

# MVPN using P2MP/tree based BIER

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

IETF-101 London

Jingrong Xie (Huawei)

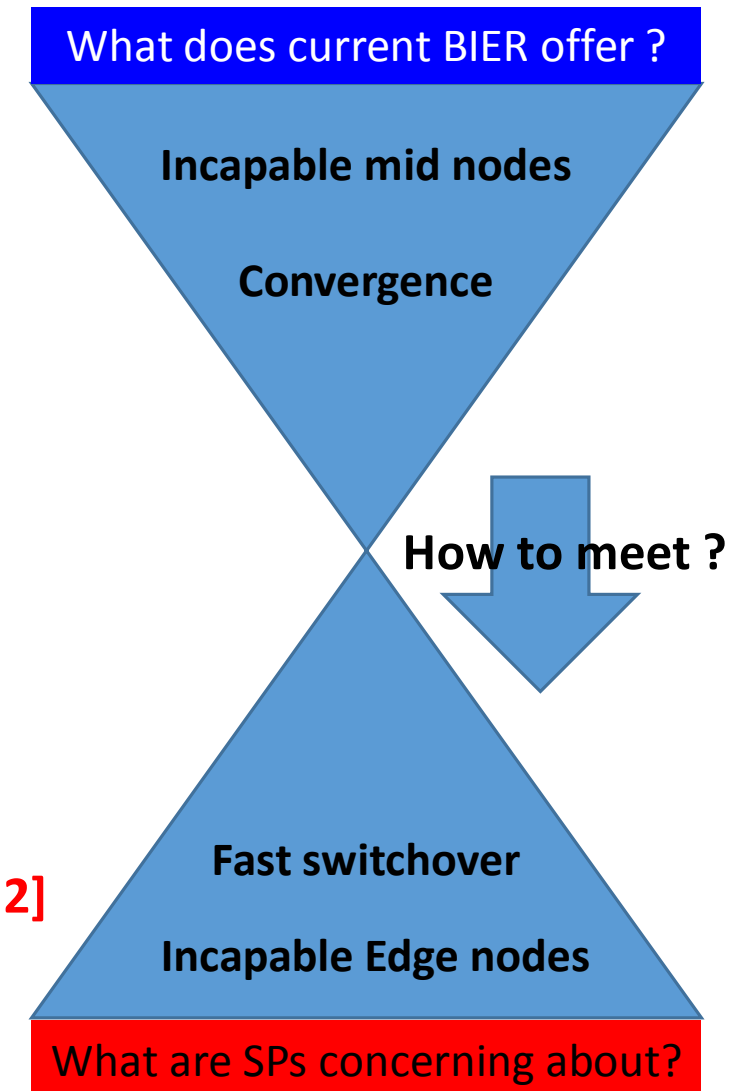
Mike McBride (Huawei)

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]**

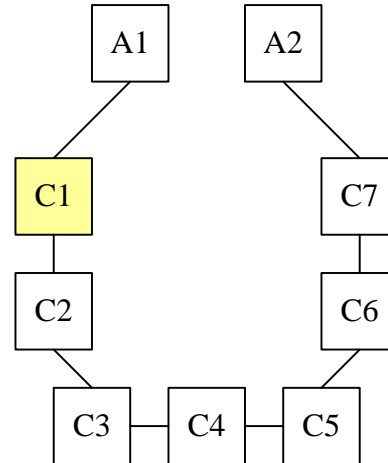
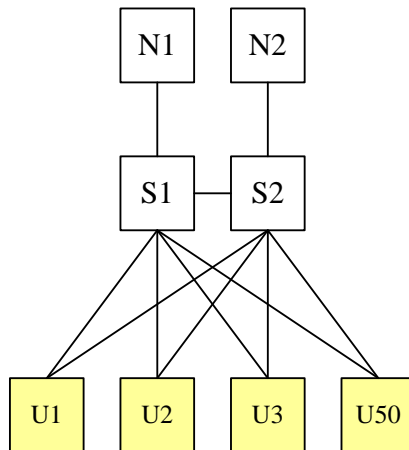


# Problem 1

- Current <draft-ietf-bier-mvpn> delivers a solution of **MVPN using SPF based BIER**.
  - many-to-many topology basis.
  - multicast is along with unicast path.
  - **It can't, however, support a multicast-specific path well, something common in legacy MVPN deployment:**
    - Live-Live Protection with two dis-joining paths:
      - RSVP-TE with explicit-path constrains.
      - PIM with explicit-rpf-vector
      - mLDP with static route.
      - MT provides similar function, but it needs more configuration, and support max 256 only.
  - Transition from legacy MVPN, without losing the **ease of many Live-Live dis-joint paths**, is lacking.

