Fragment Forwarding vs Per hop reassembly

Performance report

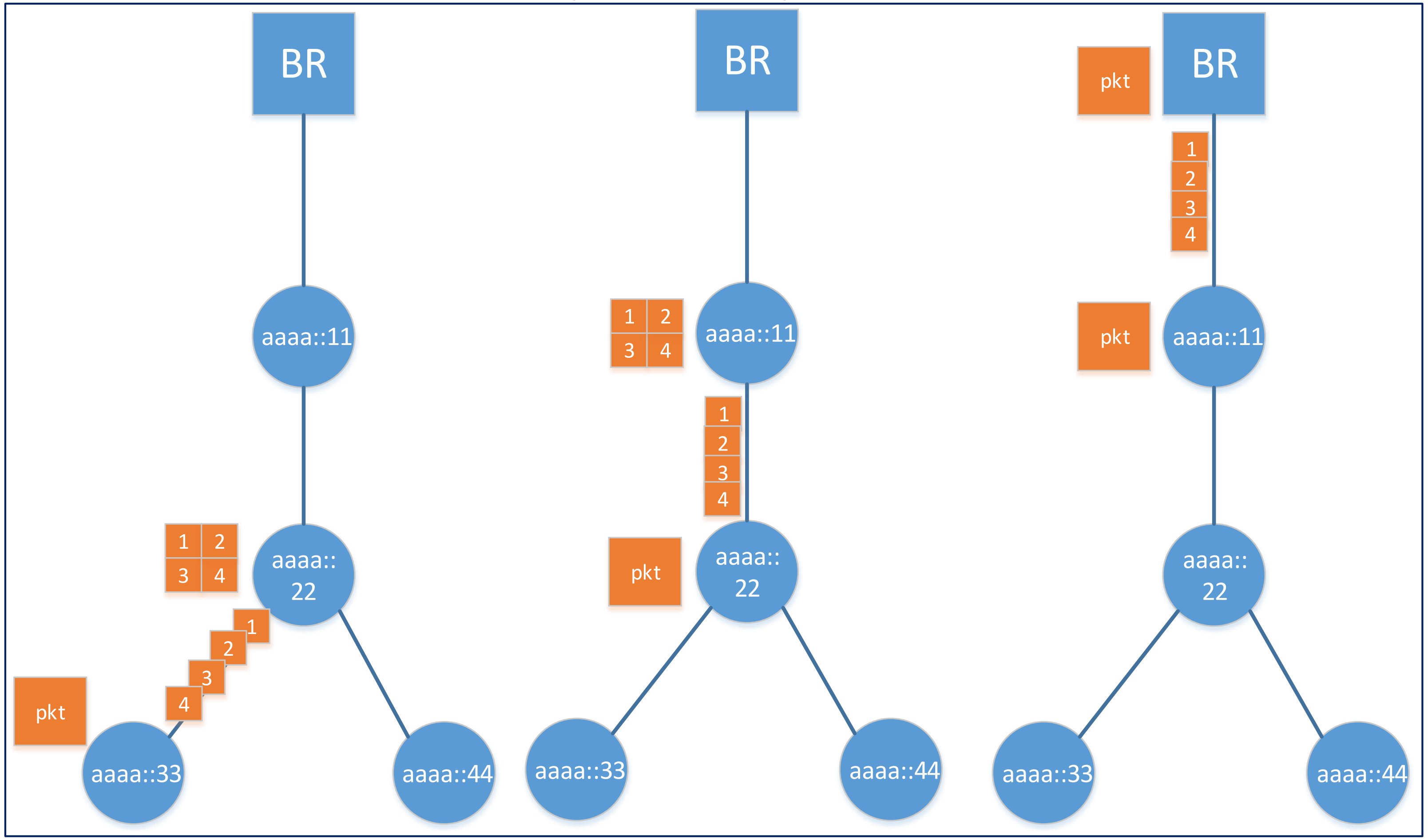
https://github.com/nyrahul/ietf-data/blob/master/6lo-fragfwd-perf-report.rst

