Respect the ORIGIN! A Best-case Evaluation of Connection Coalescing in The Wild

Sudheesh Singanamalla[*,†], Muhammad Talha Paracha[*,‡], Suleman Ahmad[*], Jonathan Hoyland[*], Luke Valenta[*], Yevgen Safronov[*], Peter Wu[*], Andrew Galloni[*], Kurtis Heimerl[†], Nick Sullivan[*], Christopher A. Wood[*], Marwan Fayed[*]

[*] Cloudflare Inc. [†] University of Washington [‡] Northeastern University

ABSTRACT

Connection coalescing, enabled by HTTP/2, permits a client to use an existing connection to request additional resources at the connected hostname. The potential for requests to be coalesced is hindered by the practice of domain sharding introduced by HTTP/1.1, because subresources are scattered across subdomains in an effort to improve performance with additional connections. When this happens, HTTP/2 clients invoke additional DNS queries and new connections to retrieve content that is available at the same server. ORIGIN Frame is an HTTP/2 extension that can be used by servers to inform clients about other domains that are reachable on the same connection. Despite being proposed by content delivery network (CDN) operators and standardized by the IETF in 2018, the extension has no known server implementation and is supported by only one browser. In this paper, we collect and characterize a large dataset. We use that dataset to model connection coalescing and identify a least-effort set of certificate changes that maximize opportunities for clients to coalesce. We then implemented and deployed ORIGIN Frame support at a large CDN. To evaluate and validate our modeling at scale, 5000 certificates were reissued. Passive measurements were conducted on production traffic over two weeks, during which we also actively measured on the 5000 domains.

CCS CONCEPTS

• **Networks** → **Application layer protocols**; *Transport protocols*; **Public Internet**; *Network privacy and anonymity*; Network measurement; Network performance modeling; • **Security and privacy** → Network security.

KEYWORDS

Network Modelling, Measurement, Privacy, Protocols, Standards

ACM Reference Format:
Sudheesh Singanamalla, Muhammad Talha Paracha, Suleman Ahmad, Jonathan Hoyland, Luke Valenta, Yevgen Safronov, Peter Wu, Andrew Galloni, Kurtis Heimerl, Nick Sullivan, Christopher A. Wood, Marwan Fayed. 2022. Respect the ORIGIN! A Best-case Evaluation of Connection Coalescing in The Wild. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22), October 25–27, 2022, Nice, France.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3517745.3561453

IMC '22, October 25–27, 2022, Nice, France

ACM ISBN 978-1-4503-9259-4/22/10.
https://doi.org/10.1145/3517745.3561453

1 INTRODUCTION

The HTTP protocol and specifications evolve in response to needs and insights generated by web applications [9, 40]. Slower to evolve are the conventions with which web applications are developed and deployed. Understanding the gaps can help to identify best practices, generate consensus, and shed light on shortcomings.

One recent advancement is *connection coalescing*, a feature enabled by the HTTP/2 specification that enables connection re-use for different hostnames and domains, so long as the server is authorized to serve them [40]. The benefits of connection coalescing are best understood with the context of HTTP, which began as a one request, one connection, protocol [11]. Pipelined requests and persistent connections were later introduced in HTTP/1.1 [27] to improve performance [7]. The gains were themselves bottlenecked by head-of-line blocking [35, 39]. This led to a practice of domain sharding, in which browsers are 'tricked' into initiating new and parallel connections to multiple subdomains [30]. As a result, the burden of managing multiple connections, scheduling requests, and optimizing rendering pipelines shifted to browsers.

As the web continued to grow and evolve, web page load times (PLT) were understood to be a key performance metric, affecting user experiences and business revenues significantly [24, 52, 61]. To address continued performance limitations of HTTP/1.1, SPDY, a completely backward-compatible extension to HTTP/1.1, was introduced [60]. SPDY is a binary protocol, in contrast to text-based HTTP/1.1, and supports stream multiplexing where requests to and responses from the server use a single interleaved TCP connection [60]. SPDY also introduced prioritization of requests allowing browsers to request critical resources blocking page rendering first. A modified variant of SPDY was standardized by the IETF as HTTP/2 in 2012 and published as RFC 7540 in 2015 [9]. Despite the ability with HTTP/2 to revert sharding, little about website maintenance practices has changed. Alongside, HTTP/2's ability to coalesce websites that rely on otherwise unrelated domains is further complicated by certificate issuance, management, and maintenance to keep pace with websites and changes.

ORIGIN Frames were later proposed by CDN operators, and introduced into HTTP/2, so that content operators could signal coalescable connections in a way that reduces the certificate management burdens and facilitates scale and operational flexibility [40], while leaving the verification and correctness to the clients. To the best of our knowledge, however, there exists no server-side

implementation of ORIGIN. Even without ORIGIN, studies show that opportunities to coalesce based on IP address are being ignored or overlooked. Instead, browsers use redundant and excessive numbers of connections to request content because of longer-standing domain sharding and DNS load-balancing conventions [44], speculative optimizations [56], and varying implementations of Same-Origin Policy implementations [49].

At the outset of this study, we confirmed browser behaviours through testing and code inspection. Chromium (and Chrome) implements only IP address based coalescing; for any subresource in the webpage, this requires that DNS returns the same IP address used to establish the existing connection. In contrast, Firefox's IP-based coalescing permits transitivity among IP address sets returned by DNS. Also, to the best of our knowledge, only Firefox supports ORIGIN Frame on the client-side.

In this study we seek to understand the 'missed opportunities'. Specifically we seek to answer three questions:

(i) How much of the Internet is coalescable?
(ii) What changes are required to make that happen? and
(iii) Can it be done?

We begin by collecting a dataset of more than 315,796 websites from Tranco's top-500K list [41]. Equipped with this dataset we make the following contributions:

(1) We model our expectations based on characterizing requests to sub-resources in a page and their destinations by identifying those requests which could have been coalesced, and conservatively adjust through reconstructed page-load timelines on the basis of coalescing.
(2) We use our model to identify the numbers and types of changes required in certificates to enable coalescing, since the same changes would be needed for both IP based and ORIGIN Frames based coalescing.
(3) We implement and deploy ORIGIN Frames at a major CDN where 5000 certificates were re-issued for large-scale evaluation, consisting of both passive and active measurements of IP and ORIGIN coalescing. The implementation is open-source [19, 20] and has been submitted upstream to the golang H2 protocol implementation [50].

The model identifies best-case coalescing for the domains in the dataset. To facilitate support and deployment, we use our model to devise a least-effort modification strategy for certificate modifications. A third party domain used by 50% of the top 1M websites [53], managed by our deployment CDN, makes the domain an ideal candidate for evaluating the impact of coalescing. Our measurements indicate clear reductions in the number of TLS connections to the third party domain in both passive and active measurements. As a part of this work, we have initiated discussions with golang maintainers to merge our ORIGIN Frame changes upstream to the open source H2 protocol implementation in `net/http` package.

In deployment post-analysis we reflect on our experience (§ 6) to identify gaps and opportunities, and to describe a client failure according to rules of disclosure. We also argue that, however counter-intuitive, *performance should not be assumed inherent and should be avoided as a primary motivator*. This position is informed by both modelling and deployment, was a surprising outcome to the authors — and is the polar-opposite from where the study started. We instead argue that the primary motivations for ORIGIN Frames is client privacy, coupled with operational ease for operators. We further posit that ORIGIN opens resource scheduling opportunities for both client and server, without risk of scheduling violations by their parallel connections that compete for transmission resources.

2 BACKGROUND AND CONTEXT

2.1 Structure of a Webpage

Modern web pages inherit optimizations and options from HTTP/1.1. Among them are sharded domains, across which sub-resources are distributed to trick clients into opening simultaneous connections. According to the HTTP Archive [5], desktop and mobile devices make 73 and 69 median requests respectively, to different domains. The set of requested resources include cascading style sheets (CSS), Javascript files, images, and fonts. For example, sub-resources for a webpage `www.example.com` might be sharded on `images.example.com` and `static.example.com`, that may themselves be found at the same physical origin web server. Resources may also be delivered via Content Delivery Networks (CDNs) in an effort to improve performance, availability, security, or costs.

2.2 Certificates and ORIGIN Frames

Web servers host certificate files that are cryptographically signed and issued by a certificate authority (CA) that has verified ownership of the hostnames. During the TLS handshake, browsers retrieve and use the certificate to validate the web server as authorized to deliver content for those domains. A single certificate can cover many hostnames in its Subject Alternative Name (SAN) field. A client connection to any name in the set may be re-used to retrieve content for other names in the SANs.

Unfortunately the appearance of a hostname on a SAN in itself offers no guarantee that the name will be reachable for two reasons. First, re-use requires that the existing connection can by used by the server to authoritatively respond for those hostnames on the connected server IP address. Clients that attempt re-use for hostnames that are not configured on the connected IP address will receive a `421 Misdirected Request` error, incurring additional RTT penalties for their efforts before launching DNS queries and separate TCP+TLS connections. Notably, this event does not invalidate the certificate, and also does not violate the server's ability to authenticate domains listed on the SAN; the `421` is permissible and expected behaviour because resources, for example, can change or be moved to different servers or sockets. Second, formal definitions of Same Origin Policy are absent or incomplete [57], which leaves each browser to decide its own [49]. Clients interacting with TLS terminating middleboxes using the TLS SNI extension to serve content might unintentionally send confidential information to a server which may not be the intended target despite its authoritativeness.

The ORIGIN Frame [40] for HTTP/2 [9] relaxes the strictest definitions of Same Origin Policy that require the triples of scheme, host, and port to be identical, and instead allows for multiple hosts, and port information to relate to the same origin, while within the formal definitions of authorized origin [8]. The ORIGIN identifies an explicit 'origin-set' to the client, with hostnames on the existing TLS connection that are both authorized and should not incur `421` errors. Names in the origin-set should also appear in the certificate

SAN. Provided the certificate passes all the necessary checks, no additional validation is required. We are unaware of any server-side implementations of ORIGIN Frames, but when used correctly, ORIGIN Frames enable clients to request authoritative resources without new DNS queries and TLS handshakes.

