           Path Computation Element (PCE) Communication Protocol (PCEP)
                         draft-ietf-pce-pcep-18.txt

Status of this Memo

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. 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.

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. Introduction

[RFC4655] describes the motivations and architecture for a Path
Compuation Element (PCE) based model for the computation of
Multiprotocol Label Switching (MPLS) and Generalized (GMPLS) Traffic
Engineering Label Swtich Paths (TE LSPs). The model allows for the
separation of PCE from Path Computation Client (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 a 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 PCC and a 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.

2. 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 at least two
different domains where a domain can 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 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.

   The message formats in this document are specified using Backus Naur
   Format (BNF) encoding as specified in [I-D.farrel-rtg-common-bnf].

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 has the required information
   (usually including network topology and resource information) so as
   to perform the computation of a path for a TE LSP. Such information
   can be gathered by routing protocols or by some other means. The way
   in which the information is gathered is out of the scope of this
   document.

   Similarly, no assumption is made about 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
   reference, [RFC4674] defines a list of requirements for dynamic PCE
   discovery and IGP-based solutions for such PCE discovery are
   specified in [RFC5088] and [RFC5089].

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 path for MPLS
   and GMPLS TE LSPs 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 LSPs to a PCE and the PCE may
   reply with a set of computed paths if one or more paths can be found
   that satisfies the set of constraints.

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 paths if the request can be satisfied, or a negative reply
   if not. The negative reply may 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 PCEs may be either statically configured on a
   PCC or dynamically discovered. The mechanisms used to discover one

or more PCEs 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.

Each PCEP message is regarded as a single transmission unit and parts
of messages MUST NOT be interleaved. So, for example, a PCC sending
a PCReq and wishing to close the session, must complete sending the
request message before starting to send a Close message.

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 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.

Only one PCEP session can exist between a pair of PCEP peers at any
one time. Only one TCP connection on the PCEP port can exist between
a pair of PCEP peers at any one time.

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

```
             +-+-+                   +-+-+
             |PCC|                   |PCE|
             +-+-+                   +-+-+
               |                       |
               | Open msg              |
               |--------               |
               |        \    Open msg  |
               |         \   ---------|
               |          \/           |
               |          /\           |
               |         /   -------->|
               |        /              |
               |<------      Keepalive|
               |              --------|
               |Keepalive   /          |
               |--------   /           |
               |        \/             |
               |        /\             |
               |<------    ---------->|
               |                       |
```
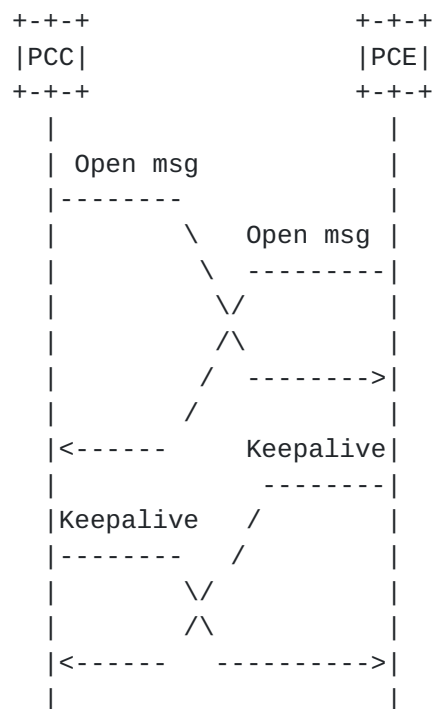
Figure 1: PCEP Initialization phase (initiated by a PCC)

(Note that once the PCEP session is established, the exchange of
Keepalive messages is optional)

4.2.2. Session Keepalive

Once a session has been established, a PCE or PCC may want to know
that its PCEP peer is still available for use.

It can rely on TCP for this information, but it is possible that the
remote PCEP function has failed without disturbing the TCP
connection. It is also possible to rely on the mechanisms built into
the TCP implementations, but these might not provide sufficiently
timely notifications of failures. Lastly, a PCC could wait until it
has a path computation request to send and use its failed
transmission or the failure to receive a response as evidence that
the session has failed, but this is clearly inefficient.

In order to handle this situation, PCEP includes a keepalive
mechanism based on a Keepalive timer, a Dead timer, and a Keepalive
message.

Each end of a PCEP session runs a Keepalive timer. It restarts the
timer every time it sends a message on the session. When the timer

expires, it sends a Keepalive message. Other traffic may serve as
Keepalive (see Section 6.3).

The ends of the PCEP session also run Dead timers, and they restart
them whenever a message is received on the session. If one end of
the session receives no message before the Dead timer expires, it
declares the session dead.

Note that this means that the Keepalive message is unresponded and
does not form part of a two-way keepalive handshake as used in some
protocols. Also note that the mechanism is designed to reduce to a
minimum the amount of keepalive traffic on the session.

The keepalive traffic on the session may be unbalanced according to
the requirements of the session ends. Each end of the session can
specify (on an Open message) the Keepalive timer that it will use
(i.e., how often it will transmit a Keepalive message if there is no
other traffic) and a Dead timer that it recommends its peer to use
(i.e., how long the peer should wait before declaring the session
dead if it receives no traffic). The session ends may use different
Keepalive timer values.

The minimum value of the Keepalive timer is 1 second, and it is
specified in units of 1 second. The recommended default value is 30
seconds. The timer may be disabled by setting it to zero.

The recommended default for the Dead timer is 4 times the value of
the Keepalive timer used by the remote peer. This means that there
is never any risk of congesting TCP with excessive Keepalive
messages.

4.2.3. Path Computation Request Sent By a PCC to a PCE

```
               +-+-+                    +-+-+
               |PCC|                    |PCE|
               +-+-+                    +-+-+
 1)Path computation |                     |
 event              |                     |
 2)PCE Selection    |                     |
 3)Path computation |---- PCReq message--->|
 request sent to    |                     |
 the selected PCE   |                     |
```
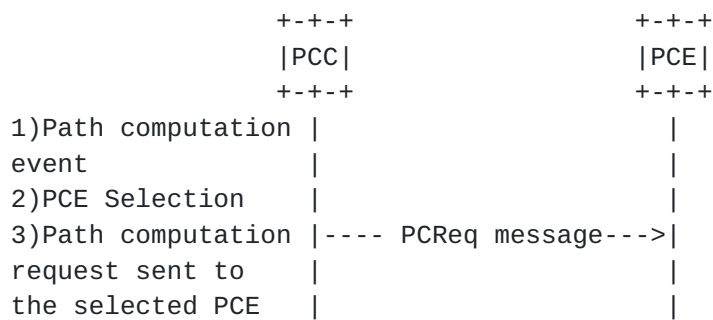
            Figure 2: Path Computation request

   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 paths, the PCC first selects one or more PCE. 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 the PCE 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.

   Note that multiple path computation requests may be outstanding from
   one PCC to a PCE at any time.

   Details about the PCReq message can be found in Section 6.4

4.2.4. Path Computation Reply Sent By The PCE to a PCC

```
              +-+-+                    +-+-+
              |PCC|                    |PCE|
              +-+-+                    +-+-+
                |                        |
                |---- PCReq message--->|
                |                        |1) Path computation
                |                        |request received
                |                        |
                |                        |2)Path successfully
                |                        |computed
                |                        |
                |                        |3) Computed paths
                |                        |sent to the PCC
                |                        |
                |<--- PCRep message ---|
                |     (Positive reply)  |
```
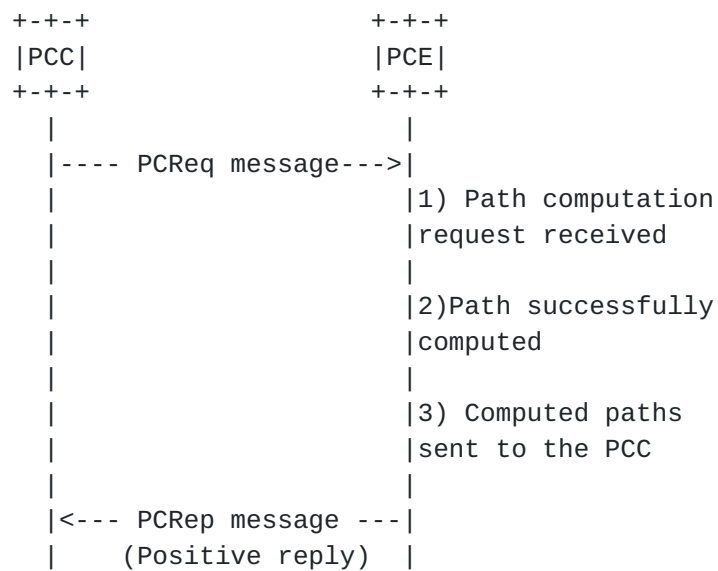
    Figure 3a: Path Computation Request With Successful
              Path Computation

```
              +-+-+                    +-+-+
              |PCC|                    |PCE|
              +-+-+                    +-+-+
                |                        |
                |                        |
                |---- PCReq message--->|
                |                        |1) Path computation
                |                        |request received
                |                        |
                |                        |2) No Path found that
                |                        |satisfies the request
                |                        |
                |                        |3) Negative reply sent to
                |                        |the PCC (optionally with
                |                        |various additional
                |                        |information)
                |<--- PCRep message ---|
                |     (Negative reply)  |
```
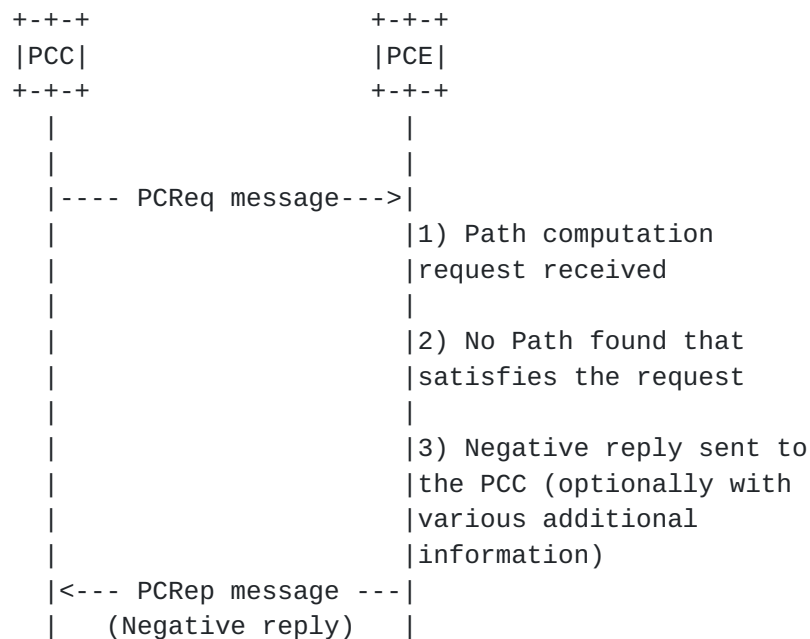
    Figure 3b: Path Computation Request With Unsuccessful
              Path Computation

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

o 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 paths 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).

o 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.

