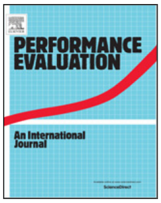
Performance Evaluation 135 (2019) 102036

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva



On the utility of unregulated IP DiffServ Code Point (DSCP) usage by end systems

Runa Barik [a,*], Michael Welzl [a], Ahmed Elmokashfi [b], Thomas Dreibholz [b], Safiqul Islam [a], Stein Gjessing [a]

[a] *Department of Informatics, University of Oslo, Oslo, Norway*
[b] *Simula Metropolitan Centre for Digital Engineering, Simula Research Laboratory, Oslo, Norway*

ARTICLE INFO

Article history:
Available online 26 August 2019

Keywords:
DiffServ
DiffServ Code Point
QoS
WebRTC
Latency

ABSTRACT

DiffServ was designed to implement service provider quality of service (QoS) policies, where routers change and react upon the DiffServ Code Point (DSCP) in the IP header. However, nowadays, applications are beginning to directly set the DSCP themselves, in the hope that this will yield a more appropriate service for their respective video, audio and data streams. WebRTC is a prime example of such an application. We present measurements, for both IPv4 and IPv6, of what happens to DSCP values along Internet paths after an end system has set them without any prior agreement between a customer and a service provider. We find that the DSCP is often changed or zeroed along the path, but detrimental effects from using the DSCP are extremely rare; moreover, DSCP values sometimes remain intact (potentially having an effect on traffic) for several AS hops. This positive result motivates an analysis of the potential latency impact from such DSCP usage, for which we present the first measurement results. We find that routers at approximately 3% of more than 100,000 links differentiate between the WebRTC DSCP values (EF, AF42 and CS1) and consistently reduce delay in comparison with probes carrying a zero value (CS0) under congestion. In contrast, routers at around 2% of these links increase the delay by a comparable amount under congestion, uniformly for EF, AF42 and CS1.

1. Introduction

The Differentiated Services Code Point (DSCP) [1,2] field in the IP header is commonly used for marking and differentiating traffic within a single domain. This is often done at the ingress of a network, and in some cases *within* the network to shape traffic. Egress points are likely to remove or change a DSCP marking. Internet Service Providers (ISP) which do not use or trust the DSCP might set it to zero at the ingress. Accordingly, the DSCP is traditionally not meant to be set by end systems. However, setting the DSCP was found to occasionally work, and there may not be much harm in trying to use this field. It is therefore now proposed as a method for WebRTC [3].

Motivated by [3], we would like to better understand the effect of such DSCP markings. As a first step, it is necessary to understand what happens to the DSCP value along an Internet path. If, for example, a client's default gateway already zeroes the DSCP, no router beyond it can use this field to differentiate packets. The longer into an Internet path a value

[*] Corresponding author.
E-mail addresses: runabk@ifi.uio.no (R. Barik), michawe@ifi.uio.no (M. Welzl), ahmed@simula.no (A. Elmokashfi), dreibh@simula.no (T. Dreibholz), safiquli@ifi.uio.no (S. Islam), steing@ifi.uio.no (S. Gjessing).

https://doi.org/10.1016/j.peva.2019.102036

survives, the more likely it is for the mechanism in [3] to be useful. If setting DSCP values "works", but routers do not currently implement any special treatment for DSCP-marked packets that end systems may emit, there is at least reason to hope that large-scale applications like WebRTC could provoke a change in the behavior of ISPs. Either way, to understand the potential of the mechanism, we must first investigate what routers and other middleboxes do to the DSCP field itself.

Such an investigation was the scope of the earlier version of this work presented in [4], as well as the scope of the related work that we discuss in the next section. We repeat the description of our DSCP-"survival" tests in Section 3 and present results from [4] in Sections 4 and 5. Then, following our conclusion in [4] that analyzing the performance impact would be a logical next step, we turn to a novel analysis of the influence that the DSCP value has on delay in Section 6. We present new measurements where we actively produced congestion, using a tool that we crafted for this purpose. To the best of our knowledge, this measurement makes the present paper the first study in the literature which measures the delay impact that end systems perceive when they set the DSCP to a specific value without any prior agreement between a customer and a service provider. Section 7 concludes.

2. Related work

The increasing popularity of middleboxes has motivated several efforts to characterize their deployment and assess their impact on data plane performance. Medina et al. [5,6] actively probed a set of web servers using TBIT [7] to assess the interaction between middleboxes and transport protocols. Honda et al. [8] developed TCPExposure to test whether TCP options are supported. TraceBox [9] improved over TCPExposure by proposing a Traceroute-like approach to pinpoint routers that alter or discard TCP options. Craven et al. [10] proposed TCP HICCUPS, a tool that reveals TCP header manipulation to both ends of a TCP connection. PATHspider [11] allows for A/B testing of a baseline configuration against an experimental configuration of different protocols and protocol extensions.[1] Other papers focused on investigating specific types of middleboxes, such as web proxies [12], transparent HTTP proxies in cellular networks [13], firewalls and NATs policies in cellular networks [14], and carrier grade NATs [15]. Trammel et al. [16] proposed correlating measurements from diverse vantage points to build a map of middlebox-induced path impairments in the Internet.

So far, only a handful of studies focused on the DSCP field: following a smaller-scale [17] and a one-sided (one side of a measured path is controlled) study [18], Custura et al. [19] analyze the DSCP modification behavior by middleboxes in mobile broadband (MBB) edge networks. MBB networks usually deploy middleboxes that interfere with traffic; the analyses of this paper [19] therefore provide a valuable starting point for further analyses. Both this MBB study in [19] and the broader work in [20] find that the DSCP field is generally often rewritten, and particularly often zeroed, by routers along a path. Specifically, the authors of [20] have identified cases of remarking that seem to reflect a historic interpretation of the header field, sometimes leading to an undesirable result which they call "priority inversion". Even so, the overall recommendation in this and the other related work [4,17–21] is that end systems should use the DSCP.

The analysis of DSCP modification behavior in the present paper complements these measurement studies in prior work in that we focus on heterogeneous fixed networks (i.e. research networks, business-grade connections, consumer-grade connections like ADSL) and investigate IPv4 vs. IPv6. Moreover, to the best of our knowledge, with the exception of our own preliminary analysis of the dataset from [4] in [21], the current paper presents the first analysis of the delay impact of the DSCP, specifically under congestion, which we actively produce.

3. *Fling* measurement methodology

For the first part of this work, we use the *fling*, an active measurement tool that allows testing whether an arbitrary sequence of packets can be exchanged between a client and a server using a pre-defined pcap and json files. We have installed *fling* in a testbed which we call *fling* middlebox measurement platform[2] [22]. For more details, see http://fling-frontend.nntb.no/index.html. For our tests, *fling* platform used 34 IPv4 and 18 IPv6 nodes as servers. Regarding the choice of the measurement hosts, we believe that a typical WebRTC use case is users at the edge calling each other or calling a company's technical support via the browser. Hence, having endpoints at the edge seems to be the right choice. This has motivated us to choose NorNet Core nodes, since they have "consumer-grade" connectivity. We also have a server from Amazon cloud as part of our dataset. We ran the client tool from 160 (IPv4)/65 (IPv6) Ark[3] [23], PlanetLab[4] and NorNet Core[5] nodes to perform a simple UDP packet exchange with different DSCP values. Details of this infrastructure are provided in Section 3.1. Fig. 1 presents the geographical locations of the *fling* clients and servers. This setup gave us measurements across a total of approximately 10k unidirectional IPv4 paths and more than 2k unidirectional IPv6 paths. We use WebRTC to motivate the choice of DSCP values in our study, but consider our results to be applicable to a broader class of applications. To obtain more general results and reduce the potential impact from other factors that might influence e.g. delay or reachability, we used direct UDP probes instead of real WebRTC traffic. We leave measuring WebRTC itself for future work. We tested the DSCP values CS1, AF42, and EF due to their importance for WebRTC (see Table 1). We conducted this measurement campaign once, from the 2nd to the 5th of June 2017.

[1] https://pathspider.readthedocs.io/en/stable/.

[2] *fling*: http://fling-frontend.nntb.no.

[3] Ark: https://www.caida.org/projects/ark/.

[4] PlanetLab: https://www.planet-lab.org.

[5] NorNet: https://www.nntb.no.

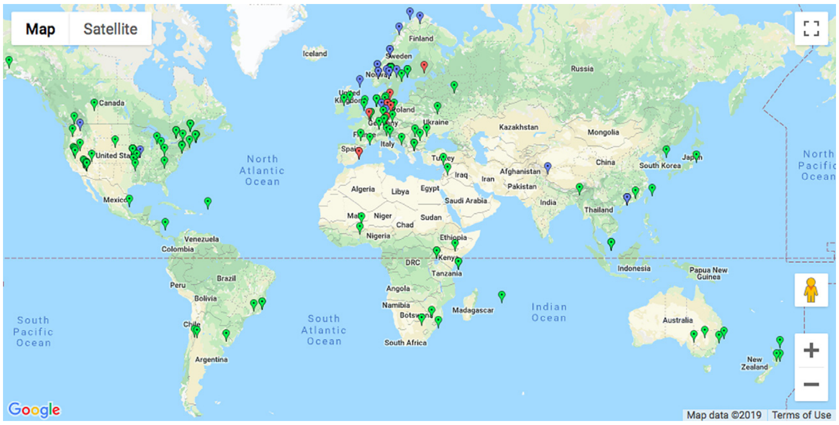


Fig. 1. Geographical *fling* host locations. Green: Ark clients, Red: PlanetLab clients, Blue: servers. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
DSCP values that we encountered in our measurements and their meaning. WebRTC flow types: 1: Audio; 2: Interactive video with or without audio; 3: Non-interactive video with or without audio; 4: Data.

DSCP value	Description (RFC 4594 [24])	WebRTC [3] flow type/Priority
CS1[a]	Low-priority data	Any/Very Low
AF42[a]	Multimedia conferencing	1/Medium or High
EF[a]	Telephony	1/Medium or High
CS0	Standard	Any/Low
AF11, AF12, AF13	High-throughput data	4/Medium (AF11 only)
AF21	Low-latency data	4/High
AF31	Multimedia streaming	3/High
AF41	Multimedia conferencing	2/High
CS4	Real-time interactive	Not defined
CS5	Signaling	Not defined
CS7	Reserved for future use	Not defined
1, 2, 4, 6, 41	Undefined values	Not defined

[a] Values that we used as input.

