Network Working Group
Internet-Draft
Intended status: Experimental
Expires: September 6, 2018

T. Eckert
Huawei
G. Cauchie
Bouygues Telecom
W. Braun
M. Menth
University of Tuebingen

March 5, 2018

# Protection Methods for BIER-TE
draft-eckert-bier-te-frr-03

## Abstract

This document proposes protection methods for the BIER-TE architecture [I-D.ietf-bier-te-arch]. These include 1+1 (live-live) path/tree [RFC7431] redundancy, 1:1 path/tree protection, as well as fast reroute (FRR) methods. The latter may protect against link and/or node failures and leverage infrastructure tunnels, BIER-in-BIER encapsulation, or header modification for implementation.

In particular, this memo describes FRR for BIER-TE in detail. FRR for BIER-TE requires support from the BIER-TE controller and the BFRs that are attached to a link/adjacency for which FRR protection is desired. FRR relies on the BIER header described in [RFC8279] which is also used by BIER-TE. It does not require extensions or modifications to existing BIER-TE tables. However, the presented FRR procedures need some additional forwarding plane logic on the BFR. An additional table is needed that carries information about pre-computed backup paths. When a failure is detected, the information from this table is used to modify the bitstring in the BIER header before forwarding a packet over a backup path. To prevent undesired packet duplication, packets should be tunneled on the backup paths.

## Status of This Memo

Copyright Notice

Table of Contents

## 1. Introduction

The BIER-TE architecture is defined in [I-D.ietf-bier-te-arch] and does not provide any protection mechanisms. However, there are several approaches to protect multicast traffic against network failures. This draft gives an overview of 1+1 (live-live) path/tree redundancy, 1:1 path/tree protection, as well as fast reroute (FRR) methods including link and node protection. They may leverage either existing mechanisms with some extensions for BIER-TE, BIER-TE point-to-multipoint tunnels, or renounce on tunneling at all with the downside of reduced coverage.

This document describes additions to the BIER-TE architecture that facilitate FRR for link and node protection using BIER-TE encapsulation. Similar extensions are needed for node-protection FRR with existing mechanisms and must be supported by and must be supported by the BIER-TE controller and BFRs attached to a link/adjacency for which FRR support is required. The FRR operation modifies the BIER header to facilitate local bypass of failed elements. It is possible to add and remove multicast links to the header, use unicast tunnels, e.g., MPLS tunnels, to implement the bypass, or to leverage BIER-in-BIER encapsulation.

The BIER-TE FRR method only requires a small set of additional forwarding entries that are stored in a separate table called the BTAFT. The entries depend only on the topology and not on the multicast flows.

## 2. Overview of FRR Mechanisms for BIER-TE

In the following we give an overview of various protection methods that may be applied to protect BIER-TE-based multicast trees.

BIER-TE can be combined with a range of mechanisms to provide resilience in the face of failures such as link or nodes. In this section, we give an overview of the key options.

## 2.1. Path/Tree Diversity

### 2.1.1. 1+1 Path/Tree Redundancy

In 1+1 redundancy, often also called live-live, traffic is sent twice across the network, path-engineered in a way that no single point of failure (link or node) is in the path for both copies towards every single receiver. For point-to-multipoint transmission, this basically requires disjoint ent-to-end paths. For point-to-multipoint transmission, distribution structures are needed that fulfill the earlier mentioned criterion. They usually resemble

"orthogonal" tree structures. See [BIERFRRANALYSIS] for illustration. For multicast transmission, 1+1 redundancy can be very attractive because the cost of dual transmission can often be negligible vs. the overall bandwidth saving of doing replication in the network.

Path-engineering for 1+1 redundancy can become quite advanced depending on the topology - but it does not require any new functionalities from BIER-TE. It just requires appropriate engineering and potentially additional application or BIER edge functions such as traffic duplication at the sender and traffic deduplication at the receiver.

Consider the following topology example: each sender or receiver has connections to two BFRs to overcome the failure of either BFR. The application on the source creates two copies for every packet. It sends one "a-packet" copy onto its a-link and the other "b-packet" copy onto its b-link. Snd,Rcvr could be BFIR,BFER or they could send native multicast and the BFR they connect to are BFIR, BFER.

