Abstract

The Path Computation Element Communication Protocol (PCEP) provides mechanisms for Path Computation Elements (PCEs) to perform path computations in response to Path Computation Clients (PCCs) requests. The stateful PCE extensions provide stateful control of Multiprotocol Label Switching (MPLS) Traffic Engineering Label Switched Paths (TE LSP) via PCEP, for a model where the PCC delegates control over one or more locally configured LSPs to the PCE.

This document describes the automatic bandwidth adjustment of such LSPs under the Active Stateful PCE model.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 6, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

[RFC5440] describes the Path Computation Element Protocol (PCEP) as the communication between a Path Computation Client (PCC) and a Path Control Element (PCE), or between PCE and PCE, enabling computation...
of Multiprotocol Label Switching (MPLS) for Traffic Engineering Label Switched Path (TE LSP).

[I-D.ietf-pce-stateful-pce] specifies extensions to PCEP to enable stateful control of MPLS TE LSPs. In this document focus is on Active Stateful PCE where LSPs are configured on the PCC and control over them is delegated to the PCE.

Over time, based on the varying traffic pattern, an LSP established with certain bandwidth may require to adjust the reserved bandwidth over time automatically. Ingress Label Switch Router (LSR) samples the traffic rate at each sample-interval (BwSample) to determine the traffic information as Maximum Average Bandwidth (MaxAvgBw). Further adjustment to the reserved bandwidth should be made at every adjustment-interval automatically.

Enabling Auto-Bandwidth on a LSP results in the LSP automatically adjusting its bandwidth based on the actual traffic flowing through the LSP. A LSP can therefore be setup with some arbitrary (or zero) bandwidth value such that the LSP automatically monitors the traffic flow and adjusts its bandwidth every adjustment-interval period. The bandwidth adjustment uses the make-before-break signaling method so that there is no interruption to traffic flow. This is described in detail in Section 4.1. [I-D.ietf-pce-stateful-pce-app] describes the usecase for auto-bandwidth adjustment for passive and active stateful PCE.

There are two approaches to automatic bandwidth adjustments in case of active stateful PCE –

- PCE to decide adjusted bandwidth:
  - Active stateful PCE can use other information such as historical trending data, application-specific information about expected demands and central policy information along with realtime actual flow volumes to make smarter bandwidth adjustment to delegated LSPs. Since LSP has delegated control to the PCE, it is inherently suited that it should be stateful PCE that decides the bandwidth adjustments. But this requires PCC to report the realtime bandwidth usage as well as the configuration knobs etc.

- PCC to decide adjusted bandwidth:
  - This approach would be similar to passive stateful PCE model, where the headend (PCC) monitor and calculate the new adjusted bandwidth and request the computed adjusted bandwidth to be updated. The passive stateful PCE would use path request/reply
mechanism where as in active stateful PCE report/update mechanism is used to adjust the bandwidth. This approach only require PCC to report the calculated bandwidth to be adjusted. But this approach does not utilize the optimization advantages offered by the active stateful PCE.

This document defines extensions needed to support Auto-Bandwidth feature along with mechanism to provide traffic information of the LSPs in a stateful PCE model using PCEP.

This document does not exclude use of any other mechanism employed by stateful PCE to learn real time traffic information etc. But at the same time, using the same protocol (PCEP in this case) for updating and reporting the LSP parameters as well as to support automatic bandwidth adjustment is operationally beneficial.

1.1. Requirements Language

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

2. Terminology

The following terminology is used in this document.

Active Stateful PCE: PCE that uses tunnel state information learned from PCCs to optimize path computations. Additionally, it actively updates tunnel parameters in those PCCs that delegated control over their tunnels to the PCE.

Delegation: An operation to grant a PCE temporary rights to modify a subset of tunnel parameters on one or more PCC’s tunnels. Tunnels are delegated from a PCC to a PCE.

PCC: Path Computation Client: any client application requesting a path computation to be performed by a Path Computation Element.

PCE: Path Computation Element. An entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

TE LSP: Traffic Engineering Label Switched Path.

Note the additional terms defined in Section 4.1.
3. Motivation

An active stateful PCE can update the bandwidth for a delegated LSP via mechanisms described in [I-D.ietf-pce-stateful-pce]. Note that further extension are needed because of following reasons:

1. To identify the LSPs that would like to use this feature. Not all LSPs in some deployments would like their bandwidth to be dependent on the live traffic but be constant as set by the operator. Incase of PCC initiated LSP, they would be configured at PCC and PCEP should support a mechanism to identify the LSP with auto bandwidth feature enabled at the PCE. Where as for PCE initiated LSP, PCEP should support mechanisms to request PCC to provide live traffic information.

2. Further for LSP with auto bandwidth feature enabled, operator should be able to specify the knobs to control this feature like the bandwidth-range etc and PCEP should support their encoding.

3. PCC would need to report the live traffic information using the same protocol (PCEP in this case) making the network operations easier.

Extensions as specified in this document is one of the way for PCE to learn this information. But at the same time a stateful PCE MAY choose to learn this information from other means like management, performance tools.

4. Architectural Overview

4.1. Auto-Bandwidth Overview

Auto-Bandwidth feature allows an LSP to automatically and dynamically adjust its reserved bandwidth over time, i.e. without network operator intervention. The bandwidth adjustment uses the make-before-break adaptive signaling method so that there is no interruption to traffic flow.

The new bandwidth reservation is determined by sampling the actual traffic flowing through the LSP. If the traffic flowing through the LSP is lower than the configured or current bandwidth of the LSP, the extra bandwidth is being reserved needlessly and being wasted. Conversely, if the actual traffic flowing through the LSP is higher than the configured or current bandwidth of the LSP, it can potentially cause congestion or packet loss. With Auto-Bandwidth feature, the LSP bandwidth can be set to some arbitrary value (even zero) during initial setup time, and it will be periodically adjusted over time based on the actual bandwidth requirement.
Note the following terms:

Maximum Average Bandwidth (MaxAvgBw): The maximum average bandwidth is the unit to measure the current traffic demand between a time interval. This is the maximum value of the averaged traffic pattern in a particular time interval.

Sample-Interval: The time interval in which the traffic rate is collected as a sample.

Adjustment-Interval: The time interval in which the bandwidth adjustment should be made based on the MaxAvgBw.

Minimum Bandwidth: The minimum bandwidth that should be reserved for the LSP.

Maximum Bandwidth: The maximum bandwidth that can be reserved for the LSP.

Report-Threshold: This value indicates when the current live traffic bandwidth sample (BwSample) must be reported to stateful PCE via PCRpt message. Only if the percentage difference between the current BwSample and the last BwSample is greater than or equal to the threshold percentage the LSP bandwidth is reported to PCE.

Adjust-Threshold: This value indicates when the bandwidth must be adjusted. Only if the percentage difference between the current MaxAvgBw and the current bandwidth allocation is greater than or equal to the threshold percentage the LSP bandwidth is adjusted to the current bandwidth demand.

4.2. Deploying Auto-Bandwidth Feature

The traffic rate is repeatedly sampled at each sample-interval (which can be configured by the user and the default value as 5 minutes). The sampled traffic rates are accumulated over the adjustment-interval period (which can be configured by the user and the default value as 24 hours).

The ingress LSR reports the live traffic information to the stateful PCE via the PCRpt message, to avoid multiple reports, the Report-Threshold percentage is used. Only if the percentage difference between the current BwSample and the last BwSample is greater than or equal to the threshold percentage the LSP bandwidth is reported to PCE.

Stateful PCE will adjust the bandwidth of the LSP to the highest sampled traffic rate amongst the set of samples taken over the
adjustment-interval. Note that the highest sampled traffic rate could be higher or lower than the current LSP bandwidth. Only if the current MaxAvgBw and the current bandwidth allocation is greater than or equal to the Adjust-Threshold percentage the LSP bandwidth is adjusted to the current bandwidth demand.

Also to avoid multiple LSP re-signaling, sometimes operator set up longer adjustment intervals. However long adjustment-interval can also result in an undesirable effect of masking sudden changes in traffic patterns. To avoid this, the stateful PCE MAY pre-maturely expire the adjustment-interval to accommodate sudden bursts in traffic.

5. Extensions to the PCEP

5.1. AUTO-BANDWIDTH-ATTRIBUTE TLV

The AUTO-BANDWIDTH-ATTRIBUTE TLV can be included as an optional TLV in the LSP object as described in [I-D.ietf-pce-stateful-pce]. Whenever the LSP with Auto-Bandwidth feature enabled is delegated, AUTO-BANDWIDTH-ATTRIBUTE TLV is carried in PCRpt message. The TLV provides PCE with the ‘local configurable knobs’ of this feature. In case of PCE Initiated LSP ([I-D.ietf-pce-pce-initiated-lsp]) with this feature enabled, this TLV is included in LSP object with PCInitiate message.

The format of the AUTO-BANDWIDTH-ATTRIBUTE TLV is shown in the following figure:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Type=[TBD]          |           Length=16           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Sample Int   |    Adj Int    | Rpt Threshold | Adj Threshold |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Minimum Bandwidth                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Maximum Bandwidth                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Reserved             |           Flags             |L|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

AUTO-BANDWIDTH-ATTRIBUTE TLV format

The type of the TLV is [TBD] and it has a fixed length of 16 octets.
The value contains the following fields:

**Sample Int (8 bits):** The Sample-Interval, time interval in which the traffic rate is collected at the PCC.

**Adj Int (8 bits):** The Adjustment-Interval, time interval in which the bandwidth adjustment should be made.

**Rpt Threshold (8 bits):** The Report-Threshold value is encoded in percentage. Only if the percentage difference between the between the current BwSample and the last BwSample is greater than or equal to the threshold percentage the real time bandwidth sample is reported to PCE.

**Adj Threshold (8 bits):** The Adjust-Threshold value is encoded in percentage. Only if the percentage difference between the current MaxAvgBw and the current bandwidth allocation is greater than or equal to the threshold percentage the LSP bandwidth is adjusted to the current bandwidth demand.

**Minimum Bandwidth (32 bits):** The minimum bandwidth allowed is encoded in IEEE floating point format (see [IEEE.754.1985]), expressed in bytes per second. Refer to Section 3.1.2 of [RFC3471] for a table of commonly used values.

**Maximum Bandwidth (32 bits):** The maximum bandwidth allowed is encoded in IEEE floating point format (see [IEEE.754.1985]), expressed in bytes per second. Refer to Section 3.1.2 of [RFC3471] for a table of commonly used values.

**Flags (16 bits):** One flag is currently defined:

* L (Live-Traffic - 1 bit): If set, PCC SHOULD report the live traffic information flowing on the LSP as per the Report-Threshold set. Otherwise PCC only reports the calculated bandwidth to be adjusted to the PCE.

Unassigned flags MUST be set to zero on transmission and MUST be ignored on receipt.

**Reserved (16 bits):** This field MUST be set to zero on transmission and MUST be ignored on receipt.

If the above parameters are not specified by the user, based on the local policy at Ingress (PCC) the default value can be encoded.
If no default value is specified at Ingress, value ‘zero’ can be encoded for the particular field. The stateful PCE can then apply its own default value based on the local policy.

5.2. BANDWIDTH Object

As per [RFC5440], the BANDWIDTH object is defined with two Object-Type values:

- Requested Bandwidth: BANDWIDTH Object-Type is 1.
- Re-optimization Bandwidth: Bandwidth of an existing TE LSP for which a reoptimization is requested. BANDWIDTH Object-Type is 2.

The new BANDWIDTH object type 3 [TBD] is used to specify the BwSample determined from the existing TE LSP Traffic flow at every sample-interval when L bit is set in AUTO-BANDWIDTH-ATTRIBUTE TLV. The Report-Threshold percentage is used to determine if there is a need to report the current BwSample.

If Live-Traffic (L-Bit) is not set, PCC only reports the calculated bandwidth to be adjusted (MaxAvgBw) to the PCE. This is done via the existing ‘Requested Bandwidth with BANDWIDTH Object-Type as 1’.

5.3. The PCRpt Message

When the delegated LSP is enabled with the Auto-Bandwidth adjustment feature with Live-Traffic (L-Bit) set, PCC SHOULD include the BANDWIDTH object of type 3 [TBD] in the PCRpt message. The definition of the PCRpt message (see [I-D.ietf-pce-stateful-pce]) is unchanged.

When LSP is delegated to a PCE for the very first time, BANDWIDTH object of type 1 is used to specify the requested bandwidth in the PCRpt message. To report the live traffic flow information (as the BwSample) the BANDWIDTH object of type 3 [TBD] is encoded in further PCRpt message.

If Live-Traffic (L-Bit) is not set, PCC SHOULD include the BANDWIDTH object of type 1 to specify the he calculated bandwidth to be adjusted to the PCE.

5.4. The PCInitiate Message

For PCE Initiated LSP ([I-D.ietf-pce-pce-initiated-lsp]) with Auto-Bandwidth feature enabled, AUTO-BANDWIDTH-ATTRIBUTE TLV is included in LSP object with the PCInitiate message. The rest of the processing remains unchanged.
6. Security Considerations

This document defines a new BANDWIDTH type and AUTO-BANDWIDTH-ATTRIBUTE TLV which does not add any new security concerns beyond those discussed in [RFC5440] and [I-D.ietf-pce-stateful-pce] in itself. Some deployments may find the live traffic bandwidth information as extra sensitive and thus should employ suitable PCEP security mechanisms like TCP-AO or [I-D.ietf-pce-pceps].

7. Manageability Considerations

7.1. Control of Function and Policy

The Auto-Bandwidth feature MUST BE controlled per tunnel at Ingress (PCC), the values for parameters like sample-interval, adjustment-interval, minimum-bandwidth, maximum-bandwidth, report-threshold, adjust-threshold, Live-Traffic (L-Bit) SHOULD BE configurable by an operator.

7.2. Information and Data Models

[RFC7420] describes the PCEP MIB, there are no new MIB Objects for this document.

7.3. Liveness Detection and Monitoring

Mechanisms defined in this document do not imply any new liveness detection and monitoring requirements in addition to those already listed in [RFC5440].

7.4. Verify Correct Operations

Mechanisms defined in this document do not imply any new operation verification requirements in addition to those already listed in [RFC5440].

7.5. Requirements On Other Protocols

Mechanisms defined in this document do not imply any new requirements on other protocols.

7.6. Impact On Network Operations

Mechanisms defined in this document do not have any impact on network operations in addition to those already listed in [RFC5440].
8. IANA Considerations

8.1. PCEP TLV Type Indicators

This document defines the following new PCEP TLVs; IANA is requested to make the following allocations from this registry.

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>TBD</td>
<td>AUTO-BANDWIDTH-ATTRIBUTE</td>
<td>[This I.D.]</td>
</tr>
</tbody>
</table>

8.2. AUTO-BANDWIDTH-ATTRIBUTE

This document requests that a registry is created to manage the Flags field in the AUTO-BANDWIDTH-ATTRIBUTE TLV in the LSP object. New values are to be assigned by Standards Action [RFC5226]. Each bit should be tracked with the following qualities:

- Bit number (counting from bit 0 as the most significant bit)
- Capability description
- Defining RFC

The following values are defined in this document:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Live-Traffic (L-Bit)</td>
<td>[This I.D.]</td>
</tr>
</tbody>
</table>

8.3. BANDWIDTH Object

This document defines new object type for the BANDWIDTH object; IANA is requested to make the following allocations from this registry.

<table>
<thead>
<tr>
<th>Object-Class Value</th>
<th>Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>BANDWIDTH</td>
<td>[This I.D.]</td>
</tr>
<tr>
<td>Object-Type</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3: MaxAvgBw determined from the existing TE LSP Traffic flow.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

9. Acknowledgments

We would like to thank Venugopal Reddy, Reeja Paul, Sandeep Boina and Avantika for their useful comments and suggestions.
10. References

10.1. Normative References


10.2. Informative References


Institute of Electrical and Electronics Engineers,
"Standard for Binary Floating-Point Arithmetic", IEEE
Appendix A. Contributor Addresses

He Zekun  
Tencent Holdings Ltd,  
Shenzhen P.R.China

Email: kinghe@tencent.com

Xian Zhang  
Huawei Technologies  
Research Area F3-1B,  
Huawei Industrial Base,  
Shenzhen, 518129, China

Phone: +86-755-28972645  
Email: zhang.xian@huawei.com

Young Lee  
Huawei Technologies  
1700 Alma Drive, Suite 100  
Plano, TX  75075  
US

Phone: +1 972 509 5599 x2240  
Fax: +1 469 229 5397  
EMail: leeyoung@huawei.com

Authors’ Addresses

Dhruv Dhody  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560037  
India

EMail: dhruv.ietf@gmail.com

Udayasree Palle  
Huawei Technologies  
Divyashree Techno Park, Whitefield  
Bangalore, Karnataka 560037  
India

EMail: udayasree.palle@huawei.com

Conveying path setup type in PCEP messages
draft-ietf-pce-lsp-setup-type-01.txt

Abstract

A Path Computation Element can compute traffic engineering paths (TE paths) through a network that are subject to various constraints. Currently, TE paths are label switched paths (LSPs) which are set up using the RSVP-TE signaling protocol. However, other TE path setup methods are possible within the PCE architecture. This document proposes an extension to PCEP to allow support for different path setup methods over a given PCEP session.

