Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 1, 2021

H. Chen
M. McBride
Futurewei
Y. Fan
Casa Systems
M. Toy
Verizon
A. Wang
China Telecom
L. Liu
Fujitsu
X. Liu
Volta Networks

September 28, 2020

SRv6 Point-to-Multipoint Path
draft-chen-pim-srv6-p2mp-path-01

Abstract

This document describes a solution for a SRv6 Point-to-Multipoint (P2MP) Path/Tree to deliver the traffic from the ingress of the path to the multiple egresses/leaves of the path in a SR domain. There is no state stored in the core of the network for a SR P2MP path like a SR Point-to-Point (P2P) path in this solution.

Requirements Language

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

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 https://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 April 1, 2021.

Table of Contents

1. Introduction

The Segment Routing (SR) for unicast or Point-to-Point (P2P) path is described in [RFC8402]. For SR multicast or Point-to-Multipoint (P2MP) path/tree, it may be implemented through using multiple SR P2P paths. The function of a SR P2MP path/tree from an ingress node to multiple (say n) egress/leaf nodes is implemented by n SR P2P paths. These n P2P paths are from the ingress to those n egress/leaf nodes of the P2MP path/tree. This solution may waste some network resources such as link bandwidth.

An alternative solution proposed in [I-D.shen-spring-p2mp-transport-chain] uses a number of P2MP chain tunnels to implement a P2MP path/tree from an ingress to n egress/ leaf nodes. Each P2MP chain tunnel is a tunnel from the ingress to a leaf node as its tail end and may have some leaf nodes as its bud nodes along the tunnel. This alternative solution improves the usage of network resources over the solution above using pure P2P paths. However, these two solutions are based on SR P2P paths.

A solution for a SR P2MP path/tree using a P2MP multicast tree is proposed in [I-D.voyer-pim-sr-p2mp-policy]. For a SR P2MP path/tree from an ingress/root to multiple egress/leaf nodes, a multicast P2MP tree is created to deliver the traffic from the ingress/root to the egress/leaf nodes. The state of the tree is instantiated in the forwarding plane by a controller such as PCE at Root node, intermediate Replication nodes and Leaf nodes of the tree. This is not consistent with the SR principles in which no state is stored at the core of the network.

This document describes a new solution for a SRv6 Point-to-Multipoint (P2MP) Path/Tree to deliver the traffic from the ingress of the path to the multiple egresses/leaves of the path in a SR domain. This solution uses a P2MP multicast tree without storing its state in the core of the network for a SR P2MP path/tree like a SR P2P path.

2. Overview of P2MP Multicast Tree

For a SR P2P path from its ingress to its egress, a segment list for the path is provided to the ingress. The ingress pushes the list into a packet, and the packet is delivered to the egress according to the segment list without any state in the core of the network.

For a SR P2MP path from its ingress to multiple egress/leaf nodes, a segment list for the P2MP path is provided to the ingress. The ingress pushes the list into a packet, and the packet is delivered to the multiple egress/leaf nodes according to the segment list without any state in the core of the network.

Figure 1 shows a SR P2MP path from ingress/root R to four egress/leaf nodes L1, L2, L3 and L4. Nodes P1, P2, P3 and P4 are the transit nodes of the P2MP path.

Suppose that X-m is the segment identifier (SID) of node X. X-m is an adjacent SID or node SID. For simplicity, we assume X-m is a node SID in the illustrations below. R-m, P1-m, P2-m, P3-m, P4-m, L1-m, L2-m, L3-m and L4-m are the SIDs of the nodes on the SR P2MP path. They are multicast SIDs or replication SIDs in general.

A multicast SID is a SID from a multicast SID block. In a SR domain supporting SR multicast, each node has a multicast node SID, which is globally significant; each adjacency of a node has a multicast adjacency SID, which is locally significant. A multicast SID of a node on a SR P2MP path is associated with the SIDs of the next hop (or say downstream) nodes. When the node receives a packet with its multicast SID, it duplicates and sends the packet to each of the next hop nodes according to their SIDs.

If node P on a SR P2MP path has B (B > 1) next hop nodes along the path, the SID of node P, P-m, MUST be a multicast SID when it is in the segment list for the P2MP path. The SIDs of the B next hop nodes just follow P-m in the segment list. When node P receives the packet with P-m, it duplicates and sends the packet to each of the B next hop nodes along the P2MP path.

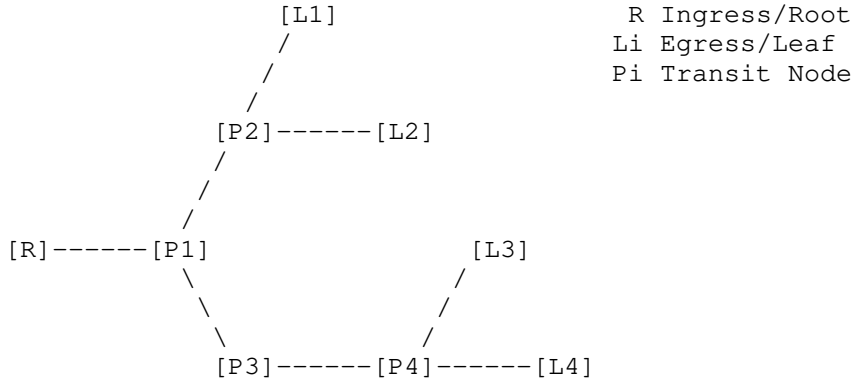

Figure 1: SR P2MP Path from R to L1, L2, L3 and L4

<P1-m, P2-m, P3-m, L1-m, L2-m, P4-m, L3-m, L4-m> is a segment list for the SR P2MP path in Figure 1 to be pushed into a packet at ingress/root R. Node P1 has 2 next hop nodes P2 and P3 along the P2MP path. The next hop nodes' SIDs P2-m and P3-m follow P1-m, which is P1's multicast SID. When P1 receives a packet transported by the P2MP path, it duplicates and sends the packet to the next hop nodes P2 and P3 according to P1-m, P2-m and P3-m.

The number of branches or next hops from node P1 is a value of one argument in P1-m, called N-Branches. The value of N-Branches in P1-m is 2. With this information, node P1 duplicates and sends the packet to 2 next hop nodes P2 and P3, which are indicated by the 2 SIDs P2-m and P3-m following P1-m.

The number of SIDs of the nodes under node P1 is a value of another argument in P1-m, called N-SIDs. The value of N-SIDs in P1-m is 7, indicating that there are 7 SIDs following P1-m in the segment list.

There are 2 branches or next hops (i.e., L1 and L2) from node P2 and 2 SIDs (i.e., L1-m and L2-m) of the nodes under node P2. The values of N-Branches and N-SIDs in P2-m are 2 and 2. with this information, before sending the packet to node P2, node P1 pushes the SIDs under node P2 into the packet (i.e., the packet has a new segment list with the SIDs under node P2. The new segment list replaces the old one in the packet).

There are 1 branch or next hop (i.e., P4) from node P3 and 3 SIDs (i.e., P4-m, L3-m and L4-m) of the nodes under node P3. The values of N-Branches and N-SIDs in P3-m are 1 and 3 respectively. with this information, before sending the packet to node P3, node P1 pushes the SIDs under node P3 into the packet.

Each node on the SR P2MP path sends the packet to its next hop nodes according to the segment list and no state is stored in any transit node (i.e., the core of the network). The packet is delivered to the egress/leaf nodes from the ingress.

3. Encoding P2MP Multicast Tree

For each sub-tree STi of a SR P2MP path from the ingress node of the P2MP path, suppose that

- the multicast SID of the next hop node NHi is mSIDi;
- there are Bi branches (i.e., outgoing interfaces) to the next hop node BNHj (j = 1, ..., Bi) from node NHi along the sub-tree, the multicast SID of BNHj is mSIDij;
- the number of branches (i.e., outgoing interfaces) under the node BNHj (j = 1, ..., Bi) is BBj; and the number of SIDs of the nodes under each of the Bi branches from node BNHj is NSj (j = 1, ..., Bi).

