PCE Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 29, 2017

S. Litkowski
Orange
S. Sivabalan
Cisco

October 26, 2016

# Enhancing redundant stateful PCE architecture to support LSP association constraint based computation

draft-litkowski-pce-state-sync-00

## Abstract

[I-D.ietf-pce-stateful-pce] defines stateful extensions for Path Computation Element Communication Protocol (PCEP). A Path Computation Client (PCC) can synchronize an LSP state information to a Path Computation Element (PCE). [I-D.ietf-pce-stateful-pce] allows for PCE redundancy where a PCC can have redundant PCEP sessions towards multiple PCEs. In such a case, a PCC gives control on a LSP to only a single PCE, and only one PCE is responsible for path computation for this delegated LSP. There are some use cases where path computation for a particular LSP is linked to another: the most common use case is path disjointness. The set of LSPs that are dependant to each other may start from different head-ends. In such a case, we cannot guarantee that at any time all the head-ends (acting as PCCs) will delegate their LSP to the same PCE. This scenario where a group of dependant LSPs are delegated to multiple PCEs is called a split-brain scenario. This split-brain scenario may lead to computation loops between PCEs. This document proposes a solution to enhance redundant stateful PCE architecture to overcome those issues.

## Requirements Language

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

## Status of This Memo

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

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

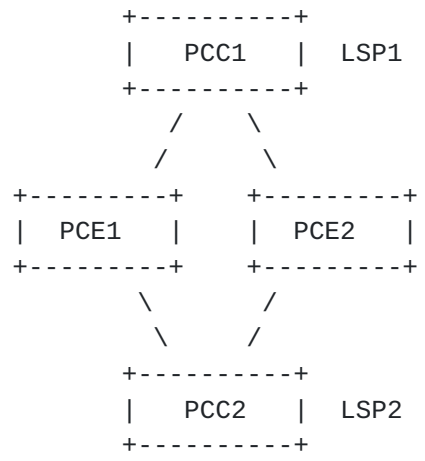
Copyright Notice



Table of Contents

## 1. Introduction and problem statement

When using stateful PCE ([I-D.ietf-pce-stateful-pce]), a Path Computation Client (PCC) can synchronize LSP state information to a Path Computation Element (PCE). In a resiliency case, PCC can may redundant PCEP sessions towards multiple PCEs. In such a case, a PCC gives control on a LSP to only a single PCE, and only one PCE is responsible for the path computation for this delegated LSP: PCC achieves this by setting the D flag only to the active PCE. The election of the active PCE is controlled on a per PCC basis. The PCC usually elects the active PCE by a local configured policy (by setting a priority). Upon PCEP session failure, or active PCE failure, PCC may decide to elect a new active PCE by sending new PCRpt with D flag set to this new active PCE. When the failed PCE or PCEP session comes back online, it will be up to the vendor to implement preemption. Doing preemption may lead to some traffic disruption on the existing path if path results from both PCEs are not exactly the same. By considering a network with multiple PCCs and implementing multiple stateful PCEs for redundancy purpose, there is no guarantee that at any time all the PCCs delegate their LSPs to the same PCE.

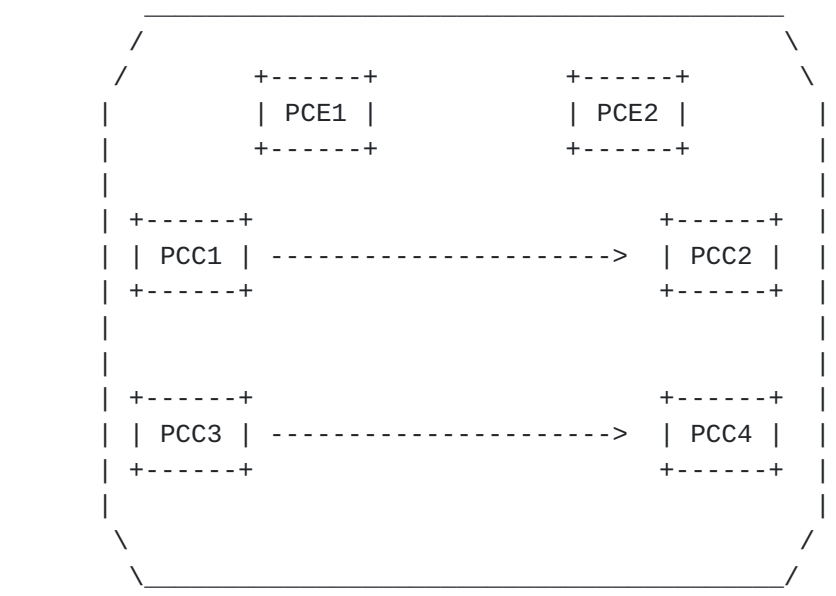


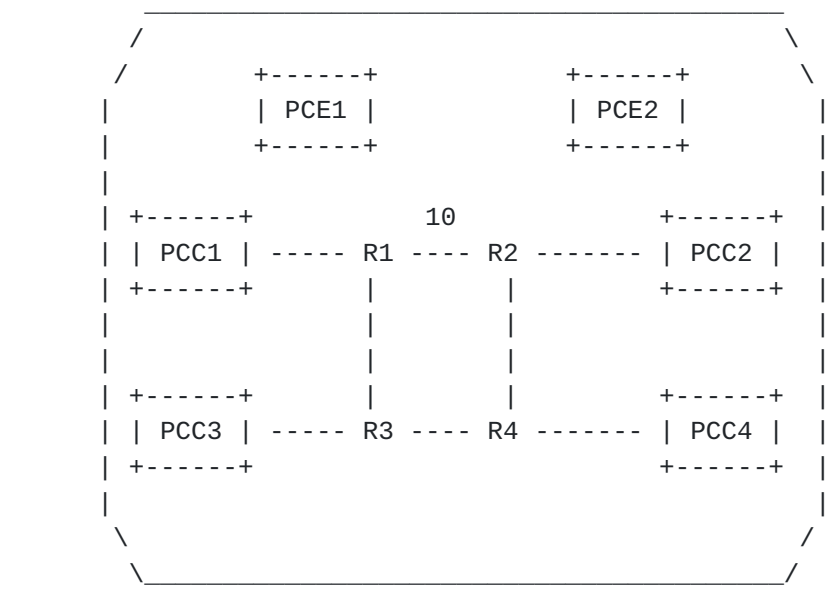

In the example above, we consider that by configuration, both PCCs will firstly delegate their LSP to PCE1. So PCE1 is responsible for computing a path for LSP1 and LSP2. If the PCEP session between PCC2 and PCE1 fails, PCC2 will delegate LSP2 to PCE2. So PCE1 becomes responsible only for LSP1 path computation while PCE2 is responsible for the path computation of LSP2. When the PCC2-PCE1 session is back

online, PCC2 will keep using PCE2 as active PCE (no preemption in this example). So the result is a permanent situation where each PCE is responsible for a subset of path computation.

