The Invisible Side of Certificate Transparency: Exploring the Reliability of Monitors in the Wild

Bingyu Li, Jingqiang Lin, *Senior Member, IEEE,* Fengjun Li, Qiongxiao Wang, Wei Wang, Qi Li, Guangshen Cheng, Jiwu Jing, *Member, IEEE,* and Congli Wang

***Abstract*—To detect fraudulent TLS server certificates and improve the accountability of certification authorities (CAs), certificate transparency (CT) is proposed to record certificates in publicly-visible logs, from which the monitors fetch all certificates and watch for suspicious ones. However, if the monitors, either domain owners themselves or third-party services, fail to return a complete set of certificates issued for a domain of interest, potentially fraudulent certificates may not be detected and then the CT framework becomes less reliable. This paper presents the first systematic study on CT monitors. We analyze the data in 88 public logs and the services of 5 active third-party monitors regarding 3,000,431 certificates of 6,000 selected Alexa Top-1M websites. We find that although CT allows ordinary domain owners to act as monitors, it is impractical for them to perform reliable processing by themselves, due to the rapidly increasing volume of certificates in public logs (e.g., on average about 5 million records or 28.29 GB daily for the minimal set of logs that need to be monitored in 2018, or more than 7 million records per day in 2020, according to the Chrome CT policy). Moreover, our study discloses that (*a*) none of the third-party monitors guarantees to return the complete set of certificates for a domain, and (*b*) for some domains, even the union of the certificates returned by the five third-party monitors can probably be incomplete. As a result, the certificates accepted by CT-enabled browsers are not actually visible to the claimed domain owners, even when CT is adopted with well-functioning logs. The risk of invisible fraudulent certificates in public logs raises doubts on the reliability of CT in practice.**

***Index Terms*—Certificate Transparency, Trust Management, Public Key Infrastructure.**

This work was supported in part by the National Natural Science Foundation of China under Grant 62002011, Grant 61772518, and Grant 62132011, China Postdoctoral Science Foundation under Grant 2021T140042 and Grant 2021M690304, Open Project of State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, under Grant 2020-ZD-05, BNRist under Grant BNR2020RC01013, NSF IIS-2014552, DGE-1565570, and the Ripple University Blockchain Research Initiative. *(Corresponding author: Jingqiang Lin.)*

Bingyu Li is affiliated with School of Cyber Science and Technology, Beihang University (BUAA), Beijing, China; E-mail: libingyu@buaa.edu.cn. Jingqiang Lin and Guangshen Cheng are with School of Cyber Security, University of Science and Technology of China; E-mail: linjq@ustc.edu.cn, yygqer200503@mail.ustc.edu.cn. Fengjun Li is with the Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045; E-mail: fli@ku.edu. Qiongxiao Wang, Wei Wang and Congli Wang are with State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences; E-mail: {wangqiongxiao, wangwei, wangcongli}@iie.ac.cn. Qi Li is with Institute for Network Sciences and Cyberspace, Tsinghua University, and Beijing National Research Center for Information Science and Technology (BNRist), Beijing 100084, China; E-mail: qli01@tsinghua.edu.cn. Jiwu Jing is with School of Computer Science and Technology, University of Chinese Academy of Sciences; E-mail: jwjing@ucas.ac.cn.

A preliminary version of this manuscript appeared under the title "Certificate Transparency in the Wild: Exploring the Reliability of Monitors" in Proc. 26th ACM Conference on Computer and Communications Security (CCS 2019) [1].

I. Introduction

TRADITIONAL X.509 public key infrastructures (PKIs) have been built on the foundation that certificate authorizties (CAs) are fully trusted and fully responsible for issuing certificates [2]. However, several security incidents indicate that this trust may be too absolute. Many accredited CAs have been be compromised or deceived to issue fraudulent TLS server certificates [3], [4], [5], [6], [7], [8], [9], which bind a domain name (e.g., www.facebook.com or www.gmail.com) to a pair of keys held by man-in-the-middle (MitM) or impersonation attackers, but not the claimed domain owners.

Several approaches attempt to tame the absolute authority of CAs from different perspectives [10], [11], [12], but none of them is widely deployed [13]. Compared with them, certificate transparency (CT) is a more promising solution to detect fraudulent certificates and improve the accountability of CAs [14], [15]. It has been supported by many mainstream browsers and TLS software, including Chrome [16], Apple platforms [17], Firefox [18], Nginx, OpenSSL, Microsoft AD Certificate Service and Azure Key Vault.

In the CT framework, a certificate is submitted to multiple public servers called *logs* by the CA that issues it or sometimes by the domain owner for which it is issued. In response, the log generates a signed certificate timestamp (SCT). In TLS negotiations, the server certificate is delivered along with SCTs; otherwise, it will be rejected by CT-enabled browsers. CT ensures that any certificate acceptable to CT-enabled browsers/software is recorded in publicly-visible logs, so that it is visible to *monitors* for further checks.

It is worth noting that CT does not prevent a CA from issuing fraudulent certificates. CT logs only record all certificates submitted to them and sign the SCTs, without checking whether a certificate is issued with the domain owner's authorization. Hence, a fraudulent certificate is still acceptable to CT-enabled browsers after being submitted to the logs by the attacker (e.g., a compromised CA). Moreover, CT does not provide a mechanism to detect the fraudulent certificates itself. Instead, it builds a publicly auditable platform to enable interested parties to identify all the relevant certificates and verify their trustworthiness. The CT framework relies on monitors to retrieve *all* the certificates belonging to the inquired domain in a timely and reliable means to assist the detection of fraudulent certificates. If a fraudulent certificate is missing in the search result returned to users, the attacker could launch MitM or impersonation attacks, without triggering any alert. The longer the fraudulent certificates stay undetected in the system, the

more the damage they may cause to the PKI ecosystem. So the quality of the search results provided by the CT monitors, especially the *completeness* of the results, affects the overall security enhancement by the CT framework.

However, there is little study in the literature about the *reliability* of the services provided by CT monitors. In practice, third-party monitors claim to reliably return all known, unexpired certificates for a domain [19], [20], [21], but none of them provides any mechanism on the completeness of the returned results. This work is among the first to study the reliability of CT monitors. We expect a *reliable* monitor to return the *complete* set of certificates for any inquired domain name. With the assistance of such reliable monitors, a domain owner can quickly identify any suspicious certificates issued for its domain. However, this requires the monitor to monitor a large number of, if not all, public CT logs. In practice, it needs to fetch all certificates at least in the default logs that are pre-included in CT-enabled browsers. Meanwhile, the monitor should process the fetched certificates properly to produce the correct answer for each inquiry in certificate search services.

While the monitors are essential to CT, it remains unclear whether the monitors in the wild provide reliable services. A domain owner is allowed to act as a monitor to watch for certificates related to its domain name [14], [22]. Meanwhile, there are five active *third-party monitors*, [1] crt.sh [23], SSLMate [24], Censys [25], Google Monitor [19] and Facebook Monitor [26], providing certificate search services. They fetch certificates from the public logs and return certificates related to the inquired domain name.

This paper presents the first systematic study on CT monitors. We analyzed the certificates in 88 public logs and the certificate search services of 5 active third-party monitors regarding 3,000,431 certificates of 6,000 selected Alexa Top-1M websites [27]. All data were collected and all certificate searches were conducted on October 27, 2018, except two experiments for ordinary domains in January 2019. Our study uncovers several problems on the implementation and deployment of CT monitors. First, *acting as a monitor raises storage and network bandwidth requirements that are beyond the capacity of most ordinary domain owners.* By October 2018, there are over 2.87 billion certificates in 88 public logs, which consume about 15.86 TB of storage space. Among them, 50 logs servers that accept the certificates trusted by common TLS clients and serve normally (referred as *regular logs* in this paper) maintain 2.77 billion records at a size of 15.31 TB. Moreover, the number is increasing dramatically, at an average rate of 6,542,421 records per day in 88 logs and 6,275,652 per day in 50 regular logs in 2018. It is extremely costly for an ordinary domain owner to act as the monitor by itself, to fetch and process the rapidly increasing volume of certificates (about 30 GB per day at least) in the public logs.

Moreover, our study of the third-party monitor services shows that *none of the third-party monitors guarantees to return the complete set of valid certificates recorded in the public logs for a domain*. We studied two groups of domains, one for most popular domains (Alexa Top-1K websites) and the other for less popular ones (5,000 domains randomly selected from Alexa Top-1M websites), and searched certificates for all these domains from 5 third-party monitors. In both cases, none of the monitors returned the complete sets of certificates for all the inquired domain names, this also supports our first finding that an ordinary domain owner is less capable of acting as a monitor to process all records in the logs. If the incomplete search results contain some fraudulent certificates in public logs, they will be invisible to the claimed domain owners and thus evade any detection attempt. Therefore, the incompleteness of the research results would degrade the effectiveness of the entire CT framework.

To the best of our knowledge, this is the first work to analyze CT monitors in the wild. Existing studies on CT [28], [29], [30], [31], [13], [32] focus on the deployment of log servers and the adoption in websites or browsers. While providing different views of CT, these large-scale studies do not investigate the reliability of monitors in practice. On the other hand, our study identifies challenges in the implementation of CT monitors and discloses the vulnerabilities in third-party monitors.

In summary, the contributions of this paper are as follow. We perform the first systematic study of the reliability of monitors for the purpose of studying the effectiveness of CT. In particular, (*a*) we investigate various types of log sets and find that each monitor can monitor a minimal set of logs while ensuring the reliability, (*b*) measure the reliability of all mainstream third-party monitors and evaluate the completeness of certificates returned by their certificate search services, and (*c*) analyze the possible causes of incomplete certificate search results and discuss several improvements to enhance the reliability of monitors.

The rest of this paper is organized as follows. Section II describes the CT framework, and Section III presents the requirements of a reliable monitor. Sections IV and V study the rapidly increase of records in public logs, and the defective certificate search services of third-party monitors, respectively. Section VI analyzes the incomplete certificate search results from the third-party monitors. Section VII surveys the related work, and Section VIII concludes this paper.

II. Certificate Transparency

The CT framework [14], [15] records CA-issued certificates in publicly-visible logs. The goal is to make it impossible for a CA to issue TLS server certificates for a domain while keeping them invisible to the claimed domain owner. As shown in Figure 1, CT introduces the following components, in addition to the traditional PKI system:

Log server. A log server maintains append-only logs that record certificates. The logs are publicly-visible, and anyone can fetch certificates from the logs. The records are organized as a Merkle hash tree, and the root node is periodically signed by the log server, called the signed tree head (STH).

Monitor. Monitors regularly watch for suspicious certificates in the public logs. A monitor fetches records from the logs,

[1]There were 10 third-party monitors in the Internet during our data collection and experiments, but three of them did not provide certificate search services, two new third-party monitors started services in 2019, after our experiments.

