PCE Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 1, 2017

S. Litkowski
Orange
S. Sivabalan
Cisco
D. Dhody
Huawei

February 28, 2017

Inter Stateful Path Computation Element communication procedures
draft-litkowski-pce-state-sync-01

Abstract

The Path Computation Element Communication Protocol (PCEP) provides mechanisms for Path Computation Elements (PCEs) to perform path computations in response to Path Computation Clients (PCCs) requests. The stateful PCE extensions allow stateful control of Multi-Protocol Label Switching (MPLS) Traffic Engineering Label Switched Paths (TE LSPs) using PCEP.

A Path Computation Client (PCC) can synchronize an LSP state information to a Stateful Path Computation Element (PCE). The stateful PCE extension allows a redundancy scenario 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. The document does not state the procedures related to an inter-PCE stateful communication.

There are some use cases, where an inter-PCE stateful communication can bring additional resiliency in the design for instance when some PCC-PCE sessions fails. The inter-PCE stateful communication may also provide a faster update of the LSP states when an event occurs. Finally, when, in a redundant PCE scenario, there is a need to compute a set of paths that are part of a group (so there is a dependency between the paths), there may be some cases where the computation of all paths in the group is not handled by the same PCE: this situation is called a split-brain. This split-brain scenario may lead to computation loops between PCEs or suboptimal paths computation.

This document describes the procedures to allow a stateful communication between PCEs for various use-cases and also the procedures to prevent computations loops.

Requirements Language

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

Status of This Memo

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

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

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

This Internet-Draft will expire on September 1, 2017.

Copyright Notice

Table of Contents

1. Introduction and problem statement

1.1. Reporting LSP changes

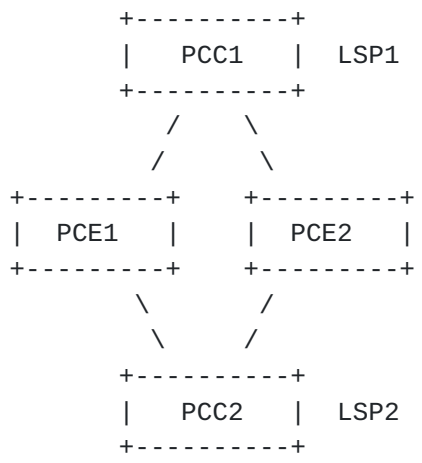
When using a stateful PCE ([I-D.ietf-pce-stateful-pce]), a Path Computation Client (PCC) can synchronize an LSP state information to the stateful Path Computation Element (PCE). If the PCC grants the control on the LSP to the PCE, the PCE can update the LSP parameters at any time.

In a multi PCE deployment (redundancy, loadbalancing...), with the current specification defined in [I-D.ietf-pce-stateful-pce], the PCC will be in charge of reporting the other PCEs of the LSP parameter change which brings additional hops and delays in notifying the overall network of the LSP parameter change.

This delay may affect the reaction time of the other PCEs, if they need to take action after being notified of the LSP parameter change.

Apart from the synchronization from the PCC, it is also useful if there is synchronization mechanism between the stateful PCEs. As stateful PCE make changes to its delegated LSPs, these changes

(pending LSPs and the sticky resources [RFC7399]) can be synchronized immediately to the other PCEs.



In the figure above, we consider a loadbalanced PCE architecture, so PCE1 is responsible to compute paths for PCC1 and PCE2 is responsible to compute paths for PCC2. When PCE1 triggers an LSP update for LSP1, it sends a PCUpdate message to PCC1 for LSP1 containing the new parameters. PCC1 will take the parameters into account and will send a PCReport to PCE1 and PCE2 reflecting the changes. PCE2 will so be notified of the change only after receiving the PCReport from PCC1.

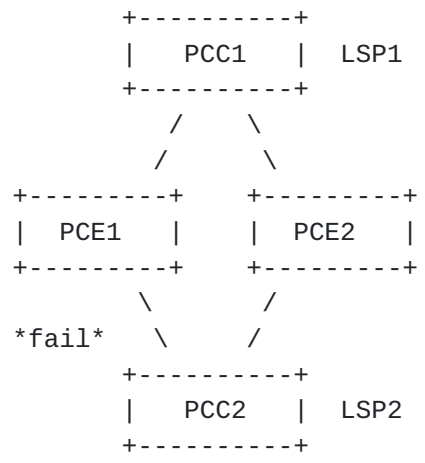
Let's consider that the LSP1 parameters changed in a such way that LSP1 will take over ressources from LSP2 with an higher priority. After receiving the report from PCC1, PCE2 will so try to find a new path for LSP2. If we consider that there is a round trip delay of about 150msec between the PCEs and PCC1 and a round trip delay of 10msec between the two PCEs, if will take more than 150msec for PCE2 to be notified of the change.

Adding a PCEP session between PCE1 and PCE2 may allow to reduce to the notification time, so PCE2 can react more quickly by taking the pending LSPs and sticky resources into account during path computation and reoptimization.

1.2. Split-brain

In a resiliency case, a PCC has redundant PCEP sessions towards multiple PCEs. In such a case, a PCC gives control on an LSP to a single PCE only, and only this PCE is responsible for the path computation for the delegated LSP: the PCC achieves this by setting the D flag only to the active PCE. The election of the active PCE to delegate an LSP is controlled by each PCC. The PCC usually elects the active PCE by a local configured policy (by setting a priority).

