Workgroup: Network Working Group
Internet-Draft: draft-chen-pim-mrh6-07
Published: 28 March 2024
Intended Status: Standards Track
Expires: 29 September 2024
Authors: H. Chen (Futurewei), M. McBride (Futurewei), Y. Fan (Casa Systems), Z. Li (Huawei)
X. Geng (Huawei), M. Toy (Verizon), G. Mishra (Verizon), Y. Liu (China Mobile)
A. Wang (China Telecom), L. Liu (Fujitsu), X. Liu (Alef Edge)

Stateless Traffic Engineering Multicast using MRH

Abstract

This document describes a stateless traffic engineering (TE) multicast along an explicit P2MP Path/Tree using an IPv6 extension header called TE multicast routing header (MRH). The MRH with the path encoded in link numbers is added into a packet to be multicast at the ingress. The packet is delivered to the egresses along the path. There is no state stored in the core of the network.

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] [RFC8174] when, and only when, they appear in all capitals, as shown here.

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 29 September 2024.

Table of Contents

1. Introduction

[I-D.ietf-pim-sr-p2mp-policy] proposes a solution for a SR P2MP path. A multicast P2MP tree is created for the path and the state of the tree is instantiated in the forwarding plane by a controller at Root, intermediate Replication nodes and Leaves of the tree.

[I-D.chen-pim-srv6-p2mp-path] proposes a stateless solution for a SR P2MP path. The overhead for encoding the P2MP path using SIDs in SRH may be large.

This document describes a stateless traffic engineering (TE) multicast along an explicit P2MP Path/Tree using an IPv6 extension header called TE multicast routing header (MRH). The MRH with the path encoded in link numbers is added into a packet to be multicast at the ingress. The packet is delivered to the egresses along the path. There is no state stored in the core of the network.

2. Overview of TE Multicast using MRH

Figure 1 shows an example network having nodes P1, P2, P3, P4, PE1 to PE10. For each of the links connected to a node, a number called local link number (or link number for short) is assigned to it. For example, PE1 has three links: link from PE1 to CE1, link from PE1 to P1, and link from PE1 to PE10. These three links have link numbers 1, 2 and 3 respectively on PE1. P1 has five links: P1 to PE1, P1 to P2, P1 to P3, P1 to PE8 and P1 to PE9. These five links have link numbers 1, 2, 3, 4 and 5 respectively on P1. P3 has two links: P3 to P1 and P3 to P4. These two links have link numbers 1 and 2 respectively on P3.

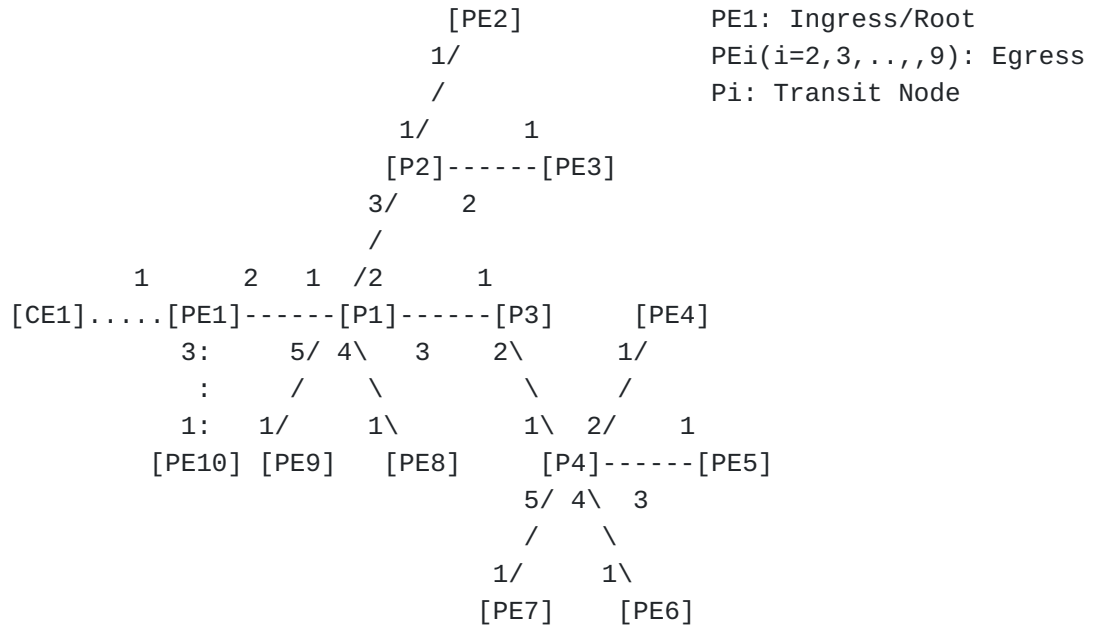

Figure 1: Network with P2MP Path from ingress PE1 to egresses PE2, PE3, ..., PE9

The P2MP path from ingress PE1 via P1 towards egresses PE2 to PE9 in Figure 1 is represented by the link numbers along the path: PE1's link number 2; P1's link numbers 2, 3, 4 and 5; P2's link numbers 1 and 2; P3's link number 2; and P4's link numbers 2, 3, 4 and 5.

A controller such as PCE as a controller has the link numbers of the links originating at every node. The controller can send the ingress the P2MP path represented by the link numbers of the links along the path.

After receiving a packet from traffic source CE1, ingress PE1 encapsulates the packet in a MRH with a P2MP path represented by the link numbers along the path. The packet in the MRH is transmitted along the path to the egresses of the path.

When a node such as P1 receives a packet with the MRH, the node gets/pops each of its link numbers, finds the address of the next hop from a neighbor address table using the link number (i.e., the link number such as 3 of the link from the node to the next hop such as P3), and sends the packet to the next hop (such as P3).

Figure 2 shows the neighbor IPv6 address table of P1. The table has five rows of link number, link type and IPv6 address for the five links of P1. The first row has link number 1, link type P2P for link from P1 to next hop PE1 and PE1's IPv6 address. The 2nd row has link number 2, link type P2P for link from P1 to next hop P2 and P2's IPv6 address. The 3rd row has link number 3, link type P2P for link from P1 to next hop P3 and P3's IPv6 address. The 4-th row has link number 4, link type P2P for link from P1 to next hop PE8 and PE8's IPv6 address. The 5-th row has link number 5, link type P2P for link from P1 to next hop PE9 and PE9's IPv6 address.

Link number	Link type	Address of next hop
1	P2P	IPv6 address of PE1
2	P2P	IPv6 address of P2
3	P2P	IPv6 address of P3
4	P2P	IPv6 address of PE8
5	P2P	IPv6 address of PE9

Figure 2: Neighbor IPv6 address table of P1

Using link number 1, P1 gets PE1's IPv6 address from the table; using link number 2, P1 gets P2's IPv6 address from the table; using link number 3, P1 gets P3's IPv6 address from the table; and so on.

3. Encoding of P2MP Path/Tree

Figure 3 shows the encoding of the P2MP path/tree in Figure 1 from PE1 via P1 to PE2, PE3, ..., PE9. Each link on the tree is encoded or represented by the link number of the link. For example, the link from PE1 to P1 on the tree is encoded by link number 2 of the link on PE1. The link from P1 to P2 on the tree is encoded by link number 2 of the link on P1. The link from P1 to P3 on the tree is encoded by link number 3 of the link on P1.

