Networking Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 6, 2008

JP. Vasseur, Ed.
Cisco Systems
JL. Le Roux, Ed.
France Telecom

July 5, 2007

# Path Computation Element (PCE) communication Protocol (PCEP) draft-ietf-pce-pcep-08.txt

## Status of this Memo



## Copyright Notice



## Abstract

This document specifies the Path Computation Element communication Protocol (PCEP) for communications between a Path Computation Client (PCC) and a Path Computation Element (PCE), or between two PCEs. Such interactions include path computation requests and path computation replies as well as notifications of specific states related to the use of a PCE in the context of Multiprotocol Label Switching (MPLS) and Generalized (GMPLS) Traffic Engineering. The

PCEP protocol is designed to be flexible and extensible so as to easily allow for the addition of further messages and objects, should further requirements be expressed in the future.

## Requirements Language

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

## Table of Contents

# 1. Terminology

Terminology used in this document

AS: Autonomous System.

Explicit path: full explicit path from start to destination made of a list of strict hops where a hop may be an abstract node such as an AS.

IGP area: OSPF area or IS-IS level.

Inter-domain TE LSP: A TE LSP whose path transits across at least two different domains where a domain can either be an IGP area, an Autonomous System or a sub-AS (BGP confederations).

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.

PCEP Peer: an element involved in a PCEP session (i.e. a PCC or a PCE).

TED: Traffic Engineering Database that contains the topology and resource information of the domain. The TED may be fed by IGP extensions or potentially by other means.

TE LSP: Traffic Engineering Label Switched Path.

Strict/loose path: mix of strict and loose hops comprising of at least one loose hop representing the destination where a hop may be an abstract node such as an AS.

Within this document, when describing PCE-PCE communications, the requesting PCE fills the role of a PCC. This provides a saving in documentation without loss of function.

## 2. Introduction

[RFC4655] describes the motivations and architecture for a PCE-based model for the computation of MPLS and GMPLS TE LSPs. The model allows for the separation of PCE from PCC, and allows for the cooperation between PCEs. This necessitates a communication protocol between PCC and PCE, and between PCEs. [RFC4657] states the generic





requirements for such protocol including the requirement for using the same protocol between PCC and PCE, and between PCEs. Additional application-specific requirements (for scenarios such as inter-area, inter-AS, etc.) are not included in [RFC4657], but there is a requirement that any solution protocol must be easily extensible to handle other requirements as they are introduced in application-specific requirements documents. Examples of such application-specific requirements are [RFC4927], [I-D.ietf-pce-interas-pcecp-reqs] and [I-D.ietf-pce-inter-layer-req].

This document specifies the Path Computation Element communication Protocol (PCEP) for communications between a Path Computation Client (PCC) and a Path Computation Element (PCE), or between two PCEs, in compliance with [RFC4657]. Such 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 MPLS and GMPLS Traffic Engineering.

PCEP is designed to be flexible and extensible so as to easily allow for the addition of further messages and objects, should further requirements be expressed in the future.

## 3. Assumptions

[RFC4655] describes various types of PCE. PCEP does not make any assumption and thus does not impose any constraint on the nature of the PCE.

Moreover, it is assumed that the PCE gets the required information so as to perform the computation of TE LSP that usually requires network topology and resource information. Such information can be gathered by routing protocols or by some other means, the gathering of which is out of the scope of this document.

Similarly, no assumption is made on the discovery method used by a PCC to discover a set of PCEs (e.g. via static configuration or dynamic discovery) and on the algorithm used to select a PCE. For the sake of reference [RFC4674] defines a list of requirements for dynamic PCE discovery and IGP-based solutions for such PCE discovery are specified in [I-D.ietf-pce-disco-proto-ospf] and [I-D.ietf-pce-disco-proto-isis].

## 4. Architectural Protocol Overview (Model)

The aim of this section is to describe the PCEP model in the spirit of [RFC4101]. An architecture protocol overview (the big picture of





the protocol) is provided in this section. Protocol details can be found in further sections.

### 4.1. Problem

The PCE-based architecture used for the computation of MPLS and GMPLS TE LSP is described in [RFC4655]. When the PCC and the PCE are not collocated, a communication protocol between the PCC and the PCE is needed. PCEP is such a protocol designed specifically for communications between a PCC and a PCE or between two PCEs in compliance with [RFC4657]: a PCC may use PCEP to send a path computation request for one or more TE LSP(s) to a PCE and the PCE may reply with a set of computed path(s) if one or more path(s) that satisfy the set of constraints can be found.

## 4.2. Architectural Protocol Overview

PCEP operates over TCP, which fulfils the requirements for reliable messaging and flow control without further protocol work.

Several PCEP messages are defined:

- Open and Keepalive messages are used to initiate and maintain a PCEP session respectively.

- PCReq: a PCEP message sent by a PCC to a PCE to request a path computation.

- PCRep: a PCEP message sent by a PCE to a PCC in reply to a path computation request. A PCRep message can either contain a set of computed path(s) if the request can be satisfied or a negative reply otherwise in which case the negative reply may also indicate the reason why no path could be found.

- PCNtf: a PCEP notification message either sent by a PCC to a PCE or a PCE to a PCC to notify of a specific event.

- PCErr: a PCEP message sent upon the occurrence of a protocol error condition.

- Close message: a message used to close a PCEP session.

The set of available PCE(s) may either be statically configured on a PCC or dynamically discovered. The mechanisms used to discover one or more PCE(s) and to select a PCE are out of the scope of this document.

A PCC may have PCEP sessions with more than one PCE and similarly a





PCE may have PCEP sessions with multiple PCCs.

### 4.2.1. Initialization Phase

The initialization phase consists of two successive steps (described in a schematic form in Figure 1):

1) Establishment of a TCP connection (3-way handshake) between the

PCC and the PCE.

2) Establishment of a PCEP session over the TCP connection.

Once the TCP connection is established, the PCC and the PCE (also referred to as "PCEP peers") initiate a PCEP session establishment during which various session parameters are negotiated. These parameters are carried within Open messages and include the Keepalive timer, the Deadtimer and potentially other detailed capabilities and policy rules that specify the conditions under which path computation requests may be sent to the PCE. If the PCEP session establishment phase fails because the PCEP peers disagree on the session parameters or one of the PCEP peers does not answer after the expiration of the establishment timer, the TCP connection is immediately closed. Successive retries are permitted but an implementation should make use of an exponential back-off session establishment retry procedure.

Keepalive messages are used to acknowledge Open messages and once the PCEP session has been successfully established, Keepalive messages may be exchanged between PCEP peers to ensure the liveness of the PCEP session.

A single PCEP session can exist between a pair a PCEP peers.

Details about the Open message and the Keepalive messages can be found inSection 6.2 and Section 6.3 respectively.



+-+-+ +-+-+

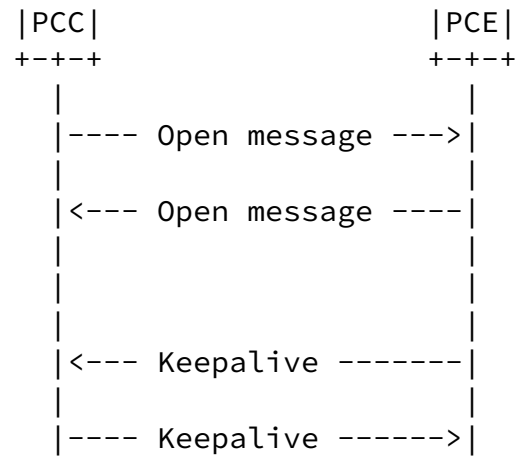



Figure 1: PCEP Initialization phase (initiated by a PCC)

(Note that the exchange of Keepalive messages is optional)

### 4.2.2. Path computation request sent by a PCC to a PCE

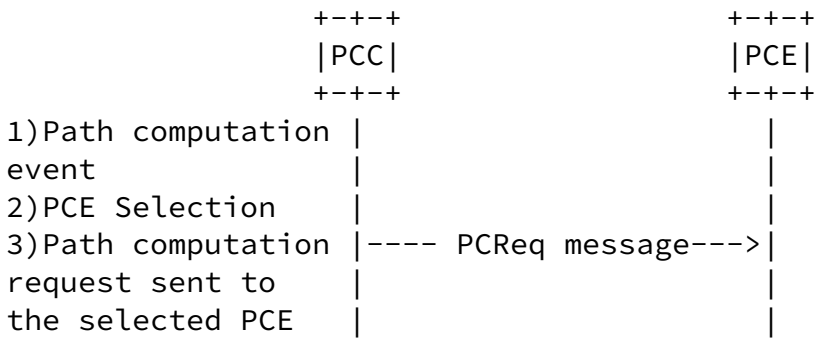



Figure 2: Path computation request

Once a PCC has successfully established a PCEP session with one or more PCEs, if an event is triggered that requires the computation of a set of path(s), the PCC first selects one or more PCE(s). Note that the PCE selection decision process may have taken place prior to the PCEP session establishment.

Once the PCC has selected a PCE, it sends a path computation request to the PCE (PCReq message) that contains a variety of objects that specify the set of constraints and attributes for the path to be computed. For example "Compute a TE LSP path with source IP address=x.y.z.t, destination IP address=x'.y'.z'.t', bandwidth=B Mbit/s, Setup/Hold priority=P, ...". Additionally, the PCC may desire to specify the urgency of such request by assigning a request

priority. Each request is uniquely identified by a request-id number and the PCC-PCE address pair. The process is shown in a schematic form in figure 2.

Details about the PCReq message can be found in Section 6.4

4.2.3. Path computation reply sent by the PCE to a PCC

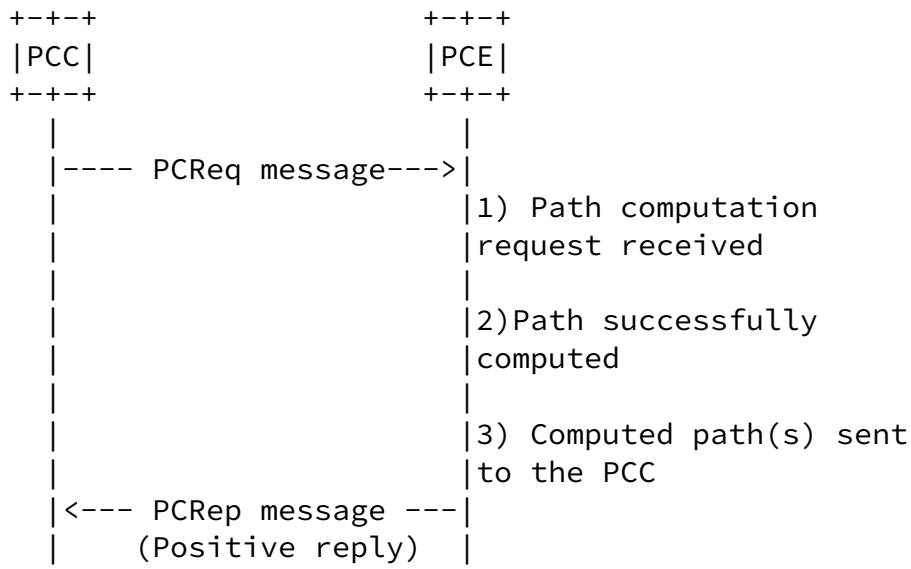



Figure 3a: Path computation request with successful path computation

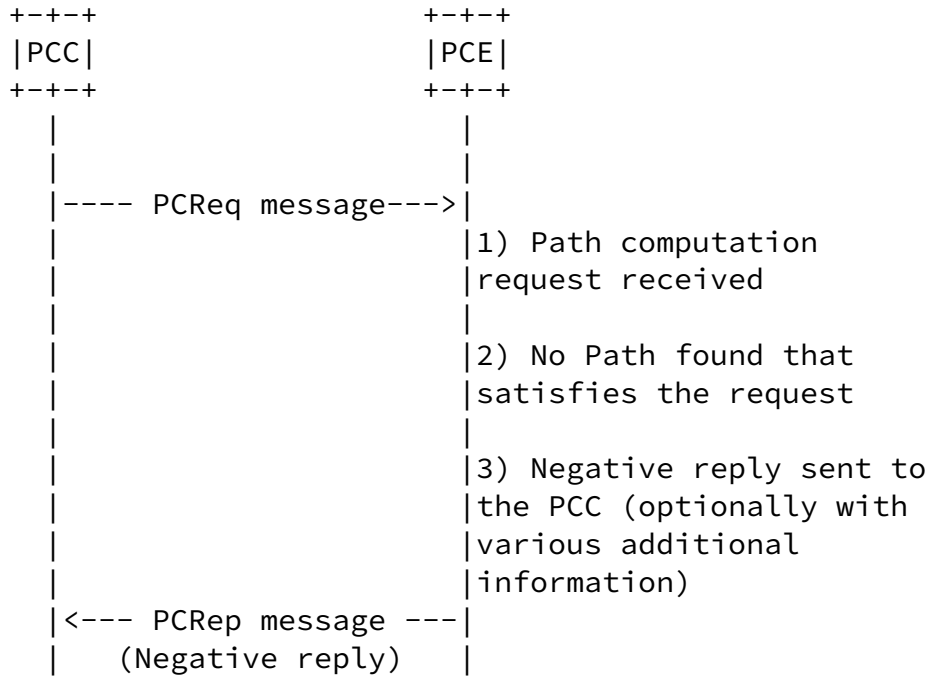



Figure 3b: Path computation request with unsuccessful path computation

Upon receiving a path computation request from a PCC, the PCE triggers a path computation, the result of which can either be:

- Positive (Figure 3-a): the PCE manages to compute a path that satisfies the set of required constraints, in which case the PCE returns the set of computed path(s) to the requesting PCC. Note that PCEP supports the capability to send a single request that requires the computation of more than one path (e.g. computation





of a set of link-diverse paths).

- Negative (Figure 3-b): no path could be found that satisfies the set of constraints. In this case, a PCE may provide the set of constraints that led to the path computation failure. Upon receiving a negative reply, a PCC may decide to resend a modified request or take any other appropriate action.

Details about the PCRep message can be found in Section 6.5.

#### 4.2.4. Notification

There are several circumstances whereby a PCE may want to notify a PCC of a specific event. For example, suppose that the PCE suddenly gets overloaded thus potentially leading to unacceptable response times. The PCE may want to notify one or more PCCs that some of their requests (listed in the notification) will not be satisfied or may experience unacceptable delays. Upon receiving such notification, the PCC may decide to redirect it(s) path computation request(s) to another PCE should an alternate PCE be available. Similarly, a PCC may desire to notify a PCE of a particular event such as the cancellation of pending request(s).

