

DETNET
Internet-Draft
Intended status: Informational
Expires: January 13, 2022

T. Eckert
Futurewei Technologies USA
S. Bryant
Stewart Bryant Ltd
July 12, 2021

Problems with existing DetNet bounded latency queuing mechanisms
draft-eckert-detnet-bounded-latency-problems-00

Abstract

The purpose of this memo is to explain the challenges and limitations of existing (standardized) bounded latency queuing mechanisms for desirable (large scale) MPLS and/or IP based networks to allow them to support DetNet services. These challenges relate to low-cost, high-speed hardware implementations, desirable network design approaches, system complexity, reliability, scalability, cost of signaling, performance and jitter experience for the DetNet applications. Many of these problems are rooted in the use of per-hop, per-flow (DetNet) forwarding and queuing state, but highly accurate network wide time synchronization can be another challenge for some networks.

This memo does not intend to propose a specific queuing solution, but in the same way in which it describes the challenges of mechanisms, it reviews how those problem are addressed by currently proposed new queuing mechanisms.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 13, 2022.

Internet-Draft

bounded-latency-problems

July 2021

Copyright Notice

Copyright (c) 2021 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Summary	3
1.1.	Problem: High speed forwarding, high scale fan-in/fan-out	3
1.2.	Solution goal: Lightweight, per-hop, per-flow stateless transit hop forwarding	4
1.3.	Requirement: Support for existing stateless / steering solutions	4
1.4.	Requirement: PCE to ingress/egress LSR only flow signaling	4
1.5.	Requirement: Support for DiffServ QoS model on transit hops.	4
1.6.	Requirement: Low jitter bounded latency solutions.	4
1.7.	Requirement: Dynamic, application signalled DetNet flows	5
2.	Evolution of IP/MPLS network technologies and designs	5
2.1.	Guaranteed Service with RSVP	5
2.2.	Hardware forwarding and DiffServ	6
2.3.	MPLS and RSVP-TE	6
2.4.	Path Computation Engines (PCE)	7
2.5.	Segment Routing (SR)	8
2.6.	BIER	8
2.7.	Summary	8
3.	Additional current considerations	9
3.1.	Impact of application based state in networks	9
3.2.	Experience from IP multicast	9
3.3.	Service Provider and Private MPLS Networks	10
3.4.	Mission-specific vs. shared infrastructures	11
3.5.	PTP and challenges with clock synchronization	12

3.6.	Jitter - in-time versus on-time	13
4.	Challenges for high-speed packet forwarding hardware	15
5.	A reference network design	16
6.	Standardized Bounded Latency algorithms	19
6.1.	Guaranteed Service (GS)	19

6.2.	TSN Asynchronous Traffic Shaping (TSN-ATS)	19
6.3.	Cyclic Queuing and Forwarding (CQF)	20
7.	Candidate solution directions	22
7.1.	Packet tagging based CQF	22
7.2.	Packet tagging based CQF with SR	23
7.3.	Per-hop latency indications for Segment Routing	23
7.4.	Latency Based Forwarding	24
8.	Conclusions	25
9.	Security Considerations	25
10.	IANA Considerations	25
11.	Acknowledgements	25
12.	Informative References	26
	Authors' Addresses	28

[1.](#) Summary

The architectural evolution of IP/MPLS networks ([Section 2](#)) in service provider and other "larger-than-building" ([Section 3.3](#)), shared-infrastructure service networks ([Section 3.4](#)) has led to a range of requirements against per-hop forwarding mechanisms which are currently not supported by the current DetNet MPLS forwarding plane [[RFC8964](#)] and per-hop, per-flow queueing model [[RFC8655](#)], [Section 3.2](#), especially with respect to the QoS support of per-hop bounded latency. The authors of this memo think that solutions for these requirements are relatively easily added to the existing DetNet architecture by adding support for already existing and/or proposed, but not standardized per-hop forwarding and queueing options.

The following sub-sections summarize the problem, solution goals and requirements as perceived by the authors. The reasoning for these is explained in the following sections.

Note that requirements are somewhat overlapping in so far as solving one of them also solves others, but each addresses the problems from a different perspective, and are therefore easier understood for different stakeholders. For example: Operators that do want to see

support of DetNet for example for Segment Routing (SR) would not think that this is "naturally" the same as DetNet supporting the DiffServ architecture, even though solutions would have a hard time to support only one of the two.

1.1. Problem: High speed forwarding, high scale fan-in/fan-out

Forwarders with bounded latency need to support interface speeds of 100 Gbps up to Tbps, likely over a period of 10 years from initial deployment of possible DetNet solutions. Hundreds of interfaces may need to be supported in a single forwarder (fan-in/fan-out).

Eckert & Bryant

Expires January 13, 2022

[Page 3]

Internet-Draft

bounded-latency-problems

July 2021

Supporting bounded latency at these speeds and fan-in/fan-out raises cost and feasibility challenges beyond those that had led to past IETF IntServ (GS) standards ([[RFC2210](#)], [[RFC2212](#)]) or more recent TSN bounded latency solutions.

Note that these high speed and scale requirements even cause challenges when DetNet bounded latency traffic is intended to be used for only a small percentage of the interfaces traffic.

1.2. Solution goal: Lightweight, per-hop, per-flow stateless transit hop forwarding

Both high-speed hardware and network architecture design (for reasons of simplicity and minimization of shared risk functions) do favor architectures that support a lightweight transit hop forwarding plane design that requires no forwarding plane or control plane operations whose scale support depends on the number of services/service-instances (e.g.: DetNet flows) offered, but at best only on the size of the network (e.g.: no per-flow, per-hop state).

1.3. Requirement: Support for existing stateless / steering solutions

There should be DetNet bounded latency options that work in conjunction with per-transit-hop stateless traffic forwarding such as through Shortest Path First (SPF) routing with IP/MPLS), engineered steering (e.g.: SR) and stateless replication, such as Bit Indexed Explicit Replication with/without Tree Engineering (BIER, BIER-TE).

1.4. Requirement: PCE to ingress/egress LSR only flow signaling

There should be DetNet bounded latency options that for the purpose of traffic engineering (including assurance of bounded latency across the network) only require per-flow Path Computation Engine (PCE) signaling to network ingress/egress router, but not to transit hop routers.

[1.5.](#) Requirement: Support for DiffServ QoS model on transit hops.

There should be DetNet bounded latency options that support the DiffServ QoS model instead of only the IntServ model.

[1.6.](#) Requirement: Low jitter bounded latency solutions.

There should be DetNet bounded latency options that together with the other requirements also provide a better than worst-case jitter for DetNet traffic.

[1.7.](#) Requirement: Dynamic, application signalled DetNet flows

The DetNet architecture should support signaling and forwarding that would make support for automatically application instantiated DetNet flows scalable and lightweight to operate.

[2.](#) Evolution of IP/MPLS network technologies and designs

To help readers understand especially the per-hop stateless requirement from above, the following sections summarizes the historical evolution of technologies and operational principles that the authors think are relevant to understand the requirements outlined above and asks to see supported in DetNet.

