Article

Cross-protocol Unfairness between Adaptive Streaming Clients over HTTP/3 and HTTP/2: A Root-Cause Analysis

Chanh Minh Tran [1]*, Tho Nguyen Duc [1], Phan Xuan Tan [2]* and Eiji Kamioka [1]*

[1] Graduate School of Engineering and Science, Shibaura Institute of Technology, Japan; nb20501@shibaura-it.ac.jp (T.N.D.)
[2] Department of Information and Communications Engineering, Shibaura Institute of Technology, Japan
* Correspondence: nb20502@shibaura-it.ac.jp (C.M.T); tanpx@shibaura-it.ac.jp (P.X.T.); kamioka@shibaura-it.ac.jp (E.J.)

Abstract: With the introduction of HTTP/3, whose transport is no longer the traditional TCP protocol but the novel QUIC protocol, research for solutions to the unfairness of Adaptive Streaming over HTTP (HAS) has become more challenging. That is, because of different transport layers, the HTTP/3 may not be available for some networks and the clients have to use HTTP/2 for their HAS applications instead. Therefore, the scenario that HAS over HTTP/3 (HAS/3) compete against HTTP/2 (HAS/2) must be considered seriously. However, there have been a shortage of investigations on the performance and the origin of the unfairness in such a cross-protocol scenario in order to produce proper solutions. Therefore, this paper provides a performance evaluation and root-cause analysis of the cross-protocol unfairness between HAS/3 and HAS/2. It is concluded that, due to differences in the congestion control mechanisms of QUIC and TCP, HAS/3 clients obtain larger congestion windows, thus requesting higher video bitrates than HAS/2. As the problem lies in the transport layer, existing client-side ABR-based solutions for the unfairness from the application layer may perform suboptimally for the cross-protocol case.

Keywords: adaptive streaming, HTTP/3, QUIC, cross-protocol, unfairness, congestion control

1. Introduction

Adaptive Streaming over HTTP (HAS) has become the de facto standard for most of the online video services on the Internet nowadays, thanks to its ability to instantaneously adapt the video quality with the network condition [1]. In HAS, a video at the streaming server is encoded into multiple quality versions (in terms of bitrates), each of which is split into multiple small segments with same duration. Then, every streaming client is equipped with an Adaptive Bitrate Algorithm (ABR) in its video player to continuously monitor the network condition, therefore requesting the suitable bitrate for every video segment. Accordingly, undesirable incidents such as playback stalls and quality variation can be reduced, thus maintaining a high quality of experience (QoE) for the users.

Nevertheless, in order to cope with the constantly increasing number of online video users [2], as well as their daily usage time of the service [3], the performance of the HAS services against multiple competing clients still requires further optimization. In multi-client scenarios, it has been proven that due to the mismatch of the clients' downloading states originating from their ABRs on the application layer, some clients may overestimate their bandwidths and request higher video bitrates than the others [4–6]. Such a phenomenon is defined as the unfairness problem, which causes QoE deterioration and negative impact on the user retention rate. Therefore, over the years, various research has been conducted and various solutions to the unfairness have been proposed [7–10].

On the other hand, the protocol stacks have seen some significant changes within the recent years that complicates the efforts of solving the unfairness of HAS. In fact, the HTTP/3 protocol [11] was proposed and is expected to be standardized in near future. While HTTP/2 differs from HTTP/1.1 by the novel features on the application layer [12], the major difference between HTTP/3 and its successors lies in the transport protocol. That is, while HTTP/1.1 and HTTP/2 have been running on top of the well-tuned TCP

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

protocol for decades, the HTTP/3 utilizes the novel QUIC protocol [13] for its transport. QUIC actually runs atop the UDP protocol, which is often blocked or limited by network entities due to known security risks [14–16]. This means that, as QUIC relies on UDP, there are realistic scenarios that a client fails to use HTTP/3 because of configurations of its network system and has to use HTTP/2 or HTTP/1.1 instead (Figure 1). Thus, streaming providers must support both HTTP/3 and the former HTTP versions at the same time to ensure service availability. This raises the need for the performance investigation of the unfairness with regard to the cross-protocol scenario, where streaming clients using HTTP/3 concurrently compete against ones using HTTP/2 or HTTP/1.1.

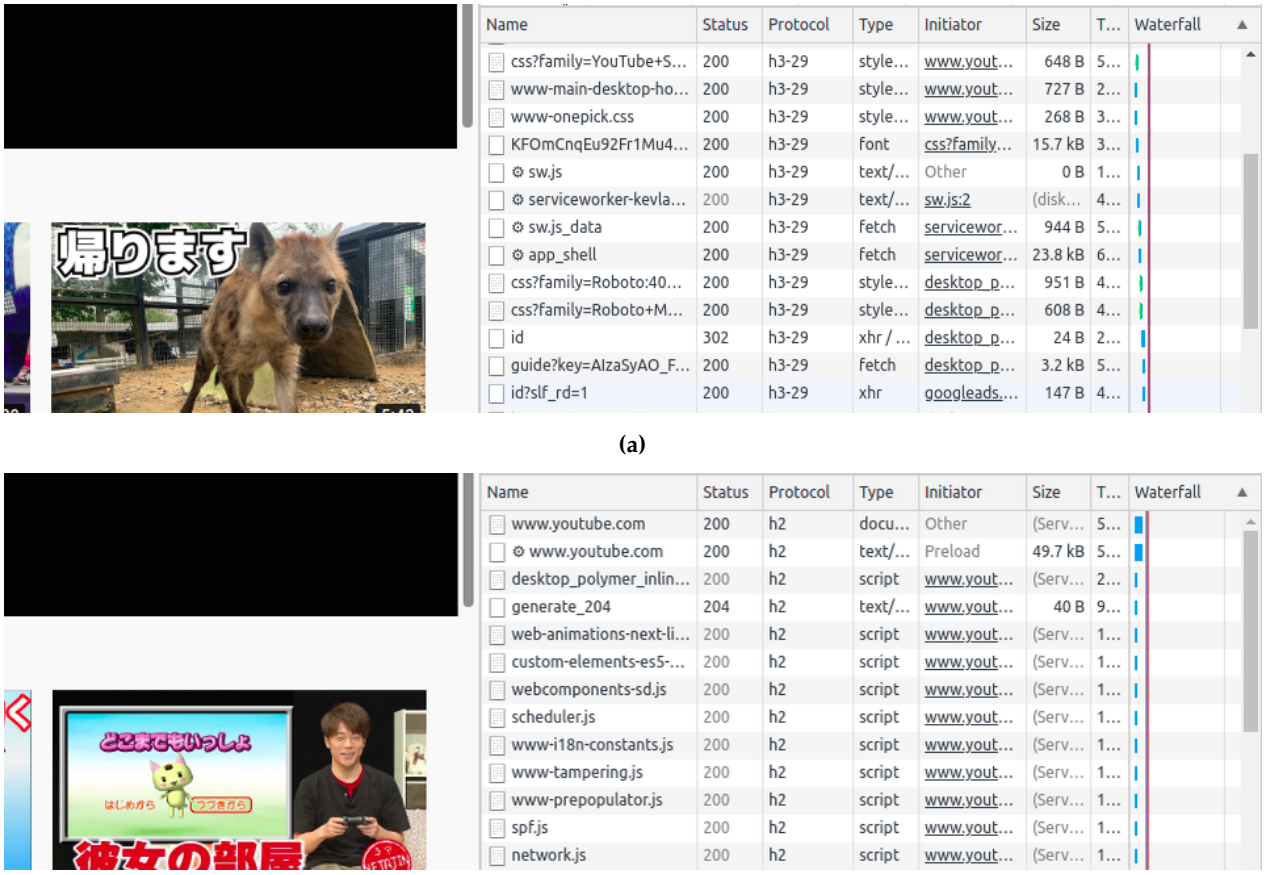

Figure 1. An example realistic case of Youtube where the HTTP/3 cannot be utilized. (a) a normal client that successfully sends HTTP/3 requests (h3-29); (b) a firewall controlled client that fails to send HTTP/3 requests and has to use HTTP/2 (h2) instead

In this manner, research about the cross-protocol unfairness between HAS over HTTP/3 (HAS/3) and over HTTP/2 (HAS/2) or HTTP/1.1 (HAS/1.1) should attract more attention. Yet, to the best of our knowledge, only a few works have attempted to investigate the unfairness in such a case for HAS and no solution has ever been proposed due to the following limitations. Firstly, as HTTP/3 and QUIC are still under standardization, their features and characteristics vary upon drafted versions and their implementations vary upon releases. Consequently, performance conclusions in the existing works are contradictory against one another. Secondly, those works only provide observations and lack of root cause analysis. Without such crucial information, research for an effective solution to the cross-protocol unfairness could not be conducted.