4.2.5. Notification

There are several circumstances in which a PCE may want to notify a
PCC of a specific event. For example, suppose that the PCE suddenly
gets overloaded, 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 its path computation requests 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 requests.

```
                    +-+-+                    +-+-+
                    |PCC|                    |PCE|
                    +-+-+                    +-+-+
   1)Path computation |                        |
   event              |                        |
   2)PCE Selection    |                        |
   3)Path computation |---- PCReq message--->|
   request X sent to  |                        |4) Path computation
   the selected PCE   |                        |request queued
                      |                        |
                      |                        |
   5) Path computation|                        |
   request X cancelled|                        |
                      |---- PCNtf message -->|
                      |                        |6) Path computation
                      |                        |request X cancelled
```
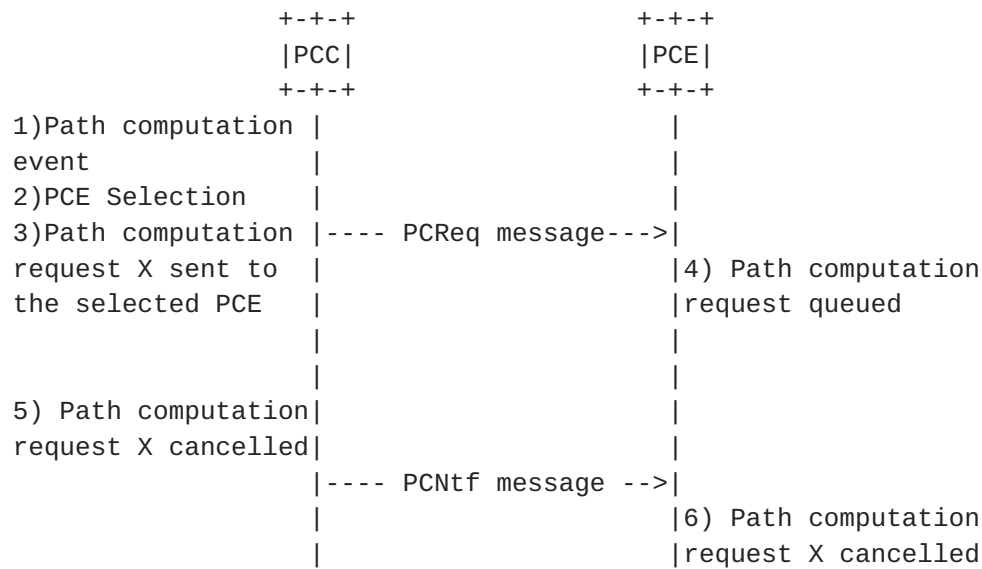
   Figure 4: Example of PCC Notification (Cancellation
   Notification)
   Sent To a PCE

```
                    +-+-+                    +-+-+
                    |PCC|                    |PCE|
                    +-+-+                    +-+-+
   1)Path computation |                        |
   event              |                        |
   2)PCE Selection    |                        |
   3)Path computation |---- PCReq message--->|
   request X sent to  |                        |4) Path computation
   the selected PCE   |                        |request queued
                      |                        |
                      |                        |
                      |                        |5) PCE gets overloaded
                      |                        |
                      |                        |
                      |                        |6) Path computation
                      |                        |request X cancelled
                      |                        |
                      |<--- PCNtf message----|
```
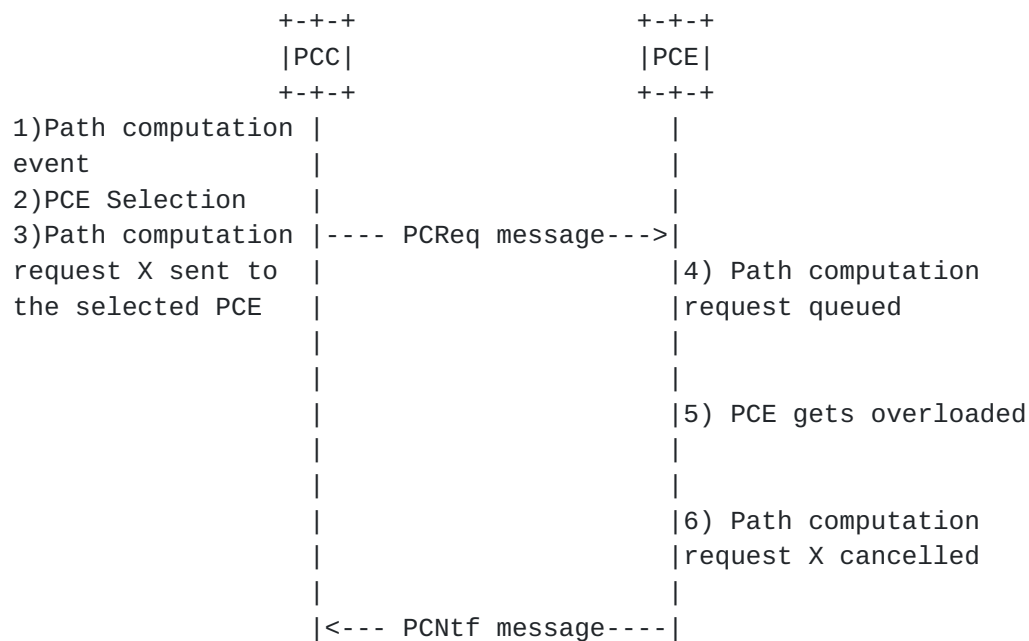
   Figure 5: Example of PCE Notification (Cancellation
   Notification) Sent To a PCC

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

4.2.6. **Error**

   The PCEP Error message (also referred to as a PCErr message) is sent
   in several situations: when a protocol error condition is met or when
   the request is not compliant with the PCEP specification (e.g.,
   capability not supported, reception of a message with a mandatory
   missing object, policy violation, unexpected message, unknown request
   reference, ...).

```
                    +-+-+                   +-+-+
                    |PCC|                   |PCE|
                    +-+-+                   +-+-+
   1)Path computation |                       |
   event              |                       |
   2)PCE Selection    |                       |
   3)Path computation |---- PCReq message--->|
   request X sent to  |                       |4) Reception of a
   the selected PCE   |                       |malformed object
                      |                       |
                      |                       |5) Request discarded
                      |                       |
                      |<-- PCErr message  ---|
                      |                       |
```
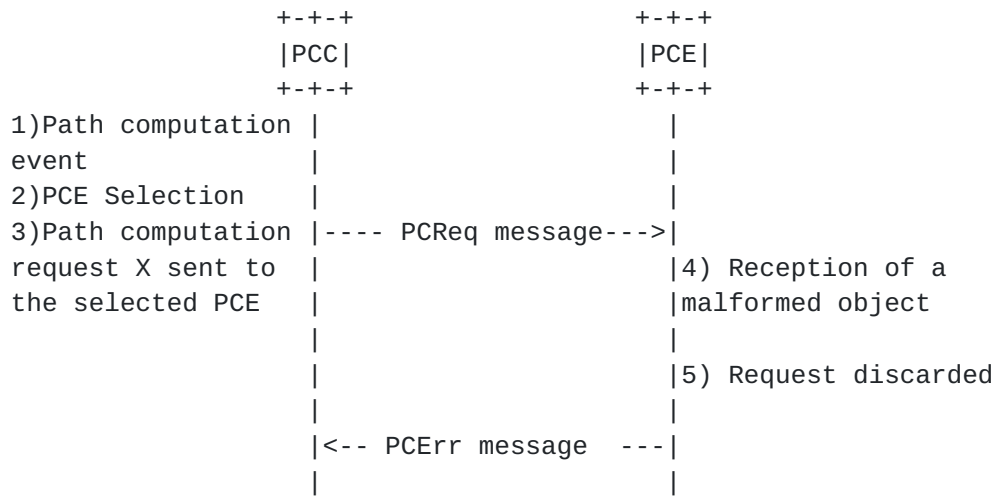
   Figure 6: Example of Error message Sent By a PCE To a PCC
   In Reply To The Reception Of a Malformed Object

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

4.2.7. **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.

4.2.8. **Intermitent versus Permanent PCEP Session**

   An implementation may decide to keep the PCEP session alive (and thus
   the corresponding TCP connection) 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 a PCEP session for each
   PCEP request (for instance when sending a path computation request is
   a rare event).

5. **Transport Protocol**

   PCEP operates over TCP using a registered TCP port (to be assigned by
   IANA). This allows the requirements of reliable messaging and flow
   control to be met without further protocol work. All PCEP message
   MUST be sent using the registered TCP port for the source and
   destination TCP port.

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 7.15).
   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 7.2). When this flag is set in an
   object in a PCReq, the PCE MUST take the information carried in the
   object into account during the path computation. For example, the
   METRIC object defined in Section 7.8 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 the constraint is taken into
   account. In this case, if the 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) (see [I-D.farrel-rtg-common-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

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Ver |  Flags  | Message-Type |        Message-Length          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
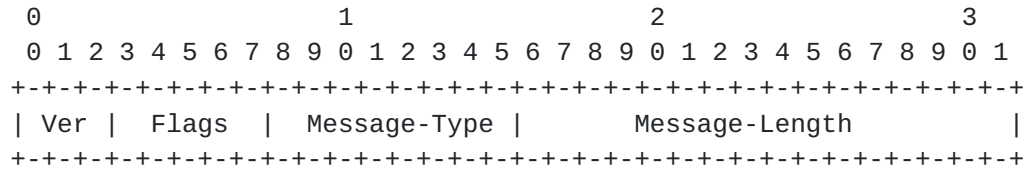
Figure 7: PCEP Message Common Header

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. They MUST be set to zero on transmission and
MUST be ignored on receipt.

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 Appendix A.

Any message received prior to an Open message MUST trigger a protocol
error condition causing a PCErr message to be sent with Error-Type

'PCEP session establishment failure' and Error-Value 'reception of an
invalid Open message or a non Open message' and the PCEP session
establishment attempt MUST be terminated by closing the TCP
connection.

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 7.3).

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 Appendix A 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 Appendix A 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 characteristics 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 7.15. Successive
retries are permitted but an implementation SHOULD make use of an
exponential back-off session establishment retry procedure.

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

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

A PCEP implementation is free to process received requests in any
order. For example, the requests may be processed in the order they
are received, re-ordered and assigned priority according to local
policy, re-ordered according to the priority encoded in the RP Object
(Section 7.4.1), or processed in parallel.

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 Keepalive message is also
used in response to an Open message to acknowledge that an Open
message has been received and that the PCEP session characteristics
are acceptable. 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. Deployments
SHOULD consider carefully the impact of using low values for the
Keepalive timer as these might not give rise to the expected results
in periods of temporary network instability.

