Transport Area Working Group
Internet-Draft
Intended status: Informational
Expires: August 26, 2021

G. White, Ed.
CableLabs

February 22, 2021

Operational Guidance for Deployment of L4S in the Internet
draft-white-tsvwg-l4sops-02

Abstract

This document is intended to provide additional guidance to operators of end-systems, operators of networks, and researchers beyond that provided in [I-D.ietf-tsvwg-ecn-l4s-id] and [I-D.ietf-tsvwg-aqm-dualq-coupled] in order to ensure successful deployment of L4S [I-D.ietf-tsvwg-l4s-arch] in the Internet. The focus of this document is on potential interactions between L4S flows and Classic ECN ([RFC3168]) flows in Classic ECN bottleneck links. The document discusses the potential outcomes of these interactions, describes mechanisms to detect the presence of [RFC3168] bottlenecks, and identifies opportunites to prevent and/or detect and resolve fairness problems in such networks.

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 August 26, 2021.

Table of Contents

1. Introduction 2
2. Per-Flow Fairness 4
3. Detection of Classic ECN Bottlenecks 6
4. Operator of an L4S host 6
4.1. Edge Servers 8
4.2. Other hosts 9
5. Operator of a Network Employing RFC3168 FIFO Bottlenecks . . 9
5.1. Configure AQM to treat ECT(1) as NotECT 9
5.2. ECT(1) Tunnel Bypass 10
5.3. Configure Non-Coupled Dual Queue 10
5.4. WRED with ECT(1) Differentation 11
5.5. Disable RFC3168 ECN Marking 11
5.6. Re-mark ECT(1) to NotECT Prior to AQM 11
6. Contributors 11
7. IANA Considerations 12
8. Security Considerations 12
9. Informative References 12
Author's Address 13

1. Introduction

Low-latency, low-loss, scalable throughput (L4S) [I-D.ietf-tsvwg-l4s-arch] traffic is designed to provide lower queuing delay than conventional traffic via a new network service based on a modified Explicit Congestion Notification (ECN) response from the network. L4S traffic is identified by the ECT(1) codepoint, and network bottlenecks that support L4S should congestion-mark ECT(1) packets to enable L4S congestion feedback. However, L4S traffic is also expected to coexist well with classic congestion controlled traffic even if the bottleneck queue does not support L4S. This includes paths where the bottleneck link utilizes packet drops in response to congestion (either due to buffer overrun or active queue management), as well as paths that implement a 'flow-queuing' scheduler such as fq_codel [RFC8290]. A potential area of poor interoperability lies in network bottlenecks employing a shared queue that implements an Active Queue Management (AQM) algorithm that provides Explicit Congestion Notification signaling according to [RFC3168]. Although RFC3168 has been updated (via [RFC8311]) to reserve ECT(1) for experimental use only (also see [IANA-ECN]), and

its use for L4S has been specified in [I-D.ietf-tsvwg-ecn-l4s-id], not all deployed queues have been updated accordingly. It has been demonstrated ([Fallback]) that when a set of long-running flows comprising both classic congestion controlled flows and L4S-compliant congestion controlled flows compete for bandwidth in such a legacy shared RFC3168 queue, the classic congestion controlled flows may achieve lower throughput than they would have if all of the flows had been classic congestion controlled flows. This 'unfairness' between the two classes is more pronounced on longer RTT paths (e.g. 50ms and above) and/or at higher link rates (e.g. 50 Mbps and above). The lower the capacity per flow, the less pronounced the problem becomes. Thus the imbalance is most significant when the slowest flow rate is still high in absolute terms.