Requirements Language

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

Status of This Memo

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

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

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

[RFC5440] describes the Path Computation Element Protocol (PCEP) for communication between a Path Computation Client (PCC) and a Path Control Element (PCE) or between one a pair of PCEs. A PCC requests a path subject to various constraints and optimization criteria from a PCE. The PCE responds to the PCC with a hop-by-hop path in an Explicit Route Object (ERO). The PCC uses the ERO to set up the path in the network.

[I-D.ietf-pce-stateful-pce] specifies extensions to PCEP that allow a PCC to delegate its LSPs to a PCE. The PCE can then update the state of LSPs delegated to it. In particular, the PCE may modify the path of an LSP by sending a new ERO. The PCC uses this ERO to re-route the LSP in a make-before-break fashion. [I-D.ietf-pce-pce-initiated-lsp] specifies a mechanism allowing a PCE to dynamically instantiate an LSP on a PCC by sending the ERO and characteristics of the LSP. The PCC signals the LSP using the ERO and other attributes sent by the PCE.
So far, the PCEP protocol and its extensions implicitly assume that the TE paths are label switched, and are established via the RSVP-TE protocol. However, other methods of LSP setup are not precluded. When a new path setup method (other than RSVP-TE) is introduced for setting up a path, a new capability TLV pertaining to the new path setup method MAY be advertised when the PCEP session is established. Such capability TLV MUST be defined in the specification of the new path setup type. When multiple path setup methods are deployed in a network, a given PCEP session may have to simultaneously support more than one path setup types. In this case, the intended path setup method needs to be either explicitly indicated or implied in the appropriate PCEP messages (when necessary) so that both the PCC and the PCE can take the necessary steps to set up the path. This document introduces a generic TLV called "PATH-SETUP-TYPE TLV" and specifies the base procedures to facilitate such operational model.

2. Terminology

The following terminologies are used in this document:

ERO: Explicit Route Object.
LSR: Label Switching Router.
PCC: Path Computation Client.
PCE: Path Computation Element
PCEP: Path Computation Element Protocol
TLV: Type, Length, and Value.

3. Path Setup Type TLV

When a PCEP session is used to set up TE paths using different methods, the corresponding PCE and PCC must be aware of the path setup method used. That means, a PCE must be able to specify paths in the correct format and a PCC must be able take control and take forwarding plane actions appropriate to the path setup type.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Type              |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Reserved            |      PST      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: PATH-SETUP-TYPE TLV

PATH-SETUP-TYPE TLV is an optional TLV associated with the RP ([RFC5440]) and the SRP ([I-D.ietf-pce-stateful-pce]) objects. Its format is shown in the above figure. The type of the TLV is to be
defined by IANA. The one octet value contains the Path Setup Type (PST). This document specifies the following PST value:

- PST = 0: Path is setup via RSVP-TE signaling protocol (default).

The absence of the PATH-SETUP-TYPE TLV is equivalent to an PATH-SETUP-TYPE TLV with an PST value of 0. It is recommended to omit the TLV in the default case. If the RP or SRP object contains more than one PATH-SETUP-TYPE TLVs, only the first TLV MUST be processed and the rest MUST be ignored.

If a PCEP speaker does not recognize the PATH-SETUP-TYPE TLV, it MUST ignore the TLV in accordance with ([RFC5440]). If a PCEP speaker recognizes the TLV but does not support the TLV, it MUST send PCErr with Error-Type = 2 (Capability not supported).

4. Operation

When requesting a path from a PCE using a PCReq message ([RFC5440]), a PCC MAY include the PATH-SETUP-TYPE TLV in the RP object. If the PCE is capable of expressing the path in a format appropriate to the setup method used, it MUST use the appropriate ERO format in the PCRep message. If the path setup type cannot be inferred from the ERO or any other object or TLV in the PCRep message, PATH-SETUP-TYPE TLV may be included in the RP object of the PCRep message. Regardless of whether PATH-SETUP-TYPE TLV is used or not, if the PCE does not support the intended path setup type it MUST send PCErr with Error-Type = TBD (Traffic engineering path setup error) (recommended value is 21) and Error-Value = 1 (Unsupported path setup type) and close the PCEP session. If the path setup types corresponding to the PCReq and PCRep messages do not match, the PCC MUST send a PCErr with Error-Type = 21 (Traffic engineering path setup error) and Error-Value = 2 (Mismatched path setup type) and close the PCEP session.

In the case of stateful PCE, if the path setup type cannot be unambiguously inferred from ERO or any other object or TLV, PATH-SETUP-TYPE TLV MAY be used in PCRpt and PCUpd messages. If PATH-SETUP-TYPE TLV is used in PCRpt message, the SRP object MUST be present even in cases when the SRF-ID-number is the reserved value of 0x00000000. Regardless of whether PATH-SETUP-TYPE TLV is used or not, if a PCRpt message is triggered due to a PCUpd message (in this case SRF-ID-number is not equal to 0x00000000), the path setup types corresponding to the PCRpt and PCUpd messages should match. Otherwise, the PCE MUST send PCErr with Error-Type = 21 (Traffic engineering path setup error) and Error-Value = 2 (Mismatched path setup type) and close the connection.
In the case of PCE initiated LSPs, a PCE MAY include PATH-SETUP-TYPE TLV in PCInitiate message if the message does not have any other means of indicating path setup type. If a PCC does not support the path setup type associated with the PCInitiate message, the PCC MUST send PCErr with Error-Type = 21 (Traffic engineering path setup error) and Error-Value = 1 (Unsupported path setup type) and close the PCEP session. Similarly, as mentioned above, if the path setup type cannot be unambiguously inferred from ERO or any other object or TLV, the PATH-SETUP-TYPE TLV MAY be included in PCRpt messages triggered by PCInitiate message. Regardless of whether PATH-SETUP-TYPE TLV is used or not, if a PCRpt message is triggered by a PCInitiate message, the path setup types corresponding to the PCRpt and the PCInitiate messages should match. Otherwise, the PCE MUST send PCErr message with Error-Type = 21 (Traffic engineering path setup error) and Error-Value = 2 (Mismatched path setup type).

5. Security Considerations

No additional security measure is required.

6. IANA Considerations

IANA is requested to allocate a new TLV type (recommended value is TBD) for PATH-SETUP-TYPE TLV specified in this document.

This document requests that a registry is created to manage the value of the path Setup Type field in the PATH-SETUP-TYPE TLV.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Traffic engineering path is setup using RSVP signaling protocol</td>
<td>This document</td>
</tr>
</tbody>
</table>

This document also defines a new Error-Type (recommended 21) and new Error-Values for the following new error conditions:

<table>
<thead>
<tr>
<th>Error-Type</th>
<th>Meaning</th>
<th>Error-Value=1: Unsupported path setup type</th>
<th>Error-Value=2: Mismatched path setup type</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>Invalid traffic engineering path setup type</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
7. Acknowledgements

We like to thank Marek Zavodsky for valuable comments.

8. Normative References

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

[I-D.ietf-pce-stateful-pce]


Authors’ Addresses

Siva Sivabalan
Cisco Systems, Inc.
2000 Innovation Drive
Kanata, Ontario K2K 3E8
Canada

Email: msiva@cisco.com

Jan Medved
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95134
USA

Email: jmedved@cisco.com
Ina Minei
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA  94043
USA

Email: inaminei@google.com

Edward Crabbe

Robert Varga
Pantheon Technologies SRO
Mlynske Nivy 56
Bratislava, 821 05
Slovakia

Email: robert.vargad@pantheon.sk

Jeff Tantsura
Ericsson
300 Holger Way
San Jose, CA  95134
USA

Email: jeff.tantsura@ericsson.com
Abstract

Segment Routing (SR) enables any head-end node to select any path without relying on a hop-by-hop signaling technique (e.g., LDP or RSVP-TE). It depends only on "segments" that are advertised by Link-State Interior Gateway Protocols (IGPs). A Segment Routed Path can be derived from a variety of mechanisms, including an IGP Shortest Path Tree (SPT), explicit configuration, or a Path Computation Element (PCE). This document specifies extensions to the Path Computation Element Protocol (PCEP) that allow a stateful PCE to compute and initiate Traffic Engineering (TE) paths, as well as a PCC to request a path subject to certain constraint(s) and optimization criteria in SR networks.

Requirements Language

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

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.
Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 10, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

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

Table of Contents

1. Introduction ........................................ 3
2. Terminology ........................................ 4
3. Overview of PCEP Operation in SR Networks ............... 5
4. SR-Specific PCEP Message Extensions .................... 6
5. Object Formats ....................................... 7
   5.1. The OPEN Object .................................. 7
   5.1.1. The SR PCE Capability TLV ...................... 7
   5.2. The RP/SRP Object ................................ 8
   5.3. ERO Object ...................................... 8
   5.3.1. SR-ERO Subobject ............................... 9
   5.3.2. NAI Associated with SID ....................... 10
   5.3.3. ERO Processing ................................ 12
   5.4. RRO Object ..................................... 13
   5.4.1. RRO Processing ............................... 13
6. Backward Compatibility ................................ 14
7. Management Considerations ............................. 14
   7.1. Policy .......................................... 14
   7.2. The PCEP Data Model ............................. 14
8. Security Considerations ................................ 14
9. IANA Considerations .................................. 14
   9.1. PCEP Objects .................................... 14
   9.2. PCEP-Error Object ................................ 14
   9.3. PCEP TLV Type Indicators ......................... 15
   9.4. New Path Setup Type ............................. 15

1. Introduction

SR technology leverages the source routing and tunneling paradigms. A source node can choose a path without relying on hop-by-hop signaling protocols such as LDP or RSVP-TE. Each path is specified as a set of "segments" advertised by link-state routing protocols (IS-IS or OSPF). [I-D.filsfils-rtgwg-segment-routing] provides an introduction to SR architecture. The corresponding IS-IS and OSPF extensions are specified in [I-D.ietf-isis-segment-routing-extensions] and [I-D.ietf-ospf-segment-routing-extensions], respectively. SR architecture defines a "segment" as a piece of information advertised by a link-state routing protocols, e.g. an IGP prefix or an IGP adjacency. Several types of segments are defined. A Node segment represents an ECMP-aware shortest-path computed by IGP to a specific node, and is always global within SR/IGP domain. An Adjacency Segment represents unidirectional adjacency. An Adjacency Segment is local to the node which advertises it. Both Node segments and Adjacency segments can be used for SR Traffic Engineering (SR-TE).

The SR architecture can be applied to the MPLS forwarding plane without any change, in which case an SR path corresponds to an MPLS Label Switching Path (LSP). This document is relevant to only MPLS forwarding plane, and assumes that a 32-bit Segment Identifier (SID) represents an absolute value of MPLS label entry. In this document, "Node-SID" and "Adjacency-SID" denote Node Segment Identifier and Adjacency Segment Identifier respectively.

A Segment Routed path (SR path) can be derived from an IGP Shortest Path Tree (SPT). SR-TE paths may not follow IGP SPT. Such paths may be chosen by a suitable network planning tool and provisioned on the source node of the SR-TE path.

[RFC5440] describes Path Computation Element Protocol (PCEP) for communication between a Path Computation Client (PCC) and a Path Computation Element (PCE) or between one a pair of PCEs. A PCE or a PCC operating as a PCE (in hierarchical PCE environment) computes paths for MPLS Traffic Engineering LSPs (MPLS-TE LSPs) based on various constraints and optimization criteria. [I-D.ietf-pce-stateful-pce] specifies extensions to PCEP that allow a stateful PCE to compute and recommend network paths in compliance
Stateful PCEP extensions provide synchronization of LSP state between a PCC and a PCE or between a pair of PCEs, delegation of LSP control, reporting of LSP state from a PCC to a PCE, controlling the setup and path routing of an LSP from a PCE to a PCC. Stateful PCEP extensions are intended for an operational model in which LSPs are configured on the PCC, and control over them is delegated to the PCE.

A mechanism to dynamically initiate LSPs on a PCC based on the requests from a stateful PCE or a controller using stateful PCE is specified in [I-D.ietf-pce-pce-initiated-lsp]. Such mechanism is useful in Software Driven Networks (SDN) applications, such as demand engineering, or bandwidth calendaring.

It is possible to use a stateful PCE for computing one or more SR-TE paths taking into account various constraints and objective functions. Once a path is chosen, the stateful PCE can initiate an SR-TE path on a PCC using PCEP extensions specified in [I-D.ietf-pce-pce-initiated-lsp] using the SR specific PCEP extensions described in this document. Additionally, using procedures described in this document, a PCC can request an SR path from either stateful or a stateless PCE. This specification relies on the PATH-SETUP-TYPE TLV and procedures specified in [I-D.ietf-pce-lsp-setup-type].

2. Terminology

The following terminologies are used in this document:

- **ERO**: Explicit Route Object
- **IGP**: Interior Gateway Protocol
- **IS-IS**: Intermediate System to Intermediate System
- **LSR**: Label Switching Router
- **MSD**: Maximum SID Depth
- **NAI**: Node or Adjacency Identifier
- **OSPF**: Open Shortest Path First
- **PCC**: Path Computation Client
- **PCE**: Path Computation Element
- **PCEP**: Path Computation Element Protocol
3. Overview of PCEP Operation in SR Networks

In SR networks, an ingress node of an SR path appends all outgoing packets with an SR header consisting of a list of SIDs (or MPLS labels in the context of this document). The header has all necessary information to guide the packets from the ingress node to the egress node of the path, and hence there is no need for any signaling protocol.

In a PCEP session, LSP information is carried in the Explicit Route Object (ERO), which consists of a sequence of subobjects. Various types of ERO subobjects have been specified in [RFC3209], [RFC3473], and [RFC3477]. In SR networks, an ingress node of an SR path appends all outgoing packets with an SR header consisting of a list of SIDs (or MPLS labels in the context of this document). SR-TE LSPs computed by a PCE can be represented in one of the following forms:

- An ordered set of IP address(es) representing network nodes/links: In this case, the PCC needs to convert the IP address(es) into the corresponding MPLS labels by consulting its Traffic Engineering Database (TED).

- An ordered set of SID(s).

- An ordered set of both MPLS label(s) and IP address(es): In this case, the PCC needs to convert the IP address(es) into the corresponding SID(s) by consulting its TED.

This document defines a new ERO subobject denoted by "SR-ERO subobject" capable of carrying a SID as well as the identity of the node/adjacency represented by the SID. SR-capable PCEP speakers should be able to generate and/or process such ERO subobject. An ERO containing SR-ERO subobjects can be included in the PCEP Path Computation Reply (PCRep) message defined in [RFC5440], the PCEP LSP Initiate Request message (PCInitiate) defined in [I-D.ietf-pce-pce-initiated-lsp], as well as in the PCEP LSP Update Request (PCUpd) and PCEP LSP State Report (PCRpt) messages defined in [I-D.ietf-pce-stateful-pce].
When a PCEP session between a PCC and a PCE is established, both PCEP speakers exchange information to indicate their ability to support SR-specific functionality. Furthermore, an LSP initially established via RSVP-TE signaling can be updated with SR-TE path. This capability is useful when a network is migrated from RSVP-TE to SR-TE technology. Similarly, an LSP initially created with SR-TE path can be updated to signal the LSP using RSVP-TE if necessary.

A PCC MAY include an RRO object containing the recorded LSP in PCReq and PCRpt messages as specified in [RFC5440] and [I-D.ietf-pce-stateful-pce] respectively. This document defines a new RRO subobject for SR networks. Methods used by a PCC to record SR-TE LSP are outside the scope of this document.