2.3 Browsers Behaviours Differ

Connection re-use is optional for browsers. We inspect the source code for Chromium [1] and Firefox [2] to understand the conditions required for each, which are confirmed with testing. Coalescing in both browsers begins with a DNS query for subresources, despite being defined as optional in the specification [40]. Matching addresses is regarded as an additional check on the authority of the server for the desired content. However, this may be overly strict because DNS operators have long been able to return any or all addresses from a set for load-balancing or other purposes [14].

The difference between browsers is in their treatment of transitivity as implied by IP addresses. Consider a webpage where the DNS response has two addresses $\{IP_A, IP_B\}$ with a connection established on IP_A, and a subsequent DNS query for a sub-resource that returns $\{IP_B, IP_C\}$. In this example, Chromium keeps only IP_A in its connected set and discards IP_B, causing the transitivity with IPs for the subresource to be lost. Since IP_A in the connected set has no match with the address-set for the sub-resource, Chromium initiates a separate connection for the subresource. Firefox, alongside the connected-set, additionally caches the available-set of addresses returned in the DNS response. Upon seeing IP_B in common, Firefox assumes by transitivity that the resources available on IP_B and IP_C must also be available on IP_A, which it finds in the connected set and hence elects to re-use.

Recall that the design of ORIGIN requires that any name in the origin set should also appear in the parent certificate that clients use to validate authenticity [40]. In this context, IP matching may be overly prohibitive. The semantics of IP matching is also unclear as IP-to-name relationships become increasingly tenuous [26], meaning that DNS queries are decreasingly useful. We posit that the additional scrutiny is both unnecessary and potentially misleading.

In the next sections we collect and characterize data to model and understand how much coalescing might be possible if ORIGIN Frames were respected without the need to (falsely) validate on IP.

3 UNDERSTANDING WEB RESOURCES

In this section we describe the method used to collect and characterize resources used and successful requests made by browsers to more than 300K websites among the top 500K websites in the Tranco dataset [41].

3.1 Data Collection

Our dataset is collected from the top 500K websites in the Tranco Million list [41] taken on 14th February 2021. We used 100 2vCPU 7.5GB memory virtual machines in Google Cloud's East US datacenter, each running a modified privately hosted version of Web Page Test (WPT) [38]. Data was collected between 14 to 18 February 2021 with no network traffic shaping, allowing browsers to use available bandwidth freely. A designated primary VM instance was responsible for queuing list entries and instructing replicas to execute the data collection. For each root web page in the queue, WPT instances initiated a new Google Chrome (v88.0) session to eliminate DNS, and resource caching effects. On successful completion, Chrome developer tools were used to retrieve and write the page load data as an HTTP Archive format (HAR) file with a full timeline of events, as well as additional information about certificates and their validation. Resulting HAR files were stored into a cloud storage bucket.

[1] `net/http/http_stream_factory.cc`
[2] `netwerk/protocol/http/Http2Session.cpp`

Tranco sites		Per-website Median Values			
Rank	**Success**	**#Reqs**	**PLT (ms)**	**#DNS**	**#TLS**
1-100K	68244	89	6168.0	17	20
100K-200K	64163	83	5720.0	14	17
200K-300K	63334	80	5601.0	14	16
300K-400K	59827	79	5565.0	13	15
400K-500K	60228	78	5724.0	13	15
Total	**315796**	81	5746.0	14	16
		μ = 113	μ = 8088.0	μ = 22.55	μ = 26.59

Table 1: Successful automated collection of top-ranked Tranco websites, with median page-level attributes. Failures caused by non-200 HTTP errors and CAPTCHAs appear to distribute evenly across the set.

From the 500K websites that the WPT instances attempted to retrieve, 315,397 (63.51%) succeeded. The remaining trials were removed from the dataset because they either received non-200 response codes or were met with CAPTCHA challenges. Our snapshot of 315796 webpages results in 35,882,587 total network requests for completing the page loads. A coarse view of the trials is summarized in Table 1, binned by popularity rank into 100K size buckets. Interestingly, median values across the set appear to be unaffected by popularity rank. Across the whole dataset, webpages initiate requests for a median 81 subrequests (mean: 113 subrequests, interquartile range: 90). These additional HTTP requests incur medians of 14 additional DNS queries and 16 TLS handshakes and verifications, which contribute to a median 5746.0ms page load time for rendering the full page.

Finally, the destination IPs for each request were resolved to its origin autonomous system (AS) so that we could identify the number of requests serviced by any AS for each page. Additionally, we parsed the certificate chains of the TLS for the root webpage and new TLS connections triggered by subresource requests to (i) identify certificate issuers for the hostnames, (ii) inspect the presence of the Subject Alternative Name (SAN) extension, and (iii) validate that DNS names resolve to the IP address used. Each request was also parsed to determine the version of HTTP used.

Sources of Selection Bias Our dataset consists of a single snapshot. A longitudinal study to capture code churn, sub-resource modifications, and other website changes is out of scope. All data is collected by machines in a single datacenter, which might experience race-conditions or network effects that could be elucidated by geographically distributed scanners. We used the same machines throughout our evaluations for consistency, and saw no observable change in datacenter provisioning or capacity over time. However, our single datacenter source of data collection on 500K websites

Rank	AS Number	Org. Name	#Req	%
1	AS 15169	Google	7932198	22.10
2	AS 13335	Cloudflare	4937395	13.75
3	AS 16509	Amazon 02	3017176	8.40
4	AS 14618	Amazon AES	2019308	5.62
5	AS 54113	Fastly	1281402	3.57
6	AS 16625	Akamai AS	1087172	3.02
7	AS 32934	Facebook	998685	2.78
8	AS 20940	Akamai Intl. B.V.	583700	1.62
9	AS 16276	OVH SAS	548107	1.52
10	AS 24940	Hetzner Online GmbH	469293	1.30
Total				**63.68**

Table 2: The top-10 destination ASes for resource requests in our dataset. Two providers in the list each operate two ASes in the top-10.

could have triggered some of the DDoS protection measures and CAPTCHAs, which was mitigated by pacing the rate of website requests. The values in Table 1 give an indication that any possible bias or skew is unaffected by popularity.

3.2 Ethics & Data Privacy

This work raises no ethical concerns or issues. Active measurements were conducted only by authors in the study, and was limited to a small number of requests to each website in the measurement and sample sets. Access to data from passive measurements was highly restricted to a subset of authors; queries were limited in scope to the bare minimum needed without exposing details of client activity. Retention policies were in place, and even affected our ability to revisit some earlier measurements in the study. Finally, some of the proposals and recommendations in this work could affect operators that issue or manage certificates. We carefully balance feasibility, benefits, and risks as informed by our real-world deployment.

3.3 Page-load Characterization

Alongside the aggregate view of our dataset, a webpage-level characterization of the dataset helps to frame the potential for connection coalescing as modelled later in Section 4.

AS-level In our dataset, requests were made to a total of 13316 ASes. The top-10 destination ASes for requests to *sub-resources* are listed in Table 2, and belong to eight unique providers. The top-3 providers (Google, Cloudflare, and Amazon), represented by 4 ASes, service ~50% of all requests in the dataset. Collectively, the top-10 ASes (0.06%) service more than 60% of the total requests in our dataset. Looking beyond the table entries, it takes 51 ASes (0.38% of ASes in the dataset) to service 80% of the requests.

The number of unique ASes needed to fully load a webpage is summarized by Figure 1. The proportion of webpages that rely on a given number of ASes (blue-line, y_1-axis) shows that 6.5% of webpages request subresources from a single AS; these webpages could immediately benefit from connection re-use. The largest bin indicates 14% of webpages need two ASes to fully load i.e. pages that have a dependency on one additional AS for subresources. The corresponding CDF (red-line, y_2-axis) shows that 6 ASes are needed to fully load more than 50% of all webpages, suggesting very high colocation of content with the necessary subresources.

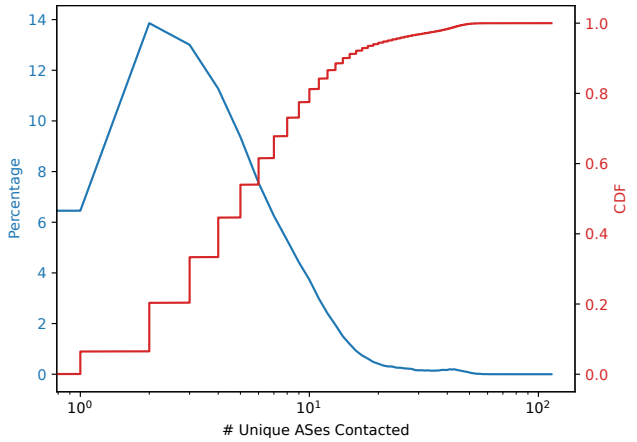


Figure 1: The frequency distribution ($y1$-axis) and CDF ($y2$-axis) of the number of unique ASes needed to load a webpage in the dataset

Protocol	# Requests	%
HTTP/2	26425963	73.64
HTTP/1.1	6850532	19.09
H3-Q050	123455	0.34
QUIC	26570	0.07
HTTP/1.0	12870	0.03
HTTP/0.9	15	4.1e-5
N/A	2443182	6.80
Total	**35882587**	**100.0**

Secure	35356995	98.53
Insecure	525592	1.47
Total	**35882587**	**100.0**

Table 3: Requests by application protocol and version (top), and proportion of encrypted requests (bottom).

Notably, from this perspective, the potential for connection re-use is optimistically approximated by the number of unique ASes needed to retrieve all subresources in a webpage. In practice, coalescing opportunities may be superceded by operational factors such as commitments to service level agreements, or helped by flexible mappings between sockets, names, and IP addresses [26].