- Rahul Jadhav & Rabi Sahoo

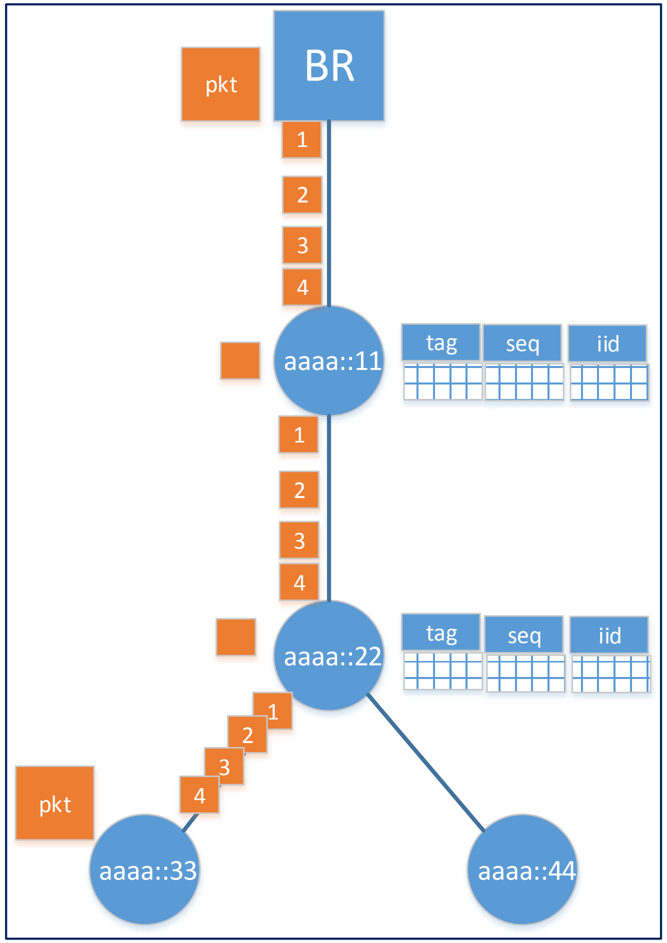
IETF 103, Bangkok

Briefly about fragment forwarding

Per hop reassembly – RFC 4944



Fragment forwarding

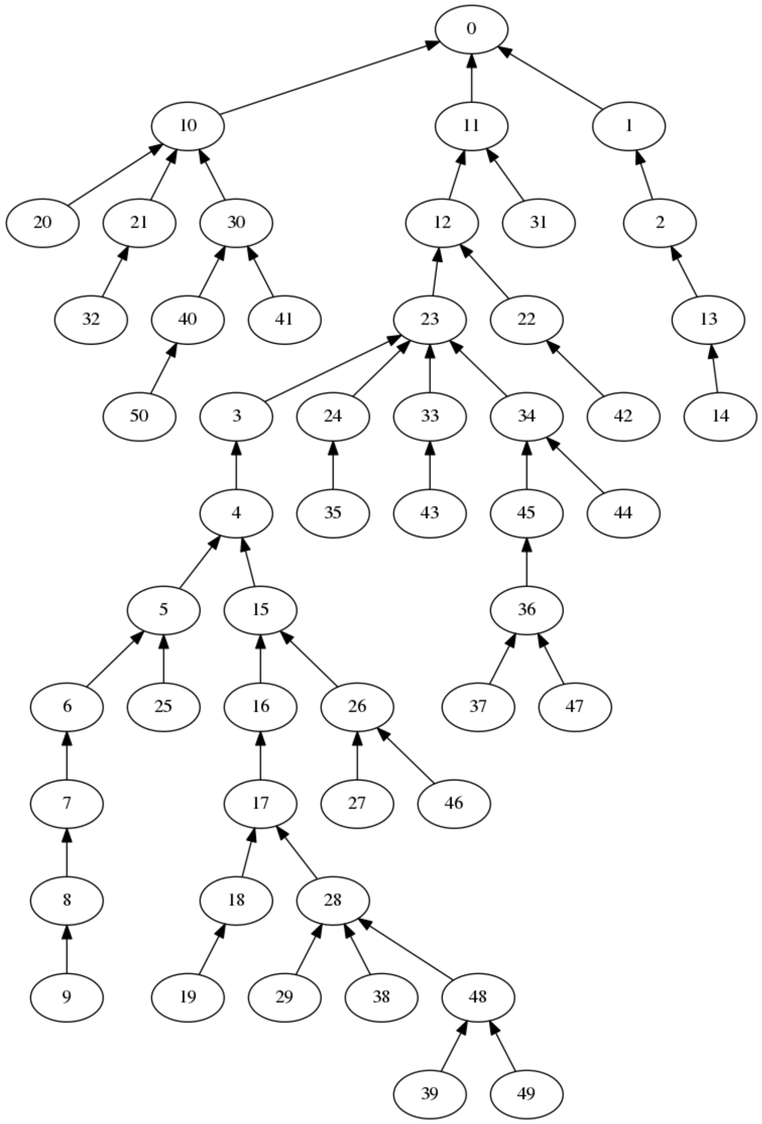


Our motivation

- Understand
  - Latency/PDR implications of using fragment forwarding(FF)
  - Focus not much on memory utilization
    - Fragment forwarding clearly improves memory utilization
- Motivation
  - Use of EAP-PANA (as defined by Wi-SUN) causes fragmentation during authentication
    - Can FF help improve PDR/latency such that network convergence time is reduced?

Test configuration

- L2 configuration
  - 802.15.4 in unslotted single channel 2.4GHz mode
  - Carrier sensing enabled but no RTS/CTS
    - LoWPAN does not use RTS/CTS because of high overhead
  - L2 MTU = 127 Bytes
  - Max mac retry = 3 (with exp backoff)
- Network Configuration
  - # of nodes = 50
  - Grid (10x5) Topology
    - Inter-node distance (x,y) = (80m, 100m)
- RPL Routing
  - MRHOF with ETX as routing metric
  - Trickle parameters, MRHOF thresholds same for all tests

