Measuring What is Not Ours: A Tale of 3^{rd} Party Performance

Utkarsh Goel[1], Moritz Steiner[2], Mike P. Wittie[1],
Martin Flack[2], and Stephen Ludin[2]

[1] Montana State University – {utkarsh.goel, mwittie}@cs.montana.edu
[2] Akamai Technologies, Inc. – {moritz, mflack, sludin}@akamai.com

Abstract. Content Providers make use of, so called *3^rd Party* (*3P*) services, to attract large user bases to their websites, track user activities and interests, or to serve advertisements. In this paper, we perform an extensive investigation on how much such *3Ps* impact the Web performance in mobile and wired last-mile networks. We develop a new Web performance metric, the **3^rd Party Trailing Ratio**, to represent the fraction of the critical path of the webpage load process that comprises of only *3P* downloads. Our results show that *3Ps* inflate the webpage load time (PLT) by as much as 50% in the extreme case. Using URL rewriting to redirect the downloads of *3P* assets on *1^st Party* infrastructure, we demonstrate speedups in PLTs by as much as 25%.

1 Introduction

Content Providers (CPs) such as Facebook, Google, and others seek to attract large number of users to their websites and to generate high revenue. As a result, CPs strive to develop attractive and interactive websites that keep their 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 [8, 10]. Further, webpage analytic APIs and performance monitoring tools allow CPs to monitor the user-perceived performance of their websites [9]. However, as CPs continue to evolve their websites with increasing number of features, the webpage load time (PLT) starts to increase – resulting in poor user experience [6, 13].

To speed up the delivery of static Web content to end-users, CPs make contracts with Content Delivery Networks (CDNs), such as Akamai. CDN servers are distributed deep inside many last mile wired and mobile ISPs worldwide and thus provide low-latency paths to end-users [23, 25]. Additionally, CDNs are motivated to adopt new and upcoming faster Internet technologies, such as HTTP/2 and IPv6 to achieve even faster content delivery for their CP customers [16, 19, 22]. Although CDNs are effective in reducing download times of Web objects they serve, as CPs continue to enhance their websites by embedding *external* resources that the surrogate CDN does not serve, it becomes challenging for the CDN to speed up components of webpages beyond its control [15, 17]. More generally, the usage of external resources have increased in last few years and have thus imposed a much harder challenge on CDNs to improve PLTs.

The performance of such *external* resources have been a great area of interest in the Web performance community. Previous attempts to classify *external* resources as *3^rd Party* (*3P*) involves comparing object hostnames to the hostname

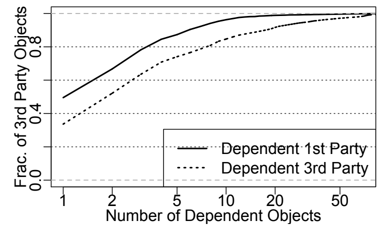

Fig. 1: Dependency on *3P* assets.

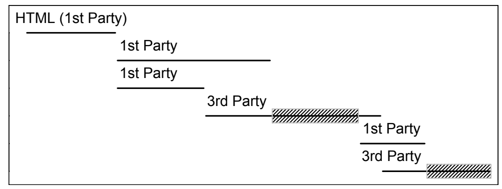

Fig. 2: A waterfall diagram with one *3P* and two *1P* objects.

of the base page URL. However, such techniques often lead to inaccurate classification. For example, while the two hostnames `www.qq.com` and `btrace.qq.com` appear to be from the same party, objects from `www.qq.com` are served from a surrogate CDN infrastructure, whereas objects from `btrace.qq.com` are served from an origin infrastructure. To bring clarity to classification of *3P* assets, we refer the server infrastructure that serves the base page HTML as the *1st Party* (*1P*) provider, such as a CDN provider acting as surrogate infrastructure for its CP customers. Additionally, we refer as to *3P* as any asset embedded in the webpage that is not served by the same infrastructure as the base page HTML. The downloads of such assets cannot be optimized by *1P* provider.

Current *3P* performance analysis techniques only investigate the overall load time of *3P* assets [6, 11], however, such techniques fail to investigate the existence of *3P* assets on webpage critical path [27]. Moreover, previous work measures *3P* performance by comparing PLTs for a webpage with and without *3P* resources [3]. However, we show in Figure 1 that such techniques may not result in accurate comparison of PLTs, as removing a *3P* resource may also remove other resources that are dependent on the removed resource. For example, while 50% of the *3P* resources initiate download of at least one other resource on the webpage, many *3P* resources initiate downloads of upto 10 other resources.

