Hardware-Based Evaluation of Scalable and Resilient Multicast with BIER in P4

Daniel Merling, Steffen Lindner, Michael Menth
daniel.merling@uni-tuebingen.de

http://kn.inf.uni-tuebingen.de
Outline

- Traditional IP multicast
- BIER
- Implementation of BIER in P4
- BIER Fast Reroute
IP Multicast (IPMC)

IP multicast (IPMC) efficiently distributes one-to-many traffic

⇒ Traditional IPMC core network requires

1. State per multicast group to know next-hops (NHs) of a packet
2. Signaling in core network when Group subscriptions change

⇒ Scalability of traditional IPMC is limited
Bit Index Explicit Replication (BIER)

- Efficient transport mechanism for IPMC traffic
  - Domain concept
  - Core routers do not require state per IPMC group
Bit Index Explicit Replication (BIER)

- Efficient transport mechanism for IPMC traffic
  - Domain concept
  - Core routers do not require state per IPMC group
Bit Index Explicit Replication (BIER)

Efficient transport mechanism for IPMC traffic
- Domain concept
- Core routers do not require state per IPMC group
High-level programming language to describe data plane
- Compiler maps P4 program onto programmable pipeline of target

Programmable

Packet in
Parser
Ingress pipeline
Match-action
Deparser
Packet recirculation

Packet out
Parser
Egress pipeline
Match-action
Deparser
Paket is sent to switch-intern recirculation port

- Port can be overloaded if too many packets are recirculated
  
  Additional physical ports in loopback mode
Create packet clones and forward them to all relevant next-hops

One pipeline iteration:

- Original BIER packet
- Cloned BIER packet
Create packet clones and forward them to all relevant next-hops

One pipeline iteration:

Original BIER packet

Cloned BIER packet

Entire packet processing:
Throughput Measurements: Setup

- Measure end-to-end throughput
  - Change amount of recirculation traffic
    - Number of next-hops
  - Measure throughput always at last next-hop!
Throughput Measurements: Setup

- Measure end-to-end throughput
  - Change amount of recirculation traffic
    - Number of next-hops
  - Measure throughput always at last next-hop!

- Tofino
  - P4 programmable high-performance switch ASIC
  - In Edgecore Wedge 100BF-32X

- 100 Gb/s traffic generator
Throughput Measurements: Results

Sending rate = 100 Gb/s

BIER Throughput (Gb/s)

Number of recirculation ports

Number of next-hops: 1 2 3 4
BIER Fast Reroute (BIER-FRR)

- BIER patent mentions fast reroute (FRR) for BIER based on loop-free alternates
  - LFAs cannot guarantee full coverage for single link failures [1] [2]
  - LFAs may cause microloops [1] [2]
  - Sometimes multiple BIER packets are sent over the same link [3]
BIER Fast Reroute (BIER-FRR)

- BIER patent mentions fast reroute (FRR) for BIER based on loop-free alternates
  - LFAs cannot guarantee full coverage for single link failures [1] [2]
  - LFAs may cause microloops [1] [2]
  - Sometimes multiple BIER packets are sent over the same link [3]

- Tunnel-based BIER-FRR [4]
  - Connectivity is restored faster in routing underlay
    - Unicast FRR
    - Faster recomputation

- Comparison in [3]
Time measurement until bmv2 1 receives traffic again after failure of primary path

Controller is directly connected to Tofino
Restoration Time: Results

- w/o BIER-FRR and w/o IP-FRR: 166.15 ms
- w/ BIER-FRR but w/o IP-FRR: 95.84 ms
- w/ BIER-FRR and w/ IP-FRR: 0.59 ms
Conclusion

► Hardware-based P4 implementation of BIER for 100G hardware
  ▪ Depending on amount of recirculation traffic, physical ports in loopback mode may be necessary to prevent packet loss
  ▪ In realistic scenarios only very few physical recirculation ports are sufficient

► Tunnel-based BIER-FRR significantly reduces restoration time in case of failures