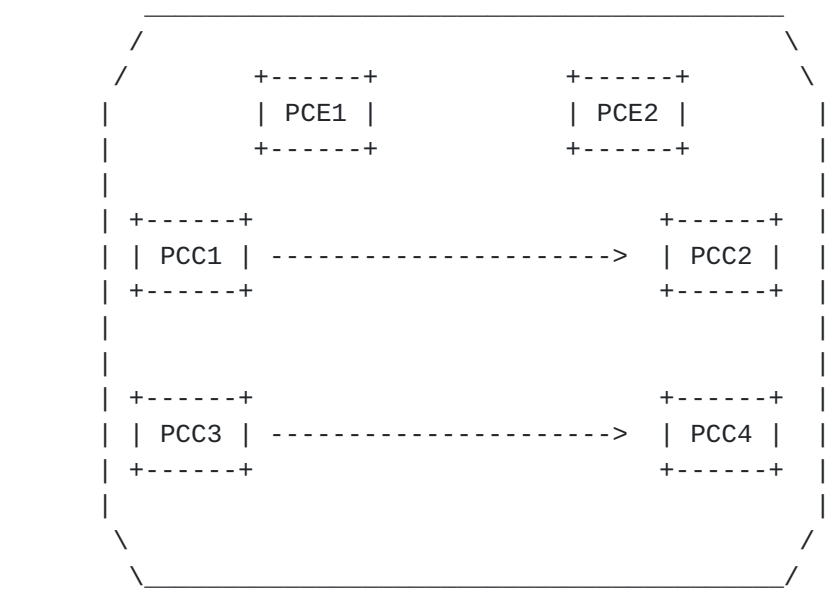
Upon PCEP session failure, or active PCE failure, PCC may decide to elect a new active PCE by sending new PCRpt message with D flag set to this new active PCE. When the failed PCE or PCEP session comes back online, it will be up to the vendor to implement preemption. Doing preemption may lead to some 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.

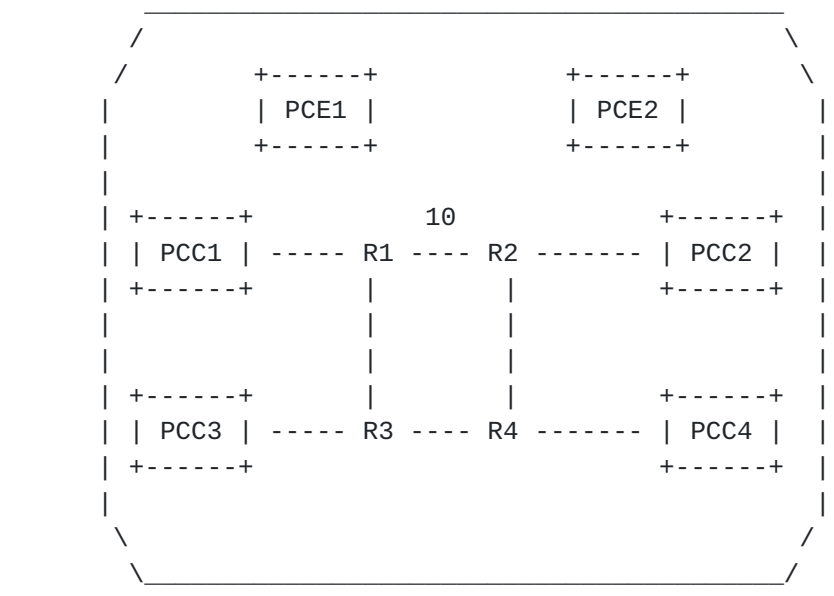


In the example above, we consider that by configuration, both PCCs will firstly delegate their LSP to PCE1. So PCE1 is responsible for computing a path for LSP1 and LSP2. If the PCEP session between PCC2 and PCE1 fails, PCC2 will delegate LSP2 to PCE2. So PCE1 becomes responsible only for LSP1 path computation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

Further, 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 (see [I-D.ietf-pce-association-diversity]). The set of LSPs that are dependant to each other may start from a different head-end.





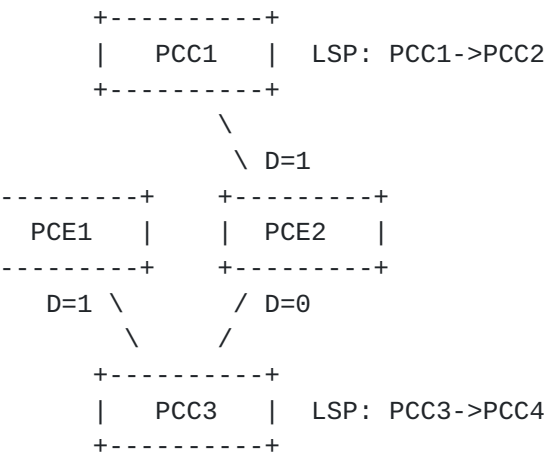
In the figure above, we want to create two link-disjoint LSPs: PCC1->PCC2 and PCC3->PCC4. In the topology, all link metrics are equal to 1 except the link R1-R2 which has a metric of 10. The PCEs are responsible for the path computation and PCE1 is the active PCE for all PCCs in the nominal case.

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.