[2.1.](#) Guaranteed Service with RSVP

The original (first and only) IETF standardized packet forwarding layer standardized queuing option for bounded latency in the IETF is "Guaranteed Service", [[RFC2212](#)] (GS), see the DetNet bounded latency document, [[DNBL](#)] [section 6.5](#). At the time the RFC was published (1997), the standardized signaling was proposed to be RSVP [[RFC2205](#)], and the use of RSVP with GS was standardized in [[RFC2210](#)].

The function to support GS bounded latency in the forwarding plane is the per-flow reshaping on every forwarder hop along the path where GS packets of one flow may get delayed in the egress interface queue due to packets from other GS flows. In typical networks, this is every hop along the path.

Early (1990/2000) forwarders for which RSVP was implemented were using so-called "software" forwarding. This meant that the forwarding plane was implemented through a general purpose CPU without additional hardware support for QoS functions such as shaping or queuing. While these forwarders did support traffic flow shaping, GS was never implemented on them and their RSVP implementations did also not support (but ignored) the RSVP TSPEC/RSPEC signaling parameters used for bounded latency. Instead, RSVP implementations only supported the parameters for bandwidth reservation, which was henceforth called Call Admission Control (CAC).

In one instance, a software forwarder implementation with RSVP supported the Controlled Load (CL) service [[RFC2211](#)], which does not provide for bounded but instead for controlled latency. This service is achieved by creating a per-flow queue and applying weighted fair queuing (WFQ) with weights according to the reserved bandwidth of the flows (see [[RFC2211](#), section 11]). This functionality did not proliferate into later generations of routers because the execution cost of WFQ was too high for a multitude of flows and the scheduling

accuracy was too inaccurate in interrupt driven CPU software forwarding with higher speed interfaces (100Mbps...1Gbps).

[2.2.](#) Hardware forwarding and DiffServ

With the rise of forwarding planes with "acceleration" through ASIC based Forwarding Plane Elements (FPE) instead of general purpose CPUs and/or dedicated QoS hardware, the ability of forwarders to support shaping evolved to only be supported, if at all, on DiffServ (DS) boundary nodes, but not on DS interior nodes. This included both shaping as well as complex queuing such as WFQ.

The DS architecture, [[RFC2475](#)], was specifically targeted to enable the evolving, now common Service Provider network services architecture, in which "high-touch" service functions are only

performed on so-called Provider Edge (PE) routers, which as required are DS boundary nodes, whereas the hop-by-hop forwarding through so-called Provider (P) (core) routers is meant to utilize only a reduced set of forwarding functions, specifically excluding per-hop, per-flow QoS forwarding plane functions such as shaping or policing. DiffServ therefore allowed to build higher speed, lower cost forwarding plane P routers. It also enabled to build equally higher speed, lower costs PE routers by supporting boundary node functions only on (lower speed) customer facing interfaces/line cards, but not on core facing interfaces.

[2.3.](#) MPLS and RSVP-TE

With the advent of MPLS [[RFC3031](#)], RSVP was extended to support MPLS through the RSVP-TE [[RFC3209](#)] extensions. RSVP-TE manages p2p (later on also p2mp) MPLS Label Switched Paths (LSP), which when signaled through RSVP-TE are also called RSVP-TE tunnels. These can be seen as the equivalent of IP flows that RSVP manages for IP. RSVP-TE tunnels can support a variety of traffic engineering functions, but none of the implementations known to the authors ever implemented GS or CL services, specifically because hardware forwarding for service provider networks was not designed to support these QoS functions for P Label Switched Routers (LSR).

Because CL/GS were not targeted with RSVP-TE, the signaling extensions for Interior Gateway Protocols (IGP) required in the classical RSVP-TE reservation model (such as [[RFC8570](#)] for IS-IS) have no parameters to signal per-hop GS queuing latency or buffer capacity utilization. In result, the existing IGP signaling for RSVP-TE only supports RSVP-TE to perform bandwidth but not non-queuing path latency resource calculations and therefore no latency based traffic engineering.

[2.4.](#) Path Computation Engines (PCE)

Even though RSVP-TE implementations support only DiffServ (but not GS/CL) with respect to per-hop QoS functions, its traffic-steering (path selection) and signaling model introduced per-flow (per-tunnel) control plane and forwarding plane overhead onto every P-hop. Through the 200x's, this RSVP-TE overhead was seen as undesirable complexity and overhead by many service providers using it. There

was also a much larger number of service providers that desired some of the benefits provided by RSVP-TE, but who were not willing to commit to the complexity, costs and operational risk introduced into the network by complex per-flow signaling of RSVP-TE. The on-path, per-hop signaling of RSVP-TE for example introduced so much overhead, that reconvergence of RSVP-TE paths after a failure or recovery took as much as 20 minutes in networks with 10,000 or more RSVP-TE tunnels.

The design of RSVP-TE's (decentralized) on path signaling model specifically showed problematic under high resource utilization. In the original, decentralized RSVP-TE deployment model, ingress PE LSR would perform so-called Constrained Shortest Path Forwarding (CSPF) calculations to determine the shortest path with enough free resources for a new flow. Afterwards the ingress PE would signal the path via RSVP-TE. The IGP would signal to all ingress PE how many (bandwidth) resources were left on every link. Under high load, when multiple ingress PE were performing this process in parallel this would cause high load, churn and reservation collisions.

These problems of de-centralized RSVP-TE plus IGP signaling lead to the introduction of a so-called Path Computation Element (PCE) based architecture, in which the (competing and uncoordinated) traffic engineering computations on every de-centralized RSVP-TE ingress LSR were replaced by a centralized PCE function (or at least a coordinated PE function), which would send the calculated results back as a path object to the headend LSR, in result limiting the functions of RSVP-TE to the signaling of a steered traffic path through the network to establish the hop-by-hop LSP. The use of a PCE can likewise eliminate all the reservation state dependent signaling from the RSVP-TE IGP extensions, because all the reservation calculations solely need to happen only on the PCE. Nevertheless, the PCE does not eliminate the per-hop signaling overhead of RSVP-TE to establish LSPs and hence it did not eliminate for example the majority of the platform and convergence cost of RSVP-TE in the network, especially for the control plane of P nodes and could hence not resolve the concerns of service providers who had chosen not to adopt RSVP-TE.

The introduction of centralized PCE had obsoleted most of the reasons for RSVP: headends did not need to do path calculation, and P router did not need to manage the available and allocated bandwidth for TE tunnels. In most service-provider use-cases this left RSVP-TE only serving as a very complex solution to do traffic steering, and the PCE was doing the rest. This ultimately lead to the design of the Segment Routing [\[RFC8402\]](#) architecture, and its mapping to the MPLS forwarding plane, SR-MPLS [\[RFC8660\]](#). Later, a mapping to IPv6 was defined with SRv6 [\[RFC8986\]](#). SR relies on strict or loose hop-by-hop hop source routing information, contained in each packet header, therefore eliminating the need to set up per-path flow state via RSVP-TE, and allowed in conjunction with DiffServ for hop-by-hop QoS a complete per-hop, per-flow stateless forwarding solution, arguably therefore lightweight, easy to implement at high performance and scalable to large number of flows.