Connection-level In our dataset 98.53% of the requests (35356995) are HTTPS connections, which validate server authority for the requested resources. The remaining 1.47% using HTTP can still re-use connections but require IP address matching via DNS. The breakdown of HTTP protocol versions used to navigate to the webpages are presented in Table 3. The majority of requests use HTTP/2 at 73.64%, with 19.09% of requests using HTTP/1.1 as negotiated by the clients. The 0.41% proportion of HTTP/3 and QUIC requests outnumber HTTP/1.0 and HTTP/0.9. Among the total requests made during our data collection, we find that 16.24% (5827389) initiated new TLS handshakes and validated corresponding certificates. The top-10 issuers of certificates for sub-resources are shown in Table 4. The top-5 distinct certificate issuers cover 59.25% of all certificates

Certificate Issuer	# Validations	%
Google Trust Services CA 101	1507140	25.86
Let's Encrypt (R3)	558798	9.58
Amazon	533391	9.15
Cloudflare Inc ECC CA-3	443727	7.61
DigiCert SHA2 High Assurance Server CA	411107	7.05
DigiCert SHA2 Secure Server CA	405377	6.95
Sectigo RSA DV Secure Server CA	402757	6.91
GoDaddy Secure Certificate Authority - G2	181798	3.11
DigiCert TLS RSA SHA256 2020 CA1	166198	2.85
GeoTrust RSA CA 2018	93168	1.59
Total (16.24% total requests)	**5827389**	**100.0**

Table 4: Breakdown of top certificate issuers by number of cerificates validated in the requests

Content Type	# Req	%
application/javascript	5118428	14.26
image/jpeg	4674423	13.02
image/png	3831287	10.67
text/html	3703184	10.32
image/gif	3221105	8.97
text/css	2796536	7.79
text/javascript	2428665	6.76
application/json	1270236	3.53
application/x-javascript	1208062	3.36
font/woff2	963218	2.68
image/webp	960160	2.67
text/plain	904010	2.52

Table 5: Requests breakdown by Top 12 content types

issued; if we count by the top-5 providers (aggregating multiple DigiCert services) this proportion grows to 69.05%.

3 of the 8 certificate issuers in Table 4 also can host content or terminate connections on origins' behalf. Providers of both services *may* be able to modify certificates to enable coalescing; in practice this ability is limited by certificate management complexities, or when customers choose to use separate or multiple providers.

Sub-resources An analysis of the requests organized and ranked by sub-resource content types is shown in Tables 5 and 6 respectively, in addition to grouping the content type by top ASes servicing those subresource requests. In both rankings across the dataset and for the top-3 ASes, Javascript is the most requested subresource content type. The top-10 subresource content types in the dataset are responsible for 86.55% of all subresource requests. When organized by provider, the most requested subresource types are similar. However, Google is unique for servicing a high number of requests for HTML and fonts resources, and for listing javascript as a `text/javascript` media type, which is now obsolete [31]. Lastly, ahead of our work to model the potential connection coalescing in the wild, the top-10 hostnames used to request subresources are shown in Table 7. The proportion of requests to these 10 hostnames is 12.5% of total requests.

The ORIGIN Frame is intended to facilitate, even encourage, connection coalescing. The characterization of our dataset suggests opportunities to do so. In the next section we use our dataset to model coalescing in the wild and improvements that are possible.

ASN	Content Type	#Req	%
Google (AS 15169)	text/javascript	1720366	21.69
	text/html	1141738	14.39
	image/gif	869759	10.96
	font/woff2	792852	9.99
Cloudflare (AS 13335)	application/javascript	1102387	22.32
	image/jpeg	959775	19.43
	image/png	590532	11.96
	text/css	529758	10.72
Amazon 02 (AS 16509)	application/javascript	644643	21.36
	image/jpeg	442718	14.67
	image/png	405607	13.44
	text/css	205720	6.81

Table 6: Breakdown of Top 4 content type requested from Top 3 Autonomous Systems (by requests)

Hostname	#Req	%
fonts.gstatic.com	802714	2.23
www.google-analytics.com	599600	1.67
www.facebook.com	567854	1.58
www.google.com	547569	1.52
tpc.googlesyndication.com	433845	1.21
cm.g.doubleclick.net	425743	1.18
googleads.g.doubleclick.net	416113	1.15
pagead2.googlesyndication.com	402448	1.12
fonts.googleapis.com	349679	0.97
cdn.shopify.com	315205	0.87

Table 7: Breakdown of Top 10 subresources hostnames requested during all page loads

4 MODELLING BEST CASE COALESCING

In this section we use the dataset collected and annotated in the previous section to better understand connection coalescing in the wild, and identify opportunities for improvement.

4.1 Reconstruction of Request Timelines

To understand connection coalescing as it exists and as it could be, we first extracted the webpage request timelines from each HAR file in our dataset. From these, the individual subresource requests were inspected and analyzed to identify those that could have been coalesced if the corresponding hostnames had been included in stream 0 as an ORIGIN Frame. Each timeline was then reconstructed first by finding the timelines' event labels {*block*, *send*, *wait*, *receive*} for the affected subrequests. We then modified those timestamps, conservatively, by omitting the smallest DNS query and TCP/TLS connection establishment times for blocking requests.

Consider a waterfall representation where multiple coalescable requests start at the same time but have varying DNS response times. The smallest time to remove is the minimum of the DNS response times for these queries and the difference between the larger response times is retained in the new timeline reconstruction. The CPU time beforehand to decide and dispatch the subresource queries and connections is unmodified in an effort to model browsers' dependency graph computation time. The resulting request timelines represent page loads on the basis of coalescing

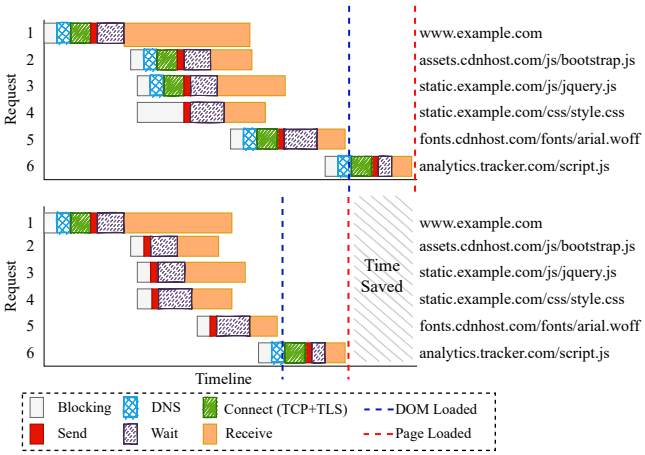


Figure 2: Timeline of requests involved in a page load event (top) with a reconstructed timeline due to ORIGIN Frames (bottom) showing an improvement in DOM loading and page load time.

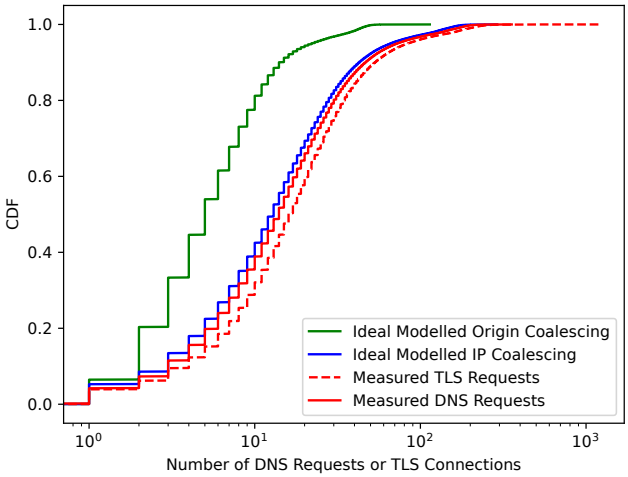


Figure 3: Comparison of the measured number of DNS requests and TLS connections using Chrome to the ideal IP based coalescing and origin coalescing settings

that could have been possible given ideal certificate SAN, ORIGIN Frames from operators, and client support.

Our model, informed by earlier observations (§ 3.3), assumes every server in each ASN can authoritatively serve all content for that ASN. We believe this to be reasonable for smaller operators that are likely to control larger proportions of their content (e.g., universities). For larger operators, however optimistic the assumption, prior works suggests that this approach is feasible [26]. The decision to equate an ASN with its ability to coalesce connections forms the core of our model and identifies both operational opportunities, as well as a potential limitation of the model's projections.

An example reconstruction is depicted in Figure 2 in which 6 object requests are needed to load `www.example.com` served by the CDN `cdnhost.com`. The top timeline draws from the original HAR file, and beneath it is the reconstruction if the requests were coalesced. As the browser parses and validates the base HTML, requests 2, 3, and 4 are initiated for critical subresources. The `css` resource retrieved by request 4 prompts the browser to initiate request 5 for a font, and finally leads to a *low priority* request 6.

The request for a base-page can never be coalesced since it initiates the first connection. In the example in Figure 2, request 2 to `assets.cdnhost.com`, given an ORIGIN Frame, could be coalesced onto the existing connection because the resource is available on the same CDN that serves the base-page. Similarly, each of requests 3, 4, and 5, are to sharded domains that are reachable on existing connections and can be coalesced. The last request (6) attempts to fetch resources from a hostname which is not serviced by the same CDN and hence cannot be coalesced. For purpose of modelling and prediction, we discard *DNS* and *Connect* events for coalescable requests, then reconstruct the timeline, accordingly. Before deciding that sharded domains can be coalesced, we first inspect IP address sets returned by DNS. For sets that reveal transitivity, we then identify the origin AS of the IP addresses. The IP to ASN mappings are drawn from an internal database at Cloudflare.

The set of names that should appear in an ORIGIN Frame for a website are those that could have been coalesced. The timeline reconstructions, coupled with characterizations in §3.3, give a model to predict the impacts of ORIGIN Frame-based coalescing.

Notably, this example makes clear that the *schedule* of events to load a page is compacted as a result of coalescing. One implication is that page load times (PLT) should improve; our evaluations (§5) suggest otherwise. We revisit this notion (§6) to argue reasons that PLT and other performance improvements should neither be assumed, nor relied upon. In lieu, we argue the immediate benefit of coalescing includes improved privacy, and also that it opens *scheduling* opportunities at the endpoints that cannot be violated by competing connection characteristics.

4.2 Predicting Queries, Connections, and Certificate Validations

In an *ideal* coalescing, the number of DNS queries, TLS handshakes, and certificate validations is equal to the number of separate services (not domains or hostnames) needed to serve all webpage resources. The ideal requires that, and for the purpose of prediction, we assume that ORIGIN Frames and certificate SANs are correctly configured with each service sending a valid corresponding ORIGIN Frame when a new connection is initiated.

