Reproducibility Report: Web Performance Pitfalls

Curt Polack
TU Munich, Germany
curt.polack@tum.de

ABSTRACT

In this paper, we re-implement a work published in proceedings of the 20th International Conference for Passive and Active Measurement (PAM) in 2019: "Web Performance Pitfalls". The paper used a number of web-scraping and performance evaluation tools to perform measurements of ca. 2000 popular websites. These measurements were used to demonstrate and evaluate differences in measurements by tool, standard, etc. that can cause different and possibly misleading results in papers that examine web performance.

This paper details the process of reproducing the original results and conclusions using the data and scripts provided in the original paper. It also details the use of the aforementioned scripts to perform a new set of measurements for ca. 1000 popular websites. The difficulties of obtaining new measurements and their results are expounded upon and the results compared with those of the original paper.

The original data provided produced results that coincide with the results and plots in the original paper. The measurement scripts provided for Chrome and Firefox with Selenium did not work in either the environment used in the original paper or in a modern Linux environment. The measurements that were collected using Firefox with Marionette and analyzed by the scripts provided did, however, generate similar results to those produced by the original data regarding two major web performance metrics: load time and number of objects. The new measurement's results did not correspond to those of the original dataset regarding object counts.

CCS CONCEPTS

• **Networks** → **Network performance evaluation**; **Network measurement**;

KEYWORDS

Web performance, Measurement

1 INTRODUCTION

Web browsing is one of the most widely-used applications used in today's Internet ecosystem. Measuring and understanding the performance of web browsing is, therefore, of the utmost importance. A number of metrics have been developed and used as benchmarks to accurately reflect the performance of web applications for users. The diversity of web pages, user devices, browsers, metrics (e.g. browser-centric and user-centric) as well as the lack of clearly established standards lead to difficulties when quantifying performance.

A number of studies (e.g. Varvello et al. [9]) over the years have attempted to measure web performance in some form. Studies that perform research on web performance are required to provide adequate details regarding chosen metrics, tools, and data sources to ensure that results are correctly interpreted, comparable, and reproducible. Enghardt et al. began with a survey of numerous studies regarding web performance and summarized the different tools and metrics used. Although one third of the surveyed studies did not precisely specific metrics and/or data sources, the most commonly used tools were identified and incorporated into subsequent measurements.

Enghardt et al. continue by creating a test environment in which different tools and metrics, researched during the aforementioned survey, can be compared with one another. The tools that were used in the majority of the surveyed papers were utilized: Firefox with Selenium, Firefox with Marionette, and Chrome with DevTools. The paper continues by comparing the differences in metrics as well as results and outlining the most prominent pitfalls that should be avoided when measuring web performance. These pitfalls include the inclusion of HTTP redirects in timing statistics and the choice of correct measurements for object sizes and counts. The paper concludes with a set of guidelines for web performance studies. This paper examines the original measurements as well as new measurements using modern versions of the aforementioned tools to examine the validity of the recommendations / pitfalls in the original paper.

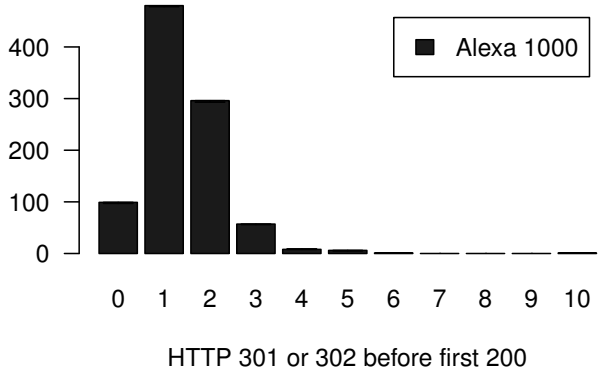
2 BACKGROUND

Among the most important web metrics are load times, number & size of objects, and page size. Each of the aforementioned metrics can be measured according to a numerous definitions and using data from diverse data sources. The following section provides an overview of these definitions and data sources used in this paper.

