Impact of DANE on Webpage Load Time

Kota Yagi*, Katsuki Isobe†, Daishi Kondo†, and Hideki Tode†
*College of Engineering, Osaka Prefecture University, Japan
†Graduate School of Informatics, Osaka Metropolitan University, Japan
Email: seb01140@st.osakafu-u.ac.jp, sb22688c@st.omu.ac.jp, daishi.kondo@omu.ac.jp, tode@omu.ac.jp

***Abstract*—An authentication technique based on the domain name system (DNS), called DNS-based authentication of named entities (DANE), has been used to support the simple mail transfer protocol. However, DANE has not been extensively used for hypertext transfer protocol secure. This is possibly attributed to the increase in latency caused by DNS security extensions, which are essential for DANE. Unlike email services, web-service access is negatively impacted by latency which considerably affects the quality of experience for end users. This study assessed the extent to which DANE affects webpage load time. To investigate the load time, we developed *DANEWebPerf*, a tool—comprising a web browser, DNS cache server, and proxy that enables DANE use—to measure the webpage load time. Our experiments showed that using DANE increased the average webpage load time by at most 1190.9 ms. Therefore, DANE for the web does not significantly impact the quality of experience of end users.**

***Index Terms*—Domain name system-based authentication of named entities, Latency measurements, Webpage load time**

I. Introduction

The public key infrastructure (PKI) [1] uses public-key cryptography for encryption, decryption, and digital signatures. One of the roles of a certificate authority (CA), which is an entity of PKI, is to issue certificates. A trusted CA can issue certificates for any domain. In contrast, a compromised CA may issue a certificate to an attacker who is not the domain owner. Cases of fraudulent certificate issuance have been reported in the past [2], [3], undermining trust in the conventional PKI.

To address PKI vulnerabilities, an authentication technique that uses the domain name system (DNS), called DNS-based authentication of named entities (DANE) [4], has been proposed. It relies on DNS security extensions (DNSSEC) [5]–[7], a mechanism used to authenticate the origin of DNS responses and ensure their integrity. It issues and verifies certificates by comparing `TLSA` (`TLSA` does not stand for anything) records, including the certificate information registered in the DNS, with certificates obtained from transport layer security (TLS) servers. The trust anchor in DANE is the public key of the DNS root zone; thus, DANE uses DNSSEC, whereas the conventional PKI uses the CA public key as the trust anchor. DANE issues and verifies CA-independent certificates, improving the trustworthiness of PKI. DANE is primarily used in the simple mail transfer protocol (SMTP) and the hypertext transfer protocol secure (HTTPS) [8]. However, their deployment situations differ. As of February 2024, approximately four million SMTP domains used DANE [9], and Microsoft is currently deploying DANE on its mail servers [10]. By contrast, its application in HTTPS is limited as most major web browsers do not support it [11], [12]; therefore, third-party tools [13], [14] are required to use DANE on these browsers. Consequently, DANE-based web browsing has not become widespread.

A reason for the minimal adoption of DANE for HTTPS may be the increased latency caused by DNSSEC [15], [16]. This is because DNSSEC requires more name resolutions to validate the public key. Unlike email services, web services are considerably affected by latency, which negatively impacts the quality of experience (QoE) of end users when accessing websites. To address this problem, the request for comments (RFC) 9102 [17], a TLS DNSSEC chain extension, enables the DNSSEC validation of the `TLSA` records without requiring name resolution by sending the required information directly from the server to the client during the TLS handshake. However, this RFC is experimental and there are no plans to implement it in Firefox [18]. Although Aishwarya *et al.* [19] investigated the performance of DANE on HTTPS based on end-user bandwidth and central processing unit (CPU) utilization, they did not evaluate the increase in latency due to DANE.

To bridge this knowledge gap, this study investigated the extent to which DANE affects web performance in terms of load time, which is directly related to end-user QoE. The load time represents the time between the start of page loading and the execution of the onLoad event [20], which is a document object model (DOM) event that is executed once page loading is complete. We developed a tool to measure webpage load time, called *DANEWebPerf*; this tool comprises a web browser, DNS cache server, and proxy that enables DANE. We prepared four measurement scenarios, namely, with or without DNS caching and with or without DANE, and measured the webpage load time in each scenario at two locations: Asia Pacific (Tokyo) and Europe (Frankfurt). The results showed that using DANE with DNS caching reduced the average webpage load time by 706.9 and 105.4 ms in Tokyo and Frankfurt, respectively, compared with its use without DNS caching. Additionally, the average load time with DANE increased by 1190.9 ms at most. Therefore, we conclude that DANE does not have a considerable negative impact on the QoE of end users.

The remainder of this paper is organized as follows. Section II provides an overview of DNSSEC, DANE, and `TLSA` records, and presents related work. Section III explains the proposed method for measuring web performance using

DANE, while Section IV presents the experimental results. Finally, Section V concludes the paper by summarizing our findings and presenting directions for future research.

II. Background

A. DNSSEC

The DNSSEC [5]–[7] feature ensures that a DNS response is created by the legitimate owner of a domain (origin authentication) and the resource record is not altered or missing in the DNS response (integrity assurance). After receiving a resource record signature (`RRSIG`) record containing the digital signature attached to the requested resource record, the DNSSEC validator (e.g., recursive DNS resolver) retrieves the DNS public key (`DNSKEY`) record (containing the public key) and the delegation signer (`DS`) record (containing the hash value of the public key) from the relevant authoritative DNS servers. The validator verifies the public key for DNSSEC validation based on the public key of the DNS root zone, which is the trust anchor, and verifies the digital signature in the `RRSIG` record.

Migault *et al.* [21] evaluated the performance impacts of DNSSEC and demonstrated that DNSSEC validation causes an increase in latency for processing DNS queries and responses in the recursive DNS resolver. Subsequently, Huston *et al.* [22] noted that additional DNS queries for `DNSKEY` and `DS` records during DNSSEC validation increase latency. However, they also noted that DNS caching can decrease the latency.

B. DANE/TLSA

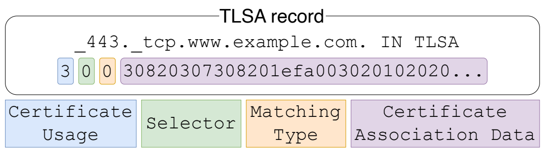

TLSA record
_443._tcp.www.example.com. IN TLSA
3
0
0
30820307308201efa003020102020...
Certificate Usage
Selector
Matching Type
Certificate Association Data

Fig. 1. Structure of a `TLSA` record.

