MVPN using P2MP/tree based BIER

draft-xie-bier-mvpn-mpls-p2mp-01

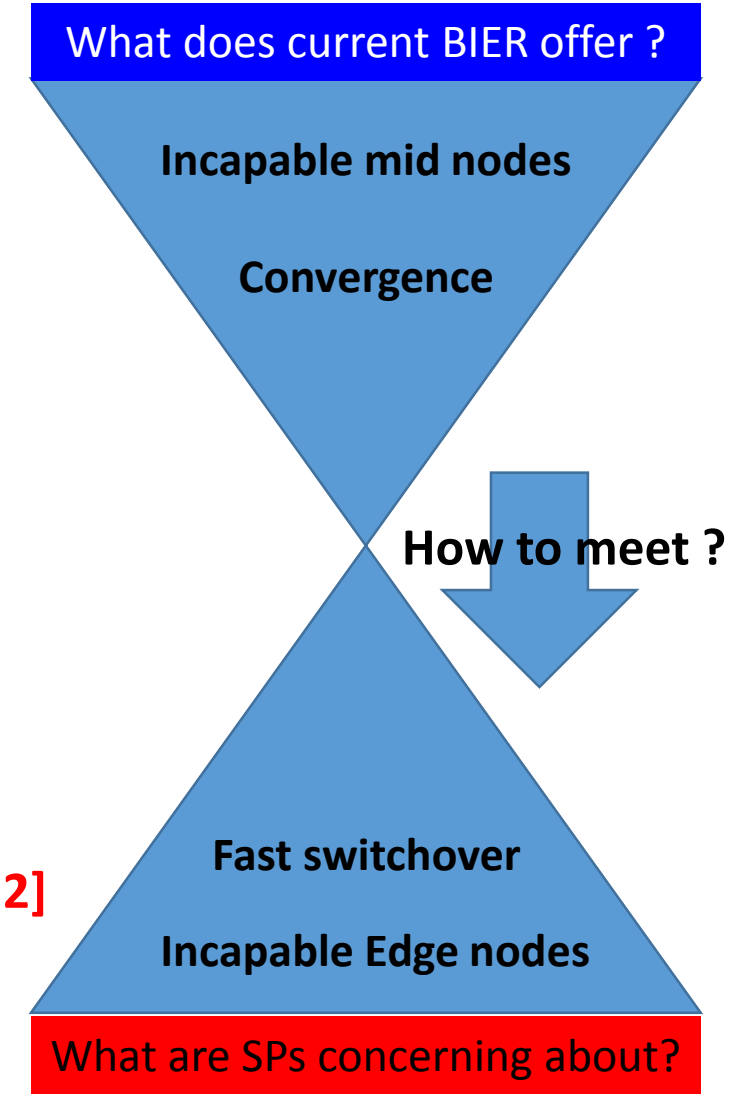
IETF-101 London

Jingrong Xie (Huawei)
Mike McBride (Huawei)
Mach(Guoyi) Chen (Huawei)
Liang Geng (China Mobile)

BIER Transition: Problem Statement

- **BIER offers a radical simplification over current IP multicast :**
  - **BIER packet forwarding/replication is along the unicast paths.**
  - **key operational benefits of BIER: deterministic convergence.**

- **Concerns from SP's perspective:**
  - **Convergence is not enough !**
    - **Fast/Lossless Switchover available ? ----[Problem 1]**
  - **Not only Mid Nodes !**
    - **Possible to Deploy with Incapable Edge nodes ? ----[Problem 2]**