Realizing the aforementioned drawbacks, this paper presents an up-to-date performance evaluation and a comprehensive root-cause investigation on the cross-protocol unfairness of HAS/3 against its successors HAS/2 and HAS/1.1. In fact, the HTTP/2 nowadays accounts for 64% of HTTP requests over the internet and is expected to continue growing linearly [17]. Such usage growth, in addition to the more advanced features the

HTTP/2 provides [12], it is believed that HTTP/1.1 will become deprecated in near future time. Therefore, this work focuses mainly on the competition of HAS/3 and HAS/2 clients. Our analysis demonstrates that, with the newest documentation and implementation, HTTP/3 clients tend to experience higher video qualities than HTTP/2 clients when they compete under the same bottleneck network. Taking an in-depth look at their transport - QUIC for HTTP/3 versus TCP for HTTP/2, it is found that, despite both implementations running the same congestion control algorithm, QUIC is able to acquire a larger congestion window than TCP. Such a behavior actually originates from the different mechanisms of the congestion control and the characteristics of QUIC and TCP [18]. In detail, we argue that the differences in loss epoch life cycle, acknowledgement (ACK) ranges and minimum congestion window size, as well as the unreliable nature of the UDP (QUIC's based protocol), allow QUIC to receive ACK frames more frequently than TCP. As a consequence, QUIC updates its congestion window more aggressively than TCP, resulting in a higher occupation of the bandwidth. Moreover, since such an underperformance is transparent to the application layer, it has been proven that the existing unfairness solutions utilizing the client-side ABR-based approach on the application layer may fail to achieve desirable effectiveness. Instead, follow-up research should focus on either examining server- or network-based approaches on the transport layer, or tweaking the functionalities and parameters of the transport QUIC. In summary, the distinguished contributions of this work are as follows:

- An up-to-date investigation of the cross-protocol unfairness between HAS/3 and HAS/2 is provided, showing that HAS/3 clients unfairly experience higher video quality.
- A comprehensive analysis of such an unfairness problem is conducted, which concludes that its origin lies in the differences in congestion control mechanisms and the characteristics of the transport layer - QUIC for HTTP/3 and TCP.
- Based on the root-cause analysis, we provide suggestions for proper solutions to the cross-protocol unfairness between HAS/3 and HAS/2.

The remainder of this paper is organized as follows: Section 2 reviews the existing research about the cross-protocol unfairness between HAS/3 and HAS/2. The hypothesis on the cause of the cross-protocol unfairness and the methodology and experimental setup for validation are presented in Section 3. The experimental results are analyzed and discussed in Section 4 and 5, respectively. Finally, Section 6 concludes this paper.

2. Related Work

As explained in the previous section, the differences in the transport protocol between HTTP/3 and HTTP/2 or HTTP/1.1 urges the need for tackling the cross-protocol unfairness of these traffics. However, research about such a problem for HAS is surprisingly limited. The work in [19] provided a performance study of several ABRs under the HTTP-over-QUIC, which is the former name of HTTP/3 before November 2018 [20]. The author did not state which version of QUIC had been tested in their work. However, they did mention the Google QUIC, which is the very first variant of today's IETF QUIC. Based on their evaluation results, it was shown that the streaming clients using HTTP-over-QUIC failed to experience the video bitrate as high as that of the HTTP-over-TCP clients. The author argued that such an underperformance of the HTTP-over-QUIC clients was because most ABRs were tuned to work well with TCP so they did not utilized the features of QUIC effectively. In their later work [21], the QUIC retransmission scheme was employed with Google QUIC version Q043, showing promising improvements in video quality and bitrate stability. Interestingly, while they discussed that their solution might result in unfairness among the clients running HTTP-over-QUIC, they also observed that such a kind of client also starved out the HTTP-over-TCP clients. Such an observation was contradictory to their own previous work in [19]. On the other hand, a performance investigation similar to that of [19] was conducted in [22] with the Google QUIC server v39. The finding in [22] showed that, while the HAS clients running over QUIC could perform fairly against one another,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

they provided 37% higher bitrate than the HAS over TCP clients in most cross-protocol cases.

It can be noticed that the findings in those existing works were contradictory to one another and were actually outdated as the Google QUIC is now deprecated to make way for the IETF QUIC [23]. Moreover, their works only provided observation and lack of in-depth analysis on the origin of the problem. Thus, to the best of our knowledge, there have been no solution proposed for the cross-protocol fairness between HAS/3 and HAS/2 clients due to such a serious drawback. In order to set light for clarifying the root-cause of the cross-protocol unfairness, in the following sections, we present our hypothesis, experimental methodology and performance analysis with the latest version of the HTTP/3 and QUIC.

3. Hypothesis and Methodology

This section describes the hypothesis on the existence and the cause of the cross-protocol unfairness among HAS/3 and HAS/2 clients in detail, as well as the methodology and experimental settings for validating it.

3.1. Hypothesis

In HAS, a client's ABR typically decides the video bitrate based on its observation of the available bandwidth. Apparently, an unfair bandwidth consumption of a client towards one another can lead to an unfair bitrate selection. Nevertheless, the bandwidth fairness is actually well-handled by the congestion control of the TCP protocol in the transport layer [24,25] and existing research also concluded that the unfairness in bitrate selection of single-protocol HAS/2 or HAS/1.1 was the result of inefficient ABRs in the application layer [4]. Therefore, before the introduction of HTTP/3, previous investigations and solutions to the unfairness only focused on optimizing the ABR from the client side.

On the other hand, this work considers the cross-protocol scenario, where TCP is no longer the only transport layer of all clients. In fact, the transport QUIC of HAS/3 also implements TCP's congestion control. However, as stated in the draft RFC, there exists several differences, which are considered as enhancements, in the mechanism of QUIC's congestion control compared with TCP's [18]. The differences that are most relevant to this paper are as follows:

- Immediate reaction to packet loss: When a packet loss is detected, QUIC typically reacts every round trip. For the case of TCP, it may have to wait for multiple round trips before progressing with follow-up actions.
- More ACK ranges for loss recovery: While TCP utilizes only 3 Selective ACK for loss recovery, QUIC supports many ACK ranges to speed up such a process, especially in high loss environments.
- Increasing minimum congestion window: QUIC recommends the minimum congestion window is two packets long, while that of the TCP is only one packet.

Thus, it is likely that the performance of their congestion control is dissimilar and causes uneven bandwidth distribution. As the clients' ABRs rely heavily on the bandwidth obtained by their transport, their bitrate selections are affected, which causes unfairness in video bitrates. To clarify this hypothesis, the experimental methodology and settings of this paper is presented in the remaining parts of this section.