We call this situation a split-brain scenario as there are multiple computation brains running at the same time while a central computation unit was required.

There are use cases where a particular LSP path computation is linked to another LSP path computation: the most common use case is path disjointness. The set of LSPs that are dependant to each other may start from a different head-end.



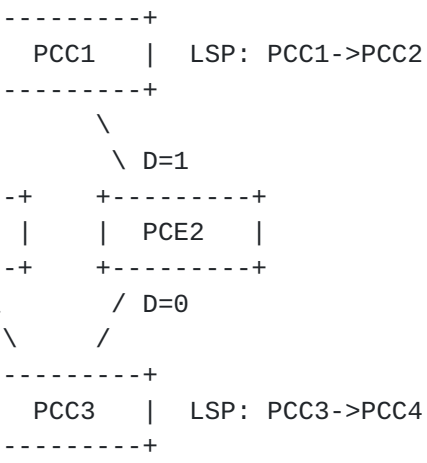

In the figure above, we want to create two link-disjoint LSPs: PCC1->PCC2 and PCC3->PCC4. In the topology, all link metrics are equal to 1 except the link R1-R2 which has a metric of 10. The PCEs are responsible for the path computation and PCE1 is the active PCE for all PCCs in the nominal case.

Scenario 1:

In the nominal case (PCE1 as active PCE), we first configure PCC1->PCC2 LSP, as the only constraint is path disjointness, PCE1 sends a PCUpdate message to PCC1 with the ERO: R1->R3->R4->R2->PCC2 (shortest path). PCC1 signals and installs the path. When PCC3->PCC4 is configured, the PCE already knows the path of PCC1->PCC2 and can compute a link-disjoint path : the solution requires to move PCC1->PCC2 onto a new path to let room for the new LSP. PCE1 sends a PCUpdate message to PCC1 with the new ERO: R1->R2->PCC2 and a PCUpdate to PCC3 with the following ERO: R3->R4->PCC4. In the nominal case, there is no issue for PCE1 to compute a link-disjoint path.

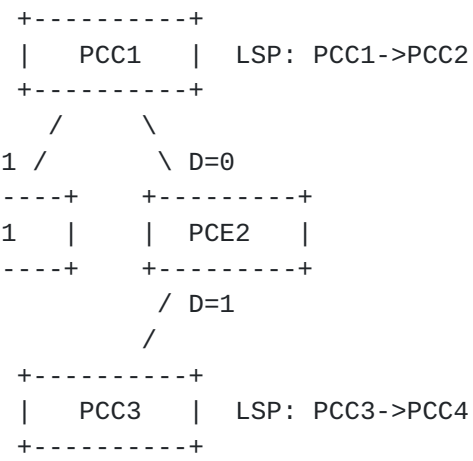
Scenario 2:

Now we consider that PCC1 losts its PCEP session with PCE1 (all other PCEP sessions are UP). PCC1 delegates its LSP to PCE2.



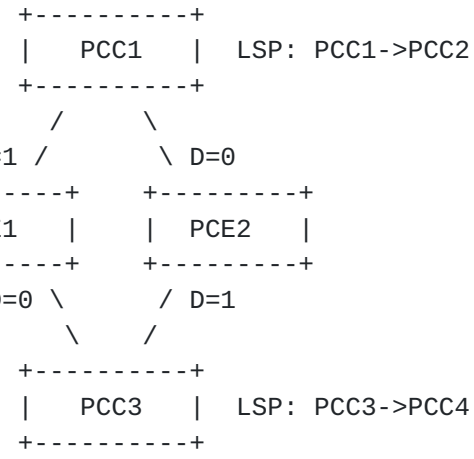

We first configure PCC1->PCC2 LSP, as the only constraint is path disjointness, PCE2 (which is the new active PCE for PCC1) sends a PCUpdate message to PCC1 with the ERO: R1->32->R4->R2->PCC2 (shortest path). When PCC3->PCC4 is configured, PCE1 is not aware anymore of LSPs from PCC1, so it cannot compute a disjoint path for PCC3->PCC4 and will send a PCUpdate message to PCC2 with a shortest path ERO: R3->R4->PCC4. When PCC3->PCC4 LSP will be reported to PCE2 by PCC2, PCE2 will ensure disjointness computation and will correctly move PCC1->PCC2 (as it owns delegation for this LSP) on the following path: R1->R2->PCC2. With this sequence of event and this PCEP session topology, disjointness is ensured.

Scenario 3:




With this new PCEP session topology, we first configure PCC1->PCC2, PCE1 computes the shortest path as it is the only LSP in the disjoint-group that it is aware of: R1->R3->R4->R2->PCC2 (shortest path). When PCC3->PCC4 is configured, PCE2 must compute a disjoint path for this LSP. The only solution found is to move PCC1->PCC2 LSP on another path, but PCE2 cannot do it as it does not have delegation for this LSP. In this setup, PCEs are not able to find a disjoint path.

