You Call This Archaeology? Evaluating Web Archives for Reproducible Web Security Measurements

Florian Hantke
CISPA Helmholtz Center for Information Security
florian.hantke@cispa.de

Stefano Calzavara
Università Ca' Foscari Venezia
stefano.calzavara@unive.it

Moritz Wilhelm
CISPA Helmholtz Center for Information Security
moritz.wilhelm@cispa.de

Alvise Rabitti
Università Ca' Foscari Venezia
alvise.rabitti@unive.it

Ben Stock
CISPA Helmholtz Center for Information Security
stock@cispa.de

ABSTRACT

Given the dynamic nature of the Web, security measurements on it suffer from reproducibility issues. In this paper we take a systematic look into the potential of using web archives for web security measurements. We first evaluate an extensive set of web archives as potential sources of archival data, showing the superiority of the Internet Archive with respect to its competitors. We then assess the appropriateness of the Internet Archive for historical web security measurements, detecting subtleties and possible pitfalls in its adoption. Finally, we investigate the feasibility of using the Internet Archive to simulate live security measurements, using recent archival data in place of live data. Our analysis shows that archive-based security measurements are a promising alternative to traditional live security measurements, which is reproducible by design; nevertheless, it also shows potential pitfalls and shortcomings of archive-based measurements. As an important contribution, we use the collected knowledge to identify insights and best practices for future archive-based security measurements.

ACM Reference Format:
Florian Hantke, Stefano Calzavara, Moritz Wilhelm, Alvise Rabitti, and Ben Stock. 2023. You Call This Archaeology? Evaluating Web Archives for Reproducible Web Security Measurements. In *Proceedings of ACM CCS 2023 (ACM CCS).* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Web measurements are a popular tool in the security community to assess whether state-of-the-art defense mechanisms are used on the Web and whether security best practices are getting traction in the wild. For example, many papers at reputable computer security venues measured the adoption and correct configuration of defense mechanisms for web applications, such as Content Security Policy (CSP) [42], HTTP Strict Transport Security (HSTS) [22] and cookie security attributes [7]. Other relevant measurements in the past assessed the security implications of remote script inclusions [33] and provided insights on the web tracking ecosystem [14].

As a matter of fact, however, web measurements are surprisingly hard to carry out in a reproducible and scientifically rigorous way. First, the Web is *ephemeral*: what we observe today is not what we will observe tomorrow, because live websites are routinely subject to changes. Moreover, the Web is *erratic*: different vantage points might yield different observations of web pages even at the same time, due to the dynamic nature of modern websites and other factors like the ever-increasing popularity of content delivery networks [19]. This means that, although papers are clear about their measurement methodology and authors may be willing to share their code, reproducing and validating the results of published web measurements is virtually impossible. This issue has been acknowledged by the web security community and recent papers investigated ideas to make web measurements more reproducible [2, 13, 18]. However, such efforts are primarily concerned with defining guidelines and recommendations to ensure papers include sufficient information about their measurement methodology to allow other researchers to assess whether what has been measured is meaningful and again perform the study under the same conditions. This is certainly useful, yet we are not aware of research that proposed concrete tools to mitigate the negative effects on reproducibility arising from the ephemeral and erratic nature of the Web. Of course, a viable approach towards full reproducibility would be following the measurement best practices defined by the research community [2, 13, 18] and sharing the collected datasets. However, guidelines are informal, may be incomplete, and there is no evidence that they have been followed correctly by researchers during dataset construction.

This paper explores a different solution in the design space of reproducible web measurements. In particular, we observe that there already exist effective countermeasures to the ephemeral nature of the Web in the form of public *web archives*, like the Internet Archive (IA, available at archive.org). Such services periodically crawl web pages and archive a copy, which is later made available to requesting clients, thus enabling everyone to access the same data and reproduce findings easily. Indeed, web archives have already been used for web security measurements in the past, in particular, to perform historical analyses aimed at understanding the evolution of relevant security aspects of the web platform [25, 33, 36, 41].

ACM CCS, November 26-30, 2023, Copenhagen

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Research Questions and Contributions

Although the idea of using web archives for web security measurements has been explored in the past, prior work only focused on historical studies and provided just limited insights on using archives correctly. We here focus on the following questions:

- *What are the best sources of archival data available to date and how can we compare their effectiveness?* Prior security studies are all based on data from the IA alone, with the exception of [36], which also used Common Crawl (commoncrawl.org) as a second data source. However, these are not the only available options. The idea of web archives is, in fact, so well established that the Memento protocol provides a standard access interface to archival data [11, 12] and more than 30 web archives are included in the Memento Time Travel project.[1] Having multiple data sources may be useful to improve the coverage of archive-based measurements with respect to different domains and historical periods, hence their quality should be rigorously evaluated.
- *Can we trust the correctness of archival data and, in particular, the security conclusions drawn by their analysis?* Prior studies normally assumed archival data to be largely correct, most notably due to the lack of a ground truth. While assuming that archives operate correctly sounds like an acceptable practice, we are not just concerned about archives failing at crawling time, exhibiting buggy behavior, or being actively tampered with, but we are also interested in the bias introduced by the over-reliance on a specific archive. Indeed, each archive operates from a specific vantage point and by means of periodic snapshots. Hence the points of view of different archives might legitimately be different and biased.
- *Can we leverage recent archival data as an effective substitute of live data for web security measurements?* Although web archives have been used for historical measurements, they have never been used to emulate traditional live measurements. Still, if recent archival data closely matched live data (at least in terms of the enabled security inferences), archive-based measurements could be a viable, reproducible alternative to otherwise ephemeral live security measurements, reproducible alternative to otherwise ephemeral live security measurements. This approach might enable new web measurements that are easily reproducible by design.

We thus here make the following contributions:

(1) We systematically analyze a set of public web archives with respect to the quantity and freshness of their archival data collected from January 2016 to July 2022. Our analysis shows that the IA essentially dominates all its competitors in all our experiments, because it does not just crawl more domains but also does that more frequently. Notably, combining the IA with all the other analyzed archives only gives a negligible advantage in terms of the raw archival data that we are able to collect. Hence the use of the IA as a single source of archival data is a plausible option (Section 3).
(2) We assess the correctness of the data available in the IA for historical security measurements. Our analysis is based on two case studies (security headers and JavaScript inclusions), where we identify a few subtleties and pitfalls that may affect the correctness of security inferences (Section 4).
(3) We investigate the feasibility of performing web security measurements on top of the data stored in the IA in place of traditional live data, i.e., we assess whether one can make web measurements reproducible by using archival data which is temporally close to the live measurement date. Our experiments show that archival data are crawled often enough to support fine-grained analyses, enable reproducibility, and allow one to draw security inferences close to those of live data. However, they also identify limitations of archive-based measurements that other researchers should be well aware of (Section 5).
(4) We distill insights and best practices for future archive-based security measurements based on the results of our systematic investigation (Section 6).

Our data collection code is available online: https://github.com/cispa/internet-archive-study. All the data that is unavailable via web archives is made available upon request.

2 BACKGROUND

We review here the key ingredients needed to follow the paper.

2.1 Reproducibility

Reproducibility is a broad term whose different facets have been formally defined by the ACM.[2] To clarify the goals of this paper, we first review the ACM terminology: (*i*) *Repeatability*: the same team can reliably repeat their own computation; (*ii*) *Reproducibility*: a different team can obtain the same results using the authors' own artifacts; (*iii*) *Replicability*: a different team can obtain the same results using artifacts which they developed independently.

The Web's ephemeral and erratic nature hinders the mere repeatability of web measurements, because two different execution of the same data collection pipeline may yield different results at different times. By using web archives, we may instead be able to achieve replicability of web measurements because most archival data is intended to be made indefinitely available to requesting clients. Having clarified this point, we now lighten terminology by informally using just the term "reproducibility" in the paper.

2.2 Web Archives

We use the term *web archive* to refer to any service which periodically stores copies of public web pages, associates them to their archival date, and makes them available to requesting clients. A variety of archives exist on the Web, however, in this paper, we primarily focus on those supporting the standard Memento protocol for web archiving [11, 12]. Our focus on Memento has the advantage of defining a systematic criterion to identify existing web archives to use in our study, with the additional benefit that they can be accessed using the same standardized protocol.

Besides Memento-based archives, we also consider Common Crawl as a possible alternative source of archival data. Common Crawl archives parts of the Web once a month and stores the content

[1] https://timetravel.mementoweb.org/

[2] https://www.acm.org/publications/policies/artifact-review-and-badging-current

as one snapshot. The reason why we use Common Crawl is that it contains a massive amount of data: its October 2022 snapshot includes more than 2.55 billion pages, with its index alone being larger than 2TB; moreover, Common Crawl was already used in a previous web security measurement [15, 36]. The content archived on Common Crawl is stored in form of large compressed files consisting of lists of WARC files. These WARC files hold meta information such as the requested datetime, content type, or content size, followed by the archived content.

2.3 Client-Side Web Security

Web browsers implement many security mechanisms to mitigate a wide range of threats. The baseline defense mechanism of web browsers is the Same Origin Policy (SOP), which establishes a security perimeter at the *origin* boundary. Origin is defined as a combination of protocol, hostname, and port. Data owned by an origin is isolated from read and write accesses by scripts running in a different origin, e.g., a script running at https://foo.com/bar cannot access the DOM of a page at https://baz.com. Note that when a page loads a script through a `<script>` tag, the script inherits the page's origin and thus acquires its privileges, i.e., remote script inclusions should be performed carefully.

Besides SOP, HTTP security headers play a prominent role in web application security. These headers are set in HTTP responses with security policies to be enforced by the web browser. In this paper, we focus on five security headers, which are among the most popular and newest in the wild and received significant attention from the research community.

2.3.1 X-Frame-Options (XFO). The XFO header allows a web page to restrict the set of pages authorized to load it within an iframe. This is useful to prevent clickjacking or other types of frame-based attacks [8, 17]. The XFO header can be set to three different values: (*i*) `SAMEORIGIN` only allows framing on web pages sharing the same origin of the page setting the header; (*ii*) `DENY` entirely forbids framing on every web page; (*iii*) `ALLOW-FROM` *o* only allows framing on web pages hosted on the origin *o*. The last option is now deprecated and unsupported by modern web browsers.