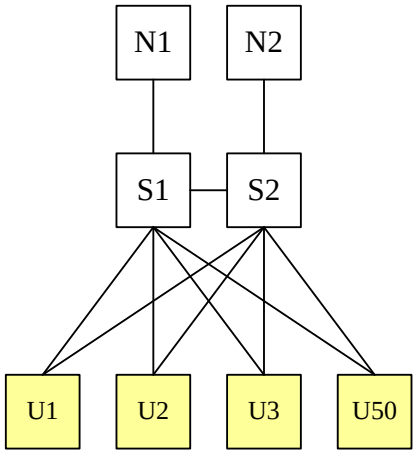


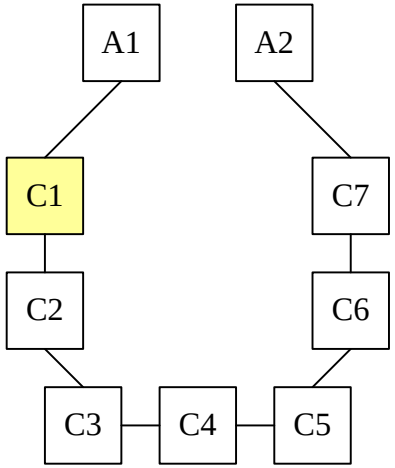
Problem 1

- **Current <draft-ietf-bier-mvpn> delivers a solution of MVPN using SPF based BIER.**
  - **many-to-many topology basis.**
  - **multicast is along the unicast paths.**
  - **It can't, however, support a multicast-specific path well, something common in legacy MVPN deployment:**
    - Live-Live Protection with two dis-joint paths:
      - RSVP-TE with explicit-path (configured on edge nodes).
      - PIM with explicit-rpf-vector (configured on edge nodes).
      - mLDP with static route.
      - MT can provide similar function, but it needs more boring configurations.
  - **Transition from legacy MVPN, without losing the ease of Live-Live dis-joint paths, is lacking. ----> [Problem 1].**

Problem 2

- **Current <RFC8279> provides a solution to support Incapable Mid nodes.**
  - **However, it cannot support deployment on a network with Incapable Edge nodes.**
  - **Unfortunately, it is common in some SP-networks that most of the nodes are Edge nodes.**
    - Example 1: A Hub-Spoken topology in Metro network.
    - Example 2: A Ring topology in backhaul network.





- **Transition from legacy MVPN, in networks with incapable edge nodes, is lacking. ---->[Problem 2]**

The Two Problems: Well-known ?

- **Benoit Claise's Discuss on draft-ietf-bier-architecture-07 (06 Jul 2017)**
  - https://www.ietf.org/mail-archive/web/bier/current/msg01275.html
  - **Operational model which required two simultaneous M/C flows from separate sources. ---->[Problem 1]**
- **BIER Algorithms**
  - https://datatracker.ietf.org/doc/draft-zzhang-bier-algorithm/
  - **Computing Maximum Disjoint Trees ---->[Problem 1]**
  - **Handling BIER Incapable Routers, and Dealing with Ingress Replication Degradation. ---->[Problem 2]**
- **mLDP Extensions for Multi-Topology Routing**
  - https://datatracker.ietf.org/doc/draft-wijnands-mpls-mldp-multi-topology/
  - **Building a Multi-Point LSPs it can follow a particular topology and algorithm. ---->[Problem 1]**

The Two Problems: What possibilities ?

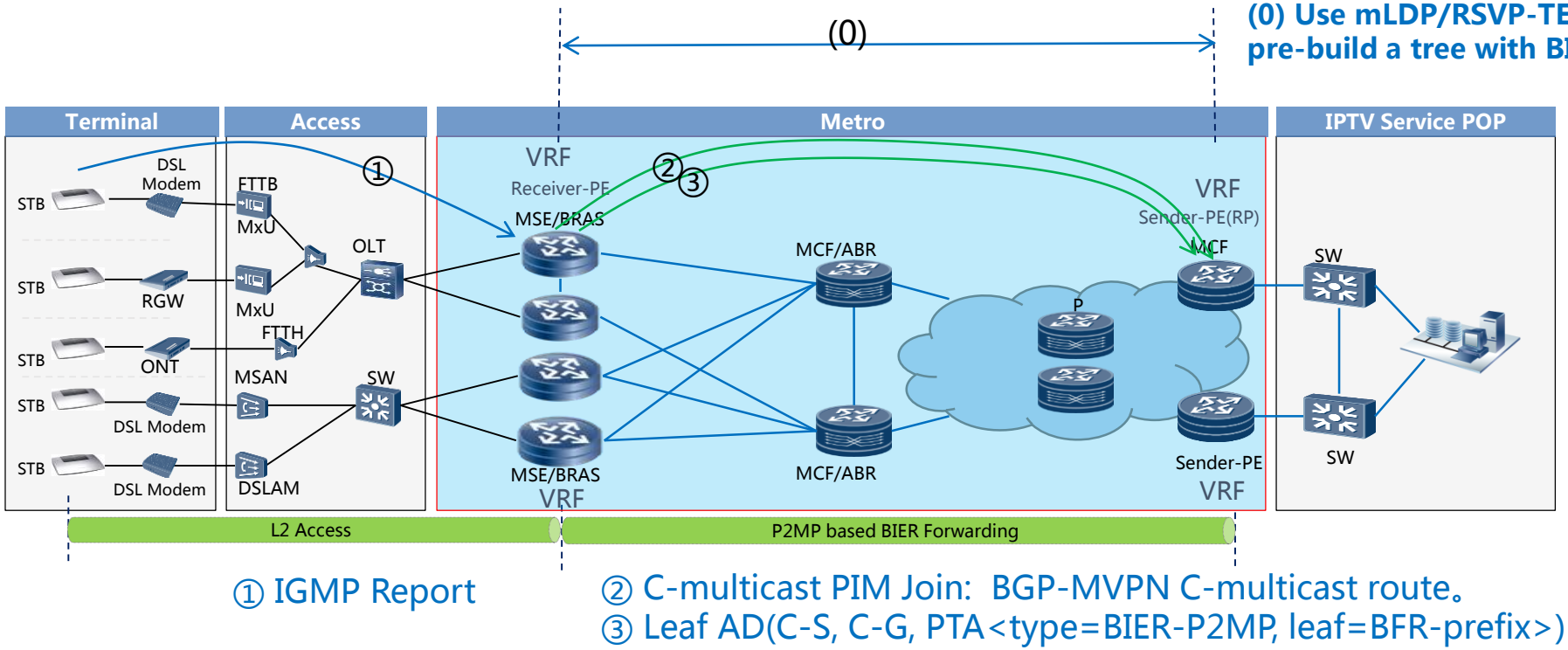
- **Problem 1: Two Disjoint Trees :**
  - **Computing & Algorithming, And then ?**
    - **Just Computing & computing & computing ?**
    - **Just Build it ? ----This draft is determined to Just Build it ! See following pages.**