3.2. Methodology

3.2.1. Experimental Scenarios

In order to confirm the cross-protocol unfairness between HAS/3 and HAS/2 with the latest documentation and implementation of HTTP/3, as well as to validate our hypothesis on the root cause of the problem, an experimental evaluation is conducted. The experiment is run with several total bandwidth limits B to assess the performance of the clients across different competitiveness of the network (i.e., a lower value of B means a more serious competition). In this experiment, the clients are examined with $B \in \{3, 5\}$ (mbps).

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

In addition, the number of each type of client (i.e., HAS/3 or HAS/2 clients) also varies. This is to verify whether the unfairness is explicitly towards a particular type of client. In this experiment, the total number of all clients is varied from 2 to 3 clients. For the case of 2 clients, there is only the scenario that 1 HAS/3 client competes with 1 HAS/2 client. While with 3 clients, there are two subscenarios to be considered: 1 HAS/3 client competes with 2 HAS/2 clients, and 2 HAS/3 clients compete with 1 HAS/2 client. In summary, the details of the evaluated scenarios are shown in Table 1

Total number of clients	Scenario	Denotation
2 clients	1 HAS/3 client and 1 HAS/2 client	S_{11}
3 clients	1 HAS/3 client and 2 HAS/2 clients	S_{12}
	2 HAS/3 clients and 1 HAS/2 client	S_{21}

Table 1: Summary of the experimental scenarios

3.2.2. Investigation Metrics

Existing works on the unfairness of HAS often utilized the average unfairness index $\overline{F}$ of the video bitrates selected by the clients' ABRs as a numerical metric of the unfairness level [6,7,10]. Therefore, in this paper, this metric will be considered to investigate such a problem. $\overline{F}$ is estimated by averaging the Jain Fairness index F [26] from the beginning to the end of a streaming session, with F calculated as in Equation 1.

$$F = \sqrt{1 - \frac{\left(\sum_{c=1}^{\mathbb{C}} r_{c,t}\right)^2}{\mathbb{C} * \sum_{c=1}^{\mathbb{C}} {r_{c,t}}^2}} \tag{1}$$

where $r_{c,t}$ denotes the bitrate r selected by the client c at time t. A larger value of $\overline{F}$ indicates a more severe unfairness. Additionally, the average bitrate $\bar{r}$ of each client throughout its streaming session will be measured in order to judge whether higher bitrates are only available at a specific type of client.

Furthermore, in this work, it is hypothesized that the differences in congestion control of the transport protocol QUIC and TCP actually causes the unfairness in bandwidth consumption among the clients. In order to clarify this hypothesis, the time-varying congestion window of each client, which is the output of the congestion control algorithm, is also collected.

3.3. Experimental Setup

In this subsection, the description of the experimental settings for investigating the cross-protocol unfairness between HAS/3 and HAS/2 is provided. Figure 2 depicts the experiment topology.

The video server and clients were deployed on separate virtual machines running Ubuntu 20.04 LTS with 4GB of RAM and 4 processor cores. Those machines were actually virtualized on a Core i5 physical machine running Ubuntu 20.04 LTS with 80GB of RAM. For enabling HTTP/3, the server applied the implementation of *quic-go* v0.20.1 [27] which supported version 34 of the drafted HTTP/3 and QUIC, which was the latest draft at the time we conducted this experiment. As for HTTP/2, the server simply used the native *http* package of Golang [28]. Meanwhile, the total bandwidth limit B was configured using the linux *tc* [29]; the congestion window was extracted from the qlog files of *quic-go* for HTTP/3 and linux *ss* [30] for HTTP/2.

The streaming application was packaged at the server based on the dash.js framework [31], which was run via a Firefox web browser from the client side. The server provided 300 2-second segments of the open-source *Big Buck Bunny* video with 11 quality versions, i.e. {100, 200, 300, 500, 700, 900, 1200, 1500, 2000, 2500, 3000} (kbps). At the client, the maximum buffer was set to 30 seconds. In this experiment, the clients' performances were tested with two types of ABR: general ABR and fairness ABR. For the general ABR, the default ABR

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

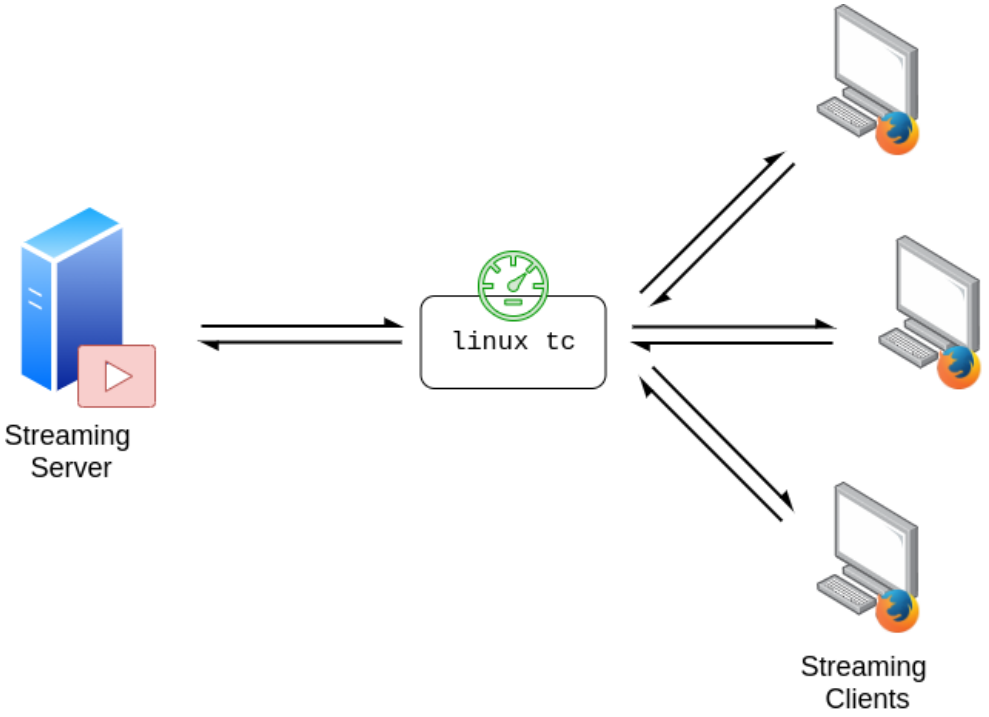

Figure 2. Experimental topology

of dash.js was employed, which was actually a dynamic ABR that conditionally switches between its own throughput ABR and the well-known buffer-based BOLA [32]. The dash.js ABR was not tuned for solving the unfairness issues, whereas we were also curious about whether existing fairness ABRs could perform well in such a cross-protocol scenario. For this reason, the FESTIVE [7], which was among the most famous baseline fairness ABRs, was considered. The FESTIVE utilized the harmonic mean of the estimated bandwidth, gradual and stateful bitrate transition, and randomized segment download scheduler to overcome the unfairness from the client side.

The experiment of each scenario described in 3.2 was tested 10 times to ensure the performance consistency. In the following section, the detailed results and analysis are presented.

4. Results and Analysis

This section provides the results and analysis of our experiment on the cross-protocol unfairness between HAS/3 and HAS/2 clients. For the numerical results ($\overline{F}$ and $\overline{r}$) of each scenario, we show the average values of all 10 runs. As for the visualized time-varying figures, due to similar behaviors, only one representative run for each scenario is shown.

4.1. General ABR

In this subsection, the results collected for the general ABR are analyzed. Table 2 summarizes the average bitrate $\overline{r}$ of every client. In this table, $C^X_{HAS/Y}$ denotes the client X running HAS/Y with $Y \in \{2,3\}$. Meanwhile, the average unfairness index $\overline{F}$ of all scenarios are shown in Table 3.