We argue that the key to minimize *3P* impact on PLT is to first understand which specific *3P* assets lie on webpages' critical path. In this paper, we extend our previous work of evaluating the impact of *3Ps* on PLT over mobile networks [21]. Specifically, we investigate *3P* impact on PLT over wired and well-provisioned datacenter networks and suggest a potential solution to mitigate their impact through experimental evaluation. Specifically, we make the following four contributions in this paper:

Analysis of webpage structure: We make extensive use of the open-source data available at the HTTP Archive [2] to expose the characteristics of *3P* assets embedded into the top 16,000 Alexa webpages [7], currently served by four major CDN providers. Specifically, for *3P* assets in each webpage in our dataset, we calculate the number of unique domain names resolved, HTTP requests sent, total bytes, and total uncompressed bytes downloaded, among many other characteristics.

Extensive Measurement: To measure the impact of *3P* downloads on Web performance, we devise a new Web performance metric, `3`$^{\texttt{rd}}$ `Party Trailing Ratio` (`3PTR`), that represents the PLT fraction of the download time of *3P* assets on webpage critical path. As shown in Figure 2, the `3PTR` is the PLT fraction that is accounted for by the sum of the download times of *3P* objects whose download times do not overlap with any *1P* object, as highlighted by the shaded areas. To calculate `3PTR` from HTTP Archive (HAR) files, we encourage readers to experiment with `http://nl.cs.montana.edu/tptr`.

Next, using cellular and wired clients of Gomez Mobile and Gomez Last-Mile testbeds [4], we run several active experiments for three months in 2016 to calculate `3PTR` for hundreds of webpages and identify which *3P* resources impact PLTs. We also use measurement data from HTTP Archive to calculate `3PTR` for the top 16,000 Alexa webpages loaded from a well-provisioned datacenter network [2].

Problems Discovered and Solutions: In our analysis of *3P* performance, we discover two major problems. *First*, we identify that for many webpages, *3P* assets that lie on the webpage critical path contribute up to 50% of the total PLT. To the best of our knowledge, there is currently no known best-practice as to how *1Ps* could optimize *3P* downloads to mitigate their impact on the PLT.
Solution: We investigate how *1P* providers could safely redirect *3P* downloads onto their infrastructures for faster delivery of *3P* assets. Based on our measurements, we demonstrate that rewriting *3P* URLs in a way that enables *1P* servers to deliver *3P* assets improves PLTs by up to 25%. The faster PLTs are achieved as rewritten URLs eliminate DNS lookups to *3P* hostnames, the clients download *3P* assets from *1Ps* using an existing TCP connection to the *1P* server, and that the *1P* (surrogate CDN) servers are likely closer to clients than the *3P* servers. Additionally, *1P* servers could compress any uncompressed *3P* assets before transferring them to clients. And finally, *1Ps* could use new content delivery protocols, such as HTTP/2 and IPv6 for even faster delivery that many *3Ps* do not employ.

Second, using the HTTP Archive data we identify that several *3P* vendors do not compress Web objects even when clients indicate support for compression in HTTP request headers. Incidentally, we identify that some *1P* providers deliver uncompressed objects as well, even when clients indicate support for compression. Our investigation suggests that this behavior is due to misconfigured HTTP response headers on *1P* servers.
Solution: We made recommendations to several *1P* providers, providing them with a list of URLs to configure compression for the objects that they currently serve uncompressed.

2 Data Collection

We use the open-sourced HTTP Archive dataset, an initiative by Google, Mozilla, and other industry leaders, to analyze structures of different websites [2]. The HTTP Archive data is collected using the WebPageTest framework, where webpages are loaded over virtual machines inside a datacenter [14]. The page loads are then translated into a format similar to HTTP Archive format (HAR) containing the timing data and as well as the HTTP request and response headers for each object embedded in the webpage under test.

For our analysis, we extract only the HTTP request and response headers pertaining to the top 16,000 Alexa webpages. In particular, for each requested object we extract HTTP headers indicating the response size, `Cache-Control`, associated hostname, and whether the response was compressed when the client indicates support for compression in the HTTP request headers. Since many *3P* assets load after the `onLoad event` triggered by the Web browser and since we only focus on understanding how much *3P* downloads impact the PLT, we consider the measurement data for objects loaded only until the `onLoad event`.[3]

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 we use in this study. If a hostname for an object does not resolve to a CNAME associated to the *1P* serving the base page HTML, we consider that object as a *3P* asset, with respect to that *1P*. Additionally, if the hostname does not resolve to any CNAME, we consider that hostname as *3P* for all four *1P* CDN providers. While many *1P* providers use anycast addressing for their CDN servers, the four CDN providers we use in this study perform DNS-based addressing and resolve hostnames to CNAMEs associated to them.

Finally, for each webpage, we calculate the total number of domain names resolved and HTTP requests sent for objects that we label as *3P*. We also calculate the total number of bytes, total number of uncompressed bytes, and total number of cacheable bytes delivered by various *3P* vendors by parsing the `Content-Encoding` and `Cache-Control` headers in the HTTP response, respectively. Our total dataset consists of structures for 16,000 webpages requesting a total of 1.6 M objects, out of which about 525 K (32%) objects belong to different *3P* providers.