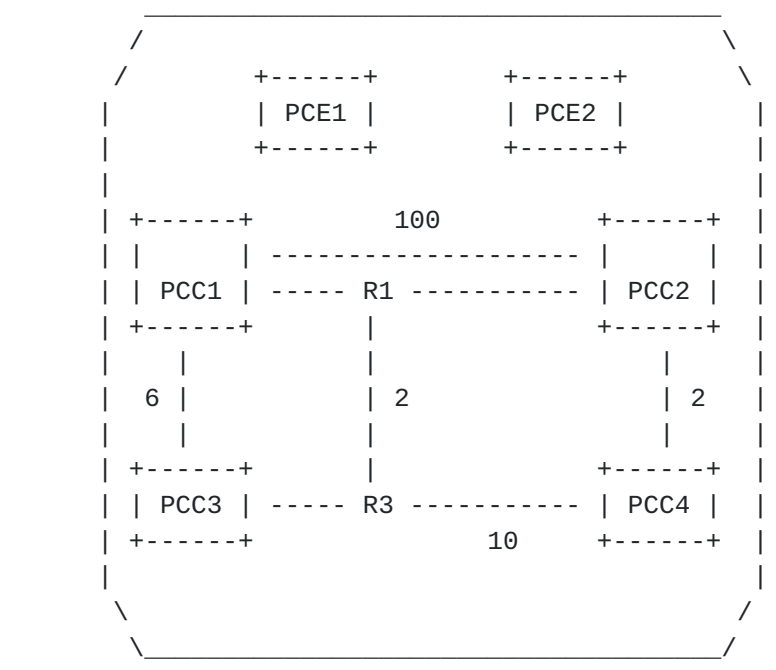
Scenario 4:




With this new PCEP session topology, we consider that PCEs are configured to fallback to shortest path if disjointness cannot be found. We first configure PCC1->PCC2, PCE1 computes shortest path as it is the only LSP in the disjoint-group that it is aware of: R1->R3->R4->R2->PCC2 (shortest path). When PCC3->PCC4 is configured, PCE2 must compute a disjoint path for this LSP. The only solution found is to move PCC1->PCC2 LSP on another path, but PCE2 cannot do it as it does not have delegation for this LSP. PCE2 then provides

shortest path for PCC3->PCC4: R3->R4->PCC4. When PCC3 receives the ERO, it reports it back to both PCEs. When PCE1 becomes aware of PCC3->PCC4 path, it recomputes the CSPF and provides a new path for PCC1->PCC2: R1->R2->PCC2. The new path is reported back to all PCEs by PCC1. PCE2 recomputes also CSPF to take into account the new reported path. The new computation does not lead to any path update.

Scenario 5:

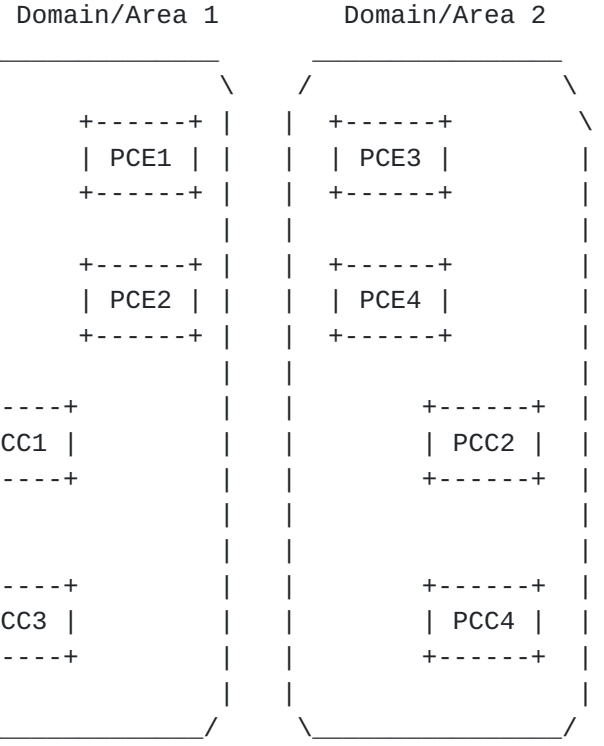



Now we consider a new network topology with the same PCEP session topology as the previous example. We configure both LSPs almost at the same time. PCE1 will compute a path for PCC1->PCC2 while PCE2 will compute a path for PCC3->PCC4. As each other is not aware of the path of the second LSP in the group (not reported yet), each PCE is computing shortest path for the LSP. PCE1 computes ERO: R1->PCC2 for PCC1->PCC2 and PCE2 computes ERO: R3->R1->PCC2->PCC4 for PCC3->PCC4. When these shortest paths will be reported to each PCE. Each PCE will recompute disjointness. PCE1 will provide a new path for PCC1->PCC2 with ERO: PCC1->PCC2. PCE2 will provide also a new path for PCC3->PCC4 with ERO: R3->PCC4. When those new paths will be reported to both PCEs, this will trigger CSPF again. PCE1 will provide a new more optimal path for PCC1->PCC2 with ERO: R1->PCC2 and PCE2 will also provide a more optimal path for PCC3->PCC4 with ERO: R3->R1->PCC2->PCC4. So we come back to the initial state. When those paths will be reported to both PCEs, this will trigger CSPF

again. An infinite loop of CSPF computation is then happening with a permanent flap of paths because of the split-brain situation.

This permanent computation loop comes from the inconsistency between the state of the LSPs as seen by each PCE due to the split-brain: each PCE is trying to modify at the same time its delegated path based on the last received path information which defacto invalidates this receives path information.

Scenario 6: multi-domain




In the example above, we want to create disjoint LSPs from PCC1 to PCC2 and from PCC4 to PCC3. All the PCEs have the knowledge of both domain topologies (e.g. using BGP-LS). For operation/management reason, each domain uses its own group of redundant PCEs. PCE1/PCE2 in domain 1 have PCEP sessions with PCC1 and PCC3 while PCE3/PCE4 in domain 2 have PCEP sessions with PCC2 and PCC4. As PCE1/2 do not know about LSPs from PCC2/4 and PCE3/4 do not know about LSPs from PCC1/3, there is no possibility to compute the disjointness constraint. This scenario can also be seen as a split-brain scenario. This multi-domain architecture (with multiple groups of PCEs) can also be used in a single domain, where an operator wants to limit the failure domain by creating multiple groups of PCEs maintaining a subset of PCCs. As for the multi-domain example, there

will be no possibility to compute disjoint path starting from head-ends managed by different PCE groups.

In this document, we will propose a solution that address the possibility to compute LSP association based constraints (like disjointness) in split-brain scenarios while preventing computation loops.

