HTTP/2 Performance in Cellular Networks

Utkarsh Goel†, Moritz Steiner, Mike P. Wittie†,
Martin Flack, and Stephen Ludin

†Montana State University
{utkarsh.goel, mwittie}@cs.montana.edu

Akamai Technologies, Inc.
{moritz, mflack, sludin}@akamai.com

1. INTRODUCTION

The need for faster Web communication motivates Content Delivery Networks (CDNs) to improve the performance of their Web content delivery to end users. The widely-adopted HyperText Transfer Protocol (HTTP/1.1) reduces request-response times by pipelining requests at the server side. However, with HTTP/1.1 (h1), resources are loaded serially over the TCP connection, which creates a head-of-line (HOL) blocking. To mitigate HOL blocking and to enable downloading of resources in parallel for faster page loads, Web browsers open up to six TCP connections for each host name on the page.

The HTTP/2 (h2) protocol, standardized in 2015, follows a different approach to eliminate HOL blocking to achieve faster page load times (PLTs) [5]. h2 uses only one TCP connection to exchange all request and response payloads through multiplexing and interleaving. Previous studies have shown both improvement and degradation in PLT when using h2 with respect to h1 [6, 7, 9, 10]. The disagreement about h2 performance from these studies motivates further investigation as to whether and under what conditions h2 brings the performance benefits that were originally envisioned [5].

In this poster, we take a novel approach to investigate h2 performance by first understanding the dynamic nature of cellular network characteristics (in terms of loss, latency, and bandwidth) and then exposing its implications on the PLTs when using h2. Our goal is to understand how h2 impacts PLT when cellular network delay is interpreted as loss by server-side TCP sockets. Therefore, for the remainder of this paper we focus only on connections with loss, as interpreted by TCP. Specifically, we make the following three contributions in this paper.

Dataset: Our preliminary analysis of real world cellular network characteristics is based on 6200 TCP connections captured over several hours on an Akamai CDN cluster hosted inside a datacenter of a major cellular network provider in the US. We observe that about 2000 connections (32%) experience loss, of which about 500 connections experience loss more than once within the first few seconds. The median connection duration, total number of TCP segments, and bytes exchanged between clients and servers during our capture is about 2.3 seconds, 26 segments, and 15 KB respectively.

Measurement: Unlike existing network emulators that introduce packet loss on a link at random times during emulation, *we model the emulation on real-world TCP traces to adequately represent the correlation of losses in cellular networks, as interpreted by TCP*. Our simulation is designed carefully to emulate cellular networks in terms of packet loss interpreted by the server (in the form of retransmissions), time between loss events, round trip time (RTT), and bandwidth. From such a comprehensive view of cellular network characteristics, we argue that our technique improves existing network emulation techniques.

Insights: Preliminary results from our investigation of h2 performance are threefold. First, for a webpage with several hundreds of small sized objects, h2 outperforms h1, except in the scenarios when cellular networks frequently experience high loss rates. Second, for a webpage with large objects, h2 does not reduce PLTs at all and in fact PLTs are significantly higher than h1. And third, for a webpage with a relatively large number of both small and medium sized objects, h2 outperforms h1; however as the connection starts experiencing loss, the performance gain in h2 degrades, resulting in PLTs slower than h1.

2. EXPERIMENTAL SETUP

To accurately understand the characteristics of cellular networks, we selected an Akamai CDN cluster located inside the network of a major cellular ISP in the US [2]. On each server in the cluster, we ran TCPDump for several hours at different times of a day to capture incoming and outgoing TCP segments. Given that the cellular network in some cases splits TCP connections between clients and servers for HTTP sessions, but never does so for HTTPS sessions [8], we only capture TCP segments for HTTPS traffic, ensuring that the TCP segments we capture are for end-to-end connections between clients and CDN servers. Our TCP traces reflect the characteristics of a real world cellular network as the TCP connections to the selected Akamai CDN cluster are not influenced by any interference from the public Internet [2].

Next, from each packet capture file, we use tshark to extract the number of frames and bytes exchanged between the client and the server, the number of frames retransmitted by the server, and the time interval between acknowledgments [4]. We extract the above metrics for each connection at 70 ms intervals, where the first interval starts when the *TCP SYN* is received by the server. The choice of 70 ms as the time interval to generate snapshots of cellular network characteristics matches the median RTT between clients in the selected cellular network and Akamai CDN servers [8]. Therefore, in the median case, we expect the above collected metrics to change only after 70 ms.

