TLS 1.3 in Practice: How TLS 1.3 Contributes to the Internet

Hyunwoo Lee
Purdue University
West Lafayette, Indiana, USA
lee3816@purdue.edu

Doowon Kim
University of Tennessee, Knoxville
Knoxville, Tennessee, USA
doowon@utk.edu

Yonghwi Kwon
University of Virginia
Charlottesville, Virginia, USA
yongkwon@virginia.edu

ABSTRACT

Transport Layer Security (TLS) has become the norm for secure communication over the Internet. In August 2018, TLS 1.3, the latest version of TLS, was approved, providing improved security and performance of the previous TLS version. In this paper, we take a closer look at TLS 1.3 deployments in practice regarding adoption rate, security, performance, and implementation by applying temporal, spatial, and platform-based approaches on 687M connections.

Overall, TLS 1.3 has rapidly been adopted mainly due to third-party platforms such as Content Delivery Networks (CDNs) makes a significant contribution to the Internet. In fact, it deprecates vulnerable cryptographic primitives and substantially reduces the time required to perform the TLS 1.3 full handshake compared to the TLS 1.2 handshake. We quantify these aspects and show TLS 1.3 is beneficial to websites that do not rely on the third-party platforms. We also review Common Vulnerabilities and Exposures (CVEs) regarding TLS libraries and show that many of recent vulnerabilities can be easily addressed by upgrading to TLS 1.3. However, some websites exhibit unstable support for TLS 1.3 due to multiple platforms with different TLS versions or migration to other platforms, which means that a website can show the lower TLS version at a certain time or from a certain region. Furthermore, we find that most of the implementations (including TLS libraries) do not fully support the new features of TLS 1.3 such as downgrade protection and certificate extensions.

CCS CONCEPTS

• **Security and privacy** → **Web protocol security**.

KEYWORDS

TLS security, TLS 1.3, Measurement, Certificate, TLS vulnerability

ACM Reference Format:
Hyunwoo Lee, Doowon Kim, and Yonghwi Kwon. 2021. TLS 1.3 in Practice: How TLS 1.3 Contributes to the Internet. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3442381.3450057

1 INTRODUCTION

Transport Layer Security (TLS) [15, 37] has become the *de-facto* standard protocol for secure communications in web services such as online banking. As of October 2020, more than 90% of Internet traffic is communicated over TLS [20]. TLS has evolved from Secure Socket Layer (SSL) to its newest version, TLS 1.3, enhancing security and performance from its legacy versions [25]. Compared to TLS 1.2 [15], for instance, TLS 1.3 guarantees *perfect forward secrecy* by removing static RSA key exchanges. It also reduces the number of round-trips of the TLS handshake from two to one, aiming to improve the performance of the initial setup.

Due to the significant impact of TLS in the web ecosystem, there have been many studies aiming to understand various aspects of TLS. To name a few, Holz et al. [22] show the statistics of the TLS 1.3 usage and what boosts its deployment. Naylor et al. [33] and Felt et al. [19] investigate the use of HTTPS (HTTP over TLS) [36] in practice. Platon et al. [25] demonstrate how the TLS ecosystem *reacts* to high-profile security attacks. However, to the best of our knowledge, *the new TLS version*'s impact on the ecosystem has not been thoroughly studied. Since it has been more than two years since the TLS 1.3's approval (Aug. 10th, 2018), we believe it is time to analyze how adequately TLS 1.3 is deployed in practice as intended by design.

In this paper, we aim to look closely at the implications of TLS 1.3 deployment in practice, mainly focusing on *adoption*, *security*, *performance*, and *implementation*. Specifically, we collect TLS handshake messages targeting the Alexa top 1M websites on a daily basis for 837 days from North America (687M connections in total) to analyze how many websites adopt TLS 1.3 and what security benefits they obtain. Furthermore, we also evaluate the time required to perform a TLS handshake with TLS 1.3 websites (399K on Dec. 31st, 2020) from eight different regions to quantify performance gain by upgrading to TLS 1.3, compared with TLS 1.2. Overall, we conclude that TLS 1.3 makes a significant contribution to the Internet in many aspects, based on the following observations:

Adoption. The TLS 1.3 adoption rate is significantly faster than the previous versions of TLS. It took only 264 days for TLS 1.3 to be deployed by more than 15% of websites after IETF officially approves the protocol. It is remarkably faster than the adoption rate of TLS 1.2, which took around five years to achieve the same adoption rate (i.e., 15%) [4]. We find that third-party platforms (e.g., CDNs) are the main contributors to the high adoption rate at the early stage of TLS 1.3, as they have adopted the TLS 1.3 at once.

Security. TLS 1.3 adoption contributes to enhancing the overall security of the TLS ecosystem. However, we find that 19.1% of the TLS 1.3 adopted websites support TLS 1.3 unstably. The TLS versions of the sessions established with the websites are not always TLS 1.3 after adopting TLS 1.3. It varies depending on when a client accesses the websites or where a client connects to them. This unstable support may weaken a certain website's security since

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450057

a website can show the lower TLS version at a specific time or from a particular region. Therefore, stakeholders should carefully manage the TLS version both temporally and geographically while upgrading to TLS 1.3.

Performance. Our results indicate that the time taken for a TLS 1.3 full handshake is reduced compared to TLS 1.2 by 57.9% – 77.1% on average, depending on the regions. In particular, websites served on third-party platforms (e.g., *Cloudflare*) are often geographically located near clients, leading to 27.9% – 69.0% of the performance gains. On the other hand, websites running over cloud platforms (usually farther from the clients geographically) gain performance enhancements of up to 91.1%, which may motivate individual websites to upgrade to TLS 1.3 for more secure and faster web services.

Implementation. We inspect whether the new features of TLS 1.3 are enabled on server-side and client-side applications or implemented in TLS libraries. 98 (0.03%) of the TLS 1.3 websites *do not* support downgrade protection (details in §2 and §7.1). Furthermore, it takes 516 days after TLS 1.3 was approved when a web browser first check downgrade sentinels sent by servers. Many TLS libraries do not implement the parsing module for certificate extension fields introduced for certificate-related extensions such as signed certificate timestamps [27] and OCSP stapling [34].

The paper is organized as follows. We summarize the TLS handshake with new features in TLS 1.3 and our research topics (§2). Then, we describe what dataset we used in this paper and how we collected them (§3). Based on the dataset, we explain our results regarding adoption (§4), security (§5), performance (§6), and implementation (§7). We review related work (§8) and finalize this paper with concluding remarks (§9).