B	S_{11}		S_{12}			S_{21}		
	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/3}$
3 mbps	1412	1007	1105	706	717	897	702	918
5 mbps	2325	1552	1684	1187	1180	1536	1115	1483

Table 2: The average bitrate $\overline{r}$ (kbps) of each client running the general ABR

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

B	S_{11}	S_{12}	S_{21}
3 mbps	0.1895	0.2711	0.2122
5 mbps	0.2105	0.2398	0.2259

Table 3: The average unfairness index $\overline{F}$ of the general ABR

The numerical results in Table 3 quantitatively demonstrate that, across all scenarios, the bitrate decision of same-type HAS clients were relatively indistinguishable. In contrast, it was obvious that the HAS/3 clients ended up requesting higher bitrates than the HAS/2 clients. Particularly, the bitrates selected by the HAS/3 clients were 42.05% higher than those of the HAS/2 clients on average. This led to the results of the unfairness index $\overline{F}$ in Table 3. Although such an unfairness performance was actually predictable because the general ABR wasn't tuned for solving such a problem, the fact that the high bitrates were reserved only for the HAS/3 clients was very interesting. Taking a closer look at this observation, Figure 3 and 4 depicts the time-varying bitrates, congestion windows (*cwnd*) and moving average congestion windows (*cwnd_ma*) of the clients under 3 mbps and 5 mbps, respectively.

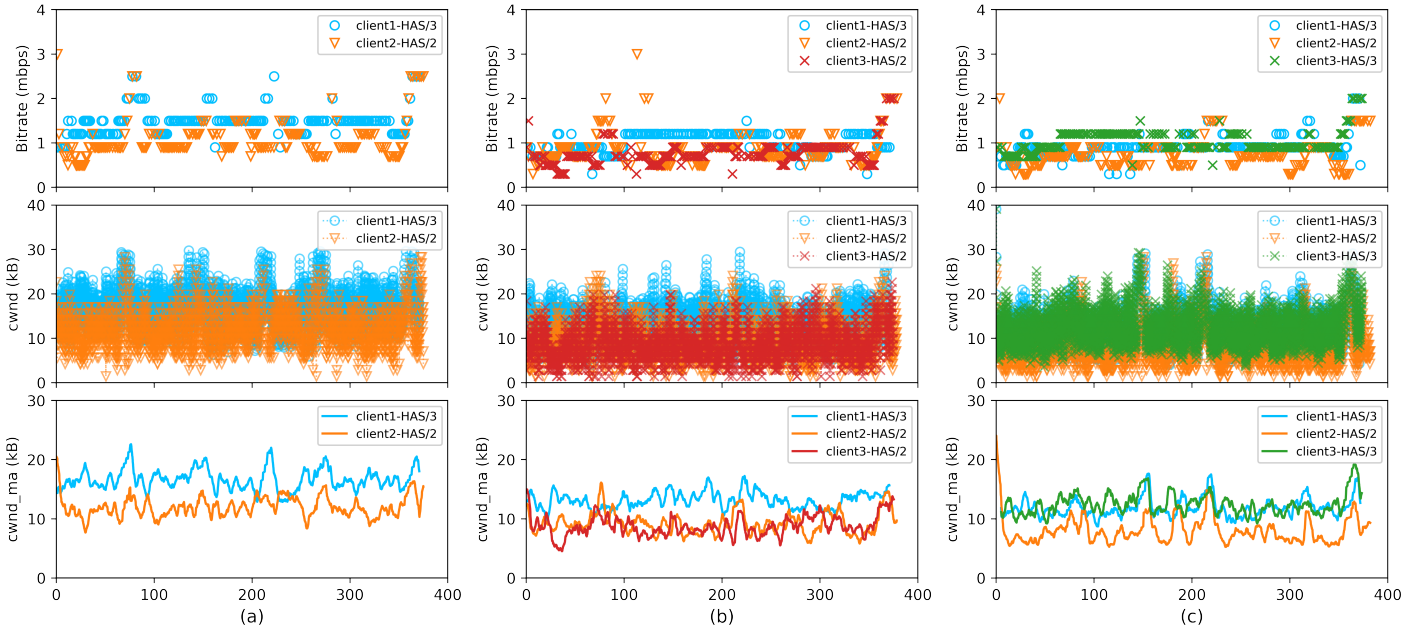

Figure 3. The representative time-varying bitrates, *cwnd* and *cwnd_ma* of the clients running the general ABR with $B = 3$ (mbps), under the scenario (a) S_{11}, (b) S_{12} and (c) S_{21}

It can be observed from the Figure 3 and 4 that the bitrates of both types of clients varied terribly; they were occasionally increased or decreased with large amplitudes and within short periods. This behavior actually harmed the fairness as such abrupt and large changes might significantly increase the clients' bitrates difference at some time intervals. More importantly, the time-varying *cwnd* and *cwnd_ma* from Figure 3 and 4 expresses that the HAS/3 clients always received higher *cwnd* than the HAS/2 clients. Meanwhile, the *cwnd* among the HAS/3 clients or among the HAS/2 clients were relatively comparable. It means that, despite running the same congestion control algorithm, the transport QUIC of HAS/3 clients helped them obtain more *cwnd*, thus receiving more portion of the bandwidth than the HAS/2 clients running on TCP. This phenomenon strongly supports our hypothesis in 3.1, that the performance of the congestion control between QUIC and TCP were unalike, which harmed the fairness of their bandwidth consumption. As a

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

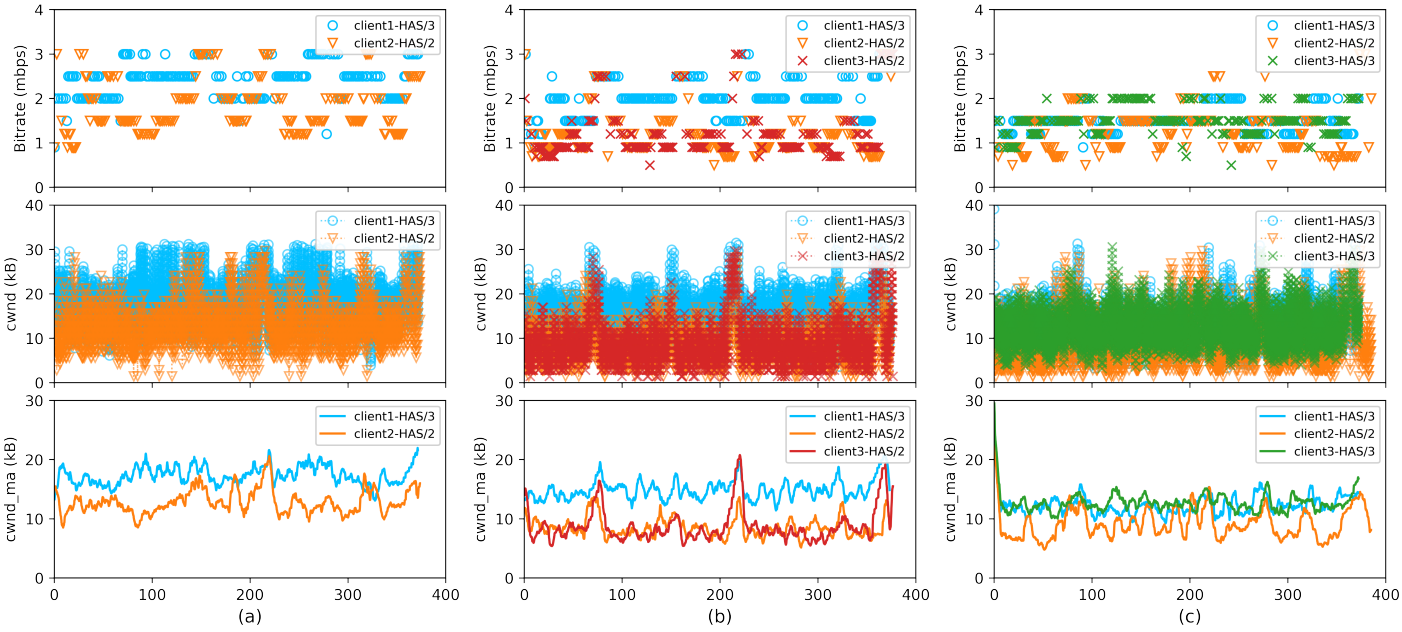

Figure 4. The representative time-varying bitrates, *cwnd* and *cwnd_ma* of the clients running the general ABR with $B = 5$ (mbps), under the scenario (a) S_{11}, (b) S_{12} and (c) S_{21}

result, the HAS/3 clients running on QUIC were able to request higher bitrates. Moreover, since this problem lies in the transport layer, it is highly possible that such a behavior also occurred with the fairness ABR. As the ABR worked on the application layer, it could not control but only relied on the bandwidth given by its transport layer. To validate this expectation, the next subsection shows similar analysis results of the fairness ABR.

