Transport Area Working Group
Internet-Draft
Intended status: Informational
Expires: 8 January 2026

G. White, Ed.
CableLabs

7 July 2025

Operational Guidance on Coexistence with Classic ECN during L4S Deployment
draft-ietf-tsvwg-l4sops-08

Abstract

This document provides guidance in order to ensure successful deployment of Low Latency Low Loss Scalable throughput (L4S) in the Internet. Other L4S documents provide guidance for running an L4S experiment, but this document is focused solely on potential interactions between L4S flows and flows using the original ('Classic') ECN over a Classic ECN bottleneck. The document discusses the potential outcomes of these interactions, describes mechanisms to detect the presence of Classic ECN bottlenecks, and identifies opportunities to prevent and/or detect and resolve fairness problems in such networks. This guidance is aimed at operators of end-systems, operators of networks, and researchers.

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 8 January 2026.

Table of Contents

1. Introduction

Low-latency, low-loss, scalable throughput (L4S) [RFC9330] 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 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]. RFC3168 has been updated (via [RFC8311]) to reserve ECT(1) (also see [IANA-ECN]), and its use for L4S has been specified in [RFC9331]. However, any deployed RFC3168 AQMs might not be updated, and RFC8311 still prefers that routers not involved in L4S experimentation treat ECT(1) and ECT(0) as equivalent. It has been demonstrated [Briscoe] 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 bandwidth delay product of a flow, the less pronounced the problem becomes. Thus the imbalance is often most significant when the slowest flow rate is still high in absolute terms.

The root cause of the unfairness is that the L4S architecture redefines the congestion signal (CE mark) and congestion response in the case of packets marked ECT(1) (used by L4S senders), whereas a RFC3168 queue does not differentiate between packets marked ECT(0) (used by classic senders) and those marked ECT(1), and provides CE marks identically to both types. 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) bottlenecks that implement RFC3168 ECN, but the situation can also potentially occur with per-flow queuing, e.g., fq_codel [RFC8290], 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.

This issue was considered in November 2015 (and reaffirmed in April 2020) when the WG decided on the identifier to use for L4S, as recorded in Appendix B.1 of [RFC9331]. 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 there is the possibility that they could be deployed/enabled in the future. Since any negative impact of this coexistence issue would not be directly experienced by the party experimenting with L4S endpoints, but rather by the other users of the bottleneck, there is 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 bottleneck queue. 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 site. 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 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 converges "towards a rate that is as independent of RTT as is possible without compromising stability or utilization" (see [RFC9331]). As a result, L4S promotes a level of per-flow fairness beyond what is ordinarily considered for classic senders, the 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 sharing a bottleneck queue [Ha]. 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 might be more important than long-term, same-RTT, per-flow fairness, it might not be helpful 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 [Ware] to classic flows need to be understood. Most importantly, if there are situations in which the introduction of L4S traffic would degrade both the absolute and relative performance of classic traffic significantly, i.e. to the point that it would be considered starvation while L4S was not starved, 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. Flow Queuing Systems

As noted above, the concern around RFC3168 coexistence mainly concerns single-queue systems where classic and L4S traffic are mixed. In a flow-queuing system, 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]), because of VPN tunneling (see Section 6.2 of [RFC8290]), or due to deliberate configuration (see Section 7, Paragraph 5).

It is believed that the majority of FQ deployments in bottlenecks today (e.g., Cake [Hoiland-Jorgensen]) 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 deployments, and the introduction of L4S traffic raises the possibility that tunnels containing mixed classic and L4S traffic would exist, in which case FQ implementations that have not been updated to be L4S-aware could exhibit similar unfairness properties as single queue AQMs. Section 7 discusses some remedies that can be implemented by operators of FQ equipment in order to minimize this risk. Additionally, end-host mitigations such as separating L4S and Classic traffic into distinct VPN tunnels could be employed.

4. Detection of Classic ECN Bottlenecks

The IETF encourages researchers, end system deployers and network operators to conduct experiments to identify to what degree RFC3168 bottlenecks 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.

4.1. Recent Studies

A small number of recent studies have attempted to gauge the level of RFC3168 AQM deployment in the internet.

