1. Proposed improved template
2. Evaluations

TCQF / CSQF: draft-eckert-detnet-tcqf ,draft-chen-detnet-sr-based-bounded-latency

gLBF: draft-eckert-detnet-glbf

3. Revisited ECQF evaluation (Appendix)

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

for the authors of above drafts

IETF DETNET Interim, 09/2023 rev 1.0 9/11/2023

Proposed improved Template

And why

Enhancements to evaluation sheet

- New column to indicate whether requirement is per-hop (P), or ingress (PE).
- Split 3.1 into its four constituent requirements (3.1.1 - 3.1.4)
  - No evaluation for 3.1 overall, just the four individual ones

- Modified 3.1.4 to “Support aperiodic flows” – indicate whether on P or PE
  - Requirement for no-clock-sync already covered by 3.1.3
  - Requirement draft implies this is only possible with per-hop asynchronous mechanisms. This is not true.
  - E.g.: TCQF/CSQF can support aperiodic flows via ingress-PE flow-interleaving (adds latency on PE hop)
    - E.g.: Delay on ingress by up to 500 usec to keep small per-hop (cycle) latency on all following hops (spread bursts over 500 usec)
  - Asynchronous per-hop (P) method (rfc2211/Qcr/gLBF) come at the price of increased per-hop guaranteed latency which adds up across long paths! – See draft-eckert-detnet-flow-interleaving

- 3.2.1 Added evaluation for single-hop propagation jitter (NEW)
  - Because of: RAW (reflections, interference, retransmission), length deviation (wired)

- 3.8(1) List explicit evaluation for “support tight jitter (per-hop)”
  - as mentioned in requirements document as one possible option of traffic (e.g.: industrial control loops)

<Mechanism> evaluation (<notes>)

Sec.	P/PE	Requirements	Eval	Notes
3.1.1	P/PE	Async across TSN subdomains		changes from overall section 3 eval to individuals
3.1.2	P	Tolerate Clock Jitter/Wander		
3.1.3	P	No Full Time Sync required		
3.1.4	P/PE	Support for aperiodic flows		
3.2	P	Large Single-hop Propagation Latency		
3.2.1	P	Support single-hop propagation jitter		New
3.3	P	Support Higher Link Speed		
3.4(1)	P	Scalable to Large Number of Flows		Details from prior reviews
3.4(2)	P	Tolerate High Utilization		Details from prior reviews
3.5	P	Link/Node failures, Topo Changes		
3.6	P	Prevent Flow Fluctuation from Disrupting Service		
3.7	P	Scalable to Large Number of Hops with Complex Topology		
3.8(1)	P	Support tight jitter/sync-control loops		New

Evaluations used / suggested

3.1.1	YES if hop-by-hop mechanism has defined ingress function, and/or assumed to be combined with some pre-existing ingress function. Describe or reference to ingress function in notes.
3.1.2	Suggest YES if permitted drift (short term variation of clock frequency) on a 100Gbps (or faster) link can be as large as 1 usec (e.g.: no as tight as 100 nsec or less).
3.1.3	YES if MTIE (maximum time interval error) over arbitrary periods can be infinite. Or (simpler): if clock wander over 24 hours can be larger than 100ppm ~= 8 seconds.Partial: clock sync requirements are proven to work for large-scale networks
3.1.4	Indicate whether support for aperiodic flows happens per-hop (P) or on PE.
3.2	YES if 1000km links can be supported (on every hop)
3.1.2	NEW: YES if jitter in link latency can be supported (length variation, L2 retransmission jitter on mobile/radio links,...)
3.3	YES if 100Gbps or faster links can be supported without loss of performance (e.g.: higher dead times) than for slower links.
3.4(1)	YES if mechanism does not require per-hop, per-flow state requiring per-packet lookup and read/write cycles.
3.4(2)	YES if 100Gbps or faster long latency links permit same close to 100% DetNet traffic
3.5	YES if Segment Routing and fast-reroute options with resource reservation can be supported
3.6	YES if mechanism does not have burst-accumulation across multiple network hops (no per-hop increase in flow burstyness) AND if mechanism allows to add/delete flows without impacting prior latency/jitter guarantees of pre-existing flows
3.7	YES if 3.6 is YES AND if flow interleaving can be applied to the mechanism across flows from different ingress/egress PE
3.8(1)	YES if end-to-end jitter is independent of number of hops.