DNS queries and TLS connections. The measured and ideal number of DNS queries and TLS connections needed to load a webpage are summarized by their CDFs in Figure 3. Interestingly, in our dataset the measured distributions differ between DNS queries and TLS connections, with medians of 14 and 16, respectively. Closer inspection suggests the differences are a result of race conditions in the browser. Examples include, (i) the practice of "happy eyeballs" [47] in which clients simultaneously initiate connections over both IPv4 and IPv6 to mitigate the performance penalty for failure on either; (ii) also speculative behaviour [1] which causes sets of sub-resources to be fetched in parallel by multiple network-state management threads (e.g., fonts from a CSS file). These types of

race conditions generate additional DNS queries or make multiple connections for the same sets of resources.

Alongside measured values in Figure 3 we plot distributions under ideal coalescing conditions. For IP-based coalescing, we inspect each webpage load in the dataset and reduce any set of two or more connections to the same IP address, to one connection; in other words, our model assumes no changes and looks for 'missed opportunities'. The median number of DNS queries and TLS connections for IP coalescing is 13, only ~7% less than measured DNS but ~19% for TLS and is presented as the solid blue line in Figure 3. These values are the upper-bound in which no two hostnames are listed on a single certificate. Certificate changes are operationally complex, as discussed in Section 5. In contrast, the ideal coalescing with ORIGIN Frames indicated by the solid green line in Figure 3 show marked improvement with a median of 5 each of DNS queries and TLS requests. In our dataset, this reduces median DNS queries by ~64% and TLS connections by ~67%.

Certificate Validations. As ORIGIN-based coalescing reduces the number of TLS connections, it also reduces the number of cryptographic certificate validations. As a result, the ideal ORIGIN coalescing for our dataset in Figure 3 predicts a best-case median of five certificate validations. For this distribution the interquartile range reduces from 22 in the dataset to 6; the 75th percentile shows a 76.67% improvement reducing the number of certificate validations from 30 to 9. One implication on the client is that the cryptographic computation overhead for validations is substantially reduced. The performance impact is an open question that we return to in Section 6.

4.3 Predicting Certificate Modifications

A properly supported ORIGIN Frame reduces the number of TCP and TLS connections as predicted by our model. However, full reduction of DNS queries and certificate validations presented in our model requires that the names in the ORIGIN are also listed in the certificate SANs. In the absence of the hostnames in the certificate, the client must query DNS and, if the IP address matches the address of the current connection could, *at best*, reuse the already established TCP connection to initiate a new TLS connection.

In practice, the reuse of an existing TCP+TLS connection for a new TLS connection requires termination of the existing TLS connection. Instead of keeping the existing TCP connection, browsers create new TCP+TLS connections. ORIGIN Frames reduce this burden on clients. By adding SANs to certificates and coalescing, fewer TCP and TLS connections need to be established. This raises a series of questions about the certificate changes needed in support.

Generate new certificates or modify existing ones? Certificate reuse is expected and common practice, but the scale and quantity of certificates to modify or generate has operational constraints: A single large certificate with all hosted names (as many as millions [26]) is unreasonable; equally, having as many as certificates as webpages is challenging for operators to manage effectively. A full measurement and understanding is regarded as out of scope: The scale (Let's Encrypt issues over 2.5 million certificates daily [3]), as well as individual operators' constraints, deserves its own investigation and is necessarily left for future work.

[3]https://letsencrypt.org/stats/

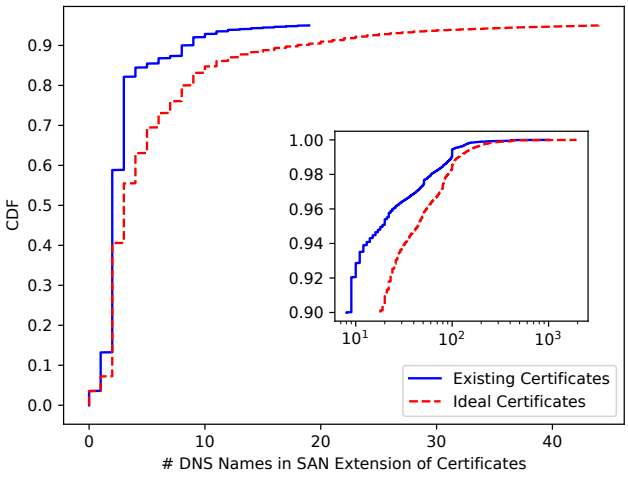


Figure 4: Comparison of the number of DNS SAN names in Existing Certificates (blue) to those after proposed certificate modifications (red). The top-most outset y-axis value is 0.94.

In this work we demonstrate feasibility (§5). Our model takes a compromise position and assumes no change in the number of certificates. Instead, we determine the volume of changes to existing certificates needed to enable perfect coalescing. To understand those changes we identify the validated certificates for each website in our dataset. In each website's certificate we identify and add the individual hostnames needed to load the webpage that are available from the same provider but absent from the SAN.

How would SAN sizes change? Figure 4 shows the distributions of SAN entries of the existing set (solid blue line), and the predicted number of additions to the SANs (dashed red line). In our dataset of 315,796 websites we change only the certificate for the website visited. From the SANs that changed, Figure 4 shows a median shift from 2 SAN DNS names to 3. The shift increases at the 75th percentile from 3 to 7. The distribution above the 94th percentile has a long tail, as shown in the inset.

What is the scale of required changes? The distribution of entries in the SAN is independent of the scale of change to each individual certificate, which we plot in Figure 5 ranked by the size of the SAN (in red). For each certificate the corresponding size of the SAN change is plotted in green, and the ranked order of predicted size after the changes in blue. In our dataset, 195693 (62.41%) of the 315,796 website certificates require no modifications, as represented at logscale in the tail of the figure. With no more than 10 changes, 290509 (92.66%) of websites are able to coalesce additional hostnames. We also identify long tail effects where ≈1% (3129) of the websites need more than 78 new DNS entries to their certificates. Our analysis finds that 230 websites already serve certificates with more than 250 DNS names in their certificates' SAN indicated by the red dots on the scatter plot. By updating existing certificates, we identify that this number increases by 130% to 529 websites which now serve certificates with more than 250 DNS SAN names with

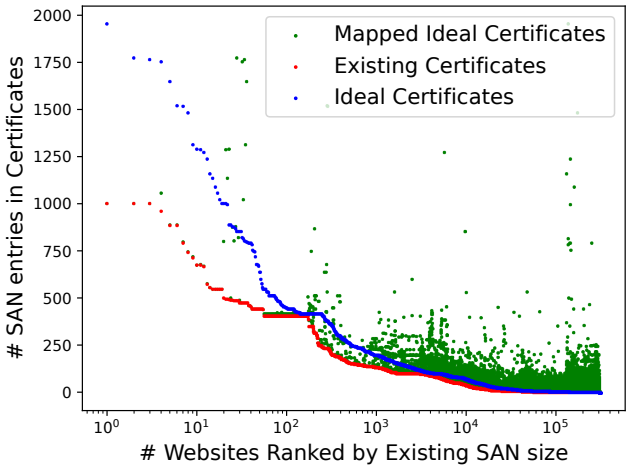


Figure 5: Tail distribution of number of SAN entries in existing certificates (red) with those after their modification (blue) for all navigated websites in the study. The changes needed per certificate in the existing distribution are shown as mapped ideal certificates (green).

the largest certificate in our analysis now containing 1951 DNS SAN names indicated by the blue dots in Figure 5.

A detailed ranking of the top-10 measured and predicted SAN sizes is presented in Table 8. For example, the top-ranked measured and predicted SAN size of 2 is unchanged. Interestingly, there is only one change in the SAN sizes that rank in the top-10: From the measured ranking, size 10 SAN entries drop out are replaced by size 7 — otherwise the set of sizes in the top-9 is unchanged. Beyond view from Table 8, we predict that 259315 (81.94%) websites would have up to 11 DNS SAN names in their modified certificates, which indicates minimal changes for the majority of the websites. While large certificates do reduce the number of certificates needing validation, they have additional implications on performance that we discuss in Section 6.5.

In our dataset there are also 11,131 websites with no SAN names in the certificate. Among those, only two certificates need changes to include coalescable hostnames. The remaining 11,129 (99.98%) websites serve their subresources, or have no coalescable hostnames, or both. They account for ~3% of the total websites in the study and can be seen in Figure 4 at $x = 0$.

Most effective changes Our analysis identifies popular cloud providers and CDN services that can provision certificates for their users. These certificates can be modified to include other subresources offered by the same CDN that serves the website. We identify individual providers and, alongside, the most frequently used hostnames for subresources in our dataset. For example, Table 9 shows the top three providers (Cloudflare, Amazon, and Google) in our dataset collectively responsible for serving 37.5% of the websites. Each entry also lists the five most used domains for those websites' subresources available at the same provider.

Among this set 78,155 websites in our snapshot are served by Cloudflare, comprising 24.74% of the websites measured, followed by Amazon Web Services (7.75%) and Google (5.09%). The most

Rank	#DNS SAN Entries				Rank Change
	Measured	Count	Ideal	Count (Pct. Change)	
1	2	143037	2	104604 (-26.86%)	=
2	3	73124	3	46772 (-36.03%)	=
3	1	30278	4	23701 (-21.72%)	↑3
4	0	11131	5	20130 (+80.84%)	↑5
5	8	8343	8	12408 (+48.72%)	=
6	4	7223	1	11456 (+50.60%)	↓3
7	9	6380	6	11297 (+77.07%)	↑1
8	6	4141	0	11129 (+168.75%)	↓4
9	5	3149	9	9833 (+212.25%)	↓2
10	10	2573	7	9295 (+261.25%)	↑2

Table 8: Distribution of the total SAN names in certificates after addition of DNS SAN names to certificates.