4.2. Fairness ABR

Similar to the previous subsection, Table 4 shows the measured $\overline{F}$, Table 5 summarizes $\bar{r}$ of every client, while Figure 5 and 6 illustrates the time-varying performance under 3 mbps and 5 mbps, respectively, in terms of bitrates, *cwnd* and *cwnd_ma* of the fairness ABR.

B	S_{11}	S_{12}	S_{21}
3 mbps	0.0814	0.1095	0.0991
5 mbps	0.0697	0.0909	0.0712

Table 4: The average unfairness index $\overline{F}$ of the fairness ABR

B	S_{11}		S_{12}			S_{21}		
	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/3}$
3 mbps	1106	927	866	706	689	804	653	792
5 mbps	1720	1461	1326	1071	1138	1234	1064	1200

Table 5: The average bitrate $\bar{r}$ (kbps) of each client running the fairness ABR

It can be inferred from Table 4 that the conditions of the cross-protocol unfairness across all scenarios were significantly better than the general ABR. Indeed, the fairness ABR provided an average of 44.69% improvement of $\overline{F}$ compared to the general ABR. Yet, Table 5 concludes a similar tendency to 4.1: the HAS/3 clients still experienced higher video bitrates than HAS/2 clients, despite the fact that the gap between them were minimized to 19.47% higher for HAS/3 on average. Observing from the Figure 5 and 6, the clients' bitrates

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

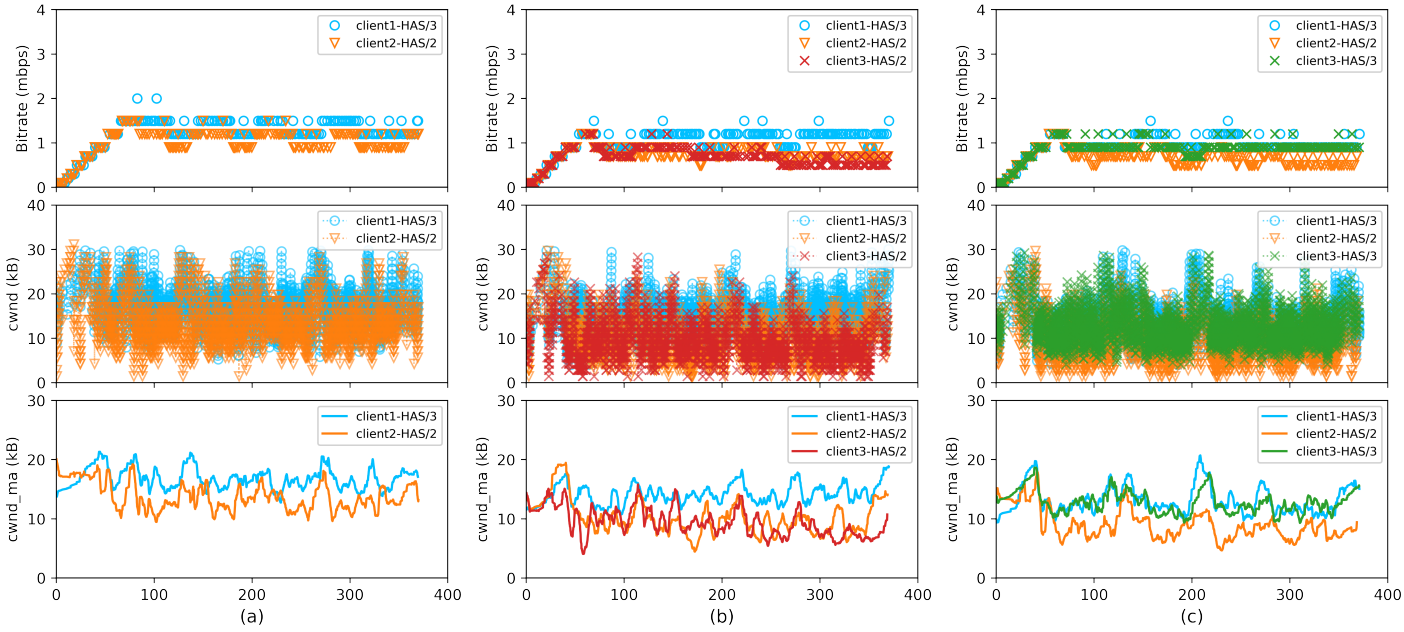

Figure 5. The representative time-varying bitrates, *cwnd* and *cwnd_ma* of the clients running the fairness ABR with $B = 3$ (mbps), under the scenario (a) S_{11}, (b) S_{12} and (c) S_{21}

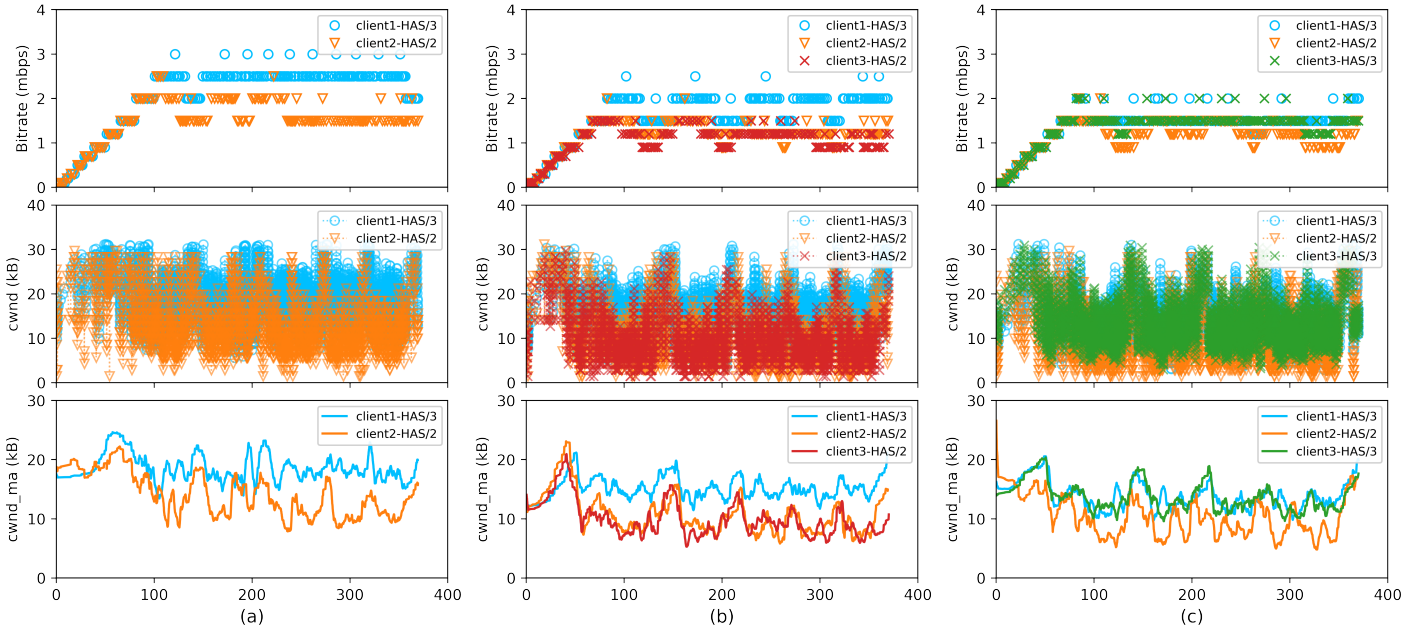

Figure 6. The representative time-varying bitrates, *cwnd* and *cwnd_ma* of the clients running the fairness ABR with $B = 5$ (mbps), under the scenario (a) S_{11}, (b) S_{12} and (c) S_{21}