To collect measurement data pertaining to *3P* impact on PLT, we conduct several active experiments using the Gomez Mobile testbed to load 60 mobile-specific webpages served by the production servers of a major CDN provider [1, 4]. We also conduct active experiments using Gomez Wired Last-Mile testbed to load a set of 376 webpages designed for larger screens from the same CDN. The selected webpages are limited to a few hundred because of the operational costs related to running Gomez experiments and that the chosen webpages are among the most popular sites served by the CDN. Next, we configure both Gomez mobile and wired clients to load each website 400 times and record the browser exposed Navigation and Resource Timing data after each page load [5, 12]. The Navigation and Resource Timing data we obtain from Gomez consists of timestamps when the page starts to load, 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 the *onLoad event* is triggered by the Web browser. Our configured Gomez clients also record the hostnames associated with each requested object, which we use to identify whether the object downloaded is a *3P* asset or a *1P* asset, similarly to how we identify this information using the HTTP Archive data. In addition to using Gomez clients, we use measurement data from the HTTP Archive to extract Resource Timing data pertaining to each object downloaded for the top 16000 Alexa webpages.

[3] We refer to the time Web browsers take to trigger the `onLoad event` as the webpage load time (PLT) [5].

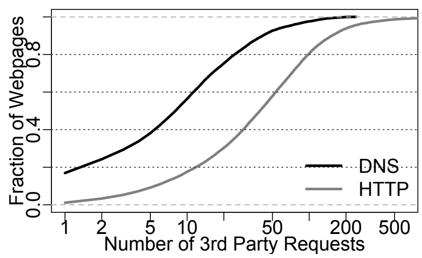

Fig. 3: Distribution of the number of DNS lookup and HTTP requests made to download *3P* assets.

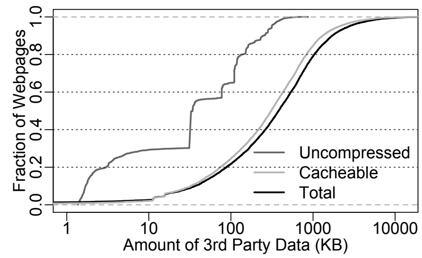

Fig. 4: Distribution of total, uncompressed, and cacheable bytes downloaded from *3P* vendors.

Such a comprehensive measurement allows us to understand the impact of *3P* assets on PLTs when loaded under different network conditions, such as cellular, wired, and well-provisioned datacenter networks.

3 Exposing characteristics of *3P* assets

Using the HTTP Archive data, in Figure 3 we show the distribution of the number of unique domain names resolved and total number of HTTP requests sent by clients to download *3P* assets for different webpages. In general, we observe that 50% of the webpages resolve atleast 10 unique *3P* domain names and issue a total of about 50 HTTP requests to different *3P* vendors. For mobile clients, where radio latency and the latency to cellular DNS servers is a few hundred milliseconds, resolving multiple *3P* domain names introduces significant latency to the overall PLT [23, 22, 26]. Further, such a large number of DNS lookups could result in many round trips to establish several new TCP connections to distant *3P* servers – introducing additional delay to the object load times, especially during the TCP slow start phase of each connection.

Next, in Figure 4, we show the distribution of the total amount of data downloaded from *3P* servers, and as well as the total number of uncompressed bytes transferred by *3P* servers, when clients indicate support for compression in the HTTP request headers. 50% of the webpages download atleast 400 KB data from different *3P* providers, out of which at least 40 KB of data is transferred uncompressed, and almost all of the data transferred by *3P* servers is cacheable by clients or any intermediate Web proxy. The opportunity to cache *3P* data allows *1Ps* to compress and serve requests from their infrastructures.

4 Third Party Trailing Ratio

3P assets embedded on a webpage require multiple DNS lookups and download of hundreds of kilobytes of data, however, *3P* assets that do not lie on the webpage critical path do not impact the PLT. Therefore, we investigate the time spent by *3P* downloads on the critical paths of webpages. For the purposes of this investigation, we devise a new Web performance metric, `3`$^{\texttt{rd}}$ `Party Trailing Ratio` (`3PTR`), that represents the fraction of PLT that is spent only by *3P* downloads and during which no *1P* asset is downloading in parallel, as denoted by the two shaded areas in Figure 2.

To calculate `3PTR`, we employ a two step process as follows: First, using start and end timestamps of all object downloads, we calculate all non-overlapping

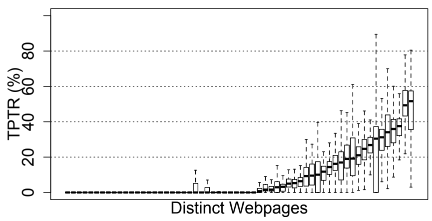

Fig. 5: `3PTR` distributions for webpages served to Gomez Mobile.