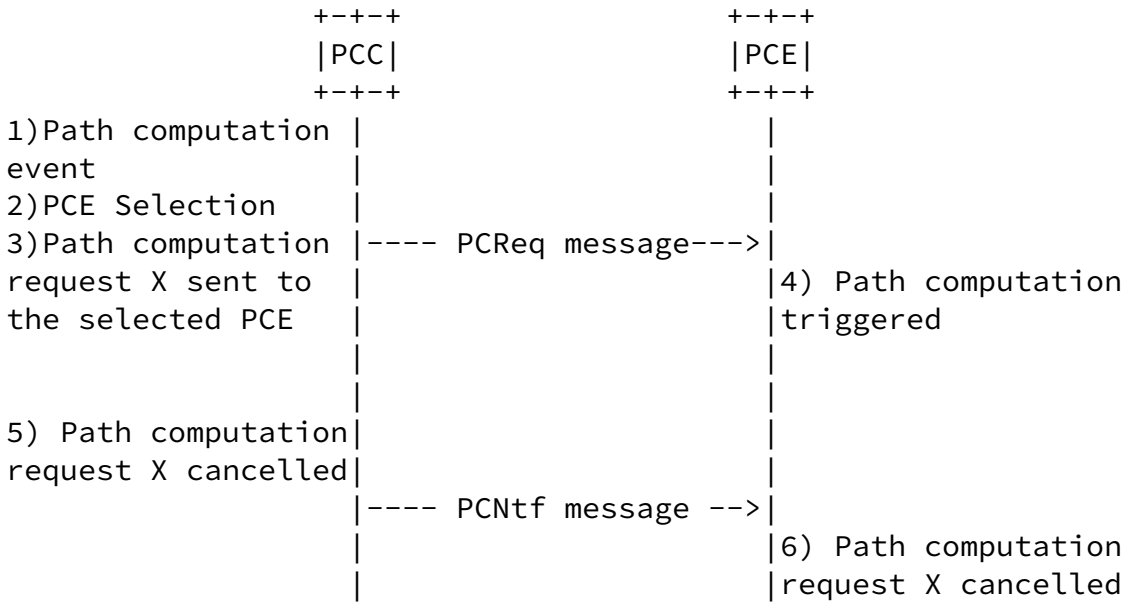



Figure 4: Example of PCC notification (cancellation notification) sent to a PCE

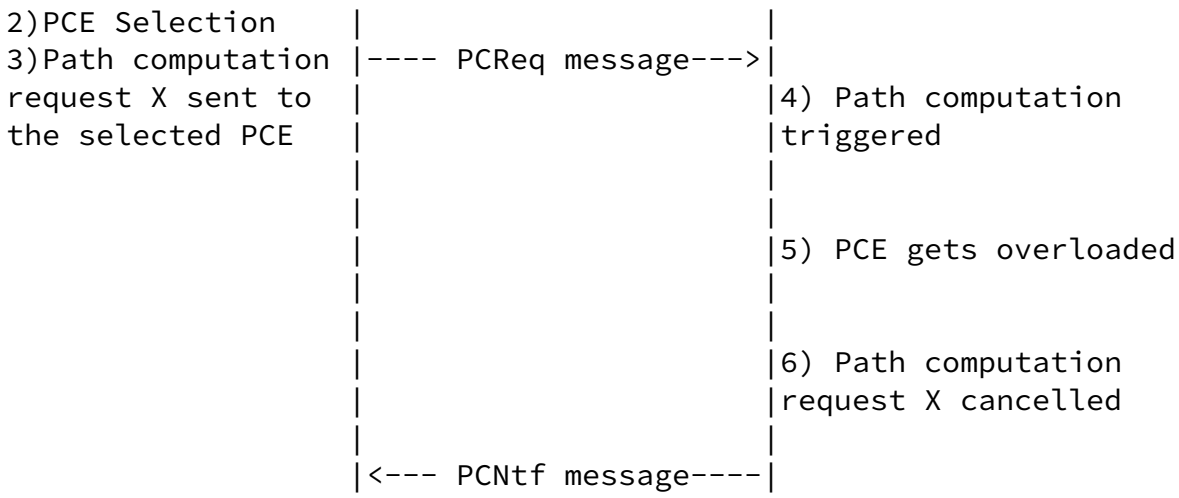



Figure 5: Example of PCE notification (cancellation notification) sent to a PCC

Details about the PCNtf message can be found in Section 6.6.

### 4.2.5. Error

PCEP Error messages are sent when a protocol error condition is met (e.g. unknown object, non supported object, policy violation, ...).





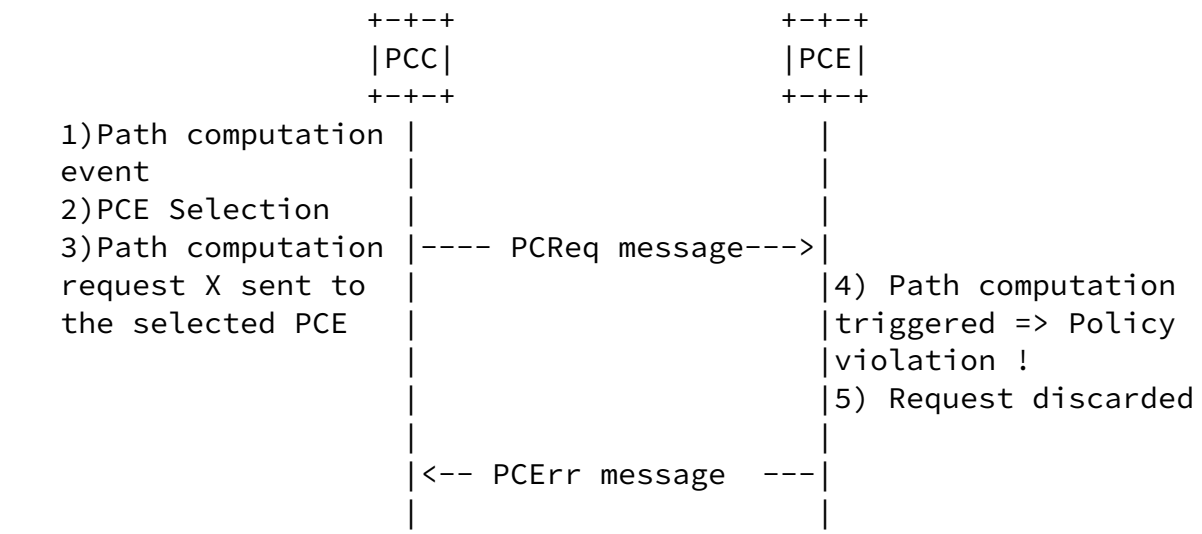



Figure 6: Example of Error message (policy violation) sent by a PCE

Details about the PCErr message can be found in Section 6.7.

### 4.2.6. Termination of the PCEP Session

When one of the PCEP peers desires to terminate a PCEP session it first sends a PCEP Close message and then closes the TCP connection. If the PCEP session is terminated by the PCE, the PCC clears all the states related to pending requests previously sent to the PCE. Similarly, if the PCC terminates a PCEP session the PCE clears all pending path computation requests sent by the PCC in question as well as the related states. A Close message can only be sent to terminate a PCEP session if the PCEP session has previously been established.

In case of TCP connection failure, the PCEP session is immediately terminated.

Details about the Close message can be found in Section 6.8.

## 5. Transport protocol

PCEP operates over TCP using a well-known TCP port (to be assigned by IANA). This allows the requirements of reliable messaging and flow control to be met without further protocol work.

An implementation may decide to keep the TCP connection alive for an unlimited time (this may for instance be appropriate when path computation requests are sent on a frequent basis so as to avoid to open a TCP connection each time a path computation request is needed, which would incur additional processing delays). Conversely, in some other circumstances, it may be desirable to systematically open and





close the TCP connection for each PCEP request (for instance when sending a path computation request is a rare event).

## 6. PCEP Messages

A PCEP message consists of a common header followed by a variable length body made of a set of objects that can either be mandatory or optional. In the context of this document, an object is said to be mandatory in a PCEP message when the object MUST be included for the message to be considered as valid. A PCEP message with a missing

mandatory object MUST trigger an Error message (see Section 7.14). Conversely, if an object is optional, the object may or may not be present.

A flag referred to as the P flag is defined in the common header of each PCEP object (see Section 7.1) that can be set by a PCEP peer to enforce a PCE to take into account the related information during the path computation. For example, the METRIC object defined in Section 7.7 allows a PCC to specify a bounded acceptable path cost. The METRIC object is optional but a PCC may set a flag to ensure that such constraint is taken into account. Similarly to the previous case, if such constraint cannot be taken into account by the PCE, the PCE MUST trigger an Error message.

For each PCEP message type, rules are defined that specify the set of objects that the message can carry. We use the Backus-Naur Form (BNF) to specify such rules. Square brackets refer to optional sub-sequences. An implementation MUST form the PCEP messages using the object ordering specified in this document.

## 6.1. Common header

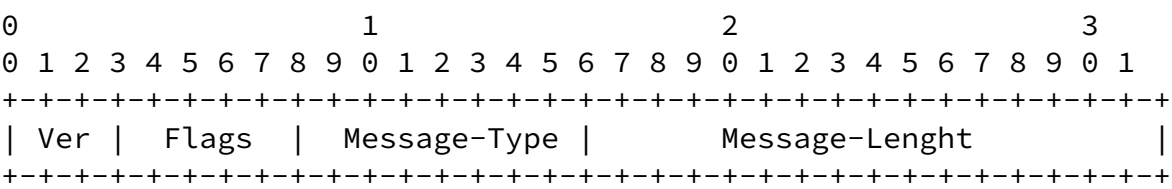



Figure 7: PCEP message common header

Ver (Version - 3 bits): PCEP version number. Current version is version 1.

Flags (5 bits): no flags are currently defined. Unassigned bits are considered as reserved and MUST be set to zero on transmission.

Message-Type (8 bits):





The following message types are currently defined (to be confirmed by IANA).

| Value | Meaning |
|---|---|
| 1 | Open |

| | |
|---|---|
| 2 | Keepalive |
| 3 | Path Computation Request |
| 4 | Path Computation Reply |
| 5 | Notification |
| 6 | Error |
| 7 | Close |

Message-Length (16 bits): total length of the PCEP message expressed in bytes including the common header.

## 6.2. Open message

The Open message is a PCEP message sent by a PCC to a PCE and a PCE to a PCC in order to establish a PCEP session. The Message-Type field of the PCEP common header for the Open message is set to 1 (To be confirmed by IANA).

Once the TCP connection has been successfully established, the first message sent by the PCC to the PCE or by the PCE to the PCC MUST be an Open message as specified in Section 10. Any message received prior to an Open message MUST trigger a protocol error condition and the PCEP session MUST be terminated. The Open message is used to establish a PCEP session between the PCEP peers. During the establishment phase the PCEP peers exchange several session characteristics. If both parties agree on such characteristics the PCEP session is successfully established.

Open message
<Open Message>::= <Common Header>
<OPEN>
The Open message MUST contain exactly one OPEN object (see Section 7.2). Various session characteristics are specified within the OPEN object. Once the TCP connection has been successfully established the sender MUST start an initialization timer called OpenWait after the expiration of which if no Open message has been received it sends a PCErr message and releases the TCP connection (see Section 10 for details).

Once an Open message has been sent to a PCEP peer, the sender MUST start an initialization timer called KeepWait after the expiration of which if neither a KeepAlive message has been received nor a PCErr message in case of disagreement of the session characteristics, a PCErr message MUST be sent and the TCP connection MUST be released (see Section 10 for details).

The KeepWait timer has a fixed value of 1 minute.

Upon the receipt of an Open message, the receiving PCEP peer MUST determine whether the suggested PCEP session characteristics are acceptable. If at least one of the characteristic(s) is not acceptable by the receiving peer, it MUST send an Error message. The Error message SHOULD also contain the related Open object: for each unacceptable session parameter, an acceptable parameter value SHOULD be proposed in the appropriate field of the Open object in place of the originally proposed value. The PCEP peer MAY decide to resend an Open message with different session characteristics. If a second Open message is received with the same set of parameters or with parameters that are still unacceptable, the receiving peer MUST send an Error message and it MUST immediately close the TCP connection. Details about error message can be found in Section 7.14.

If the PCEP session characteristics are acceptable, the receiving PCEP peer MUST consequently send a Keepalive message (defined in Section 6.3) that would serve as an acknowledgment.

The PCEP session is considered as established once both PCEP peers have received a Keepalive message from their peer.

## 6.3. Keepalive message

A Keepalive message is a PCEP message sent by a PCC or a PCE in order to keep the session in active state. The Message-Type field of the PCEP common header for the Keepalive message is set to 2 (To be confirmed by IANA). The Keepalive message does not contain any object.

PCEP has its own keepalive mechanism used to ensure of the liveness of the PCEP session. This requires the determination of the frequency at which each PCEP peer sends keepalive messages. Asymmetric values may be chosen; thus there is no constraint mandating the use of identical keepalive frequencies by both PCEP peers. The DeadTimer is defined as the period of time after the expiration of which a PCEP peer declares the session down if no PCEP message has been received (keepalive or any other PCEP message: thus, any PCEP message acts as a keepalive message). Similarly, there is no constraints mandating the use of identical DeadTimers by both PCEP peers. The minimum KeepAlive timer value is 1 second.

Keepalive messages are used to acknowledge an Open message if the receiving PCEP peer agrees on the session characteristics and to ensure the liveness of the PCEP session. Keepalive messages are sent at the frequency specified in the OPEN object carried within an Open

message. Because any PCEP message may serve as Keepalive an





implementation may either decide to send Keepalive messages at fixed intervals regardless on whether other PCEP messages might have been sent since the last sent Keepalive message or may decide to differ the sending of the next Keepalive message based on the time at which the last PCEP message (other than Keepalive) has been sent.

Note that sending Keepalive messages to maintain the session alive is optional and PCEP peers may decide to not send Keepalive messages once the PCEP session is established.

Keepalive message
<Keepalive Message>::= <Common Header>

## 6.4. Path Computation Request (PCReq) message

A Path Computation Request message (also referred to as a PCReq message) is a PCEP message sent by a PCC to a PCE so as to request a path computation. The Message-Type field of the PCEP common header for the PCReq message is set to 3 (To be confirmed by IANA).

There are two mandatory objects that MUST be included within a PCReq message: the RP and the END-POINTS objects (see section Section 7). If one of these objects is missing, the receiving PCE MUST send an error message to the requesting PCC. Other objects are optional.

```
   The format of a PCReq message is as follows:
   <PCReq Message>::= <Common Header>
                      [<SVEC-list>]
                      <request-list>

   where:
      <svec-list>::=<SVEC>[<svec-list>]
      <request-list>::=<request>[<request-list>]

      <request>::= <RP>
                   <END-POINTS>
                   [<LSPA>]
                   [<BANDWIDTH>]
                   [<BANDWIDTH>]
                   [<metric-list>]
                   [<RRO>]
                   [<IRO>]
                   [<LOAD-BALANCING>]
   where:

   <metric-list>::=<METRIC>[<metric-list>]
```

The SVEC, RP, END-POINTS, LSPA, BANDWIDTH, METRIC, RRO, IRO and LOAD-BALANCING objects are defined in Section 7. The special case of two BANDWIDTH objects is discussed in details in Section 7.6.

## 6.5. Path Computation Reply (PCRep) message

The PCEP Path Computation Reply message (also referred to as a PCRep message) is a PCEP message sent by a PCE to a requesting PCC in response to a previously received PCReq message. The Message-Type field of the PCEP common header is set to 4 (To be confirmed by

IANA).

The PCRep message MUST contain at least one RP object. For each reply that is bundled into a single PCReq message, an RP object MUST be included that contains a Request-ID-number identical to the one specified in the RP object carried in the corresponding PCReq message (see Section 7.3 for the definition of the RP object).

A PCRep message may contain a set of computed path(s) corresponding to either a single path computation request with load-balancing (see Section 7.15) or multiple path computation requests originated by a requesting PCC. The PCRep message may also contain multiple acceptable paths corresponding to the same request.

The bundling of multiple replies to a set of path computation





requests within a single PCRep message is supported by PCEP. If a PCE receives non-synchronized path computation requests by means of one or more PCReq messages from a requesting PCC it MAY decide to bundle the computed paths within a single PCRep message so as to reduce the control plane load. Note that the counter side of such an approach is the introduction of additional delays for some path computation requests of the set. Conversely, a PCE that receives multiple requests within the same PCReq message MAY decide to provide each computed path in separate PCRep messages or within the same PCRep message.

If the path computation request can be satisfied (the PCE finds a set of path(s) that satisfy the set of constraint(s)), the set of computed path(s) specified by means of ERO object(s) is inserted in the PCRep message. The ERO object is defined in Section 7.8. Such a situation where multiple computed paths are provided in a PCRep message is discussed in detail in Section 7.12. Furthermore, when a PCC requests the computation a set of paths for a total amount of bandwidth of X by means of a LOAD-BALANCING object carried within a PCReq message, the ERO of each computed path may be followed by a BANDWIDTH object as discussed in section Section 7.15.