2 BACKGROUND & MOTIVATION

2.1 The TLS 1.3 Protocol

Transport Layer Security (TLS), the successor to Secure Socket Layer (SSL), was designed by *Netscape* in 1994. In the last decade, the latest TLS version 1.3 [37] has been deployed, addressing critical vulnerabilities of its predecessor (i.e., TLS 1.2 [15]), such as the BEAST and FREAK attacks [25]. The standardization work for TLS 1.3 began in August 2013 and was finished in August 2018 with security and performance improvements. This section provides a brief overview of TLS 1.3, focusing on the distinct differences from its predecessor, TLS 1.2.

Security Improvements of TLS 1.3. TLS 1.2 is vulnerable to man-in-the-middle attacks and downgrade attacks. For example, *POODLE* [11] exploits the CBC-mode padding vulnerability when falling back to SSL 3.0. To this end, TLS 1.3 introduces a downgrade protection mechanism. When clients negotiate a TLS 1.3 server with older TLS versions (or SSL 3.0), the TLS 1.3 server must include one of two predefined values (`DOWNGRD01` or `DOWNGRD00`) in *server random*, as a downgrade signal. This mechanism is similar to the `TLS_FALLBACK_SCSV` [32] that aims to protect a session from being downgraded due to the web browser's TLS fallback mechanism. TLS 1.3 also introduces certificate *extension* fields in the `Certificate` message to efficiently process certificate-related TLS extensions. Currently, RFC8446 describes signed certificate timestamps (SCTs) [27] and OCSP stapling [34] for the extensions, but is not limited to only them. Note that TLS implementations need to be updated to process the new `Certificate` message, even if the TLS implementations have functions related to SCTs and OCSP stapling.

Performance Improvements of TLS 1.3. TLS 1.3 reduces the two round-trip times (RTT) for a handshake down to only one RTT. Specifically, the `ClientHello` and `ServerHello` messages are combined with the key exchange messages in the second round-trip in TLS 1.2. Moreover, the *early_data* extension is introduced to resume a TLS session with the previously visited website without delay (so-called "0-RTT"). For resumed sessions, there is no handshake procedure before sending application data. It allows clients to send application data along with the first handshake message. TLS 1.2, by contrast, requires one RTT before sending application data.

2.2 Motivation

In this paper, we analyze TLS 1.3 deployment in practice comprehensively via measuring the real-world websites. We focus on the practice of TLS 1.3 deployment from four aspects, each of which is analyzed from temporal, spatial, and platform-based viewpoints.

Adoption. The first aspect is the overall trend of TLS 1.3 adoption on websites in the wild. We take a closer look at how many websites currently support TLS 1.3 and who leads the deployment of TLS 1.3 in practice. Furthermore, we want to know if there are any different phenomena in the TLS 1.3 adoption according to the Alexa rank or the platforms (e.g., CDNs). To this end, we raise the following research questions:

- How many websites currently support TLS 1.3? Specifically, are there any specific trends during the TLS 1.3 deployments?
- Who leads the TLS 1.3 deployments in practice? (e.g., top Alexa websites or third-party platforms?)

Security. One of the main goals of TLS 1.3 is to improve the security of TLS. Therefore, we investigate what security benefits that websites gain when they upgrade to TLS 1.3. Moreover, we see whether websites stably support TLS 1.3 during our observation period. For example, if we observe a website that supports TLS 1.3 disables it and falls back to a TLS 1.2 website, we aim to investigate the case to understand the reasons behind it. The research questions that we raise regarding security are as follows:

- How many vulnerable servers are reduced (or secured) during our observation period due to the TLS 1.3 upgrading?
- Do the websites in the wild stably support TLS 1.3?

Performance. Another essential goal of TLS 1.3 is to improve performance by streamlining the handshake process. We measure how much TLS 1.3 decreases the time required to complete a full handshake compared to TLS 1.2 across different regions. We also analyze which factors may accelerate or impede performance gain. Particularly, we raise the following research questions:

- How much performance gain do websites obtain by upgrading to TLS 1.3?
- Are the performance gains similar across the regions? Who is the particular beneficiary?

Implementation. The TLS 1.3 libraries should be correctly implemented for users to enable the benefits of TLS 1.3. We measure how properly TLS libraries, web servers, and client applications are prepared for the new features of TLS 1.3. In particular, we investigate the downgrade protection in the `ServerHello` message, and the certificate extensions including *signed_certificate_timestamp* (SCT) and *certificate_status* (a.k.a, OCSP stapling). Moreover, we review Common Vulnerabilities and Exposures (CVEs) to understand how TLS libraries are correctly implemented. To this end, we raise the following research questions.

- Have websites and TLS libraries been properly prepared for the new features of TLS 1.3?
- Are there any vulnerabilities of TLS libraries that are addressed by TLS 1.3 deployment?

3 DATA COLLECTION

In this section, we describe the datasets that we use to answer the research questions presented in §2.2. We make three types of datasets—*Security Parameters (D1)*, *Handshake Messages (D2)*, and *Platform Information (D3)*—using our client-side applications.[1]

Security Parameters (D1). To understand the adoption rate and the security impact of TLS 1.3, we collect two `hello` messages in the TLS protocol (`ClientHello` and `ServerHello`). Those `hello` messages are to negotiate the TLS version and other security parameters between endpoints. To collect this data, we implement a client-side application based on `OpenSSL-1.1.1a` that implements the officially approved TLS 1.3 protocol. Our client application sends `ClientHello` to the intended server and terminates the handshake right after receiving `ServerHello`, recording the two `hello` messages and the IP addresses of the target servers.

The collection is performed for each of the Alexa 1M websites on a daily basis from a machine with Intel Xeon E3 CPUs and 8GB RAM. We utilize a single snapshot of the Alexa 1M websites generated in April 2018 during our observation period, which is from Sept. 17th, 2018 to Dec. 31st, 2020 (837 days in total). Throughout our observation period, around 84% of the websites were consistently collected. There were network outages for 17 days, which are pruned out from the dataset.

Handshake Messages (D2). To analyze the TLS 1.3 features supported in the TLS 1.3 web servers (399K on Dec. 31st, 2020) and to compare the initial setup time of TLS 1.3 with that of TLS 1.2, we also collect both TLS 1.2 and TLS 1.3 full handshake messages while measuring the elapsed time to establish the session. The data is collected from AWS machines (2.3GHz CPUs and 8GB memory) in eight different regions—Eastern North America (Ohio), Western North America (California), South America (San Paulo), Western Europe (Paris), South Africa (Cape City), East Asia (Seoul), Southeast Asia (Mumbai), and Oceania (Sydney).

Platform Information (D3). To better understand who upgrades the TLS versions of the Alexa 1M websites and find any different trends due to the platforms, we categorize the TLS websites into two classes based on who is responsible for managing the TLS libraries. They can be defined as 1) *first-party responsibility* and 2) *third-party responsibility*. In the former, website owners are responsible for upgrading the TLS libraries since the websites are running over an infrastructure-as-a-service platform such as Amazon Web Services. We consider these websites as *first-party responsibility* (FPR). On the other hand, if the websites use a CDN network (e.g., Cloudflare) or a website builder (e.g., Squarespace) to deliver contents, the platform providers are responsible for managing the TLS libraries; in other words, the website owners (or administrators) are not responsible for it. We classify these websites as *third-party responsibility* (TPR).