Sub-tree STi is encoded as segment list

```
      < mSIDi,  mSIDi1, ..., mSIDiBi,   SegSeq1, ...,  SegSeqBi  >,
        \___/  \_____________________/   \______/      \________/
SIDs of  NHi    Bi branches/next-hops   sub-tree       sub-tree
                BNHj of node NHi        from BNH1      from BNHBi
```

where mSIDi contains the number of branches, Bi, in its N-Branches field, and the number of SIDs under mSIDi in its N-SIDs field; mSIDij

(j = 1, ..., Bi) contains the number of branches, BBj, in its N-Branches field and the number of SIDs, NSj, in its N-SIDs field; SegSeqj (j = 1, ..., Bi) is the SID sequence in the segment list encoding the sub-trees from node BNHj.

For the P2MP path in Figure 1 from ingress node R to egress nodes L1, L2, L3 and L4, there is one sub-tree from R.

For this sub-tree,

- the next hop node is P1 and the multicast SID of P1 is P1-m;
- there are 2 branches to the next hop nodes P2 and P3 from node P1 along the sub-tree; the number of SIDs of the nodes under P1 is 7; the multicast SIDs of P2 and P3 are P2-m and P3-m respectively;
- the numbers of SIDs of the nodes under these two branches are 2 and 3 respectively. The SIDs of the nodes under P2 are L1-m and L2-m. The SIDs of the nodes under P3 are P4-m, L3-m and L4-m.

The sub-tree is encoded as segment list

```
      < P1-m,  P2-m, P3-m,          L1-m, L2-m,  P4-m, L3-m, L4-m >,
        \__/  \____________/         \________/   \________________/
SIDs of  P1   2 branches/next-hops   sub-tree     sub-tree
              P2 and P3 of node P1   from P2      from P3
```

where

P1-m's N-Branches field is set to 2 and its N-SIDs field to 7;
P2-m's N-Branches field is set to 2 and its N-SIDs field to 2;
P3-m's N-Branches field is set to 1 and its N-SIDs field to 3;

L1-m and L2-m are the SID sequence in the segment list encoding the sub-trees from P2;

P4-m, L3-m and L4-m are the SID sequence in the segment list encoding the sub-trees from P3; and

P4-m's N-Branches field is set to 2 and its N-SIDs field to 2.

Figure 2 shows in details the segment list, which is an encoding of the P2MP multicast tree for the SR P2MP path from R to L1, L2, L3 and L4.

	N-Branches	N-SIDs		
P1's Multicast SID Locator	2	7	Arguments	P1-m
P2's Multicast SID Locator	2	2	Arguments	P2-m
P3's Multicast SID Locator	1	3	Arguments	P3-m
L1's Multicast SID Locator	0	0	Arguments	L1-m
L2's Multicast SID Locator	0	0	Arguments	L2-m
P4's Multicast SID Locator	2	2	Arguments	P4-m
L3's Multicast SID Locator	0	0	Arguments	L3-m
L4's Multicast SID Locator	0	0	Arguments	L4-m

Figure 2: Encoding of P2MP Multicast Tree from R to L1 - L4

SID P1-m indicates that there are 2 branches and 7 SIDs under P1. SID P2-m indicates that there are 2 branches and 2 SIDs under P2. SID P3-m indicates that there are 1 branch and 3 SIDs under P3. SIDs L1-m and L2-m indicate that there is no branch under them. SID P4-m indicates that there are 2 branches and 2 SIDs under P4. L3-m and L4-m indicate that there is no branch under them.

4. Procedures/Behaviors

This section describes the procedures or behaviors on the ingress, transit and egress/leaf node of a SR P2MP path to deliver a packet received from the path to its destinations.

4.1. Procedure/Behavior on Ingress Node

For a packet to be transported by a SR P2MP Path, the ingress of the P2MP path duplicates the packet for each sub-tree of the SR P2MP path branching from the ingress, pushes the segment list encoding the sub-tree into the packet by executing H.Encaps [I-D.ietf-spring-srv6-network-programming] and sends the packet to the next hop node along the sub-tree.

For example, there is one sub-tree from the ingress R of the SR P2MP path in Figure 1 via next hop node P1 towards egress/leaf nodes L1, L2, L3 and L4.

For this sub-tree, the ingress R duplicates the packet, set the destination address (DA) to P1-m (i.e., multicast SID of node P1), pushes the segment list without P1-m (i.e., <P2-m, P3-m, L1-m, L2-m, P4-m, L3-m, L4-m>) encoding the sub-tree in a Segment Routing Header (SRH) of the packet by executing H.Encaps and sends the packet to DA (i.e., node P1). The contents of the multicast SIDs P1-m, P2-m, P3-m, L1-m, L2-m, P4-m, L3-m, L4-m are shown in Figure 2.

Suppose that the duplicated packet is Pkt0 for the sub-tree. The execution of H.Encaps pushes an IPv6 header (i.e., SRH) to Pkt0 and sets some fields in the header to produce an encapsulated packet Pkt'. Pkt' is represented in the following:

```
Pkt' = (SA=R, DA=P1-m)( L4-m, L3-m,..., P3-m,P2-m; SL=7)Pkt0
                       \_____________________________/
          corresponds to: <P2-m,P3-m, ..., L3-m,L4-m>
```

where DA=P1-m means that the destination address (DA) is set to P1-m; SA=R means that the source address (SA) is set to R; SL=7 means that the number of Segments Left (SL) is 7.

4.2. Procedure/Behavior on Transit Node

When a transit node of a SR P2MP path receives a packet transported by the P2MP path, the DA of the packet is a multicast SID of the node and the packet contains a segment list for the sub-trees under the transit node. The DA and the segment list comprise the information for encoding the sub-trees.

For example, when node P1 receives a packet transported by the SR P2MP path in Figure 1, the packet's DA is P1-m (which is a multicast SID of node P1) and the segment list in the packet is <P2-m, P3-m, L1-m, L2-m, P4-m, L3-m, L4-m>.

The N-Branches field (which has value of n) of the DA indicates that there are n branches (or say sub-trees) under the transit node. The N-SIDs field of the DA indicates the number of SIDs for these n sub-trees under the transit node. The multicast SIDs of the next hop nodes of these n sub-trees are the first n multicast SIDs of the segment list in the packet.

For example, the N-Branches field (which has value of 2) of DA = P1-m indicates that there are 2 branches (or say sub-trees) under node P1. The N-SIDs field (which has value of 7) of the DA = P1-m indicates that there are 7 SIDs for these 2 sub-trees under node P1.

The first multicast SID (P2-m) of the segment list is the SID of the next hop node (P2) of the first sub-tree; The second multicast SID

(P3-m) of the segment list is the SID of the next hop node (P3) of the second sub-tree.

After the multicast SIDs of the next hop nodes, there are n blocks of SIDs for those n sub-trees. The N-SIDs field (which has value of B1) of the first multicast SID of the next hop nodes indicates that there are B1 SIDs in the first block for the first sub-tree; the N-SIDs field (which has value of B2) of the second multicast SID of the next hop nodes indicates that there are B2 SIDs in the second block for the second sub-tree after the first block; and so on.

For example, there are 2 blocks of SIDs for the 2 sub-trees under node P1 after the multicast SIDs P2-m and P3-m of the next hop nodes P2 and P3. The N-SIDs field of P2-m (the first multicast SID of the next hop nodes) has value of 2, indicating that there are 2 SIDs in the first block for the first sub-tree, which are L1-m and L2-m.

The N-SIDs field of P3-m (the second multicast SID of the next hop nodes) has value of 3, indicating that there are 3 SIDs in the second block for the second sub-tree after the first block, which are P4-m, L3-m and L4-m.

The transit node duplicates the packet without top header for each sub-tree under it and adds a new header with a new segment list built from the SID block for the sub-tree to the duplicated packet by executing H.Encaps. It sets the DA of the packet to the multicast SID of the next hop node along the sub-tree and sends the packet to the DA.

For example, node P1 duplicates the packet for the first sub-tree towards L1 and L2 and adds a new header with a new segment list <L1-m, L2-m>. It sets DA = P2-m (multicast SID of next hop P2), and sends the packet to the DA (i.e., P2).