The root cause of the unfairness is that a legacy RFC3168 queue does not differentiate between packets marked ECT(0) (used by classic senders) and those marked ECT(1) (used by L4S senders), and provides an identical congestion signal (CE marks) to both types, while the L4S architecture redefines the CE mark and congestion response in the case of ECT(1) marked packets. The result is that the two classes respond differently to the CE congestion signal. The classic senders expect that CE marks are sent very rarely (e.g. approximately 1 CE mark every 200 round trips on a 50 Mbps x 50ms path) while the L4S senders expect very frequent CE marking (e.g. approximately 2 CE marks per round trip). The result is that the classic senders respond to the CE marks provided by the bottleneck by yielding capacity to the L4S flows. The resulting rate imbalance can be demonstrated, and could be a cause of concern in some cases.

This concern primarily relates to single-queue (FIFO) bottleneck links that implement legacy RFC3168 ECN, but the situation can also potentially occur in fq_codel [RFC8290] bottlenecks when flow isolation is imperfect due to hash collisions or VPN tunnels.

While the above mentioned unfairness has been demonstrated in laboratory testing, it has not been observed in operational networks, in part because members of the Transport Working group are not aware of any deployments of single-queue Classic ECN bottlenecks in the Internet. Additionally, this issue was considered and is discussed in Appendix B.1 of [I-D.ietf-tsvwg-ecn-l4s-id]. It was recognized that compromises would have to be made because IP header space is extremely limited. A number of alternative codepoint schemes were compared for their ability to traverse most Internet paths, to work over tunnels, to work at lower layers, to work with TCP, etc. It was decided to progress on the basis that robust performance in presence of these single-queue RFC3168 bottlenecks is not the most critical issue, since it was believed that they are rare. Nonetheless, there is the possibility that such deployments exist, and hence an interest

in providing guidance to ensure that measures can be taken to address the potential issues, should they arise in practice.

2. Per-Flow Fairness

There are a number of factors that influence the relative rates achieved by a set of users or a set of applications sharing a queue in a bottleneck link. Notably the response that each application has to congestion signals (whether loss or explicit signaling) can play a large role in determining whether the applications share the bandwidth in an equitable manner. In the Internet, ISPs typically control capacity sharing between their customers using a scheduler at the access bottleneck rather than relying on the congestion responses of end-systems. So in that context this question primarily concerns capacity sharing between the applications used by one customer. Nonetheless, there are many networks on the Internet where capacity sharing relies, at least to some extent, on congestion control in the end-systems. The traditional norm for congestion response has been that it is handled on a per-connection basis, and that (all else being equal) it results in each connection in the bottleneck achieving a data rate inversely proportional to the average RTT of the connection. The end result (in the case of steady-state behavior of a set of like connections) is that each user or application achieves a data rate proportional to N/RTT, where N is the number of simultaneous connections that the user or application creates, and RTT is the harmonic mean of the average round-trip-times for those connections. Thus, users or applications that create a larger number of connections and/or that have a lower RTT achieve a larger share of the bottleneck link rate than others.

While this may not be considered fair by many, it nonetheless has been the typical starting point for discussions around fairness. In fact it has been common when evaluating new congestion responses to actually set aside N & RTT as variables in the equation, and just compare per-flow rates between flows with the same RTT. For example [RFC5348] defines the congestion response for a flow to be '"reasonably fair" if its sending rate is generally within a factor of two of the sending rate of a [Reno] TCP flow under the same conditions.' Given that RTTs can vary by roughly two orders of magnitude and flow counts can vary by at least an order of magnitude between applications, it seems that the accepted definition of reasonable fairness leaves quite a bit of room for different levels of performance between users or applications, and so perhaps isn't the gold standard, but is rather a metric that is used because of its convenience.

In practice, the effect of this RTT dependence has historically been muted by the fact that many networks were deployed with very large

("bloated") drop-tail buffers that would introduce queuing delays well in excess of the base RTT of the flows utilizing the link, thus equalizing (to some degree) the effective RTTs of those flows. Recently, as network equipment suppliers and operators have worked to improve the latency performance of the network by the use of smaller buffers and/or AQM algorithms, this has had the side-effect of uncovering the inherent RTT bias in classic congestion control algorithms.