2.3.2 Content Security Policy (CSP). The CSP header allows a web page to enforce declarative security policies, addressing a range of different threats [36, 39]. In particular, existing literature identifies three key use cases for CSP: (*i*) *XSS mitigation*: CSP allows the specification of a set of allowed origins for remote script inclusions and can block a page's ability to run inline scripts, inline event handlers, and string-to-code transformation functions like eval, which are the most common XSS vectors; (*ii*) *Framing control*: the frame-ancestors directive of CSP is intended as a modern replacement of XFO since it allows the specification of a set of origins which are allowed to frame the page; (*iii*) *TLS enforcement*: CSP supports the `upgrade-insecure-requests` and `block-all-mixed-content` directives to forbid plain HTTP requests and enforce the use of HTTPS.

2.3.3 HTTP Strict Transport Security (HSTS). HSTS enforces communication towards a web host to happen exclusively via the HTTPS protocol: all requests usually performed via HTTP are automatically upgraded to HTTPS [16, 22]. HSTS uses the `max-age` directive to express the duration of protection in seconds, i.e., for how long the web browser should perform the HTTP to HTTPS upgrade. The protection can be extended to all subdomains of the activating host using the `includeSubDomains` option. For example, if the homepage of https://foo.com activates HSTS with `max-age` equal to 31536000 and the `includeSubDomains` option, any request towards a subdomain of foo.com will be automatically upgraded to HTTPS for one year (31536000 seconds). To prevent attacks against the very first response before observing the HSTS header, site operators can ask for inclusion in the HSTS preload list,[3] which major web browsers use to determine which hosts they should automatically protect according to the HSTS discipline.

2.3.4 Cross-Origin Policies. Attacks like XS-Leaks frequently rely on the attacker having a handle to the victim application's window object [21, 35]. This can be done, e.g., by using the `window.open` functionality in browsers and using the return value for that call. To protect a site from this, operators can nowadays specify the Cross-Origin-Opener-Policy (COOP) header [27]. This enables fine-grained control of whether the opener can retain a handle. It can be `unsafe-none`, i.e., any opener can keep the handle. Alternatively, it can be set to `same-origin` or `same-origin-allow-popus`, allowing the opener to retain the handle if it shares the opened document's origin. In addition to COOP, sites can also ensure inclusion behavior for other types of content, e.g., images. For that, they can set the Cross-Origin-Embedder-Policy (COEP) header [26] to either `require-corp` (requiring external resources to set the CORP header), `credentialless` (ensuring that cross-origin resources are requested without credentials, i.e., cookies) or `unsafe-none` (default behavior in COEP's absence, allows any inclusion). The Cross-Origin-Resource-Policy (CORP) [28], in turn, allows an operator to disallow access from other origins unless requests are CORS-enabled. CORP can be set to `cross-origin` (allowing any page to include the resource), `same-origin` (only same-origin pages can embed the file), or `same-site`.

2.3.5 Referrer-Policy. Normally, a browser sends along the so-called `Referer` (sic) header in outgoing requests. Referrer-Policy (RP) [31] indicates the URL of the page that caused the request, e.g., when clicking a link, it includes the page from which the user came. Similarly, when including subresources such as images or scripts, browsers also provide the current document's URL. This has obvious privacy implications, in particular in cases where session management is implemented through tokens in the URL. To overcome this privacy threat, browsers have long since implemented the more privacy-friendly `Origin` header [29]. Rather than the full URL, this header only contains the document's origin. With RP, a site operator can control if the Referer header is sent along and what information it shall contain. The default value, `unsafe-url` sends the full URL in all outgoing requests. In contrast, `origin` only provides the origin, whereas `no-referrer` strips the header entirely. On top, site operators can also control cases for protocol downgrades and cross-origin requests: `no-referrer-when-downgrade` sends the full URL if the resource's protocol is at least as secure the current URL (e.g., HTTPS to HTTPS); `origin-when-cross-origin` sends the origin only if

[3] https://hstspreload.org/

Archive	API Endpoint	Hits	Fresh Hits
End of Term Web Archive	http://eot.us.archive.org/eot/[DATE]/[URL]	4,061	3,713
Internet Archive	https://web.archive.org/web/[DATE]/[URL]	4,061	3,713
Archive-It	http://wayback.archive-it.org/all/[DATE]/[URL]	3,531	2,786
Arquivo.pt	https://arquivo.pt/wayback/[DATE]mp_/[URL]	3,412	338
Library of Congress	http://webarchive.loc.gov/all/[DATE]/[URL]	2,496	0
Croatian Web Archive	https://haw.nsk.hr/wayback/[DATE]/[URL]	2,064	202
Stanford Web Archive	https://swap.stanford.edu/[DATE]mp_/[URL]	2,020	96
Archive-It (10702)	http://wayback.archive-it.org/10702/[DATE]/[URL]	1,948	0
Icelandic Web Archive	http://wayback.vefsafn.is/wayback/[DATE]/[URL]	1,846	986
York University Digital Library	https://digital.library.yorku.ca/wayback/[DATE]/[URL]	379	216
Bibliotheksverbund Bayern	https://langzeitarchivierung.bib-bvb.de/wayback/[DATE]/[URL]	50	1
Bibliotheque et Archives nationales du Quebec	https://waext.banq.qc.ca/wayback/[DATE]/[URL]	42	0
Museum of the Czech Web	https://wayback.webarchiv.cz/wayback/[DATE]/[URL]	1	1

Table 1: Working endpoints of the different Memento-based web archives, with their corresponding number of hits (for 2022-07-15). The shaded rows represent archives which are further evaluated in this work.

a cross-origin request is made, but the full URL to same-origin resources; `same-origin` sends the full URL to same-origin resources, but nothing to cross-origin resources; `strict-origin` only sends the origin if the protocol remains the same (and nothing in case of protocol downgrade); and `strict-origin-when-cross-origin` sends the full URL to same-origin resources, the origin to equal-protocol cross-origin URLS, and nothing in less secure protocols.

2.3.6 Permissions-Policy. With Permissions-Policy (PP) [30], a site can have fine-grained control over the APIs accessible by JavaScript, both for first- and, more importantly, third-party resources. This allows site operators to constrain the usage of sensitive features such as geo location, or camera usage. For example, by configuring the PP header as `geolocation=(self "trusted.com")`, the site's own origin resources, as well as those in frames coming from trusted.com are allowed to access the device's geographical location. Note that the policy only applies to documents, i.e., if the first party includes a script from untrusted.com, this is running in the context of the first party. That is, it has the same capabilities as the first party (in this case, that script can access the geo location).

3 ASSESSMENT OF EXISTING WEB ARCHIVES

In this section, we measure the effectiveness of different web archives with respect to the quantity and freshness of their archival data.

3.1 Archive Selection

Besides Common Crawl, we consider a public list[4] of web archives supporting the Memento protocol, provided as part of the Memento Time Travel project. We parse the list to collect a set of 64 API endpoints from 38 web archives and we perform a preliminary experiment to identify a set of archives for our study. In particular, we contact all the endpoints to access archival data for the homepage of the top 5,000 domains in the Tranco list [34] downloaded on February 25th, 2022 (ID Z2QWG). We ask for data archived on July 15th, 2022, with a connection timeout of 40 seconds. Note that, since web archives may provide both older and newer content than the requested date in their responses, we are not penalizing archives that did not crawl the Web around July 2022.

At the end of our experiment, we associate to each endpoint a corresponding number of *hits*, i.e., the number of domains for which we successfully received a response from the archive; the more hits we get, the more the archive provides coverage of popular domains. However, this information alone is moot because archival data are not necessarily fresh enough to support useful conclusions. We thus recompute the number of hits for the different archives after setting a *maximum temporal threshold* of six weeks on the collected responses, i.e., if the temporal distance between the requested date and the archival date (available in the `Memento-Datetime` header) exceeds six weeks, we disregard the archival data. We use the term *fresh hits* to refer to hits within the temporal threshold. Note that, since archives may return responses archived on a later date than the requested one, the six weeks threshold actually enforces a rather large window of twelve weeks (±6 weeks). We argue that any archival data falling out of this window is useless for realistic web measurements as it would be too far away from the requested date.

We report all endpoints which had at least one hit in Table 1, along with their number of hits and fresh hits. The table is sorted by the decreasing number of hits and supports several interesting observations. First, although we contacted 38 endpoints, only 13 endpoints had at least one hit, i.e., roughly two-thirds of the tested endpoints are likely outdated, leading to timeouts or other types of failures for all requests sent. As for the remaining endpoints, the four worst-performing ones are operated by libraries and museums, hence they likely index just a small number of resources of interest, i.e., at most 379 domains, amounting to around 8% of the Tranco top 5,000. These archives are certainly not amenable for general web measurements. As for the other nine endpoints, we exclude the End of Term Web Archive because it turned out to be a mirror of the more popular IA and a second endpoint of Archive-It, which indexes just a subset of its resources. The seven archives we consider for further evaluation are shaded in gray in the table. Note that all the shaded archives show an appropriate performance in terms of the number of hits, i.e., they archive data from a significant amount of domains, but some provide just a low number of fresh hits. The reason we do not filter them more aggressively is that even archives with very few fresh hits in 2022 may still be useful for historical measurements performed in previous years.

3.2 Coverage and Freshness of Archival Data

Our analysis covers a six-year timeframe from January 15th, 2016, to July 15th, 2022, and is based on periodic archival snapshots of the Tranco top 5,000, taken every twelve weeks. For this experiment, we store in our dataset not just responses with status code 200,

[4] http://labs.mementoweb.org/aggregator_config/archivelist.xml

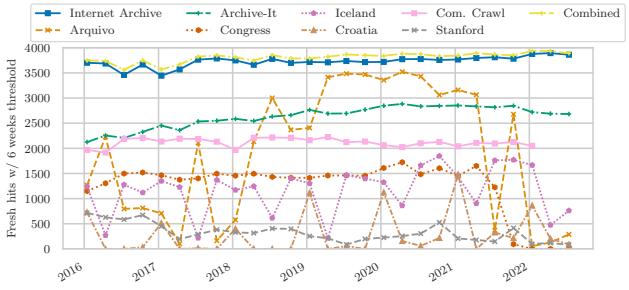


Figure 1: Number of fresh hits for different archives

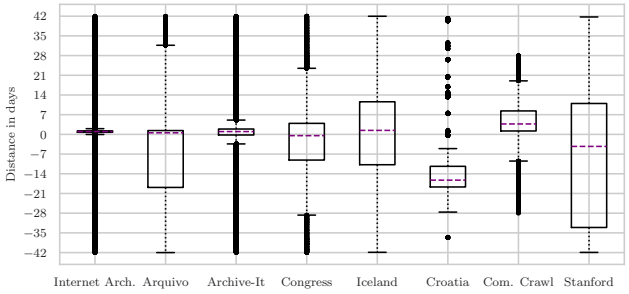


Figure 2: Distribution of the observed gaps between requested date and archival date for different archives

but also responses with status code 404, which are provided when an archive tried to crawl a web page but could not find it; this is valuable information when analyzing potential coverage of archives. Responses with a different status code turned out to be negligible in number and were ignored for simplicity. In addition, we discarded responses lacking the `Memento-Datetime` header because we have no information about their freshness.

