Deterministic Networking (DetNet) Data Plane MPLS TC Tagging for Cyclic Queuing and Forwarding (MPLS-TC TCQF)

draft-eckert-detnet-mpls-tc-tcqf-00

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

Stewart Bryant, University of Surrey ICS (s.bryant@surrey.ac.uk)

09/09/2021

Why ?

- Bounded latency solution for DetNet …
- Derived from IEEE TSN “Cyclic Queuing and Forwarding” (CQF 802.1Qch-2017)
  - CQF Problem: assigns pakets to cyclic buffer by receive timestamp
    - Limits range to few Km hops, then hop latency exceeds cycle time.
    - Requires sub cycle-time accurate cock synchronization
- Tagged CQF (TCQF)
  - Solves CQF issues by carrying cycle identifier in packet header field
  - Allows for more than 2 cycles
- Benefits
  - Simple/cos-effective queuing for high-speedlow-cost hardware, 100Gbps validated
  - No per-hop, per-flow queuing state – no per-hop, per-queue controller plane interaction needed
    - Scale, operational simplicity
    - Can optionally support DiffServ model for DetNet (not required)
  - No limitation for link propagation latency -> enables wide-area network DetNet services
  - Simple controller plane calculus for admitting flows and calculating latency
    - E.g.: Latency calculated in number of cycle times for each hop, admission based on bytes per cycle for a flow
  - Tightly bounded jitter (cycle-time eg 100usec, 20usec,...)
    - Can support all TSN traffic classes up to isochronous
    - Bounded latency

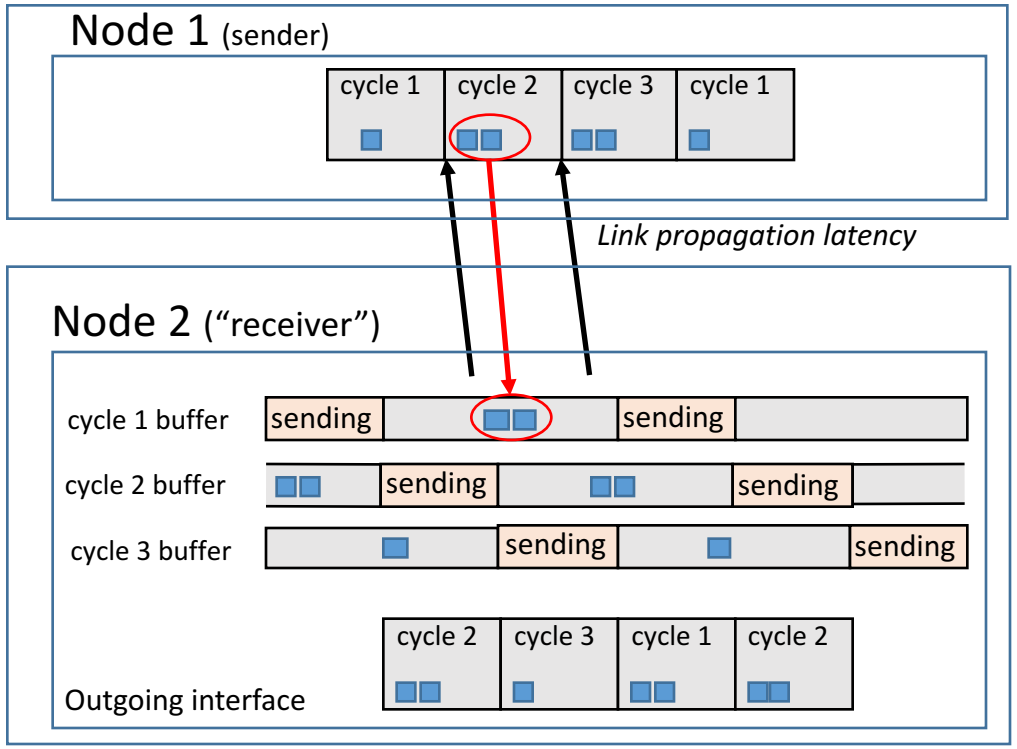
Tagged CQF with Multiple Buffers

Cycle Identifier (1..N, N = 2,3,..) in packet header

Receiving router preconfigured with cycle mapping table input to output cycle.