Provider	#Sites	Hostname	Count	%
Cloudflare (AS 13335)	78155 (24.74%)	cdnjs.cloudflare.com	12675	16.21
		*.cdnjs.cloudflare.com	12675	16.21
		sni.cloudflaressl.com	9839	12.58
		ajax.cloudflare.com	8816	11.28
		cdn.jsdelivr.net	6793	8.69
Amazon-02 (AS 16509)	24494 (7.75%)	cloudfront.net	4907	20.03
		*.cloudfront.net	4907	20.03
		*.hotjar.com	3620	14.77
		hotjar.com	3613	14.75
		*.s3.amazonaws.com	2944	12.01
Google (AS 15169)	16078 (5.09%)	google-analytics.com	13776	85.68
		*.google-analytics.com	13776	85.68
		urchin.com	13776	85.68
		www.googletagmanager.com	13300	82.72
		fps.goog	13300	82.72

Table 9: Top 5 frequently requested hostnames to include in certificates issued by Top 3 hosting and certificate providers

used domain for subresources is `cdnjs.cloudflare.com`, which is requested by 16.21% of websites served by Cloudflare. These webpages currently incur network and endhost resources for DNS and TLS that could otherwise be eliminated by coalescing if the hostnames were included in all Cloudflare issued certificates. Similarly, `cloudfront.net` has multiple sub resources that are used by 20.03% of Amazon users and the inclusion of the hostname in the DNS SAN of the certificates would improve clients' ability to coalesce connections. The inclusion of google-analytics hostnames in certificates from Google resources would result in improved connections management and reuse of connection from 85.68% of webpages served by Google.

Observations summarized by Table 9 suggest that large gains await with least-effort by adding a set of most-used subresource hostnames to existing certificates, and enabling ORIGIN Frame. It behoves large providers, in particular, to consider adding their most-used hostnames to certificates by default, and implementing support for ORIGIN Frame. On the client-side, since no certificate changes are needed, only ORIGIN Frame support is needed. In the absence of support, clients must fail-open according to the specification by ignoring the ORIGIN Frame [10, 55]; our deployment experience suggests that standards compliance is not a given (§ 6.7).

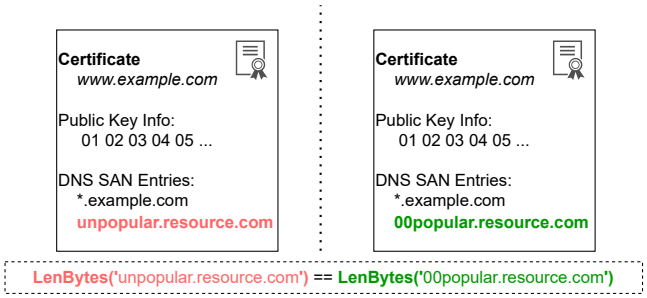


Figure 6: Experiment Setup - Certificate Issuance

5 VALIDATION WITH PRODUCTION TRAFFIC

In this section we validate our modelling expectations at a global CDN which serves a significantly large number of websites [54]. Two experiments were deployed months apart with different designs to satisfy the deployment CDN's operational constraints, beginning with IP-based coalescing. The results were viewed as sufficiently positive to invest resources to implement and evaluate ORIGIN Frame support. We used both multi-week passive, and single snapshot active measurements for the experimental deployments. The designs and results are described below.

One unavoidable caveat is that our validations are performed with changes at one single provider of service. While global in reach, the deployment CDN is one among many included in our model and its predictions (§4).

5.1 Sample Group and Certificates' Setup

Our deployments coalesce a third party (i.e., subresources) domain, hosted by the same CDN, that is used by ~50% of the top 1M websites [53], and receives over 5 Billion daily requests, worldwide [13]. Due to the importance of this third party domain, its request pipeline is provisioned with high-level traffic-engineering and service-level agreements (SLA). Notably, this means that changes to either the IP address or the request pipeline has implications for the CDN's wider service, and necessarily shapes experimental designs. For example, candidate domains were restricted to lower-tier service-level agreements (SLAs) to mitigate risk.

Our initial sample set was the 5000 domains with the most requests to the third party, as indicated by the `Referer` field in the CDN's third party logs. To respect privacy, the `Referer` field was truncated at the domain, and omitted subpages in the URL. From this set we found and removed 22% of websites that could not trigger the required actions from the active measurements because only their subpages request the third party resources we attempt to observe the connection coalescing to. Each domain was randomly assigned into an experimental or control group.

Recall that both IP and ORIGIN based coalescing require all coalescable domains to appear in the certificate SAN. We modified all existing certificates for the 5000 domains served by the CDN, as depicted in Figure 6. For the experimental group, the corresponding certificates were renewed with the third party domain (20 bytes) added to the SAN. To ensure integrity between control and experimental sets, we selected a valid and identical size third party domain used by *none* of the control domains. The certificates were also renewed for these domains to include the unused domain. As a result all certificate modifications in both treatment groups consisted of the same number of byte additions.

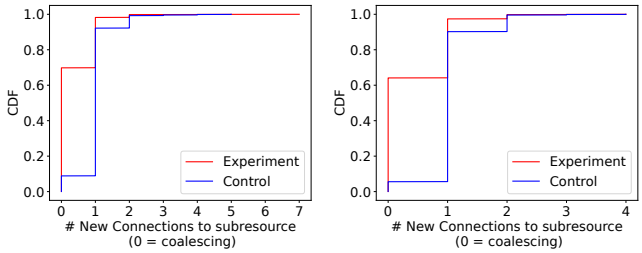
5.2 Evaluating IP Coalescing

Deployment Setup In addition to certificate changes, IP coalescing requires that (i) DNS returns the same IP address for domains to which connections could be coalesced, and that (ii) the webserver can respond to and satisfy content requests for all those domains, on that IP address. In our experiment we elected to use one single address for all 5000 domains in the experiment and for the two third party domains. The necessary changes were facilitated by mechanisms supporting addressing agility [26].

One operational challenge in this deployment was that the SLA associated with the sample group was very different from the SLA for the two third party domains. We limited exposure by (i) deploying to two medium-sized datacenters, and (ii) using a new and unallocated IP address. Corresponding changes were made to DNS to respond with that address, and to web servers to accept connections, for the control group and third party domain. Web servers were additionally configured to service requests where the HTTP `Host` for the third party domain was different from the SNI in the TLS connection (to pass domain-fronting checks). The deployment duration was two weeks in August 2021.

Server-side Passive Measurement A randomly sampled 1% of HTTP requests were logged. Requests to the third party domain from the experiment and control groups were tracked by the `Referer` field. In the default logging pipeline all requests in a TLS connection were with a unique identifier that, coupled with the referrer, was sufficient to count non-coalesced requests to the third party. However, there exists no obvious flag, field, or marker in TLS nor HTTP that would indicate a request is in a coalesced connection. Therefore, we modified the pipeline to additionally (i) set a flag bit for requests when the HTTP `Host` differed from the TLS SNI, (ii) treatment label (experiment/control), and (iii) label each request with its arrival order in the connection. We take the flag bit set as a reasonable signal of connection coalescing. From these requests, we look for arrivals ≥ 2, making sure to count the corresponding unique identifier only once. Requests to third party domain from the experiment and control treatments groups were confirmed to have similar profiles (request rate, popularity). Across all browsers and client requests to the websites in the two treatment groups in the study, we observed a 56% reduction in the rate of TLS connections to the third party domain from the experimental group relative to the control group.

Client-side Active Measurement If connections are being coalesced we would expect to see zero new TLS connections from the client to the third party. To confirm this behaviour we repeated the methodology described in Section 3 on the sample set. The number of new connections are captured by Figure 7a. Results shown are for Firefox (v91) for later comparison, since only Firefox has client-side support for ORIGIN Frame. In the control group, the proportion of 0 and 1 new connections is approximately 9% and 83%, respectively; and overall 90% making between 1 to 5 new connections. None of the control group initiated more than 7 new connections to the third party domain. In the experiment group, approximately 70% of



(a) IP-Based Coalescing (b) ORIGIN Frame

Figure 7: Connection reuse seen through the reduction in the number of TLS connections made during page load events as a part of active measurement.

visits trigger no new connections at all. A further 28% make one new connection, and no website uses more than 4 new connections.

Coalescing support via IP addresses needs alignment in both DNS responses and content reachability. This creates a burden on service operators, who must ensure consistency between content and address systems that are otherwise unrelated. However, their separation is needed to help maintain SLAs and different request pipeline optimizations for different domains or traffic types. ORIGIN Frames diminish these burdens on the operator.

5.3 Evaluating ORIGIN Frame Coalescing

Deployment Setup ORIGIN Frame removes the need to align IP addresses for different domains, and reduces operational costs and complexities on CDNs. On balance with the nominal changes to certificates predicted by our model in Section 4.3, the maintenance burden of ORIGIN Frames is isolated from the alternative, and preferable. IP-based coalescing, by comparison, requires co-orchestration with DNS and name-to-IP mappings, which reduces an operators ability to provision their systems or meet SLAs. In addition, misconfiguration risks hostnames being unreachable. In contrast, the consequence of a misconfigured ORIGIN Frame, for example with unreachable names, is to cause clients to 'fail-open' and revert to normal behaviour without coalescing.

Unfortunately, there exists no production-grade implementation of ORIGIN Frames in HTTP/2 web servers. To mitigate risk on the CDN's request pipeline, we implemented and integrated a custom connection-termination process, with ORIGIN support, into the production environment [19, 20]. Our custom process was configured to accept and service requests for the sample group. ORIGIN Frames were populated with either the third party or control domain to match the sample's certificate. Finally, the third party DNS changes were undone, which immediately restored the CDN operator to its standard traffic-engineering practices and SLAs.

Unlike the regional-restrictions needed to minimize risk with IP-based coalescing, the ORIGIN Frame implementation was deployed globally at over 275 points of presence [21]. To facilitate observability, the sample group was moved onto an isolated anycast address. Doing so meant that the CDN's network operations could, if needed, shift experimental traffic away from a datacenter by withdrawing the corresponding IP prefix from BGP announcements.