We designate a test as "failed" if any packet of the test was dropped. To eliminate the effect of sporadic random drops, we only decide that a packet was dropped as a result of a specific DSCP value if it is consistently dropped in three tests. If a packet is dropped or modified on the path, depending on the direction of its traversal, the sender of the packet resends it with increasing TTL (TraceBox-like test — our own implementation of tracebox supporting all types of protocol packets defined in a fling test) to attempt to pinpoint the router that interfered with the DSCP value. In this process, we collect ICMP packets with Time-to-Live Exceeded messages from the network nodes and parse them to see whether they contain the original packet that triggered the ICMP response. If a device remarks the DSCP upon forwarding, we can only observe this behavior correctly by considering the ICMP error message from a node (we hereby call it *responder*) at the next hop on the path.

3.1. Infrastructure

We ran the *fling* client from 111 vantage points on the CAIDA's Ark platform. These vantage points, being located in people's homes, universities and offices, are spread across 44 countries (e.g. 36 from US, 7 from CA, 5 from DE, 4 from ZA, etc.). From these vantage points, we got 111 IPv4 and 46 IPv6 addresses for our measurements. We also received information about the vantage points, such as the Autonomous System (AS) number, organization name, AS classification and geographic locations of the vantage points.

PlanetLab [25] is a group of computers available as a testbed for computer networking and distributed systems research. Its nodes are mostly devices located at universities. We ran the *fling* tool from 14 IPv4 PlanetLab Europe (PLE) nodes (e.g. 8 from DE, 2 from GR, 2 from ES, 1 from FR and 1 from NO), as these allow raw sockets out of the box. PlanetLab Central (PLC) nodes support *safe* raw sockets for ICMP, UDP and TCP packets, but these require binding to a local port which would prohibit receiving ICMP messages that may not carry the correct port number.

The NorNet Core testbed [26] is a large-scale Internet testbed for multi-homed systems. Unlike PlanetLab, NorNet Core also provides support for IPv6. Furthermore, NorNet Core sites are not only connected to a site's local research

Table 2
End-to-end transparency results from 9992 (our work, with vantage points in Ark, PlanetLab and NorNet testbeds) and 9202 ([19], with MBB vantage points in the "MONROE" testbed) IPv4 paths. We combine AF41 and AF42 because they are in the same general category [24].

DSCP value	Paths where DSCP value was preserved (our work)	Paths where DSCP value was preserved ([19], Table V)
EF	23%	23.8%
CS1	30%	24%
AF42/AF41	22%	23.1%

network ISP, but also have "consumer-grade" connectivity with many home-user ADSL and fiber subscriptions. This makes NorNet Core a realistic Internet test platform for experiencing the "normal" user's QoS. Furthermore, we got the possibility to run *fling* directly on the routers of the testbed, providing unrestricted access to the public IP addresses.

We host our *fling* servers on the routers in NorNet Core. We have a total of 31 IPv4 and 18 IPv6 nodes from NorNet Core for *fling* servers, covering 5 countries (21 from Norway, 4 from Germany, 3 from China, 2 from America, 1 from Sweden). We also deployed three additional *fling* servers in the United Kingdom (it has an IPv4 and an IPv6 addresses), India (only IPv4 address), and the USA (only IPv4 address). Further, we ran the *fling* client tool on all the above nodes.

3.2. Data processing

After running the measurements, we collected the data from a total of 160 IPv4 and 65 IPv6 nodes (IPv6 addresses and their corresponding IPv4 addresses belong to the same interfaces). Of the 160 IPv4 addresses, 112 addresses are public and 48 addresses are behind NAT boxes. We extracted all the IPv4 and IPv6 addresses from the ICMP packets obtained from our TraceBox-like test. For IP-to-AS mapping, we used the Whois database provided by Team Cymru[6] and ignored invalid mappings (bogons). To check if multiple IPs belong to the same router while dealing with DSCP, we resolved all IPv4 router aliases using the tools Kapar[7] followed by MIDAR.[8] For IPv6 addresses, we used the tool Speedtrap.[9] We extracted 7721 IPv4 and 1503 IPv6 addresses. After performing alias resolution, we collected 416 routers in IPv4 and 127 routers in IPv6 with multiple addresses. Since we do not carry out our TraceBox-like test when all packets reach the other end unmodified, the number of paths for which we have TraceBox-like information is smaller than the total number of paths. The whole measurement gave us TraceBox-like test details for a total of 8217 unidirectional paths for IPv4, and 1585 for IPv6. We categorize all client nodes from the three platforms into three groups:

1. Home Networks: This covers 39 IPv4 (12 from NorNet Core and 27 from Ark) and 11 IPv6 (7 from NorNet Core and 4 from Ark) addresses of residential nodes.
2. Research and Education Networks: We consider the PlanetLab nodes as part of research and education networks. In this group, in addition to the 14 nodes from PlanetLab, we have 19 nodes from NorNet Core and 38 nodes from Ark (i.e., a total of 71 IPv4 addresses). In the IPv6 case, we have 34 nodes. Of these 34 nodes, 23 nodes are from Ark and the remaining 11 nodes belong to NorNet Core.
3. Commercial Networks: Business, commercial and infrastructure type nodes fall in this group. We have a total of 50 IPv4 and 20 IPv6 addresses from commercial networks. Of the 50 IPv4 addresses, only 4 belong to NorNet Core and the rest is from Ark. Only 1 IPv6 address is from NorNet Core; the remaining 19 IPv6 nodes belong to Ark.

4. Path traversal

First, we analyze the DSCP effect on path traversal from the end-hosts' perspective and compare some of our results to the results published in [19]. The authors of [19] ran the PATHSpider [11] tool in the "MONROE" mobile platform to analyze the DSCP treatment in MBB edge networks. The tool was run from 107 mobile vantage points spanning 12 MBB providers against 86 IPv4 addresses, covering 9202 source–destination pairs. PATHSpider operates quite similar to the TraceBox-based part of fling, making the results comparable. Different from our work, [19] focuses only on IPv4, but also considers TCP, finding only a marginal difference in behavior between TCP and UDP. Table 2 summarizes some findings from our own tests and [19]. Differences between codepoints are notable, but not very large: CS1 survives end-to-end (e2e) in slightly less than a third of the cases, compared to slightly less than a quarter for EF and AF42.

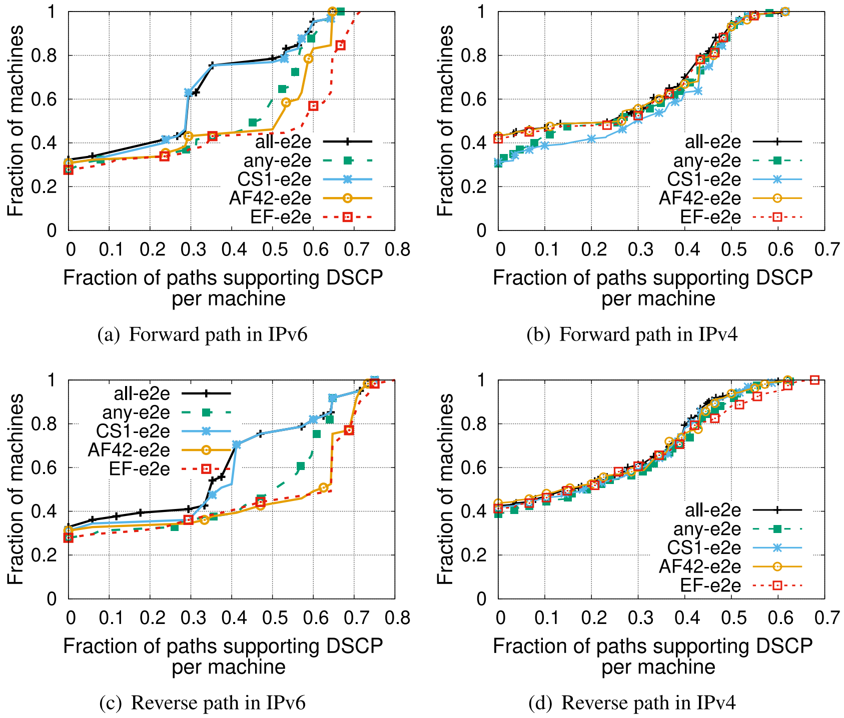
Figs. 2(a) and 2(c) show the cumulative distributions of the fraction of paths per machine supporting a DSCP end-to-end in IPv6 networks in both the forward (client to server) and reverse (server to client) directions. Note that we removed

[6] Team Cymru: https://www.team-cymru.org/IP-ASN-mapping.html.

[7] Kapar, an IP alias resolution tool based on topology inference: https://www.caida.org/tools/measurement/kapar/.

[8] MIDAR, a Monotonic ID-Based Alias Resolution tool: https://www.caida.org/tools/measurement/midar/.

[9] Speedtrap, IPv6 Alias Resolution technique: https://www.caida.org/tools/measurement/scamper/.



(a) Forward path in IPv6

(b) Forward path in IPv4

(c) Reverse path in IPv6

(d) Reverse path in IPv4

Fig. 2. End-to-end DSCP "survival" in forward (client-to-server) and backward direction (server-to-client).

all paths that exhibited two or more instances of DSCP remarking, of which the last remarked it to the original value (e.g., CS1 → AF42 → CS1). In the figures, the lines all-e2e and any-e2e represent a DSCP value surviving on a path for all and at least one of the tested DSCP values, respectively. Approximately 27% of machines observe zero paths where the DSCP survived end-to-end for at least one DSCP value, while approximately 32% of machines observe no support for all the codepoints.

On average, EF and AF42 are much more likely to survive (≈40%) end-to-end than CS1 (≈30%). We found this to be primarily the effect of one AS, AS2116, which consistently mapped CS1 to AF11 on 121 out of our 1170 total IPv6 paths. This AS consistently changed CS1 to AF11, while AF42 and EF remained unchanged.

Figs. 2(b) and 2(d) show the cumulative distributions of the fraction of paths per machine supporting DSCP end-to-end in IPv4 networks in both the forward (client to server) and reverse (server to client) directions. For both plots, reverse and forward paths exhibit similar characteristics, except that CS1's slightly better chance of end-to-end "survival" in the IPv4 case is only visible in the forward direction.