- **Problem 2: Incapable Edge node:**
  - **RFC8279 has make it clear, Ingress replication do not fit to incapable Edge nodes.**
    - **Why ?**
    - **What's the alternatives ? ----This draft introduces some. See following pages.**

Applicability Statement of this draft

- **This document introduces:**
  - A seamless transition mechanism from legacy ng-MVPN ---->[Problem 1]
    - By applying a BIER encapsulation in data-plane to eliminate per-flow states.
    - While preserving existing features, such as Live-Live dis-joint paths, by using existing protocols.
  - Seamless Live-Live protection developed from Live-Live protection ---->[Problem 1]
    - Considering the ECMP/Entropy feature is not supported in P2MP (see RFC6790)
    - The Entropy field of BIER Header is useless, so re-use it as a per-flow sequence-number.
  - Seamless deployment on networks with Incapable Edge nodes and/or Mid nodes ---->[Problem 2]
    - Exploring of P2MP/tree based BIER forwarding in detail. This is mentioned but not explored in RFC8279.
    - Support Incapable Edge nodes, which is not supported by RFC8279.
    - Support Incapable Mid nodes, without using the P2P replication in RFC8279.

MVPN using P2MP based BIER : The Whole picture



- Main part of Transition Step: to borrow the BIER MPLS encapsulation to eliminate per-flow states.
- Most of the existing remains: MVPN/IPTV service, Live-Live Protection with dis-joint paths. -->**Problem 1**.
- Still deployable when there are some Mid/Edge nodes do not support BIER. -->**Problem 2**.
- Some bit-level stuff in the following pages.....

MVPN using P2MP based BIER (RSVP-TE)

