PCE Working Group   M. Koldychev
Internet-Draft   S. Sivabalan
Intended status: Standards Track   Cisco Systems, Inc.
Expires: August 27, 2020   M. Negi
                                                        Huawei Technologies
                                                                D. Achaval
                                                                      Nokia
                                                                   H. Kotni
                                                      Juniper Networks, Inc
                                                          February 24, 2020

                       PCEP Operational Clarification
                     draft-koldychev-pce-operational-01

Abstract

   This document is meant to provide better clarity about how PCEP
   operates and hence to facilitate better interoperability between
   different equipment vendors. The content of this document has been
   compiled based on the feedback from several multi-vendor interop
   exercises. Several constructs are introduced to facilitate this,
   such as the LSP-DB and the ASSO-DB.

Requirements Language

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
   "OPTIONAL" in this document are to be interpreted as described in BCP
   14 [RFC2119] [RFC8174] when, and only when, they appear in all
   capitals, as shown here.

Table of Contents

[1](#). Introduction

   Path Computation Element (PCE) Communication Protocol (PCEP)
   [[RFC5440](#)] enables the communication between a Path Computation Client
   (PCC) and a Path Control Element (PCE), or between two PCEs based on
   the PCE architecture [[RFC4655](#)].

   PCEP Extensions for the Stateful PCE Model [[RFC8231](#)] describes a set
   of extensions to PCEP to enable active control of Multiprotocol Label
   Switching Traffic Engineering (MPLS-TE) and Generalized MPLS (GMPLS)

tunnels. [RFC8281] describes the setup and teardown of PCE-initiated
LSPs under the active stateful PCE model, without the need for local
configuration on the PCC, thus allowing for dynamic centralized
control of a network.

PCEP Extensions for Establishing Relationships Between Sets of LSPs
[RFC8697] introduces a generic mechanism to create a grouping of LSPs
which can then be used to define associations between a set of LSPs
and a set of attributes (such as configuration parameters or
behaviors) and is equally applicable to stateful PCE (active and
passive modes) and stateless PCE.

The PCEP protocol has evolved from a simple stateless model into a
stateful model with more features being added. Due to subtle
differences in interpretation of existing PCEP standards, it was
found that networking equipment vendors often had to adjust their
implementations, in order to interoperate. This informational
document is meant to clarify these subtle differences and agree on a
final model that all major vendors have agreed on and that all other
vendors can adopt. This document applies to RSVP-TE and Segment-
Routing.

2. Terminology

The following terminologies are used in this document:

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

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

PCEP: Path Computation Element Protocol.

MBB: Make-Before-Break. A procedure during which the head-end of a
   traffic-engineered path wishes to move traffic to a new path
   without losing any traffic, by first "making" a new path and then
   "breaking" the old path.

Association parameters: As described in [RFC8697], the combination
   of the mandatory fields Association type, Association ID and
   Association Source in the ASSOCIATION object uniquely identify the
   association group. If the optional TLVs - Global Association
   Source or Extended Association ID are included, then they MUST be
   included in combination with mandatory fields to uniquely identify
   the association group.

   Association information: As described in [RFC8697], the ASSOCIATION
      object could also include other optional TLVs based on the
      association types, that provides 'information' related to the
      association type.

   ERO: Explicit Route Object is the path of the LSP encoded into a
      PCEP object. To represent an empty ERO object, i.e., without any
      subobjects, we use the notation "ERO={}". To represent an ERO
      object containing some given sequence of subobjects, we use the
      notation "ERO={A}".

3. PCEP LSP Database

   We introduce the concept of the LSP-DB, as a database of actual LSP
   state in the network. This concept is not explicitly defined in
   [RFC8231], but is fully compatible with it. We use the LSP-DB to
   describe how certain actions are performed, because it is easier to
   define actions as a function of database state, rather than as a
   function of previously received messages. The structure and format
   of the LSP-DB MUST be common among all dataplane types (i.e., RSVP-
   TE/SR-TE/SRv6), all instantiation methods (i.e., PCC-initiated/PCE-
   initiated), all destination types (i.e., point-to-point/point-to-
   multipoint).

   Note that we use the term "Tunnel" somewhat loosely here, to mean
   "the object identified by the PLSP-ID". It may or may not be an
   actual tunnel in the implementation. For example, working and
   protect paths can be implemented as one tunnel interface, but in PCEP
   we would refer to them as two different Tunnels, because they would
   have different PLSP-IDs.