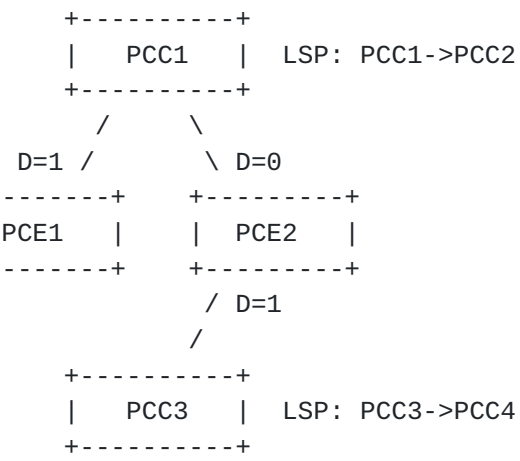
Scenario 2:

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



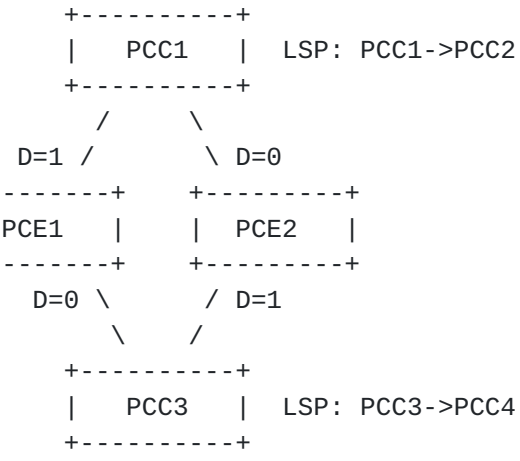
We first configure PCC1->PCC2 LSP, as the only constraint is path disjointness, PCE2 (which is the new active PCE for PCC1) sends a PCUpdate message to PCC1 with the ERO: R1->32->R4->R2->PCC2 (shortest path). When PCC3->PCC4 is configured, PCE1 is not aware anymore of LSPs from PCC1, so it cannot compute 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.

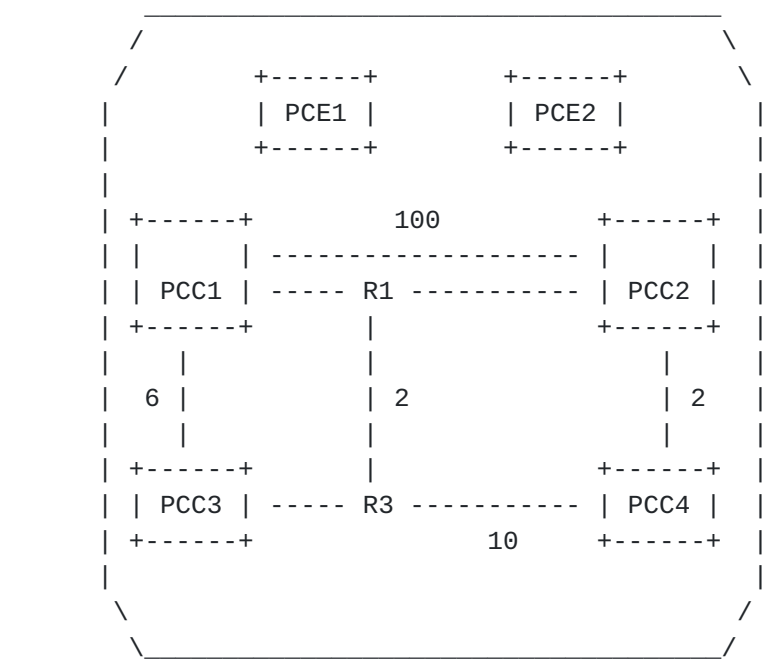
Scenario 4:



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:

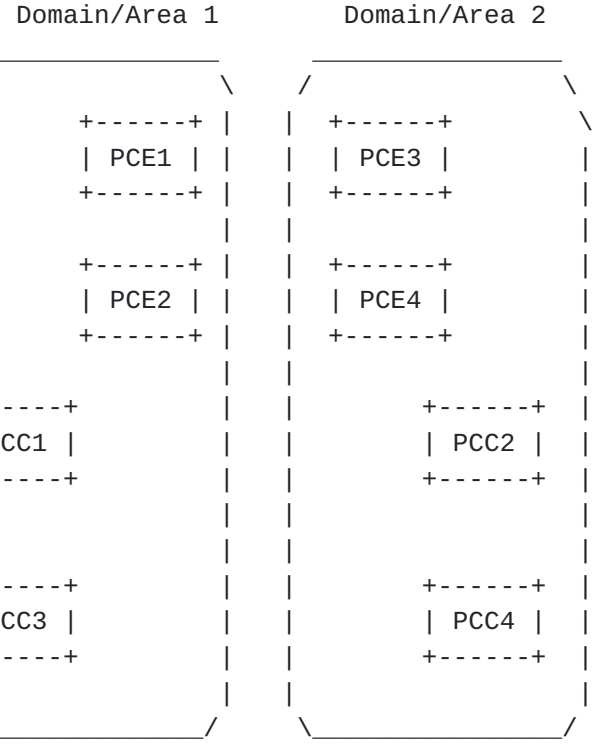


Now we consider a new network topology with the same PCEP session topology as the previous example. We configure both LSPs almost at the same time. PCE1 will compute a path for PCC1->PCC2 while PCE2 will compute a path for PCC3->PCC4. As each other is not aware of the path of the second LSP in the group (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.