Packet put into output cycle buffer – and sent when its cycle is up.

More cycles allow for more variation in forwarding: MTIE and link propagation variation



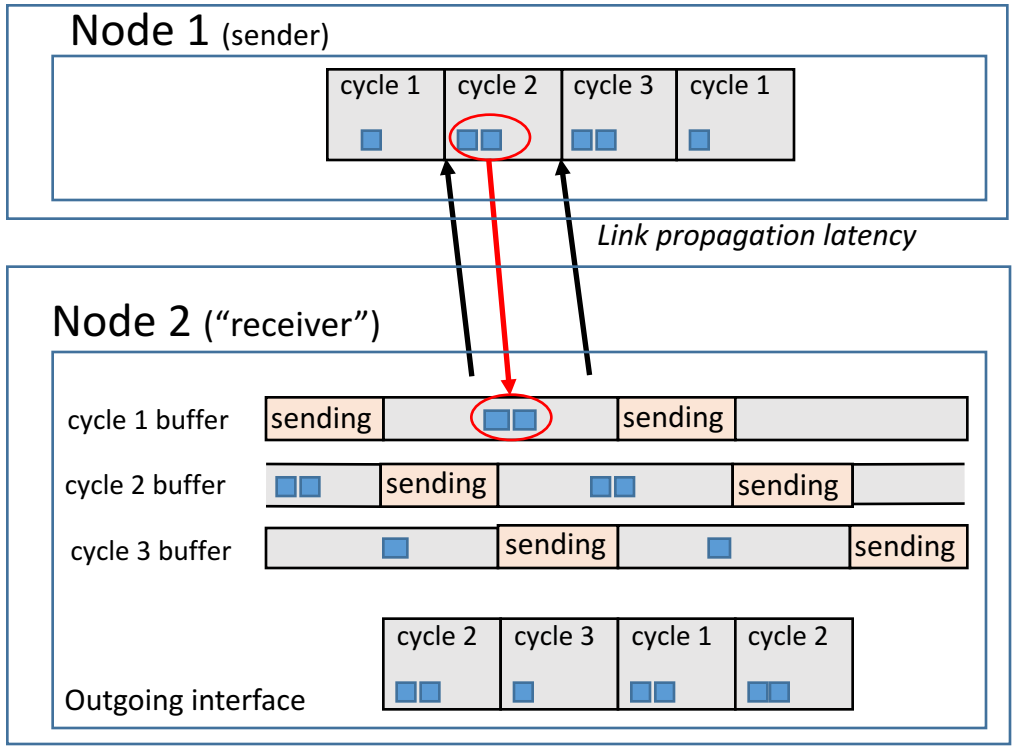
Tagged CQF with Multiple Buffers

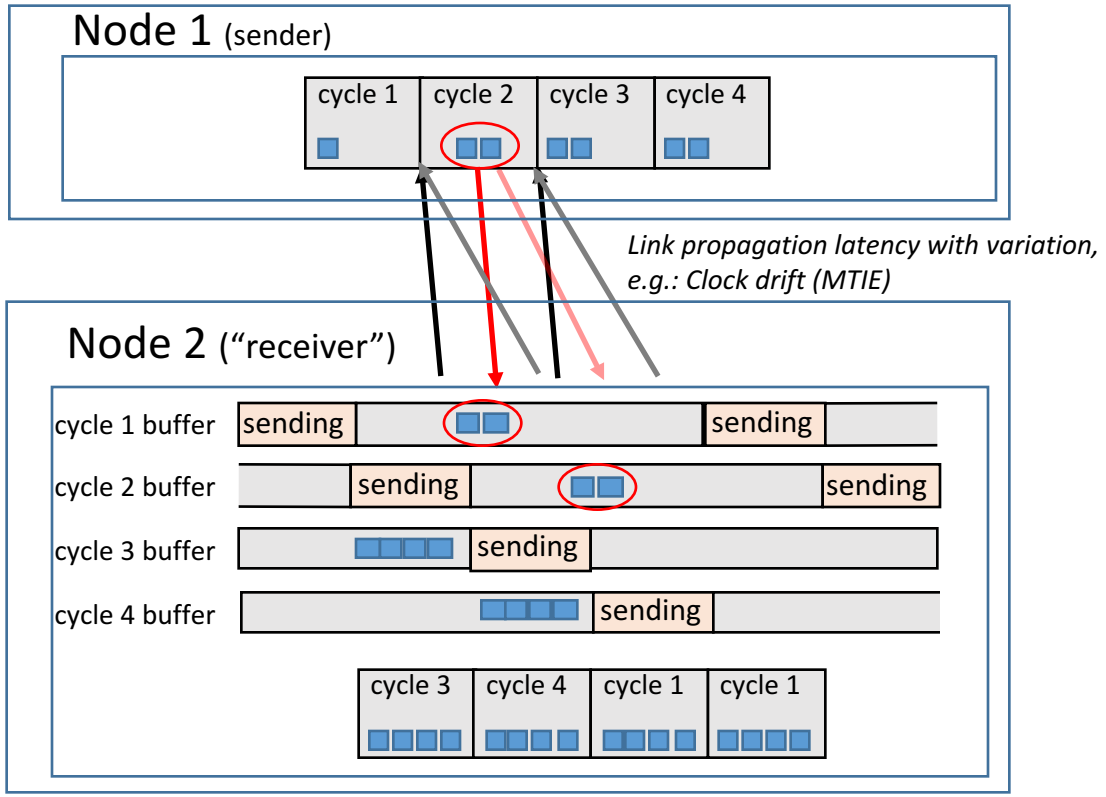
Cycle Identifier (1..N, N = 2,3,..) in packet header

Receiving router preconfigured with cycle mapping table input to output cycle.

Packet put into output cycle buffer – and sent when its cycle is up.

More cycles allow for more variation in forwarding: MTIE and link propagation variation