# 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.

# 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, which cause controversy and confusing in BIER-WG:**
  - <https://datatracker.ietf.org/doc/draft-zhang-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 alternatives to solve ?

- **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:**

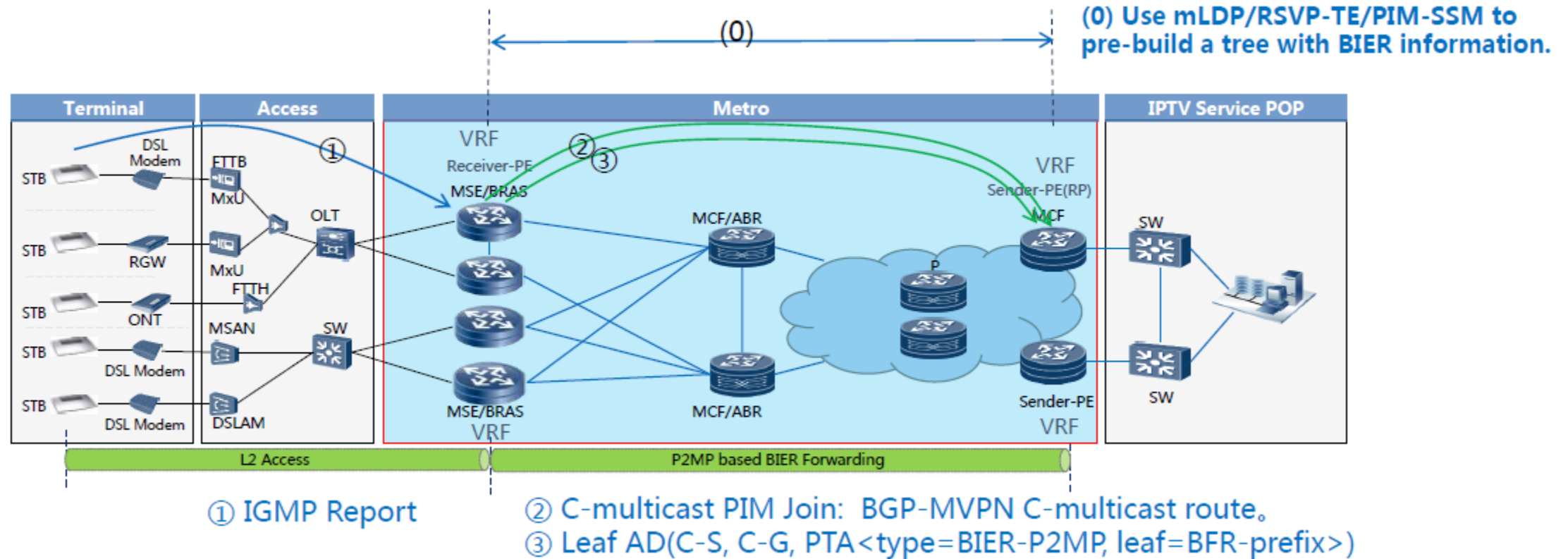
- **RFC8279 has clarified, Ingress replication do not fit to incapable Edge nodes.**

- Why ?
    - What's the alternatives ? ----This draft **introduces some**. See following pages.

# Applicability Statement

- 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 -->[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 support 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 about MVPN PTA in the following pages.....