In 2020, Akamai conducted a study (https://mailarchive.ietf.org/arch/msg/tsvwg/2tbRHphJ8K_CE6is9n7iQy-VAZM/) of "downstream" (server to client) CE marking broken out by ASN on two separate days, one in late March, the other in mid July [Holland]. They concluded that prevalence of CE-marking was low across the ~800 ASNs observed (0.19% - 0.30% of ECT client IPs ever saw a CE mark), but it was growing, and that they could not determine whether the CE marking was due to a single queue or FQ. They also observed that RFC3168 AQMs are not uniformly distributed. There were three small ISPs where prevalence of CE-marking was above ~70%, indicating a likely deployment by the ISP. There were another four small ASNs where the prevalence was between 10% and 20%, which may also indicate deployment by the ISP. There were also roughly six larger ASNs (and perhaps 20 small ASNs) where the prevalence was between 3% and 8%.

In 2017, Apple reported on their observations of ECN marking by networks, broken out by country [Bhooma]. They reported four countries that exceeded the global baseline seen by Akamai, but one of these (Argentine Republic) was later discovered to be due to a bug (https://datatracker.ietf.org/meeting/106/materials/slides-106-tsvwg-sessa-72-l4s-drafts-00#page=15), leaving three countries: China 1% of paths, Mexico 3.2% of paths, France 6% of paths. The percentage in France appears consistent with reports (https://mailarchive.ietf.org/arch/msg/tsvwg/UyvpwUiNw0obd_EylBBV7kDRIHs/) that fq_codel has been implemented in DSL home routers deployed by Free.fr.

In December 2020 - January 2021, Pete Heist worked with a small cooperative WISP in the Czech Republic to collect data on CE-marking [I-D.heist-tsvwg-ecn-deployment-observations]. Overall, 18.6% of paths saw possible RFC3168 AQM activity, which appears to place this ISP in the small group with moderately high RFC3168 prevalence reported by Akamai. This ISP was known to have deployed RFC3168 fq_codel equipment in some of their subnets, and in other subnets there were 33 IPs where possible AQM activity was observed via CE-marks and/or ECE flags, corresponding to approximately 10% of paths. It was agreed (https://mailarchive.ietf.org/arch/msg/tsvwg/Rj7GylByZuFa3_LTCMvEfb-CYpw/) that these were likely to be due to fq_codel implementations in home routers deployed by members of the cooperative.

The interpretation of these studies seems to be that there are no known deployments of FIFO RFC3168, all of the known RFC3168 deployments are fq_codel, the majority of the currently unknown deployments are likely to be fq_codel, and there may be a small number of networks where CE-marking is prevalent (and thus likely ISP-managed) where it is currently unknown as to whether the source is a FIFO or an FQ system.

Other studies (e.g., [Trammel], [Bauer], [Mandalari]) have examined ECN traversal, but have not reported data on prevalence of CE-marking by networks. Another [Roddav] examined traces from a Tier 1 ISP link in 2018 and observed that 94% of the non-zero ECN marked packets were CE, which appears to reflect a misconfiguration of equipment using that link, as opposed to providing evidence of RFC3168 AQM deployment.

4.2. Future Experiments

The design of future 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 (FQ) system. It is believed that the vast majority, if not all, of the RFC3168 AQMs in use at bottlenecks are flow-queuing systems (e.g., fq_codel [RFC8290] or COBALT [Palmei]).

[Briscoe] contains recommendations on some of the mechanisms that can be used to detect RFC3168 bottlenecks. In particular, Section 4 of [Briscoe] outlines an approach for out-band-detection of RFC3168 bottlenecks.

5. Operator of an L4S host

From a host’s perspective, support for L4S only involves the sender via ECT(1) marking & L4S-compatible congestion control. The receiver is involved in ECN feedback but can generally be agnostic to whether ECN is being used for L4S [RFC9330]. Between these two entities, it is primarily incumbent upon the sender to evaluate the potential for presence of RFC3168 FIFO bottlenecks and make decisions whether or not to use L4S congestion control. While is is possible for a receiver to disable L4S functionality by not negotiating ECN, 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.

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., usually in a progressive manner. 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. This sort of progressive (incremental) deployment helps to ensure that any issues are discovered when the scale of those issues is relatively small.

Some of the recommendations in this section involve the sender determining (through various means) the likelihood of a particular path having a bottleneck that implements single queue RFC3168 AQM. Since this determination can be imprecise, there exists some risk that a path is incorrectly classified. In the case of false-positives (where a path is erroneously believed to contain RFC3168), discontinuing the use of L4S on that path would result in a lost opportunity for low-latency low-loss service, and thus likely an unnecessary degradation in the quality of experience for the user. In the case of false-negatives, the use of L4S has the potential to result in a reduction in the throughput of non-L4S flows while the L4S flow is active. In environments where the risk of false-negatives is significant, it is recommended that hosts limit the use of L4S congestion control to application-limited flows that are especially sensitive to latency, latency variation and loss.