1.3. Applicability to H-PCE

[I-D.dhodylee-pce-stateful-hpce] describes general considerations and use cases for the deployment of Stateful PCE(s) using the Hierarchical PCE [RFC6805] architecture. In this architecture there is a clear need to communicate between a child stateful PCE and a parent stateful PCE. The procedures and extensions as described in Section 3 are equally applicable to H-PCE.

2. Proposed solution

Our solution is based on :

- The creation of the inter-PCE stateful PCEP session with specific procedures.

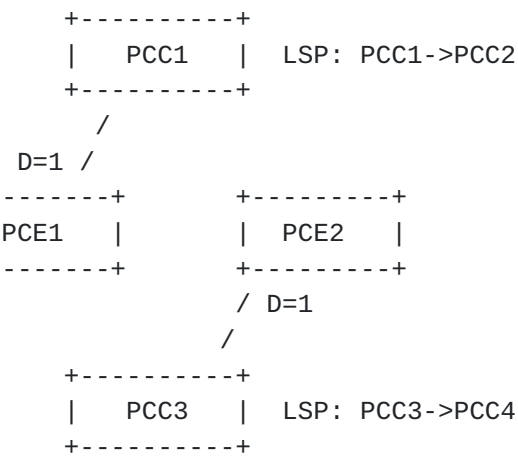
- A Master/Slave relationship between PCEs.

2.1. State-sync session

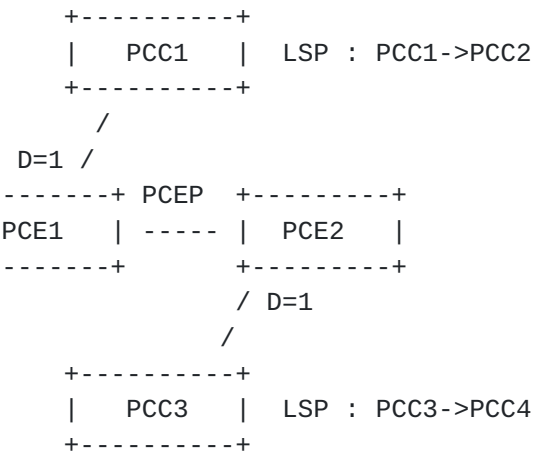
We propose to create a PCEP session between the stateful PCEs. Creating such session is already authorized by multiple scenarios like the one described in [RFC4655] (multiple PCEs that are handling part of the path computation) and [RFC6805] (hierarchical PCE) but was only focused on stateless PCEP sessions. As stateful PCE brings additional features (LSP state synchronization, path update ...), thus 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 send PCReport messages 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 a 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 addition, this session will allow for a PCE to update the other PCEs providing a faster synchronization mechanism than relying on PCCs only.

2.2. Master/Slave relationship between PCE

As seen in Section 1, performing a path computation in a split-brain scenario (multiple PCEs responsible for computation) may provide a non optimal LSP placement, no path or computation loops. To provide the best efficiency, an LSP association constraint based computation requires that a single PCE performs the path computation for all LSPs in the association group. Note that, it could be all LSPs belonging to a particular association group, or all LSPs from a particular PCC, or all LSPs in the network that need to be delegated to a single PCE based on the deployment scenarios.

We propose to add a priority mechanism between PCEs to elect a single computing PCE. 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. The priority could be set per association, per PCC, or for all LSPs. How this priority is set or advertised is out of scope of this document. The rest of the text consider association group as an example.

When a single PCE is performing the computation for a particular association group, no computation loop can happen and an optimal placement will be provided. 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.

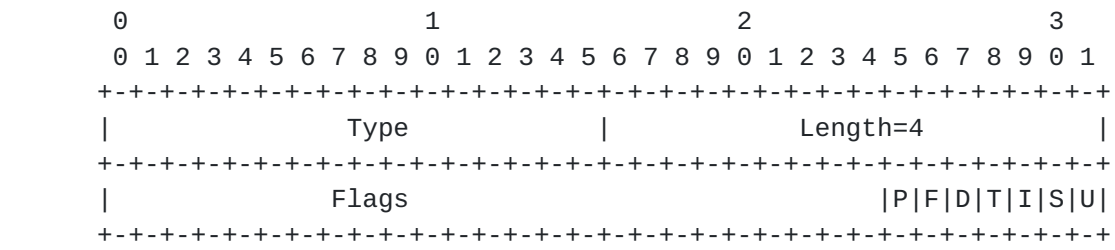
3. Procedures and protocol extensions

3.1. Opening a state-sync session

3.1.1. Capability advertisement

A PCE indicates its support of state-sync procedures 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 P (INTER-PCE-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 :

P (INTER-PCE-CAPABILITY - 1 bit): If set to 1 by a PCEP Speaker, the PCEP speaker indicates that the session MUST follow the state-sync procedures as described in this document. The P bit MUST be set by both speakers: if a PCEP Speaker receives a STATEFUL-PCE-CAPABILITY TLV with P=0 while it advertised P=1 or if both set P flag to 0, the session SHOULD open but the state-sync procedures MUST NOT be applied on this session.

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

3.2. State synchronization

When the INTER-PCE-CAPABILITY has been negotiated, each PCEP speaker will behave as a PCE and as a 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 send a PCRpt message towards its neighbor with S flag set for each LSP in its LSP database learned from a PCC. (PCC role)
- MUST send the End Of Synchronization Marker towards its neighbor when all LSPs have been reported. (PCC role)
- MUST wait for the LSP synchronization from its neighbor to end (receiving an End Of Synchronization Marker). (PCE role)

The process of synchronization runs in parallel on each PCE (no defined order).