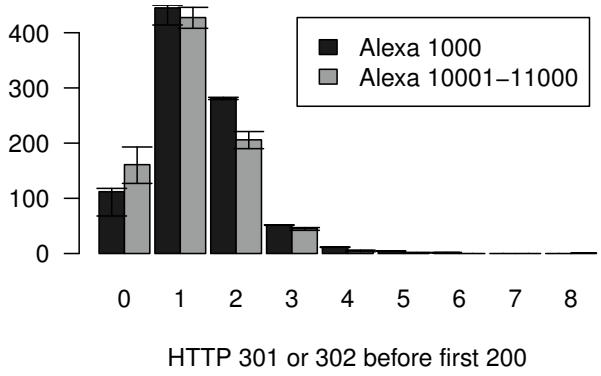
2.1 Load Times

The time required for loading a web page correlates strongly with user experience [5]. A browser normally loads a web page in multiple steps: load and parse the base document, construct a Document Object Model (DOM), load and process referenced objects, render / display the results. Although Page Load Time (PLT), defined as the time until the onLoad event, is most often used to measure page load times, there are a number of other metrics. For example, Time To First Paint (TTFP) and Above The Fold Time (AFT) are triggered before PLT when the content is first displayed and available to the user. The start times used for calculating load times can always differ (e.g. navigationStart, fetchStart). One of the main takeaways from [7] is that redirects can strongly affect these measurements and should be accounted for; Figure 2 shows the impact redirects can have on PLT and Figure 1 the prevalence of such redirects.

There is also a plurality of data sources for calculating load times. The standardized API for navigation timings [1] is used to fetch load times based on browser events. It is important to note that events / metrics provided by the navigation timings API are not necessarily defined in the same way for all browsers. HTTP Archive files [2] and the Resource Timings API [3] also provide load time data. The

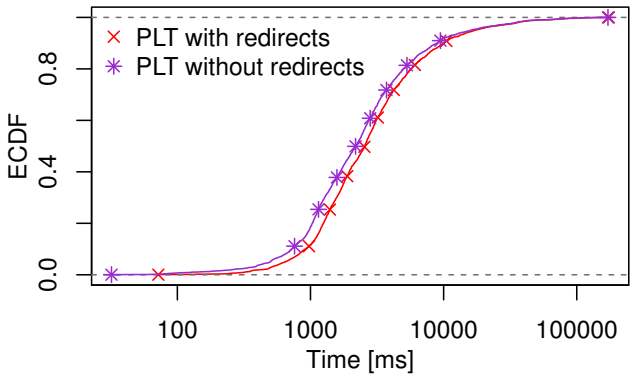


(a) New Measurements

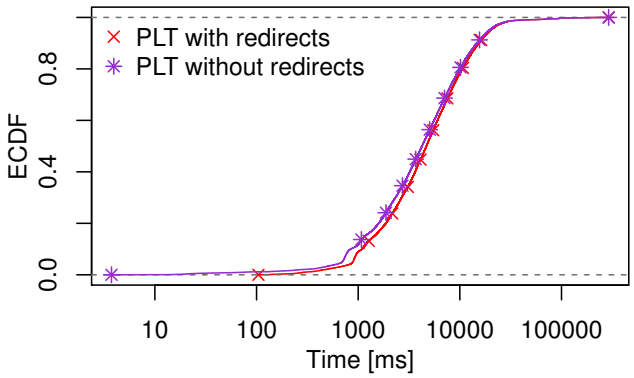


(b) Original Measurements

Figure 1: Number of initial Redirects



(a) New Measurements



(b) Original Measurements

Figure 2: Page Load Time (PLT) with and without initial redirects

majority of modern browsers (e.g. Chrome, Firefox) implement the Navigation Timings and Resource Timings APIs; HAR files can be exported using developer tools. Both Chrome and Firefox provide remote debugging and page load automation interfaces (i.e. Chrome DevTools & Firefox Marionette). There are also third-party automation frameworks that allow for the same basic functionality as well as extended functions (e.g. keyboard input); Selenium, one of the most popular such tools, is used in the original paper.

2.2 Number and Size of Objects

In order to estimate the complexity of web pages, metrics such as Object Index, Object Count, and Byte Index are employed. Since web pages are often constantly loading - even after the initial page load - object counts should only count objects loaded by the onLoad event. Calculating a count of the initial objects can be done using objects in the DOM or HTTP request-response pairs. Object size normally reflects the encoded size (i.e. the count of bytes transferred over the network) but can also reflect the decoded (i.e. decompressed) number of bytes. Byte Index refers to the integral of the total sizes of objects loaded over time and is an important metric for the size of Objects[4].