Keepalive messages are sent at the frequency specified in the OPEN
object carried within an Open message according to the rules
specified in Section 7.3. 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) was
sent.

Note that sending Keepalive messages to keep the session alive is
optional and PCEP peers may decide to not send Keepalive messages
once the PCEP session is established in which case the peer that does
not receive Keepalive messages does not expect to receive them and
MUST NOT declare the session as inactive.

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 to request a path
computation. A PCReq message may carry more than one path
computation request. 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 or both 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>]
                [<metric-list>]
                [<RRO>[<BANDWIDTH>]]
                [<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 detail in Section 7.7.

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 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. A PCRep message may contain positive and negative
   replies.

   A PCRep message may contain a set of computed paths corresponding to
   either a single path computation request with load-balancing (see
   Section 7.16) 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 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 7.4 for the definition of the RP object).

   If the path computation request can be satisfied (the PCE finds a set
   of paths that satisfy the set of constraints), the set of computed
   paths specified by means of ERO objects is inserted in the PCRep
   message. The ERO is defined in Section 7.9. The situation where
   multiple computed paths are provided in a PCRep message is discussed
   in detail in Section 7.13. Furthermore, when a PCC requests the
   computation of a set of paths for a total amount of bandwidth 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 7.16.

   If the path computation request cannot be satisfied, the PCRep
   message MUST include a NO-PATH object. The NO-PATH object (described
   in Section 7.5) may also contain 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 be sent either by a PCE to a PCC, or by a PCC to a PCE, 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 7.14. The PCNtf message MAY also contain RP objects (see
   Section 7.4 when the notification refers to particular path
   computation requests.

   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
in several situations: when a protocol error condition is met or when
the request is not compliant with the PCEP specification (e.g.,
reception of a malformed message, reception of a message with a
mandatory missing object, policy violation, unexpected message,
unknown request reference, ...). The Message-Type field of the PCEP
common header is set to 6 (To be confirmed by IANA).

The PCErr message is sent by a PCC or a PCE in response to a request
or in an unsolicited manner. If the PCErr message is sent in
response to a request, the PCErr message MUST include the set of RP
objects related to the pending path computation requests that
triggered the error condition. In the later case (unsolicited), no
RP object is inserted in the PCErr message. For example, 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 7.15.

The format of a PCErr message is as follows:
```
<PCErr Message> ::= <Common Header>
                    ( <error-object-list> [<Open>] ) | <error>
                    [<error-list>]

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

The procedure upon the receipt of a PCErr message is defined in
Section 7.15.

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). If more than one CLOSE object is present, the first
   MUST be processed and subsequent objects ignored.

   Upon the receipt of a valid Close message, the receiving PCEP peer
   MUST cancel all pending requests, it MUST close the TCP connection
   and MUST NOT send any further PCEP messages on the PCEP session.

6.9. **Reception of Unknown Messages**

   A PCEP implementation that receives an unrecognized PCEP message MUST
   send a PCErr message with Error-value=2 (capability not supported).

   If a PCC/PCE receives unrecognized messages at a rate equal of
   greater than MAX-UNKNOWN-MESSAGES unknown message requests per
   minute, the PCC/PCE MUST send a PCEP CLOSE message with close
   value="Reception of an unacceptable number of unknown PCEP message".
   A RECOMMENDED value for MAX-UNKOWN-MESSAGES is 5. The PCC/PCE MUST
   close the TCP session and MUST NOT send any further PCEP messages on
   the PCEP session.

7. **Object Formats**

   PCEP objects have a common format. They begin with a common object
   header (see Section 7.2). This is followed by object-specific fields
   defined for each different object. The object may also include one
   or more type-length-value (TLV) encoded data sets. Each TLV has the
   same structure as described in Section 7.1.

7.1. **PCE TLV Format**

   A PCEP object may include a set of one or more optional TLVs.

   All PCEP TLVs have the following format:

Type: 2 bytes
Length: 2 bytes
Value: variable
A PCEP object TLV is comprised of 2 bytes for the type, 2 bytes
specifying the TLV length, and a value field.

The Length field defines the length of the value portion in bytes.
The TLV is padded to 4-bytes alignment; padding is not included in
the Length field (so a three byte value would have a length of three,
but the total size of the TLV would be eight bytes).

Unrecognized TLVs MUST be ignored.

IANA management of the PCEP Object TLV type identifier codespace is
described in Section 9.

7.2. 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:
```
 0              1               2               3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Object-Class  |   OT  |Res|P|I|   Object Length (bytes)       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                         (Object body)                      //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
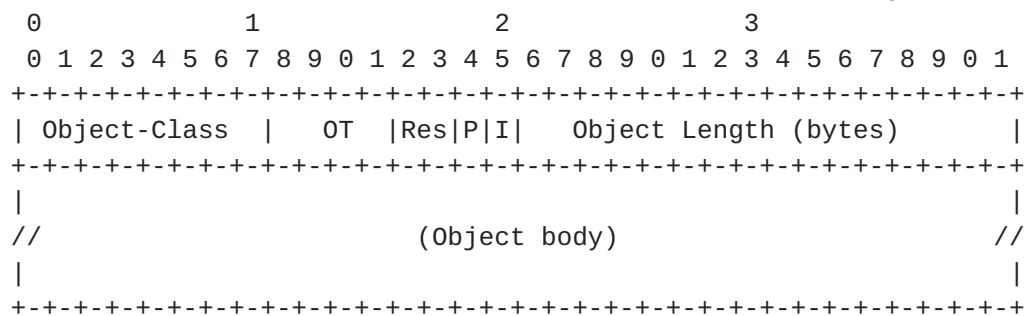
Figure 8: PCEP common object header

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.

o 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.

o 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 has been set in the corresponding
   PCReq 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" along with
the corresponding RP object. Note that if a PCReq includes multiple
requests, only requests for which an object with the P flag set is
unknown/unrecognized MUST be rejected.

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.

7.3. 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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Ver |  Flags  |   Keepalive   |   DeadTimer   |      SID      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                        Optional TLVs                        //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
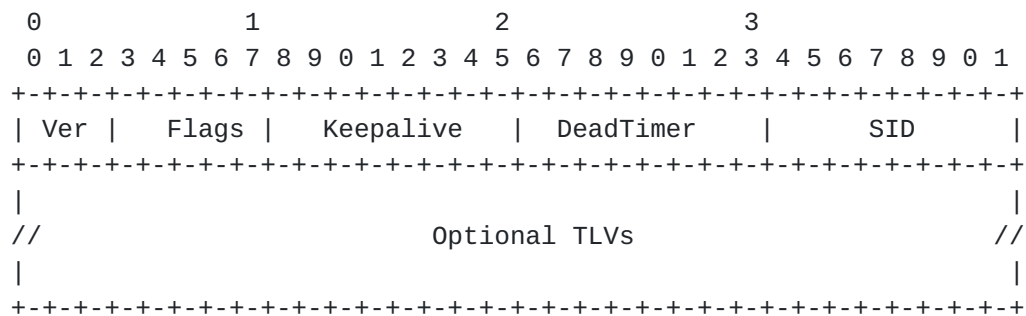
                     Figure 9: OPEN Object format

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

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

Keepalive (8 bits): maximum period of time (in seconds) between two
consecutive PCEP messages sent by the sender of this message. The
minimum value for the Keepalive is 1 second. When set to 0, once the
session is established, no further Keepalive messages are 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 SHOULD be set to 0 and MUST be ignored 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=40 seconds. This means that A sends Keepalive messages (or
any other PCEP message) to B every 10 seconds and B can declare the
PCEP session with A down if no PCEP message has been received from A
within any 40 second period.

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 used for logging and
troubleshooting purposes. Each increment SHOULD have a value of 1
and may cause a wrap back to zero.

The SID is used to disambiguate instances of sessions to the same
peer. A PCEP implementation could use a single source of SIDs across
all peers, or one source for each peer. The former might constrain
the implementation to only 256 concurrent sessions. The latter
potentially requires more states. There is one SID number in each
direction.

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.

7.4. 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 RP object is used to specify various characteristics
of the path computation request.

The P flag of the RP object MUST be set in PCReq and PCReq messages
and MUST be cleared in PCNtf and PCErr messages. If the RP objet is
received with the P flag set incorrectely according to the rules
states above, the receiving peer MUST send a PCErr message with
Error-type=10 and Error-value=1. The corresponding path computation
request MUST be cancelled by the PCE without further notification.

7.4.1. 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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Flags                     |O|B|R| Pri |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Request-ID-number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                      Optional TLVs                          //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
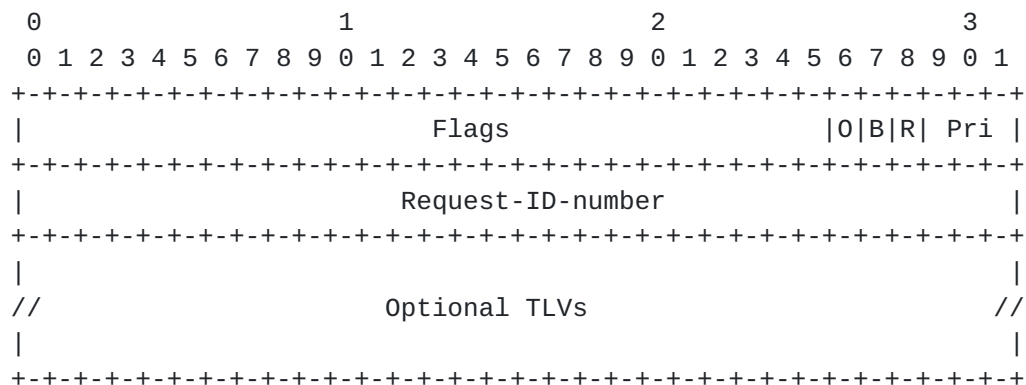
Figure 10: RP object body format

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

Flags (24 bits)

The following flags are currently defined:

o 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
   that the PCC set the priority field to 0 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 PCCs.
   The ability of a PCE to support request prioritization MAY be
   dynamically discovered by the PCCs 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.

o R (Reoptimization - 1 bit): when set, the requesting PCC specifies
   that the PCReq message relates to the reoptimization of an
   existing TE LSP. For all TE LSPs except 0-bandwidth LSPs, when
   the R bit is set, an RRO (see Section 7.10) MUST be included in
   the PCReq message to show the path of the existing TE LSP. Also,
   for all TE LSPs except 0-bandwidth LSPs, then the R bit is set,
   the existing bandwidth of the TE LSP to be reoptimized MUST be
   supplied in a BANDWIDTH object (see Section 7.7). This BANDWIDTH
   object is in addition to the instance of that object used to
   describe the desired bandwidth of the reoptimized LSP. For
   0-bandwidth LSPs, the RRO and BANDWIDTH objects that report the
   characteristics of the existing TE LSP are optional.

o 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, TE Links, and resource
   requirements (e.g., latency and jitter) in each direction. When
   cleared, the TE LSP is unidirectional.

o 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. They MUST be set to zero
on transmission and MUST be ignored on receipt.