DANE [4] is a DNS-based authentication technique that uses a `TLSA` record to associate a domain name with the information used for server certificate validation. The `TLSA` record must be protected using DNSSEC. The fully qualified domain name (FQDN) used to retrieve the `TLSA` record is defined as *_(port_number)*`.`*_(protocol)*`.`*(FQDN)*, where *(port_number)*, *(protocol)*, and *(FQDN)* denote the port number, protocol, and FQDN of the server, respectively. Fig. 1 shows the `TLSA` record for an HTTPS server running on Port 443 of `www.example.com`. It comprises the following four fields:

- `Certificate Usage`
  This field describes the server certificate validation methods and is represented by an integer between 0 and 3. Each method may be expressed as follows: `PKIX-TA(0)`, `PKIX-EE(1)`, `DANE-TA(2)`, and `DANE-EE(3)` [23] (Fig. 2). In `PKIX-TA(0)`, the `Certificate Association Data` represent the certificate information of the public CA that can be used to verify the end entity (EE) certificate (i.e., the server certificate). In `PKIX-EE(1)`, the `Certificate Association Data` indicate the information of the EE certificate signed by the public CA. In `DANE-TA(2)`, the `Certificate Association Data` indicate the information of the certificate of the private CA, which can introduce a new trust anchor. Finally, in `DANE-EE(3)`, the `Certificate Association Data` include the information of the EE certificate, and the domain owner can issue a valid certificate for the domain.
- `Selector`
  This field indicates the part of the server certificate to be compared with the `Certificate Association Data`. If the value is 0, the entire certificate is compared, whereas only the public key is compared when the value is 1.
- `Matching Type`
  This field indicates the format of the `Certificate Association Data` value, which is an integer between 0 and 2, where 0 denotes the original data, whereas the values of 1 and 2 denote the SHA-256 and SHA-512 hash values, respectively.
- `Certificate Association Data`
  This field stores the data to be compared with the certificate provided by the server.

C. Related Work

Zhu *et al.* [8] investigated the adoption of `TLSA` records for SMTP, HTTPS, and extensible messaging and presence protocol (XMPP) in generic top-level domains (gTLDs), such as `com` and `net`. Their dataset comprised `com` and `net` zone files [24] (as of December 3, 2014) obtained from the Internet Corporation for Assigned Names and Numbers. Among the 485k DNSSEC secured `com` and `net` zones investigated, they found only 997 `TLSA` records. Additionally, they discovered that the adoption rates of `TLSA` records were 50.7%, 39.5%, and 9.8% for SMTP, HTTPS, and XMPP, respectively, indicating that DANE was mainly used for SMTP and HTTPS.

Lee *et al.* [25] investigated the use of DANE for SMTP and revealed that although DANE adoption has gradually increased, its operation had some issues (e.g., 36% `TLSA` records were invalid owing to DNSSEC validation failure, whereas 14% server certificates did not match the `TLSA` records). Additionally, Lee *et al.* [26] explored the causes of DANE operation problems and identified DNSSEC deployment failure and public key changes due to automatic certificate issuance as the primary causes.

We investigated the performance impacts on clients using DANE for web browsing by focusing on webpage load time, which directly affects the QoE of end users. To the best of our knowledge, such effects have not been investigated in previous studies.

III. Measurement Method

A. Performance Metric

To investigate the impact of DANE on the web performance for the client, we used webpage load time—which is directly

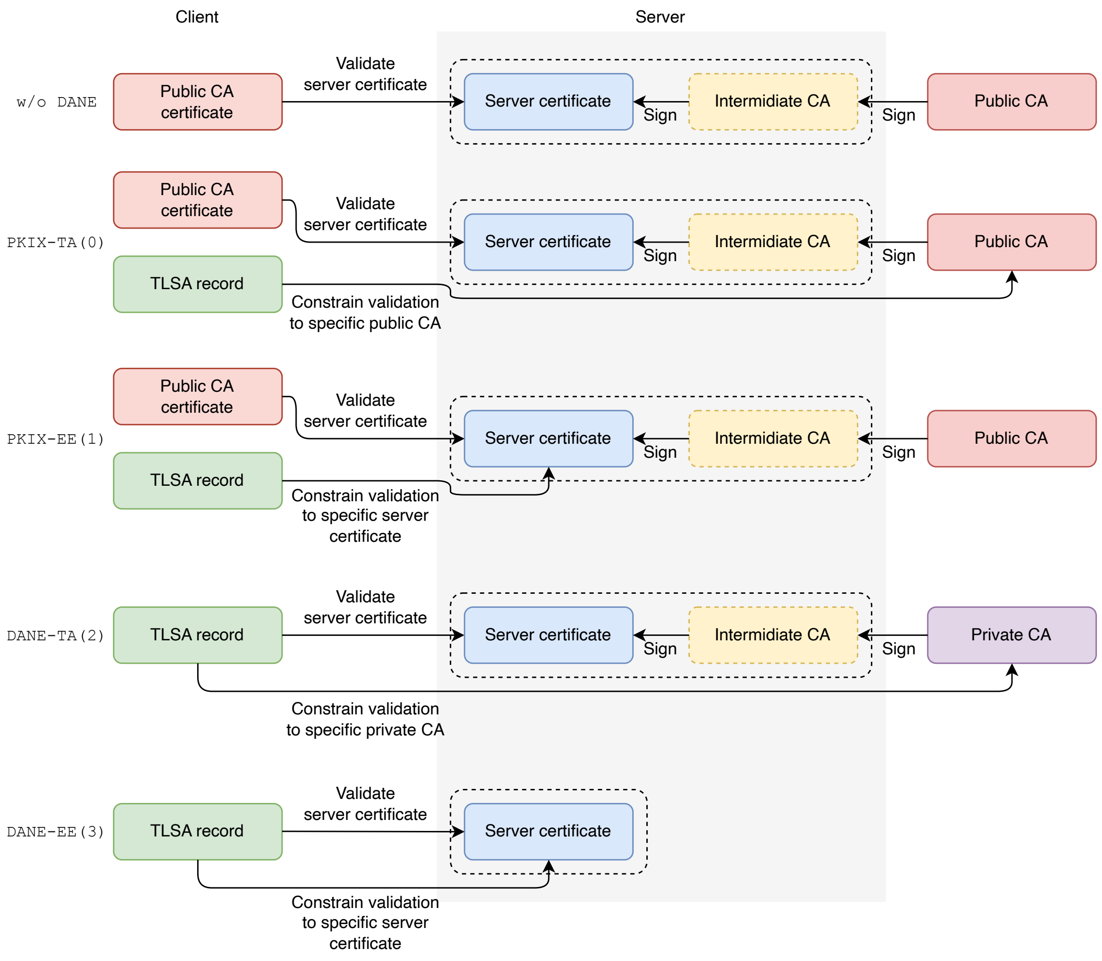

Client
Server
w/o DANE
Public CA certificate
Validate server certificate
Server certificate
Sign
Intermidiate CA
Sign
Public CA
PKIX-TA(0)
Public CA certificate
TLSA record
Validate server certificate
Constrain validation to specific public CA
PKIX-EE(1)
Constrain validation to specific server certificate
DANE-TA(2)
TLSA record
Private CA
Constrain validation to specific private CA
DANE-EE(3)
Constrain validation to specific server certificate