BIER-TE bitstrings need to be set up for a-packets and b-packets to pass through the topology counter-circular to each other so that any individual link or node failure never interrupt both a-packets and b-packets

```
                    Snd1,Rcv1
              a-link/      \
                   /        \
         ------ BFR1a ---- BFR1b -------
```

```
                       /                                \
                  - BFR2a          (RING1)            BFR4b -
                 /    \                                /    \
         Snd2,Rcv2     \                              /    Snd4,Rcv4
                  -- BFR2b ---- BFR3a -- BFR3b --- BFR4a ----
                       /              \    /             \
                      /             Snd3,Rcv3             \
                  - BFR5a                                  BFR6b -
                 /    \              (RING2)               /    \
         Snd5,Rcv5     \                                  /   Rcv5
                  -- BFR5b --- BFR6a --- BFR6b --- BFR6a -----
                                    \    /
                                     Rcv6
```

The application side in Snd,Rcv needs to be able to eliminate the duplication resulting from receiving both a-packet and b-packet copies (unless there is a failure). This is easily done if there is any encapsulation (such as transport layer) where sequence numbers are used. Otherwise such a layer has to be added.

If sender and/or receivers can not support the duplication on the sending side and/or deduplication on the receiving side, it is easy to derive variations of these designs in which network ingress/egress devices take over these rules. The functionality that is least common in network devices is per-packet deduplication based on sequence numbers in the packets. Only few network transport service encapsulations do currently provide sequence numbers, e.g., L2TPv3.

Because the described ingress/egress functionalities are not specific to BIER-TE but would equally apply to the solution if built with native IP multicast or RSVP-TE/P2MP, this document does not propose any further functionality required for this type of solutions.

Compared to RSVP-TE/P2MP, one specific benefit of BIER-TE is that trees can be optimized without re-signaling solely by the BIER-TE sender adding/deleting bits representing leaf trees to receivers that join/leave the content.

### 2.1.2. 1:1 Path/Tree Protection

If duplicate transmission is not desirable, variations of the above scheme can be devised where only one copy is sent to each receiver.

Such solutions require receiver feedback to the sender and are therefore most feasible for a limited receiver set deployment, e.g., regional multi-system operator (MSO) distribution networks to hybrid fiber-coaxial (HFC) headends. For example, by default only the a-tree could be transmitted, and upon reception failure at any subset of receivers, the sender would transmit to the subset of the b-tree covering those receivers. Upon recovery of the a-tree, signaling from the receiver could equally stop transmission of the b-tree toward this receiver.

For a more common path utilization across the network in the no-failure scenario, senders could instead start with 50% receiver populated a-tree and b-tree as well.

These options heavily depend on the ability to change the set of receivers in a tree without signaling. The enabling/disabling of sending to a particular branch of the network can be done within a single round-trip starting with the signaling of the reception failure by a receiver.

### 2.2. 1:1 Link/Node Protection

In this section we discuss the link and node protection for BIER-TE using existing mechanisms and the native BIER-Te FRR approach.

#### 2.2.1. Link Protection using Existing Mechanisms

BIER-TE can be used in conjunction with reactive 1:1 link protection as it is also possible in RSVP-TE/P2MP. When a node sees a failure on a link, it assumes that the link has failed and uses a pre-established backup path to get packets to the node on the other side of the link. In example below, BFR2 would use a pre-established path via l7,l2,l3,l4 to send packets to BFR3 when it sees a failure of link1.

```
                                      l3
                        ------ BFR4 ------- BFR5 ------
                       /                    /  \
                      / l2              l4 /    \ l5
                     /                    /      |
                 BFR1 ---- BFR2 --X-- BFR3 ---- BFR6 --
                       l7         l1      \      |
                                           \     | l6
                                            \    |
                                             -- BFR7 --
```

In BIER-TE, this link protection can be done either by using some pre-existing traffic engineered tunneling mechanisms such as RSVP-TE/P2P or Segment Routing paths to get packets to BFR3 in the link failure case. These options do not require enhancements to BIER-TE.