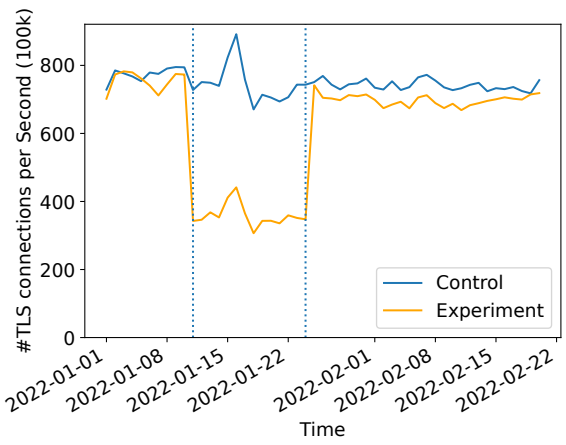


Figure 8: Reduction in Number of TLS connections made to coalesced sub resource observed during the duration of the experimental treatment

Finally, the decisions to implement a custom process and deploy globally were taken only after we could test and confirm that ORIGIN is either ignored or handled correctly by in-support browser versions. The deployment duration was two weeks in January 2022.

Server-side Passive Measurement The same logging pipe-line, 1% random sampling, and signals described in Section 5.2 during the IP based coalescing measurement were hardened for global deployment, and used to count coalescing with ORIGIN Frames. Global data was further filtered by user-agents corresponding to Firefox, the only browser at the time of deployment with support for ORIGIN. The daily number of new TLS connections for the sample group are shown longitudinally in Figure 8. The daily new TLS connections from the experimental group to the third party initiate approximately half as many new TLS connections compared to the control, and relative to daily counts before and after testing. Only in our active measurements did we discover that further coalescing was obstructed by use of the HTML `crossorigin` attribute set to anonymous in subrequests providing support for CORS [58]. We further discuss the implications of these mechanisms in section 6.2.

Client-side Active Measurement We again used the methodology described in Section 3 to measure the chosen websites directly. The number of new connections under ORIGIN using Firefox (v96) are captured by Figure 7b. In the control group, the proportion of 0 and 1 new connections is approximately 6% and 84%, respectively; and overall 94% making 1 to 4 new connections. In the experiment group, approximately 64% of visits trigger no new connections at all. A further 33% make one new connection. As a general improvement over natural IP coalescing, none of the sample set visits make more than 4 new connections to the third party domain with ORIGIN enabled, which is less than both groups made with IP coalescing, alone. While we expected higher percentages of connections to be coalesced with the usage of ORIGIN Frames (Figure 7b) compared to IP based coaslecing (Figure 7a), the lower percentage observed in our active measurements could be explained by the churn in resources required by the websites in the study due to modifications to the websites in our study by their developers given the

time elapsed between both the experimental setups (August 2021 and January 2022), possible sampling and artifact biases.

Given the positive results from IP coalescing described in Section 5.2, in this deployment we also perform a longitudinal view of new TLS connections to the third party from websites during the deployment of ORIGIN Frames, shown in Figure 8. The difference in new connections initiated by control and experimental groups during the deployment is in stark contrast to new connections initiated before and after. The average reduction during the experiment is clear at ≈50%, but also less than expected. Upon later inspection of websites in the experimental group, we observed the usage of `XMLHttpRequest` and `fetch` to make requests to the subresource which do not coalesce connections, similar to subresources obtained with a value *anonymous* for the HTML attribute crossorigin.

6 DISCUSSION

Our results indicate that ORIGIN Frame support can significantly improve rates of connection coalescing. In this section we discuss some of the implications of our study, both for operators and the web ecosystem.

6.1 On the Question of Performance

Connection coalescing opens resource scheduling opportunities. Re-ordering and restructuring of webpages has been shown to greatly affect page loads [34, 61]. However, the sequence of resources transmitted over multiple connections may be altered by network effects, and received by the client with different ordering and timings. An unintended ordering may then delay receipt of higher priority objects on a parallel connection, or contribute to head-of-line blocking in the application. In contrast, coalesced resources are always received in the ordering intended to optimize the critical path [59].

On performance in general, we *emphatically refrain from 'faster' as an assumed outcome or primary motivation.* Our preliminary evidence suggests 'no worse' is appropriate. The subtle interplay between resource object sizes, competing connections, and congestion control [3] is subject to network conditions. Bottleneck-share capacity, for example, diminishes as fewer connections compete for bottleneck resources. Yet many pipelined small objects may favour bytes transmitted in steady state on one connection, over bytes delayed by slow-start and discovery over many connections. Additionally, outside of bottlenecks, web interactions are dominated by latency over goodput [6, 28, 48].

This position is reinforced by our own attempts to demonstrate improvements in page load times (PLT), as captured in Figure 9. Modelling on our dataset as shown in (top) Figure 9, suggests that IP based (dashed blue line) and ORIGIN (solid green line) coalescing would improve median PLT by ~10% and ~27%, respectively. These predictions establish an upper bound for ideal conditions in which every server or service provider fully enables IP or ORIGIN Frame support for connection coalescing as indicated by our model. The nature of the model also preserves measured throughput speeds that may change with fewer coalesced connections.

As part of the model's predictions we also isolated the deployment CDN for validation with our experiments. The dotted-black line in (top) Figure 9 suggests that ORIGIN Frame support only at this CDN would yield a modest by comparison improvement of

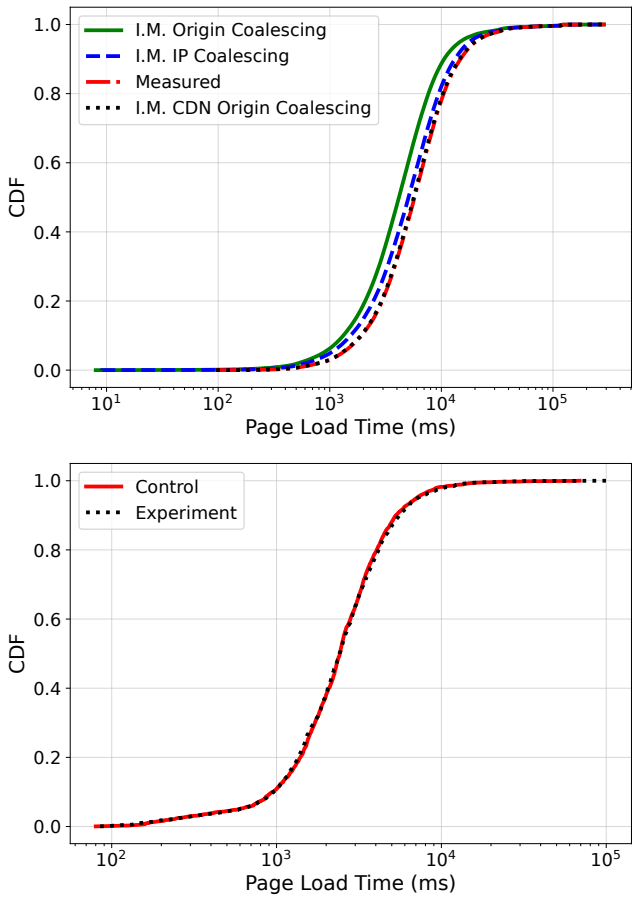


Figure 9: (*Top*) Model predictions of connection coalescing impact on Page Load Times (PLT); dotted-black line predicts only the deployment CDN. (*Bottom*) Measured PLTs at the deployment CDN with ORIGIN support indicate potentially minor improvement, but 'no worse' performance. (I.M = Ideal Modelled)

~1.5% in the median PLT. The measured PLTs in our experimental deployment with ORIGIN Frame improved by ~1%, as shown in (bottom) Figure 9 with a dotted-black line. This could be explained by differences in sample groups between measured dataset and our validation, or that our sample group differs from domains in the dataset presented in § 3. The performance differences between experimental and control groups for IP based coalescing were similarly indistinguishable, suggesting that the predicted differences between IP and ORIGIN coalescing in our model may be harder to discern in practice; Data retention policies at the CDN prevented us from revisiting that data. We emphasize, however, that while coalescing may be equally achieved via IP or ORIGIN, operator complexities make the former difficult to support.

Client caches are intentionally deleted between measurements in all of our active measurements to prevent bias from caching effects. Coalescing behaviour may, for example, be difficult to observe because of order-effects from browsing to webpage A before

B instead of from browing B before A; coalescing may also be preempted altogether by cached objects that can be loaded from local storage. We also must clarify that there is no reason to expect that clients' performance improved during the passive measurements (§5) strictly because of caching. This is because our deployment makes the domain coalescable, independent of the subresources served by that domain. Client activity may or may not rely on the same object from among thousands that are available [17]. Unfortunately we have no access to clients' telemetry that might concretely validate this position.

It is also possible that conservative browser behaviours play a role, yet can be safely changed to match modelled expectations (§6.8). In general, the relationship in context of coalescing between page load metrics, caching, and network conditions, deserves rigorous future study.

6.2 Notions of Trust and Privacy

X-Origin Security Browsers can trust existing connections for hostnames in the ORIGIN set with valid entries in the certificate's SANs. Today, browser client support exists only through Firefox. Other clients may be discouraged by the lack of clarity in the CORS standard [58]. Wider adoption may hinge on refinement or instruction in the Fetch standard and Same-Origin Policy [57]. The Online Certificate Status Protocol (OCSP) is independent, but notification of certificate revocation may give further confidence in the certificate's validity without DNS [45].

Fingerprinting, SNI, and Plaintext UDP DNS Whether ORIGIN Frames facilitate or challenge fingerprinting techniques is an open question. However, each coalesced connection hides an otherwise exposed plaintext SNI, and at least one DNS query if transmitted over UDP or TCP on port 53. These otherwise plaintext signals expose user activity, and interests. If adopted, the emerging Encrypted Client Hello (ECH) encrypts the SNI [43], and DNS over TLS (DoT) and HTTPS (DoH) protect user queries [32, 33]. ORIGIN Frame removes those signals from the network path, arguably improving privacy by reducing the amount of cleartext information leaked on path to network adversaries and on-lookers.

Web Packaging and Proxy Based Accelerators An alternative approach which is actively being considered by organizations and standards bodies is the usage of web packaging signed HTTP exchanges [62]. These packaging bundles allow content to be decoupled from origin servers and served by intermediate nodes without losing the cryptographic guarantees of connecting to the origin and behaves equivalent to scenarios with complete connection coalescing. Efforts are ongoing to improve privacy with the increased usage of web package signed HTTP bundles using private prefetch proxies [15]. These approaches while potentially improving performance beyond ORIGIN Frames pose privacy, and fairness challenges by preventing traffic from actually reaching the origin servers. This has been actively adopted by news websites using Accelerated Mobile Pages (AMP) but has also been heavily criticized [4].