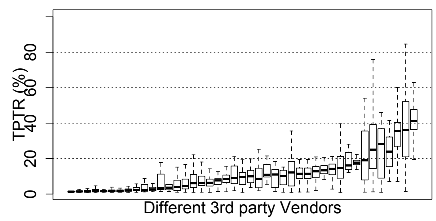

Fig. 6: `3PTR` distributions of *3P* providers served to Gomez Mobile.

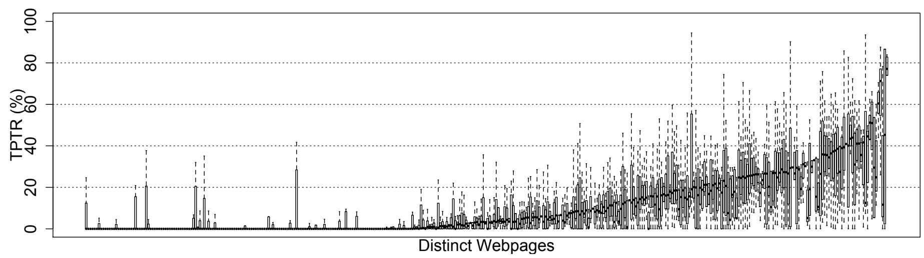

Fig. 7: `3PTR` distributions for different webpages served to wired clients.

time intervals of *1P* and *3P* downloads independently [20]. Second, using the above time intervals, for each *3P* interval we identify whether there is any time duration that does not overlap with any *1P* interval. The sum of all such *3P* time intervals results in the *3P* delay. Finally, the percentage of PLT that belongs to *3P* delay is referred to `3PTR`.

In Figure 5, we show the `3PTR` distributions for 60 webpages served by a major CDN provider, where we load each webpage 400 times from Gomez Mobile clients connected to cellular networks. For figure clarity, we sort pages along the x-axis based on the median `3PTR` value. In general, we observe that *3P* downloads do not impact PLT for about half of the webpages in our dataset. With these webpages, when *3P* assets are being downloaded, one or more longer *1P* assets are also being downloaded in parallel. Therefore, for these webpages, the *3P* downloads do not lie on the critical path. However, for other webpages, *3P* downloads contribute to up to 50% of the total PLT, in the median case. For these webpages, when *3P* assets are downloaded, none of the *1P* assets are being downloaded. Therefore, for these webpages, *3P* downloads lie on the webpage critical path and thus introduce additional latency to the overall PLT. Note that the variation in `3PTR` in Figure 5 arises from the variation in the network conditions, or server processing time. Specifically, as the load time of a *3P* asset changes, the `3PTR` changes as well.

In Figure 6, we separate `3PTR` based on *3P* providers. Specifically, for each *3P* provider on the critical path, we show a boxplot distribution of the `3PTR` contributed by that *3P* provider. From the figure we observe that while some *3P* providers impact PLT of some pages by as low as 5%, other *3Ps* contribute up to 40% of PLT for some webpages. Therefore, to speedup websites it is first important to understand which *3P* provider impacts PLT and then mitigate its impact.

We observe similar impact of *3P* on PLT when loading a different set of 376 webpages using Gomez Wired Last-Mile clients. In Figure 7, we show that the median `3PTR` is zero for about 40% of the webpages. For the rest 60% of the

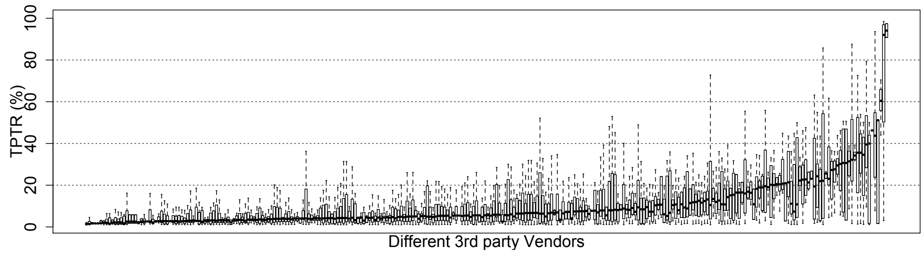

Fig. 8: 3PTR distributions for various *3P* providers for pages served to Gomez Wired Last-Mile clients.

webpages, *3Ps* contribute as much as 50% of the PLT in the median case. As observed earlier, the variation in 3PTR comes from the variation in load times of *3P* assets. Additionally and similarly to Figure 6, in Figure 8 we observe that some *3Ps* impact PLTs of some webpages as low as 1%, while other *3Ps* impact PLT as much as 50%.