From the collected packet captures, we make four observations. **First**, about 32% of the TCP connections experience loss (in the form of retransmissions from the server), which is much higher compared to what we observe in wired last-mile

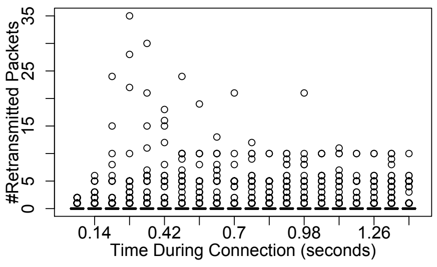


Figure 1: Clustered Loss.

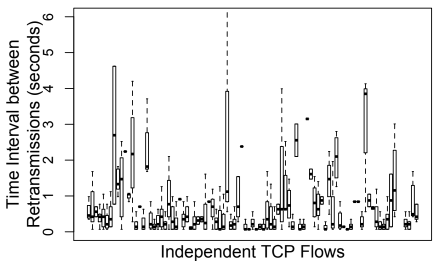


Figure 2: Multiple loss events.

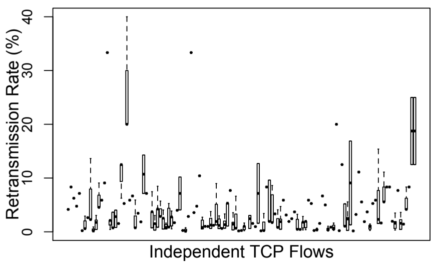
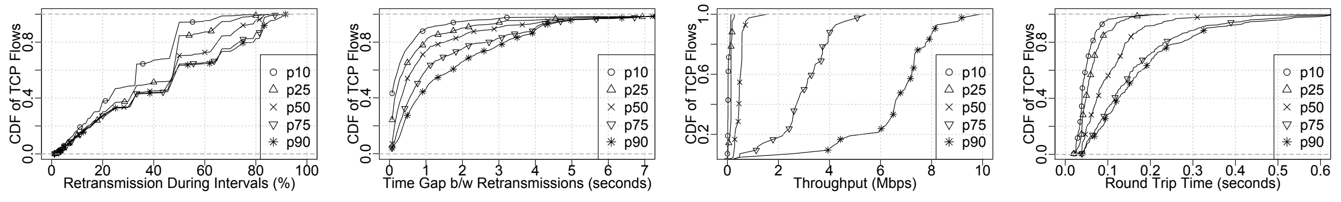


Figure 3: Retransmission Rate (%).



(a) Loss (b) Time b/w Retransmission (c) Throughput (d) RTT

Figure 4: Distributions of 10th, 25th, 50th, 75th, and 90th percentile observed across all TCP connections.

networks. Although retransmissions could happen because of socket timeouts caused by temporary congestion in the network, as opposed to only when packets are lost, TCP congestion control interprets both types of events as loss.

Second, losses in cellular networks are often clustered, such that when a loss event occurs, many consecutive TCP segments are retransmitted by server. In Figure 1, we show a boxplot of the distribution of the number of packets retransmitted by the server within continuous 70 ms intervals across all TCP connections. Each outlier circle in the figure represents one or more TCP connections. The x-axis represents the timestamp when a 70 ms time interval finishes. The y-axis represents the number of packets retransmitted by server during a time interval. Since the majority of TCP connections do not experience retransmission within every 70 ms interval, the 25th, 50th, and 75th percentile values of the distributions are all zero. From the figure we observe that many connections experience multiple retransmissions at different times during the connection. For example, for the time interval finishing at 420 ms, we observe that for multiple TCP connections servers retransmit 5, 10, and even 20 packets. We observed such clustered retransmissions throughout the life of TCP connections, though for figure clarity we only show clustered loss for only the first 1.5 seconds.

Third, TCP connections experience retransmissions at multiple times during their lifetime. In Figure 2, for connections experiencing retransmissions, we show the distribution of time gaps between retransmission clusters. Since we record retransmission after every 70 ms, each time gap between retransmission is at least 70 ms. From the figure we observe that for several connections, subsequent retransmission clusters appear within one second.

And **fourth**, when a retransmission cluster appears, the majority of connections experience about 5-15% packet loss during 70 ms time intervals. We support the above claim via Figure 3, where for each TCP connection we show the distribution of retransmitted TCP segments (in percent) at the occurrence of every retransmission cluster.