Justification: PE functions

- On ingress to a DetNet domain, all flows need to be processed on a per-flow basis because received traffic flows need to be policed to comply with their admitted traffic envelope.
  - Complexity / scalability on PE node can therefore never be as high as on P node.
  - *Note: in simple ring topologies, every node is a P and PE node. If a ring has 20 nodes, then a node will typically be PE for only 1/20'th of the overall flows. So P scalability is even important in these topologies. Larger ring topologies will have ring nodes that are hubs for a larger number of spokes, and hence ring nodes are only P nodes.*
- Hop-by-hop forwarding mechanism may need specific ingress PE processing beyond policing. This needs to be evaluated.
  - Example: cycle mechanisms need per-flow timing of admitting packets of a flow into specific cycles. This is a specific version of a generic policer.
- draft-eckert-detnet-flow-interleaving makes the argument that policing and timing of flows on ingress PE is a generic function that DetNet is currently missing, but which if defined could be combined with all hop-by-hop mechanisms and act as the common additional PE function
  - This would be a simplified version of the TSN timed gates function (which in TSN is allowed to be used on every hop, but in "scaling" DetNet only on ingress PE
- In current sheet, only one requirement is identified as "only applicable to PE". Other requirements have not been identified to have specific different requirements for PE, so only P is indicated.

TCQF / CSQF evaluation

Common evaluation because:

- Same common per-hop logic
  - Same ingress processing (PE)
  - N-cycle forwarding hop-by-hop
    - E.g.: any TSN switch could do it, 100Gbps low-cost validation exist
  - Packet is assigned to cycle by a packet header field.
- Difference
  - TCQF: packet header field is single cycle-field, rewritten on every hop
    - Allows to use IP-DSCP, MPLS-TC – and thus work in IP/IPv6 and MPLS networks – without SR.
    - Can also be combined with SR (orthogonal)
  - CSQF: cycle derived on every hop from Segment Routing header
    - N SIDs per hop to indicate N different cycles
    - IPv6: SRH, future compressed SRH, SR-MPLS
- We think:
  Any implementation that can do TCQF and supports SR can also do CSQF
- *Differences relevant to evaluation only in one point*

TCQF & CSQF evaluation

Sec.	P/PE	Requirements	Evaluation	Notes
3.1.1	PE	Async across TSN subdomains	Yes	TCQF defines async ingres. Same applicable to CSQF
3.1.2	P	Tolerate Clock Jitter/Wander	Yes	E.g.: with 4 cycles, max jitter/wander 1 cycle time. **Main benefit over (E)CQF** *ECQF can not do this: arrival time inaccuracy leads to wrong cycle assumption*
3.1.3	P	No Full Time Sync required	partial	Same as CQF, ECQF, TAS. **Main benefit of gLBF (which has YES) Partial: ca. 90% lower accuracy clock sync req. than with CQF/ECQF**
3.1.4	PE	Support for aperiodic flows	Yes	Via aperiodic burst shape/delay on ingress PE or overprovisioning.
3.2	P	Large Single-hop Propagation Latency	Yes	**Main benefit over TAS, CQF**. Same as ECQF.
3.2.1	P	Support single-hop propagation jitter	Yes	**Benefit over ECQF , CQF, TAS** (radio, retransmissions, length deviation)
3.3	P	Support Higher Link Speed	Yes	200km/100Gbps proven, can scale well beyond that. **Not proven for ECQF**
3.4(1)	P	Scalable to Large Number of Flows	Yes	No per-hop, per-flow state, read/write memory access requirement
3.4(2)	P	Tolerate High Utilization	Yes	**Solves CQF issue, equal/better than ECQF**: No dead times
3.5	P	Link/Node failures, Topo Changes	Yes	Can support Segment Routing and hence all re-route, path-fixing options **(TCQF, CSQF). CSQF**: even defined solely for SR!
3.6	P	Prevent Flow Fluctuation from Disrupting Service	Yes	Like CQF/ECQF: No burst accumulation/jitter-increase due to per-hop cycle based reshapeper-hop reshaping.
3.7	P	Be Scalable to a Large Number of Hops with Complex Topology	Good (TCQF) Best (CSQF)	Assuming flow interleaving on edge, like CQF/ECQF **CSQF more flexible** than TSQF, CQF/ECQF
3.8(1)	P	Support tight jitter/sync-control loops	Yes	Network size independent hop-by-hop/end-to-end jittter ~ O(0)