5.1. Server Type

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

5.1.1. General purpose servers (e.g., web servers)

* Out-of-band active testing could be performed by the server. For example, a JavaScript application could run simultaneous downloads (i.e. with and without L4S) during page reading time in order to survey for presence of RFC3168 FIFO bottlenecks on paths to users (e.g., as described in Section 4 of [Briscoe]).

* In-band testing could be built in to the transport protocol implementation at the sender in order to perform detection (see Section 5 of [Briscoe], though note that this mechanism does not differentiate between FIFO and FQ).

Depending on the details of the L4S congestion control implementation, taking action based on the detection of RFC3168 FIFO bottlenecks may not be needed for short transactional transfers that are unlikely to achieve the steady-state conditions where unfairness

is likely to occur. For longer file transfers, it may be possible to fall-back to Classic behavior in real-time (i.e. when doing in-band testing), or to cache those destinations where RFC3168 has been detected, and disable L4S for subsequent long file transfers to those destinations.

5.1.2. Specialized servers handling long-running sessions (e.g., cloud gaming)

* Out-of-band active testing could be performed at each session startup

* Out-of-band 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 [Briscoe] could be performed during the session

5.2. Server deployment environment

The responsibilities of and actions taken by a sender may additionally 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.

5.2.1. 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.

It is recommended that L4S experimental deployments begin with such servers.

For example, if a particular network is known to have deployed legacy [RFC3168] FIFO bottlenecks, usage of L4S for long capacity-seeking file transfers on that network could 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:

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

* Consult with network operators on presence of L4S bottlenecks

* 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.

5.2.2. 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 earlier section (based on server type) can be taken to minimize the risk of unfairness.

It is recommended that operators of such hosts consider carefully whether these hosts are appropriate for early experimentation with L4S.

The interpretation of studies on ECN usage and their deployment context (see Section 4.1) has so far concluded that RFC3168 FIFO bottlenecks are likely to be rare, and so detections using these techniques may also prove to be rare. Additionally, the most recent large scale study [Holland] indicated that there were a small number of networks in which RFC3168 bottlenecks are more prevalent than the global average. Therefore, it may be possible for a host to maintain a list of networks where L4S should not be enabled, and, for other networks, 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 block-list of domains that implement RFC3168 FIFO bottlenecks 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 link, and therefore of the particular current path between two endpoints.

It has also been suggested that a public allow-list of domains that are participating in the L4S experiment could be maintained. This approach would not be useful, given the presence of an L4S domain on the path does not imply the absence of RFC3168 AQMs upstream or downstream of that domain. Also, the approach cannot cater for domains with a mix of L4S and RFC3168 AQMs.

6. Operator of a Network Employing RFC3168 FIFO Bottlenecks

While it is more preferable for L4S senders to detect problems themselves, a network operator who has deployed equipment in a likely bottleneck location (i.e. a link that is expected to frequently be fully saturated) that is configured with a legacy [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.

6.1. Preferred Options

The options in this section preserve the ability of the bottleneck to CE-mark ECT(1) packets as well as ECT(0) packets. The result of these options is that hosts utilizing classic (RFC3168) ECN and hosts utilizing L4S ECN receive the benefit of ECN. Further with these options, the hosts that choose to use L4S ECN see the benefit of reduced latency and latency-variation compared to hosts that choose instead to use classic ECN.

6.1.1. Upgrade AQMs to an L4S-aware AQM

If the RFC3168 AQM implementation can be upgraded to enable support for L4S, either via [RFC9332] or via an L4S-aware FQ implementation, this is the preferred approach to addressing potential unfairness, because it additionally enables all of the benefits of L4S.

Section 4.2 of [RFC9330] contains a description of the options available, including a discussion about L4S-aware FQ implementations.

6.1.2. Configure Non-Coupled Dual Queue with Shallow Target

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

* 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 [RFC9332]. 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 [RFC9332]. As is discussed in [RFC9331], this reordering would be either zero risk or very low risk.

If classification based on the ECN field isn’t possible in the bottleneck, this option may still be useful if an external system can be configured to reflect the ECN codepoint to another field that could then be used as an alternative identifier to classify traffic into Queue #1. For example, if at network ingress an edge router can apply a local-use DSCP to ECT(1) & CE packets, the bottleneck can then utilize a DSCP classifier. Similarly, in MPLS networks, ECT(1) & CE packets could use a different EXP value [RFC5129] than classic packets. More generally, any tunneling protocol can be used to proxy the ECN value of the encapsulated packet to its outer header, enabling bottlenecks to classify packets based on their input virtual interface.

