Are 3rd Parties Slowing Down the Mobile Web?

Utkarsh Goel
Montana State University
utkarsh.goel@montana.edu

Moritz Steiner
Akamai Technologies, Inc.
moritz@akamai.com

Wontaek Na
Akamai Technologies, Inc.
wna@akamai.com

Mike P. Wittie
Montana State University
mwittie@cs.montana.edu

Martin Flack
Akamai Technologies, Inc.
mflack@akamai.com

Stephen Ludin
Akamai Technologies, Inc.
sludin@akamai.com

1. INTRODUCTION

Content Providers (CPs) such as Facebook, Google, and others desire that their websites attract large user bases and generate high revenue. As a result, CPs strive to develop attractive and interactive websites that keep users engaged. JavaScript libraries from Online Social Networks, advertisements, and user tracking beacons allow CPs to personalize webpages based on end-users' interests, while various CSS frameworks make websites aesthetically pleasing [7, 9]. Further, webpage analytic APIs and performance monitoring tools allow CPs to monitor the user-perceived performance with their websites [8, 11]. However, as CPs continue to evolve their websites with more and more attractive features the webpage load time (PLT) starts to increase, which results in poor user experience with their websites [5, 12, 27].

To speed up Web content delivery to end-users CPs make contracts with Content Delivery Networks (CDNs), such as Akamai. These CDNs are distributed deep inside many last mile wired and mobile ISPs worldwide and thus provide low-latency paths to end-users [24]. Additionally, CDNs are motivated to adopt new and upcoming Internet standards to achieve even faster content delivery for CPs' websites [15, 17, 19, 21]. Although CDNs are effective in reducing download times of Web objects they serve, as CPs continue to enhance their websites by adding *external* APIs and tools it becomes challenging for CDNs to further speed up webpages [5, 12, 27].

We refer to external APIs and tools as *3rd Party* assets because the performance of such assets is not under the control of the *1st Party* (such as a CDN provider acting as surrogate infrastructure for its CP customers) that serves the base page HTML. In general, we define *3rd Party* as any asset embedded in the webpage that is not served by the same infrastructure serving the base page HTML and thus the downloads of such assets cannot be optimized by *1st Party*.

In this paper, we take a novel approach to expose the impact of *3rd Party* downloads on mobile Web performance. We showcase results from our ongoing investigation to understand whether *3rd Party* downloads contribute to critical paths of webpages [26]. Specifically, we make the following three contributions:

S3'16, October 03-07, 2016, New York City, NY, USA

DOI: http://dx.doi.org/10.1145/2987354.2987359

Data Analysis: We make extensive use of the open-sourced data available at HTTP Archive to expose the characteristics of *3rd Party* assets embedded into the top 16,000 Alexa webpages, which we observe to be currently served by four major CDN providers [2, 6]. Specifically, we investigate the number of unique domain names resolved, HTTP requests sent, total bytes, and total uncompressed bytes downloaded for *3rd Party* assets for each webpage in our dataset.

Measurement: To understand the impact of *3rd Party* downloads on PLT, we conduct several active experiments over a period of three months in 2016 using 52 cellular clients of Gomez Mobile testbed [3]. We load 60 real webpages about 400 times each from the production servers of a major CDN provider, where each webpage comprises of several *3rd Party* assets. For the purposes of this investigation, we devise a new Web performance metric, **3rd Party Trailing Ratio**, that represents the time spent in downloading *3rd Party* assets, when no *1st Party* asset is downloaded in parallel.

Inferences Drawn: Based on our preliminary analysis of *3rd Party* download times, we observe that for many webpages *3rd Party* downloads contribute to up to 50% of the page load time, in the median case. Additionally, we identify that several *3rd Party* vendors do not compress Web objects even when clients show support for compression in HTTP headers.

To the best of our knowledge, there is currently no known best-practice as to how *1st Parties* should optimize *3rd Party* downloads to mitigate the impact of *3rd Party* downloads on the overall PLT. Therefore, in this paper we motivate the need to explore research directions for *1st Parties* to enable them reduce PLT currently inflated by various *3rd Party* downloads.

2. DATA COLLECTION

We now discuss our datasets to investigate the characteristics of *3rd Party* assets embedded in different webpages. We then measure the impact that *3rd Party* assets make on PLT, by conducting active experiments in real-world cellular networks.

Understanding webpage structures: We use the open-sourced HTTP Archive dataset, an initiative by Google, Mozilla, and other industry leaders, to understand structures of different websites [2]. The HTTP Archive data is collected using the WebPageTest framework, where webpages are loaded over several virtual machines inside a datacenter [13]. The page loads are then translated into a format similar to HTTP Archive format (HAR) containing the timing data as well as HTTP request and response headers for each individual HTTP request issued for the webpage under test.

