A first look at HTTP/3 adoption and performance

A First Look at HTTP/3 Adoption and Performance

Gianluca Perna[a], Martino Trevisan[a,*], Danilo Giordano[a], Idilio Drago[b]

[a] *Politecnico di Torino, Italy*
[b] *University of Turin, Italy*

Abstract

The third version of the Hypertext Transfer Protocol (HTTP) is in the final standardization phase by the IETF. In addition to better security and greater flexibility, it promises performance benefits. HTTP/3 uses a more efficient header compression scheme and replaces TCP with QUIC, a transport protocol over UDP that was originally proposed by Google and is also currently being standardized. Although initial implementations of HTTP/3 already exist and some websites have announced their support, few studies have been conducted to assess its benefits.

We measure the adoption and performance of HTTP/3 and show how it has been adopted by some of the leading Internet companies such as Google, Facebook, and Cloudflare in 2020. We conduct a large-scale measurement campaign on thousands of websites using HTTP/3 to understand the extent to which it outperforms HTTP/2 in web browsing applications. We find that websites using HTTP/3 often host most web page objects on third-party servers that only support HTTP/2 or even HTTP/1.1. Websites that load objects from a limited number of third-party domains are the ones that see larger performance gains. However, our experiments show that HTTP/3 offers significant benefits only in high-latency or mobile networks. Finally, we run an experimental campaign to study the impact of HTTP/3 on video streaming applications. In this direction, our results show that HTTP/3 currently does not provide benefits.

Keywords: HTTP/3; Performance; Measurements.

1. Introduction

The Hypertext Transfer Protocol (HTTP) is used to access the vast majority of services on the Internet, from websites to social networks and collaborative platforms. HTTP was born in the early 90s, and its first version (HTTP 1.1) was standardized in 1997 [5]. It was not until 2014 that the second version (HTTP/2 [1]) was standardized, including significant changes to the protocol's framing mechanisms. HTTP/3 is the third version of HTTP and is currently in the final standardization phase at the IETF [2]. HTTP/3 promises performance benefits and security improvements over HTTP/2. One major change is that HTTP/3 replaces TCP as the transport layer in favor of QUIC, a UDP-based transport protocol originally proposed by Google and currently an IETF standard [8]. In addition, HTTP/3 introduces a more effective header compression mechanism and uses TLS 1.3 [15] (or higher) to improve security.

HTTP/3 is expected to take the place of HTTP/2 in the next few years, and some of the leading Internet companies have already announced plans to support it starting in 2020, such as CloudFlare CDN[1] and Facebook.[2] However, very few works [17, 13] have examined HTTP/3 deployments. More importantly, the impact of the protocol on Web performance has not been widely measured yet. Such efforts are important to externally validate the benefits of the protocol, which have only been evaluated by the few service providers that have deployed it.

We fill this gap by conducting a large-scale measurement study of HTTP/3 adoption and performance. We first rely on the HTTPArchive dataset[3] to examine the extent to which the Web ecosystem has adopted HTTP/3. Then, we run additional campaigns to measure the benefits introduced by HTTP/3. Considering websites using different versions of the HTTP protocol, we measure various metrics known to indicate user Quality of Experience (QoE). Finally, we emulate different network conditions on the network path to assess whether, and to what extent, HTTP/3 improves performance in different scenarios.

Using the open source HTTPArchive dataset, we find thousands of websites that support HTTP/3. Most of them are hosted by a handful of Internet hypergiants, i.e., Facebook, Google, and Cloudflare. We then automatically visit websites that support HTTP/3 using the different

*Corresponding author: Martino Trevisan (martino.trevisan@polito.it)
Email address: martino.trevisan@polito.it (M. Trevisan), gianluca.perna@polito.it (G. Perna), danilo.giordano@polito.it (D. Giordano), idilio.drago@unito.it (I. Drago).

[1] https://blog.cloudflare.com/http3-the-past-present-and-future/
[2] https://engineering.fb.com/2020/10/21/networking-traffic/how-facebook-is-bringing-quic-to-billions/
[3] https://httparchive.org/

HTTP versions and under different network conditions to measure performance in terms of QoE-related metrics. We visit a total of 14 707 websites while emulating artificial latency, packet loss and limited bandwidth. We perform 2 647 260 visits over a one-month period to determine the benefits of HTTP/3 for normal web browsing activities. We then supplement the analysis with additional ad-hoc campaigns to measure specific aspects, such as to examine mobile browsing scenarios and video streaming usage.

We find that the benefits of HTTP/3 only emerge under certain network conditions and vary significantly across websites. Our main results are:

- Google, Facebook, and Cloudflare are the early adopters of HTTP/3 and host almost the totality of currently websites supporting HTTP/3.
- The majority of web page objects in websites that support HTTP/3 are still hosted on third-party servers that do not support HTTP/3.
- In current deployments HTTP/3 brings significant performance benefits in high latency scenarios and limited benefits in very low bandwidth ones.
- As expected, sites that require fewer connections to load objects benefit the most.
- The benefits of HTTP/3 are significant in mobile scenarios, such as for users browsing from smartphones and tablets.
- Performance gains largely depend on the infrastructure hosting the website, possibly due to optimizations on the server side.
- Adaptive video streaming services do not seem to benefit from HTTP/3 in terms of key QoE-related metrics.

This paper extends our prior work [21] in several directions. First, we extend our temporal analysis to 2021, expanding the scope of the paper. We evaluate the impact of HTTP/3 on mobile browsing scenarios. Moreover, our results now include not only web browsing, but also adaptive video streaming. Finally, to ease reproducibility of our results and enable additional measurement campaigns, the scripts used to set up and run our experiments are now available on GitHub.[4]

The paper is organized as follows: Section 2 describes HTTP/3 and illustrates related work. Section 3 presents our datasets and data collection methodology. Section 4.1 illustrates our results on HTTP/3 adoption, while Section 5 and Section 6 evaluate the performance of HTTP/3 in web browsing and video streaming, respectively. Section 7 discusses our results, while Section 8 concludes the paper.

[4] `https://github.com/SmartData-Polito/h3-benchmark`

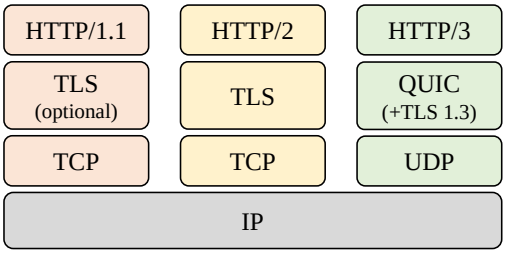


Figure 1: Protocol stack for different HTTP versions

2. Background and related work

2.1. HTTP/3

HTTP/3 is the third version of the well-known Hypertext Transfer Protocol, which was developed in the 1990s for transmitting multimedia content and hyper-textual documents over the Internet. Its version 1.1 was replaced by its second version HTTP/2 only in 2014. HTTP/2 implements several new features, most notably to improve the way data is framed and transported. HTTP/2 promises to make the web faster, although some studies question its benefits [22, 14].

HTTP/3 is currently in the final standardization phase and has reached the 34^{th} draft version [2], making it stable and ready for real-world deployment. Key improvements over the version 2 include more efficient header compression, enhanced security features based on TLS 1.3 and, most importantly, the use of QUIC at the transport layer. The resulting protocol stack is therefore heavily modified, as we show in Figure 1. QUIC, originally developed by Google, is a transport protocol based on UDP [8]. QUIC reworks TCP, moves congestion control to the user space, and enables faster handshaking. It also solves the long-standing problem of head-of-line blocking and allows multiple independent streams within the same connection. QUIC enables independent retransmission of partial streams and decouples them from congestion control. This operation is expected to improve users' QoE with faster website responsiveness, especially in scenarios with poor network conditions. HTTP/3 also mandates the use of TLS 1.3 [15], which is integrated directly into the QUIC layer. Finally, it allows 1-RTT handshakes and 0-RTT resumption, which further reduces session setup time.

2.2. Related Work

Given its recent conception, few works have addressed HTTP/3. Saif *et al.* [17] conduct experiments where they control both the client and the server when accessing a single web page. They study the impact of delay, packet loss, and throughput on HTTP/3 performance without finding major effects. In contrast, we conduct a large-scale measurement campaign that controls only the client and examines thousands of HTTP/3 websites in production. This setup allows us to consider both real-world network conditions and server implementations.