Comparison CQF / ECQF / TCQF & CSQF - *"Yes" – with other TSN means*

Sec.	P/PE	Requirements	CQF	ECQF	TCQF/CSQF	Notes
3.1.1	PE	Async across TSN subdomains	"Yes"	"Yes"	Yes	TCQF defines async ingres. Same applicable to CSQF
3.1.2	P	Tolerate Clock Jitter/Wander	No	No	Yes	E.g.: with 4 cycles, max jitter/wander 1 cycle time. **Main benefit over (E)CQF** *ECQF can not do this: arrival time inaccuracy leads to wrong cycle assumption*
3.1.3	P	No Full Time Sync required	No	No	partial	Same as CQF, ECQF, TAS. **Main benefit of gLBF (which has YES)** **Partial: ca. 90% lower accuracy clock sync req. than with CQF/ECQF**
3.1.4	PE	Support for aperiodic flows	"Yes"	"Yes"	Yes	Via aperiodic burst shape/delay on ingress PE or overprovisioning.
3.2	P	Large Single-hop Propagation Latency	No	Yes	Yes	**Main benefit over TAS, CQF**. Same as ECQF.
3.2.1	P	Support single-hop propagation jitter	No	No	Yes	**Benefit over ECQF , CQF, TAS** (radio, retransmissions, length deviation)
3.3	P	Support Higher Link Speed	partial	partial	Yes	200km/100Gbps proven, can scale well beyond that. **Not proven for ECQF**
3.4(1)	P	Scalable to Large Number of Flows	partial	partial	Yes	No per-hop, per-flow state, read/write memory access requirement
3.4(2)	P	Tolerate High Utilization	No	Yes	Yes	**Solves CQF issue, equal/better than ECQF**: No dead times
3.5	P	Link/Node failures, Topo Changes	No (TSN)	No (TSN)	Yes	Can support Segment Routing and hence all re-route, path-fixing options **(TCQF, CSQF). CSQF**: even defined solely for SR!
3.6	P	Prevent Flow Fluctuation from Disrupting Service	Yes	Yes	Yes	Like CQF/ECQF: No burst accumulation/jitter-increase due to per-hop cycle based reshapeper-hop reshaping.
3.7	P	Be Scalable to a Large Number of Hops with Complex Topology	No (TSN)	No (TSN)	Good (TCQF) Best (CSQF)	Assuming flow interleaving on edge, like CQF/ECQF **CSQF more flexible** than TSQF, CQF/ECQF
3.8(1)	P	Support tight jitter/sync-control loops	Yes	Yes	Yes	Network size independent hop-by-hop/end-to-end jittter ~ O(0)

3.4 explanations

- No per-hop, per-flow state in routers
  - No need for per-packet “single-cycle” read, calculate, write processing of router flow-state memory as in e.g.: TSN-ATS interleaved regulators.
- Large scale simulation result in 2020 published IEEE research paper
- 100Gbps low-cost FPGA router based, 2000km long network path validation in China
  - Includes feasibility of network wide clock sync, e.g.: invalidates req. 3.1.3 in the context of PTP as required for TCQF/CSQF
- Minimum HW complexity, number of queues / buffers required.
- Reduced accuracy of clock synchronization / cycle timing vs. CQF / ECQF
  - Low per-packet clock accuracy across complex, large router
  - Because cycle number is in packet header, sender to receiver can have clock jitter / wander.