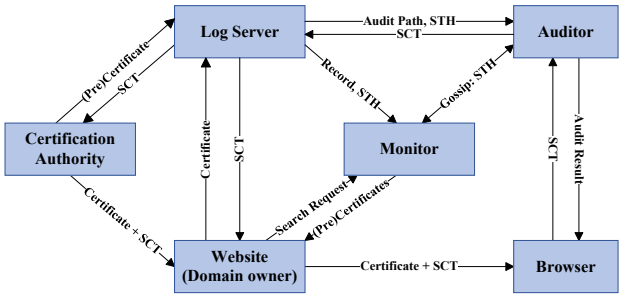


Fig. 1. The framework of certificate transparency

decodes the certificates, and checks certificates of interest. A domain owner may assume the monitor role to search for certificates of interest, and there are also third-party monitors which process the records in public logs to provide certificate search services for users.

Auditor. Auditors ensure the proper behaviors of log servers. By comparing two STHs, an auditor verifies that a log is append-only, i.e., any particular version of the log is a superset of any previous version. It also verifies that each SCT corresponds to a record in the logs by verifying the audit path, the shortest list of additional nodes in the Merkle hash tree to compute the root node.

When a certificate is submitted, the log server responds with an SCT as a promise to append it to the public log within the maximal merge delay (MMD). There are two methods to deliver the SCTs to browsers in TLS negotiations [15]. (*a*) *As TLS extensions*. After a CA issues the certificate for a website, the CA or the website submits it to obtain an SCT, which is delivered as TLS extensions later. (*b*) *Using certificate extensions*. Before a CA issues a certificate, it creates a *precertificate*, which binds the same data but is formatted in a way different from the final certificate. The precertificate is submitted to return SCTs, and then the certificate is issued with the SCTs embedded as a certificate extension. A log may record either a certificate or its corresponding precertificate, or both. In the rest of this paper, we use the term *(pre)certificates* to denote the records in public logs and the raw search results from third-party monitors.

As a decentralized system, CT does not rely on any fully-trusted log server, instead, it employs redundant logs, auditors and monitors, to collectively ensure the trustworthiness of the system [33]. For example, gossip [34], [35] is implemented by exchanging information (e.g., STHs and SCTs) with other components (e.g., monitors, websites, and browsers), to detect the misbehavior of log servers, such as providing inconsistent views to different entities, failing to include submitted (pre)certificates within the MMD, etc.

III. Acting as a Monitor: Requirements and Challenges

The CT framework is proposed to enable the rapid discovery of fraudulent and misissued certificates. In this work, we follow the same threat model and assumptions as the ones adopted by CT [14], [15]. That is, (*a*) the CAs might be compromised or deceived by attackers, so the certificates submitted to the logs might be fraudulent or misissued, and (*b*) the correctness of log servers' behaviors is ensured by redundant auditors and monitors.

If there exists a vulnerability in the implementations of the monitor functionality, the attackers would actively exploit this vulnerability to evade the detection of fraudulent certificates. Such fraudulent certificates, which are recorded in the public logs but actually *invisible* to the public, will be accepted by CT-enabled browsers in the MitM or impersonation attacks. This not only makes the CT framework unreliable, but also may induce more severe problems as CT-compliant certificates are supposed to be more trustworthy.

This work focuses on CT monitors, which are responsible for fetching certificates from the logs and watching for certificate of a domain of interest. As the monitors are essential to facilitate the detection of fraudulent certificates, we expect them to be *reliable*. Although reliability has not yet been formally defined or officially declared as a required property of CT monitors, several third-party monitor services claim to provide reliable certificate monitor functions for users [19], [20], [21] to return *all certificates of a domain* recorded in public logs. In this paper, a reliable monitor is required to return the *complete* set of all certificates issued for the domain of interest.

This requirement of reliability poses two technical challenges. First, a monitor should fetch all certificates that are recorded in public logs. Since certificates are required to be duplicated among logs [17], [36], [33] and the amount of records is increasing dramatically over time, the monitor may not fetch records from all public logs. It may alternatively select an appropriate set of logs to monitor, which provides certain guarantees to the completeness of records and the timeliness of the processing. Moreover, a third-party monitor should return the complete set of all valid (or unexpired) certificates related to any inquired domain name and also its subdomains, which are bound in the (pre)certificates in different forms (e.g., as a wildcard subdomain name), based on the records it fetches.

IV. The (Pre)Certificates in Public Logs

Many organizations operate public CT logs. This section studies these logs, especially the (pre)certificates they maintain, to understand the requirements imposed to an entity assuming the monitor role, from the perspectives of storage and network capacities.

A. Public Logs

We created a list of public logs in October 2018. It includes a total of 88 logs collected from the list maintained by Google [37], and the websites of CA companies and third-party monitors. From the type of (pre)certificates it records [37], a log server falls into one of the four categories: (*a*) *trusted-cert* (72 logs in this category), the logs run for certificates trusted by common TLS clients; (*b*) *untrusted-cert* (only one), for certificates not trusted by common TLS clients; (*c*) *expired-cert* (only one), for expired certificates; and (*d*) *testing* (14), for testing purposes only.

Meanwhile, based on the running status [38], [39], the 88 logs are classified as: (*a*) *good* (40 logs in this category), the servers are running normally; (*b*) *ceased* (19 logs), they are inactive and no longer running; (*c*) *frozen* (3), the servers are up, but do not accept new (pre)certificates any more; (*d*) *warning* (13), the logs are running but with some errors; and (*e*) *pending* (13), they are not monitored by any third-party monitor. Frozen and warning logs hold a large number of (pre)certificates and still provide services, while pending logs are either not publicly announced or for testing purposes only, which are not recommended for regular use [39].

The above categorizations distinguish the public logs according to their functional and operational status. Next, we define two sets of logs that need further investigation.

Regular logs. We call the logs that accept trusted certificates (i.e., trusted-cert logs) and are currently active (i.e., logs in good, frozen or warning status) as *regular logs*. To be accepted by CT-enabled browsers/platforms, a certificate must be publicly-visible in some regular log. This set denotes the *maximal* set of logs that a reliable monitor needs to monitor. Among the 88 public logs, 50 logs are included in this set, which consists of 16 Google-operated logs and 34 non-Google-operated ones.

Google-approved log. Chrome, the first browser that supports CT, pre-installs the public keys of 32 logs to verify the SCTs in TLS negotiations [37]. Among these 32 logs, 6 are now disqualified – five were ceased and one is still running but disqualified. They are still pre-included in Chrome for backward compatibility. So we focus on the remaining 26 qualified logs, and denote them as *Google-approved logs*.

Apple-approved log. Apple did not create its own list of approved logs but used the same list as Chrome until 2019. Now, it enforces its own CT policy. Based on the list we obtained in October 2020, Apple pre-installs the public keys of 63 logs to verify the SCTs in TLS negotiations [17]. We analyzed the Apple-approved logs based on the status of each log server. Among these 63 logs, 3 are disqualified, and 5 are not usable yet because some TLS clients still keep an out-of-date log list and do not accept SCTs from these logs. The (pre)certificates recorded in another 13 logs have expired. So, we focus on the remaining 42 qualified logs, and denote them as *Apple-approved logs* in the remainder. It is worth noting that, there are 12 Apple-approved logs not in the list of regular logs collected in October 2018.

B. The Rapid Increase of (Pre)Certificate Records

We collected the historical STHs of each log, which are archived in SSLMate [40], to explore the amount of records in public logs. Figure 2 shows the numbers of records in the sets of regular logs, Google-approved logs and Apple-approved logs from January 1 to October 27, 2018 (42 weeks in total). These numbers are extracted directly from STHs of the logs, and duplicated (pre)certificates are counted.

Regular logs. Figure 2(a) shows the cumulative growths of records in regular logs, Google-operated regular logs, and non-Google-operated regular logs, respectively. Until October 27, 2018, 2,771,590,678 (pre)certificates are recorded in 50 regular logs, at an average growth of 6,275,652 records per day. This data set contains a large number of duplicated records, since CT policies require each certificate to be submitted to multiple logs [17], [36]. In Figure 2(a) the amount in regular logs increases relatively slowly from January to May 2018, at an average rate of 4,721,304 records per day. The number of records was rather small until 2016 [32]. Since June 2018, the average growth rate significantly increases to 7,778,870 records per day, with about 65% in Google-operated logs. This rapid increase since June 2018 was also reported in a recent study [28]. It is consistent with the mandatory enforcement of CT in Chrome and Apple platforms in 2018 that only CT-compliant certificates will be marked as trusted [17], [16].

We randomly sampled 42,752 records from these logs and the average size of each record is about 5.93 KB. So storing 2,771,590,678 (pre)certificates from all 50 regular logs requires at least 15.31 TB of storage space. Moreover, with the average growth rate of 7,778,870 records per day, it requires an additional storage of 43.99 GB per day for the newly appended (pre)certificates. Meanwhile, downloading only the new (pre)certificates from regular logs demands a 5Mbps network bandwidth dedicated to this task.

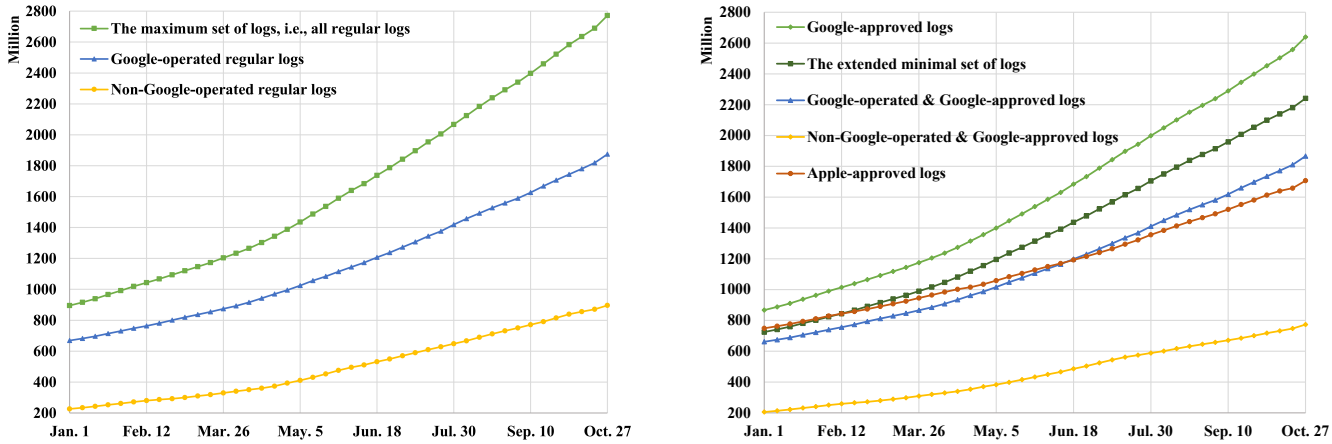
Google-approved logs. Monitoring all 50 regular logs imposes a requirement of very large storage and network bandwidth capacities to a monitor, which is probably beyond the capacity of ordinary domain owners. A reliable monitor may choose to monitor only a subset of important logs, such as the Google-approved logs.

There are 2,639,608,856 (pre)certificates in total in the 26 Google-approved logs until October 27, 2018, which is about 95%[2] of the records in all 50 regular logs. As shown in Figure 2(b), the average growth of certificates in Google-approved logs is about 5,928,983 records per day. In particular, since June 2018, 7,242,755 records (about 40.96 GB) on average are appended every day to these logs. Therefore, monitoring only the Google-approved logs still requires very huge storage and network bandwidth capacities.