[2.6.](#) BIER

In the same way as SR eliminated the need for hop-by-hop traffic steering forwarding state from RSVP-TE in P-routers for unicast traffic, Bit Indexed Explicit Replication [\[RFC8279\]](#) (BIER) solves this problem for shortest path multicast replication state across P-routers, by replacing it with a BIER packet header [\[RFC8296\]](#) and therefore eliminating any per-application/flow, per-hop forwarding state for multicast in P-routers. BIER also removed the associated overhead of prior ingress replication solutions Service Providers where looking into to avoid the per-hop state.

Finally, BIER-TE [\[I-D.ietf-bier-te-arch\]](#) adds traffic steering with replication to the BIER architecture and calls this Tree Engineering. Likewise, this is without the need for per-hop/per-flow steering or replication state.

[2.7.](#) Summary

Service Provider networks have evolved especially in the past 25 years into an architecture, where high-speed, low-cost and high-reliability are based on designs that eliminate or reduce as much as possible any form of unnecessary control-plane and even more so per-flow, per-application plane complexity from P-routers/transit-nodes.

This has led to the development of the DiffServ QoS architecture that eliminated IntServ/per-flow QoS from P-routers, and later on to the evolution from MPLS/RSVP-TE to SR and BIER that eliminated per-flow/tunnel forwarding/steering and replication state from the same P-nodes.

Finally, early experience with Traffic Engineering churn under high load and today's requirements for often NP-complete optimization lead to an architectural preference for off-path/centralized model for TE calculations via PCE to also free P-routers from signaling complexity and perform dynamic/service-dependent signaling only to PE-routers.

[3.](#) Additional current considerations

The following subsections look at further into the background for why per-hop, per-flow state can be problematic and discuss problems beyond this core issue.

[3.1.](#) Impact of application based state in networks

RSVP-TE was (and is) solely used for services where the operator of a domain explicitly provisions RSVP-TE tunnels across its domain (for example using a PCE) and can therefore fairly easily know the worst-case scaling impact. For example the number of tunnels does not arise as a chance value arising through dynamic subscriber action, and the number of tunnels in the network is primarily impacted by topological changes and the (over time relatively rare) occurrences of additional services and/or service instances being provisioned. For RSVP-TE there was never (to the knowledge of the authors) an end-to-end application layer interface such as there was for RSVP over IP, for example as supported by earlier versions of Microsoft Windows QoS enabled IP sockets.

When per-flow operations including per-hop signaling or even worse per-hop forwarding plane or QoS state is not a result of well-controlled provisioning or well-plannable/predictable failure behavior but instead driven by applications not under the control of network operators, the per-hop state requirements can become much more an operational and cost problem, because of its unpredictability.

[3.2.](#) Experience from IP multicast

The widest experience with dynamic, application based signaling in Service Provider networks likely exist for IP multicast, where creation of per-hop forwarding/replication state is triggered by applications not under the control of network operations but by customer managed applications/application-instances. Managing the amount of state and the control plane load on P-routers was and is one of the major concerns when operationalizing IP Multicast services in SPs.

Service Provider L2-VPN and L3-VPN services can offer IP Multicast via architectures such as [[RFC6513](#)] that attempt to solve/reduce the

problem of customer application driven, per-multicast application in a variety of ways, but they all come with their own problems:

- o In ingress-replication, the ingress-PE sends a separate unicast copy to every egress-PE. This creates significant excess traffic on links close to the ingress-PE and potentially higher-cost ingress-PE attachment speeds.
- o In L3VPN aggregates-trees, the traffic for multiple trees is sent across a common tree reaching the superset of all egress-PE of all included trees. This reduces the number of trees from one per-customer application to a lower number of aggregates this, but it creates potentially significant excess traffic towards egress-PE that do not need all the aggregated traffic and may even result in a requirement for access core access link speeds for those egress routers.

Finally, the per P-router stateless BIER solution solved these issues. It does not require any per P-router, per tree state creation, and achieves a 256x better traffic efficiency than ingress replication (with 256 long BIER bit strings).

[3.3](#). Service Provider and Private MPLS Networks

With DetNet services being targeted primarily for so-called private networks such as (but not limited to) those for industrial, theme parks, power supply systems, road, river, airport and train transportation networks, it is important to understand how concerns for SP networks will apply to such private networks:

While the aforementioned evolution of MPLS networks focused on large-scale service provider networks, the very same architectural evolution is or will also happen in any private MPLS networks in the same way as the DiffServ architecture equally became the only widely adopted QoS architecture in any larger scale (campus or beyond) private networks.

While some of the scaling, cost, performance and reliability issues mentioned above for service providers may not equally apply to

smaller scale private networks, past experience has shown that that it is unlikely for a critical mass for different solutions to develop across a large variety of vertical private type of networks. For this reason, in the past any larger scale enterprise networks have preferred to adopt solutions that had proven themselves through SP deployments and that were based on cross-vendor IETF based architecture principles and widely, interoperable vendor implementations.

Another reason for private network operators looking for service provider calls designs is that it also simplifies potential service provider based management of the network and/or outsourcing of the network to a service provider. This was seen often when large enterprises that had to support multi-tenants evolved from ad-hoc network virtualization solutions (such as VRF-lite) over to BGP/MPLS-VPN designs and later outsourced those very networks.

In that same line of future proofing, networking technologies first developed for enterprises would also be picked up and reused in Service Provider networks as long as they would fit. IP Multicast for example had (since about 1996) ca. 10 years of deployment for business critical enterprise use cases (such as financial market data distribution), before it was adopted widely for IPTV in service providers.

[3.4.](#) Mission-specific vs. shared infrastructures

Whereas the previous section points to the practice and benefits to share technologies between private and SP network, this section highlights one core additional requirement of SP networks not found in most private networks from which pre-DetNet deterministic service requirements will likely originate.

In architectural terms, the desire and need to minimize or avoid per-application/flow forwarding/control-plane state and per-hop control plane interactions (be it through on-path signaling or direct PCE to P-router signaling) is not primarily a matter of SP/private networks or not even of size, but foremost a matter of whether or not the network itself is seen as the (a) communications fabric of a large distributed application or (b) as an independently running shared infrastructure across a potentially wide variety of application/

services with diverging requirements.

(a) is the dominant view of the network specifically from many (single) mission specific networks such as many industrial networks and even non-public High Performance Compute (HPC) center architectures. In either of these case, it is a single architectural entity that can control both network infrastructure and application to build a mission optimized compound.

For example, switches in HPC Data Centers had traditionally very shallow interface packet buffering for cost reasons, resulting in inferior performance under peak load with predominant older TCP congestion control stacks. Instead of using better, more expensive switches, it was easier to improve application device TCP stacks, leading for example to BBR TCP. While this is very much in line with the desired Internet architecture that is putting a significant

Eckert & Bryant

Expires January 13, 2022

[Page 11]

Internet-Draft

bounded-latency-problems

July 2021

responsibility onto transport layer protocols in hosts (not limited to TCP) to behave "fair" or "ideal", the reality even in many private missions centric networks such as manufacturing plant is different. Dealing with misbehaving user devices or applications is one of the main challenge. In the example, that is the case when a DC is offering public cloud services, where TCP stacks can not be controlled, and hence deeper buffers and/or better AQM are a core requirement.

In general: In networks following the (b) shared infrastructure design principle, any resource that needs to be shared across different services or even service instances becomes a potential three party reliability and costing issue between the provider running the network and the two (or more) parties whose services utilize the common resource. Minimizing the total amount of complex, failure-prone and hard to quantify in a cost-effective manner shared resources is thus at the base of any shared infrastructure network design.