To identify the two categories (i.e., FPR and TPR), we perform the following platform identification process, as shown in Figure 1.

First, as a preliminary step, we identify each website's IP addresses and the related organizations (of the IPs). We also prepare for a list of known TPR platforms, such as Cloudflare, with their publicly announced IP ranges.

Second, we consider a website as FPR if its domain name and the IP address owner are the same. Google is an example of FPR.

Third, we denote a website as TPR if the website runs over a platform included in the list of known TPR platforms.

Fourth, we check whether a website is running over an anycast infrastructure. Specifically, we identify the anycasting IP address by comparing i) the round-trip times between two vantage points and ii) the sum of the round-trip times from each vantage point to each domain, proposed in prior work [29]. If the latter is significantly smaller than the former (less than 50%), we conclude that the IP address might be used for anycasting, hence classified as TPR.

Finally, we refer to a website as TPR if the round-trip times between clients of eight different regions and the website accessed by more than one IP address are significantly low. Otherwise, we classify the website as FPR.

To this end, we identify 240,512 websites (60.34%) as FPR and 158,081 (39.66%) as TPR.

Ethical consideration. To minimize the ethical concerns during our data collection, we restrict the number of requests we send to the public servers. Specifically, only one TCP handshake is performed per domain once a day. TLS `hello` messages were exchanged with our client, which is trivial.

4 TLS 1.3 ADOPTION

In this section, we first measure the adoption ratio of TLS 1.3 among Alexa top 1M sites. Then, we analyze which factors affect the adoption of TLS 1.3, concluding that it is mainly led by TPR platforms such as CDNs and web hosting companies since they can upgrade TLS libraries simultaneously.

TLS 1.3 Adoption Rate Over Time. We find that the ratio of TLS 1.3 adoption is continuously increasing—from 11.78% on Sept. 17th, 2018 to 48.09% on Dec. 31st, 2020 as shown in Figure 3. Note that the TLS 1.3 adoption ratio has increased at a substantially higher rate than the legacy TLS versions. In particular, it takes only 264 days (Apr. 30th, 2019) after TLS 1.3 (RFC 8446) was officially approved (Aug. 10th, 2018) to reach over 15% adoption. In contrast, the shift from TLS 1.1 to TLS 1.2 needed around five years to reach the 15% adoption after the approval date of TLS 1.2 [4].[2] The TLS 1.2

[1]We release all datasets used in this paper, the source codes of client-side applications to generate the datasets, and the scripts to analyze the datasets at a public repository: https://github.com/tls13contribution/tls13.git

[2]The TLS 1.2 RFC document was published in Aug. 2008. SSL Pulse (https://www.ssllabs.com/ssl-pulse/) reported that the ratio of TLS 1.2 adoption on web servers exceeded 15% out of 170K popular TLS websites on June 2013.

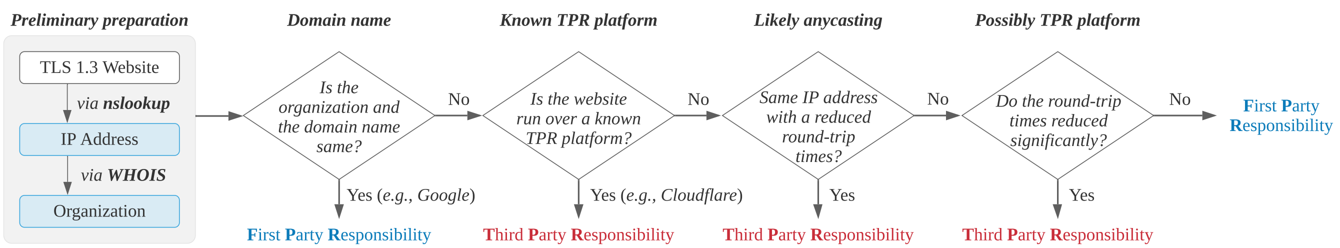

Figure 1: Platform Identification. We classify platforms that websites are running over into two categories to conduct a platform-based analysis. They are called the first-party responsibility (FPR) and the third-party responsibility (TPR). The important difference between FPR and TPR is who is responsible for managing the TLS servers and libraries.

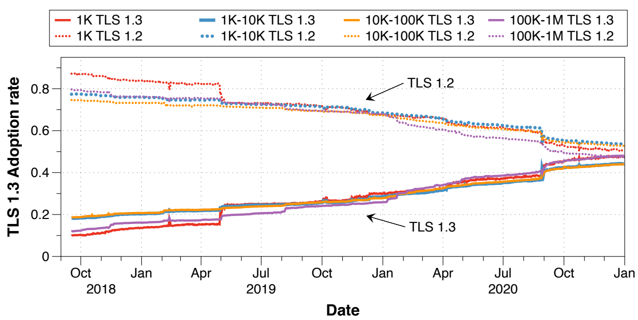

Figure 2: TLS 1.3 adoption ratio by Alexa rank. There is no significant difference in the trend of TLS 1.3 adoption between Alexa top 1K, 10K, 100K, and 1M sites.

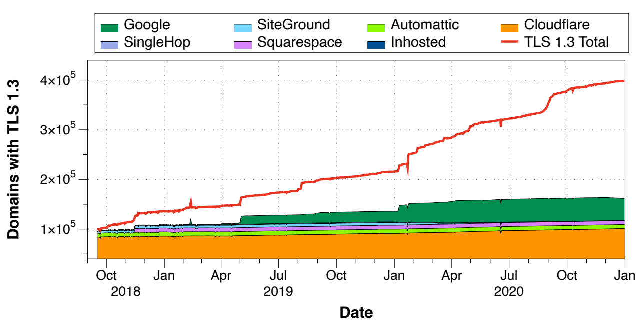

Figure 3: TLS 1.3 adoption ratio by platforms. The ratio is continuously increasing, by around 0.042% per day. The main contributors are TPR platforms whose administrators can upgrade their TLS libraries for all the websites they are hosting.

upgrades were mainly influenced by security events such as BEAST and Snowden's revelation [25]. On the other hand, TLS 1.3 is being proactively deployed. It motivates us to investigate what causes such fast deployment of TLS 1.3.

Popular Websites and TLS 1.3. Several studies show higher-ranked websites are likely to adopt new security features quickly. For example, HTTPS and SMTP security extensions are deployed further in popular sites [19, 23]. We investigate whether there is a correlation between the adoption rate of TLS 1.3 and the Alexa ranks of the websites. We consider the four cases—Alexa top 1K, 10K (1K–10K), 100K (10K–100K), and 1M (100K–1M) sites—to understand the correlation.