Google-operated & Google-approved logs. Among the 26 Google-approved logs, 9 are operated by Google and the other 17 are not. According to the Chrome CT policy [36], a CT-qualified certificate is recorded in at least one Google-operated log and one non-Google-operated log. Thus, to be accepted by Chrome (or other TLS software adopting this policy), a TLS server certificate has to be recorded in some Google-approved log that is operated by Google.

The 9 Google-operated & Google-approved logs compose the *minimal* set of CT logs that need to be monitored by reliable monitors, especially for the certificates compliant with the Chrome CT policy. Less than 0.2% websites adopting the CT framework but *not* complying with this policy [28], so this set works for almost all websites in the wild. As shown in Figure 2(b), monitoring this minimal set still consumes huge storage space and network bandwidth. There are 1,866,390,690 (pre)certificates in these logs by October 27,

[2]The data reported here are before deduplication, so the certificates not in Google-approved logs are actually much fewer than 5%.



(a) The (pre)certificates in 50 regular logs.

(b) The records in Google-approved logs and Apple-approved logs.

Fig. 2. The rapidly-increasing records in the public logs.

2018. In particular, since June 2018, the amount increases at a daily growth of 5,002,599 records (i.e., about 28.29 GB).

Apple-approved logs. Apple enforces a CT policy [17] that a certificate is trusted only if it is sent along with any two or more SCTs from the approved logs. Different from Google, Apple does not impose additional restrictions of the log servers that records certificates. This policy implies that all but not a subset of Apple-approved logs need to be monitored by a reliable monitor.

Until October 27, 2018, there were totally 1,707,049,738 (pre)certificates in 42 Apple-approved logs. As shown in Figure 2(b), the average growth in Apple-approved logs is 3,204,567 records per day, while from June 2018 to October 2018, 3,811,569 records (about 21.56 GB) on average are appended every day to these logs.[3]

Summary. It requires a monitor to process on average 7,778,870 (pre)certificates at a size of 43.99 GB per day for monitoring all regular logs. Alternatively, a monitor processes about 28.29 GB per day for the minimal set of the Chrome CT policy, or 21.56 GB per day following the Apple CT policy. As each certificate has its validity period, renewed certificates will be constantly submitted to the logs. Moreover, the amount will keep increasing as the wide adoption of CT in the future.

The large amounts of (pre)certificates bring challenges for any entity which assumes the monitor role, including (*a*) the huge capacities of storage and bandwidth, and (*b*) the timely processing of the rapidly-increasing data. We believe, the large requirements of storage, bandwidth and processing prevent an ordinary domain owner from acting as the monitor by itself. In fact, even professional third-party monitors have problems in meeting the second requirement and have to keep lots of fetched-but-unprocessed data in backlogs. For example, some (pre)certificates have been kept in backlogs by crt.sh for a long period of time (several days or even over a year) [23].

[3]The total number and the growth in the Apple-approved set are lower than Google-approved logs, because in 2018 11 Apple-approved logs have not started to work yet.

C. The CAs Accepted by Public Logs

Mainstream browsers and platforms pre-install a number of root CAs that they trust by default, called *mainstream CAs* in this paper. There are 371 root CA certificates in Microsoft Windows, 168 in Apple macOS, and 148 in Mozilla NSS.[4] The union[5] consists of 386 root CAs [41], [42], [43] in October 2018. Next, we study the coverage of mainstream CAs accepted by logs and derive a practical minimal set of logs that we recommend the monitors to monitor.

1) The support to mainstream CAs: Each log holds an accept list of CAs and accepts only the (pre)certificates issued by these CAs. Using the *get-roots* command [15], we obtained the list of root CAs accepted by each log in October 2018. Table I lists the number of unique CAs accepted by each set of logs (denoted as $\#_{CA}$) and the number of unique CAs *not* accepted (denoted as $\#^{\ominus}_{CA}$).

As shown in Table I, 50 regular logs support 581 unique root CA certificates in total, among which 381 belong to mainstream CAs. While 200 root CA certificates accepted by the regular logs do not belong to current mainstream CAs, 5 mainstream CAs are *not* accepted by any regular log. Most of these 200 certificates are expired certificates of mainstream CAs. A log operator rarely removes expired CA certificates from the accept list [44], mainly for compatibility purposes. Meanwhile, the log operators may include some CAs trusted by some platforms other than Microsoft Windows, Mozilla NSS, and Apple macOS [44].

However, it is problematic if any mainstream CA is not accepted by the regular logs or the Google-operated regular logs. Totally 9 mainstream CAs are not accepted by any Google-operated log (see Table XIII in Appendix B for details), and 5 out of these 9 CAs are not accepted by any

[4]In the certificate trust list of Mozilla, the *intermediate* CA "GlobalSign Extended Validation CA - SHA256 - G2" is tracked as a trust anchor, but not in the root CA list of others. Its parent root CA is trusted by Microsoft Windows and Apple macOS, which is accepted by Google-operated & Google-approved logs. We ignore this intermediate CA, for its parent root CA has been counted.

[5]Chrome uses the CA list of Mozilla NSS as its certificate trust list on Linux, and on Windows it directly adopts the root CA storage of operating system.

TABLE I
THE CAS ACCEPTED BY LOG SERVERS.

Log server set	$\#_{Log}$	$\#_{CA}$	$\#^{\ominus}_{CA}$
Regular	50	581	5
Google-approved	26	580	5
Google-operated	25	727	9
Google-operated & Google-approved	9	537	9
The extended minimal set	9+2	542	5
Apple-approved	42	624	$0^{\dagger}$

of the regular logs. However, because the Chrome CT policy requires that, in TLS negotiations, at least one SCT is signed by Google-operated logs [36], the certificates issued by these 9 mainstream CAs will always be rejected by Chrome and any other CT-enabled software adopting this policy. On the other hand, the certificates issued by these 5 mainstream CAs which are unsupported by any regular log, will be rejected by Chrome always, but acceptable to browsers that currently do not support CT such as Microsoft IE and Edge. These mainstream CAs are excluded from the CT framework, so any fraudulent certificates issued by them will not be detected. It is unclear why they are not accepted by the regular logs, but it raises potential issues that need to be addressed to facilitate the wide adoption of CT.

The Apple-approved logs were collected in October 2020, and we also re-created the list of mainstream CAs in October 2020. The 42 Apple-approved logs accept 624 unique root CA certificates in total, and cover all 341 mainstream CAs (see Section IV-C3 for details).

Summary. The CT framework improves the accountability of CAs, but its coverage of CAs is still insufficient. Although it is widely agreed that a log server operating in the public interest should accept any publicly-trusted CA pre-installed in major browsers [15], [44], some publicly-trusted mainstream CAs have not accepted by any of the public logs. To improve the accountability of the TLS/HTTPS ecosystem, the 9 mainstream CAs that are currently not accepted by Google-operated logs should be accepted by some of them in the future. This will inevitably impose higher requirements on reliable monitors, because more (pre)certificates will be recorded in public logs as more CAs are accepted.

2) The extended minimal set of logs: As discussed above, the minimal set of logs should be extended to support more mainstream CAs that are currently not accepted. In particular, among the 9 unsupported CAs, 4 of them are accepted by non-Google-operated regular logs but not by the Google-operated ones. To support these publicly-trusted mainstream CAs, we have to include some non-Google-operated regular logs into the minimal set. Fortunately, we do not need to extend the minimal set for the Apple-approved logs, because *all* the mainstream CAs are accepted by the set of 42 Apple-approved logs.

A CA accepted by a log server does *not* mean all (pre)certificates issued by the CA will be recorded in the log. It only means, if a (pre)certificate issued by the CA is submitted to the log, it will be accepted. Thus, in order to fetch as complete (pre)certificates issued by certain CAs as possible, we need to monitor more active regular logs. Since (pre)certificates are duplicated across the non-Google-operated logs, we consider two criteria for selecting more logs: (*a*) the logs with larger daily growth of appended records; and (*b*) the logs supporting the mainstream CAs that are not accepted by the 9 logs in the minimal set.

We finally identify two non-Google-operated logs to be included into the *extended minimal* set, as shown in Table I. If either one of these two logs is removed from this set, some mainstream CAs will not be accepted. Meanwhile, adding more non-Google-operated logs into this set will not increase the total number of accepted mainstream CAs. Therefore, the final extended minimal set consists of 9 Google-approved & Google-operated logs and 2 non-Google-operated logs, which are also Google-approved. By October 27, 2018, this extended minimal set of logs maintains 2,241,161,280 records in total, and since June 2018, on average 6,096,426 (pre)certificates (about 34.48 GB) per day are submitted to these 9+2 logs.

3) The improvement of CA coverage until October 2020: In May 2019 we posted the incomplete coverage of mainstream CAs by CT logs, in the Google CT Working Group forum [44]. The list of (un)accepted CAs depends entirely on the log operators, and Google-operated logs regularly add root CAs to the list of accepted CAs to include the trusted root CA programs of Apple, Microsoft and Mozilla.

Next, we investigate the improved coverage of mainstream CAs, as CT is adopted more and more widely. First of all, in October 2020 we re-created the list of mainstream CAs. There are 307 root CA certificates in Microsoft Windows, 139 in Mozilla NSS, and 178 in Apple macOS. The union consists of 341 root CAs [41], [42], [43].

There are 13 Google-operated & Google-approved logs in October 2020, and these 13 logs support 581 unique root CAs in total, among which 334 belong to mainstream CAs. Only 7 mainstream CAs are *not* accepted by any Google-operated & Google-approved log. All these 7 CAs are trusted *only* in Microsoft Windows, and they are pre-installed *not* for TLS server certificates but for code signing, file system encryption, timestamping, and/or document signing. Meanwhile, the 42 Apple-approved logs accept 624 unique root CA certificates in total, and cover all current 341 mainstream CAs.

Finally, until December 20, 2020, there are 5,893,773,152 (pre)certificates recorded in 13 Google-operated & Google-approved logs, at an average growth of 7,342,465 records per day in 2020. At the same time, 8,109,398,634 (pre)certificates are recorded in the 42 Apple-approved logs, at an average growth of 10,776,617 records (or 60.94 GB) per day in 2020. Thus, monitoring these logs requires even huger storage and network bandwidth, while the CA coverage has been improved from 2018 to 2020.

V. THE INCOMPLETE CERTIFICATES RETURNED BY THIRD-PARTY MONITORS

In this section, we investigate the third-party monitors and evaluate the completeness of their certificate search services. To systematically assess these search services, we conducted experiments for different kinds of domain names. Finally, we

analyzed the search results from the perspectives of issuing CA, certificate type, and certificate format, attempting to understand the missing certificates.

A. Third-Party Monitors

Our study on public logs indicates that it is costly and impractical for ordinary domain owners to act as CT monitors by themselves. A more practical solution is to rely on professional third-party monitors to maintain a *complete* copy of all certificates in public logs and return a *complete* set of all certificates related to a domain name inquired by the owner.

A third-party monitor usually provides the running status of public logs, including STH, uptime, certificate scope, etc. [15], and some of them also provide certificate search services. They fetches (pre)certificates from public logs, and allow the users (e.g., a domain owner) to search for certificates of interest.