If the path computation request cannot be satisfied, the PCRep message MUST include a NO-PATH object. The NO-PATH object (described in Section 7.4) may also comprise other information (e.g reasons for the path computation failure).

The format of a PCRep message is as follows:

```
<PCRep Message> ::= <Common Header>
                    <response-list>
```

where:

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


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

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

<path>::= <ERO><attribute-list>
```

where:

```
<attribute-list>::=[<LSPA>]
                   [<BANDWIDTH>]
                   [<metric-list>]
                   [<IRO>]

<metric-list>::=<METRIC>[<metric-list>]
```

## 6.6. Notification (PCNtf) message

The PCEP Notification message (also referred to as the PCNtf message) can either be sent by a PCE to a PCC or by a PCC to a PCE so as to notify of a specific event. The Message-Type field of the PCEP common header is set to 5 (To be confirmed by IANA).

The PCNtf message MUST carry at least one NOTIFICATION object and may contain several NOTIFICATION objects should the PCE or the PCC intend to notify of multiple events. The NOTIFICATION object is defined in Section 7.13. The PCNtf message MAY also contain an RP object (see Section 7.3 when the notification refers to a particular path computation request.

The PCNtf message may be sent by a PCC or a PCE in response to a request or in an unsolicited manner.





The format of a PCNtf message is as follows:

```
<PCNtf Message>::=<Common Header>
                  <notify-list>

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

<notify>::= [<request-id-list>]
             <notification-list>

<request-id-list>:==<RP><request-id-list>

<notification-list>:=<NOTIFICATION><notification-list>
```

## 6.7. Error (PCErr) Message

The PCEP Error message (also referred to as a PCErr message) is sent when a protocol error condition is met. The Message-Type field of the PCEP common header is set to 6 (To be confirmed by IANA).

The PCErr message is either sent by a PCC or a PCE in response to a request or in an unsolicited manner. In the former case, the PCErr message MUST include the set of RP objects related to the pending path computation request(s) that triggered the protocol error condition. In the later case (unsolicited), no RP object is inserted in the PCErr message. No RP object is inserted in a PCErr when the error condition occurred during the initialization phase. A PCErr message MUST contain a PCEP-ERROR object specifying the PCEP error condition. The PCEP-ERROR object is defined in section Section 7.14.

The format of a PCErr message is as follows:

```
<PCErr Message> ::= <Common Header>
                    <error-list>
                    [<Open>]

<error-list>:==<error>[<error-list>]
<error>::=[<request-id-list>]
           <error-obj-list>

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

<error-obj-list>:==<PCEP-ERROR>[<error-obj-list>]
```

The procedure upon the reception of a PCErr message is defined in Section 7.14.

## 6.8. Close message

The Close message is a PCEP message that is either sent by a PCC to a PCE or by a PCE to a PCC in order to close an established PCEP





session. The Message-Type field of the PCEP common header for the Close message is set to 7 (To be confirmed by IANA).

Close message