We find that all the bins show continuous increases with similar patterns. As shown in Figure 2, in general, top-ranked websites have more TLS 1.3 adoption rates. However, in terms of the increment rate, the lower-ranked sites are likely to deploy TLS 1.3 faster. Interestingly, when we see the adoption rate of the sites below 1K, it was the lowest, which means the highest-ranked websites are more conservative in adopting TLS 1.3 than the lower-ranked websites. The sites between 200K and 300K show a higher adoption rate than the sites between 100K and 200K.

We conclude that there is no strong positive correlation between the Alexa ranks and the TLS 1.3 adoption rate from these observations. In other words, the trend of TLS 1.3 adoption shows a different result from that of HTTPS or SMTP security extension supporting domains.

Platform-based Adoption. To better understand the main contributors for the fast deployment of TLS 1.3, we compare the overall tendency of the adoption ratio of the popular platform providers. Specifically, we select the seven most popular platform providers:

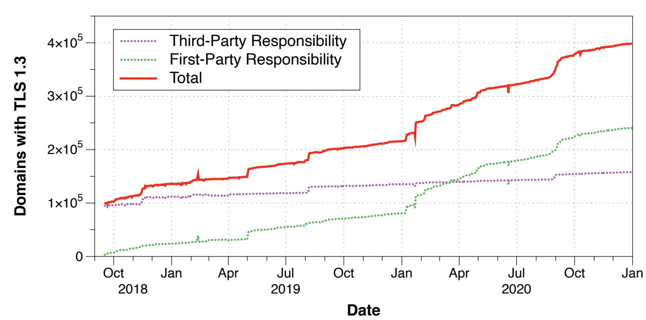

Figure 4: TLS 1.3 adoption ratio by who has responsibilities for managing the configuration of TLS servers. First-party responsibility: the web server administrators are responsible for managing their TLS server. Third-party responsibility: A third-party provider is responsible for managing the TLS server, such as Cloudflare.

Table 1: Changes of TLS version upgrade. Most of the websites are directly upgraded from TLS 1.2, while some websites show unstable support for TLS 1.3.

Pattern	FPR	TPR	Total
1.0 → 1.3	1,267 (0.5%)	543 (0.3%)	1,810 (0.5%)
1.1 → 1.3	20 (0.0%)	4 (0.0%)	24 (0.0%)
1.2 → 1.3	174,870 (72.7%)	70,265 (44.5%)	245,135 (61.5%)
1.3	11,702 (4.9%)	63,815 (40.4%)	75,517 (19.0%)
Unstable	52,653 (21.9%)	23,454 (14.8%)	76,107 (19.1%)
Total	240,512 (100.0%)	158,081 (100.0%)	398,593 (100.0%)

Cloudflare, Inhosted Lp., SiteGround, Squarespace Inc., Automattic Inc., SingleHop LLC., and Google LLC.

Figure 3 shows the changes of the overall TLS 1.3 deployment and the changes of TLS 1.3 deployment of the platform providers over time. The line demonstrates the overall TLS 1.3 deployment, while the bar graph shows a cumulative TLS 1.3 deployment of the selected platform providers. We observe that the seven companies cover most of the support of TLS 1.3 at the early stage, implying that these major platforms initially drive the rate of deployment.

We also compare the overall trend with the trend of websites served by FPR platforms and TPR platforms. As demonstrated in Figure 4, the result shows that TPR platforms deploy TLS 1.3 at once; thus, we can observe a few step-like increasing trends from the graph. On the other hand, the number of websites that support TLS 1.3 over FPR platforms is gradually increasing, showing a similar shape with the overall trend. We find that after Mar. 20th 2020, websites over FPR platforms account for more than 50% of the TLS 1.3 adoption. From these observations, we conclude that TPR platforms mainly lead the TLS 1.3 adoption at the early stage of TLS 1.3. However, the recent increase is caused by websites served over FPR platforms.

Furthermore, there are two interesting points regarding the platform providers. *First*, we see a sharp increase between Nov. 11st, 2018 and Nov. 16th, 2018. It is mainly because *Inhosted* initiated support for TLS 1.3 for 1,696 websites on Nov. 14th, 2018 and *Squarespace* enabled TLS 1.3 for 4,789 websites on Nov. 15th, 2018 and other 3,001 websites on Nov. 16th, 2018. *Second*, there is a peak between Feb. 10th, 2018 and Feb. 14th, 2018. It was related to the Google platform. On Feb. 10th, 2018, only 649 websites supported TLS 1.3 over the platform. The number of TLS 1.3 sites over Google increased to 6,760 and 11,962 websites on Feb. 11st and 12nd, respectively, which dropped to 664 websites on Feb. 14th.

5 SECURITY

One of the main goals of TLS 1.3 is to enhance the security of TLS. By upgrading to TLS 1.3, websites can obtain several security benefits such as enforcing forward-secret and AEAD cipher suites. Moreover, we discuss the critical security issues when web servers unstably support TLS 1.3.

Table 2: Websites of unstable TLS 1.3 are analyzed during a period from Sept. 17th, 2018 to Dec. 31st, 2020; Case #1: a machine is downgraded again after being upgraded to TLS 1.3; Case #2: websites migrate or extend their servers to other cloud or CDN networks where the versions are downgraded.

	FPR	TPR
Case #1	32,013 (60.8%)	5,392 (23.0%)
Case #2	12,597 (23.9%)	15,099 (64.4%)
Both	2,031 (3.9%)	1,636 (7.0%)
Others	6,012 (11.4%)	1,327 (5.7%)
Total	52,653 (100.0%)	23,454 (100.0%)

Table 3: Average (and median) of downgraded days per case is measured; FPR websites show longer downgraded days than TPR websites.

Case	FPR	TPR
Case #1	97.4 (78)	80.7 (47)
Case #2	211.5 (157)	121.8 (43)
Both	236.5 (188)	139.7 (61)

5.1 Security Benefits

To better understand the security benefits of TLS 1.3, we measure the highest TLS versions that each website supports in our observation period. We first create TLS version traces of the websites supporting TLS 1.3 to see the TLS version changes daily using the Security Parameters (D1) dataset (more detail in §3). Each trace consists of a series of three elements: *date*, *IP address*, and *TLS version*. Finally, we obtain 398,593 traces of websites that support TLS 1.3 on Dec. 31st, 2020 and find 350 different patterns of the traces in total. We assume that different IP addresses indicate different servers in this experiment.

As shown in Table 1, the majority of the websites have adopted TLS 1.3. It enhances the security of the TLS ecosystem. Notably, we find that 61.5% of TLS 1.3 websites are directly upgraded from TLS 1.2 during our observation period. There are very few servers (0.5%) upgraded from TLS 1.0 or 1.1 to 1.3.