Fig. 2. Server certificate validation method with the DANE (inspired by Fig. 1 in Zhu *et al.* [8]).

related to QoE—as the performance metric. We obtained the loading time from the HTTP archive (HAR) file [27], which is a JSON-formatted archive file containing detailed information regarding the webpage loaded by the web browser and can be obtained from most major web browsers. The webpage load time can be extracted from the onLoad event within the pageTimings object in the HAR file. It represents the time (in ms) between the start of page loading and the execution of the onLoad event [20], a DOM event executed once the page loading is complete.

B. *DANEWebPerf*

We developed a tool called *DANEWebPerf*[1] to measure the webpage load time described in Section III-A. This tool can measure the load time under four conditions: with or without DNS caching and with or without DANE. *DANEWebPerf* comprises Firefox (web browser), Unbound (DNS cache server), and Let's DANE [14] (proxy that enables DANE use) (Fig. 3), with each independent Docker container running on the same Docker network. *DANEWebPerf* accesses the website associated with the FQDN to be measured in the headless mode where the browser window is not displayed.

[1]https://github.com/yagikota/danewebperf

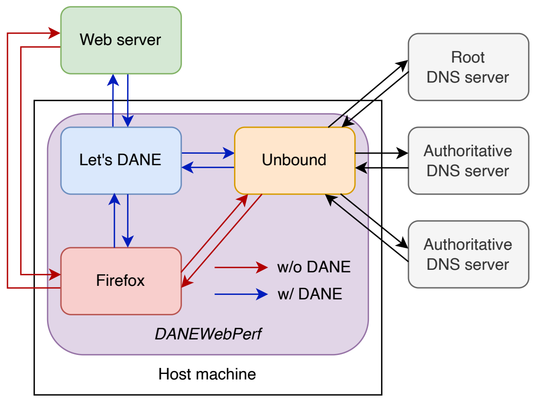


Fig. 3. Overview of *DANEWebPerf*.

It contains three Docker containers, which are described as follows.

- Firefox
  We used the Docker image employed by Houn-

sel *et al.* [28]. Within the container, Firefox 67.0.1 was run in the headless mode and controlled via Selenium 3.141.0. The container allows setting a timeout value for the webpage load time, which was set to 30 s in this study. When DANE is not used, the internet protocol (IP) address of the DNS cache server container is passed to the Firefox container at startup. When DANE is used, the IP address of the Let's DANE container is passed to the Firefox container at startup, which reads the self-signed certificate generated by Let's DANE as the root certificate. The reason is described below.

- Unbound
  We used the existing Docker image [29] as the DNS cache server. The container employed Unbound 1.19.1 with DNSSEC validation. The caching function could be controlled by the environment variables provided. To enable the caching function, in this study, the minimum time-to-live (TTL) for the positive cache, maximum TTL for the positive cache, and maximum TTL for the negative cache were set to 86400, 86400, and 3600 s, respectively. These values were sufficiently large to generate cache hits because a maximum of 30 s was required to obtain the webpage load time for a single website owing to the timeout setting. To disable the caching function, the three values were set to 0 s to ensure that no cache hits occurred.
- Let's DANE
  Let's DANE is a lightweight proxy that enables the use of DANE in applications such as web browsers. At the time this paper was written, only the server certificate verification method of `DANE-EE(3)` was available. Let's DANE queries the DNS for `TLSA` records and verifies server certificates using DANE on behalf of the web browser. If the verification is successful, Let's DANE generates a new server certificate. Firefox imports the self-signed certificate issued by Let's DANE as a root certificate; thus, Firefox can use the new server certificate received from Let's DANE to access the website. Additionally, Firefox uses the normal root certificate to access websites whose domains do not support DANE.

C. Measurement Scenarios

We considered the following four measurement scenarios:

- (w/o DNS cache, w/o DANE)
  In this scenario, the DNS cache is disabled and DANE is not used. This is based on the assumption that the user accesses infrequently accessed websites via conventional web browsing. The Firefox container sends a DNS query to the Unbound container to access the desired website.
- (w/o DNS cache, w/ DANE)
  In this scenario, DANE is used but the DNS cache is disabled. This is based on the assumption that the user accesses infrequently accessed websites when browsing with DANE. The Firefox container uses the Let's DANE container as a proxy and sends a DNS query to the Unbound container to access the desired website.
- (w/ DNS cache, w/o DANE)
  In this scenario, the DNS cache is enabled and DANE is not used. A case is assumed wherein a frequently accessed website is accessed using conventional web browsing. The Firefox container makes a DNS query to the Unbound container to store the DNS cache and access the desired website. Subsequently, another Firefox container is launched in the same Docker network and sends a DNS query to the Unbound container to access the same website. Thus, the Firefox container can use the DNS cache, and the webpage load time with the container is obtained.
- (w/ DNS cache, w/ DANE)
  In this scenario, both DANE and the DNS cache are enabled. This is assumed to be a case wherein frequently accessed websites are accessed by web browsing with DANE. The Firefox container uses the Let's DANE container as a proxy to send DNS queries to the Unbound container to access the desired website and store the DNS cache. Subsequently, another Firefox container and a Let's DANE container are launched in the same Docker network, and a DNS query is sent to the Unbound container to access the same website. Thus, the Firefox container can use the DNS cache, and the webpage load time with the container is obtained.

Notably, (w/o DNS cache, w/o DANE), (w/o DNS cache, w/ DANE), (w/ DNS cache, w/o DANE), and (w/ DNS cache, w/ DANE) are used to represent the four scenarios in Section IV.

IV. Experiments

A. Hardware Setup and Locations

We ran *DANEWebPerf* on Amazon EC2 instances to measure the webpage load time. Regarding the hardware setup for the instances, we configured the Amazon Machine Image, model, number of virtual CPUs, memory size, network bandwidth, and elastic block store (EBS)[2] bandwidth to Debian12, c5.4xlarge, 16, 32 GiB, up to 10 Gbps, and 4750 Mbps, respectively.

Additionally, *DANEWebPerf* was deployed in Asia Pacific (Tokyo) and Europe (Frankfurt).