# MVPN using P2MP based BIER(RSVP-TE)

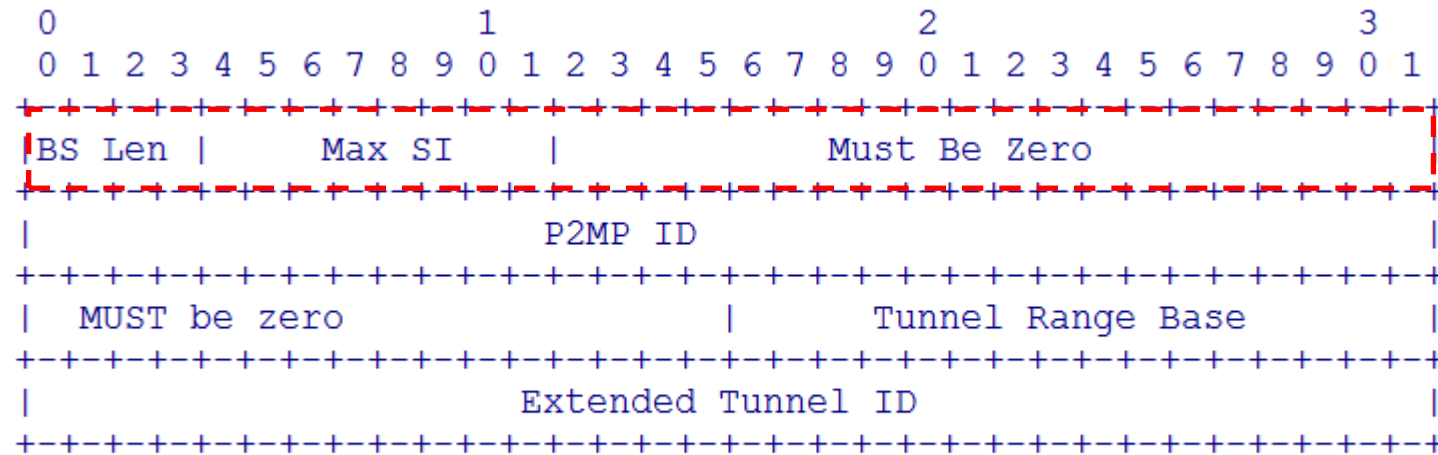


Figure 1: PTA of RSVP-TE built P2MP BIER

- 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)

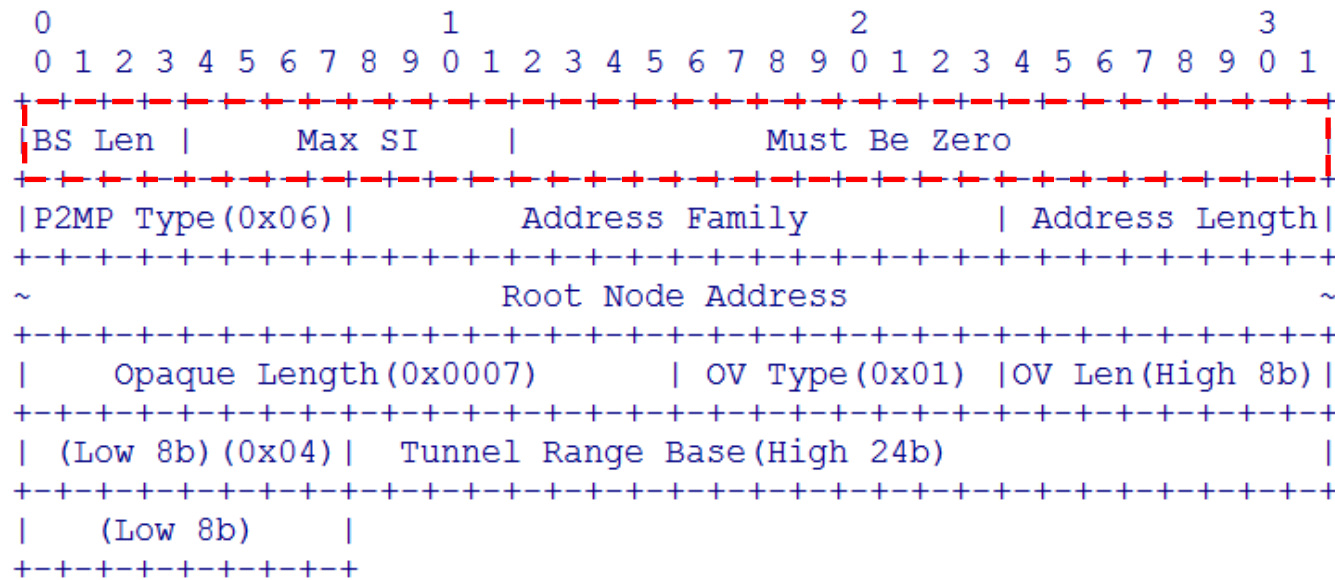


Figure 2: PTA of mLDP built P2MP BIER

- 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)