   Note that the term "LSP", which stands for "Label Switched Path", if
   taken too literally would restrict our discussion to MPLS dataplane
   only. In this document, we allow the term "LSP" to refer to any
   path, regardless of the dataplane format. So that an LSP can refer
   to MPLS and SRv6 dataplane paths.

3.1. Structure

   [RFC8231] states that the LSP-IDENTIFIERS TLV contains the key that
   MUST be used to differentiate different LSPs during make before break
   procedure. We further clarify here that PCEP LSPs exist in a 2-tier
   structure. The top tier is the "Tunnel", identified by the PLSP-ID
   and/or SYMBOLIC-NAME, while the lower tier is the "LSP", identified
   by the values in LSP-IDENTIFIERS TLV. A single Tunnel may contain
   multiple LSPs at the same time, i.e., a Tunnel is a container for
   LSPs. A Tunnel MUST have at least one LSP and when the last LSP is
   removed from the Tunnel, the Tunnel itself is removed.

3.2. Synchronization

   The stateful PCE MUST maintain the PCE LSP-DB, which stores Tunnels
   and LSPs. The PCE LSP DB is only modified by PCRpt messages. No
   other PCEP message may modify the PCE LSP DB. The PCC MUST also
   maintain the PCC LSP DB, which it MUST synchronize with the PCE LSP
   DB by sending PCRpt messages.

   The PCC adds/removes entries to/from its LSP-DB based on what LSPs it
   creates/destroys in the network. There can be many trigger types for
   updating the PCC LSP-DB, some examples include PCUpd messages, local
   computation on the PCC, local configuration on the PCC, etc. The
   trigger type does not affect the content of the PCC LSP-DB, i.e., the
   content of the PCC LSP-DB is updated identically regardless of the
   trigger type.

   Whenever a PCC modifies an entry it its PCC LSP-DB, it MUST send a
   PCRpt message to the PCE (or multiple PCEs), to synchronize this
   change. Ensuring this synchronization is always in place allows one
   to define behavior as a function of LSP-DB state, instead of defining
   behavior as a function of what PCEP messages were sent or received.

   The PCE MUST always act on the latest state of the PCE LSP DB. Note
   that this does not mean that the PCE cannot use information from
   outside of LSP-DB. For example, the PCE can use other mechanisms to
   collect traffic statistics and use them in the computation. However,
   these traffic statistics are not part of the LSP-DB, but only
   reference it.

   The LSP-DB on both the PCC and the PCE only stores the actual state
   in the network, it does not store the desired state. For example,
   consider the case of PCE Initiated LSP, configured on the PCE. When
   the operator modifies the configuration of this LSP, that is a change
   in desired state. The actual state has not yet changed, so LSP-DB is
   not modified yet. The LSP-DB is only modified after the PCE sends
   PCInit/PCUpd message to the PCC and the PCC decides to act on that
   message. When the PCC acts on message, it would update its own PCC
   LSP DB and immediately send PCRpt to the PCE to synchronize the
   change. When the PCE receives the PCRpt msg, it updates its own PCE
   LSP DB. After this, the PCC LSP DB and PCE LSP DB are in sync.

3.3. Stateful Bringup

   [RFC8231] in section 5.8.2, allows delegation of an LSP in
   operationally down state, but at the same time mandates the use of
   PCReq, before sending PCRpt. In this document, we would like to make
   it clear that sending PCReq is optional.

We shall refer to the process of sending PCReq before PCRpt as
"stateless bringup". In reality, stateless bringup introduces
overhead and is not possible to enforce from the PCE, because the
stateless PCE is not supposed to keep any per-LSP state about
previous PCReq messages. It was found that many vendors choose to
ignore this requirement and send the PCRpt directly, without going
through PCReq. This section will serve to explain and to validate
this behavior.

Even though all the major vendors today are moving to the stateful
PCE model, it does not deprecate the need for stateless PCEP. The
key property of stateless PCEP is that PCReq messages MUST NOT modify
the state of the PCE LSP-DB in any way. Therefore, PCReq messages
are useful for many OAM ping/traceroute applications where the PCC
wishes to probe the network without having any effect on the existing
LSPs.

The PCC MAY delegate an empty LSP to the PCE and then wait for the
PCE to send PCUpd, without sending PCReq. We shall refer to this
process as "stateful bringup". The PCE MUST support the original
stateless bringup, for backward compatibility purposes. Supporting
stateful bringup should not require introducing any new behavior on
the PCE, because as mentioned earlier, the PCE MUST NOT modify LSP-DB
state based on PCReq messages. So whether the PCE has received a
PCReq or not, it MUST process the PCRpt all the same.