In summary, this document:

- Defines a new PCEP capability, new ERO subobject, new RRO subobject, a new TLV, and new PCEP error codes.
- Specifies how two PCEP speakers can establish a PCEP session that can carry information about SR-TE paths.
- Specifies processing rules of ERO subobject.
- Defines a new path setup type carried in the PATH-SETUP-TYPE TLV for SR-TE LSP.

The extensions specified in this document complement the existing PCEP specifications to support SR-TE path. As such, the PCEP messages (e.g., Path Computation Request, Path Computation Reply, Path Computation Report, Path Computation Update, Path Computation Initiate, etc.) MUST be formatted according to [RFC5440], [I-D.ietf-pce-stateful-pce], [I-D.ietf-pce-pce-initiated-lsp], and any other applicable PCEP specifications.

4. SR-Specific PCEP Message Extensions

As defined in [RFC5440], a PCEP message consists of a common header followed by a variable length body made up of mandatory and/or optional objects. This document does not require any changes in the format of PCReq and PCRpt messages specified in [RFC5440], PCInitiate message specified in [I-D.ietf-pce-pce-initiated-lsp], and PCRpt and PCUpd messages specified in [I-D.ietf-pce-stateful-pce]. However, PCEP messages pertaining to SR-TE LSP MUST include PATH-SETUP-TYPE TLV in the RP or SRP object to clearly identify that SR-TE LSP is intended. In other words, a PCEP speaker MUST not infer whether or not a PCEP message pertains to SR-TE LSP from any other object or TLV.
5. Object Formats

5.1. The OPEN Object

This document defines a new optional TLV for use in the OPEN Object.

5.1.1. The SR PCE Capability TLV

The SR-PCE-CAPABILITY TLV is an optional TLV associated with the OPEN Object to exchange SR capability of PCEP speakers. The format of the SR-PCE-CAPABILITY TLV is shown in the following figure:

```
+-----------------------------------------------+
<table>
<thead>
<tr>
<th>Type=TBD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length=4</td>
</tr>
<tr>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>------------------------------------------</td>
</tr>
</tbody>
</table>
+------------------------------------------+

Figure 1: SR-PCE-CAPABILITY TLV format
```

The code point for the TLV type is to be defined by IANA. The TLV length is 4 octets.

The 32-bit value is formatted as follows. The "Maximum SID Depth" (1 octet) field (MSD) specifies the maximum number of SIDs that a PCC is capable of imposing on a packet. The "Flags" (1 octet) and "Reserved" (2 octets) fields are currently unused, and MUST be set to zero on transmission and ignored on reception.

5.1.1.1. Exchanging SR Capability

By including the SR-PCE-CAPABILITY TLV in the OPEN message destined to a PCE, a PCC indicates that it is capable of supporting the head-end functions for SR-TE LSP. By including the TLV in the OPEN message destined to a PCC, a PCE indicates that it is capable of computing SR-TE paths.

The number of SIDs that can be imposed on a packet depends on PCC’s data plane’s capability. The default value of MSD is 0 meaning that a PCC does not impose any limitation on the number of SIDs included in any SR-TE path coming from PCE. Once an SR-capable PCEP session is established with a non-default MSD value, the corresponding PCE cannot send SR-TE paths with SIDs exceeding that MSD value. If a PCC needs to modify the MSD value, the PCEP session MUST be closed and re-established with the new MSD value. If a PCEP session is
established with a non-default MSD value, and the PCC receives an SR-TE path containing more SIDs than specified in the MSD value, the PCC MUST send a PCErr message with Error-Type 10 (Reception of an invalid object) and Error-value 3 (Unsupported number of Segment ERO).

The SR Capability TLV is meaningful only in the OPEN message sent from a PCC to a PCE. As such, a PCE does not need to set MSD value in outbound message to a PCC. Similarly, a PCC ignores any MSD value received from a PCE. If a PCE receives multiple SR-PCE-CAPABILITY TLVs in an OPEN message, it processes only the first TLV is processed.

5.2. The RP/SRP Object

In order to setup an SR-TE LSP using SR, RP or SRP object MUST PATH-SETUP-TYPE TLV specified in [I-D.ietf-pce-lsp-setup-type]. This document defines a new Path Setup Type (PST) for SR as follows:

- PST = 1: Path is setup using Segment Routing Traffic Engineering technique.

5.3. ERO Object

An SR-TE path consists of one or more SID(s) where each SID MAY be associated with the identifier that represents the node or adjacency corresponding to the SID. This identifier is referred to as the 'Node or Adjacency Identifier' (NAI). As described later, a NAI can be represented in various formats (e.g., IPv4 address, IPv6 address, etc). Furthermore, a NAI is used for troubleshooting purposes and, if necessary, to derive SID value as described below.

The ERO object specified in [RFC5440] is used to carry SR-TE path information. In order to carry SID and/or NAI, this document defines a new ERO subobject referred to as "SR-ERO subobject" whose format is specified in the following section. An ERO object carrying an SR-TE path consists of one or more ERO subobject(s), and MUST carry only SR-ERO subobject. Note that an SR-ERO subobject does not need to have both SID and NAI. However, at least one of them MUST be present.

When building the MPLS label stack from ERO, a PCC MUST assume that SR-ERO subobjects are organized as a last-in-first-out stack. The first subobject relative to the beginning of ERO contains the information about the topmost label. The last subobject contains information about the bottommost label.
5.3.1. SR-ERO Subobject

An SR-ERO subobject consists of a 32-bit header followed by the SID and the NAI associated with the SID. The SID is a 32-bit number. The size of the NAI depends on its respective type, as described in the following sections.

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|L|    Type     |     Length    |  ST   |     Flags     |F|S|C|M|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              SID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                        NAI (variable)                       //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: SR-ERO Subobject format

The fields in the SR-ERO Subobject are as follows:

The ‘L’ Flag indicates whether the subobject represents a loose-hop in the LSP [RFC3209]. If this flag is unset, a PCC MUST not overwrite the SID value present in the SR-ERO subobject. Otherwise, a PCC MAY expand or replace one or more SID value(s) in the received SR-ERO based on its local policy.

Type is the type of the SR-ERO subobject. This document defines the SR-ERO subobject type, and requests a new codepoint from IANA.

Length contains the total length of the subobject in octets, including the L, Type and Length fields. Length MUST be at least 8, and MUST be a multiple of 4. As mentioned earlier, an SR-ERO subobject MUST have at least SID or NAI. The length should take into consideration SID or NAI only if they are not null. The flags described below used to indicate whether SID or NAI field is null.

SID Type (ST) indicates the type of information associated with the SID contained in the object body. The SID-Type values are described later in this document.
Flags is used to carry any additional information pertaining to SID. Currently, the following flag bits are defined:

* M: When this bit is set, the SID value represents an MPLS label stack entry as specified in [RFC5462] where only the label value is specified by the PCE. Other fields (TC, S, and TTL) fields MUST be considered invalid, and PCC MUST set these fields according to its local policy and MPLS forwarding rules.

* C: When this bit as well as the M bit are set, then the SID value represents an MPLS label stack entry as specified in [RFC5462], where all the entry’s fields (Label, TC, S, and TTL) are specified by the PCE. However, a PCC MAY choose to override TC, S, and TTL values according to its local policy and MPLS forwarding rules.

* S: When this bit is set, the SID value in the subobject body is null. In this case, the PCC is responsible for choosing the SID value, e.g., by looking up its TED using the NAI which, in this case, MUST be present in the subobject.

* F: When this bit is set, the NAI value in the subobject body is null.

Editorial Note: we need to decide how to treat an SR-ERO subobject in which both NAI and SID are null.

SID is the Segment Identifier.

NAI contains the NAI associated with the SID. Depending on the value of ST, the NAI can have different format as described in the following section.

5.3.2. NAI Associated with SID

This document defines the following NAIs:

‘IPv4 Node ID’ is specified as an IPv4 address. In this case, ST value is 1, and the Length is 8 or 12 depending on either SID or NAI or both are included in the subobject.
'IPv6 Node ID’ is specified as an IPv6 address. In this case, ST and Length are 2, and Length is 8, 20, or 24 depending on either SID or NAI or both are included in the subobject.

‘IPv4 Adjacency’ is specified as a pair of IPv4 addresses. In this case, ST value is 3. The Length is 8, 12, or 16 depending on whether SID or NAI or both are included in the subobject, and the format of the NAI is shown in the following figure:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Local IPv4 address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Remote IPv4 address                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: NAI for IPv4 Adjacency

‘IPv6 Adjacency’ is specified as a pair of IPv6 addresses. In this case, ST value is 4. The Length is 8, 36 or 40 depending on whether SID or NAI or both included in the subobject, and the format of the NAI is shown in the following figure:

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//               Local IPv6 address (16 bytes)                 //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//               Remote IPv6 address (16 bytes)                //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: NAI for IPv6 Adjacency

‘Unnumbered Adjacency with IPv4 NodeIDs’ is specified as a pair of Node ID / Interface ID tuples. In this case, ST value is 5. The Length is 8, 20, or 24 depending on whether SID or NAI or both included in the subobject, and the format of the NAI is shown in the following figure:
Figure 5: NAI for Unnumbered adjacency with IPv4 Node IDs

Editorial Note: We are yet to decide if another SID subobject is required for unnumbered adjacency with 128 bit node ID.

5.3.3. ERO Processing

A PCEP speaker that does not recognize the SR-ERO subobject in PCRep, PCInitiate, PCUpd or PCRpt messages MUST reject the entire PCEP message and MUST send a PCE error message with Error-Type=3 ("Unknown Object") and Error-Value=2 ("Unrecognized object Type") or Error-Type=4 ("Not supported object") and Error-Value=2 ("Not supported object Type"), defined in [RFC5440].

When the SID represents an MPLS label (i.e. the M bit is set), its value (20 most significant bits) MUST be larger than 15, unless it is special purpose label, such as an Entropy Label Indicator (ELI). If a PCEP speaker receives a label ERO subobject with an invalid value, it MUST send the PCE error message with Error-Type = 10 ("Reception of an invalid object") and Error Value = TBD ("Bad label value"). If both M and C bits of an ERO subobject are set, and if a PCEP speaker finds erroneous setting in one or more of TC, S, and TTL fields, it MUST send a PCE error with Error-Type = 10 ("Reception of an invalid object") and Error-Value = TBD ("Bad label format").

If a PCC receives a stack of SR-ERO subobjects, and the number of stack exceeds the maximum number of SIDs that the PCC can impose on the packet, it MAY send a PCE error with Error-Type = 10 ("Reception of an invalid object") and Error-Value = TBD ("Unsupported number of Segment ERO subobjects").

When a PCEP speaker detects that all subobjects of ERO are not identical, and if it does not handle such ERO, it MUST send PCE error with Error-Type = 10 ("Reception of an invalid object") and Error-Value = TBD ("Non-identical ERO subobjects").
If a PCEP speaker receives an SR-ERO subobject in which both SID and NAI are absent, it MUST consider the entire ERO object invalid and send a PCE error with Error-Type = 10 ("Reception of an invalid object") and Error-Value = TBD ("Both SID and NAI are absent in ERO subobject").

5.4.  RRO Object

A PCC can record SR-TE LSP and report the LSP to a PCE via RRO. An RRO object contains one or more subobjects called "SR-RRO subobjects" whose format is shown below:

```
+-----------------------------------------------+--------------------------+-------------------------+
| Type | Length | ST | Flags | F|S|C|M |
+-----------------------------------------------+--------------------------+-------------------------+
| SID  |
+-----------------------------------------------+--------------------------+-------------------------+
//                        NAI (variable)                        //
+-----------------------------------------------+--------------------------+-------------------------+
```

Figure 6: SR-RRO Subobject format

The format of SR-RRO subobject is the same as that of SR-ERO subobject without L flag.

A PCC MUST assume that SR-RRO subobjects are organized such that the first subobject relative to the beginning of RRO contains the information about the topmost label, and the last subobject contains information about the bottommost label of the SR-TE LSP.

5.4.1.  RRO Processing

Processing rules of SR-RRO subobject are identical to those of SR-ERO subobject.

If a PCEP speaker receives an SR-RRO subobject in which both SID and NAI are absent, it MUST consider the entire RRO object invalid and send a PCE error with Error-Type = 10 ("Reception of an invalid object") and Error-Value = TBD ("Both SID and NAI are absent in RRO subobject").

If a PCE detects that all subobjects of RRO are not identical, and if it does not handle such RRO, it MUST send PCE error with Error-Type = 10 ("Reception of an invalid object") and Error-Value = TBD ("Non-identical RRO subobjects").
6. Backward Compatibility

A PCEP speaker that does not support the SR PCEP capability cannot recognize the SR-ERO or SR-RRO subobjects. As such, it MUST send a PCEP error with Error-Type = 4 (Not supported object) and Error-Value = 2 (Not supported object Type) as per [RFC5440].

7. Management Considerations

7.1. Policy

PCEP implementation:

- Can enable SR PCEP capability either by default or via explicit configuration.
- May generate PCEP error due to unsupported number of SR-ERO or SR-RRO subobjects either by default or via explicit configuration.

7.2. The PCEP Data Model

A PCEP MIB module is defined in [I-D.ietf-pce-pcep-mib] needs be extended to cover additional functionality provided by [RFC5440] and [I-D.ietf-pce-pce-initiated-lsp]. Such extension will cover the new functionality specified in this document.

8. Security Considerations

The security considerations described in [RFC5440] and [I-D.ietf-pce-pce-initiated-lsp] are applicable to this specification. No additional security measure is required.

9. IANA Considerations

9.1. PCEP Objects

IANA is requested to allocate a new ERO subobject and a new RRO subobject types (recommended values = 5 and 6 respectively).

9.2. PCEP-Error Object

This document defines new Error-Type and Error-Value for the following new conditions:

<table>
<thead>
<tr>
<th>Error-Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Reception of an invalid object.</td>
</tr>
</tbody>
</table>
9.3. PCEP TLV Type Indicators

This document defines the following new PCEP TLVs:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>SR-PCE-CAPABILITY</td>
<td>This document</td>
</tr>
</tbody>
</table>

9.4. New Path Setup Type

This document defines a new setup type for the PATH-SETUP-TYPE TLV as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Traffic engineering path is setup using Segment Routing technique.</td>
<td>This document</td>
</tr>
</tbody>
</table>

10. Contributors

The following people contributed to this document:

- Lakshmi Sharma (Cisco Systems)

11. Acknowledgements

We like to thank Ina Minei, George Swallow, Marek Zavodsky and Tomas Janciga for the valuable comments.

12. References
12.1. Normative References

[I-D.filsfils-rtgwg-segment-routing]

[I-D.ietf-isis-segment-routing-extensions]

[I-D.ietf-ospf-segment-routing-extensions]

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

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

[I-D.ietf-pce-pcep-mib]

[I-D.ietf-pce-stateful-pce]


12.2. Informative References


Authors' Addresses

Siva Sivabalan
Cisco Systems, Inc.
2000 Innovation Drive
Kanata, Ontario K2K 3E8
Canada

Email: msiva@cisco.com

Jan Medved
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA  95134
US

Email: jmedved@cisco.com
A YANG Data Model for Path Computation Element Communications Protocol (PCEP)
draft-pkd-pce-pcep-yang-02

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 August 27, 2015.

Copyright Notice

Copyright (c) 2015 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
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
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 base PCEP YANG module, "ietf-pcep" which provides the basic PCEP [RFC5440] data model.

[Editor’s Note: Further modules augmenting the data model with advanced features maybe handled in a future revision or a separate document.]