There are 10 third-party monitors in the Internet, namely crt.sh, SSLMate, Censys, Google Monitor, Facebook Monitor, CT-Observatory, Edgecombe, Merkle Town, Hardenize and Entrust CT. [6] Among them, CT-Observatory was suspended since September 2018, and Hardenize is still under construction until January 2019. Edgecombe and Merkle Town provide only the running status of logs but not the search service. So this work focuses on the remaining 5 monitors, but this investigation is ready to be extended to explore more monitors when they are available.

B. Exploring the Certificate Search Services Tentatively

We conducted a small-scale experiment, by our domains with known certificates as the controlled input, to explore the search services of 5 third-party monitors. We applied two sets of certificates for 17 subdomains of two second-level domains (i.e., warnings.xyz and wclcttest.cn) from Let's Encrypt [46]. There are in total 102 certificates, and 3 certificates for each subdomain.[7] We searched for these certificates in 5 monitors to find whether and when the monitors return a copy of them.

All 102 certificates (in particular, 102 final certificates and 102 precertificates) are returned from crt.sh, SSLMate, Censys, and Google Monitor. Every (pre)certificate is returned within less than 8 hours after it was submitted to the logs. However, Facebook Monitor does not return all certificates until February 6, 2019 (about four weeks after the certificate issuance). As shown in Table II, 18 certificates are missing in the first set, and 3 are missing in the second.[8] Also, the processing delay in Facebook Monitor is very large – some (pre)certificates are returned after it has been submitted to the logs for about 55 hours.

[6]Entrust CT search started to work in February 2018, but it is now still in intermittent availability due to maintenance [45]. So our experiments do not include it. Moreover, our small-scale preliminary experiments in 2020 show that Entrust CT search misses about 56-64% (pre)certificates, even worse than Google and Facebook Monitors.

[7]Let's Encrypt enforces an upper limit of the certificates that a domain owner can apply in one week. So, we applied the first set of 51 certificates on January 8, 2019, and the second set on January 14, 2019.

[8]We contacted Facebook for this problem on January 17, 2019, but did not receive any meaningful reply. We regularly checked the results returned by Facebook Monitor and found it returned all 102 certificates after May 2019. It indicates a processing delay of more than four weeks.

TABLE II
THE INCOMPLETE RESULTS FROM FACEBOOK MONITOR.

	$\#_{pre}$	$\#_{final}$	$\#_{unicert}$	$\#_{domain}$
Certificate set 1	22	25	33	8
Certificate set 2	13	45	48	14

$\#_{pre}$ and $\#_{final}$ denote the number of *precertificates* and *final certificates* returned in the raw results, out of the 51 (pre)certificates in logs; $\#_{unicert}$ denotes the number of *unique* certificates in the results; $\#_{domain}$ denotes the number of domains with *incomplete* results, out of the 17 domains.

This tentative small-scale experiment, exposes a critical problem of existing third-party monitors (e.g., Facebook Monitor) that they may not return all the certificates issued for a certain domain. This motivates us to conduct a systematic study on the certificate search services and uncover potential issues if any.

C. Searching for the Certificates of Alexa Top Websites

To systematically assess the performance of the search services of 5 monitors, we conducted two sets of experiments to search for the certificates issued for Alexa Top-1K and Top-1M websites, respectively. The rationale is to study if each monitor returns a complete set of certificates for popular domains (with hundreds of certificates per domain) and ordinary domains (with 1–100 certificates per domain), for fraudulent certificates are detected to be issued binding the domain names of both popular websites [4], [5] and ordinary ones [7], [8]. Some incidents have even shown that the attack would affect an ordinary domain longer than popular domains [9].

1) The certificate search services: We describe the certificate search interfaces of the 5 third-party monitors under study.

crt.sh. The SQL interface is provided. The search is performed by comparing (*a*) commonName (CN) and organizationUnitName (OU) in the subject field, and (*b*) dNSName, emailAddress and iPAddress in the certificate extension of subject alternative name (SAN).

SSLMate. It provides an HTTP GET/POST API, and the certificate search is performed by comparing CN in the subject field and dNSName in the SAN extension.

Censys. The Python API is provided. Similar to crt.sh, Censys allows flexible combinations of the subject field and the SAN extension, on which the comparisons are performed.

Google Monitor. It does not provide any API, and we have to access the service on web pages. We developed a tool to access the URLs, and parse the results on web pages. The search is performed by comparing CN and dNSName.

Facebook Monitor. The Facebook Graph API is provided for developers to access Facebook Monitor. The search is performed by comparing CN and dNSName.

While providing free services, the monitors enforce limitations. For example, Censys supports only 250 searches per month, 0.4 action per second, and up to 1,000 records per search for free. Meanwhile, it offers commercial plans. We adopt Censys Pro Plan (25,000 searches per month, up to 25,000 records per search, and 1.0 action per second) in the experiments. Facebook Monitor dynamically controls the search speed based on the number of concurrent users.

TABLE III
THE SERVICE INTERFACES AND SEARCH POLICIES OF THE THIRD-PARTY MONITORS.

	Input vs. Result†			API	Comparison Scope	Valid/Expired Option	Case-Insensitive
	B.A	B.A w/ subdomain opt.	*.B.A				
crt.sh	–a–	–b–	—c–	✓	Customized	✓	✓
SSLMate	a—	abc	—c–	✓	CN & dNSName	✓	✓
Censys	ab–	–	—c–	✓	Customized	•[1]	•[2]
Google Monitor	a–c	–b–	—c–	–	CN & dNSName	✓	✓
Facebook Monitor	ab–	–	abcd	✓	CN & dNSName	–	✓

✓: The feature is supported. •: The feature is partially supported. –: The feature is unsupported, or this part of related certificates is not returned.
†: For a domain, a monitor returns different combinations of some related certificates binding (*a*) the domain name, which is expected; (*b*) any subdomain name, either wildcard or not, expected; (*c*) the related wildcard domain name, expected if the searched domain is third-level or higher-level and its parent domain is not in the TLD list; and (*d*) any unrelated domain or subdomain name matching the input wildcard domain name (e.g., we input *.B.A for C.B.A, but it returns X.B.A, Y.B.A and even W.Z.B.A, etc.), which is unexpected.
1: Censys filters out expired certificates in the results, but it works with delays in marking newly-expired certificates.
2: The service of Censys is case-insensitive, if the input is lower-case; if the input contains any upper-case character, it becomes case-sensitive.

2) Dealing with different search policies: When a domain name is input, we expect a third-party monitor to return all unexpired certificates binding the inquired domain name and its subdomains, in any valid form (e.g., as a wildcard subdomain name).

However, five monitors return different results for a same input of domain name, due to their very different search policies. For fair comparisons and systematical investigations, we need to explore the differences in their search policies, and design strategies for each monitor to conduct customized searches for each inquiry. This ensures the results include all related (pre)certificates from the monitors.

Certificates to be returned. We specify the related certificates of a domain that are expected to be returned. This specification helps us to fairly compare the results returned from five monitors and systematically assess their completeness. Given a domain name such as "B.A", we expect the result includes all certificates in their validity periods and binding any of the following types of domain names in CN or dNSName, in a case-insensitive way: (*a*) the domain name inquired (i.e., B.A); and (*b*) any subdomain name with wildcard or not (i.e., *.B.A, X.B.A, Y.B.A, WWW.B.A, etc.)

In addition, if the inquired domain is a third-level or higher-level domain (e.g., C.B.A) and its parent domain (i.e., B.A) is not in the top-level domain (TLD) list [47], (*c*) any certificate binding the related wildcard domain name (i.e., *.B.A) is also expected to be returned. For example, Alexa Top-1K websites consist of 893 second-level domains and 107 third-level ones, among which 8 third-level domains have a parent domain not in the TLD list. For these domains, we expect the certificates with the corresponding wildcard domain names (e.g., "*.tumblr.com" for "media.tumblr.com").

The search policies of third-party monitors. The monitors do not disclose their search policies in details. We had to exhaustively explore a variation of combinations of comparison statements to infer the policies, e.g., with upper/lower-case (wildcard) domain names of different levels, available options enabled/disabled, etc. We show our findings in Table III to provide an overview of the search policies adopted by the five third-party monitors. In particular,

When a domain name (e.g., B.A) is input without an explicit subdomain option, in addition to the certificates binding that domain name, Censys and Facebook Monitor also return the ones binding subdomain names (i.e., X.B.A, *.B.A, etc.). If it is a third-level or higher-level domain (e.g., C.B.A), Google Monitor also returns the certificates for the related wildcard subdomain name (i.e., *.B.A).

When a domain name is input with the subdomain option (if supported), crt.sh and Google Monitor return only the certificates binding the subdomain names (i.e., X.B.A, *.B.A, etc.), but not the input domain name, while SSLMate returns the ones binding the domain name (i.e., B.A) and any subdomain name (i.e., X.B.A, *.B.A, etc.). If it is a third-level or higher-level domain, SSLMate also returns the certificates binding the related wildcard domain name (i.e., *.B.A).

*When a wildcard domain name is input for the certificates of a domain (e.g., we input *.B.A to search for the certificates of C.B.A),* Facebook Monitor returns the certificates binding any domain name matching the wildcard input (i.e., X.B.A, Y.B.A and even W.Z.B.A), while others return only the ones exactly binding the wildcard domain name (i.e., *.B.A).

Customized comparison statements for five monitors. We composed the comparison statements customized for each monitor according to its search policy. As shown in Table III, to return the expected parts of related certificates for each inquired domain name, we need to conduct (*a*) one search in SSLMate, Censys and Facebook Monitor; (*b*) two searches in crt.sh and Google Monitor; and (*c*) one additional search in crt.sh, Censys and Facebook Monitor for a third-or-higher-level domain whose parent domain is not in the TLD list. Besides, we need to filter out the *unexpected* part by ourselves, after the addition search in Facebook Monitor.

Other pre- and post-processing. The search services of all monitors except Censys are not case-sensitive. Censys is case-insensitive if the inquired domain name is lower-case, but if the input contains any upper-case character, it becomes case-sensitive. So, we converted all inputs into lower-case in our experiments.

crt.sh, SSLMate, Censys, and Google Monitor implement the option to exclude expired certificates, while Facebook Monitor does not. But Censys works with delays in marking an expired certificate, which causes some newly-expired (for one or two days) certificates to appear in the results. So, we need to filter out the expired certificates returned from Censys

TABLE IV
THE RESULTS FOR ALEXA TOP-1K WEBSITES.

	$\#_{cert}$	$\#_{unicert}$	P_{cn}	$\#_{domain}$
crt.sh	407,660	327,019	14.4%	104
SSLMate	201,954	201,954	47.1%	164
Censys	418,382	333,993	12.6%	120
Google Monitor	268,152	181,664	52.3%	546
Facebook Monitor	327,805	252,189	34.0%	289

$\#_{cert}$: the number of valid (pre)certificates searched;
$\#_{unicert}$: the number of *unique* certificates, after we deduplicate the results;
P_{cn}: the proportion of unique certificates *not* returned, compared with the reference sets;
$\#_{domain}$: the number of domains with *incomplete* results, compared with the reference sets (excluding the 18 domains with no certificate).