5. DSCP treatment in different network parts

In this section, we investigate how DSCP codepoints are treated in different parts of the network — inside the local network, within and beyond the first-hop ISP (the ISP hosting the vantage point). The goal of this analysis is to determine which parts of the end-to-end path are likelier to remark DSCP codepoints. The total number of client ASes is 118 for IPv4 addresses and 54 for IPv6 addresses. With one exception (noted in Section 5.3), all the behavior described here happened *persistently*, i.e. routers and gateways exhibited the same behavior as they appeared in multiple paths — all measurements were done from multiple clients to multiple servers.

5.1. Will a host's DSCP mark make it to its own ISP?

In the following, we quantify DSCP treatment in the local network. We blame the default gateway for remarking the DSCP if we discover a change at TTL = 1.[10]

We observe that the default gateways of 8 commercial, 11 residential and 9 research clients always reset the DSCP values to 0 (CS0), irrespectively of the input value. These are 17% (28 out of 160: 6 from NorNet, 1 from PlanetLab and 21 from Ark) of the total number of machines with IPv4 addresses. However, it happens for only 10% (7 out of 65: 2

[10] By the default gateway, we strictly refer to the first router hop that connects the vantage point to the rest of the Internet.

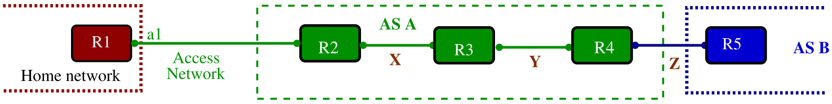


Fig. 3. IP a1 belongs to AS *A* and *R*2 is the access router in AS *A*. We categorize DSCP changes as follows: X (*access network*): DSCP changed by *R*2 and detected by *R*3, Y (*in-network*): DSCP changed by *R*3 ($R2 \neq R3$) and detected by *R*4; Z (*egress*): DSCP changed by *R*4 (AS *A*) and detected by R5 (AS *B*).

Table 3
DSCP change in first-hop ISP; "Mixed" means that ASes changed the DSCP in multiple places (multiple routers remarking the DSCP)

	#ASes (IPv4)	#ASes (IPv6)	#ASes setting DSCP to 0 (IPv4)	#ASes setting DSCP to 0 (IPv6)
DSCP changeat access network	15	3	11	3
DSCP changeat in-network	6	5	5	5
DSCP changeat egress	13	3	9	3
DSCP changeat mixed	6	2	4	1
No change	57	34	0	0
Total	97	47	29	12

from NorNet and 5 from Ark) of the machines with IPv6 addresses. These are 4 commercial, 2 research, and 1 residential clients.

We also observe that some of the gateways persistently reset the DSCP values to a fixed value, irrespective of the incoming DSCP value. For IPv4, the gateways of 2 out of 51 commercial Ark clients change the incoming DSCP values to a decimal value of 63 (unregistered codepoint). Similarly, 2 of the 39 residential Ark nodes detected a change of the DSCP value to CS1 (which commonly encodes "less than best effort" behavior, see Table 1). Only one residential Ark client in an IPv6 network experiences a persistent DSCP re-write; it is always changed to the unregistered decimal value of 4.

Some default gateways generate specific DSCP values depending on incoming values. 2 commercial and 1 residential clients (all Ark) in IPv4 detected different DSCP values depending on input values. For example, for one commercial client we observed that the AF42 (multimedia conferencing, see Table 1) codepoint is changed to CS4 (real-time interactive); EF (telephony) is changed to CS5 (signaling), while CS1 remains unchanged. This mapping could just be the result of bleaching the lower 3 bits (setting the lower 3 bits to zero) of incoming DSCP values. The default gateways of one commercial and one residential node appear to bleach the higher 3 bits of the DSCP. However, we did not see a single case of such DSCP mapping for IPv6 probes.

To summarize, we have seen that default gateways treat the DSCP in a myriad of ways. However, overall, only 21% and 12% of our IPv4 and IPv6 probes experienced a change of the DSCP value at the default gateway. This is encouraging, considering an earlier work that has shown an overall end-to-end bleaching rate of 59% [19]: even with IPv4, the DSCP value often remained intact, and IPv6 makes this success even more likely, possibly giving an additional reason to favor IPv6 over IPv4. Except for one case, DSCP values are typically not demoted. We are left with 125 IPv4 and 57 IPv6 clients that did not experience a DSCP change at the first hop. The number of client ASes for these IPv4 and IPv6 addresses are 97 and 47, respectively.[11] Next, we will look at what happens further "down the road", inside these client ASes.

5.2. *DSCP treatment by first-hop ISP*

We now turn to investigating how DSCP markings, which survive the local network, are treated by the first-hop ISP. Of the remaining 125 IPv4 clients, 67 are affected by their own ISPs. These clients belong to 41% of all measured ASes. The chance is lower for IPv6, with only 20 of the remaining 57 addresses experiencing a change of DSCP in their own ISP. Overall, the majority of first-hop ISPs do not interfere with DSCP codepoints; 59% and 67% for IPv4 and IPv6 respectively. We categorize the DSCP change in the ISP as *access network*, *egress* and *in-network*, depending on the location inside the ISP network, where the change occurs. Fig. 3 explains how we categorize DSCP changes. If DSCP remarking routers in an AS appear in multiple places (i.e., different routers in different places), we call the DSCP remarking in that AS *mixed*.

To study the *access network*, we consider the first public IP after the default gateway to be the first-hop ISP ingress router. This is usually a router that is two hops away from the client i.e. TTL = 2. This mapping can be incorrect if the first-hop ISP configures its access network to use private IPs e.g. in presence of carrier grade NATs. However, our dataset does not involve paths with private IP hops with TTL>1. Table 3 presents the total number of ASes that remark the DSCP at the access network in both IPv4 and IPv6 ISP networks. 15 out of 97 ASes in IPv4 networks remark the DSCP at the access network. Most of these set DSCP to CS0. Four ASes, however, map incoming DSCP to a new value as it enters the AS.

[11] This is larger than the total number of client ASes minus the ASes of machines with an interfering gateway because some client ASes have several clients.

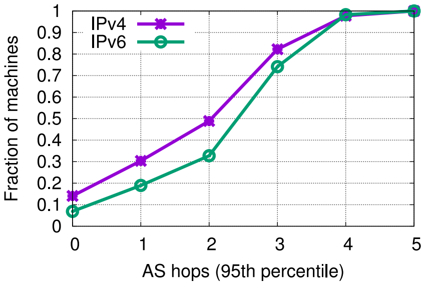


Fig. 4. The number of ASes hops before the DSCP is remarked.

Table 4
A summary of the number of clients' DSCP getting affected in different parts of the network in IPv4 and IPv6.

Parts of the network	#clients in IPv4	#clients in IPv6
Local network	35	8
First hop AS	67	20
Beyond the first hop AS	58	37
Total	160	65

For example, irrespective of all incoming DSCP values, one AS remarks to AF11 (i.e. high-throughput data), while another AS changes to CS1 (i.e. less than best effort) at the access network in IPv4. Another two ASes bleach the higher 3 bits of incoming IPv4 DSCP. This mapping results in unregistered codepoints (other than CS1 and CS0), making the traffic unclassified. IPv6 is different, with only 3 out of 47 ASes remarking the DSCP (all to CS0).

In Table 3, we see that 6 ASes for IPv4 clients remark DSCP *in-network* and 5 of them remark to DSCP 0. Only one AS remarks CS1 and EF to the decimal value of 1, while AF42 is reset to 0. For IPv6, we detect 3 ASes, and all of them reset the DSCP to 0.

We categorize the location of the DSCP change as *egress* when the last router of the client's AS changes the DSCP and the *responder* belongs to another AS. In Table 3, we see that 13 ASes in IPv4 remark the DSCP at the egress, and 9 of them simply reset it to CS0. In case of IPv6, three ASes reset the DSCP to 0 at the egress. In two ASes in IPv4, we see *egress* remarking with the mapping CS1→0, EF→6, AF42→4. One AS changes the DSCP to a decimal value of 2 when it forwards the traffic to another AS. At the egress of yet another AS, we observe a DSCP remarking to the unregistered codepoint 41.

Six out of 97 ASes (i.e., 6%) in IPv4 and 2 out of 47 ASes in IPv6 were categorized as *mixed*. In the IPv4 case, four of them reset the DSCP to CS0. In a particular AS, at the access network for both, IPv4 and IPv6 clients, we see a DSCP change only for AF42 (to AF41), while CS1 and EF remain unchanged.

To understand the importance of marking in the first-hop ISP, we investigate the number of ASes that let a DSCP value "survive" before it first becomes changed. Fig. 4 provides a high-level overview of how long into the path DSCP codepoints remain unchanged. For each client we identify all paths with DSCP changes and the responsible AS-hop (the AS-hop where the DSCP value changes for the first (and possibly only) time. Then we use the 95th percentile of the responsible AS-hop as a measure of where changes typically occur, and subtract 1 from the value for plotting to show the AS number just *before* the AS that changed it. The DSCP markings survive the first hop AS (i.e. own AS) for 70% and 80% of the IPv4 and IPv6 clients, respectively. Notably, DSCP markings by 5% of all clients traverse 4 ASes without being remarked. In IPv4, probes from 50% of home network nodes survive the source's own ASes whereas more than 70% in research and commercial networks survive their own ASes.

To summarize, we found that the most common re-marking behavior of the first-hop ASes (which we assumed to be the clients' and server's own ISP) was to zero the DSCP. Generally, as with the default gateways, chances of DSCP "survival" seem better with IPv6 than with IPv4. Also, with the exception of one AS, it appears that the choice of input DSCP value does not matter: the DSCP either remained unchanged, was zeroed, or changed into undefined values in our tests. Beyond the first-hop AS, 58 of the total of 160 IPv4 clients (IPv6: 37 out of 65) still have their DSCP value intact (see Table 4). This is perhaps already good news for an application that may want to use the DSCP. Even more importantly, we have seen that a chosen DSCP can sometimes traverse several ASes before first becoming remarked. This also means that the DSCP could potentially be quite useful on short paths (e.g., within the same ISP).