were more in control and tended to varied within specific ranges. This behavior was the results of the gradual and stateful update mechanism of the fairness ABR: the client stayed on a every bitrate level for a specific number of segments before, if necessary, increasing or decreasing it to the closet higher or lower level. Thus, there was no abruptly large bitrate changes and the difference of the clients' average bitrates was smaller. Moreover, their bitrate selections were relatively identical for a specific amount of time at the beginning. This was because the fairness ABR forced all clients to choose the lowest bitrate at the beginning of their streaming sessions to avoid playback stalls. Such a strategy not only explains why the results of $\bar{r}$ of all types of client with all total bandwidth limit were smaller

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

when running the fairness ABR, but also contributes to the minimization of the bitrates difference discussed earlier. Nevertheless, as expected, the HAS/3 clients were given higher *cwnd*, thus requesting higher bitrates due to higher occupation of bandwidth. As a result, although the bitrate selections were similar at the beginning due to the ABR's strategy, they ended up varying unfairly based on their estimations of bandwidth afterwards.

In summary, the results of both types of ABR demonstrate that, under a cross-protocol scenario, the bitrate selection among the clients were always unfairly higher on the HAS/3 clients. This is because the transport QUIC allowed its HAS/3 clients to utilize higher *cwnd* than the TCP of the HAS/2 clients. Therefore, the root-cause of such a cross-protocol unfairness lies in the congestion control mechanisms of QUIC and TCP, rather than in the application layer as argued in previous investigations that only focused on the single-protocol unfairness.

5. Discussion

Based on the analysis in Section 4, it is found that the transport QUIC was able to get higher *cwnd* for its HAS/3 clients than TCP for HAS/2, leading to an unfair distribution of the shared bandwidth. For this reason, even the referenced fairness ABR could not perform optimally to provide similar bitrates for both types of HAS client. Such a phenomenon is actually explainable by examining the enhancements of QUIC's congestion control in comparison with TCP's as described in 3.2. Those enhancements actually empower QUIC to transmit data packets much faster than TCP. In addition, QUIC runs atop UDP, which is naturally faster than TCP due to the unreliable characteristics [16]. As for those reasons, QUIC is able to receive ACK frames faster than TCP, so that QUIC updates its *cwnd* more aggressively and occupies more than the fair share of bandwidth. In order to confirm this explanation, Table 6 and 7 shows the average number of *cwnd* updates of the clients using the general and the fairness ABR, respectively, in the previous experiment; while Figure 7 illustrates an example zoomed-in time-varying *cwnd* of the clients using the fairness ABR under the scenario S_{11} with $B = 3$ (mbps).

B	S_{11}		S_{12}			S_{21}		
	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/3}$
3 mbps	4407	2586	3776	2569	2610	3894	2611	3971
5 mbps	6916	3631	6662	3637	3600	6250	3851	6048

Table 6: The average number of *cwnd* updates of each client running the general ABR

B	S_{11}		S_{12}			S_{21}		
	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/2}$	$C^1_{HAS/3}$	$C^2_{HAS/2}$	$C^3_{HAS/3}$
3 mbps	3443	2212	3425	2076	2050	3397	2054	3301
5 mbps	5360	3159	5001	3091	2969	4836	3110	4815

Table 7: The average number of *cwnd* updates of each client running the fairness ABR

Obviously, from Table 6 and 7, the HAS/3 clients updated the *cwnd* much more frequently than the HAS/2 clients. The illustration on Figure 7 also supports this conclusion. It can be noticed that the number of *cwnd* updates of all clients running the general ABR were higher than those running the fairness ABR across all scenarios and bandwidth limits. This was simply because the clients running the general ABR often abruptly requested bitrates too higher than the fair portion of the total bandwidth limit, leading to higher packet loss due to insufficient bandwidth and increasing the frequency of *cwnd* updates

In conclusion, our hypothesis on the root-cause of the cross-protocol unfairness of HAS/3 and HAS/2 was verified, that the dissimilarities in congestion control of QUIC and TCP led to unfair bandwidth allocation and, finally, unfair bitrate selections. As the the problem arises from the transport layer, follow-up research on the cross-protocol unfairness

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

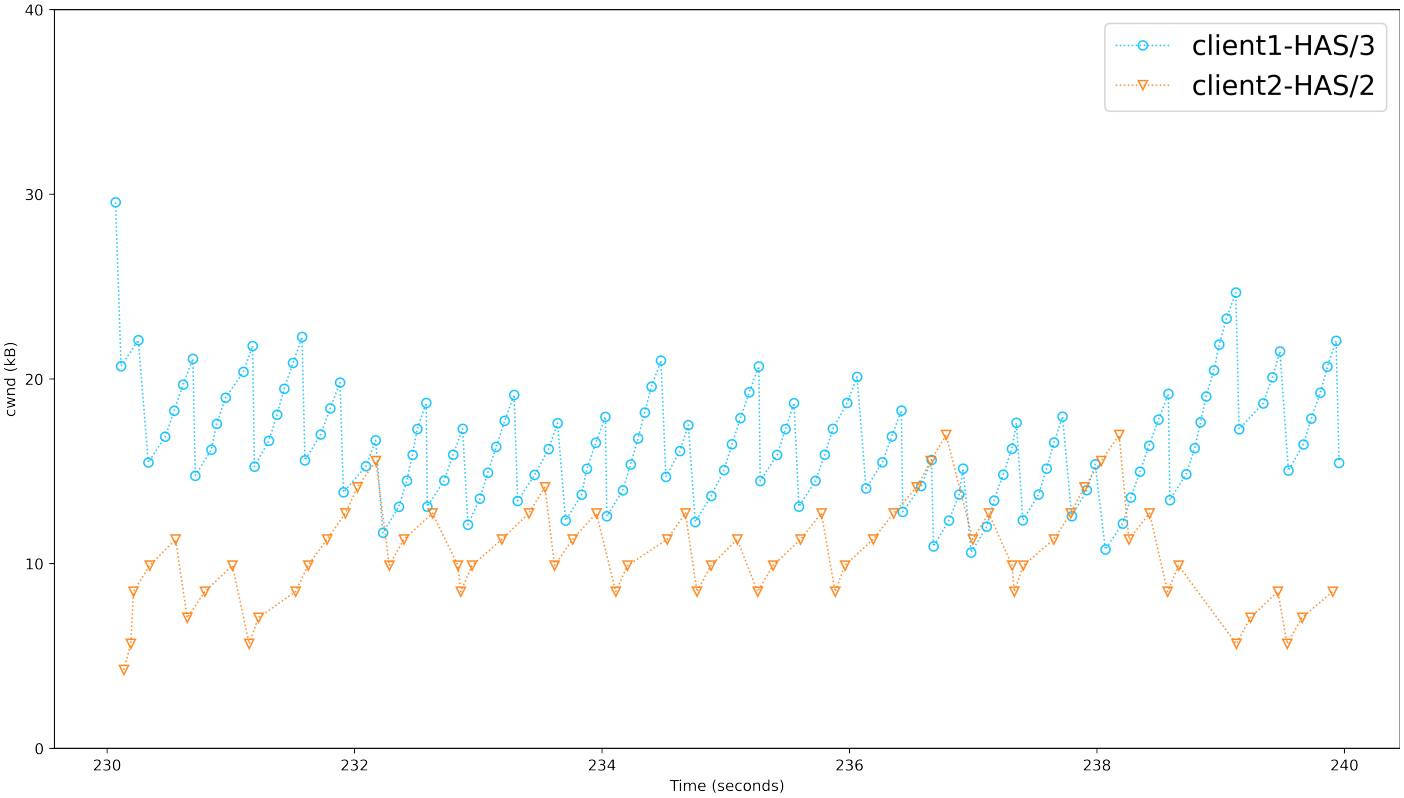

Figure 7. An example zoomed-in time-varying *cwnd* of the clients using the fairness ABR under the scenario S_{11} with $B = 3$ (mbps)

can investigate existing methods working on this layer, such as server- or network-based solutions. For example, [10,33] utilized a bandwidth allocation module that assigned an equal and separate bandwidth slice for each client. Figure 8 visualizes the bitrate selection performance of the scenario S_{11} with $B = 3$ (mbps) when applying such a method.

