

Deterministic Networking (DetNet) Data Plane Flow interleaving

for scaling detnet data planes with minimal end-to-end latency and large number of flows

draft-eckert-detnet-flow-interleaving-01

Toerless Eckert, Futurewei Technologies USA (tte@cs.fau.de),

IETF 117 DetNet WG meeting , rev 0.1

<https://github.com/toerless/detnet/tree/main/slides>

Draft Goal / Summary

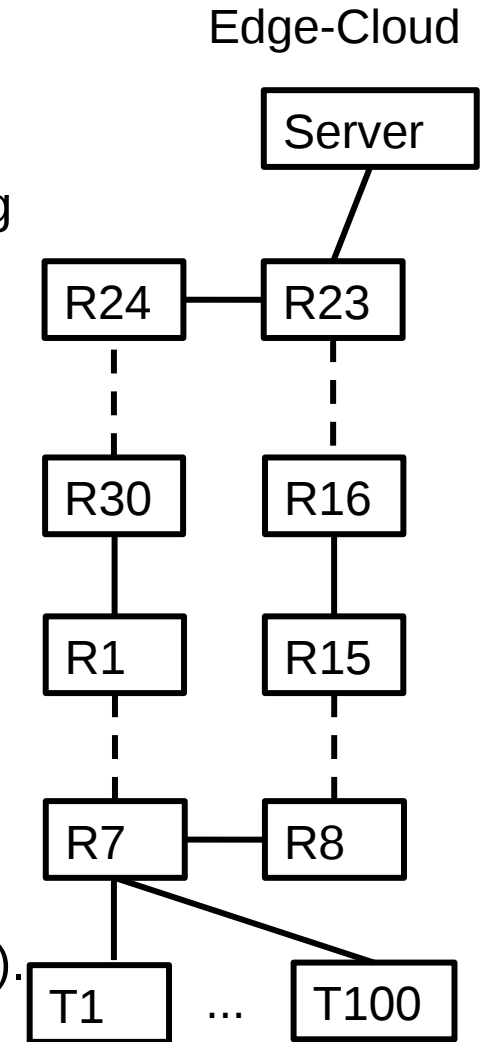
- TSN has “Gates” to exactly time packets from flow
 - Flexible tool for different purposes.
 - Used hop-by-hop to build CQF
 - Used on ingress and even hop-by-hop for “flow-interleaving”
 - Other uses ?
- DetNet Architecture should have Gates for Flow Interleaving
 - I do not think we included them yet into our architecture
 - Should be necessary only as ingress/edge function into DetNet domain (“PE function” only)
 - Could leverage existing TSN HW capabilities where they exist.
 - But difficult to adopt/deploy if we just hope a mix of DetNet and TSN Gates thrown in will work.
 - Explicitly defining our spec needed (IMHO)
 - Could be simplified if needed for scale (TBD details)
- This draft just meant to be making that argument. Not the specification (yet)

Why / Use-cases

- Maximize possible DetNet utilization while maintaining minimal end-to-end queuing latency
 - Benefits proportional to number of hops
 - Benefits higher with lower link speeds
- Does work best in with on-time per-hop queuing mechanisms
 - Example: draft-eckert-detnet-tcqf (TCQF)
draft-chen-detnet-sr-based-bounded-latency (CSQF)
 - Text was actually split out from text to compare TCQF, SCQF
 - *Note: On-time queuing does not depend on this any more than in-time queuing does. But on-time can benefit from it more!*