Table 1: Description of the employed datasets.

Dataset	Runs	Goal
HTTPArchive	53 107 185	HTTP/3 Adoption
BrowserTime-Web	2 647 260	Browsing Performance
BrowserTime-Mobile	1 800	Mobile Browsing Performance
BrowserTime-Video	3 60	Video Streaming Performance

Table 2: Network configurations used in the experiments.

Parameter	Tested settings
Latency [ms]	Native, 50, 100, 200
Loss [%]	Native, 1, 2, 5
Bandwidth [Mbit/s]	Native, 5, 2, 1

Marx *et al.* [13] compare 15 HTTP/3 implementations and find great heterogeneity in how congestion control, prioritization, and packetization work. They perform single file downloads without providing large scale measurements in the wild, which we provide here. Cloudflare benchmarks its own HTTP/3 implementation in draft 27 in [19] and finds that it is $1-4\%$ slower than HTTP/2. However, their experiments are limited to the website `blog.cloudflare.com`. Guillen *et al.* [7] propose a control algorithm for adaptive streaming tailored for HTTP/3. Saif *et al.* [16] measure performance benefits of using HTTP/3 instead of MQTT and MQTT-over-QUIC in IoT scenarios. Lovell *et al.* [11] compare HTTP/3 support across millions of websites and show that the most popular websites have not yet explored HTTP/3, while less popular websites have higher HTTP/3 adoption.

QUIC has been the subject of numerous studies. Wolsing *et al.* [23] show that QUIC performs better than TCP thanks to its fast connection establishment. Manzoor *et al.* [12] show that QUIC performs worse than TCP in wireless mesh networks due to the poor interaction of the protocol with the WiFi layer in this scenario. Carlucci *et al.* [3] found that QUIC reduces the overall page load time. Kakhi *et al.* [9] conducted a large-scale measurement campaign on QUIC and found that it outperforms TCP in most cases. However, these works target Google's QUIC versions, while the current IETF standard has made significant progress [10]. Moreover, they focus exclusively on the transport layer and neglect the improvements introduced by HTTP/3, which we measure in this work.

3. Datasets and performance metrics

We rely on several datasets to study (i) the adoption of HTTP/3 and its performance on (ii) normal web browsing, (iii) mobile browsing, and (iv) video streaming. We summarize these datasets in Table 1.

3.1. HTTP/3 Adoption

We examine HTTP/3 adoption using the HTTPArchive, an open dataset available online.[5] The dataset contains metadata derived from visits to a list of more than 5 million URLs provided by the Chrome User Experience Report.[6] The list of URLs is compiled using navigation data from real Chrome users and provides a representative view of the most popular websites and services accessed worldwide.[7] Each month, all URLs visited with the Google Chrome browser are taken from a U.S.-based data center and the resulting navigation data is published. For each visit, the dataset contains information about page characteristics, load performance, and HTTP transactions in HAR format[8] including request and response headers.

Of fundamental importance to our analyzes are the HTTP responses, which contain the eventual `Alt-Svc` header used by servers to announce support for HTTP/3. By setting the `Alt-Svc` header, the server tells the client that subsequent connections can use HTTP/3, while also indicating its support for specific design versions (e.g., 27 or 29).

We download the HTTPArchive dataset from November 2019, when we first observe sites supporting HTTP/3. We monitor the HTTPArchive through September 2021. We use the data to examine the trend of HTTP/3 adoption. The data is 6.6 TB. Since we are interested in examining HTTP/3 adoption on *websites*, we discard all visits to internal pages (less than half of the total) and keep only visits to home pages. We refer to this dataset as *HTTPArchive*.

3.2. HTTP/3 Performance

Our goal is to compare the performance of the three HTTP versions when accessing heterogeneous types of content. To this end, we collect three datasets: (i) *BrowserTime-Web* , including visits to websites supporting HTTP/3 from a regular browser, (iii) *BrowserTime-Mobile* , targeting mobile websites under mobile network conditions, and (iii) *BrowserTime-Video* , targeting video streaming.

3.2.1. Web browsing

To automate website testing, we rely on BrowserTime, a docked tool for performing automated visits to websites with a large number of configurable parameters.[9] We use BrowserTime to instrument Google Chrome to visit web pages with a specific HTTP version. Importantly for our goal, Google Chrome provides the ability to specify a set of domains to be contacted on the first visit using HTTP/3, i.e., without prior specification via the `Alt-Svc` header. We restrict ourselves to Chrome, since we are not aware of similar features in other browsers (e.g., Firefox).

[5] `https://httparchive.org/`, visited on February 4, 2021.

[6] `https://developers.google.com/web/tools/chrome-user-experience-report`

[7] HTTPArchive previously adopted the Alexa Top 1M Websites list, but switched to the Chrome User Experience Report when Alexa discontinued its ranking in July 2018.

[8] `http://www.softwareishard.com/blog/har-12-spec/`

[9] `https://www.sitespeed.io/documentation/browsertime/`

We are interested in studying the impact of HTTP/3 under different network conditions. For this reason, we perform our measurements under different *network configurations*.[10] We conduct our experiments with two high-end servers connected to the Internet via 1 Gbit/s Ethernet and located on our university campus. We call this baseline scenario *Native*, as indicated in Table 2.

We then enforce the network configurations during the visits using the Linux tool `tc` tool. For each network configuration, we change one of the three network parameters enforcing: (i) additional latency or (ii) additional packet loss or (iii) bandwidth limit. For each parameter, we use 4 different settings listed in Table 2. In the case of latency, we simulate an increasing *Round Trip Time* (RTT) and therefore only apply it in one link, namely the uplink. For loss and bandwidth constraint, we enforce the configuration for both uplink and downlink. For each network configuration, we visit each website (i) with only HTTP/1.1, (ii) with HTTP/1.1 and HTTP/2, and (iii) with all three versions of the protocol. All visits to the same website are performed sequentially, cleaning up all state between repetitions, i.e., browser cache, TCP connections, etc.

We collect the *BrowserTime-Web* dataset by visiting websites that currently support HTTP/3. At the time we run this experimental campaign (December 2020), we find 14 707 websites that announce support for HTTP/3 and test them all.

The visits are repeated 5 times to get more reliable results. So we visit each website 4 *times*$3 \times 3 \times 5 = 180$ times. Next, we visit these websites with three HTTP versions (HTTP/1.1, HTTP/2, and HTTP/3) to quantify potential performance improvements. In total, we performed 2 647 260 visits over a period of one month. The metadata of the visits accounts for 189 GB, and we call this dataset *BrowserTime*.

3.2.2. Browsing under mobile networks

We also evaluate the impact of HTTP/3 for mobile users, i.e., users of smartphones or tablets connected via 3G or 4G mobile networks. We conduct an additional measurement campaign in which we emulate both mobile devices and mobile network conditions.

For the former, we rely on BrowserTime's ability to mimic mobile devices by setting the appropriate user agent string in Google Chrome and limiting the size of the view port when rendering the page. We emulate an iPhone 6 and an iPad tablet. For the latter, we use ERRANT [20], a data-driven open-source emulator for mobile access networks. Briefly, ERRANT uses more than 100 thousand speed-test measurements obtained from real mobile networks to simulate network profiles for different Radio Access Technologies (RATs) (3G or 4G) and signal strengths (bad, medium, and good). Each network profile describes both typical behavior and inherent network variability. Then, ERRANT uses the Linux tool tc-netem Linux tool to enforce the selected network profile that emulates both the typical behavior and the network variability.

In this experimental campaign, we target a random subset of 100 websites that support HTTP/3. We have reduced our sample in this experiment to limit both the time needed to complete the measurements and the generated traffic. To ensure a general coverage of the list of websites used in other experiments, we perform a stratified random sampling across content providers. Specifically, we take 25 websites for the top three content providers (Google, Facebook, and CloudFlare) plus 25 from the remaining websites.

We visit each website using both emulated devices (tablet and smartphone). As an additional comparison step, we revisit the website with the default desktop setting, as in the *BrowserTime* dataset. We test the websites with 6 ERRANT profiles, namely the combination of the two RATs (3G and 4G) and three signal strengths (bad, medium and good). For each profile, we run 10 experiments, firing a total of 54 000 visits. We refer to this dataset as *BrowserTime-Mobile* .