3.7 Explanations / justifications

- Per-hop cycles avoid burst-accumulation (3.6). Strict per-hop latency.
- Controller-plane manages per-flow resources per-hop.
  - No per-hop forwarding plane config/state impact when flows are added/removed / re-routing

- Addl. Improvement: Allocation of per-hop, per-cycle resources by controller-plane as for example explained in draft-eckert-detnet-flow-interleaving
  - Timed gates on ingress PE router, no other forwarding plane impact.
  - Examples shows path-length linear compute cost when allocating new flows, up to high utilization

- Same logic applies to all of TCQF/CSQF, CQF/ECQF
  - CSQF more flexible here than TCQF, CQF/ECQF – because it can define cycles per-hop in the packet header
    - SR header as proposed, compressed SR header (SPRING) or DetNet header (e.g.: 4-bit/hop should be enough).

  - Degree of improvement (higher utilization) for CSQF over TCQF ... 20% ?

  - 3.5: SR resilience options may be biggest benefit of CSQF over TCQF.

gLBF evaluation

gLBF evaluation **(vs. CQF/ECQF/gLBF highlighted)**

Sec.	P/PE	Requirements	Evaluation	Notes
3.1.1	PE	Async across TSN subdomains	Yes	Naturally async, no special ingress function needed
3.1.2	P	Tolerate Clock Jitter/Wander	Yes	*Max per-hop jitter/wander just needs to be known/over-estimated during config*
3.1.3	P	Tolerate Time Asynchrony	**Yes**	**Main benefit over CQF / ECQF / TCQF / CSQF**
3.1.4	P/PE	Support for aperiodic flows	Yes	Directly via P, or via additional ingress PE timed gates to increase utilization at lower end-to-end latency
3.2	P	Large Single-hop Propagation Latency	Yes	*Max hop propagation latency to be known/over-estimated during net. config.*
3.2.1	P	Support single-hop propagation jitter	Yes	*Requires clock sync only across jittery link – not other links*
3.3	P	Support Higher Link Speed	TBD	Not yet proven in high-speed ASIC implementation. Target timed FIFO implementation model suggested for next-gen high-speed, low-cost.Impl.
3.4(1)	P	Scalable to Large Number of Flows	Yes	No per-hop, per-flow state, read/write memory access requirement
3.4(2)	P	Tolerate High Utilization	Yes	No dead times
3.5	P	Link/Node failures, Topo Changes	Yes	Can support all Segment Routing re-route/path-control options.
3.6	P	Prevent Flow Fluctuation from Disrupting Service	Yes	No burst accumulation/jitter-increase due to per-hop latency based re-shaping.
3.7	P	Be Scalable to a Large Number of Hops with Complex Topology	Yes/TBD	Flow interleaving TBD
3.8(1)	P	Support tight jitter/sync-control loops	Yes	Network size independent hop-by-hop/end-to-end jittter ~ O(0)

Explanations

3.4

- No per-hop, per-flow state in routers
- No per-hop, per-flow state in routers
  - No need for per-packet “single-cycle” read, calculate, write processing of router flow-state memory as in e.g.: TSN-ATS interleaved regulators.

3.7

- TSN-ATS per-hop latency and calculus, but without per-hop/flow shaper (3.4)
  - Damper mechanism to prohibit burst accumulation, replaces shapers.
  - No control-plane to-per-hop signaling required.
  - Flow changes only impact controller-plane, ingres-edge-router
- Applicability of flow interleaving for high utilization under review

ECQF Eval revisited

CQF / ECQF eval revisited

- *These slides only for comparison during discussion*

- ECQF 3.1 evaluation wrong. Was Yes, should be No:
  - Definition Asynchrony: drift (MTIE) proportional to time, synchronous: drift (MTIE) has upper bound across arbitrary periods.
  - Assume N (2,3,4) cycles: when clock drift exceeds N-cycle time, cycle mechanism will fail.