The time-varying bitrate selection clearly demonstrate superior fairness performance compared with the client-side fairness ABR tested in 4.2. This is because, as explained in [10,33], when every client had its own specific bandwidth, it basically didn'tcompete with one another and only maximized its bitrates based on the assigned bandwidth. Consequently, since all clients were given an equal bandwidth, their bitrate selections ended up being fair regardless of their HTTP versions or transport protocols. Nevertheless, the server- and network-based solutions have been questioned about their consistent efficiency in large-scale network due to extra computational complexity and overhead, or about deployment feasibility as they require additional network entities [34,35]. The cost benefit regarding these matters should be carefully considered before applying such solutions in real life.

On the other hand, the transport QUIC is actually implemented on the user space of both endpoints [36]. Therefore, modifications of the protocol's functionalities and parameters become more feasible as they don't experience the ossification problem caused by conservative network blocks on-the-fly [37,38]. For this reason, future research can also consider tweaking the functionalities and parameters of QUIC for obtaining a fairer bandwidth for the HAS/3 clients against HAS/2 and/or HAS/1.1.

6. Conclusion and Future Work

This paper presents an up-to-date performance analysis of the HAS/3 and HAS/2 clients under a cross-protocol scenario and confirms that HAS/3 clients always unfairly acquire higher bitrates than its successors. Looking into the transport layer, it is found that the root-cause of this underperformance lies in the aggressive occupation of the congestion

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 June 2021 doi:10.20944/preprints202106.0361.v1

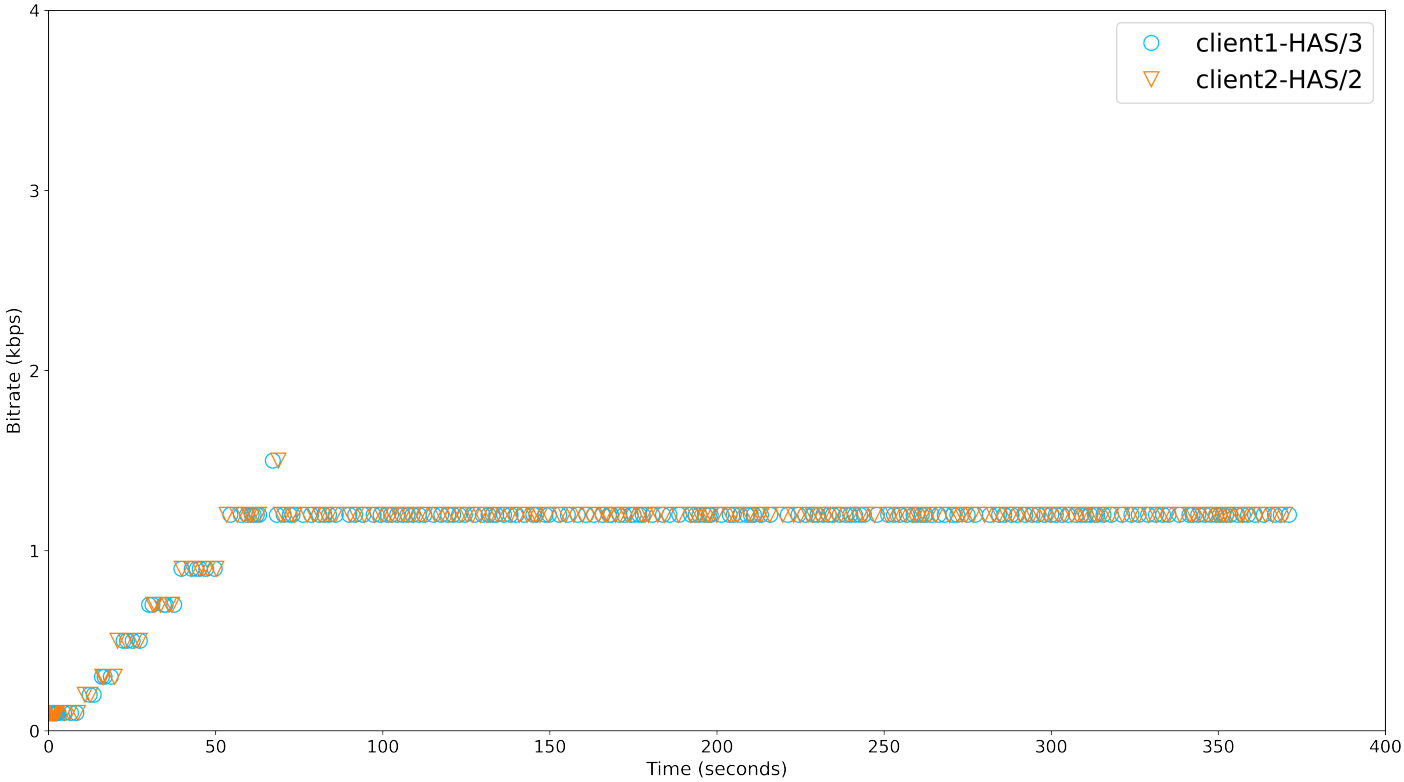

Figure 8. An example time-varying bitrate selection of the scenario S_{11} with $B = 3$ (mbps) when applying a bandwidth allocation method

window of QUIC - the transport of HAS/3. Such a behavior originates from enhancements in the congestion control mechanism of QUIC, as well as its different characteristics with TCP. As a result, it is proven that the existing client-side fairness ABR fails to provide fair bitrates for the clients due to irrelevant working layers. Based on this conclusion, for future work, we will focus on tuning the congestion control algorithm of QUIC. In addition, the feasibility of server- and network-based unfairness solutions will also be examined under such a cross-protocol scenario.

Author Contributions: Conceptualization, Chanh Minh Tran, Tho Nguyen Duc and Phan Xuan Tan; Methodology, Chanh Minh Tran, Tho Nguyen Duc and Phan Xuan Tan; Supervision, Phan Xuan Tan and Eiji Kamioka; Writing – original draft, Chanh Minh Tran and Tho Nguyen Duc; Writing – review & editing, Phan Xuan Tan and Eiji Kamioka.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Stockhammer, T. Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles. Proceedings of the Second Annual ACM Conference on Multimedia Systems; Association for Computing Machinery: New York, NY, USA, 2011; MMSys '11, p. 133–144. doi:10.1145/1943552.1943572.
2. Global Digital Video 2019. Technical report, eMarketer, 2019.
3. Online Video Forecasts 2019. Technical report, Zenith, 2019.
4. Akhshabi, S.; Begen, A.C.; Dovrolis, C. An Experimental Evaluation of Rate-Adaptation Algorithms in Adaptive Streaming over HTTP. Proceedings of the Second Annual ACM Conference on Multimedia Systems; Association for Computing Machinery: New York, NY, USA, 2011; MMSys '11, p. 157–168. doi:10.1145/1943552.1943574.
5. Akhshabi, S.; Anantakrishnan, L.; Begen, A.C.; Dovrolis, C. What Happens When HTTP Adaptive Streaming Players Compete for Bandwidth? Proceedings of the 22nd International Workshop on Network and Operating System Support for Digital Audio and Video; Association for Computing Machinery: New York, NY, USA, 2012; NOSSDAV '12, p. 9–14. doi:10.1145/2229087.2229092.