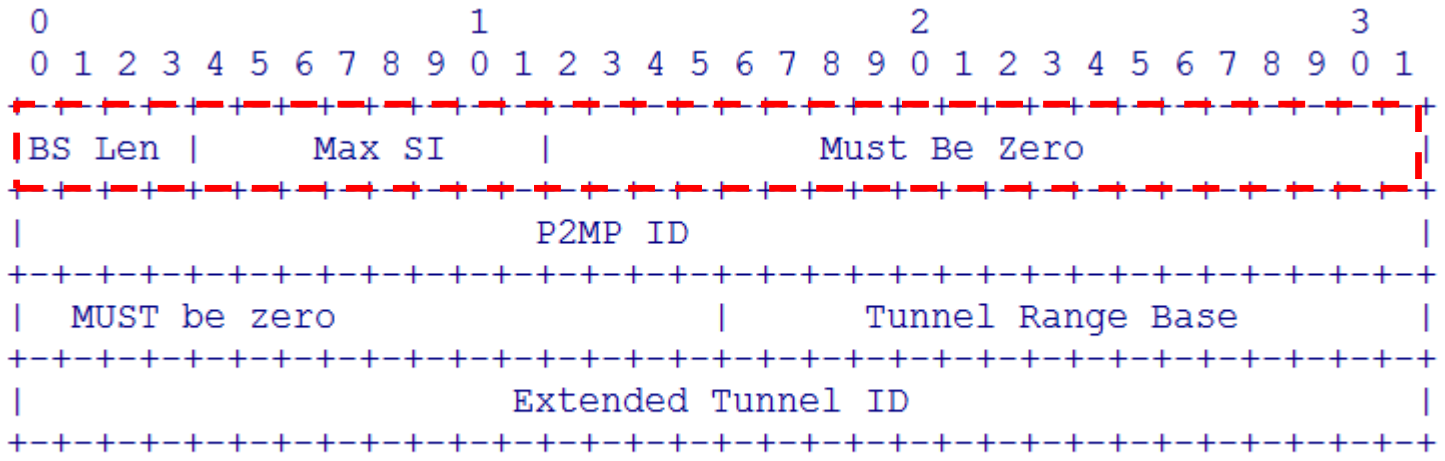


Figure 1: PTA of RSVP-TE built P2MP BIER

- One fixed BSL used. E.g. 256
- Existing feature such as RSVP-TE explicit path can be inherited.
- A batch of ‘RSVP-TE P2MP’ tunnels identified by (Tunnel Number, Tunnel Range Base)
  - R1…R256 join ‘RSVP-TE P2MP’ tunnel identified by <P2MP ID, Tunnel Range Base, Ext Tunnel ID>
  - R257…R512 join ‘RSVP-TE P2MP’ tunnel identified by <P2MP ID, Tunnel Range Base + 1, Ext Tunnel ID>
  - ……

MVPN using P2MP based BIER (mLDP)

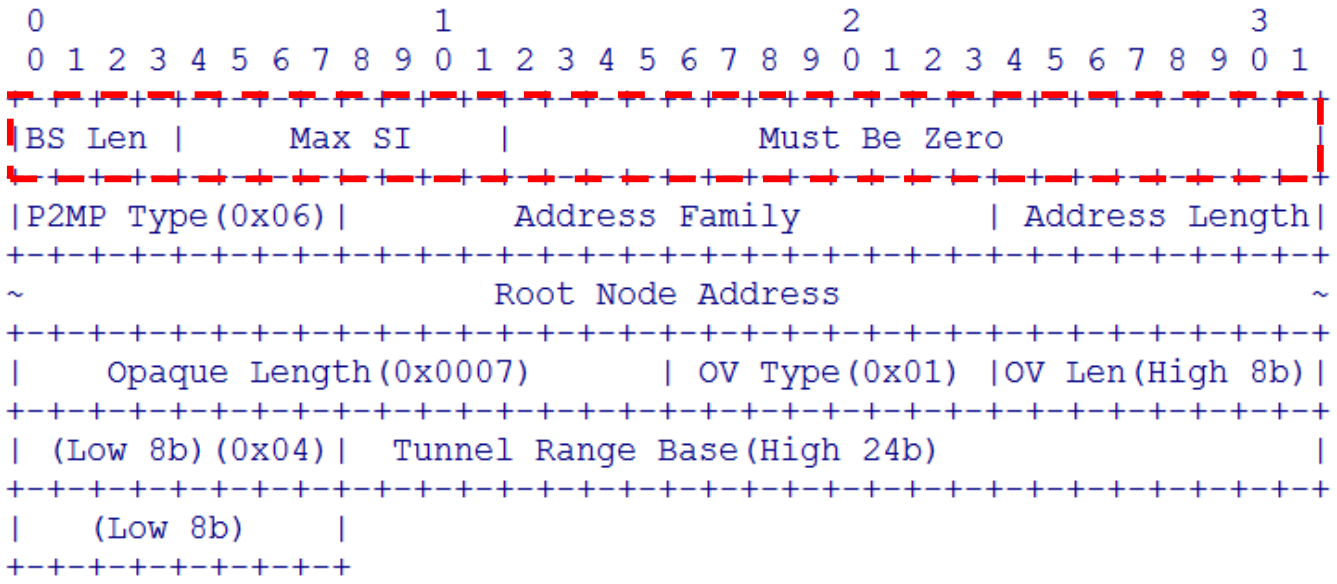


Figure 2: PTA of MLDP built P2MP BIER

- One fixed BSL used. E.g 256
- Existing feature such as mLDP using static route can be inherited.
- A batch of ‘mLDP P2MP’ tunnels identified by (Tunnel Number, Tunnel Range Base)
  - R1…R256 join ‘mLDP P2MP’ tunnel identified by FEC<Root Node Address, Tunnel Range Base>
  - R257…R512 join ‘mLDP P2MP’ tunnel identified by FEC<Root Node Address, Tunnel Range Base + 1>
  - ……

MVPN using P2MP based BIER (PIM)

