Fast ReRoute (FRR) Extensions for BIER-TE

draft-eckert-bier-te-frr

Toerless Eckert, Gregory Cauchie, Wolfgang Braun, Michael Menth

http://kn.inf.uni-tuebingen.de
BIER-TE Drafts Since Buenos Aires

- Requested in IETF 95
  - Remove FRR from BIER-TE draft

- Update from bier-te-arch-03 to bier-te-arch-04
  - Removed FRR method and put into a new draft-eckert-bier-te-frr-00
Failures are detected by BFR component and local detour for packets will be initiated.

FRR modifies bitstring to implement the detour:
- Backup path is encoded by adding bits
- Some bits must be removed to prevent duplicates or even loops
- The Add- and ResetBitmasks are dependent of the failed element but are applied to all multicast flows using the failed segment
- Highly scalable

Backup/detour paths have to be computed depending on the failure policy (link or node)
Packet must be redirected at B

Multicast tree 1 delivers packets to C and D because A sends a packet directly to D
  - Local_decap(D) must be reset to avoid duplicate

Multicast tree 2 with reset local_decap(D) does not deliver at D

Tunneling can avoid reset bits at the backup path
Node Failure Protection

- Improved creation of backup paths using tunnels
- Send packet only to **DS-NNH**
  - DS-NNH is identified by the DS-NNH-BOI: F → D1
  - Reset contains failed interface and incoming adjacencies of DS-NNHs
  - Add contains only the path to the NNH D1 but not D2 because F → D2 is not set
- **No duplicates or losses!**
- Additional entry for NNH adjacency necessary in BTAFT!
Backup Path Options (1)

- Simply add the backup path and reset necessary bits
  - Simple to implement but lots of adjacencies have to be reset to avoid duplicates
  - Does not provide full (or rather low) coverage

![Diagram showing backup path and reset bits]
Use forward_routed() adjacencies to tunnel the packet to avoid intermediate BFRs to look at the bit string and cause duplicates

- Requires additional bits in the bitstring to encode tunnels; bit space is already limited
- Routing underlay must provide the tunnels; state requirements?
- P2MP tunnels? Unicast tunnels can cause high loads for node failures

---

**Backup Path Options (2)**

- **001000110111101**
- **011000010111101**

Reset bits
Backup path
Dedicated forward_routed
Use BIER-in-BIER tunneling to form the tunnel

- Additional BIER header may cause large overhead because bitstring varies in size (64 – 4096 bits)
- Supports P2MP detour paths for node failures

```
1000110111101
```

```
0111000000000
```

```
1000010111101
```
Currently we study BIER-TE FRR

- Different detour implementation options
- Failure policies (link or node)
- Failure scenarios (single link and node failures)

Compare BIER-TE with 1+1 (live-live) protected BIER (MoFRR/MRTs)

Preliminary results

- Full coverage for BIER-TE cannot be achieved without tunnels; coverage is rather low or does more harm than good
- Hop lengths are mostly lower for BIER-TE than for MoFRR
- Unicast detour paths cause high link loads when node protection is active
- BIER-TE generally requires less capacity than 1+1