2.1 Emulating Cellular Networks

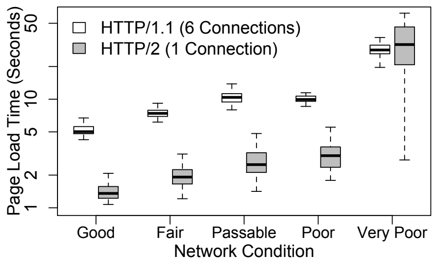
Based on our observations, we design our emulation to closely represent the network conditions of a real world cellular network. Specifically, we introduce clustered loss only at specific times during the connection, in addition to modifying bandwidth, and RTT every 70 ms. In Figure 4, we show distributions of 10th, 25th, 50th, 75th, and 90th percentile retransmission rate, time gap between retransmission clusters, throughput, and RTT, as observed across all TCP connections respectively. Using these distributions, in Table 1, we develop simulation scenarios to emulate various mobile network conditions. For example, when emulating a *Good* experience, we select the 10th percentile (p10) loss distribution from Figure 4(a), 10th percentile RTT (p10) distribution from Figure 4(d), 90th percentile (p90) throughput (used as bandwidth in our emulation) distribution from Figure 4(c), and 90th percentile (p90) distribution of time gap between loss from Figure 4(b). Note again, that we only emulate connections that experience loss.

We design our emulation to dynamically update network links between client and server in terms of the above four distributions, as shown in Table 1. Specifically, at every 70 ms we update RTT by randomly selecting a value from its respective distribution. Since bandwidth in cellular networks is attributed to base stations and is therefore not dependent on loss and RTT [11], we also change bandwidth every 70 ms. Finally, we introduce packet loss only when the selected time gap value has passed during emulation.

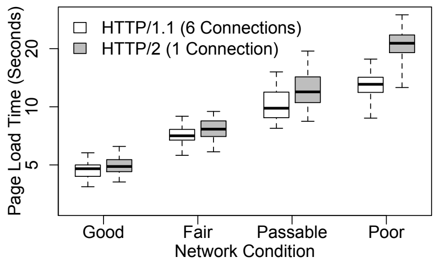
Next, we setup a network topology using three machines to emulate a client, a Web server, and a bridge through which client and server connect. On the client, we run Chromium Telemetry to load different webpages over 100 times using Google Chrome browser [1]. On the server, we configure Apache to support `h1` and `h2` on different virtual hosts. On the bridge, we emulate different network experiences using `TC NETEM` between the client and the server. Depending on the emulated scenario, we configure the bridge to modify the network loss, time of next loss, RTT, and bandwidth after every 70 ms according to the scenarios defined in Table 1.

3. PRELIMINARY RESULTS

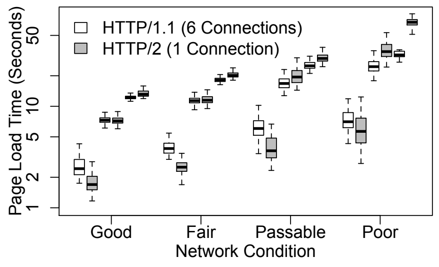
In Figure 5(a), we compare PLTs for a webpage with 365 objects of average size of 2 KB, loaded in turn over `h1`



(a) 365 objects of 2 KB each



(b) 10 objects of 435 KB each



(c) 136 objects of 1 KB to 620 KB each

Figure 5: Distribution of page load times when loaded over `h1` and `h2` under various network conditions.

Table 1: Emulation of various network conditions.

Condition	Loss	Time Gap	Throughput	RTT
Good	p10	p90	p90	p10
Fair	p25	p75	p75	p25
Passable	p50	p50	p50	p50
Poor	p75	p25	p25	p75
Very Poor	p90	p10	p10	p90

and `h2` connections across the different emulated network conditions. From the figure we observe that PLTs over `h2` are lower than PLTs over `h1`, because in the case of `h1` with 6 TCP connections, the server can only send 6 objects in parallel (due to HOL blocking), whereas in the case of `h2` with many streams multiplexed onto one connection, the server sends a large number of objects in parallel. Further, as the network condition becomes *Very Poor*, the PLTs increase for both `h1` and `h2`, but more so for `h2`. For `h1`, the impact of packet loss on any one of the 6 connections only affects the download of the object on that particular connection. Whereas, in the case of `h2`, since all object downloads are multiplexed over a single TCP connection, every packet loss affects all ongoing object downloads.