6.1.3. Approximate Fair Dropping

The Approximate Fair Dropping ([AFD]) algorithm tracks individual flow rates and introduces either packet drops or CE-marks to each flow in proportion to the amount by which the flow rate exceeds a computed per-flow fair-share rate. Where an implementation of AFD or an equivalent algorithm is available, it could be enabled on an interface with a single-queue RFC3168 AQM as a fairly lightweight way to inject additional ECN marks into any significantly higher rate flows. See also [Cisco-N9000].

6.1.4. Replace RFC3168 FIFO with RFC3168 FQ

As discussed in Section XREF, implementations of RFC3168 with an FQ scheduler (e.g., fq_codel or Cake) significantly reduce the likelihood of experiencing any unfairness between Classic and L4S traffic.

6.1.5. Do Nothing

If it is infeasible to implement any of the above options, it may be preferable for an operator of RFC3168 FIFO bottlenecks to leave them unchanged. In many deployment situations the risk of fairness issues may be very low, and the impact if they occur may not be particularly troublesome. This could, for instance, be true in bottlenecks where there is a high degree of flow aggregation or in high-speed bottlenecks (e.g., greater than 100 Mbps).

6.2. Non-Preferred Options

The options in this section come with a downside that they treat ECT(1) packets as NotECT, and thus don’t provide the latency/loss benefit to flows marked ECT(1) (i.e. L4S flows). In the case that there is a strong concern about per-flow fairness between L4S flows and Classic flows in an RFC3168 FIFO bottleneck, and none of the remedies in the previous section can be implemented, the options listed in this section could be considered. These options are non-preferred because bottlenecks that implement them create a dilemma for operators of hosts, in that the application could see better performance if it uses classic (RFC3168) ECN rather than L4S ECN.

6.2.1. Configure Non-Coupled Dual Queue Treating ECT(1) as NotECT

* 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 is the same as in Section 6.1.2

This option could potentially be implemented using an identifier other than the ECN field, as discussed in Section 6.1.2.

6.2.2. WRED with ECT(1) Differentiation

This configuration is similar to the option described in Section 6.2.1, 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

This option could potentially be implemented using an identifier other than the ECN field, as discussed in Section 6.1.2.

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

If equipment is configurable in such a 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.

6.2.4. ECT(1) Tunnel Bypass

Tunnel ECT(1) traffic through the RFC3168 bottleneck with the outer header indicating Not-ECT, by using either an ECN tunnel ingress in Compatibility Mode [RFC6040] or a Limited Functionality ECN tunnel [RFC3168].

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

6.3. Last Resort Options

If serious issues are detected, where the presence of L4S flows is determined to be the likely cause, and none of the above options are implementable, the options in this section can be considered as a last resort. These options are not recommended.

6.3.1. Disable RFC3168 Support

Disabling an [RFC3168] AQM from CE marking both ECT(0) traffic and ECT(1) traffic eliminates the unfairness issue. A downside to this approach is that classic senders will no longer get the benefits of Explicit Congestion Notification at this bottleneck either. This alternative is only mentioned in case there is no other way to reconfigure an RFC3168 AQM.

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

Remarking ECT(1) packets as NotECT (i.e. bleaching ECT(1)) ensures that they are treated identically to classic NotECT senders. However, this action is not recommended because a) it would also prevent downstream L4S bottlenecks from providing high fidelity congestion signals; b) it could lead to problems with future experiments that use ECT(1) in alternative ways to L4S; and c) it would violate requirements in [RFC9331]. This alternative is mentioned as an absolute last resort in case there is no other way to reconfigure an RFC3168 AQM.

Note that the CE codepoint must never be bleached, otherwise it would black-hole congestion indications.