2. Requirements Language

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

3. Terminology and Notation

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

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

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

- PCEP entity: a local PCEP speaker.
- PCEP peer: to refer to a remote PCEP speaker.
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]:

- Brackets "[" and "]" enclose list keys.
- Curly braces "{" and "}" contain names of optional features that make the corresponding node conditional.
- Abbreviations before data node names: "rw" means configuration (read-write), and "ro" state data (read-only).
- Symbols after data node names: "?" means an optional node and "*" denotes a "list" or "leaf-list".
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (":").
- 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] |
+--------+-----------------+-----------+
```

Table 1: Prefixes and corresponding YANG modules

4. Objectives

This section describes some of the design objectives for the model:
In case of existing implementations, it needs to map the data model defined in this document to their proprietary native data model. To facilitate such mappings, the data model should be simple.

The data model should be suitable for new implementations to use as is.

Mapping to the PCEP MIB Module should be clear.

The data model should allow for static configurations of peers.

The data model should include read-only counters in order to gather statistics for sent and received PCEP messages, received messages with errors, and messages that could not be sent due to errors.

It should be fairly straightforward to augment the base data model for advanced PCE features.

5. The Design of PCEP Data Model

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

module: ietf-pcep
  +--rw pcep
    |   +--rw entity* [addr]
    |      +--rw addr      inet:ip-address
    |      +--rw enabled?  boolean
    |      +--rw role      pcep-role
    |      +--rw description?  string
    |      +--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 domain
    |            |   +--rw domain-type?  pce-domain-type
    |            |   +--rw domain?      pce-domain
    |            +--rw neigh-domains
| +--rw domain* [domain-type domain] |
|    | +--rw domain-type pce-domain-type |
|    |    +--rw domain pce-domain |
|    |    +--rw capability |
|    |    +--rw gmpls? boolean |
|    |    +--rw bi-dir? boolean |
|    |    +--rw diverse? boolean |
|    |    +--rw load-balance? boolean |
|    |    +--rw synchronize? boolean |
|    |    +--rw objective-function? boolean |
|    |    +--rw add-path-constraint? boolean |
|    |    +--rw prioritization? boolean |
|    |    +--rw multi-request? boolean |
|    |    +--rw gco? boolean |
|    |    +--rw p2mp? boolean |
|    |    +--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 |
|    |    +--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 peers |
|    |    +--rw peer* [addr] |
|    |    +--rw addr inet:ip-address |
|    |    +--rw description? string |
|    |    +--rw pce-info |
|    |    +--rw scope |
|    |    +--rw intra-area-scope? boolean |
|    |    +--rw intra-area-pref? uint8 |
|    |    +--rw inter-area-scope? boolean |
|    |    +--rw inter-area-scopedefault? boolean |
|    |    +--rw inter-area-pref? uint8 |
|    |    +--rw inter-as-scope? boolean |
|    |    +--rw inter-as-scope-default? boolean |
|    |    +--rw inter-as-pref? uint8 |
Internet-Draft                  PCE-YANG                   February 2015

|              |  +--rw inter-layer-scope?          boolean
|              |  +--rw inter-layer-pref?           uint8
|              |  +--rw domain
|              |  +--rw domain-type?   pce-domain-type
|              |  +--rw domain?        pce-domain
|              |  +--rw neigh-domains
|              |  +--rw domain* [domain-type domain]
|              |     +--rw domain-type    pce-domain-type
|              |     +--rw domain         pce-domain
|              |  +--rw capability
|              |     +--rw gmpls?                 boolean
|              |     +--rw bi-dir?                boolean
|              |     +--rw diverse?               boolean
|              |     +--rw load-balance?          boolean
|              |     +--rw synchronize?           boolean
|              |     +--rw objective-function?    boolean
|              |     +--rw add-path-constraint?   boolean
|              |     +--rw prioritization?        boolean
|              |     +--rw multi-request?         boolean
|              |     +--rw gco?                   boolean
|              |     +--rw p2mp?                  boolean
|     +--ro pcep-state
|     +--ro entity* [addr]
|         +--ro index?              uint32
|         +--ro addr                inet:ip-address
|         +--ro admin-status?       pcep-admin-status
|         +--ro oper-status?        pcep-admin-status
|         +--ro role?               pcep-role
|     +--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 domain
|             |  +--ro domain-type?   pce-domain-type
|             |  +--ro domain?        pce-domain
|         +--ro neigh-domains
|             |  +--ro domain* [domain-type domain]
|             |     +--ro domain-type    pce-domain-type
|             |     +--ro domain         pce-domain
|         +--ro capability
<pre>++--ro gmpls? boolean
++--ro bi-dir? boolean
++--ro diverse? boolean
++--ro load-balance? boolean
++--ro synchronize? boolean
++--ro objective-function? boolean
++--ro add-path-constraint? boolean
++--ro prioritization? boolean
++--ro multi-request? boolean
++--ro gco? boolean
++--ro p2mp? boolean
++--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
++--ro request-timer? uint32
++--ro max-sessions? uint32
++--ro max-unknown-reqs? uint32
++--ro max-unknown-mgs? uint32
++--ro peers
  ++--ro peer* [addr]
  |   ++--ro addr inet:ip-address
  |   ++--ro role? pcep-role
  |   ++--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 domain
  |       |     |   ++--ro domain-type? pce-domain-type
  |       |     |   ++--ro domain? pce-domain
</pre>
++-ro neigh-domains
   ++-ro domain* [domain-type domain]
      ++-ro domain-type pce-domain-type
      ++-ro domain pce-domain
++-ro capability
   ++-ro gmpls? boolean
   ++-ro bi-dir? boolean
   ++-ro diverse? boolean
   ++-ro load-balance? boolean
   ++-ro synchronize? boolean
   ++-ro objective-function? boolean
   ++-ro add-path-constraint? boolean
   ++-ro prioritization? boolean
   ++-ro multi-request? boolean
   ++-ro gco? boolean
   ++-ro p2mp? boolean
   ++-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 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-svec-sent? yang:counter32
++-ro num-svec-req-sent? yang:counter32
++-ro num-req-sent-pend-ref? 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?
++-ro num-req-rcvd?
++-ro num-svec-rcvd?
++-ro num-svec-req-rcvd?
++-ro num-req-rcvd-pend-rep?
++-ro num-req-rcvd-ero-sent?
++-ro num-req-rcvd-nopath-sent?
++-ro num-req-rcvd-cancel-sent?
++-ro num-req-rcvd-error-sent?
++-ro num-req-rcvd-cancel-rcvd?
++-ro num-req-rcvd-cancel-rcvd?
++-ro num-rep-rcvd-unknown?
++-ro num-req-rcvd-unknown?
++-ro num-req-sent-closed?
++-ro num-req-rcvd-closed?
++-ro sessions
    +++-ro session* [initiator]
        +++-ro initiator pcep-initiator
        +++-ro state-last-change? yang:timestamp
        +++-ro state? pcep-sess-state
        +++-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 discontinuity-time? yang:timestamp
        +++-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?
        +++-ro num-keepalive-rcvd?
        +++-ro num-unknown-rcvd?
        +++-ro num-corrupt-rcvd?
        +++-ro num-req-sent?
---ro num-svec-sent?  yang:counter32
++--ro num-svec-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-svec-rcvd?  yang:counter32
++--ro num-svec-req-rcvd?  yang:counter32
++--ro num-svec-req-rcvd?  yang:counter32
++--ro num-req-rcvd-pend-rep?  yang:counter32
++--ro num-req-rcvd-ero-rcvd?  yang:counter32
++--ro num-req-rcvd-nopath-sent?  yang:counter32
++--ro num-req-rcvd-cancel-sent?  yang:counter32
++--ro num-req-rcvd-error-sent?  yang:counter32
++--ro num-req-rcvd-cancel-rcvd?  yang:counter32
++--ro num-rep-rcvd-unknown?  yang:counter32
++--ro num-req-rcvd-unknown?  yang:counter32
++--ro num-requested rcvd?  yang:counter32
++--ro num-svec-sent?  yang:counter32
++--ro num-svec-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-svec-rcvd?  yang:counter32
++--ro num-svec-req-rcvd?  yang:counter32
++--ro num-svec-req-rcvd?  yang:counter32
++--ro num-req-rcvd-pend-rep?  yang:counter32
++--ro num-req-rcvd-ero-rcvd?  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

notifications:

---n pcep-session-up
  |---ro entity-addr?  leafref
  |---ro peer-addr?  leafref
  |---ro session-initiator?  leafref
  |---ro state-last-change?  yang:timestamp
  |---ro state?  pcep-sess-state

---n pcep-session-down
  |---ro entity-addr?  leafref
  |---ro peer-addr?  leafref
  |---ro session-initiator?  pcep-initiator
  |---ro state-last-change?  yang:timestamp
  |---ro state?  pcep-sess-state

---n pcep-session-local-overload
  |---ro entity-addr?  leafref
  |---ro peer-addr?  leafref
  |---ro session-initiator?  leafref
  |---ro overloaded?  boolean
  |---ro overload-time?  uint32

---n pcep-session-local-overload-clear
  |---ro entity-addr?  leafref
  |---ro peer-addr?  leafref
  |---ro overloaded?  boolean

---n pcep-session-peer-overload
  |---ro entity-addr?  leafref
  |---ro peer-addr?  leafref
  |---ro session-initiator?  leafref
  |---ro peer-overloaded?  boolean
5.1. The Entity Lists

The PCEP yang module may contain status information for multiple logical local PCEP entities. There are several scenarios in which there may be more than one local PCEP entity, it is listed in [RFC7420].

The data model for PCEP presented in this document uses a flat list of entities. Each entity in the list is identified by its IP address (addr-type, addr). Furthermore, each entity has a mandatory "role" leaf (the local entity PCEP role). The ietf-inet-types [RFC6991] is used.

There is one list for configuration ("/pcep/entity"), and a separate list for the operational state of all entities ("/pcep-state/entity").

The PCEP MIB module [RFC7420] uses 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.

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").

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.

A means and procedure to handle to following PCE features needs to be decided:

- Capability and Scope
7. 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@2015-02-xx.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;
    }

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

    contact "WG Web: <http://tools.ietf.org/wg/pce/>
            WG List: <mailto:pce@ietf.org>"
The YANG module defines a generic configuration and operational model for PCEP common across all of the vendor implementations.

revision 2015-02-23 { 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";

    }
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 {

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 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 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";
"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 pce-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 {

typedef pce-domain {
  type union {
    type domain-ospf-area;
    type domain-isis-area;
    type domain-as;
  }
  description
    "The PCE Domain";
}

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
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{
    type uint32 {
        range "0..65535";
    }
}
units "seconds";
default 60;
description "The value of SyncTimer in seconds is used in the case of synchronized path computation request using the SVEC object. Consider the case where a PCReq message is received by a PCE that contains the SVEC object referring to M synchronized path computation requests. If after the expiration of the SyncTimer all the M path computation requests have not been received, a protocol error is triggered and the PCE MUST cancel the whole set of path computation requests.

The aim of the SyncTimer is to avoid the storage of unused synchronized requests should one of them get lost for some reasons (for example, a misbehaving PCC).

Zero means that the PCEP entity does not use the SyncTimer.";
reference "RFC 5440: Path Computation Element (PCE) Communication Protocol (PCEP)"
}

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

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

leaf max-unknown-reqs{
type uint32;
default 5;
description "The maximum number of unrecognized requests and
replies that any session on this PCEP entity is willing to accept per minute before terminating the session.

A PCRep message contains an unrecognized reply if it contains an RP object whose request ID does not correspond to any in-progress request sent by this PCEP entity.

A PCReq message contains an unrecognized request if it contains an RP object whose request ID is zero.

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

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

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.";
}
"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 pce-domain{
  description
  "This grouping specifies a PCE-Domain where the PCE has topology visibility and through which the PCE can compute paths."
  leaf domain-type{
    type pce-domain-type;
    description
    "The PCE domain type."
  }
  leaf domain{
    type pce-domain;
    description
    "The PCE domain."
  }
} // pce-domain

grouping pce-capability{
  description
  "This grouping specifies a PCE-capability 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 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{
  type boolean;
  description
  "Synchronized paths computation.";
}
leaf 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{
  type boolean;
  description
    "Support for Global Concurrent Optimization
     (GCO).";
}
leaf p2mp{
  type boolean;
  description
    "Support for P2MP path computation.";
}
} // pce-capability

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 domain{
    description
      "The PCE domain";
    uses pce-domain{
      description
        "The PCE local domain."
    }
  }
  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 pce-domain{
        description
          "The PCE neighbour domain.";
      }
    }
  }
}
container capability{
    description
        "The PCE capability";
    uses pce-capability{
        description
            "The PCE entity supported capabilities.";
    }
}

//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 avg-rsp-time = 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 lwm-rsp-time = 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.";
        )
    }
}
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 hwm-rsp-time = 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-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-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-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.";
}
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-rep-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.";
}
//pcep-stats

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

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

  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
/*
* Configuration data nodes
* /
container pcep{

description
"Parameters for list of configured PCEP entities on the device.";

list entity{
    key "addr";

description
"The configured 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 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";
}

container pce-info {
  must "((../role == 'pce')" + " or " + "((../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";
}

description
  "The Local PCE Entity PCE 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 peers{
    description
        "The list of configured peers for the entity";

    list peer{
        key "addr";

        description
            "The peer configured for the entity."

        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";
        }

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

        description
            "The Local PCE Entity PCE information";

    } // peer
} // peers
} // entity
} // pcep

/*
 * Operational data nodes
 */

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

list entity{
    key "addr";
    unique "index";
    description
    "The operational PCEP entity on the device.";
    leaf index{
        type uint32;
        description
        "The index of the operational PECP
        entity";
    }
    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 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.";
}

container pce-info {
  when "((../role == 'pce')" +
  " or " +
  "((../role == 'pcc-and-pce'))"
  {
    description
    "When PCEP entity is PCE";
  }
  uses pce-info {
    description
    "Local PCE information";
  }
  description
  "The Local PCE Entity PCE information";
}
uses pcep-entity-info{
    description
    "The operational information related to the PCEP entity.";
}

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.";
        }
    }

    container pce-info {
        uses pce-info {
            description
            "PCE Peer information";
        }
        description
        "The Local PCE Entity PCE information";
    }

    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 " +
    "session-up-time = 0))";
}
leaf session-fail-time{
  type yang:timestamp;
  must "(../num-sess-setup-fail != 0 or " +
    "(../num-sess-setup-fail = 0 and "  +
    "session-fail-time = 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 "  +
    "session-fail-up-time = 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.";
}

uses pcep-stats{
"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."
}

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 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 remote-id = 0))" {
    error-message
    "Invalid remote-id";
    description
    "If state is tcp-pending or open-wait then this leaf is not used and MUST be set to zero.";
}
leaf keepalive-timer {
    type uint32 {
        range "0..255";
    }
    units "seconds";
    must "(.state = 'session-up' + "or" + "(.state != 'session-up' + "and keepalive-timer = 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 " + "peer-keepalive-timer = 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 " +
       "peer-dead-timer = 0))" {
    error-message
      "Invalid Peer Dead
timer";
    description
      "If state is tcp-
pending or open-wait	hen 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 " +
  "ka-hold-time-rem = 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"
        " overload-time = 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 "
        "peer-overload-time = 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."
};

} // 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;

} // session

} // sessions

} // peer

} // entity

} // pcep-state

*/

} // pcep-stats

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

} // session

} // sessions

} // peer

} // 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;

} // session

} // sessions

} // peer

} // entity

} // pcep-state

*/

} // pcep-stats
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";
        must "{../overloaded = true or " + "{../overloaded != true and " + "overload-time = 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.";
    }
} //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";
    must "((../peer-overloaded = true or " +
      "(../peer-overloaded != true and "  +
      "peer-overload-time = 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.";
  }
} //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>

8. 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.

9. Manageability Considerations

9.1. Control of Function and Policy

9.2. Information and Data Models

9.3. Liveness Detection and Monitoring

9.4. Verify Correct Operations
9.5. Requirements On Other Protocols


10. IANA Considerations

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


Registrant Contact: The PCE WG of the IETF.

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

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

Name: ietf-pcep
Prefix: pcep
Reference: This I-D

11. 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.

12. References

12.1. Normative References


12.2.  Informative References


Appendix A. Contributor Addresses

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 560037
India

EMail: udayasree.palle@huawei.com

Vishnu Pavan Beeram
Juniper Networks
USA

EMail: vbeeram@juniper.net

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

EMail: zhang.xian@huawei.com

Authors’ Addresses

Rohit Pobbathi
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560037
India

EMail: rohit.pobbathi@huawei.com
Abstract

[I-D.xu-spring-pce-based-sfc-arch] describes a PCE-based SFC architecture in which the PCE is used to compute service function paths in SPRING networks. Based on the above architecture, this document describes extensions to the Path Computation Element Protocol (PCEP) that allow a PCE to compute and instantiate service function paths in SPRING networks. The extensions specified in this document are applicable to both the stateless PCE model and the stateful PCE model.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on April 30, 2015.
Copyright Notice

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

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

Table of Contents

1. Introduction ............................................. 3
2. Terminology ............................................ 4
3. Overview of PCEP Extensions for SFC in SR Networks .......... 5
4. PCEP Message Extensions for SR-based SFC .................. 5
   4.1. PCReq Message ................................... 5
   4.2. PCRep Message ................................... 5
   4.3. PCUpd Message ................................... 6
   4.4. PCRpt Message ................................... 6
5. Object Formats .......................................... 7
   5.1. OPEN Object ..................................... 7
      5.1.1. SR-SFC PCE Capability TLV ..................... 7