6.3 Coalescing is tied to Content Operators

Intuitively, the ability to coalesce connections is tied to the domains and resources available via a CDN or content service provider that satisfies Same-Origin Policy [57]. Our measurements (§3) indicate that in today's Internet, popular Javascript, CSS libraries, and fonts needed to load and power webpages, are obtained from a comparatively small number of service providers. CDNs and content services have evolved into critical infrastructure. This study suggests positive ways in which such providers might further improve their service, user experience, and privacy – but also recognizes that coalescing opportunities favour CDNs and large organizations that manage multiple web resource infrastructure over that of single-origin operations.

6.4 Low Impact on Certificate Transparency

Certificate Transparency Logs (CT Logs) are append-only ledgers of certificate and are critical components of the certificate infrastructure in the Web Public Key Infrastructure (PKI). Certificate authorities (CAs) issuing certificates to a website for a hostname also write the certificate to multiple CT logs, operated by different organizations and monitored publicly [36, 37]. Modifying 37.59% (120103) of the websites contributes to a 5-10% increase in the number of certificates issued by CAs daily [2, 18]. The unbalanced publishing of certificate records between CAs and CT log operators results in increased stress among a few large CT log operators (e.g., Cloudflare, Google) but can be mitigated by improving load distribution among log operators [46]. The global certificate issuance rate is 257034 certificates per hour indicating that a bursty one time operation of certificate modifications as described in section 4.3 would not adversely affect CT log infrastructure performance [18].

6.5 SAN, ORIGIN, or Secondary Certificates

All forms of coalescing requires certificate modifications, but which modifications to make? Intuitively, an operator could generate a single certificate with all its hostnames in the SAN. This practice is permitted by IETF certificate standards, but is undesirable for two reasons. First, there is no natural IP coalescing unless the names all *reliably* share an IP address, which can restrict operators ability to meet SLAs, balance load, or be resilient and agile [26, 29].

Second, large certificates can be problematic both for the connection and user experience. Browsers fail to navigate to hostnames which serve extremely large certificates (e.g., 10000 DNS SAN names)[4] and present an SSL protocol error to the user. Also, in the absence of an explicit limit on DNS names in the IETF specification, different certificate authorities operate with different limits on their certificates. Let's Encrypt, DigiCert, and GoDaddy limit the issuance of certificates to 100 names per certificate [22, 23, 25] while Comodo uses 2000 as the limit [51]. Our analysis within the top 500K websites identifies that the cPanel Inc, DFN-Verein Global Issuing CA, and GlobalSign CloudSSL CA - SHA256- G3 currently issue certificates with more than 800 DNS SAN names.

The usage of large certificates affects the TLS handshake when setting up a connection since the certificate no longer fits within the 16KB TLS record size incurring additional packets and round trip times to deliver the certificate to the client [16]. Our study suggests that connection coalescing needs only a few (up to 7) additions to the certificates. However, the absence of "best practice" conventions may lead providers to generate unnecessarily large

[4] https://10000-sans.badssl.com/

certificates. Further investigations are needed to understand the certificate practices and sizes in the long tail of the Internet that lay outside the top-million.

A possible alternative may exist as a result of secondary certificate authentication [12], a draft proposal introduced to avoid large certificates and their negative impacts. The proposed changes to certificates introduce an ORIGIN Frame equivalent called *certificate frames*. These are blocking stream operations sent in stream 0 and provide alternative ways to obtain server certificates other than through the initial TLS handshake. If needed, large certificates can be broken into smaller certificates to match the TLS record sizes and sent using the proposed certificate frames. However, the usage of certificate frames incurs the transmission of complete X.509 certificates by the servers each with embedded public key, and signature which are the largest segments of the certificate increasing their size compared to the required certificate modification. However, the certificate frames could provide some flexibility to service providers to send certificates on demand without modifying the main certificate of the webpage or including the popular DNS name in multiple certificates, and could benefit from additional investigation.

Recall that our model, in the context of ORIGIN Frame, assumes changes to only those certs that need changing with only the coalescable names. As shown in Section 4.3, this least-effort approach indicates that no more than 3 additional hostnames in the certificate are needed to benefit 50% of websites, 7 additional hostnames at the 75th percentile, and 10 or fewer hostnames for 92% of websites. The usage of certificate frames have yet to see adoption, and are beyond scope of this work. A comparative study is suggested as future work for the measurement community.

6.6 What about HTTP/3 and QUIC?

The popularity of HTTP/3 is continuing to grow and would eventually replace the current usage of HTTP/2. The usage of ORIGIN Frames would continue to add significant value and contribute towards improving page load times and in-turn user experiences. The replacement of TCP with QUIC and introducing streams at the transport layer presents many opportunities for coalescing. QUIC and TLS1.3 introduce 0-RTT handshake mechanisms further reducing the amount of time needed to perform the handshakes. Clients using HTTP/2 protocol can leverage optimizations from TCP Fast Open to include portions of the TLS handshake within the SYN packets reducing 1 RTT. However, today HTTP/3 has no proposed standard for ORIGIN Frames and coalescing based on the ORIGIN Frames leaving the ORIGIN Frames restricted currently to HTTP/2.

6.7 Non-compliant HTTP/2 Software Stacks

An unanticipated outcome of our study was that it exposed non-compliant HTTP/2 implementations. The HTTP/2 specification mandates that clients ignore and discard unknown frames [9], a mandate respected by all operating systems and supported browsers. However client communications may be subject to separate network stacks running, for example, antivirus software with Internet security features, or corporate firewalls, and networks that install custom managed self-signed certificates.

During our experiments with ORIGIN Frame, a developer of anti-virus and Internet security software products contacted the CDN to enquire about an increased number of failed connections to the websites in our study. Following the rules of disclosure, a collaborative diagnosis pinpointed the issue to an unknown HTTP/2 frames handling bug in the developer's network agent. Rather than ignore the unrecognized frame as is required by the specification, the network agent instead would tear down the TLS connection and prevent clients from accessing the websites in our experiments. Upon careful consideration by the CDN, and given the sensitive nature of the antivirus software developers' operations, the CDN agreed to limit disclosure and pause ORIGIN testing for a limited duration. In September 2022, the antivirus provider confirmed to the CDN that the issue in the product had been fixed.

6.8 Next Steps for ORIGIN Support

Based on our experiences and observations, we describe a set of next steps for wider support of ORIGIN Frame in the Internet ecosystem.

Content Operators In this study we have identified opportunities for operators that host third party resources. With ORIGIN Frame, our models suggest that adding the most popular of those domains to the appropriate certificates (see Table 9) would reduce the number of TLS connections made to the resources, thus improving server compute overheads and benefitting clients that can and choose to coalesce their connections.

Web Servers, and Browser Clients In our evaluations we confirmed that the Firefox browser conservatively continues to make new and subrequest *blocking* DNS requests to hostnames in the ORIGIN Frame, despite their inclusion in the modified TLS certificate. These additional queries could be avoided, conferring privacy and other benefits to users. Doing so does not alter isolation techniques, for example, as exists in the render process to prevent security attacks [42]. For improved adoption, we also recommend web server software to integrate support for configuring ORIGIN Frames. As a part of this work, we have initiated discussions with golang developers to merge our ORIGIN-supporting changes upstream to the `net/http` implementation [19, 20, 50].

7 CONCLUSION

ORIGIN Frames are extremely useful hints when respected by browsers and need server modifications. Our experiments reveal an important role that ORIGIN Frames could play in today's Internet. Our analysis on a large-scale dataset finds that adding no more than 10 DNS names to 37.59% of the certificates will reduce certificate validations (i.e., new TLS handshakes) by 68.75%, while reducing the number of render blocking DNS queries by 64.28%. Clients additionally reap these benefits in privacy by reducing cleartext DNS exposure to network on-lookers. The results presented in this paper, generated at a large CDN, indicate our proposed changes are feasible and offer a glimpse into the potential of ORIGIN Frames.

ACKNOWLEDGMENTS

The authors would like to thank Kyle Schomp for shepherding the revisions of this paper. The authors also thank Larry Archer, Michel Bamps, Petros Gigis, Vasileios Giotsas, Vânia Gonçalves, John Graham-Cumming, Mihir Jham, and Avani Wildani for their valuable discussion, feedback, and support.

REFERENCES

[1] 2012. Issue 116982: Chromium opens useless TCP connections. Chromium Bugs. https://bugs.chromium.org/p/chromium/issues/detail?id=116982#c6

[2] Josh Aas. 2021. Preparing to Issue 200 Million Certificates in 24 Hours. https://letsencrypt.org/2021/02/10/200m-certs-24hrs.html.

[3] Neil Agarwal, Matteo Varvello, Andrius Aucinas, Fabián Bustamante, and Ravi Netravali. 2020. *Mind the Delay: The Adverse Effects of Delay-Based TCP on HTTP.* Association for Computing Machinery, New York, NY, USA, 364–370.

[4] Amelia Andersdotter and Signatories. 2018. A letter about Google AMP. http://ampletter.org/.

[5] HTTP Archive. 2021. Report: The State of the Web. https://httparchive.org/reports/state-of-the-web.

[6] Catalin-Alexandru Avram, Kenneth Salem, and Bernard Wong. 2014. Latency amplification: Characterizing the impact of web page content on load times. In *2014 IEEE 33rd International Symposium on Reliable Distributed Systems Workshops*. IEEE, 20–25.

[7] Paul Barford and Mark Crovella. 1999. A performance evaluation of hyper text transfer protocols. In *Proceedings of the 1999 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*. 188–197.

[8] Adam Barth. 2011. The Web Origin Concept. RFC 6454. https://doi.org/10.17487/RFC6454

[9] Mike Belshe, Roberto Peon, and Martin Thomson. 2015. Hypertext transfer protocol version 2 (HTTP/2).

[10] Mike Belshe, Roberto Peon, and Martin Thomson. 2015. Hypertext Transfer Protocol Version 2 (HTTP/2). RFC 7540. https://doi.org/10.17487/RFC7540

[11] Tim Berners-Lee, Roy Fielding, and Henrik Frystyk. 1996. Hypertext transfer protocol–HTTP/1.0.