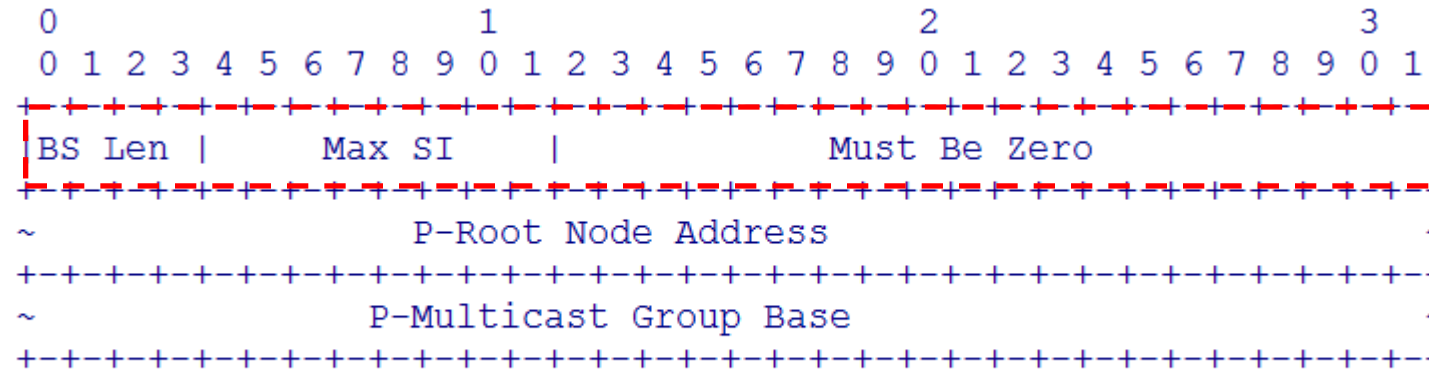
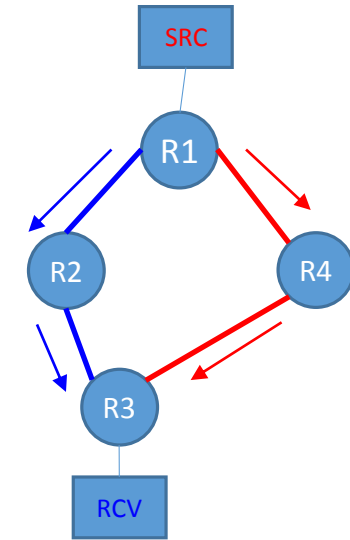
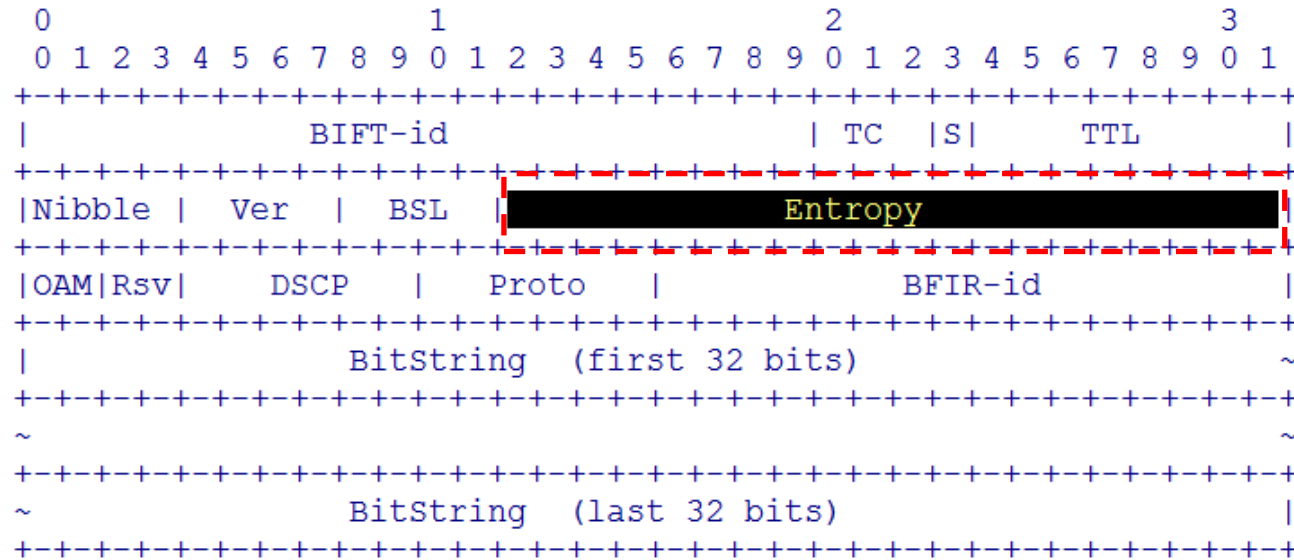