   5.2. RP/SRP Object ................................... 8
   5.3. Include Route Object ................................ 8
   5.4. SR-SFC-ERO Object ................................ 8
      5.4.1. SR-SFC-ERO Subobject ........................... 9
      5.4.2. NSI Associated with SID ....................... 10
      5.4.3. SR-SFC-ERO Processing ......................... 11
6. IANA Considerations ..................................... 11
   6.1. PCEP Objects ..................................... 11
   6.2. PCEP-Error Object .................................. 11
   6.3. PCEP TLV Type Indicators ............................ 11
   6.4. New Path Setup Type ................................ 11
   6.5. New IRO Sub-object Type ............................ 11
7. Security considerations .................................. 12
8. Acknowledgement ......................................... 12
9. References ............................................... 12
   9.1. Normative References ............................... 12
   9.2. Informative References ............................. 12
Authors’ Addresses ........................................ 13
1. Introduction

Service Function Chaining [I-D.ietf-sfc-architecture] provides a flexible way to construct services. When applying a particular Service Function Chain (SFC) to the traffic classified by the Classifier, the traffic needs to be steered through an ordered set of Service Functions (SF) in the network. This ordered set of SFs in the network, referred to as a Service Function Path (SFP), is an instantiation of the SFC in the network. For example, as shown in Figure 1, an SFP corresponding to the SFC of \{SF1, SF3\} can be expressed as \{SFF1, SF1, SFF2, SF3\}.

![Figure 1: PCE-based Service Function Chaining in SR Network](image)

[I-D.xu-spring-pce-based-sfc-arch] describes a PCE-based SFC architecture in which the PCE is used to compute an SFP (i.e., instantiate a service function chain) in SPRING networks (a.k.a., Segment Routing networks or SR networks in short). This document describes extensions to the PCEP on basis of that architecture. The extensions specified in this document are applicable to both the stateless PCE model defined in [RFC5440] and the stateful PCE model defined in [I-D.ietf-pce-stateful-pce].
2. Terminology

This section contains definitions for terms used frequently throughout this document. However, many additional definitions can be found in [RFC5440], [I-D.sivabalan-pce-segment-routing] and [I-D.xu-spring-pce-based-sfc-arch].

PCC: Path Computation Client

PCE: Path Computation Element

PCEP: Path Computation Element Protocol

ERO: Explicit Route Object

Service Function Chain (SFC): A service function chain defines a set of abstract service functions and ordering constraints that must be applied to packets and/or frames selected as a result of classification.

SF Identifier (SF ID): A unique identifier that represents a service function within an SFC-enabled domain.

Service Function Forwarder (SFF): A service function forwarder is responsible for delivering traffic received from the network to one or more connected service functions according to information carried in the SFC encapsulation, as well as handling traffic coming back from the SF.

Service Function Path (SFP): The SFP provides a level of indirection between the fully abstract notion of service chain as a sequence of abstract service functions to be delivered, and the fully specified notion of exactly which SFF/SFs the packet will visit when it actually traverses the network. Specifically, it is an ordered list of SFFs and SF IDs.

Compact SFP: An ordered list of SFFs.

SID: Segment Identifier

Service Function SID: A locally unique SID indicating a particular service function on an SFF.

SR: Segment Routing

SR-specific SFP: An ordered list of node SIDs (representing SFFs) and Service Function SIDs.
Compact SR-specific SFP: An ordered list of node SIDs (representing SFFs).

3. Overview of PCEP Extensions for SFC in SR Networks

As discussed in [I-D.xu-spring-pce-based-sfc-arch], the PCC provides an ordered list of SF IDs to the PCE and indicates to the PCE that what type SFs and paths are requested (e.g., an SFP, or a compact SFP, or an SR-specific SFP, or a compact SR-specific SFP) through the path computation request message, and then the PCE responds with a corresponding path through the path computation response message. This specification is applicable to both stateful and stateless PCEs.

4. PCEP Message Extensions for SR-based SFC

4.1. PCReq Message

This document does not specify any changes to the PCReq message format. This document requires the PATH-SETUP-TYPE TLV [I-D.sivabalan-pce-lsp-setup-type] to be carried in the RP Object in order for a PCC to request a particular type of path. Four new Path Setup Types need to be defined for SR-based SFC, or SR-SFC in short (Section 5.2). This document also requires the Include Route Object (IRO) to be carried in the PCReq message in order for a PCC to specify SFC. A new IRO sub-object type needs to be defined for SF (Section 5.3).

4.2. PCRep Message

This document defines the format of the PCRep message carrying an SFP. The message is sent by a PCE to a PCC in response to a previously received PCReq message, where the PCC requested an SFP. The format of the SFC-specific PCRep message is as follows:
<PCRep Message>::=<Common Header>
    <response-list>

Where:

<response-list>::=<response>[<response-list>]

<response>::=<RP>
    [<NO-PATH>]
    [<path-list>]

Where:

<path-list>::=<SR-SFC-ERO>[<path-list>]

The RP and NO-PATH Objects are defined in [RFC5440]. The <SR-SFC-ERO> object contains the SFP and is defined in Section 5.4.

4.3. PCUpd Message

This document defines the format of the PCUpd message carrying an SFP update. The message is sent forwardly by a PCE to a PCC to update an previously computed SFP.

The format of the PCUpd message is as follows:

<PCUpd Message>::=<Common Header>
    <update-request-list>

Where:

<update-request-list>::=<update-request>[<update-request-list>]

<update-request>::=<SRP><path-list>

Where:

<path-list>::=<SR-SFC-ERO>[<path-list>]

4.4. PCRpt Message

PCPRpt message sent from a PCC to PCE as a respond to a PCUpd message or in an unsolicited manner (e.g., during state synchronization).

The format of the PCUpd message is as follows:
<PCUpd Message>::=<Common Header>  
   <state-report-list>
Where:
   <state-report-list>::=<state-report>[<state-report-list>]
   <state-report>::=[<SRP>]<path-list>
Where:
   <path-list>::=<SR-SFC-ERO>[<path-list>]
5. Object Formats
5.1. OPEN Object
This document defines a new optional TLV for use in the OPEN Object.
5.1.1. SR-SFC PCE Capability TLV
The SR-SFC-PCE-CAPABILITY TLV is an optional TLV for use in the OPEN Object to negotiate SR-SFC capability on the PCEP session. The format of the SR-SFC-PCE-CAPABILITY TLV is shown in the following Figure 2:

0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Type=TBD              |       Length=4                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Reserved             |  Flags        |      MSD      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
Figure 2: SR-SFC-PCE-CAPABILITY TLV format
The code point for the TLV type is to be defined by IANA. The TLV length is 4 octets. The 32-bit value is formatted as follows. The "Maximum SID Depth" (1 octet) field (MSD) specifies the maximum number of SIDs that a PCC is capable of imposing on a packet. The "Flags" (1 octet) and "Reserved" (2 octets) fields are currently unused, and MUST be set to zero and ignored on receipt.
5.1.1.1. Negotiating SR-SFC Capability

The SR-SFC capability TLV is contained in the OPEN object. By including the TLV in the OPEN message to a PCE, a PCC indicates its support for SFPs. By including the TLV in the OPEN message to a PCC, a PCE indicates that it is capable of computing SFPs.

5.2. RP/SRP Object

In order to setup an SFP, the RP or SRP object MUST carry a PATH-SETUP-TYPE TLV specified in [I-D.sivabalan-pce-lsp-setup-type]. This document defines four new Path Setup Types (PST) for SR-SFC as follows:

- PST = 2: The path is an SFP.
- PST = 3: The path is a compact SFP.
- PST = 4: The path is an SR-specific SFP.
- PST = 5: The path is a compact SR-specific SFP.

5.3. Include Route Object

The IRO (Include Route Object) MUST be carried within PCReq messages to indicate a particular SFC. Furthermore, the IRO MAY be carried in PCRep messages. When carried within a PCRep message with the NO-PATH object, the IRO indicates the set of service functions that cause the PCE to fail to find a path.

This document defines a new sub-object type for the SR-SFC as follows:

<table>
<thead>
<tr>
<th>Type</th>
<th>Sub-object</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Service Function ID</td>
</tr>
</tbody>
</table>

5.4. SR-SFC-ERO Object

Generally speaking, an SR-SFC-ERO object consists of one or more ERO subobjects described in the following sub-sections to represent a particular type of service function path. In the ERO subobject, each SID is associated with an identifier that represents either a service node or a service function. This identifier is referred to as the 'Node or Service Identifier' (NSI). As described later, an NSI can be represented in various formats (e.g., IPv4 address, IPv6 address, SF identifier, etc). Specifically, in the SFP case, the NSI of every ERO subobject contained in the SR-SFC-ERO object represents a service...
node or a service function while the SID of each ERO subobject is set to null. In the compact SFP case, the NSI of every ERO subobject contained in the SR-SFC-ERO object only represents an SFF meanwhile the SID of every ERO subobject is set to null. In the SR-specific SFP, the NSI of every ERO subobject contained in the SR-SFC-ERO object represents an SFF or a service function while the SID of every ERO subject MUST NOT be null. In the compact SR-specific SFP, the NSI of every ERO subobject contained in the SR-SFC-ERO object represents an SFF meanwhile the SID of every ERO subobject MUST NOT be null.

5.4.1. SR-SFC-ERO Subobject

An SR-SFC-ERO subobject (as shown in Figure 3) consists of a 32-bit header followed by the SID and the NSI associated with the SID. The SID is a 32-bit or 128 bit number. The size of the NSI depends on its respective type, as described in the following sub-sections.

<table>
<thead>
<tr>
<th>0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
</tr>
<tr>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>SID (variable: 4 or 16 octets)</td>
</tr>
<tr>
<td>NSI (variable)</td>
</tr>
</tbody>
</table>

Figure 3: SR-SFC-ERO Subobject Format

The fields in the ERO Subobject are as follows:

- **‘L’ Flag**: indicates whether the subobject represents a loose-hop in the explicit route [RFC3209]. If this flag is unset, a PCC MUST not overwrite the SID value present in the SR-SFC-ERO subobject. Otherwise, a PCC MAY expand or replace one or more SID value(s) in the received SR-SFC-ERO based on its local policy.

- **Type**: is the type of the SR-SFC-ERO Subobject. This document defines the SR-SFC-ERO Subobject type. A new code point will be requested for the SR-SFC-ERO Subobject from IANA.

- **Length**: contains the total length of the subobject in octets, including the L, Type and Length fields. Length MUST be at least 4, and MUST be a multiple of 4.

- **NSI Type (NSIT)**: indicates the type of NSI associated with the SID. The NSI-Type values are described later in this document.
Flags: is used to carry any additional information pertaining to SID. Currently, the following flag bits are defined:

M: When this bit is set, the SID value represents an MPLS label stack entry as specified in [RFC5462], where only the label value is specified by the PCE. Other fields (TC, S, and TTL) fields MUST be considered invalid, and PCC MUST set these fields according to its local policy and MPLS forwarding rules.

C: When this bit as well as the M bit are set, then the SID value represents an MPLS label stack entry as specified in [RFC5462], where all the entry’s fields (Label, TC, S, and TTL) are specified by the PCE. However, a PCC MAY choose to override TC, S, and TTL values according its local policy and MPLS forwarding rules.

S: When this bit is set, the SID value in the subobject body is null. In this case, the PCC is responsible for choosing the SID value, e.g., by looking up its Traffic Engineering Database (TED) using node/service identifier in the subobject body.

F: When this bit is set, the NSI value in the subobject body is null.

P: When this bit is set, the SID value represents an IPv6 address.

SID: is the 4-octect or 16-octect Segment Identifier

NSI: contains the NSI associated with the SID. Depending on the value of NSIT, the NSI can have different format as described in the following sub-section.

5.4.2. NSI Associated with SID

This document defines the following NSIs:

‘IPv4 Node ID’: is specified as an IPv4 address. In this case, NSIT and Length are 1 and 12 respectively.

‘IPv6 Node ID’: is specified as an IPv6 address. In this case, NSIT and Length are 2 and 24 respectively.

‘Service Function ID’: is specified as an SF ID. In this case, NSIT and Length are TBD.
5.4.3. SR-SFC-ERO Processing

TBD.

6. IANA Considerations

6.1. PCEP Objects

IANA is requested to allocate an ERO subobject type (recommended value= 6) for the SR-SFC-ERO subobject.

6.2. PCEP-Error Object

TBD.

6.3. PCEP TLV Type Indicators

This document defines the following new PCEP TLVs:

<table>
<thead>
<tr>
<th>Value</th>
<th>Meaning</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>SR-SFC-PCE-CAPABILITY</td>
<td>This document</td>
</tr>
</tbody>
</table>

6.4. New Path Setup Type

This document defines a new setup type for the PATH-SETUP-TYPE TLV as follows:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>The path is an SFP.</td>
<td>This document</td>
</tr>
<tr>
<td>3</td>
<td>The path is a compact SFP.</td>
<td>This document</td>
</tr>
<tr>
<td>4</td>
<td>The path is an SR-specific SFP.</td>
<td>This document</td>
</tr>
<tr>
<td>5</td>
<td>The path is a compact SR-specific SFP.</td>
<td>This document</td>
</tr>
</tbody>
</table>

6.5. New IRO Sub-object Type

This document defines a new IRO sub-object type for the SFC as follows:

<table>
<thead>
<tr>
<th>Type</th>
<th>Sub-object</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Service Function ID</td>
</tr>
</tbody>
</table>
7. Security considerations

This document does not introduce any new security considerations.

8. Acknowledgement

TBD.

9. References

9.1. Normative References

[I-D.ietf-pce-stateful-pce]

[I-D.ietf-sfc-architecture]


9.2. Informative References

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


Authors’ Addresses

Xiaohu Xu
Huawei
Email: xuxiaohu@huawei.com

Jianjie You
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, 210012
China
Email: youjianjie@huawei.com

Siva Sivabalan
Cisco Systems
2000 Innovation Drive
Kanata, Ontario K2K 3E8
Canada
Email: msiva@cisco.com

Himanshu Shah
Ciena
3939 North First Street
San Jose, CA 95134
USA
Email: hshah@ciena.com
Extensions to Path Computation Element Protocol (PCEP) to Support Resource Sharing-based Path Computation
draft-zhang-pce-resource-sharing-03.txt

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on August 27, 2015.

Copyright Notice

Copyright (c) 2014 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.

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].

Abstract

Resource sharing in a network means two or more Label Switched Paths (LSPs) use common piece(s) of resource along their paths. This can help save network resource and useful in scenarios such as LSP recovery or when two LSPs do not need to be active at the same time. A Path Computation Element (PCE) is a centralized entity, responsible for path computation. Given this feature and its access to the network resource information and possibly active LSPs information, it can be used to support resource-sharing-based path computation with better efficiency.

This document extends the Path Computation Element Protocol (PCEP) in order to support resource sharing-based path computation.

Table of Contents

1. Introduction and Motivation................................. 3
2. Motivation .................................................. 4
   2.1. Use Case 1 ............................................. 4
   2.2. Use Case 2 ............................................. 6
3. Extensions to PCEP .......................................... 8
   3.1. Resource Sharing Object............................... 8
   3.2. Processing Rules..................................... 9
   3.3. Carrying RSO in a PCEP Message ....................... 10
4. Security Considerations..................................... 11
5. IANA Considerations ....................................... 11
   5.1. New Object Type.................................... 11
6. References ................................................. 12
   6.1. Normative References ................................. 12

Zhang et al
Expires August 2015
1. Introduction and Motivation

A Path Computation Element (PCE) provides an alternative way for providing path computation function, and it is especially useful in the scenarios where complex constraints and/or a demanding amount of computation resource are required [RFC4655]. The development of PCE standardization has evolved from stateless to stateful. A stateful PCE has access to the LSP database information of the network(s) it serves as a computation engine [Stateful-PCE]. Unless specified, this document assumes a PCE mentioned is a stateful PCE (either passive or active).

Resource sharing denotes that two or more Label Switched Paths (LSPs) share common piece(s) of resource, (such as a common time slot of a link in an Optical Transport Network (OTN)). This is usually useful in the scenario where only one LSP is active and the benefit herein is to save network resources. A simple example of this is dynamically calculating a LSP for an existing LSP undergoing a link failure. Note that the resource sharing can be worked out using a stateless PCE, but the mechanism may be complex and is out the scope of this draft.

This document considers the following requirement: resource sharing with one or multiple existing LSPs.

In a single domain, this is a common requirement in the recovery cases especially in order to increase traffic resilience against failure while reducing the amount of network resource used for recovery purpose [RFC4428].

The current protocol supporting the communication between a PCE and a Path Computation Client (PCC), i.e. PCE Protocol (PCEP), allows for re-optimization of an existing LSP [RFC5440]. This is achieved by setting R bit in the Request Parameter (RP) object, together with some additional information if applicable, in the Path Computation Request (PCReq) message sent from a PCC to the PCE. To support this type of resource sharing, a PCC needs to ask a PCE to compute a new path with the constraints of sharing resource with one or multiple existing LSPs. It is worth noting the ‘resource sharing’ in this draft not only means one LSP re-using the same link(s) of another LSP, but also the same slice of bandwidth. This may occur when an LSP is required for re-routing, or online re-optimization. Current PCEP specifications do not provide such function.
As mentioned in [stateful-PCE], the PLSP-ID is unique during a PCEP session between PCC and PCE. Such identification is helpful in supporting the above resource sharing requirement for standardization of stateful PCEs. With a unique identifier, the configuration of PCCs is greatly simplified. Instead of determining all the resources to be shared, the PCC could request resource sharing directly from PCE.

The resource sharing can also be required in an inter-layer PCEP session. This is similar to the previous requirement. However, it is more complex and therefore deserves a more detailed explanation here.

In a multi-layer network, Label Switched Paths (LSPs) in a lower layer are used to carry higher-layer LSPs across the lower-layer network [RFC5623]. Therefore, the resource sharing constraints in the higher layer might actually relate to the resource sharing in the lower layer. Thus, it is useful to consider how this can be achieved and whether additional extensions are needed using the models defined in [RFC5623].

In the next sections, use cases are provided to show what information needs to be exchanged to fulfill these requirements. This memo then provides extensions to PCEP to enable this function.

2. Motivation

2.1. Single PCE Use Case

Figure 1 shows a single domain network with a stateful PCE. Assume a working LSP (N1-N2-N3) exists in the network, when there is failure on the link N2-N3, it is desired to set up a restoration path for this working LSP. Suppose N1 serves as the PCC and sends a request to the stateful PCE for such an LSP. Before sending the request, N1 may need to check what policy should be applied for the path computation. For example, it might value resource sharing and prefer to share as much resource with the working LSP as possible and specify this policy in the PCReq message. If resources are shared between the old and new LSPs, there will be some ‘interruption’ when the traffic is switched from the old LSP to the new LSP.

On the other hand, in some scenarios there are different policies, for example the LSP should be restored without any interruption with best effort. An example can be found in Fig. 1 without failure on N2-N3 link, instead, an online re-optimization is needed for the working LSP (N1-N2-N3) from the stateful PCE. In such cases, the best choice is to set up a backup LSP for the working LSP with
totally separate routing (for example N1-N5-N4-N3), and move the
traffic to that backup LSP. After that the working LSP can be torn
down, which will not result in any interruption during the
optimization procedure. This can actually be implemented with
existing PCEP mechanism. However, if there is no such separate path,
existing PCEP will reply error. A secondary option for this case is
to set up an LSP and complete such re-optimization with resource
sharing, even if some interruption introduced. Given the resource
from the LSP to be interrupted, there may be some solutions instead
of Path Compute error due to the lack of resource.

A simple illustration is provided below:

```
+--------------+
|              |
| Stateful PCE |
|              |
+--------------+
```

```
+--------+          +--------+          +--------+
| N1 +------X----+ N2 +------N3 |
| +--------+          +--------+          +--------+
|              |                      |              |
| +---------+       +---------+       +---------+|
|            |              |            |              |
| +--------+ N5 +-----+ +--------+ N4 +-----+|
|         |              |         |              |
| +--------+          +--------+          +--------+
```

Figure 1: A Single Domain Example

Available recovery paths computed by the stateful PCE:

LSP1: N1-N2-N4-N3
LSP2: N1-N5-N4-N3

If resource sharing is preferred, the stateful PCE will reply with
LSP1 information. Instead, if PCC prefer to have less interruption,
PCE will reply with LSP2 information.

Another piece of information that needs to be conveyed to the PCE is
the information about the working path LSP. Note this simple use
case assumes end-to-end recovery. But in order to be applicable to
use cases such as shared mesh protection purpose, where the head-end
or tail-end nodes may be different, this information is necessary in the message exchange between PCCs and PCEs, so that the stateful PCE knows which LSP the path computation request wants to share the resource.

Besides, parameter changes during the resource sharing computation also need to be considered. For example, the bandwidth of the request LSP may be different with the existing LSP, while resource sharing is still preferred by the PCC. PCE should consider the sharing request together with the policy and available resource(s) in the network. Details can be found in Section 3.3.

2.2. Multiple PCEs Use Case

Figure 2 shows a two-layer network example, with each layer managed by a PCE. As discussed in Section 3 of [RFC5623], there are three models for inter-layer path computation. They are single PCE computation, multiple PCE with inter-PCE communication and multiple PCE without inter-PCE communication, respectively. For the single PCE computation, the process would be similar to that of the use case in Section 2.1. Thus, this model is not discussed further.
An inter-layer path computation example is shown in Fig. 2, assume a LSP (LSP1: H2-H3) has been established already, visible as H2-H3 from view of higher-layer PCE and H2-L1-L2-H3 from the global view (or from the view of lower-layer PCE). A new request comes at H2 to establish a new LSP (LSP2: from H2 to H5), given the constraint it can share resource with LSP1. This requirement is possible if only one of the LSPs needs to be active and resource sharing is the target.

If multiple PCE with inter-PCE communication model is employed, the path computation request sent by H2 to higher-layer PCE will be forwarded to lower-layer PCE since there is no resource readily available in the higher layer. So it leaves the lower-layer PCE to compute a path in the lower layer in order to support the higher layer request. In this case, lower-layer PCE is required to compute a path between H2 and H5 under the constraint that it can share the resource with that of the LSP1. At this moment the lower-layer PCE has the knowledge on the explicit routing that LSP1 go through (H2-L1-L2-H3). So when lower-layer PCE computes the path for LSP2, it can consider the resource used by LSP1 as available with higher priority. For example, lower-layer PCE may choose H2-L1-L2-L4-H5 as the computation result. On the other hand, if the path computation policy is to have a separate path with LSP1, the lower-layer PCE may choose H2-L1-L3-L4-H5.

During this procedure higher-layer PCE can only use LSP1 information (such as its five-tuple LSP information) as the information, an issue to solve is how lower-layer PCE can resolve this information to the actual resource usage in its own layer, i.e. lower layer. This could be solved by edge LSR L1 reporting this higher-lower layer LSP correlation to the lower-layer PCE as part of the LSP information during the LSP state synchronization process. If needed, it can be later updated when there is a change in this information. Alternatively, the lower-layer PCE can get this information from other sources, such as network management system, where this information should be stored.

If multiple PCE without inter-PCE communication model is employed, the path computation request in the lower layer will be initiated the border LSR node, i.e., L1. The process would be similar to that of the previous scenario. A point worth noting is that the border LSR node may be able to resolve the higher layer LSP information itself, such as mapping it to the corresponding LSP in the lower layer, in this way lower-layer PCE does not need to perform this
function. Otherwise, the mapping method mentioned above can still be used.

3. Extensions to PCEP

This section provides PCEP extensions. Currently the text focuses only on passive stateful PCE and corresponding PCReq. But if active stateful PCE delegation is used, we would like to convey the same information via RSO in PCRpt. In the passive stateful PCE architecture, a PCC is allowed to specify resource sharing when sending a PCReq message. It also details the processing rule and error codes needed.

3.1. Resource Sharing Object

The PCEP Resource Sharing Object (RSO) is optional. It MAY be carried within a PCRep message from the network element (or other PCCs) so as to indicate the desired resource sharing requirements to be applied by the stateful PCE during path computation.

The RSO object format is compliant with the PCEP object format defined in [RFC5440].

The RSO Object-Class is TBA.

The RSO Object-type is 1.

The format of the RSO object body is:

```
+----------------+----------------+----------------+----------------+
| RSO Flags      | Reserved       | Optional TLVs  |
+----------------+----------------+----------------+
                   +----------------+              |
                   | RSO Flags      |               |
                   | Reserved       |               |
                   +----------------+               |
```

Figure 3: RSO Object Format

RSO flags (16 bits): the objective of the resource sharing. Currently, the following objectives are defined:

D (1 bit): sharing as little as possible.

R (1 bit): sharing as much as possible
It is possible that multiple computation results satisfy the request. Among these results, D set to 1 will select the most separate one, while R set to 1 will select the one sharing most resources. Both D and R set to 0 don’t specify any constraint and will result in a random selection among these results. The combination of D=1 and R=1 is not allowed.

Reserved (2 bytes): This field MUST be set to zero on transmission and MUST be ignored on receipt.

Optional TLVs may be needed to indicate the LSP(s) with which the resource is shared. The LSP Info TLV, include the IPv4-LSP-IDENTIFIERS TLV and IPv6-LSP IDENTIFIERS TLV, are defined in the same way as in [stateful-pce].

3.2. Processing Rules

To request a path allowing sharing resource with one or multiple existing LSPs, a PCC includes a RSO object in the PCReq message.

On receipt of a PCReq message with a RSO object, a stateful PCE MUST proceed as follows:

- If the RSO object is unknown/unsupported, the PCE will follow procedures defined in [RFC5440]. That is, the PCE sends a PCErr message with error type 3 or 4 (Unknown / Not supported object) and error value 1 or 2 (unknown / unsupported object class / object type), and the related path computation request is discarded.

- If TLV(s) present in the RSO object are unknown/unsupported and the P bit is set, the PCE MUST send a PCErr message with error type 3 or 4 (Unknown / Not supported object) and error value 4 (Unrecognized/Unsupported parameter), and the related path computation request MUST be discarded as defined in [RFC5440].

- If the resource sharing information is extracted correctly, the PCE MUST apply the requested resource sharing requirement.

The procedure of setting R and/or D bit follows the rules defined in Section 3.1. The RSO flags may be locally configured on the requesting nodes via external entities, such as a network management system or the entity that impose the resource sharing requirement.
3.3. Carrying RSO in a PCEP Message

The RSO is applied to an individual path computation request and the format of the PCReq message is updated as follows:

<PCReq Message> ::= <Common Header>
[<svec-list>]
<request-list>

where:

<svec-list> ::= <SVEC>
[<OF>]
[<metric-list>]
[<svec-list>]

<request-list> ::= <request> [<request-list>]

<request> ::= <RP>

<END-POINTS>
[<LSPA>]
[<BANDWIDTH>]
[<metric-list>]
[<OF>]
[<RRO>[<BANDWIDTH>]]
[<IRO>]
[<RSO>]
[<LOAD-BALANCING>]

Zhang et al  Expires August 2015
and where:

\[
<\text{metric-list}> :::= <\text{METRIC}>[<\text{metric-list}>]
\]

4. Security Considerations

Security of PCEP is discussed in [RFC5440] and [RFC6952]. The extensions in this document do not change the fundamentals of security for PCEP.

However, the introduction of the RSO provides a vector that may be used to probe for information from a network. For example, a PCC that wants to discover the path of an LSP with which it is not involved, can issue a PCReq with an RSO and may be able to get back quite a lot of information about the path of the LSP through issuing multiple such requests for different endpoints and analyzing the received results. To protect against this, a PCE should be configured with access and authorization controls such that only authorized PCCs (for example, those within the network) can make computation requests, only specifically authorized PCCs can make requests using the RSO, and resource sharing requests relating to specific LSPs are further limited to a select few PCCs. How such access controls and authorization is managed is outside the scope of this document, but it will at the least include Access Control Lists.

Furthermore, a PCC must be aware that setting up an LSP that shares resources with another LSP may be a way of attacking the other LSP, for example by depriving it of the resources it needs to operate correctly. Thus it is important that, both in PCEP and the associated signaling protocols, only authorized resource sharing is allowed.

5. IANA Considerations

5.1. New Object Type

IANA manages the PCEP Objects code point registry (see [RFC5440]). This is maintained as the "PCEP Objects" sub-registry of the "Path Computation Element Protocol (PCEP) Numbers" registry.

This document defines a new PCEP object, the RSO object, to be carried in PCReq messages. IANA is requested to make the following allocation in the "PCEP Objects" sub-registry:

<table>
<thead>
<tr>
<th>Object Class</th>
<th>Name</th>
<th>Object Type</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Zhang et al</td>
<td>Expires August 2015</td>
<td>[Page 11]</td>
</tr>
</tbody>
</table>
5.2 RSO flags field

IANA is requested to create and maintain a new sub-registry named "RSO flags". The following flags are defined in this document:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Flag name</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>D</td>
<td>sharing as little as possible</td>
<td>[this document]</td>
</tr>
<tr>
<td>1</td>
<td>R</td>
<td>sharing as much as possible</td>
<td>[this document]</td>
</tr>
</tbody>
</table>

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to indicate requirements levels", RFC 2119, March 1997.


6.2. Informative References


7. Authors’ Addresses

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

Haomian Zheng
Huawei Technologies
Email: zhenghaomian@huawei.com

Oscar Gonzalez de Dios
Telefonica I+D
Don Ramon de la Cruz 82-84
Madrid 28045
Spain
EMail: ogondio@tid.es

Victor Lopez
Telefonica I+D
Don Ramon de la Cruz 82-84
Madrid 28045
Spain
EMail: vlopez@tid.es

Contributor’s Address :

Dhruv Dhody
Huawei Technologies
Email: dhruv.dhody@huawei.com

Igor Bryskin
In certain networks deployment scenarios, service providers would like to keep all the existing MPLS functionalities in both MPLS and GMPLS while removing the complexity of existing signaling protocols such as LDP and RSVP-TE. In [I-D.zhao-pce-central-controller-user-cases], we propose to use the PCE [RFC5440] as a central controller (PCECC) so that LSP can be calculated/ signaled/initiated and label forwarding entries are downloaded through a centralized PCE server to each network devices along the LSP path while leveraging the existing PCE technologies as much as possible.

This draft specify the procedures and PCEP protocol extensions for using the PCE as the central controller and user cases where LSPs are calculated/setup/initiated and label forwarding entries are downloaded through extending the existing PCE architectures and PCEP.

This document also discuss the role of PCECC in Segment Routing (SR).

Status of This Memo

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

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

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

This Internet-Draft will expire on September 3, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the
document authors. All rights reserved.

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

Table of Contents

1. Introduction ................................................. 3
   1.1. Requirements Language ................................. 5
2. Terminology ................................................. 5
3. PCECC Modes ................................................ 5
4. PCEP Requirements ........................................... 6
5. Procedures for Using the PCE as the Central Controller
   (PCECC) .................................................... 6
   5.1. Stateful PCE Model ..................................... 7
   5.2. New LSP Functions ..................................... 7
   5.3. PCECC Capability Advertisement ...................... 8
   5.4. Label Range Reservation ............................... 8
   5.5. LSP Operations ........................................ 9
      5.5.1. Basic PCECC Mode ................................. 9
         5.5.1.1. PCECC LSP Setup ............................ 9
         5.5.1.2. Label Download ............................. 11
         5.5.1.3. Label Cleanup ............................... 11
         5.5.1.4. PCE Initiated PCECC LSP .................... 12
         5.5.1.5. PCECC LSP Update ........................... 13
         5.5.1.6. PCECC LSP State Report ..................... 14
      5.5.2. PCECC Segment Routing (SR) ....................... 15
         5.5.2.1. PCECC SR-BE ................................ 15
         5.5.2.2. PCECC SR-TE ................................ 16
6. PCEP messages ............................................... 17
   6.1. The PCLRResv message ................................ 17
   6.2. The PCLLabelUpd message ............................. 18
7. PCEP Objects ................................................. 19
7.1.  OPEN Object ............................................. 19
  7.1.1.  PCECC Capability TLV .............................. 20
7.2.  LABEL-RANGE Object .................................... 20
7.3.  PATH-SETUP-TYPE TLV ................................... 21
7.4.  Label Object ........................................... 21
  7.4.1.  NextHop TLV ........................................ 22
7.5.  FEC Object .............................................. 24

8.  Security Considerations ................................. 25

9.  Manageability Considerations ............................. 26
  9.1.  Control of Function and Policy ....................... 26
  9.2.  Information and Data Models .......................... 26
  9.3.  Liveness Detection and Monitoring .................... 26
  9.4.  Verify Correct Operations ............................. 26
  9.5.  Requirements On Other Protocols ....................... 26

10.  IANA Considerations .................................... 26
11.  Acknowledgments ......................................... 26
12.  References ............................................... 26
  12.1.  Normative References ................................ 26
  12.2.  Informative References ............................... 27
Authors’ Addresses ........................................... 28

1.  Introduction

In certain network deployment scenarios, service providers would like to have the ability to dynamically adapt to a wide range of customer’s requests for the sake of flexible network service delivery, Software Defined Networks (SDN) has provides additional flexibility in how the network is operated compared to the traditional network.

The existing networking ecosystem has become awfully complex and highly demanding in terms of robustness, performance, scalability, flexibility, agility, etc. By migrating to the SDN enabled network from the existing network, service providers and network operators must have a solution which they can evolve easily from the existing network into the SDN enabled network while keeping the network services remain scalable, guarantee robustness and availability etc.

Taking the smooth transition between traditional network and the new SDN enabled network into account, especially from a cost impact assessment perspective, using the existing PCE components from the current network to function as the central controller of the SDN network is one choice, which not only achieves the goal of having a centralized controller, but also leverages the existing PCE network components.
The Path Computation Element communication Protocol (PCEP) provides mechanisms for Path Computation Elements (PCEs) to perform route computations in response to Path Computation Clients (PCCs) requests. PCEP Extensions for PCE-initiated LSP Setup in a Stateful PCE Model [I-D.ietf-pce-stateful-pce] describes a set of extensions to PCEP to enable active control of MPLS-TE and GMPLS tunnels.

[I-D.ietf-pce-pce-initiated-lsp] describes the setup and teardown of PCE-initiated LSPs under the active stateful PCE model, without the need for local configuration on the PCC, thus allowing for a dynamic MPLS network that is centrally controlled and deployed.

[I-D.ietf-pce-remote-initiated-gmpls-lsp] complements [I-D.ietf-pce-pce-initiated-lsp] by addressing the requirements for remote-initiated GMPLS LSPs.

Segment Routing (SR) technology leverages the source routing and tunneling paradigms. A source node can choose a path without relying on hop-by-hop signaling protocols such as LDP or RSVP-TE. Each path is specified as a set of "segments" advertised by link-state routing protocols (IS-IS or OSPF). [I-D.ietf-spring-segment-routing] provides an introduction to SR technology. The corresponding IS-IS and OSPF extensions are specified in [I-D.ietf-isis-segment-routing-extensions] and [I-D.ietf-ospf-segment-routing-extensions], respectively.

A Segment Routed path (SR path) can be derived from an IGP Shortest Path Tree (SPT). Segment Routed Traffic Engineering paths (SR-TE paths) may not follow IGP SPT. Such paths may be chosen by a suitable network planning tool and provisioned on the source node of the SR-TE path.

It is possible to use a stateful PCE for computing one or more SR-TE paths taking into account various constraints and objective functions. Once a path is chosen, the stateful PCE can instantiate an SR-TE path on a PCC using PCEP extensions specified in [I-D.ietf-pce-pce-initiated-lsp] using the SR specific PCEP extensions described in [I-D.ietf-pce-segment-routing].

PCECC may further use PCEP protocol for SR label distribution instead of IGP extensions with some benefits.

Current MPLS label has local meaning. That is, MPLS label is always allocated by the downstream node to the upstream node. Then the MPLS label is only identified by the neighboring upstream node and downstream node. The label allocation is done locally and signaled through the LDP/RSVP-TE/BGP protocol. To ease the label allocation and signaling mechanism, PCE can be conveniently used as a central
controller with Label download capability. Further PCE can also be used to manage the label range and SRGB etc.

The PCECC solution introduced in [I-D.zhao-pce-central-controller-user-cases] allow for a dynamic MPLS network that is eventually controlled and deployed without the deployment of RSVP-TE protocol or extended IGP protocol with node/adjacency segment identifiers signaling capability while providing all the key MPLS functionalities needed by the service providers.

This draft specify the procedures and PCEP protocol extensions for using the PCE as the central controller and user cases where LSPs are calculated/setup/initiated/downloaded through extending the existing PCE architectures and PCEP.

1.1. Requirements Language

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

2. Terminology

The following terminology is used in this document.


PCC: Path Computation Client: any client application requesting a path computation to be performed by a Path Computation Element.

PCE: Path Computation Element. An entity (component, application, or network node) that is capable of computing a network path or route based on a network graph and applying computational constraints.

TE: Traffic Engineering.

3. PCECC Modes

The following PCECC modes are supported -

 o Basic PCECC.
 o PCECC SR.
   * PCECC SR-BE (Best Effort).
* PCECC SR-TE (Traffic Engineered).

In basic PCECC mode, the forwarding is similar to RSVP-TE signalled LSP without the RSVP-TE signaling. The PCECC allocates and download the label entries along the LSP. The rest of processing is similar to the existing stateful PCE mechanism.

In case of SR, there are two modes for SR-BE and SR-TE. For SR-BE, the forwarding is similar to LDP LSP without LDP signaling or IGP-SR extension. The SR Node label are allocated and distributed in the domain centrally by the PCE via PCEP. Each node (PCC) rely on local IGP for the nexthop calculation. For SR-TE, the forwarding uses label stack similar to IGP based SR-TE without IGP-SR extension. The SR node and adj labels are allocated and distributed in the domain centrally by the PCE via PCEP by PCECC. Rest of the processing is similar to existing stateful PCE with SR mechanism.

4. PCEP Requirements

Following key requirements associated PCECC should be considered when designing the PCECC based solution:

1. PCEP speaker supporting this draft MUST have the capability to advertise its PCECC capability to its peers.

2. Path Computation Client (PCC) supporting this draft MUST have a capability to communicate local label range or global label range or both to PCE.

3. Path Computation Element (PCE) supporting this draft SHOULD have the capability to negotiate a global label range for a group of clients and communicate the final global label range to PCC.

4. PCEP speaker not supporting this draft MUST be able to reject PCECC related message with a reason code that indicates no support for PCECC.

5. PCEP SHOULD provide a means to identify PCECC based LSP in the PCEP messages.

6. PCEP SHOULD provide a means to update (or cleanup) the label-download or label-map entry to the PCC.

5. Procedures for Using the PCE as the Central Controller (PCECC)
5.1. Stateful PCE Model

Active stateful PCE is described in [I-D.ietf-pce-stateful-pce]. PCE as a central controller (PCECC) reuses existing Active stateful PCE mechanism as much as possible to control the LSP.

5.2. New LSP Functions

This document defines the following new PCEP messages and extends the existing messages to support PCECC:

(PCLRResv): a PCEP message sent by a PCC to a PCE to ask for the label range reservation or a PCE to a PCC to send the reserved label range. The PCLRResv message described in Section 6.1.

(PCLabelUpd): a PCEP message sent by a PCE to a PCC to download or cleanup the Label entry. The PCLabelUpd message described in Section 6.2.

(PCRpt): a PCEP message described in [I-D.ietf-pce-stateful-pce]. PCRpt message MAYBE used to send PCECC LSP Reports.

(PCInitiate): a PCEP message described in [I-D.ietf-pce-pce-initiated-lsp]. PCInitiate message is used to setup PCE-Initiated LSP based on PCECC mechanism.

(PCUpd): a PCEP message described in [I-D.ietf-pce-stateful-pce]. PCUpd message is used to send PCECC LSP Update.

The new LSP functions defined in this document are mapped onto the messages as shown in the following table.

<table>
<thead>
<tr>
<th>Function</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCECC Capability advertisement</td>
<td>Open</td>
</tr>
<tr>
<td>Label Range Reservation</td>
<td>PCLRResv</td>
</tr>
<tr>
<td>Label entry Update</td>
<td>PCLabelUpd</td>
</tr>
<tr>
<td>Label entry Cleanup</td>
<td>PCLabelUpd</td>
</tr>
<tr>
<td>PCECC Initiated LSP</td>
<td>PCInitiate</td>
</tr>
<tr>
<td>PCECC LSP Update</td>
<td>PCUpd</td>
</tr>
<tr>
<td>PCECC LSP State Report</td>
<td>PCRpt</td>
</tr>
<tr>
<td>PCECC LSP Delegation</td>
<td>PCRpt</td>
</tr>
</tbody>
</table>
5.3. PCECC Capability Advertisement

During PCEP Initialization Phase, PCEP Speakers (PCE or PCC) advertise their support of PCECC extensions. A PCEP Speaker includes the "PCECC Capability" TLV, described in Section 7.1.1 of this document, in the OPEN Object to advertise its support for PCECC extensions.

The presence of the PCECC Capability TLV in PCC’s OPEN Object indicates that the PCC is willing to function as a PCECC client.

The presence of the PCECC Capability TLV in PCE’s OPEN message indicates that the PCE is interested in function as a PCECC server.

The PCEP protocol extensions for PCECC MUST NOT be used if one or both PCEP Speakers have not included the PCECC Capability TLV in their respective OPEN message. If the PCEP Speakers support the extensions of this draft but did not advertise this capability then a PCErr message with Error-Type=19 (Invalid Operation) and Error-Value=[TBD] (Attempted LSP setup/download/label-range reservation if PCECC capability was not advertised) will be generated and the PCEP session will be terminated.

L flag and G flag defined in PCECC Capability TLV specifies the local and global label range reservation capability.

A PCC or a PCE MUST include both PCECC-CAPABILITY TLV and STATEFUL-PCE-CAPABILITY TLV in OPEN Object to support the extensions defined in this document. If PCECC-CAPABILITY TLV is advertised and STATEFUL-PCE-CAPABILITY TLV is not advertised in OPEN Object, it SHOULD send a PCErr message with Error-Type=19 (Invalid Operation) and Error-value=[TBD] (stateful pce capability was not advertised) and terminate the session.

5.4. Label Range Reservation

After PCEP initial state synchronization, the label range is reserved.

If L flag is advertised in OPEN Object by PCEP speakers, a PCC reserves a local label range and is communicated using PCLRResv message to a PCE. The PCE maintains the local label range of each node and further during LSP setup, a label is assigned to each node from the corresponding local label range reserved.

If G flag is advertised in OPEN Object by PCEP speakers, a PCC reserves a global label range and is advertised in PCLRResv message to a PCE. The PCE MAY negotiate and reserves the global label range.
and also sends the negotiated global label range in PCLRResv message to the PCC. Please refer [I-D.li-mpls-global-label-framework] for MPLS global label allocation.

A PCC MUST send PCLRResv message immediately after the initial LSP synchronization completion. A PCE SHOULD not send PCLLabelUpd message to a PCC before PCLRResv message received. If the PCC received PCLLabelUpd message and not initiated label range reservation, it SHOULD send a PCErr message with Error-type=[TBD] (label range not reserved) and Error-value=[TBD].

The label range reservation sequence is shown below.

```
+---+     +---+
|PCC|     |PCE|
+---+     +---+
      |--- PCLRResv, label type=1 --->|local label range reserved
      |(100-500)                        |global label range negotiated
      |label type=2                     |
      |(600-1000)                       |
      |<--- PCLRResv, label type=2 ---|global label range reserved
      |(700-1000)                       |
```

[Editor’s Note: This section of the document would be updated with more details about Label Block Negotiation, Reservation, Adjustment etc in a future revision of the document.]

5.5. LSP Operations

The PCEP messages pertaining to PCECC MUST include PATH-SETUP-TYPE TLV [I-D.sivabalan-pce-lsp-setup-type] in the SRP object to clearly identify the PCECC LSP is intended.

5.5.1. Basic PCECC Mode

5.5.1.1. PCECC LSP Setup

In order to setup a LSP based on PCECC mechanism, a PCC MUST delegate the LSP by sending a PCCRpt message with Path Setup Type set for basic PCECC (see Section 7.3) and D (Delegate) flag (see [I-D.ietf-pce-stateful-pce]) set in the LSP object.
The LSP-ID in LSP-IDENTIFIER TLV (which usually corresponds to the RSVP-TE LSP-ID) for PCECC LSP MUST always be generated by the PCE. In the first PCRpt message of PCECC LSP, LSP ID of LSP-IDENTIFIER TLV is set to zero.

When a PCE received PCRpt message with P and D flags set, it generates LSP ID; calculates the path and assign labels along the path; and setup the path by sending PCLabelUpd message to each node along the path of the LSP.

The PCE SHOULD send the PCUpd message with the same PLSP-ID to the Ingress PCC in response to the delegate PCRpt message.

The PCECC LSPs MUST be delegated to a PCE at all times.

LSP deletion operation for PCECC LSP is same as defined in [I-D.ietf-pce-stateful-pce]. If the PCE received PCRpt message for LSP deletion then it does Label cleanup operation as described in Section 5.5.1.3 for the corresponding LSP.

The Basic PCECC LSP setup sequence is as shown below.

```
+-------+                           +-------+
   |PCC   |                           |  PCE  |
   |Ingress|                           +-------+
+------|       |                               |
   | PCC  +-------+                               |
   | Transit|       | |--- PCRpt,PLSP-ID=1, P=1, D=1  --->| PCECC LSP |
   |       |       |     |                                   |
   |PCC   |  |     |                                   |
   |Egress  |  |     |                                   |
+--------+  |     |                                   |
     |<------ PLabelUpd,PLSP-ID=1  ------------------ | Label  
     |       |     |       (LSP ID=1)                  | download |
     |       |     |                                   |
     |       |<----- PLabelUpd,PLSP-ID=1  ----------- | Label  
     |       |     |      (LSP ID=1)                   | download |
     |       |     |                                   |
     |       |<-- PCUpd,PLSP-ID=1, P=1, D=1 ---- | PCECC LSP  
     |       |     |      (LSP ID=1)                   | Update   |
```
The PCECC LSP are considered to be ‘up’ by default. The Ingress MAY further choose to deploy a data plane check mechanism and report the status back to the PCE via PCRpt message.

5.5.1.2. Label Download

In order to setup an LSP based on PCECC, the PCE sends a PCLabelUpd message to each node of the LSP to download the Label entry as described in Section 5.5.1.1.

The LSP object in PCLabelUpd MUST include the LSP-IDENTIFIER TLV.

If a node (PCC) received a PCLabelUpd message but failed to download the Label entry, it MUST send a PCErr message with Error-type=[TBD] (label download failed) and Error-value=[TBD].

5.5.1.3. Label Cleanup

In order to delete an LSP based on PCECC, the PCE sends a PCLabelUpd message to each node along the path of the LSP to cleanup the Label entry.

If the PCC received a PCLabelUpd message but does not recognize the label, the PCC MUST generate a PCErr message with Error-Type 19(Invalid operation) and Error-Value=3, "Unknown Label".

The R flag in SRP object defined in [I-D.ietf-pce-pce-initiated-lsp] specifies the deletion of Label Entry in the PCLabelUpd message.
5.5.1.4. PCE Initiated PCECC LSP

The LSP Instantiation operation is same as defined in [I-D.ietf-pce-pce-initiated-lsp].

In order to setup a PCE Initiated LSP based on PCECC mechanism, a PCE sends PCInitiate message with Path Setup Type set for basic PCECC (see Section 7.3) to the Ingress PCC.

The Ingress PCC MUST also set D (Delegate) flag (see [I-D.ietf-pce-stateful-pce]) and C (Create) flag (see [I-D.ietf-pce-pce-initiated-lsp]) in LSP object of PCRpt message. The PCC responds with first PCRpt message with the status as "GOING-UP" and assigned PLSP-ID.

The rest of the PCECC LSP setup operations are same as those described in Section 5.5.1.1.

The LSP deletion operation for PCE Initiated PCECC LSP is same as defined in [I-D.ietf-pce-pce-initiated-lsp]. The PCE should further perform Label entry cleanup operation as described in Section 5.5.1.3 for the corresponding LSP.

The PCE Initiated PCECC LSP setup sequence is shown below.
5.5.1.5. PCECC LSP Update

In case of a modification of PCECC LSP with a new path, a PCE sends a PCUpd message to the Ingress PCC.

When a PCC received a PCUpd message for an existing LSP, a PCC MUST follow the make-before-break procedure. On successful traffic switch over to the new LSP, PCC sends a PCRpt message to the PCE for the deletion of old LSP. Further the PCE does cleanup operation for the old LSP described in Section 5.5.1.3.

The PCECC LSP Update and make-before-break sequence is shown below.
The modified PCECC LSP are considered to be 'up' by default. The Ingress MAY further choose to deploy a data plane check mechanism and report the status back to the PCE via PCRpt message.

5.5.1.6. PCECC LSP State Report

As mentioned before, an Ingress PCC MAY choose to apply any OAM mechanism to check the status of LSP in the Data plane and MAY further send its status in PCRpt message to the PCE.
5.5.2. PCECC Segment Routing (SR)

Segment Routing (SR) as described in [I-D.ietf-spring-segment-routing] depends on "segments" that are advertised by Interior Gateway Protocols (IGPs). The SR-node allocate and advertise the SID (node, adj etc) and flood via the IGP. This document proposes a new mechanism where PCE allocate the SID (label) centrally and uses PCEP to advertise the SID. In some deployments PCE (and PCEP) are better suited than IGP because of centralized nature of PCE and direct TCP based PCEP session to the node.

5.5.2.1. PCECC SR-BE

Each node (PCC) is allocated a node-SID (label) by the PCECC. The PCECC sends PCLabelUpd to update the label map of each node to all the nodes in the domain. Each node (PCC) uses the local information to determines the next-hop and download the label forwarding instructions accordingly. The PCLabelUpd message in this case MUST not have LSP object but uses new FEC object.

