Re-check Your Certificates! Experiences and Lessons Learnt from Real-world HTTPS Certificate Deployments

Wenya Wang[1], Yakang Li[1], Chao Wang[2], Yuan Yan[1], Juanru Li[1], and Dawu Gu[1]

[1] Shanghai Jiao Tong University, China
{ducky_97,liyakang_n,loccs,jarod,dwgu}@sjtu.edu.cn
[2] The Ohio State University, USA
wang.15147@osu.edu

Abstract. HTTPS is the typical security best practice to protect data transmission. However, it is difficult to correctly deploy HTTPS even for administrators with technical expertise, and mis-configurations often lead to user-facing errors and potential vulnerabilities. One major reason is that administrators do not follow new features of HTTPS ecosystem evolution, and mistakes were unnoticed and existed for years.
In this paper, we conduct a large-scale and persistent study on HTTPS certificate deployment to investigate whether administrators follow the certificate management trend. We empirically evaluate HTTPS certificate deployment concerning five new issues of improper configurations, and discuss how four usability factors may influence the mistakes. We monitored domain names and their certificates in China Education and Research Network (CERNET). Using data collected from more than 30,000 domain names of 113 universities in 12 weeks, we gained a panorama of HTTPS deployment in academia and summarized typical mistakes that administrators tend to make. Our results demonstrated that incorrect deployments were common, and a stable ratio of administrators did not follow the latest HTTPS guidelines. We also observed that certain usability factors (e.g., certificate shared by multiple domain names) potentially correlate with insecure HTTPS websites.

Keywords: HTTPS, PKI, digital certificate, CERNET

1 Introduction

Recent years have witnessed the significant growth of HTTPS among websites worldwide. By March 2021, 71% of the one million most visited websites worldwide have deployed HTTPS as their default [1]. To encourage web administrators and users to adopt HTTPS, many web browsers (e.g., Google Chrome) are enforcing HTTPS connections. However, the adoptions of HTTPS, especially the HTTPS certificate configuration part, is not easy for most websites administrators. Configuring a server to support HTTPS involves a wide variety of technical issues and is often error-prone. Efforts from both industry and academia have also been made to find insecurely deployed HTTPS websites. For instance, The Mozilla Observatory [2] has helped over 240,000 websites on how to configure their sites securely. Acer *et al.* [3] classified the top causes of HTTPS error warnings to implement actionable warnings for Chrome users. Unfortunately, the rapid evolution of HTTPS ecosystem (e.g., the use of TLS 1.3 protocol and `Let's Encrypt Free Certificates`) urges administrator to continuously upgrade security best practices. Therefore, a large potion of websites using HTTPS are still vulnerable due to recently emerged deployment issues, and this brings new challenges to administrators who are struggling for deploying HTTPS correctly.

We can blame certificate issuers for some deployment issues. Kumar *et al.* [4] systematically analyzed mechanical errors made by CAs when issuing certificates, and found that small CAs regularly made mistakes after analyzing 240 million browser-trust certificates. Schwittmann *et al.* [5] tested domain validation authorities and concluded that all major CAs were vulnerable in multiple ways because of insecure protocols they used. On the other hand, website administrators are also responsible for a variety of deployment problems. Recent researches have revealed that even for expert system administrators, the complexity of HTTPS certificates deployment often becomes a main obstacle for website security. Singanamalla *et al.* [6] measured HTTPS adoption of government websites around the world and found an overall lower https rate. Ukrop *et al.* [7] also found that self-signed certificates and name constrained certificates were over trusted by people in the IT industry, and notifications sometimes make no sense.

To better understand what are the most critical factors that lead to an incorrectly deployed HTTPS certificate in 2021, in this paper we conduct an empirical study of the HTTPS certificate deployment against more than 30,000 domain names and their corresponding certificate in China Education and Research Network (CERNET). We first surveyed a series of recent research papers on HTTPS certificate issuing, deployment, and revocation, summarized the trend of modern certificate management over the past five years. Then we evaluated the correctness of HTTPS deployment in CERNET by considering five new issues of HTTPS certificates deployment: subject alternative name (SAN) mismatching, long validity period, broken certificate chain, certificate opacity and obsolete crypto algorithms. In addition, we investigate four usability factors (certificate indicator, certificate issuer, certificate sharing and certificate revocation) and the potential correlation with above incorrect deployments. We conducted the study by consecutively monitoring how domain names of 113 universities configured their HTTPS certificates and relevant web/DNS parameters in 12 weeks (from November 4th, 2020 to January 20th, 2021), and summarizing the overall status of incorrect HTTPS certificate deployment in CERNET.

Our investigation shows that current deployment status of HTTPS certificates is not optimistic: even for administrators of CERNET (who are more likely to learn recently proposed guidelines and recommendations of HTTPS deployments), a stable portion of them (27%) did not adopt correct configurations. We found the top three deployment issues are **SAN mismatching**, **long validity**, and **broken certificate chain**. We also observed that certificate renewal would reproduce broken certificate chain issues and short lifespans would cause certificate expiration problems. Regarding the usability factors, we observed certain status of certificate (e.g., certificate sharing) strongly correlated with deployment issues.

Our contributions include:

- We summarized recent policies, guidelines, and recommendations of certificate management over the past five years, and evaluated how administrators followed those new changes at a large scale. The results showed that HTTPS certificate mis-configuration is widespread, and we found many popular web services (e.g., email services) were endangered.
- We analyzed the ratio of incorrectly deployed certificates/domain names in a certain period of time (12 weeks), and found the trend of improperly deploying certificates is not downwards: although many certificates were renewed and correctly configured, some updates introduced new mis-configurations.
- According to our study, we discussed whether typical usability factors (e.g., the selection of certificate issuer) correlate with incorrectly deployed certificates. Our analysis demonstrated that some factors did affect the certificate deployment and should be concerned by administrators.

2 Certificate Deployment in 2020s: Trends and Changes

In this section, we present a variety of details that reflect recent features and changes of HTTPS certificate management, and discuss best practices of certificate deployment.

HTTPS Certificates HTTPS encrypts and authenticates data transmission to protect its integrity and confidentiality from attackers. During HTTPS handshake, the client examines the fields and chains of certificates delivered by the server for authentication. `Subject Alternative Name` (SAN) field consists of `DNS names`. The `DNS name` is the criterion for identifying whether a domain name could match the certificate. Based on SAN, HTTPS certificates are divided into three types. Single-domain type has only one `DNS name` and can only be used by a domain name (e.g., `domain.com`). Wildcard type containing two `DNS names` can match a domain name and its subdomain name (e.g., `domain.com` and `*.domain.com`). And multi-domain type with multiple `DNS Names` could match the domain names with different suffixes (e.g., `domain-A.com` and `domain-B.org`). Long life span certificates are gradually being deprecated, as they generally use outdated configurations and would increase the risks. For example, `SHA-1`-to-`SHA-2` transition takes 3 years due to the slow replacement of long-term certificates.