Example use-case

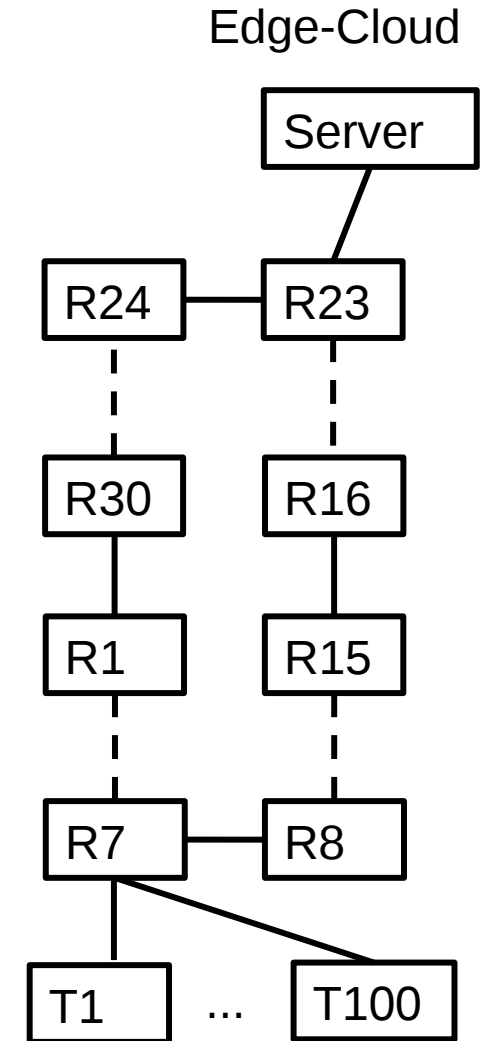
- Metro-Ring, e.g.: 30 hops (Bay-Area: 400 miles = 3 msec RTT light speed)
- TCQF 20 usec cycle time: $15 * 20 * 1.5$ (cycle) = 0.45 msec max one way queuing latency. Great low queuing latency re. distance RTT.
- 100 Gbps links = $166 * 1500$ byte packet/cycle
- Assume video based remote control applications
 - 50 FPS video, remote driving, video based machine control
 - Assume each flow 1 * 1500 byte packet / frame (20 msec) (simplified)
- Aka: many very low bitrate flows:
 - Each flow only needs a 1500 byte packet in one out of 1000 cycles.
- We can aggregate flows from any ingress R_i to egress R23, but that aggregation already requires to use gates to schedule each individual flow to be aggregated.
- Single server: 100 (clients) * 30 (ring routers) = 3000 flows (just one example app).
 - Still only 1.8 Gbps total – no bandwidth problem. Just low-latency management issue!



E.g.: radio towers

Flow Interleaving

- Solution: DetNet Controller Plane performs admission across e.g.: 100 msec worth of 20usec cycles (e.g.: 5000 cycles).
 - Each flow out of potentially tenths of thousands of flows (with low bitrates) get admitted to a set of periodic cycles, e.g.: 20 msec apart. Flows are “distributed” across that time scale
 - Not that difficult. Draft has example algorithm to do such “cycle-admission-control”. $O(\text{path-length}) * O(\#\text{cycles/period})$ complexity.
 - Complexity of finding optimized paths in complex topologies much higher. And that is done today in every large SP network.
- Gates on ingress router is enforcing that packets for a flow are only released into DetNet domain at the right cycle(s)



E.g.: radio towers

~~This is just a problem using cycles!~~

- Let's compare *QF (CQF, TCQF, CSQF), with TSN-ATS:
- In TSN-ATS, per-hop guaranteed queuing latency is the sum of the burst size of flows
- To guarantee e.g.: 20 usec per-hop queuing latency, TSN-ATS could also only admit 166 flows with a burst-size of 1500 bytes
- Same problem of: how do we break through the 166 barrier
- But not the same good solutions to it as we have with *QF
- TSN-ATS is “in-time”: Jitter adds up: 20 usec/hop - compared to 20 usec end-to-end with *QF:
 - Can not manage network capacity/latency in 20usec increments
 - Maybe manage latency in worst-case intervals: $30 \text{ (hops)} * 20 \text{ usec} = 600 \text{ usec}$ “virtual cycles”.
 - Aka: 30 times less possible utilization. The one example application 1.8 Gbps could barely be made to fit 100 Gbps.
- On-time mechanisms can really only increase per-hop burst-size/latency to manage utilization.
 - Aggregation of multiple flows from same ingress to same egress no affected (gates can perfectly be used for that).

TCQF vs CSQF

- With TCQF, the cycle on every hop along the path is pre-determined by the initial cycle in which it is sent into the TCQF domain
- When doing frame interleaving, some cycles along the path may be filling up faster than others... And may not be possible to avoid
 - Example: application with 50 fps and 60 fps have cycle interference
- With CSQF it is possible to spread out traffic across adjacent cycles easily and avoid these “overloaded cycles”
- How important is this CSQF benefit ?
 - Needs probably more complex real-world example than I have done so far
 - But good to know that we easily have this option.

Summary

We should include Gates into DetNet architecture

They help with flow-interleaving to increase utilization at low latency in DetNet.

A) With any per-hop queuing mechanism for flows from same ingress to same egress

B) When using on-time per-hop queuing mechanism (*QF),
they permit even higher utilization by allowing interleaving
between flows from different ingress to different egress routers.

Benefit A) may be possible to do in a proprietary fashion (although not preferable), but B) requires a standard, so that controller-plane can coordinate the gates across multiple routers.