For each link from its upstream node to its downstream (or say next hop) node on the tree, three fields are used for the link: 1). link

number (Link-No for short) field for storing the link number on the upstream node, 2). number of branches (N-Branches for short) for storing the number branches/sub-trees of the tree from the next hop node, and 3). size of branches+ (S-Branches+ for short) for storing the size of branches of the tree from the next hop node and the fields following. The size is in bytes.

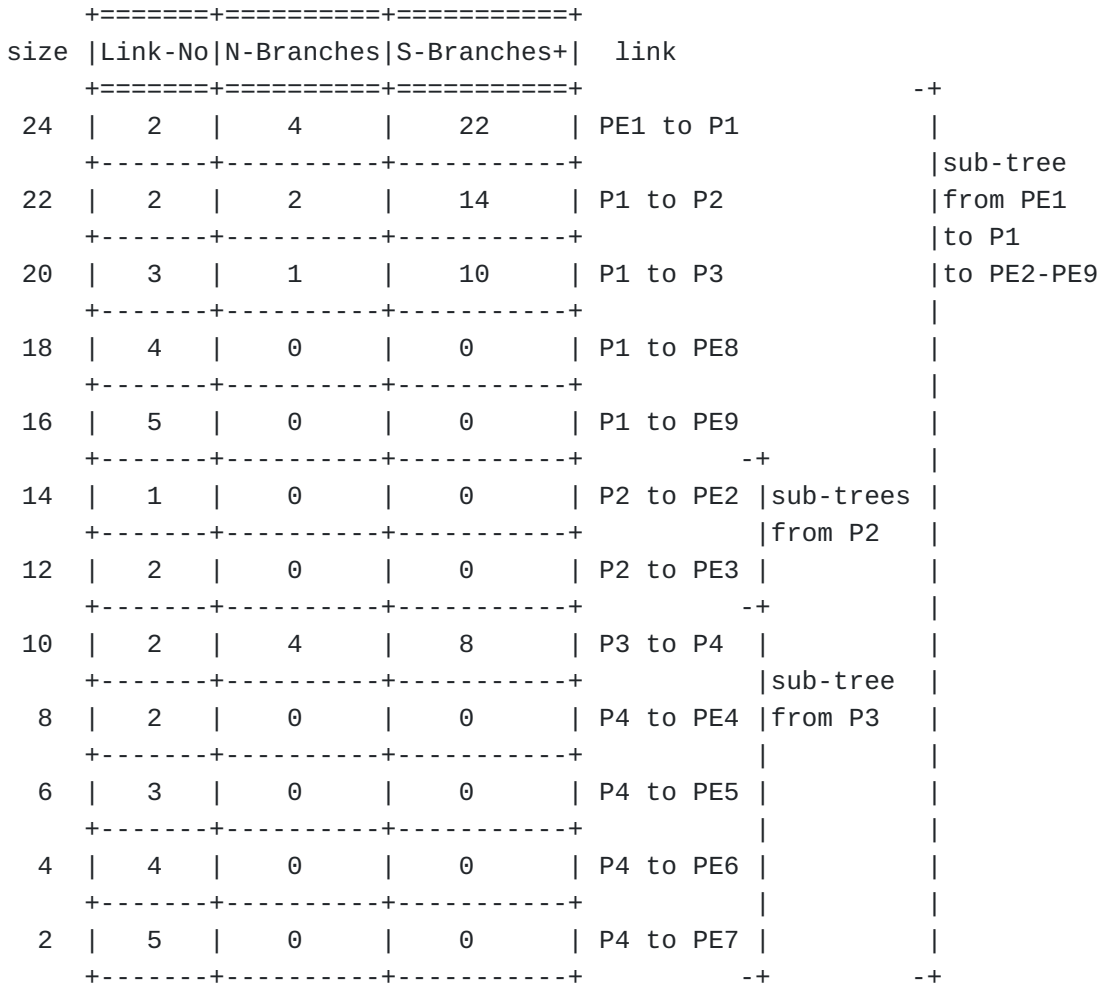

Figure 3: Encoding of tree from PE1 via P1 to PE2, PE3, ..., PE9

For example, for the link from PE1 to P1 on the tree, Link-No field has value of 2 since the link number is 2 for the link on PE1; N-Branches field has value of 4 since there are 4 branches from P1; S-Branches+ field has value of 22 (bytes) since 22 bytes are used for storing/encoding 4 branches from P1 when the three fields Link-No, N-Branches and S-Branches+ for a link occupy 2 bytes in total.

For the link from P1 to P2 on the tree, Link-No field has value of 2 since the link number is 2 for the link on P1; N-Branches field has value of 2 since there are two branches from P2; S-Branches+ field

has value of 14 (bytes) since 14 bytes are used for storing/encoding the two branches (i.e., sub-trees) from P2 and the fields following.

For the link from P1 to P3 on the tree, Link-No field has value of 3 since the link number is 3 for the link on P1; N-Branches field has value of 1 since there is one branch (i.e., sub-tree) from P3; S-Branches+ field has value of 10 (bytes) since 10 bytes are used for storing/encoding the branch from P3 and no other fields following.

For the link from P2 to egress PE2 on the tree, Link-No field has value of 1 since the link number is 1 for the link on P2; N-Branches field has value of 0 since there is no branch (i.e., sub-tree) from PE2; S-Branches+ field has value of 0 (bytes).

Figure 4 shows an enhancement of encoding the P2MP path/tree in Figure 1 using L flag. The L flag added for a link indicates whether the next hop is a leaf node. When L is set to one indicating the next hop is a leaf (i.e., egress), the "N-Branches" and "S-Branches+" fields are removed. There are 8 links to leaves (i.e., egress nodes) on the tree, which are 2 links from P2 to PE2 and PE3; 4 links from P4 to PE4, PE5, PE6 and PE7; and 2 links from P1 to PE8 and PE9. These 8 links have their L flags set to one. The N-Branches and S-Branches+ fields for these 8 links are removed from the encoding of the tree. This reduces the space/memory for encoding the tree.