This again points to the model, where all network control can happen on the edge, and due to the absence of per-hop, per-flow state there simply is no shared flow state table that needs to be managed across multiple different services/subscribers.

[3.5.](#) PTP and challenges with clock synchronization

Some bounded latency solution require accurate clock synchronization across network nodes performing the bounded latency algorithm. The most commonly used (family of) protocol(s) for this is the Precision Time Protocol (PTP), standardized in IEEE1588 and various market specific profiles thereof.

PTP can achieve long-term Maximum Time Interval Errors (MTIE) of as little as 10th of nsec. MTIE is the maximum time difference between the clocks of two PTP nodes measured over long period of time.

Implementing PTP in devices comes at a range of design requirements. At high degree of accuracy, PTP requires accordingly accurate local oscillators that includes hardware such as regulated heating to operate under constant temperature. It includes accurate distribution of clock across all components of the system, which can be especially challenging in modular, large-scale devices, and accurate insertion and retrieval of timestamp field into packet headers.

While PTP is becoming more and more widely available, consistent support of high accuracy across all target type of switches and routers in wide area networks cannot be taken for granted to be a

feasible new requirement raised for DetNet when it did not exist in before. Today, PTP is often found in mobile network fronthauls, but not their backhauls or any other broadband aggregation, distribution or core networks. This is because there is, as of today, no strong business case requirement for PTP at high precision in those networks, whereas technologies such as eCPRI raise such requirements against mobile fronthauls. Instead, those other networks most often resort to at best msec accuracy NTP protocol deployments which is typically sufficient for control-plane and operational event tracing as its main, accuracy defining use-case.

The larger the network and more multi-vendor varied the deployed equipment is, the higher will also be the operational cost of maintaining and controlling the accuracy of a PTP service. This primarily has been cited in the past as a reason to not deploy PTP even if hardware was supporting it. This operational challenge will especially apply when PTP support may be required for only a small percentage of traffic in a high speed wide area network. The revenue

from the service needs to cover the operational cost incurred by its exclusive components (hardware, software and operations).

[3.6.](#) Jitter - in-time versus on-time

This section discusses how low-jitter bounded latency applications can be highly beneficial for DetNet applications.

Depending on the bounded latency algorithm, the jitter experienced by packets varies based on the amount of competing traffic. In algorithms and their resulting end-to-end service which this memo calls "in-time", such as GS and [\[TSN-ATS\]](#), the experienced latency in the absence of any competing traffic is zero, and in the presence of the maximum amount of permissible competing traffic, latency is the maximum, guaranteed bounded latency. In result, the jitter provided by these algorithms is the highest possible.

In algorithms and their resulting end-to-end service which this memo calls "on-time", the experienced latency is completely or most significantly independent of the amount of competing traffic and the jitter therefore null or minimal. In these algorithms, the network buffers packets when they are earlier than guaranteed, whereas in-time algorithms deliver packets (almost) as fast as possible.

This memo argues that on-time queuing algorithms provide an additional value-add over in-time algorithms, especially for use in metropolitan or wide-area networks. Whatever algorithm is used, the receiving application only has a guarantee for the maximum bounded latency, and the real (shorter) latency of any received packet is no indication for the latency of the next packet. Instead, the receiver

application has to be prepared for each and any future packet to arrive with the worst possible, e.g.: the bounded latency.

The majority of applications require some higher layer function synchronously to the sender application: Rendering of audio/video and other media information needs to happen at the same frequency or event intervals at which the media was encoded. When these applications receive packets earlier than the time at which they can be processed (which is equal or close to the bounded latency), these applications buffer media in a so-called playout buffer and release them only at that target time. Likewise, remote control loops

including industrial Programmable Logic Controller (PLC) loops or remote controlling of robots or cars is typically based on synchronous operations. In these applications, early packets are also delayed to then be processed "synchronously" later.

In all cases, where applications need to buffer (or otherwise remember) received data when it is too early, in-time queueing latency raises the challenge to application developers to be able to predict the networks worst possible jitter, and this can be particularly challenging for embedded, if not constrained receiver devices with minimum memory to buffer/remember. When these devices are designed against one particular type of network with well-known low jitter, then they will not necessarily operate correctly in networks with larger jitter. And in metropolitan and WAN networks, jitter with in-time services can be highly variable based on its design and the relative location of the communicating nodes in the topology (see [Section 5](#) for an example network design).

One example of such issues was encountered when digital TV receivers (Set Top Boxes, STB) designed for (mostly synchronous) digital cable transmission where evolved to become IPTV STB, but the playout buffer of < 50 msec was not sufficient to compensate for a > 50 msec jitter experienced in IP metropolitan networks.

Note that this section does not claim that all applications will benefit from on-time service, nor that no application would benefit more from in-time service than from on-time service. Nevertheless, the authors are not aware of instances of [\[RFC8578\]](#) application for whom in-time service would be more beneficial than on-time service. Of course, this comparison is only about the benefit to the application and other factors such as the cost/scale of the service for the network itself have also to be taken into account.

[4.](#) Challenges for high-speed packet forwarding hardware

The problems of cost and operational feasibility in shared-infrastructure networks specifically applies to scaling of hardware

resources such as per-application-flow forwarding or QoS state in high-speed network routers: Even if the business case makes it clear that only e.g. 1 Gbps worth of traffic may require this advanced state (such as multicast replication or per-flow shaping for bounded latency), it will be more expensive to build this functionality into a 100 Gbps transit switch/router than into a 1 Gbps switch/router. This too is based on experience from migrating services of low-speed mission specific networks, such as IP multicast onto high speed, shared-infrastructure service provider networks.

The reason for this higher cost at higher speed is that the 1 Gbps worth of "advanced" traffic still has to be built into 100 times faster hardware and each of the "advanced" packets forwarded would need to be replicated/shaped 100 times faster.

This packet processing issue may look like it applies equally to both per-hop, per-flow stateful based forwarding as well as solely in-packet based mechanisms, in practice, per-flow state may require a lot more high-speed memory access because of the need to access an entry from a state table. In most cases, this table space can only be made to work at line rate packet processing when it is on-chip, hence it is not only most expensive, it is also crucial to scale right. And as the 1 vs. 100 Gbps example above showed, it is very hard to come by an appropriate scale smaller than "would work for 100% of traffic" - because network operators providing shared infrastructure networks really do not want to be responsible for predicting how individual services may grow in adoption by making a specific hardware selection that constrains any such growth.

Last, but not least, on-chip high-speed state tables become even more expensive when they do not only have to be read only, but also when they have to be written at line rate and even worse, when they have to operate for line-rate speed read/write/read control loops:

The main issue with scaling state in hardware routers is that designs will be hesitant to work against unclear growth predictions. Even if at some point in time only 1 Gbps of DetNet traffic was expected to be required on a 100 Gbps platform, hardware designers will be much more likely to scale against the worst (best) case service growth expectation so that customers will not feel that they would buy into a product that becomes obsolete under success.