A complete certificate chain is composed with the leaf certificates up to the root certificates. While TLS handshaking, the server should also transmit the intermediate certificates. Only the client links them to a trusted root certificate stored in local, the client would trust the leaf certificate. Up to February 2021, about 1.7% of servers did not configure the proper certificate chain [8]. Existing technologies such as AIA Fetching technology, Intermediate Certificate Cache technology and Intermediate Certificate Preloading help fix the broken chains, some researchers argue that AIA Fetching and Intermediate Certificate Cache technology would compromise user privacy [9] [10].

Certificate Authorities Before 2015, applying for an HTTPS certificate costed much, and the installing process was sophisticated [11]. `Let's Encrypt` launched in late 2015 greatly improved this situation. `Let's Encrypt` issues the Domain

Validation (DV) certificates free and automatically, and also helps maintain the certificates (e.g., renew the certificates). As of November 2020, 232 million websites install 144 million active certificates issued by `Let's Encrypt` [12]. CAs like `Encryption Everywhere Program`, cloud providers like `Cloudflare` also provide similar services for their customers.

And the other types, namely Extended Validation (EV), Individual Validation (IV) and Organization Validation (OV) provide more trust than DV certificates, especially EV certificates. As the certificates contain more details about the owners, and the verification process is more rigorous. CAs generally claim that EV would make phishing attacks harder and have higher warranties. However, attackers can still get an EV certificate with the same company name. And some EV certificates of well-known websites were not renewed in time. `LinkedIn` in 2017 forgot to renew its EV certificate [13], and `Instagram` in 2015 forgot to renew its EV certificate [13], Modern browsers have removed special indicators in the URL bar for EV certificates since 2019 [14] [15] [16].

Certificate Transparency and Revocation Records Mis-issued certificates or the certificates with compromised private keys do great harm to the privacy of users. The revocation mechanism restrains the trust in malicious certificates, and certificate transparency provides an auditing and monitoring system for HTTPS ecosystem.

Online Certificate Status Protocol (OCSP) is the most widely adopted certificate revocation mechanism [17] [18]. The client could perform an OCSP request about the certificate revocation status before the page finishes loading. This usually causes time delays. OCSP Stapling alleviates the latency problem to some extent, as servers can staple the OCSP response in the TLS handshakes. However, OCSP Stapling could be stripped by the attackers, and OCSP Must-Staple proposed in 2015 prevents this from happening [19]. Administrators could set OCSP Must-staple signal in the certificate extensions, and client like Firefox would block access to sites that set Must-staple but do not deliver OCSP Stapling [20]. But some browsers like Google Chrome consider that this would create new access-blocking issues and do not support OCSP Must-Staple [17].

In order to remedy incorrect or malicious certificates issuance by CAs, Certificate Transparency (CT) provides an auditing and monitoring system for HTTPS ecosystem. Certificate Transparency has been widely adopted [21]. Before a CA issues a certificate, it sends the pre-certificate of the certificate to CT logs first, and the logs respond with signed certificate timestamps (SCTs), which represents the promises of submission with the Maximum Merge Delay. And the CA embes the SCTs in the certificate before issuing. There are three ways to bind SCTs: certificate embedding, TLS extension and OCSP stapling. Certificate embedding is the most common and reliable way [22].

Certificate Sharing The prevalence of wildcard and multi-domain certificates promotes certificate sharing between multiple domain names. CDN service, cloud service, and hosting service providers usually install one certificate for multiple customers. Although it is feasible and convenient for multiple domain names to share a certificate, security risks also increase, since copies of private keys are stored on different servers, and some of the servers may not securely protected [23]. Once one of the server is compromised, all domains that share the same certificate would be severely threatened. Actually, researchers have shown that sharing a certificate with low-profile domain names will reduce the security of the system [24] [25].

3 Investigating Certificate Deployment

To systematically study the deployment status of recent HTTPS certificate deployment and summarize experiences from real-world websites, in this paper we aim to answer the following research questions by conducting large-scale and lasting analysis against certificates in use.

RQ1: What is the overall trend of certificate deployment? Despite the fact that 71.0% of all the websites have deployed HTTPS as their default protocol [1] in 2021, there still exist several issues that affect the security of network communication. Specifically, we observed that many websites did not follow new features and recommendations that nowadays HTTPS certificate management should adopt. Therefore, although they did not affected by those well-known bad practices such as self-signed certificate, various new issues are still disturbing website administrators. In response, we aim to investigate the overall trend of how administrators handle new changes and corresponding issues. Furthermore, we concern about the ratio of domain names affected by new deployment issues, the distributions of each issues, and their variations in a certain period of time. With a long-term analysis rather than an analysis against a snapshot, we try to investigate whether a deployment issue a temporary mistake or is it a common situation.

RQ2: How user factors affect the deployment of certificates? To further understand the root cause of incorrect certificate deployments, we consider several user factors that may mislead administrators. For instance, we would like to examine whether the sharing of certificate between multiple domain names is a dependent factor of an ill-deployed certificate.

RQ3: How many users and what kinds of web services are affected? To evaluate the actual risks of those incorrectly deployed certificates, we would like to check if those popular web services (e.g., email service, identity authentication service) are suffering from certificate deployment issues (and how many users are threatened due to the problems), or only those less frequently accessed websites mis-configured their certificates.

4 Evaluation

We report in this section our investigation results on recently emerged HTTPS certificates deployment issues. We conducted a 12-week investigation against more than 30,000 domain names and their HTTPS certificates to examine to what extent website administrators have followed the latest trends of HTTPS certificate management, or if they made mistakes due to the unfamiliarity of those new changes. We first detail the built dataset of our investigation (Section 4.1), and then elaborate how recent certificates deployment issues affected the studied domain names/certificates (Section 4.2). Finally, we discuss four factors that may lead to incorrect certificates deployment (Section 4.3), and show the influence of incorrectly deployed certificates against popular web services (Section 4.4).

Ethical Statement Our study did not tamper with any website with insecurely deployed certificate and executed a full responsible disclosure process by contacting the network administrators of relevant universities. In further, we only used port 80 and 443 to access the websites and did not perform any port scanning actions that might result in abuse of the hosts in the target.