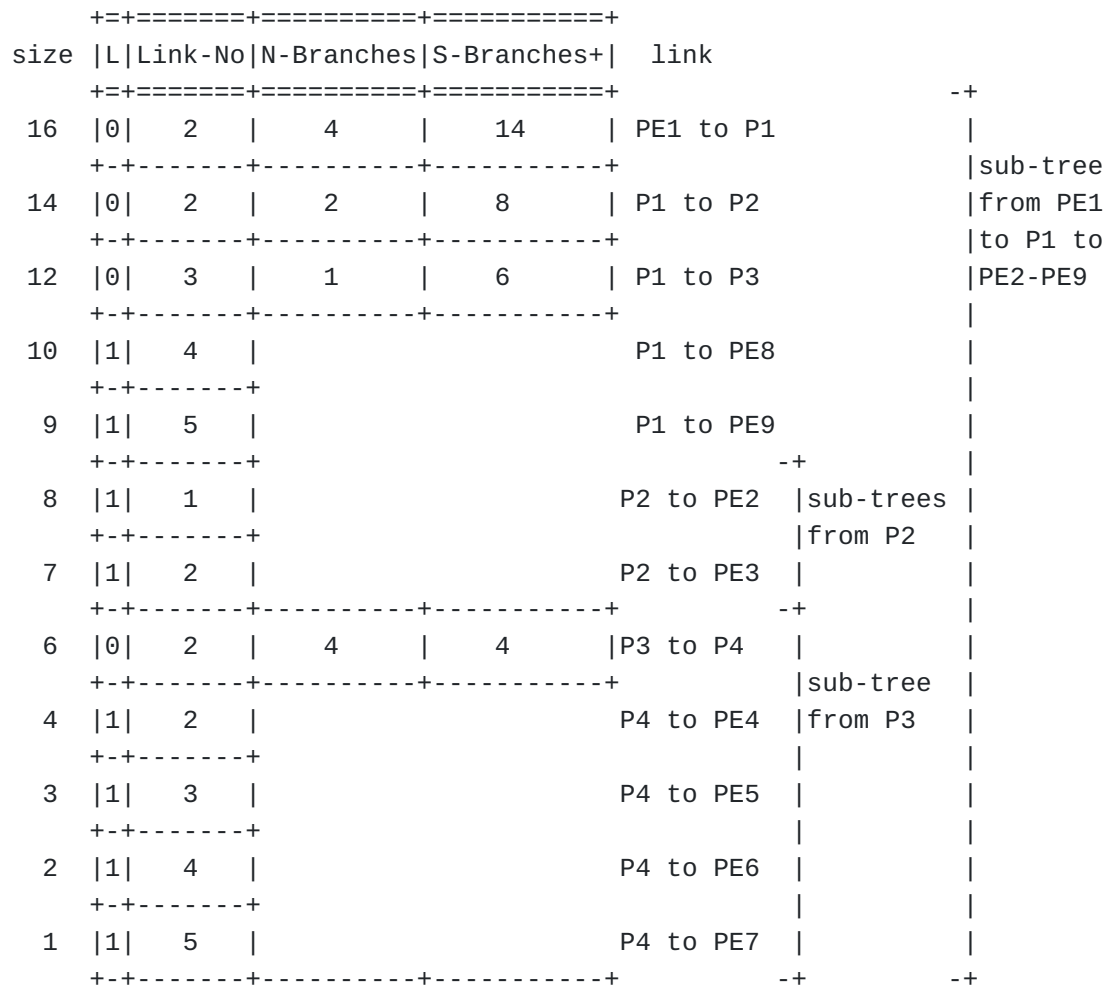

Figure 4: Encoding of tree from PE1 via P1 to PE2, PE3, ..., PE9 with L flag

For example, suppose that two fields L and Link-No with padding (padding is not shown) for a link to leaf occupy 1 byte; four fields L, Link-No, N-Branches and S-Branches+ for a link to a transit node occupy 2 bytes. In this case, the S-Branches+ field for link from PE1 to P1 is 14 (bytes), which is the size of the fields for 11 links (i.e., link from P1 to P2, link from P1 to P3, ..., link from P4 to PE7) encoding the sub-trees from P1 to PE2, PE3, ..., PE9. The S-Branches+ field for link from P1 to P2 is 8 (bytes), which is the size of the fields for 7 links (i.e., link from P2 to PE2, link from P2 to PE3, link from P3 to P4, and 4 links from P4 to PE4 - PE7) encoding the sub-trees from P1 to PE2 and PE3, and the fields following them.

Encoding the tree without L flag occupies 24 bytes in total as illustrated in Figure 3. Encoding the tree with L flag occupies 16 bytes in total as illustrated in Figure 4. 8 bytes (or say 33% of space/memory) are saved/reduced.

Figure 5 shows an enhancement of encoding the branch/sub-tree from P3 to PE4, PE5, PE6 and PE7 in Figure 1 using B flag. The B flag added for a link indicates whether the link bits+ are used to represent the link numbers and other link related information such as whether links' remote end nodes (i.e., next hops) are leaves. When B flag for the link from upstream node such as P3 to downstream node such as P4 is set to one indicating the link bits+ are used, the links from downstream node such as P4 on the tree are encoded or represented by link bits+. A link bits+ contains a P (short for plus leaf) field, S-Bits (short for size of the bits in a unit such as byte) field, and Bits field.

P field has a value such as 1 indicating that a bit with value of 1 (one) in Bits field means that the corresponding link is on the branch and the link's next hop is a leaf. There are four links to leaves (i.e., egress nodes) from P4, which are link from P4 to PE4 with link number 2, link from P4 to PE5 with link number 3, link from P4 to PE6 with link number 4, and link from P4 to PE7 with link number 5. These four links are represented by the 2-th bit, 3-th bit, 4-th bit and 5-th bit in the Bits field (from left to right) respectively.

S-Bits field has a value such as 1 indicating the size of the Bits field in a unit such as byte.

Suppose that the fields (i.e., L, B, Link-No, No-Branches and S-Branches+) for the link from P3 to P4 occupy 2 bytes; P and S-Bits occupy 1 byte. In this case, the link bits+ for the links from P4 occupy 2 bytes, encoding the branch from P3 uses 4 bytes in total.

The number of branches from P4 is the number of bits with value of one in the Bits field. The N-Branches field for the link from P3 to P4 is used for other purpose. For example, N-Branches field and S-Branches+ field combined to represent the size of the branches plus the fields following (i.e., S-Branches+).

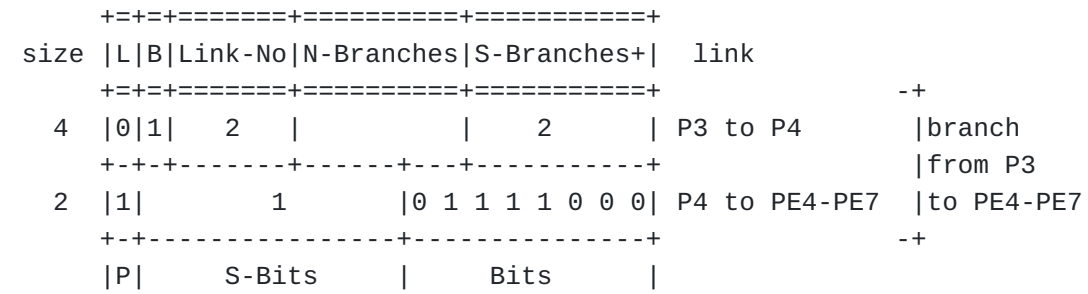

Figure 5: Encoding of branch from P3 via P4 to PE4, PE5, PE6, PE7 with B flag

Figure 6 shows an enhancement of encoding the branch part from PE1 via P1 to P2, P3, PE8 and PE9 on the P2MP tree/path in Figure 1 using B flag. B flag for the link from upstream node PE1 to downstream node P1 is set to one indicating that the link bits+ are used to represent the information about the links on the tree from downstream node P1.

P field has a value such as 0 indicating that a bit with value of 1 (one) in Bits field means that the corresponding link from downstream node P1 is on the tree.

S-Bits field has a value such as 1 indicating the size of Bits field in a unit such as byte.

Bits field of one byte has the second (2-th) bit, third (3-th) bit, 4-th bit and 5-th bit set to one (from left to right), indicating four links from P1 on the tree: the link from P1 to P2 with link number 2, link from P1 to P3 with link number 3, link from P1 to PE8 with link number 4 and link from P1 to PE9 with link number 5. The fields for each of these links do not have Link-No field, which is called reduced fields.