Suppose that the duplicated packet is Pkt0 for the sub-tree. The execution of H.Encaps pushes a new IPv6 header (i.e., SRH) to Pkt0 and sets some fields in the header to produce an encapsulated packet Pkt'. Pkt' is represented in the following:

```
    Pkt' = (SA=P1, DA=P2-m)( L2-m, L1-m;  SL=2)Pkt0.
                            \___________/
          corresponds to:   <L1-m, L2-m>
```

where DA=P2-m means that the destination address (DA) is set to P2-m; SA=P1 means that the source address (SA) is set to P1; SL=2 means that the number of Segments Left (SL) is 2.

Node P1 duplicates the packet for the second sub-tree via P3 towards L3 and L4 and adds a new header with a new segment list <P4-m, L3-m, L4-m>. It sets DA = P3-m (multicast SID of next hop P3), and sends the packet to the DA (i.e., P3).

4.3. Procedure/Behavior on Egress Node

When an egress node of a SR P2MP path receives a packet transported by the P2MP path, the DA of the packet is a SID of the egress node. The egress node sends the packet to its destination accordingly. If the SID is a multicast SID of the egress, the N-Branches field and N-SIDs field are all zeros.

5. Protection

Protections for a SR P2MP path can be classified into two types: global protection and local protection.

5.1. Global Protection

For a primary SR P2MP path from an ingress node R1 to multiple egress nodes Li (i = 1, ..., n), a backup SR P2MP path from an ingress node R1' to multiple egress nodes Li' (i = 1, ..., n) is set up to provide global protection for the primary SR P2MP path. If R1' is the same as R1, the failure of the ingress node R1 of the primary SR P2MP path is not protected; otherwise (i.e., R1' and R1 are different and connected to the same traffic source), the failure of the ingress node R1 is protected. If Li' is the same as Li (i = 1, ..., n), the failure of the egress nodes Li (i = 1, ..., n) of the primary SR P2MP path is not protected; otherwise (i.e., Li' and Li are different and connected to the same destination), the failure of the egress nodes Li is protected.

When a failure happens on the primary SR P2MP path and is detected by the source of the traffic or other entity, the traffic to be transported by the primary SR P2MP path is switched to the backup SR P2MP path, which sends the traffic from its ingress node R1' to its egress nodes Li' (i = 1, ..., n).

5.2. Local Protection

Local protection or say Fast Reroute (FRR) of a node and adjacency segment on a SR P2P path is proposed in [I-D.ietf-rtgwg-segment-routing-ti-lfa] and [I-D.ietf-rtgwg-srv6-egress-protection]. It can be applied to FRR of a node and adjacency segment on a SR P2MP path in a similar way. But FRR for SR P2MP path is more complicated.

More details will be added later.

6. IANA Considerations

TBD

7. Security Considerations

TBD

8. Acknowledgements

The authors would like to thank Acee Lindem and Daniel Voyer for their valuable comments and suggestions on this draft.

9. References

9.1. Normative References

[I-D.ietf-6man-segment-routing-header]
Filsfils, C., Dukes, D., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", draft-ietf-6man-segment-routing-header-26 (work in progress), October 2019.

[I-D.ietf-rtgwg-segment-routing-ti-lfa]
Litkowski, S., Bashandy, A., Filsfils, C., Decraene, B., Francois, P., Voyer, D., Clad, F., and P. Camarillo, "Topology Independent Fast Reroute using Segment Routing", draft-ietf-rtgwg-segment-routing-ti-lfa-04 (work in progress), August 2020.

[I-D.ietf-rtgwg-srv6-egress-protection]
Hu, Z., Chen, H., Chen, H., Wu, P., Toy, M., Cao, C., Liu, L., and X. Liu, "SRv6 Path Egress Protection", draft-ietf-rtgwg-srv6-egress-protection-01 (work in progress), July 2020.

[I-D.ietf-spring-srv6-network-programming]
Filsfils, C., Camarillo, P., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "SRv6 Network Programming", draft-ietf-spring-srv6-network-programming-20 (work in progress), September 2020.

[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>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <https://www.rfc-editor.org/info/rfc8754>.

9.2. Informative References

[I-D.shen-spring-p2mp-transport-chain]
Shen, Y., Zhang, Z., Parekh, R., Bidgoli, H., and Y. Kamite, "Point-to-Multipoint Transport Using Chain Replication in Segment Routing", draft-shen-spring-p2mp-transport-chain-02 (work in progress), April 2020.

[I-D.voyer-pim-sr-p2mp-policy]
Voyer, D., Filsfils, C., Parekh, R., Bidgoli, H., and Z. Zhang, "Segment Routing Point-to-Multipoint Policy", draft-voyer-pim-sr-p2mp-policy-02 (work in progress), July 2020.

[I-D.voyer-spring-sr-replication-segment]
Voyer, D., Filsfils, C., Parekh, R., Bidgoli, H., and Z. Zhang, "SR Replication Segment for Multi-point Service Delivery", draft-voyer-spring-sr-replication-segment-04 (work in progress), July 2020.

Authors' Addresses

Huaimo Chen
Futurewei
Boston, MA
USA

Email: Huaimo.chen@futurewei.com

Mike McBride
Futurewei

Email: michael.mcbride@futurewei.com

Yanhe Fan
Casa Systems
USA

Email: yfan@casa-systems.com

Mehmet Toy
Verizon
USA

Email: mehmet.toy@verizon.com

Aijun Wang
China Telecom
Beiqijia Town, Changping District
Beijing, 102209
China

Email: wangaj3@chinatelecom.cn

Lei Liu
Fujitsu

USA

Email: liulei.kddi@gmail.com

Xufeng Liu
Volta Networks

McLean, VA
USA

Email: xufeng.liu.ietf@gmail.com

Network Working Group
Internet-Draft
Updates: 3376, 3810 (if approved)
Intended status: Standards Track
Expires: January 13, 2021

M. Sivakumar
Juniper Networks
S. Venaas
Cisco Systems, Inc.
Z. Zhang
ZTE Corporation
H. Asaeda
NICT

July 12, 2020

IGMPv3/MLDv2 Message Extension
draft-ietf-pim-igmp-mld-extension-01

Abstract

IGMP and MLD protocols are extensible, but no extensions have been defined so far. This document provides a well-defined way of extending IGMP and MLD, using a list of TLVs (Type, Length and Value).

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 https://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 January 13, 2021.

Copyright Notice

Table of Contents

1. Introduction

In this document, we describe a generic method to extend IGMPv3 [RFC3376] and MLDv2 [RFC3810] messages to accommodate information other than what is contained in the current message formats. This is done by allowing a list of TLVs (Type, Length and Value) to be used in the Additional Data part of IGMPv3 and MLDv2 messages. This document defines a registry for such TLVs, while other documents will define the specific types and their values, and their semantics. The extension would only be used when at least one TLV is to be added to the message. This extension also applies to the lightweight versions of IGMPv3 and MLDv2 as defined in [RFC5790].

The extension will be part of additional data as mentioned in [RFC3810] Section 5.1.12 (resp. [RFC3376] Section 4.1.10) for query messages and [RFC3810] Section 5.2.12 (resp. [RFC3376] Section 4.2.11) for report messages.

One such TLV is being defined in [I-D.ietf-bier-mld]

2. Conventions used in this document

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.

3. Extension Format

A previously reserved bit in the IGMPv2 and MLDv2 headers is used to indicate whether this extension is used. It is set to 1 if it is used, otherwise 0. When this extension is used, the Additional Data of IGMPv3 and MLDv2 messages would be formatted as follows. Note that this format contains a variable number of TLVs. It MUST contain at least one TLV.