TABLE V
THE NUMBER OF DOMAINS WITH INCOMPLETE RESULTS IN EACH GROUP OF WEBSITES.

Φ	$= 1$	(1, 10]	(10, 100]	> 100
D_Φ, $\sum = 982$	50	235	471	226
crt.sh	0	8	28	68
SSLMate	0	7	53	104
Censys	0	3	33	84
Google Monitor	0	59	269	218
Facebook Monitor	0	14	102	173

Φ: the range of the number of certificates for a domain;
D_Φ: the number of domains in a group of websites. For every domain in this group, the sum of unique certificates is within the range Φ.

and Facebook Monitor.

Censys returns certificates with a CN containing the string of the input domain name, even when CN is not a domain name. For example, when we search the domain sohu.com, the certificate with CN "Developer ID Application: Sohu.com Inc. (NASDAQ: SOHU) (X3XWZ5HCGK)" appears in the result. In the experiment of Alexa Top-1K websites, 7 certificates are returned for 5 domain names due to this feature. In addition, the results from Censys include some non-publicly-trusted or testing certificates, which are recorded in the untrusted-cert or testing logs it monitors. Such certificates are not accepted by browsers or returned by other monitors, so we filtered them out before the analysis.

3) The incompleteness of search results for Alexa Top-1K websites: We conducted the experiment to search for certificates issued for Alexa Top-1K websites on October 27, 2018. After filtering out invalid TLS server certificates such as expired certificates and code-signing certificates, we obtain a raw dataset with a total of 1,623,953 valid (pre)certificates from 5 third-party monitors.

Deduplicated unique certificates. The raw search results from crt.sh, Censys, Google and Facebook Monitors may include a certificate and its corresponding precertificate at the same time, while SSLMate returns deduplicated results (either the certificate or the equivalent precertificate, but not both). Although a precertificate is invalid in TLS negotiations, it corresponds to a server certificate and the misissuance of precertificates is considered equivalent to the misissuance of the final certificate [15]. So we treat a certificate and its corresponding precertificate as *equivalent* to each other. We define the four-tuple (*NotBefore, NotAfter, SerialNumber, Issuer*) as the index to identify a (pre)certificate, and deduplicate the data from each monitor as well as the union of all searched certificates. Finally, we obtain a dataset of 382,051 deduplicated unique certificates in total for Alexa Top-1K websites.

The reference set. Before analyzing the completeness of the returned results, we face a challenge that it is difficult to obtain the ground truth data about the complete set of valid certificates issued for each inquired domain. In fact, if such ground truth for any domain is easily available, the CT framework is not needed to detect fraudulent certificates. So we define a reference set to approximate the complete set, which is the union of the certificates returned by all five monitors for each inquired domain. Note that conceptually this reference set is only a subset of the real complete set.

Results for Alexa Top-1K websites. We compare the certificates returned from each monitor against the reference set to assess its completeness. For 18 domains among the 1,000 domains, no (pre)certificate is returned by any monitor. We accessed these 18 websites manually, and confirmed that they did not hold valid certificates in public logs. In summary, 10 websites support HTTPS but configure certificates in a problematic means. The remaining 8 websites do not support HTTPS. No certificate is returned for these 18 domains, due to the websites' configurations, but not the monitors' defective processing (see Appendix A for details).

For the other 982 domains, no monitor returns the complete set of known certificates. The results are shown in Table IV. We assess the performance of five monitors from two aspects, (*a*) the proportion of unique certificates *not* returned; and (*b*) the number of domains of which the searched certificates are *incomplete*. Google Monitor is the worst, which fails to return 52.3% of the certificates for 546 domains. Censys yields the best result in terms of the numbers of certificates returned (87.4%), while crt.sh is the best in terms of the numbers of domains with complete results (878 domains). However, even the best services miss 12.6% of certificates and return defective results for 104 out of 982 domains.

Finally, we explore the relationship between the number of certificates a domain holds and the incompleteness of the results. We divide the 982 websites into 4 groups according to the number of valid certificates each domain holds in the reference sets. As shown in Table V, there are 50 domains with only one certificate, 235 domains with more than 1 but less than 10 certificates, 471 domains with more than 10 but less than 100 certificates, and 226 domains with more than 100 certificates. For each group, Table V shows the number of domains with incomplete results from each monitor.

In general, more popular domains (with more certificates) have larger probabilities of incomplete results. Nearly half of the domains have 10–100 certificates. The probability that a domain in this group receives an incomplete result is 26.9%, 32.3%, 27.5%, 49.3%, and 35.3%, from five monitors, respectively. On the other hand, for each of the 50 domains with only one certificate, its certificate is searched successfully by all five monitors. However, for other three groups, no monitor returns the complete results for all domains. Note that,

TABLE VI
THE RESULTS FOR ALEXA TOP-1M WEBSITES.

	$\#_{cert}$	$\#_{unicert}$	$\#_{dn}$	$\#_{unicert/dn}$
Top-1K	1,623,953	382,051	982	389.05
Top-(1K, 5K]	770,257	114,061	968	117.83
Top-(5K, 20K]	341,202	48,869	938	52.10
Top-(20K, 100K]	146,863	19,558	878	22.28
Top-(100K, 500K]	72,610	9,669	834	11.59
Top-(500K, 1M]	45,546	6,462	772	8.37

$\#_{cert}$: the number of valid (pre)certificates searched;
$\#_{unicert}$: the number of unique certificates;
$\#_{dn}$: the number of domains returning at least one (pre)certificate;
$\#_{unicert/dn}$ = $\#_{unicert}$ / $\#_{dn}$: the average number of certificates per domain.

if fraudulent certificates are issued for a domain, the number of certificates related to the domain name will become greater than one.

4) The results for more ordinary domains: Based on the above analysis, we extended our study to more ordinary domains. We conducted the second experiment on January 6, 2019, and randomly selected 1,000 domains from each of the five segments of Alexa Top-1M websites: Top-(1K, 5K], Top-(5K, 20K], Top-(20K, 100K], Top-(100K, 500K], and Top-(500K, 1M].

After the same pre- and post-processing, we collected a total of 1,376,478 (pre)certificates for the 5,000 websites. We deduplicated the data and obtained the reference sets with 198,619 unique certificates. The statistics are shown in Table VI. These are less popular domains, with fewer certificates per domain. In particular, in the Top-(500K, 1M] segment, only 772 out of 1,000 websites hold certificates, and on average each holds less than 9 certificates.

We compare the results returned from each monitor with the reference sets. Table VII lists the proportion of unique certificates that are *not* returned from each monitor. In general, the probability of missing certificates decreases for less popular websites. Censys and crt.sh perform better than others, and in the best case only 0.1% of certificates are not returned by Censys in the searches of Alexa Top-(100K, 500K] websites. Google Monitor is the worst, and at least 6.7% of certificates are missing in these experiments.

Table VII also summarizes the number of domains with incomplete results for each segment. Censys performs better than others, but there are always incomplete results for some websites in each segment of Alexa Top-1M websites. In the best case, 7 out of 1,000 domain names return incomplete certificates from Censys. Facebook Monitor is the worst, and out of the 1,000 domains in each segment, there are always hundreds of domain names not returning complete results.

We particularly study the domains with one certificate and the ones with less than 10 certificates to find if the monitors have any tendency towards missing the certificates of less (or more) popular domains (see Table XIV in Appendix B for details). For less popular sites, Facebook Monitor misses more certificates than others. For ordinary domains, all monitors perform similarly. Overall, Facebook Monitor always returns incomplete results for about 15% of domains, while Censys performs the best and returns incomplete results for only a few domains of each segment.

Table VIII counts the amounts of domains with different number of missing certificates. In general, whether it is a popular or ordinary website with a great or small number of certificates, the number of missing certificates is more likely to be less than 10. For example, the probability that a domain returns an incomplete result with less than 10 missing certificates from a monitor, is 81.7% (i.e., (114+118)/(114+118+43+9) = 81.7%), 77.8%, 81.5%, 67.6%, and 86.5%, respectively. Even in the Top-1K segment, the probability is about 75.0%, 72.0%, 80.0%, 53.3%, and 75.8%, respectively. Meanwhile, in the Top-(500K, 1M) segment, this probability is more than 90% (see Table XV in Appendix B for details). That is, for any monitor, most of the defective certificate search results miss less than 10 certificates for a domain. Our further investigation shows that there is no common feature among them.

5) The results from the perspective of issuing CA: We analyze the missing results from the perspective of issuing CA. If the missing certificates are issued by trusted mainstream CAs, they will be accepted by CT-enabled browsers but invisible to the domain owners; otherwise, these certificates will be rejected by browsers and then the invisibility does not degrade the security in fact. Note that lots of CAs are accepted by public logs, but not included in the list of mainstream CAs (see Section IV-C for details).

A total of 314,344 unique (pre)certificates are missing in the results from some of the third-party monitors. These missing certificates are issued by 87 root CAs: 314,337 (pre)certificates are issued by 86 mainstream CAs, and only 7 are issued by an untrusted CA. It means that almost all missing certificates will be accepted in mainstream platforms. Thus, if any of these 314,337 (pre)certificates was fraudulent, the MitM attacks exploiting such a certificate would succeed without any warnings in TLS sessions and this certificate could probably not be detected by the legitimate domain owner.

Only 7 certificates in the missing set are issued by an untrusted CA,[9] for 7 domain names. These certificates will be rejected by the mainstream platforms. If these 7 certificates are filtered out from the reference set, the number of domains with incomplete results by crt.sh, SSLmate, and Facebook Monitor will be reduced by 4, 3, and 2, respectively, but without any improvement of the results by Google Monitor and Censys.

6) The results from the perspective of certificate type: There are three types of TLS server certificates, with different levels of assurance [48]: (*a*) Domain-validated (DV), only the ownership of the domain name is verified by CAs; (*b*) Organization-validated (OV), more steps are conducted to confirm the existence of the applicant organization; and (*c*) Extended-validation (EV), the most efforts are conducted by the CA to validate the certificate request.

Platforms have different CT policies for different types of certificates, so we wondered whether these different requirements have relationships with the returned/missing results from CT monitors; that is, the certificate type might affect

[9]This is an Apple-operated root CA, and the subject DN is "CN=Developer ID Certification Authority, OU=Apple Certification Authority, O=Apple Inc., C=US". It is somehow strange that it is not in the certificate trust list of Apple macOS [43].

TABLE VII
THE *incomplete* RESULT COMPARED WITH THE REFERENCE SETS, FOR EACH SEGMENT OF ALEXA TOP-1M WEBSITES.