[12] Mike Bishop, Nick Sullivan, and Martin Thomson. 2020. *Secondary Certificate Authentication in HTTP/2.* Internet-Draft draft-ietf-httpbis-http2-secondary-certs-06. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-ietf-httpbis-http2-secondary-certs-06 Work in Progress.

[13] Zack Bloom. 2019. An Update on CDNJS. Cloudflare Blog. https://blog.cloudflare.com/an-update-on-cdnjs/

[14] Thomas P. Brisco. 1995. DNS Support for Load Balancing. RFC 1794. https://doi.org/10.17487/RFC1794

[15] Michael Buettner and Kenji Baheux. 2020. Continuing our journey to bring instant experiences to the whole web. https://blog.chromium.org/2020/12/continuing-our-journey-to-bring-instant.html.

[16] Paul Calvano. 2020. SAN Certificates: How Many Alt-Names are too Many? https://discuss.httparchive.org/t/san-certificates-how-many-alt-names-are-too-many/1867.

[17] cdnjs. 2022. Libraries - cdnjs - The #1 free and open source CDN built to make life easier for developers. Website. https://cdnjs.com/libraries

[18] Cloudflare. 2021. Merkle Town - Explore the Certificate Transparency Ecosystem. https://ct.cloudflare.com/.

[19] Cloudflare. 2022. Go Networking. https://github.com/cloudflare/net-originframe

[20] Cloudflare. 2022. ORIGIN Frame Implementation in Go. https://github.com/cloudflare/go-originframe

[21] Cloudflare. 2022. The Cloudflare global network. https://www.cloudflare.com/en-gb/network/

[22] Go Daddy. 2021. One multi-domain certificate does it all. https://www.godaddy.com/web-security/multi-domain-san-ssl-certificate.

[23] DigiCert. 2021. Secure Multiple Domain Names with a Single SSL/TLS Certificate. https://www.websecurity.digicert.com/security-topics/san-ssl-certificates.

[24] Kit Eaton. 2012. How One Second Could Cost Amazon USD 1.6 Billion In Sales. https://www.fastcompany.com/1825005/how-one-second-could-cost-amazon-16-billion-sales.

[25] Let's Encrypt. 2021. Rate Limits - Let's Encrypt. https://letsencrypt.org/docs/rate-limits/.

[26] Marwan Fayed, Lorenz Bauer, Vasileios Giotsas, Sami Kerola, Marek Majkowski, Pavel Odintsov, Jakub Sitnicki, Taejoong Chung, Dave Levin, Alan Mislove, Christopher A. Wood, and Nick Sullivan. 2021. The Ties That Un-Bind: Decoupling IP from Web Services and Sockets for Robust Addressing Agility at CDN-Scale. In *Proceedings of the 2021 ACM SIGCOMM 2021 Conference* (Virtual Event, USA) *(SIGCOMM '21)*. ACM, 433–446. https://doi.org/10.1145/3452296.3472922

[27] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. 1999. Hypertext transfer protocol–HTTP/1.1.

[28] Tobias Flach, Nandita Dukkipati, Andreas Terzis, Barath Raghavan, Neal Cardwell, Yuchung Cheng, Ankur Jain, Shuai Hao, Ethan Katz-Bassett, and Ramesh Govindan. 2013. Reducing web latency: the virtue of gentle aggression. In *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*. 159–170.

[29] Ashley Flavel, Pradeepkumar Mani, David Maltz, Nick Holt, Jie Liu, Yingying Chen, and Oleg Surmachev. 2015. FastRoute: A Scalable Load-Aware Anycast Routing Architecture for Modern CDNs. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. USENIX Association, Oakland, CA, 381–394. https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/flavel

[30] Ilya Grigorik. 2013. Making the web faster with HTTP 2.0. *Commun. ACM* 56, 12 (2013), 42–49.

[31] Bjoern Hoehrmann. 2006. Scripting Media Types. RFC 4329. https://doi.org/10.17487/RFC4329

[32] Paul Hoffman and Patrick McManus. 2018. Dns queries over https (doh). *Internet Requests for Comments, RFC Editor, RFC* 8484 (2018).

[33] Zi Hu, Liang Zhu, John Heidemann, Allison Mankin, Duane Wessels, and Paul E. Hoffman. 2016. Specification for DNS over Transport Layer Security (TLS). RFC 7858. https://doi.org/10.17487/RFC7858

[34] Ronny Ko, James Mickens, Blake Loring, and Ravi Netravali. 2021. Oblique: Accelerating Page Loads Using Symbolic Execution. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, 289–302.

[35] Balachander Krishnamurthy and Craig E Wills. 2000. Analyzing factors that influence end-to-end web performance. *Computer Networks* 33, 1-6 (2000), 17–32.

[36] Ben Laurie, Adam Langley, and Emilia Kasper. 2013. Certificate Transparency. RFC 6962. https://doi.org/10.17487/RFC6962

[37] Bingyu Li, Jingqiang Lin, Fengjun Li, Qiongxiao Wang, Qi Li, Jiwu Jing, and Congli Wang. 2019. Certificate Transparency in the Wild: Exploring the Reliability of Monitors. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (London, United Kingdom) *(CCS '19)*. Association for Computing Machinery.

[38] Patrick Meenan and Contributors. 2021. Web Page Test - Web Performance Testing Code. https://github.com/WPO-Foundation/webpagetest.

[39] Henrik Frystyk Nielsen, James Gettys, Anselm Baird-Smith, Eric Prud'hommeaux, Håkon Wium Lie, and Chris Lilley. 1997. Network performance effects of HTTP/1.1, CSS1, and PNG. In *Proceedings of the ACM SIGCOMM'97 conference on Applications, technologies, architectures, and protocols for computer communication*. 155–166.

[40] M Nottingham and E Nygren. 2018. The origin http/2 frame. *RFC8336, IETF, Mar* (2018).

[41] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2018. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings of the 2018 Network and Distributed Systems Security Symposium*.

[42] Charles Reis, Alexander Moshchuk, and Nasko Oskov. 2019. Site isolation: process separation for web sites within the browser. In *28th USENIX Security Symposium (USENIX Security 19)*. 1661–1678.

[43] Eric Rescorla, Kazuho Oku, Nick Sullivan, and Christopher A. Wood. 2022. *TLS Encrypted Client Hello.* Internet-Draft draft-ietf-tls-esni-14. Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-ietf-tls-esni-14 Work in Progress.

[44] Constantin Sander, Leo Blöcher, Klaus Wehrle, and Jan Rüth. 2021. Sharding and HTTP/2 Connection Reuse Revisited: Why Are There Still Redundant Connections?. In *Proceedings of the 21st ACM Internet Measurement Conference* (Virtual Event) *(IMC '21)*. Association for Computing Machinery, 292–301.

[45] Stefan Santesson, Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Dr. Carlisle Adams. 2013. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 6960. https://doi.org/10.17487/RFC6960

[46] Quirin Scheitle, Oliver Gasser, Theodor Nolte, Johanna Amann, Lexi Brent, Georg Carle, Ralph Holz, Thomas C Schmidt, and Matthias Wählisch. 2018. The rise of Certificate Transparency and its implications on the Internet ecosystem. In *Proceedings of the Internet Measurement Conference 2018*. 343–349.

[47] David Schinazi and Tommy Pauly. 2017. Happy Eyeballs Version 2: Better Connectivity Using Concurrency. RFC 8305. https://doi.org/10.17487/RFC8305

[48] Brandon Schlinker, Italo Cunha, Yi-Ching Chiu, Srikanth Sundaresan, and Ethan Katz-Bassett. 2019. Internet performance from facebook's edge. In *Proceedings of the Internet Measurement Conference*. 179–194.

[49] Jörg Schwenk, Marcus Niemietz, and Christian Mainka. 2017. Same-Origin Policy: Evaluation in Modern Browsers. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 713–727. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/schwenk

[50] Sudheesh Singanamalla, Jonathan Hoyland, Suleman Ahmad, Peter Wu, and Marwan Fayed. 2022. proposal: net/http: Enabling support for HTTP/2 ORIGIN Frames RFC 8336. https://github.com/golang/go/issues/55121

[51] Comodo SSL Store. 2021. Multi Domain SSL Certificate. https://comodosslstore.com/comodo-mdc-ssl.aspx.

[52] Akamai Technologies. 2017. Akamai Online Retail Performance Report: Milliseconds Are Critical. Web Performance Analytics Show Even 100-Millisecond Delays Can Impact Customer Engagement and Online Revenue. https://blog.cloudflare.com/ecdsa-the-digital-signature-algorithm-of-a-better-internet/.

[53] Web Technology Surveys (W3 Techs). 2022. Usage of CDNJS broken down by ranking. https://w3techs.com/technologies/breakdown/cd-cdnjs/ranking

[54] Web Technology Surveys (W3 Techs). 2022. Usage statistics and market share of Cloudflare. https://w3techs.com/technologies/details/cn-cloudflare

[55] Martin Thomson and Cory Benfield. 2022. HTTP/2. RFC 9113. https://doi.org/10.17487/RFC9113

[56] Chromium Project Bug Tracker. 2012. Issue 116982: Chromium opens useless TCP connections. https://bugs.chromium.org/p/chromium/issues/detail?id=116982.
[57] W3C. [n.d.]. Same Origin Policy. https://www.w3.org/Security/wiki/Same_Origin_Policy
[58] WHATWG W3C. 2020. Cross-Origin Resource Sharing. https://www.w3.org/TR/2020/SPSD-cors-20200602/
[59] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2013. Demystifying Page Load Performance with WProf. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*. USENIX Association, Lombard, IL, 473–485.
[60] Xiao Sophia Wang, Aruna Balasubramanian, Arvind Krishnamurthy, and David Wetherall. 2014. How Speedy is SPDY?. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. 387–399.
[61] Xiao Sophia Wang, Arvind Krishnamurthy, and David Wetherall. 2016. Speeding up web page loads with shandian. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)*. 109–122.
[62] Kinuko Yasuda. 2018. Signed HTTP Exchanges. https://developers.google.com/web/updates/2018/11/signed-exchanges.