3.2.3. Video Streaming

In addition to web browsing performance, we evaluate the impact of HTTP/3 on video streaming. Among the dozens of protocols for video streaming, most providers have moved to solutions based on streaming over HTTP. The most widely used solution is called Dynamic Adaptive Streaming over HTTP (DASH), which splits the video into chunks of a few seconds that the client retrieves via HTTP requests.

DASH supports adaptive streaming by allowing the client to choose the best video resolution among those available on the server, depending on network conditions. We run an experimental campaign for video streaming in a controlled test environment.

Popular commercial video streaming services such as Twitch, Prime Video, or Netflix do not yet support HTTP/3 by the time of writing. YouTube, on the other hand, supports HTTP/3 and HTTP/1.1, but surprisingly not HTTP/2. As our goal in this paper is to assess the benefits of HTTP/3 considering *also* HTTP/2, no streaming service currently in production could serve as basis for our analysis. We therefore prefer to use a controlled environment on both the client and server side to measure performance across protocols.

In our setup, an instrumented browser runs a DASH web client that plays a video hosted on our server set to support HTTP/1.1, HTTP/2, and HTTP/3. We use the popular and open source player `Dash.js`[11] and a `nginx` web server set to use Cloudflare's *quiche* HTTP/3 and QUIC implementation.[12] The server hosts a 9-minute video delivered

[10] We have included configurations covering the typical network conditions previously observed in real measurements [20]

[11] `https://reference.dashif.org/dash.js/`

[12] `https://docs.quic.tech/quiche/`

in 150 chunks, available in 10 bitrates ranging from 250 kbit/s to 14 Mbit/s.

We test the same network conditions as in Table 2 with all three HTTP versions. Each experiment lasts 9 minutes and we repeat it 10 times. In total, we run 360 video sessions. We call this dataset *BrowserTime-Video* .

3.3. Performance metrics

We rely on different performance metrics for the web browsing and video streaming scenarios. For each case, we select metrics that are known to be good proxies for users' Quality of Experience (QoE).

First, BrowserTime collects various statistics during emulated web browsing, including QoE-related performance metrics. We track two metrics that are correlated with users' QoE [4] during web browsing:

- **onLoad**: The time when the browser fires the `onLoad` event –i.e., when all elements of the page, including images, stylesheets and scripts, have been downloaded and parsed;
- **SpeedIndex**: Suggested by Google,[13] it represents the time at which the visible parts of the page are displayed. It is computed by recording the video of the browser screen and tracking the visual progress of the page during rendering.

Note that we use these metrics for both mobile and non-mobile browsing.

For video streaming, we rely on the following QoE-related metrics [18]:

- **Video resolution**: Image quality is fundamental to QoE and can be estimated from video resolution. We determine the bitrate by parsing the requested URLs. We then calculate both the average encoding bitrate per video session and the number of requests for chunks in each bitrate.
- **Playback Startup Delay (*PSD* for brevity):** This is the time between the user request for a video and the start of playback. Most players wait until a buffer (a few seconds) is filled before starting playback. We calculate the startup delay by measuring the time until the client receives the first video chunk.
- **Frequency of video downscale:** We evaluate how video resolution evolves in video sessions and track resolution switches. While switching is normal for adaptive video (e.g., to prevent video freezes), frequent switching affects QoE. We count how often the browser experiences a *downscale* in the requested bitrate.

[13] `https://web.dev/speed-index/`

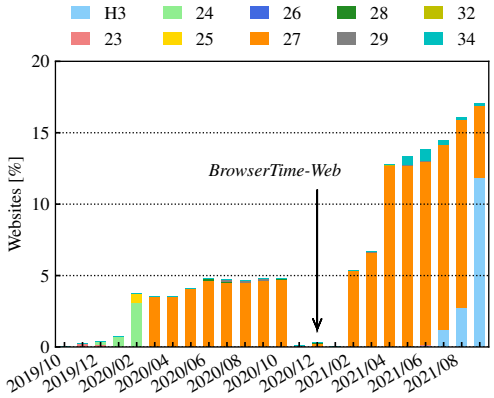


Figure 2: Percentage of websites in HTTPArchive that announce support to HTTP/3, separately by IETF draft (*HTTPArchive* dataset).

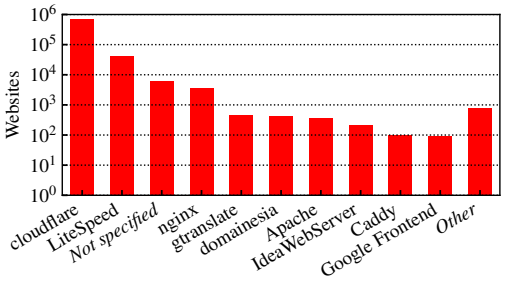


Figure 3: `Server` in HTTP response (December 2020) (*HTTPArchive* dataset).

4. Dissecting HTTP/3 adoption

We now provide an overview of the adoption of HTTP/3. Since announcing HTTP/3 support does not equate to delivering content over this protocol, we also quantify the amount of content delivered over HTTP/3.

4.1. Websites supporting HTTP/3

We use the *HTTPArchive* dataset to examine the extent to which HTTP/3 has been adopted since it was first proposed. The first IETF draft was published in January 2017, but we do not observe the first websites adopting HTTP/3 until late 2019. Since then, the number of websites supporting HTTP/3 has steadily increased. Figure 2 shows the trend for the last months of 2019, all of 2020, and the first 9 months of 2021. Using the `Alt-Svc` header, we can observe the HTTP/3 draft version supported by the server, shown with different colors in the figure. In case a website provides more than one version, we consider the last one seen in *HTTPArchive*. As of September 2021, we observe a significant number of websites supporting draft_34, which is on its way to becoming the final IETF standard for HTTP/3.

The figure shows that the number of websites supporting HTTP/3 has slowly increased, reaching 0.7 % of the total

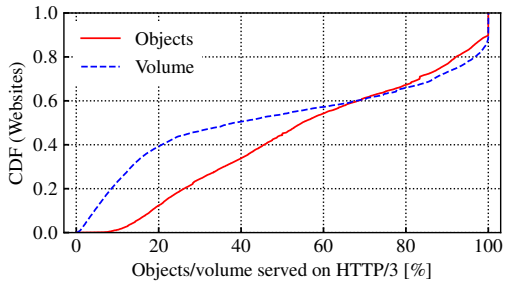


Figure 4: Share of objects/volume served using HTTP/3 on enabled websites (*BrowserTime-Web* dataset).

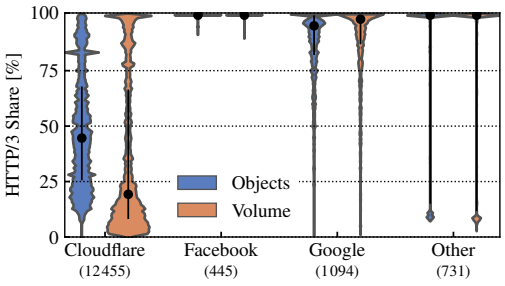


Figure 5: Share of objects/volume served on HTTP/3, separately by provider (*BrowserTime-Web* dataset).

in early 2020. At that time, only Google and Facebook offered HTTP/3 for their websites. In February 2020, the number of websites supporting HTTP/3 jumped. This increase was due to CloudFlare enabling HTTP/3 for the websites it hosts. The percentage of websites supporting HTTP/3 increases over 4 % during this period, reaching a maximum of 4.8 % of websites (203 k) in October 2020. In November, the number of websites suddenly dropped to less than 0.1% (4 024 in absolute numbers). This gap was caused by CloudFlare suspending support for HTTP/3 due to performance issues.[14] CloudFlare re-enabled HTTP/3 for a subset of websites in December 2020. Since our *BrowserTime-Web* campaign took place in December 2020, we only consider these 14 707 websites for the following results.

The number of HTTP/3enabled websites remained low in the following months, mainly due to changes in the Cloudflare CDN configuration. Starting from February 2021, Cloudflare finally enabled HTTP/3 and we observe that the number of enabled websites returned to the October 2020 level. Since then, HTTP/3 support has reached 17.01%. In September 2021, 11.84% of websites announce support for the latest version of HTTP/3.