Next, we investigate how `h2` performs with multiplexed responses when each object to be transferred over the same TCP connection has a large payload. In Figure 5(b), we compare PLTs for a webpage with 10 large objects of size 435 KB each. In general, we observe that `h1` outperforms `h2` across all network conditions, especially in *Poor* conditions where loss occurs frequently. We argue that in the case of `h2` with one connection, the initial congestion window (ICW) size of the server during the TCP slow start is only one-sixth of the cumulative window sizes of `h1` with 6 parallel TCP connections. As a result, the server sends six times lesser data over `h2` during the TCP slow start phase. We confirm the claim as when we compare PLTs without any loss, we observe `h1` to still outperform `h2`. Further, as the network conditions get worse, in terms of loss rate and frequency of loss events, the congestion window of the single `h2` connection does not grow. Whereas in the case of six `h1` connections, loss only impacts one or more connections and thus the cumulative congestion window size remains larger than the window size of single `h2` connection.

Finally, in Figure 5(c), we compare PLTs for a webpage (modeled from HTTP Archive data [3]) with 136 objects whose size ranges from 1 KB to 620 KB. In addition to measuring PLTs over `h1` and `h2` in different network conditions, we investigate the impact of webpage size on PLT, while keeping the number of objects constant. The first, second, and third pairs of boxplots in Figure 5(c) under each network condition represent the distribution of PLTs for webpages of size 2 MB, 8 MB, and 12 MB, respectively. From the figure we observe that for a 2 MB page, PLTs over `h2` are lower than PLTs over `h1`, as (similarly to Figure 5(a)) server sends multiple small sized objects in parallel during the TCP slow start, whereas server sends only six objects in parallel in the case of `h1`.

Although for the 8 MB page, PLTs over `h2` and `h1` are comparable under *Good* and *Fair* network conditions, however, as the conditions get worse, PLTs over `h2` become larger than `h1`. Moreover, for the 12 MB page, the PLTs over `h2` are always higher than `h1`. Similarly to Figure 5(b), under lossy conditions the congestion window on the server does not grow as much and as fast as it grows cumulatively for six `h1` connections, thus affecting the PLTs over `h2` when downloading large objects. We observe that the slow start phase is less important here as most of the PLT comes from the congestion avoidance phase. To confirm whether the ICW impacts PLTs, we loaded webpages 25 MB in size (not shown in the paper) and observed no statistical significance in the difference between `h2` and `h1` PLTs.

4. FUTURE WORK

To best of our knowledge, there is currently no known best practice as to how TCP and `h2` should be tuned such that the impact of loss on single `h2` connection is minimized. We propose to side-step from the standard of using single `h2` connection for the complete webpage and investigate the mobile Web performance with multiple `h2` connections, especially when cellular networks experience loss.

5. REFERENCES

[1] Chromium Telemetry. `https://goo.gl/gFxs7x`, Dec. 2014.
[2] Akamai Accelerated Network Partner. `https://goo.gl/ZhkNH1`, Jun. 2016.
[3] HTTP Archive: Interesting stats. `http://httparchive.org/interesting.php`, Jun. 2016.
[4] Tshark. `https://goo.gl/YmGrjE`, Jun. 2016.
[5] M. Belshe, R. Peon, and E. M. Thomson. Hypertext Transfer Protocol Version 2 (HTTP/2), RFC 7540, May 2015.
[6] H. de Saxc, I. Oprescu, and Y. Chen. Is HTTP/2 really faster than HTTP/1.1? In *IEEE INFOCOM*, Apr. 2015.
[7] J. Erman, V. Gopalakrishnan, R. Jana, and K. K. Ramakrishnan. Towards a SPDY'Ier Mobile Web? In *ACM CoNext*, Dec. 2013.
[8] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. Detecting Cellular Middle-boxes using Passive Measurement Techniques. In *Passive and Active Measurements Conference (PAM)*, Mar. 2016.
[9] M. Varvello, K. Schomp, D. Naylor, J. Blackburn, Alessandro, Finamore, and K. Papagiannaki. Is The Web HTTP/2 Yet? In *Passive and Active Measurements Conference (PAM)*, Mar. 2016.
[10] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall. How Speedy is SPDY? In *USENIX NSDI*, Apr. 2014.
[11] K. Winstein, A. Sivaraman, and H. Balakrishnan. Stochastic Forecasts Achieve High Throughput and Low Delay over Cellular Networks. In *USENIX NSDI*, Apr. 2013.