7. Operator of a Network Employing RFC3168 FQ Bottlenecks

A network operator who has deployed flow-queuing systems that implement RFC3168 (e.g., fq_codel or CAKE using default hashing) at network bottlenecks will likely see fewer potential issues when L4S traffic is present on their network as compared to operators of RFC3168 FIFOs. As discussed in Section 3, the flow queuing mechanism will typically isolate L4S flows and Classic flows into separate queues, and the scheduler will then enforce per-flow fairness. As a result, the potential fairness issues between Classic and L4S traffic that can occur in FIFOs will typically not occur in FQ systems. That said, FQ systems commonly treat a tunneled traffic aggregate as a single flow, and thus a tunneled traffic aggregate that contains a mix of Classic and L4S traffic will utilize a single queue, and the traffic within the tunnel could experience the same fairness issue as has been described for RFC3168 FIFOs. This unfairness is compounded by the fact that the FQ scheduler will already be causing unfairness

to flows within the tunnel relative to flows that are not tunneled (each of which gets the same bandwidth share as does the tunnel). Additionally, many of the deployed RFC3168 FQ systems currently implement an AQM algorithm (either CoDel or COBALT) that is designed for Classic traffic and reacts sluggishly to L4S (or unresponsive) traffic, with the result being that L4S senders could in some cases see worse latency performance than Classic senders.

While the potential unfairness result is arguably less impactful in the case of RFC3168 FQ bottlenecks, it is believed that RFC3168 FQ bottlenecks are currently more common than RFC3168 FIFO bottlenecks. The most common deployments of RFC3168 FQ bottlenecks are in home routers running OpenWRT firmware where the user has turned the feature on.

As is the case with RFC3168 FIFOs, the preferred remedy for a network operator that wishes to enable the best performance possible with regard to L4S, is for the network operator to update RFC3168 FQ bottlenecks to be L4S-aware (in the case of linux bottlenecks, see Appendix A). In cases where that is infeasible, several of the remedies described in the previous section can be used to reduce or eliminate these issues.

* Configure AQM to treat ECT(1) as NotECT

* Disable RFC3168 Support

* Re-mark ECT(1) to NotECT Prior to AQM

Note that some FQ schedulers can be configured to intentionally aggregate multiple flows into each queue. This might be used, for instance, to implement per-user or per-host fairness rather than per-flow fairness. In this case, if the flow aggregates contain a mix of Classic and L4S traffic, one would expect to see the same potential unfairness as is seen in the FIFO case. The same remedies mentioned above would apply in this case as well.

8. Conclusion of the L4S experiment

This section gives guidance on how L4S-deploying networks and endpoints should respond to either of the two possible outcomes of the IETF-supported L4S experiment.

8.1. Termination of a successful L4S experiment

If the L4S experiment is deemed successful, the IETF would be expected to move the L4S specifications to standards track. Networks would then be encouraged to continue/begin deploying L4S-aware nodes and to replace all non-L4S-aware RFC3168 AQMs already deployed as far as feasible, or at least restrict RFC3168 AQM to interpret ECT(1) equal to NotECT. Networks that participated in the experiment would be expected to track the evolution of the L4S standards and adapt their implementations accordingly (e.g., if as part of switching from experimental to standards track, changes in the L4S RFCs become necessary).

8.2. Termination of an unsuccessful L4S experiment

If the L4S experiment is deemed unsuccessful (e.g., due to lack of deployment of compliant end-systems or AQMs), it might need to be terminated: any L4S network nodes should then be un-deployed and the ECT(1) codepoint usage should be released/recycled as quickly as possible, recognizing that this process may take some time. To facilitate this potential outcome, [RFC9331] requires L4S hosts to be configurable to revert to non-L4S congestion control, and networks to be configurable to treat ECT(1) the same as ECT(0).

9. Contributors

Thanks to Bob Briscoe, Jake Holland, Koen De Schepper, Olivier Tilmans, Tom Henderson, Asad Ahmed, Gorry Fairhurst, Sebastian Moeller, Pete Heist, Ingemar Johansson, and members of the TSVWG mailing list for their contributions to this document.

10. IANA Considerations

None.

11. Security Considerations

This is an informational document that discusses operational strategies related to the deployment of L4S [RFC9330], and does not introduce new security considerations. The security considerations for L4S and ECN deployment described in [RFC3168], [RFC9330], [RFC9331] and [RFC9332] apply.

12. Normative References

[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>.

[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>.

[RFC9330] Briscoe, B., Ed., De Schepper, K., Bagnulo, M., and G. White, "Low Latency, Low Loss, and Scalable Throughput (L4S) Internet Service: Architecture", RFC 9330, DOI 10.17487/RFC9330, January 2023, <https://www.rfc-editor.org/info/rfc9330>.

[RFC9331] De Schepper, K. and B. Briscoe, Ed., "The Explicit Congestion Notification (ECN) Protocol for Low Latency, Low Loss, and Scalable Throughput (L4S)", RFC 9331, DOI 10.17487/RFC9331, January 2023, <https://www.rfc-editor.org/info/rfc9331>.