5.3. DSCP treatment beyond the first hop AS

In the following, we investigate the cases where DSCP markings survived past the first-hop ASes. The total number of ASes detected in our measurement is 298 for IPv4 addresses and 119 for IPv6 addresses. Excluding the clients' ASes from our measurement data, we found 180 ASes in IPv4 and 65 ASes in IPv6 core networks, respectively. We find that about 32% of the core ASes remark the DSCP codepoints. This percentage is consistent for both IPv4 and IPv6.

Table 5
An example showing the DSCP values (CS1, AF42, and EF) changing multiple times on a path.

TTL	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ASes	35 432	35 432	1299	174	174	174	174	174	680	680	680	680	680	680	680
CS1 (8)	8	8	8	8	18	18	18	18	8	8	8	8	0	0	0
AF42 (36)	36	32	32	8	18	18	18	18	8	8	8	8	0	0	0
EF (46)	46	40	40	8	18	18	18	18	8	8	8	8	0	0	0

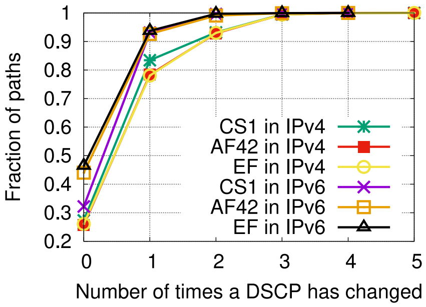


Fig. 5. The number of times the DSCP values have changed over all the IPv4 and IPv6 paths.

On some paths, DSCP values were changed multiple times. Table 5 shows an example where they were changed particularly often, for a client from AS35432 that sent the codepoints CS1, AF42 and EF to a server in AS680. On this path, we saw the following decimal values of codepoints: 0 (CS0); 8 (CS1); 18 (AF21); 32 (CS4); 36 (AF42); 40 (CS5); 46 (EF). The table also presents the list of ASes whose routers respond with a codepoint received (inside the ICMP payload) over different TTL values. We explain the DSCP treatments by different ASes on the path in the following:

- [TTL 1-2]: At the default gateway, we observe a mapping EF→CS5, AF42→CS4, CS1→CS1 (remarking lower 3-bits to 000).
- [TTL 2-3]: No DSCP change was detected at the *egress* of the client's own AS (AS35432, CABLENET-AS, CY).
- [TTL 3-4]: At the *ingress* to AS1299, we detect a DSCP change to CS1(CS4→CS1, CS5→CS1).
- [TTL 4-5]: At the *ingress* to AS174, the codepoints were changed to AF21.
- [TTL 8-9]: We observe a DSCP change to CS1 at the *egress* to AS174.
- [TTL 12-]: At an *in-network* router of AS680, DSCP was modified to CS0.

From the table, we observe that CS1 has changed 3 times while both AF42 and EF have changed 5 times.

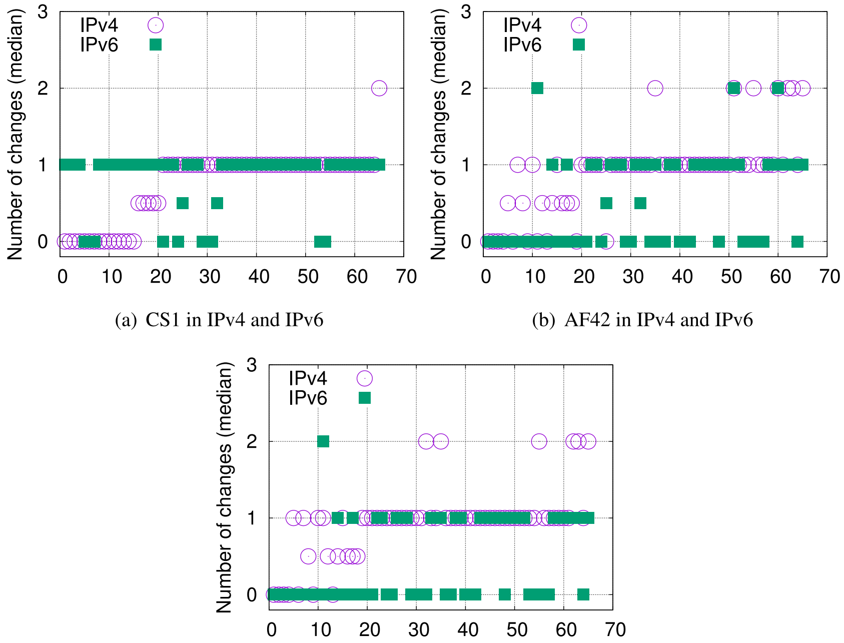
Fig. 5 plots the fraction of paths both in IPv4 and IPv6 networks exhibiting the number of times a given codepoint (CS1, AF42 and EF) has changed over them. 25%–27% of the paths in IPv4 networks do not change the DSCP codepoints whereas in IPv6 networks, AF42 and EF survive end-to-end on ≈44% of the paths, while CS1 survives on ≈32% of the paths.

We then compare the treatments of the codepoints inside the core networks for the clients that have both IPv4 and IPv6 addresses (a total of 65 clients). For each client, we consider all the forward paths and compute the number of modifications to each DSCP codepoint on each path. Fig. 6(a) presents the dot plot of the medians of the number of modifications for CS1 for all the clients with IPv4 and IPv6 addresses. The clients are sorted on the basis of increasing order of their median values for IPv4 paths. Figs. 6(b) and 6(c) represent the dot plots for codepoints AF42 and EF. 36 clients with EF and 34 clients with AF42 have zero median in IPv6 paths while only 10 clients with CS1 have zero median. This shows clients with EF or AF42 are more likely to remain unchanged. However, IPv4 clients show a very different picture: only 7–15 clients with all codepoints observe no change in the DSCP values.

We collect and categorize the remarking policies followed by all the ASes that lie between the client and server ASes. Table 6 presents all the identified DSCP policies along with the number of ASes that have exhibited them. The ASes in which a certain DSCP policy is applied are not mutually exclusive; for example, AS1299 exhibited seven different policies R-1, R-2, R-5 (CS0, CS1, AF11, AF21 and 2).

Looking closer at individual ASes, we find evidence that some ASes implement policies based on AS relationships. For example, Cogent marks traffic from its peers as AF21 (low-latency data) and traffic from its customers as AF11 (high-throughput data).[12] Other ASes, however, do not seem to account for business relationships when remarking packets. In fact some ASes, like Telia, can remark packets from the same origin AS differently, which hints at an absence of a customer- or peering-agreement-specific' remarking policy.

[12] This marking behavior appears to be in line with a description given at https://groups.google.com/a/measurementlab.net/forum/#!msg/discuss/vcQnaZJO6nQ/ltfi_3Aif9gJ; this statement is, however, relatively old, and we have no evidence that this policy is still in place.

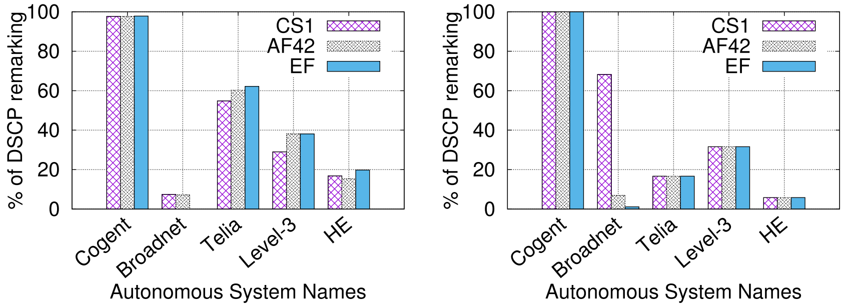


(a) CS1 in IPv4 and IPv6

(b) AF42 in IPv4 and IPv6

(c) EF in IPv4 and IPv6

Fig. 6. Each dot represents the *median* of the distributions of the number of times a given DSCP has changed over all the forward paths from a host.



(a) IPv4

(b) IPv6

Fig. 7. Percentage of DSCP remarking in ASes.

Table 6
DSCP policies. In the following, policy R-5 is indicated by the name of the codepoint — e.g., policy "CS0" means a remarking of "all→CS0". 32% (58 out of 180 ASes in IPv4 core networks and 22 out of 65 ASes in IPv6 networks) have DSCP policies. We detected 11 258 paths where DSCP policies are applied in the core IPv4 networks for all codepoints.

DSCP policy	Description	#ASes (out of 58)	Remarking paths in IPv4 core
R-1	Remark higher 3-bits to 000 (e.g., AF42→4)	18	20.3%
R-2	Remark higher 3-bits to 001 (AF42→AF12)	5	13.1%
R-3	Remark higher 3-bits to 010 (AF42→AF22)	2	0.1%
R-4	Remark lower 3-bits to 000 (AF42→CS4)	2	0.3%
R-5	Others e.g. all→ {AF11, AF21, AF31, CS1, CS0, CS4, CS7, 1, 2, 4, 6}	48	66.2%

Figs. 7(a) and 7(b) present the percentage of DSCP remarking by routers in a set of ASes for IPv4 and IPv6 topologies, respectively. These ASes are the top 5 ASes in each topology in terms of number of measured paths that cross them after removing educational networks. We remove education networks as we would like to focus on prominent transit providers.

Table 7
DSCP policy in Level-3.

DSCP Policy	Neighboring ASes (observed)	
	In IPv4 networks	In IPv6 networks
R-1	1299, 8966	1299
R-2	1299, 12 956, 3701	–
CS0	5511, 6079, 6774, 29 695, 30 950	6774, 8422, 29 695, 30 950
CS1	680, 2119, 7922	680
AF11	2495	2495
AF21	174, 3741, 2200	–
CS7	34 224	–

Thus, the top-five networks include four major global transit providers: Level-3 (AS3356), Telia (AS1299), Cogent (AS174), and HE (AS6939). In addition, they include Broadnet (AS2116), which is a major Norwegian transit provider. Broadnet is present on the top-five list because most of the NorNet servers and clients are located in Norway.

For each AS, we plot the percentage of DSCP remarking for each input DSCP value (CS1, AF42 and EF). Cogent remarks almost all IPv4 and IPv6 packets, while HE allows most marked packets to pass, especially for IPv6. Telia is more aggressive for IPv4, but Level-3 demonstrates a comparable chance of remarking for both IPv4 and IPv6. Finally, Broadnet allows most packets to pass, except that it always remarks IPv6 packets with a CS1 codepoint, which get remarked. Next, we provide more details about the remarking policies of these ASes.