The L4S architecture aims to significantly improve this situation, by requiring senders to adopt a congestion response that eliminates RTT bias as much as possible (see [I-D.ietf-tsvwg-ecn-l4s-id]). As a result, L4S promotes a level of per-flow fairness beyond what is ordinarily considered for classic senders, the legacy RFC3168 issue notwithstanding.

It is also worth noting that the congestion control algorithms deployed currently on the internet tend toward (RTT-weighted) fairness only over long timescales. For example, the cubic algorithm can take minutes to converge to fairness when a new flow joins an existing flow on a link [Cubic]. Since the vast majority of TCP connections don't last for minutes, it is unclear to what degree per-flow, same-RTT fairness, even when demonstrated in the lab, translates to the real world.

So, in real networks, where per-application, per-end-host or per-customer fairness may be more important than long-term, same-RTT, per-flow fairness, it may not be that instructive to focus on the latter as being a necessary end goal.

Nonetheless, situations in which the presence of an L4S flow has the potential to cause harm [Harm] to classic flows need to be understood. Most importantly, if there are situations in which the introduction of L4S traffic would degrade classic traffic performance significantly, i.e. to the point that it would be considered starvation, these situations need to be understood and either remedied or avoided.

Aligned with this context, the guidance provided in this document is aimed not at monitoring the relative performance of L4S senders compared against classic senders on a per-flow basis, but rather at identifying instances where RFC3168 bottlenecks are deployed so that operators of L4S senders can have the opportunity to assess whether any actions need to be taken. Additionally this document provides guidance for network operators around configuring any RFC3168 bottlenecks to minimize the potential for negative interactions between L4S and classic senders.

3. Detection of Classic ECN Bottlenecks

The IETF encourages researchers, end system deployers and network operators to conduct experiments to identify to what degree legacy RFC3168 bottlecks exist in networks. These types of measurement campaigns, even if each is conducted over a limited set of paths, could be useful to further understand the scope of any potential issues, to guide end system deployers on where to examine performance more closely (or possibly delay L4S deployment), and to help network operators identify nodes where remediation may be necessary to provide the best performance.

The design of such experiments should consider not only the detection of RFC3168 ECN marking, but also the determination whether the bottleneck AQM is a single queue (FIFO) or a flow-queuing system. It is believed that the vast majority, if not all, of the RFC3168 AQMs in use at bottleneck links are flow-queuing systems (e.g. fq_codel [RFC8290] or [COBALT]). When flow isolation is successful, the FQ scheduling of such queues isolates classic congestion control traffic from L4S traffic, and thus eliminates the potential for unfairness. But, these systems are known to sometimes result in imperfect isolation, either due to hash collisions (see Section 5.3 of [RFC8290]) or because of VPN tunneling (see Section 6.2 of [RFC8290]). It is believed that the majority of fq_codel deployments in bottleneck links today (e.g. [Cake]) employ hashing algorithms that virtually eliminate the possibility of collisions, making this a non-issue for those deployments. But, VPN tunnels remain an issue for fq_codel deployments, and the introduction of L4S traffic raises the possibility that tunnels containing mixed classic and L4S traffic would exist, in which case fq_codel implementations that have not been updated to be L4S-aware could exhibit similar unfairness properties as single queue AQMs. Until such queues are upgraded to support L4S or treat ECT(1) as not-ECT traffic, end-host mitigations such as separating L4S and Classic traffic into distinct VPN tunnels could be employed.

[Fallback] contains recommendations on some of the mechanisms that can be used to detect legacy RFC3168 bottlenecks. TODO: summarize the main ones here.

4. Operator of an L4S host

From a host's perspective, support for L4S involves both endpoints: ECT(1) marking & L4S-compatible congestion control at the sender, and ECN feedback at the receiver. Between these two entities, it is primarily incumbent upon the sender to evaluate the potential for presence of legacy RFC3168 FIFO bottlenecks and make decisions whether or not to use L4S congestion control. A general purpose

receiver is not expected to perform any testing or monitoring for RFC3168, and is also not expected to invoke any active response in the case that such a bottleneck exists. That said, it is certainly possible for receivers to disable L4S functionality by not negotiating ECN support with the sender.