B. Dataset

The dataset for this study was required to contain FQDNs for which DANE is valid. Therefore, we obtained a list of FQDNs from the Hall of Fame [30] that satisfied all web security metrics [31] from Internet.nl [32], a security testing tool for web and mail services. One of the metrics involved verified the validity of DNSSEC, which is a prerequisite for enabling the DANE (see Section II-B). As of March 28, 2024, the FQDN list included 36321 FQDNs. We then extracted the `TLSA` records for HTTPS from the list using ZDNS [33]. The number of FQDNs registering `TLSA` records for HTTPS was 5130, and the number of `TLSA` records was 7482. Fig. 4 shows the classification of the obtained `TLSA` records. Among

[2]EBS is a block storage service attached to an Amazon EC2 instance.

the 7482 `TLSA` records, 4680 were for `DANE-EE(3)`; this resulted in 4022 FQDNs. As indicated in Section III-B, Let's DANE supports only `DANE-EE(3)`; therefore, this list of 4022 FQDNs is the dataset used in Section IV-C. Fig. 5 shows the number of FQDNs for the top 10 TLDs in the dataset.

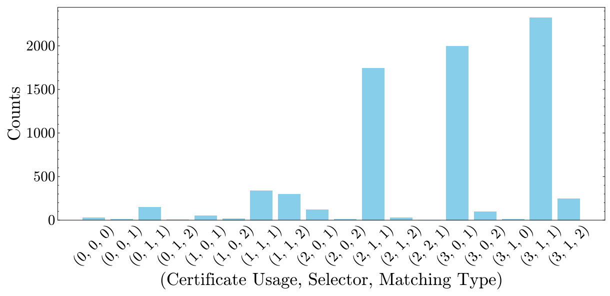


Fig. 4. Classification of the obtained `TLSA` records.

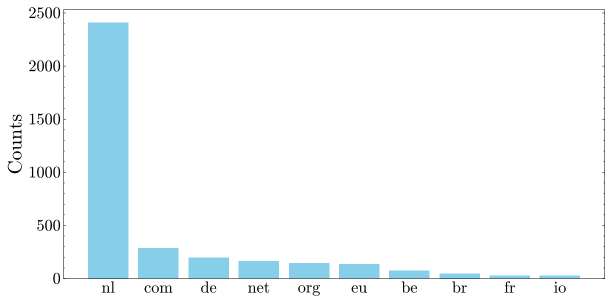


Fig. 5. Number of FQDNs for the top 10 TLDs in our dataset.

C. Evaluations

We launched 10 Amazon EC2 instances each in Tokyo and Frankfurt on April 21, 2024, and ran *DANEWebPerf* based on the four measurement scenarios and dataset generated in Section IV-B. Because we confirmed an execution error of *DANEWebPerf* in the 6th instance in Frankfurt, we removed the data obtained in the 6th instance in Tokyo, and finally performed evaluations on the basis of the data obtained from nine instances at each location. Loading a webpage involves generating HTTP requests and obtaining the HTTP responses. The first HTTP request was generated using the dataset generated in Section IV-B. Even if the FQDN of the first HTTP request supports DANE, there is no guarantee that those of future HTTP requests generated by the HTTP responses support it. Fig. 6 shows the classification of webpage load statuses. In the legend, "Success w/o DANE" indicates that the page was successfully loaded without using DANE; "Success w/ DANE" implies that the page was successfully loaded using DANE; and "Success w/o DANE and w/ DANE" suggests that the page was successfully loaded, but there was a mixture of FQDNs that supported DANE and those that did not. Finally, "Failure" indicates that the page failed to load (i.e., the HAR file was not generated). Fig. 6 shows different percentages based on location because we set a timeout value of 30 s for webpage load time, as described in Section III-B. Section IV-C1 focuses on the Success w/o DANE and Success w/ DANE data and presents the experimental results for FQDNs that successfully loaded the page in all four measurement scenarios across nine instances in each location. While the experiments were conducted across nine instances in both Tokyo and Frankfurt, some FQDNs were excluded from the evaluations if the FQDNs did not complete all scenarios. Section IV-C2 presents the evaluations, except for the cases of webpage load failures (i.e., the evaluations are for the Success w/o DANE, Success w/ DANE, and Success w/o DANE and w/ DANE data). The purpose of Section IV-C1 is to evaluate webpage load time using a dataset of websites that fully implement DANE. In contrast, Section IV-C2 evaluates the load time for all successfully loaded pages. Thus, Section IV-C1 represents a scenario where DANE is fully implemented, while Section IV-C2 reflects the current state of the web with DANE.

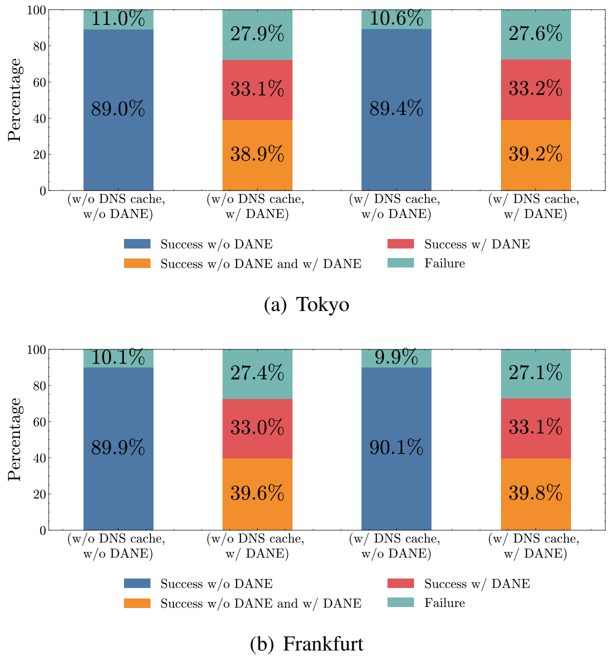
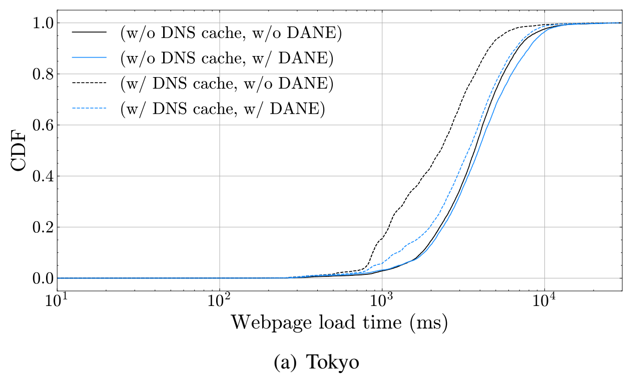


Fig. 6. Classification of the webpage load statuses.