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

When a PCEP Speaker sends a PCReport on a state-sync session, it MUST add the SPEAKER-IDENTITY-TLV (defined in [I-D.ietf-pce-stateful-sync-optimizations]) in the LSP Object, the value used will refer to the PCC owner of the LSP. If a PCEP Speaker receives a PCReport on a state-sync session without this TLV, it MUST discard the PCReport and it MUST reply with a PCErr message using error-type=6 (Mandatory Object missing) and error-value=TBD1 (SPEAKER-IDENTITY-TLV missing).

3.3. Maintaining LSP states from different sources

When a PCE receives a PCReport on a state-sync session, it stores the LSP information into the original PCC address context (as the LSP belongs to the PCC). A PCE SHOULD maintain a single state for a particular LSP.

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 some PCEs (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 an 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 must override the state that was previously received from the PCC.

The PCEP Speaker MUST track the list of sources it learned a particular LSP state from.

When it receives a PCReport requesting an LSP deletion from a particular source, it SHOULD remove this particular source from the list of sources associated with this LSP.

When the list of sources becomes empty for a particular LSP, the LSP state MUST be removed. This means that all the sources must send a PCRpt with R=1 for an LSP to make the PCE removing the LSP state.

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 for each such change.

When a PCE receives a new PCReport from a PCC, if the LSP state information has changed compared to the previous information (or if it is a new reported LSP), the PCE MUST forward the PCReport to all its state-sync sessions and MUST add the appropriate SPEAKER-IDENTITY-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 SPEAKER-IDENTITY-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. This helps to prevent message loops between PCEs. As a consequence, a full mesh of PCEP sessions between PCEs is required.

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. Thus an implementation supporting this document MUST consider SPEAKER-IDENTITY-TLV and PLSP-ID together to uniquely identify an LSP on the state-sync session.

3.5. Computation priority between PCEs and sub-delegation

A 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 a more optimized LSP placement and will prevent computation loops.

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. Note that there is no need for PCC to be aware of this. The computation priority is a number 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 provide a rank: the 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 is called a sub-delegation (the PCE subdelegates the control of

the LSP to another PCE). 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. When a PCE receives a PCReport with D flag set on a state-sync session, as a regular PCE, it becomes granted to update the LSP.

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. It is a local policy decision.

When a PCE has the delegation for an LSP and needs to update this LSP, it MUST send a PCUpdate message to all state-sync sessions and to the PCC session on which it received the delegation. The D-Flag would be unset in the PCUpdate for state-sync sessions where as D-Flag would be set for the PCC. In case of subdelegation, the computing PCE will send the PCUpdate only to all state-sync sessions (as it has no direct delegation from a PCC). The D-Flag would be set for the state-sync session to the PCE that sub-delegated this LSP and the D-Flag would be unset for other state-sync sessions.

The PCUpdate sent over a state-sync session MUST contain the SPEAKER-IDENTITY-TLV in the LSP Object (the value used must identify the target PCC). The PLSP-ID used is the original PLSP-ID generated by the PCC and learned from the forwarded PCReport. If a PCE receives a PCUpdate on a state-sync session without the SPEAKER-IDENTITY-TLV, it MUST discard the PCUpdate and MUST reply with a PCError message using error-type=6 (Mandatory Object missing) and error-value=TBD1 (SPEAKER-IDENTITY-TLV missing).

When a PCE receives a valid PCUpdate on a state-sync session, it SHOULD forward the PCUpdate to the appropriate PCC (identified based on the SPEAKER-IDENTITY-TLV value) that delegated the LSP originally and SHOULD remove the SPEAKER-IDENTITY-TLV from the LSP Object. The acknowlegment of the PCUpdate is done through a cascaded mechanism, and the 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.

3.6. Passive stateful procedures

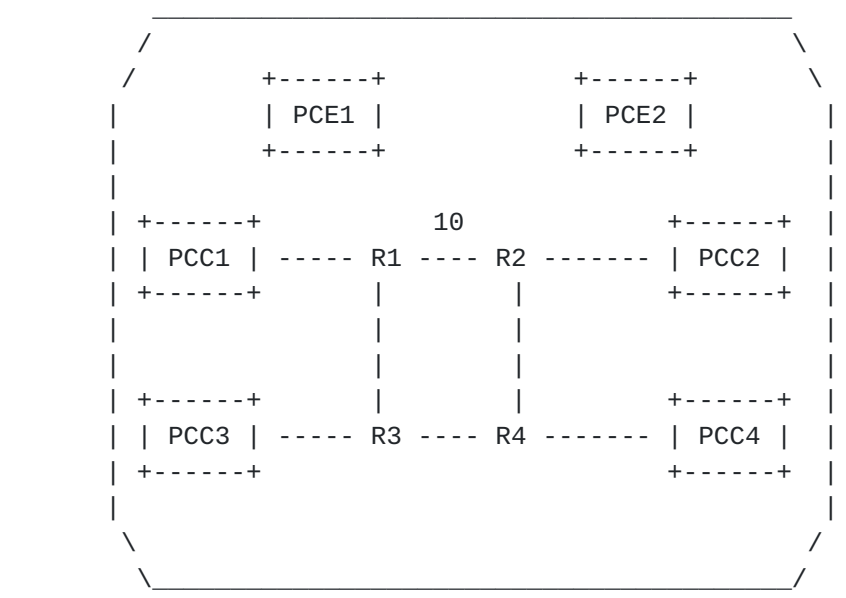
In the passive stateful PCE architecture, the PCC is responsible of triggering a path computation request using a PCRequest message to its PCE. Similarly to PCReports which remains unchanged for passive mode, if a PCE receives a PCRequest for an LSP and if this PCE finds that it does not have the highest computation priority of this LSP, or groups..., it MUST forward the PCRequest to the highest priority PCE over the state-sync session. When the highest priority PCE receives the PCRequest, it computes the path and generates a PCReply only to the PCE that is received the PCRequest from. This PCE will then forward the PCReply to the requesting PCC. The handling of LSP object and the SPEAKER-IDENTITY-TLV in PCRequest and PCReply is similar to PCReport/PCUpdate.