5.3.1. Cogent

Cogent remarks all incoming packets, except for 3%, to either AF21 or AF11. The unchanged 3% correspond to the case where Cogent receives packets with AF21 and AF11 codepoints, i.e. the same values that are used when remarking. For example, AS2914, AS701 remark any DSCP values to AF21 at the egress before forwarding the packets to Cogent. We also detect the same codepoint AF21 when packets traverse from Cogent to AS220 or AS4538.

5.3.2. Broadnet

In IPv6 networks, we observe only *ingress* and *in-network* DSCP remarking in *Broadnet* — no DSCP remarking is observed at the egress. 68% of packets with CS1 are remarked; these packets are mainly traversing from AS2603 which is an educational network interconnecting Nordic countries. Some routers inside AS2116 networks also remark CS1 to AF11 or CS0. However, in IPv4 networks, we observe only *in-network* DSCP remarking with CS1 to AF11 or CS0, AF42 to CS0 and EF remains unchanged. Incoming packets with DSCP value CS0 traverse this AS unchanged.

5.3.3. HE

HE seems to remark packets as they exit its network. There is no consistent remarking policy for all affected neighbors. However, the low extent of remarking and the lack of a consistent policy hint that these remarkings are possibly not carried by HE but rather by its neighboring ASes. This in fact stems from a limitation in our methodology for identifying which end of a link has actually manipulated the DSCP codepoint. For example, in Fig. 3, R5 may remark the DSCP codepoint, for packets received from R4, before sending the ICMP response. Consequently, we blame the remarking on R4.

5.3.4. Telia

Telia appear to remark packets exchanged with 48 neighboring IPv4 ASes out of a total of 83 that are visible in our dataset, as well as 4 out of the 28 visible IPv6 neighbors. We count a total of 8 remarking policies that happen at both egresses and ingresses. This large number of inconsistent policies highlights the need for further investigations to determine whether Telia can actually be blamed for all the remarkings.

5.3.5. Level-3

Table 7 presents the DSCP policies along with the neighboring ASes in IPv4 and IPv6 networks. For *Level-3*, we only observe remarkings at the egress affecting 17 out of 65 IPv4 neighboring ASes and 7 out of 23 neighboring IPv6 ASes. The marking policies are consistent for neighbors that are present in both the IPv4 and IPv6 topologies, but inconsistent across neighbors. As for HE, this observation hints the measured remarkings are likely to be carried out by Level-3 neighbors.

In summary, the measured ASes employ a diverse set of remarking policies. Accordingly, it is unclear whether transit ASes are actively leveraging the DSCP codepoints in signaling. One AS — Cogent, however, seems to follow a specific remarking policy depending on whether packets traverse from a customer or a peer AS. We also observe that both Level-3 and HE do not seem to implement a remarking policy. This is good news since Level-3 and HE are the largest transit providers for IPv4 and IPv6, respectively.

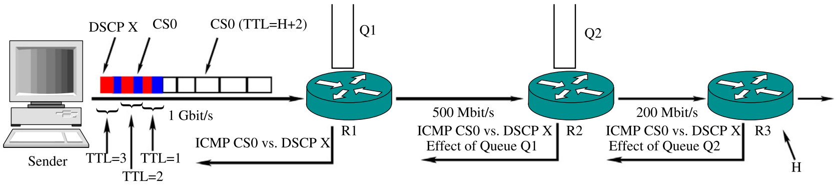


Fig. 8. FloodBox operation. The sender transmits a CS0 burst, followed by a pair of CS0 packets and DSCP X probes per TTL value.

6. Latency performance evaluation

So far, we have investigated the survivability of DSCP values. Survivability – at least up to a certain number of hops – is a necessary but not a sufficient condition for the DSCP value to be useful. To understand whether using a nonzero DSCP value can benefit end hosts, we measure its impact on delay. For this, we need to produce congestion, and we should be able to obtain measurements per hop to reduce the impact of noise from multiple traversed routers. We now present FloodBox,[13] our extension of TraceBox [9] that produces congestion via traffic floods (in bursts).

6.1. Measurement tool

We assume that, in line with [24], routers implement QoS using congestion-responsive elements such as token buckets, Active Queue Management (AQM) and DSCP-based scheduling of packets from queues that grow upon congestion. Then, WebRTC traffic could benefit from using the DSCP codepoints only while the routers face congestion. We therefore implemented a function in FloodBox that creates congestion in the Internet path to measure the delay impact. Using the tool, we first measure the number of hops H that a particular codepoint survives (similar to our use of TraceBox in Section 3). Then, we transmit several "congestion probes" towards the destination, TTL-limited to avoid flooding the final destination. Because we cannot know if the router at hop H possibly reacts upon the DSCP before changing it, we need to probe up to $H + 1$. We use $H + 2$ instead as a limit to increase the chance of obtaining useful measurements in case of minor routing deviations. The idea is to transmit bursts that should make a queue grow, followed by measurement packets that should notice the DSCP delay impact.

Each probe consists of a burst of 200 UDP packets of size 1500 bytes (including IP header), containing random data with CS0. This burst is immediately followed by 2*(H+1) TTL-limited UDP packets (one type for codepoint CS0 and one type for the tested DSCP value). For example, if we find DSCP value AF42 surviving for 2 hops along a 7-hop path, we send 6 such packets after the burst: CS0 with TTL = 1, AF42 with TTL = 1, CS0 with TTL = 2, AF42 with TTL = 2, etc. (as shown in Fig. 8). We repeat these complete congestion probes 20 times in order to create enough congestion in the network path. This also means for each DSCP test, each router on a path gets a maximum of 40 packets which may cause it to generate ICMP "Time-to-Live Exceeded" messages. In the background, we run *tcpdump* which collects these ICMP messages. Note that, while routers may not always emit these messages and might sometimes delay sending them, they would only spoil our measurements if they consistently delay them differently for different DSCP values.

Fig. 8 depicts our approach of computing link delay improvements due to different codepoints. For example, on the link R1–R2, we compute the link delay improvement for a particular DSCP value X as follows:

$$\delta_{\mathrm{R1-R2}}(X) = \mathrm{RTT}_{\mathrm{R2}}(CS0) - \mathrm{RTT}_{\mathrm{R2}}(\mathrm{X}) - \mathrm{RTT}_{\mathrm{R1}}(CS0) + \mathrm{RTT}_{\mathrm{R1}}(\mathrm{X})$$

where $\mathrm{RTT}_{\mathrm{R}i}(X)$ is the round-trip time from the sender to the router Ri, obtained using the UDP packet carrying DSCP X in the forward path and ICMP "Time-to-Live Exceeded" message in the reverse path. FloodBox only uses data from paths that have routers in the same TTL sequence for CS0 and DSCP X, and we ignore the routers that appear in non-adjacent TTLs marking them as part of invalid links.

By sending these bursts at high speeds, we increase the probability of producing the desired congestion downstream in the network. A burst can at best produce transient congestion at an Internet router. This is not always guaranteed to have an effect — active queue management algorithms, for example, operate as a function of an average queue length, and might not react to a short-term burst. However, the specifications of EF [27] and LBE [28] recommend using a scheduler to discriminate traffic belonging to these DiffServ classes. It therefore seems likely that, when traffic is scheduled into separate physical queues, even a short burst could experience a delay effect.

Following the implementation of Iperf,[14] FloodBox uses a UDP socket to transfer multiple packets with a fixed TTL and a DSCP value of CS0 (configured using *setsockopt* system call) from a single buffer, as shown in Fig. 9. For the measurement packets, we create a raw socket and craft packets at layer 2 with specific DSCP values and varying TTLs.

[13] FloodBox: https://github.com/runabk/tracebox.

[14] Iperf: https://iperf.fr.

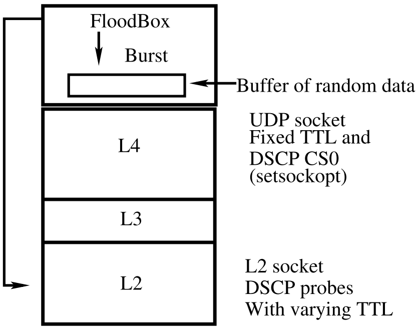


Fig. 9. FloodBox architecture. FloodBox uses two types of sockets: an L4 socket, responsible for generating fixed TTL and CS0 UDP packets, and an L2 socket, responsible for specific DSCP values and varying-TTL UDP packets.

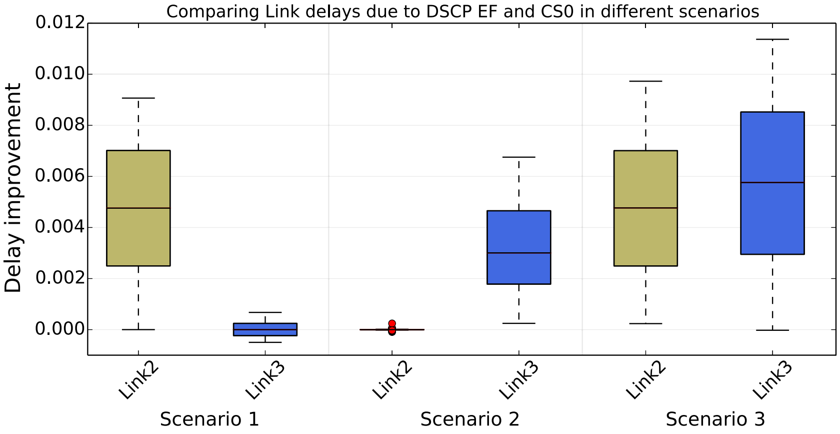


Fig. 10. Link delay improvements (s) in the local test. **Scenario 1**: for Link2 at R1, 950 Mbit/s with Cake, and for Link3 at R2, 1 Gbit/s with pfifo_fast; **Scenario 2**: for Link2 at R1, 1 Gbit/s with pfifo_fast and for Link3 at R2, 950 Mbit/s with Cake; and **Scenario 3**: for Link2 at R1, 950 Mbit/s with Cake, and for Link3 at R2, 900 Mbit/s with Cake.