An example of stateful bringup follows. In our example the PCC
starts off by using LSP-ID of 0. The value 0 does not hold any
special meaning, any other 16-bit value could have been used.

PCC has no LSP yet, but wants to establish a path. PCC sends
PCRpt(R-FLAG=0, D-flag=1, OPER-FLAG=DOWN, PLSP-ID=100, LSP-ID=0,
ERO={}).

```
   +----------------------------------------------------------------+
   | TUNNEL           | LSP                                         |
   +------------------+---------------------------------------------+
   | PLSP-ID=100      | LSP-ID=0, D-flag=1, OPER=DOWN, ERO={}       |
   +----------------------------------------------------------------+
```

Figure 1: Content of LSP DB

PCC received a PCUpd from the PCE and has decided to install the
ERO={A} from that PCUpd. PCC sends PCRpt(R-FLAG=0, D-flag=1, OPER-
FLAG=UP, PLSP-ID=100, LSP-ID=0, ERO={A}).

```
+----------------------------------------------------------------+
| TUNNEL           | LSP                                         |
+------------------+---------------------------------------------+
| PLSP-ID=100      | LSP-ID=0, D-flag=1, OPER=UP, ERO={A}        |
+----------------------------------------------------------------+
```

Figure 2: Content of LSP DB

3.4. Successful MBB

Below we give an example of doing MBB to switch the tunnel from one
path to another. We represent the path encoded into the ERO object
as ERO={A} and ERO={B}.

PCC has an existing LSP in UP state, with LSP-ID=2. PCC sends
PCRpt(R-FLAG=0, PLSP-ID=100, LSP-ID=2, ERO={A}, OPER-FLAG=UP).

```
+----------------------------------------------------------------+
| TUNNEL           | LSP                                         |
+------------------+---------------------------------------------+
| PLSP-ID=100      | LSP-ID=2, ERO={A}, OPER=UP                  |
+----------------------------------------------------------------+
```

Figure 3: Content of LSP DB

PCC initiates the MBB procedure by creating a new LSP with LSP-ID=3.
It does not matter what triggered the creation of the new LSP, it
could have been due to a new path received via PCUpd (if the given
tunnel is delegated), or it could have been local computation on the
PCC (if the tunnel is locally computed on the PCC), or it could have
been a change in configuration on the PCC (if the tunnel's path is
explicitly configured on the PCC). It is important to emphasize that
the procedure for updating the LSP-DB is common, regardless of the
trigger that caused the change.

PCC sends PCRpt(R-FLAG=0, PLSP-ID=100, LSP-ID=3, ERO={B}, OPER-
FLAG=UP).

```
+----------------------------------------------------------------+
| TUNNEL           | LSP                                         |
+------------------+---------------------------------------------+
| PLSP-ID=100      | LSP-ID=2, ERO={A}, OPER=UP                  |
|                  | LSP-ID=3, ERO={B}, OPER=UP                  |
+----------------------------------------------------------------+
```

Figure 4: Content of LSP DB

   After some time, the PCC decides to destroy the old LSP. PCC sends
   PCRpt(R-FLAG=1, PLSP-ID=100, LSP-ID=2).

```
   +------------------------------------------------------------------+
   | TUNNEL           | LSP                                           |
   +------------------+-------------------------------------------------+
   | PLSP-ID=100      | LSP-ID=3, ERO={B}, OPER=UP                    |
   +------------------------------------------------------------------+
```

                       Figure 5: Content of LSP DB

3.5. Aborted MBB

   The MBB process can abort when the newly created LSP is destroyed
   before it is installed as traffic carrying. This scenario is
   described below.

   PCC has an existing LSP in UP state, with LSP-ID=2. PCC sends
   PCRpt(R-FLAG=0, OPER-FLAG=UP, PLSP-ID=100, LSP-ID=2).

```
   +------------------------------------------------------------------+
   | TUNNEL           | LSP                                           |
   +------------------+-------------------------------------------------+
   | PLSP-ID=100      | LSP-ID=2, OPER=UP                             |
   +------------------------------------------------------------------+
```

                       Figure 6: Content of LSP DB

   MBB procedure is initiated, a new LSP is created with LSP-ID=3. LSP
   is currently being established, so its oper state is DOWN. PCC sends
   PCRpt(R-FLAG=0, OPER-FLAG=DOWN, PLSP-ID=100, LSP-ID=3).

```
   +------------------------------------------------------------------+
   | TUNNEL           | LSP                                           |
   +------------------+-------------------------------------------------+
   | PLSP-ID=100      | LSP-ID=2, OPER=UP                             |
   |                  | LSP-ID=3, OPER=DOWN                           |
   +------------------------------------------------------------------+
```

                       Figure 7: Content of LSP DB

   MBB procedure is aborted. PCC sends PCRpt(R-FLAG=1, PLSP-ID=100,
   LSP-ID=3).

```
   +---------------------------------------------------------------+
   | TUNNEL            | LSP                                        |
   +-------------------+-------------------------------------------+
   | PLSP-ID=100       | LSP-ID=2, OPER=UP                          |
   +---------------------------------------------------------------+
```

Figure 8: Content of LSP DB

4. PCEP Association Database

PCEP Association is a group of zero or more LSPs.