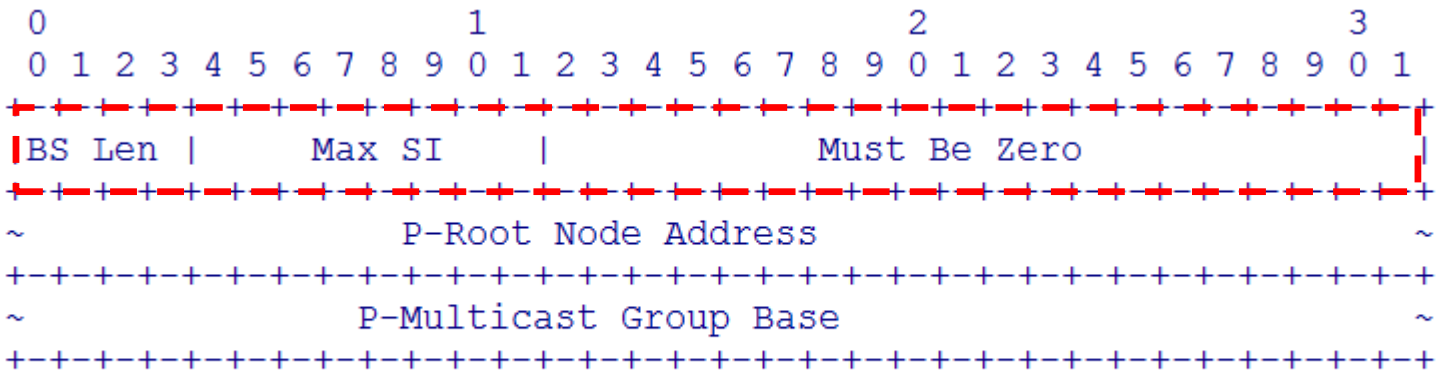
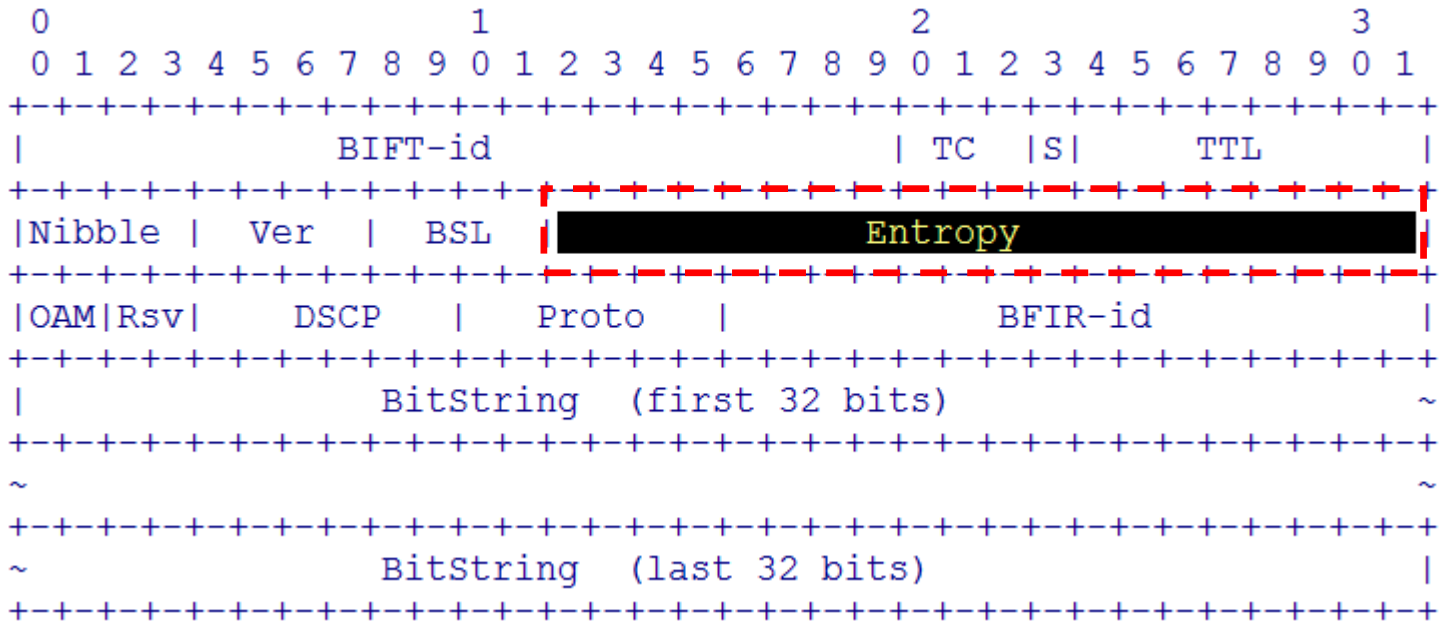
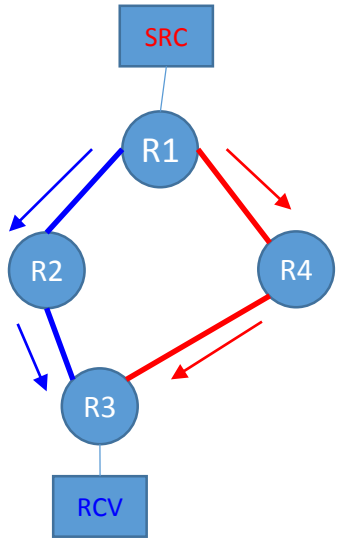