```
<Close Message>::= <Common Header>
```

<CLOSE>

The Close message MUST contain exactly one CLOSE object (see Section 6.8).

Upon the receipt of a Close message, the receiving PCEP peer MUST cancel all pending requests and MUST close the TCP connection.

# 7. Object Formats

## 7.1. Common object header

A PCEP object carried within a PCEP message consists of one or more 32-bit words with a common header which has the following format:

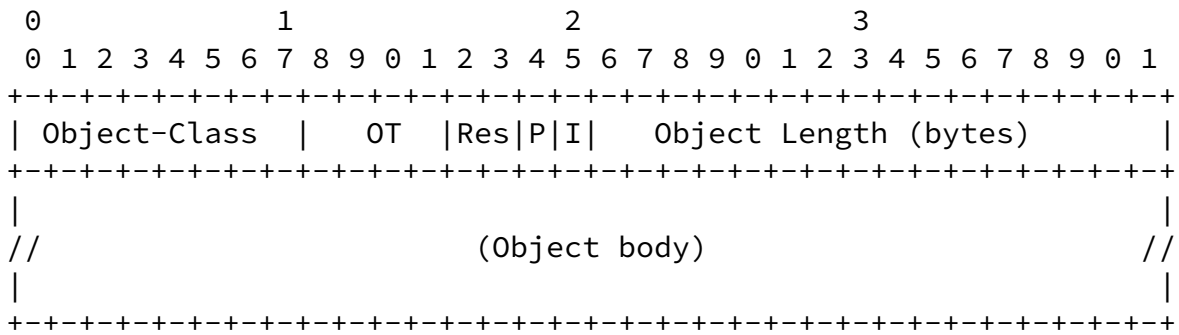



Figure 8: PCEP common object header

Object-Class (8 bits): identifies the PCEP object class.

OT (Object-Type - 4 bits): identifies the PCEP object type.

The Object-Class and Object-Type fields are managed by IANA.

The Object-Class and Object-Type fields uniquely identify each PCEP object.

Res flags (2 bits). Reserved field. This field MUST be set to zero on transmission and MUST be ignored on receipt.

- P flag (Processing-Rule - 1-bit): the P flag allows a PCC to specify in a PCReq message sent to a PCE whether the object must be taken into account by the PCE during path computation or is just optional. When the P flag is set, the object MUST be taken into account by the PCE. Conversely, when the P flag is cleared,

the object is optional and the PCE is free to ignore it if not supported.

- I flag (Ignore - 1 bit): the I flag is used by a PCE in a PCRep message to indicate to a PCC whether or not an optional object was processed. The PCE MAY include the ignored optional object in its reply and set the I flag to indicate that the optional object was ignored during path computation. When the I flag is cleared, the PCE indicates that the optional object was processed during the path computation. The setting of the I flag for optional objects is purely indicative and optional. The I flag has no meaning in a PCRep message when the P flag had been set in the corresponding PCRep message.

If the PCE does not understand an object with the P flag set or understands the object but decides to ignore the object, the entire PCEP message MUST be rejected and the PCE MUST send a PCErr message with Error-Type="Unknown Object" or "Not supported Object".

Object Length (16 bits). Specifies the total object length including the header, in bytes. The Object Length field MUST always be a multiple of 4, and at least 4. The maximum object content length is 65528 bytes.

## 7.2. OPEN object

The OPEN object MUST be present in each Open message and MAY be present in a PCErr message. There MUST be only one OPEN object per Open or PCErr message.

The OPEN object contains a set of fields used to specify the PCEP version, Keepalive frequency, DeadTimer, PCEP session ID along with various flags. The OPEN object may also contain a set of TLVs used to convey various session characteristics such as the detailed PCE capabilities, policy rules and so on. No TLVs are currently defined.

OPEN Object-Class is to be assigned by IANA (recommended value=1)

OPEN Object-Type is to be assigned by IANA (recommended value=1)

The format of the OPEN object body is as follows:

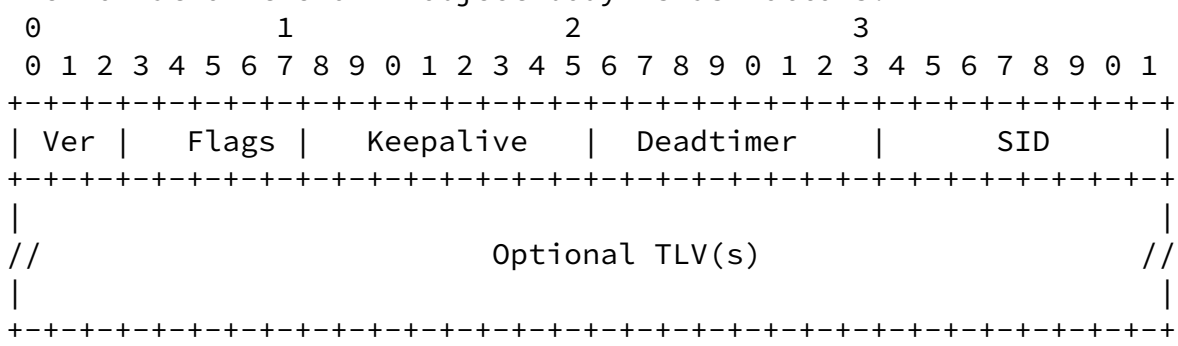



Figure 9: OPEN Object format

Ver (3 bits): PCEP version. Current version is 1.

Flags (5 bits): No Flags are currently defined. Unassigned bits are considered as reserved and MUST be set to zero on transmission.

Keepalive (8 bits): maximum period of time (in seconds) between the sending of PCEP messages. The minimum value for the Keepalive is 1 second. When set to 0, once the session is established, no further Keepalive messages need to be sent to the remote peer. A RECOMMENDED value for the keepalive frequency is 30 seconds.

DeadTimer (8 bits): specifies the amount of time after the expiration of which the PCEP peer can declare the session with the sender of the Open message down if no PCEP message has been received. The DeadTimer MUST be set to 0 if the Keepalive is set to 0. A RECOMMENDED value for the DeadTimer is 4 times the value of the Keepalive.

Example

A sends an Open message to B with Keepalive=10 seconds and Deadtimer=30 seconds. This means that A sends Keepalive messages (or ay other PCEP message) to B every 30 seconds and B can declare the PCEP session with A down if no PCEP has been received from A.

SID (PCEP session-ID - 8 bits): unsigned PCEP session number that identifies the current session. The SID MUST be incremented each time a new PCEP session is established and is mainly used for logging and troubleshooting purposes.

Optional TLVs may be included within the OPEN object body to specify PCC or PCE characteristics. The specification of such TLVs is outside the scope of this document.

When present in an Open message, the OPEN object specifies the proposed PCEP session characteristics. Upon receiving unacceptable





PCEP session characteristics during the PCEP session initialization phase, the receiving PCEP peer (PCE) MAY include an OPEN object within the PCErr message so as to propose alternative acceptable session characteristic values.

## 7.3. RP Object

The RP (Request Parameters) object MUST be carried within each PCReq and PCRep messages and MAY be carried within PCNtf and PCErr messages. The P flag of the RP object MUST be set in PCReq and PCReq messages and MUST be cleared in PCNtf and PCErr messages. The RP object is used to specify various characteristics of the path computation request.

### 7.3.1. Object definition

RP Object-Class is to be assigned by IANA (recommended value=2)

RP Object-Type is to be assigned by IANA (recommended value=1)

The format of the RP object body is as follows:

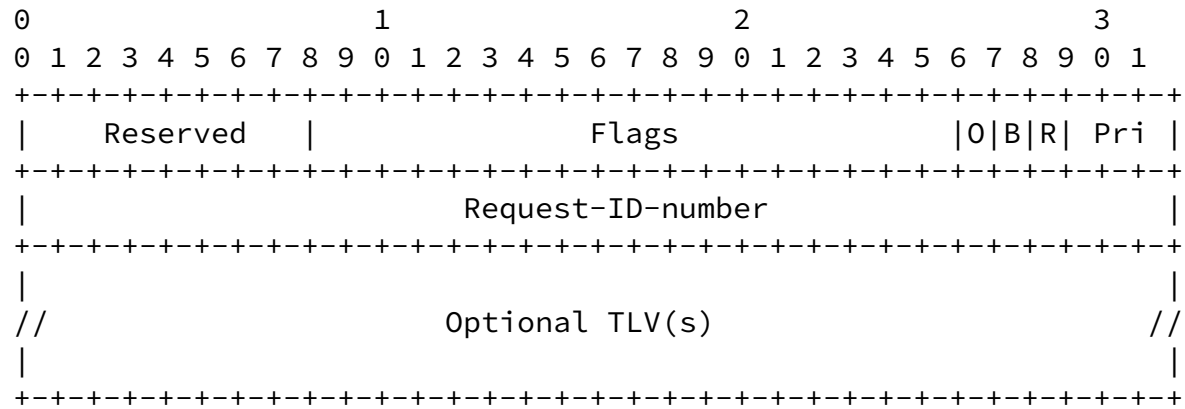



Figure 10: RP object body format

The RP object body has a variable length and may contain additional TLVs. No TLVs are currently defined.

Reserved (8 bits): Reserved: This field MUST be set to zero on transmission and MUST be ignored on receipt.

Flags (24 bits)

The following flags are currently defined:

- Pri (Priority - 3 bits): the Priority field may be used by the requesting PCC to specify to the PCE the request's priority from 1 to 7. The decision of which priority should be used for a





specific request is of a local matter and MUST be set to 0 when unused. Furthermore, the use of the path computation request priority by the PCE's scheduler is implementation specific and out of the scope of this document. Note that it is not required for a PCE to support the priority field: in this case, it is RECOMMENDED to set the priority field to 0 by the PCC in the RP object. If the PCE does not take into account the request priority, it is RECOMMENDED to set the priority field to 0 in the RP object carried within the corresponding PCRep message, regardless of the priority value contained in the RP object carried within the corresponding PCReq message. A higher numerical value of the priority field reflects a higher priority. Note that it is the responsibility of the network administrator to make use of the priority values in a consistent manner across the various PCC(s). The ability of a PCE to support requests prioritization may be dynamically discovered by the PCC(s) by means of PCE capability discovery. If not advertised by the PCE, a PCC may decide to set the request priority and will learn the ability of the PCE to support request prioritization by observing the Priority field of the RP object received in the PCRep message. If the value of the Pri field is set to 0, this means that the PCE does not support the handling of request priorities: in other words, the path computation request has been honoured but without taking the request priority into account.

- R (Reoptimization - 1 bit): when set, the requesting PCC specifies that the PCReq message relates to the reoptimization of an existing TE LSP in which case, in addition to the TE LSP

attributes, the current path of the existing TE LSP to be reoptimized MUST be provided in the PCReq (except for 0-bandwidth TE LSP) message by means of an RRO object defined in Section 7.9.

- B (Bi-directional - 1 bit): when set, the PCC specifies that the path computation request relates to a bidirectional TE LSP that has the same traffic engineering requirements including fate sharing, protection and restoration, LSRs, and resource requirements (e.g. latency and jitter) in each direction. When cleared, the TE LSP is unidirectional.

- O (strict/lOose - 1 bit): when set, in a PCReq message, this indicates that a loose path is acceptable. Otherwise, when cleared, this indicates to the PCE that a path exclusively made of strict hops is required. In a PCRep message, when the O bit is set this indicates that the returned path is a loose path, otherwise (the O bit is cleared), the returned path is made of strict hops.

Unassigned bits are considered as reserved and MUST be set to zero on





transmission.

Request-ID-number (32 bits). The Request-ID-number value combined with the source IP address of the PCC and the PCE address uniquely identify the path computation request context. The Request-ID-number MUST be incremented each time a new request is sent to the PCE. The value 0x0000000 is considered as invalid. If no path computation reply is received from the PCE, and the PCC wishes to resend its request, the same Request-ID-number MUST be used. Conversely, different Request-ID-number MUST be used for different requests sent to a PCE. The same Request-ID-number may be used for path computation requests sent to different PCEs. The path computation reply is unambiguously identified by the IP source address of the replying PCE.

### 7.3.2. Handling of the RP object

If a PCReq message is received without containing an RP object, the PCE MUST send a PCErr message to the requesting PCC with Error-type="Required Object missing" and Error-value="RP Object missing".

If the O bit of the RP message carried within a PCReq message is cleared and local policy has been configured on the PCE to not provide explicit path(s) (for instance, for confidentiality reasons), a PCErr message MUST be sent by the PCE to the requesting PCC and the pending path computation request MUST be discarded. The Error-type is "Policy Violation" and Error-value is "O bit cleared".

R bit: when the R bit of the RP object is set in a PCReq message, this indicates that the path computation request relates to the reoptimization of an existing TE LSP. In this case, the PCC MUST also provide the strict/loose path by including an RRO object in the PCReq message so as to avoid/limit double bandwidth counting if and only if the TE LSP is a non 0-bandwidth TE LSP. If the PCC has not requested a strict path (O bit set), a reoptimization can still be requested by the PCC but this implies for the PCE to be either stateful (keep track of the previously computed path with the associated list of strict hops) or to have the ability to retrieve the complete required path segment. Alternatively the PCC MUST be able to inform PCE of the working path with associated list of strict hops in PCReq. The absence of an RRO in the PCReq message for a non 0-bandwidth TE LSP when the R bit of the RP object is set MUST trigger the sending of a PCErr message with Error-type="Required Object Missing" and Error-value="RRO Object missing for reoptimization".

If the PCC receives a PCRep message that contains a RP object referring to an unknown Request-ID-Number, the PCC MUST send a PCErr





message with Error-Type="Unknown request reference".

## 7.4. NO-PATH Object

The NO-PATH object is used in PCRep messages in response to an unsuccessful path computation request (the PCE could not find a path satisfying the set of constraints). When a PCE cannot find a path satisfying a set of constraints, it MUST include a NO-PATH object in the PCRep message. The NO-PATH object is used to report the impossibility to find a path that satisfies the set of constraints.

There are potentially several categories of issues that can lead to a negative reply. For example, the PCE chain might be broken (should there be more than one PCE involved in the path computation) or no

path obeying the set constraints could be found. The "NI (Nature of Issue)" field in the NO-PATH object is used to report the error category.

Optionally, if the PCE supports such capability, the NO-PATH object MAY contain an optional NO-PATH-VECTOR TLV defined below used to provide more information on the reasons that led to a negative reply and the PCRep message MAY also contain a list of objects that specify the set of constraints that could not be satisfied. The PCE MAY just replicate the set of object(s) that was received that was the cause of the unsuccessful computation or MAY optionally report a suggested value for which a path could have been found.

NO-PATH Object-Class is to be assigned by IANA (recommended value=3)

NO-PATH Object-Type is to be assigned by IANA (recommended value=1)

The format of the NO-PATH object body is as follows:

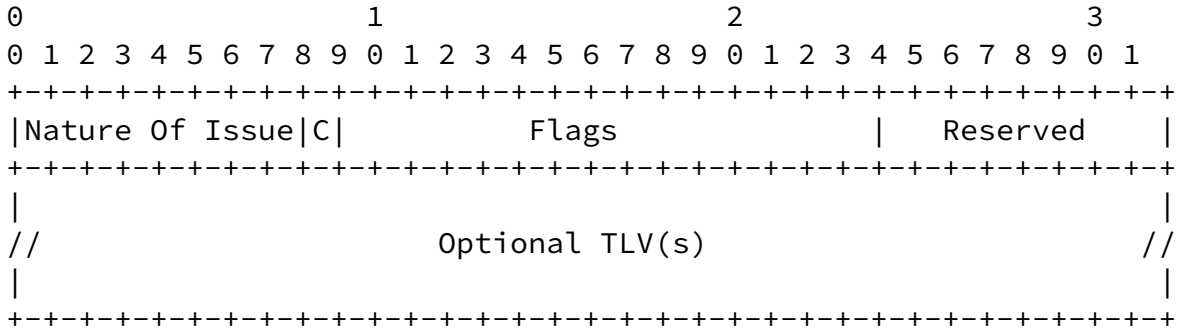



Figure 11: NO-PATH object format

NI - Nature Of Issue (8 bits): the NI field is used to report the nature of the issue that led to a negative reply. Two values are currently defined:





0x00: No path satisfying the set of constraints could be found

0x01: PCE chain broken

Flags (16 bits).

The following flag is currently defined:

C flag (1 bit): when set, the PCE indicates the set of unsatisfied constraints (reasons why a path could not be found) in the PCRep message by including the relevant PCEP objects. When cleared, no reason is specified. When the C bit is set, the NI field value MUST be 0x00.

Reserved (8 bits): This field MUST be set to zero on transmission and MUST be ignored on receipt.

The NO-PATH object body has a variable length and may contain additional TLVs. The only TLV currently defined is the NO-PATH-VECTOR TLV defined below.

Example: consider the case of a PCC that sends a path computation request to a PCE for a TE LSP of X MBits/s. Suppose that PCE cannot find a path for X MBits/s. In this case, the PCE must include in the PCRep message a NO-PATH object. Optionally the PCE may also include the original BANDWIDTH object so as to indicate that the reason for the unsuccessful computation is the bandwidth constraint (in this case, the NI field value is 0x00 and C flag is set). If the PCE supports such capability it may alternatively include the BANDWIDTH Object and report a value of Y in the bandwidth field of the BANDWIDTH object (in this case, the C flag is set) where Y refers to the bandwidth for which a TE LSP with the same other characteristics could have been computed.

When the NO-PATH object is absent from a PCRep message, the path computation request has been fully satisfied and the corresponding path(s) is/are provided in the PCRep message.

An optional TLV named NO-PATH-VECTOR MAY be included in the NO-PATH object in order to provide more information on the reasons that led to a negative reply.

The NO-PATH-VECTOR TLV is composed of 1 byte for the type, 1 byte specifying the number of bytes in the value field, followed by a fix length value field of 32-bits flags field used to report the reason(s) that led to unsuccessful path computation.

The NO-PATH-VECTOR TLV is padded to eight-byte alignment.

TYPE: To be assigned by IANA

LENGTH: 4

VALUE: 32-bits flags field

IANA is requested to manage the space of flags carried in the NO-PATH-VECTOR TLV (see Section 9).

The following flags are currently defined:

- 0x01: PCE currently unavailable
- 0x02: Unknown destination
- 0x03: Unknown source

### 7.5. END-POINT Object

The END-POINTS object is used in a PCReq message to specify the source IP address and the destination IP address of the path for which a path computation is requested. Note that the source and destination addresses specified in the END-POINTS object may or may not correspond to the source and destination IP address of the TE LSP but rather to a path segment. Two END-POINTS objects (for IPv4 and IPv6) are defined.

END-POINTS Object-Class is to be assigned by IANA (recommended value=4)