For each archive and snapshot, we count the number of fresh hits, i.e., hits for which the archival date is within ±6 weeks from the requested date. Note that the chosen window ensures that all the content fetched from the archives was archived between two consecutive snapshots (twelve weeks away) while avoiding overlaps between snapshots. This is important for meaningful measurements, otherwise a snapshot for June 2020 might, e.g., retrieve data archived in December 2019. Figure 1 shows the number of fresh hits for different archives over time. The figure shows that the IA performs much better than all the other archives across all the temporal snapshots that we considered. To further highlight the superiority of the IA, the figure shows in the "Combined" line the number of fresh hits that we would achieve by combining the IA with all the other archives: as we can see, the advantage enabled by the introduction of additional archives is negligible. Finally, the figure also shows that the performance of archives is not always invariant over time and some archives show significantly worse performance in 2022 than in 2016. This is interesting because one might expect that some of the popular domains in the recent Tranco list that we use were not so popular in 2016 and hence were not crawled by the archives. As it turns out, however, some archives just slowed down archiving over time or even stopped doing that entirely, e.g., the Library of Congress does not seem to crawl fresh data anymore. For Common Crawl, the July 2022 snapshot was still not added at the time of our data collection, leading to its line stopping one snapshot earlier.

To better investigate the freshness of archival data, we report in Figure 2 a box plot of the temporal distances between the requested dates and the corresponding archival dates observed for fresh hits in our dataset. We consider all snapshots which were available for any of the requested dates. The figure shows that the IA could serve fresh archival data for most of our requests, because its distribution is strongly skewed towards 0, i.e., the requested date and the archival date coincide in most cases, while the other archives suffer from larger fluctuations in general. A seeming exception is Archive-It, which also has a high density around the requested date; this archive covers fewer domains overall (see also Figure 1), but for those covered, it provides a high freshness.

3.3 Impact of Domain Popularity

Our previous experiment just provides a partial picture of the state of archives because it is based just on the top 5,000 domains of Tranco. To mitigate this bias, we randomly sample 5,000 domains from Tranco (which includes one million domains) and we create a new list of domains with varying levels of popularity. We use stratified random sampling for this task, i.e., we randomly sample 500 domains from each of the ten buckets of 100k domains from the full Tranco list. We then collect two snapshots for the top 5,000 domains of Tranco and the random list of 5,000 domains, one at the beginning (January 15, 2016) and one at the end (July 15, 2022) of our time window. The reason why we consider two snapshots is because we showed that the performance of archives is not invariant over time. We finally compute the number of hits, the number of fresh hits, and the average temporal distance between the requested date and the archival date. As it turns out, the IA is the only archive that is powerful enough to provide reasonable performance on the random set of domains sampled from Tranco because it could provide fresh data for around 2,700 domains in both snapshots (roughly 55%); as for the other archives, the best result was 341 domains (7%). Note that we exclude Common Crawl here given that it does not provide a feasible and cost-efficient way to query for *all* available snapshots, but rather those within a small time window. Hence, it could only ever produce fresh hits.

Most security papers conduct analyses on the top 10k or top 100k [8, 33, 36]. Notably, however, a few studies also focus on the entirety of the top 1M domains [23, 43]. Hence, we investigate the utility of the analyzed archives for the long tail of domains. For that, we analyze the hit rate per sampled 100k bucket in Figure 3. The figure shows that the IA is again superior to all its competitors for all buckets. Note that, although Archive-It and Arquivo.pt apparently show good performance on the top 100k domains, their performance drops when focusing on fresh hits. The only archive able to provide fresh hits for a significant number of domains across the Tranco top 1M is again the IA.

Figure 3: Hit rates across popularity buckets

4 HISTORICAL SECURITY MEASUREMENTS

We now focus on the correct use of the IA for *historical* security measurements, as performed by prior work which measured the evolution of different web security aspects over time [33, 36, 41]. We just focus on the IA because we showed that it is unquestionably superior to its competitors in terms of both coverage of domains and content freshness.

4.1 Methodology

We investigate the quality of the data available in the IA to assess the correctness of historical web security measurements performed on top of them. We are particularly concerned about the bias that might be introduced by the over-reliance on this single data source, since prior research showed that web measurements can be influenced by the choice of a specific vantage point [13, 37]. Our first idea to investigate this was the following: for each snapshot in our dataset, compare the data from the IA against the temporally closest data returned by an independent data source, i.e., any other web archive considered in our prior evaluation. Unfortunately, this simple idea does not work, because the IA is so much better than its competitors that it is impossible to find fresh (±6 weeks from the requested date) independent data for comparison in the vast majority of cases.

We thus leverage the observation that the IA actually aggregates information from multiple data sources (including some of the archives in Table 1). For each stored response in the IA, the corresponding source filename is given in the `x-archive-src` header. A source is a collection of multiple responses provided by the same contributor. Feeding this filename to the Metadata API (https://archive.org/metadata/<source>) shows more information about the source, such as the contributor or the crawler name. Since the IA aggregates data from multiple contributors, we can cross-compare its data to reason about the bias coming from the use of specific contributors. In particular, while we cannot directly request and cross-compare data from various archives with the IA as we originally planned, we can reason on how different contributors (including archives) inside the IA might affect our historical view of the Web. Concretely, for each URL u and timestamp t, we collect up to 20 additional nearby snapshots, where with "nearby" we mean within ±5 days from t; we denote with $N(u, t)$ such set of archived responses, called *neighborhood*. Since all the responses in the same neighborhood are close in time, we expect them to coincide in the vast majority of cases, thus enabling their comparison. We perform both *syntactic* comparisons, where we compare the raw content of responses up to straightforward normalization of dynamic elements (e.g., CSP nonces), and *semantic* comparisons, where we compare the security inferences drawn from the collected data. This way, we can estimate both the quality of the available data and the sensitivity of web measurements with respect to small temporal drifts - an essential property for their generalization.

Our investigation is performed on the top 5,000 domains of Tranco, using the archival data of the IA from January 2016 to July 2022, collected as described before (including all response status codes). We focus on two significant case studies: security headers and JavaScript inclusions. These case studies are representative because security headers received considerable attention in the literature [8, 22, 42] and some historical measurements even used archival data [36, 41]. As for JavaScript inclusions, a prior study [33] used the number of remote hosts used for script inclusion as valuable information to reason about JavaScript security because any script included in a web page runs within its origin and inherits the corresponding privileges, hence script inclusion from a single malicious host is enough to completely compromise security. We perform a similar analysis at the *site* level rather than at the host level because the notion of site better captures the concept of a third party [40]. We also investigate the prevalence of trackers among the sites used for script inclusion, similar as performed in prior work [25]. Note that although we borrow from prior work to design case studies worth considering, our focus is different: we are primarily interested in the result of such measurements to reason about the correctness of archival data and historical analyses performed over them. Hence, we do not dive deep into the historical implications of the results themselves, but we rather check how the use of data in the IA may affect (and have affected) such experiments to provide methodological advice.

4.2 Security Headers

We first use the IA to investigate the deployment of security headers in the wild from 2016 to 2022.

4.2.1 Syntactic Analysis. Our first experiment focuses on the syntactic differences in header values observed within the same neighborhood (after header normalization). The graphs in Figure 4 plot the average and maximum number of different header configurations in the same neighborhood observed over time. We include only neighborhoods that make use of the specific header at least once. Thus, the graphs show that the newer headers have only been appearing regularly for the last three years. They also show that the more recent headers show a less stable trend for the average differences. This can be explained by the low number of neighborhoods that use these headers (Table 2), i.e., already a few differences have high impact on the average. Nevertheless, the average number of different header values is close to 1 for all headers, i.e., responses in the same neighborhood largely agree with respect to the configuration of security headers. This is a positive finding because it means that the bias coming from the use of specific contributors of archival

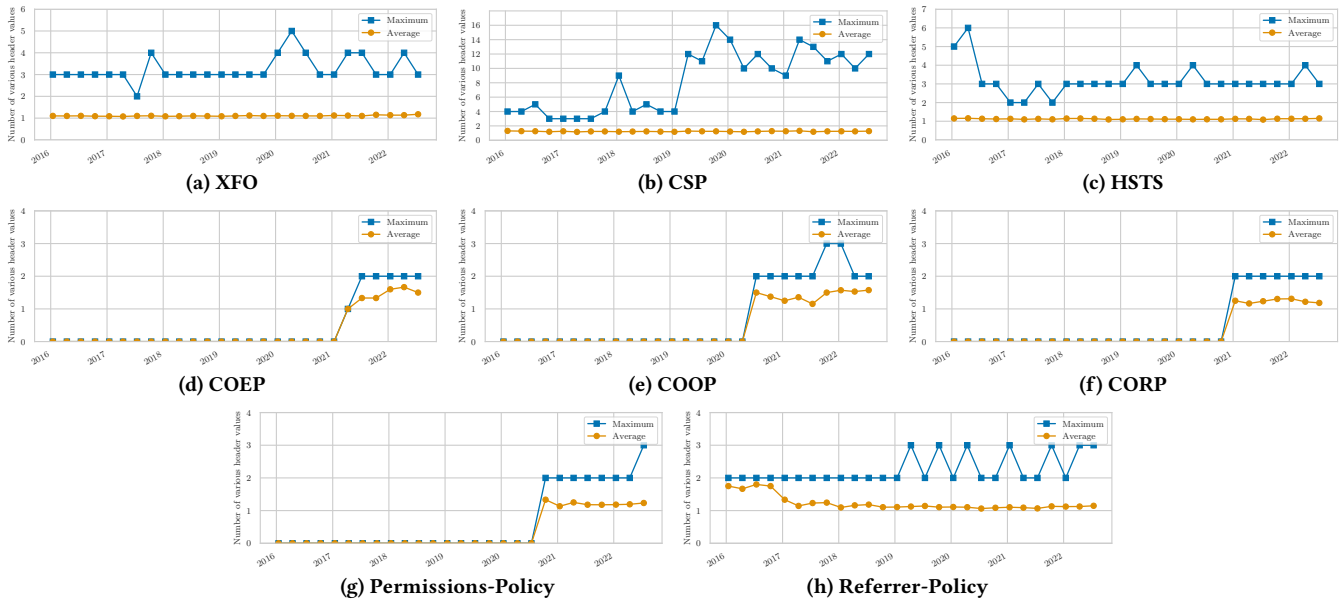


Figure 4: Number of different security header configurations within the same neighborhood

data is limited when measuring this aspect. However, the maximum number of different header values within the same neighborhood can be much higher than the average: we found neighborhoods where XFO was configured in 5 different ways, CSP was configured in 16 different ways, and HSTS was configured in 6 different ways. In contrast, the other five headers only show a maximum of three different configurations, likely due to their simpler syntax.