1) Evaluations with Success w/o DANE and Success w/ DANE Data: Fig. 7 shows the cumulative distribution functions (CDFs) of the webpage load time at the two locations, and Table I presents the statistics of the webpage load time. When the DNS cache was used and DANE was enabled, the average webpage load time was reduced by 706.9 and 105.4 ms in Tokyo and Frankfurt, respectively, compared with the case where the DNS cache was not used and DANE was enabled. This is because the DNS cache eliminates the latency caused by DNSSEC validation. Additionally, the average webpage load time with DANE increased by 1190.9 ms at most. Although Let's DANE incurs the overhead of creating a new server certificate upon successful server certificate validation by DANE, as described in Section III-B, we believe that there



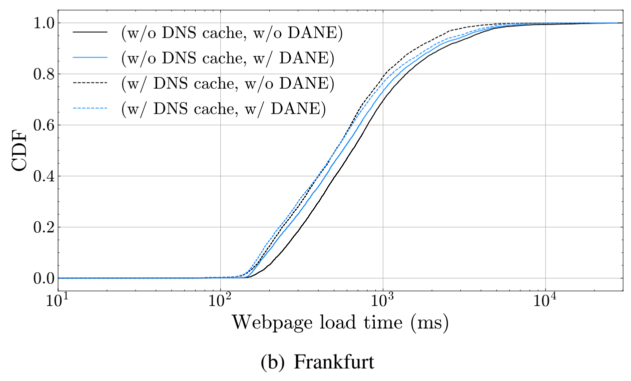
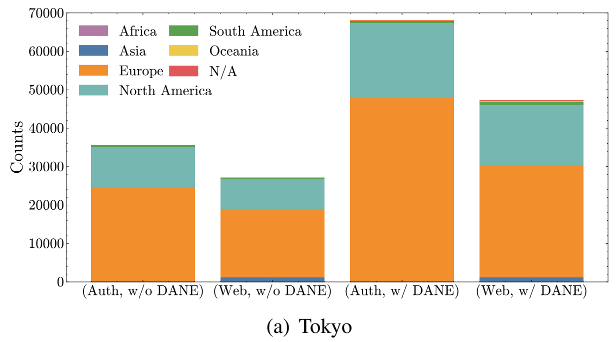


Fig. 7. Plots of CDFs of webpage load time.

TABLE I
STATISTICS OF THE WEBPAGE LOAD TIME.

Measurement scenario	Tokyo				Frankfurt			
	Average	90th percentile	95th percentile	99th percentile	Average	90th percentile	95th percentile	99th percentile
(w/o DNS cache, w/o DANE)	4121.6 ms	6869.2 ms	8238.2 ms	12655.2 ms	1114.5 ms	2309.0 ms	3598.4 ms	7096.3 ms
(w/o DNS cache, w/ DANE)	4472.8 ms	8000.2 ms	9333.5 ms	12805.2 ms	952.2 ms	2071.9 ms	3232.0 ms	5902.0 ms
(w/ DNS cache, w/o DANE)	2575.0 ms	4546.0 ms	5444.1 ms	8984.7 ms	758.0 ms	1614.0 ms	2256.4 ms	3978.3 ms
(w/ DNS cache, w/ DANE)	3765.9 ms	6610.2 ms	7758.6 ms	10524.6 ms	846.8 ms	1914.9 ms	2901.2 ms	4890.0 ms

is no significant impact on the QoE of end users even during web browsing with DANE using such plug-ins. Web browsers that natively support TLS DNSSEC chain extension [17] could achieve a shorter webpage load time.



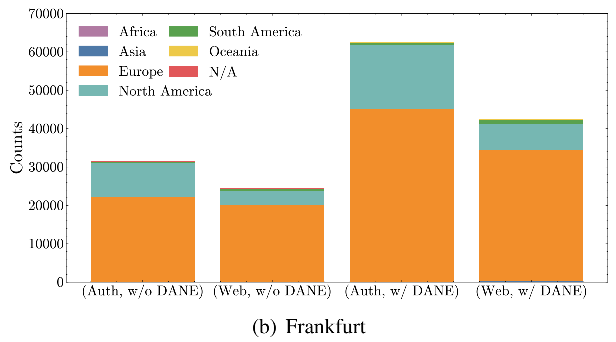
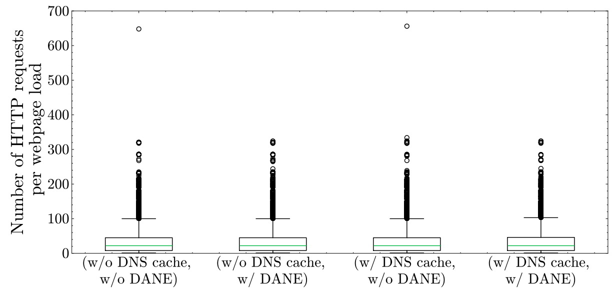


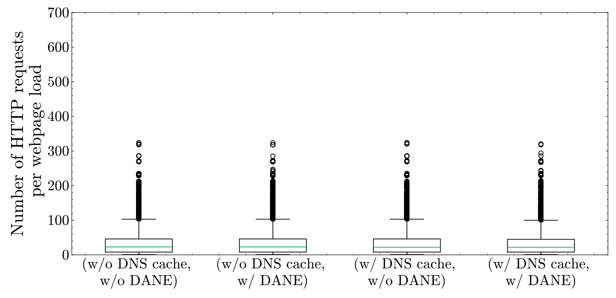
Fig. 8. Number of IP addresses of the authoritative DNS and web servers in (w/o DNS cache, w/o DANE) and (w/o DNS cache, w/ DANE) and their IP locations.

Fig. 7 and Table I indicate that the webpage load time tends to be shorter in Frankfurt than in Tokyo. We used IP location API [34] to investigate whether the geographical location of the authoritative DNS and web servers affects the webpage load time. We collected the IP addresses of the authoritative DNS and web servers observed in the (w/o DNS cache, w/o DANE) and (w/o DNS cache, w/ DANE) cases by extracting the DNS responses from the authoritative DNS servers of all FQDNs saved in the HAR files. Fig. 8 shows the number of IP addresses of the authoritative DNS and web servers observed in the (w/o DNS cache, w/o DANE) and (w/o DNS cache, w/ DANE) cases along with their IP locations. In the figure, Auth and Web represent the authoritative DNS and web servers, respectively, and are divided into cases wherein DANE is used and not used. The legend represents each region and N/A indicates IP addresses such as those where no region was assigned. Fig. 8 illustrates that the authoritative DNS and web servers were mostly located in Europe; therefore, the webpage load time tended to be shorter in Frankfurt than in Tokyo. This is relevant to our dataset obtained from Internet.nl [32], wherein most FQDNs include TLDs of `nl` (see Fig. 5).