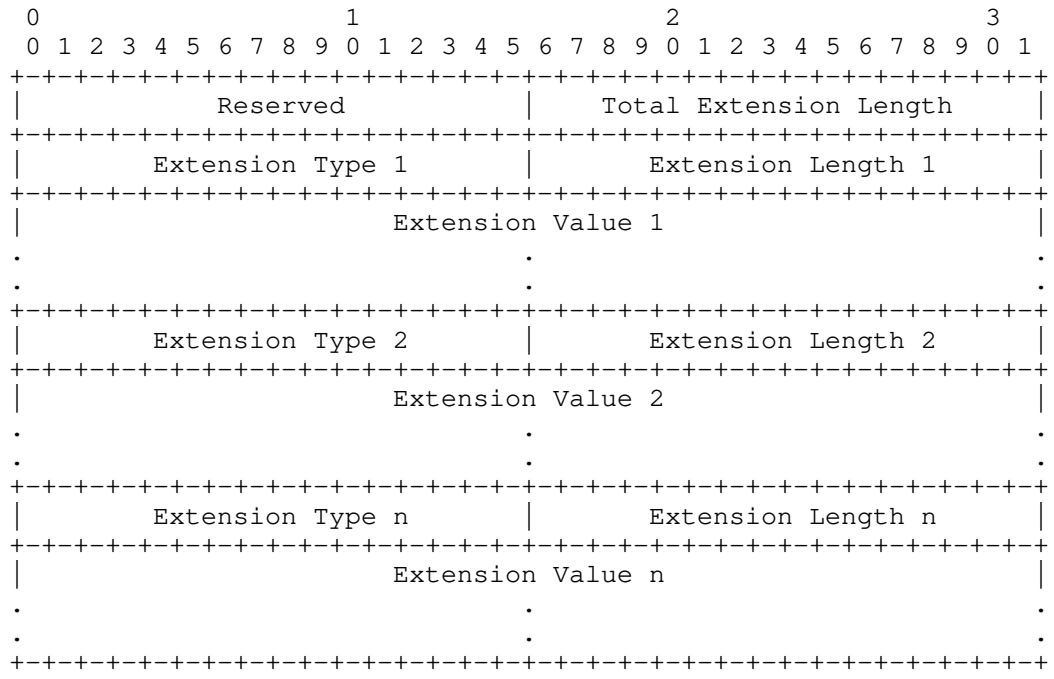

Figure 1: Extension Format

Reserved: 2 octets. Reserved. MUST be set to 0. MUST be ignored when received.

Total Extension Length: 2 octets. The remaining length of the extension. This value MUST be equal to ((2 + 2) * n) + Extension Length 1 + Extension Length 2 + ... Extension Length n. That is, it is the sum of the lenghts of all the TLVs, including the type field (2 octets), and the length field (2 octets) of each TLV. The total number of octets used by the extension is the value of

this field plus 4 (including the 2 octets Reserved field and the 2 octets of this field).

Extension Type: 2 octets. This identifies a particular Extension Type as defined in the IGMP/MLD Extension Type Registry.

Extension Length: 2 octets. This specifies the length in octets of the following Extension Value field.

Extension Value: This field contains the value. The length and the contents of this field is according to the specification of the Extension Type as defined in the IGMP/MLD Extension Type Registry. The length MUST be as specified in the Extension Length field.

Note that there may be additional data following this extension. The Total Extension Length field would indicate where this extension ends, and the additional data starts. Also, there is a possibility that an implementation uses the Additional Data part of IGMP/MLD messages, but not according to this extension scheme. When a message is received, it MUST be verified that the Total Extension Length is equal to ((2 + 2) * n) + Extension Length 1 + Extension Length 2 + ... Extension Length n, where n is the number of TLVs. Note that the value of n is not known ahead of time. An implementation would walk through the TLVs and add the 4 octet overhead and the length of each TLV, until the sum is larger or equal to the Total Extension Length, or until the end of the IGMP/MLD message, whichever happens first. Any additional data after this MUST be ignored, except the data MUST be included in checksum computations. If the sum is not equal to the Total Extension Length, then it is assumed that this extension is not being used, and this specification does not apply.

3.1. Multicast Listener Query Extension

The MLD query format with extension is shown below. The E-bit is set to 1 to indicate that the extension is present.

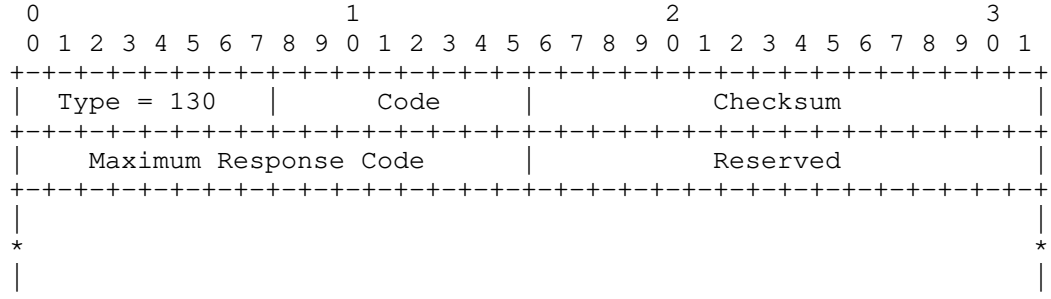

Figure 2: MLD Query Extension

3.2. Version 2 Multicast Listener Report Extension

The MLD report format with extension is shown below. The E-bit is set to 1 to indicate that the extension is present.

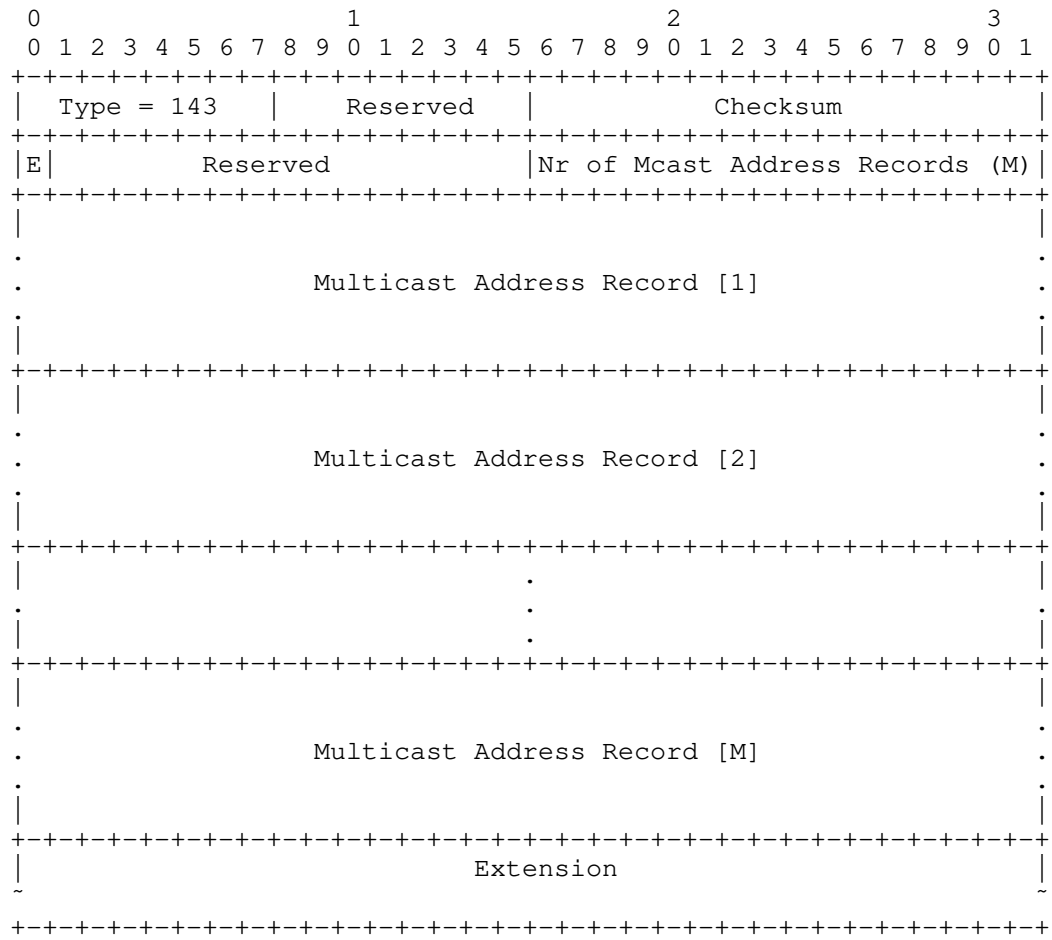

Figure 3: MLD Report Extension

3.3. IGMP Membership Query Extension

The IGMP query format with the extension is shown below. The E-bit is set to 1 to indicate that the extension is present.