The majority of websites supporting HTTP/3 are hosted by large enterprises running their own server applications. We break down these numbers in Figure 3, which lists the 10 most popular servers, as indicated in the HTTP `Server` header. As expected, CloudFlare hosts the most websites that support HTTP/3 (672 909, notice the log y- scale). In second position, we find LiteSpeed, a high-performance web server that supports HTTP/3. Looking at the server IP addresses, we notice that some popular cloud providers use it (e.g., OVH). For 5 957 websites, there is no reference to the `server` in the HTTP responses, and most of them belong to Facebook's domains - e.g., `facebook.com` and `instagram.com`. Google also supports HTTP/3, with `gtranslate` (Google Translate) and Google front-end servers. The remaining websites run other servers (e.g., nginx and Apache).

[14] `https://community.cloudflare.com/t/community-tip-http-3-with-quic/117551`, visited on 2/20/2021.

4.2. Content served over HTTP/3

Next, we examine the extent to which objects are served by enabled websites using HTTP/3. This is because even if a website supports HTTP/3, not *all* of its objects are served over HTTP/3. Objects may be downloaded from external CDNs, cloud providers, or third-parties not supporting the same protocol. This is the case, for example, with ads and trackers, which are usually hosted on a different third-party infrastructure. We use the *BrowserTime-Web* dataset, which allows us to observe the protocol used to deliver each object that makes up the websites we visit.

In Figure 4, we consider all visits made with HTTP/3 enabled. For each visit, we compute the fraction of objects served via HTTP/3. Since each website is accessed multiple times, we calculate the average of the values over all visits. It is clear that at least the main HTML document is always sent over HTTP/3, but the remaining objects can be served with older HTTP versions. The figure shows the distribution of the percentage of objects transmitted over HTTP/3 (solid red line) and the byte-wise distribution (dashed blue line). Note that in 18% of cases all objects are delivered over HTTP/3, which means that the web page contains only elements hosted on HTTP/3-enabled servers. Web pages with 90% or more objects (volume) on HTTP/3 are 36 (41) % and only 9 (28) % have less than 20 % of objects (volume).

Next, we break down the above analysis by provider – i.e., by the company/CDN hosting the website. We get it by looking at the HTTP header `server`, the name of the website, and the IP addresses of the server. As shown in Figure 3, we find that HTTP/3 is mainly used by (i) Cloudflare CDN, (ii) Facebook, and (iii) Google. The remaining 595 websites (i.e., Other) largely belong to self-hosted websites running updated versions of the *nginx* web server.

Figure 5 shows the percentage of objects and volume served over HTTP/3, separately by provider. Websites hosted by Cloudflare tend to be more heterogeneous, with half of the objects accessed via non-HTTP/3 servers (on median). In addition, only 24% of the volume is served via HTTP/3. This is likely due to the diversity of websites that rely on the provider. These websites may use

complex web pages consisting of multiple third-party objects stored on external providers that do not yet rely on HTTP/3. In contrast, Facebook and Google serve almost all objects using HTTP/3. For Google, the long tail of the distribution is due to Blogspot, where the creator can add content from external sources. Finally, if we look at the Other category, almost all objects are served using HTTP/3. These websites are usually simple and consist of a few objects stored on the same self-hosted servers along with the main HTML document.

5. Web Browsing performance

Now we investigate how HTTP/3 affects QoE-related web browsing performance metrics and whether the observed improvements can be related to the provider hosting the content (Section 5.1), website characteristics (Section 5.2), or mobile networks (Section 5.3).

We now investigate the impact of HTTP/3 on website performance. For this purpose, we use the *BrowserTime-Web* dataset in which the 14 707 web pages were visited multiple times under different network conditions. In addition to computing performance in the native scenario (i.e., 1 gpbs Ethernet in a campus network), we use `tc-netem` to enforce additional latency, packet loss, and bandwidth constraints. We then contrast the QoE-related performance indicators of the site (onLoad and SpeedIndex) by (i) displaying their absolute value and (ii) computing a metric that we call *H3 Delta*. Given a website and a given network scenario, we obtain the *H3 Delta* as the relative deviation of the metric when using HTTP/3 ($h3$) instead of HTTP/2 ($h2$). Since we always perform 5 visits for each case, we consider the median values. The *H3 Delta* for a website w in scenario s is calculated as follows:

$$\text{H3-Delta}(w,s) = \frac{\text{median}(w,s,h3) - \text{median}(w,s,h2)}{\max(\text{median}(w,s,h3), \text{median}(w,s,h2))} \quad (1)$$

By definition, H3 Delta(w,s) is bound in $[-1,1]$ and negative if a website loads faster under HTTP/3, and positive if not. We calculate the *H3 Delta* for both onLoad and SpeedIndex.

We illustrate how the values of the metric change under different network conditions by first focusing on the additional latency in Figure 6. Using boxplots, we show the distribution of onLoad (top) and SpeedIndex (bottom), separately by HTTP version (coloured boxes). The boxes range from the first to the third quartile, the whiskers indicate the 10^{th} and the 90^{th} percentiles, while the black dashes represent the median. When no additional latency is added (*native* case), we observe that the median onLoad time is about 2s, while SpeedIndex is about 1s, without much difference between HTTP versions. When adding additional latency, websites load slower as more time is needed to download the page objects, 6 seconds on median with 200 ms of additional latency. Not shown here for brevity, also packet loss and limited bandwidth also cause

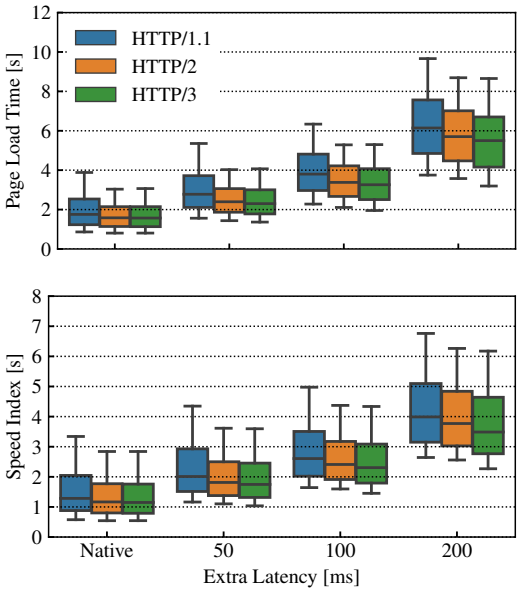


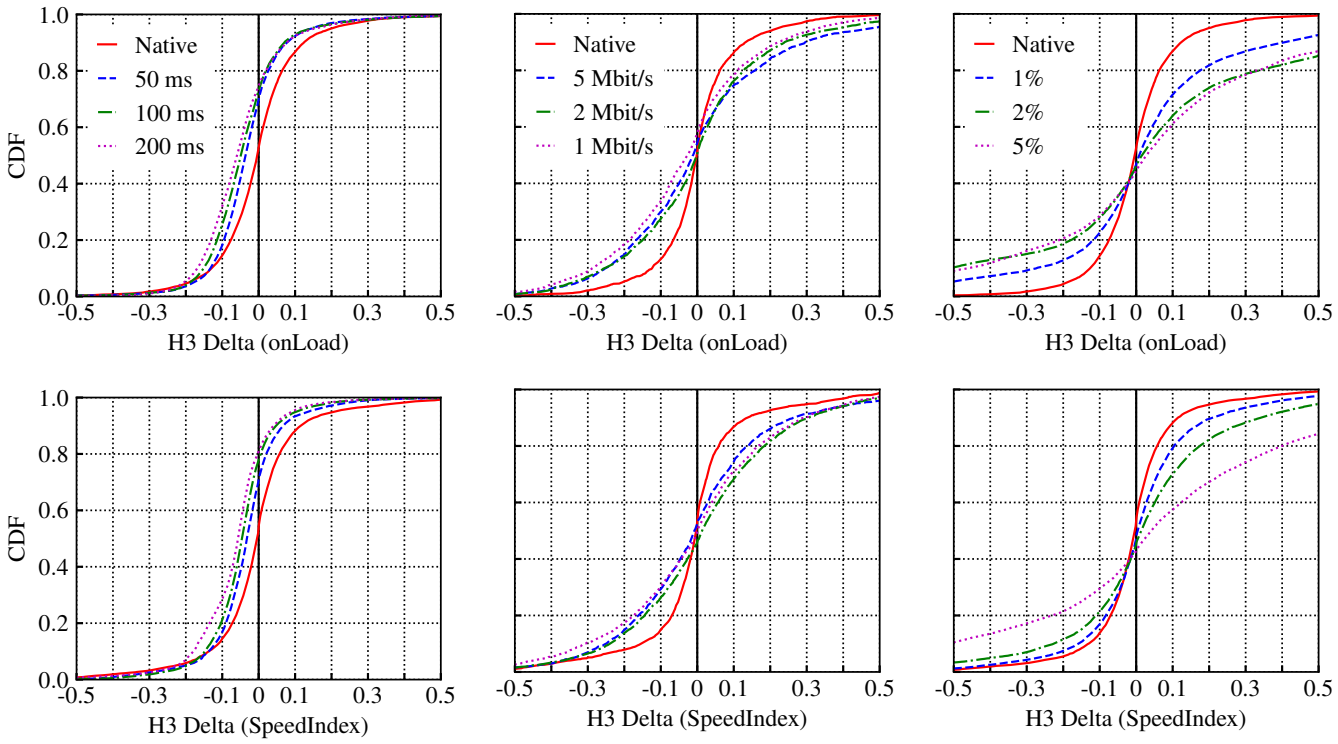
Figure 6: onLoad (top) and SpeedIndex (bottom) with extra latency, separately for HTTP/1.1, HTTP/2 and HTTP/3 (*BrowserTime-Web* dataset).