For the purposes of this work, we extract only the HTTP request and response headers pertaining to the top 16,000 Alexa webpages,

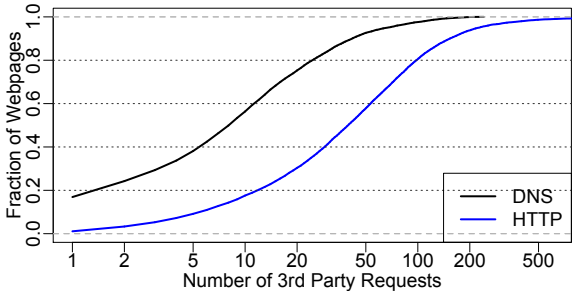

Figure 1: Distribution of the number of DNS lookup and HTTP requests made to download *3rd Party* assets.

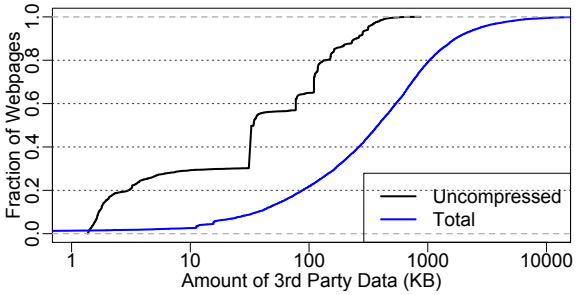

Figure 2: Distribution of total bytes and uncompressed bytes downloaded from *3rd Party* vendors.

which we observe are currently served to Android mobile devices by four major CDN providers. In particular, for each requested object we extract headers indicating the response size, hostname associated with the download, and whether the response was compressed by the *3rd Party* server. Since many *3rd Party* assets often load after the *onLoad event* triggered by the Web browser and since we only focus on understanding how much *3rd Party* downloads impact the PLT, we consider the measurement data for objects loaded only until the *onLoad event*. Our total dataset consists of about 16,000 webpages requesting a total of 1.6 M objects, out of which about 525 K objects belong to *3rd Party* vendors.

Next, for each hostname we perform a `dig` operation to check whether the hostname resolves to a canonical name (CNAME) associated with any of the four CDN providers used in this study. If a hostname for an object does not resolve to a CNAME associated to the *1st Party* serving the base page HTML, we consider that object as a *3rd Party* asset, with respect to the *1st Party* under analysis. Additionally, if the hostname does not resolve to any CNAME, we consider that hostname as *3rd Party* for all four *1st Party* CDN providers we consider in this study.

Finally, for each webpage we calculate the total number of domain names resolved and HTTP requests sent for objects that we label as *3rd Party* assets. We also calculate the total number of bytes, as well as the total number of uncompressed bytes delivered by various *3rd Party* vendors even when mobile clients indicate support for compression in HTTP requests.

Measurement data for *3rd Party* impact: To collect Web performance data pertaining to mobile page loads, we conduct several active experiments on Gomez Mobile testbed to download 60 different webpages served by the production servers of a major CDN provider [1, 3]. Next, we configure Gomez Mobile clients to load each website about 400 times and record Navigation and Resource Timing data for each page load [4, 10]. The Navigation and Resource Timing data we obtain consists of timestamps when the page load starts, timestamps when each object starts and finishes loading (including the time to perform DNS lookup, TCP handshake time, SSL handshake time, time to receive the first bit, and the object download time), and the timestamp when *onLoad* event is triggered by the Web browser. Our configured Gomez clients also record the hostname associated to each requested object, which we use to identify whether or not the object is downloaded from a *3rd Party* vendor, similarly to how we identify this using the HTTP Archive data.

3. RESULTS

In Figure 1, we show the distribution of the number of unique domain names resolved and total number of HTTP requests issued to download *3rd Party* assets for webpages served by the four major CDN providers used in this study. In general, we observe that

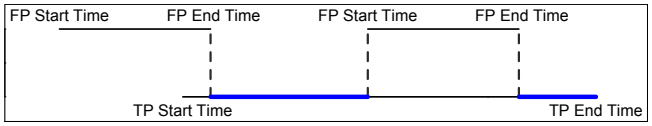

Figure 3: An example waterfall diagram showing one *3rd Party* and two *1st Party* downloads during page load.

in the median case, webpages resolve about 10 unique *3rd Party* domain names and issue about 50 HTTP requests to download *3rd Party* assets. For mobile clients where radio latency and the latency to cellular DNS servers is a few hundred milliseconds, we speculate that resolving multiple *3rd Party* domain names could introduce significant latency to the overall PLT [20, 19, 25]. Additionally, such a large number of domain name lookups could potentially result in establishing many new TCP connections, negatively affecting the object downloads as each connection to *3rd Party* server has to go through a TCP slow start phase.