### 2.2.2. Node Protection using Existing Mechanisms

BFR2 can not distinguish whether link1 or BFR3 fails. It simply needs to assume one or the other and perform link or node protection. If it assumes node failure and wants to perform node protection, the solution becomes more complex. Effectively, BFR2 needs to get the packets over to BFR5,BFR6 and BFR7 or the subset of those next-next-hops that is required. Using pre-established p2p backup tunnels (RSVP-TE/P2P or SR) allows to send just to the required subset of next-next-hops at the expense of sending the traffic up to three times across the path consisting of the links l7,l2,l3. The required subset of next-next-hops consists of the downstream next-next-hops. Downstream means that the next-next-hops are the direct multicast children of the next-hop. This is similar to the FRR concept discussed in Section 3.1. Using a backup RSVP-TE/P2MP tree eliminates this higher load but would deliver to all next-next-hops or it would be necessary to set up backup RSVP-TE/P2MP trees for all possible subsets of next-next-hops (which may not scale well).

### 2.2.3. Native 1:1 Protection using BIER-in-BIER Encapsulation

A simpler solution for node protection leverages BIER-in-BIER encapsulation. In this approach, BFR2 would encapsulate the BIER-TE packet into another BIER-TE packet whose bitset was pre-built to carry the packet via l7,l2,l3,l5,l6 to BFR5,BFR6,BFR7 optionally reducing the bitset based on the bitset of the encapsulated BIER-TE packet. This document describes this option in Section 3.

If there are no available infrastructure tunnels deployed, then BIER-in-BIER encapsulation could equally be used for the simple case of link protection.

### 2.2.4. Native BIER-TE Node and Link Protection

An even simpler solution is to not encapsulate the BIER-TE packet into another BIER-TE packet but instead to rewrite the bitstring of the BIER-TE packet to make the packets get delivered via l2,l3 to BFR5, via l5 to BFR6 if that is part of the original bitstring and via l6 to BFR7, if that is part of the original bitstring.

This document specifies the necessary rules for the BIER-TE forwarding machinery for such bitstring bitstring rewrites to enable the most lightweight and efficient form of node protection with BIER-TE.

This solution will not work in all topologies though because the set of bits necessary to pass the traffic across a backup path/tree may cause undesired traffic forwarding (duplicates/loops).

## 3. FRR Extension for Native 1:1 Protection with BIER-TE

In this section, we explain the FRR extension to support native 1:1 protection with BIER-TE. These extensions do not require changes to the BIER header or to forwarding mechanism. The protection procedure runs directly before the BIER-TE forwarding procedure.

Note that the FRR extensions require the use of a new table which is described in Section 3.2. The table is only filled with entries that depend on the network topology and do not depend on the multicast flows.

An implementation of BIER-TE FRR based on BIER-in-BIER encapsulation in the networking programming language P4 is given in [BIER-FRR-P4]. The source code is thoroughly documented and contains examples how the BIER-TE BIFT and BIER-TE BTAFT tables can be filled.

### 3.1. FRR Key Concepts

In this section we use the following example to explain the key concepts of BIER-TE FRR. The example shows a multicast tree from BFR1 to BFR2, BFR6, BFR9. The path to BFR2 is represented by the bits p1, p3 and p4. The bits p1, p6, p7 and the bits p2, p8 represent the path towards BFR6 and BFR 9, respectively. Local_decap bits for all BFR2,BFR6, and BFR9 are also used.

```
                  BFR1-------+
                   |         |
                   |         |
    p4      p3     v p1      v p2
BFR2<---BFR3<-----BFR4------BFR5
         |         |      p5 |
         |         |         |
      p8 |         v p6      v p8
        BFR6<-----BFR7-----BFR9
           p7    p9  p10
```

First, we consider that the link from P towards F fails. The failure can be protected by the backup paths over BFR3->BFR6->BFR7: p3, p8, p9 (BP1) and BFR5->BFR9->BFR7: p5, p8, p10 (BP2). The use of backup path BP1 does not cause duplicates. Backup path BP2 would cause duplicates because the local_decap bit for D2 is still set in bitstring at P. Two options exist to avoid duplicates.

1. We reset the local_decap bit for D2: This solution prevents the duplicate packet. However, this method can lead to lost packets in other examples.