We now investigate which fraction of neighborhoods may observe syntactic differences for the respective security mechanisms. The second group of lines in Table 2 shows this: we find that across all headers, at least 10% of the neighborhoods have at least two configurations. The second line highlights that many sites are affected in multiple snapshots since the number of affected sites is much lower than the affected neighborhoods. Notably, though, the vast majority of syntactic differences originate from the respective header not being observed in some snapshots. This has significant implications for works that rely on syntactic changes [36], which should therefore aggregate multiple data points in the same neighborhood. The more mature headers, such as XFO, HSTS, and CSP, are more coherently deployed within the same neighborhood. Newer mechanisms like COEP and COOP show much higher variance within the same neighborhood: in Section 5, we show that this is related to the fact that these headers are not always collected correctly by the IA, due to specific behavior of Google-owned servers acting as early adopters, thus skewing results.

We discuss a few examples where we observed different header configurations within the same neighborhood:

- Some XFO headers are incorrectly populated with different numbers of occurrences of the same directive, e.g., DENY vs. DENY,DENY. This wrong configuration of the web server may lead to a lack of framing control in legacy browsers [8].
- Some CSPs make use of nonces without putting them within single quotes, as required by the standard, leading to different header values even after our normalization routine (which only operates on correctly quoted nonces). This wrong configuration may void mitigating capabilities for XSS or introduce breakage in web applications.
- Some HSTS headers make use of dynamic `max-age` values, hence different header configurations are archived upon multiple crawls. This dynamic behavior was already observed for live data in the past [37].

Again, a measurement that considers only singular requests could suffer from errors. Instead, we recommend collecting multiple responses for a single snapshot (neighborhoods) and aggregating them to improve robustness, as discussed in this section.

4.2.2 Semantic Analysis. We next abstract from purely syntactic differences between header values and focus on measuring the security guarantees offered by the headers. In particular, we consider the following definitions of security:

- XFO: we consider an XFO header to be safe if and only if it provides some restriction on framing, i.e., framing is forbidden on some origins. Concretely, we consider an XFO header safe when it is set to `SAMEORIGIN` or `DENY`. We do not consider `ALLOW-FROM` safe because it is unsupported by modern browsers and was never supported by Chrome in the first place, so its effectiveness in practice is questionable.
- CSP: since CSP has three use cases, we consider different definitions of safety: (*i*) XSS mitigation: the CSP does not suffer from trivial bypasses, e.g., the CSP does not allow script inclusion from any web origin as required by the definition of safe CSP in [9]; (*ii*) Framing control: similar to XFO, we require the policy to provide some restriction on framing, i.e., stop framing on some origins; (*iii*) TLS enforcement: we require

	XFO	CSP	HSTS	COEP	COOP	CORP	PP	RP
Neighborhoods setting header at least once	38,417	14,463	31,090	26	157	163	1,451	6,780
- Sites setting header at least once	2,646	1,399	2,464	9	48	47	437	875
Neighborhoods with syntactically different headers	3,870 (10.1%)	2,587 (17.9%)	3,486 (11.2%)	12 (46.2%)	72 (45.9%)	36 (22.1%)	272 (18.7%)	704 (10.4%)
- Affected sites	1,133 (42.1%)	716 (51.2%)	1,044 (42.4%)	6 (66.7%)	35 (72.9%)	18 (38.3%)	168 (38.4%)	313 (35.8%)
- only because of missing header	*963 (36.4%)*	*398 (28.4%)*	*832 (33.8%)*	*6 (66.7%)*	*29 (60.4%)*	*16 (34.0%)*	*162 (37.1%)*	*275 (31.4%)*
Neighborhoods with semantically different headers	3,555 (9.3%)	1,210 (8.4%)	522 (1.7%)	9 (34.6%)	3 (1.9%)	16 (9.8%)	-	505 (7.4%)
- Affected sites	1,077 (40.7%)	377 (26.9%)	210 (8.5%)	5 (55.6%)	2 (4.2%)	4 (8.5%)	-	223 (25.5%)
- only because of missing header	*938 (35.4%)*	*250 (17.9%)*	*141 (5.7%)*	*5 (55.6%)*	*2 (4.2%)*	*3 (6.4%)*	-	*193 (22.1%)*

Table 2: Neighborhoods affected by different header configurations

the policy to use either the `upgrade-insecure-requests` or `block-all-mixed-content` directives, which have the same effect of forbidding plain HTTP communication.
- HSTS: we consider an HSTS header to be safe if and only if it satisfies the necessary conditions required for inclusion in the HSTS preload list, i.e., the `max-age` directive is set to a duration of at least one year and the `includeSubDomains` option is activated.
- COOP, COEP, and CORP: since COOP is meant to restrict access from cross-origin openers, we argue that `same-origin` and `same-origin-allow-popups` are deemed safe, while `unsafe-none` is deemed unsafe. Similarly, COEP and CORP are unsafe when set to `unsafe-none` and to `cross-origin`.
- For RP, the primary threat model is to not leak sensitive URL parameters to third parties. Therefore, `unsafe-url` and `no-referrer-when-downgrade` are deemed unsafe, whereas all other values are deemed safe.

Finally, for PP, there is not clear binary choice between safe and unsafe configurations. Therefore, rather than attempt to come up with an arbitrary definition, we omit it from further analysis.

Based on these definitions of safety, responses in $N(u,t)$ can be partitioned in a safe subset $N^+(u,t)$ and an unsafe subset $N^-(u,t)$, and we can estimate how much the responses in $N(u,t)$ agree with respect to safety by computing the *Gini impurity* score as follows:

$$Gini(N(u,t)) = 1 - \left(\frac{|N^+(u,t)|}{|N(u,t)|}\right)^2 - \left(\frac{|N^-(u,t)|}{|N(u,t)|}\right)^2$$

The value of Gini impurity ranges from 0, when all the responses in $N(u,t)$ belong to the same class, to 0.5, when responses are equally split between the two classes. We first estimate the number of cases where the Gini impurity is greater than 0: these cases can compromise the correctness of security measurements because different responses in the same neighborhood enable different security inferences. The results for each security mechanism are shown in the lower three rows of Table 2. Across all neighborhoods, we find that between 1.7% (for HSTS) and 34.6% (for COEP[5]) are impure, i.e., they contain observations which have differing security implications. We note, compared to the fraction of impure neighborhoods, that a higher percentage of sites that deployed the respective mechanism at least once are affected. However, the results also show that for the vast majority of sites, this is because they were simply missing the header in at least one snapshot. Considering how many sites are affected (in a potential longitudinal measurement, e.g., [36]), we find that relying on a single snapshot for each site threatens the validity of measurement results. We further study the reasons for these cases in the following section.

[5]COEP likely is an outlier due to its low prevalence.

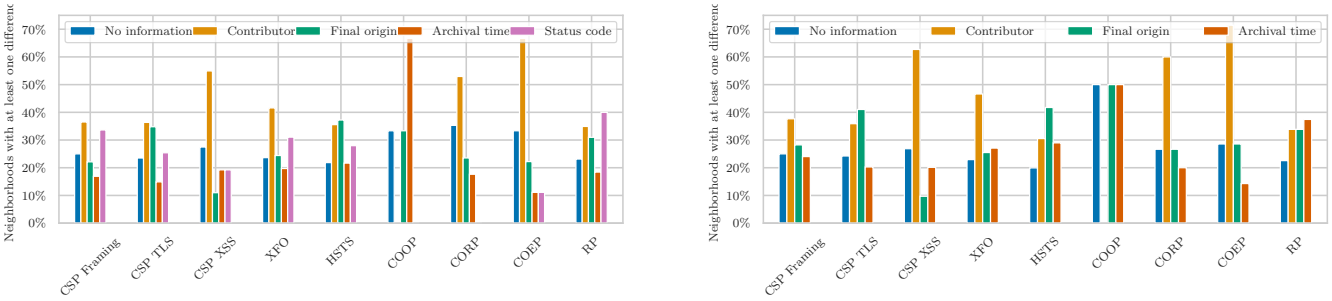
4.2.3 Attribution of Differences. To investigate why such significant differences occur between neighbors, we identify the response feature f which contributes the most to impurity by computing the *information gain* (reduction of Gini impurity) obtained by splitting $N(u,t)$ in two subsets (i.e., safe and unsafe) based on the value of f. We consider the following features of the responses as potentially important for Gini impurity:

(1) *Contributor*: different contributors of archival data might influence the response content due to its vantage point and generic reasons, e.g., bugs in a crawling script;
(2) *Status code*: for example, error pages might enforce different security policies than normal web pages;
(3) *Final origin*: different origins (after redirects) may point to different applications, enforcing different security policies;
(4) *Archival time*: changes to security policies might occur even in our short time window of ten days.

Note that this analysis does not come without caveats. First, multiple features may contribute to explaining the same security difference: for example, different contributors may be redirected to different origins due to their geolocation. We address this issue by considering all these features as equally important when they lead to the same information gain. Moreover, it is also possible that none of the considered features leads to a clear information gain: when none of the features leads to an information gain of at least 0.1, we do not perform any attribution.