## 2. Proposed solution

Our solution is based on :

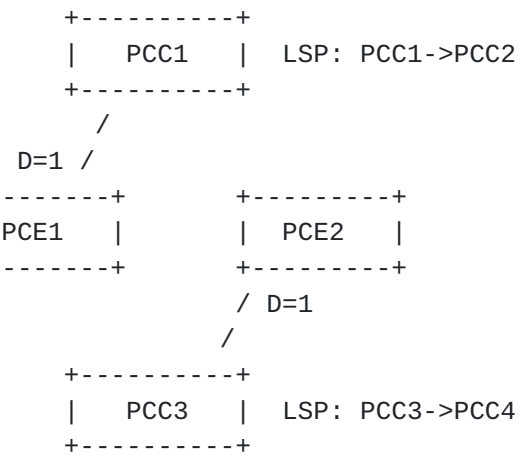
- The creation of the inter-PCE stateful PCEP session.
- A master/slave mechanism between PCEs.

### 2.1. State-sync session

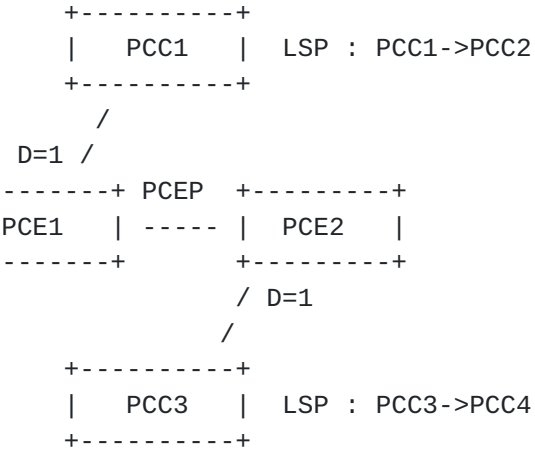
We propose to create a PCEP session between the PCEs. Creating such session is already authorized by multiple scenarios like the one described in [RFC4655] (multiple PCEs that are handling part of the path computation) and [RFC6805] (hierarchical PCE) but was mainly focused on stateless PCEP sessions. As stateful PCE brings additional features (LSP state synchronization, path update ...), some new behaviors need to be defined.

This inter-PCE PCEP session will allow exchange of LSP states between PCEs that would help some scenario where PCEP sessions are lost between PCC and PCE. This inter-PCE PCEP session is called a state-sync session.

For example, in the scenario below, there is no possibility to compute disjointness as there is no PCE aware of both LSPs.



If we add a state-sync session, PCE1 will be able to forward some PCRpt for its LSP to PCE2 and PCE2 will do the same. All the PCEs will be aware of all LSPs even if PCC->PCE session are down. PCEs will then be able to compute disjoint paths.




The procedures associated with this state-sync session are defined in Section 3.

Adding this state-sync session does not ensure that path with LSP association based constraints can always been computed and does not prevent computation loop, but it increases resiliency and ensures that PCEs will have the state information for all LSPs in the group.

## 2.2. Master/slave computation

As seen in Section 1, performing computation in a split-brain scenario (multiple PCEs responsible for computation) may provide less optimal LSP placement, no solution or computation loops. To provide the best efficiency, LSP association constraint based computation requires that a single PCE performs the path computation for all LSPs in the association group.

We propose to add a priority mechanism. Using this priority mechanism, PCEs can agree on the PCE that will be responsible for the computation for a particular association group, or set of LSPs.

When a single PCE is performing computation for a particular association group, no computation loop can happen. The other PCEs will only act as state collectors and forwarders.

In the scenario described in Section 2.1, PCE1 and PCE2 will decide that PCE1 will be responsible for the path computation of both LSPs. If we first configure PCC1->PCC2, PCE1 computes shortest path at it

is the only LSP in the disjoint-group that it is aware of: R1->R3->R4->R2->PCC2 (shortest path). When PCC3->PCC4 is configured, PCE2 will not perform computation even if it has delegation but forwards the PCRpt to PCE1 through the state-sync session. PCE1 will then perform disjointness computation and will move PCC1->PCC2 onto R1->R2->PCC2 and provides an ERO to PCE2 for PCC3->PCC4: R3->R4->PCC4.

# 3. Procedures and protocol extensions

## 3.1. Opening a state-sync session

### 3.1.1. TCP session opening

The state-sync session follows the initialization phase as described in [RFC5440], except that :

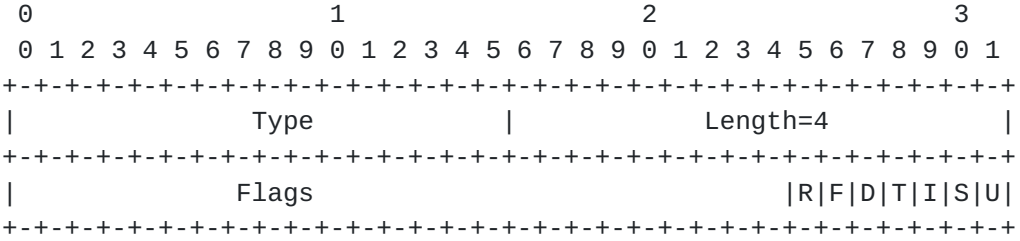
- Both PCEs are acting as TCP client and TCP servers.
- The TCP session that uses the highest IP client source address will be kept while the other will be rejected. This allows to solve TCP session setup collision.

When the TCP session is opened, initialization phase follows with OPEN message exchange as usual.

### 3.1.2. Capability advertisement

A PCE indicates its support of state-sync during the PCEP Initialization phase. The Open object in the Open message MUST contains the "Stateful PCE Capability" TLV defined in [I-D.ietf-pce-stateful-pce]. A new R (STATE-SYNC-CAPABILITY) flag is introduced to indicate the support of state-sync.

The format of the STATEFUL-PCE-CAPABILITY TLV is shown in the following figure:




This document only updates the Flags field with :

R (STATE-SYNC-CAPABILITY - 1 bit): If set to 1 by a PCEP Speaker, the PCEP speaker indicates that the session MUST follow the state-sync procedures. The R bit MUST be set by the two speakers: if a PCEP Speaker receives a STATEFUL-PCE-CAPABILITY TLV with R=0 while it advertised R=1, it MUST consider the Open parameters as unacceptable and follow the procedures described in [RFC5440].

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

The U flag MUST be set when sending the STATEFUL-PCE-CAPABILITY TLV with the R flag set. S flag can be set if optimized synchronization is required as per [I-D.ietf-pce-stateful-sync-optimizations].