4.1 Dataset and Experiment Setup

Our investigation utilized China Education and Research Network (CERNET) [26] as our research target. A typical feature of CERNET is that its members (i.e., universities), unlike enterprises or government departments, manage their belonging domain names and certificates in a de-centralized style. That is, even for the same university, all of its belonging web servers and certificates are often managed by different departments/institutes rather than a single bureau. Therefore, this network becomes a very typical example to study the diversification of HTTPS certificate managements (and how this diversity affects security).

To build a dataset for the certificate deployment analysis, we first collected all HTTPS-accessible domain names of CERNET. Our investigation monitored domain names of 113 major universities [27] of CERNET. More specifically, since in CERNET the domain and subdomain names of one specific university possess the same suffix in the form of `name-abbreviation.edu.cn` (e.g., `pku.edu.cn` is the common suffix for all domain names belonging to *Peking University* (PKU), and all subdomain names of PKU must follows the format of `*.pku.edu.cn` [28]). We started from generating an initial "seed" list of each university containing its common suffix, and then automatically collected their subdomain names of the domain in the list through utilizing both Sublist3r [29] subdomain name enumeration tool and ctr.sh [30] certificate transparency (CT) log search engine. After the collection, our dataset finally obtained 31,211 domain names, among which 19,493 (62.45%) domain names were accessible during our experiment period, and 11,718 (35.54%) domain names suffered from connection issues.

Table 1: The overall HTTP(S) connectivity.

HTTP	HTTPS	Number
✓	✗	7,820 (40.12%)
✓	✓	5,376 (27.58%)
✓[1]	✓	5,284 (27.11%)
✓	✓[2]	503 (2.58%)
✗	✓	420 (2.15%)
✓[1]	✓[2]	90 (0.46%)

[1]: redirected to HTTPS;
[2]: redirected to HTTP;

Next, we examined the HTTP(S) connectivity of the 19,493 accessible domain names. We utilized Requests [31] tool to access a domain name through both HTTP and HTTPS protocol to check its connectivity. Table 1 shows the overall status of HTTP/HTTPS connectivity in our first snapshot (Nov. 8th, 2020) of our dataset. In all 19,493 accessible domain names, we found 11,673 (59.88%) of them supported HTTPS connection. In detail, 420 domain names could only be connected through HTTPS. The HTTP connections of 5,284 domain names were directed HTTPS. 5,376 supported both HTTP and HTTPS, and 503 domain names redirected HTTPS connections to HTTP. Note that HTTP and HTTPS connections of 90 domain names were both redirected. This would not cause loop redirects, as at least one of them was finally redirected to other domain name.

The last step of our investigation checks the deployment correctness of each certificate (binding to one or multiple domain names). Particularly, we developed an analysis tool based on OpenSSL to first capture the certificate and then inspect whether its deployment suffered from five typical issues (Section 4.2) due to the evolution of HTTPS ecosystem. Note that in our investigation we did not only capture a snapshot of the entire analyzed domain names/certificates, but instead we kept monitoring their status and changes throughout a period of 12 weeks. That is, we captured a snapshot of all collected domain names once a week from November 4th, 2020 to January 20th, 2021. On average, for each snapshot a two-day analysis was required to obtain a 76 MB data that covers all those domain names.

4.2 Certificate Deployment Issues

In the following, we answered the proposed Research Questions 1 by elaborating our evaluation focusing on five new HTTPS certificate deployment issues over the past five years:

- **Subject Alternative Name (SAN) Mismatching.** If the tested domain name does not match anyone of the `DNS Names` listed in the SAN field of the HTTPS certificate, we consider the deployment as an SAN mismatching (which would be alerted by most web browsers). Note that if the domain name only matches the `commonName` field in the certificate, browsers nowadays would still reject the certificate because they only check SAN field [32] [33] since 2016.
- **Long Validity Period.** Modern browsers require a certificate *issued on or after September 1st, 2020* should not possess a validity period longer than 398 days [34] [35] [36]. In our investigation, we consider a certificate with a long validity period (i.e., 398+ days) as an incorrectly deployed one. And in particular, for the certificates *issued before September 1st, 2020*, we consider an 825+ days period (as CA/Browser Forum [37] suggested) as potentially risky.
- **Broken Certificate Chain.** If the certificates provided by the domain names could not link the leaf certificate to the root certificate, we would think the deployment has the broken certificate chain issue. Browsers such as Firefox (lower than Version 75) [38] directly block HTTPS connections due to the lack of trust. Though in 2019 Intermediate Certificate Preloading adopted by Firefox (upper than Version 75) help fix this issue [38], technologies that are still in use (e.g., AIA Fetching technology and Intermediate Certificate Cache technology) are considered to violate user privacy [9] [10].
- **Certificate Opacity.** To allow users to audit the reliability of the used certificate, a domain name must deliver the SCTs as well as the certificate during a TLS handshake. According to regulations of Chrome [39] and Safari [40],

we consider a certificate *issued on or after April 2018* without delivering SCT as invalid.

– **Obsolete Crypto Algorithms.** As practical collision of MD5 and SHA-1 had already been found [41] [42], both crypto algorithms have been abandoned by browsers since 2017 [43] [44]. Therefore, we consider a certificate signed by them as invalid. Moreover, NIST have recommended that the public key length of RSA should be longer than 1024 bits [45], and crypto suites using RSA-1024 or lower are also considered as deprecated in our investigation.

To evaluate whether a domain name (and its HTTPS certificate) has suffered the above five issues, in our investigation we first fetched the certificate of the domain name. We used the 98 root certificates trusted by Chrome Root Store [46], Apple OS [47], Mozilla [48], and Windows Root Certificate Program [49] as our trusted root certificate list. To evaluate the broken certificate chain issue, we used the error message of OpenSSL to verify the certificate chain. For the other four issues, we simply used pyOpenSSL [50] to parse the certificate, extracted information of each field and then compared them with our pre-defined criterion. This helped us judge whether the certificate suffered from any deployment issues. In particular, since the most common way to deliver the SCT is to attach it as an extension of the certificate [22], we directly check the SCT list in certificate extensions to obtain the SCT information.