Whereas steering state, such as MPLS label entries can easily scale to hundreds of thousands, the same is not clear about shapers or

interleaved regulators. They are more challenging because they require fast (on-chip) read-write memory for the state variables, especially when forwarding is parallelized across multiple execution unit. This does incur additional complexity to split up the state and its packets across multiple execution units and/or to provide consistent cross-execution units shared read/writeable memory.

Even only writeable (but not cross-execution units then also readable) memory has traditionally been a sparse resource the faster the forwarding engines are. This can be seen from (often very limited) scale of packet monitoring state such as for IPfix.

But the main issue of per-hop, per-flow forwarding state that could be quite dynamic because it might be triggered by applications is the control plane to forwarding-plane-state interactions. Updating hardware forwarding engine state tables is often one of the key performance limits of routers. Adding significant additional state with likely ongoing changes is easily seen as a big contributor to churn in the control plane and likely reason for stability and reduced peak performance under key events such as reconvergence of all or large parts of IGP or BGP routing tables.

[5.](#) A reference network design

The following picture shows an example, worst-case network topology of interest (in the opinion of the authors) for bounded latency considerations. This section does not claim that greenfield rollouts may or want to use all aspects of this topology. What this memo does claim is that many existing brownfield networks, especially large metropolitan areas show all or many of these aspects, and that it would be prudent for bounded latency network technologies to support networks like these so as to not create new constraints against network designers by only supporting physical network topologies optimized for a particular type of service (bounded latency).

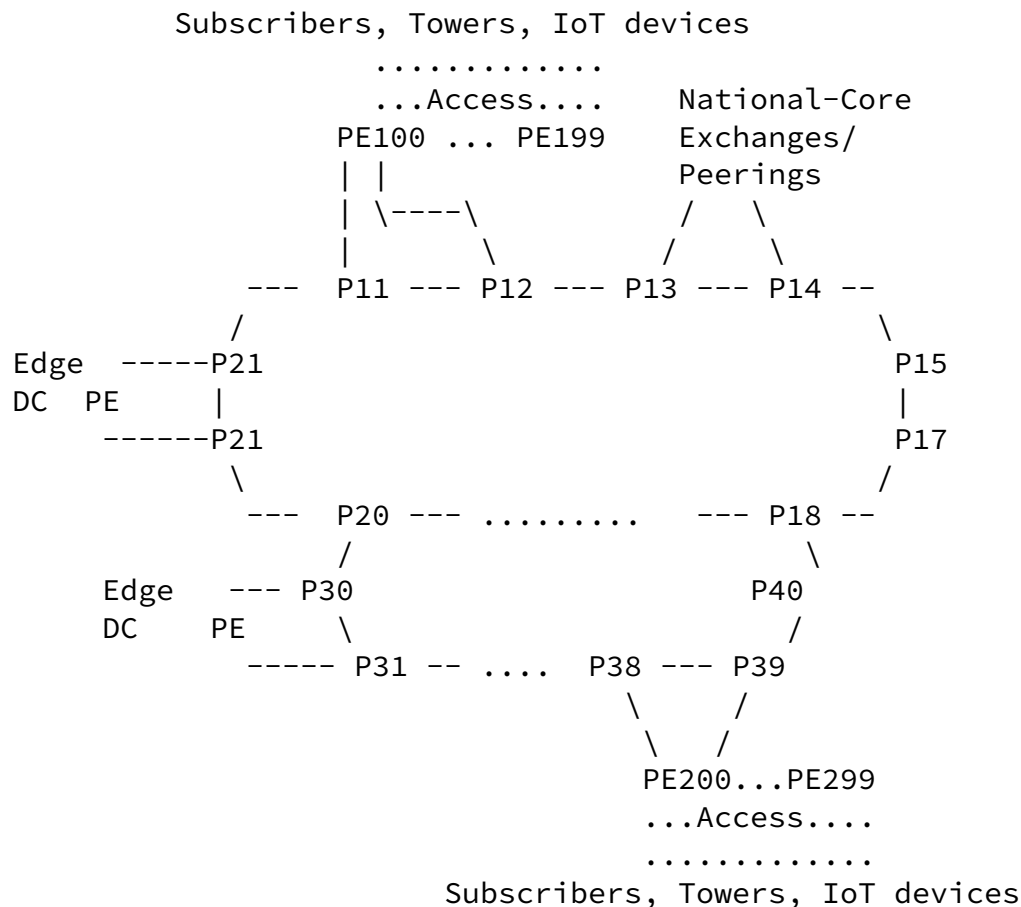


Figure 1: Reference Network Topology

An example metropolitan scale network as shown in Figure 1 may consist of one or more rings of forwarders. A ring provides the minimum cost $n+1$ redundancy between the ring nodes, especially when, as is common in metropolitan networks, new fibre cannot cost-effectively be put into new optimum trenches, but existing fibre and/or trenches have to be used. This is specifically true when the area includes not dense populated suburban areas (higher cost per subscriber and mile for rollouts).

Multiple, so-called subtended rings typically occur when existing networks are expanded into new areas: A new ring is simply connected at two most economic points into the existing infrastructure. Likewise, such a topology may become more complicated over time by

addition of capacity, which resulting from TE planning calculations may not follow any of the pre-existing ring paths.

Edge Data-Center (DC), connections to Exchanges/Peerings or national cores of the provider itself, as well as all subscribers including Mobile Network Towers, and IoT devices connect to these ring directly via PE edge-forwarders and (more often) via additional CE type devices. P nodes may also double as PE nodes.

In densely populated regions, P, or PE nodes may have a high number of attached devices, shown in the picture with the example of 100 PE forwarder connecting to a single P forwarder (or rather two P for redundancy and therefore support of PREOF).

In summary, the following aspects of these networks are relevant for bounded latency:

- o Link speeds today are at least 100 Gbps and will be Tbps in the near future. Even if only a small percentage of that traffic has to support bounded latency, the queuing mechanism need to support these high-speed interfaces.
- o Fan-in/out at PE or P nodes may be (worst case) in the order of hundred(s) of incoming interfaces. Bounded latency mechanisms whose number of queues depend on the number ($\#I$) of interfaces in a more than linear fashion, such as ($\#I^2$) in the case of [\[TSN-ATS\]](#), may introduce significant challenges for cost-effective hardware.
- o Through the advent of decentralized edge Data Center and peerings between different operators and content providers, traffic flows of interest will not solely be between one central site from/to subscribers hub&spoke. Instead arbitrary, traffic engineered paths across the topology between any two edges need to be supportable in scale with the bounded latency queuing mechanism.
- o The total number of edge ($\#E$) nodes (PE or CE) for a bounded latency service can easily be in the thousands. Aggregation of bounded latency flows on the order of ($\#E^2$), which is the best option in per-hop, per-flow solutions such as [\[TSN-ATS\]](#), is likely insufficient to significantly reduce the number of flows that need to be managed across P nodes in such bounded latency queuing

mechanisms.

- o The total number of P nodes may be in the hundreds and bounded latency flows in the tenths of thousands. It should also be expected that such flows are not necessarily long-term static but may need to be provisionable in the time-scale order of for example telephone calls (such as flows supporting remote control of devices or operations). Bounded latency solutions that require per-flow, per-node state maintenance on the P nodes themselves may therefore be undesirable from a network operational/complexity/reliability perspective, but also from a hardware engineering cost perspective, especially with respect to the control plane cost of dynamically setting up per-flow bounded latency for flow whenever there is a new flow or all of them

Eckert & Bryant

Expires January 13, 2022

[Page 18]

Internet-Draft

bounded-latency-problems

July 2021

whenever there are topology or load changes that make rerouting desirable.