	Top-1K		Top-(1K, 5K]		Top-(5K, 20K]		Top-(20K, 100K]		Top-(100K, 500K]		Top-(500K, 1M]	
	P_{cn}	$\#_{dn}$	P_{cn}	$\#_{dn}$	P_{cn}	$\#_{dn}$	P_{cn}	$\#_{dn}$	P_{cn}	$\#_{dn}$	P_{cn}	$\#_{dn}$
crt.sh	14.4%	104	3.9%	78	0.3%	46	0.4%	29	0.4%	16	0.8%	11
SSLMate	47.1%	164	10.5%	100	1.3%	61	0.7%	33	0.4%	19	0.9%	15
Censys	12.6%	120	0.3%	52	1.0%	27	0.5%	14	0.1%	7	0.2%	7
Google Monitor	52.3%	546	22.2%	421	16.2%	294	10.2%	198	8.6%	117	6.7%	73
Facebook Monitor	34.0%	289	12.9%	307	5.0%	393	5.3%	259	5.7%	226	6.0%	160

P_{cn}: the proportion of unique *certificates not* returned, for each segment of Alexa Top-1M websites;
$\#_{dn}$: the number of domains of which the results are *incomplete* compared with the reference sets, for each segment of Alexa Top-1M websites.

TABLE VIII
THE NUMBER OF DOMAINS WITH DIFFERENT NUMBERS OF MISSING CERTIFICATES, FOR ALEXA TOP-1M WEBSITES.

θ	$=0$	$=1$	(1, 10]	(10, 100]	(100, $+\infty$)
crt.sh	5,088	114	118	43	9
SSLMate	4,980	140	165	66	21
Censys	5,145	89	96	35	7
Google Monitor	3,723	414	700	409	126
Facebook Monitor	3,738	583	831	175	45

θ: the range of the number of missing certificates for a domain.

TABLE IX
THE RESULTS OF ALEXA TOP-1M WEBSITES IN EACH TYPE OF CERTIFICATES.

	$\#_{sct}$	$\#_{domain}$	$\#_{unicert}$	$\#_{dn}$	P_{cn}
EV	3.7	774	27,063	265 (34.2%)	52.3%
OV	3.9	3,683	398,465	1,372 (37.3%)	66.9%
DV	3.4	5,367	155,142	1,847 (34.4%)	59.6%

$\#_{sct}$: the average number of SCTs per certificate;
$\#_{domain}$: the number of domains which return at least one (pre)certificate of this type;
$\#_{unicert}$: the number of *unique* valid (pre)certificates searched;
$\#_{dn}$: the number of domains for which at least one monitor returns *incomplete* results, compared with the reference sets;
P_{cn}: the proportion of unique missing certificates which did not return from at least one monitor, compared with the reference sets.

the performance of the search services. In the beginning of CT enforcements, most platforms display warnings on the violation of CT policies only for EV certificates [16], [17], [28]. Moreover, CT policies enforced by different platforms vary across certificate types and over time. For example, Google requires CT compliance for the EV certificates issued after 1 Jan 2015, or they will not be recognized as EV certificates in Chrome; but this policy was not applied to all types of certificates until April 30, 2018 [16]. Meanwhile, Apple requires that all certificates issued after October 15, 2018 must meet the Apple CT policy, as only EV certificates were required to meet this CT policy in Apple platforms before that day [17]. Because these experiments were carried out in October 2018 and January 2019, some of the certificates collected must be issued before the strict enforcement of CT-compliance verifications.

We explore whether a CA or a third-party monitor processes EV certificates in special ways, including the number of SCTs, and the performance of the search services. Table IX lists the missing results, from the perspective of certificate type. It seems the incomplete results are not related to the certificate type; i.e., for a certificate of any type, the possibility that a monitor does not return this certificate is close (i.e., 52.3% for EV certificates, 66.9% for OV and 59.6% for DV). In terms of the proportion of domains for which some monitor returns incomplete results, the numbers are also very close (i.e., 34.2% for EV certificates, 37.3% for OV and 34.4% for DV). Moreover, a CA does not submit an EV certificate to more log servers (i.e., in average 3.7 SCTs for each EV certificate, while an OV certificate is submitted to 3.9 logs). In summary, EV certificates which are expected to be more secure in practice, do not express any significant priority in the CT framework from our experiments.

We further study the result of each certificate type of that are not returned from each monitor, for each segment of Alexa Top-1M websites (see Table XVI in Appendix B for details). It seems that no monitor gives any priority to EV certificates in the processing: different types of certificates are processed indistinguishably by any of the five monitors.

7) The results from the perspective of certificate format: As mentioned above, the misissuance of precertificates is considered equivalent to the misissuance of the final certificate [15], so we count a certificate and its corresponding precertificate as *equivalent* to each other in Section V-C3.

Next, we try to find whether the certificate format (i.e., a certificate or a precertificate) affects the returned results. Table X lists the proportion of unique *precertificates* that are *not* returned from each monitor compared with the set of all (pre)certificates it misses. Censys and SSLMate perform better than others, and in the best case only 12.3% of results not returned by Censys are precertificates in the searches of Alexa Top-(5K, 20K] websites. In general, Google Monitor is the worst at handling precertificates, and at least 69.6% of missing results in these experiments are precertificates.

Table X also lists the domains which only do not return *precetificates*. In general, Censys and SSLMate perform better, while Google performs worst. But for the less popular sites, Facebook Monitor is the worst. For example, for Alexa Top-(500K, 1M] websites, Facebook Monitor returns incomplete results for 160 domains, among which 151 domains miss only precertificates.

D. Summary and Discussion

Our experiment results uncover that *none* of the five active third-party monitors provides reliable certificate search services that guarantee to return the *complete* set of certificates

TABLE X
THE PROPORTION OF UNIQUE *precertificates not* RETURNED, COMPARED WITH THE SETS OF MISSING RESULTS.

	Top-1K		Top-(1K, 5K]		Top-(5K, 20K]		Top-(20K, 100K]		Top-(100K, 500K]		Top-(500K, 1M]	
	P_{cn}	$\#_{dn!pre}$	P_{cn}	$\#_{dn!pre}$	P_{cn}	$\#_{dn!pre}$	P_{cn}	$\#_{dn!pre}$	P_{cn}	$\#_{dn!pre}$	P_{cn}	$\#_{dn!pre}$
crt.sh	70.4%	30	41.4%	16	35.2%	11	36.8%	9	17.9%	3	39.2%	2
SSLMate	64.4%	21	46.4%	14	12.3%	10	20.0%	8	16.3%	3	33.3%	2
Censys	53.6%	16	38.9%	6	22.3%	4	23.6%	4	14.3%	1	16.7%	0
Google Monitor	76.9%	396	69.6%	347	92.9%	250	93.7%	171	95.1%	104	90.8%	63
Facebook Monitor	67.0%	59	50.1%	55	49.5%	127	54.3%	102	54.1%	87	54.6%	151

P_{cn}: the proportion of unique *precertificates not* returned, for each segment of Alexa Top-1M websites;
$\#_{dn!pre}$: the number of domains that only do not return *precetificates*, compared with the reference sets.

for an inquired domain name. All five monitors demonstrate defects in their search services, in the experiments with both popular websites and less popular ones. This exposes a critical problem that will degrade the reliability of not only the monitor but also the CT framework. If a fraudulent certificate of a certain domain is not returned from the certificate search services, it would never be detected.

This problem may be more severe than we expose as above, because the reference sets we used are only the approximations of the real complete sets for there is no available ground truth for Alexa Top websites. That is, *even the union of results from 5 third-party monitors could not guarantee to include complete certificates for a certain domain.* To illustrate this problem, we construct the base reference sets for Alexa Top-1K websites with only the results from crt.sh, and add the results from SSLMate, Censys, Google Monitor and Facebook Monitor one by one to enlarge the reference sets. The number of deduplicated unique certificates increases from 329,019 to 334,605, 373,634, 376,308 and finally 382,051. Correspondingly, the number of domains with complete results increases from 878 to 888, 949, 954 and finally 982. We argue that, when there are more active third-party monitors (such as CT-Observatory, Hardenize and Entrust CT search, if they provide regular services in the future), or even when the monitors monitor more logs and process more (pre)certificates, the reference sets will probably be larger than the ones in our experiments.

VI. THE CAUSES OF INCOMPLETE CERTIFICATE SEARCH RESULTS

In this section, we discuss the causes of incomplete search results and propose suggestions for more reliable monitors.

A. Potential Causes due to Unmonitored Logs

Since each (pre)certificates is required to be submitted to multiple logs [36], [17] and the public logs maintain a great number of duplicated (pre)certificates, it may not be necessary for a third-party monitor to monitor all logs for efficiency and cost reasons. So, we investigate the set of logs that each monitor (un)monitors to find if the log coverage causes incomplete results. While we believe the third-party monitors, especially the ones with certificate search services, should provide the list of logs they monitor to the public to help assess the quality of their services [22], Google and Facebook Monitors do not disclose such information. In Sections IV-B and IV-C, we investigate various types of log sets and define multiple log sets as reference sets for third-party monitors for various purposes. In Table XI, we summarize the numbers of logs in five different sets that are *not* monitored by each of the remaining 6 monitors.[10]

TABLE XI
THE LOGS (UN)MONITORED BY MONITORS.

	$\#_{Log}$	$\#_{r}^{\ominus}$	$\#_{go}^{\ominus}$	$\#_{ga}^{\ominus}$	$\#_{go+ga}^{\ominus}$	$\#_{e}^{\ominus}$
crt.sh	46	9	8	4	1	1
SSLMate	77	1	1	0	0	0
Censys	46	19	12	5	0	0
CT-Observatory†	20	39	20	18	4	6
Edgecombe	77	2	7	0	0	0
Merkle Town	39	11	10	4	0	0

$\#_{r}^{\ominus}$, $\#_{go}^{\ominus}$, $\#_{ga}^{\ominus}$, $\#_{go+ga}^{\ominus}$, $\#_{e}^{\ominus}$: the numbers of regular logs, Google-operated logs, Google-approved logs, Google-operated & Google-approved logs, and logs in the extended minimal set, which are *not* monitored.
†: The data were collected in August 2018 before it was shut down.

No monitor covers all 50 regular logs, and only SSLMate and Edgecombe monitor all 26 Google-approved logs. We define the extended minimal set of logs as a reference set for third-party monitors to balance the requirement of reliability and the cost. In Table XI, we find that most monitors actually cover more logs than these sets. In particular, among the 3 monitors with active certificate search services, SSLMate and Censys monitor all 9 Google-operated & Google-approved logs as well as all logs in the extended minimal set. crt.sh, on the contrary, only monitors 8 out of 9 Google-operated & Google-approved log.

We compare the logs (un)monitored by monitors and the search results in the experiments of Alexa Top websites. By directly inputting the SHA256 fingerprint of a (pre)certificate (but not the domain name) to crt.sh, Censys or Google Monitor, a user is informed which logs the (pre)certificate is recorded in or fetched from. We utilized this special function of crt.sh, Censys and Google Monitor to analyze in which logs the missing results are recorded, and Table XII lists the missing results of Alexa Top websites. These missing certificates are submitted to at least 47 log servers totally. Among the 47 logs that record these missing certificates, 15 are unmonitored by crt.sh, 13 are unmonitored by SSLMate, and 29 are unmonitored by Censys. Note that Google and Facebook Monitors do not disclose the lists of monitored logs.

[10]This information was collected in 2018, when the certificate search experiments were conducted.

TABLE XII
THE MISSING RESULTS FOR ALEXA TOP-1M WEBSITES.