Figure 3: PTA of PIMSSM built P2MP BIER

- 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)
  - R<sub>1</sub>...R<sub>256</sub> join 'PIM-SSM tree' identified by (P-Root Node Address, P-Multicast Group Base)
  - R<sub>257</sub>...R<sub>512</sub> 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):** 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):** brings the sequence-number out, check with the following IP(S,G), on a per-flow basis.
- Which Level is this function belonging to ?
  - RTP has been doing similar thing for decades. And **IEEE 802.3ab is doing similar thing.**
  - Each layer is responsible for its own reliability and so does bier. So it seems to be an Edge specific BIER Layer function, and it seems to be harmless and competitive.

# Underlay protocols to support P2MP based BIER

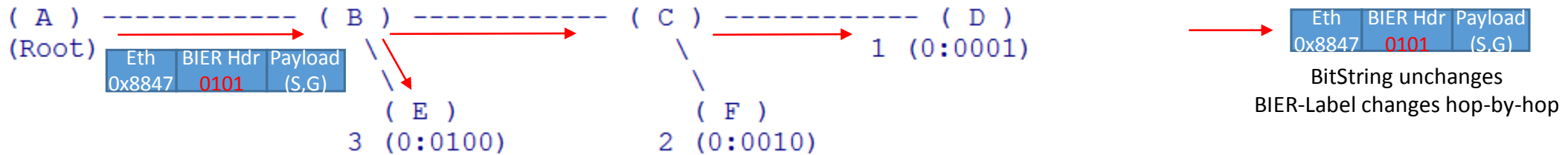
- draft-xie-mpls-ldp-bier-extensions-00
- draft-xie-mpls-rsvp-bier-extensions-00
- draft-xie-pim-bier-extensions-00
- Configuring and Computing on **Edge**.
- Let one of these protocols to go and run an errand, to Build the tree !

# Summary

- The **2 Problems**: Live-Live protection / BIER-incapable Edge nodes.
- Abstract of the **3 seamless** points: Combination of P2MP / BIER.
- The **Philosophy**: Need a tree, then build it !
- The **Applicability**: BIER transition from ng-MVPN.
- The **Question** since IETF100: Simplification or complication?
  - From the **BIER transition perspective**:
  - it makes **the deployment indecisive/complicated** when using current IGP BIER.
    - some valuable things such as Live-Live lost, though some running protocol removed.
    - many Edge nodes are required to upgrade HW, though some running protocol removed.
  - it makes **the deployment determined/simplification** to introduce a tree into BIER.
    - Main BIER benefits got, while existing features preserved, and edge nodes un-upgraded.
    - One more thing, BIER configuration on **Edge** only will be a simplification+, as chap 6 stated.

- Questions and Comments are welcome.
- Thank you !
- And some more bit-level stuff about forwarding procedure in the following pages...

# P2MP/tree based BIER forwarding procedure(1)



Forwarding Table on A (FTN and NHLFE)	
FTN	(S,G, TreeID, Flag= <b>CheckBS</b>   Root, <b>BSL</b> )
NHLFE1	(TreeID, OutInterface<to B>, OutLabel<alloc by B>, <b>F-BM</b> <0111>)
Forwarding Table on B (ILM and NHLFE)	
ILM	(inLabel<alloc by B>, action<Rep to TreeID>, Flag= <b>CheckBS</b>   Branch, <b>BSL</b> )
NHLFE1	(TreeID, outInterface<to C>, outLabel<alloc by C>, <b>F-BM</b> <0011>)
NHLFE2	(TreeID, outInterface<to E>, outLabel<alloc by E>, <b>F-BM</b> <0100>)
Forwarding Table on E (ILM and LEAF)	
ILM	(inLabel<alloc by D>, action<Rep to TreeID>, Flag= <b>CheckBS</b>   Leaf, <b>BSL</b> )
LEAF	(TreeID, <b>F-BM</b> <0100>, flag= <b>PopBIERincluding</b> )