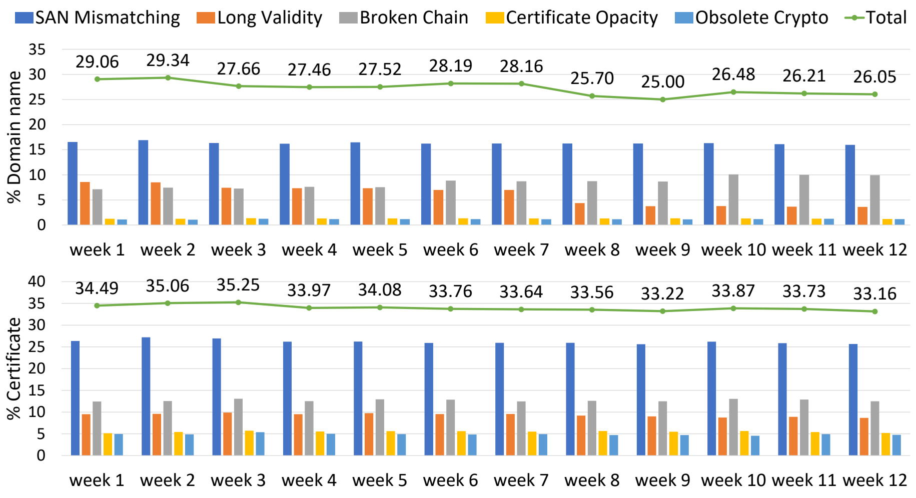


Fig. 1: Deployment Issues

Overall Findings In summary, we found during the entire 12-week investigation period, around 27% domain names with nearly 33% certificates suffered from at least one deployment issues. Figure 1 depicts the overall status during the 12 weeks. Our observation is that during our experiment, the ratio of incorrectly deployed certificates/domains is stable: it only slightly decreased between week 8 to week 9 due to the replacement of long-term certificates to short-term certificates of 306 domain names, and rebounded again in week 10 for 107 domain names emerged broken certificate chain issues. Among all issues, the most frequently occurred one is SAN mismatching, 16% domain names suffered from this issue. 9% certificates had excessively long validity duration. 8% domain names installed broken certificate chain. 5% of the evaluated certificates did not set SCTs, and 5% were signed by obsolete algorithms or had short public key. In the following, we discuss each issue of incorrectly deployed certificate, respectively.

SAN Mismatching On average, we found nearly 16% domain names mismatched their certificates. Among the certificate mismatched by their domain names, we found 90 (8.40%) certificates of 252 domain names set CN field but not SAN field. Lack of SAN or SAN mismatching would cause the identity of the domain name to be unauthenticated, and the browsers would also prevent users from accessing the domain name.

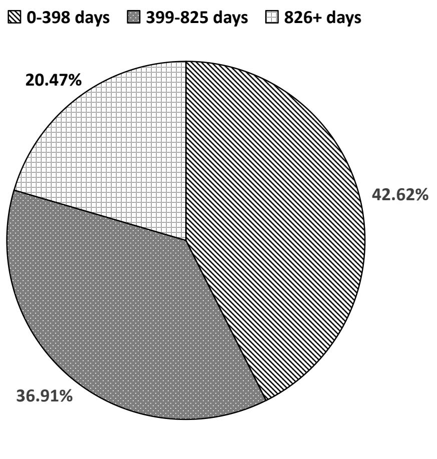
Table 2: SAN mismatching status of domain names and certificates (Nov. 8th).

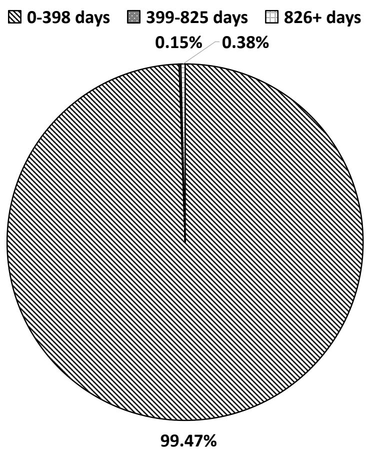
Certificate Type	**Certificates (Mismatched)**	**Domains (Affected)**
Single-domain	625 (64)	778 (182)
Multi-domain	100 (37)	1,978 (666)
Wildcard	256 (92)	8,665 (832)
Total	**981 (193)**	**11,421 (1,680)**

Moreover, we classified the certificates with SAN field into three types. As shown in Table 2, wildcard certificates were wildly installed by most domain names (77.62%), and the configurations of wildcard certificates had the lowest mismatch rate (8.76%). This indicates that wildcard certificates could match most of the using domain names. Another interesting insight is that 1,171 domain names of 1,680 mismatching domain names (69.70%) were multiple level subdomain names but the SAN field could only match the first-level subdomain names: wildcard certificate (e.g., `*.pku.edu.cn`) only matches first level subdomain name (e.g., `mail.pku.edu.cn`) but does not match second level subdomains (e.g., `its.lb.pku.edu.cn`). This implies that many administrators misunderstand the semantic of a wildcard certificate.

Long Validity Period Among 12 weeks, we found 127 certificates with long validity period, which used by 1,067 domain names (i.e., more than 398 days or 825 days). Since the 398-day limit was enforced since *September 1st, 2020* [34] [35] [36], we further divided the certificates into two categories: issued before and after September 1, 2020. Figure 2 shows the distribution of certificate lifespans in 12 weeks, respectively. We found before September 1st, 2020, only 42.62% issued certificates adopted a validity period less than 398 days. In comparison, after that date, 99.47% certificates followed the new regulation. This indicates that most CAs have shortened the validity period of the certificate.

As Figure 1 shows, the ratio of long-term certificates adoption kept decreasing. We observed especially in week 8 (Dec 23rd, 2020), the renewal of seven certificates significantly affected 306 domain names. However, certificates with short validity period introduced new issues. We observed that administrators often forgot to renew certificates, and thus a certificate with shorter lifespan has a larger chance to be used in real world after its expiration: the distribution of 125 expired certificates (230 domain names) in our first snapshot (Nov. 8th, 2020) demonstrates that nearly 70% expired but still-in-use certificates possessed a short validity period.





(a) Certificate Lifespans (Before Sep. 1st) (b) Certificate Lifespans (After Sep. 1st)

Fig. 2: Distribution of Certificate Lifespans

Broken Certificate Chain We found around administrators of 8% domain names did not install intermediate certificates. In week 10 (Jan 6th, 2021), the number of domain names with the broken certificate chain issue increased from 1,042 to 1,211. For domain names that have newly generated the issue, we found 98.16% of them were caused by certificate updates.

We evaluated the cause of this issues. We found 89.00% domain names with this issue only provided the leaf certificates. This indicates that the main reason for this issue is that the administrators have no idea of installing a complete certificate chain. And the left 11.00% domain names offered wrong intermediate certificates. Among them, most domain names (90.23%) transferred their leaf certificates as intermediate certificates. 9.77% domain names provided the intermediate certificates of previously used certificates, which indicates that administrators forgot to change the certificate chain after updating. These suggest that administrators would not recheck the certificate chains after deployment.