a similar degradation in performance indicators. Figure 6 shows that HTTP/1.1 has the worst performance at high latency, while HTTP/3 shows the greatest benefits. With an additional latency of 200 ms, websites onLoad in a median of 6.4, 5.8 and 5.4 s with HTTP versions 1.1, 2, and 3, respectively.

To better capture the differences between HTTP/3 and HTTP/2, we now examine the *H3 Delta* in Figure 7, where we show the distribution across the 14 707 websites for both onLoad (top row) and SpeedIndex (bottom row). The three columns respectively refer to scenarios with additional latency, limited bandwidth, and packet loss, respectively. Solid red lines represent the *native* case. Dashed lines represent scenarios with emulated network conditions as indicated in Table 2.

Starting with latency, we confirm what has already been shown in Figure 6. In the native case, we do not observe a general trend: looking at the solid red lines, we find that in about 50 % of cases, websites load faster with HTTP/3, and in the remaining cases HTTP/3 is slower. When latency is high, HTTP/3 offers significant advantages over HTTP/2. With an additional 50 ms of latency, 70 (71) % of websites have a lower onLoad time (SpeedIndex), meaning they load faster. The number of websites that load faster increases to 73 (77) % at a latency of 100 ms latency. At 200 ms, the number of websites that load faster reaches 74 (77) %, and the median *H3 Delta* is −0.059 (−0.056).

When we focus on bandwidth-limited experiments (central plots in Figure 7), different considerations hold. We observe (limited) benefits only for the onLoad time when bandwidth is limited to 1 Mbit/s, with 57 % of websites loading faster with HTTP/3. Note that this benefit does not come from indirect higher latency due to queuing de-



(a) Latency. (b) Bandwidth. (c) Packet loss.

Figure 7: *H3 Delta* on different scenarios. onLoad (top) and SpeedIndex (bottom). Negative values indicate that HTTP/3 is faster (*BrowserTime-Web* dataset).

lays (also called bufferbloat), since we limit host queues to 32 kB. In other cases, no clear trend emerges, but we do notice greater variability in the *H3 Delta* measure due to the constrained configuration. For example, in the case of SpeedIndex, 52, 45, 49 % of websites load faster with HTTP/3 with 5, 2 and 1 Mbps bandwidth. Similar considerations apply to packet loss (rightmost graphs in Figure 7). Despite greater variability, we cannot see an overall trend, and the *H3 Delta* values are evenly distributed above and below 0.

In summary, we observe limited improvements in onLoad time with very low bandwidth and substantial benefits for both metrics in the high latency case. We do not testify performance improvements of HTTP/3 in high packet loss scenarios. In fact, in several tested cases, some websites may even perform worse when HTTP/3 is enabled.

5.1. HTTP/3 performance by provider

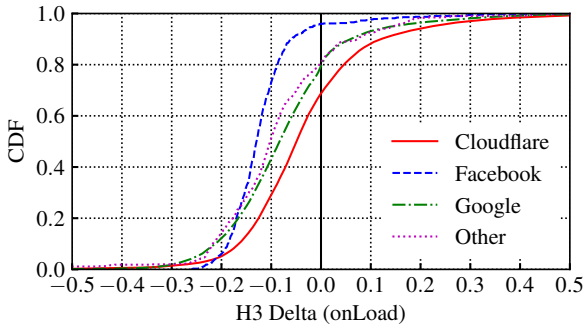
Next, we investigate whether the performance gains of HTTP/3 might be related to the provider hosting the websites. Since we observed considerable performance benefits for HTTP/3 only in cases with high latency or low bandwidth, we limit our analysis to these cases.

Figure 8 shows the distribution of *H3 Delta* for onLoad, separated by provider. We focus on scenarios with 200 ms of additional latency and 1 Mbit/s bandwidth limit. We find that the *H3 Delta* varies significantly by provider. When we focus on latency (Figure 8a), Facebook websites show the highest performance gain (*H3 Delta* −0.13 in median), which is represented by the blue dashed line in the figure. In addition, 95% of websites load faster with HTTP/3 than with HTTP/2. Cloudflare (red solid line) shows the lowest benefits, with only 72% of websites loading faster. Google and the rest of the websites are in the middle. Similar considerations hold for SpeedIndex, which is not shown here for brevity.

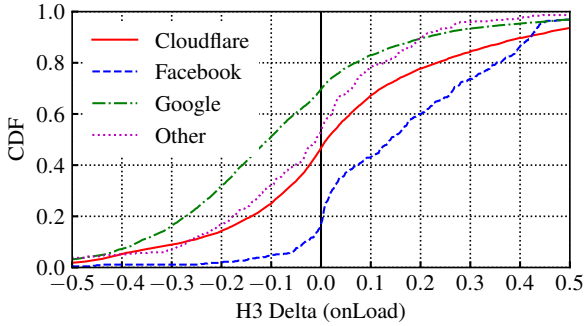
With limited bandwidth (Figure 8b), we observe a completely different situation. Here, Facebook generally has the worst performance with HTTP/3, with 91% of its websites loading faster with HTTP/2. Conversely, Google (green dashed line) shows the best values, with a median *H3 Delta* −0.14 and 79% of its websites loading faster with HTTP/3. Cloudflare and the rest of the websites do not show a clear trend, with about half of the websites loading faster with HTTP/3.

5.2. Page characteristics

Now we examine the characteristics of the pages and possible correlations with performance when using HTTP/3. To this end, we compute several metrics that describe the loading process of a web page and contrast them to understand if they have correlations with *H3 Delta*. For each visit to the 14 707 websites in our dataset, we calculate the following metrics in addition to the*H3 Delta*:



(a) Latency.



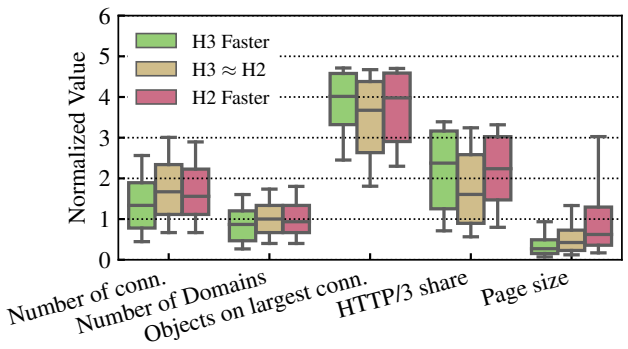
(b) Bandwidth.

Figure 8: onLoad *H3 Delta* by website provider for scenarios with extra-latency and bandwidth limit (*BrowserTime-Web* dataset).