The PCE ASSO DB is populated by PCRpt messages and MAY also be
populated via configuration on the PCE itself. An Association is
identified by the Association Parameters. The Association parameters
contain many fields, so for convenience we will group all the fields
into a single value. We will use ASSO_PARAM=A, ASSO_PARAM=B, to
refer to different PCEP Associations: A and B, respectively.

4.1. 2 LSPs in same Association

Below, we give an example of LSPs joining the same Association.

PCC creates the first LSP. PCC sends PCRpt(R-FLAG=0, PLSP-ID=100,
LSP-ID=1, ASSO_PARAM=A, ASSO_R_FLAG=0).

```
   +---------------------------------------------------------------+
   | ASSO              | LSP                                        |
   +-------------------+-------------------------------------------+
   | ASSO_PARAM=A      | PLSP-ID=100, LSP-ID=1                      |
   +---------------------------------------------------------------+
```

Figure 9: Content of PCE ASSO DB

PCC creates the second LSP. PCC sends PCRpt(R-FLAG=0, PLSP-ID=200,
LSP-ID=1, ASSO_PARAM=A, ASSO_R_FLAG=0).

```
   +---------------------------------------------------------------+
   | ASSO              | LSP                                        |
   +-------------------+-------------------------------------------+
   | ASSO_PARAM=A      | PLSP-ID=100, LSP-ID=1                      |
   |                   | PLSP-ID=200, LSP-ID=1                      |
   +---------------------------------------------------------------+
```

Figure 10: Content of PCE ASSO DB

PCC updates the first LSP, the PCC is NOT REQUIRED to send the
ASSOCIATION object in this PCRpt, since the LSP is already in the

Association. PCC sends PCRpt(R-FLAG=0, PLSP-ID=100, LSP-ID=1). The
content of the PCE ASSO DB is unchanged. Note that the PCC MUST send
the ASSOCIATION OBJECT in the first PCRpt during SYNC state, even if
it has already issued a PCRpt with the association object sometime in
the past with this PCE. The synchronization steps outlined in
[RFC8697] are to be followed.

```
+---------------------------------------------------------------+
| ASSO            | LSP                                         |
+-----------------+---------------------------------------------+
| ASSO_PARAM=A    | PLSP-ID=100, LSP-ID=1                       |
|                 | PLSP-ID=200, LSP-ID=1                       |
+---------------------------------------------------------------+
```

Figure 11: Content of PCE ASSO DB

PCC decides to delete the second LSP. PCC sends PCRpt(R-FLAG=1,
PLSP-ID=200, LSP-ID=1).

```
+---------------------------------------------------------------+
| ASSO            | LSP                                         |
+-----------------+---------------------------------------------+
| ASSO_PARAM=A    | PLSP-ID=100, LSP-ID=1                       |
+---------------------------------------------------------------+
```

Figure 12: Content of PCE ASSO DB

PCC decides to remove the first LSP from the Association, but not
delete the LSP itself. PCC sends PCRpt(R-FLAG=0, PLSP-ID=100, LSP-
ID=1, ASSO_PARAM=A, ASSO_R_FLAG=1). The PCE ASSO DB is now empty.

```
+---------------------------------------------------------------+
| ASSO            | LSP                                         |
+-----------------+---------------------------------------------+
| ASSO_PARAM=A    |                                             |
+---------------------------------------------------------------+
```

Figure 13: Content of PCE ASSO DB

4.2. Switch Association during MBB

Each new LSP (identified by the LSP-ID) does not inherit the
Association membership of any previous LSPs within the same Tunnel.
This is done so that a Tunnel can have two LSPs that are in different
Associations, this may be required when switching from one
Association to another.

Below, we give an example a Tunnel going through MBB and switching
from Association A to Association B.