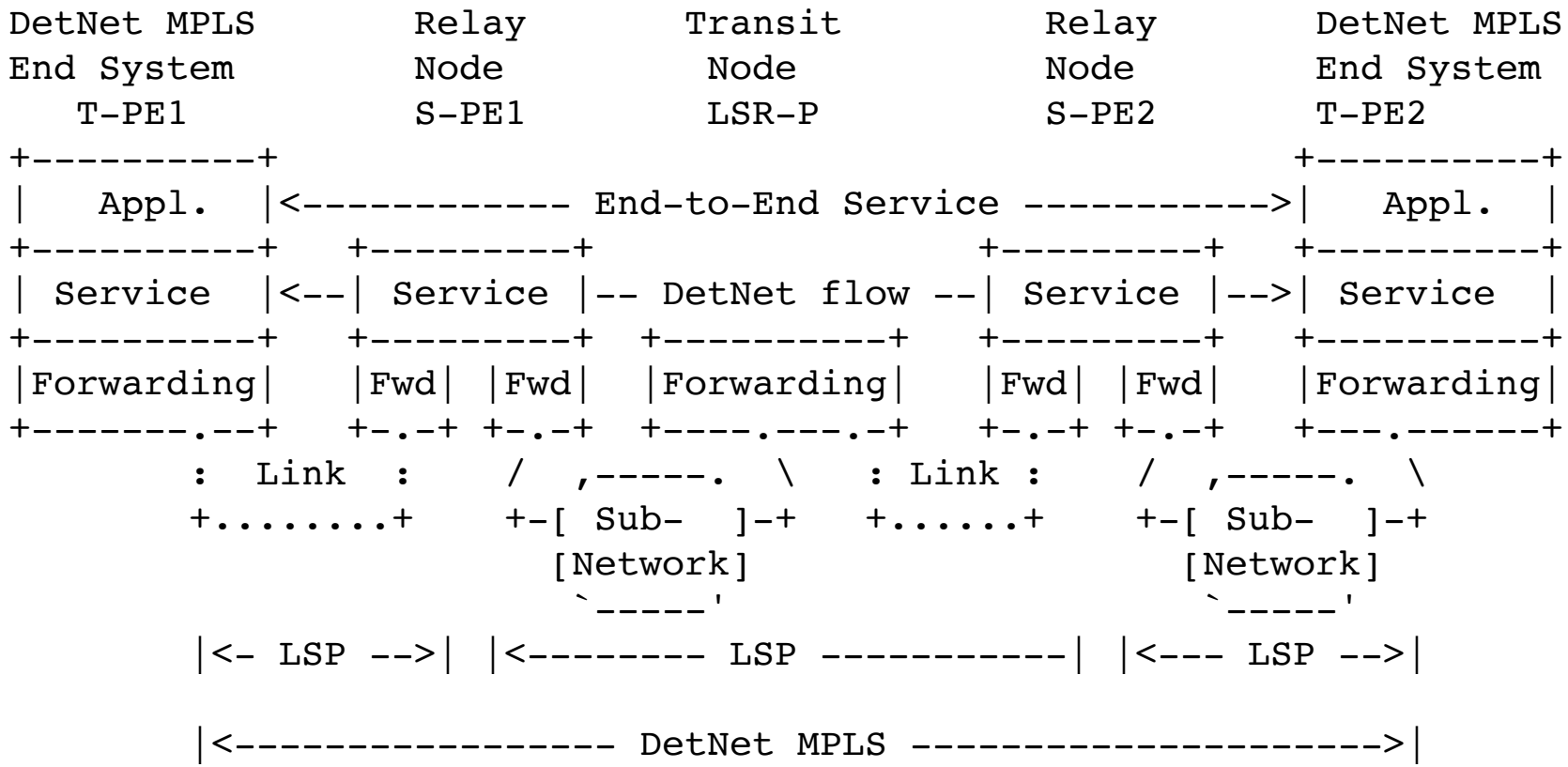
How

- Multiple Buffer, Tagged CQF decription of operations, benefits:
  - draft-qiang-detnet-large-scale-detnet (expired), draft-dang-queuing-with-multiple-cyclic-buffers
  - Informational – do not specify actual forwarding plane specific encap/forwarding behavior.

- This draft
  - Proposed standards track TCQF spec for DetNet MPLS Data Plane
    - Using Traffic Class (TC) of Top of Stack (ToS) MPLS Label Stack Entry (LSE)

  - Complies with MPLS DiffServ [RFC3270] TC-inferred PSC LSP (E-LSP) behavior
    - Occupies one TC value by cycle (e.g.: 4 TC for prior slide 4 cycle setup)

Section 2/4: Using TCQF with DetNet MPLS



- Can run T-PE1 to T-PE2 – or on any subset (S-PE1 – S-PE2)
- TBD: ingres function: shape DetNet flows into cyclic queuing (some known TSN gate functions or similar).
- TCQF state not tied to DetNet flow state in forwarding plane on midpoints (but in controller plane for admission control)
  - ToS LSE that holds TC with TCQF semantic can be Servic Label (S-Label) or Forwarding Label (F-Label) – up to controller plane
- Draft discusses when DetNet service layer operation require to re-do ingres function
  - Only Packet Reordering Function (PRF) – causes change in burstyness of flow packets.