## 3.2. State synchronization

When STATE-SYNC-CAPABILITY has been negotiated, each PCEP speaker will behave as a PCE and PCC at the same time regarding the state synchronization as defined in [I-D.ietf-pce-stateful-pce]. This means that each PCEP Speaker:

- MUST generate a PCReport message with S flag set for each LSP in its LSP database learned from a PCC and for which it has delegation and send it to its neighbor. (PCC role)

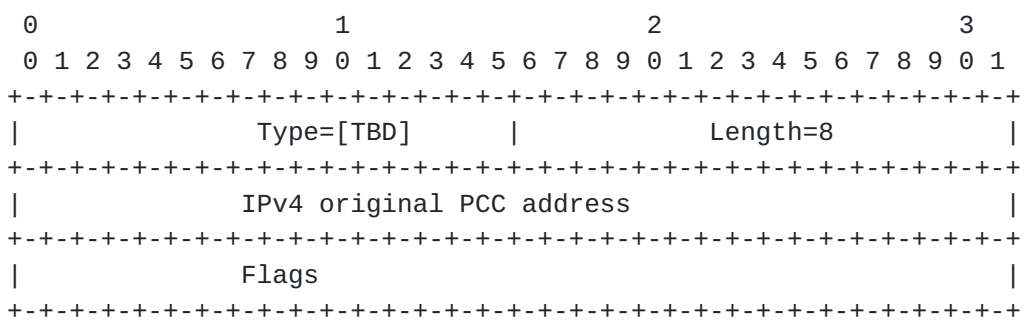
- MUST generate the End Of Synchronization Marker when all delegated LSPs have been reported and send it to its neighbor. (PCC role)

- MUST wait for LSP states with S=1 and the End Of Synchronization Marker from its neighbor. (PCE role)

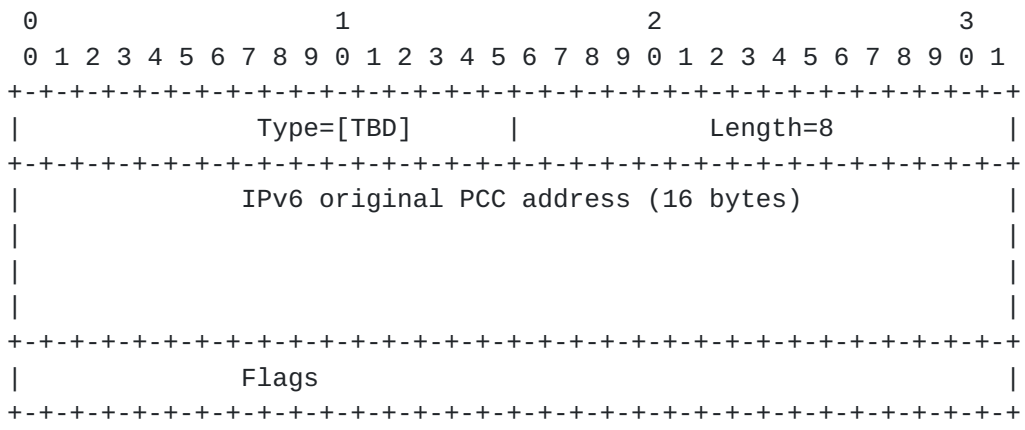
Optimized synchronization can be used as defined in [I-D.ietf-pce-stateful-sync-optimizations].

Two new TLVs are added in the LSP Object ([I-D.ietf-pce-stateful-pce]). These TLV are called IPV4-STANDBY-SYNC-TLV, IPV6-STANDBY-SYNC-TLV and have the following format :

IPV4-STANDBY-SYNC-TLV :




IPV6-STANDBY-SYNC-TLV :




IPv[4|6] original PCC address: MUST be set to the address of the PCC the LSP state was learned from.

Flags: reserved, MUST be set to 0 on transmission and MUST be ignored on receipt.

When a PCEP Speaker sends a PCReport on a state-sync session, it MUST add the IPV4-STANDBY-SYNC-TLV or IPV6-STANDBY-SYNC-TLV. If a PCEP Speaker receives a PCReport on a state-sync session without those TLVs, it MUST discard the PCReport and it MUST reply with a PCError message using error type 6 (Mandatory Object missing) and error-value [TBD] (STANDBY-SYNC-TLV missing).

### 3.3. Maintaining LSP states from different sources

When a PCEP Speaker receives a PCReport on a state-sync session, it MUST store the LSP information into the original PCC address context (as the LSP belongs to the PCC, not the neighboring PCE that reported it).

A PCEP Speaker may receive a state information for a particular LSP from different sources : the PCC that owns the LSP (through a regular PCEP session) and one other PCE (through PCEP state-sync session). A PCEP Speaker MUST always keep the last received state information in its LSP database, overriding the previously received information. For example, a PCE first receives a report for a LSP1 from a PCC, and it then receives a report for LSP1 through a PCEP state-sync session. The last information received from the state-sync session will so override the state that was previously received from the PCC.

### 3.4. Incremental updates and report forwarding rules

During the life of an LSP, its state may change (path, constraints, operational state ...) and a PCC will advertise a new PCReport to the PCE.

When a PCE receives a new PCReport from a PCC with D flag set for an existing LSP, if the LSP state information has changed compared to the previous information, the PCE MUST forward the PCReport to all its state-sync sessions and MUST add the appropriate STANDBY-SYNC-TLV in the PCReport.

When a PCE receives a new PCReport from a PCC with D flag unset for a previously delegated LSP, the PCE MUST forward the PCReport to all its state-sync sessions with the R flag set (Remove) and MUST add the appropriate STANDBY-SYNC-TLV in the PCReport.

When a PCE receives a new PCReport from a PCC with R flag set for delegated LSP, the PCE MUST forward the PCReport to all its state-sync sessions keeping the R flag set (Remove) and MUST add the appropriate STANDBY-SYNC-TLV in the PCReport.

When a PCE receives a PCReport from a state-sync session, it MUST NOT forward the PCReport to other state-sync sessions.

When a PCReport is forwarded, all the original objects and values are kept. As an example, the PLSP-ID used in the forwarded PCReport will be the same as the original one used by the PCC.

### 3.5. Computation priority between PCEs and sub-delegation

Computation priority is necessary to ensure that a single PCE will perform the computation for all the LSPs in an association group: this will allow for more optimized LSP placement and computation loop prevention mechanism.