To obtain the number of objects, one can count the number of HTTP request-response pairs in HAR files. The number of objects can also be obtained using the Resource Timings API. Both the Resource Timings API and HAR files supply the encoded and decoded body size. Alternatively, the number of objects can be extracted from packet capture traces if all elements can be decrypted; if this is not the case, object sizes can differ due to TLS padding.

3 EXPERIMENTAL SETUP & RE-IMPLEMENTATION

This section details the experimental setup used both in the original paper [7] as well as this paper. It also documents the process of and difficulties during the data collection and evaluation. Lastly, it describes the origins of the datasets used.

3.1 Experimental Setup

Enghardt et al. used the following tools: (1) Firefox 61.0.2 with Selenium 3.14.0 and geckodriver 0.21.0, (2) Firefox 62.0.2 with Marionette, and (3) Chrome 69 with Chrome DevTools. This multitude of tools was used in order to provide data for a comparison of different frameworks as well as web pages. All pages were loaded on a Thinkpad L450 running Debian 9 (Stretch). The extension used to export HAR files was har-export-trigger 0.6.1. The Thinkpad (i.e. vantage point) was connected directly to a university network to avoid bandwidth / latency issues. To minimize the effects of DNS

caching and resolver delay, a recursive resolver within close range (in terms of the network) was used.

The hardware and software employed in this paper were similar yet more modern. Ubuntu 20.04. was run in a virtual machine on a Matebook X Pro (1st Generation). The use of the following tools was attempted / responsible for the results: (1) Firefox 82 with Selenium 3.141.59 and geckodriver 0.29.0, (2) Firefox 82 with Marionette, and (3) Chrome 88 with Chrome DevTools. The extension used to export HAR files was, as in the original version, har-export-trigger 0.6.1. as no new versions were released since the original paper. The vantage point was connected directly to a stable home network using ethernet; the connection was not shared for the duration of the tests. A recursive DNS resolver close to the vantage point, in terms of the network, was used as well. During testing of scripts in the new environment, there were difficulties; a second virtual machine with the original experimental setup (Debian Stretch, etc.) was used to reproduce / solve these issues. Despite trying the versions specified in the original paper as well as the more modern versions, it was not possible to consistently extract HAR files using Firefox with Selenium and Chrome with PyDevTools as specified in the original paper.

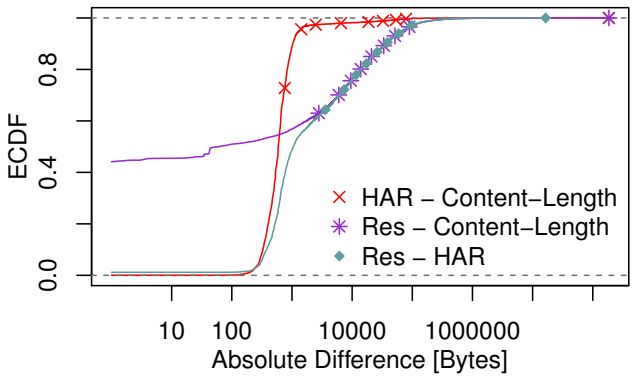
3.2 Scripts

All scripts used in [7] are available in [6]. The scripts are grouped into three categories: measurement / load, computation, and evaluation. The measurement scripts include separate scripts for each of the aforementioned tools (e.g. Firefox with Selenium) as well as a script to parse the exported data and capture packets for comparison with and verification of the results. The computation scripts take the raw results as input and output standardized data that is then used by the evaluation scripts to provide percentiles, averages, and graphs.