Certificate Opacity Most modern browsers require certificates *issued on or after April 2018* to provide Signed Certificate Timestamp (SCT), otherwise they fail to trust those certificates. Using certificate extension is the most common way to support certificate transparency (CT) policy. Among 1,789 certificates issued after the specified date, we observed 1,704 (95.25%) set SCT field in certificate extensions, and they were all issued by trustful CAs. This implies that most CAs support Certificate Transparency. The left 85 certificates without setting SCT field were all self-signed certificates. These certificates were unauditable: The users could not audit whether the certificates were issued incorrectly or the private keys of them were compromised. We also evaluated SCT deployment of the certificates issued before the specified date. We found 199 domain names installed four certificates without setting SCT. They were still considered correct by most browsers. And none of the domain names sent SCT through OCSP stapling or TLS extensions which are the other two ways to deliver SCT. This implies that unless CAs set the SCT, the administrators would not add it on their own.

Table 3: Adoption of Signature Algorithms.

Algorithm	**Certificates**	**Domain Names**
SHA256-RSA	1,662	12,865
SHA384-ECDSA	171	88
SHA256-ECDSA	5	5
SHA384-RSA	3	2
SHA1-RSA	**68**	**164**
MD5-RSA	**2**	**8**

Table 4: Adoption of Public Keys.

Public Key Size	**Certificates**	**Domain Names**
RSA 2048	1,191	12,298
RSA 4096	447	660
ECDSA 256	198	108
RSA 3072	20	13
ECDSA 384	11	8
RSA 1024	**44**	**122**

Obsolete Crypto Algorithms Among the 1911 certificates collected, we found 70 certificates (3.66%) used obsolete signature algorithms and 44 certificates (2.30%) had insecure public key length. Table 3 shows signature hash algorithms used by 1,911 certificates. Among these certificates, crypto suites of 70 certificates (3.66%) were considered to be weak (`MD5 with RSA` of two certificates and `SHA-1 with RSA` of 68 certificates), which would cause certificates to be forged. The result shows that MD5 algorithm has been deprecated, but `SHA-1` was still trusted over some administrator. In Table 4, we found the public key of 44 certificates (2.30%) is RSA 1024 which is in-approval by NIST [45]. We also observed that the above invalid certificates were all self-signed certificates, which indicates that the administrators choose the incorrect algorithms or public key sizes manually. This implies that certificates can be easily misconfigured by administrators.

4.3 Influence of Usability Factors

We answered the proposed Research Questions 2 in this section by analyzing the potential connection between four usability factors and certificate deployment correctness. In detail, we consider four usability factors: the type of certificate indicators, the selection of certificate issuers, the sharing of certificate between multiple domain names, and the revocation option made by certificate applicants.

Certificate Indicator Excluding self-signed (or self-signed in certificate chain) certificates, we classified certificates collected in the dataset of November 8th into three types: EV, IV/OV, and DV. In detail, we identified EV certificates by checking whether `Certificate Policy` field of a certificate belonged to EV policy Object Identifier (OID) set [51]. And for the left certificates, we utilized `Subject` field for classification, as IV/OV certificates contain more information than DV certificates. As shown in Figure 3, we found most of the certificates (75.94%) were DV, while IV/OV certificates were wildly installed on most domain names(65.48%).

We separately evaluated the deployment status of each type certificates. Though most CAs stated that EV certificates generally have the most quality warranty, followed by IV/OV, and finally DV. However in Figure 3(a), we found the proportion of invalid deployment in EV certificates is the highest (83.70%) and DV certificates have the lowest invalid proportion (24.99%). For IV/OV certificates, we found 98.97% of the domain names share certificates with others, and 55.09% certificates are deployed incorrectly as shown in Figure 3(b). This indicates that administrators usually deploy an IV/OV certificate on multiple domain names with deployment issues.

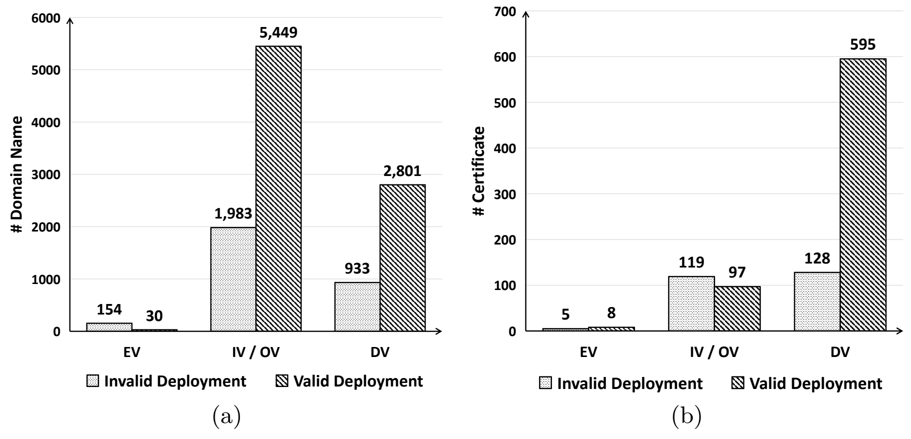
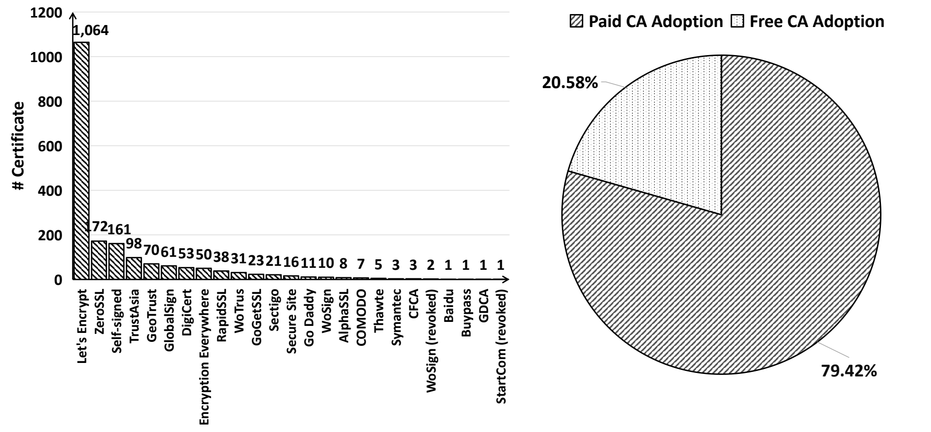


Fig. 3: Distribution of Certificate Types (Nov. 8th)

Certificate Issuer We also checked the connections between certificate issuers and deployment conditions. We extracted the name of CA from `commonName` in `Issuer Name` field of the certificates. Figure 4(a) shows the distribution of involved certificate issuers in our evaluation. 1,750 (91.57%) certificates were issued by trustful CAs, 161 certificates were self-signed certificates and the issuers of three certificates were distrusted by most browsers. As Figure 4(a) depicts, the most frequently used CAs is `Let's Encrypt` (55.68% are issued by this certificate authority).