Moreover, there are 4,829 (TLS 1.3 supported) websites that have upgraded to use *forward-secret* cipher suites from *non-forward-secret* cipher suites, providing higher security for the websites. Furthermore, 17,094 sites have changed non-AEAD cipher suites to AEAD cipher suites by upgrading to TLS 1.3.

5.2 Unstable TLS Versions

We observe that 76,107 cases of unstable TLS versions in our trace: TLS 1.3 is supported on a particular day but falls back to TLS 1.2 later. There are 4,926 highly unstable cases (1.2%, out of the 398,593 traces). They have changed their highest TLS version *more than ten times*.

This instability indicates that these websites do not always guarantee the security benefits of TLS 1.3. To understand the instabilities, we take a closer look at the unstable cases from Sept. 17th, 2018 to Dec. 31st, 2020. Two representative scenarios cause the instability—1) downgraded servers and 2) migration to servers with lower TLS versions, the statistics shown in Table 2. We also find that the instability occurs because of the multiple platform services, especially when one platform supports the lower TLS version than the others. An example can be a website that uses three platform services where one supports only TLS 1.0, while others enable TLS 1.3. Note that we demonstrate the number of "unstable days" in Table 3. It shows how many days the websites sustain their lower TLS versions since the TLS 1.3 session has been established.

5.2.1 Downgraded TLS Versions. The two cases (downgraded servers and migration to servers with lower TLS versions) can cause the instability of TLS versions.

Case #1: Downgraded Servers. The most prevalent case of the FPR websites, which accounts for 60.8% of unstable FPR traces, is that the TLS versions of web servers are downgraded to TLS 1.2 even after upgrading to 1.3. For example, one website starts to support TLS 1.3 on Dec. 18th, 2018, in our dataset, but it (with the same IP address) is downgraded to TLS 1.2 on Jan. 16th, 2019. About three weeks later, it again supports TLS 1.3 after Feb. 8th, 2019.

Case #2: Migration to Servers with Lower TLS Versions. There are websites that support TLS 1.3 for some periods but are downgraded to TLS 1.2 because they change their platforms (e.g., CDNs). For example, one website is hosted on a platform supporting TLS 1.3 before Mar. 20th, 2019. After then, we find that the website's IP addresses are changed to another platform that does not support TLS 1.3. Similar cases account for 23.9% of the TLS 1.3 FPR websites and 64.4% of the TLS 1.3 TPR websites.

5.2.2 Regional Differences. We investigate the correlation between the regional differences and the instability of TLS versions. Specifically, we use the Handshake Messages (D2) dataset to see the TLS version of the sessions between clients from eight different regions and each TLS 1.3 website. We find 357 cases in which clients from different regions establish TLS sessions with different TLS versions.

For example, our client application establishes a TLS 1.3 session with a specific website in Eastern North America, while it establishes a TLS 1.2 session with the website in East Asia. The IP addresses used to connect to the servers were different, hosted by two distinct platforms, providing only TLS 1.2.

We argue that this instability should be resolved because the security of a website relies on its lowest TLS version. An adversary who is aware of the instability of a particular website may attack a weak server in a different region to exploit the vulnerabilities in a lower TLS version.

Takeaways. We observe that many websites gain the security benefits such as *forward secrecy* after upgrading their TLS versions to TLS 1.3. However, we also find a security issue where websites unstably support the TLS version. The instability of TLS versions happens when 1) downgrading TLS versions and 2) migrating to servers with lower TLS versions. Web server administrators are recommended to be sure to support TLS 1.3 when migrating to other platforms. Moreover, when they utilize TPR platforms such as multi-CDNs, they are also recommended to check whether their platform services stably support TLS 1.3 from different regions.

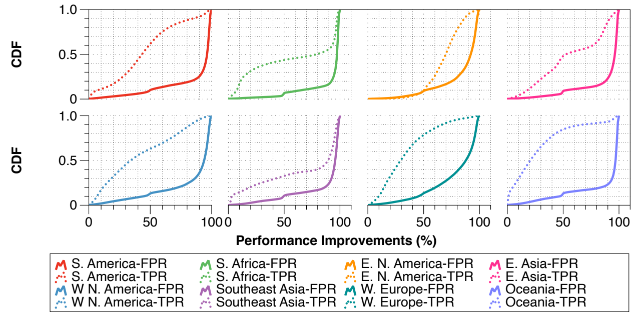

Figure 5: Delay latency of the TLS 1.3 handshake is measured from eight different regions.

6 PERFORMANCE ENHANCEMENT

In this section, we quantify how much delay is reduced by TLS 1.3. We measure the elapsed time of both the TLS 1.2 and TLS 1.3 full handshakes and calculate the performance gain defined as

$$(1 - \frac{\text{(Elapsed Time for TLS 1.3 Full Handshake)}}{\text{(Elapsed Time for TLS 1.2 Full Handshake)}}) \times 100\ (\%)$$

The measurement results are summarized in Table 4. The average performance gain of TLS 1.3 compared to TLS 1.2 is more than 57.9% (in Western Europe). We observe the most significant improvements in South Africa since it is located (on average) geographically farther from the Alexa 1M websites than the other regions. Note that the average round-trip time of South Africa to the Alexa 1M websites is 138.08 ms that is longer than those of other regions. Our manual inspection result shows that the Alexa 1M servers are *mostly located in North America (Eastern and Western) and Western Europe*. From the observations, we conclude that the longer the delay is taken between the server and the client, the more significant performance improvements one may get.

This trend is clearly shown when we consider the platforms for our analysis. As described in Figure 5, the websites running over TPR platforms experience smaller delay improvements compared to those on FPR platforms. This is because the TPR platforms are usually distributed on a global scale and hence servers tend to be closer to the clients; thus, the networking distances account for the variation in delay performance gains. Note that the correlation between the round-trip time and the FPR gain is 0.87.

Takeaways. From the above observations, we conclude that TLS 1.3 can be more beneficial to websites which cannot use CDN services due to budget or other reasons. We believe this result may motivate individual websites to be upgraded to TLS 1.3 for both security and performance.

7 IMPLEMENTATION OF TLS LIBRARY

In this section, we investigate whether TLS libraries in the wild are correctly and faithfully implement with the new features of TLS 1.3. Specifically, we focus on analyzing two aspects of the implementation: 1) new TLS 1.3 features and 2) common vulnerabilities and exposures (CVEs). We first look at TLS 1.3 libraries to check if they implement the downgrade attack mechanism and the parsing routine for the certificate extension fields. Then, we investigate

Table 4: Averaged round trip time and performance gain from eight different regions to TLS 1.3 web servers. We measure round trip times toward Alexa top 1M sites from each region to quantify how far a region is from web servers and the potential benefit of the TLS 1.3 handshake. We find a high correlation between round trip times and averaged gains of FPR websites ($R^2 = 0.9$).