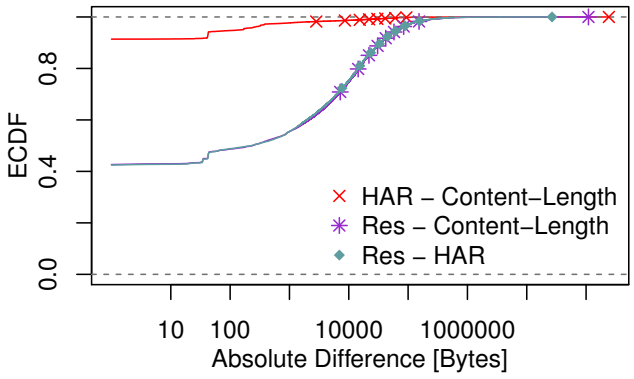
The scripts for Firefox with Selenium and Chrome with DevTools were unable to export HAR files in both the Ubuntu virtual machine with all updates installed or a Debian virtual machine using the software versions specified in the original paper. The inability to export the HAR files that were key to the measurements which compared tools led to the decision to focus on reproducing measurements using Firefox with Marionette; this decision meant that results for chrome could not be reproduced. Multiple of the computation and evaluation scripts also exhibited a number of smaller errors, e.g. the compute script was not compatible with the original data format. The adjusted / corrected scripts are available in the following on GitHub[1].

3.3 Datasets

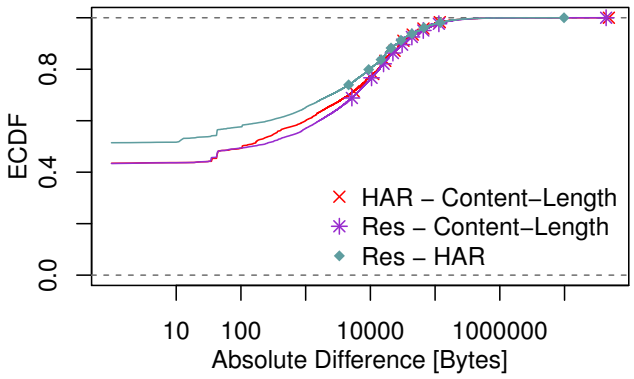
Alexa Top Lists are, despite some well-known limitations, among the most used data sources for research regarding the internet [8]. For this reason, Enghardt et al. used two distinct website lists in their paper: Alexa 1000 (September 18, 2018) and Alexa 10001-11000 (September 30, 2018). Each page was accessed a total of 10 times using different framework over a period of several days. In this paper, I used Alexa 1000 (January 18, 2021) and performed 3 measurements for each website over a period of several days. The original[2] and reproduced[3] datasets collected can be found online. The captured packets (pcap format) that were used to verify results were also not available in the original dataset due to their size.



(a) New Measurements



(b) Original Measurements (Firefox Only)



(c) Original Measurements (Chrome Only)

Figure 3: Object sizes: differences due to metric for all objects

4 EVALUATION

In their original work, Enghardt et al. identified four pitfalls and their effects in the collected datasets. This section examines three

[1] https://github.com/curtp67/web-measurement-tools-reproduction

[2] http://dx.doi.org/10.14279/depositonce-8100

[3] https://syncandshare.lrz.de/getlink/fiFSuxPcpDVkZhwfeTVNDYxc

major pitfalls: Redirects, Object Sizes, and Byte Index more closely using both data from the original dataset and the newly collected measurements.

4.1 Redirects

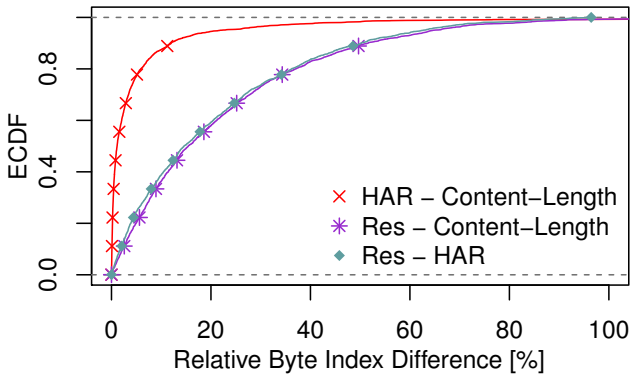
As mentioned in Section 2, Enghardt et al. makes the observation that initial redirects can increase PLT substantially. As seen in Figure 2, the dataset provided in the original paper also makes the more specific observation that this effect is especially pronounced for short page loads. While the new measurements / dataset does confirm that initial redirects can have a large effect on PLT, the difference in short load times is less pronounced.