Prior to deployment of any new technology, it is commonplace for the parties involved in the deployment to validate the performance of the new technology, via lab testing, limited field testing, large scale field testing, etc. The same is expected for deployers of L4S technology. As part of that validation, it is recommended that deployers consider the issue of RFC3168 FIFO bottlenecks and conduct experiments as described in the previous section, or otherwise assess the impact that the L4S technology will have in the networks in which it is to be deployed, and take action as is described further in this section.

If pre-deployment testing raises concerns about issues with RFC3168 bottlenecks, the actions taken may depend on the server type:

- General purpose servers (e.g. web servers)
  - Active testing could be performed by the server. For example, a javascript application could run simultaneous downloads during page reading time in order to survey for presence of legacy RFC3168 FIFO bottlenecks on paths to users.
  - Passive testing could be built in to the transport protocol implementation at the sender in order to perform detection (see [Fallback]).
  - Taking action based on the detection of RFC3168 FIFO bottlenecks is likely not needed for short transactional transfers (e.g. sub 10 seconds) since these are unlikely to achieve the steady-state conditions where unfairness has been observed.
  - For longer file transfers, it may be possible to fall-back to Classic behavior in real-time, or to simply disable L4S for future long file transfers to clients where legacy RFC3168 has been detected.
- Specialized servers handling long-running sessions (e.g. cloud gaming)
  - Active testing could be performed at each session startup

* Active testing could be integrated into a "pre-validation" of the service, done when the user signs up, and periodically thereafter

* In-band detection as described in [Fallback] could be performed during the session

In addition, the responsibilities of and actions taken by a sender may depend on the environment in which it is deployed. The following sub-sections discuss two scenarios: senders serving a limited known target audience and those that serve an unknown target audience.

4.1. Edge Servers

Some hosts (such as CDN leaf nodes and servers internal to an ISP) are deployed in environments in which they serve content to a constrained set of networks or clients. The operator of such hosts may be able to determine whether there is the possibility of [RFC3168] FIFO bottlenecks being present, and utilize this information to make decisions on selectively deploying L4S and/or disabling it (e.g. bleaching ECN). Furthermore, such an operator may be able to determine the likelihood of an L4S bottleneck being present, and use this information as well.

For example, if a particular network is known to have deployed legacy [RFC3168] FIFO bottlenecks, deployment of L4S for that network should be delayed until those bottlenecks can be upgraded to mitigate any potential issues as discussed in the next section.

Prior to deploying L4S on edge servers a server operator should:

o Consult with network operators on presence of legacy [RFC3168] FIFO bottlenecks

o Consult with network operators on presence of L4S bottlenecks

o Perform pre-deployment testing per network

If a particular network offers connectivity to other networks (e.g. in the case of an ISP offering service to their customer's networks), the lack of RFC3168 FIFO bottleneck deployment in the ISP network can't be taken as evidence that RFC3168 FIFO bottlenecks don't exist end-to-end (because one may have been deployed by the end-user network). In these cases, deployment of L4S will need to take appropriate steps to detect the presence of such bottlenecks. At present, it is believed that the vast majority of RFC3168 bottlenecks in end-user networks are implementations that utilize fq_codel or Cake, where the unfairness problem is less likely to be a concern.

While this doesn't completely eliminate the possibility that a legacy [RFC3168] FIFO bottleneck could exist, it nonetheless provides useful information that can be utilized in the decision making around the potential risk for any unfairness to be experienced by end users.

4.2. Other hosts

Hosts that are deployed in locations that serve a wide variety of networks face a more difficult prospect in terms of handling the potential presence of RFC3168 FIFO bottlenecks. Nonetheless, the steps listed in the ealier section (based on server type) can be taken to minimize the risk of unfairness.