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

- 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)

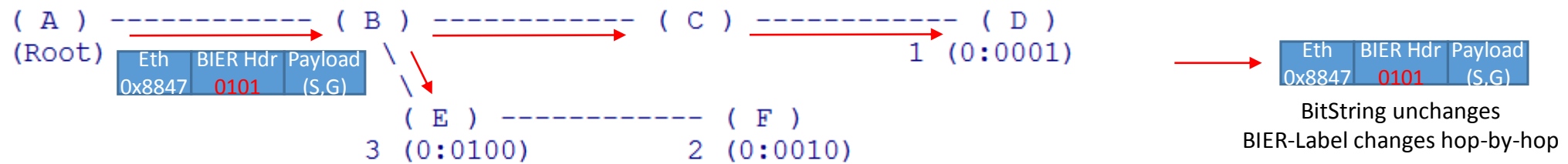


Figure 5: P2MP-based BIER Topology with BUD nodes

Forwarding Table on A (FTN and NHLFE)	
FTN	(S,G, TreeID, Flag=CheckBS   Root, BSL)
NHLFE1	(TreeID, OutInterface<to B>, 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

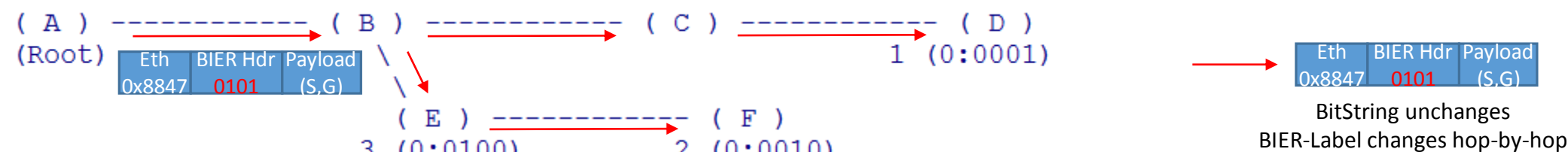


Figure 5: P2MP-based BIER Topology with BUD nodes

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

Forwarding Table on B (ILM and NHLFE)	
ILM	(inLabel<alloc by B>, action<Rep to TreeID>, Flag=Branch, <b>BSL</b> )
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, <b>BSL</b> )
NHLFE1	(TreeID, outInterface<to F>, outLabel<alloc by F>)
LEAF	(TreeID, <b>flag=PopBIERincluding</b> )

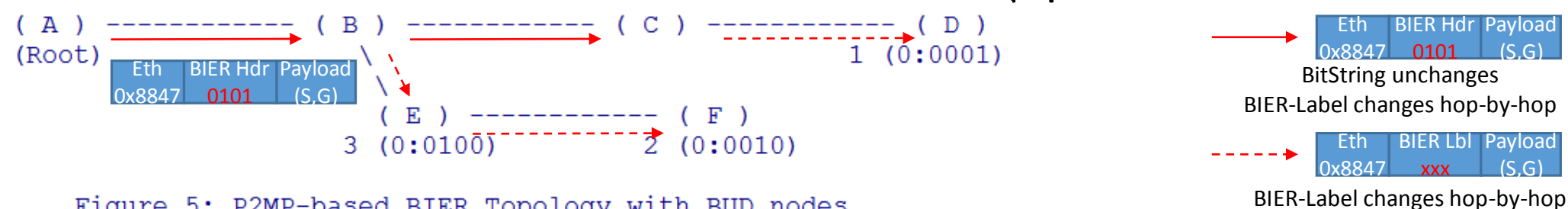
Forwarding Table on C (ILM and NHLFE)	
ILM	(inLabel<alloc by C>, action<Rep to TreeID>, Flag=Branch)
NHLFE1	(TreeID, outInterface<to D>, outLabel<alloc by D>)

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

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

- 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



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.