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
is used for disambiguation between pending requests and thus it MUST
be changed (such as by incrementing it) each time a new request is
sent to the PCE, and may wrap.

The value 0x00000000 is considered as invalid.

If no path computation reply is received from the PCE (e.g. request
dropped by the PCE because of memory overflow), and the PCC wishes to
resend its request, the same Request-ID-number MUST be used. Upon
receiving a path computation request from a PCC with the same
Request-ID-number the PCE SHOULD treat the request as a new request

but an implementation MAY choose to cach path computation replies in
order to quickly handle restranmission without having to handle twice
a path computation request should have the first request been dropped
or lost. Upon receiving a path computation reply from a PCE with the
same Request-ID-number the PCC SHOULD silently discard the path
computation reply.

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.

7.4.2. Handling of the RP Object

If a PCReq message is received that does not contain 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 paths (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 requires that the PCE either be
stateful (keep track of the previously computed path with the
associated list of strict hops), or have the ability to retrieve the
complete required path segment. Alternatively the PCC MUST inform
the PCE of the working path with the 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 a PCC/PCE receives a PCRep/PCReq message that contains a RP object
referring to an unknown Request-ID-Number, the PCC/PCE MUST send a
PCErr message with Error-Type="Unknown request reference". This is
used for debugging purposes. If a PCC/PCE receives PCRep/PCReq at a

rate equal of greater than MAX-UNKOWN-REQUESTS unknown requests per
minute, the PCC/PCE MUST send a PCEP CLOSE message with close
value="Reception of an unacceptable number of unknown requests/
replies". A RECOMMENDED value for MAX-UNKOWN-REQUESTS is 5. The
PCC/PCE MUST close the TCP session and MUST NOT send any further PCEP
messages on the PCEP session.

The reception of a PCEP message that contains a RP object referring
to a Request-ID-number=0x00000000 MUST be treated similarly to an
unkown request.

7.5. 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.

There are several categories of issue 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 and used to
provide more information on the reasons that led to a negative reply.
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 objects 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 (in which case the value
differs from the value in the original request).

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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Nature Of Issue|C|          Flags          |     Reserved     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                      Optional TLVs                          //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
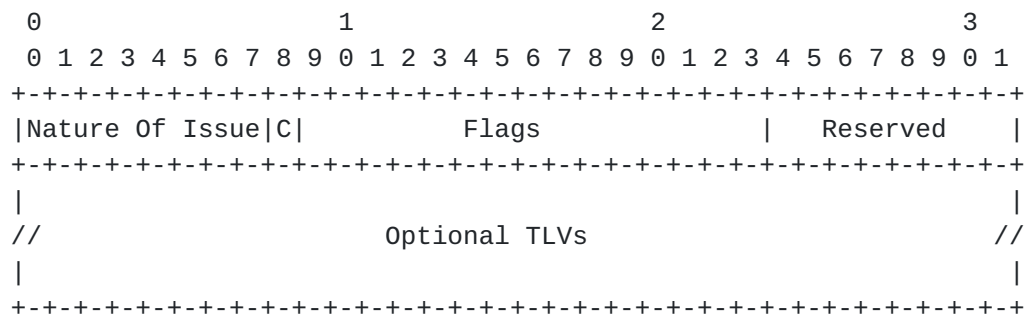
Figure 11: NO-PATH Object Format

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

The Nature Of Issue field value can be used by the PCC for various
purposes:

o Constraint adjustement before re-issuing a new path computation
   request,

o Explicit selection of a new PCE chain,

o Logging of the error type for futher action by the network
   admistrator

IANA management of the NI field codespace is described in Section 9.

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
failing constraints are specified. The C flag has no meaning and is
ignored unless the NI field is set to 0x00.

Unassigned bits are considered as reserved. They 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.

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
paths 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 compliant with the PCEP TLV format defined in
section 7.1 and is comprised of 2 bytes for the type, 2 bytes specifying
the TLV length (length of the value portion in bytes) followed by a fixed
length value field of 32-bit flags field.

TYPE: To be assigned by IANA (suggested value=1)
LENGTH: 4
VALUE: 32-bit 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:

o Bit number: 31 - PCE currently unavailable

o Bit number: 30 - Unknown destination

o Bit number: 29 - Unknown source

7.6. 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. The P flag of the END-POINT
object MUST be set. If the END-POINT objet is received with the P
flag cleared, the receiving peer MUST send a PCErr message with
Error-type=10 and Error-value=1. The corresponding path computation
request MUST be cancelled by the PCE without further notification.

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)

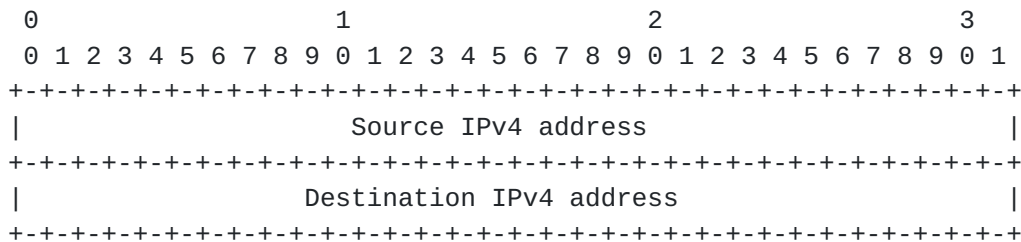The format of the END-POINTS object body for IPv4 (Object-Type=1) is
as follows:

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

        Figure 12: END-POINTS Object Body Format for IPv4

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

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                              |
|                 Source IPv6 address (16 bytes)               |
|                                                              |
|                                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                              |
|               Destination IPv6 address (16 bytes)            |
|                                                              |
|                                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
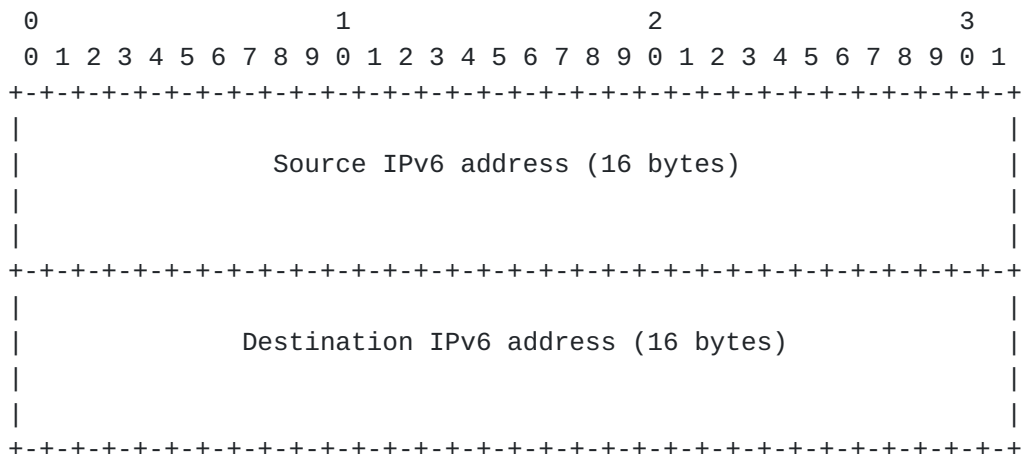
        Figure 13: END-POINTS Object Body Format for IPv6

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

If more than one END-POINTS object is present, the first MUST be
processed and subsequent objects ignored.

7.7. BANDWIDTH Object

The BANDWIDTH object is used to specify the requested bandwidth for a
TE LSP. The notion of bandwidth is similar to the one used for RSVP
signaling in [RFC2205], [RFC3209] and [RFC3473].

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 values 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 values are defined for the BANDWIDTH object:

o Requested bandwidth: BANDWIDTH Object-Type is to be assigned by
   IANA (recommended value=1)

o 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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Bandwidth                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
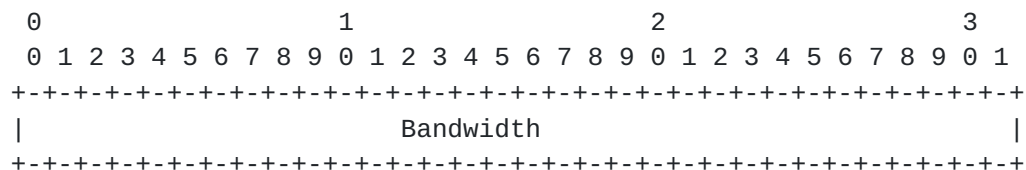
                 Figure 14: BANDWIDTH Object Body Format

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

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

7.8. METRIC Object

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

In a PCReq message, a PCC MAY insert one of more METRIC objects:

o 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.

   o 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 metrics are only meaningful if
   used consistently: for instance, if the delay of a computed path
   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 no constraints need be applied to
   any of the metrics.

   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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Reserved            |    Flags  |C|B|       T       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         metric-value                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
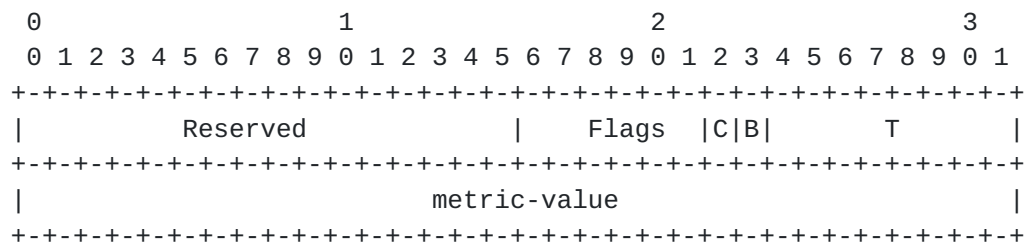
                    Figure 15: METRIC Object Body Format

   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:

o T=1: IGP metric

o T=2: TE metric

o T=3: Hop Counts

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

o B (Bound - 1 bit): When set in a PCReq message, the metric-value
   indicates a bound (a maximum) for the path metric that must not be
   exceeded for the PCC to consider the computed path as acceptable.
   The path metric must be less than or equal to the value specified
   in the Metric-value field. When the B flag is cleared, the
   metric-value field is not used to reflect a bound constraint.

o C (Computed Metric - 1 bit): When set in a PCReq message, this
   indicates that the PCE MUST provide the computed path metric value
   (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 (See [IEEE.754.1985]).

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.

The presence of two METRIC objects of the same type with a different
value of the B-Flag in a PCEReq message is allowed. Furthermore, it
is also allowed to insert in a PCReq message two METRIC objects with
different types that have both their B-Flag cleared: in this case, an
objective function must be used by the PCE to solve a multi-parameter
constraint problem.