To see whether our tool is able to produce congestion at the required speed, we carried out a local test across two routers, connected with 1 Gbit/s links (this would be fast enough for our measurement, as the NorNet nodes that we use in the next section were connected with 1 Gbit/s as well). A test with Iperf showed that, using our Linux routers and user-space software on the sender side, we were able to achieve throughput of around 976 Mbit/s. Linux routers now implement QoS based on DSCP codepoints with "Cake" (see [29] for details). We used the default settings of Cake, and used DSCP codepoints EF and CS0 in FloodBox for our local tests. The local testbed consists of three routers R1, R2 and R3; we name the links Link2 for R1–R2 and Link3 for R2–R3. Below we evaluate our tool for three different scenarios.

(1) A bottleneck at the first router, achieved by limiting its outgoing link speed to 950 Mbit/s, using Cake. The second router is fully connected with 1 Gbit/s and uses a DropTail queuing discipline. This tests the basic ability of our tool to identify different delays due to the DSCP value choice.
(2) A bottleneck at the second router, achieved by limiting its outgoing link speed to 950 Mbit/s, using Cake. The first router is fully connected with 1 Gbit/s and uses a DropTail queuing discipline. This tests the ability of our tool to distinguish latency values at a later bottlenecked hop at this high speed. The expected behavior of this test is equal to the first (but identifying the latency difference at a different router).
(3) A bottleneck at the first router, achieved by limiting its outgoing link speed to 950 Mbit/s, using Cake, and an additional bottleneck at the second router, achieved by limiting its outgoing link speed to 900 Mbit/s, using Cake. This tests the ability of our tool to correctly identify different DSCP treatment from multiple hops at high speeds.

Fig. 10 shows that our tool was able to clearly distinguish bottlenecked links. While we see that, at such high speeds, distributions of our 100 tests begin to overlap in the first scenario in the figure. However, the 25–75 percent quartile ranges are still distinct; the second scenario showed no overlaps at all. The third scenario with two bottlenecks shows values in the same rough range, which is the actual determined per-router effect of EF support (first and second scenarios show that per-router latency changes are indeed separately detected).

Table 8
Vantage points of FloodBox.

Country	Number of nodes	Provider	Type of link	Node type
Norway	9	Uninett	Fiber, research	NorNet
Germany	1 1	DFN Amazon.com	Fiber, research N/A	NorNet EC2
USA	1	Amazon.com	N/A	EC2
Canada	1	Amazon.com	N/A	EC2
Brazil	1	Amazon.com	N/A	EC2
India	1	Amazon.com	N/A	EC2
Singapore	1	Amazon.com	N/A	EC2
China	1	CERNET	Fiber, research	NorNet
Korea	1	KREONET	Fiber, research	NorNet

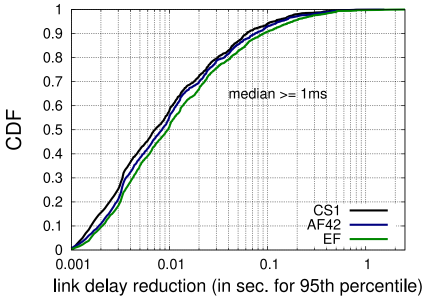


Fig. 11. Latency reduction for DSCP values CS1, AF42 and EF on links that saw an improvement above 1 ms. Delay values are a hint of an effect and do not relate to the absolute end-to-end delay, as explained in the text.

6.2. Performance evaluation in the internet

For our tests, we obtained 52,112 ASes using the Whois database (IP-to-AS mapping) from the dataset collected in [30] and retrieved one IP address per AS from this dataset. To these destination IPs, we ran FloodBox from 12 NorNet nodes and 6 Amazon EC2 nodes (shown in Table 8), using interfaces with link speeds of 1 Gbit/s. Because of the large amount of traffic (more than 1.25 TB) that we produced (which would have resulted in high costs for the Amazon nodes), we were forced to stop some measurements earlier than others, but we ensured that every one of the 18 source hosts carried out tests towards at least 10,000 destination IP addresses out of our set of 52,112.[15] While intentionally producing congestion inevitably causes harm to other traffic that would traverse the same routers at the same time as our tests, we restrain the potential negative effect from our measurement on other traffic by TTL-limiting all packets, including the initial burst. Moreover, we transferred the data strictly from high-speed links, where congestion would quickly disappear. After the measurement campaign, we received one complaint regarding a falsely triggering firewall, which we consider a minor reaction given the total amount of traffic that was produced.

A FloodBox test begins by measuring H as described in Section 6.1, using CS1. For each destination IP, we stored the router addresses encountered on the paths. This allowed us to reduce the number of total tests (thereby limiting congestion incurred) as follows: for every new destination IP, we ran FloodBox only if we observed at least one router address that is not contained in the set of stored addresses. We stopped the measurement after collecting around 100,000 unique links (unique pairs of router IP addresses that responded with consecutive TTLs in the combined data from all our source hosts). Specifically, we ended up with 111,874 links and 50,077 invalid links (router pairs with a TTL gap greater than one). Our measurement traffic traversed 211,928 unique Internet paths.

In around 95% of links, the link delay difference δ falls close to zero for any of the three DSCP codepoints. We therefore analyzed the links with δ more than 1 ms or less than -1 ms. We observed a median latency improvement of more than 1 ms with 2193 (2%), 1994 (1.8%) and 1820 (1.6%) links for EF, AF42 and CS1 respectively. Some of these links overlap; the total number of links where any codepoint led to a latency reduction of more than 1 ms is 3540 ($\approx$3% of all links). The gain distribution of the 95th percentile is depicted in Fig. 11, and it clearly shows that EF sees the largest gain, followed by AF42, followed be CS1. With the exception of CS1 seeing any gain at all (rather than a delay increase, as this is scavenger-type traffic), this matches expectations of how DiffServ QoS could be implemented. In Fig. 11, we can see that the top 40%–50% of samples show a link delay improvement by more than 10 ms. We could also notice that the top 10% exhibit a link delay improvement by more than 100 ms.

[15] pcap files of all the outgoing traffic except for the initial burst per congestion probe (200 packets per probe, i.e. 20 × 200 = 4000 packets per test) and incoming ICMP responses are available from: http://fling-frontend.nntb.no/research/tools/floodbox-traces.tgz.

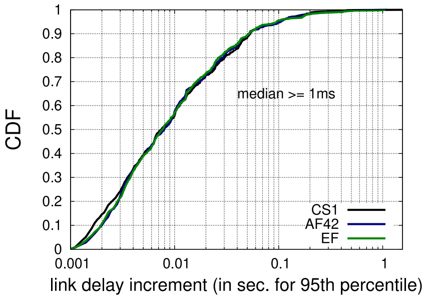


Fig. 12. Latency increment for DSCP values CS1, AF42 and EF on links that saw an increase above 1 ms (1% of all links). Delay values are a hint of an effect and do not relate to the absolute end-to-end delay, as explained in the text.

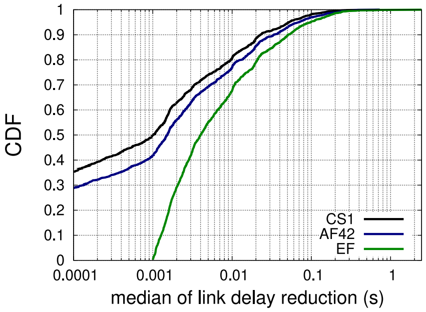


Fig. 13. Latency reduction for DSCP values CS1, AF42 and EF on links that saw an EF improvement above 1 ms (2009 links for all codepoints). Delay values are a hint of an effect and do not relate to the absolute end-to-end delay, as explained in the text.

On 1064 (0.9%), 1048 (0.9%) and 1124 (1%) links, the median per-link latency was increased by more than 1 ms for EF, AF42 and CS1 respectively. Some of these links overlap; the total number of links where any codepoint led to a latency increment of more than 1 ms is 2210 (≈2% of all links). From Fig. 12 we can see that, at some links, codepoints appear to be treated worse than CS0. The number of links where this happens is around half as large as in the positive case, and the magnitude of the impact is roughly similar: while in the positive case, at around 70% of the links the delay was reduced by 20 ms or less, in the negative case, at around 70% of the links it increased by 20 ms or less.

In order to better understand differential treatment, we considered the codepoints in all the cases where the delay improvement with EF was greater than 1 ms (i.e., we looked at a set of an equal number of links: 2009 for EF, AF42 and CS1), depicted in Fig. 13. The difference is significant: in at least 30% of these cases, there was no improvement at all with AF42 or CS1.

Even when we notice a difference between EF, AF42 and CS1, the absolute improvement numbers in the previous figures are quite small. At this point, we remind the reader that these numbers were produced with TTL-limited probes following our self-produced "congestion" packet trains from 1 Gbit/s links. The goal was to identify whether there is any special DSCP-based delay impact at all, and whether there is differential treatment for the three codepoints; the absolute delay gain that we report is less meaningful, as it may not match what an application sees under a practical congestion situation. This also makes it less interesting to further analyze the case of increasing delay, as the lines in Fig. 12 overlap, meaning that this detrimental behavior was the same for all DSCP values.

The absolute delay impact may differ greatly when the extent of congestion is larger. Also, we avoided doing end-to-end measurements in order to reduce noise; however, paths may contain multiple congested links at DSCP-supportive routers. To see whether this is likely, we checked the number of DSCP-supportive (more than 1 ms delay gain over CS0) routers on the total distinct paths in our measurements. 24 798 (11%) of the 211,928 measured Internet paths have at least one link that reduces the latency due to the DSCP values. Of these paths, 97.6% contained only one of the latency-affecting links; thus, it generally seems unlikely that WebRTC traffic would experience a DSCP effect from more than one router. We then found that this link was the same (only 3 hops away from the source host) on close to half of the paths. Removing these cases leaves us with approximately 6% of the total number of paths.

To further understand the importance of our per-link results, we analyze how they are spread across ASes. As before, we only consider cases where delay was reduced by at least 1 ms, and to simplify the discussion, we limit our investigation to the most influential codepoint (EF); this gives us a total of 550 ASes with the distribution of links per AS shown in Fig. 14.