END-POINTS Object-Type is to be assigned by IANA (recommended value=1 for IPv4 and 2 for IPv6)

The format of the END-POINTS object body for IPv4 (Object-Type=1) is as follows:

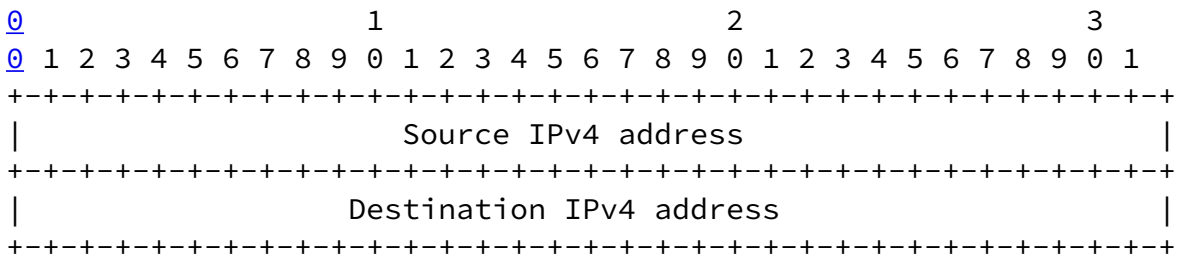



Figure 12: END-POINTS object body format for IPv4

The format of the END-POINTS object for IPv6 (Object-Type=2) is as follows:

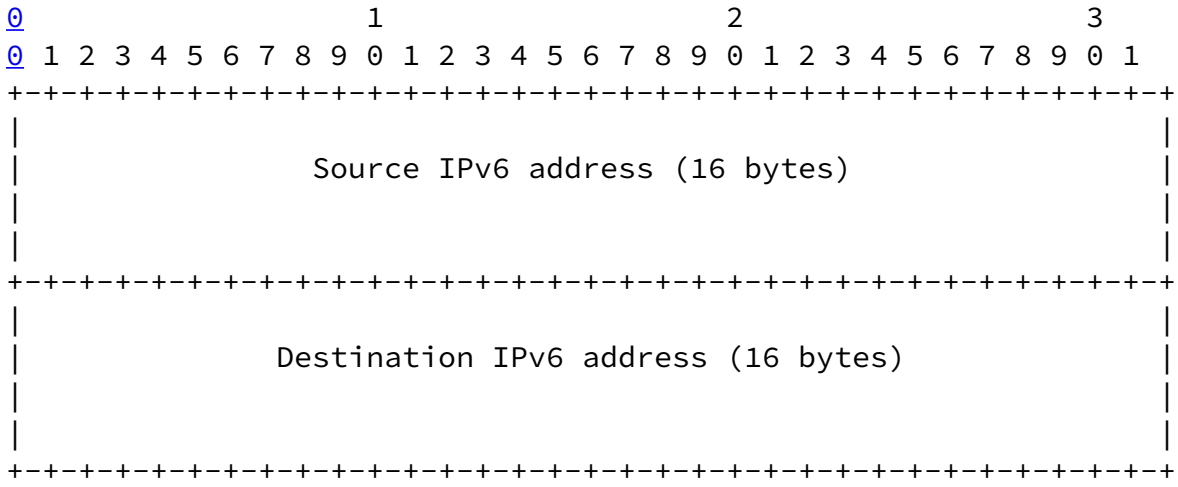



Figure 13: END-POINTS object body format for IPv6

The END-POINTS object body has a fixed length of 8 bytes for IPv4 and 32 bytes for IPv6.

## 7.6. BANDWIDTH Object

The BANDWIDTH object is used to specify the requested bandwidth for a TE LSP.

If the requested bandwidth is equal to 0, the BANDWIDTH object is optional. Conversely, if the requested bandwidth is non equal to 0, the PCReq message MUST contain a BANDWIDTH object.

In the case of the reoptimization of a TE LSP, the bandwidth of the existing TE LSP MUST also be included in addition to the requested bandwidth if and only if the two values differ. Consequently, two Object-Type are defined that refer to the requested bandwidth and the bandwidth of the TE LSP for which a reoptimization is being





performed.

The BANDWIDTH object may be carried within PCReq and PCRep messages.

BANDWIDTH Object-Class is to be assigned by IANA (recommended value=5)

Two Object-Type are defined for the BANDWIDTH object:

- Requested bandwidth: BANDWIDTH Object-Type is to be assigned by IANA (recommended value=1)
- Bandwidth of an existing TE LSP for which a reoptimization is requested. BANDWIDTH Object-Type is to be assigned by IANA (recommended value=2)

The format of the BANDWIDTH object body is as follows:

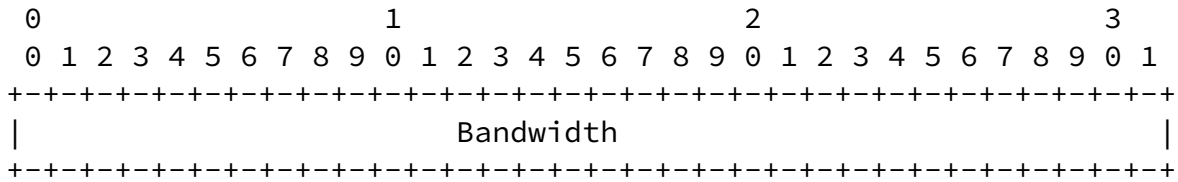



Figure 14: BANDWIDTH object body format

Bandwidth: 32 bits. The requested bandwidth is encoded in 32 bits in IEEE floating point format, expressed in bytes per second.

The BANDWIDTH object body has a fixed length of 4 bytes.

## 7.7. METRIC Object

The METRIC object is optional and can be used for several purposes.

In a PCReq message, a PCC MAY insert a METRIC object:

- To indicate the metric that MUST be optimized by the path

computation algorithm (IGP metric, TE metric, Hop counts). Currently, three metrics are defined: the IGP cost, the TE metric (see [RFC3785]) and the number of hops traversed by a TE LSP.

- To indicate a bound on the path cost that MUST NOT be exceeded for the path to be considered as acceptable by the PCC.

In a PCRep message, the METRIC object MAY be inserted so as to provide the cost for the computed path. It MAY also be inserted within a PCRep with the NO-PATH object to indicate that the metric constraint could not be satisfied.





The path computation algorithmic aspects used by the PCE to optimize a path with respect to a specific metric are outside the scope of this document.

It must be understood that such path metric is only meaningful if used consistently: for instance, if the delay of a path computation segment is exchanged between two PCEs residing in different domains, consistent ways of defining the delay must be used.

The absence of the METRIC object MUST be interpreted by the PCE as a path computation request for which the PCE may choose the metric to be used.

METRIC Object-Class is to be assigned by IANA (recommended value=6)

METRIC Object-Type is to be assigned by IANA (recommended value=1)

The format of the METRIC object body is as follows:

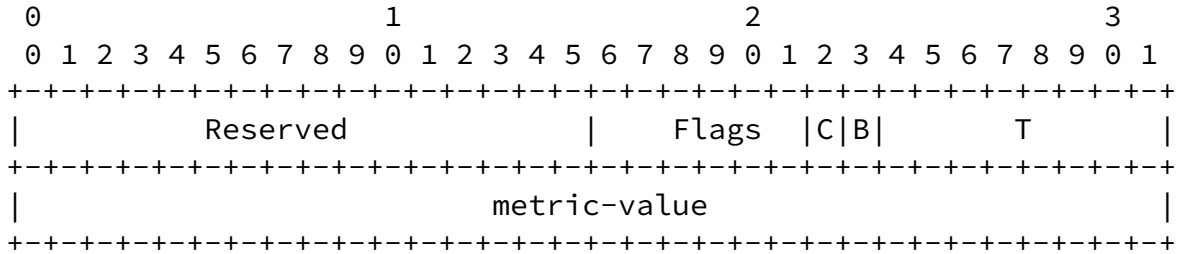



Figure 15: METRIC object body format

The METRIC object body has a fixed length of 8 bytes.

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

T (Type - 8 bits): Specifies the metric type.

Three values are currently defined:

- T=1: IGP metric
- T=2: TE metric
- T=3: Hop Counts

Flags (8 bits): Two flags are currently defined:

- B (Bound - 1 bit): When set in a PCReq message, the metric-value indicates a bound (a maximum) for the path cost that must not be exceeded for the PCC to consider the computed path as acceptable.





When the B flag is cleared, the metric-value field is not used to reflect a bound constraint.

- C (Cost - 1 bit): When set in a PCReq message, this indicates that the PCE MUST provide the computed path cost (should a path satisfying the constraints be found) in the PCRep message for the corresponding metric.

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

Metric-value (32 bits): metric value encoded in 32 bits in IEEE floating point format.

Multiple METRIC Objects MAY be inserted in a PCRep or the PCReq message. There MUST be at most one instance of the METRIC object for each metric type with the same B flag value. If two or more instances of a METRIC object with the same B flag value are present for a metric type, only the first instance MUST be considered and other instances MUST be ignored.

In a PCReq message the presence of multiple METRIC objects can be

used to specify a multi-parameters (e.g. a metric may be a constraint or a parameter to minimize/maximize) objective function or multiple bounds for different constraints where at most one METRIC object must be used to indicate the metric to optimize (B-flag is cleared): the other METRIC object MUST be used to reflect bound constraints (B-Flag is set).

A METRIC object used to indicate the metric to optimize during the path computation MUST have the B-Flag cleared and the T-Flag set to the appropriate value. When the path computation relates to the reoptimization of an exiting TE LSP (in which case R-Flag of the RP object is set) an implementation MAY decide to set the metric-value field to the cost of the TE LSP to be reoptimized with regards to a specific metric type.

A METRIC object used to reflect a bound MUST have the B-Flag set, the T-Flag and metric-value field set to the appropriate values.

In a PCRep message, unless not allowed by PCE policy, at least one METRIC object MUST be present that reports the computed path cost in particular if the C bit of the METRIC object was set in the corresponding path computation request (the B-flag MUST be cleared); optionally the PCRep message MAY contain additional METRIC objects that correspond to bound constraints, in which case the metric-value MUST be equal to the corresponding path metric cost (the B-flag MUST be set). If no path satisfying the constraints could be found by the





PCE, the METRIC objects MAY also be present in the PCRep message with the NO-PATH object to indicate the constraint metric that could be satisfied.

Example: if a PCC sends a path computation request to a PCE where the metric to optimize is the IGP metric and the TE metric must not exceed the value of M, two METRIC object are inserted in the PCReq message:

- First METRIC Object with B=0, T=1, C=1, metric-value=0x0000
- Second METRIC Object with B=1, T=2, metric-value=M

If a path satisfying the set of constraints can be found by the PCE and no policy preventing to provide the path cost is in place, the

PCE inserts one METRIC object with B=0, T=1, metric-value= computed IGP path cost. Additionally, the PCE may insert a second METRIC object with B=1, T=2, metric-value= computed TE path cost.

## 7.8. Explicit Route Object

The ERO is used to encode a TE LSP. The ERO is carried within a PCRep message to provide the computed TE LSP should have the path computation been successful.

The contents of this object are identical in encoding to the contents of the Explicit Route Object defined in [RFC3209], [RFC3473] and [RFC3477]. That is, the object is constructed from a series of sub-objects. Any RSVP ERO sub-object already defined or that could be defined in the future for use in the ERO is acceptable in this object.

PCEP ERO sub-object types correspond to RSVP ERO sub-object types.

Since the explicit path is available for immediate signaling by the MPLS or GMPLS control plane, the meanings of all of the sub-objects and fields in this object are identical to those defined for the ERO.

ERO Object-Class is to be assigned by IANA (recommended value=7)

ERO Object-Type is to be assigned by IANA (recommended value=1)

## 7.9. Route Record Object

The RRO is used to record the route followed by a TE LSP. The PCEP RRO is exclusively carried within a PCReq message so as to specify the route followed by a TE LSP for which a reoptimization is desired.





The contents of this object are identical in encoding to the contents of the Route Record Object defined in [RFC3209], [RFC3473] and [RFC3477]. That is, the object is constructed from a series of sub-objects. Any RSVP RRO sub-object already defined or that could be defined in the future for use in the RRO is acceptable in this object.

The meanings of all of the sub-objects and fields in this object are

identical to those defined for the RRO.

PCEP RRO sub-object types correspond to RSVP RRO sub-object types.

RRO Object-Class is to be assigned by IANA (recommended value=8)

RRO Object-Type is to be assigned by IANA (recommended value=1)

## 7.10. LSPA Object

The LSPA object is optional and specifies various TE LSP attributes to be taken into account by the PCE during path computation. The LSPA (LSP Attributes) object can either be carried within a PCReq message or a PCRep message in case of unsuccessful path computation (in this case, the PCRep message also contains a NO-PATH object and the LSPA object is used to indicate the set of constraint(s) that could not be satisfied). Most of the fields of the LSPA object are identical to the fields of the SESSION-ATTRIBUTE object defined in [RFC3209] and [RFC4090]. When absent from the PCReq message, this means that the Setup and Holding priorities are equal to 0, and there are no affinity constraints.

LSPA Object-Class is to be assigned by IANA (recommended value=9)

LSPA Object-Types is to be assigned by IANA (recommended value=1)





The format of the LSPA object body is:

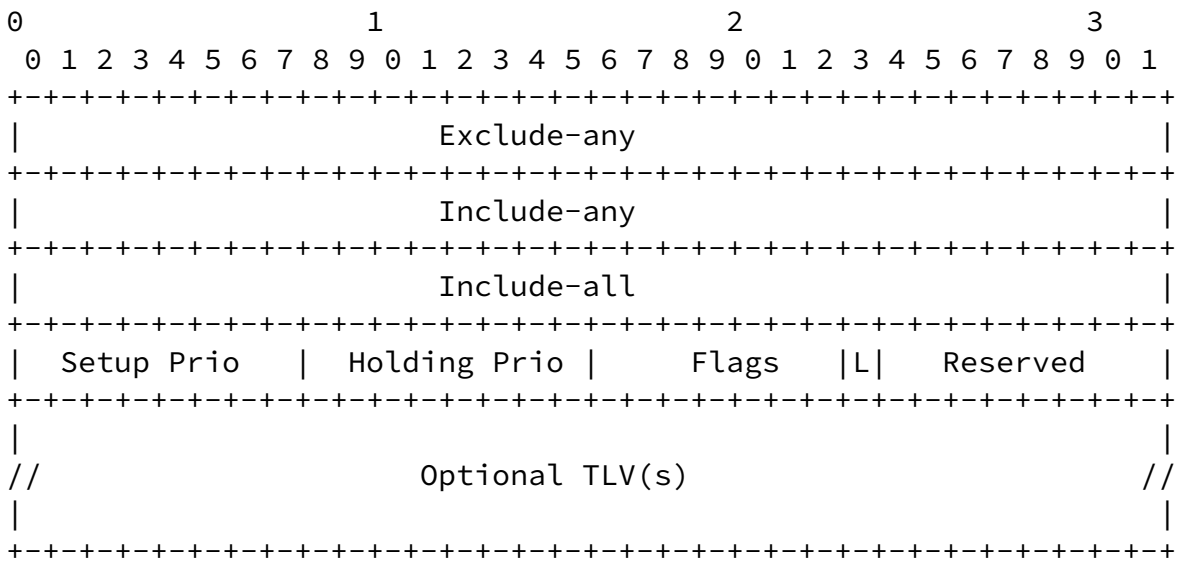



Figure 16: LSPA object body format

Setup Prio (Setup Priority - 8 bits). The priority of the session with respect to taking resources, in the range of 0 to 7. The value 0 is the highest priority. The Setup Priority is used in deciding whether this session can preempt another session.

Holding Prio (Holding Priority - 8 bits). The priority of the session with respect to holding resources, in the range of 0 to 7. The value 0 is the highest priority. Holding Priority is used in deciding whether this session can be preempted by another session.

Flags (8 bits)

The flag L corresponds to the "Local protection desired" bit ([RFC3209]) of the SESSION-ATTRIBUTE Object.

L Flag (Local protection desired). When set, this means that the computed path must include links protected with Fast Reroute as defined in [RFC4090].

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

Reserved (8 bits): This field MUST be set to zero on transmission and MUST be ignored on receipt.

Note that the Optional TLV field may contain additional TE LSP attributes as defined in [RFC4420].

### 7.11. Include Route Object Object

The IRO (Include Route Object) is optional and can be used to specify that the computed path MUST traverse a set of specified network elements. The IRO MAY be carried within PCReq and PCRep messages. When carried within a PCRep message with the NO-PATH object, the IRO indicates the set of elements that fail the PCE to find a path.

IRO Object-Class is to be assigned by IANA (recommended value=10)

IRO Object-Type is to be assigned by IANA (recommended value=1)

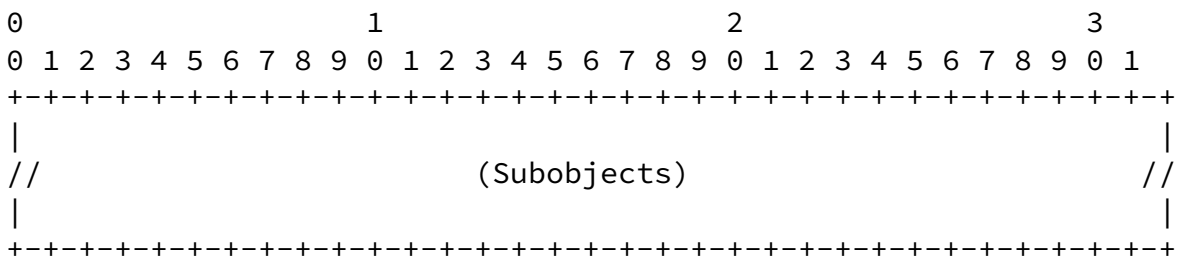



Figure 17: IRO body format

Subobjects The IRO is made of sub-object(s) identical to the ones defined in [RFC3209], [RFC3473] and [RFC3477] for use in EROs.

The following subobject types are supported.

| Type | Subobject |
|---|---|
| 1 | IPv4 prefix |
| 2 | IPv6 prefix |
| 4 | Unnumbered Interface ID |
| 32 | Autonomous system number |

The L bit of such sub-object has no meaning within an IRO.

### 7.12. SVEC Object

#### 7.12.1. Notion of Dependent and Synchronized path computation requests

Independent versus dependent path computation requests: path computation requests are said to be independent if they are not related to each other. Conversely a set of dependent path computation requests is such that their computations cannot be performed independently of each other (a typical example of dependent requests is the computation of a set of diverse paths).

Synchronized versus non-synchronized path computation requests: a set of path computation requests is said to be non-synchronized if their

respective treatment (path computations) can be performed by a PCE in a serialized and independent fashion.





There are various circumstances where the synchronization of a set of path computations may be beneficial or required.

Consider the case of a set of N TE LSPs for which a PCC needs to send path computation requests to a PCE. The first solution consists of sending N separate PCReq messages to the selected PCE. In this case, the path computation requests are non synchronized. Note that the PCC may chose to distribute the set of N requests across K PCEs for load balancing purpose. Considering that M (with M<N) requests are sent to a particular PCEi, as described above, such M requests can be sent in the form of successive PCReq messages destined to PCEi or bundled within a single PCReq message (since PCEP allows for the bundling of multiple path computation requests within a single PCReq message). That said, even in the case of independent requests, it can be desirable to request from the PCE the computation of their paths in a synchronized fashion that is likely to lead to more optimal path computations and/or reduced blocking probability if the PCE is a stateless PCE. In other words, the PCE should not compute the corresponding paths in a serialized and independent manner but it should rather "simultaneously" compute their paths. For example, trying to "simultaneously" compute the paths of M TE LSPs may allow the PCE to improve the likelihood to meet multiple constraints. Consider the case of two TE LSPs requesting N1 MBits/s and N2 MBits/s respectively and a maximum tolerable end-to-end delay for each TE LSP of X ms. There may be circumstances where the computation of the first TE LSP irrespectively of the second TE LSP may lead to the impossibility to meet the delay constraint for the second TE LSP. A second example is related to the bandwidth constraint. It is quite straightforward to provide examples where a serialized independent path computation approach would lead to the impossibility to satisfy both requests (due to bandwidth fragmentation) while a synchronized path computation would successfully satisfy both requests. A last example relates to the ability to avoid the allocation of the same resource to multiple requests thus helping to reduce the call set up failure probability compared to the serialized computation of independent requests.

Dependent path computation are usually synchronized. For example, in the case of the computation of M diverse paths, if such paths are

computed in a non-synchronized fashion this seriously increases the probability of not being able to satisfy all requests (sometimes also referred to as the well-know "trapping problem"). Furthermore, this would not allow a PCE to implement objective functions such as trying to minimize the sum of the TE LSP costs. In such a case, the path computation requests must be synchronized: they cannot be computed independently of each other. Conversely a set of independent path computation requests may or may not be synchronized.





The synchronization of a set of path computation requests is achieved by using the SVEC object that specifies the list of synchronized requests that can either be dependent or independent.

PCEP supports the following three modes:

- Bundle of a set of independent and non-synchronized path computation requests,
- Bundle of a set of independent and synchronized path computation requests (SVEC object defined below required),