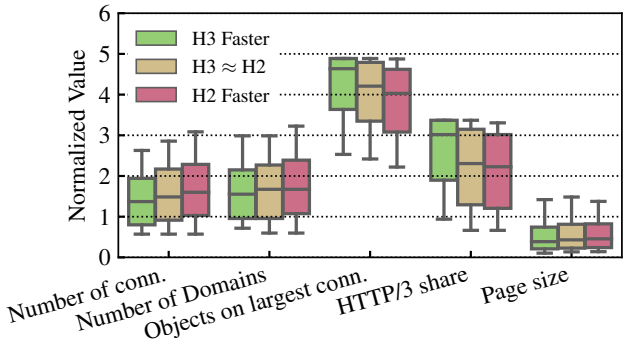
- **Number of connections** made by the browser to load the website when using HTTP/3.
- **Number of domains** contacted while loading the website (i.e., including third-party domains).
- **Share of objects on the largest connection**, which measures the percentage of objects transferred over the connection where the most objects were requested. Recall that HTTP/3 best practices recommend avoiding domain splitting to improve performance.
- **Share of objects served on HTTP/3**, which is used to investigate possible correlations between the share of objects transferred over HTTP/3 (in Figure 4) and the *H3 Delta* metric.
- **Page Size** to break down performance for small and large web pages.

In Figure 9, we compare the distribution of the above metrics and group the websites by classes defined by the onLoad *H3 Delta*:

- **H3 Faster**: websites loading faster with HTTP/3, i.e., onLoad *H3 Delta* < -0.1.
- **H3 $\approx$ H2**: websites having a similar loading time in HTTP/2 and HTTP/3, i.e., onLoad *H3 Delta* $\in [-0.1, 0.1]$.



(a) Latency.



(b) Bandwidth.

Figure 9: Visit characteristics vs. *H3 Delta* class (normalized values, *BrowserTime-Web* dataset).

- **H2 Faster**: websites loading faster with HTTP/2, i.e., onLoad *H3 Delta* > 0.1.

In the figure, boxes with different colors represent these three classes. The y- axis represents the metrics normalized by scaling to a unit variance for ease of visualization and comparison. Again, we examine the scenarios with added latency (top row) and limited bandwidth (bottom row) as they provide the most interesting insights. With additional latency, H3 Faster websites are 32%, H2 Faster 11% and H3 $\approx$ H2 are 57%. With limited bandwidth, they are 38%, 25% and 37%, respectively.

We first focus on the left group of boxes in Figure 9, which shows the (normalized) number of connections the browser established to load the web page. Green boxes indicate that websites that make fewer connections (smaller metric values) are faster with HTTP/3 than with HTTP/2. This is true for both scenarios, i.e., low latency and low bandwidth. Similar considerations apply when we focus on the second set of boxes, which represents the number of domains contacted. Indeed, we find that the number of connections per site and the number of domains contacted are 0.91-correlated (Pearson correlation). The third set of boxes provides a similar perspective, measuring how web page objects are split across multiple connections/domains. The web pages that benefit the most from

HTTP/3 are those that tend to collect objects on a single connection – see the highest position of the green boxes, which means more objects are on a single connection. This is very clear when bandwidth is limited (Figure 9b) and not when latency is high (Figure 9a).

Serving most objects using HTTP/3 (instead of HTTP/2) also has a positive effect, as we can see from the fourth group of boxes in Figure 9. Again, this is most evident in the case of bandwidth limitations (Figure 9b), while it is hard to see a clear trend in the case of additional latency (Figure 9a). Finally, the case of web page size (last box group) is interesting. In high latency scenarios, the websites that benefit from HTTP/3 are small, while large websites tend to do better with HTTP/2. When bandwidth is tight, the picture is more even: Again, it is the small websites that load faster with HTTP/3, albeit only moderately.

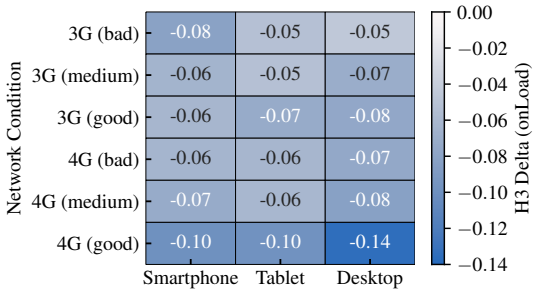
In summary, those websites that benefit from HTTP/3 are those that limit the number of connections and third-party domains, fully adopt HTTP/3 for all website objects, and limit page size. These considerations hold for high latency or limited bandwidth scenarios, while we do not observe a clear trend for optimal network conditions or high packet loss, where the distributions of the metrics largely overlap.

5.3. Performance for mobile users

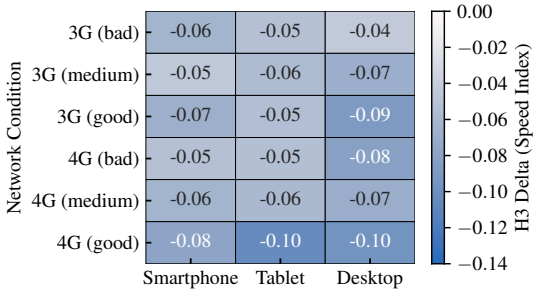
We now use the *BrowserTime-Mobile* dataset to investigate how HTTP/3 impacts the browsing performance of Internet users. To this end, we evaluate the impact of using different devices (i.e., smartphone and table) that we emulate with the BrowserTime features. Then, as described in Section 3.2.2, we emulate different mobile network conditions using ERRANT [20]. Here, we restrict our analysis to the 100 selected websites and measure the gain of HTTP/3 by computing the *H3 Delta*.

To give a compact overview of our results, we plot the median *H3 Delta* across all 100 websites in Figure 10 in the form of a heatmap, separately by OnLoad (Figure 10a) and SpeedIndex (Figure 10b). We emulate three different devices: an iPhone 6 smartphone, an iPad tablet, and the default desktop version of Chrome. We arrange them in the columns of the heatmap. The different rows represent the 6 ERRANT network profiles, including 2 Radio Access Technology (RATs) (3G and 4G) and three signal quality levels. We refer the reader to [20] for the details of the emulated network conditions. In summary, ERRANT emulates the typical latency, uplink and downlink bandwidths measured in a large-scale speed test measurement campaign with 4 European network operators. Importantly, ERRANT emulates not only the average conditions, but also their variability measured during the speed tests at training time.

Starting from OnLoad in Figure 10a, we find that HTTP/3 performs better than HTTP/2 in all scenarios – note the negative median of the H3 Delta between -0.05 and -0.14. Websites that benefit from HTTP/3 range



(a) onLoad.



(b) SpeedIndex.

Figure 10: Web Browsing Performance of Mobile Users, separately by user device type and emulated network (*BrowserTime-Mobile* dataset).

from 66% to 88% depending on device and network conditions. Desktop websites generally see the greatest improvements – see the last column in the figure-but smartphones and tablets also see improvements. In terms of network conditions, the improvements are more pronounced on Good 4G, the best network profile we tested. Similar conclusions apply to the SpeedIndex (Figure 10b). Visits with an emulated mobile device cause the browser to render the *mobile* versions of the web pages. As a result, they tend to be easier to render and therefore smaller in terms of bytes. In fact, the average page size is 680 kB for desktop versions, which reduces to 630 kB in the case of tablet versions and 402 kB for smartphone versions. We also find that latency on mobile networks is on the order of $50 - 150$ms, as measured in the real experimental campaign behind ERRANT [20]. These results are consistent with those in Section 5 and show that HTTP/3 offers significant benefits in high latency scenarios.