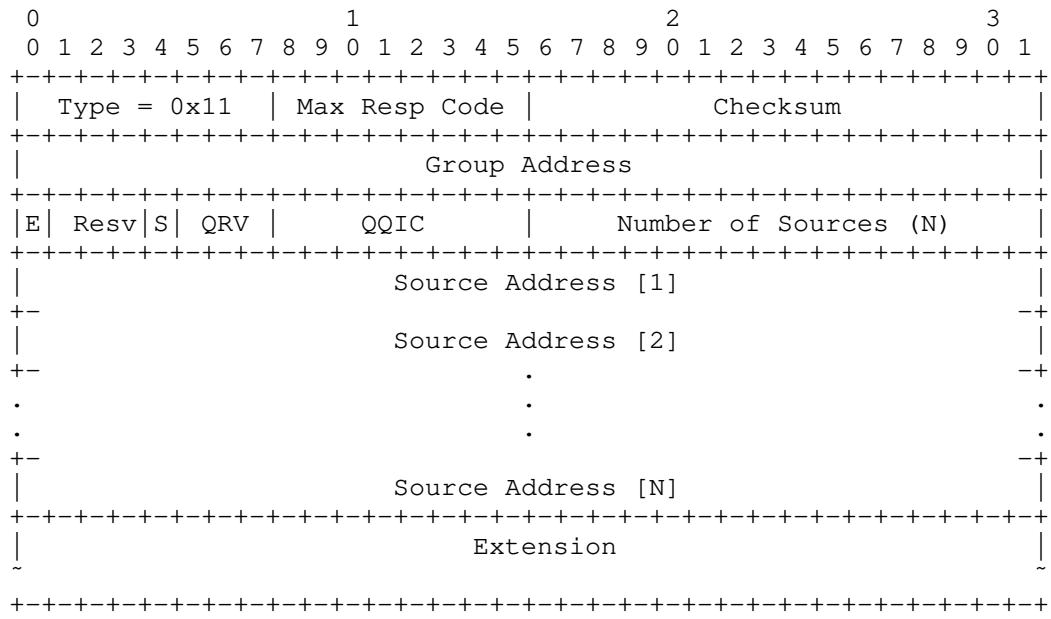

Figure 4: IGMP Query Extension

3.4. IGMP Version 3 Membership Report Extension

The IGMP report format with the extension is shown below. The E-bit is set to 1 to indicate that the extension is present.

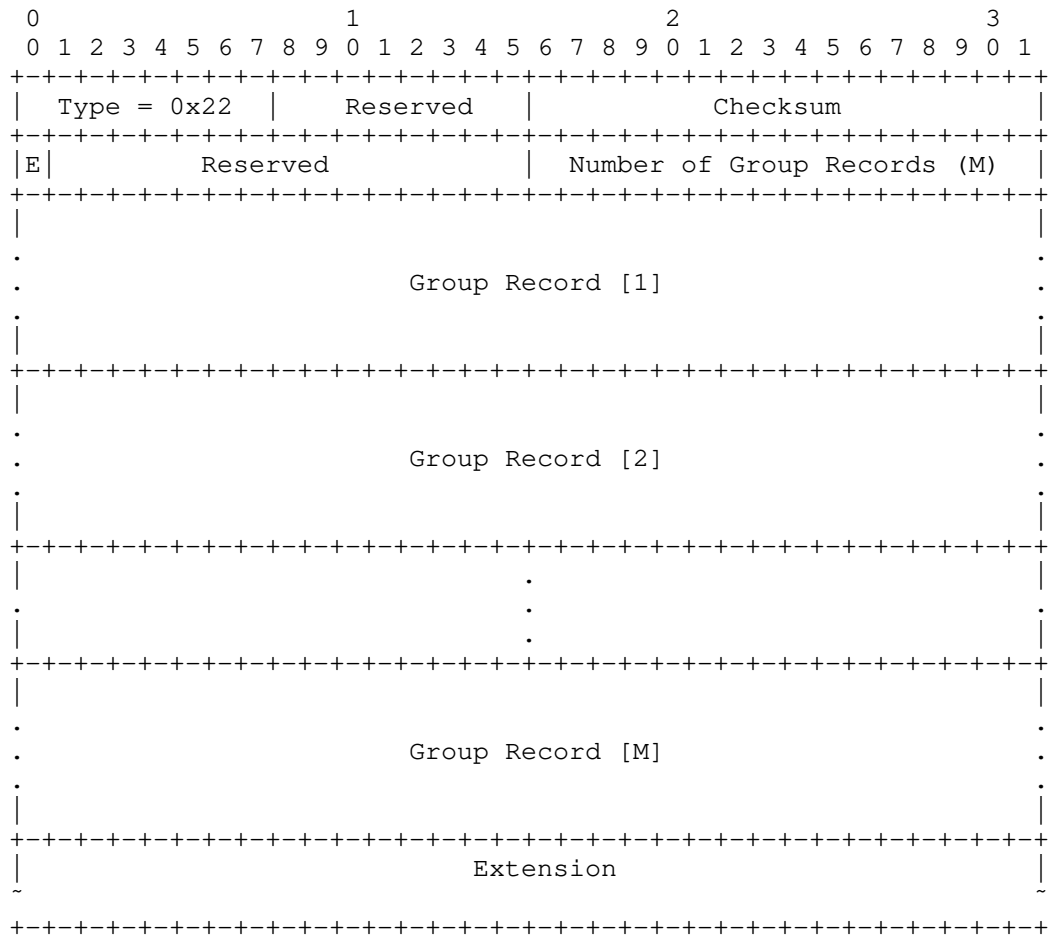

Figure 5: IGMP Report Extension

4. Applicability and backwards compatibility

IGMP and MLD implementations, host implementations in particular, rarely change, and it expected to take a long time for them to support this extension mechanism. Also as new extensions are defined, it may take a long time before they are supported. Implementations that do not support this extension mechanism will simply ignore the extension, provided they are compliant with IGMPv3 and MLDv2 RFCs, and behave as if the extension is not present. Implementations that support this extension MUST behave as if it is not present if they support non of the extension types in an IGMP/MLD message. If they support at least one of the types, they will

process the supported types according to the type specifications, and ignore any unsupported types.

When defining new types, care must be taken to ensure that nodes that support the type can co-exist with nodes that don't, on the same subnet. There could be multiple routers where only some support the extension, or multiple hosts where only some support the extension. Or a router may support it and none of the hosts, or all hosts may support it, but none of the routers.

The extension mechanism do not support IGMPv1, IGMPv2 and MLDv1. As nodes may send older version message, they would also not be able to send messages using this extension.

5. Security Considerations

This document extends MLD (resp. IGMP) message formats. As such, there is no impact on security or changes to the considerations in [RFC3810] and [RFC3376]. The respective types defined using this extension may impact security and must be considered as part of the respective specifications.

6. IANA Considerations

A new registry called "IGMP/MLD Extension Types" should be created with registration procedure "IETF Review" as defined in [RFC8126] with this document as a reference. The registry should be common for IGMP and MLD and can perhaps be added to the "Internet Group Management Protocol (IGMP) Type Numbers" section. The initial content of the registry should be as below.

Type	Length	Name	Reference

7. References

7.1. 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>.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, DOI 10.17487/RFC3376, October 2002, <https://www.rfc-editor.org/info/rfc3376>.

[RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <https://www.rfc-editor.org/info/rfc3810>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

7.2. Informative References

[I-D.ietf-bier-mld] Pfister, P., Wijnands, I., Venaas, S., Wang, C., Zhang, Z., and M. Stenberg, "BIER Ingress Multicast Flow Overlay using Multicast Listener Discovery Protocols", draft-ietf-bier-mld-04 (work in progress), March 2020.

[RFC5790] Liu, H., Cao, W., and H. Asaeda, "Lightweight Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Version 2 (MLDv2) Protocols", RFC 5790, DOI 10.17487/RFC5790, February 2010, <https://www.rfc-editor.org/info/rfc5790>.