Figure 3: PTA of PIMSSM built P2MP BIER

- One fixed BSL used. E.g 256
- Existing feature such as PIM explicit rpf vector can be inherited.
- A batch of ‘PIM-SSM trees’ identified by (P-Root Node Address, P-Multicast Group Base)
  - R1…R256 join ‘PIM-SSM tree’ identified by (P-Root Node Address, P-Multicast Group Base)
  - R257…R512 join ‘PIM-SSM tree’ identified by (P-Root Node Address, P-Multicast Group Base + 1)
  - ……

Seamless Live-Live protection





- Re-Use Entropy as per-flow sequence-number.
- **Ingress PE (R1):** when forwarding packet from SRC to R2/R4, it imposes a sequence-number in the Entropy subfield, per-flow per-packet.
- **Transit PE (R2/R4):** not need to care about Entropy.
- **Egress PE(R3):** when forwarding packet to local receiver, it brings the sequence-number out, check with the following IP-header(S,G), on a per-flow basis.

MVPN using P2MP based BIER: Underlay protocols

- draft-xie-mpls-ldp-bier-extensions-00
- draft-xie-mpls-rsvp-bier-extensions-00
- draft-xie-pim-bier-extensions-00

- Configuring and Computing on Edge.
- Let one of these protocols to go and run an errand, to Build the tree !

Next Step

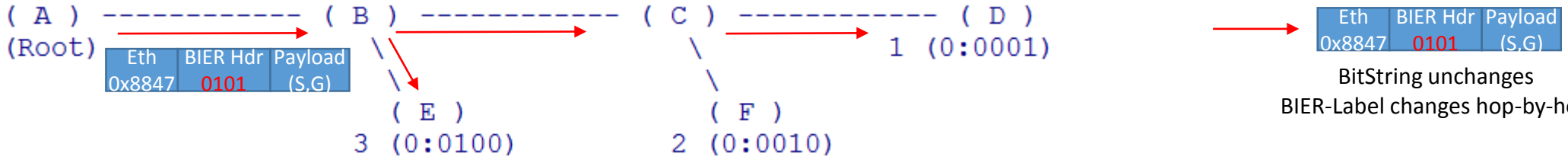
- Make sense ?
  - The Two Problems: Live-Live protection & BIER-incapable Edge nodes.
  - The Combination: of P2MP & BIER.
  - The Philosophy: Need a tree, then build it !
  - The Applicability: seamless transition from ng-MVPN.

- Questions and Comments are welcome.

Thanks !

And some more bit-level stuff in the following pages…

P2MP/tree based BIER forwarding procedure(1)



Forwarding Table on A (FTN and NHLFE)	
FTN	(S,G, TreeID, Flag=CheckBS|Root, BSL)
NHLFE1	(TreeID, OutInterface<toB>, OutLabel<alloc by B>, F-BM<0111>)

Forwarding Table on B (ILM and NHLFE)	
ILM	(inLabel<alloc by B>, action<Rep to TreeID>, Flag=CheckBS|Branch, BSL)
NHLFE1	(TreeID, outInterface<to C>, outLabel<alloc by C>, **F-BM<0011>)**
NHLFE2	(TreeID, outInterface<to E>, outLabel<alloc by E>, **F-BM<0100>)**

Forwarding Table on E (ILM and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=CheckBS|Leaf, BSL)
LEAF	(TreeID, **F-BM<0100>, flag=PopBIERincluding)**

Forwarding Table on C (ILM and NHLFE)	
ILM	(inLabel<alloc by C>, action<Rep to TreeID>, Flag=CheckBS|Branch, BSL)
NHLFE1	(TreeID, outInterface<to D>, outLabel<alloc by D>, **F-BM<0001>**
NHLFE2	(TreeID, outInterface<to F>, outLabel<alloc by F>, **F-BM<0010>**

Forwarding Table on D (ILM and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=CheckBS|Leaf, BSL)
LEAF	(TreeID,**F-BM<0001>, flag=PopBIERincluding)**

Forwarding Table on F (ILM and LEAF)	
ILM	(inLabel<alloc by F>, action<Rep to TreeID>, Flag=CheckBS|Leaf, BSL)
LEAF	(TreeID,**F-BM<0010>, flag=PopBIERincluding)**

- CheckBS means, when Replicate to every NHLFE or LEAF of a Tree, Check the result by AND'ing the BitString in packet and the F-BM in the NHLFE/LEAF, Forward packet only when result is not zero. It is called **P-CAPABILITY**.
- PopBIERincluding(p16-p18)/PopBIERexcluding(p19) means, to pop the BIER header including/excluding the BIER Label in packet. It is called **D-CAPABILITY**.