Figure 10 clearly provides a rough median of the H3 Delta, but the absolute metrics exhibit large variability due to the varying characteristics of the 100 websites and the variable network conditions imposed by ERRANT. We illustrate this variability using the Good 4G profile as an example by plotting the distribution of onLoad time using violin plots in Figure 11. The distributions are wide, as we notice from the black vertical line within each violin, which represents the interquartile range (IQR). For example, in the case of a smartphone using HTTP/1.1, the onLoad time ranges from 2s to 4s. For HTTP/2 and 3, the onLoad

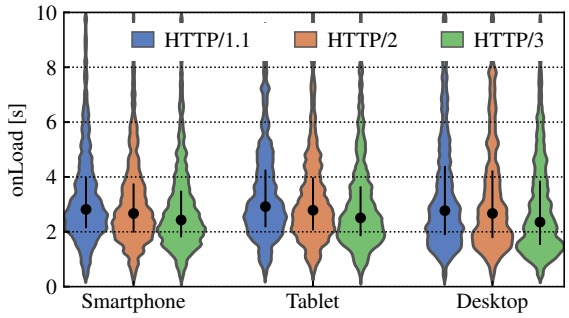


Figure 11: onLoad time with emulated 4G good network (*BrowserTime-Mobile* dataset).

time tends to decrease. The median is 2.8s for HTTP/1.1, which drops to 2.6s for HTTP/2 and 2.4s for HTTP/3. Similar considerations hold for the other cases, with the median onLoad value decreasing by about 0.2s for newer HTTP versions. In summary, our experiments show that HTTP/3 provides significant benefits in mobile networks, even when users use mobile devices that retrieve the lighter mobile versions of websites.

6. Performance of Adaptive Video Streaming

6.1. Metrics

In this section, we study the performance of HTTP/1.1, HTTP/2, and HTTP/3 in video streaming using the *BrowserTime-Video* dataset. To do so, we conduct experiments with different network conditions NC and measure the quality of video streaming using metrics known to correlate with subjective QoE [6, 18]. In particular, we focus on three QoE-related metrics, namely: (i) *video resolution*, (ii) *playback startup delay (PSD for brevity)*, and (iii) *number of downscale adjustments*.

To ease the comparison between different HTTP versions, we compute the *Speedup* of PSD similar to the H3-Delta defined in Equation 1. Specifically, given two experiments j and k, we compute the speedup as follows:

$$\text{Speedup}(j,k) = \frac{PSD(j) - PSD(k)}{\max(PSD(j), PSD(k)} \quad (2)$$

Since we are interested in comparing experiments with the same network conditions $nc \in NC$, we define the set of experiments $S_{i,nc}$ for HTTP version $i \in \{1,2,3\}$. Finally, we compare successive HTTP versions by measuring how the median Speedup changes. Thus, given a network condition nc and an HTTP version i, the median Speedup $MS_{i,nc}$ is simply:

$$MS_{i,nc} = Median_{j \in S_{i,nc}, k \in S_{i-1,nc}}(\text{Speedup}(j,k)) \quad i \in \{2,3\} \quad (3)$$

Table 3: Summary of the takeaways from *BrowserTime-Video* .

	Resolution	**Speedup**	**Downscale**
Native	No difference	No difference	No difference
Loss	No difference	minor difference	None for h3 Rare for h1 and h2
Bandwidth	h3 worse	h3 little slower h2 h2 faster h1	Many for h3 Few for h2 and h1
Latency	No difference	h2 faster h1 h3 slower h2	No difference

6.2. Results

We summarize our key findings on video streaming results in Table 3. Starting from the case without network impairments (i.e., the *Native* case in Table 3), we note that regardless of the HTTP version used, the chunks are always delivered at the highest resolution ($4k$) and without downscaling. Also, when looking at the Average Speedup (see also Figure 12), we cannot see any radical differences between the HTTP versions. In sum, when the network has good quality, all HTTP versions perform similarly.

When we introduce controlled *Packet Loss*, some initial differences start to appear. Specifically, for HTTP/1.1 and 2, we observe some rare video adjustments (downscale) where the resolution of the chunks jumps between $1920 \times 1080p$ and $4k$. In contrast, HTTP/3 proves to be more stable, with all chunks delivered at $4k$ resolution. Considering other metrics (video resolution and Speedup), we see nearly the same figures across all HTTP versions, e.g., with video chunks delivered at either $4k$ or $1920 \times 1080p$.

More interesting differences emerge when we consider bandwidth limitations. We find that HTTP/3 results in lower resolution video and more frequent video adjustments. We show this effect in Figure 13, which shows the distribution of the number of downloaded chunks in relation of their corresponding bitrate.[15] Recall that we varied the bandwidth between 1 and 5 Mbit/s and, as expected, the video quality does not exceed $1024 \times 576p$. HTTP/1.1 and HTTP/2 show the same performance, while for HTTP/3 we find a higher percentage of chunks delivered at the lower resolutions.

This result is also detailed in Table 4, which gives the average number of resolution downscales for each scenario. At only 1 Mbit/s, all HTTP versions struggle and result in a high (and similar) number of downscales (first line in the table). At 5 Mbit/s, the situation improves and the video settles at $1024 \times 576p$ (third line in the table). Interestingly, HTTP/1.1 and HTTP/2 show stable performance with almost no adaptation in the 2 Mbit/s case, while HTTP/3 averages 13.3 downscales per experiment for the 150 chunks (second line in the table).

To explore further this phenomenon, Figure 14 shows an example of the quality level of the chunks for an HTTP/2

[15] We do not use resolution because some of the chunks in our setup are encoded at two bitrates.

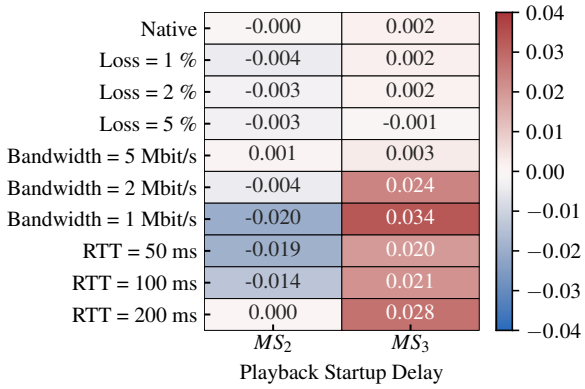


Figure 12: Median Speedup per network condition (*BrowserTime-Video* dataset).

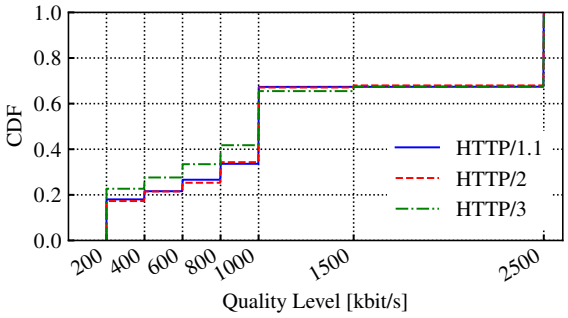


Figure 13: Distribution of chunks resolution with limited bandwidth (*BrowserTime-Video* dataset).

Table 4: Mean number of downscale per experiment.

	HTTP1/1	**HTTP/2**	**HTTP/3**
Bandwidth = 1 Mbit/s	38.7	43.3	38.0
Bandwidth = 2 Mbit/s	1.0	1.0	13.3
Bandwidth = 5 Mbit/s	1.9	2.0	2.0

and an HTTP/3 video session at 2 Mbit/s. On the second chunk, the player downgrades with HTTP/2, but then immediately stabilizes the quality level to 1000 kbit/s ($640 \times 360p$). In contrast, the player with HTTP/3 is more eager for high-resolution chunks and fetches the video several times at the 1500 kbit/s quality level, i.e., $768 \times 432p$. However, the lack of bandwidth results in downgrading to the lowest possible resolution.

While we cannot clearly identify the causes for this behavior, we conjecture that it is caused by complex interactions between several components. Recalling that HTTP/3 runs on top of UDP/QUIC with a different congestion control algorithm, the client player seems more aggressive in this scenario. For example, we observe cases in which the player requests multiple times the same chunk with different resolutions, probably as a reaction to delays in lower network layers.

Looking at the Speedup metric, we see that HTTP/2 improves greatly over HTTP/1.1, while HTTP/3 has the worse performance, especially at very low bandwidth (0.034). We provide further details about this result in Figure 12, which show the median Speedup for HTTP/3 (h3) and HTTP/2 (h2) for different network conditions. The bluer the color, the greater the benefits of the newer HTTP version. Conversely, the red color indicates that the newer HTTP version results in a larger PSD, i.e., slower transmission of the first chunk

Finally, when we impose additional latency (last row in Table 3), almost all chunks are delivered at the highest resolution, and we find that video resolution adjustments are very rare (i.e., no differences between HTTP versions for this metric). Looking at the median in the last three lines of Figure 12, we find that HTTP/3 tends to be slower than HTTP/2, i.e., the first video chunks take more time to be delivered to the client. We again conjecture that this effect is due to the different congestion and flow control algorithms implemented in QUIC, and their interactions with the player. Investigating how performance could be improved by changing the video server configuration (note that we use *nginx HTTP server* with *quiche* HTTP/3 implementation) is left for future work.