Finally, using the measurement data from HTTP Archive, in Figure 9 we show the 3PTR distribution for the top 16,000 Alexa webpages. For example, we see that for about 50% of the webpages served by CDNs A, B, and C, *3Ps* contribute at least 20% of the total PLT, even when webpages are loaded from a cloud datacenter network. For webpages served by CDN D we see that about 65% of the webpages have zero 3PTR, because many webpages served by CDN D are for its own products that do not contain any *3P* assets.

5 Selecting Third Party Objects for Optimization

Based on our analysis of *3P* impact on PLT in different types of networks, we argue for *1Ps* (such as a CDN provider) to rewrite critical *3P* URLs and redirect requests onto their infrastructures to reduce 3PTR. Specifically, rewriting critical *3P* URLs eliminates DNS lookup time for multiple *3P* hostnames, as a rewritten URL can point to the hostname of the basepage that the browser has resolved already. Additionally, URL rewriting allows clients to connect to already warmed-up TCP connections to much closer *1P* servers and download *3P* content while eliminating TCP slow start and time to setup new TCP connections to distant *3P* servers.

Next, when the request to download a *3P* resource arrives at the *1P* server, the *1P* delivers the requested content in one of the following two ways: 1) either from the server's cache; or 2) by retrieving the requested resource from the *3P* server over a proactively established TCP connection. For example, while the first request for a *3P* resource is fetched from *3P* servers, subsequent requests for the same resource are served from *1P* cache. While it is possible that many clients request a specific resource URL, the response for which needs to be personalized according to the user profile, the *1Ps* will need to always fetch the resource from the original *3P* server. For such resources, the client requests contain a cookie in the HTTP headers that enables *3P* servers to customize responses accordingly.

Rewriting *3P* URLs for resources that require a *3P* cookie in the request, or in the response, introduces challenges for *1Ps* to reliably perform URL rewriting. Specifically, many *3P* providers process cookies to perform visitor

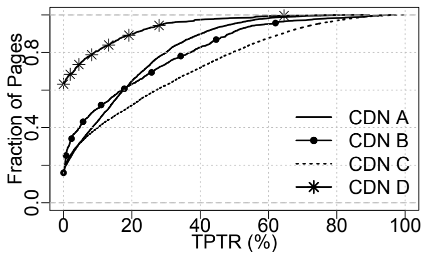

Fig. 9: `3PTR` distribution for webpages served by four CDN providers.

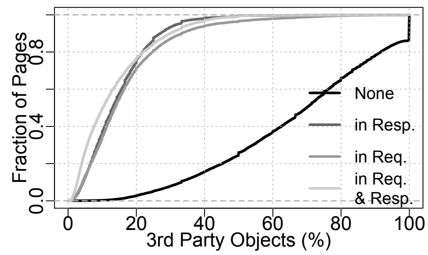

Fig. 10: Distributions of cookie-based requests and responses.

counts for each resource, track user activities, generate responses based on user's recent activities, among others. Therefore, when *1Ps* proxy *3P* traffic on their infrastructure, requests may appear to originate from a smaller pool of *1P* server IP addresses – negatively impacting the visitor count and user tracking services for *3P* providers. Although, *1Ps* could add an `x-Forwarded-For` header in the forwarded HTTP requests, *3P* servers will need to process this header to accurately track users. Finally, if many *3P* requests containing user cookies originate from a unreasonably small pool of *1P* IP addresses, *3P* servers may interpret these requests as a part of a Denial-of-Service (DOS) attack.

In Figure 10, we show the number of *3P* objects that require cookies in requests and/or responses for the top 16,000 Alexa webpages. From the figure we observe that for about 50% of the total websites, at least 70% of the *3P* objects do not require cookies in requests and responses. Therefore, it is promising for *1P* providers to speed up webpages by rewriting URLs for those critical *3P* resources that do not require cookies neither in HTTP requests, nor in HTTP responses. We argue that for each webpage that a *1P* provider serves, the provider could proactively download *3P* resources to identify those that do not contain any cookies and thereafter apply URL rewriting to redirect requests for only those *3P* resources to its own infrastructure before sending the basepage HTML to the client.

6 Third Party Content Acceleration via URL Rewriting

We clone several webpages on a major CDN provider's infrastructure, where each webpage has two versions: 1) where *3P* resources are downloaded from *3P* servers, and 2) where URLs of *3P* resources are rewritten to download from *1P* servers. In Figures 11-16, we show distributions of 200 PLTs for different webpages loaded under different mobile and wired network conditions. Note that the y-axis in these figures is on a log scale. To measure PLTs under different mobile network conditions, we utilize our previous work on simulating cellular networks [24]. For simulating wired network conditions, we only control end-to-end (E2E) latency between clients and servers, as in our observations packet loss on wired networks is minimal and bandwidth is not the limiting factor.