Section 3: Data model (and textual forwarding description)

```
module ietf-detnet-tcqf
  augment TBD
    +--rw tcqf-config
    |  +--rw cycles uint16
    |  +--rw cycle-time uint16
    |  +--rw cycle-clock-offset uint32
    |  +--rw tcqf-if-config* [oif-name]
    |     +--rw oif-name       if:interface-ref
    |     +--rw cycle-clock-offset int32
    |     +--rw tcqf-iif-cycle-map* [iif-name]
    |        +--rw iif-name    if:interface-ref
    |        +--rw iif-cycle-map* [iif-cycle]
    |           +--rw iif-cycle  uint8
    |           +--rw oif-cycle  uint8
    |
    +--rw tcqf-mpls-tc-tag* [name]
       +--rw name          if:interface-ref
       +--rw cycle* [cycle]
          +--rw cycle  uint8
          +--rw tc    uint8
```

- YANG data model to configure TCQF: domain-wide cycle-time, cycles, clock-offsets
- Per TCQF Interface config: per-input-interface cycle-map table
- Encapsulation dependent: per-interface cycle-ID to packet-tag mapping (here only MPLS-TC encap specified)

Section 5: Data plane pseudocode (1)

Receive … TCQF enqueue packet

```
tcqf = ietf-detnet-tcqf

void receive(pak) {
  // Receive side TCQF - remember cycle in
  // packet internal header
  iif = pak.context.iif
  if(tcqf.tcqf-if-config[iif]) { // TCQF enabled
    if(tcqf.tcqf-mpls-tc-tag[iif]) { // TC-TCQF
      pak.context.tcqf_cycle =
        map_tc2cycle(pak.mpls_header.lse[tos].tc,
             tcqf.tcqf-mpls-tc-tag[iif])
    } else // future encap/tagging options for TCQF
  }
  // Forwarding including any label stack operations
  oif = pak.context.oif = forward_process(pak)

  // ... optional  DetNet PREOF functions here
  // ... if router is DetNet service node
```

```
// Output interface TCQF enqueuing

if(pak.context.tcqf_cycle && // non TCQF packets value is 0
  tcqf.tcqf-if-config[oif]) { // TCQF enabled
  // Map tcqf_cycle for iif to oif mapping table

  cycle = pak.context.tcqf_cycle =  map_cycle(cycle,
       tcqf.tcqf-if-config[oif].tcqf-iif-cycle-map[[iif])

  // Map cycle to TC value of ToS LSE
  if(tcqf.tcqf-mpls-tc-tag[iif]) { // TC-TCQF
     pak.mpls_header.lse[tos].tc =
       map_cycle2tc(cycle, tcqf.tcqf-mpls-tc-tag[oif])
  } else // future encap/tagging options for TCQF

  // Enqueue into cycle buffer
  tcqf_enqueue(pak, oif.cycleq[cycle])
```

Section 5: Data plane pseudocode (2)
TCQF packet sending

```
// Started when TCQF is enabled on an interface

   void send_tcqf(oif) {
     // Calculate correct time to start sending cycle 1
     cycle = 1
     cc =  tcqf.tcqf-config.cycle-time *
           tcqf.tcqf-config.cycle-time
     o =   tcqf.tcqf-config.cycle-clock-offset
     nextcyclestart = floor(tnow / cc) * cc + cc + o

     // Send cycles forever
     while(1) {
       while(tnow < nextcyclestart) { } // Wait until next cycle starts
       while(pak = dequeue(oif.cycleq(cycle)) {
         send(pak)
       }
       cycle = (cycle + 1) mod tcqf.tcqf-config.cycles + 1
       nextcyclestart += tcqf.tcqf-config.cycle-time
     }
   }
```

More

- Details in doc, not explained in slides:
  - Example controller plane math to calculate cycle mapping

- Missing / TBD ?!
  - Ingres function, shapping into cycles
    - Do we need this in TCQF specific ? TSN Gates would work.
    - We would likely want/need those for other DetNet QoS options too ?
    - E.g.: TBD what to do
  - Complete YANG model specification
    - Just writing exercise
  - What else ?

Q & A