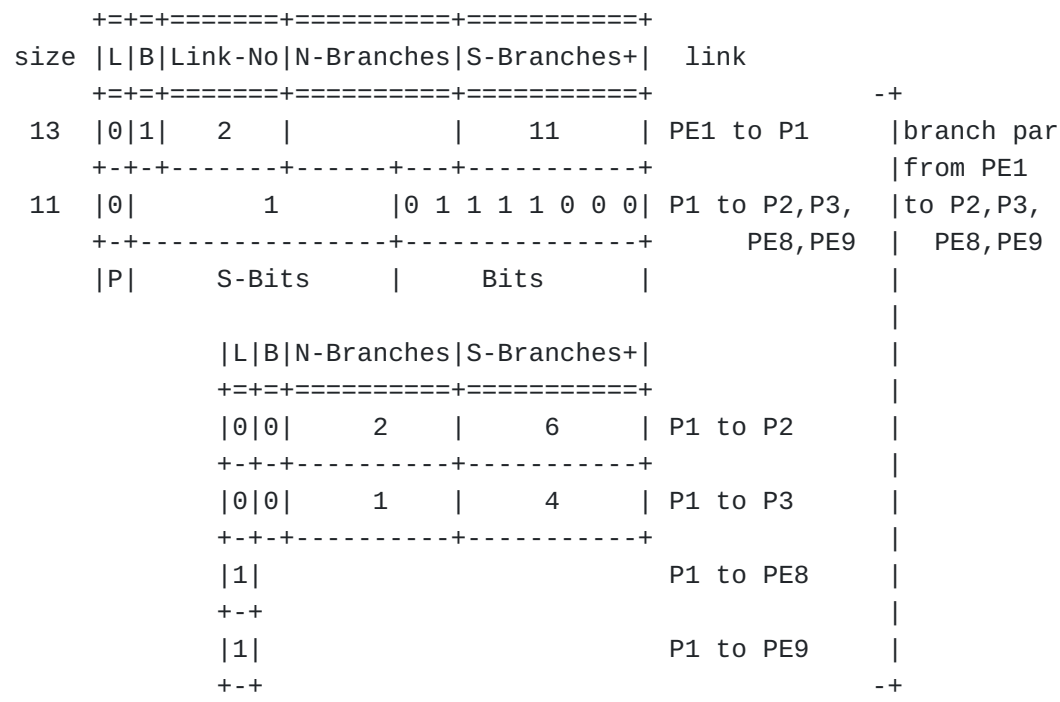

Figure 6: Encoding of branch part from PE1 via P1 to P2, P3, PE8, PE9 with B flag

The reduced fields (i.e., L = 0, B = 0, No-Branches = 2, S-Branches+ = 6) for the link from P1 to P2 indicates that the link's next hop is not leaf, link bits+ is not used for the branches from P2, the

number of branches from P2 is 2 and the size of the branches from P2 plus the fields following is 6.

The reduced fields (i.e., L = 0, B = 0, No-Branches = 1, S-Branches+ = 4) for the link from P1 to P3 indicates that the link's next hop is not leaf, link bits+ is not used for the branches from P3, the number of branches from P3 is 1 and the size of the branches from P3 plus the fields following is 4.

The reduced fields (i.e., L = 1) for the link from P1 to PE8 indicates that the link's next hop PE8 is a leaf.

The reduced fields (i.e., L = 1) for the link from P1 to PE9 indicates that the link's next hop PE9 is a leaf.

Suppose the fields (i.e., L, B, Link-No, No-Branches and S-Branches+) for a link to a transit node occupy 2 bytes; P and S-Bits occupy 1 byte; the reduced fields (i.e., L, B, No-Branches and S-Branches+) for a link to a transit node occupy 11 bits, the reduced fields (i.e., L) for a link to a leaf occupy 1 bit. In this case the reduced fields for four links from P1 to P2, P3, PE8 and PE9 uses 3 bytes (i.e., 24 bits). Encoding the branch part from PE1 via P1 to P2, P3, PE8 and PE9 uses 7 bytes.

Figure 7 shows an enhancement of encoding the P2MP path/tree in Figure 1 using both L and B flags.

The branch part from PE1 via P1 to P2, P3, PE8 and PE9 is the same as the one in Figure 6 and occupies 7 bytes.

The link from P2 to PE2 is represented by L = 1 and Link-No = 1. The link from P2 to PE3 is represented by L = 1 and Link-No = 2. These two links occupy 2 bytes. In an option, when L = 1, the Link-No field and Pad are combined to represent link number.

The branch from P3 is the same as the one in Figure 5 and occupies 4 bytes. Encoding the tree of 13 nodes uses 13 bytes in total.

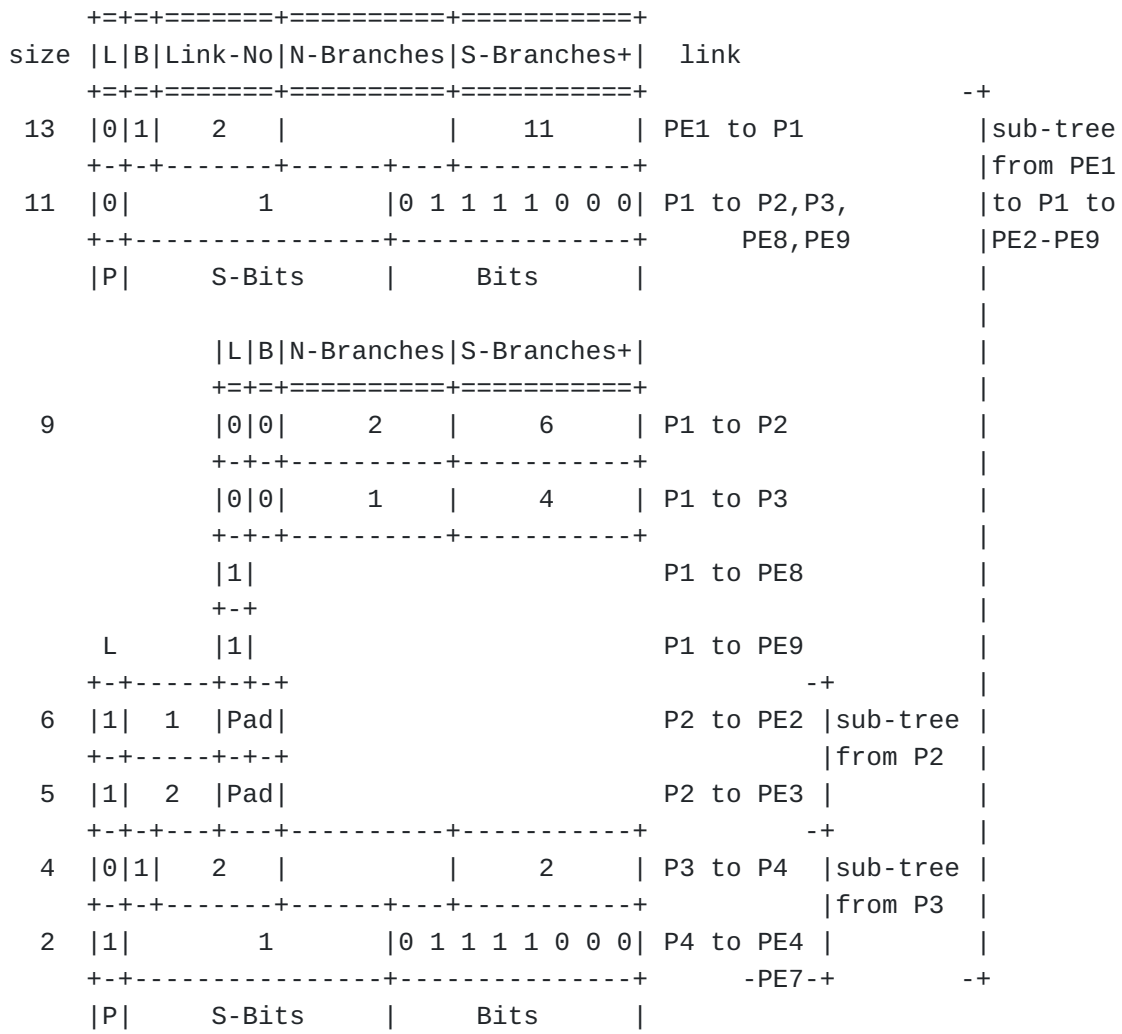

Figure 7: Encoding of tree from PE1 to PE2 - PE9 with L and B

4. Multicast Routing Header (MRH)

Figure 8 shows a Multicast Routing Header (MRH) in an IPv6 packet. The IPv6 packet comprises an IPv6 header with a destination address (DA) and source address (SA) of IPv6, a routing header with Routing type (TBD) indicating MRH and an IP multicast datagram. The routing header is indicated by the Next Header in the IPv6 header.