P2MP/tree based BIER forwarding procedure(2)

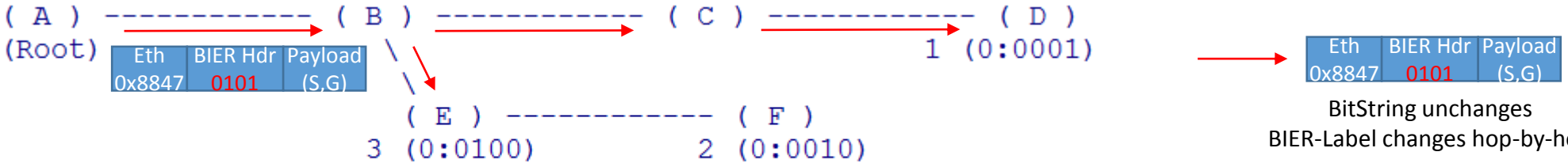


Figure 5: P2MP-based BIER Topology with BUD nodes

Forwarding Table on A (FTN and NHLFE)	
FTN	(S,G, TreeID, Flag=CheckBS|Root, BSL)
NHLFE1	(TreeID, OutInterface<toB>, OutLabel<alloc by B>, F-BM<0111>)

Forwarding Table on B (ILM and NHLFE)	
ILM	(inLabel<alloc by B>, action<Rep to TreeID>, Flag=CheckBS|Branch, BSL)
NHLFE1	(TreeID, outInterface<to C>, outLabel<alloc by C>, **F-BM<0001>)**
NHLFE2	(TreeID, outInterface<to E>, outLabel<alloc by E>, **F-BM<0110>)**

Forwarding Table on E (ILM and NHLFE and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=CheckBS|Bud, BSL)
NHLFE1	(TreeID, outInterface<to F>, outLabel<alloc by F>, **F-BM<0010>)**
LEAF	(TreeID, **F-BM<0100>, flag=PopBIERincluding)**

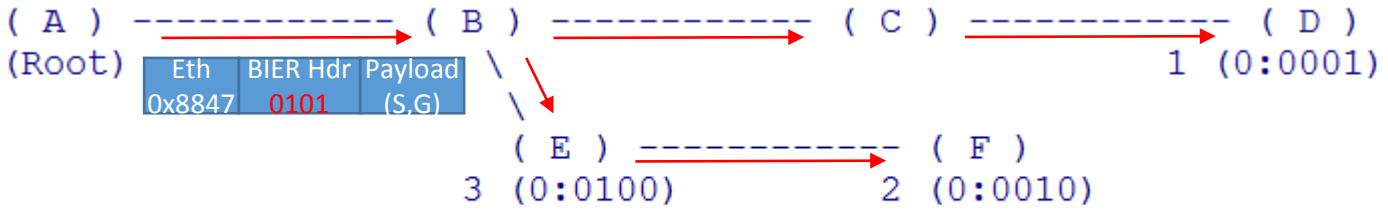
Forwarding Table on C (ILM and NHLFE)	
ILM	(inLabel<alloc by C>, action<Rep to TreeID>, Flag=CheckBS|Branch, BSL)
NHLFE1	(TreeID, outInterface<to D>, outLabel<alloc by D>, **F-BM<0001>**

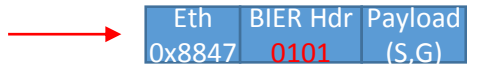
Forwarding Table on D (ILM and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=CheckBS|Leaf, BSL)
LEAF	(TreeID,**F-BM<0001>, flag=PopBIERincluding)**

Forwarding Table on F (ILM and LEAF)	
ILM	(inLabel<alloc by F>, action<Rep to TreeID>, Flag=CheckBS|Leaf, BSL)
LEAF	(TreeID,**F-BM<0010>, flag=PopBIERincluding)**

- A Leaf/BUD node need both P-CAPABILITY and D-CAPABILITY.
- A Branch node need P-CAPABILITY.