Fig. 9 shows the number of HTTP requests per webpage load. The median for all measurement scenarios at each location was 22 or 23. Additionally, the outliers result in a longer webpage load time in Fig. 7 and Table I. These outliers originate from `www.bb-schipborg.nl`. When measuring the load time for this site, there were cases wherein the JavaScript scripts included in the HTML were executed and cases wherein they were not. When the scripts were executed, the number of HTTP requests increased, resulting in a longer load time. Conversely, when the scripts were not executed, the



(a) Tokyo



(b) Frankfurt

Fig. 9. Number of HTTP requests per webpage load.

load time was shorter.

2) Evaluations with Success w/o DANE, Success w/ DANE, and Success w/o DANE and w/ DANE Data: Figs. 10, 11, and 12 and Table II present the measured results (excluding the cases of webpage load failures). The characteristics of the webpage load time are similar to those obtained in Section IV-C1. The IP locations of the authoritative DNS and web servers shown in Fig. 11 differ from those shown in Fig. 8, indicating that the IP locations of FQDNs for the webpages that do not use DANE are in Asia and North America.

Fig. 13 illustrates the relationship between the number of HTTP requests per webpage load and that using DANE per webpage load in the (w/ DNS cache, w/ DANE) and (w/o DNS cache, w/ DANE) cases, excluding the cases of webpage load failures. Section IV-C1 shows the results obtained when these two numbers are the same (i.e., Success w/ DANE in Fig. 6). Fig. 13 shows that 70.2 and 70.8% of webpages were loaded by more than 80% of DANE-enabled HTTP requests; however, 2.7 and 2.8% of webpages were loaded using DANE only for the first HTTP request in Tokyo and Frankfurt, respectively.

D. Limitations

This study had three limitations. First, we focused only on the server certificate verification method of `DANE-EE(3)`, as described in Section III-B. However, different webpage load times may be obtained when using `DANE-TA(2)`. Compared with `DANE-EE(3)`, `DANE-TA(2)` may experience additional latency due to the involvement of intermediate CAs, as shown in Fig. 2.

Second, our dataset included many FQDNs with TLDs of `nl`; therefore, the dataset was skewed. Additionally, we considered using the Chrome User Experience Report [35], a dataset that includes the top one million URLs that reflect how Chrome users access popular pages on the web (https://github.com/zakird/crux-top-lists/blob/main/data/global/202401.csv.gz) to collect `TLSA` records for HTTPS. However, as of February 26 and 27, 2024, we could extract only 163 `TLSA` records for HTTPS from the dataset, and it contained only 104 `TLSA` records for `DANE-EE(3)`.

Third, we did not consider the breakdown of the webpage load time. The webpage load time comprises several factors, such as resource record fetching, DNSSEC validation, server certificate verification by DANE, and HTTPS fetching. The analysis of this breakdown will allow the identification of the bottlenecks in various aspects of the webpage load time and clarify the factors responsible for the weak negative impact of DANE for the web on the QoE of end users.

These limitations should be addressed in future research.

V. Conclusion

Although DANE improves the trustworthiness of PKI by issuing and verifying CA-independent certificates, the problem of latency increment caused by DNSSEC limits the adoption of DANE for HTTPS. Latency is a major problem for web services because it has a significant negative impact on the QoE of end users during website access. This study investigated the impact of DANE on web performance using *DANEWebPerf*, a tool that measures webpage load time. The increase in the average webpage load time with DANE was at most 1190.9 ms. Therefore, we conclude that DANE for the web does not have a considerable negative impact on the QoE of end users.

Future research should focus on measuring the webpage load time using several server certificate verification methods with DANE, and employ various datasets for measurements and analysis of the breakdown of webpage load times. It will also be interesting to measure webpage load times using web browsers that support the TLS DNSSEC chain extension once the function becomes extensively supported.

References

[1] S. Boeyen, S. Santesson, T. Polk, R. Housley, S. Farrell, and D. Cooper, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 5280, May 2008, doi: 10.17487/RFC5280.

[2] "Comodo Report of Incident - Comodo detected and thwarted an intrusion on 26-MAR-2011," Accessed: Feb. 8, 2024. [Online]. Available: https://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html

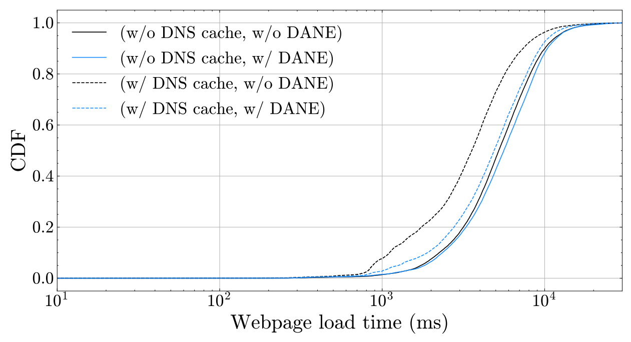
[3] "Hackers spied on 300,000 Iranians using fake Google certificate — Computerworld," Accessed: Feb. 8, 2024. [Online]. Available: https://www.computerworld.com/article/2510951/hackers-spied-on-300-000-iranians-using-fake-google-certificate.html

[4] P. E. Hoffman and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA," RFC 6698, Aug. 2012, doi: 10.17487/RFC6698.

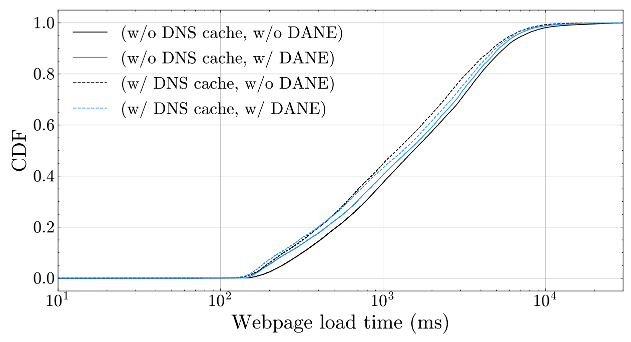
[5] S. Rose, M. Larson, D. Massey, R. Austein, and R. Arends, "DNS Security Introduction and Requirements," RFC 4033, Mar. 2005, doi: 10.17487/RFC4033.

[6] ——, "Resource Records for the DNS Security Extensions," RFC 4034, Mar. 2005, doi: 10.17487/RFC4034.

[7] ——, "Protocol Modifications for the DNS Security Extensions," RFC 4035, Mar. 2005, doi: 10.17487/RFC4035.