	$\#_{unicert}$	$\#_{sct}$	$\#_{log}$
Top-1K	260,869	4.1	40
Top-(1K, 5K]	31,874	3.4	29
Top-(5K, 20K]	15,159	3.7	41
Top-(20K, 100K]	3,125	4.2	28
Top-(100K, 500K]	1,696	4.3	24
Top-(500K, 1M]	1,621	5.2	24
Top-1M	314,344	4.0	47

$\#_{unicert}$: the number of *unique* missing certificates which are not returned from at least one monitor, compared with the reference sets;
$\#_{sct}$: the average number of SCTs per missing certificate;
$\#_{log}$: the number of *unique* log servers recording these missing certificates.

However, the missing certificates shall have very little to do with the list of logs unmonitored by a monitor. Among the 47 logs, we identity 10 logs that record the highest number of missing certificates, and 96.6% of missing certificates can be fetched from these 10 logs. Meanwhile, we find that crt.sh, SSLMate and Censys monitor all these 10 logs, but 19.0%, 61.4%, and 15.6% of these missing certificates, are not returned by crt.sh, SSLMate and Censys, respectively. In fact, as shown in Table XI, SSLMate has a better log coverage than crt.sh and Censys, but it misses more certificates than the other two in our experiments, as shown in Table VII. Finally, Table XII shows that a missing certificate is on average recorded in 4.0 log servers, even a little more than the average number of all certificates in the reference sets (which is less than 4.0 for any certificate type in Table IX).

Then, we analyze the relationship between these missing certificates and the 8 Google-operated & Google-approved logs monitored by crt.sh (as well as SSLMate and Censys). We find that each of these missing certificates is submitted to at least one of them. This confirms that the missing (pre)certificates of crt.sh, SSLMate and Censys are not caused by the unmonitored logs.

We further study the observations in the experiments with Google and Facebook Monitors, although they do not disclose the logs they are monitoring. In the preliminary experiments on Alexa Top-1K websites in September 2018, we found that Google Monitor did not return any record in the Argon2019 log. However, in the experiment on October 27, 2018 (in Section V-C), some records in Argon2019 were returned by Google Monitor. It is not possible for us to tell if this is caused by any adjustment of logs monitored by Google Monitor or other issues. The log coverage cannot explain the failure of Facebook Monitor in our experiment with Let's Encrypt in Section V-B. Let's Encrypt submitted the 102 precertificates to 5 logs (i.e., Argon2019 and Icarus operated by Google, and non-Google-operated Nimbus2019, Sabre and Yeti2019). Each precertificate is submitted to at least two and at most five logs, and all the 102 final certificates are submitted to Argon2019. From other (pre)certificates returned from Facebook Monitor, we can tell it is monitoring all the 5 logs. But, we cannot tell why it missed 21 certificates in this experiment.

We also checked the behavior of log servers. Edgecombe [38] exchanges the STHs of most public logs with Chromium STHSets [49], Google [50], and SSLMate, by the gossip validation [34]. From the results disclosed by Edgecombe, there is no log misbehavior that provides inconsistent views to different monitors.

B. Causes due to Monitor Implementation Issues

We analyze the results of each monitor, and discover several issues that may cause the missing records. However, since the internal mechanism and architecture of CT monitors are unknown to the public, we can only study this problem from the perspective of external users. We do believe our findings only reveal a subset of causes, and more unknown bugs or vulnerabilities in the monitors' services exist. For example, all the causes cannot fully explain the missing (pre)certificates in our experiment in Section V-B.

We have reported the issues uncovered in the paper to all five CT monitors, and exchanged detailed experimental results with them from January 2019 to September 2020 (see the remainder for details). SSLMate and crt.sh replied that they would initiate investigations on the reported issues, but we have not received further feedbacks.

Delayed processing. The monitors may not be able to process fetched records in time and thus store them in so-called backlogs. We find the records from some large logs (e.g., Pliot, Rocketeer and Argon2019) are not processed in time by crt.sh. The SCTs indicate that many of the (pre)certificates have been stored in backlogs for several days and even over a year without being processed. The delay causes some incomplete results of crt.sh. We compared the backlogs on August 27 and October 27, 2018, and the former is much larger than the latter. It is consistent with our observation of much more missing certificates in the preliminary experiment in August 2018 – for 500 domains in the Alexa Top-1K list, the returned results were incomplete. There are also delays in the Censys service – some (pre)certificates newly-appended in the logs for one or two days are not returned. Based on this observation, we argue that popular domains with a large number of certificates are likely to have more records in the backlogs than less popular domains, because the former issues certificates more frequently.

Unreturned precertificate. In Tables X Google Monitor misses a significant proportion of precertificates, regardless of the popularity of the domains. We find that, if a certificate is recorded in public logs only in the format of precertificate, it is probably not returned by Google Monitor. But Google Monitor does return some precertificates. For example, for Alexa Top-1K websites, Google Monitor returns incomplete results for 546 domains, among which 396 domains miss only precertificates. These missing precertificates accounts for 76.9% of all missing results of Alexa Top-1K websites from Google Monitor. For Alexa Top-(20K,100K] websites, Google Monitor returns incomplete results for 198 domains, among which 171 domains miss only precertificates; for Alexa Top-(100K,500K] websites, the precertificates accounts for up to 95.1% of all missing results. This indicates the processing of precertificates in Google Monitor may be faulty.

The Google Monitor team disclosed that they fixed a bug in the precertificate processing in July 2019, which caused

about 86% of missing certificates. They also explained that another 10% of missing certificates were caused by the defect of handling the underscore character ('_') in domain names.

Incident not recovered in time. We observe a large number of missing results are related to specific time windows. For example, the SCTs of most (pre)certificates missed by Censys have timestamps between UTC 2017-10-11/12/13/14, and Google Monitor misses many (pre)certificates whose SCTs were signed on UTC 2015-11-20. It is probably due to interruptions during data retrieval or processing. While the causes of the interruption vary, the results indicate that the services of these monitors are not fault-tolerant.

We disclosed our findings to Censys. After their investigation, we were told that this might be caused by the incidents during their data migration in July and August 2018. Our later experiments showed that Censys was fixing this problem gradually – our experiment on Aug 21, 2018 discovered missing certificates for 200 domains among Alexa Top-1K websites, while on October 27, this number was reduced to 120 domains. But the recovery process is taking a longer period than we expect. Until January 2019, we still observed missing certificates with timestamps in these time windows. Besides, a small number of certificates out of these time windows were still not returned, as shown in Table VII, for which Censys did not provide a clear explanation. Besides, they confirmed the problem of marking certificate tags that we reported, such as "(un)expired" tags.

Unsupported domain. SSLMate misses many certificates due to its restricted services to four domains (i.e., amazonaws.com, cloudfront.net, blogspot.com and fbsbx.com). The requests about these domains are directly terminated with the error "not_allowed_by_plan". However, the certificates of these 4 domains are recorded in at least 37 logs, among which SSLMate monitors 27 at a regular basis. SSLMate claims it only accepts domain names of registered domains or their subdomains [24], and deliberately excludes these 4 domains to protect user privacy (e.g., customers' hostnames in amazonaws.com, cloudfront.net, blogspot.com and fbsbx.com) [28]. Similarly, Google Monitor returns no result for blogspot.com and cloudfront.net, but it does not provide any specific explanation or error message.

Facebook Monitor returns the warning, "provided domain is invalid," for some domains (e.g., btrc.gov.bd and sabay.com.kh). Among 6,000 selected Alexa Top-1M websites, the requests of 9 domains are terminated with such errors. We doubt the error is *not* caused by the privacy concerns, since Facebook Monitor returns non-empty results for domains with obvious privacy problems (e.g., cloudfront.net and blogspot.com). Moreover, Facebook Monitor returns no results for many other domains.

Interface limitation. Some monitors limit the number of records returned in each search. Censys has a restriction of at most 25,000 records per search and Facebook Monitor sets this value to 5,000. In SSLMate, a query of zendesk.com, for which many (pre)certificates are supposed to be returned, receives the error "This query took too long to complete."

Any failure to meet this requirement causes missing results. However, several domains have certificates larger than this limit. For example, among Alexa Top-1K websites, there are 4 domains exceeding the 250,000-record limit, i.e., amazonaws.com (43,306), zendesk.com (34,341), cisco.com (39,045), and att.com (32,496), and 29 domains exceeding the 5,000 limit. When the relevant records exceed the limits, one has to issue multiple queries and combine the results, which may result in some (pre)certificates overlapped or missing.

In summary, in the experiments, the interface limitation causes a number of certificates not returned, but its impact is limited - it affects only a very small amount of domains.

C. Potential Countermeasures

Our experiments show that the third-party monitors are not fault-tolerant, due to the scale of the (pre)certificates and the complexity in processing them. The monitors, like other CT components, should not be assumed by default as fully trustworthy, since CT is by nature a decentralized system. From these considerations, we propose to design countermeasures to improve the reliability of CT monitors.

Reliability audit of monitor services. CT auditors and the gossip protocol [34] are designed to detect the misbehavior of CT logs. We believe a similar audit mechanism should be implemented to detect the misbehavior or problematic behavior of CT monitors, especially on service reliability. In particular, we propose to regularly evaluate the state of online monitor services and identify the problems in these services through two black-box testing approaches.

The first approach, similar to the experiments in this paper, is to periodically collect certificate search results for the same domain name from multiple monitors to generate reference sets for tested domains. We can start from the set of all logs - the maximal set of logs supporting the reliability that we expect - to derive a conceptual, minimal set of logs with the same expected reliability guarantee. Meanwhile, the monitors have full flexibility to re-define their own reference sets that reflect their considerations from multiple perspectives, such as the types of certificates recorded by the log, the running status of the log servers, the CT policy, and the CAs supported by the logs. By comparing the results from a single monitor with the results in the reference sets, the auditor and the monitor can detect if any valid record is not correctly returned. The number of missing records and the number of affected domains can be used to evaluate the reliability of the services of CT monitors. The second black-box testing approach involves a set of CAs to regularly issue multiple types of certificates for a selected set of test domain names and submit them to the logs. The auditor or the monitor itself can issue queries about the test domain names to check the monitoring states of these certificates, e.g., whether they are timely monitored and whether the monitoring results are complete. This is similar to the black-box test method for checking the correctness of data query in database systems [51], [52].

MaaS for elastic resource allocation. With the rapidly increase of certificates in logs, a monitor has to deal with the scalability challenge and allocate its limited resources among heavy tasks. Incapable of addressing this challenge

leads to several issues as we observed in this study, such as erroneous processing, delayed incident recovery, etc., which might lead to delayed or failed detection of fraudulent certificates. Therefore, we propose to enable elastic resource allocation in third-party monitors so that they can always have sufficient resources to correctly process the great number of (pre)certificates retrieved from logs.

To implement elastic resource allocation for monitors, we can possible deploy the monitor functions in a cloud computing platform, e.g., to enable monitoring as a service (MaaS) – certificate search and monitoring services [53]. MaaS in the cloud brings several advantages: (*a*) resources are allocated on demand so that certificates from the logs will be processed in a timely and efficient manner; and (*b*) the cloud platform provides a continuous unified service by enabling redundancy for monitoring resources, and mitigate the impact of single point of failure on the monitor service.