In Figures 11 and 12, we select a webpage with `3PTR` of about 49% and compare its PLTs in various mobile and wired network conditions respectively. Our results show that rewriting *3P* URLs for webpages with such high `3PTR` values result in significantly lower PLTs compared to original page. For example, under *Fair* mobile conditions, the median PLT and the `3PTR` is reduced by 28%

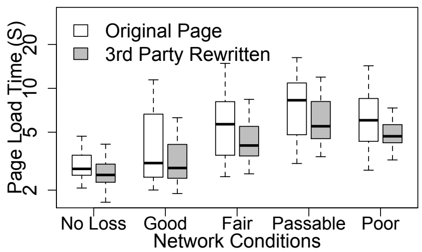

Fig. 11: PLTs in cellular conditions for a page with TPTR of 49%.

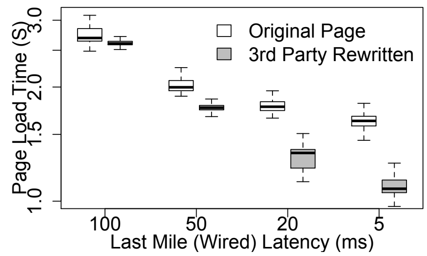

Fig. 12: PLTs in wired conditions for a page with TPTR of 49%.

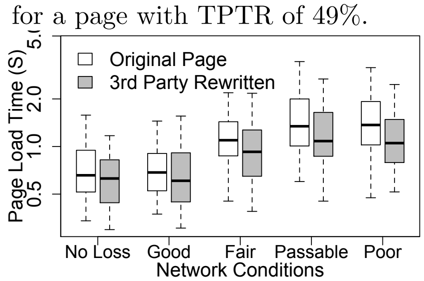

Fig. 13: PLTs in cellular conditions for a page with TPTR of 25%.

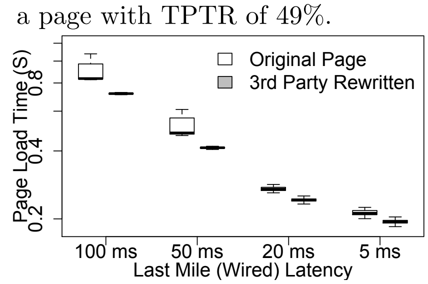

Fig. 14: PLTs in wired conditions for a page with TPTR of 25%.

by rewriting URLs of *3P* assets on the webpage critical path. Additionally, in a last-mile wired network with E2E latency of 20 ms (typical latency between clients and CDN providers), we observe that the median PLT and the `3PTR` with rewritten *3P* URLs is 24% lower than original webpages.

Similarly, when comparing PLTs for webpages with `3PTR` of 25% and 5% in Figures 13-14 and 15-16 respectively, we observe reduced PLTs by rewriting *3P* URLs. However, for these webpages the improvements are less pronounced than we observe in Figures 11 and 12, as the `3PTR` for these webpages is less. For example in Figures 13-14, the median PLTs and `3PTR` of a webpage with rewritten *3P* URLs under *Fair* mobile conditions and 20 ms E2E wired latency are 15% and 10% lower than original webpage, respectively. Similarly, in Figures 15-16, the median PLTs and the `3PTR` under same conditions are 3% and 2.2% lower than for the original webpage.

Note: For CP customers that desire to enable *3P* content acceleration for their webpages, rewriting of all *3P* objects served over HTTPS should be performed only when the CDN provider makes legal agreements with individual *3P* providers to terminate HTTPS connections to their servers and cache the requested content. Additionally, URL rewriting does not introduce any operational complexity to CPs. As CDN providers fetch HTML from their CP customers, CDNs could parse the HTML and apply URL rewriting to *3P* objects that lie on the critical path. Further, as CDNs cache *3P* objects, these objects can be refreshed similarly to how CDNs refresh objects from their CP customers.

7 Discussion

The improvements in PLTs depend on the value of `3PTR` – higher the value of `3PTR`, the more potential for reducing PLTs exists. While our URL rewriting technique demonstrates improvements in PLTs, we argue that for certain *3Ps*,

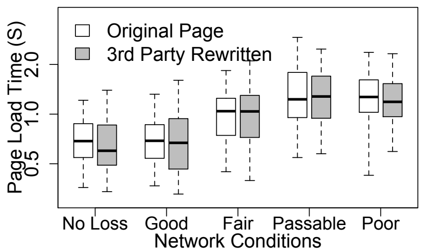

Fig. 15: PLTs in cellular conditions for a page with TPTR of 5%.

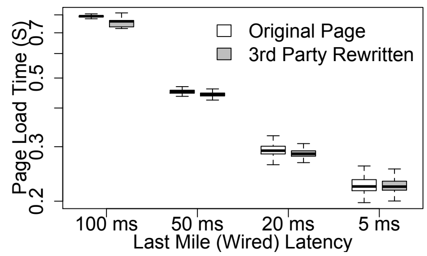

Fig. 16: PLTs in wired conditions for a page with TPTR of 5%.