All PCEs in the network that are handling LSPs in a common LSP association group SHOULD be aware of each other including the

computation priority of each PCE. The computation priority is a number (1 byte) and the PCE having the highest priority SHOULD be responsible for the computation. If several PCEs have the same priority value, their IP address SHOULD be used as a tie-breaker to rank PCEs: highest IP address as more priority. How PCEs are aware of the priority of each other is out of scope of this document, but as example learning priorities could be done through IGP informations or local configuration.

The definition of the priority MAY be global so the highest priority PCE will handle all path computations or more granular, so a PCE may have highest priority for only a subset of LSPs or association-groups.

A PCEP Speaker receiving a PCReport from a PCC with D flag set that does not have the highest computation priority, SHOULD forward the PCReport on all state-sync sessions (as per Section 3.4) and SHOULD set D flag on the state-sync session towards the highest priority PCE, D flag will be unset to all other state-sync sessions. This behavior is similar to the delegation behavior handled at PCC side and can be seen as a sub-delegation. When a PCEP Speaker sub-delegates a LSP to another PCE, it looses the control on the LSP and cannot update it anymore by its own decision.

If the highest priority PCE is failing or if the state-sync session between the local PCE and the highest priority PCE failed, the local PCE MAY decide to delegate the LSP to the next highest priority PCE or to take back control on the LSP.

When a PCE receives a PCReport with D flag set on a state-sync session, as a regular PCE, it is authorized to update the LSP. When it needs to update the LSP, it sends a PCUpdate on the state-sync session towards the PCE that sub-delegated the LSP. The LSP Object in the PCUpdate message MUST contain the STANDBY-SYNC-TLV (IPv4 or IPv6). If a PCE receives a PCUpdate on a state-sync session without the STANDBY-SYNC-TLV, it MUST discard the PCUpdate and MUST reply with a PCError message using error type 6 (Mandatory Object missing) and error-value [TBD] (STANDBY-SYNC-TLV missing).

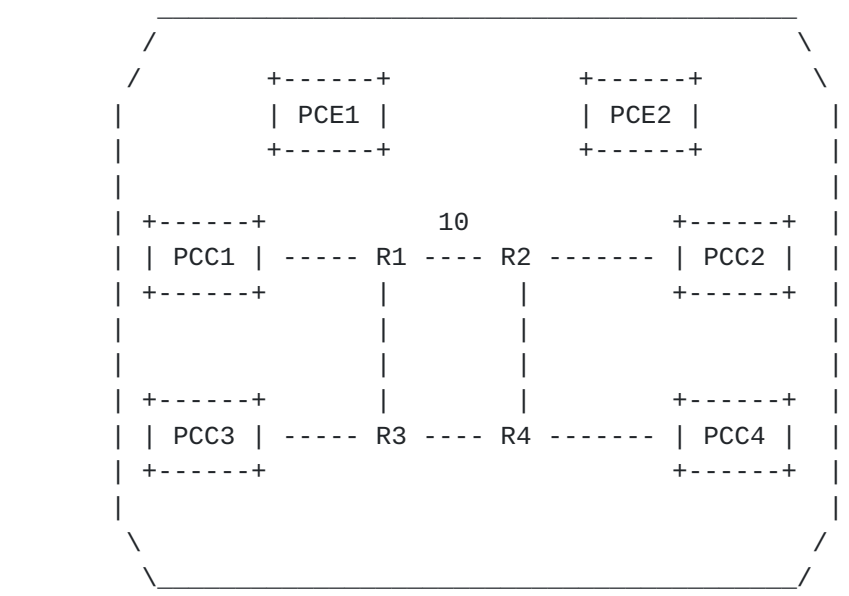
When a PCE receives a valid PCUpdate on a state-sync session, it SHOULD forward the PCUpdate to the appropriate PCC that delegated the LSP originally and SHOULD remove the STANDBY-SYNC-TLV from the LSP Object. Acknowlegment of the PCUpdate is done through a cascaded mechanism, and the original PCC is the only responsible of triggering the acknowledgment: when the PCC receives the PCUpdate from the local PCE, it acknowledges it with a PCReport as per [I-D.ietf-pce-stateful-pce]. When receiving the new PCReport from the PCC, the local PCE uses the defined forwarding rules on the

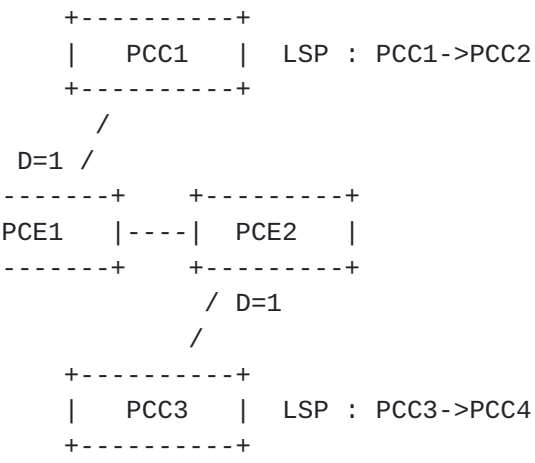
state-sync session so the acknowledgment is relayed to the computing PCE.

A PCE SHOULD NOT compute a path using an association-group constraint if it has delegation for only a subset of LSPs in the group. In this case, an implementation MAY use a local policy on PCE to decide if PCE does not compute path at all for this set of LSP or if it can compute a path by relaxing the association-group constraint.