- Bundle of a set of dependent and synchronized path computation requests (SVEC object defined below required).

### 7.12.2. SVEC Object

Section 7.12.1 details the circumstances under which it may be desirable and/or required to synchronize a set of path computation requests. The SVEC (Synchronization VECtor) object allows a PCC to request the synchronization of a set of dependent or independent path computation requests. The SVEC object is optional and may be carried within a PCReq message.

The aim of the SVEC object carried within a PCReq message is to request the synchronization of M path computation requests. The SVEC object is a variable length object that lists the set of M path computation requests that must be synchronized. Each path computation request is uniquely identified by the Request-ID-number carried within the respective RP object. The SVEC object also contains a set of flags that specify the synchronization type.

SVEC Object-Class is to be assigned by IANA (recommended value=11)

SVEC Object-Type is to be assigned by IANA (recommended value=1)

One Object-Type is defined for this object to be assigned by IANA with a recommended value of 1.





The format of the SVEC object body is as follows:

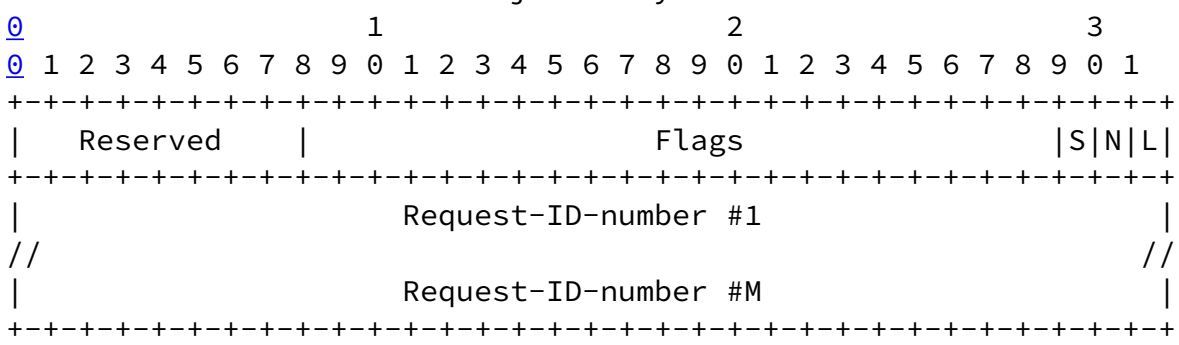



Figure 18: SVEC body object format

Reserved (8 bits): This field MUST be set to zero on transmission and MUST be ignored on receipt.

Flags (24 bits): Defines the potential dependency between the set of path computation requests.

- L (Link diverse) bit: when set, this indicates that the computed paths corresponding to the requests specified by the following RP objects MUST NOT have any link in common.

- N (Node diverse) bit: when set, this indicates that the computed paths corresponding to the requests specified by the following RP

objects MUST NOT have any node in common.

- o S (SRLG diverse) bit: when set, this indicates that the computed paths corresponding to the requests specified by the following RP objects MUST NOT share any SRLG (Shared Risk Link Group).

In case of a set of M synchronized independent path computation requests, the bits L, N and S are cleared.

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

The flags defined above are not exclusive.

### 7.12.3. Handling of the SVEC Object

The SVEC object allows a PCC to specify a list of M path computation requests that MUST be synchronized along with a potential dependency. The set of M path computation requests may be sent within a single PCReq message or multiple PCReq message. In the later case, it is RECOMMENDED for the PCE to implement a local timer activated upon the receipt of the first PCReq message that contains the SVEC object after the expiration of which, if all the M path computation requests





have not been received, a protocol error is triggered (this timer is called the SyncTimer). In this case the PCE MUST cancel the whole set of path computation requests and MUST send a PCErr message with Error-Type="Synchronized path computation request missing".

Note that such PCReq message may also contain non-synchronized path computation requests. For example, the PCReq message may comprise N synchronized path computation requests related to RP 1, ... , RP N listed in the SVEC object along with any other path computation requests.

## 7.13. NOTIFICATION Object

The NOTIFICATION object is exclusively carried within a PCNtf message and can either be used in a message sent by a PCC to a PCE or by a PCE to a PCC so as to notify of an event.

NOTIFICATION Object-Class is to be assigned by IANA (recommended

value=12)

NOTIFICATION Object-Type is to be assigned by IANA (recommended value=1)

One Object-Type is defined for this object to be assigned by IANA with a recommended value of 1.

The format of the NOTIFICATION body object is as follows:

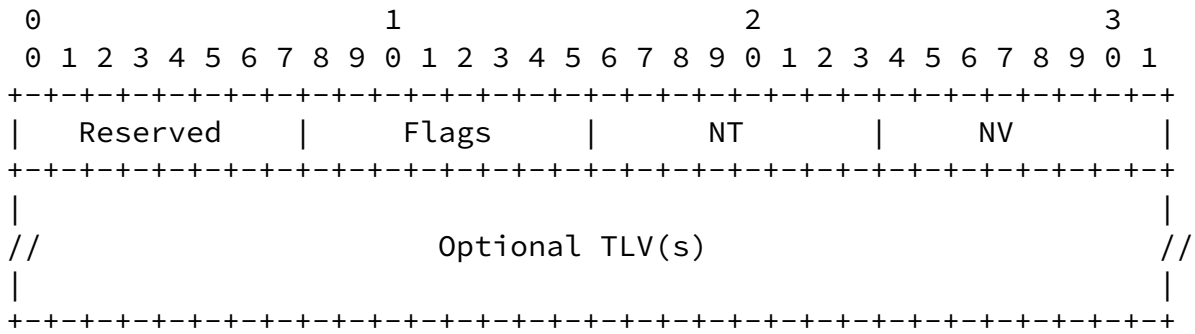



Figure 19: NOTIFICATION body object format

Reserved (8 bits): This field MUST be set to zero on transmission and MUST be ignored on receipt.

Flags (8 bits): no flags are currently defined. Unassigned flags MUST be set to zero on transmission and MUST be ignored on receipt.

NT (Notification Type - 8 bits): the Notification-type specifies the class of notification





NV (Notification Value - 8 bits): the Notification-value provides addition information related to the nature of the notification.

Both the Notification-type and Notification-value should be managed by IANA.

The following Notification-type and Notification-value values are currently defined:

o Notification-type=1: Pending Request cancelled

* Notification-value=1: PCC cancels a set of pending request(s). A Notification-type=1, Notification-value=1 indicates that the PCC wants to inform a PCE of the cancellation of a set of pending request(s). Such event could be triggered because of external conditions such as the receipt of a positive reply from another PCE (should the PCC have sent multiple requests to a set of PCEs for the same path computation request), a network event such as a network failure rendering the request obsolete or any other event(s) local to the PCC. A NOTIFICATION object with Notification-type=1, Notification-value=1 is carried within a PCNtf message sent by the PCC to the PCE. The RP object corresponding to the cancelled request MUST also be present in the PCNtf message. Multiple RP objects may be carried within the PCNtf message in which case the notification applies to all of them. If such notification is received by a PCC from a PCE, the PCC MUST silently ignore the notification and no errors should be generated.

* Notification-value=2: PCE cancels a set of pending request(s). A Notification-type=1, Notification-value=2 indicates that the PCE wants to inform a PCC of the cancellation of a set of pending request(s). Such event could be triggered because of PCE overloaded state or because of missing path computation requests that are part the set of synchronized path computation requests. A NOTIFICATION object with Notification-type=1, Notification-value=2 is carried within a PCNtf message sent by a PCE to a PCC. The RP object corresponding to the cancelled request MUST also be present in the PCNtf message. Multiple RP objects may be carried within the PCNtf message in which case the notification applies to all of them. If such notification is received by a PCE from a PCC, the PCE MUST silently ignore the notification and no errors should be generated.

o Notification-type=2: Overloaded PCE

* Notification-value=1: A Notification-type=2, Notification-value=1 indicates to the PCC(s) that the PCE is currently in an





overloaded state. If no RP objects are comprised in the PCNtf message, this indicates that no other requests SHOULD be sent to that PCE until the overloaded state is cleared: the pending requests are not affected and will be served. If some pending

requests cannot be served due to the overloaded state, the PCE MUST also include a set of RP object(s) that identifies the set of pending requests that are cancelled by the PCE and will not be honored. In this case, the PCE does not have to send an additional PCNtf message with Notification-type=1 and Notification-value=2 since the list of cancelled requests is specified by including the corresponding set of RP object(s). If such notification is received by a PCE from a PCC, the PCE MUST silently ignore the notification and no errors should be generated.

Optionally, a TLV named OVERLOADED-DURATION may be included in the NOTIFICATION object that specifies the period of time during which no further request should be sent to the PCE. Once this period of time has elapsed, the PC should no longer be considered in congested state.

The OVERLOADED-DURATION TLV is composed of 1 byte for the type, 1 byte specifying the number of bytes in the value field, 2 bytes for an "Unused" field (the value of which MUST be set to 0), followed by a fix length value field of 4 bytes specifying the estimated PCE congestion duration in seconds. The OVERLOADED-DURATION TLV is padded to eight-byte alignment.

TYPE: To be assigned by IANA
LENGTH: 4
VALUE: period of time (in seconds) during which the PCE should be considered as overloaded.

* Notification-value=2: A Notification-type=2, Notification-value=2 indicates that the PCE is no longer in congested state and is available to process new path computation requests. An implementation MUST make sure that a PCE sends such notification to every PCC to which a Notification message (with Notification-type=2, Notification-value=1) has been sent unless an OVERLOADED-DURATION TLV has been included in the corresponding message and the PCE wishes to wait for the expiration of that period of time before receiving new requests. If such notification is received by a PCE from a PCC, the PCE MUST silently ignore the notification and no errors should be generated. It is RECOMMENDED to support some dampening notification procedure on the PCE so as to avoid too frequent congestion state and congestion state release notifications. For example, an implementation could make use of an hysteresis approach using a dual-thresholds mechanism

triggering the sending of congestion state notifications. Furthermore, in case of high instabilities of the PCE resources, an additional dampening mechanism SHOULD be used (linear or exponential) to pace the notification frequency and avoid path computation requests oscillation.

* Alternatively, PCE may decide to signal its (non) overloaded state using a IGP-based notification mechanism as defined in [I-D.ietf-pce-disco-proto-isis]and [I-D.ietf-pce-disco-proto-ospf]. A PCE may also decide to signal its overloaded state using PCEP and its no longer overloaded state using an IGP-based notification and vice-versa.

## 7.14. PCEP-ERROR Object

The PCEP-ERROR object is exclusively carried within a PCErr message to notify of a PCEP error.

PCEP-ERROR Object-Class is to be assigned by IANA (recommended value=13)

PCEP-ERROR Object-Type is to be assigned by IANA (recommended value=1)

One Object-Type is defined for this object to be assigned by IANA with a recommended value of 1.

The format of the PCEP-ERROR object body is as follows:

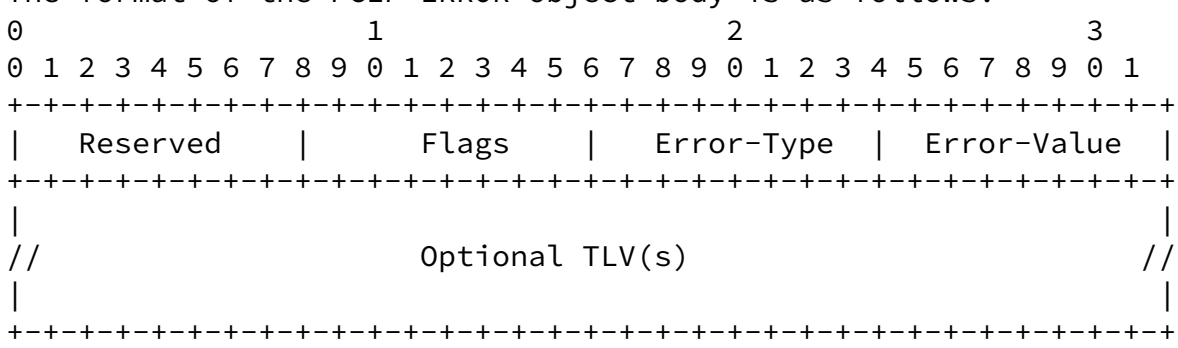



Figure 20: PCEP-ERROR object body format

A PCEP-ERROR object is used to report a PCEP error and is characterized by an Error-Type that specifies the type of error and an Error-value that provides additional information about the error type. Both the Error-Type and the Error-Value should be managed by IANA (see the IANA section).

Reserved (8 bits): This field MUST be set to zero on transmission and MUST be ignored on receipt.

Flags (8 bits): no flag is currently defined. This flag MUST be set to zero on transmission and MUST be ignored on receipt.

Error-type (8 bits): defines the class of error.

Error-value (8 bits): provides additional details about the error.

Optionally the PCEP-ERROR object may contain additional TLV so as to provide further information about the encountered error.

A single PCErr message may contain multiple PCEP-ERROR objects.

For each PCEP error, an Error-type and an Error-value are defined.

| Error-Type | Meaning |
|---|---|
| 1 | PCEP session establishment failure |
| | Error-value=1: reception of a malformed message |
| | Error-value=2: no Open message received before the expiration of the OpenWait timer |
| | Error-value=3: unacceptable and non negotiable session characteristics |
| | Error-value=4: unacceptable but negotiable session characteristics |
| | Error-value=5: reception of a second Open message with still unacceptable session characteristics |
| | Error-value=6: reception of a PCErr message proposing unacceptable session characteristics |
| | Error-value=7: No Keepalive or PCErr message received before the expiration of the KeepWait timer |
| 2 | Capability not supported |
| 3 | Unknown Object |
| | Error-value=1: Unrecognized object class |
| | Error-value=2: Unrecognized object Type |
| 4 | Not supported object |
| | Error-value=1: Not supported object class |
| | Error-value=2: Not supported object Type |
| 5 | Policy violation |
| | Error-value=1: C bit of the METRIC object set (request rejected) |
| | Error-value=2: O bit of the RP object set (request rejected) |
| 6 | Mandatory Object missing |
| | Error-value=1: RP object missing |
| | Error-value=2: RRO object missing for a reoptimization request (R bit of the RP object set) when bandwid is not equal to 0. |
| | Error-value=3: END-POINTS object missing |
| 7 | Synchronized path computation request missing |

8 Unknown request reference
9 Attempt to establish a second PCEP session

Error-Type=1: PCEP session establishment failure.

If a malformed message is received, the receiving PCEP peer MUST send a PCErr message with Error-type=1, Error-value=1.

If no Open message is received before the expiration of the OpenWait timer, the receiving PCEP peer MUST send a PCErr message with Error-type=1, Error-value=2 (see Section 10 for details).

If one or more PCEP session characteristic(s) are unacceptable by the receiving peer and are not negotiable, it MUST send a PCErr message with Error-type=1, Error-value=3.





If an Open message is received with unacceptable session characteristics but these characteristics are negotiable, the receiving PCEP peer MUST send a PCErr message with Error-type-1, Error-value=4 (see Section 6.2 for details).

If a second Open message is received during the PCEP session establishment phase and the session characteristics are still unacceptable, the receiving PCEP peer MUST send a PCErr message with Error-type-1, Error-value=5 (see Section 6.2 for details).

If a PCErr message is received during the PCEP session establishment phase that contains an Open message proposing unacceptable session characteristics, the receiving PCEP peer MUST send a PCErr message with Error-type=1, Error-value=6.

If neither a Keepalive message nor a PCErr message is received before the expiration of the KeepWait timer during the PCEP session establishment phase, the receiving PCEP peer MUST send a PCErr message with Error-type=1, Error-value=7.

Error-Type=2: the PCE indicates that the path computation request cannot be honored because it does not support one or more required capability. The corresponding path computation request MUST be cancelled.

Error-Type=3 or Error-Type=4: if a PCEP message is received that

carries a PCEP object (with the P flag set) not recognized by the PCE or recognized but not supported, then the PCE MUST send a PCErr message with a PCEP-ERROR object (Error-Type=3 and 4 respectively). In addition, the PCC MAY include in the PCErr message the unknown or not supported object. The corresponding path computation request MUST be cancelled by the PCE without further notification.

Error-Type=5: if a path computation request is received that is not compliant with an agreed policy between the PCC and the PCE, the PCE MUST send a PCErr message with a PCEP-ERROR object (Error-Type=5). The corresponding path computation MUST be cancelled. Policy-specific TLV(s) carried within the PCEP-ERROR object may be defined in other documents to specify the nature of the policy violation.

Error-Type=6: if a path computation request is received that does not contain a mandatory object, the PCE MUST send a PCErr message with a PCEP-ERROR object (Error-Type=6). If there are multiple mandatory objects missing, the PCErr message MUST contain one PCEP-ERROR object per missing object. The corresponding path computation MUST be cancelled.

Error-Type=7: if a PCC sends a synchronized path computation request





to a PCE and the PCE does not receive all the synchronized path computation requests listed within the corresponding SVEC object after the expiration of the timer SyncTimer defined in Section 7.12.3, the PCE MUST send a PCErr message with a PCEP-ERROR object (Error-Type=7). The corresponding synchronized path computation MUST be cancelled. It is RECOMMENDED for the PCE to include the REQ-MISSING TLV(s) (defined below) that identifies the missing request(s).