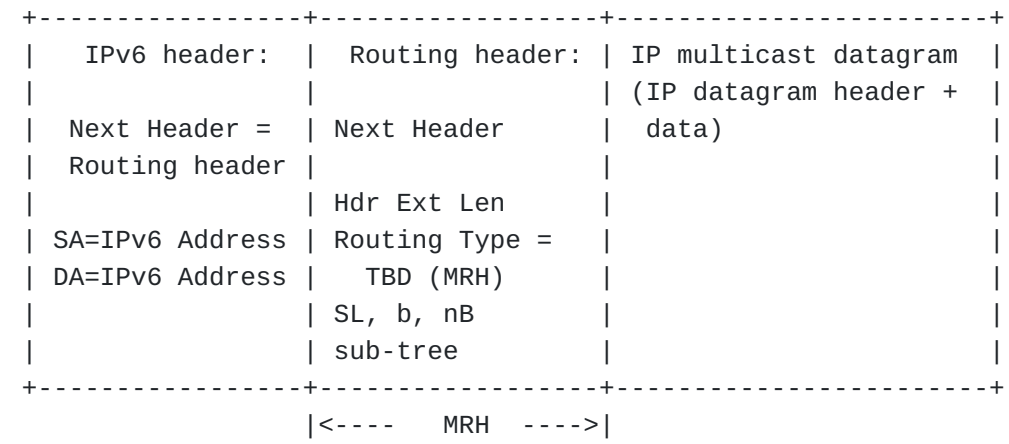

Figure 8: Multicast Routing Header (MRH) in IPv6 packet

The format of the MRH is shown in Figure 9.

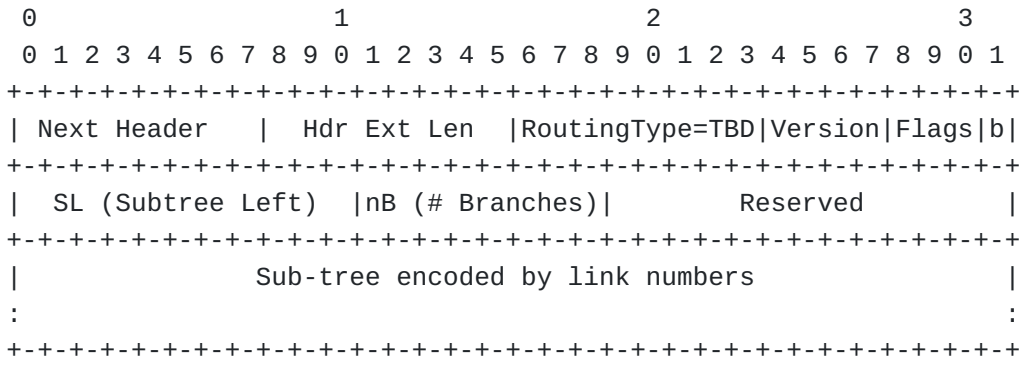

Figure 9: Format of Multicast Routing Header (MRH)

The MRH has the following fields:

Next Header: The type of the header after the MRH. Either another extension header or the type of IP multicast datagram in the packet.

Hdr Ext Len: Its value indicates the length of the MRH in a unit of 64 bits (i.e., 8 bytes) excluding the first 8 bytes.

Routing Type: Its value TBD indicates that the routing header is a Multicast Routing Header (MRH) for TE multicast.

Version: The Version of the MRH. This document specifies Version zero.

Flags: A 1-bit b flag is defined. b = 1 indicates the links directly from the root of the sub-tree are encoded by bits+.

Sub-tree Left (SL): Its value as a pointer points to the sub-tree.

Number of Branches (nB):
Its value indicates the number of branches from the root of the sub-tree.

Sub-tree: Its value encodes the TE multicast sub-tree using link numbers.

The P2MP path/tree from PE1 via P1 to PE2 - PE9 in Figure 1 is encoded by the link numbers of the links on the tree as illustrated in Figure 7. The link from ingress PE1 to P1 is encoded by the link number of the link on PE1, which is 2, the number of branches from P1 on the tree, and the size of the branches+ from P1, which is 11. The number of branches from P1 is the number of bits with value 1 in the Bits field for the links from P1 to P2, P3, PE8 and PE9, which is 4 since there are 4 bits with value 1 in the Bits field.

For an IP multicast packet to be transmitted by the P2MP path/tree, PE1 constructs an IPv6 packet for each sub-tree/branch from PE1 and sends the packet containing a MRH and the IP multicast packet to the next hop along the sub-tree.

The P2MP path/tree has one sub-tree from PE1 via P1 to PE2 - PE9. PE1 finds P1's IPv6 address from PE1's neighbor IPv6 address table using the link number 2 and sets DA (destination address) of the IPv6 packet to P1's IPv6 address and SA (source address) of the IPv6 packet to PE1's IPv6 address. PE1 builds the MRH based on the encoding of the tree in Figure 7 through setting b flag to 1, which is the value of B for the link from PE1 to P1, SL to 11, which is the value of S-Branches+ for the link from PE1 to P1. nB is not set since B = 1. Figure 10 shows the IPv6 packet to be sent to P1, which is received by P1. Note that some details are not shown.