As we can see, 60% of ASes contain only one of these links, and the tail end of the distribution (upper 10%) contain more than 6 links, ranging up to as many as 122 in one extreme case. We categorize the links as *ingress*, *egress* and *in-network*,

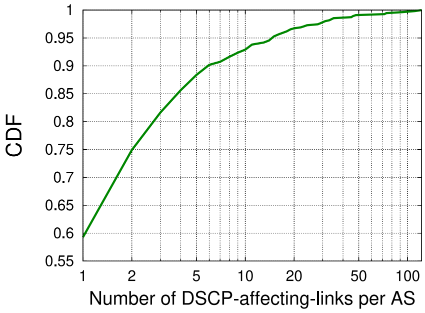


Fig. 14. Distribution of links reducing delay for DSCP (EF) per Ases on links that saw an EF improvement above 1 ms.

Table 9
Behavior of top ASes, sorted backwards by the number of EF-reactive links. Ingress, In-network and Egress columns show the number of links with median delay in brackets.

ASes	Ingress	In-network	Egress
AS4134 (Chinanet-Backbone)	0	120 (1.7 ms)	2 (2.5 ms)
AS7018 (ATT-Internet4)	0	104 (23.9 ms)	7 (1.2 ms)
AS12389 (Rostelecom)[a]	61 (5.6 ms)	38 (5.2 ms)	18 (2.7 ms)
AS6939 (Hurricane)[a]	9 (20 ms)	47 (7 ms)	22 (11 ms)
AS6762 (Seabone-net Telecom italia)[a]	49 (5 ms)	15 (3 ms)	20 (1 ms)
AS1239 (Sprintlink)	0	45 (3.3 ms)	3 (1.6 ms)
AS7922 (Comcast)	1 (4.4 ms)	2 (2.3 ms)	32 (3.7 ms)
AS4775 (Globe-Telecom)[a]	9 (54 ms)	29 (49 ms)	0
AS6461 (ZAYO)	3 (10 ms)	15 (49 ms)	17 (5 ms)
AS3257 (GTT-Backbone)	5 (2.6 ms)	9 (4.7 ms)	17 (3.8 ms)
AS16509 (Amazon-02)	0	29 (19 ms)	2 (6 ms)
AS15412 (Reliance Globalcom)	9 (5 ms)	17 (5 ms)	3 (7.5 ms)

[a] Some links in these ASes were detected as Ingress *and* In-network.

Table 10
Behavior of ASes that were encountered in both FloodBox and *fling* measurements (see Section 5.3).

ASes	Ingress	In-network	Egress
AS3356 (Level-3)	9 (4.5 ms)	4 (7.8 ms)	16 (6 ms)
AS1299 (Telia)	1 (1.5 ms)	2 (24 ms)	10 (2.6 ms)
AS174 (Cogent)	2 (1.3 ms)	0	0
AS2116 (Broadnet)	2 (1 ms)	1 (3.4 ms)	0

depending on the location of the link in the AS where we observe a latency reduction. For example, we consider a link as *ingress* when a router – through which the traffic enters that AS from another AS or Internet exchange point – attached to the link reacts to codepoints along the forward path. Similarly, we call a link as *egress* when the routers attached to the link belong to different ASes. Table 9 provides an overview of the top ASes in our distribution, sorted by the number of DSCP-reacting links — i.e., the AS in the first line in the table contains the largest number of links that reduced the delay upon seeing EF.

The AS-level investigation also showed some ASes (Table 10) that we encountered in our earlier measurements in Section 5.3. Our FloodBox measurement confirms the *fling* measurement: earlier, we discussed that Cogent only remarked the DSCP field in the ingress; matching this behavior, we only see a delay effect from EF in Cogent's ingress. Broadnet remarked in the ingress and in-network, and here, we also see a delay effect in the same places. Level-3 and Telia appear to have a common DiffServ deployment where the egress (and, for Telia, also the ingress) remarks, and all three types of routers react to the EF codepoint. We also detected 61 out of 2193 links that belong to Internet exchange points, reducing the latency upon seeing the DSCP codepoint EF.

Concluding our AS study, Fig. 15 offers a look at how "deep" inside the network most of the EF delay benefit occurred — e.g., was it always at the source host's ISP? The lines for 2, 3, 4 and 6 AS-level hops mostly overlap, meaning that the delay effect occurred at various distances. We saw an upper limit of 6 ASes (with only 53 links at such a large distance). The diverging lines for 1 and 5 hops only represent a relatively small number of links; the large delay effect at 1 hop might be a result of the FloodBox burst being perfectly intact only at these distances and diluted at further distances — it serves as a reminder that the absolute delay change that we report is less meaningful than the number and location of such changes.

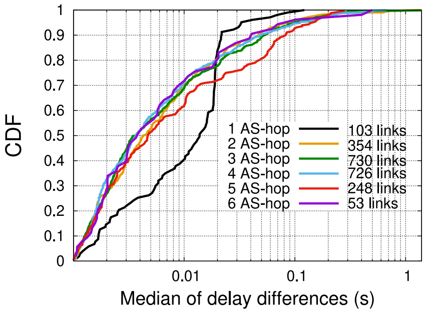


Fig. 15. Distribution of link delay differences for different AS-hops.

7. Conclusion

We set out on this investigation to answer whether a mechanism like the proposed QoS scheme for WebRTC [3] can work, in the hope of being able to give implementation advice. We were, in fact, afraid that this advice might turn out to be complex: different input DSCP values can provoke different output DSCP values, and so highly detrimental things might happen, e.g., a low-latency request turning into lower-than-best effort marking. We found that such behavior hardly exists. Generally, the DSCP was kept intact, zeroed, or statically set to a certain value irrespective of the input. When the output *was* indeed a function of the input, the result was usually one of a set of undefined values, rather than a clear change of the expected semantics. From our results, but also concurring with previously published work, we conclude that the DSCP is often changed, and particularly often zeroed. There is, however, hardly any evidence of a DSCP choice being counterproductive.

We followed our investigation of DSCP "survival" with a study of how the DSCP value can influence delay. Using our FloodBox tool that actively produces congestion and investigates delay changes on different links inside the network, we were able to find a consistent delay improvement on around 3% (in case of EF, slightly less for AF42 and CS1) of measured links, which occurred on 11% of all paths and belonged to 550 ASes. By "consistent", we mean that the delay improvement over CS0 was seen in at least 20 consecutive measurements where we always sent pairs of the tested DSCP value and CS0. With the large majority of paths containing only one of these links, and around 60% of ASes containing only one of them, the likelihood of seeing a DSCP delay effect from more than one router on a path is small. When there was an effect, EF tended to work best, followed by AF42, followed by CS1 — this matches expectations, except for the better-than-normal delay with CS1. There were also cases where a non-zero DSCP choice increased the delay, but without differentiating the three DSCP values and only on 2% of the links. This is only the result of our specific measurement traffic and may look different for specific payload when devices carry out deep packet inspection (DPI).

We conclude that, while the latency effect is generally limited and will only rarely occur, there *is* a measurable delay gain (but also a slight risk of a disadvantage) from using the DSCP, although our study does not quantify the amount (our measurements only provide a hint that the effect exists). In particular, it can be beneficial for latency-critical applications to mark their traffic as EF. We recommend that such applications incorporate tests to fall back to CS0 or a different value upon consistent loss. We saw that the number of ASes that a value "survives" can be significant, with different ASes implementing different DSCP policies, and the delay impact can occur at a significant distance from the source (up to 6 AS hops in our measurements). Thus, if an application has a choice between paths, possibly traversing different ISPs (as it is the case for our NorNet Core nodes), it is worth testing them to see where the DSCP works better. Finally, we recommend to favor IPv6 over IPv4 if the DSCP is to be used: on IPv6 paths, the DSCP is much more likely to remain intact.

The Transport Services (TAPS) Working Group of the Internet Engineering Task Force (IETF)[16] is currently developing a new API for applications to use [31,32]. This API decouples applications from the underlying transport protocols and their specific features, offering only the services that are relevant to applications themselves. Using TAPS, an application can choose a service, which a TAPS system may map onto a specific DSCP value choice together with fall-back behavior as described above. This functionality is already implemented in the open-source TAPS library NEAT [33].

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.peva.2019.102036.

[16] https://datatracker.ietf.org/wg/taps/.

Acknowledgments

This work was partially funded by the European Union's Horizon 2020 Research and Innovation Programme through the NEAT project under Grant Agreement No. 644334. The views expressed are solely those of the authors.