	Eastern N. America	Western N. America	South America	Western Europe	South Africa	East Asia	South East Asia	Oceania
Round Trip Time (ms)	51.7	61.2	102.9	40.5	138.1	120.9	136.2	126.6
Average of Gain (%)	76.7	63.1	66.1	57.9	76.8	72.9	77.1	59.0
Average of FPR Gain (%)	84.6	83.3	86.9	78.9	91.1	88.9	87.3	86.4
Average of TPR Gain (%)	69.0	41.1	45.2	34.3	58.9	57.8	66.6	27.9

Table 5: We investigate whether TLS Libraries incorporate the two new features of TLS 1.3. Note that not all of the TLS implementations support them.

TLS Library	Version	Downgrade Protection	Certificate Extensions
Apple CoreTLS	167	○	○
BoringSSL	Latest*	●	●
Fizz	Latest*	○	○
Mozilla NSS	3.61	●	●
OpenSSL	1.1.1i	●	●
WolfSSL	4.6.0	●	○

*: The source code is cloned from the public repository on Feb. 5th, 2021. ●: fully supported. ○: not fully supported.

whether web servers and web browsers properly employ the features. Finally, we review the CVE reports of TLS libraries to measure the security threats caused by the TLS implementations, especially relevant to TLS 1.3. Hence, we focus on the CVEs reported after TLS 1.3 was approved.

As shown in Table 5, of the total seven TLS libraries supporting TLS 1.3, only `BorningSSL`, `Mozilla NSS`, and `OpenSSL` fully support the two new features.

7.1 Downgrade Attack Protection

Recall that TLS 1.3 prevents downgrading attacks by inserting a downgrade sentinel (`DOWNGRD0` or `DOWNGRD1`) in the last 8 bytes of the server's random value when a client attempts to connect over TLS 1.2 (c.f., §2). If the client supports TLS 1.3, the client must abort the connection attempt with the downgrade sentinel over TLS 1.2 and send the web server an "illegal parameter" alert message.

Servers. To measure how many TLS 1.3 web servers correctly provide the downgrade protection, we run our client application that performs TLS 1.2 handshakes with TLS 1.3 websites. Then, we inspect the `ServerHello` messages. The result shows that most of the TLS 1.3 servers embed the downgrade sentinels in their `ServerHello` while 98 servers (0.03%) do not embed the sentinels. We find that 39 (out of the 98 servers, 39.8%) are over the Facebook platforms that may use `Fizz` [3] for its TLS library. Note that `Fizz` has *not* implemented the downgrade protection mechanism (c.f., Table 5).

Clients. To check whether web browsers (i.e., clients) correctly respond to the alert message, we conduct an experiment in which an active man-in-the-middle adversary performs the downgrade attack. Specifically, the adversary composes a `ClientHello` message in TLS 1.2 by dropping the *SupportedVersion* field in the message. Then, we relay the message to our TLS 1.3 server. In turn, our web server sends back to the client a `ServerHello` message that contains `DOWNGRD1`. We check whether our controlled web server receives the "illegal parameter" alert message from a web browser. Our experiment is conducted only with web browsers that support TLS 1.3, including Firefox (Linux/Android), Chrome (Linux/Android), and Edge (Android).

The results show that none of the browsers send the "illegal parameter" alert message until Firefox (version 72) first understands downgrade sentinels in `ServerHello` (as of Jan. 7th, 2020). Note that it is 516 days after TLS 1.3 was approved (Aug. 10th, 2018). Before then, all the browsers send a "bad mac" alert message when they detect the handshake messages tampered with from the `Finished` message. Chrome started to enable the downgrade protection mechanism on Apr. 13rd, 2020 (613 days after the TLS 1.3 approval), while Edge does not support it yet (version 46.01). It may not be critical since web browsers have removed the insecure TLS fallback mechanism [2] that necessitates the downgrade protection mechanism. However, for compatibility reasons, the browser vendors occasionally enable the TLS fallback mechanism to see servers' tolerance [1], which may cause clients to remain exposed to security threats.

7.2 Certificate Extensions

TLS 1.3 also introduces an extension field in the structure of the `Certificate` message. There are two examples—1) signed certificate timestamps (SCTs) [27] and 2) OCSP stapling [34]—described in [37]. Although they are not new features of TLS 1.3, revisions of TLS implementations are required to parse the new fields and call the functions related to SCTs and OCSP stapling. Thus, we first check whether or not TLS libraries properly process the certificate extensions. Then, we measure how many SCTs and OCSP stapling are used in practice.

We find that only three out of six TLS libraries properly handle the certificate extension fields, as shown in Table 5. It means that if a server sends an SCT or an OCSP response together with a certificate, only the client based on the three libraries can process the SCTs and the OCSP response.

Table 6: CVEs regarding the TLS Libraries. We categorize the vulnerabilities into three classes: 1) the vulnerability introduced due to TLS 1.3, 2) the vulnerability that can be addressed if TLS 1.3 is adopted, and 3) the vulnerability that is not related to any particular version of TLS.

TLS Library	Total	Category 1	Category 2	Category 3
BoringSSL	2	0	1	1
Fizz	2	0	0	2
Mozilla NSS	3	0	1	2
OpenSSL	25	0	4	21
WolfSSL	18	2	1	15
Total	**62**	**2**	**13**	**47**

SCT. Of the total 399K TLS 1.3 websites, 71 websites (0.02%) include their SCTs in the certificate extension fields. Among them, three SCTs show a signature error. However, many TLS libraries do not have SCT-related implementations yet. It means that even though the server-side prepares the SCTs, many of the clients based on TLS libraries other than `BoringSSL` and `Mozilla NSS` cannot parse and process the SCTs.

OCSP Stapling. 98,861 websites (28.4% out of 399K TLS 1.3 websites) provide OCSP responses in the certificate extension fields. 39.3% out of 101,155 responses fail in verification; particularly, 71 of them have signature errors, and the others have parsing errors. Compared to the SCTs, many TLS libraries already support this feature, meaning that the servers need to reduce the error rate of their OCSP responses.

7.3 Vulnerabilities and Exposures

The security of TLS in practice depends on the implementation of TLS libraries. To understand how secure the TLS libraries are, we review known CVEs of the six TLS libraries. In particular, we investigate the vulnerabilities announced after the approval of TLS 1.3 (Aug. 10th, 2018) to focus on vulnerabilities relevant to TLS 1.3.

We find 62 CVE entries in total, categorized into three classes: (1) the vulnerability introduced due to TLS 1.3, (2) the vulnerability that can be addressed if TLS 1.3 is used, and (3) the vulnerability that is TLS version agnostic. Table 6 shows the result.