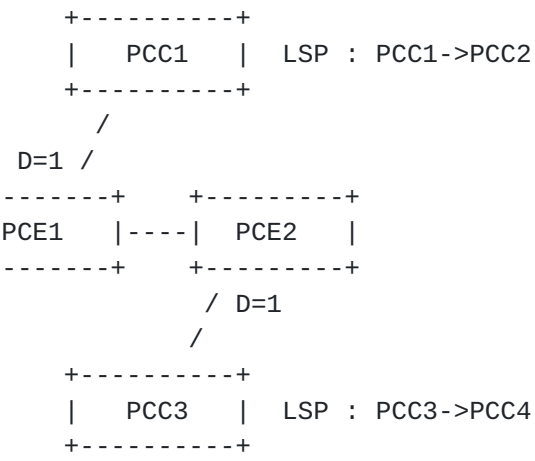
3.7. PCE initiation procedures

TBD

4. Examples

4.1. Example 1





PCE1 computation priority 100
PCE2 computation priority 200

With this PCEP session topology where computation priority is global for all LSPs, we still want to have link disjoint LSPs PCC1->PCC2 and PCC3->PCC4.

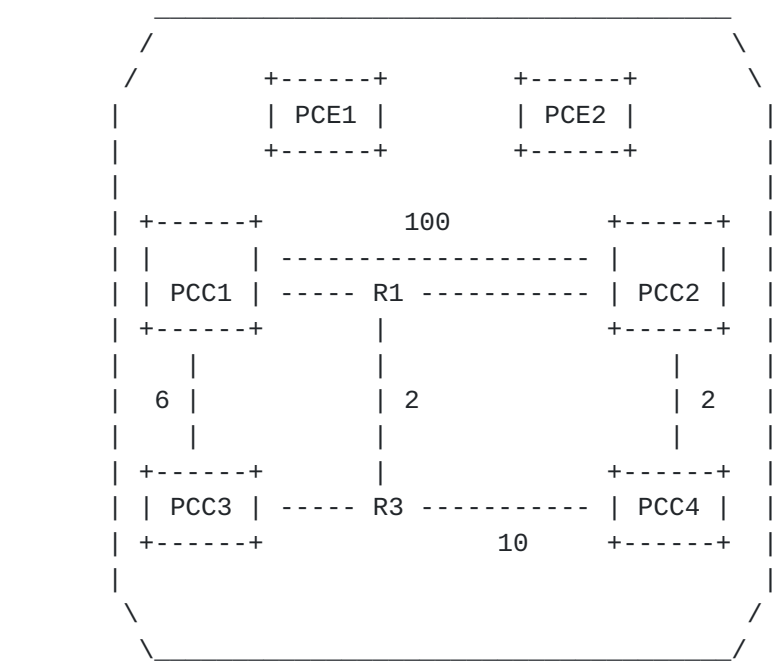
We first configure PCC1->PCC2, PCC1 delegates the LSP to PCE1, but as PCE1 does not have the highest computation priority, it will sub-delegate the LSP to PCE2 by sending a PCReport with D=1 and including the SPEAKER-IDENTITY-TLV over the state-sync session. PCE2 receives the PCReport and as it has delegation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

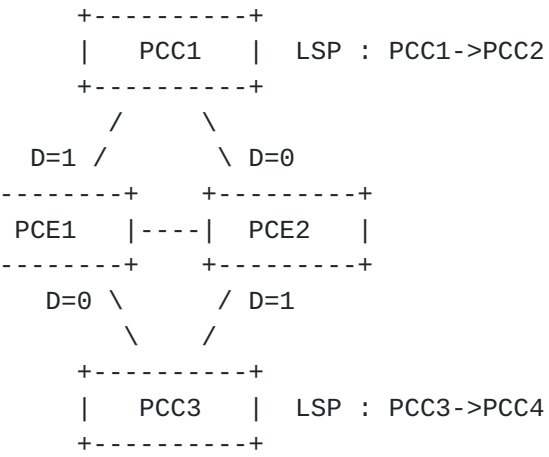
TLV). PCC1 acknowledges the PCUpdate by a PCReport to PCE1. PCE1 forwards the PCReport to PCE2.