Figure 5a shows that the *contributor* is the most important feature to explain security differences within neighborhoods for almost all the headers, followed by the *status code*. The *final origin* and the *archival time* play a less significant role in the attribution of security differences (except for HSTS and CSP-TLS). We then observe that:

(1) The impact of the contributor is significant, yet it can be mitigated by restricting the set of trusted contributors within the web measurement. In our dataset, we observe that many responses (24%) do not bear any information about their contributor, which is set to NULL. Remarkably, the NULL contributor is the one that contributes to explaining most of the contributor-related security differences, i.e., it often disagrees with the other contributors. Web measurements can be made more robust by filtering out the NULL contributor: this does not significantly affect the feasibility of historical analyses, as less than 3% of the considered neighborhoods include only data provided by this contributor.



(a) All status codes **(b) Just 2XX status codes**

Figure 5: Feature importance for the attribution of security differences within neighborhoods (that show at least one difference)

(2) The impact of the status code is significant, yet it can be easily mitigated by focusing just on responses with a specific status code, e.g., in the 2XX class. Figure 5b shows a variant of Figure 5a where we only keep responses with 2XX status code. As we can see, the picture is not very different, and the contributor is still the most important feature to attribute security differences even when filtering out error pages.
(3) The impact of the final origin is often limited, yet it should be zeroed out in correct web measurements. This particularly holds true for measurements related to TLS enforcement (CSP-TLS and HSTS), for which the results show it as having high explanation power. This is to be expected, given that, e.g., HSTS headers have no meaning when sent through HTTP. While different recipes may be used to select the final origin to analyze, e.g., its inclusion in Tranco, it is important to ensure the chosen final origin is always the same across the web measurement because the IA aggregates responses for different final origins under the same URL.
(4) The impact of the archival time is often limited in our time window of ten days. We can nevertheless further limit the impact of archival time by making neighborhoods smaller, e.g., using a time window of just five days.

4.3 JavaScript Inclusions

In this section, we use the IA to investigate the state of JavaScript inclusions. We focus on inclusions from remote sites, given their security implications according to SOP. To this end, we parse the static HTML content available in our dataset and collect every URL that is loaded in any script element. Additionally, we use this information to estimate the prevalence of web tracking in the wild (as done in [25]). In particular, we tag a site as a tracker if it belongs to the Disconnect[6] or EasyPrivacy[7] filter lists to estimate how often web trackers are included in popular websites.

In our first experiment, we measure how the average number of remote sites used for script inclusion changed over time. This is shown in Figure 6. The plot has three lines: the first one uses the responses which are temporally closest to the analyzed date, while the other two lines show the union of sites detected in each neighborhood (i.e., a site appeared in at least one snapshot within the neighborhood), and the intersection of detected sites (i.e., a site appeared in all snapshots within the neighborhood). This way, we can estimate upper and lower bounds for the sets of sites used for script inclusion, thus estimating the bias coming from the use of the specific data point corresponding to the temporally closest date. All three lines in the plot show that the number of remote sites used for script inclusion did not change a lot over the years. However, although the lines agree on the general trend, the actual number of detected sites may differ. At most, we detect 8% fewer sites when taking the temporally closest response instead of the union of sites, while we detect 12% fewer sites when taking the intersection of sites instead of the temporally closest response. We thus recommend using multiple data points within a neighborhood to make web measurements more robust, e.g., by reporting lower and upper bounds, as discussed in our experiment.

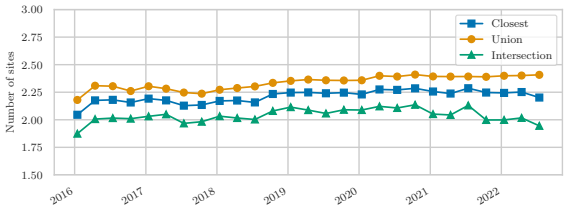
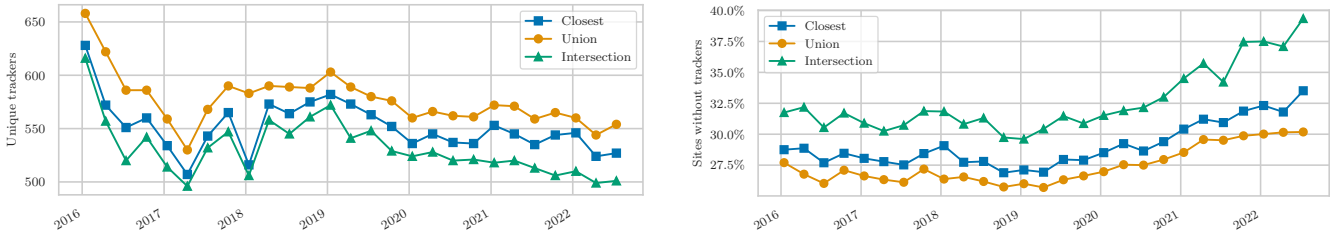


Figure 6: Script inclusion statistics for third-party sites

We now zoom in on the prevalence of trackers over time. In particular, we plot how many of the trackers of Disconnect and EasyPrivacy were found at least once in our dataset at different points in time. This is shown in Figure 7a. The plot uses the same three types of observations (temporally closest, union, and intersection) as before. The figure shows that the number of unique trackers observed over the years appears to be decreasing. Figure 7b additionally shows that the number of websites that do not include any tracker has been slowly increasing over the years. In particular, it shows the percentage of websites that do not include any tracker *directly* peaked at around 40% in 2022. This trend seems counter-intuitive at first glance. However, prior work showed that as of 2021, more than 60% of websites dynamically load third parties [40]. With this in mind, we conjecture that our original focus on static HTML content was introducing a bias in our findings.

[6]https://disconnect.me/trackerprotection
[7]https://easylist.to



(a) Number of unique trackers per date

(b) Percentage of websites without any tracker per date

Figure 7: Prevalence of web tracking in the wild based on archival data

We thus implemented a dynamic approach and crawled the archived pages from our previous experiments using Chromium via Playwright. Due to the nature of modern websites, the dynamic analysis significantly increased the number of requests sent to the IA. Instead of one request per website, we now measured more than 100 requests per site on average. This increase in requests for dynamic analyses not only means more use in bandwidth but, more importantly, it also means a drastic increase in the time it takes to request all websites and a significant risk of being rate-limited by the IA. Therefore, to put less load on the IA, we added a 5s sleep after each page request and conducted our experiment just on the oldest and the newest snapshot in our experiments, i.e., January 2016 and July 2022. Despite all the care put into preventing rate-limiting, our crawler still frequently got responses with status code 429 (Too Many Requests). Therefore, we could only compare 2,026 sites. This shows that large-scale analyses of dynamic HTML content are particularly difficult to perform using the IA, which is a significant bottleneck.

Regardless of the bottleneck, we highlight the importance of dynamic approaches for some measurements (e.g., tracking behavior), as the comparison of the static and dynamic crawls in Table 3 shows. The first two rows of the table show the average number of remote script inclusions and trackers. In both rows and for both years, the crawler was able to find more trackers with the dynamic approach compared to the static approach. The other rows in the table show a similar picture for the number of unique trackers and the number of websites with trackers. Importantly, the dynamic crawl revealed that the number of websites that include some tracker is much higher than expected according to the static crawl. While both 2016 and 2022 statically show around 73% of sites with trackers, the dynamic ones reveal 95 and 92%. Counterintuitively, the number of sites with a tracker also seems to decline in the dynamic crawl of 2022 (albeit the fluctuation is similar to what was observed in [25]). To confirm the correctness of the IA, we ran another experiment, visiting both live and archived versions of an April 2023 sample of sites. For the overlapping sites which returned a 200 status code in both live and archived versions, 1571 (archived) and 1574 (live) sites contained trackers, thus confirming the accuracy of the IA.

	2016		2022	
	static	dyn.	static	dyn.
Avg. no. of trackers	1.95	6.80	1.71	7.23
Avg. no. of remote inclusions	3.41	6.63	3.51	7.04
Unique trackers	465	851	405	899
Websites with trackers (total)	1,497	1,931	1,481	1,855
Websites with trackers (perc.)	73.89%	95.31%	73.10%	91.56%

Table 3: Static vs. dynamic crawls of the IA for the set of 2,026 sites available in both 2016 and 2022

5 LIVE SECURITY MEASUREMENTS

We now investigate whether archive-based measurements can be a feasible and reproducible alternative to traditional live security measurements. We just focus on security headers here since our previous experiments showed that JavaScript inclusions need to be studied dynamically, which would put even more load on the IA.

5.1 Stability of Live Data

The IA operates by means of periodic snapshots and cannot crawl every website every day. This means that its view of the Web is potentially coarse-grained: any change taking place between two consecutive snapshots cannot be observed by the IA, i.e., security can be realistically measured through the IA only if it does not change too frequently.

To understand whether the granularity of the snapshots taken by the IA makes it a reasonable vantage point to measure security, we collect live data from the top 20,000 domains of Tranco once per day for fifteen days, and we store them in a local database. We refer to such live data collections as L_i for $1 \leq i \leq 15$ and we use the notation $L_i(d)$ to denote the response from domain d in L_i, if any. We then assess to which extent data collected from live domains are *stable* over our time window, where we say that domain d is stable in the time window if and only if for every two live data collections L_i, L_j we have that $L_i(d)$ and $L_j(d)$ are equivalent from a security perspective. We measure the stability of security headers in two ways. Syntactic stability requires equality over (normalized) header values, while semantic stability just ensures that a safe header configuration does not turn unsafe or vice-versa. Having a high number of stable domains is a necessary condition to ensure that security can be meaningfully measured using periodic snapshots like those taken by the IA.