- Note on 3.3 evaluation:
  - 10Gbps (fastest known CQF) -> 400 Gbps: 40x more accurate clock synchronization needed (impossible) to keep same percentage dead time. This improvement in accuracy is impossible.
  - Therefore, dead time for 100G needs to be 10x higher than for 10G, and 40x higher than for 400G. Equally applies to ECQF.
  - When packet arrives, arrival time determines assumed sending time. If sending time is within margin of accuracy, receiver time may be in the wrong cycle!

CQF evaluation

- **CQF (Cyclic Queuing and Forwarding), refer to IEEE 802.1Qch**

sectionc	Requirements	Evaluation	Notes
3.1	Tolerate Time Asynchrony	No	Rely on nanosecond clock synchronization across the entire network, where all network nodes align their cycle boundaries.
3.2	Support Large Single-hop Propagation Latency	No	Link delay must be much smaller than cycle duration and considered negligible, so 2-buffer mode can work. Otherwise, 3-buffer (or more) mode is needed.
3.3	Accommodate the Higher Link Speed	Partial	More buffer space is required for a specific length of cycle duration. Smaller cycle size may be chosen, but with a much smaller available zone.
3.4(1)	Be Scalable to the Large Number of Flows	Partial	Transmission gates are associated with each aggregated queue, but the stream filtering and policing actions per stream should be maintained.
3.4(2)	Tolerate High Utilization	No	The cycle duration includes a time zone called dead time (DT) contributed by Output delay, Link delay, Frame preemption delay, Processing delay, which can not be used to send packets.
3.5	Prevent Flow Fluctuation from Disrupting Service	Partial	Requires corresponding flows setup algorithms to allocate resources appropriately among multiple flows to avoid transmission conflicts.
3.6	Tolerate Failures of Links or Nodes and Topology Changes		Not related to queuing mechanisms directly.
3.7	Be Scalable to a Large Number of Hops with Complex Topology	Partial/No (No from first eval)	It is more difficult to select the cycle time. Need making trade-offs between end-to-end delay and cycle duration.
3.8	Support Multi-Mechanisms in Single Domain and Multi-Domains		Not related to a single queuing mechanism directly.

ECQF evaluation

- **ECQF (Enhancements to CQF), refer to IEEE 802.1Qdv**

section	Requirements	Evaluation	Notes
3.1	Tolerate Time Asynchrony	~~Yes~~ **NO**	Does not rely on time synchronization. Time-based CQF need frequency lock (frequency synchronization too). Count-based CQF can be more relaxed.
3.2	Support Large Single-hop Propagation Latency	Yes	The cycle phase difference between two nodes is allowed. CPAP detect message covers link propagation delay.
3.3	Accommodate the Higher Link Speed	Partial	More buffer space is required for a specific length of cycle duration. Smaller cycle size may be choosen, but with a much smaller available zone.
3.4(1)	Be Scalable to the Large Number of Flows	Partial	Transmission gates are associated with each aggregated queue, but the stream filtering and policing actions per stream should be maintained. Count-based bin need maintain state machine per flow.
3.4(2)	Tolerate High Utilization	No	Cyle size is always far less than burst interval, so overprovision (caused by burst / cycle) may cause low utilization. NOTE: based on the bandwidth reservation method in the document.
3.5	Prevent Flow Fluctuation from Disrupting Service	Partial	Time-based CQF: may ensure CQF flows to be protected. Count-based CQF: may discard excess data above the contracted amount.
3.6	Tolerate Failures of Links or Nodes and Topology Changes		Not related to queuing mechanisms directly.
3.7	Be Scalable to a Large Number of Hops with Complex Topology	Partial	It is more difficult to select the cycle time. Need making trade-offs between end-to-end delay and cycle duration.
3.8	Support Multi-Mechanisms in Single Domain and Multi-Domains		Not related to a single queuing mechanism directly.