## 4. Examples

### 4.1. Example 1




PCE1 computation priority 100
PCE2 computation priority 200

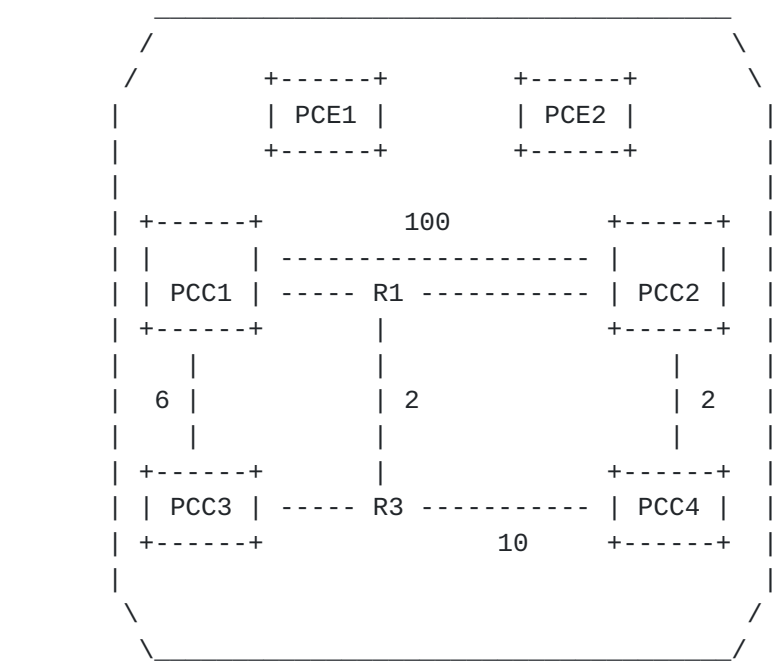
With this PCEP session topology, we still want to have link disjoint LSPs PCC1->PCC2 and PCC3->PCC4.

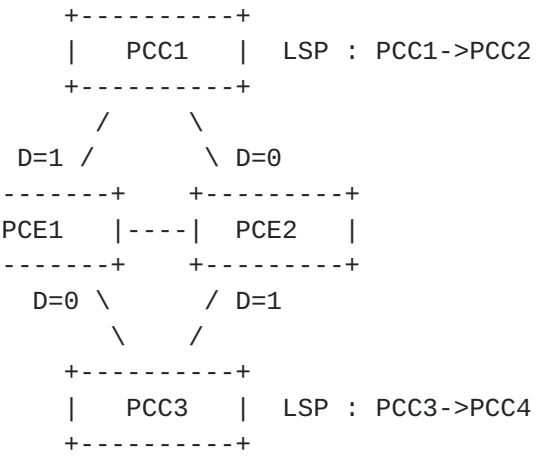
We first configure PCC1->PCC2, PCC1 delegates the LSP to PCE1, but as PCE1 does not have the highest computation priority, it will sub-delegate the LSP to PCE2 by sending a PCReport with D=1 and including the STANDBY-SYNC-TLV on the state-sync session. PCE2 receives the PCReport and as it has delegation for this LSP, it computes shortest path: R1->R3->R4->R2->PCC2. It then sends a PCUpdate to PCE1 (including the STANDBY-SYNC-TLV) with the computed ERO. PCE1 forwards the PCUpdate to PCC1 (removing the STANDBY-SYNC-TLV). PCC1 acknowledges the PCUpdate by a PCReport to PCE1. PCE1 forwards the PCReport to PCE2.

When PCC3->PCC4 is configured, PCC3 delegates the LSP to PCE2, PCE2 can compute a disjoint path as it has knowledge of both LSPs and has delegation also for both. The only solution found is to move PCC1->PCC2 LSP on another path, PCE2 can move PCC3->PCC4 as it has delegation for it. It creates a new PCUpdate with new ERO: R1->R2-PCC2 towards PCE1 which forwards to PCC1. PCE2 sends a PCUpdate to PCC3 with the path: R3->R4->PCC4.

In this setup, PCEs are able to find a disjoint path while without state-sync and computation priority they could not.

### 4.2. Example 2

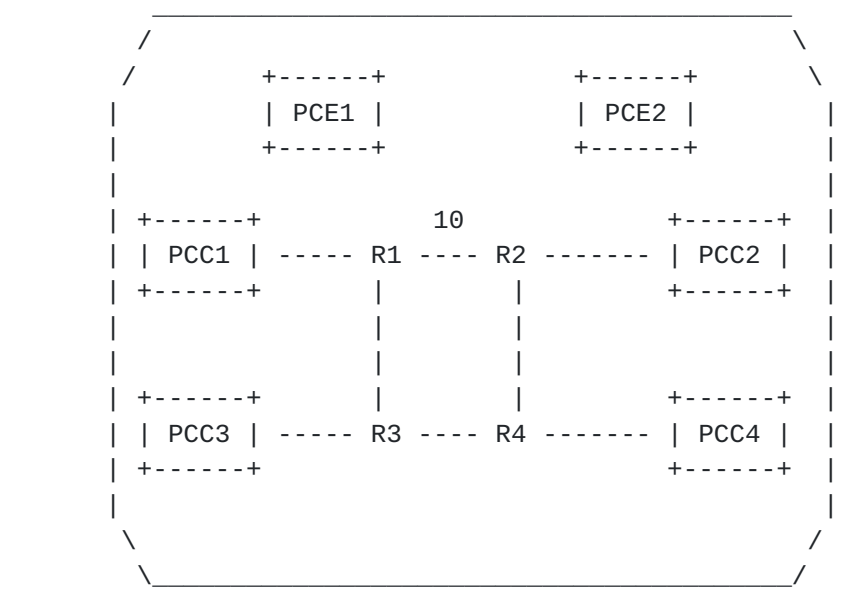


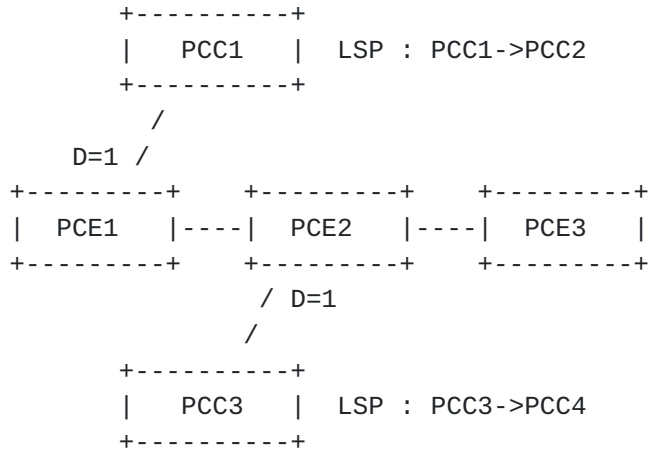




PCE1 computation priority 200
PCE2 computation priority 100

In this example, we configure both LSPs almost at the same time. PCE1 sub-delegates PCC1->PCC2 to PCE2 while PCE2 keeps delegation for PCC3->PCC4, PCE2 computes a path for PCC1->PCC2 and PCC3->PCC4 and can achieve disjointness computation easily. No computation loop happens in this case.