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

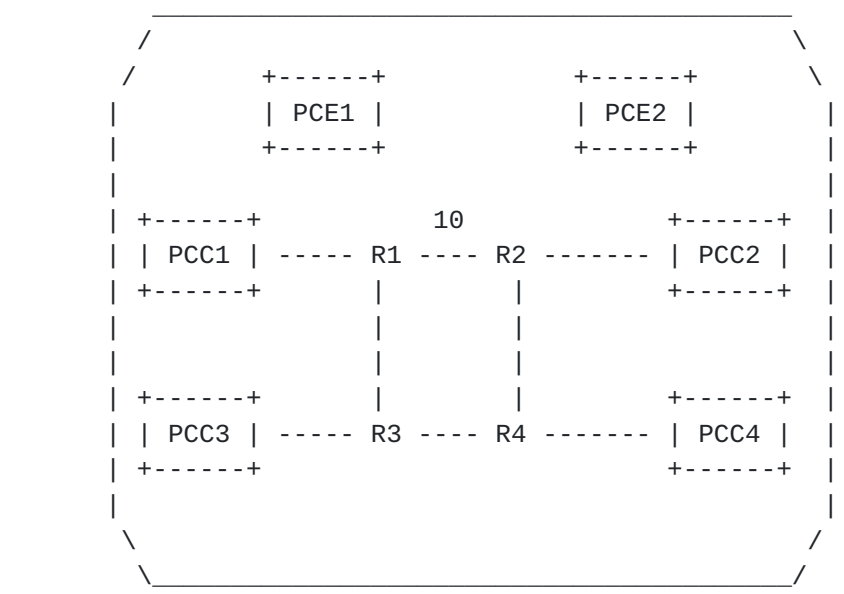


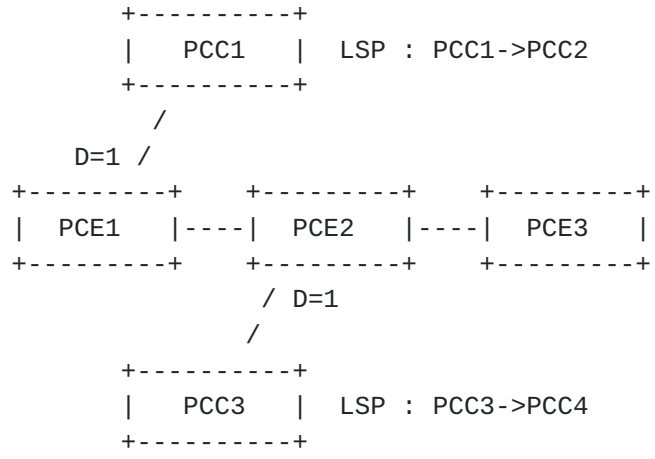


PCE1 computation priority 200
PCE2 computation priority 100

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.

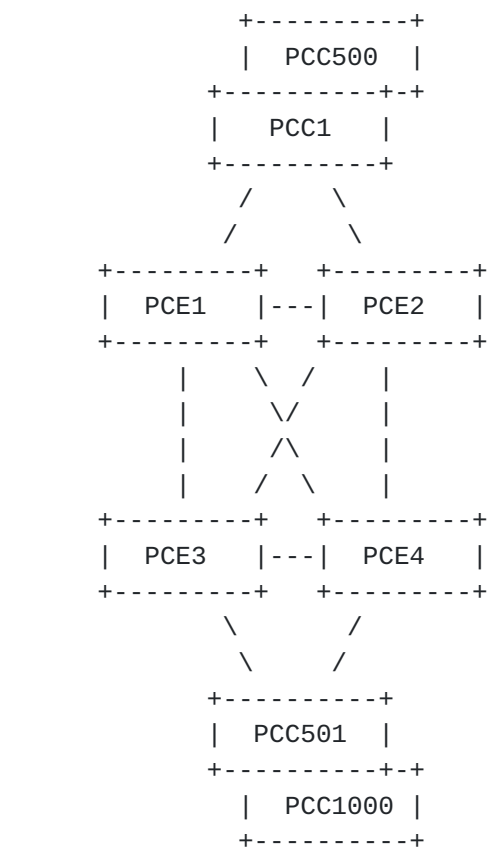
We first configure PCC1->PCC2, PCC1 delegates the LSP to PCE1, but as PCE1 does not have the highest computation priority, it will sub-delegate the LSP to PCE2 (as it cannot reach PCE3 through a state-sync session). PCE2 cannot compute a path for PCC1->PCC2 as it does not have the highest priority and 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. Thus it is advised for all PCEs that support state-sync to have a full mesh sessions between each other.

5. Using Master/Slave computation and state-sync sessions to increase scaling

The Primary/Backup 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 perform 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.