Observations. *First*, there are two cases only related to TLS 1.3 in WolfSSL: CVE-2019-15651 [13] and CVE-2020-12457 [12]. Both are vulnerabilities in the TLS 1.3 handshake protocol where CVE-2019-15651 is related to the certificate extensions while CVE-2020-12457 is associated with the change cipher spec message. Note that TLS 1.3 changes the handshake protocol significantly from its previous versions. As a result, TLS libraries introduce new implementations of the state machines for TLS 1.3, often including new vulnerabilities like the two CVEs mentioned above [12, 13].

Second, TLS 1.3 deprecates various specifications including several cryptographic primitives and static DH and CBC-related cipher suites. As a result, vulnerabilities related to those deprecated specifications can be addressed by simply adopting TLS 1.3. For example, the recent Raccoon attack [31], identified as CVE-2020-1968, which is performed to acquire the Diffie-Hellman (DH) shared secret through side-channel attacks, cannot be done over TLS 1.3, since it prevents the DH key from being reused.

Takeaways. We find that many critical security features of TLS 1.3 are not fully implemented yet in the client-side libraries, compared with web servers, leading to various security concerns. Moreover, our analysis confirms that many TLS libraries' vulnerabilities can be quickly addressed by adopting TLS 1.3.

8 RELATED WORK

In this section, we discuss related work in two key areas: measuring the Web PKI ecosystem and the TLS deployment.

The Web PKI Ecosystem. The Web PKI ecosystem has been well studied and understood after network scanner tools were introduced (e.g., ZMap [18] and ICSI Notary [6]). These scanners help researchers collect representative datasets in the wild (such as X.509 certificates [10]) within a relatively short time and discover security problems in the Web PKI: including (1) vulnerabilities in the wild [5, 18, 21, 42], (2) revocation [9, 26, 30], (3) aftermath of the *Heartbleed* bug [16, 43], (4) private key sharing [7], (5) certificate transparency [24, 28, 38–40], and (6) invalid certificates [8, 26]. These measurement studies help improve the security of the entire Web PKI ecosystem. Moreover, the TLS interceptions have also been studied [14, 17, 41]. The interceptions are mainly conducted by middleboxes such as anti-virus software and security gateways. The studies have reported that the negative effects: specifically, incorrect certificate validation or security downgrade.

TLS Deployment. Unlike the measurements of the Web PKI, the deployment and security of TLS 1.3 are little known. To name a few, Holz et al. [22] showed the statistics of the TLS 1.3 usage and what boosts its deployment; however, it does not present the security implication, performance, and implementation of TLS 1.3. Moreover, Razaghpanah et al. [35] analyzed the cipher suite list and the TLS extensions (specifically, weak cipher suites and vulnerable protocol versions) on Android using passive datasets collected from *Lumen Privacy Monitor*, a free Android app. In this work, TLS 1.3 was not discussed. Recently, Kotzias et al. [25] first examined how the TLS ecosystem has evolved over approximately six years (February 2012 – April 2018) using passive and active datasets. They observed correlations between the TLS ecosystem's evolution and new TLS attacks; in other words, there have been significant improvements to the TLS ecosystem after new TLS attacks were discovered. However, this study barely measured TLS 1.3 deployment; rather they focused on the draft version 28 of TLS 1.3, since the study was conducted before TLS 1.3 was officially approved by the *IETF*. In contrast to these three measurement studies, our work focuses on the official TLS 1.3 examining the differences from TLS 1.2 in terms of deployments, security, performance, and implementation of the libraries that support TLS 1.3.

9 CONCLUSION

In this paper, we present a comprehensive analysis of TLS 1.3 in terms of its *adoption*, *security*, *performance*, and *implementation*. To answer the research questions from the four aspects, we conduct a temporal, spatial, and platform-based analysis on our datasets, consisting of TLS handshake messages from the Alexa top 1M websites for 837 days.

Our research leads to the following observations. First, the adoption rate of TLS 1.3 has rapidly increased compared to the previous versions of TLS, led mainly by third-party platforms such as CDNs. Second, we found that websites (19.1%) suffer from unstable support for TLS 1.3 during our observation period. Some websites downgraded to the lower version of TLS even after upgrading to TLS 1.3. Other cases were due to the website migration from the platform that supports TLS 1.3 to the one that does not. There were also cases that this instability happens due to the regional differences. Third, TLS 1.3 achieves reduced delays over TLS 1.2, which is more beneficial for first-party responsibility platforms than third-party responsibility ones. Fourth, we observe that many implementations of TLS libraries do not properly support the new features of TLS 1.3 such as downgrade protections. Finally, our study on CVEs of TLS libraries reveals that many vulnerabilities can be mitigated by simply adopting TLS 1.3.

ACKNOWLEDGMENTS

We thank the anonymous referees for their constructive feedback. The research was supported, in part, by NSF under awards 1916499, 1908021, and 1850392. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsor.