Figure 8 shows how the *syntactic* stability rate of domains changes across the live data collections $L_1, \ldots, L_k$ for increasing values of k. We only compute this rate with respect to the number of domains making use of a given security header at least once within that

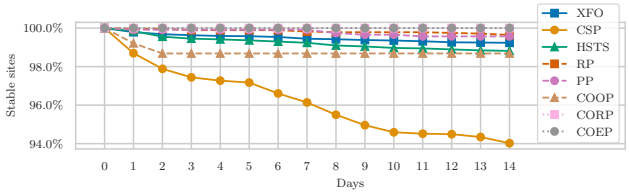


Figure 8: Syntactic stability of security headers found in live data between 22/03/2023 and 05/04/2023

Figure 9: Insertions, deletions, and updates of snapshots

month. The figure shows that syntactic stability is above 98% after fifteen days for all headers except CSP, while semantic stability (that we do not plot to improve readability) is even higher, ranging above 99%. The header facing the most changes is CSP, the most complex mechanism out of those considered in terms of syntax, however, syntactic stability is above 94% even for CSP. Given this extended period of stability for the vast majority of domains, we conclude that live security headers can realistically be analyzed by means of periodic archival snapshots. Recall from Figure 2 that the distance between the requested date and the archival date is normally less than one day for the IA; hence stability over our conservative time window of fifteen days gives strong assurance that security can be measured through archival data. Remarkably, stability does not seem to differ for mature headers such as XFO, CSP, and HSTS and newer headers such as COOP and PP. This is interesting because one might expect that less mature headers undergo more changes, for example, as the result of preliminary testing of header deployment in the wild.

5.2 Stability of Archival Data

Measurements performed on top of the IA are expected to be reproducible by design; however, reproducibility might break when the IA updates its content. On the one hand, the IA routinely gets updated as the result of its periodic Web crawling: even if data is missing for one exact date, a snapshot close after that date might be crawled. On the other hand, requesting data for a particular date (rather than an exact timestamp) might also change the archived content. For example, introducing a snapshot (e.g., from Common Crawl) closer to the requested date may lead to a different conclusion to be drawn with respect to security analyses.

To understand whether the IA can be useful for reproducible measurements, we ask for the same archival data (based on the same requested day r) for the top 20,000 domains of Tranco once per day for fifteen days, and we store them. We only consider fresh data archived within ±1 days from the requested date r, given our current focus on emulating live measurements. We refer to such live data collections as A_i for $1 \leq i \leq 15$ and we use the notation $A_i(d)$ to denote the (fresh) response from domain d in A_i, if any. We then assess to which extent archival data are *stable* over time, where we say that domain d is stable in the time window if and only if for every two data collections A_i, A_j we have that $A_i(d) = A_j(d)$. This means that three cases can violate stability:

(1) Insertions: a snapshot for a previously non-archived domain d has been archived, i.e., there exists i such that $A_i(d)$ is undefined, but $A_{i+1}(d)$ is fresh;
(2) Updates: existing archival data for a domain d are replaced by new archival data, i.e., there exists i such that both $A_i(d)$ and $A_{i+1}(d)$ are fresh, but the latter has a more recent timestamp;
(3) Deletions: archival data for a domain d become unavailable, i.e., there exists some i such that $A_i(d)$ is fresh, but $A_{i+1}(d)$ returns a response with a status code other than 200.

The latter two operations are particularly dangerous for reproducibility because they imply that the *exact* data used within a web measurement may become unavailable at a later time. Insertions are easier to deal with during experiment design, given that researchers can delay their analysis until the data has become stable, i.e., no new snapshots are added closer to the requested date.

Figure 9 shows the number of domains with at least one working (i.e., status code 200) snapshot over time (as the blue line, with the left y-axis). Note that given our threshold of ±1 day, we only find data for 9,056 domains. Here, we observe that within the first three days after the requested date, saturation is already reached since there are virtually no snapshots for domains that lacked a previous entry, i.e., no more new insertions. Notably, though, we observe three distinct spikes in updates (shown in the green bars, right-hand y-axis): 1,332 domains, which had been archived before, get a temporarily closer snapshot on day 1. Moreover, after 2 and 3 days, respectively, another 1,720 and 1,760 domains receive updates. This can be attributed to the IA consuming external sources to update its own database. After four days, where a very small number of domains still undergo updates, further updates are negligible in number and become invisible in the figure. Note that the figure also indicates deletions, yet they are always invisible and thus negligible in practice. This leads us to conclude that by consuming external archives, the IA gets fresher data which only in very rare cases leads to a previously archived result becoming unavailable, hence reproducibility is not at stake.

5.3 Live Data vs. Archival Data

Having assessed the potential effectiveness of the IA for realistic and reproducible live security analyses, we run a final experiment to establish whether archival data actually matches live data. In particular, we perform both a syntactic and a semantic comparison between archival data and live data, similar to what we performed when measuring the stability of live data. For this, we collect live data of the top 20,000 sites for April 14, 2023 and compare it with archival data for that day, collected on April 17, 2023 (taking into account our previous findings about delayed additions to the IA). We then compare the collected security headers for those URLs with a valid response from both the live crawl and the archived version. Here, we again rely on the tolerance of ±1 day from the requested date. Since prior work [37] showed that web measurements could

	Germany (9,056 sites)			USA (8,487 sites)			Australia (8,457 sites)		
	usage	**syn. diff.**	**sem. diff.**	**usage**	**syn. diff.**	**sem. diff.**	**usage**	**syn. diff.**	**sem. diff.**
X-Frame-Options	4,923	198 (4.0%)	190 (3.9%)	4,493	160 (3.6%)	153 (3.4%)	4,465	177 (4.0%)	173 (3.9%)
Content-Security-Policy	2,640	214 (8.1%)	159 (6.0%)	2,417	162 (6.7%)	118 (4.9%)	2,411	188 (7.8%)	140 (5.8%)
Strict Transport Security	5,206	378 (7.3%)	367 (7.0%)	4,766	311 (6.5%)	297 (6.2%)	4,746	334 (7.0%)	322 (6.8%)
Referrer-Policy	1,845	77 (4.2%)	60 (3.3%)	1,741	51 (2.9%)	36 (2.1%)	1742	54 (3.1%)	40 (2.3%)
Permissions-Policy	774	228 (29.5%)	228 (29.5%)	755	219 (29.0%)	219 (29.0%)	754	219 (29.0%)	219 (29.0%)
Cross-Origin-Opener-Policy	295	207 (70.2%)	205 (69.5%)	301	209 (69.4%)	209 (69.4%)	306	212 (69.3%)	212 (69.3%)
Cross-Origin-Resource-Policy	93	33 (35.5%)	25 (26.9%)	102	28 (27.5%)	20 (19.6%)	101	28 (27.7%)	20 (19.8%)
Cross-Origin-Embedder Policy	23	1 (4.3%)	0 (0.0%)	22	0 (0.0%)	0 (0.0%)	22	1 (4.5%)	0 (0.0%)
Any header	6,935	602 (8.7%)	536 (7.7%)	6412	512 (8.0%)	447 (7.0%)	6381	541 (8.5%)	484 (7.6%)

Table 4: Differences between live data and archival data

be affected by the choice of a specific vantage point, we perform three different live data collections from Germany, California, and Australia, respectively, in this experiment.

The results for the security header comparison are shown in Table 4. For each vantage point, the table shows the number of sites successfully visited in both the live and archived versions, i.e., both returned status codes 200. At first glance, we observe that relative to the usage of headers, Germany has the highest syntactic and semantic differences, closely followed by Australia. US shows the smallest number of differences. Given the IA is primarily fed from crawls in the US, this is to be expected. For the following discussion, we thus focus on the differences for the US-based crawl.

Notably, when using the IA or crawling live data, the URL initially visited does not necessarily belong to the same registered domain as the final URL. As with our previous analysis, we therefore first looked at cases in which the host of the requested URLs *after* redirects matched between live and archive. Out of the 512 cases of differences in the US analysis, 283 had a differing final origin. As expected, the numbers for Germany and Australia were even higher (344 and 316, respectively) because of geo-specific registered domains. Of the remaining 229 domains, 17 had a much older snapshot in the IA (beyond one day of tolerance). This leaves us with 212 (out of 6,412, i.e., 3.3%) domains for which no obvious explanation exists.

Looking at the table in more detail, it also highlights that more recent additions of security headers, such as Permissions-Policy or COOP, show differences of up to 70% when comparing live with archival data. To understand potential reasons for these (and all other differences), we investigated the sites affected. This revealed that the very large majority of domains deploying these newer security mechanisms are Google domains (google.tld), which selectively send the header or are based on user-agent sniffing. Based on this finding, we conduct an additional check against *all* 212 domains with unexplained header differences; making one request against the live site with the User-Agent set to `archive.org_bot` (https://archive.org/details/archive.org_bot) and one pretending to be a recent Chrome browser. This revealed that a total of 93 sites employed User-Agent sniffing, simply not returning security headers at all for the IA bot. Of the remaining 119, 34 had a different title (mostly indicating error pages in one case and the actual site in the other), and a further four had a similar title, yet the content varied significantly (most likely returned from another server with another configuration). Overall, this leaves us with 81 cases of differences (across *all* headers) that cannot be explained. Relative to the number of sites that deployed any header, this leaves us with 1.26% (81/6,412) without explanation. These findings are in line with recent work from Roth et al. [37], who showed that 127/8,174 sites (1.55%) with any security header in their study had seemingly random differences. Hence, while the IA may show differences of up to 9% compared to live analysis, researchers can a priori design their experiment to limit the impact or filter out bogus data (e.g., caused by User-Agent sniffing). This way, the IA can be used to get similar quality data to live measurements.

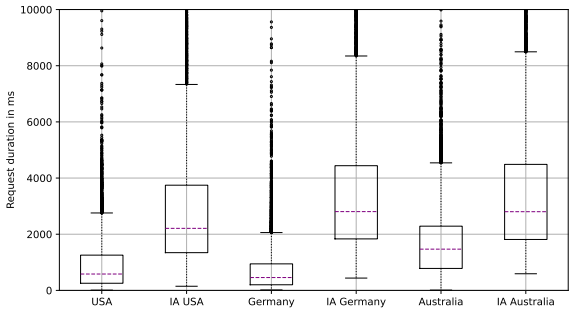


Figure 10: Duration for live and archival requests for the three vantage points Germany, USA, and Australia

5.4 Performance Evaluation

The IA can represent a bottleneck for large-scale measurements. In live measurements, requests are spread across different web servers, while in archive-based measurements, all the requests are sent to the IA. This also means that the rate-limiting policy of the IA may hinder the amount of parallelism that would be required to carry out large-scale measurements in a reasonable amount of time. Of course, testing the rate-limiting policy of the IA would be unethical because it would necessarily require sending a high number of concurrent requests to the IA, which may affect its functionality. We thus limit our investigation to *request duration*, i.e., the amount of time passed between the creation of an HTTP request and the reception of the corresponding HTTP response.