A METRIC object used to indicate the metric to optimize during the
path computation MUST have the B-Flag cleared and the C-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 computed value of the metric 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
C-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 metric
if the C bit of the METRIC object was set in the corresponding path
computation request (the B-flag MUST be cleared). The C-flag has no
meaning in a PCRep message. 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
computed path metric (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:

o First METRIC Object with B=0, T=1, C=1, metric-value=0x0000

o 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 there is no policy that prevents the return of the computed
metric, 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.

7.9. Explicit Route Object

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

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

PCEP ERO sub-object types correspond to RSVP-TE 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)

7.10. Reported Route Object

   The RRO is exclusively carried within a PCReq message so as to report
   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-TE RRO sub-object already defined or that could be
   defined in the future for use in the RSVP-TE 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 RSVP-TE RRO.

   PCEP RRO sub-object types correspond to RSVP-TE 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)

7.11. 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 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 constraints that could not be
   satisfied). Most of the fields of the LSPA object are identical to
   the fields of the SESSION-ATTRIBUTE (C-Type = 7) 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. See section 4.7.4 of [RFC3209] for a
   detailed description of the use of resource affinities.

   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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Exclude-any                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Include-any                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Include-all                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Setup Prio   | Holding Prio |     Flags     |L|   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                       Optional TLVs                         //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
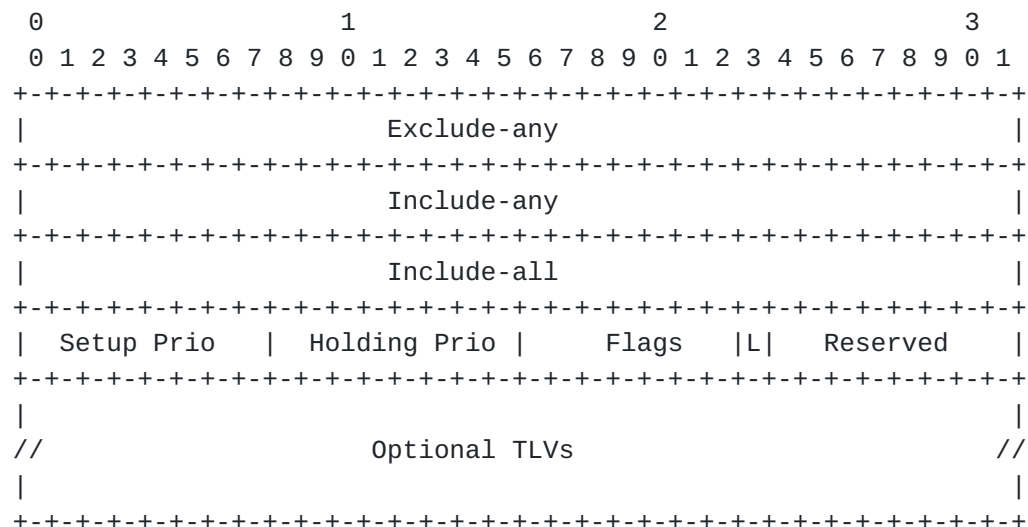
Figure 16: LSPA Object Body Format

Setup Prio (Setup Priority - 8 bits). The priority of the TE LSP
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 TE LSP
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 Optional TLVs may be defined in the future to carry
additional TE LSP attributes such as those defined in [RFC4420].

7.12. **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 cause de PCE to fail to find a
   path.

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

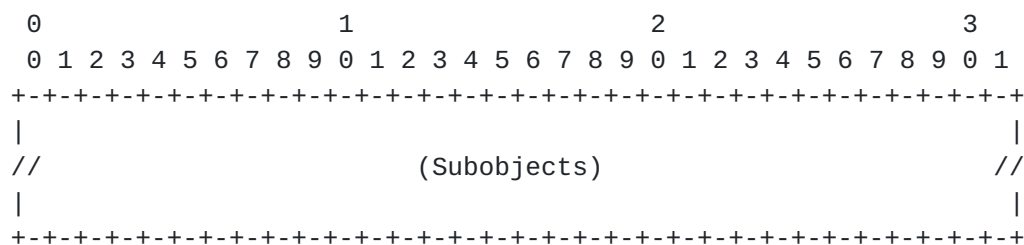   IRO Object-Type is to be assigned by IANA (recommended value=1)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                       (Subobjects)                          //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                     Figure 17: IRO Body Format

   Subobjects: The IRO is made of subobjects identical to the ones
   defined in [RFC3209], [RFC3473] and [RFC3477] where the IRO subobject
   type is identical to the subobject type defined in the related
   documents.

   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.

7.13. **SVEC Object**

7.13.1. **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 purposes. 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:

o Bundle of a set of independent and non-synchronized path
   computation requests,

o Bundle of a set of independent and synchronized path computation
   requests (SVEC object defined below required),

o Bundle of a set of dependent and synchronized path computation
   requests (SVEC object defined below required).

7.13.2. SVEC Object

Section 7.13.1 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)
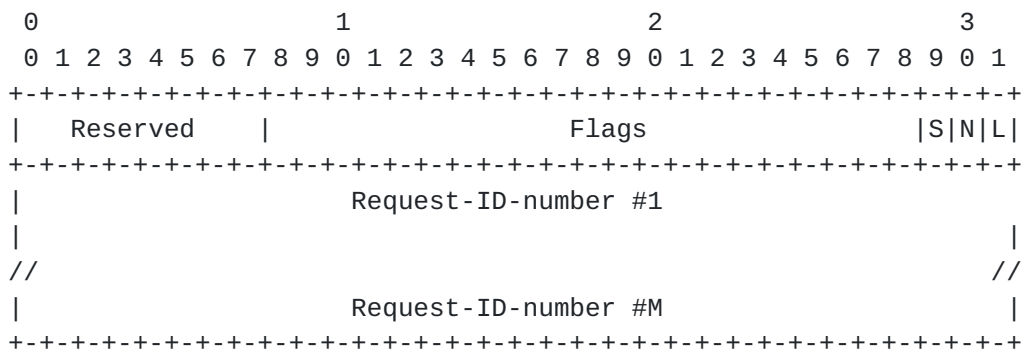
The format of the SVEC object body is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Reserved    |                  Flags                |S|N|L|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Request-ID-number #1                     |
|                                                             |
//                                                           //
|                     Request-ID-number #M                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

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.

o 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.

o 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.

7.13.3. 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 messages. 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). When a PCE receives a path computation
request that cannot be satisfied (for example, because the PCReq
message contains an object with the P bit set that is not supported),
the PCE sends a PCErr message for this request (see Section 7.2, the
PCE MUST cancel the whole set of related 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 that are processed as normal.

7.14. 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)

The format of the NOTIFICATION body object is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Reserved    |     Flags     |      NT       |      NV       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                      Optional TLVs                          //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
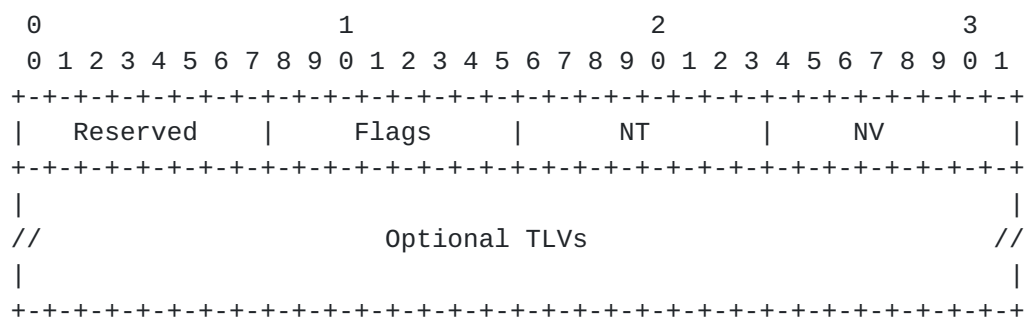
                Figure 19: NOTIFICATION Body Object Format

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 requests. A
      Notification-type=1, Notification-value=1 indicates that the
      PCC wants to inform a PCE of the cancellation of a set of
      pending requests. Such an 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 events 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 a 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 requests. A
      Notification-type=1, Notification-value=2 indicates that the
      PCE wants to inform a PCC of the cancellation of a set of
      pending 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 that the PCE is currently in an
  overloaded state. If no RP objects are included 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 objects 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 objects. If
  such notification is received by a PCE from a PCC, the PCE MUST
  silently ignore the notification and no errors should be
  generated.

* A PCE implementation SHOULD use a dual threshold mechanism used
  to determine whether it is in a congestion state with regards
  to specific resources monitoring (e.g. CPU, memory, ...). The
  use of such thresholds is to avoid oscillations between
  overloaded/non-overloaded state that may result in oscillations
  of request targets by the PCCs."

*

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 PCE should no longer be considered in congested
state.

The OVERLOADED-DURATION TLV is compliant with the PCEP TLV format
defined in section 7.1 and is comprised of 2 bytes for the type,
2 bytes specifying the TLV length (length of the value portion in bytes)
followed by a fix length value field of 32-bits flags field.

TYPE: To be assigned by IANA (suggested value=2)
LENGTH: 4
VALUE: 32-bits flags field indicates the estimated PCE congestion
duration in seconds.

* Notification-value=2: A Notification-type=2, Notification-
  value=2 indicates that the PCE is no longer in overloaded state
  and is available to process new path computation requests. An
  implementation SHOULD 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.

When a PCC receives an overload indication from a PCE it should
consider the impact on the entire network. It must be remembered
that other PCCs may also receive the notification and so many path
computation requests could be redirected to other PCEs. This may, in
turn, cause further overloading at PCEs in the network. Therefore,
an application at a PCC receiving an overload notification should
consider applying some form of back-off (e.g. exponential) to the
rate at which it generates path computation requests into the
network. This is especially the case as the number of PCEs reporting
overload grows.

7.15. 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)

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

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Reserved    |     Flags     |  Error-Type   |  Error-Value  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                      Optional TLVs                          //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
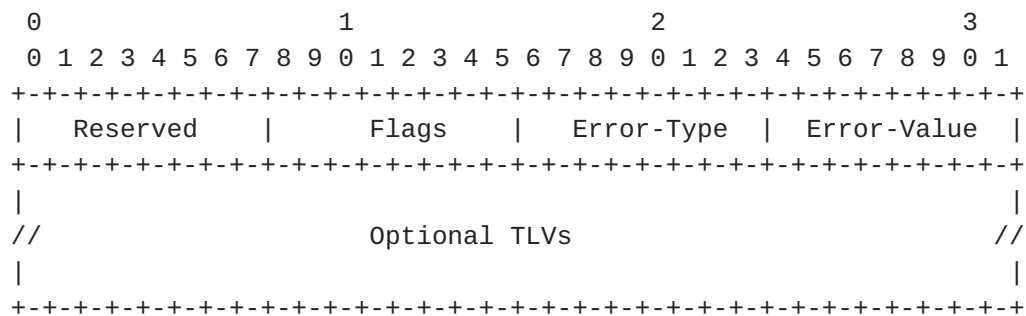
                Figure 20: PCEP-ERROR Object Body Format

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 an invalid Open message or
                                  a non Open 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
                                   bandwidth 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
    10   Reception of an invalid object
                    Error-value=1: reception of an object with P flag not set
                    although the P-flag must be set according to this
                    specification.