7. Discussion and future work

We dissected the performance of HTTP/3 under different network conditions. However, we only performed the measurements with Google Chrome, as we are not aware of any other browser that can be instrumented to use HTTP/3 from the first connection – i.e., without the need to observe the `Alt-Svc` header beforehand. In addition, we always visited websites using a new browser profile with an empty cache and no pre-existing connections. This obviously limits the scope of our study, as we cannot measure how HTTP/3 affects performance on subsequent visits or with a warm HTTP cache, as it will be the case for real users relying on HTTP/3. Also, some websites display a privacy banner that prevents large portions of the web page from loading until the user has given consent to cookies. It will be necessary to test these websites simulating real users behavior, thus accepting the privacy banner.

We have limited ourselves to a subset of the websites that use HTTP/3. In fact, we only included a fraction of websites hosted on the CloudFlare CDN, as its HTTP/3 support has been partially disabled during the time we run our large-scale *BrowserTime-Web* campaign. Since then, CloudFlare has re-enabled HTTP/3 support on most of its websites and new providers start supporting HTTP/3. Therefore, our measurements need to be continuously run (i) to observe how the web ecosystem embraces HTTP/3, (ii) to assess how web pages change to optimize the performance, especially considering domain sharding.

HTTP/3 and QUIC are new standards and may be subject to changes in the near future. Thus, updates on our measurements will be naturally needed. Similarly, we have

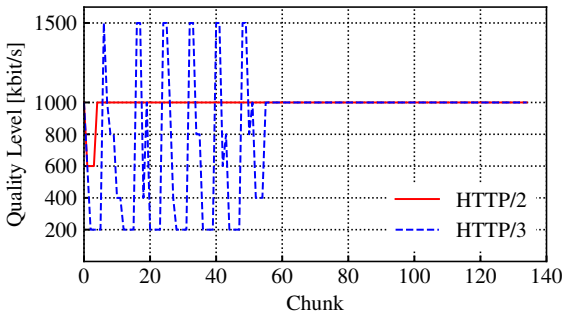


Figure 14: Example sessions with 2 Mbit/s bandwidth (*BrowserTime-Video* dataset).

not investigated how different client and server configurations affect HTTP/3 performance – e.g., the interactions between HTTP/3 and congestion control settings. In addition, while we covered various scenarios using state-of-the-art tools, it is known that network emulation is complex. Similar measurement studies relying on data about actual users are still needed to confirm our results.

8. Conclusions

We presented a study of HTTP/3 adoption and performance, quantifying the performance benefits of HTTP/3 in various network scenarios. We demonstrated that some of the leading Internet companies have started to adopt HTTP/3 in 2020. However, most of the early adopters still host the majority of web objects on third-party HTTP/2 servers. With a large-scale measurement campaign, we examined the performance of HTTP/3 under various network conditions, targeting thousands of websites. We found that HTTP/3 brings performance benefits in high latency scenarios. When packet loss is high or bandwidth is low, the performance of HTTP/3 and HTTP/2 is about the same. Mobile users accessing websites over mobile networks with smartphones and tablets can also expect benefits. We found large differences in performance depending on the infrastructure where the website is hosted. In general, websites that load objects from a limited number of third-party domains benefit from HTTP/3 the most. Differently, we did not observed benefits for adaptive video streaming setups in controlled environments.

Acknowledgements

This work has been supported by Eutelsat Communications S.A., by Telecom Italia S.p.A and by the SmartData@PoliTO center on Big Data and Data Science.

References

[1] Belshe, M., Peon, R., and Thomson, M. (2015). Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540.

[2] Bishop, M. (2021). Hypertext Transfer Protocol Version 3 (HTTP/3). Internet-Draft draft-ietf-quic-http-34, Internet Engineering Task Force. Work in Progress.

[3] Carlucci, G., De Cicco, L., and Mascolo, S. (2015). Http over udp: An experimental investigation of quic. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, page 609–614, New York, NY, USA. Association for Computing Machinery.

[4] da Hora, D. N., Asrese, A. S., Christophides, V., Teixeira, R., and Rossi, D. (2018). Narrowing the gap between qos metrics and web qoe using above-the-fold metrics. In *International Conference on Passive and Active Network Measurement*, pages 31–43. Springer.

[5] Fielding, R. T., Nielsen, H., Mogul, J., Gettys, J., and Berners-Lee, T. (1997). Hypertext Transfer Protocol – HTTP/1.1. RFC 2068.

[6] Guarnieri, T., Drago, I., Vieira, A. B., Cunha, I., and Almeida, J. (2017). Characterizing qoe in large-scale live streaming. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–7.

[7] Guillen, L., Izumi, S., Abe, T., and Suganuma, T. (2019). Sand/3: Sdn-assisted novel qoe control method for dynamic adaptive streaming over http/3. *Electronics*, **8**(8), 864.

[8] Iyengar, J. and Thomson, M. (2021). Quic: A udp-based multiplexed and secure transport. RFC 9000, RFC Editor.

[9] Kakhki, A. M., Jero, S., Choffnes, D., Nita-Rotaru, C., and Mislove, A. (2017). Taking a long look at quic: an approach for rigorous evaluation of rapidly evolving transport protocols. In *Proceedings of the 2017 Internet Measurement Conference*, pages 290–303.

[10] Kosek, M., Shreedhar, T., and Bajpai, V. (2021). Beyond quic v1–a first look at recent transport layer ietf standardization efforts. *arXiv preprint arXiv:2102.07527*.

[11] Lovell, D., Pollard, B., Marx, R., and Hantsis, S. (2021). Part IV Chapter 24, HTTP).

[12] Manzoor, J., Cerdà-Alabern, L., Sadre, R., and Drago, I. (2020). On the performance of quic over wireless mesh networks. *Journal of Network and Systems Management*, **28**(4), 1872–1901.

[13] Marx, R., Herbots, J., Lamotte, W., and Quax, P. (2020). Same standards, different decisions: A study of quic and http/3 implementation diversity. In *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, pages 14–20.

[14] Rajiullah, M., Lutu, A., Khatouni, A. S., Fida, M.-R., Mellia, M., Brunstrom, A., Alay, O., Alfredsson, S., and Mancuso, V. (2019). Web experience in mobile networks: Lessons from two million page visits. In *The World Wide Web Conference*, pages 1532–1543.

[15] Rescorla, E. (2018). The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446.

[16] Saif, D. and Matrawy, A. (2021). A pure http/3 alternative to mqtt-over-quic in resource-constrained iot. *arXiv preprint arXiv:2106.12684*.

[17] Saif, D., Lung, C.-H., and Matrawy, A. (2020). An early benchmark of quality of experience between http/2 and http/3 using lighthouse. *arXiv preprint arXiv:2004.01978*.

[18] Seufert, M., Egger, S., Slanina, M., Zinner, T., Hoßfeld, T., and Tran-Gia, P. (2014). A survey on quality of experience of http adaptive streaming. *IEEE Communications Surveys & Tutorials*, **17**(1), 469–492.

[19] Tellakula, S. (2020). Comparing HTTP/3 vs. HTTP/2 Performance. `https://blog.cloudflare.com/http-3-vs-http-2/`.

[20] Trevisan, M., Khatouni, A. S., and Giordano, D. (2020). Errant: Realistic emulation of radio access networks. *Computer Networks*, **176**, 107289.

[21] Trevisan, M., Giordano, D., Drago, I., and Khatouni, A. S. (2021). Measuring http/3: Adoption and performance. In *2021 19th Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 1–8.

[22] Varvello, M., Schomp, K., Naylor, D., Blackburn, J., Finamore, A., and Papagiannaki, K. (2016). Is the web http/2 yet? In *International Conference on Passive and Active Network Measurement*, pages 218–232. Springer.

[23] Wolsing, K., Rüth, J., Wehrle, K., and Hohlfeld, O. (2019). A performance perspective on web optimized protocol stacks: Tcp+ tls+ http/2 vs. quic. In *Proceedings of the Applied Networking Research Workshop*, pages 1–7.