The REQ-MISSING TLV is composed of 1 byte for the type, 1 byte specifying the number of bytes in the value field, 2 bytes for an "Unused" field (the value of which MUST be set to 0), followed by a fix length value field of 4 bytes specifying the request-id-number that correspond to the missing request. The REQ-MISSING TLV is padded to eight-byte alignment.

TYPE: To be assigned by IANA
LENGTH: 4
VALUE: request-id-number that corresponds to the missing request

Error-Type=8: if a PCC receives a PCRep message related to an unknown path computation request, the PCC MUST send a PCErr message with a PCEP-ERROR object (Error-Type=8). In addition, the PCC MUST include in the PCErr message the unknown RP object.

Error-Type=9: if a PCEP peer detects an attempt from another PCEP peer to establish a second PCEP session, it MUST send a PCErr message with Error-type=9, Error-value=1. The existing PCEP session MUST be preserved and all subsequent messages related to the tentative establishment of the second PCEP session MUST be silently ignored.

## 7.15. LOAD-BALANCING Object

There are situations where no TE LSP with a bandwidth of X could be found by a PCE although such bandwidth requirement could be satisfied by a set of TE LSPs such that the sum of their bandwidths is equal to X. Thus it might be useful for a PCC to request a set of TE LSPs so that the sum of their bandwidth is equal to X MBits/s, with potentially some constraints on the number of TE LSPs and the minimum bandwidth of each of these TE LSPs. Such request is made by inserting a LOAD-BALANCING object in a PCReq message sent to a PCE.

The LOAD-BALANCING object is optional.

LOAD-BALANCING Object-Class is to be assigned by IANA (recommended value=14)

LOAD-BALANCING Object-Type is to be assigned by IANA (recommended





value=1)

The format of the LOAD-BALANCING object body is as follows:

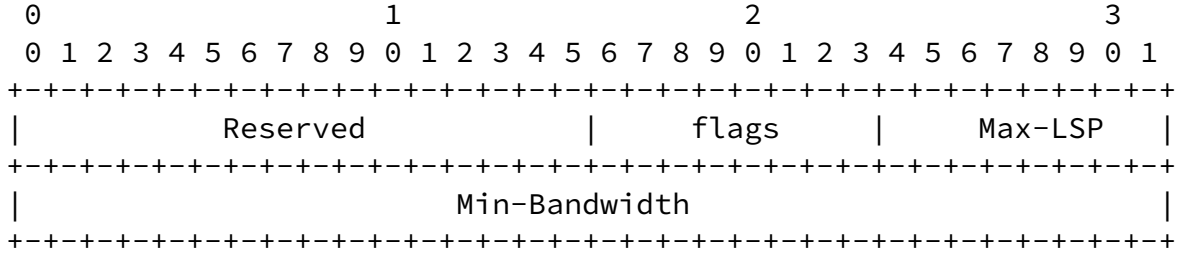



Figure 21: LOAD-BALANCING object body format

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

Flags (8 bits): No Flag is currently defined. The Flag field MUST be set to zero on transmission and MUST be ignored on receipt).

Max-LSP (8 bits): maximum number of TE LSPs in the set

Min-Bandwidth (32 bits). Specifies the minimum bandwidth of each element of the set of TE LSPs. The bandwidth is encoded in 32 bits in IEEE floating point format, expressed in bytes per second.

The LOAD-BALANCING object body has a fixed length of 8 bytes.

If a PCC requests the computation of a set of TE LSP(s) so that the sum of their bandwidth is X, the maximum number of TE LSP is N and each TE LSP must at least have a bandwidth of B, it inserts a BANDWIDTH object specifying X as the required bandwidth and a LOAD-BALANCING object with the Max-LSP and Min-Bandwidth fields set to N and B respectively.

## 7.16. CLOSE Object

The CLOSE object MUST be present in each Close message. There MUST be only one CLOSE object per Close message. If a Close message is received that contains more than one CLOSE object, the first CLOSE object is the one that must be processed. Other CLOSE object(s) MUST be silently ignored.

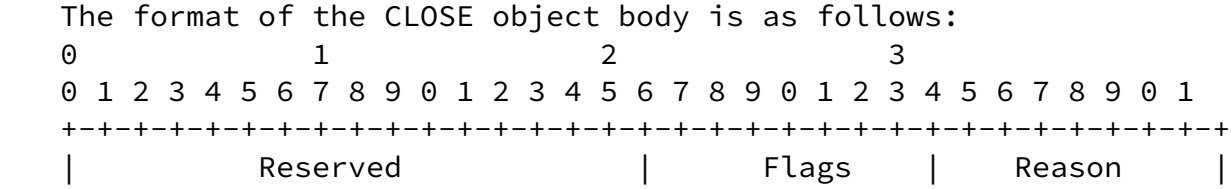
CLOSE Object-Class is to be assigned by IANA (recommended value=15)

CLOSE Object-Type is to be assigned by IANA (recommended value=1)





The format of the CLOSE object body is as follows:



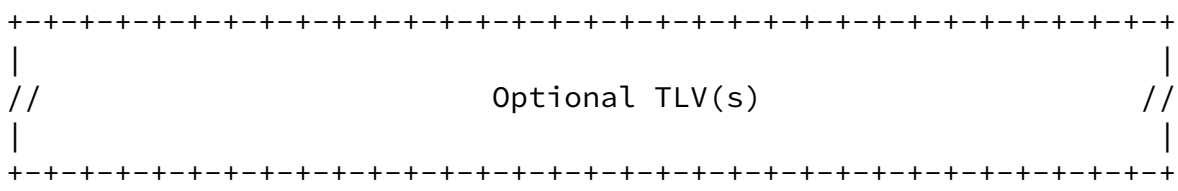



Figure 22: CLOSE Object format

Reason (4 bits): specifies the reason for closing the PCEP session. The setting of this field is optional. The following values are currently defined.

Reasons

| Value | Meaning |
|---|---|
| 1 | No explanation provided |
| 2 | DeadTimer expired |
| 3 | Reception of a malformed PCEP message |

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

Flags (4 bits): No Flags are currently defined. The Flag field MUST be set to zero on transmission and MUST be ignored on receipt.

Optional TLVs may be included within the CLOSE object body. The specification of such TLVs is outside the scope of this document.

## 8. Manageability Considerations

This section follows the guidance of [I-D.ietf-pce-manageability-requirements].

### 8.1. Control of Function and Policy

A PCEP implementation SHOULD allow configuring the following PCEP session parameters on a PCEP peer:

- The local keepalive and Deadtimer (i.e. parameters send by the PCEP speaker in an Open message),
- The maximum acceptable remote keepalive and dead timers (i.e.parameters sent by a peer in an Open message),

- Negotiation enabled or disabled,
- If negotiation is allowed, the minimum acceptable Keepalive and Deadtimer timers sent by a PCEP peer,
- The SyncTimer,
- The maximum number of sessions that can be setup,
- Request timer: amount of time a PCC waits for a reply before resending its path computation requests (potentially to an alternate PCE).

These parameters may be configured as default parameters for any PCEP session the PCEP speaker participates in, or may apply to a specific session with a given PCEP peer or a specific group of sessions with a specific group of PCEP peers. A PCEP implementation SHOULD allow configuring the initiation of a PCEP session with a selected subset of discovered PCEs. Note that PCE selection is a local implementation issue. A PCEP implementation SHOULD allow configuring a specific PCEP session with a given PCEP peer. This includes the configuration of the following parameters:

- The IP address of the PCEP peer,
- The PCEP speaker role: PCC, PCE or both,
- Whether the PCEP speaker should initiate the PCEP session or wait for initiation by the peer,
- The PCEP session parameters, as listed above, if they differ from the default parameters,
- A set of PCEP policies including the type of operations allowed for the PCEP peer (e.g. diverse path computation, synchronization, etc.)

A PCEP implementation MUST allow restricting the set of PCEP peers that can initiate a PCEP session with the PCEP speaker (e.g. list of authorized PCEP peers, all PCEP peers in the area, all PCEP peers in the AS).

## 8.2. Information and Data Models

A PCEP MIB module is defined in [I-D.kkoushik-pce-pcep-mib] that describes managed objects for modeling of PCEP communication including:

- PCEP client configuration and status,
- PCEP peer configuration and information,
- PCEP session configuration and information,
- Notifications to indicate PCEP session changes.

## 8.3. Liveness Detection and Monitoring

PCEP includes a keepalive mechanism, allowing checking the liveliness of a PCEP peer and a notification procedure allowing a PCE to advertise its congestion state to a PCC. Also, procedures in order to monitor the liveliness and performances of a given PCE chain (in case of Multiple-PCE path computation) are defined in [I-D.vasseur-pce-monitoring].

## 8.4. Verifying Correct Operation

Verifying the correct operation of a PCEP communication can be performed by monitoring various parameters. A PCEP implementation SHOULD provide the following parameters:

- Response time (minimum, average and maximum), on a per PCE Peer basis,
- PCEP Session failures,
- Amount of time the session has been in active state,
- Number of corrupted messages,
- Number of failed computations,
- Number of requests for which no reply has been received after the expiration of a configurable timer and by verifying that a returned path fit in with the requested TE parameters.

A PCEP implementation SHOULD log error events (e.g. corrupted messages, unrecognized objects, etc.).

## 8.5. Requirements on Other Protocols and Functional Componentssection

PCEP does not put any new requirements on other protocols. As PCEP relies on the TCP transport protocol, PCEP management can make use of TCP management mechanisms (such as the TCP MIB defined in [RFC4022]). The PCE Discovery mechanisms ([I-D.ietf-pce-disco-proto-isis], [I-D.ietf-pce-disco-proto-ospf]) may have an impact on PCEP. To





avoid that a high frequency of PCE Discovery/Disappearance trigger high frequency of PCEP session setup/deletion, it is RECOMMENDED to introduce some dampening for establishment of PCEP sessions.

### 8.6. Impact on Network Operation

In order to avoid any unacceptable impact on network operations, an implementation SHOULD allow limiting the number of session that can be setup on a PCEP speaker, and MAY allow limiting the rate of messages sent by a PCEP speaker and received from a peer. It MAY also allow sending a notification when a rate threshold is reached.

## 9. IANA Considerations

### 9.1. TCP Port

PCEP will use a well-known TCP port to be assigned by IANA.

### 9.2. PCEP Messages

Each PCEP message has a Message-Type.

| Value | Meaning |
|---|---|
| 1 | Open |
| 2 | Keepalive |
| 3 | Path Computation Request |
| 4 | Path Computation Reply |
| 5 | Notification |
| 6 | Error |
| 7 | Close |

### 9.3. PCEP Object

IANA assigns value to PCEP parameters. Each PCEP object has an

Object-Class and an Object-Type.

| Object-Class | Name | |
|---|---|---|
| 1 | OPEN | |
| | Object-Type | |
| | | 1 |
| 2 | RP | |
| | Object-Type | |
| | | 1 |
| 3 | NO-PATH | |
| | Object-Type | |
| | | 1 |
| 4 | END-POINTS | |
| | Object-Type | |
| | | 1 : IPv4 addresses |
| | | 2: IPv6 addresses |
| 5 | BANDWIDTH | |
| | Object-Type | |
| | | 1: Requested bandwidth |
| | | 2: Bandwidth of an existing TE LSP for which a reoptimization is performed. |
| 6 | METRIC | |
| | Object-Type | |
| | | 1 |
| 7 | ERO | |
| | Object-Type | |
| | | 1 |
| 8 | RRO | |
| | Object-Type | |
| | | 1 |
| 9 | LSPA | |
| | Object-Type | |

| | | |
|---|---|---|
| | | 1 |
| 10 | | IRO<br>Object-Type<br>1 |
| 11 | | SVEC<br>Object-Type<br>1 |
| 12 | | NOTIFICATION<br>Object-Type<br>1 |
| 13 | | PCEP-ERROR<br>Object-Type<br>1 |
| 14 | | LOAD-BALANCING<br>Object-Type |





| | | |
|---|---|---|
| | | 1 |
| 15 | | CLOSE<br>Object-Type<br>1 |

## 9.4. Notification Object

A NOTIFICATION object is characterized by a Notification-type that specifies the class of notification and a Notification-value that provides additional information related to the nature of the notification. Both the Notification-type and Notification-value are managed by IANA (see IANA section).

| Notification-type | Name |
|---|---|
| 1 | Pending Request cancelled<br>Notification-value<br>1: PCC cancels a set of pending request<br>2: PCE cancels a set of pending request |
| 2 | PCE Congestion<br>Notification-value |

1: PCE in congested state
2: PCE no longer in congested state

## 9.5. PCEP Error Object

PCEP-ERROR objects are used to report a PCEP error and are characterized by an Error-Type which specifies the type of error and an Error-value that provides additional information about the error type. Both the Error-Type and the Error-Value are managed by IANA.





For each PCEP error, an Error-type and an Error-value are defined.

| Error-Type | Meaning |
|---|---|
| 1 | PCEP session establishment failure |
| | Error-value=1: reception of a malformed message |
| | Error-value=2: no Open message received before the expiration of the OpenWait timer |
| | Error-value=3: unacceptable and non negotiable session characteristics |
| | Error-value=4: unacceptable but negotiable session characteristics |
| | Error-value=5: reception of a second Open message with still unacceptable session characteristics |
| | Error-value=6: reception of a PCErr message proposing unacceptable session characteristics |

| | |
|---|---|
| | Error-value=7: No Keepalive or PCErr message received before the expiration of the KeepWait timer |
| | Error-value=8: PCEP version not supported |
| 2 | Capability not supported |
| 3 | Unknown Object |
| | Error-value=1: Unrecognized object class |
| | Error-value=2: Unrecognized object Type |
| 4 | Not supported object |
| | Error-value=1: Not supported object class |
| | Error-value=2: Not supported object Type |
| 5 | Policy violation |
| | Error-value=1: C bit of the METRIC object set (request rejected) |
| | Error-value=2: O bit of the RP object cleared (request rejected) |
| 6 | Mandatory Object missing |
| | Error-value=1: RP object missing |
| | Error-value=2: RRO missing for a reoptimization request (R bit of the RP object set) |
| | Error-value=3: END-POINTS object missing |
| 7 | Synchronized path computation request missing |
| 8 | Unknown request reference |
| 9 | Attempt to establish a second PCEP session |

## 9.6. CLOSE Object

The CLOSE object MUST be present in each Close message in order to close a PCEP session. The reason field of the CLOSE object specifies the reason for closing the PCEP session. The reason field of the CLOSE object is managed by IANA.





Reasons

| Value | Meaning |
|---|---|
| 1 | No explanation provided |
| 2 | DeadTimer expired |
| 3 | Reception of a malformed PCEP message |

## 9.7. NO-PATH-VECTOR TLV

IANA is requested to manage the space of flags carried in the NO-PATH-VECTOR TLV defined in this document, numbering them in the usual IETF notation starting at zero and continuing through 31.

New bit numbers may be allocated only by an IETF Consensus action.

Each bit should be tracked with the following qualities: - Bit number - Defining RFC - Name of bit

Currently two bits are defined.

Here are the suggested values:

0x01: PCE currently Unavailable
0x02: Unknown Destination
0x03: Unknown Source

## 10. PCEP Finite State Machine (FSM)

The section describes the PCEP Finite State Machine (FSM).

PCEP Finite State Machine

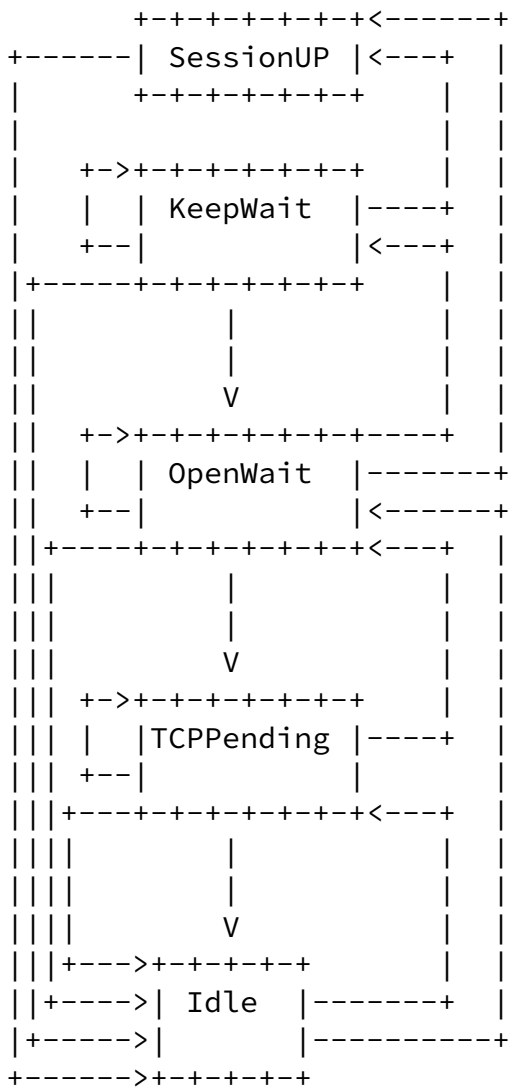



Figure 23: PCEP Finite State Machine for the PCC

PCEP defines the following set of variables:

Connect: timer (in seconds) started after having initialized a TCP connection using the PCEP well-known TCP port. The value of the TCPConnect timer is 60 seconds.

ConnectRetry: specifies the number of times the system has tried to establish a TCP connection with a PCEP peer without success.

ConnectMaxRetry: Maximum number of times the system tries to establish a TCP connection using the PCEP well-known TCP port before going back to the Idle state. The value of the ConnectMaxRetry is 5.

OpenWait: timer that corresponds to the amount of time a PCEP peer will wait to receive an Open message from the PCEP peer after the expiration of which the system releases the PCEP resource and go back to the Idle state. The OpenWait timer has a fixed value of 60 seconds.

KeepWait: timer that corresponds to the amount of time a PCEP peer will wait to receive a KeepAlive or a PCErr message from the PCEP peer after the expiration of which the system releases the PCEP resource and go back to the Idle state. The KeepWait timer has a fixed value of 60 seconds.

OpenRetry: specifies the number of times the system has received an Open message with unacceptable PCEP session characteristics.

The following two states variable are defined:

RemoteOK: the RemoteOK variable is a Boolean set to 1 if the system has received an acceptable Open message.

LocalOK: the LocalOK variable is a Boolean set to 1 if the system has received a Keepalive message acknowledging that the Open message sent to the peer was valid.

Idle State:

The idle state is the initial PCEP state where PCEP (also referred to as "the system") waits for an initialization event that can either be manually triggered by the user (configuration) or automatically triggered by various events. In Idle state, PCEP resources are allocated (memory, potential process, ...) but no PCEP messages are accepted from any PCEP peer. The system listens the well-known PCEP TCP port.

The following set of variable are initialized:

TCPRetry=0,

LocalOK=0,

RemoteOK=0,

OpenRetry=0.

Upon detection of a local initialization event (e.g. user configuration to establish a PCEP session with a particular PCEP peer, local event triggering the establishment of a PCEP session with a PCEP peer such as the automatic detection of a PCEP peer, ...), the





system:

- Initiates of a TCP connection with the PCEP peer,
- Starts the Connect timer,
- Moves to the TCPPending state.