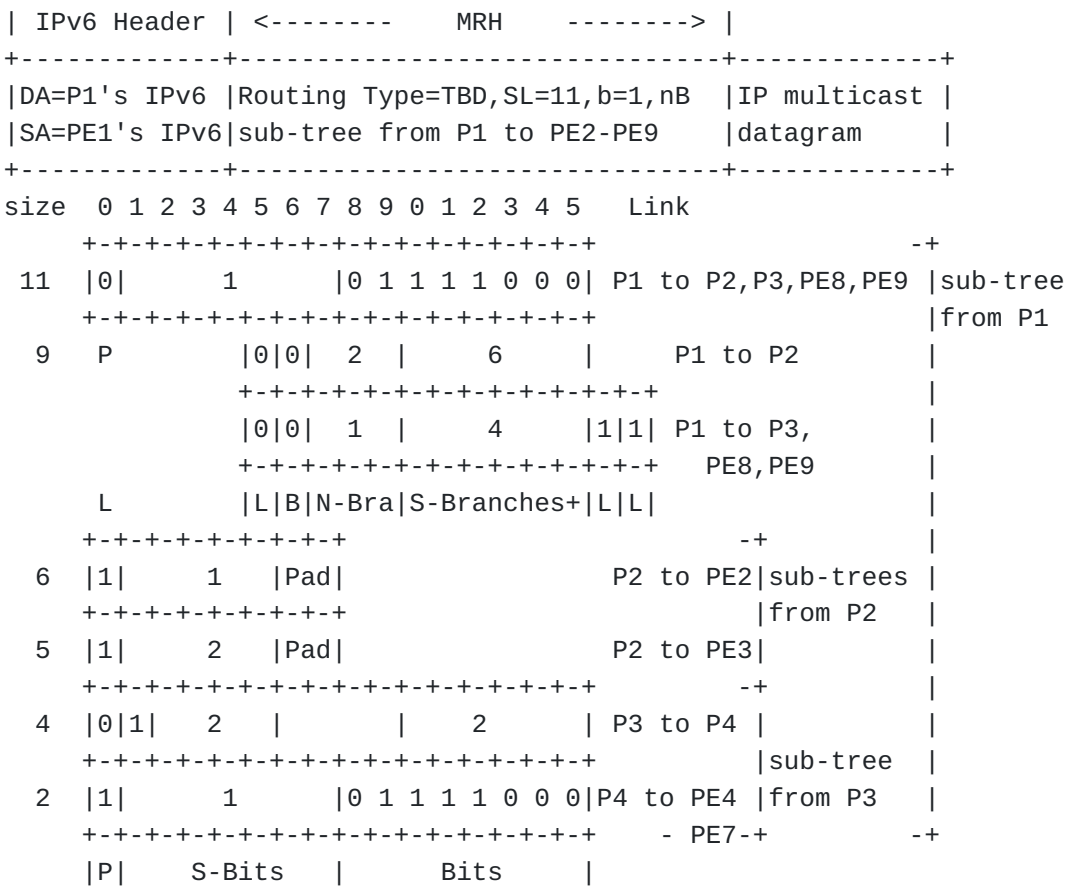

Figure 10: IPv6 packet with MRH received by P1

The number of branches from P1 is 4, which is the number of bits with value 1 in the Bits field for the links from P1 to P2, P3, PE8 and PE9. This is determined by b = 1 and the Bits+ fields pointed by SL = 11.

There are 4 links from P1. The 2-th bit in the Bits field has value 1 indicating the link with link number 2, which is the link from P1 to P2. The reduced fields (i.e., L = 0, B = 0, No-Branches = 2, S-Branches+ = 6) for the link indicates that the link's next hop P2 is not leaf, link bits+ is not used for the branches from P2, the number of branches from P2 is 2 and the size of the branches from P2 plus the fields following is 6.

The 3-th bit has value 1 indicating the link with link number 3, which is the link from P1 to P3. The reduced fields (i.e., L = 0, B = 0, No-Branches = 1, S-Branches+ = 4) for the link indicates that the link's next hop P3 is not leaf, link bits+ is not used for the branches from P3, the number of branches from P3 Is 1 and the size of the branches from P3 plus the fields following is 4.

The 4-th bit has value 1 indicating the link with link number 4, which is the link from P1 to PE8. PE8 is a leaf, which is indicated

by the first L = 1. The 5-th bit has value 1 indicating the link with link number 5, which is the link from P1 to PE9. PE9 is a leaf indicated by the second L = 1.

The link number of the link from P2 to leaf PE2 is 1 on P2. The link number of the link from P2 to leaf PE3 is 2 on P2. The link number of the link from P3 to P4 is 2 on P3. The size of branches from P4 is 2. The link numbers of the links from P4 to leaf PE4, PE5, PE6 and PE7 are 2, 3, 4 and 5 on P4 respectively.

After receiving the IPv6 packet from PE1, P1 determines whether the packet's next header is a MRH through checking if the next header is routing header and routing type in the routing header is TBD for MRH. When the next header is the MRH, P1 duplicates the packet for each link/branch/sub-tree from P1 and sends the packet copy with an updated MRH (note: only SL, b and nB are updated) to the next hop along the branch. Since b = 1, the number of bits with value 1 in the Bits field of Bits+ pointed by SL = 11 is the number of branches/links from P1 according to Figure 10, which is 4.

The 2-th bit in the Bits field has value 1 indicating the link with link number 2, which is the link from P1 to P2. The first reduced fields is for link from P1 to P2 and indicates the number of branches from P2 is 2 and the size of branches from P2 plus the ones following them is 6.

P1 duplicates the packet for the link with link number 2, finds P2's IPv6 address from P1's neighbor IPv6 address table using the link number 2 of the link from P1 to P2 and sets DA of the packet to P2's IPv6 address. P1 updates the MRH based on the encoding of the sub-tree in Figure 10 through setting SL, b and nB to S-Branches+ = 6, B = 0 and N-Branches = 2 for the link from P1 to P2 respectively. Figure 11 shows the packet to be sent to P2, which is received by P2.

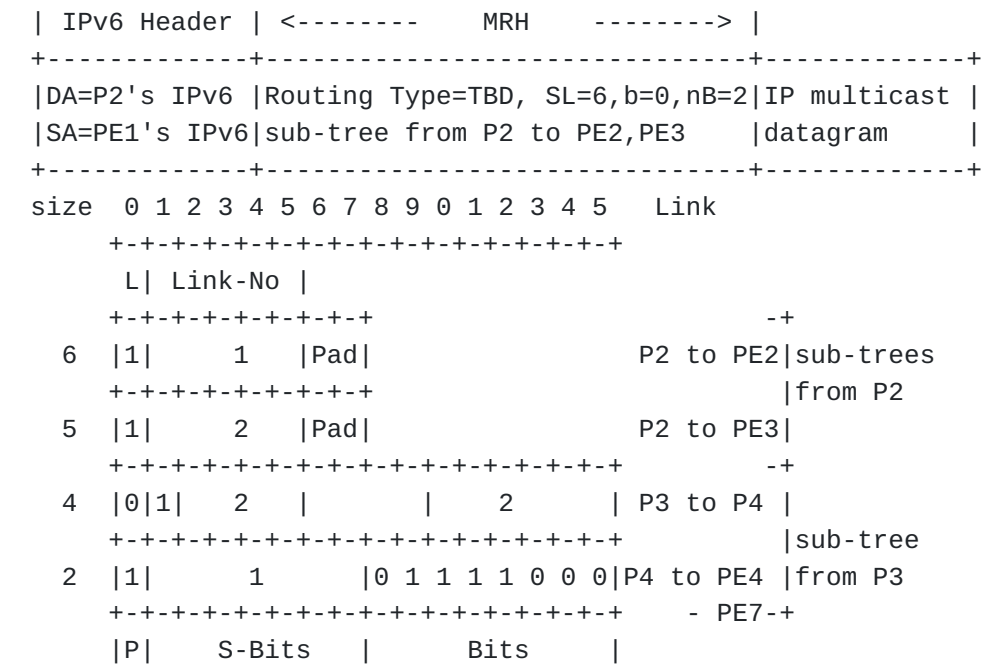

Figure 11: IPv6 packet with MRH received by P2

The number of branches from P2 is 2. The link number of the link from P2 to PE2 is 1 on P2. The link number of the link from P2 to PE3 is 2 on P2. PE2 and PE3 are leaves (i.e., egresses), which are indicated by L = 1.

The fields for encoding the branch/sub-tree from P3 follows the branches/sub-trees from P2.

The 3-th bit has value 1 indicating the link with link number 3, which is the link from P1 to P3. The second reduced fields is for link from P1 to P3 and indicates the number of branches from P3 is 1 and the size of branches from P3 plus the ones following them is 4.

P1 duplicates the packet for the link with link number 3 (which is link from P1 to P3) and sends the packet copy with an updated MRH to P3.

The 4-th bit has value 1 indicating the link with link number 4, which is the link from P1 to PE8. PE8 is a leaf, which is indicated by the first L = 1.