Beyond queuing concerns, path selection too specifically for deterministic services is a challenge in these networks:

- o Path lengths may be significantly longer than e.g. 3 hops. In large metropolitan networks, they can reach 20 or more hops. Speed of light end-to-end in these networks will be in the order of low number of msec. End-to-end queuing latency can be in the same range, if not higher.
- o To avoid undesirable re-routing under failure when PREOF and engineered disjoint paths are used, traffic steering needs to support efficiently supportable hop-by-hop traffic steering. In networks designed for source-routing (e.: SR routing), efficiently encoded strict-hop-by-hop steering for as much as those (e.g.: 20) hops may be desirable to support.

[6.](#) Standardized Bounded Latency algorithms

[DNBL] gives an overview of the math for the most well-known existing deterministic bounded latency algorithms/solutions. This section reviews the relevant currently standardized algorithms from the perspective of the above listed problems for high-speed, high-scale, shared services infrastructures and to provide additional background

about them.

[6.1.](#) Guaranteed Service (GS)

GS is described in section 6.5 of [\[DNBL\]](#). [Section 2.1](#) describes its historical evolution and challenges. We skip further detailing of its issues here to concentrate on IEEE Time Synchronous Networking - Asynchronous Traffic Shaping [\[TSN-ATS\]](#), which in general is seen as superior to GS for high speed hardware implementation. All the concerns described in the TSN-ATS section apply equally or even more to GS.

[6.2.](#) TSN Asynchronous Traffic Shaping (TSN-ATS)

Section 6.4 of [\[DNBL\]](#) describes the bounded latency used for TSN Asynchronous Traffic Shaping [\[TSN-ATS\]](#). Like GS, this bounded latency solution also relies on per-flow shaper state, except that it uses optimized shapers called "Interleaved Regulator" as explained in section 4.2.1 of [\[DNBL\]](#).

The concept and simplification in interleaved regulators over traditional shapers and the concept of interleaved regulators is a

resulting from mathematical work done in the last 10 years starting with [\[UBS\]](#).

In a system with e.g. $N=10,000$ flows each with a shaper, the forwarder needs to have 10,000 shapers each of which would need to calculate the earliest feasible send-time of the first queued packet of the flow and all these send-times would need to be compared by a scheduler picking the absolute first packet to send. Of course it is unlikely that the router would have to queue at least one packet for all queues at any point in time, but the complexity to implement the scheduler scales with N .

With interleaved regulators, there is still the per-flow state required to hold each flows traffic parameters and its next-packet earliest departure time, but instead of requiring a scheduler to compare N entries, packets are queued into one out of $(\#IIF, \#PRIO)$ FIFO queues, one queue for all the packets arriving from the same Incoming InterFace (IIF) and targeted the same worst-case queuing latency/PRIOrity (PRIO) on this hop. The shaper now only needs to

calculate the earliest departure time of the head of each of these $M = \#IIF * \#PRIO$ queues and the complexity of a scheduler to select the first packet across those interleave regulators is therefore reduced by a factor of $O(N/M)$.

Unfortunately, while industrial ethernet switches today often have no more than 24 IIF, aggregation routers in metropolitan networks may have thousands of IIF, so the benefit of interleaved regulators over per-flow shaper will likely be much higher in classical TSN environments than it would be for example likely DetNet target routers in metropolitan networks.

In addition, the aforementioned core problems for shapers ([Section 4](#)), namely control plane, read/write/read cycle access and scale equally apply to interleaved regulators, so the main optimization benefits of interleaved regulators is for the original targets of [\[UBS\]](#) / [\[TSN-ATS\]](#): low-speed (1..10Gbps switches) with limited number of interfaces – but to a much lower degree for likely important type of DetNet deployments.

[6.3.](#) Cyclic Queuing and Forwarding (CQF)

TSN Cyclic Queuing and Forwarding as described in [\[DNBL\]](#), [section 6.6](#), is a per-flow, per-transit-hop stateless forwarding mechanism, which solves the concerns with per-hop, per-flow state issues described earlier in this memo. It also provides an on-time service in which the per-hop and end-to-end jitter is very small, namely in the order of a cycle time.

[\[CQF\]](#) operates by forwarders sending packets in periodic cycles. These cycles are derived from clock synchronization: The start of each cycle (and by implication the end of the prior cycle) are simply periodically increasing clock timestamps that have to be synchronized across adjacent forwarders, usually via PTP. This method to operate cycles allows [\[CQF\]](#) to operate without additional [\[CQF\]](#) data packet headers, but it is also the reason for the two issues of [\[CQF\]](#), and both relate to the so-called dead time (DT).

For the receiving node to correctly associate a [\[CQF\]](#) packet to the same cycle as the sending node, the last bit of the last packet in the cycle on the sending node needs to be received by the receiving

node before the cycle ends.

[DNBL] explains that DT is the sum of latencies 1,2,3,4 as of [DNBL] Figure 1, but that is missing the MTIE between the forwarders: If a cycle is for example 10 usec, and the PTP MTIE is 1 usec, then only 9 usec of the cycle could be used (without even yet considering the other factors contributing to MTIE). If MTIE is not taken into account, a packet might arrive in time from the perspective of the sending forwarder, but not in the perspective of the 1 usec earlier receiving node.

In practice, MTIE should be equal or lower than 1% of the cycle time. When forwarders and links increase in speed, cycle times could become proportionally smaller to reduce per-hop cycle time latency. When this is done, MTIE needs to equally become smaller, raising the costs of the solution. Therefore, [COF] has a challenge with higher speed networks.

The second and even more important problem is that DT includes the link latency (2 in [DNBL], Figure 1). With a speed of light in fibre of 200,000 Km, link latency is 10 usec for 2 Km. This makes [COF] very problematic and limited in metropolitan and wide-area networks. If the longest link of a network was 10 Km, this would cause a DT on that link of 50 usec and with a cycle time of 100 usec, only 50% bandwidth could be used for cycle-time (bounded latency) traffic (excluding all other DT factors).

When links are subject to thermal expansion also known as sag on hanging wires, such as broadband copper wires (Cable Networks), their length can also change by as much as 20% between noon and night temperatures, which without changes in the design has to be taken into account as part of DT.

In conclusion, [COF] solves many of the problems discussed in this memo, but it's reliance on timestamp synchronized cycles may pose undesirable challenges with the required accuracy of PTP in high

speed network and especially limits [COF] ability to support wider-scale networks due to DT.

7. Candidate solution directions

As this memo outlines, per-hop, per-flow stateless forwarding is the one core requirement for to support Gbps speed metropolitan or wide-area networks.

This section gives an overview and evaluation from the perspective of the authors of this memo of currently known non-standardized proposals for per-hop-stateless forwarding with the explicit goal and/or possibility of bounded latency forwarding and in relationship. to the concerns and desires described in the previous sections.