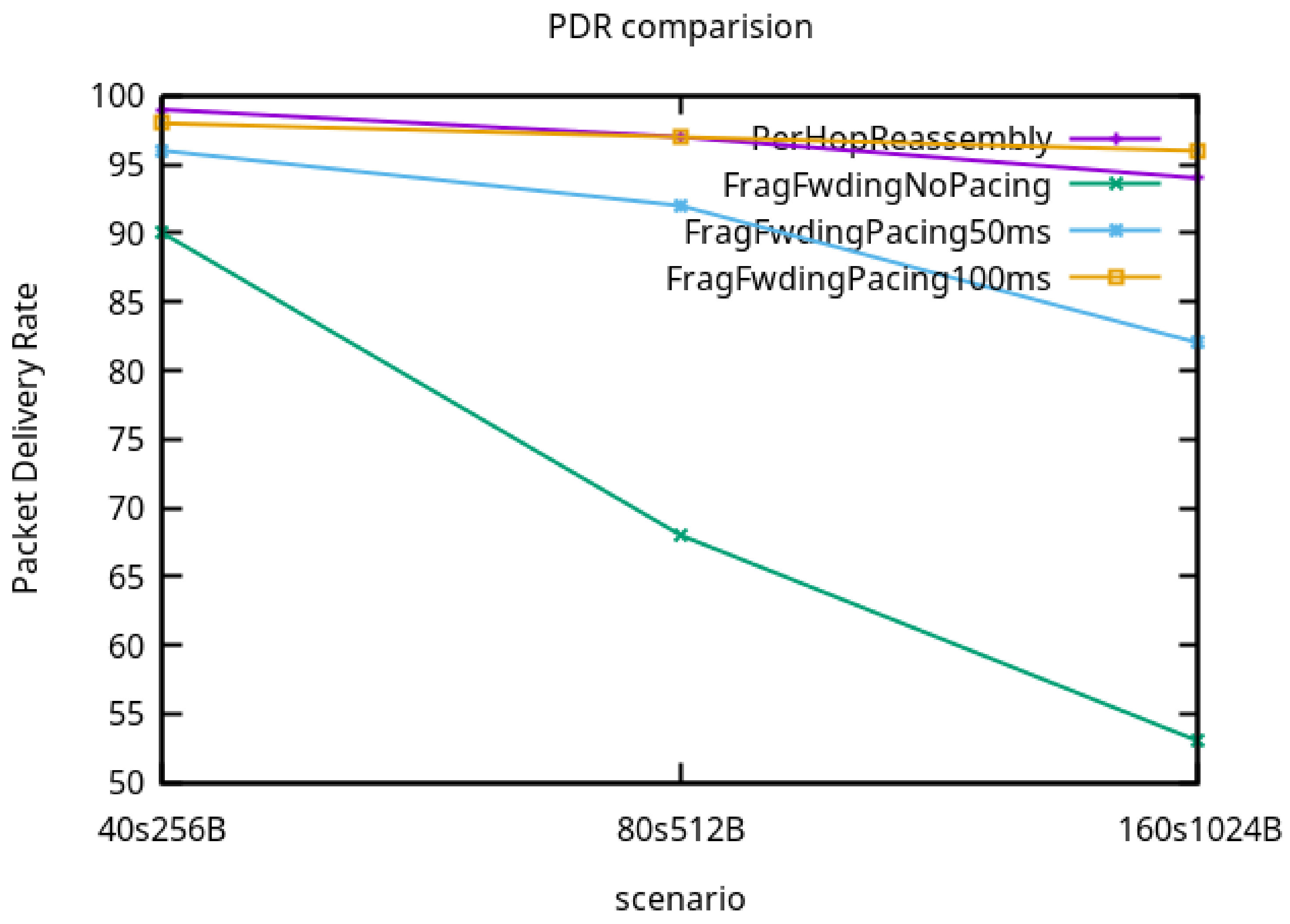


Sample Topology in tree format

Data transmission

- Send frequency for every node
  - 40s with UDP payload of 256B, results in 3 fragments
  - 80s with 512B, results in 5 fragments
  - 160s with 1024B, results in 9 fragments
  - Please note that every node app adds random delay between 0.5s to 5s before transmitting
  - All the data destined to BR