When Mid/Leaf/Bud Nodes don’t support P-CAPABILITY





BitString unchanges
BIER-Label changes hop-by-hop

Figure 5: P2MP-based BIER Topology with BUD nodes

Forwarding Table on A (FTN and NHLFE)	
FTN	(S,G, TreeID, Flag=CheckBS|Root, BSL)
NHLFE1	(TreeID, OutInterface<toB>, OutLabel<alloc by B>, F-BM<0111>)

Forwarding Table on B (ILM and NHLFE)	
ILM	(inLabel<alloc by B>, action<Rep to TreeID>, Flag=Branch, BSL)
NHLFE1	(TreeID, outInterface<to C>, outLabel<alloc by C>)
NHLFE2	(TreeID, outInterface<to E>, outLabel<alloc by E>)

Forwarding Table on E (ILM and NHLFE and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=Bud, BSL)
NHLFE1	(TreeID, outInterface<to F>, outLabel<alloc by F>)
LEAF	(TreeID, **flag=PopBIERincluding**)

Forwarding Table on C (ILM and NHLFE)	
ILM	(inLabel<alloc by C>, action<Rep to TreeID>, Flag=Branch)
NHLFE1	(TreeID, outInterface<to D>, outLabel<alloc by D>)

Forwarding Table on D (ILM and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=Leaf, BSL)
LEAF	(TreeID, **flag=PopBIERincluding**)

Forwarding Table on F (ILM and LEAF)	
ILM	(inLabel<alloc by F>, action<Rep to TreeID>, Flag=Leaf, BSL)
LEAF	(TreeID, **flag=PopBIERincluding**)

- When any node (either Branch, Leaf or BUD node) don’t support P-CAPABILITY, just downshift to P2MP/tree forwarding without check the BitString of packet. It is a local behavior.
- Can apply as long as the edge nodes have D-CAPABILITY, which is supposed to be simple for a programmable HW.

When Leaf/Bud Nodes even don't support D-CAPABILITY

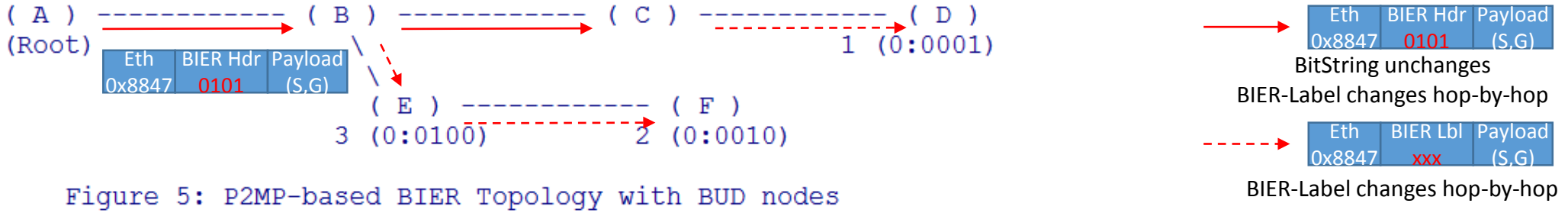


Figure 5: P2MP-based BIER Topology with BUD nodes

Forwarding Table on A (FTN and NHLFE)	
FTN	(S,G, TreeID, Flag=CheckBS|Root, BSL)
NHLFE1	(TreeID, OutInterface<toB>, OutLabel<alloc by B>, F-BM<0111>)

Forwarding Table on B (ILM and NHLFE)	
ILM	(inLabel<alloc by B>, action<Rep to TreeID>, Flag=CheckBS|Branch, BSL)
NHLFE1	(TreeID, outInterface<to C>, outLabel<alloc by C>, **F-BM<0001>)**
NHLFE2	(TreeID, outInterface<toE>, outLabel<byE>, **F-BM<0110>,** Flag=**PopBIERexcluding**)

Forwarding Table on E (ILM and NHLFE and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=Bud)
NHLFE1	(TreeID, outInterface<to F>, outLabel<alloc by F>**)**
LEAF	(TreeID, **flag=PopLabel)**

Forwarding Table on C (ILM and NHLFE)	
ILM	(inLabel<alloc by C>, action<Rep to TreeID>, Flag=CheckBS|Branch, BSL)
NHLFE1	(TreeID, outInterface<toD>, outLabel<byD>, **F-BM<0001>,** Flag=**PopBIERexcluding**)

Forwarding Table on D (ILM and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag=Leaf)
LEAF	(TreeID, **flag=PopLabel)**

Forwarding Table on F (ILM and LEAF)	
ILM	(inLabel<alloc by F>, action<Rep to TreeID>, Flag=Leaf)
LEAF	(TreeID, **flag=PopLabel)**

- Node D don't support D-CAPABILITY, then configure on D to receive a Label packet rather than a BIER packet, and do a <PopBIERexcluding> when C send the replicated packet to D.
- Node E don't support D-CAPABILITY, then configure on E and F to receive a Label packet rather than a BIER packet, and do a <PopBIERexcluding> when B send the replicated packet to E.