Authors' Addresses

Mahesh Sivakumar
Juniper Networks
64 Butler St
Milpitas CA 95035
USA

Email: sivakumar.mahesh@gmail.com

Stig Venaas
Cisco Systems, Inc.
Tasman Drive
San Jose CA 95134
USA

Email: stig@cisco.com

Zheng(Sandy) Zhang
ZTE Corporation
No. 50 Software Ave, Yuhuatai District
Nanjing 210000
China

Email: zhang.zheng@zte.com.cn

Hitoshi Asaeda
National Institute of Information and
Communications Technology
4-2-1 Nukui-Kitamachi
Koganei, Tokyo 184-8795
Japan

Email: asaeda@nict.go.jp

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 28, 2021

D. Voyer, Ed.
Bell Canada
C. Filsfils
R. Parekh
Cisco Systems, Inc.
H. Bidgoli
Nokia
Z. Zhang
Juniper Networks

July 27, 2020

Segment Routing Point-to-Multipoint Policy
draft-ietf-pim-sr-p2mp-policy-00

Abstract

This document describes an architecture to construct a Point-to-Multipoint (P2MP) tree to deliver Multi-point services in a Segment Routing domain. A SR P2MP tree is constructed by stitching a set of Replication segments together. A SR Point-to-Multipoint (SR P2MP) Policy is used to define and instantiate a P2MP tree which is computed by a PCE.

Requirements Language

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

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 https://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 January 28, 2021.

Table of Contents

1. Introduction

A Multi-point service delivery could be realized via P2MP trees in a Segment Routing domain [RFC8402]. A P2MP tree spans from a Root node to a set of Leaf nodes via intermediate Replication nodes. It consists of a Replication segment [I-D.ietf-spring-sr-replication-segment] at the root node, one or more Replication segments at Leaf nodes and intermediate Replication nodes. The Replication segments are stitched together.

A Segment Routing P2MP policy, a variant of the SR Policy [I-D.ietf-spring-segment-routing-policy], is used to define a P2MP tree. A PCE is used to compute the tree from the Root node to the set of Leaf nodes via a set of replication nodes. The PCE then instantiates the P2MP tree in the SR domain by signaling Replication segments to Root, replication and Leaf nodes using various protocols (PCEP, BGP, NetConf etc.).

2. P2MP Tree

A P2MP tree in a SR domain connects a Root to a set of Leaf nodes via a set of intermediate Replication nodes. It consists of a Replication segment at the root stitched to Replication segments at intermediate Replication nodes eventually reaching the Leaf nodes.

The Replication SID of the Replication Segment at Root node is called Tree-SID. The Tree-SID SHOULD also be used as Replication SID of Replication segments at Replication and Leaf nodes. The Replication segments at Replication and Leaf nodes MAY use Replication SIDs that are not same as the Tree-SID.

The Replication segment at Root of a P2MP tree MUST be associated with that P2MP tree (i.e. <Root, Tree-ID> identifier in SR P2MP policy section below) to map a Multi-point service to the tree. A Replication segment that terminates a P2MP tree at a Leaf node MUST be associated with the P2MP tree to determine the context for a Multi-point service. The The information that can be used to derive this association is specific to encoding of the protocol (PCEP, BGP, NetConf etc.) used to instantiate the Replication segment for a P2MP tree. Replication segments at intermediate Replication nodes of a tree are also associated with that tree.

A PCE MAY decide not instantiate Replication segments at Leaf nodes of a P2MP tree if it is known a priori that Multi-point services mapped to the P2MP tree can be identified using a context that is globally unique in SR domain. Multi-point service contexts assigned from "Domain-wide Common Block" (DCB) [I-D.ietf-bess-mvpn-evpn-aggregation-label] are an example of such

globally unique contexts. A Segment Routing Global Block (SRGB) [RFC8402] MAY be used to allocate globally unique Multi-point service contexts, but it is NOT RECOMMENDED to do so as the service contexts only need to be unique at service edge nodes. In this case, Replication nodes connecting to Leaf nodes SHOULD use Penultimate-Hop Pop (PHP) behavior to pop Tree-SID from a packet.

A packet steered into a P2MP tree is replicated by the Replication segment at Root node to each downstream node in the Replication segment, with the Replication SID of the Replication Segment at the downstream node. A downstream node could be a Leaf node or an intermediate Replication node. In the latter case, replication continues with the Replication segments until all Leaf nodes are reached. A packet is steered into a P2MP tree in two ways:

- Based on a local policy-based routing at the Root node.

- Based on steering via the Tree-SID at the Root node.

2.1. Sharing Replication segments across P2MP trees

Two or more P2MP trees MAY share a Replication segment at Root or Replication nodes if at minimum as the first condition below is satisfied. A tree always has its own Replication segment at its root even if shares another Replication segment. A tree that shares another Replication segment may or may not have its own Replication segment on its Leaf nodes. If not, the second and third conditions apply to such situations.

1. The Leaf nodes reached via a shared Replication segment must be subset of Leaf or Replication nodes of the P2MP trees that shares this segment. Note if a Replication segment is shared, all its downstream Replication segments are also shared.

2. Some Multi-point services realized by the P2MP trees may need service context (e.g. packets are for certain VPNs, and/or from certain nodes). If the trees do not have their own Replication segments at their Leaf nodes then the packets transported on the P2MP trees MUST carry a service context that does not rely on the tree or root identification, e.g. a service label assigned from Domain-wide Common Block or common SRGB.

3. For some Multi-point services using P2MP trees that share Replication segments, packets transported on these trees MAY require a Tree context (e.g. MVPN Extranet [RFC7900] to avoid certain ambiguities - see Section 2.3.1 of RFC 7900). In this case, the trees MUST have their own Replication segments on the Leaf nodes. This is similar to "tunnel stacking" concept.

Sharing of a Replication segment for P2MP trees is OPTIONAL. Exact procedures to ensure validity of above conditions across PM2P services on nodes of a Segment Routing domain are outside the scope of this document.

3. SR P2MP Policy

The SR P2MP policy is a variant of an SR policy [I-D.ietf-spring-segment-routing-policy] and is used to instantiate SR P2MP trees.

A SR P2MP Policy is identified by the tuple <Root, Tree-ID>, where:

- Root: The address of Root node of P2MP tree instantiated by the SR P2MP Policy
- Tree-ID: A identifier that is unique in context of the Root. This is an unsigned 32-bit number.

A SR P2MP Policy is defined by following elements:

- Leaf nodes: A set of nodes that terminate the P2MP trees.
- Candidate Paths: See below.

A SR P2MP policy is provisioned on a PCE to instantiate the P2MP tree. The Tree-SID SHOULD be used as Binding SID of the P2MP policy. A PCE computes the P2MP tree and instantiates Replication segments at Root, Replication and Leaf nodes. When Replication segments are not shared across P2MP trees, the Root and Tree-ID of the SR P2MP policy are mapped to Replication-ID element of the Replication segment identifier i.e the SR Replication segment identifier is <Root, Tree-ID, Node-ID>. A shared Replication segment MAY be identified with zero Root-ID address (0.0.0.0 for IPv4 and :: for IPv6) and a Replication-ID that is unique in context of Node address where the Replication segment is instantiated when it is not associated a particular tree.

A SR P2MP Policy has one or more Candidate paths. The active Candidate path is selected based on the tie breaking rules amongst the candidate-paths as specified in[I-D.ietf-spring-segment-routing-policy]. Each candidate path has a set of topological/resource constraints and/or optimization objectives which determine the P2MP tree for that Candidate path. Tree-SID is an identifier of the P2MP tree of the candidate path in the forwarding plane. It is instantiated in the forwarding plane at Root node, intermediate Replication nodes and Leaf nodes. The Tree-SID MAY be different at Replication and Leaf nodes.

4. Using Controller to build a P2MP Tree

A P2MP tree can be built using a Path Computation Element (PCE). This section outlines a high-level architecture for such an approach.