Figure 10 shows the distribution of the request time for live measurements and archive-based measurements from the three vantage points for a total of 13,107 domains that could successfully be retrieved (both live and archive) from all vantage points. Requests to the IA are fastest from the US, requiring, on average 3.3s, compared to 3.9s from Australia and 4.0s from Germany. Considering the live requests, which took, on average 0.92, 1.74, and 0.75 seconds, respectively, we note a significant slowdown for single

requests. Hence, the slowdown varies from 2.2x (Australia) to 5.3x (Germany). The seemingly lower slowdown for Australia, however, actually relates to the high latency that occurred for live requests, likely because many popular services primarily serve customers from Europe and US (and hence optimize their content distribution accordingly).

6 BEST PRACTICES FOR RESEARCHERS

Our principled investigation of web archives provides many relevant insights that future research may leverage.

6.1 Sources of Archival Data

We experimentally showed that the IA is unquestionably superior to all its competitors for web measurements. Compared to other archives, the IA: (*i*) provides higher coverage of domains in the Tranco top 5,000; (*ii*) provides fresher data on average on the Tranco top 5,000; and (*iii*) is the only archive that can realistically generalize beyond Tranco top 5,000, thus enabling large-scale studies. Moreover, we showed that combining the information in the IA with additional data sources only provides a negligible advantage in practice (Figure 1). Our analysis thus shows that using the IA as a single source of archival data is a reasonable choice for archive-based measurements.

6.2 Correct Use of Archival Data

We showed that we can mostly trust the data available in the IA. However, leveraging it for research purposes is not as straightforward as one might think. In particular, for syntactic analysis, even for mature headers, measured results might be off for up to 20%. Therefore, our analysis identified several potential pitfalls which may affect the precision of archive-based measurements, and we thus propose a few recipes to improve future studies:

(1) The IA aggregates information from multiple sources, hence its point of view of the Web might not reflect any actual vantage point. To improve the robustness of web measurements and mitigate the risk of bias, it is useful to collect neighborhoods of temporally close data rather than single data points, using different aggregation and filtering techniques as discussed in this paper. Since the majority of incorrect conclusions comes from missing headers (see Table 2), checking more snapshots is a simple way to rectify that issue.
(2) Different contributors to the IA might disagree with each other. This might come from their specific vantage point or be attributed to other issues, e.g., bugs in their data collection. Restricting the set of trusted contributors might reduce the study's variance and improve its conclusions' robustness. Concretely, we observed that the NULL contributor is the one that is most often in disagreement with other observations.
(3) The IA stores responses with different status codes, including error pages. Restricting the focus to 2XX status codes might improve the robustness of web measurements, although we note that some studies may want to also consider error pages. For example, XSS protection should also be applied to error pages to protect the web origin.
(4) The same neighborhood of archival data might include responses coming from different origins, yielding responses that are generally incomparable. This is subtler to deal with than status codes because the choice of the final origin to be preferred is not necessarily straightforward. However, it is crucial to always focus on the same origin throughout a longitudinal measurement to avoid "apples to oranges" comparisons between different points in time.
(5) The dynamic nature of script inclusions and web tracking complicates archive-based measurements because the static HTML content from the IA only provides limited insights. Unfortunately, the rate-limiting policy of the IA makes dynamic analysis via standard web browsers challenging to perform. Requesting just a subset of resources of interest, e.g., substituting images with placeholders, may help reduce the amount of traffic generated toward the IA.

6.3 Archives for Live Measurements

Our investigation showed that the IA can be used to perform security measurements that closely approximate traditional live measurements while being easily reproducible by design. On the other hand, archive-based measurements have specific limitations. In particular, our experiments showed that:

(1) The IA frequently crawls the most popular domains of Tranco, hence the temporal distance between the requested date and the archival date even drops to zero for most domains after a few days of waiting. This implies that the archival snapshots cover popular domains roughly on a daily basis and are thus frequent enough to measure security because live data exhibit just small security differences even after a window of fifteen days.
(2) When requesting archival content for a given date, the IA does not seem to undergo content changes after a stabilization window of around four days from that date. Hence, archive-based measurements performed after stabilization are largely reproducible. To further mitigate the effect of updates to the IA, researchers can share the end URLs used to collect data from the IA rather than the original request URLs.
(3) Archival data closely match live data from multiple vantage points for the vast majority of domains and a number of security headers, including XFO, CSP, and HSTS. This means that archives are almost as good as traditional crawlers for security measurements to measure security trends when following the previously mentioned best practices (Section 6.2), yet they better support reproducibility.
(4) Some domains cannot be analyzed by means of the IA as they implement browser switches, causing different responses for crawlers, which are not representative of live data. While this practice is not particularly widespread, our analysis specifically for COOP showed that by a single large entity (i.e., Google) deploying this mechanism, results could be significantly skewed. This can be accounted for by testing live domains for such behavior. Such domains can then simply be skipped to avoid biases.
(5) Archive-based measurements are slower than traditional live measurements. The average overhead of archive-based measurements over live measurements ranges from 2.2x to

5.3x, depending on the vantage point. Also, the rate-limiting policy of the IA may impact the amount of parallelism in the crawling phase. Still, considering mid-to-large scale studies, the increased runtime is offset by the ability to have reproducible measurement results. Notably, though, given the rate-limiting enforced by the IA, it is not feasible to replace full-blown dynamic live crawls given the significant overhead caused by loading vast amounts of resources beyond the static HTML content.

7 RELATED WORK

We relate our work to historical web measurements, reproducibility of web measurements, and other work on web archives.

Historical Web Measurements. We are not the first to use web archives for historical web security measurements, however, prior work is largely based on the IA alone without any strong motivation and provides just limited insights on how to use it correctly for security analyses. Our paper instead considers a plethora of different web archives, compares their effectiveness, and presents potential pitfalls of archive-based security measurements. Our systematic analysis identified useful methodological guidelines, summarized in Section 6. In a sense, our analysis extends to the web security setting prior work on web privacy by Lerner et al., leveraging the IA to study the evolution of web tracking from 1996 to 2016 [25]. Their work already documented challenges in using the IA, such as the bubble escapes that return responses from the live web. However, there are significant differences between the two studies: (*i*) we evaluate an extensive list of web archives and not just the IA, highlighting its advantage over all its competitors; (*ii*) we focus on web security rather than web privacy, hence we investigate different aspects such as security headers; (*iii*) we identify new pitfalls in the use of the IA coming from the existence of different contributors therein, which we tackle through the introduction of neighborhoods; finally, (*iv*) we investigate the use of the IA as a reproducible alternative to traditional live security measurements and set out best practices for future works in this space.

The first security measurement based on archival data we are aware of is due to Nikiforakis et al., who used the IA for a large-scale evaluation of remote JavaScript inclusion from 2000 to 2010 [33]. Stock et al. used the IA to analyze the history of client-side security from 1997 to 2016, discussing the evolution of JavaScript code and the changes in the adoption of security headers [41]. Roth et al., instead, used the IA to study the evolution of the CSP deployment from 2012 to 2018 [36]. An interesting observation in their study is that the data fetched from the IA largely matches those stored by Common Crawl, however, their analysis of the correctness of the IA is limited to simple syntactic differences in the CSP headers.

Other recent security-unrelated historical web measurements include an analysis of the evolution of privacy policies over time [5], an empirical study of the changes of the structural properties of web pages [10] and a perspective on how the Web evolved over the last 25 years in terms of website popularity and complexity [1].

Reproducible Web Measurements. To the best of our knowledge, we are the first to propose the use of web archives as a more reproducible alternative to traditional live measurements. Reproducibility of web measurements is a hot topic nowadays. Ahmad et al. first observed that the choice of a specific web crawler might have a significant impact on web measurements and the inferences drawn from them [2]. Jueckstock et al. similarly showed that specific browser configurations and network access methods might affect web security and privacy measurements [18]. These works pointed out that web measurements are generally hard to reproduce, which motivated additional work by Demir et al. [13]. Their research further confirmed the problem and identified general guidelines to improve the reproducibility of web measurements.

All these papers made the important contribution of identifying relevant shortcomings in published research, however, they only provided limited insights in terms of actionable steps to improve on this limitation. In particular, while guidelines are great for supporting the reproducibility of the measurement methodology, they do not suffice to counter the inherent irreproducibility coming from the ephemeral and erratic nature of the Web. Our focus on web archives, instead, provides a countermeasure against such significant issues. Reconciling our work with prior research, e.g., by studying whether archival data is representative of data that humans would be actually exposed to, is an important research direction for future work.

Other Work on Web Archives. Web archives received attention from the research community, in particular when it comes to the quality and quantity of archival data. Murphy et al. presented the first validity study of the IA with respect to archived web pages, website age, and website updates in 2008 [32]. More recent studies took a more in-depth look into the accuracy of archived copies of popular web pages, detecting that only one out of five pages actually existed as presented [3], and proposed solutions to mitigate this problem [6, 20]. Remarkably, [3] first suggested the combination of multiple archives to improve the completeness of web archiving, although at the expense of temporal coherence. Other work also proposed the use of aggregate information from multiple web archives: AlSum et al. showed that the top three web archives in their experiments could provide coverage comparable to their full set of twelve archives [4]. None of these studies, however, studied the use of web archives for security measurements.

Orthogonally, Soska and Christin used archival data to train classifiers for detecting vulnerable websites before they turn malicious [38], while Lerner et al. identified several flaws in how the IA stored archived data, which allowed attackers to compromise the users' view of an archived web page [24]. Though security-related, these are not related to web security measurements.

8 CONCLUSION

We investigated the use of web archives to carry out reproducible security measurements. After showing the superiority of the IA over other alternatives, we analyzed its correct use to perform historical measurements, and we showed the potential of archive-based measurements as a reproducible alternative to traditional live measurements. Our analysis identified insights and best practices for future archive-based security measurements, providing useful guidance to other web security researchers.

In future work, we plan to further investigate whether archival data enable other types of reproducible web security measurements.

In particular, we showed that dynamic analyses are difficult to perform on top of the IA, hence we would like to design and implement a reliable analysis platform for archival data. Moreover, we plan to assess whether archival data is representative of real data that users would have access to and thus allow conclusions that are more meaningful in practice [13].