The percentage of redirects taking an abnormally long time remained relatively constant between both datasets: In the original dataset, 11.20482% (Firefox Datasets Only: 13.9249%) of redirects take more than one second; The new dataset provides a similar result with 12.34043% of redirects taking more than one second. The number of redirects increased slightly in the new dataset: in the original dataset, the mean number of redirects performed by Firefox using Marionette was 1.258; the new dataset had a mean number of redirects of 1.393. The new dataset confirms that redirects can have an impact on PLT, albeit less significant than in the original dataset. The suggestion for papers to make a conscious choice regarding the inclusion or exclusion of redirects in timing is, therefore, still valid.

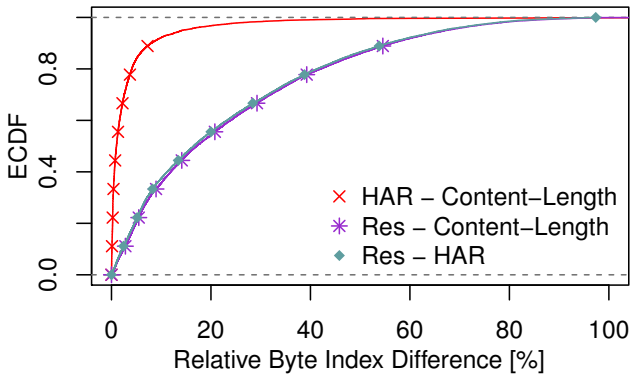
4.2 Object Sizes

In the original paper, Enghardt et al. compare the object sizes for all pages (including those transferred over encrypted connections). In Figure 4, the object size differences across the different data sources (HAR file body size, Content-Length header in HAR file, and Resource Timings) are displayed. Content-Length header (taken from browser) is used as a close approximation for a baseline. In the original dataset (Firefox Only): Resource timings, HAR body size, and HAR Content-Length provide the exact same object size as the baseline in 42.5%, 91.3%, 42.4% of cases respectively. These percentages are drastically different in the new dataset for both HAR body size, and HAR Content-Length: Only 0.02% and 0.88% match the baseline exactly. Resource Timings, however, provide a relatively accurate measure of object sizes and math the baseline in 43.88% of all cases.

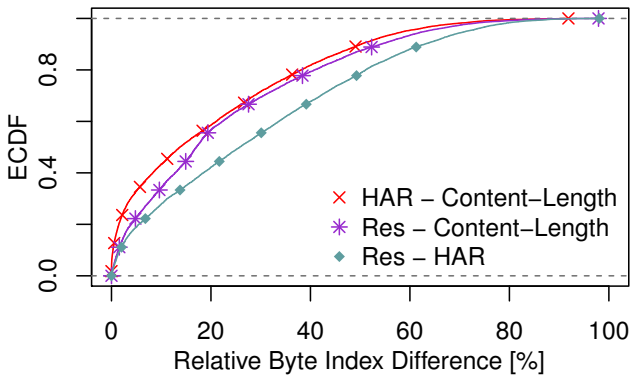
This data, as shown in Figure 3, clearly shows that the new measurements do not confirm the conclusion that Content-Length found that HAR files is the most accurate measure of object sizes. The new dataset instead leads to the conclusion that object sizes provided by Resource Timings provide the most accurate approximation of total object size. The fact that it is only an exact match in fewer than half of all cases, however, does not make it an ideal measure. The difference between HAR Content-Length is however very strongly concentrated in one range between 100 and 1000 bytes, where over 90% of measurements can be found (see Figure 3a); this suggests that there is a consistent reason for this difference and that it can be accounted for.



(a) New Measurements



(b) Original Measurements (Firefox Only)



(c) Original Measurements (Chrome Only)

Figure 4: Byte index: difference due to data source

4.3 Object Count & Byte Index

In the original paper, Enghardt et al. found that not only do object sizes differ significantly by data source but also object counts. In the original Alexa 1000 dataset, object counts for the same web pages differed between HAR files and Resource Timings by at least 7 objects. Figure 4 shows the relative difference between Byte Index for the same page load using HAR body size, Resource Timings body size and the Content Length header. As observed in the original measurements (Firefox Only), the Byte Index is close to identical for

Content-Length and HAR body size; this observation is confirmed in the new measurements, as seen in Figure 4a.