    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 Appendix A for details).

If one or more PCEP session characteristics 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 PCE 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 TLVs 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 7.13.3, 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 TLVs (defined below) that identifies the
missing requests.

The REQ-MISSING TLV is compliant with the PCEP TLV format defined
in section 7.1 and is comprised of 2 bytes for the type, 2 bytes
specifying the TLV length (length of the value portion in bytes)
followed by a fix length value field of 4 bytes.

TYPE: To be assigned by IANA (suggested value=3)
LENGTH: 4
VALUE: 4 bytes that indicates the 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.

7.16. 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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Reserved            |     flags     |    Max-LSP     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Min-Bandwidth                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
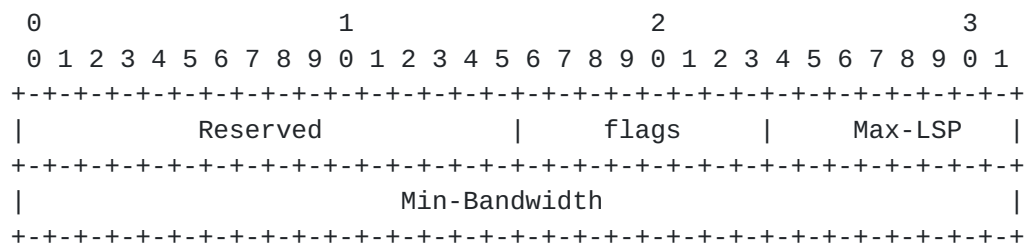
Figure 21: LOAD-BALANCING Object Body Format

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 (see [IEEE.754.1985]), 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 LSPs 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.

[7.17](7.17). 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 objects 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:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Reserved            |     Flags     |    Reason     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
//                       Optional TLVs                         //
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
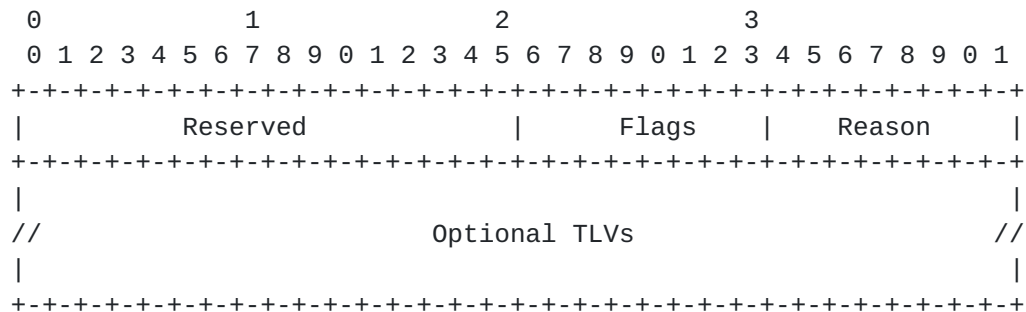
                    Figure 22: CLOSE Object Format

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

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

Reason (8 bits): specifies the reason for closing the PCEP session.
The setting of this field is optional. IANA is requested to manage
the codespace of the Reason field. The following values are
currently defined (To be confirmed by IANA).

```
Reasons
 Value        Meaning
   1          No explanation provided
   2          DeadTimer expired
   3          Reception of a malformed PCEP message
   4          Reception of an unacceptable number of
              unknown requests/replies
   5          Reception of an unacceptable number of
              unrecognized PCEP messages
```

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 the implementation:

   o The local Keepalive and DeadTimer (i.e., parameters sent by the
      PCEP peer in an Open message),

   o The maximum acceptable remote Keepalive and DeadTimer (i.e.,
      parameters received from a peer in an Open message),

   o Negotiation enabled or disabled,

   o If negotiation is allowed, the minimum acceptable Keepalive and
      Deadtimer timers received from a PCEP peer,

   o The SyncTimer,

   o The maximum number of sessions that can be setup,

   o Request timer: amount of time a PCC waits for a reply before
      resending its path computation requests (potentially to an
      alternate PCE).

   o The MAX-UNKNOWN-REQUESTS

   o The MAX-UNKNOWN-MESSAGES

   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:

   o The IP address of the PCEP peer,

   o The PCEP speaker role: PCC, PCE or both,

   o Whether the PCEP speaker should initiate the PCEP session or wait
      for initiation by the peer,

o The PCEP session parameters, as listed above, if they differ from
   the default parameters,

o 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:

o PCEP client configuration and status,

o PCEP peer configuration and information,

o PCEP session configuration and information,

o Notifications to indicate PCEP session changes.

8.3. Liveness Detection and Monitoring

PCEP includes a keepalive mechanism to check the liveliness of a PCEP
peer and a notification procedure allowing a PCE to advertise its
overload 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.ietf-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:

o Response time (minimum, average and maximum), on a per PCE Peer
   basis,

o PCEP Session failures,

o Amount of time the session has been in active state,

   o Number of corrupted messages,

   o Number of failed computations,

   o 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 Components

   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 ([RFC5088], [RFC5089]) 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 a limit to be placed on the number of
   session that can be set up on a PCEP speaker, and MAY allow a limit
   to be placed on 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

   IANA assigns values to the PCEP protocol parameters (messages,
   objects, TLVs).

   IANA is requested to establish a new top-level registry to contain
   all PCEP codepoints and sub-registries.

   The allocation policy for each new registry is by IETF Consensus: new
   values are assigned through the IETF consensus process (see
   [RFC5226]). Specifically, new assignments are made via RFCs approved
   by the IESG. Typically, the IESG will seek input on prospective
   assignments from appropriate persons (e.g., a relevant Working Group
   if one exists).

9.1. TCP Port

   PCEP will use a registered TCP port to be assigned by IANA.

9.2. PCEP Messages

   IANA is requested to create a registry for PCEP messages. Each PCEP
   message has a message type value.

   Value   Meaning   Reference
     1   Open   This document
     2   Keepalive   This document
     3   Path Computation Request   This document
     4   Path Computation Reply   This document
     5   Notification   This document
     6   Error   This document
     7   Close   This document

9.3. PCEP Object

   IANA is requested to create a registry for PCEP objects. Each PCEP
   object has an Object-Class and an Object-Type.

   Object-Class Value   Name   Reference

         1   OPEN   This document
                      Object-Type
                          1

         2   RP   This document
                      Object-Type
                          1

         3   NO-PATH   This document
                      Object-Type
                          1

         4   END-POINTS   This document
                      Object-Type
                          1: IPv4 addresses
                          2: IPv6 addresses

         5   BANDWIDTH   This document
                      Object-Type
                          1: Requested bandwidth
                          2: Bandwidth of an existing TE LSP
                          for which a reoptimization is performed.

```
    6               METRIC                        This document
                    Object-Type
                         1

    7               ERO                           This document
                    Object-Type
                         1

    8               RRO                           This document
                    Object-Type
                         1

    9               LSPA                          This document
                    Object-Type
                         1

   10               IRO                           This document
                    Object-Type
                         1

   11               SVEC                          This document
                    Object-Type
                         1

   12               NOTIFICATION                  This document
                    Object-Type
                         1

   13               PCEP-ERROR                    This document
                    Object-Type
                         1

   14               LOAD-BALANCING                This document
                    Object-Type
                         1

   15               CLOSE                         This document
                    Object-Type
                         1
```

9.4. RP Object

   New bit numbers may be allocated only by an IETF Consensus action.
   Each bit should be tracked with the following qualities:

   o Bit number

   o Capability Description

   o Defining RFC

   Several bits are defined in this document. The following values have
   been assigned:

   Codespace of the Flag field (RP Object)
     Bit   Description   Reference

     0-2   Priority   This document
      3   Reoptimization   This document
      4   Bi-directional   This document
      5   Strict/Loose   This document

9.5. Notification Object

   IANA is requested to create a registry for the Notification-type and
   Notification-value of the Notification Object and manage the code
   space.

Notification-type Name   Reference
      1   Pending Request cancelled   This document
                    Notification-value
                      1: PCC cancels a set of pending requests
                      2: PCE cancels a set of pending requests

      2   PCE Congestion   This document
                    Notification-value
                      1: PCE in congested state
                      2: PCE no longer in congested state

9.6. PCEP-ERROR Object

   IANA is requested to create a registry for the Error-type and Error-
   value of the PCEP Error Object and manage the code space.

    For each PCEP error, an Error-type and an Error-value are defined.
Error-Type Meaning
Reference
    1   PCEP session establishment failure   This
document
              Error-value=1: reception of an invalid Open message or
                             a non Open 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   This
document
    3   Unknown Object   This
document
                Error-value=1: Unrecognized object class
                Error-value=2: Unrecognized object Type
    4   Not supported object   This
document
                Error-value=1: Not supported object class
                Error-value=2: Not supported object Type
    5   Policy violation   This
document
                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   This
document
                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   This
document
    8   Unknown request reference   This
document
    9   Attempt to establish a second PCEP session   This
document
    10   Reception of an invalid object   This
document

Error-value=1: reception of an object with P flag not set
although
                    the P-flag must be set according to this specification.

9.7. 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
   4           Reception of an unacceptable number of
               unknown requests/replies
   5           Reception of an unacceptable number of
               unrecognized PCEP messages
```

[9.8](9.8). NO-PATH Object

IANA is requested to create a registry to manage the codespace of NI
field present in the NO-PATH Object.

```
 Value         Meaning                      Reference

   0     No path satisfying the set        This document
         of constraints could be found
   1     PCE chain broken                  This document
```

[9.9](9.9). METRIC Object

IANA is requested to create a registry to manage the codespace of T
field and the Flag field of the METRIC Object.