P1 duplicates the packet for the link with link number 4 and sends the packet copy with an updated MRH to PE8.

The 5-th bit has value 1 indicating the link with link number 5, which is the link from P1 to PE9. PE9 is a leaf, which is indicated by the second L = 1.

P1 duplicates the packet for the link with link number 5 and sends the packet copy with an updated MRH to PE9.

P1 duplicates the packet for the link with link number 3 (which is link from P1 to P3), finds P3's IPv6 address from P1's neighbor IPv6 address table using the link number 3 of the link from P1 to P3 and sets DA of the packet to P3's IPv6 address. P1 updates the MRH based on the encoding of the sub-tree in Figure 10 through setting SL, b and nB to S-Branches+ = 4, B = 0 and N-Branches = 1 in the reduced fields for the link from P1 to P3 respectively. Figure 12 shows the packet to be sent to P3, which is received by P3.

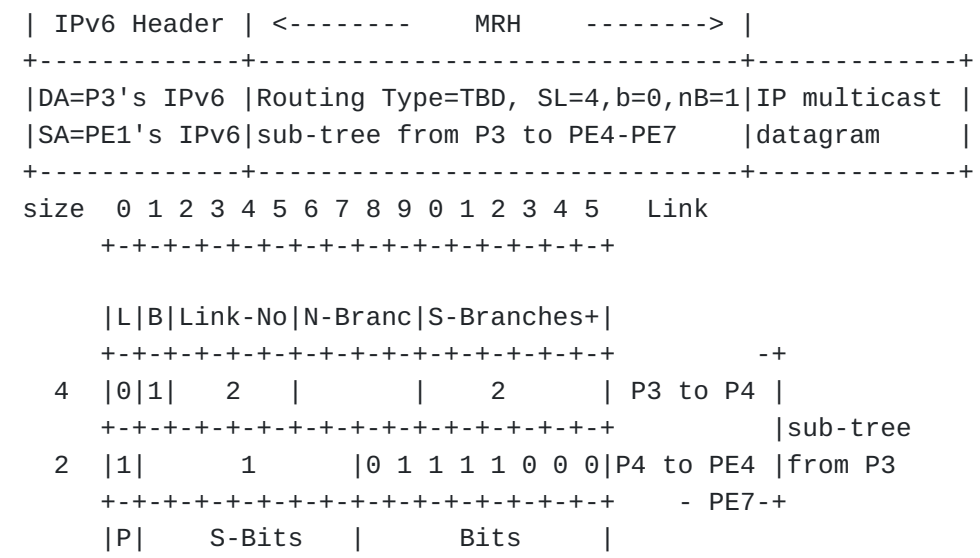

Figure 12: IPv6 packet with MRH received by P3

The number of branches from P3 is 1. The link number of the link from P3 to P4 is 2 on P3. The size of branches from P4 is 2. B = 1 for the link from P3 to P4 indicates that the links from P4 are encoded by link bits+. The number of branches from P4 is the number of bits with value 1 in the Bits field for the links from P4 to PE4 - PE7, which is 4 since there are 4 bits with value 1 in the Bits field.

After receiving the IPv6 packet from P1, P3 determines whether the packet's next header is a MRH. When the packet's next header is the MRH, P3 duplicates the packet for each link/branch/sub-tree from P3 and sends the packet copy with an updated MRH to the next hop along the branch. There is one branch from P3 according to the sub-tree remaining in the MRH in the packet received by P3. The branch/sub-tree is from the link from P3 to P4 towards PE4 - PE7.

P3 duplicates the packet for the branch/sub-tree, finds P4's IPv6 address from P3's neighbor IPv6 address table using the link number 2 of the link from P3 to P4 and sets DA of the packet to P4's IPv6

address. P3 updates the MRH based on the encoding of the sub-tree in Figure 12 through setting SL and b to S-Branches+ = 2 and B = 1 for the link from P3 to P4 respectively. nB is not set or used since b = 1. Figure 13 shows the packet to be sent to P4, which is received by P4.

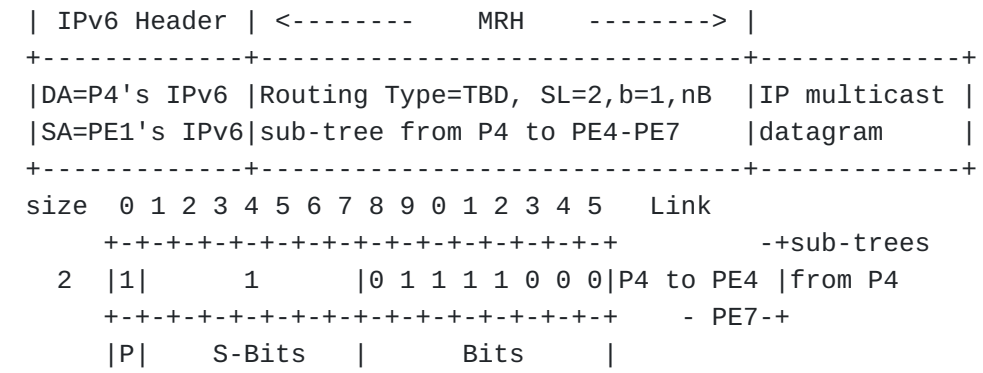

Figure 13: IPv6 packet with MRH received by P4

After receiving the IPv6 packet from P3, P4 determines whether the packet's next header is a MRH. When the packet's next header is the MRH, P4 duplicates the packet for each branch/link from P4 and sends the packet copy with an updated MRH to the next hop along the branch. There are 4 branches/links from P4 according to the sub-tree remaining in the MRH in the packet received by P4.

The 2-th bit in the Bits field has value 1 indicating the link with link number 2, which is the link from P4 to leaf PE4.

The 3-th bit has value 1 indicating the link with link number 3, which is the link from P4 to leaf PE5.

The 4-th bit has value 1 indicating the link with link number 4, which is the link from P4 to leaf PE6.

The 5-th bit has value 1 indicating the link with link number 5, which is the link from P4 to leaf PE7.

P4 duplicates the packet for the first branch, finds PE4's IPv6 address from P4's neighbor IPv6 address table using the link number 2 of the link from P4 to PE4 and sets DA of the packet to PE4's IPv6 address. P4 updates the MRH based on the encoding of the sub-tree in Figure 13 through setting SL to 0 since PE4 is a leaf. Figure 14 shows the packet to be sent to PE4, which is received by PE4. Similarly, P4 duplicates the packet for each of the other links, updates the MRH and sends the packet copy to each of PE5, PE6 and PE7.

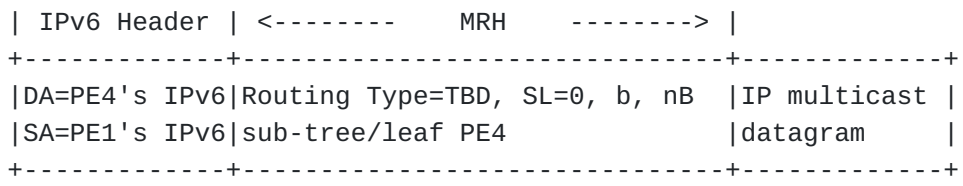

Figure 14: IPv6 packet with MRH received by PE4

After receiving the packet from P4, PE4 determines whether the packet's next header is a MRH. When the packet's next header is the MRH, PE4 checks if PE4 itself is a leaf (i.e., egress) through checking whether SL is 0. When PE4 is a leaf, PE4 decapsulates the packet and sends the IP multicast datagram to the IP multicast forwarding module.