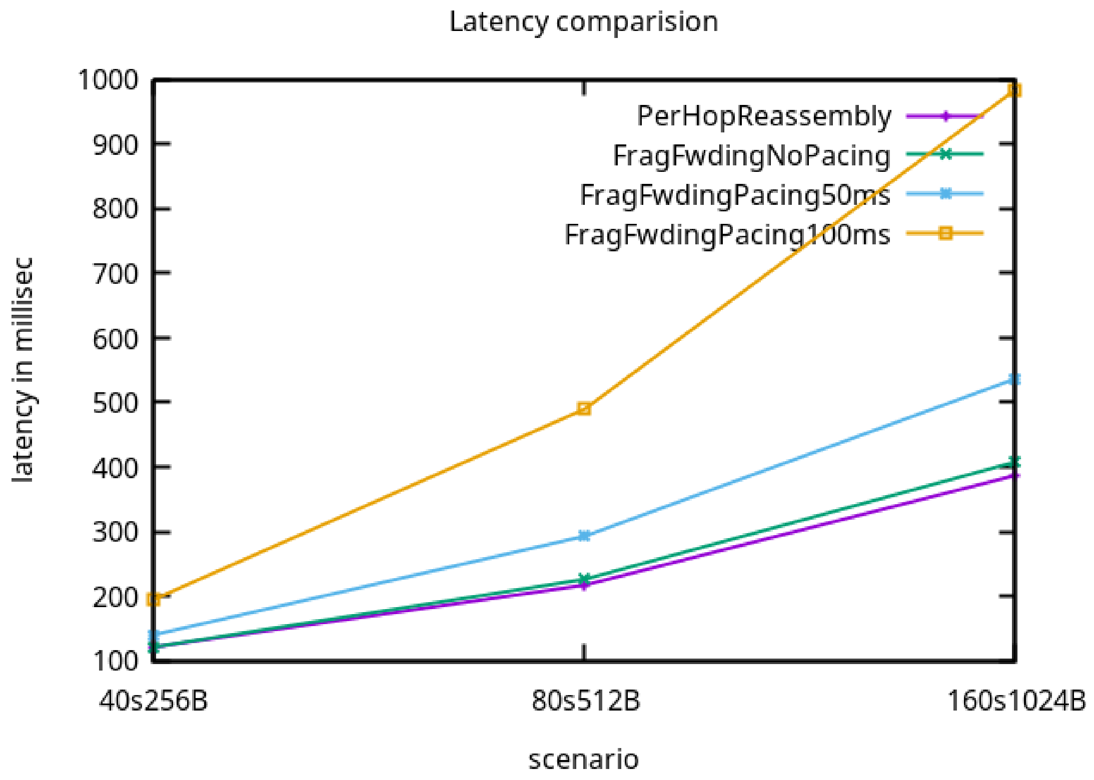
Data: PDR (Packet Delivery Rate)



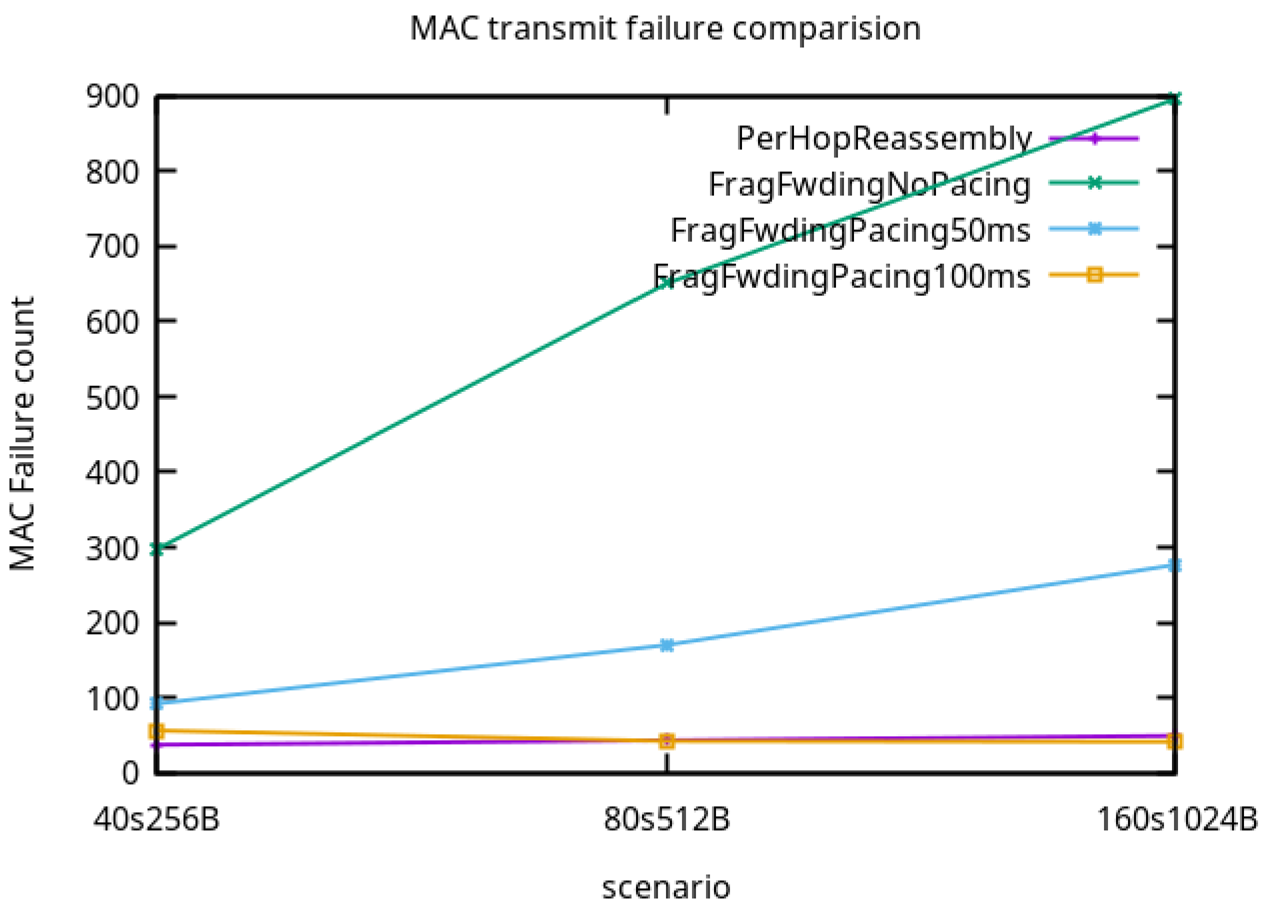
PDR of FF without pacing was sub-optimal.
Pacing improved it significantly.