PCC creates the first LSP. PCC sends PCRpt(R-FLAG=0, PLSP-ID=100,
LSP-ID=1, ASSO_PARAM=A, ASSO_R_FLAG=0).

```
+--------------------------------------------------------------------+
| ASSO            | LSP                                              |
+-----------------+--------------------------------------------------+
| ASSO_PARAM=A    | PLSP-ID=100, LSP-ID=1                            |
+--------------------------------------------------------------------+
```

                  Figure 14: Content of PCE ASSO DB

PCC creates the MBB LSP in a different Association. PCC sends
PCRpt(R-FLAG=0, PLSP-ID=100, LSP-ID=2, ASSO_PARAM=B, ASSO_R_FLAG=0).

```
+--------------------------------------------------------------------+
| ASSO            | LSP                                              |
+-----------------+--------------------------------------------------+
| ASSO_PARAM=A    | PLSP-ID=100, LSP-ID=1                            |
+-----------------+--------------------------------------------------+
| ASSO_PARAM=B    | PLSP-ID=100, LSP-ID=2                            |
+--------------------------------------------------------------------+
```

                  Figure 15: Content of PCE ASSO DB

PCC deletes the old LSP. PCC sends PCRpt(R-FLAG=1, PLSP-ID=100, LSP-
ID=1).

```
+--------------------------------------------------------------------+
| ASSO            | LSP                                              |
+-----------------+--------------------------------------------------+
| ASSO_PARAM=B    | PLSP-ID=100, LSP-ID=2                            |
+--------------------------------------------------------------------+
```

                  Figure 16: Content of PCE ASSO DB

[5](#). Computation Constraints

For any PCEP object that does not have an explicit removal flag, the
absence of that object indicates removal of the constraint specified
by that object. For example, suppose the first state-report contains
an LSPA object with some affinity constraints. Then if a subsequent
state-report does not contain an LSPA object, then this means that
the previously specified affinity constraints do not apply anymore.
Same applies to all PCEP objects, like METRIC, BANDWIDTH, etc., which
do not have an explicit flag for removal. This simply ensures that

it is possible to remove a constraint without using an explicit
removal flag.

6. Security Considerations

None at this time.

7. IANA Considerations

None at this time.

8. Acknowledgement

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
           Requirement Levels", BCP 14, RFC 2119,
           DOI 10.17487/RFC2119, March 1997,
           <https://www.rfc-editor.org/info/rfc2119>.

[RFC5440] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation
           Element (PCE) Communication Protocol (PCEP)", RFC 5440,
           DOI 10.17487/RFC5440, March 2009,
           <https://www.rfc-editor.org/info/rfc5440>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC
           2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174,
           May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8231] Crabbe, E., Minei, I., Medved, J., and R. Varga, "Path
           Computation Element Communication Protocol (PCEP)
           Extensions for Stateful PCE", RFC 8231,
           DOI 10.17487/RFC8231, September 2017,
           <https://www.rfc-editor.org/info/rfc8231>.

[RFC8281] Crabbe, E., Minei, I., Sivabalan, S., and R. Varga, "Path
           Computation Element Communication Protocol (PCEP)
           Extensions for PCE-Initiated LSP Setup in a Stateful PCE
           Model", RFC 8281, DOI 10.17487/RFC8281, December 2017,
           <https://www.rfc-editor.org/info/rfc8281>.

   [RFC8697] Minei, I., Crabbe, E., Sivabalan, S., Ananthakrishnan, H.,
              Dhody, D., and Y. Tanaka, "Path Computation Element
              Communication Protocol (PCEP) Extensions for Establishing
              Relationships between Sets of Label Switched Paths
              (LSPs)", RFC 8697, DOI 10.17487/RFC8697, January 2020,
              <https://www.rfc-editor.org/info/rfc8697>.

9.2. Informative References

   [RFC4655] Farrel, A., Vasseur, J., and J. Ash, "A Path Computation
              Element (PCE)-Based Architecture", RFC 4655,
              DOI 10.17487/RFC4655, August 2006,
              <https://www.rfc-editor.org/info/rfc4655>.

Appendix A. Contributors

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

   Email: dhruv.ietf@gmail.com

   Andrew Stone
   Nokia
   Ottawa, Canada

   Email: andrew.stone@nokia.com

Authors' Addresses

   Mike Koldychev
   Cisco Systems, Inc.
   2000 Innovation Drive
   Kanata, Ontario K2K 3E8
   Canada

   Email: mkoldych@cisco.com

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

   Email: msiva@cisco.com

Mahendra Singh Negi
Huawei Technologies

Email: mahendrasingh@huawei.com

Diego Achaval
Nokia

Email: diego.achaval@nokia.com

Hari Kotni
Juniper Networks, Inc

Email: hkotni@juniper.net