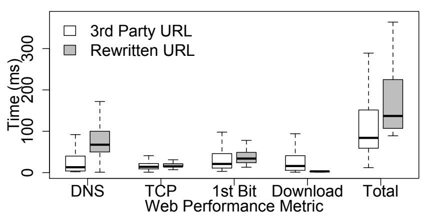

Fig. 17: Comparing performance metrics of a *3P* objects.

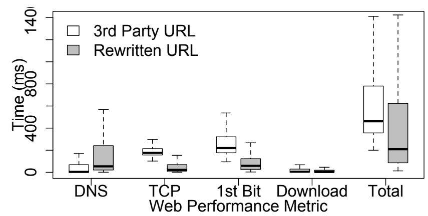

Fig. 18: Comparing performance metrics of another *3P* objects.

rewriting URLs may degrade the performance. For example, in Figure 17 we compare performance of a popular *3P* resource in terms of DNS lookup time, TCP handshake time, time to receive first bit, download time, and the total load time, when loaded from a major CDN provider network and *3P* servers respectively. We observe that DNS lookup time for the *3P* resource is significantly lower than the DNS lookup time for the *1P* CDN provider, likely because the *1P* domain name created for this experiment is not very popular and therefore is not cached by the local DNS resolver. The TCP handshake, first bit, and download time are similar when downloading the same object from *3P* or *1P* servers. As such, the total load time is governed by the DNS lookup time.

Similarly, in Figure 18, we show the same performance metrics for a different *3P* resource. We observe that while DNS lookup time is still higher for a *1P* hostname, the TCP handshake, first bit times are significantly lower when downloading the resource from a *1P* server, which translates to a lower total load time with rewritten URLs. Therefore, we argue that careful performance analysis should be performed for each critical *3P* resource before transmitting HTML with rewritten URLs to clients. For example, if DNS lookup time impacts the overall load time of the object, either the *3P* resource need not be rewritten, or the rewritten URL should use a hostname that client should have already resolved, or configure clients to coalesce TCP connections to multiple *3P* hostnames. 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 [18].

Next, in Figure 19 we show the impact of URL rewriting when the base page is served over HTTP/2 (`h2`). This webpage uses many *3P* hostnames for which the client establishes several TCP connections. When rewriting such a webpage we rewrite all critical *3P* URLs to send requests to the basepage hostname –

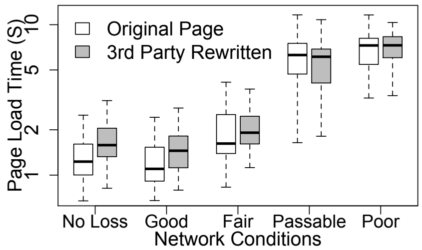

Fig. 19: PLTs distributions when rewriting URLs for an `h2` page.

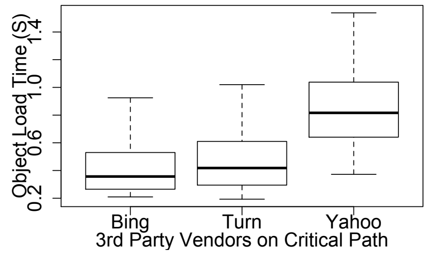

Fig. 20: Distributions showing the variation in *3P* load times.

reducing the total number of connections from several dozen to just one `h2` connection. For such webpages, single TCP connection degrades PLTs as loss interpreted by TCP due to variable radio latency in cellular networks degrades HTTP/2 performance [24]. When measuring PLTs for the same page over `h2` in wired networks, we observe that without packet loss, `h2` offers faster PLTs. Therefore, we argue that for content delivery optimized for mobile networks, it is important to consider impact on PLT of the number of TCP connections that remain after rewriting URLs.

Finally, for another webpage with over 40 different *3P* hostnames and `3PTR` of about 30%, we identify that the performance variation from a few *3P* resources (for which we could not perform URL rewriting as they contain cookies) negate the benefits of URL rewriting for other *3P* resources. As shown in Figure 20, the three *3P* resources downloaded from Bing, Turn, and Yahoo servers vary by over 1 second. For example, a resource loaded from Yahoo servers takes anywhere from 300 ms to 1.5 s. Therefore, we argue that for webpages that embed cookie-based *3P* objects with high performance variation may not assist the URL rewriting technique to improve PLTs.