Among 1,911 certificates, we found 1,441 free certificates (75.41%) used by 5,808 domains. As Section 2 introduced, free certificates usually provide automated deployment services. To further investigate whether this mechanism could reduce the probability of misconfigurations, we visualized the relationship between issuer mechanism and invalid deployment in Figure 4(b). We notice that nearly 80% domain names with invalid deployment applied certificates from paid

CAs. This indicates that the mechanism used by free-of-charge CA could help administrators deploy the certificates correctly.



(a) Certificate Authorities (b) Invalid Domain Names (Nov.8th)

Fig. 4: Certificate Authorities

Certificate Sharing We noted that the domain names tend to share the same certificates with other domain names. Overall, we observed that 11,676 domain names (89.03%) reused the same public key of 472 certificates (24.70%) with others. Among 472 certificates, the deployment of 311 certificates (65.89%) were invalid. Especially, we found 31 certificates are reused by 529 domain names crossing 82 universities, and all of the certificates had deployment issues.

Certificate sharing would connect to deployment inconsistencies in following three aspects. The first inconsistency is in the SAN matching. We found 2,695 domain names did not match the SAN field, and they reused 265 certificates with 10,427 domain names (89.30%). The second is certificate chain completeness inconsistency. 1,413 domain names sharing 203 certificates with 5,309 domain names (45.47%) had the broken certificate chain issue. The last one was the inconsistent renewal of expired certificates. We found the renewal status of 2,950 (25.27%) domain names sharing 48 certificates were not inconsistent. Take the 150 domain names of *Yunnan University* (`ynu.edu.cn`) as a example. When the sharing certificate was expired, 141 (94%) domain names renewed the certificate first, while nine of them updated later. The nine domain names included VPN service website and postgraduate management system. These inconsistencies indicate that certificate sharing would cause administrators to be unable to maintain the certificate deployment for all domain names correctly and simultaneously.

Certificate Revocation We found 1,750 certificates issued by trustful CAs provided OCSP information, which allow browsers to check their revocation status. We used OCSP request to check revocation status of the certificates that were not expired. And for the expired ones, we extracted the issuers to evaluate whether their root certificates have been unrecognized by modern browsers. We found no certificates were revoked during their lifespans. The CAs of three expired certificates (i.e., `StartCom` and `WoSign`) were revoked by most browsers, and the certificates have been expired for a long time. This indicates that this passive revocation is imperceptible for the administrators.

We also evaluated the adoption of OCSP Stapling and OCSP Must-Staple. We extracted OCSP Stapling while TLS handshaking. For domain names with OCSP Stapling, we checked whether the certificate extensions set `OID` for OCSP Must-Staple [19]. Among the 11,673 domain names installing 1,750 certificates, we found only 1,828 domain names (15.66%) configured OCSP stapling on the server side, and none of them set OCSP Must-staple in their certificates. As both OCSP Must-staple and OCSP Stapling need to be added manually, no setting OCSP Must-staple and low adoption of OCSP Stapling infer that administrators did not actively support OCSP mechanisms.

4.4 Popular Web Services with Incorrectly Deployed Certificates

In this section, we discuss how widely used web services (e.g., email, identity authentication, VPN, software license authorization) are affected by insecurely deployed HTTPS.

Email Services We found that it is common for universities to delegate their email services to third-party email service providers. However, such delegations often suffer from incorrect sharing. We found mail services of a university with 52,000+ users, a university with 18,000+ users and a university with 29,000+ users were delegated by eNetEase, and they installed the same certificates with mismatching errors. The same situation also occurs in the mail services of a university with 31,000+ users, a university with 40,000+ users and a university with 36,000+ users delegated by Tencent. Although these websites were managed by professional third-party email service providers, their used domain names and certificates were mismatched. For instance, one of the domain names adopted the certificate with SAN field `*.qiye.163.com`. We also found these domain names still supported HTTP access and thus the certificate mismatch issue may not be noticed by those third-party service providers. Denial of access through HTTPS would cause the users to connect the domain name with HTTP, and the attackers could eavesdrop the privacy of users through man-in-the-middle (MitM) attacks.

Software Distribution Services We observed the licensed software (e.g., Microsoft Windows and Office) download website of a university with 42,000+ users, and the VPN software providers of 20 universities (with 661,000+ students and faculties in total) installed extremely insecure certificates with obsolete crypto suites and invalid lifespans. Therefore, a man-in-the-middle attack could be conducted easily against software downloading, and the users may download maliciously modified software. A further manual inspection showed that those certificates were all self-signed, which indicated why they adopted such dangerous settings.

Authenticating Services We found a certificate of a university with 43,000+ users did not set SCT in the certificate extension, and the certificates was shared by 67 domain names related to 11 IP addresses. Most web services provided by these domain names involve the identity authentication of the users (students and faculties). Since this certificate was shared by so many servers, its private key data is very possible to be leaked due to insecure configurations of either the web servers or the web services. Unfortunately, we found this certificate was issued on January 4th, 2018 and its lifespan is 1,155 days, when the SCT has not yet been stipulated as required. Moreover, administrators of those 67 domains did not set SCT in their TLS extensions. Therefore, this certificate was hardly audited and less trustful.

We additionally observed a certificate of a university , issued on July 2015, was passively revoked due to its untrustful issuer (i.e., `StartCom`). Nevertheless, the website was still using this certificate. Considering that the website is providing Campus ID Card authentication service to 52,000+ users, and the untrustful CA would disclose the private key, the security and privacy of its users would severely threatened.

News Portal We found news portal websites of a university with 25,000+ users, a university with 34,000+ users, and a university with 27,000+ users installed OV certificates. These domain names play an important role in the universities, as through them, schools advertise themselves, and students and faculties get notifications. However, the certificate chains of these domain names were broken. As a result, even though latest version of browsers adopt AIA fetching or intermediate certificate cache and may not report the certificate error against this case, some browsers or embedded web components would directly block such HTTPS connections, and users are not able to access these services.