7.1. Packet tagging based CQF

To overcome the challenges outlined in [Section 6.3](#), [\[I-D.qiang-DetNet-large-scale-DetNet\]](#) and [\[I-D.dang-queuing-with-multiple-cyclic-buffers\]](#) (tagged-CQF) propose a modified [\[CQF\]](#) mechanism in which timestamp based cycle indication of [\[CQF\]](#) is replaced by indicating the senders cycle in an appropriate packet header field, so that the receiver can accordingly map the received packet to the right local cycle.

This approach completely eliminates the link-latency as a factor impacting the effectiveness of the mechanism, because in this approach, the link latency does not impact the DT. Instead the link latency is used to calculate which cycle from the sender needs to be mapped to which cycle on the receiver, and this is programmed during setup of links into the receiving routers cycle mapping table.

Depending on the number of cycles configured, it is also possible to compensate for variability in the link-latency and higher MTIE (picture TBD). If one more cycle is used for example, this would allow for MTIE to be the order of one cycle time as opposed to a likely target of 1% of cycle time as in [\[CQF\]](#), reducing the required PTP clock accuracy by a factor of 100. This possible reduction in required accuracy of operations by appropriate configuration does not only cover PTP but also extends into any forwarding operation within the nodes, e.g.: it could also reduce the cost of implementation of forwarding hardware at higher speeds accordingly.

In MPLS networks, packet tagged CQF with a small number of cycle tags (such as 3 or 4) could easily be realized and standardized by relying on E-LSP where 3 or 4 EXP code points would be used to indicate the cycle value. Given how such deterministic bounded latency traffic is not subject to congestion control, it also does not require

additional ECN EXP code points, so those would be available for e.g.: best-effort traffic that should use the same E-LSP.

7.2. Packet tagging based CQF with SR

[I-D.chen-DetNet-sr-based-bounded-latency] applies the tagged-CQF mechanisms to Segment Routing (SR) by proposing SR style header elements to indicate the per-segment/hop cycle. This eliminates the need to set up on every hop a cycle mapping table.

It is unclear to the authors of this memo how big a saving this is given how the PCE would need to update all the ingress router per-flow configurations where header imposition happens when links change, whereas the mapping table approach would require only localized changes on the affected routers.

7.3. Per-hop latency indications for Segment Routing

[I-D.stein-srtsn] describes a mechanism in which a source-routed header in the spirit of a Segment Routing (SR) header can be used to enable a per-transit-hop per-flow stateless latency control. For every hop, a maximum latency is specified. The draft outlines a control plane which similarly to packet tagging based CQF or [TSN-ATS] would put the work of admitting flows, determining their paths and admitting their resources along those paths to some form of PCE/SDN-Controller.

The basic principle of forwarding in this proposal is to put received packets into a priority heap and schedule them in order of their urgency (shortest latency) for this hop.

The draft explicitly does not prescribe specific algorithms on the forwarders to take the indicated latency for the hop into account in a way that the controller can calculate the resource availability, such as specific queuing or scheduling algorithms.

It is not entirely clear to the authors of this memo, if the sole indication of such deadline latencies is sufficient to completely eliminate per-transit-hop, per-flow state and still achieve deterministic latency because of the [UBS] work. Consider that the packets latency for a hop could be used to derive a priority queue on the hop relative to other packets with higher or lower latency for this hop,

As was shown in the research work leading up to [TSN-ATS], the priority queuing on each hop alone is not sufficient to achieve a simple, solely per-hop calculated latency bound under high load because of the problem of multi-hop burst aggregation and the

Internet-Draft

bounded-latency-problems

July 2021

resulting hard to calculate incurred upper latency bound. To overcome that calculation issue, shapers or as in [\[TSN-ATS\]](#) their optimization, interleaved regulators, are used in [\[TSN-ATS\]](#) and GS. Shapers/interleaved requires to maintain across packets from the same flow per-flow state.

Nevertheless, appropriate mathematical models for SDN controllers may be possible to develop deterministic per-hop forwarding models relying not only on the per-hop indicated latency but also on additional constraints such as limited number of hops or sufficiently low degrees of maximum admitted amount of traffic. Or else this may be used for to be developed latency models that are not 100% deterministic, but close enough in probability such that the amount of late packets would be in the same order as otherwise unavoidable problems such as BER based packet loss.

To that end, the author of [\[I-D.stein-srtsn\]](#) has conducted simulations of the proposed mechanism, contrasting it with other mechanisms. These results, which will be published elsewhere, show that this mechanism excels in cases with high load and a small number of flows with tight budgets. However, some small percentage of packets will miss their end-to-end latency bounds, and must be treated as lost packets.

Depending on the algorithms chosen, solutions may or may not rely on strong, weak, or no clock synchronization across nodes.

[7.4.](#) Latency Based Forwarding

"High-Precision Latency Forwarding over Packet-Programmable Networks", NOMS 2020 conference [\[LBF\]](#) describes a framework for per-transit-hop, per-flow stateless forwarding based on three packet parameters: The minimum and maximum desired end-to-end latency, set by the sender and not changed by the network, and the experienced latency updated by every hop. Routers supporting this LBF mechanism do also extend their routing (e.g.: IGP) to be able to calculate the non-queueing latency towards the destination. Based on the in-packet parameters and the future latency prediction are used to prioritize packets in queuing including giving them higher priority when they are late due to prior hop incurred latency, or delaying them when they are too early.

LBF was started as a more fundamental research into how application experience could be improved when they are allowed to indicate such differential min/max latency Service Level Objectives (SLO). Benefits include the ability to compensate for prior hop incurred queuing latency, but also to automatically prioritize packets on a

single hop based on their future path length, all without the need for any explicit admission control.

The LBF algorithm is completely without need for clock synchronization across nodes. Instead, it assumes mechanisms to know or learn link latency and the remaining latencies (as defined in the DetNet architecture) can be calculated locally (e.g.: latency through a forwarder).

The authors have not yet tried to define a mathematical model that would allow to derive completely deterministic behavior for this original LBF algorithm in conjunction with a PCE/SDN controller. Due to the absence of per-flow (shaper/interleaved-regulator), the authors believe that deterministic solutions would as outlined above for SRTSN ([Section 7.3](#)) likely only be possible under additional assumed constraints.

[8.](#) Conclusions

Bounded Latency for DetNet have been designed by trying to adopt solutions developed either several decades ago (GS) or recently for limited scope and scale L2 networks [[TSN-ATS](#)].

To allow DetNet solutions to explore opportunities in larger speed & scale shared network infrastructures, both private and service provider networks, it is highly desirable for DetNet WG (and/or other IETF WGs claiming responsibility in conjunction with DetNet as the driver) to explore the opportunities to standardize additional, and in the opinion of the authors better per-hop forwarding models in support of (near) deterministic bounded latency by mean of standardizing per-flow stateless/"DiffServ" style per-hop forwarding behavior (PHB) with appropriate network packet header parameters.

[9.](#) Security Considerations

This document has no security considerations (yet?).

[10.](#) IANA Considerations

This document has no IANA considerations.

[11.](#) Acknowledgements

Thanks for Yaakov Stein for reviewing and proposing text for [Section 7.3](#).