The results for Chrome, as seen in Figure 4c, show that the Byte Index for both Resource Timings and HAR differed significantly from that of Content-Length. Unfortunately, due to the difficulties described in 3.2 it was not possible to confirm these findings using new measurements in Chrome. Thus, the conclusion of the original paper that resource timings do not include all objects of a web page is confirmed by the new dataset for Firefox.

5 CONCLUSION

In this paper, we re-implemented the paper Web Performance Pitfalls, published in 2019 during the International Conference for Passive and Active Measurement (PAM). The re-implementation focused on reproducing the conclusions of the original paper using the data provided publicly as well as taking new measurements to confirm it. During the implementation process, problems with a number of the provided scripts caused smaller adjustments to be necessary and no measurements to be taken with 2/3 of the frameworks (i.e. Firefox with Selenium and Chrome with DevTools). Three new datasets were created using the scripts, based on the Alexa 1000 list.

This paper's reproduction effort focused mainly on three main pitfalls that were described in the original paper: Redirects, Object Size, and Object Count. The results required for the conclusions drawn for these pitfalls were confirmed by the original data, but only 2 / 3 were confirmed by the new measurements created using Firefox with Marionette on a modern system. The new dataset contradicts the conclusion in the original paper that the Content-Length in HAR files is the most reliable data source for object sizes but does confirm the large inconsistencies in object size between data sources. The other examined conclusions regarding redirects and object count / byte index were confirmed by the new dataset.

The plots and results found in this work and the original paper can be reproduced using the data and scripts available online, as described in Section 3. The data in both papers underlines the importance of improving the documentation of studies of web performance as well as choosing performance metrics carefully and deliberately.

REFERENCES

[1] 2012. Navigation Timing. (Dec 2012). https://www.w3.org/TR/navigation-timing/

[2] 2012. W3C Editor's Draft: HTTP Archive (HAR) format. (Aug 2012). https://w3c.github.io/web-performance/specs/HAR/Overview.html

[3] 2020. Resource Timing Level 2. (Aug 2020). https://www.w3.org/TR/resource-timing-2/

[4] Enrico Bocchi, Luca De Cicco, and Dario Rossi. 2016. Measuring the Quality of Experience of Web Users. In *Proceedings of the 2016 Workshop on QoE-Based Analysis and Management of Data Communication Networks (Internet-QoE '16)*. Association for Computing Machinery, New York, NY, USA, 37?42. https://doi.org/10.1145/2940136.2940138

[5] S. Egger, T. Hossfeld, R. Schatz, and M. Fiedler. 2012. Waiting times in quality of experience for web based services. In *2012 Fourth International Workshop on Quality of Multimedia Experience*. 86–96. https://doi.org/10.1109/QoMEX.2012.6263888

[6] Theresa Enghardt. 2019. Web Measurement Tools. (Jan 2019). https://github.com/theri/web-measurement-tools

[7] Theresa Enghardt, Thomas Zinner, and Anja Feldmann. 2019. Web Performance Pitfalls. In *Passive and Active Measurement*, David Choffnes and Marinho Barcellos (Eds.). Springer International Publishing, Cham, 286–303.

[8] Quirin Scheitle, Oliver Hohlfeld, Julien Gamba, Jonas Jelten, Torsten Zimmermann, Stephen D. Strowes, and Narseo Vallina-Rodriguez. 2018. A Long Way to the Top: Significance, Structure, and Stability of Internet Top Lists. In *Proceedings of the Internet Measurement Conference 2018 (IMC '18)*. Association for Computing Machinery, New York, NY, USA, 478?493. https://doi.org/10.1145/3278532.3278574

[9] Matteo Varvello, Kyle Schomp, David Naylor, Jeremy Blackburn, Alessandro Finamore, and Konstantina Papagiannaki. 2016. Is the Web HTTP/2 Yet?. In *Passive and Active Measurement*, Thomas Karagiannis and Xenofontas Dimitropoulos (Eds.). Springer International Publishing, Cham, 218–232.