Limitation: The one (minor) limitation of `3PTR` is that for some webpages, `3PTR` may give a lower bound on the impact of *3P* downloads on PLT. For example, when a *3P* object initiates the download of a *1P* object and the *3P* downloads in parallel with some other *1P* object, the TPTR is calculated as zero. As the *3P* object initiates the download of a *1P* object, that *3P* lies on the webpage critical path, however, `3PTR` does not consider object dependencies within a webpage when calculating impact of *3P* downloads on PLT. To detect object dependencies, the `Referrer` header in the HTTP requests can be used to identify the initiator of the request. However, the Resource Timing API does not record the `Referrer` header and thus we designed `3PTR` to utilize the start and end timestamps for each loaded object. Using HTTP Archive data, we identify that less than 2-10% of the webpages possess such dependencies and therefore `3PTR` calculates accurate *3P* impact for majority of the webpages.

8 Conclusions

Our large scale investigation on 3^{rd} *Party* performance reveals that *3Ps* can impact the overall webpage load time by up to 50%. Through extensive experimentation, we demonstrate that redirecting *3P* traffic to *1P* infrastructure improves webpage load times. We, therefore, make recommendations to *1P* providers to investigate the existence of *3P* resources the critical path of webpages and utilize URL rewriting to improve Web performance for end-users. In the future, we plan to perform even larger scale measurements on production Web traffic.

ACKNOWLEDGMENTS: We thank Ilya Grigorik, Shantharaju Jayanna, Wontaek Na, and Kanika Shah for their help. We also thank National Science Foundation for supporting this work via grants CNS-1555591 and CNS-1527097.

References

[1] Gomez Last-Mile Testbed. `https://goo.gl/BtwSWY`, Nov. 2009.
[2] HTTP Archive: Interesting stats. `http://httparchive.org/`, 2010.
[3] Performance of 3rd Party Content. `http://stevesouders.com/p3pc/`, Feb. 2010.
[4] Gomez (Dynatrace Synthetic Monitoring). `https://goo.gl/4JTjJy`, Jul. 2015.
[5] Navigation Timing. `http://w3c.github.io/navigation-timing/`, Aug. 2015.
[6] The Truth Behind the Effect of Third Party Tags on Web Performance. `https://goo.gl/24f09c`, Dec. 2015.
[7] Alexa Top Sites. `http://www.alexa.com/topsites`, Jul. 2016.
[8] Facebook for Developers. `https://developers.facebook.com/`, Jun. 2016.
[9] Google Analytics Solutions. `https://analytics.googleblog.com/`, Jun. 2016.
[10] Google Fonts. `https://fonts.google.com/`, Jun. 2016.
[11] Performance Measurement for the Real World. `https://www.soasta.com/performance-monitoring/`, Aug. 2016.
[12] Resource Timing. `https://www.w3.org/TR/resource-timing/`, Jul. 2016.
[13] Third-party content could be slowing Britain's retail websites. `https://goo.gl/1gi1Li`, Mar. 2016.
[14] WebPageTest Framework. `http://www.webpagetest.org/`, Jul. 2016.
[15] K. Alstad. Can third-party scripts take down your entire site? `https://goo.gl/V0iLfa`, Jun. 2014.
[16] M. Belshe, R. Peon, and E. M. Thomson. Hypertext Transfer Protocol Version 2 (HTTP/2), RFC 7540, May 2015.
[17] B. Bermes. Third Party Footprint: Evaluating the Performance of External Scripts. `https://goo.gl/Cqhafq`, Sept. 2014.
[18] 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.
[19] F. Chen, R. K. Sitaraman, and M. Torres. End-User Mapping: Next Generation Request Routing for Content Delivery. In *ACM SIGCOMM*, Aug. 2015.
[20] R. C. Enaganti. Merge Overlapping Intervals. `http://www.geeksforgeeks.org/merging-intervals/`, Aug. 2015.
[21] U. Goel, M. Steiner, W. Na, M. P. Wittie, M. Flack, and S. Ludin. Are 3rd Parties Slowing Down the Mobile Web? In *ACM S3 Workshop*, Oct. 2016.
[22] 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*, Oct. 2016.
[23] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. Detecting Cellular Middleboxes using Passive Measurement Techniques. In *ACM PAM*, Mar. 2016.
[24] U. Goel, M. Steiner, M. P. Wittie, M. Flack, and S. Ludin. HTTP/2 Performance in Cellular Networks. In *ACM MobiCom (Poster)*, Oct. 2016.
[25] E. Nygren, R. K. Sitaraman, and J. Sun. The Akamai Network: A Platform for High-Performance Internet Applications. In *ACM SIGOPS*, July 2010.
[26] J. P. Rula and F. E. Bustamante. Behind the Curtain: Cellular DNS and Content Replica Selection. In *ACM IMC*, Nov. 2014.
[27] X. S. Wang, A. Balasubramanian, A. Krishnamurthy, and D. Wetherall. Demystify Page Load Performance with WProf. In *USENIX NSDI*, Apr. 2013.