2. We use a tunnel from P to F over D2 to prevent BIER packet processing at the nodes at the backup path.

Tunnels may be implemented in these two different ways:

1. A remote adjacency represented by a single bit which is a tunnel in the routing underlay. For an MPLS routing underlay, this can be implemented using an MPLS label stack. In the example we would introduce an additional bit,e.g., p11, representing the tunnel.

2. BIER-in-BIER encapsulation using an additional BIER header with NextProto = BIER. This methods does not require additional bits for remote adjacencies compared to remote adjacencies but it increases the size of the packet header. In this example the new bitstring contains the bits of BP2 and an additional local_decap bit for BFR7.

Now, we consider that BFR7 fails. The backup path must send the packets to all downstream next next-hops (DS-NNHs), i.e. the next-hops of the sub-tree rooted of BFR7. BFR4 can identify the DS-NNHs by checking the bits of interest of the failed node BFR7. BFR6 is such a node because bit p7 is set. BFR9 is not downstream because there is no bit of interest from BFR7 to BFR9. Sending packets to BFR9 would causes duplicates because BFR9 is served using the branch BFR1->BFR5->BFR9.

Protection against link failures only requires knowledge of the failed adjacency. Protection against node failures requires additional knowledge of the downstream nodes of the tree. The computation of appropriate backup paths, AddBitmasks, ResetBitmasks, and BitPositions is outside of the scope of this document.

### 3.2. The BIER-TE Adjacency FRR Table (BTAFT)

The BIER-TE IF FRR Table exists in every BFR that is supporting BIER-TE FRR. It is indexed by FRR Adjacency Index that is compromised of the SI and the adjacency. Associated with each FRR Adjacency Index are the failed BitPosition (F-BP), Downstream BitPosition (DS-BP), ResetBitmask, and AddBitmask. The table can be configured to enable different actions for the AddBitMask. Either the table is configured to apply BIER-in-BIER encapsulation with a new BIER header containing the AddBitmask as new bitstring or to simply add the bits on the current bitstring.

| FRR Adjacency Index | Failed BP | Downstream BP | ResetBitmask | AddBitmask |
|---|---|---|---|---|
| 0:1 | 5 | 5 | ..0010000 | ..11000000 |

...

An FRR Adjacency is an adjacency that is used in the BIFT of the BFR. The BFR has to be able to determine whether the adjacency is up or down in less than 50msec. An FRR adjacency can be a forward_connected adjacency with fast L2 link state Up/Down state notifications or a forward_connected or forward_routed adjacency with a fast aliveness mechanism such as BFD. Details of those mechanism are outside the scope of this architecture.

The FRR Adjacency Index is the index that would be indicated on the fast Up/Down notifications to the BIER-TE forwarding plane and enables the selection of appropriate ResetBitmasks and AddBitmasks.

The failed BitPosition is the BP in the BIFT in which the FRR Adjacency is used. The downstream BitPosition is required to protect against node failures to identify the downstream adjacency as described in Section 3.1. The backup path/tree is constructed of the individual ResetBitmasks and AddBitmasks of the downstream nodes. To protect against link failures, the DS-BP field is set equally to the F-BP field.

## 3.3. FRR in BIER-TE Forwarding

The BIER-TE forwarding plane receives fast Up/Down notifications of BIER adjacencies which are used for different SIs. From the failed BitPosition in the BTAFT entry, it records which BPs are currently affected (have a down adjacency).

When a packet is received, BIER-TE forwarding checks if it has failed BPs and matching downstream BitPositions to which it would forward. If it does, it will remove the ResetBitmask bits from the packets BitString. Depending on the table configuration it will either add the AddBitmask bits to the packets BitString or construct a new BIER header for rerouted packets. Note that the original packet must be still available for non-affected bitpositions.

Afterwards, normal BIER-TE forwarding occurs, taking the modified bitstring or the additional BIER header into account. Note that the information is pre-computed by the controller so that the BFR can reroute around a failure immediately after its detection.

## 3.4. FRR in the BIER-TE Controller

The basic rules how the BIER-TE controller would calculate the ResetBitmask and the AddBitmask are as follows:

1. The BIER-TE controller decides which tunnel mode a BFR uses for the BTAFT: remote adjacencies or BIER-in-BIER tunneling.