Additionally, in Figure 2, we show the distribution of the total amount of data downloaded from *3rd Party* servers, as well the total number of uncompressed bytes transmitted by *3rd Party* servers even when clients indicate support for compression in the HTTP requests. Finally, similarly to Agababov *et al.*, we observe that several *3rd Party* vendors do not compress objects even when clients indicate support for compression in HTTP request headers [14].

Although *3rd Party* assets embedded on a webpage may require multiple domain name lookups and download of hundreds of kilobytes of data, we argue that *3rd Party* assets that do not lie on a webpage's critical path will not impact the PLT. Therefore, we now focus on quantifying the time spent by *3rd Party* downloads on the critical path of the webpage. Specifically, we devise a new Web performance metric, **3rd Party Trailing Ratio** (**TPTR**), to reflect time spent by *3rd Party* downloads during which no *1st Party* asset is being downloaded. For example, consider the example waterfall chart in Figure 3, where we show the start and end timestamps of one *3rd Party* and two *1st Party* object downloads during a webpage load [22]. We define **TPTR** as the total time spent by *3rd Party* downloads during which no *1st Party* download is overcasting the *3rd Party* download in the waterfall chart, as denoted by the two thick solid blue lines.

To calculate **TPTR**, we employ a two step process as follows: First, using start and end timestamps of all object downloads, we calculate all non-overlapping time intervals of *1st Party* and *3rd Party* downloads [18]. Second, using above intervals, for each *3rd Party* interval we identify whether there is any *1st Party* interval that intersects the *3rd Party* interval. We then calculate the *3rd Party* time interval that does not overlap with any *1st Party* interval. The sum of all such intervals results to the **TPTR**.

In Figure 4, we show the distribution of **TPTR** for 60 webpages served by a major CDN provider, each loaded 400 times from Gomez Mobile clients connected to cellular networks. For figure

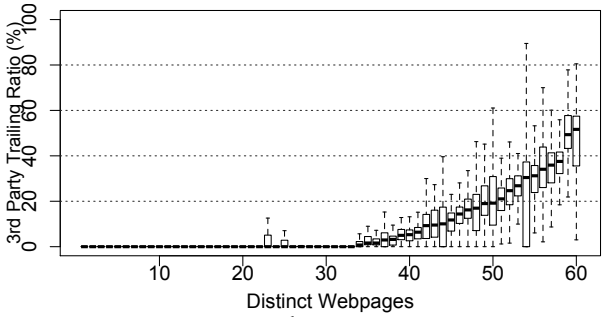

Figure 4: Distributions of 3rd Party Trailing Ratio across 60 webpages served to cellular clients.

clarity, we sort pages on x-axis based on the median **TPTR** value across different webpages. In general, we observe that *3rd Party* downloads do not impact PLT for about half of the webpages in our dataset. With these webpages, when *3rd Party* assets are being downloaded, one or more longer *1st Party* assets are also being downloaded in parallel. Therefore, for these webpages, the *3rd Party* downloads do not lie on critical path. However, for other webpages, *3rd Party* downloads contribute to up to 50% of the total PLT, in the median case. For these webpages, when *3rd Party* assets are being downloaded one or more *1st Party* downloads overlap at most a small fraction of *3rd Party* download time. Therefore, for these webpages, *3rd Party* downloads lie on the webpage critical path and thus introduce additional latency to the overall PLT.

4. FUTURE RESEARCH DIRECTIONS

Our study exposes the impact of *3rd Party* object downloads on mobile Web performance. We argue that the research community should invest effort to develop techniques that mitigate such demonstrated *3rd Party* impact. In our ongoing work, we plan to investigate if and how *1st Party* could safely redirect *3rd Party* downloads onto their infrastructures. One option to achieve this would be to rewrite *3rd Party* URLs to URLs associated to *1st Party* hostnames, such that any request to download a *3rd Party* asset could be delivered by *1st Party* servers, to which the clients are already connected.

Additionally, such URL rewriting would allow *1st Party* to use well-thought content delivery protocols, such as HTTP/2, IPv6, caching and pre-fetching static resources to deliver *3rd Party* assets. Alternatively, one could potentially coalesce TCP connections to *3rd Party* content serves onto the existing connection to the *1st Party* server [23]. In fact, a recent Internet draft by Microsoft and Mozilla details how to present additional certificates during an existing connection and serve content for the domains referenced in the additional certificates [16].