References

[1] D. Black, P. Jones, Differentiated Services (Diffserv) and Real-Time Communication, Informational RFC 7657, IETF, ISSN 2070-1721 (Nov. 2015).
[2] K. Nichols, S. Blake, F. Baker, D.L. Black, Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, Standards Track RFC 2474, IETF, ISSN 2070-1721 (Dec. 1998).
[3] P.E. Jones, S. Dhesikan, C. Jennings, D. Druta, DSCP Packet Markings for WebRTC QoS, Internet Draft draft-ietf-tsvwg-rtcweb-qos-18, IETF (Feb. 2017).
[4] R. Barik, M. Welzl, A.M. Elmokashfi, T. Dreibholz, S. Gjessing, CaN webrtc qos work? a DSCP measurement study, in: Proceedings of the 30th International Teletraffic Congress (ITC), Wien/Austria, ISBN: 978-0-9883045-5-0, 2018, pp. 167–175.
[5] A. Medina, M. Allman, S. Floyd, Measuring interactions between transport protocols and middleboxes, in: Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC), ISBN: 1-58113-821-0, 2004, pp. 336–341.
[6] A. Medina, M. Allman, S. Floyd, Measuring the evolution of transport protocols in the internet, SIGCOMM Comput. Commun. Rev. 35 (2) (2005) 37–52, ISSN 0146-4833.
[7] J. Pahdye, S. Floyd, On inferring TCP behavior, in: Proceedings of the ACM SIGCOMM Conference, ISBN: 1-58113-411-8, 2001, pp. 287–298.
[8] M. Honda, Y. Nishida, C. Raiciu, A. Greenhalgh, M. Handley, H. Tokuda, Is it still possible to extend tcp?, in: Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC), ISBN: 978-1-4503-1013-0, 2011.
[9] G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, B. Donnet, Revealing middlebox interference with tracebox, in: Proceedings of the 13th ACM Internet Measurement Conference (IMC), Barcelona, Catalonia/Spain, ISBN: 978-1-4503-1953-9, 2013.
[10] R. Craven, R. Beverly, M. Allman, A middlebox-cooperative TCP for a non-end-to-end internet, in: Proceedings of the ACM SIGCOMM Conference, ISBN: 978-1-4503-2836-4, 2014, pp. 151–162.
[11] I.R. Learmonth, B. Trammell, M. Kühlewind, G. Fairhurst, Pathspider: a tool for active measurement of path transparency, in: Proceedings of the ACM, IRTF and ISOC Applied Networking Research Workshop (ANRW), ISBN: 978-1-4503-4443-2, 2016.
[12] N. Weaver, C. Kreibich, M. Dam, V. Paxson, Here be web proxies, in: Proceedings of the Passive and Active Measurement Workshop (PAM), Los Angeles, California/U.S.A., ISBN: 978-3-319-04917-5, 2014, pp. 183–192.
[13] X. Xu, Y. Jiang, T. Flach, E. Katz-Bassett, D. Choffnes, R. Govindan, Investigating transparent web proxies in cellular networks, in: Proceedings of the Passive and Active Measurement Workshop (PAM), New York/U.S.A., ISBN: 978-3-319-15509-8, 2015, pp. 262–276.
[14] Z. Wang, Z. Qian, Q. Xu, Z. Mao, M. Zhang, An untold story of middleboxes in cellular networks, in: Proceedings of the ACM SIGCOMM Conference, ISBN: 978-1-4503-0797-0, 2011, pp. 374–385.
[15] A. Müller, F. Wohlfart, G. Carle, Analysis and topology-based traversal of cascaded large scale nats, in: Proceedings of the ACM Workshop on Hot Topics in Middleboxes and Network Function Virtualization (HotMiddlebox), ISBN: 978-1-4503-2574-5, 2013, pp. 43–48.
[16] B. Trammell, M. Kühlewind, Observing internet path transparency to support protocol engineering, in: Proceedings of IRTF/ISOC RAIM Workshop, Yokohama/Japan, 2015.
[17] R. Barik, M. Welzl, A. Elmokashfi, How to say that you're special: Can we use bits in the ipv4 header?, in: Proceedings of the ACM, IRTF and ISOC Applied Networking Research Workshop (ANRW), Berlin/Germany, ISBN: 978-1-4503-4443-2, 2016, pp. 68–70.
[18] B. Trammell, M. Kühlewind, P.D. Vaere, I.R. Learmonth, G. Fairhurst, Tracking transport-layer evolution with pathspider, in: Proceedings of the ACM, IRTF and ISOC Applied Networking Research Workshop (ANRW), Prague/Czech Republic, ISBN: 978-1-4503-5108-9, 2017, pp. 20–26.
[19] A. Custura, A. Venne, G. Fairhurst, Exploring DSCP modification pathologies in mobile edge networks, in: Proceedings of the Network Traffic Measurement and Analysis Conference (TMA), ISBN: 978-3-901882-95-1, 2017, pp. 1–6.
[20] A. Custura, R. Secchi, G. Fairhurst, Exploring DSCP modification pathologies in the internet, Comput. Commun. (ISSN: 0140-3664) 127 (2018) 86–94.
[21] M. Welzl, S. Islam, R. Barik, S. Gjessing, A.M. Elmokashfi, Investigating the delay impact of the diffserv code point (DSCP), in: Proceedings of the International Conference on Computing, Networking and Communications (ICNC): Internet Services and Applications (ICNC), Honolulu, Hawaii/U.S.A., 2019.
[22] R. Barik, M. Welzl, A.M. Elmokashfi, S. Gjessing, S. Islam, Fling: A flexible ping for middlebox measurements, in: Proceedings of the 29th International Teletraffic Congress (ITC), Genoa/Italy, ISBN: 978-0-9883045-3-6, 2017.
[23] K.C. Claffy, CAIDA: Visualizing the internet, IEEE Internet Comput. 5 (1) (2001) 88.
[24] J. Babiarz, K.H. Chan, F. Baker, Configuration Guidelines for DiffServ Service Classes, Informational RFC 4594, IETF, ISSN 2070-1721 (Aug. 2006).
[25] L. Peterson, T. Roscoe, The design principles of planetlab, Oper. Syst. Rev. (ISSN: 0163-5980) 40 (1) (2006) 11–16.
[26] E.G. Gran, T. Dreibholz, A. Kvalbein, Nornet core – a multi-homed research testbed, Comput. Netw. Spec. Issue Future Internet Testbeds (ISSN: 1389-1286) 61 (2014) 75–87.
[27] J. Mahdavi, V. Paxson, IPPM Metrics for Measuring Connectivity, RFC 2498 (Experimental) (Jan. 1999).
[28] R. Bless, A Lower Effort Per-Hop Behavior (LE PHB) for Differentiated Services, Internet-Draft draft-ietf-tsvwg-le-phb-10, Internet Engineering Task Force, work in Progress (Mar. 2019).
[29] T. Høiland-Jørgensen, D. Täht, J. Morton, Piece of cake: a comprehensive queue management solution for home gateways, in: Proceedings of the IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), 2018, pp. 37–42, ISSN 1944-0375.
[30] B.J. Goodchild, Y.-C. Chiu, R. Hansen, H. Lua, M. Calder, M. Luckie, W. Lloyd, D. Choffnes, E. Katz-Bassett, The record route option is an option!, in: Proceedings of the Internet Measurement Conference (IMC), in: IMC '17, ACM, New York, NY, USA, ISBN: 978-1-4503-5118-8, 2017, pp. 311–317.
[31] T. Pauly, B. Trammell, A. Brunstrom, G. Fairhurst, C. Perkins, P.S. Tiesel, C.A. Wood, An Architecture for Transport Services, Internet-Draft draft-ietf-taps-arch- 03, Internet Engineering Task Force, work in Progress (Mar. 2019).
[32] B. Trammell, M. Welzl, T. Enghardt, G. Fairhurst, M. Kühlewind, C. Perkins, P.S. Tiesel, C.A. Wood, An Abstract Application Layer Interface to Transport Services, Internet-Draft draft-ietf-taps-interface-03, Internet Engineering Task Force, work in Progress (Mar. 2019).
[33] N. Khademi, D. Ros, M. Welzl, Z. Bozakov, A. Brunstrom, G. Fairhurst, K. Grinnemo, D. Hayes, P. Hurtig, T. Jones, S. Mangiante, M. Tuxen, F. Weinrank, NEAT: A platform- and protocol-independent internet transport API, IEEE Commun. Mag. 55 (6) (2017) 46–54.

Runa Barik is a Ph.D. candidate at the Department of Informatics, University of Oslo (UiO), Norway. Mr. Barik received a Master of Science by Research from IIT Mandi, India in 2014. His previous research includes developing theoretical models, simulations and kernel implementations related to TCP and MPTCP. Mr. Barik has actively involved in several EU projects. He has submitted two IETF drafts and presented talks at the IETF meetings. His current research includes network measurements, detecting middleboxes and opportunistic use of various IPv4 and IPv6 Internet Protocols.

Michael Welzl is a full professor in the Department of Informatics of the University of Oslo since 2009. He received his Ph.D. (with distinction) and his habilitation from the University of Darmstadt / Germany in 2002 and 2007, respectively. His book "Network Congestion Control: Managing Internet Traffic", is the only introductory book on network congestion control. Michael has been active in the IETF and IRTF for many years, e.g. by chairing the Internet Congestion Control Research Group (ICCRG) leading the effort to form the Transport Services (TAPS) Working Group. He has also participated in several European research projects, including roles such as coordinator and technical manager.

Ahmed Elmokashfi is a Senior Research Scientist at Simula Metropolitan Centre for Digital Engineering and Simula Research Laboratory in Norway. He is currently heading the Centre for Resilient Networks and Applications (CRNA), which is part of the Simula Metropolitan Centre that is funded by the Norwegian government. CRNA focuses on conducting long-term research and innovation at a high international level, on robustness and reliability of communications networks and applications. Dr. Elmokashfi's research interest lies in network Measurements and Performance. In particular, he has been focusing on studying the resilience, scalability, and evolution of the Internet infrastructure; the measurement and quantification of robustness in mobile broadband networks; and the understanding of dynamical complex systems. Over the past few years,he has been leading and contributing to the development, operation and management the NorNet testbed infrastructure, which is a countrywide measurement setup for monitoring the performance of mobile broadband networks in Norway. Dr. Elmokashfi received his PhD degree from the University of Oslo in 2011.

Thomas Dreibholz has received his Diplom (Dipl.-Inform.) degree in Computer Science from the University of Bonn in Bonn, Germany in 2001. Furthermore, he has received his Ph.D. degree (Dr. rer. nat.) in 2007, as well as his Habilitation (Priv.-Doz.) degree in 2012 from the University of Duisburg-Essen in Essen, Germany. Now, he works as Chief Research Engineer for the Simula Metropolitan Centre for Digital Engineering (SimulaMet) in Oslo, Norway.

He has published and presented more than 70 research contributions at international conferences and in journals, on the topics of Reliable Server Pooling (RSerPool), the Stream Control Transmission Protocol (SCTP), Quality of Service (QoS), and multi-homed network infrastructures. Furthermore, he has contributed multiple Working Group and Individual Submission Drafts to the IETF standardisation processes of RSerPool and SCTP. He is also co-author of multiple RFC documents published by the IETF. In addition, he has written the RSerPool reference implementation.

Safiqul Islam is a Postdoctoral Fellow at the Department of Informatics, University of Oslo (UiO), Norway. He received his Ph.D. from UiO in 2017. His research interests include performance analysis, evaluation, and optimization of transport layer protocols. He has actively participated in several EU and International projects. He is active in the IETF and IRTF where he has contributed to several IETF/IRTF Working Groups. Prior to his Ph.D., he worked with Ericsson Research and TSLab at the Royal Institute of Technology (KTH), Sweden.

Stein Gjessing is a professor of Computer Science in Department of Informatics, University of Oslo. He received his the Cand. Real. degree in 1975 and his Dr. Philos. degree in 1985, both form the University of Oslo. He acted as Head of the Department of Informatics for 4 years from 1987. Gjessing's original work was in the field of programming languages and programming language semantics, in particular related to object oriented concurrent programming. He has worked with computer interconnects and computer architecture for cache coherent shared memory, with DRAM organization, with ring based LANs and with IP fast reroute. His current main research interests are multipath routing and transport in IP-like networks.