2. The BIER-TE controller determines whether a failure of the adjacency should be taken to indicate link or node failure. This is a policy decision.

3. The ResetBitmask has the BitPosition of the failed adjacency.

4. In the case of link protection, the AddBitmask are the segments forming a path from the BFR over to the BFR on the other end of the failed link. The path can be formed using remote adjacencies for tunneling purposes.

5. In the case of node protection, the AddBitmask are the segments forming a tree from the BFR over to all necessary downstream BFRs of the (assumed to be failed) BFR across the failed adjacency.

6. The ResetBitmask is extended with those segments that could lead to duplicate packets if the AddBitmask is added to possible bitstrings of packets using the failing BitPosition.

### 3.5. BIER-TE FRR Benefits

Compared to other FRR solutions, such as RSVP-TE/P2MP FRR, BIER-TE FRR has two key distinctions

- It maintains the goal of BIER-TE not to establish in-network per multicast traffic flow state. For that reason, the backup path/ trees are only tied to the topology, but not to individual distribution trees.

- For the case of a node failure, it allows to build a path engineered backup tree as opposed to a set of partly overlapping p2p backup tunnels.

- BIER-in-BIER encapsulation enables backup tunnels in networks that do not provide a routing layer with tunneling capabilities. It may simplify network management because additional tunnels (such as GRE) do not to be setup in the routing layer.

### 3.6. Adjustment to the BIER-TE Forwarding Pseudocode

We augment the forwarding procedure presented in the BIER-TE draft to support FRR.

The following procedure computes the ResetBitmasks and AddBitmasks when an adjacency up/down notification is triggered.  The masks can later be directly applied to the header to facilitate the backup.

```
global ResetBitMaskByBT[BitStringLength]
global AddBitMaskByBT[BitStringLength]
global FRRaffectedBP

void FrrUpDown(FrrAdjacencyIndex, UpDown)
{
    global FRRAdjacenciesDown
    local Idx = FrrAdjacencyIndex

    if (UpDown == Up)
        FRRAdjacenciesDown &= ~ 2<<(FrrAdjacencyIndex-1)
    else
        FRRAdjacenciesDown |=   2<<(FrrAdjacencyIndex-1)

    for (Index = GetFirstBitPosition(FRRAdjacenciesDown); Index ;
        Index = GetNextBitPosition(FRRAdjacenciesDown, Index))

        local BP = BTAFT[Index].BitPosition
        FRRaffectedBP |= 2<<(Index)
        ResetBitMaskByBT[BP] |= BTAFT[Index].ResetBitMask
        AddBitMaskByBT[BP]   |= BTAFT[Index].AddBitMask
}
```

The ForwardBierTePacket procedure must be modified by applying the FRR operations when necessary.

```
      void ForwardBierTePacket (Packet)
      {
          // We calculate in BitMask the subset of BPs of the BitString
          // for which we have adjacencies. This is purely an
          // optimization to avoid to replicate for every BP
          // set in BitString only to discover that for most of them,
          // the BIFT has no adjacency.

          local BitMask = Packet->BitString
          Packet->BitString &= ~MyBitsOfInterest
          BitMask &= MyBitsOfInterest

          // FRR Operations
          // Note: this algorithm is not optimal yet for ECMP cases
          // it performs FRR replacement for all candidate ECMP paths

          local MyFRRBP = BitMask & FRRaffectedBP
          for (BP = GetFirstBitPosition(MyFRRNP); BP ;
               BP = GetNextBitPosition(MyFRRNP, BP))
              BitMask &= ~ResetBitMaskByBT[BP]
              BitMask |=  ResetBitMaskByBT[BP]

          // Replication
          for (Index = GetFirstBitPosition(BitMask); Index ;
               Index = GetNextBitPosition(BitMask, Index))
              foreach adjacency BIFT[Index]

                  if(adjacency == ECMP(ListOfAdjacencies, seed) )
                      I = ECMP_hash(sizeof(ListOfAdjacencies),
                                    Packet->Entropy, seed)
                      adjacency = ListOfAdjacencies[I]

                  PacketCopy = Copy(Packet)

                  switch(adjacency)
                      case forward_connected(interface,neighbor,DNR):
                          if(DNR)
                              PacketCopy->BitString |= 2<<(Index-1)
                          SendToL2Unicast(PacketCopy,interface,neighbor)

                      case forward_routed([VRF],neighbor):
                          SendToL3(PacketCopy,[VRF,]l3-neighbor)

                      case local_decap([VRF],neighbor):
                          DecapBierHeader(PacketCopy)
                          PassTo(PacketCopy,[VRF,]Packet->NextProto)
      }
```