(a) Tokyo

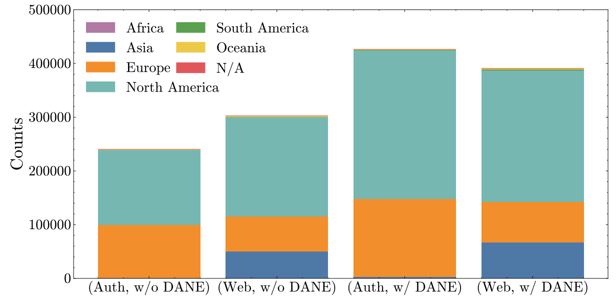


(b) Frankfurt

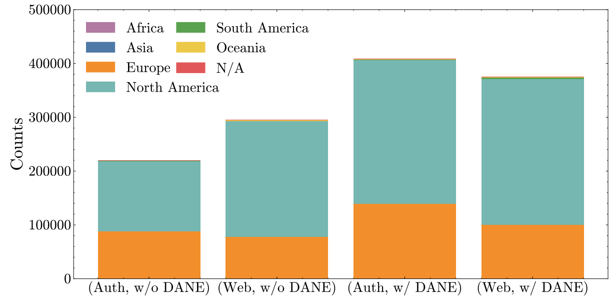
Fig. 10. Webpage load times excluding the cases of webpage load failures.

TABLE II
STATISTICS OF THE WEBPAGE LOAD TIME EXCLUDING THE CASES OF WEBPAGE LOAD FAILURES.

Measurement scenario	Tokyo				Frankfurt			
	Average	90th percentile	95th percentile	99th percentile	Average	90th percentile	95th percentile	99th percentile
(w/o DNS cache, w/o DANE)	5875.4 ms	10041.6 ms	12081.6 ms	17753.7 ms	2477.3 ms	5569.2 ms	7233.6 ms	13203.7 ms
(w/o DNS cache, w/ DANE)	6193.2 ms	10402.0 ms	12301.5 ms	18220.8 ms	2240.2 ms	5167.0 ms	6548.1 ms	10545.3 ms
(w/ DNS cache, w/o DANE)	4065.0 ms	7367.0 ms	9107.3 ms	14420.6 ms	1973.9 ms	4734.2 ms	6110.2 ms	9319.0 ms
(w/ DNS cache, w/ DANE)	5402.7 ms	9358.0 ms	10932.0 ms	15484.8 ms	2104.6 ms	4926.5 ms	6299.0 ms	10103.1 ms

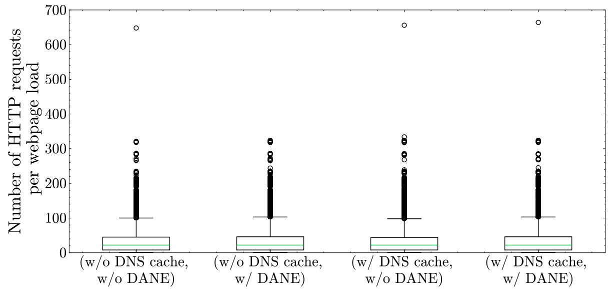


(a) Tokyo

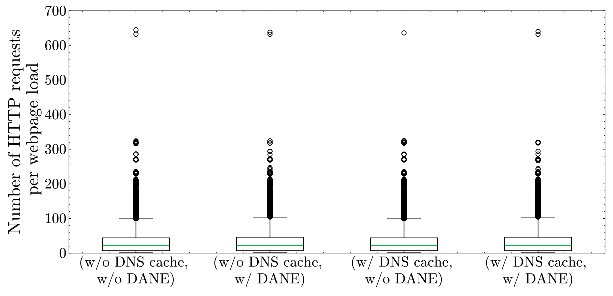


(b) Frankfurt

Fig. 11. Number of IP addresses of the authoritative DNS and web servers in the (w/o DNS cache, w/o DANE) and (w/o DNS cache, w/ DANE) cases and their IP locations, excluding the cases of webpage load failures.



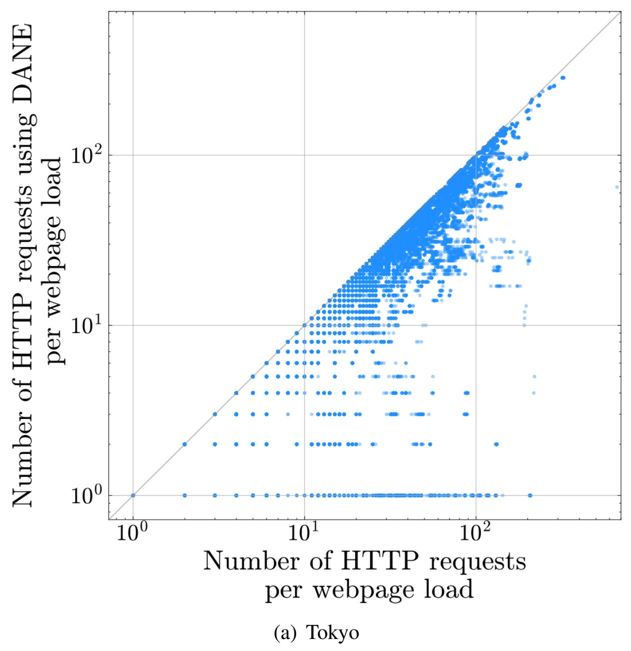
(a) Tokyo



(b) Frankfurt

Fig. 12. Number of HTTP requests per webpage load excluding the cases of webpage load failures.

[8] L. Zhu, D. Wessels, A. Mankin, and J. Heidemann, "Measuring DANE TLSA deployment," in *Proc. Traffic Monitoring and Analysis*, 2015, pp. 219–232, doi: 10.1007/978-3-319-17172-2_15.
[9] "DNSSEC-DANE-Deployment-Statistics," Accessed: Feb. 8, 2024. [Online]. Available: https://stats.dnssec-tools.org/
[10] "Implementing Inbound SMTP DANE with DNSSEC for Exchange Online Mail Flow - Microsoft Community Hub," Accessed: Feb. 8, 2024. [Online]. Available: https://techcommunity.microsoft.com/t5/exchange-team-blog/implementing-inbound-smtp-dane-with-dnssec-for-exchange-online/ba-p/3939694
[11] "1479423 - support for DNSSEC/DANE/TLSA validation," Accessed: Feb. 8, 2024. [Online]. Available: https://bugzilla.mozilla.org/show_bug