6. Wang, Y.; Tran, C.M.; Duc, T.N.; Wu, X.; Tan, P.X.; Kamioka, E. An Experimental Study on The Unfairness in Adaptive Streaming with HTTP/2 Server Push. Proceedings of the 2019 International Conference on Video, Signal and Image Processing; Association for Computing Machinery: New York, NY, USA, 2019; VSIP 2019, p. 94–98. doi:10.1145/3369318.3369329.
7. Jiang, J.; Sekar, V.; Zhang, H. Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With Festive. *IEEE/ACM Transactions on Networking* **2014**, *22*, 326–340. doi:10.1109/TNET.2013.2291681.
8. Li, Z.; Zhu, X.; Gahm, J.; Pan, R.; Hu, H.; Begen, A.C.; Oran, D. Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale. *IEEE Journal on Selected Areas in Communications* **2014**, *32*, 719–733. doi:10.1109/JSAC.2014.140405.
9. Liu, J.; Tao, X.; Lu, J. QoE-Oriented Rate Adaptation for DASH With Enhanced Deep Q-Learning. *IEEE Access* **2019**, *7*, 8454–8469. doi:10.1109/ACCESS.2018.2889999.
10. Tran, C.M.; Nguyen Duc, T.; Tan, P.X.; Kamioka, E. FAURAS: A Proxy-Based Framework for Ensuring the Fairness of Adaptive Video Streaming over HTTP/2 Server Push. *Applied Sciences* **2020**, *10*. doi:10.3390/app10072485.
11. Hypertext Transfer Protocol Version 3 (HTTP/3) - draft-ietf-quic-http-34, 2021.
12. RFC: 7540 Hypertext Transfer Protocol Version 2 (HTTP/2), 2015.
13. QUIC: A UDP-Based Multiplexed and Secure Transport - draft-ietf-quic-transport-34, 2021.
14. Berti-Equille, L.; Zhauniarovich, Y. Profiling DRDoS Attacks with Data Analytics Pipeline. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management; Association for Computing Machinery: New York, NY, USA, 2017; CIKM '17, p. 1983–1986. doi:10.1145/3132847.3133155.
15. Wangen, G.; Shalaginov, A.; Hallstensen, C. Cyber Security Risk Assessment of a DDoS Attack. Information Security; Bishop, M.; Nascimento, A.C.A., Eds.; Springer International Publishing: Cham, 2016; pp. 183–202.
16. Soni, M.; Rajput, B.S. Security and Performance Evaluations of QUIC Protocol. Data Science and Intelligent Applications; Kotecha, K.; Piuri, V.; Shah, H.N.; Patel, R., Eds.; Springer Singapore: Singapore, 2021; pp. 457–462.
17. HTTP Archive's annual state of the web report. https://almanac.httparchive.org/en/2020/http2#fig-4, accessed on 27 Apr 2021.
18. QUIC Loss Detection and Congestion Control - draft-ietf-quic-transport-34, 2021.
19. Bhat, D.; Rizk, A.; Zink, M. Not so QUIC: A Performance Study of DASH over QUIC; Association for Computing Machinery: New York, NY, USA, 2017; NOSSDAV'17, p. 13–18. doi:10.1145/3083165.3083175.
20. HTTP-over-QUIC officially becomes HTTP/3. https://daniel.haxx.se/blog/2018/11/11/http-3/, accessed on 09 Mar 2021.
21. Bhat, D.; Deshmukh, R.; Zink, M. Improving QoE of ABR Streaming Sessions through QUIC Retransmissions. Proceedings of the 26th ACM International Conference on Multimedia; Association for Computing Machinery: New York, NY, USA, 2018; MM '18, p. 1616–1624. doi:10.1145/3240508.3240664.
22. Arisu, S.; Yildiz, E.; Begen, A.C. Game of Protocols: Is QUIC Ready for Prime Time Streaming? *Int. J. Netw. Manag.* **2020**, *30*. doi:10.1002/nem.2063.
23. Chrome is deploying HTTP/3 and IETF QUIC. https://blog.chromium.org/2020/10/chrome-is-deploying-http3-and-ietf-quic.html, accessed on 27 Apr 2021.
24. TCP Congestion Control, 1999.
25. Ha, S.; Rhee, I.; Xu, L. CUBIC: A New TCP-Friendly High-Speed TCP Variant **2008**. *42*, 64–74. doi:10.1145/1400097.1400105.
26. Jain, R.; Chiu, D.; Hawe, W. A Quantitative Measure Of Fairness And Discrimination For Resource Allocation In Shared Computer Systems. *CoRR* **1998**, *cs.NI/9809099*.
27. quic-go. https://github.com/lucas-clemente/quic-go/tree/master, accessed on 09 Mar 2021.
28. golang http. https://golang.org/pkg/net/http/, accessed on 09 Mar 2021.
29. linux tc. https://man7.org/linux/man-pages/man8/tc.8.html, accessed on 09 Mar 2021.
30. linux ss. https://man7.org/linux/man-pages/man8/ss.8.html, accessed on 09 Mar 2021.
31. dash.js. https://github.com/Dash-Industry-Forum/dash.js/wiki, accessed on 09 Mar 2021.
32. Spiteri, K.; Urgaonkar, R.; Sitaraman, R.K. BOLA: Near-optimal bitrate adaptation for online videos. IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, 2016, pp. 1–9.
33. Taibi Guguen, C.; Le Bolzer, F.; Houdaille, R. Improving User Experience when HTTP Adaptive Streaming Clients Compete for Bandwidth. *SMPTE Motion Imaging Journal* **2017**, *126*, 28–34. doi:10.5594/JMI.2016.2632279.
34. Bentaleb, A.; Taani, B.; Begen, A.C.; Timmerer, C.; Zimmermann, R. A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP. *IEEE Communications Surveys Tutorials* **2019**, *21*, 562–585. doi:10.1109/COMST.2018.2862938.
35. Petrangeli, S.; Hooft, J.V.D.; Wauters, T.; Turck, F.D. Quality of Experience-Centric Management of Adaptive Video Streaming Services: Status and Challenges. *ACM Trans. Multimedia Comput. Commun. Appl.* **2018**, *14*. doi:10.1145/3165266.
36. Wang, P.; Bianco, C.; Riihijärvi, J.; Petrova, M. Implementation and Performance Evaluation of the QUIC Protocol in Linux Kernel; Association for Computing Machinery: New York, NY, USA, 2018; MSWIM '18, p. 227–234. doi:10.1145/3242102.3242106.
37. Kosek, M.; Shreedhar, T.; Bajpai, V. Beyond QUIC v1: A First Look at Recent Transport Layer IETF Standardization Efforts. *IEEE Communications Magazine* **2021**, *59*, 24–29. doi:10.1109/MCOM.001.2000877.
38. Papastergiou, G.; Fairhurst, G.; Ros, D.; Brunstrom, A.; Grinnemo, K.J.; Hurtig, P.; Khademi, N.; Tüxen, M.; Welzl, M.; Damjanovic, D.; Mangiante, S. De-Ossifying the Internet Transport Layer: A Survey and Future Perspectives. *IEEE Communications Surveys Tutorials* **2017**, *19*, 619–639. doi:10.1109/COMST.2016.2626780.