Since existing studies have hinted that RFC3168 FIFO bottlenecks are rare, detections using these techniques may also prove to be rare. Therefore, it may be possible for a host to cache a list of end host ip addresses where a RFC3168 bottleneck has been detected. Entries in such a cache would need to age-out after a period of time to account for IP address changes, path changes, equipment upgrades, etc.

It has been suggested that a public blacklist of domains that implement RFC3168 FIFO bottlenecks or a public whitelist of domains that are participating in the L4S experiment could be maintained. There are a number of significant issues that would seem to make this idea infeasible, not the least of which is the fact that presence of RFC3168 FIFO bottlenecks or L4S bottlenecks is not a property of a domain, it is the property of a path between two endpoints.

5. Operator of a Network Employing RFC3168 FIFO Bottlenecks

While it is, of course, preferred for networks to deploy L4S-capable high fidelity congestion signaling, and while it is more preferable for L4S senders to detect problems themselves, a network operator who has deployed equipment in a likely bottleneck link location (i.e. a link that is expected to be fully saturated) that is configured with a leagcy [RFC3168] FIFO AQM can take certain steps in order to improve rate fairness between classic traffic and L4S traffic, and thus enable L4S to be deployed in a greater number of paths.

Some of the options listed in this section may not be feasible in all networking equipment.

5.1. Configure AQM to treat ECT(1) as NotECT

If equipment is configurable in such as way as to only supply CE marks to ECT(0) packets, and treat ECT(1) packets identically to

NotECT, or is upgradable to support this capability, doing so will eliminate the risk of unfairness.

5.2. ECT(1) Tunnel Bypass

Using an [RFC6040] compatibility mode tunnel, tunnel ECT(1) traffic through the [RFC3168] bottleneck with the outer header indicating Not-ECT.

Two variants exist for this approach

1. per-domain: tunnel ECT(1) pkts to domain edge towards dst

2. per-dst: tunnel ECT(1) pkts to dst

5.3. Configure Non-Coupled Dual Queue

Equipment supporting [RFC3168] may be configurable to enable two parallel queues for the same traffic class, with classification done based on the ECN field.

Option 1:

- Configure 2 queues, both with ECN; 50:50 WRR scheduler
  - Queue #1: ECT(1) & CE packets - Shallow immediate AQM target
  - Queue #2: ECT(0) & NotECT packets - Classic AQM target
- Outcome in the case of n L4S flows and m long-running Classic flows
  - if m & n are non-zero, flows get 1/2n and 1/2m of the capacity, otherwise 1/n or 1/m
  - never < 1/2 each flow's rate if all had been Classic

This option would allow L4S flows to achieve low latency, low loss and scalable throughput, but would sacrifice the more precise flow balance offered by [I-D.ietf-tsvwg-aqm-dualq-coupled]. This option would be expected to result in some reordering of previously CE marked packets sent by Classic ECN senders, which is a trait shared with [I-D.ietf-tsvwg-aqm-dualq-coupled]. As is discussed in [I-D.ietf-tsvwg-ecn-l4s-id], this reordering would be either zero risk or very low risk.

Option 2:

- Configure 2 queues, both with AQM; 50:50 WRR scheduler
  - Queue #1: ECT(1) & NotECT packets - ECN disabled
  - Queue #2: ECT(0) & CE packets - ECN enabled
- Outcome
  - ECT(1) treated as NotECT
  - Flow balance for the 2 queues the same as in option 1

This option would not allow L4S flows to achieve low latency, low loss and scalable throughput in this bottleneck link. As a result it is a less prefered option.

5.4. WRED with ECT(1) Differentation

This configuration is similar to Option 2 in the previous section, but uses a single queue with WRED functionality.

- Configure the queue with two WRED classes
- Class #1: ECT(1) & NotECT packets - ECN disabled
- Class #2: ECT(0) & CE packets - ECN enabled

5.5. Disable RFC3168 ECN Marking

Disabling [RFC3168] ECN marking eliminates the unfairness issue. Clearly a downside to this approach is that classic senders will no longer get the benefits of Explict Congestion Notification.