ACKNOWLEDGMENTS

We thank Eric Geyer (Akamai), Ilya Grigorik (Google), Shantharaju Jayanna (Yahoo!), Andrew Kahn (Akamai), Patrick Meenan (Google), Ajay Kumar Miyyapuram (Cerner), and Kanika Shah (AppDynamics) for providing us constructive feedback. We also thank National Science Foundation (NSF) for supporting this work through grants NSF CNS-1555591 and NSF CNS-1527097.

5. REFERENCES

[1] Gomez. http://www.sqaforums.com/attachments/601980-SQA_Gomez_DollarThrifty_Webinar_QandA.PDF, Nov. 2009.

[2] HTTP Archive: Interesting stats. http://httparchive.org/interesting.php, 2010.

[3] Gomez (Dynatrace Synthetic Monitoring). https://www.ndm.net/apm/Compuware/gomez, Jul. 2015.

[4] Navigation Timing. http://w3c.github.io/navigation-timing/, Aug. 2015.

[5] The Truth Behind the Effect of Third Party Tags on Web Performance. http://blog.catchpoint.com/2015/03/12/truth-behind-effect-third-party-tags-web-performance/, Dec. 2015.

[6] Alexa Top Sites. http://www.alexa.com/topsites, Jul. 2016.

[7] Facebook for Developers. https://developers.facebook.com/, Jun. 2016.

[8] Google Analytics Soulutions. https://analytics.googleblog.com/, Jun. 2016.

[9] Google Fonts. https://fonts.google.com/, Jun. 2016.

[10] Resource Timing. https://www.w3.org/TR/resource-timing/, Jul. 2016.

[11] The next generation of Application Intelligence has arrived. https://www.appdynamics.com/, Jun. 2016.

[12] Third-party content could be slowing Britain's retail websites. https://www.nccgroup.trust/uk/about-us/newsroom-and-events/press-releases/2016/march/third-party-content-could-be-slowing-britains-retail-websites/, Mar. 2016.

[13] WebPageTest Framework. http://www.webpagetest.org/, Jul. 2016.

[14] V. Agababov, M. Buettner, V. Chudnovsky, M. Cogan, B. Greenstein, S. McDaniel, M. Piatek, C. Scott, M. Welsh, and B. Yin. Flywheel: Google's Data Compression Proxy for the Mobile Web. In *USENIX NSDI*, May 2015.

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

[16] M. Bishop and M. Thomson. Secondary Certificate Authentication in HTTP/2. http://www.ietf.org/internet-drafts/draft-bishop-httpbis-http2-additional-certs-01.txt, May 2016.

[17] F. Chen, R. K. Sitaraman, and M. Torres. End-User Mapping: Next Generation Request Routing for Content Delivery. In *ACM SIGCOMM*, Aug. 2015.

[18] R. C. Enaganti. Merge Overlapping Intervals. http://www.geeksforgeeks.org/merging-intervals/, Aug. 2015.

[19] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. A Case for Faster Mobile Web in Cellular IPv6 Networks. In *ACM MobiCom (Poster Session)*, Oct. 2016.

[20] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. Detecting Cellular Middle-boxes using Passive Measurement Techniques. In *ACM Passive and Active Measurements Conference (PAM)*, 2016.

[21] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. HTTP/2 Performance in Cellular Networks. In *ACM MobiCom*, Oct. 2016.

[22] M. Isham. X-Ray Your Website Performance: A BeginnerâĂŹs Guide to Waterfall Charts. https://zoompf.com/blog/2013/10/x-ray-your-website-performance-a-beginners-guide-to-waterfall-charts, Oct. 2013.

[23] S. Makineni, R. Iyer, P. Sarangam, D. Newell, L. Zhao, R. Illikkal, and J. Moses. Receive Side Coalescing for Accelerating TCP/IP Processing. In *High Performance Computing (HiPC)*, Dec. 2006.

[24] E. Nygren, R. K. Sitaraman, and J. Sun. The Akamai Network: A Platform for High-Performance Internet Applications. In *ACM SIGOPS Operating Systems Review, Vol. 44, No.3*, July 2010.

[25] J. P. Rula and F. E. Bustamante. Behind the Curtain: Cellular DNS and Content Replica Selection. In *ACM Internet Measurement Conference (IMC)*, Nov. 2014.

[26] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall. Demystify Page Load Performance with WProf. In *USENIX NSDI*, Apr. 2013.

[27] C. Williams. How one developer just broke Node, Babel and thousands of projects in 11 lines of JavaScript. http://www.theregister.co.uk/2016/03/23/npm_left_pad_chaos/, Mar. 2016.