Upon receiving a TCP connection on the well-known PCEP TCP port, if the TCP connection establishment succeeds, the system:

- Sends an Open message,
- Starts the OpenWait timer,
- Moves to the OpenWait state.

If the connection establishment fails, the system remains in the Idle state. Any other event received in the Idle state is ignored.

It is expected that an implementation will use an exponentially increase timer between automatically generated Initialization events and between retrials of TCP connection establishments.

TCPPending State

If the TCP connection establishment succeeds, the system:

- Sends an Open message,
- Starts the OpenWait timer,
- Moves to the OpenWait state.

If the TCP connection establishment fails (an error is detected during the TCP connection establishment) or the Connect timer

expires:

If ConnectRetry =ConnectMaxRetry the system moves to the Idle State

If ConnectRetry < ConnectMaxRetry the system:

- Initiates of a TCP connection with the PCEP peer,
- Increments the ConnectRetry variable,
- Restarts the Connect timer,





- Stays in the TPCPending state.

If the system detects that the PCEP peer tries to simultaneously establish a TCP connection, it stops the TCP connection establishment if and only if the PCEP peer has a higher IP address and moves to the Idle state. This guarantees that in case of "collision" a single TCP connection is established.

In response to any other event the system releases the PCEP resources for that peer and moves back to the Idle state.

OpenWait State:

In the OpenWait state, the system waits for an Open message from its PCEP peer.

If the system receives an Open message from the PCEP peer before the expiration of the OpenWait timer, PCEP checks the PCEP session attributes (Keepalive frequency, DeadTimer, ...).

If an error is detected (e.g. malformed Open message, presence of two Open objects, ...), PCEP generates an error notification, the PCEP peer sends a PCErr message with Error-Type=1 and Error-value=1. The system releases the PCEP resources for the PCEP peer, closes the TCP connection and moves to the Idle state.

If no errors are detected, PCEP increments the OpenRetry variable.

If no errors are detected, OpenRetry=1 and the session

characteristics are unacceptable, the PCEP peer sends a PCErr with Error-Type=1 and Error-value=5, the system releases the PCEP resources for that peer and moves back to the Idle state.

If no errors are detected and the session characteristics are acceptable to the local system, the system:

- Sends a Keepalive message to the PCEP peer,
- Starts the Keepalive timer,
- Sets the RemoteOK variable to 1.

If LocalOK=1 the system clears the OpenWait timer and moves to the UP state.

If LocalOK=0 the system clears the OpenWait timer, starts the KeepWait timer and moves to the KeepWait state.





If no errors are detected but the session characteristics are unacceptable and non-negotiable, the PCEP peer sends a PCErr with Error-Type=1 and Error-value=3, the system releases the PCEP resources for that peer, and moves back to the Idle state.

If no errors are detected, OpenRetry=0, the session characteristics are unacceptable but negotiable (such as the Keepalive period or the DeadTimer), the system:

- Increments the OpenRetry variable,
- Sends a PCErr message with Error-Type=1 and Error-value=4 that contains proposed acceptable session characteristics,
- If LocalOK=1, the system restarts the OpenWait timer and stays in the OpenWait state
- If LocalOK=0, the system clears the OpenWait timer, starts the KeepWait timer and moves to the KeepWait state

If no Open message is received before the expiration of the OpenWait timer, the PCEP peer sends a PCErr message with Error-Type=1 and

Error-value=2, the system releases the PCEP resources for the PCEP peer, closes the TCP connection and moves to the Idle state.

In response to any other event the system releases the PCEP resources for that peer and moves back to the Idle state.

KeepWait State

In the Keepwait state, the system waits for the receipt of a Keepalive from its PCEP peer acknowledging its Open message or a PCErr message in response to unacceptable PCEP session characteristics proposed in the Open message.

If an error is detected (e.g. malformed Keepalive message), PCEP generates an error notification, the PCEP peer sends a PCErr message with Error-Type=1 and Error-value=1. The system releases the PCEP resources for the PCEP peer, closes the TCP connection and moves to the Idle state.

If a Keepalive message is received before the expiration of the KeepWait timer, then the system sets LocalOK=1 and:

- If RemoteOK=1, the system clears the KeepWait timer and moves to the UP state.





- If RemoteOK=0, the system clears the KeepWait timer, starts the OpenWait timer and moves to the OpenWait State.

If a PCErr message is received before the expiration of the KeepWait timer:

1. If the proposed values are unacceptable, the PCEP peer sends a PCErr message with Error-Type=1 and Error-value=6 and the system releases the PCEP resources for that PCEP peer, closes the TCP connection and moves to the Idle state.

2. If the proposed values are acceptable, the system adjusts its PCEP session characteristics according to the proposed values received in the PCErr message restarts the KeepWait timer and sends a new Open message. If RemoteOK=1, the system restarts the

KeepWait timer and stays in the KeepWait state. If RemoteOK=0, the system clears the KeepWait timer, start the OpenWait timer and moves to the OpenWait state.

If neither a Keepalive nor a PCErr is received after the expiration of the KeepWait timer, the PCEP peer sends a PCErr message with Error-Type=1 and Error-value=7 and, system releases the PCEP resources for that PCEP peer, closes the TCP connection and moves to the Idle State.

In response to any other event the system releases the PCEP resources for that peer and moves back to the Idle state.

UP State

In the UP state, the PCEP peer starts exchanging PCEP messages according to the session characteristics.

If the Keepalive timer expires, the system restarts the Keepalive timer and sends a Keepalive message.

If no PCEP message (Keepalive, PCReq, PCRep, PCNtf) is received from the PCEP peer after the expiration of the DeadTimer, the system terminates PCEP session according to the procedure defined in Section 6.8, releases the PCEP resources for that PCEP peer, closes the TCP connection and moves to the Idle State.

If a malformed message is received, the system terminates the PCEP session according to the procedure defined in Section 6.8, releases the PCEP resources for that PCEP peer, closes the TCP connection and moves to the Idle State.

If the system detects that the PCEP peer tries to setup a second TCP





connection, it stops the TCP connection establishment and sends a PCErr with Error-Type=9.

If the TCP connection fails, the system releases the PCEP resources for that PCEP peer, closes the TCP connection and moves to the Idle State.

## 11. Security Considerations

PCEP could be the target of the following attacks:

- Spoofing (PCC or PCE impersonation)
- Snooping (message interception)
- Falsification
- Denial of Service

A PCEP attack may have significant impact, particularly in an inter-AS context as PCEP facilitates inter-AS path establishment. Several mechanisms are proposed below, so as to ensure authentication, integrity and privacy of PCEP Communications, and also to protect against DoS attacks.

### 11.1. PCEP Authentication and Integrity

It is RECOMMENDED to use TCP-MD5 [RFC1321] signature option to provide for the authenticity and integrity of PCEP messages. This will allow protecting against PCE or PCC impersonation and also against message content falsification.

This requires the maintenance, exchange and configuration of MD-5 keys on PCCs and PCEs. Note that such maintenance may be especially onerous to the operators as pointed out in [I-D.ietf-rpsec-bgpsecrec]. Hence it is important to limit the number of keys while ensuring the required level of security.

MD-5 signature faces some limitations, as per explained in [RFC2385]. Note that when one digest technique stronger than MD5 is specified and implemented, PCEP could be easily upgraded to use it.

### 11.2. PCEP Privacy

Ensuring PCEP communication privacy is of key importance, especially in an inter-AS context, where PCEP communication end-points do not reside in the same AS, as an attacker that intercept a PCE message





could obtain sensitive information related to computed paths and

resources. Privacy can be ensured thanks to encryption. To ensure privacy of PCEP communication, IPSec [RFC4303] tunnels MAY be used between PCC and PCEs or between PCEs. Note that this could also be used to ensure Authentication and Integrity, in which case, TCP MD-5 option would not be required.

### 11.3. Protection against Denial of Service attacks

PCEP can be the target of TCP DoS attacks, such as for instance SYN attacks, as all protocols running on top of TCP. PCEP can use the same mechanisms as defined in [RFC3036] to mitigate the threat of such attacks:

- A PCE should avoid promiscuous TCP listens for PCEP TCP connection establishment. It should use only listens that are specific to authorized PCCs.

- The use of the MD5 option helps somewhat since it prevents a SYN from being accepted unless the MD5 segment checksum is valid. However, the receiver must compute the checksum before it can decide to discard an otherwise acceptable SYN segment.

- The use of access-list on the PCE so as to restrict access to authorized PCCs.

### 11.4. Request input shaping/policing

A PCEP implementation may be subject to Denial Of Service attacks consisting of sending a very large number of PCEP messages (e.g. PCReq messages). Thus, especially in multi-Service Providers environments, a PCE implementation should implement request input shaping/policing so as to throttle the amount of received PCEP messages without compromising the implementation behavior.

## 12. Authors' addresses

This document was the collective work of several authors. The content of this document was contributed by the editors and the co-authors listed below:

Arthi Ayyangar
Nuova Systems
2600 San Tomas Expressway
Santa Clara, CA 95051
USA

Email: arthi@nuovasystems.com

Eiji Oki
NTT
Midori 3-9-11
Musashino, Tokyo, 180-8585
JAPAN

Email: oki.eiji@lab.ntt.co.jp

Alia Atlas
Google
1600 Amphitheatre Parkway
Montain View, CA 94043
USA

Email: akatlas@alum.mit.edu

Andrew Dolganow
Alcatel
600 March Road
Ottawa, ON K2K 2E6
CANADA

Email: andrew.dolganow@alcatel.com

Yuichi Ikejiri
NTT Communications Corporation
1-1-6 Uchisaiwai-cho, Chiyoda-ku
Tokyo, 100-819
JAPAN

Email: y.ikejiri@ntt.com

Kenji Kumaki
KDDI Corporation
Garden Air Tower Iidabashi, Chiyoda-ku,
Tokyo, 102-8460
JAPAN

Email: ke-kumaki@kddi.com

## 13. Acknowledgements

The authors would like to thank Dave Oran, Dean Cheng, Jerry Ash, Igor Bryskin, Carol Iturrade, Siva Sivabalan, Rich Bradford, Richard Douville, Jon Parker and Martin German for their very valuable input. Special thank to Adrian Farrel for his very valuable suggestions.

## 14. References

### 14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2205] Braden, B., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3473] Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

[RFC3477] Kompella, K. and Y. Rekhter, "Signalling Unnumbered Links in Resource ReSerVation Protocol - Traffic Engineering (RSVP-TE)", RFC 3477, January 2003.

[RFC4090] Pan, P., Swallow, G., and A. Atlas, "Fast Reroute Extensions to RSVP-TE for LSP Tunnels", RFC 4090, May 2005.

### 14.2. Informative References

[I-D.ietf-ccamp-inter-domain-rsvp-te] Ayyangar, A., "Inter domain Multiprotocol Label Switching

(MPLS) and Generalized MPLS (GMPLS) Traffic Engineering - RSVP-TE extensions", draft-ietf-ccamp-inter-domain-rsvp-te-06 (work in progress), April 2007.

[I-D.ietf-pce-disco-proto-isis]
Roux, J., "IS-IS protocol extensions for Path Computation Element (PCE) Discovery", draft-ietf-pce-disco-proto-isis-06 (work in progress),





June 2007.

[I-D.ietf-pce-disco-proto-ospf]
Roux, J., "OSPF protocol extensions for Path Computation Element (PCE) Discovery", draft-ietf-pce-disco-proto-ospf-06 (work in progress), June 2007.

[I-D.ietf-pce-inter-layer-req]
Oki, E., "PCC-PCE Communication Requirements for Inter-Layer Traffic Engineering", draft-ietf-pce-inter-layer-req-04 (work in progress), March 2007.

[I-D.ietf-pce-interas-pcecp-reqs]
Bitar, N., "Inter-AS Requirements for the Path Computation Element Communication Protocol (PCECP)", draft-ietf-pce-interas-pcecp-reqs-01 (work in progress), October 2006.

[I-D.ietf-pce-manageability-requirements]
Farrel, A., "Inclusion of Manageability Sections in PCE Working Group Drafts", draft-ietf-pce-manageability-requirements-01 (work in progress), March 2007.

[I-D.ietf-rpsec-bgpsecrec]
Christian, B. and T. Tauber, "BGP Security Requirements", draft-ietf-rpsec-bgpsecrec-07 (work in progress), February 2007.

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

Stephan, E. and K. Koushik, "PCE communication protocol(PCEP) Management Information Base", draft-kkoushik-pce-pcep-mib-00 (work in progress), February 2007.

[I-D.vasseur-pce-monitoring] Vasseur, J., "A set of monitoring tools for Path Computation Element based Architecture", draft-vasseur-pce-monitoring-03 (work in progress), May 2007.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.





[RFC3036] Andersson, L., Doolan, P., Feldman, N., Fredette, A., and B. Thomas, "LDP Specification", RFC 3036, January 2001.

[RFC3785] Le Faucheur, F., Uppili, R., Vedrenne, A., Merckx, P., and T. Telkamp, "Use of Interior Gateway Protocol (IGP) Metric as a second MPLS Traffic Engineering (TE) Metric", BCP 87, RFC 3785, May 2004.

[RFC4022] Raghunarayan, R., "Management Information Base for the Transmission Control Protocol (TCP)", RFC 4022, March 2005.

[RFC4101] Rescorla, E. and IAB, "Writing Protocol Models", RFC 4101, June 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4420] Farrel, A., Papadimitriou, D., Vasseur, J., and A. Ayyangar, "Encoding of Attributes for Multiprotocol Label Switching (MPLS) Label Switched Path (LSP) Establishment Using Resource ReserVation Protocol-Traffic Engineering (RSVP-TE)", RFC 4420, February 2006.

[RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation

Element (PCE)-Based Architecture", RFC 4655, August 2006.

[RFC4657] Ash, J. and J. Le Roux, "Path Computation Element (PCE) Communication Protocol Generic Requirements", RFC 4657, September 2006.

[RFC4674] Le Roux, J., "Requirements for Path Computation Element (PCE) Discovery", RFC 4674, October 2006.

[RFC4927] Le Roux, J., "Path Computation Element Communication Protocol (PCECP) Specific Requirements for Inter-Area MPLS and GMPLS Traffic Engineering", RFC 4927, June 2007.

## Appendix A. PCEP Variables

PCEP defines the following configurable variables:

KeepAlive timer: minimum period of time between the sending of PCEP messages (Keepalive, PCReq, PCRep, PCNtf) to a PCEP peer. A suggested value for the Keepalive timer is 30 seconds.

DeadTimer: period of timer after the expiration of which a PCEP peer





declared the session down if no PCEP message has been received.

SyncTimer: the SYNC timer is used in the case of synchronized path computation request using the SVEC object defined in Section 7.12.3. 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 SYNC timer 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. A RECOMMENDED value for the SYNC timer is 60 seconds.

## Authors' Addresses

JP Vasseur (editor)
Cisco Systems
1414 Massachusetts Avenue
Boxborough, MA 01719

USA

Email: jpv@cisco.com

JL Le Roux (editor)
France Telecom
2, Avenue Pierre-Marzin
Lannion,  22307
FRANCE

Email: jeanlouis.leroux@orange-ftgroup.com





Full Copyright Statement

THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

## Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

## Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).