5 Related Work

Certificate Ecosystem Measurement Several work has provided large-scale measurements on certificate ecosystem, especially focusing on the vulnerabilities. Razaghpanah *et al.* [52] analyzed 7,258 Android apps and saw the adoption rate of certificate security measures, such as certificate pinning, was low. Alashwali *et al.* [53] analyzed the difference in TLS security configurations and certificates between two million plain-domains and their equivalent www-domains and they found www-domains tend to have stronger security configurations than their equivalent plain-domains. Singanamalla *et al.* [6] measured HTTPS deployment of government websites across the world and found that many of the domain names use misconfigured certificates, which is the most closest and recent work to our measurements. The difference is that we focused on the new policies and mechanisms of HTTPS certificate deployment in recent years. We analyzed the new polices with a large-scale, persistent measurement on CERNET and unearthed the causes and the consequences of the invalid deployments.

New Mechanism Measurement Prior work also examined the new mechanisms from different perspectives. Some studies [11] found that `Let's Encrypt` and Certbot greatly facilitated the HTTPS certificate ecosystem. And [54] and [55] conducted control trials of HTTPS configuration to evaluate the usability of `Let’s Encrypt` and Certbot in comparison to traditional ways. They found

`Let's Encrypt` did improve the HTTPS deployment. We showed that the free-of-charge and auto-of-deployment mechanism used by `Let's Encrypt` and other CAs is indeed wildly adopted and improves HTTPS certificate deployments. Stark *et al.* [21] evaluated CT adoption on the web from many angles and found CT has been adopted widely and correctly. Nykvist *et al.* [22] analyzed SCT usage among one million domain names and found certificate extension is the major and simplest solution. We found that CA has attached SCT extensions for certificates after April 2018, which greatly reduce the tasks of certificate administrators. However, existing long-term certificates did not set SCT. Chung *et al.* [18] conducted extensive research on web PKI. They found that most certificates and clients support OCSP, while OCSP Must-Staple was not supported by major browsers and servers, and OCSP responders were still not available. We show that OCSP stapling still has low adoption and none of the certificate set OCSP Must-staple.

Study of Certificate Administrator Some work has also been done on professionals and developers associated with certificate deployment and revealed that it is difficult to deploy certificates correctly. Ukrop *et al.* [7] did an empirical study with 75 IT professionals and found that they place too much confidence on self-signed certificates and name constrained certificates, and existing error notification mechanisms need be adjusted. Zeng *et al.* [56] tested the effectiveness of the notifications of HTTPS configurations.They sent secure notifications to the administrators of certificate misconfigured sites in two ways and found that notifications had moderate impact on error recoveries. Krombholz *et al.* [57] did a research of 28 knowledgeable participants related to HTTPS deployment on Apache and interviewed 7 experienced security auditors. And they revealed that making a correct TLS configurations on Apache including certificate installations was complicated. And our analysis show that there are still round 27% domain name administrators misconfigured the HTTPS certificates and this situation was maintained throughout our measurements.

6 Conclusion

In this paper, we first summarized the new trends of HTTPS certificate management over the past five years. We examined more than 30,000 domain names of CERNET belonging to 113 universities in 12 weeks. We found a stable ratio of domain names did not follow the latest guidelines. We also discussed the usability factors of misconfigurations, and observed some factors did relate to the deployment issues. We hope that our research efforts contribute to HTTPS deployment and fosters future work on network security.

References

1. “Usage statistics of Default protocol https for websites.” https://w3techs.com/technologies/details/ce-httpsdefault.
2. “The mozilla observatory,” https://observatory.mozilla.org/.
3. M. E. Acer, E. Stark, A. P. Felt, S. Fahl, R. Bhargava, B. Dev, M. Braithwaite, R. Sleevi, and P. Tabriz, “Where the wild warnings are: Root causes of chrome

Table 5: HTTP(S) connectivity.

HTTP / HTTPS / Date	✓ ✗	✓ ✓	✓[1] ✓	✓ ✓[2]	✗ ✓	✓[1] ✓[2]	✗ ✗
20201104	7,820	5,376	5,284	503	420	90	11,718
20201111	7,843	5,398	5,265	498	415	89	11,696
20201118	8,034	5,569	5,319	494	474	91	11,269
20201125	7,891	5,762	5,330	491	451	87	11,219
20201202	7,966	5,606	5,275	495	455	88	11,299
20201209	7,936	5,632	5,329	492	396	95	11,314
20201216	7,932	5,612	5,371	502	394	93	11,294
20201223	7,906	5,546	5,533	503	339	87	11,380
20201230	7,833	5,541	5,482	500	412	88	11,227
20210106	7,714	5,546	5,398	492	483	89	11,492
20210113	7,753	5,568	5,442	496	522	81	11,381
20210120	7,620	5,615	5,416	500	474	91	11,501

[1]: redirected to HTTPS;
[2]: redirected to HTTP;

https certificate errors," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1407–1420.
4. D. Kumar, Z. Wang, M. Hyder, J. Dickinson, G. Beck, D. Adrian, J. Mason, Z. Durumeric, J. A. Halderman, and M. Bailey, "Tracking certificate misissuance in the wild," in *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 785–798.
5. L. Schwittmann, M. Wander, and T. Weis, "Domain impersonation is feasible: A study of ca domain validation vulnerabilities," in *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2019, pp. 544–559.
6. S. Singanamalla, E. H. B. Jang, R. Anderson, T. Kohno, and K. Heimerl, "Accept the risk and continue: Measuring the long tail of government https adoption," in *Proceedings of the ACM Internet Measurement Conference*, 2020, pp. 577–597.
7. M. Ukrop, L. Kraus, V. Matyas, and H. A. M. Wahsheh, "Will you trust this tls certificate? perceptions of people working in it," in *Proceedings of the 35th Annual Computer Security Applications Conference*, 2019, pp. 718–731.
8. "Ssl pulse." https://www.ssllabs.com/ssl-pulse/.
9. ""all trusted web pki certificate authority certificates known to mozilla will be cached locally"," https://lwn.net/Articles/817182/.
10. "Intermediate ca caching could be used to fingerprint firefox users," https://threatpost.com/intermediate-ca-caching-could-be-used-to-fingerprint-firefox-users/123834/.
11. J. Aas, R. Barnes, B. Case, Z. Durumeric, P. Eckersley, A. Flores-López, J. A. Halderman, J. Hoffman-Andrews, J. Kasten, E. Rescorla *et al.*, "Let's encrypt: An automated certificate authority to encrypt the entire web," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 2473–2487.
12. "2020 isrg annual report." https://www.abetterinternet.org/documents/2020-ISRG-Annual-Report.pdf.
13. "Instagram forgets to renew its ssl certificate." https://news.netcraft.com/archives/2015/04/30/instagram-forgets-to-renew-its-ssl-certificate.html.
14. "EV UI Moving to Page Info," https://chromium.googlesource.com/chromium/src/+/HEAD/docs/security/ev-to-page-info.md.