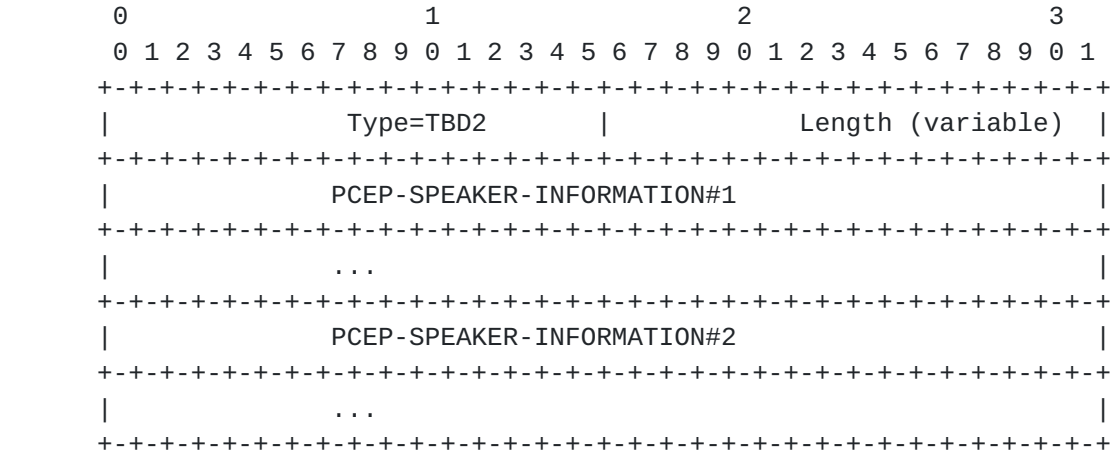
- PCE3 has priority 200 for association ID 301 up to 500, association source 0.0.0.0. All other PCEs have a decreasing priority for those associations.

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. PCEP-PATH-VECTOR-TLV

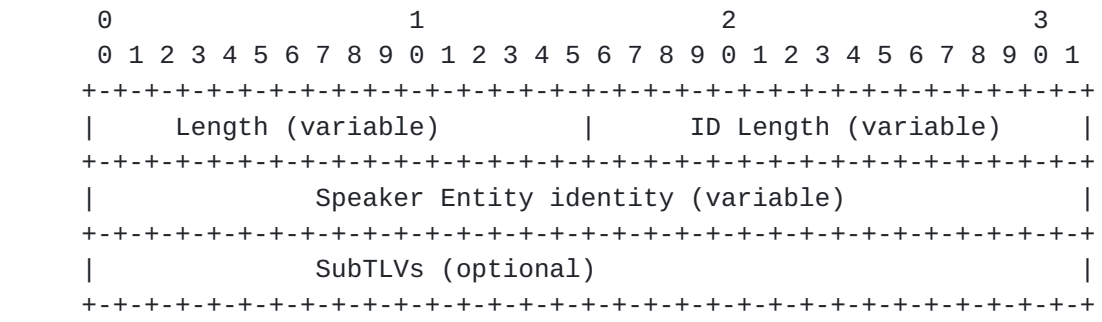
This document allows PCEP messages to be propagated among PCEP speaker. It may be useful to track informations about the propagation of the messages. One of the use case is a message loop detection mechanism, but other use cases like hop by hop information recording may also be implemented.

This document introduces the PCEP-PATH-VECTOR-TLV (type TBD2) with the following format:



The TLV format and padding rules are as per [RFC5440].

The PCEP-SPEAKER-INFORMATION field has the following format:



Length: defines the total length of the PCEP-SPEAKER-INFORMATION field.

ID Length: defines the length of the Speaker identity actual field (non-padded).

Speaker Entity identity: same possible values as the SPEAKER-IDENTIFIER-TLV. Padded with trailing zeroes to a 4-byte boundary.

The PCEP-SPEAKER-INFORMATION may also carry some optional subTLVs so each PCEP speaker can add local informations that could be recorded. This document does not define any subTLV.

The PCEP-PATH-VECTOR-TLV MAY be added in the LSP-Object. Its usage is purely optional.

The list of speakers within the PCEP-PATH-VECTOR-TLV MUST be ordered. When sending a PCEP message (PCReport, PCUpdate or PCInitiate), a PCEP Speaker MAY add the PCEP-PATH-VECTOR-TLV with a PCEP-SPEAKER-INFORMATION containing its own informations. If the PCEP message sent is the result of a previously received PCEP message, and if the PCEP-PATH-VECTOR-TLV was already present in the initial message, the PCEP speaker MAY append a new PCEP-SPEAKER-INFORMATION containing its own informations.

7. Security Considerations

TBD.

8. Acknowledgements

TBD.

9. IANA Considerations

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

9.1. 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=TBD1: SPEAKER-IDENTITY-TLV missing	This document

9.2. PCEP TLV Type Indicators

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

Value	Meaning	Reference
TBD2	PCEP-PATH-VECTOR-TLV	This document

9.3. 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	INTER-PCE-CAPABILITY	This document

10. References

10.1. 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-18 (work in progress), December 2016.

[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-08 (work in progress), January 2017.

[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>.

[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>.

10.2. Informative References

[I-D.dhodylee-pce-stateful-hpce]
Dhody, D., Lee, Y., Ceccarelli, D., Shin, J., King, D., and O. Dios, "Hierarchical Stateful Path Computation Element (PCE).", draft-dhodylee-pce-stateful-hpce-02 (work in progress), October 2016.

[I-D.ietf-pce-association-diversity]
Litkowski, S., Sivabalan, S., and C. Barth, "Path Computation Element communication Protocol extension for signaling LSP diversity constraint", draft-ietf-pce-association-diversity-00 (work in progress), January 2017.

[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>.

[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>.

[RFC7399] Farrel, A. and D. King, "Unanswered Questions in the Path Computation Element Architecture", RFC 7399, DOI 10.17487/RFC7399, October 2014, <http://www.rfc-editor.org/info/rfc7399>.

Authors' Addresses

Stephane Litkowski
Orange

Email: stephane.litkowski@orange.com

Siva Sivabalan
Cisco

Email: msiva@cisco.com

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

Email: dhruv.ietf@gmail.com