Alternatively, after receiving the packet from P3 and determining that the packet's next header is a MRH, P4 checks if each of its next hops is a leaf. When the next hop is a leaf, P4 decapsulates the packet and sends the IP multicast datagram to the next hop. Since 4 next hops PE4 - PE7 are leaves, P4 sends the IP multicast datagram to PE4 - PE7.

5. Procedures/Behaviors

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

5.1. Procedure/Behavior on Ingress Node

For a packet to be transported by a P2MP Path, the ingress of the P2MP path duplicates the packet for each sub-tree/branch of the P2MP path branching from the ingress, encapsulates the packet copy in a MRH containing the sub-tree and sends the encapsulated packet copy to the next hop node along the sub-tree.

For example, there is one sub-tree branching from the ingress of the P2MP path/tree in Figure 1. The sub-tree is from ingress PE1 via next hop node P1 towards PE2 to PE9. PE1 sends P1 the packet as shown in Figure 10.

5.2. Procedure/Behavior on Transit Node

When a transit node of a P2MP path/tree receives a packet transported by the P2MP path/tree, the node determines whether the current routing header is a MRH. After determining that it is a MRH, the node executes the procedure to duplicate the packet for each of the downstream links of the node on the P2MP path/tree and send the packet copy to next hop (i.e., downstream node) of the link.

Suppose that the transit node receives the packet from a upstream link from a upstream node to the transit node and there are n downstream links from the transit node on the P2MP path/tree (i.e., there are n branches/sub-trees from the transit node on the P2MP path/tree, where n is greater than zero).

The information about the upstream link is in b and nB field of the MRH. When b = 0, nB field contains the number of branches from the transit node. The information about n downstream links is pointed by SL. When b = 1, the number of branches from the transit node is the number of bits with value 1 in the Bits field of the Bits+ fields pointed by SL. The information about n downstream links is encoded by the Bits field and the fields following the Bits field.

For example, when node P1 receives the packet transported by the P2MP path/tree in Figure 1 from ingress PE1, the MRH is illustrated in Figure 10. The b = 1, the Bits field of the Bits+ fields pointed by SL = 11 has 4 bits with value 1. So there are 4 branches from P1. The information about 4 downstream links (i.e., link from P1 to P2, link from P1 to P3, link from P1 to PE8 and link from P1 to PE9) is encoded by the Bits field and the reduced fields for these 4 links following the Bits field.

For each of the downstream links of the transit node, the transit node duplicates the packet for the link, sets SL, b and nB in the packet copy accordingly. If the next hop is a leaf (i.e., egress), the transit node sets SL to 0; otherwise, the transit node sets SL, b and nB to the value of S-Branches+, B and N-Branches for the link respectively when B is 0. When B = 1, the transit node sets SL and b to the value of S-Branches+ and B for the link respectively. The transit node finds the IPv6 address of the next hop (i.e., the downstream node) of the link from the neighbor IPv6 address table of the transit node using the link number of the link, sets the DA of the packet copy to the IPv6 address of the next hop, and sends the packet copy to the next hop of the link.

For example, for the first downstream link from P1 (i.e., link from P1 to P2), P1 duplicates the packet for the link, sets SL to 6 (which is the value of the S-Branches+ field for the link), b to 0 (which is the value of B for the link) and nB to 2 (which is the value of the N-Branches field for the link). P1 finds the IPv6 address of the next hop P2 of the link from the neighbor IPv6 address table of P1 using the link number 2 of the link, sets the DA of the packet copy to the P2's IPv6 address, and sends the packet copy to P2. The packet copy received by P2 is shown in Figure 11.

For the second downstream link from P1 (i.e., link from P1 to P3), P1 duplicates the packet for the link, sets SL, b and nB to 4, 0 and 1 respectively, which are the values of S-Branches+, B and N-

Branches for the link from P1 to P3 respectively. P1 finds the IPv6 address of the next hop P3 of the link from the neighbor IPv6 address table of P1 using the link number 3 of the link, sets the DA of the packet copy to the P3's IPv6 address, and sends the packet copy to P3. The packet copy received by P3 is illustrated in Figure 12.

For the 3-th downstream link from P1 (i.e., link from P1 to PE8), P1 duplicates the packet for the link, sets SL to 0 since PE8 is a leaf (i.e., egress). P1 finds the IPv6 address of the next hop PE8 of the link from the neighbor IPv6 address table of P1 using the link number 4 of the link, sets the DA of the packet copy to the PE8's IPv6 address, and sends the packet copy to PE8.

5.3. Procedure/Behavior on Egress Node

When an egress node of a P2MP path receives a packet transported by the path, the DA of the packet is the IPv6 address of the egress node and there is an indication in the MRH for the leaf/egress. The egress node proceeds to process the next header in the packet.

For example, after receiving the IPv6 packet from P4, PE4 determines whether the packet's next header is a MRH. When the packet's next header is the MRH, PE4 checks if PE4 itself is a leaf (i.e., egress) through checking whether SL is 0. When PE4 is a leaf, PE4 decapsulates the packet and sends the IP multicast datagram to the IP multicast forwarding module.

6. IANA Considerations

This document requests assigning a new Routing Type in the subregistry "Routing Types" under registry "Internet Protocol Version 6 (IPv6) Parameters" as follows:

Value	Description	Reference
TBD (7 suggested)	Multicast Routing Header	This document

7. Security Considerations

TBD

8. Acknowledgements

The authors would like to thank Donald Eastlake for the valuable comments and suggestions on this draft.

9. References

9.1. Normative References

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

[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>.

[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>.

9.2. Informative References

[I-D.chen-pim-srv6-p2mp-path] Chen, H., McBride, M., Fan, Y., Li, Z., Geng, X., Toy, M., Mishra, G. S., Wang, A., Liu, L., and X. Liu, "Stateless SRv6 Point-to-Multipoint Path", Work in Progress, Internet-Draft, draft-chen-pim-srv6-p2mp-path-09, 22 October 2023, <https://datatracker.ietf.org/doc/html/draft-chen-pim-srv6-p2mp-path-09>.

[I-D.ietf-pim-sr-p2mp-policy] Voyer, D., Filsfils, C., Parekh, R., Bidgoli, H., and Z. J. Zhang, "Segment Routing Point-to-Multipoint Policy", Work in Progress, Internet-Draft, draft-ietf-pim-sr-p2mp-policy-07, 11 October 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-pim-sr-p2mp-policy-07>.

Authors' Addresses

Huaimo Chen
Futurewei
Boston, MA,
United States of America

Email: hchen.ietf@gmail.com

Mike McBride
Futurewei

Email: michael.mcbride@futurewei.com

Yanhe Fan

Casa Systems
United States of America

Email: yfan@casa-systems.com

Zhenbin Li
Huawei

Email: lizhenbin@huawei.com

Xuesong Geng
Huawei

Email: gengxuesong@huawei.com

Mehmet Toy
Verizon
United States of America

Email: mehmet.toy@verizon.com

Gyan S. Mishra
Verizon
13101 Columbia Pike
Silver Spring, MD 20904
United States of America

Phone: 301 502-1347
Email: gyan.s.mishra@verizon.com

Yisong Liu
China Mobile

Email: liuyisong@chinamobile.com

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

Email: wangaj3@chinatelecom.cn

Lei Liu
Fujitsu
United States of America

Email: liulei.kddi@gmail.com

Xufeng Liu
Alef Edge
United States of America

Email: xufeng.liu.ietf@gmail.com