```
Codespace of the T field (Metric Object)
 Value         Meaning            Reference

   1           IGP metric     This document
   2           TE metric      This document
   3           Hop Counts     This document
```

New bit numbers may be allocated only by an IETF Consensus action.
Each bit should be tracked with the following qualities:

o Bit number

o Capability Description

o Defining RFC

Several bits are defined in this document. The following values have
been assigned:

   Codespace of the Flag field (Metric Object)
     Bit   Description   Reference

      0   Bound   This document
      1   Computed metric   This document

9.10. PCEP TLV Type Indicators

   IANA is requested to create a registry for the PCEP TLVs.

     Value   Meaning   Reference

      1   NO-PATH-VECTOR TLV   This document
      2   OVERLOAD-DURATION TLV   This document
      3   REQ-MISSING TLV   This document

9.11. 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 from 0 as
   the least significant bit.

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

   Each bit should be tracked with the following qualities: - Bit number
   - Name flag - Reference

     Bit Number   Name   Reference
      31   PCE currently Unavailable   This document
      30   Unknown Destination   This document
      29   Unknown Source   This document

10. Security Considerations

10.1. Vulnerability

   Attacks on PCEP may result in damage to active networks. If path
   computation responses are changed, the PCC may be encouraged to set
   up inappropriate LSPs. Such LSPs might deviate to parts of the
   network susceptible to snooping, or might transit congested or
   teserved links. Path computation responses may be attacked by
   modification of the PCRep message, by impersonation of the PCE, or by
   modification of the PCReq to cause the PCE to perform a different
   computation from that which was originally requested.

   It is also possible to damage the operation of a PCE through a

variety of denial of service attacks. Such attacks can cause the PCE
to become congested with the result that path computations are
supplied too slowly to be of value for PCCs. This could lead to
slower than acceptable recovery times or delayed LSP establishment.
In extreme cases it may be that service requests are not satisfied.

PCEP could be the target of the following attacks:

o Spoofing (PCC or PCE impersonation)

o Snooping (message interception)

o Falsification

o Denial of Service

In inter-AS scenarios when PCE-to-PCE communication is required,
attacks may be particularly significant with commercial as well as
service-level implications.

Additionally, snooping of PCEP requests and responses may give an
attacker information about the operation of the network. Simply by
viewing the PCEP messages someone can determine the pattern of
service establishment in the network, and can know where traffic is
being routed making the network susceptible to targeted attacks and
the data within specific LSPs vulnerable.

The following sections identify mechanisms to protect PCEP against
security attacks.

10.2. TCP Security Techniques

At the time of writing, TCP-MD5 [RFC2385] is the only available
security mechanism for securing the TCP connections that underly PCEP
sessions.

As explained in [RFC2385], the use of MD5 faces some limitations and
does not provide as high a level of security as was once believed. A
PCEP implementation supporting TCP-MD5 SHOULD be designed so that
stronger security keying techniques or algorithms that may be
specified for TCP can be easily integrated in future releases.

The TCP Authentication Option [I-D.ietf-tcpm-tcp-auth-opt] specifies
new security procedures for TCP, but is not yet complete. Since it
is believed that [I-D.ietf-tcpm-tcp-auth-opt] will offer
significantly improved security for applications using TCP.

Implementations MUST support either TCP-MD5 or TCP Authentication

Option and should make the security function available as a
configuration option.

Operators will need to observe that some deployed PCEP
implementations may pre-date the completion of
[I-D.ietf-tcpm-tcp-auth-opt] and it will be necessary to configure
policy for secure communication between PCEP speakers that support
the TCP Authentication Option, and those that don't.

An alternative approach for security over TCP transport is to use the
Transport Layer Security (TLS) protocol [RFC5246]. This provides
protection against eavesdropping, tampering, and message forgery.
But TLS doesn't protect the TCP connection itself, because it does
not authenticate the TCP header. Thus it is vulnerable to attacks
such as TCP reset attacks (something against which TCP-MD5 does
protect). The use of TLS would, however, require the specification
of how PCEP initiates TLS handshaking and how it interprets the
certificates exchanged in TLS. This specification is out of the
scope of this document, but could be the subject of future work.

10.3. PCEP Authentication and Integrity

Authentication and integrity checks allow the receiver of a PCEP
message to know that the message genuinely comes from the node that
purports to have sent it and to know whether the message has been
modified.

The TCP-MD5 mechanism [RFC2385] described in the previous section
provides such a mechanism subject to the concerns listed in [RFC2385]
and [RFC4278]. These issues will be addressed and resolved by
[I-D.ietf-tcpm-tcp-auth-opt].

10.4. 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.

PCEP privacy can be ensured by encryption. TCP MAY be run over IPsec
[RFC4303] tunnels to provide the required encryption. Note IPsec can
also ensure authentication and integrity, in which case TCP-MD5 or
TCP-AO would not be required. However, there is some concern that
IPsec on this scale would be hard to configure and operate. Use of
IPSec with PCEP is out of the scope of this document and may be
addressed in a separate document.

10.5. Key Configuration and Exchange

   Authentication, tamper protection, and encryption all require the use
   of keys by sender and receiver.

   Although key configuration per session is possible, it may be
   particularly onerous to operators (in the same way as for the Border
   Gateway Protocol (BGP) as discussed in [I-D.ietf-rpsec-bgpsecrec]).
   If there is a relatively small number of PCCs and PCEs in the network
   manual key configuration MAY be consider as a valid choice by the
   operator, although it is important to be aware of the vulnerabilities
   introduced by such mechanisms as configuration errors, social
   engineering, and carelessness could all give rise to secrity
   breeches. Furthermore, manually configured keys are less likely to
   be regularly updated which also increases the security risk. Where
   there is a large number of PCCs and PCEs, the operator could find
   that key configuration and maintenance is a significant burden as
   each PCC needs to be configured to the PCE.

   An alternative to individual keys is the use of a group key. A group
   key is common knowledge among all members of a trust domain. Thus,
   since the routers in an IGP area or an AS are part of a common trust
   domain [I-D.ietf-mpls-mpls-and-gmpls-security-framework], a PCEP
   group key MAY be shared among all PCCs and PCEs in an IGP area or AS.
   The use of a group key will considerably simplify the operator's
   configuration task while continuing to secure PCEP against attack
   from outside the network. However, it must be noted that the more
   entities that have access to a key, the greater the risk of that key
   becoming public.

   With the use of a group key, separate keys would need to be
   configured for the PCE-to-PCE communications that cross trust domain
   (e.g., AS) boundaries, but the number of these relationships is
   likely to be very small.

   PCE discovery ([RFC5088] and [RFC5089]) is a significant feature for
   the successful deployment of PCEP in large networks. This mechanism
   allows a PCC to discover the existence of suitable PCEs within the
   network without the necessity of configuration. It should be obvious
   that, where PCEs are discovered and not configured, the PCC cannot
   know the correct key to use. There are three possible approaches to
   this problem that retain some aspect of security:

   o The PCCs may use a group key as previously discussed,

   o The PCCs may use some form of secure key exchange protocol with
      the PCE (such as the Internet Key Exchange protocol v2 (IKE)
      [RFC4306]. The drawback to this is that IKE implementations on

routers are not common and this may be a barrier to the deployment
of PCEP. Details are out of the scope of this document and may be
addressed in a separate document.

o The PCCs may make use of a key server to determine the key to use
   when talking to the PCE. To some extent, this is just moving the
   problem, siince the PCC's communications with the key server must
   also be secure (for example using Kerberos [RFC4120]), but there
   may some (minor) benefit in scaling if the PCC is to learn about
   several PCEs and only need to know one key server. Note that key
   servers currently have very limited implementation. Details are
   out of the scope of this document and may be addressed in a
   separate document.

PCEP relationships are likely to be long-lived even if the PCEP
sessions are repeatedly closed and re-established. Where protocol
relationships persist for a large number of protocol interactions or
over a long period of time, changes in the keys used by the protocol
peers is RECOMMENDED [RFC4107]. Note that TCP-MD5 does not allow the
key to be changed without closing and re-opening the TCP connection
which would result in the PCEP session being terminated and needing
to be restarted. That might not be an significant issue for PCEP.
Note also that the plans for the TCP Authentication Option
[I-D.ietf-tcpm-tcp-auth-opt] will allow dynamic key change (roll-
over) for an active TCP connection.

If key exchange is used (for example through IKE) then it is
relatively simple to support dynamic key updates and apply these to
PCEP.

Note that inband key management for the TCP Authentication Option
[I-D.ietf-tcpm-tcp-auth-opt] is currently unresolved.

[RFC3562] sets out some of the issues for the key management of
secure TCP connections.

10.6. Access Policy

Unauthorised access to PCE function represents a variety of potential
attacks. Not only may this be a simple Denial of Service attack (see
Section 10.7), but it would be a mechanism for an intruder to
determine important information about the network and operational
network policies simply by inserting bogus computation requests.
Furthermore, false computation requests could be used to predict
where traffic will be placed in the network when real requests are
made, allowing the attacker to target specific network resources.

PCEs SHOULD be configurable for access policy. Where authentication

   is used, access policy can be achieved through the exchange or
   configuration of keys as described in Section 10.5. More simple
   policies MAY be configured on PCEs in the form of access lists where
   the IP addresses of the legitimate PCCs are listed. Policies SHOULD
   also be configurable to limit the type of computation requests that
   are supported from different PCCs.

   It is RECOMMENDED that access policy violations are logged by the PCE
   and are available for inspection by the operator to determine whether
   attempts have been made to attack the PCE. Such mechanisms MUST be
   lightweight to prevent them from being used to support denial of
   service attacks (see Section 10.7).

10.7. Protection Against Denial of Service Attacks

   Denial of service (DoS) attacks could be mounted at the TCP level or
   at the PCEP level. That is, the PCE could be attacked through
   attacks on TCP or through attacks within established PCEP sessions.

10.7.1. Protection Against TCP DoS Attacks

   PCEP can be the target of TCP DoS attacks, such as for instance SYN
   attacks, as is the case for all protocols that run over TCP. Other
   protocol specifications have investigated this problem and PCEP can
   share their experience. The reader is referred to the specification
   of the Label Distribution Protocol (LDP) [RFC5036] for example. In
   order to protect against TCP DoS attacks, PCEP implementations can
   support the following techniques.

   o PCEP uses a single registered port for all communications. The
      PCE SHOULD listen for TCP connections only on ports where
      communication is expected.

   o The PCE MAY implement an access list to immediately reject (or
      discard) TCP connection attempts from unauthorized PCCs.

   o The PCE SHOULD NOT allow parallel TCP connections from the same
      PCC on the PCEP registered port.

   o The PCE MAY require the use of the MD5 option on all TCP
      connections rejecting (or discarding) any connection setup attempt
      that does not use MD5, and not accepting any SYN for which the MD5
      segment checksum is invalid. Note, however, that the use of MD5
      requires that the receiver use CPU resources to compute the
      checksum before it can decide to discard an otherwise acceptable
      SYN segment.

10.7.2. Request Input Shaping/Policing

   A PCEP implementation may be subject to DoS attacks within a
   legitimate PCEP session. For example, a PCC might send a very large
   number of PCReq messages causing the PCE to become congested or
   causing requests from other PCCs to be queued.

   Note that the direct use of the Priority field on the RP Object to
   prioritize received requests does not provide any protection since
   the attacker could set all requests to be of the highest priority.

   Therefore, it is RECOMMENDED that PCE implementations include input
   shaping/policing mechanisms that either throttle the requests
   received from any one PCC, or apply queuing or priority-degradation
   techniques to over-communicative PCCs.

   Such mechanisms MAY be set by default, but SHOULD be available for
   configuration. Such techniques may be considered patricularly
   important in multi- service-provider environments to protect the
   resources of one service provider from unwarranted, over-zealous, or
   malicious use by PCEs in another service provider.

11. Authors' Addresses

   The content of this document was contributed by the editors and the
   co-authors listed below:

   Arthi Ayyangar
   Juniper Networks
   1194 N. Mathilda Ave
   Sunnyvale, CA 94089
   USA

   Email: arthi@juniper.net

   Adrian Farrel
   Old Dog Consulting
   Phone: +44 (0) 1978 860944
   EMail: adrian@olddog.co.uk

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

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

Alia Atlas
British Telecom
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

12. Acknowledgements

13. References

13.1. 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.

   [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an
              IANA Considerations Section in RFCs", BCP 26, RFC 5226,
              May 2008.

13.2. Informative References

   [I-D.farrel-rtg-common-bnf]
              Farrel, A., "Reduced Backus-Naur Form (RBNF) A Syntax Used
              in Various Protocol Specifications",
              draft-farrel-rtg-common-bnf-07 (work in progress),
              November 2008.

   [I-D.ietf-mpls-mpls-and-gmpls-security-framework]
              Fang, L. and M. Behringer, "Security Framework for MPLS
              and GMPLS Networks",
              draft-ietf-mpls-mpls-and-gmpls-security-framework-03 (work
              in progress), July 2008.

   [I-D.ietf-pce-inter-layer-req]
              Oki, E., Roux, J., Kumaki, K., Farrel, A., and T. Takeda,
              "PCC-PCE Communication and PCE Discovery Requirements for
              Inter-Layer Traffic Engineering",
              draft-ietf-pce-inter-layer-req-08 (work in progress),

                    October 2008.

    [I-D.ietf-pce-interas-pcecp-reqs]
                    Bitar, N., Kumaki, K., and R. Zhang, "Inter-AS
                    Requirements for the Path Computation Element
                    Communication Protocol (PCEP)",
                    draft-ietf-pce-interas-pcecp-reqs-06 (work in progress),
                    May 2008.

    [I-D.ietf-pce-manageability-requirements]
                    Farrel, A., "Inclusion of Manageability Sections in PCE
                    Working Group Drafts",
                    draft-ietf-pce-manageability-requirements-05 (work in
                    progress), October 2008.

    [I-D.ietf-pce-monitoring]
                    Vasseur, J., Roux, J., and Y. Ikejiri, "A set of
                    monitoring tools for Path Computation Element based
                    Architecture", draft-ietf-pce-monitoring-02 (work in
                    progress), September 2008.

    [I-D.ietf-rpsec-bgpsecrec]
                    Christian, B. and T. Tauber, "BGP Security Requirements",
                    draft-ietf-rpsec-bgpsecrec-09 (work in progress),
                    November 2007.

    [I-D.ietf-tcpm-tcp-auth-opt]
                    Touch, J., Mankin, A., and R. Bonica, "The TCP
                    Authentication Option", draft-ietf-tcpm-tcp-auth-opt-01
                    (work in progress), July 2008.

    [I-D.kkoushik-pce-pcep-mib]
                    Stephan, E. and K. Koushik, "PCE communication
                    protocol(PCEP) Management Information Base",
                    draft-kkoushik-pce-pcep-mib-01 (work in progress),
                    July 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.

    [RFC3471] Berger, L., "Generalized Multi-Protocol Label Switching
                    (GMPLS) Signaling Functional Description", RFC 3471,
                    January 2003.

    [RFC3562] Leech, M., "Key Management Considerations for the TCP MD5

                    Signature Option", RFC 3562, July 2003.

   [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.

   [RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic
              Key Management", BCP 107, RFC 4107, June 2005.

   [RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The
              Kerberos Network Authentication Service (V5)", RFC 4120,
              July 2005.

   [RFC4278] Bellovin, S. and A. Zinin, "Standards Maturity Variance
              Regarding the TCP MD5 Signature Option (RFC 2385) and the
              BGP-4 Specification", RFC 4278, January 2006.

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

   [RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol",
              RFC 4306, 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.

   [RFC5036] Andersson, L., Minei, I., and B. Thomas, "LDP
              Specification", RFC 5036, October 2007.

   [RFC5088] Le Roux, JL., Vasseur, JP., Ikejiri, Y., and R. Zhang,
              "OSPF Protocol Extensions for Path Computation Element
              (PCE) Discovery", RFC 5088, January 2008.

   [RFC5089] Le Roux, JL., Vasseur, JP., Ikejiri, Y., and R. Zhang,
              "IS-IS Protocol Extensions for Path Computation Element
              (PCE) Discovery", RFC 5089, January 2008.

   [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security
              (TLS) Protocol Version 1.2", RFC 5246, August 2008.

13.3. References

   [IEEE.754.1985]
              IEEE Standard 754, "Standard for Binary Floating-Point
              Arithmetic", August 1985.

Appendix A. PCEP Finite State Machine (FSM)

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

PCEP Finite State Machine