Eckert & Bryant

Expires January 13, 2022

[Page 25]

Internet-Draft

bounded-latency-problems

July 2021

[12.](#) Informative References

- [CQF] IEEE Time-Sensitive Networking (TSN) Task Group., "IEEE Std 802.1Qch-2017: IEEE Standard for Local and Metropolitan Area Networks -- Bridges and Bridged Networks -- Amendment 29: Cyclic Queuing and Forwarding", 2017.
- [DNBL] Finn, N., Boudec, J. L., Mohammadpour, E., Zhang, J., Varga, B., and J. Farkas, "DetNet Bounded Latency", [draft-ietf-detnet-bounded-latency-06](#) (work in progress), May 2021.
- [I-D.chen-DetNet-sr-based-bounded-latency] Chen, M., Geng, X., and Z. Li, "Segment Routing (SR) Based Bounded Latency", [draft-chen-DetNet-sr-based-bounded-latency-01](#) (work in progress), May 2019.
- [I-D.dang-queuing-with-multiple-cyclic-buffers] Liu, B. and J. Dang, "A Queuing Mechanism with Multiple Cyclic Buffers", [draft-dang-queuing-with-multiple-cyclic-buffers-00](#) (work in progress), February 2021.
- [I-D.ietf-bier-te-arch] Eckert, T., Cauchie, G., and M. Menth, "Tree Engineering for Bit Index Explicit Replication (BIER-TE)", [draft-ietf-bier-te-arch-10](#) (work in progress), July 2021.
- [I-D.qiang-DetNet-large-scale-DetNet]

Qiang, L., Geng, X., Liu, B., Eckert, T., Geng, L., and G. Li, "Large-Scale Deterministic IP Network", [draft-qiang-DetNet-large-scale-DetNet-05](#) (work in progress), September 2019.

[I-D.stein-srtsn]

Stein, Y. (., "Segment Routed Time Sensitive Networking", [draft-stein-srtsn-00](#) (work in progress), February 2021.

[LBF]

Clemm, A. and T. Eckert, "High-Precision Latency Forwarding over Packet-Programmable Networks", IEEE 2020 IEEE/IFIP Network Operations and Management Symposium (NOMS 2020), doi 10.1109/NOMS47738.2020.9110431, April 2020.

[RFC2205]

Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", [RFC 2205](#), DOI 10.17487/RFC2205, September 1997, <<https://www.rfc-editor.org/info/rfc2205>>.

Eckert & Bryant

Expires January 13, 2022

[Page 26]

Internet-Draft

bounded-latency-problems

July 2021

[RFC2210]

Wroclawski, J., "The Use of RSVP with IETF Integrated Services", [RFC 2210](#), DOI 10.17487/RFC2210, September 1997, <<https://www.rfc-editor.org/info/rfc2210>>.

[RFC2211]

Wroclawski, J., "Specification of the Controlled-Load Network Element Service", [RFC 2211](#), DOI 10.17487/RFC2211, September 1997, <<https://www.rfc-editor.org/info/rfc2211>>.

[RFC2212]

Shenker, S., Partridge, C., and R. Guerin, "Specification of Guaranteed Quality of Service", [RFC 2212](#), DOI 10.17487/RFC2212, September 1997, <<https://www.rfc-editor.org/info/rfc2212>>.

[RFC2475]

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", [RFC 2475](#), DOI 10.17487/RFC2475, December 1998, <<https://www.rfc-editor.org/info/rfc2475>>.

[RFC3031]

Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", [RFC 3031](#), DOI 10.17487/RFC3031, January 2001,

<<https://www.rfc-editor.org/info/rfc3031>>.

- [RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", [RFC 3209](#), DOI 10.17487/RFC3209, December 2001, <<https://www.rfc-editor.org/info/rfc3209>>.
- [RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/BGP IP VPNs", [RFC 6513](#), DOI 10.17487/RFC6513, February 2012, <<https://www.rfc-editor.org/info/rfc6513>>.
- [RFC8279] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Przygienda, T., and S. Aldrin, "Multicast Using Bit Index Explicit Replication (BIER)", [RFC 8279](#), DOI 10.17487/RFC8279, November 2017, <<https://www.rfc-editor.org/info/rfc8279>>.
- [RFC8296] Wijnands, IJ., Ed., Rosen, E., Ed., Dolganow, A., Tantsura, J., Aldrin, S., and I. Meilik, "Encapsulation for Bit Index Explicit Replication (BIER) in MPLS and Non-MPLS Networks", [RFC 8296](#), DOI 10.17487/RFC8296, January 2018, <<https://www.rfc-editor.org/info/rfc8296>>.
- [RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", [RFC 8402](#), DOI 10.17487/RFC8402, July 2018, <<https://www.rfc-editor.org/info/rfc8402>>.

- [RFC8570] Ginsberg, L., Ed., Previdi, S., Ed., Giacalone, S., Ward, D., Drake, J., and Q. Wu, "IS-IS Traffic Engineering (TE) Metric Extensions", [RFC 8570](#), DOI 10.17487/RFC8570, March 2019, <<https://www.rfc-editor.org/info/rfc8570>>.
- [RFC8578] Grossman, E., Ed., "Deterministic Networking Use Cases", [RFC 8578](#), DOI 10.17487/RFC8578, May 2019, <<https://www.rfc-editor.org/info/rfc8578>>.
- [RFC8655] Finn, N., Thubert, P., Varga, B., and J. Farkas, "Deterministic Networking Architecture", [RFC 8655](#), DOI 10.17487/RFC8655, October 2019, <<https://www.rfc-editor.org/info/rfc8655>>.

- [RFC8660] Bashandy, A., Ed., Filsfils, C., Ed., Previdi, S., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing with the MPLS Data Plane", [RFC 8660](#), DOI 10.17487/RFC8660, December 2019, <<https://www.rfc-editor.org/info/rfc8660>>.
- [RFC8964] Varga, B., Ed., Farkas, J., Berger, L., Malis, A., Bryant, S., and J. Korhonen, "Deterministic Networking (DetNet) Data Plane: MPLS", [RFC 8964](#), DOI 10.17487/RFC8964, January 2021, <<https://www.rfc-editor.org/info/rfc8964>>.
- [RFC8986] Filsfils, C., Ed., Camarillo, P., Ed., Leddy, J., Voyer, D., Matsushima, S., and Z. Li, "Segment Routing over IPv6 (SRv6) Network Programming", [RFC 8986](#), DOI 10.17487/RFC8986, February 2021, <<https://www.rfc-editor.org/info/rfc8986>>.
- [TSN-ATS] Specht, J., "P802.1Qcr – Bridges and Bridged Networks Amendment: Asynchronous Traffic Shaping", IEEE , July 2020, <<https://1.ieee802.org/tsn/802-1qcr/>>.
- [UBS] Specht, J. and S. Samii, "Urgency-Based Scheduler for Time-Sensitive Switched Ethernet Networks", IEEE 28th Euromicro Conference on Real-Time Systems (ECRTS), 2016.

Authors' Addresses

Toerless Eckert
 Futurewei Technologies USA
 2220 Central Expressway
 Santa Clara CA 95050
 USA

Email: tte@cs.fau.de

Eckert & Bryant	Expires January 13, 2022	[Page 28]
-----------------	--------------------------	-----------

Internet-Draft	bounded-latency-problems	July 2021
----------------	--------------------------	-----------

Stewart Bryant
 Stewart Bryant Ltd

Email: sb@stewartbryant.com