[RFC9332] De Schepper, K., Briscoe, B., Ed., and G. White, "Dual-Queue Coupled Active Queue Management (AQM) for Low Latency, Low Loss, and Scalable Throughput (L4S)", RFC 9332, DOI 10.17487/RFC9332, January 2023, <https://www.rfc-editor.org/info/rfc9332>.

13. Informative References

[AFD] Pan, R., Breslau, L., Prabhakar, B., and S. Shenker, "Approximate Fairness through Differential Dropping", Computer Comm. Rev. vol.33, no.1, January 2003, <https://people.eecs.berkeley.edu/~istoica/classes/cs268/10/papers/afd.pdf>.

[Bauer] Bauer, S., Beverly, R., and A. Berger, "Measuring the State of ECN Readiness in Servers, Clients, and Routers", Proc ACM SIGCOMM Internet Measurement Conference IMC' 11, 2011, <http://conferences.sigcomm.org/imc/2011/docs/p171.pdf>.

[Bhooma] Bhooma, P., "TCP ECN: Experience with enabling ECN on the Internet", 98th IETF MAPRG Presentation , 2017, <https://datatracker.ietf.org/meeting/98/materials/slides-98-maprg-tcp-ecn-experience-with-enabling-ecn-on-the-internet-padma-bhooma-00>.

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

[Cisco-N9000] "Intelligent Buffer Management on Cisco Nexus 9000 Series Switches White Paper", Cisco Product Document 1486580292771926, 6 June 2017, <https://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/white-paper-c11-738488.html>.

[Ha] 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>.

[Hoiland-Jorgensen] 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>.

[Holland] Holland, J., "Latency & AQM Observations on the Internet", IETF MAPRG interim-2020-maprg-01, August 2020, <https://www.ietf.org/proceedings/interim-2020-maprg-01/slides/slides-interim-2020-maprg-01-sessa-latency-aqm-observations-on-the-internet-01.pdf>.

[I-D.heist-tsvwg-ecn-deployment-observations] Heist, P. and J. Morton, "Explicit Congestion Notification (ECN) Deployment Observations", Work in Progress, Internet-Draft, draft-heist-tsvwg-ecn-deployment-observations-02, 8 March 2021, <https://www.ietf.org/archive/id/draft-heist-tsvwg-ecn-deployment-observations-02.html>.

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

[Mandalari] Mandalari, AM., Lutu, A., Briscoe, B., Bagnulo, M., and O. Alay, "Measuring ECN++: Good News for ++, Bad News for ECN over Mobile", DOI 10.1109/MCOM.2018.1700739, IEEE Communications Magazine vol. 56, no. 3, March 2018, <https://ieeexplore.ieee.org/document/8316790>.

[Palmei] Palmei, J., Gupta, S., Imputato, P., Morton, J., Tahiliani, M., Avallone, S., and D. Taht, "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>.

[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>.

[Roddav] Roddav, N., Streit, K., Rodosek, G.D., and A. Pras, "On the Usage of DSCP and ECN Codepoints in Internet Backbone Traffic Traces for IPv4 and IPv6", DOI 10.1109/ISNCC.2019.8909187, ISNCC 2019, 2019, <https://ieeexplore.ieee.org/document/8909187>.

[tc-fq_codel] The Linux Foundation, "tc-fq_codel(8) — Linux manual page", Accessed 7 July 2025, <https://www.man7.org/linux/man-pages/man8/tc-fq_codel.8.html>.

[Trammel] Trammel, B., Kuehlewind, M., Boppart, D., Learmonth, I., Fairhurst, G., and R. Scheffenegger, "Enabling Internet-Wide Deployment of Explicit Congestion Notification", Proc Passive & Active Measurement Conference PAM15, 2015, <https://link.springer.com/chapter/10.1007%2F978-3-319-15509-8_15>.

[Ware] 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>.

Appendix A. Support for L4S in Linux fq_codel

As of kernel version 5.16, the linux fq_codel qdisc [tc-fq_codel] supports L4S via the ce_threshold and ce_threshold_selector parameters. Setting ce_threshold to an appropriate value (e.g. "1ms"), and setting the ce_threshold_selector to match ECT1 marked packets (i.e. "0x1/0x3") enables support for L4S.

Author's Address

Greg White (editor)
CableLabs
Email: g.white@cablelabs.com