### 4.3. Example 3




PCE1 computation priority 100
PCE2 computation priority 200
PCE2 computation priority 300

With this PCEP session topology, we still want to have link disjoint LSPs PCC1->PCC2 and PCC3->PCC4.

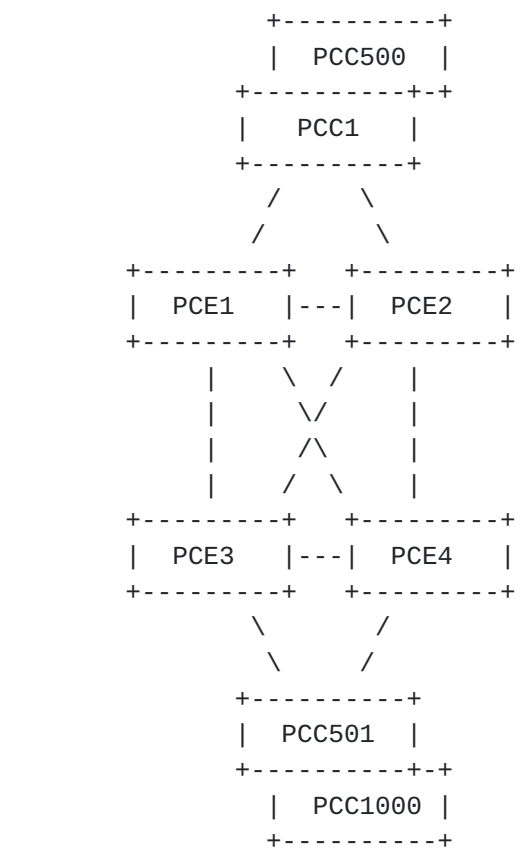
We first configure PCC1->PCC2, PCC1 delegates the LSP to PCE1, but as PCE1 does not have the highest computation priority, it will sub-delegate the LSP to PCE2 (as it cannot reach PCE3 through a state-sync session). PCE2 cannot compute a path for PCC1->PCC2 as it does not have the highest priority and cannot sub-delegate the LSP again towards PCE3.

When PCC3->PCC4 is configured, PCC3 delegates the LSP to PCE2 that performs sub-delegation to PCE3. As PCE3 will have knowledge of only one LSP in the group, it cannot compute disjointness and can decide to fallback to a less constrained computation to provide a path for PCC3->PCC4. In this case, it will send a PCUpdate to PCE2 that will be forwarded to PCC3.

Disjointness cannot be achieved in this scenario because of lack of state-sync session between PCE1 and PCE3, but no computation loop happens.

## 5. Using master/slave computation and state-sync sessions to increase scaling

The master/slave computation and state-sync sessions architecture can be used to increase the scaling of the PCE architecture. If the number of PCCs is really high, it may be too resource consuming for a single PCE to maintain all the PCEP sessions while at the same time performing all path computations. Using master/slave computation and state-sync sessions may allow to create groups of PCEs that manage a subset of the PCCs and some or no path computations. Decoupling PCEP session maintenance and computation will allow to increase scaling of the PCE architecture.




In the figure above, two groups of PCEs are created: PCE1/2 maintain PCEP sessions with PCC1 up to PCC500, while PCE3/4 maintain PCEP sessions with PCC501 up to PCC1000. A granular master/slave policy is setup as follows to loadshare computation between PCEs:

- PCE1 has priority 200 for association ID 1 up to 300, association source 0.0.0.0. All other PCEs have a decreasing priority for those associations.

- PCE3 has priority 200 for association ID 301 up to 500, association source 0.0.0.0. All other PCEs have a decreasing priority for those associations.

If some PCCs delegate LSPs with association ID 1 up to 300 and association source 0.0.0.0, the receiving PCE (if not PCE1) will sub-delegate the LSPs to PCE1. PCE1 becomes responsible for the computation of these LSP associations while PCE3 is responsible for the computation of another set of associations.

## 6. Security Considerations

TBD.

## 7. Acknowledgements

TBD.

## 8. IANA Considerations

This document requests IANA actions to allocate code points for the protocol elements defined in this document.

### 8.1. PCEP-Error Object

IANA is requested to allocate a new Error Value for the Error Type 9.

| Error-Type | Meaning | Reference |
|---|---|---|
| 6 | Mandatory Object Missing | [RFC5440] |
| | Error-value=TBD: STANDBY-SYNC-TLV missing | This document |

### 8.2. STATEFUL-PCE-CAPABILITY TLV

IANA is requested to allocate a new bit value in the STATEFUL-PCE-CAPABILITY TLV Flag Field sub-registry.

| Bit | Description | Reference |
|---|---|---|
| TBD(suggested value 25) | STATE-SYNC-CAPABILITY | This document |

### 8.3. PCEP TLV Type Indicators

IANA is requested to allocate two new TLV Type Indicator values within the "PCEP TLV Type Indicators" sub-registry of the PCEP Numbers registry, as follows:

| Value | Description | Reference |
|---|---|---|
| TBD | IPV4-STANDBY-SYNC-TLV | This document |
| TBD | IPV6-STANDBY-SYNC-TLV | This document |

## 9. References

### 9.1. Normative References

[I-D.ietf-pce-stateful-pce]
Crabbe, E., Minei, I., Medved, J., and R. Varga, "PCEP Extensions for Stateful PCE", draft-ietf-pce-stateful-pce-16 (work in progress), September 2016.

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

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

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

## 9.2. Informative References

[I-D.ietf-pce-stateful-sync-optimizations] Crabbe, E., Minei, I., Medved, J., Varga, R., Zhang, X., and D. Dhody, "Optimizations of Label Switched Path State Synchronization Procedures for a Stateful PCE", draft-ietf-pce-stateful-sync-optimizations-06 (work in progress), October 2016.

[RFC6805] King, D., Ed. and A. Farrel, Ed., "The Application of the Path Computation Element Architecture to the Determination of a Sequence of Domains in MPLS and GMPLS", RFC 6805, DOI 10.17487/RFC6805, November 2012, <http://www.rfc-editor.org/info/rfc6805>.

## Authors' Addresses

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Siva Sivabalan
Cisco

Email: msiva@cisco.com