### 3.7. Recommendations for Tunneling

Recommendations for usage of tunnels for BIER-TE FRR based on the study in [BIERFRRANALYSIS]:

1. Tunneling SHOULD be used. The study shows that header modification does not improve coverage in many topologies and may cause more harm than protection when node failures occur.

2. Using unicast tunnels may cause unnecessary extra load on overlapping links when protecting against node failures.

Moreover, they require additional bits in the BIER header.

3. BIER-in-BIER encapsulation is the recommended mechanism. It causes increased header overhead through the encapsulating BIER header. This may become an issue when the BIER header supports long bitstrings

## 4. IANA Considerations

This document requests no action by IANA.

## 5. Acknowledgements

The authors would like to thank Greg Shepherd, Ijsbrand Wijnands and Neale Ranns for their extensive review and suggestions.

## 6. Change log [RFC Editor: Please remove]

03: Updated references, no other conent changes from -02. Refresh to continue using this document for reference of new work. Unchanged. Currently unclear if the novel method proposed in this document is worth pursuing. If not, then we would change the document to solely document the applicability of pre-existing methods (and remove new options).

02: Overview of FRR mechanisms feasible for BIER-TE (Eckert)

02: Removed Section "BIER-TE and existing FRR, because it is now part of "Overview" chapter (Braun)

02: Added recommendation on native BIER-TE FRR with regard to tunneling options based on a study (Braun)

02: Added P4 implementation of BIER-TE FRR using BIER-in-BIER as reference.(Braun)

01: Update of author addresses

00: Initial version based on draft-eckert-bier-arch-03.

## 7. References

## 7.1. Normative References

[I-D.ietf-bier-te-arch]
Eckert, T., Cauchie, G., Braun, W., and M. Menth, "Traffic Engineering for Bit Index Explicit Replication (BIER-TE)", draft-ietf-bier-te-arch-00 (work in progress), January 2018.

[RFC7431] Karan, A., Filsfils, C., Wijnands, IJ., Ed., and B. Decraene, "Multicast-Only Fast Reroute", RFC 7431, DOI 10.17487/RFC7431, August 2015, <https://www.rfc-editor.org/info/rfc7431>.

[RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", RFC 8279, DOI 10.17487/RFC8279, November 2017, <https://www.rfc-editor.org/info/rfc8279>.

## 7.2. Informational References

[BIER-FRR-P4]
Braun, W., Hartmann, J., and M. Menth, "Demo: Scalable and Reliable Software-Defined Multicast with BIER and P4", IFIP/IEEE International Symposium on Integrated Network Management (IM), May 2017, <https://atlas.informatik.uni-tuebingen.de/~menth/papers/Menth17b.pdf>.

[BIERFRRANALYSIS]
Braun, W., Eckert, T., and M. Menth, "Performance Comparison of Resilience Mechanisms for Stateless Multicast Using BIER", IFIP/IEEE International Symposium on Integrated Network Management (IM), May 2017, <https://atlas.informatik.uni-tuebingen.de/~menth/papers/Menth17a.pdf>.

Authors' Addresses

Toerless Eckert
Huawei USA - Futurewei Technologies Inc.
2330 Central Expy
Santa Clara  95050
USA

Email: tte+ietf@cs.fau.de

Gregory Cauchie
Bouygues Telecom

Email: GCAUCHIE@bouyguestelecom.fr

Wolfgang Braun
University of Tuebingen

Email: wolfgang.braun@uni-tuebingen.de

Michael Menth
University of Tuebingen

Email: menth@uni-tuebingen.de