Pacing? Impact on latency?

- Add inter-fragment fixed delay on original sender side
  - We tried 50ms and 100ms fixed delay
  - Pacing allows the fragment receiver to receive and subsequently forward the fragment without interference
  - Thanks to Carsten and Pascal for this discussion
- Pacing improved PDR drastically
- But pacing induced serious latency



Reasoning: MAC transmit failure



Please note that these are MAC transmit failures.. The packets delivered in first, second, third attempt are mentioned in the performance report. 2nd/3rd attempts are also much high for FragFwdingNoPacing case.

Observations (with this L2 setup)

- FF seems to depend on pacing
  - But if you add pacing, the latency is impacted negatively
- Per hop reassembly seems to be doing better, in this case, both in terms of PDR & latency
- Clearly, L2 primitives have big impact on such schemes
- Note: fragment drop due to memory unavailability were very less
  - Grid topology has less impact of bottleneck nodes
  - traffic pattern was sparse
- Fragment-Recovery might help
  - More fragments, higher payload loss probability. Not burst losses, usually.

Tools

- Simulation tool
  - Whitefield-Framework (using NS3-lrwpan backend for realistic RF)
- Implementation
  - FF support added in forked Contiki
    - Implementation adds slack (reserves extra bytes) in the first fragment
    - Slack is needed because the first fragment size might change en-route because of varying 6lo compression at each hop
    - Timer (60sec) to clear off entries in fragment table in case all fragments do not arrive
  - Contiki already supports per-hop reassembly

More experiments needed

- Experiment with different RFs
  - 6TiSCH
  - Ad-hoc 802.11 with RTS/CTS
  - 802.11s uses L2-mesh … This will result in fragment-forwarding like behavior.
- More optimal pacing algorithms needed
  - Should pacing be done at original sender-side only?
    - Trivial to implement
  - Will it help if done at intermediate hops?
    - non-trivial to implement since there could be multiple forwarding sessions in parallel
- Experiment same using a hardware based setup

Ack: Thanks to

- **Yatch** for sharing his insights into his experiments
- **Carsten** and **Pascal** for great discussions on 6lo-FF-design-team ML