REFERENCES

[1] 2016. Enable TLS 1.3 by default. https://bugzilla.mozilla.org/show_bug.cgi?id=1310516. Accessed: 2019-04-27.
[2] 2016. TLS Fallbacks are Dead. https://textslashplain.com/2016/05/04/tls-fallbacks-are-dead/. Accessed: 2019-05-02.
[3] 2019. C++14 implementation of the TLS-1.3 standard. https://github.com/facebookincubator/fizz. Accessed: 2019-04-29.
[4] April. SSL Pulse. https://www.ssllabs.com/ssl-pulse/. Accessed: 2019-04-27.
[5] David Adrian, Karthikeyan Bhargavan, Zakir Durumeric, Pierrick Gaudry, Matthew Green, J Alex Halderman, Nadia Heninger, Drew Springall, Emmanuel Thomé, Luke Valenta, et al. 2015. Imperfect forward secrecy: How Diffie-Hellman fails in practice. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 5–17.
[6] Bernhard Amann, Matthias Vallentin, Seth Hall, and Robin Sommer. 2012. *Extracting certificates from live traffic: A near real-time SSL notary service.* Technical Report. Citeseer.
[7] Frank Cangialosi, Taejoong Chung, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, and Christo Wilson. 2016. Measurement and analysis of private key sharing in the https ecosystem. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 628–640.
[8] Taejoong Chung, Yabing Liu, David Choffnes, Dave Levin, Bruce MacDowell Maggs, Alan Mislove, and Christo Wilson. 2016. Measuring and applying invalid SSL certificates: The silent majority. In *Proceedings of the 2016 Internet Measurement Conference*. 527–541.
[9] Taejoong Chung, Jay Lok, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, John Rula, Nick Sullivan, and Christo Wilson. 2018. Is the Web Ready for OCSP Must-Staple?. In *Proceedings of the Internet Measurement Conference 2018*. 105–118.
[10] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. 2008. Internet X. 509 Public Key Infrastructure Certificate and CRL Profile.
[11] CVE. 2014. CVE - CVE-2014-3566. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3566. (Accessed on 03/26/2019).
[12] CVE. 2019. CVE - CVE-2019-15651. https://nvd.nist.gov/vuln/detail/CVE-2019-15651. (Accessed on 10/19/2020).
[13] CVE. 2020. CVE - CVE-2019-15651. https://nvd.nist.gov/vuln/detail/CVE-2020-12457. (Accessed on 10/19/2020).
[14] Xavier de Carné de Carnavalet and Mohammad Mannan. 2016. Killed by proxy: Analyzing client-end TLS interception software. In *Network and Distributed System Security Symposium*.
[15] Tim Dierks and Eric Rescorla. 2008. The transport layer security (TLS) protocol version 1.2.
[16] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, et al. 2014. The matter of heartbleed. In *Proceedings of the 2014 conference on internet measurement conference*. 475–488.
[17] Zakir Durumeric, Zane Ma, Drew Springall, Richard Barnes, Nick Sullivan, Elie Bursztein, Michael Bailey, J. Alex Halderman, and Vern Paxson. 2017. The security impact of HTTPS interception. In *Network and Distributed Systems Symposium*.
[18] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. 2013. ZMap: Fast Internet-wide scanning and its security applications. In *22nd {USENIX} Security Symposium ({USENIX} Security 13)*. 605–620.
[19] Adrienne Porter Felt, Richard Barnes, April King, Chris Palmer, Chris Bentzel, and Parisa Tabriz. 2017. Measuring HTTPS Adoption on the Web. In *Proceedings of the 26th USENIX Conference on Security Symposium* (Vancouver, BC, Canada) *(SEC'17)*. USENIX Association, Berkeley, CA, USA, 1323–1338. http://dl.acm.org/citation.cfm?id=3241189.3241292
[20] Google. 2019. HTTPS encryption on the web – Google Transparency Report. https://transparencyreport.google.com/https/overview?hl=en. (Accessed on 03/08/2019).
[21] Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J Alex Halderman. 2012. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In *Presented as part of the 21st {USENIX} Security Symposium ({USENIX} Security 12)*. 205–220.
[22] Ralph Holz, Jens Hiller, Johanna Amann, Abbas Razaghpanah, Thomas Jost, Narseo Vallina-Rodriguez, and Oliver Hohlfeld. 2020. Tracking the deployment of TLS 1.3 on the Web: A story of experimentation and centralization. *ACM SIGCOMM Computer Communication Review* 50, 3 (2020), 3–15.
[23] Hang Hu and Gang Wang. 2018. End-to-end measurements of email spoofing attacks. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 1095–1112.
[24] Daniel Kales, Olamide Omolola, and Sebastian Ramacher. 2019. Revisiting User Privacy for Certificate Transparency. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 432–447.
[25] Platon Kotzias, Abbas Razaghpanah, Johanna Amann, Kenneth G. Paterson, Narseo Vallina-Rodriguez, and Juan Caballero. 2018. Coming of Age: A Longitudinal Study of TLS Deployment. In *Proceedings of the Internet Measurement Conference 2018* (Boston, MA, USA) *(IMC '18)*. ACM, New York, NY, USA, 415–428. https://doi.org/10.1145/3278532.3278568
[26] Deepak Kumar, Zhengping Wang, Matthew Hyder, Joseph Dickinson, Gabrielle Beck, David Adrian, Joshua Mason, Zakir Durumeric, J Alex Halderman, and Michael Bailey. 2018. Tracking certificate misissuance in the wild. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 785–798.
[27] Ben Laurie, Adam Langley, and Emilia Kasper. 2013. Certificate transparency.
[28] 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*. 2505–2520.
[29] Zhihao Li, Dave Levin, Neil Spring, and Bobby Bhattacharjee. 2018. Internet anycast: performance, problems, & potential. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. ACM, 59–73.
[30] Yabing Liu, Will Tome, Liang Zhang, David Choffnes, Dave Levin, Bruce Maggs, Alan Mislove, Aaron Schulman, and Christo Wilson. 2015. An end-to-end measurement of certificate revocation in the web's PKI. In *Proceedings of the 2015 Internet Measurement Conference*. 183–196.
[31] Robert Merget, Marcus Brinkmann, Nimrod Aviram, Juraj Somorovsky, Johannes Mittmann, and Jörg Schwenk. 2021. Raccoon Attack: Finding and Exploiting Most-Significant-Bit-Oracles in TLS-DH(E). In *Proceedings of the 30th USENIX Conference on Security Symposium* (Vancouver, BC, Canada) *(SEC'21)*. USENIX Association.
[32] Bodo Möller and Adam Langley. 2015. TLS fallback Signaling Cipher Suite Value (SCSV) for preventing protocol downgrade attacks.
[33] David Naylor, Alessandro Finamore, Ilias Leontiadis, Yan Grunenberger, Marco Mellia, Maurizio Munafò, Konstantina Papagiannaki, and Peter Steenkiste. 2014. The Cost of the S in HTTPS. In *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 133–140.
[34] Yngve Pettersen. 2013. The transport layer security (TLS) multiple certificate status request extension.
[35] Abbas Razaghpanah, Arian Akhavan Niaki, Narseo Vallina-Rodriguez, Srikanth Sundaresan, Johanna Amann, and Phillipa Gill. 2017. Studying TLS Usage in Android Apps. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies* (Incheon, Republic of Korea) *(CoNEXT '17)*. ACM, New York, NY, USA, 350–362. https://doi.org/10.1145/3143361.3143400
[36] E. Rescorla. 2000. HTTP over TLS. https://tools.ietf.org/html/rfc2818
[37] Eric Rescorla. 2018. The transport layer security (TLS) protocol version 1.3.
[38] Richard Roberts and Dave Levin. 2019. When Certificate Transparency Is Too Transparent: Analyzing Information Leakage in HTTPS Domain Names. In *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society*. 87–92.
[39] 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.

[40] Emily Stark, Ryan Sleevi, Rijad Muminovic, Devon O'Brien, Eran Messeri, Adrienne Porter Felt, Brendan McMillion, and Parisa Tabriz. 2019. Does certificate transparency break the web? Measuring adoption and error rate. In *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 211–226.

[41] Louis Waked, Mohammad Mannan, and Amr Youssef. 2018. To Intercept or Not to Intercept: Analyzing TLS Interception in Network Appliances. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*. ACM, 399–412.

[42] Scott Yilek, Eric Rescorla, Hovav Shacham, Brandon Enright, and Stefan Savage. 2009. When private keys are public: Results from the 2008 Debian OpenSSL vulnerability. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*. 15–27.

[43] Liang Zhang, David Choffnes, Dave Levin, Tudor Dumitraş, Alan Mislove, Aaron Schulman, and Christo Wilson. 2014. Analysis of SSL certificate reissues and revocations in the wake of Heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference*. 489–502.