```
+-------+                           +-------+
|PCC    |                           |  PCE  |
|3.3.3.3|                           +-------+
+------|       |                               |
| PCC  +-------+                               |
| 2.2.2.2| |                                   |
+------|        | |                                   |
|PCC   +--------+ |                                   |
|1.1.1.1 |  |     |                                   |
+--------+  |     |                                   |
            |     |                                   |
<------ PCLabelUpd, FEC=1.1.1.1----------------- | Label Map
|Find                                                 update
|NextHop locally                                       |
|<---- PCLabelUpd, FEC=1.1.1.1-----------------     |
|Label=X                                                |
<--- PCLabelUpd, FEC=1.1.1.1------                  |
|Label=X                                                |
|Label Map update                                      |
```

The forwarding behavior and the end result is similar to IGP based "Node-SID" in SR. Thus, from anywhere in the domain, it enforces the ECMP-aware shortest-path forwarding of the packet towards the related node.
PCE rely on the Node label cleanup using the same PCLabelUpd message.

5.5.2.2. PCECC SR-TE

A Segment Routed Best Effort path (SR-BE path) can be derived from an IGP Shortest Path Tree (SPT) as explained above. On the other hand, SR-TE paths may not follow IGP SPT. Such paths may be chosen by a PCE and provisioned on the source node of the SR-TE path.

[I-D.ietf-pce-segment-routing] extends PCEP to allow a stateful PCE to compute and initiate SR-TE paths, as well as a PCC to request a path subject to certain constraint(s) and optimization criteria in SR networks.

For SR-TE, apart from node-SID, Adj-SID is used where each adjacency is allocated an Adj-SID (label) by the PCECC. The PCECC sends PCLabelUpd to update the label map of each Adj to the corresponding nodes in the domain. Each node (PCC) download the label forwarding instructions accordingly. Similar to SR-BE, the PCLabelUpd message in this case MUST not have LSP object but uses new FEC object.

```
+-------+                           +-------+
|PCC    |                           |  PCE  |
|3.3.3.3|                           +-------+
+------|       |                               |
| PCC  +-------+                               |
| 2.2.2.2| |                                   |
|------|        | |                                   |
|PCC   +--------+ |                                   |
|1.1.1.1 |     |                                   |
|------|     |                                   |
|<------ PCLabelUpd, FEC=10.1.1.1 / ------------- | Label Map
|       |     |          10.1.1.2                 | update
|       |     |             Label=A               |
|       |<----- PCLabelUpd, FEC=10.1.1.2--------- | Label Map
|       |     |                 10.1.1.1          | update
|       |     |             Label=B               |
|       |     |                                   |
```

The forwarding behavior and the end result is similar to IGP based "Adj-SID" in SR.
The Path Setup Type MUST be set for PCECC SR-TE (see Section 7.3). The rest of the PCEP procedures and mechanism are similar to [I-D.ietf-pce-segment-routing].

PCEP relies on the Adj label cleanup using the same PCLLabelUpd message.

6. PCEP messages

As defined in [RFC5440], a PCEP message consists of a common header followed by a variable-length body made of a set of objects that can be either mandatory or optional. An object is said to be mandatory in a PCEP message when the object must be included for the message to be considered valid. For each PCEP message type, a set of rules is defined that specify the set of objects that the message can carry. An implementation MUST form the PCEP messages using the object ordering specified in this document.

6.1. The PCLRResv message

A Label Range Reservation message (also referred to as PCLRResv message) is a PCEP message sent by a PCC to a PCE for the reservation of label range or by PCE to PCC to send reserved label range for the network. The Message-Type field of the PCEP common header for the PCLRResv message is set to [TBD].

The format of a PCLRResv message is as follows:

PCLRResv Message>::= <Common Header>
   <label-range>
   Where:
   <label-range> ::= <SRP>
   <labelrange-list>
   Where
   <labelrange-list>::=<LABEL-RANGE>[<labelrange-list>]

There are two mandatory objects that MUST be included within each <label-range> in the PCLRResv message: the SRP Object and LABEL-RANGE object.

SRP Object is defined in [I-D.ietf-pce-stateful-pce] and this document extends the use of SRP object in PCLRResv message. If the SRP object is missing, the receiving PCE MUST send a PCErr message with Error-type=6 (Mandatory Object missing) and Error-value=10 (SRP object missing).
PCC generates the value of SRP-ID-number in SRP object of PCLRResv message send to a PCE. The PCE MUST include the same SRP-ID-number in SRP object of PCLRResv message sent to the PCC in response to PCLRResv message.

LABEL-RANGE object is defined in Section 7.2. If the LABEL-RANGE object is missing, the receiving PCE MUST send a PCErr message with Error-type=6 (Mandatory Object missing) and Error-value=[TBD] (Label object missing).

[Editor’s Note: This section of the document would be updated with more details about Label Block Negotiation, Reservation, Adjustment etc in a future revision of the document.]

6.2. The PCLabelUpd message

The Label Update Message (also referred to as PCLabelUpd) is a PCEP message sent by a PCE to a PCC to download label or update the label map. The same message is also used to cleanup the Label entry. The Message-Type field of the PCEP common header for the PCLabelUpd message is set to [TBD].

The format of the PCLabelUpd message is as follows:

```plaintext
<PCLabelUpd Message> ::= <Common Header>
                <pce-label-update-list>
Where:
<pce-label-update-list> ::= <pce-label-update>
                     [<pce-label-update-list>]
<pce-label-update> ::= (<pce-label-download>|<pce-label-map>)
Where:
<pce-label-download> ::= <SRP>
                        <LSP>
                        <label-list>
<pce-label-map> ::= <SRP>
                    <LABEL>
                    <FEC>
<label-list> ::=  <LABEL>
                    [<label-list>]
```
The PCLabelUpd message is used to download label along the path of the LSP for the basic PCECC mode, as well as to update the label map for the Node and Adjacency Label in case of SR.

The SRP object is defined in [I-D.ietf-pce-stateful-pce] and this document extends the use of SRP object in PCLabelUpd message. The SRP object is mandatory and MUST be included in PCLabelUpd message. If the SRP object is missing, the receiving PCC MUST send a PCErr message with Error-type=6 (Mandatory Object missing) and Error-value=10 (SRP object missing).

The LSP object is defined in [I-D.ietf-pce-stateful-pce] and this document extends the use of LSP object in PCLabelUpd message. The LSP is an optional object and used in the basic PCECC mode in PCLabelUpd message. LSP Identifiers TLV is defined in [I-D.ietf-pce-stateful-pce], it MUST be included in the LSP object in PCLabelUpd message. If the TLV is missing, the PCC will generate a PCErr message with Error-Type=6 (mandatory object missing) and Error-Value=11 (LSP-IDENTIFIERS TLV missing) and close the session.

The LABEL object is defined in Section 7.4. The LABEL is the mandatory object and MUST be included in PCLabelUpd message. If the LABEL object is missing, the receiving PCC MUST send a PCErr message with Error-type=6 (Mandatory Object missing) and Error-value=[TBD] (LABEL object missing). More than one LABEL object MAY be included in the PCLabelUpd message for the transit LSR.

The FEC object is defined in Section 7.5. The FEC is an optional object and used in PCECC SR mode in PCLabelUpd message. The FEC object encodes the Node and Adjacency information of the Label Map.

To cleanup the SRP object must set the R (remove) bit.

7. PCEP Objects

The PCEP objects defined in this document are compliant with the PCEP object format defined in [RFC5440]. The P flag and the I flag of the PCEP objects defined in this document MUST always be set to 0 on transmission and MUST be ignored on receipt since these flags are exclusively related to path computation requests.

7.1. OPEN Object

This document defines a new optional TLV for use in the OPEN Object.
7.1.1. PCECC Capability TLV

The PCECC-CAPABILITY TLV is an optional TLV for use in the OPEN Object for PCECC capability advertisement. Advertisement of the PCECC capability implies support of LSPs that are setup through PCECC as per PCEP extensions defined in this document.

Its format is shown in the following figure:

```
+-----------------------------------------------+                     +-----------------------------------------------+
|               Type=[TBD]          |            Length=4           |                     |               Flags                         |G|L|
+-----------------------------------------------+                     +-----------------------------------------------+
```

The type of the TLV is [TBD] and it has a fixed length of 4 octets.

The value comprises a single field - Flags (32 bits):

L (LOCAL-LABEL-RANGE-CAPABILITY – 1 bit): If set to 1 by a PCEP speaker, it indicates that the PCEP speaker is capable for local label range reservation.

G (GLOBAL-LABEL-RANGE-CAPABILITY – 1 bit): If set to 1 by a PCEP speaker, it indicates that the PCEP speaker capable for global label range reservation.

Unassigned bits are considered reserved. They MUST be set to 0 on transmission and MUST be ignored on receipt.

7.2. LABEL-RANGE Object

The LABEL-RANGE object MUST be carried within PCLRResv message. The LABEL-RANGE object is used to carry the label range information based on the label type.

LABEL-RANGE Object-Class is TBD.

LABEL-RANGE Object-Type is 1.
label type (8 bit): The values defined for label type are label type
1 specifies the local label. It means the label range is non
negotiable. label type 2 specifies the global label. It means
the label range is negotiable. Refer

Range size (24 bit): It specifies the size of label range.
Label base (32 bit): It specifies the minimum label of label range.

7.3. PATH-SETUP-TYPE TLV

The PATH-SETUP-TYPE TLV is defined in
[I-D.sivabalans-pce-lsp-setup-type]; this document defines following
new PST value:

- PST = 2: Path is setup via Basic PCECC mode.
- PST = 3: Path is setup via PCECC SR-TE mode.

On a PCRpt or PCInitiate message, the PST=2 in PATH-SETUP-TYPE TLV in
SRP object indicates that this LSP was setup via a basic PCECC based
mechanism; the PST=3 indicates that this LSP was setup via a PCECC
SR-TE based mechanism.

7.4. Label Object

The LABEL Object is used to specify the Label information and MUST be
carried within PCLabelUpd message.

LABEL Object-Class is TBD.
LABEL Object-Type is 1.
The fields in the LABEL object are as follows:

Flags: is used to carry any additional information pertaining to the label. Currently, the following flag bit is defined:

* O bit (Out-label) : if the bit is set it specifies the label is the OUT label and it is mandatory to encode the nexthop information (via NEXTHOP-IPV4-ADDRESS TLV or NEXTHOP-IPV6-ADDRESS TLV or NEXTHOP-UNNUMBERED-IPV4-ID TLV in LABEL object).

Label (32-bit): The Label information encoded such that the 20 rightmost bits represent a label.

7.4.1. NextHop TLV

This document defines the following TLV for the LABEL object to associate the nexthop information incase of an outgoing label.
The NextHop TLVs are as follows:

**NEXTHOP-IPV4-ADDRESS TLV:** where Nexthop IPv4 address is specified as an IPv4 address of the Nexthop.

**NEXTHOP-IPV6-ADDRESS TLV:** where Nexthop IPv6 address is specified as an IPv6 address of the Nexthop.

**NEXTHOP-UNNUMBERED-IPV4-ID TLV:** where a pair of Node ID / Interface ID tuples is used for the Nexthop.
7.5. FEC Object

The FEC Object is used to specify the FEC information and MAY be carried within PCLabelUpd message.

FEC Object-Class is TBD.

FEC Object-Type is 1 ‘IPv4 Node ID’.

FEC Object-Type is 2 ‘IPv6 Node ID’.

FEC Object-Type is 3 ‘IPv4 Adjacency’.

FEC Object-Type is 4 ‘IPv6 Adjacency’.

FEC Object-Type is 5 ‘Unnumbered Adjacency with IPv4 NodeIDs’.  

```
+-------------------------------------------------------------+
<table>
<thead>
<tr>
<th>Local Node-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local Interface ID</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>Remote Node-ID</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
</tr>
<tr>
<td>Remote Interface ID</td>
</tr>
<tr>
<td>-------------------------------------------------------------</td>
</tr>
</tbody>
</table>
```

The FEC objects are as follows:

**IPv4 Node ID:** where IPv4 Node ID is specified as an IPv4 address of the Node. FEC Object-type is 1, and the Object-Length is 4 in this case.

**IPv6 Node ID:** where IPv6 Node ID is specified as an IPv6 address of the Node. FEC Object-type is 2, and the Object-Length is 16 in this case.

**IPv4 Adjacency:** where Local and Remote IPv4 address is specified as pair of IPv4 address of the adjacency. FEC Object-type is 3, and the Object-Length is 8 in this case.

**IPv6 Adjacency:** where Local and Remote IPv6 address is specified as pair of IPv6 address of the adjacency. FEC Object-type is 4, and the Object-Length is 32 in this case.

**Unnumbered Adjacency with IPv4 NodeID:** where a pair of Node ID / Interface ID tuples is used. FEC Object-type is 5, and the Object-Length is 16 in this case.

8. Security Considerations

TBD
9. Manageability Considerations

9.1. Control of Function and Policy

TBD.

9.2. Information and Data Models

TBD.

9.3. Liveness Detection and Monitoring

TBD.

9.4. Verify Correct Operations

TBD.

9.5. Requirements On Other Protocols

TBD.


TBD.

10. IANA Considerations

TBD

11. Acknowledgments

We would like to thank Robert Tao, Changjing Yan, Tieying Huang for their useful comments and suggestions.

12. References

12.1. Normative References


[I-D.ietf-pce-stateful-pce]

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

12.2. Informative References

[I-D.li-mpls-global-label-framework]

[I-D.zhao-pce-central-controller-user-cases]
Zhao, Q., Zhao, K., and Z. Ke, "The Use Cases for Using PCE as the Central Controller(PCECC) of LSPs", draft-zhao-pce-central-controller-user-cases-01 (work in progress), July 2014.

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

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

[I-D.ietf-isis-segment-routing-extensions]
[I-D.ietf-ospf-segment-routing-extensions]

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

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

Authors’ Addresses

Quintin Zhao
Huawei Technologies
125 Nagog Technology Park
Acton, MA  01719
USA
EMail: quintin.zhao@huawei.com

Katherine Zhao
Huawei Technologies
2330 Central Expressway
Santa Clara, CA  95050
USA
EMail: katherine.zhao@huawei.com

Zhenbin Li
Huawei Technologies
Huawei Bld., No.156 BeiQing Rd.
Beijing    100095
China
EMail: lizhenbin@huawei.com

Dhruv Dhody
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560037
India
EMail: dhruv.ietf@gmail.com

Udayasree Palle
Huawei Technologies
Divyashree Techno Park, Whitefield
Bangalore, Karnataka 560037
India
EMail: udayasree.palle@huawei.com

Boris Zhang
Telus Ltd.
Toronto
Canada
EMail: boris.zhang@telus.com