Acknowledgements. We thank the anonymous reviewers for their constructive feedback, which has greatly contributed to the improvement of this paper. This research was partially supported by DAIS - Università Ca' Foscari Venezia within the IRIDE program and by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU. Furthermore, this work was conducted in the scope of a dissertation at the Saarbrücken Graduate School of Computer Science.

REFERENCES

[1] Vibhor Agarwal and Nishanth Sastry. 2022. "Way back then": A Data-driven View of 25+ years of Web Evolution. In *The Web Conference*. https://doi.org/10.1145/3485447.3512283

[2] Syed Suleman Ahmad, Muhammad Daniyal Dar, Muhammad Fareed Zaffar, Narseo Vallina-Rodriguez, and Rishab Nithyanand. 2020. Apophanies or Epiphanies? How Crawlers Impact Our Understanding of the Web. In *The Web Conference*. https://doi.org/10.1145/3366423.3380113

[3] Scott G. Ainsworth, Michael L. Nelson, and Herbert Van de Sompel. 2015. Only One Out of Five Archived Web Pages Existed as Presented. In *ACM HT*. https://doi.org/10.1145/2700171.2791044

[4] Ahmed Alsum, Michele C. Weigle, Michael L. Nelson, and Herbert Van de Sompel. 2013. Profiling Web Archive Coverage for Top-Level Domain and Content Language. In *International Conference on Theory and Practice of Digital Libraries*. https://doi.org/10.1007/978-3-642-40501-3_7

[5] Ryan Amos, Gunes Acar, Elena Lucherini, Mihir Kshirsagar, Arvind Narayanan, and Jonathan R. Mayer. 2021. Privacy Policies over Time: Curation and Analysis of a Million-Document Dataset. In *The Web Conference*. https://doi.org/10.1145/3442381.3450048

[6] Justin F. Brunelle, Mat Kelly, Hany SalahEldeen, Michele C. Weigle, and Michael L. Nelson. 2015. Not all mementos are created equal: measuring the impact of missing resources. *Int. J. Digit. Libr.* 16, 3-4 (2015). https://doi.org/10.1007/s00799-015-0150-6

[7] Michele Bugliesi, Stefano Calzavara, Riccardo Focardi, and Wilayat Khan. 2015. CookiExt: Patching the browser against session hijacking attacks. *J. Comput. Secur.* 23, 4 (2015), 509–537. https://doi.org/10.3233/JCS-150529

[8] Stefano Calzavara, Sebastian Roth, Alvise Rabitti, Michael Backes, and Ben Stock. 2020. A Tale of Two Headers: A Formal Analysis of Inconsistent Click-Jacking Protection on the Web. In *USENIX Security*.

[9] Stefano Calzavara, Tobias Urban, Dennis Tatang, Marius Steffens, and Ben Stock. 2021. Reining in the Web's Inconsistencies with Site Policy. In *NDSS*.

[10] Xavier Chamberland-Thibeault and Sylvain Hallé. 2021. An Empirical Study of Web Page Structural Properties. *J. Web Eng.* 20, 4 (2021), 971–1002. https://doi.org/10.13052/jwe1540-9589.2044

[11] Herbert Van de Sompel, Michael L. Nelson, and Robert Sanderson. 2013. HTTP Framework for Time-Based Access to Resource States - Memento. (2013). https://doi.org/10.17487/RFC7089

[12] Herbert Van de Sompel, Michael L. Nelson, Robert Sanderson, Lyudmila Balakireva, Scott Ainsworth, and Harihar Shankar. 2009. Memento: Time Travel for the Web. *CoRR* (2009). http://arxiv.org/abs/0911.1112

[13] Nurullah Demir, Matteo Große-Kampmann, Tobias Urban, Christian Wressnegger, Thorsten Holz, and Norbert Pohlmann. 2022. Reproducibility and Replicability of Web Measurement Studies. In *The Web Conference*. https://doi.org/10.1145/3485447.3512214

[14] Steven Englehardt and Arvind Narayanan. 2016. Online Tracking: A 1-million-site Measurement and Analysis. In *ACM CCS*. https://doi.org/10.1145/2976749.2978313

[15] Florian Hantke and Ben Stock. 2022. HTML Violations and Where to Find Them: A Longitudinal Analysis of Specification Violations in HTML. In *Proceedings of the 22nd ACM Internet Measurement Conference*.

[16] Jeff Hodges, Collin Jackson, and Adam Barth. 2012. HTTP Strict Transport Security (HSTS). *RFC* 6797 (2012). https://doi.org/10.17487/RFC6797

[17] Lin-Shung Huang, Alexander Moshchuk, Helen J. Wang, Stuart Schecter, and Collin Jackson. 2012. Clickjacking: Attacks and Defenses. In *USENIX Security*.

[18] Jordan Jueckstock, Shaown Sarker, Peter Snyder, Aidan Beggs, Panagiotis Papadopoulos, Matteo Varvello, Benjamin Livshits, and Alexandros Kapravelos. 2021. Towards Realistic and Reproducible Web Crawl Measurements. In *The Web Conference*. https://doi.org/10.1145/3442381.3450050

[19] Jordan Jueckstock, Shaown Sarker, Peter Snyder, Panagiotis Papadopoulos, Matteo Varvello, Benjamin Livshits, and Alexandros Kapravelos. 2019. The Blind Men and the Internet: Multi-Vantage Point Web Measurements. *CoRR* (2019). arXiv:1905.08767 http://arxiv.org/abs/1905.08767

[20] Martin Klein, Harihar Shankar, Lyudmila Balakireva, and Herbert Van de Sompel. 2019. The Memento Tracer Framework: Balancing Quality and Scalability for Web Archiving. In *International Conference on Theory and Practice of Digital Libraries*. https://doi.org/10.1007/978-3-030-30760-8_15

[21] Lukas Knittel, Christian Mainka, Marcus Niemietz, Dominik Trevor Noß, and Jörg Schwenk. 2021. XSinator. com: From a Formal Model to the Automatic Evaluation of Cross-Site Leaks in Web Browsers.. In *CCS*.

[22] Michael J. Kranch and Joseph Bonneau. 2015. Upgrading HTTPS in mid-air: An empirical study of strict transport security and key pinning. In *NDSS*.

[23] Deepak Kumar, Zane Ma, Zakir Durumeric, Ariana Mirian, Joshua Mason, J Alex Halderman, and Michael Bailey. 2017. Security challenges in an increasingly tangled web. In *Proceedings of the 26th International Conference on World Wide Web*. 677–684.

[24] Ada Lerner, Tadayoshi Kohno, and Franziska Roesner. 2017. Rewriting History: Changing the Archived Web from the Present. In *ACM CCS*. https://doi.org/10.1145/3133956.3134042

[25] Adam Lerner, Anna Kornfeld Simpson, Tadayoshi Kohno, and Franziska Roesner. 2016. Internet Jones and the Raiders of the Lost Trackers: An Archaeological Study of Web Tracking from 1996 to 2016. In *USENIX Security*.

[26] Mozilla Developer Network. 2023. Cross-Origin-Embedder-Policy. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cross-Origin-Embedder-Policy.

[27] Mozilla Developer Network. 2023. Cross-Origin-Opener-Policy. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cross-Origin-Opener-Policy.

[28] Mozilla Developer Network. 2023. Cross-Origin-Resource-Policy. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cross-Origin-Resource-Policy.

[29] Mozilla Developer Network. 2023. Origin. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Origin.

[30] Mozilla Developer Network. 2023. Permissions Policy. https://developer.mozilla.org/en-US/docs/Web/HTTP/Permissions_Policy.

[31] Mozilla Developer Network. 2023. Referrer-Policy. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy.

[32] Jamie Murphy, Noor Hazarina Hashim, and Peter O'Connor. 2007. Take Me Back: Validating the Wayback Machine. *J. Comput. Mediat. Commun.* 13, 1 (2007). https://doi.org/10.1111/j.1083-6101.2007.00386.x

[33] Nick Nikiforakis, Luca Invernizzi, Alexandros Kapravelos, Steven Van Acker, Wouter Joosen, Christopher Kruegel, Frank Piessens, and Giovanni Vigna. 2012. You are what you include: large-scale evaluation of remote javascript inclusions. In *ACM CCS*. https://doi.org/10.1145/2382196.2382274

[34] Victor Le Pochat, Tom van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *NDSS*.

[35] Jannis Rautenstrauch, Giancarlo Pellegrino, and Ben Stock. 2023. The Leaky Web: Automated Discovery of Cross-Site Information Leaks in Browsers and the Web. In *2023 IEEE Symposium on Security and Privacy (SP)*.

[36] Sebastian Roth, Timothy Barron, Stefano Calzavara, Nick Nikiforakis, and Ben Stock. 2020. Complex Security Policy? A Longitudinal Analysis of Deployed Content Security Policies. In *NDSS*. The Internet Society.

[37] Sebastian Roth, Stefano Calzavara, Moritz Wilhelm, Alvise Rabitti, and Ben Stock. 2022. The Security Lottery: Measuring Client-Side Web Security Inconsistencies. In *USENIX Security*.

[38] Kyle Soska and Nicolas Christin. 2014. Automatically Detecting Vulnerable Websites Before They Turn Malicious. In *USENIX Security*.

[39] Sid Stamm, Brandon Sterne, and Gervase Markham. 2010. Reining in the web with content security policy. In *The Web Conference*. ACM. https://doi.org/10.1145/1772690.1772784

[40] Marius Steffens, Marius Musch, Martin Johns, and Ben Stock. 2021. Who's Hosting the Block Party? Studying Third-Party Blockage of CSP and SRI. In *NDSS*. https://doi.org/10.14722/ndss.2021.24028

[41] Ben Stock, Martin Johns, Marius Steffens, and Michael Backes. 2017. How the Web Tangled Itself: Uncovering the History of Client-Side Web (In)Security. In *USENIX Security*.

[42] Lukas Weichselbaum, Michele Spagnuolo, Sebastian Lekies, and Artur Janc. 2016. CSP Is Dead, Long Live CSP! On the Insecurity of Whitelists and the Future of Content Security Policy. In *ACM CCS*. https://doi.org/10.1145/2976749.2978363

[43] Michael Weissbacher, Tobias Lauinger, and William Robertson. 2014. Why is CSP failing? Trends and challenges in CSP adoption. In *International Workshop on Recent Advances in Intrusion Detection*.