15. "Improved security and privacy indicators in firefox 70," https://blog.mozilla.org/security/2019/10/15/improved-security-and-privacy-indicators-in-firefox-70/.
16. "Extended validation certificates are dead," https://www.troyhunt.com/extended-validation-certificates-are-dead/.
17. T. Smith, L. Dickinson, and K. Seamons, "Let's revoke: Scalable global certificate revocation," in *27th Annual Network and Distributed System Security Symposium, NDSS*, 2020.
18. T. Chung, J. Lok, B. Chandrasekaran, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, J. Rula, N. Sullivan, and C. Wilson, "Is the web ready for ocsp must-staple?" in *Proceedings of the Internet Measurement Conference 2018*, 2018, pp. 105–118.
19. P. Hallam-Baker, "X. 509v3 transport layer security (tls) feature extension," *RFC 7633*, 2015.
20. "CA/Revocation Checking in Firefox." https://wiki.mozilla.org/CA/Revocation_Checking_in_Firefox#OCSP_Must-staple.
21. E. Stark, R. Sleevi, R. Muminovic, D. O'Brien, E. Messeri, A. P. Felt, B. McMillion, and P. Tabriz, "Does certificate transparency break the web? measuring adoption and error rate," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 211–226.
22. C. Nykvist, L. Sjöström, J. Gustafsson, and N. Carlsson, "Server-side adoption of certificate transparency," in *International Conference on Passive and Active Network Measurement*. Springer, 2018, pp. 186–199.
23. F. Cangialosi, T. Chung, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "Measurement and analysis of private key sharing in the https ecosystem," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 628–640.
24. M. Zhang, X. Zheng, K. Shen, Z. Kong, C. Lu, Y. Wang, H. Duan, S. Hao, B. Liu, and M. Yang, "Talking with familiar strangers: An empirical study on https context confusion attacks," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 1939–1952.
25. M. Squarcina, M. Tempesta, L. Veronese, S. Calzavara, and M. Maffei, "Can i take your subdomain? exploring related-domain attacks in the modern web," *arXiv preprint arXiv:2012.01946*, 2020.
26. "Cernet," http://www.edu.cn/english/.
27. "Project 211." https://en.wikipedia.org/wiki/Project_211.
28. "Network Service." https://its.pku.edu.cn/service_1_dns.jsp.
29. "Sublist3r," https://github.com/aboul3la/Sublist3r.
30. "ctr.sh," https://crt.sh/.
31. "Requests," https://requests.readthedocs.io/en/master/.
32. "Deprecations and Removals in Chrome 58." https://developers.google.com/web/updates/2017/03/chrome-58-deprecations.
33. "Requirements for trusted certificates in iOS 13 and macOS 10.15." https://support.apple.com/en-us/HT210176.
34. "About upcoming limits on trusted certificates," https://support.apple.com/en-us/HT211025.
35. "Certificate lifetimes," https://chromium.googlesource.com/chromium/src/+/master/net/docs/certificate_lifetimes.md.
36. "Reducing tls certificate lifespans to 398 days," https://blog.mozilla.org/security/2020/07/09/reducing-tls-certificate-lifespans-to-398-days/.
37. "Ballot 193 – 825-day certificate lifetimes," https://cabforum.org/2017/03/17/ballot-193-825-day-certificate-lifetimes/.
38. "Preloading Intermediate CA Certificates into Firefox." https://blog.mozilla.org/security/2020/11/13/preloading-intermediate-ca-certificates-into-firefox/.
39. "Certificate transparency," https://chromium.googlesource.com/chromium/src/+/master/net/docs/certificate-transparency.md.

40. "Apple's certificate transparency policy," https://support.apple.com/en-us/HT205280.
41. X. Wang and H. Yu, "How to break md5 and other hash functions," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2005, pp. 19–35.
42. "Announcing the first sha1 collision." https://security.googleblog.com/2017/02/announcing-first-sha1-collision.html.
43. "Apple drops support for sha-1 certificates in macos catalina and ios 13," https://www.macrumors.com/2019/06/06/apple-deprecates-sha1-macos-catalina-ios-13/.
44. "A further update on sha-1 certificates in chrome," https://www.chromium.org/Home/chromium-security/education/tls/sha-1.
45. E. Barker and A. Roginsky, "Transitioning the use of cryptographic algorithms and key lengths," National Institute of Standards and Technology, Tech. Rep., 2018.
46. "Chrome Root Store." https://www.chromium.org/Home/chromium-security/root-ca-policy.
47. "Available trusted root certificates for Apple operating systems." https://support.apple.com/en-us/HT209143.
48. "CA certificates extracted from Mozilla." https://curl.se/docs/caextract.html.
49. "Release notes - Microsoft Trusted Root Certificate Program." https://docs.microsoft.com/en-us/security/trusted-root/release-notes.
50. "pyOpenSSL's documentation." https://www.pyopenssl.org/en/stable/.
51. "ev_root_ca_metadata." https://chromium.googlesource.com/chromium/src/net/+/master/cert/ev_root_ca_metadata.cc.
52. A. Razaghpanah, A. A. Niaki, N. Vallina-Rodriguez, S. Sundaresan, J. Amann, and P. Gill, "Studying tls usage in android apps," in *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*, 2017, pp. 350–362.
53. E. S. Alashwali, P. Szalachowski, and A. Martin, "Does" www." mean better transport layer security?" in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, 2019, pp. 1–7.
54. C. Tiefenau, E. von Zezschwitz, M. Häring, K. Krombholz, and M. Smith, "A usability evaluation of let's encrypt and certbot: Usable security done right," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1971–1988.
55. M. Bernhard, J. Sharman, C. Z. Acemyan, P. Kortum, D. S. Wallach, and J. A. Halderman, "On the usability of https deployment," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–10.
56. E. Zeng, F. Li, E. Stark, A. P. Felt, and P. Tabriz, "Fixing https misconfigurations at scale: An experiment with security notifications," 2019.
57. K. Krombholz, W. Mayer, M. Schmiedecker, and E. Weippl, "" i have no idea what i'm doing"-on the usability of deploying {HTTPS}," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 1339–1356.