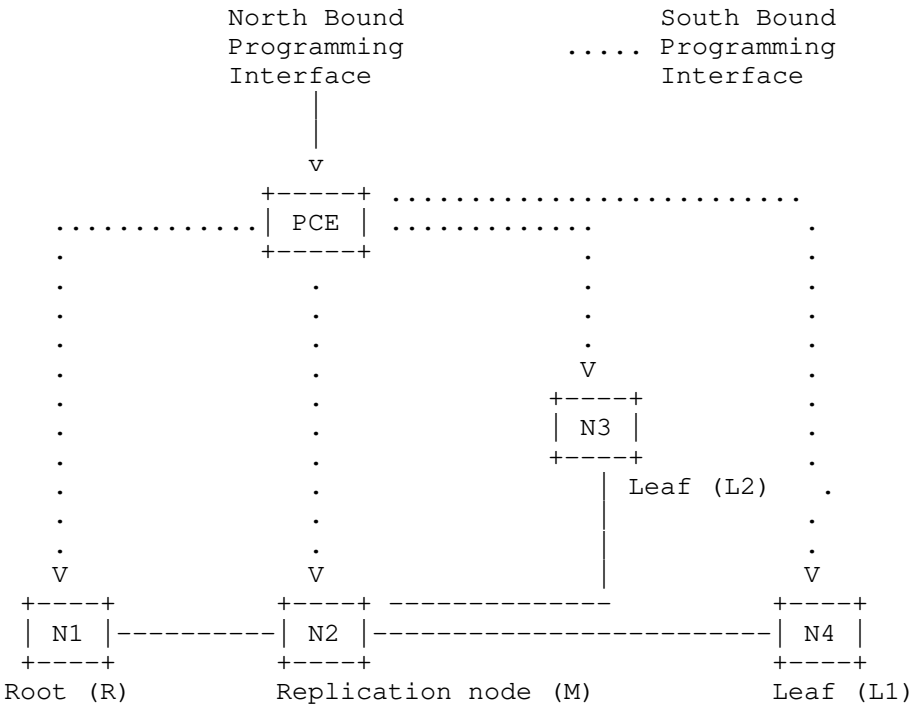

Figure 1: Centralized Control Plane Model

4.1. Provisioning SR P2MP Policy Creation

A SR P2MP policy can be instantiated and maintained in a centralized fashion using a Path Computation Element (PCE).

4.1.1. API

North-bound APIs on a PCE can be used to:

1. Create SR P2MP policy
2. Delete SR P2MP policy
3. Update SR P2MP policy

4. Create a Candidate Path for SR P2MP policy

5. Update a Candidate Path for SR P2MP policy

6. Delete a Candidate Path for SR P2MP policy

4.1.2. Invoking API

Interaction with a PCE can be via PCEP, REST, Netconf, gRPC, CLI. Yang model shall be be developed for this purpose as well.

4.2. P2MP Tree Computation

An entity (an operator, a network node or a machine) provisions a SR P2MP policy by specifying the addresses of the root (R) and set of leaves {L} as well as Traffic Engineering (TE) attributes of Candidate paths via a suitable North-Bound API. The PCE computes the tree of Active candidate path. The PCE MAY compute P2MP trees for all Candidate paths., If tree computation is successful, PCE instantiates the P2MP tree(s) using Replication segments on Root, Replication, and Leaf nodes.

Candidate path constraints shall include link color affinity, bandwidth, disjointness (link, node, SRLG), delay bound, link loss, etc. Candidate path shall be optimized based on IGP or TE metric or link latency.

The Tree SID of Candidate path of a SR P2MP policy can be either dynamically allocated by the PCE or statically assigned by entity provisioning the SR P2MP policy. Ideally, same Tree-SID SHOULD be used for Replication segments at Root, Replication, and Leaf nodes. Different Tree-SIDs MAY be used at replication node(s) if it is not feasible to use same Tree SID.

A PCE can modify a P2MP tree following network element failure or in case a better path can be found based on the new network state. In this case, the PCE may want to setup the new instance of the tree and remove the old instance of the tree from the network in order to minimize traffic loss. In this case, the instances of trees for all the Candidate paths of a P2MP policy can be identified by an Instance-ID which is unique in context of the P2MP policy. As such, the identifier of non-shared Replication segments used to instantiate these trees becomes <Root-ID, Tree-ID, Node-ID, Instance-ID>.

A PCE shall be capable of computing paths across multiple IGP areas or levels as well as Autonomous Systems (ASs).

4.2.1. Topology Discovery

A PCE shall learn network topology, TE attributes of link/node as well as SIDs via dynamic routing protocols (IGP and/or BGP-LS). It may be possible for entities to pass topology information to PCE via north-bound API.

4.2.2. Capability and Attribute Discovery

It shall be possible for a node to advertise SR P2MP tree capability via IGP and/or BGP-LS. Similarly, a PCE can also advertise its P2MP tree computation capability via IGP and/or BGP-LS. Capability advertisement allows a network node to dynamically choose one or more PCE(s) to obtain services pertaining to SR P2MP policies, as well a PCE to dynamically identify SR P2MP tree capable nodes.

4.3. Instantiating P2MP tree on nodes

Once a PCE computes a P2MP tree for Candidate path of SR P2MP policy, it needs to instantiate the tree on the relevant network nodes via Replication segments. The PCE can use various protocols to program the Replication segments as described below.

4.3.1. PCEP

PCE Protocol (PCEP)has been traditionally used:

1. For a head-end to obtain paths from a PCE.

2. A PCE to instantiate SR policies.

PCEP protocol can be stateful in that a PCE can have a stateful control of an SR policy on a head-end which has delegated the control of the SR policy to the PCE. PCEP shall be extended to provision and maintain SR P2MP trees in a stateful fashion.

4.3.2. BGP

BGP has been extended to instantiate and report SR policies. It shall be extended to instantiate and maintain P2MP trees for SR P2MP policies.

4.3.3. NetConf

TBD

4.4. Protection

4.4.1. Local Protection

A network link, node or path on the tree of a P2MP tree can be protected using SR policies computed by PCE. The backup SR policies shall be programmed in forwarding plane in order to minimize traffic loss when the protected link/node fails. It is also possible to use node local Fast Re-Route protection mechanisms (LFA) to protect link/nodes of P2MP tree.

4.4.2. Path Protection

It is possible for PCE create a disjoint backup tree for providing end-to-end path protection.

5. IANA Considerations

This document makes no request of IANA.

6. Security Considerations

There are no additional security risks introduced by this design.

7. Acknowledgements

The authors would like to acknowledge Siva Sivabalan, Mike Koldychev and Vishnu Pavan Beeram for their valuable inputs..

8. Contributors

Clayton Hassen
Bell Canada
Vancouver
Canada

Email: clayton.hassen@bell.ca

Kurtis Gillis
Bell Canada
Halifax
Canada

Email: kurtis.gillis@bell.ca

Arvind Venkateswaran
Cisco Systems, Inc.
San Jose

US

Email: arvvenka@cisco.com

Zafar Ali
Cisco Systems, Inc.
US

Email: zali@cisco.com

Swadesh Agrawal
Cisco Systems, Inc.
San Jose
US

Email: swaagraw@cisco.com

Jayant Kotalwar
Nokia
Mountain View
US

Email: jayant.kotalwar@nokia.com

Tanmoy Kundu
Nokia
Mountain View
US

Email: tanmoy.kundu@nokia.com

Andrew Stone
Nokia
Ottawa
Canada

Email: andrew.stone@nokia.com

Tarek Saad
Juniper Networks
Canada

Email:tsaad@juniper.net

9. References

9.1. Normative References

[I-D.ietf-spring-segment-routing-policy]
Filsfils, C., Talaulikar, K., Voyer, D., Bogdanov, A., and P. Mattes, "Segment Routing Policy Architecture", draft-ietf-spring-segment-routing-policy-08 (work in progress), July 2020.

[I-D.ietf-spring-sr-replication-segment]
Voyer, D., Filsfils, C., Parekh, R., Bidgoli, H., and Z. Zhang, "SR Replication Segment for Multi-point Service Delivery", draft-ietf-spring-sr-replication-segment-00 (work in progress), July 2020.