5.6. Re-mark ECT(1) to NotECT Prior to AQM

While not a recommended alternative, remarking ECT(1) packets as NotECT (i.e. bleaching ECT(1)) ensures that they are treated identically to classic NotECT senders. However, this also eliminates the possibility of downstream L4S bottlenecks providing high fidelity congestion signals.

6. Contributors

Thanks to Bob Briscoe, Jake Holland, Koen De Schepper, Olivier Tilmans, Tom Henderson, Asad Ahmed, and members of the TSVWG mailing list for their contributions to this document.

7. IANA Considerations

None.

8. Security Considerations

For further study.

9. Informative References

[Cake] Hoiland-Jorgensen, T., Taht, D., and J. Morton, "Piece of CAKE: A Comprehensive Queue Management Solution for Home Gateways", 2018, <https://arxiv.org/abs/1804.07617>.

[COBALT] Palmei, J. and et al., "Design and Evaluation of COBALT Queue Discipline", IEEE International Symposium on Local and Metropolitan Area Networks 2019, 2019, <https://ieeexplore.ieee.org/abstract/document/8847054>.

[Cubic] Ha, S., Rhee, I., and L. Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant", ACM SIGOPS Operating Systems Review , 2008, <https://www.cs.princeton.edu/courses/archive/fall16/cos561/papers/Cubic08.pdf>.

[Fallback] Briscoe, B. and A. Ahmed, "TCP Prague Fall-back on Detection of a Classic ECN AQM", ArXiv , Feb 2021, <https://arxiv.org/abs/1911.00710>.

[Harm] Ware, R., Mukerjee, M., Seshan, S., and J. Sherry, "Beyond Jain's Fairness Index: Setting the Bar For The Deployment of Congestion Control Algorithms", Hotnets'19 , 2019, <https://www.cs.cmu.edu/~rware/assets/pdf/ware-hotnets19.pdf>.

[I-D.ietf-tsvwg-aqm-dualq-coupled] Schepper, K., Briscoe, B., and G. White, "DualQ Coupled AQMs for Low Latency, Low Loss and Scalable Throughput (L4S)", draft-ietf-tsvwg-aqm-dualq-coupled-13 (work in progress), November 2020.

[I-D.ietf-tsvwg-ecn-l4s-id] Schepper, K. and B. Briscoe, "Identifying Modified Explicit Congestion Notification (ECN) Semantics for Ultra-Low Queuing Delay (L4S)", draft-ietf-tsvwg-ecn-l4s-id-12 (work in progress), November 2020.

[I-D.ietf-tsvwg-l4s-arch]
Briscoe, B., Schepper, K., Bagnulo, M., and G. White, "Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture", draft-ietf-tsvwg-l4s-arch-08 (work in progress), November 2020.

[IANA-ECN]
Internet Assigned Numbers Authority, "IANA ECN Field Assignments", 2018, <https://www.iana.org/assignments/dscp-registry/dscp-registry.xhtml#ecn-field>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <https://www.rfc-editor.org/info/rfc3168>.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, DOI 10.17487/RFC5348, September 2008, <https://www.rfc-editor.org/info/rfc5348>.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, DOI 10.17487/RFC6040, November 2010, <https://www.rfc-editor.org/info/rfc6040>.

[RFC8290] Hoeiland-Joergensen, T., McKenney, P., Taht, D., Gettys, J., and E. Dumazet, "The Flow Queue CoDel Packet Scheduler and Active Queue Management Algorithm", RFC 8290, DOI 10.17487/RFC8290, January 2018, <https://www.rfc-editor.org/info/rfc8290>.

[RFC8311] Black, D., "Relaxing Restrictions on Explicit Congestion Notification (ECN) Experimentation", RFC 8311, DOI 10.17487/RFC8311, January 2018, <https://www.rfc-editor.org/info/rfc8311>.

Author's Address

Greg White (editor)
CableLabs

Email: g.white@cablelabs.com