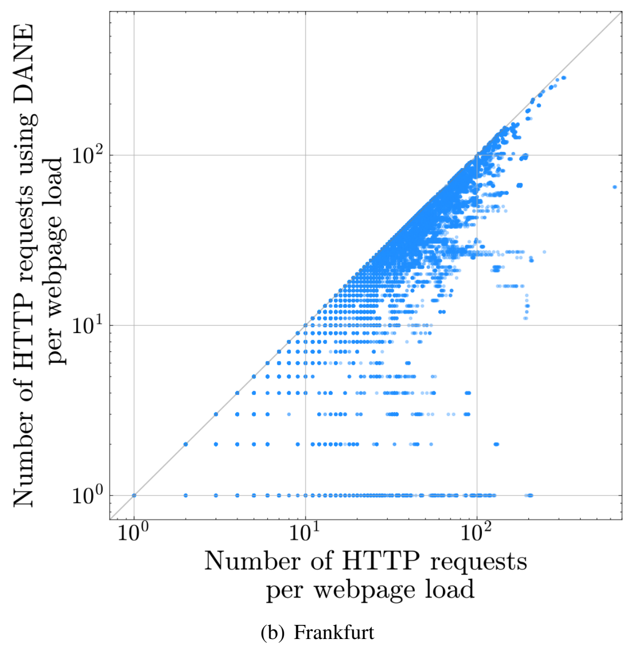


Fig. 13. Relationship between the number of HTTP requests per webpage load and that using DANE per webpage load in the (w/ DNS cache, w/ DANE) and (w/o DNS cache, w/ DANE) cases excluding the cases of webpage load failures.

.cgi?id=1479423

[12] "DNSSEC and DANE — Can I use... Support tables for HTML5, CSS3, etc," Accessed: Feb. 8, 2024. [Online]. Available: https://caniuse.com/dnssec

[13] "DNSSEC/TLSA Validator," Accessed: Feb. 8, 2024. [Online]. Available: https://www.dnssec-validator.cz/

[14] "buffrr/letsdane: Let's DANE is an experimental way to enable the use of DANE/TLSA in browsers and other apps using a lightweight proxy." Accessed: Feb. 8, 2024. [Online]. Available: https://github.com/buffrr/letsdane

[15] "ImperialViolet - Why not DANE in browsers," Accessed: Feb. 8, 2024. [Online]. Available: https://www.imperialviolet.org/2015/01/17/notdane.html

[16] S. Huque and V. Dukhovni, "DANE Overview," Jul. 2022, Accessed: Feb. 22, 2024. [Online]. Available: https://indico.dns-oarc.net/event/43/contributions/928/attachments/901/1648/dane-overview-shumon.pdf

[17] V. Dukhovni, S. Huque, W. Toorop, P. Wouters, and M. Shore, "TLS DNSSEC Chain Extension," RFC 9102, Aug. 2021, doi: 10.17487/RFC9102.

[18] "672600 - Use DNSSEC/DANE chain stapled into TLS handshake in certificate chain validation," Accessed: Mar. 15, 2024. [Online]. Available: https://bugzilla.mozilla.org/show_bug.cgi?id=672600

[19] C. Aishwarya, R. M. A, S. Hosmani, M. Sannidhan, B. Rajendran, K. Chandrasekaran, and B. Bindhumadhava, "DANE: An inbuilt security extension," in *Proc. 2015 International Conference on Green Computing and Internet of Things (ICGCIoT)*, 2015, pp. 1571–1576, doi: 10.1109/ICGCIoT.2015.7380717.

[20] "Window: load event - Web APIs — MDN," Accessed: Feb. 8, 2024. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Window/load_event

[21] D. Migault, C. Girard, and M. Laurent, "A performance view on DNSSEC migration," in *Proc. 2010 International Conference on Network and Service Management*, 2010, pp. 469–474, doi: 10.1109/CNSM.2010.5691275.

[22] G. Huston and G. Michaelson, "Measuring DNSSEC Performance," May 2013, Accessed: Feb. 24, 2024. [Online]. Available: https://www.dotnxdomain.net/ispcol/2013-05/dnssec-performance.pdf

[23] V. Dukhovni and W. Hardaker, "The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance," RFC 7671, Oct. 2015, doi: 10.17487/RFC7671.

[24] "Centralized Zone Data Service," Accessed: Feb. 8, 2024. [Online]. Available: https://czds.icann.org/home

[25] H. Lee, A. Gireesh, R. van Rijswijk-Deij, T. T. Kwon, and T. Chung, "A longitudinal and comprehensive study of the DANE ecosystem in email," in *Proc. 29th USENIX Security Symposium (USENIX Security 20)*, Aug. 2020, pp. 613–630.

[26] H. Lee, M. I. Ashiq, M. Müller, R. van Rijswijk-Deij, T. T. Kwon, and T. Chung, "Under the hood of DANE mismanagement in SMTP," in *Proc. 31st USENIX Security Symposium (USENIX Security 22)*, Aug. 2022, pp. 1–16.

[27] "HTTP Archive (HAR) format," Accessed: Feb. 8, 2024. [Online]. Available: https://w3c.github.io/web-performance/specs/HAR/Overview.html

[28] A. Hounsel, K. Borgolte, P. Schmitt, J. Holland, and N. Feamster, "Comparing the effects of DNS, DoT, and DoH on web performance," in *Proc. The Web Conference 2020*, 2020, pp. 562–572, doi: 10.1145/3366423.3380139.

[29] "Image Layer Details - secns/unbound:1.19.1 — Docker Hub," Accessed: Feb. 8, 2024. [Online]. Available: https://hub.docker.com/layers/secns/unbound/1.19.1/images/sha256-a5b72cbfd7d4f8597eeb2c566a02324143dfe2f341ed188631bc0c3995003c8b?context=explore

[30] "Hall of Fame - Websites," Accessed: Feb. 8, 2024. [Online]. Available: https://internet.nl/halloffame/web/

[31] "About the website test," Accessed: Feb. 8, 2024. [Online]. Available: https://internet.nl/test-site/

[32] "Test for modern Internet Standards like IPv6, DNSSEC, HTTPS, DMARC, STARTTLS and DANE." Accessed: Feb. 8, 2024. [Online]. Available: https://internet.nl/

[33] "zmap/zdns: Fast CLI DNS Lookup Tool," Accessed: Feb. 8, 2024. [Online]. Available: https://github.com/zmap/zdns

[34] "IP Address Lookup — Geolocation," Accessed: May 15, 2024. [Online]. Available: https://www.iplocation.net/

[35] "zakird/crux-top-lists: Downloadable snapshots of the Chrome Top Million Websites pulled from public CrUX data in Google BigQuery." Accessed: Feb. 26, 2024. [Online]. Available: https://github.com/zakird/crux-top-lists