[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>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

9.2. Informative References

[I-D.ietf-bess-mvpn-evpn-aggregation-label]
Zhang, Z., Rosen, E., Lin, W., Li, Z., and I. Wijnands, "MVPN/EVPN Tunnel Aggregation with Common Labels", draft-ietf-bess-mvpn-evpn-aggregation-label-03 (work in progress), October 2019.

[RFC7900] Rekhter, Y., Ed., Rosen, E., Ed., Aggarwal, R., Cai, Y., and T. Morin, "Extranet Multicast in BGP/IP MPLS VPNs", RFC 7900, DOI 10.17487/RFC7900, June 2016, <https://www.rfc-editor.org/info/rfc7900>.

Appendix A. Illustration of SR P2MP Policy and P2MP Tree

Consider the following topology:

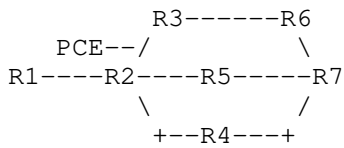

Figure 1

In these examples, the Node-SID of a node Rn is N-SIDn and Adjacency-SID from node Rm to node Rn is A-SIDmn. Interface between Rm and Rn is Lmn.

Assume PCE is provisioned following SR P2MP policy at Root R1 with Tree-ID T-ID:

SR P2MP Policy <R1,T-ID>:
Leaf Nodes: {R2, R6, R7}
Candidate-path 1:
Optimize: IGP metric
Tree-SID: T-SID1

The PCE is responsible for P2MP tree computation. Assume PCE instantiates P2MP trees by signalling non-shared Replication segments i.e. Replication-ID of these Replication Segments is <Root, Tree-ID>. If a Candidat-path can have multiple instances of P2MP trees, the Replication-ID is <Root, Tree-ID, Instance-ID>. In this example, we assume one instance of P2MP tree for a candidate-path. All Replication Segments use the Tree-SID T-SID1 as Replication-SID.

A.1. P2MP Tree with non-adjacent Replication Segments

Assume PCE computes a P2MP tree with Root node R1, Intermediate and Leaf node R2, and Leaf nodes R6 and R7. The PCE instantiates the P2MP tree by stitching Replication Segments at R1, R2, R6 and R7. Replication Segment at R1 replicates to R2. Replication Segment at R2 replicates to R6 and R7. Note nodes R3, R4 and R5 do not have any Replication Segment state for the tree.

The Replication Segment state at nodes R1, R2, R6 and R7 is shown below.

Replication Segment at R1:

Replication Segment <R1,T-ID,R1>:
Replication SID: T-SID1
Replication State:
R2: <T-SID1->L12>

Replication to R2 steers packet directly to the node on interface L12.

Replication Segment at R2:

```
Replication Segment <R1,T-ID,R2>:
 Replication SID: T-SID1
 Replication State:
   R2: <Leaf>
   R6: <N-SID6, T-SID1>
   R7: <N-SID7, T-SID1>
```

R2 is a Bud-Node. It performs role of Leaf as well as a transit node replicating to R6 and R7. Replication to R6, using N-SID6, steers packet via IGP shortest path to that node. Replication to R7, using N-SID7, steers packet via IGP shortest path to R7 via either R5 or R4 based on ECMP hashing.

Replication Segment at R6:

```
Replication Segment <R1,T-ID,R6>:
 Replication SID: T-SID1
 Replication State:
   R6: <Leaf>
```

Replication Segment at R7:

```
Replication Segment <R1,T-ID,R7>:
 Replication SID: T-SID1
 Replication State:
   R7: <Leaf>
```

When a packet is steered into the SR P2MP Policy at R1:

- Since R1 is directly connected to R2, R1 performs PUSH operation with just <T-SID1> label for the replicated copy and sends it to R2 on interface L12.

- R2, as Leaf, performs NEXT operation, pops T-SID1 label and delivers the payload. For replication to R6, R2 performs a PUSH operation of N-SID6, to send <N-SID6,T-SID1> label stack to R3. R3 is the penultimate hop for N-SID6; it performs penultimate hop popping, which corresponds to the NEXT operation and the packet is then sent to R6 with <T-SID1> in the label stack. For replication to R7, R2 performs a PUSH operation of N-SID7, to send <N-SID7,T-SID1> label stack to R4, one of IGP ECMP nexthops towards R7. R4 is the penultimate hop for N-SID6; it performs penultimate hop popping, which corresponds to the NEXT operation

and the packet is then sent to R7 with <T-SID1> in the label stack.

- R6, as Leaf, performs NEXT operation, pops T-SID1 label and delivers the payload.

- R7, as Leaf, performs NEXT operation, pops R-SID7 label and delivers the payload.

A.2. P2MP Tree with adjacent Replication Segments

Assume PCE computes a P2MP tree with Root node R1, Intermediate and Leaf node R2, Intermediate nodes R3 and R5, and Leaf nodes R6 and R7. The PCE instantiates the P2MP tree by stitching Replication Segments at R1, R2, R3, R5, R6 and R7. Replication Segment at R1 replicates to R2. Replication Segment at R2 replicates to R3 and R5. Replication segment at R3 replicates to R6. Replication segment at R5 replicates to R7. Note node R4 does not have any Replication Segment state for the tree.

The Replication Segment state at nodes R1, R2, R3, R5, R6 and R7 is shown below.

Replication Segment at R1:

```
Replication Segment <R1,T-ID,R1>:
 Replication SID: T-SID1
 Replication State:
   R2: <T-SID1->L12>
```

Replication to R2 steers packet directly to tje node on interface L12.

Replication Segment at R2:

```
Replication Segment <R1,T-ID,R2>:
 Replication SID: T-SID1
 Replication State:
   R2: <Leaf>
   R3: <T-SID1->L23>
   R5: <T-SID1->L25>
```

R2 is a Bud-Node. It performs role of Leaf as well as a transit node replicating to R3 and R5. Replication to R3, steers packet directly to the node on L23. Replication to R5, steers packet directly to the node on L25.

Replication Segment at R3:

```
Replication Segment <R1,T-ID,R3>:
 Replication SID: T-SID1
 Replication State:
   R6: <T-SID1->L36>
```

Replication to R6, steers packet directly to the node on L36.

Replication Segment at R5:

```
Replication Segment <R1,T-ID,R5>:
 Replication SID: T-SID1
 Replication State:
   R7: <T-SID1->L57>
```

Replication to R7, steers packet directly to the node on L57.

Replication Segment at R6:

```
Replication Segment <R1,T-ID,R6>:
 Replication SID: T-SID1
 Replication State:
   R6: <Leaf>
```

Replication Segment at R7:

```
Replication Segment <R1,T-ID,R7>:
 Replication SID: T-SID1
 Replication State:
   R7: <Leaf>
```

When a packet is steered into the SR P2MP Policy at R1:

- Since R1 is directly connected to R2, R1 performs PUSH operation with just <T-SID1> label for the replicated copy and sends it to R2 on interface L12.

- R2, as Leaf, performs NEXT operation, pops T-SID1 label and delivers the payload. It also performs CONTINUE operation on T-SID1 for replication to R3 and R5. For replication to R6, R2 sends <T-SID1> label stack to R3 on interface L23. For replication to R5, R2 sends <T-SID1> label stack to R5 on interface L25.

- R3 performs CONTINUE operation on T-SID1 for replication to R6 and sends <T-SID1> label stack to R6 on interface L36.

- R5 performs CONTINUE operation on T-SID1 for replication to R7 and sends <T-SID1> label stack to R7 on interface L57.

- R6, as Leaf, performs NEXT operation, pops T-SID1 label and delivers the payload.

- R7, as Leaf, performs NEXT operation, pops R-SID7 label and delivers the payload.

Authors' Addresses

Daniel Voyer (editor)
Bell Canada
Montreal
CA

Email: daniel.voyer@bell.ca

Clarence Filsfils
Cisco Systems, Inc.
Brussels
BE

Email: cfilsfil@cisco.com

Rishabh Parekh
Cisco Systems, Inc.
San Jose
US

Email: riparekh@cisco.com

Hooman Bidgoli
Nokia
Ottawa
CA

Email: hooman.bidgoli@nokia.com

Zhaohui Zhang
Juniper Networks

Email: zzhang@juniper.net