```
         +-+-+-+-+-+-+-+<------+
 +------| SessionUP |<---+  |
 |      +-+-+-+-+-+-+    |  |
 |                      |  |
 |    +->+-+-+-+-+-+-+   |  |
 |    | | KeepWait  |----+ |
 |    +--|           |<---+ |
 |+-----+-+-+-+-+-+-+    |  |
 ||          |          |  |
 ||          |          |  |
 ||          V          |  |
 ||   +->+-+-+-+-+-+-+---+ |
 ||   | | OpenWait  |-------+
 ||   +--|           |<------+
 ||+----+-+-+-+-+-+-+<---+  |
 |||         |          |  |
 |||         |          |  |
 |||         V          |  |
 ||| +->+-+-+-+-+-+-+    |  |
 ||| |  |TCPPending |----+  |
 ||| +--|           |       |
 |||+---+-+-+-+-+-+-+<---+   |
 ||||        |          |  |
 ||||        |          |  |
 ||||        V          |  |
 |||+--->+-+-+-+-+       |  |
 ||+---->| Idle  |-------+  |
 |+----->|       |----------+
 +------>+-+-+-+-+
```
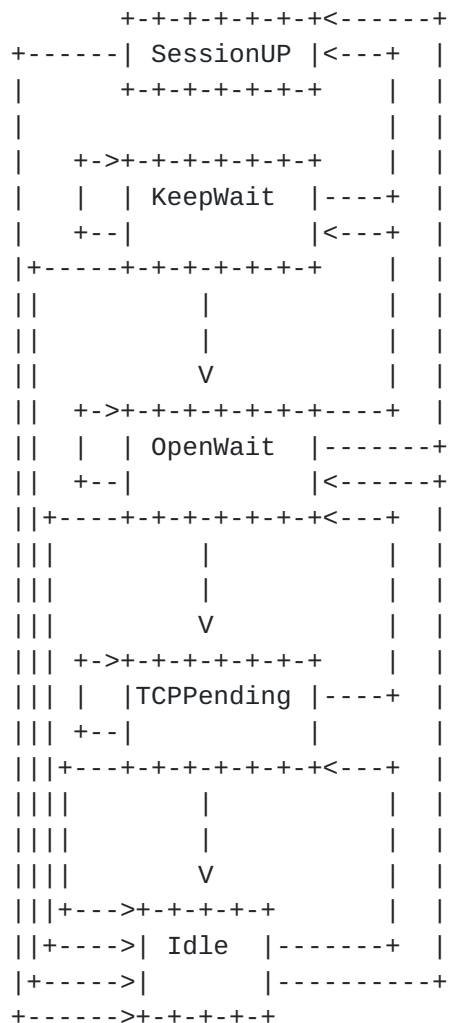
Figure 23: PCEP Finite State Machine for the PCC

PCEP defines the following set of variables:

Connect: timer (in seconds) started after having initialized a TCP
connection using the PCEP registered 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 registered 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 registered 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:

   o Initiates of a TCP connection with the PCEP peer,

   o Starts the Connect timer,

   o Moves to the TCPPending state.

   Upon receiving a TCP connection on the registered PCEP TCP port, if
   the TCP connection establishment succeeds, the system:

   o Sends an Open message,

   o Starts the OpenWait timer,

   o 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
   increasing timer between automatically generated Initialization
   events and between retries of TCP connection establishment.

   TCPPending State

   If the TCP connection establishment succeeds, the system:

   o Sends an Open message,

   o Starts the OpenWait timer,

   o 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:

   o If ConnectRetry =ConnectMaxRetry the system moves to the Idle
      State

   o If ConnectRetry < ConnectMaxRetry the system:

      1. Initiates of a TCP connection with the PCEP peer,

      2. Increments the ConnectRetry variable,

   3. Restarts the Connect timer,

   4. Stays in the TPCPending state.

   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, the system first examines all of
   its sessions that are in the OpenWait or KeepWait state. If another
   session with the same PCEP peer already exists (same IP address),
   then the system performs the following collision resolution
   procedure:

   o If the system has initiated the current session and it has a lower
      IP address than the PCEP Peer, the system closes the TCP
      connection, releases the PCEP resources for the pending session
      and moves back to the Idle state.

   o If the session was initiated by the PCEP peer and the system has a
      higher IP address that the PCEP Peer, the system closes the TCP
      connection, releases the PCEP resources for the pending session,
      and moves back to the Idle state.

   o Otherwise, the system checks the PCEP session attributes
      (Keepalive frequency, DeadTimer, ...).

   If an error is detected (e.g. malformed Open message, reception of a
   message that is not an 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, 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:

o Sends a Keepalive message to the PCEP peer,

o Starts the Keepalive timer,

o 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, and OpenRetry is 0, and the session
characteristics are unacceptable but negotiable (such as, the
Keepalive period or the DeadTimer), then the system:

o Increments the OpenRetry variable,

o Sends a PCErr message with Error-Type=1 and Error-value=4 that
   contains proposed acceptable session characteristics,

o If LocalOK=1, the system restarts the OpenWait timer and stays in
   the OpenWait state

o 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:

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

o 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 before 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.

Appendix B. 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 7.13.3.
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. The aim of the SyncTimer is to avoid the storage of unused
synchronized request should one of them get lost for some reasons
(e.g a misbehaving PCC). Thus the value of the Synctimer must be
large enough to avoid the expiration of the timer under normal
circumstances. A RECOMMENDED value for the SYNC timer is 60 seconds.

MAX-UNKNOWN-REQUESTS: A RECOMMENDED value is 5.

MAX-UNKNOWN-MESSAGES: A RECOMMENDED value is 5.

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