VII. Related Work

Understanding the TLS/HTTPS ecosystem. Several large-scale studies on the ecosystem have been finished. Holz et al. [54] analyzed the quality of TLS server certificates in the Internet, while Durumeric et al. [55] investigated the trust relationships among CAs and websites, and uncovered the insecure certificate practices. Amann et al. [56] analyzed the benign changes of the trust relationships in the wild. After analyzing 47 million certificates, Perl et al. [57] found that only 66% of the 426 root CAs trusted in mainstream platforms are used to sign TLS certificates. Huang et al. [58] investigated the MitM attacks against Facebook exploiting fraudulent certificates, and found that most were caused by antivirus software and corporate-scale content filters.

There are reports on invalid certificates in practice. 65% of TLS server certificates in the large-scale scan were invalid and most were held by end-user devices [59]. Among the websites with invalid certificates, about one third of them configure these certificates accidentally, while two thirds do deliberately [60]. Kumar et al. [61] studied the certificates in Censys and discussed the reasons of certificates with errors.

Based on the passive measurement of over 300,000 users, Akhawe et al. identified the low-risk TLS warnings that consume most user attention and proposed recommendations to browser developers that help to maintain the user attention in high-risk warnings [62]. The large-scale study in 2017 shows that most HTTPS errors are caused by client-side or network issues, but not server misconfigurations [63].

The records in CT logs help to understand the TLS/HTTPS ecosystem. VanderSloot et al. [29] integrated the certificates in logs with the data from passive measurements, active scans, and certificate search engines, to present a view of the TLS/HTTPS ecosystem. Using the data in public logs, Aertsen et al. [64] analyzed the certificate services of Let's Encrypt adopted in different organizations, hosts and domains, while the violations of certificate issuance standards were investigated [65]. With TLS server certificates from public logs and passive measurements, Cui et al. [66] analyzed the forged certificates in the wild. Based on the domain names extracted from the certificates in CT logs, machine-learning solutions are proposed to detect the domain names of phishing websites [67], [68]. Different from these studies using the data in CT logs to investigate certificate services, our work utilizes these data to analyze the services of CT monitors.

CT deployment. The deployments of CT in the Internet are investigated. Stark et al. [28] finished a study on the adoption of CT across the web, including compliance, user experience, and potential risk. Gustafsson et al. [30] characterized 11 public logs with a focus on the recorded certificates and their usage in TLS/HTTPS. Nykvist et al. [31] studied the adoption of CT in Alexa Top-1M websites and the methods to deliver SCTs. Amann et al. [13] finished a large-scale study on the adoption of various TLS/HTTPS security enhancements, including CT, HPKP, HSTS, CAA, SCSV downgrade prevention and DANE. Scheitle et al. [32] analyzed the server-side deployment of CT, and discussed the domain name information leakage caused by the certificates in public logs. B. Li et al [69] explores the TLS/HTTPS configurations of third-party monitors, compared with common websites. Korzhitskii et al [70] survey the root CA accepted by CT logs, track the changes over time, and find some potential log mismanagement and misbehavior. These works studied the deployments of CT from the perspectives of logs and websites, while we focus on the implementation and deployment of reliable monitors.

CT enhancement. Following the basic CT framework, several designs were proposed to improve the security and/or performance of CT. Dowling et al. [33] defined the security properties of logging schemes, and formally proved that CT achieves these properties. Verifiable designs enable users to verify notifications for the certificates of registered domains from monitors [53]. Sun et al. [71] proposed to support domain name owners to customize the log policy to achieve a lightweight self-monitor. An efficient gossip protocol was proposed to detect several types of log inconsistencies [35]. Li et al. [72] explore the CT framework in practice, and discuss the potential security vulnerabilities of each CT component. Roberts et al. [73] identified several types of user and enterprise information leakages through CT logs, and analyzed the extent of these information leakages. The logs are audited without exposing user privacy by zero-knowledge proofs [74], and with the support of non-public subdomains by commitments. Phan [75] proposed to store the Merkle tree of certificates using ORAM on one server, and the position mappings on another server. So these two non-colluding log servers response a browser (as an auditor) with the audit path, without disclosing the user's browsing habits. Oxford et al. [76] verified the effectiveness of CT gossip protocols and discussed the benefits and drawbacks of deploying these protocols. Matsumoto et al. [77] studied the incentives to deploy the CT framework, and proposed to detect the deployment status of a domain against the downgrade attacks.

CT extensions and variations. The idea of CT is extended from certificate signing to other services. CIRT [78] records certificates in two Merkle hash trees: one of which is in chronological order with all certificates and the other is lexicographical with only the recent ones for each certificate subject,

to achieve the transparencies of both certificate signing and revocation. Singh et al. [79] improved CIRT by bilinear-map accumulators and binary trees, to achieve the transparencies with shorter proofs. Tomescu et al. [80] introduced an append-only authenticated dictionary to construct logs, to offer both append-only proofs and lookup proofs with polylogarithmic size. PKISN [81] records all certificates and revocations in public logs in chronological order. So it is able to revoke a CA certificate, while the end-entity certificates issued before the revocation do not become invalid. PoliCert [11] records subject certificate policies and certificates in public logs, providing the cryptographic proofs of presence and absence. CONIKS [82] builds transparent key directories based on Merkle prefix trees, allowing a users to audit its public keys while keep privacy.

Software transparency [83] requires a developer to submit the updated package including all source codes and meta data, to public logs. So a monitor rebuilds all packages on every update and checks if the resulting binary matches. Ticket transparency [84] records SSO tickets in public logs to audit the IdP of SSO services, while blind signatures and Bloom filters are integrated to protect user privacy. The general transparency overlay [85] is designed, which can be instantiated to provide transparency for other services (e.g., Bitcoin). Wang et al. propose to publish certificates and also revocation status messages on blockchain by the certificate holders (i.e., domain owners) [86]. Domain owners are then granted collaborative controls over their certificates, to balance the absolute authority of CAs. CertLedger [87] proposes to conduct certificate signing/revocation and trusted CA management on blockchain, to achieve certificate/revocation transparency.

VIII. Conclusions

This paper presents the first systematic study on CT monitors in the wild. We analyze the data of 88 public logs in the Internet, and the studies show the (pre)certificates in the logs are increasing at a significant daily growth rate. The amount of records in the 50 regular logs increases by 7,778,870 records or 43.99 GB per day on average, and 5,002,599 records (about 30 GB) per day are appended in the 9 Google-operated & Google-approved logs since June 2018. The rapidly-increasing large amounts of records in the logs prevent an ordinary domain owner from assuming the role of monitor to watch for suspicious certificates by itself.

We study the certificate search services of well-known third-party monitors. Various domain names are input to search certificates from these monitors, and the search results disclose the following defects in the services: (*a*) none of the third-party monitors guarantees to return the complete set of certificates in the public logs for a domain; and (*b*) even the union of the third-party monitors can probably be unable to return the complete set, for some domains at least. The defective certificate search services of third-party monitors can cause some (fraudulent) certificates recorded in public logs but invisible to the claimed domain owners, which harms the overall reliability of CT.

This work provides an in-depth understanding of the implementations and deployments of CT monitors, and demonstrates technical challenges of monitors in practice.

References

[1] B. Li, J. Lin, F. Li, Q. Wang, Q. Li, J. Jing, and C. Wang, "Certificate transparency in the Wild: Exploring the reliability of monitors," in *26th ACM Conference on Computer and Communications Security (CCS)*, 2019, pp. 2505–2520.

[2] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. T. Polk, "IETF RFC 5280 - Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile," 2008.

[3] Comodo Group Inc, "Comodo report of incident," 2011, https://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html.

[4] B. Morton, "More Google fraudulent certificates," 2014, https://www.entrust.com/google-fraudulent-certificates/.

[5] P. Eckersley, "A Syrian MITM attack against Facebook," 2011, https://www.eff.org/deeplinks/2011/05/syrian-man-middle-against-facebook.

[6] K. Wilson, "Distrusting new CNNIC certificates," 2015, https://blog.mozilla.org/security/2015/04/02/distrusting-new-cnnic-certificates/.

[7] David Warburton, "2020 phishing and fraud report," 2020, https://www.f5.com/labs/articles/threat-intelligence/2020-phishing-and-fraud-report.

[8] SSLMate, "Timeline of certificate authority failures," 2020, https://sslmate.com/resources/certificate_authority_failures.

[9] CA Wosign, "Incidents involving the CA WoSign," 2016, https://www.mail-archive.com/dev-security-policy@lists.mozilla.org/msg03665.html#.

[10] P. Hoffman and J. Schlyter, "IETF RFC 6698 - The DNS-based authentication of named entities (DANE) transport layer security (TLS) protocol: TLSA," 2012.

[11] P. Szalachowski, S. Matsumoto, and A. Perrig, "PoliCert: Secure and flexible TLS certificate management," in *21st ACM Conference on Computer and Communications Security (CCS)*, 2014, pp. 406–417.

[12] J. Kasten, E. Wustrow, and A. Halderman, "CAge: Taming certificate authorities by inferring restricted scopes," in *17th FC*, 2013.

[13] J. Amann, O. Gasser, Q. Scheitle, L. Brent, G. Carle, and R. Holz, "Mission accomplished? HTTPS security after DigiNotar," in *17th Internet Measurement Conference (IMC)*, 2017, pp. 325–340.

[14] Google Inc, "Certificate transparency," http://www.certificate-transparency.org/.

[15] B. Laurie, A. Langley, and E. Käsper, "IETF RFC 6962 - Certificate Transparency," 2013.

[16] Google Inc, "Certificate transparency in Chrome," 2018, https://groups.google.com/a/chromium.org/forum/#!msg/ct-policy/wHILiYf31DE/iMFmpMEkAQAJ.

[17] Apple Inc, "Apple's certificate transparency policy," 2019, https://support.apple.com/en-us/HT205280.

[18] Mozilla, "Mozilla CT policy," 2019, https://groups.google.com/a/chromium.org/forum/m/#!topic/ct-policy/Xx1bv8r33ZE.

[19] Google Inc, "Google: HTTPS encryption on the web," https://transparencyreport.google.com/https/certificates.

[20] Opsmate Inc, "SSLMate statement," 2019, https://sslmate.com/blog/post/certspotter_apiv1, https://sslmate.com/certspotter/howithelps.

[21] Facebook Inc, "Facebook monitor statement," 2019, https://www.facebook.com/notes/801629873918851/.

[22] S. Kent, "IETF Draft - Attack and threat model for certificate transparency," 2019, https://datatracker.ietf.org/doc/draft-ietf-trans-threat-analysis/.

[23] Comodo CA Limited, "crt.sh: Certificate search," https://crt.sh.

[24] Opsmate Inc, "SSLMate: Cert spotter," https://sslmate.com/certspotter/.

[25] University of Michigan, "Censys," https://censys.io/.

[26] Facebook Inc, "Facebook: Certificate transparency monitoring," https://developers.facebook.com/tools/ct/search/.

[27] Alexa, "Alexa top 1M websites," 2018, http://s3.amazonaws.com/alexa-static/top-1m.csv.zip.

[28] 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 *40th IEEE S&P*, 2019.

[29] B. VanderSloot, J. Amann, M. Bernhard, Z. Durumeric, M. Bailey, and J. A. Halderman, "Towards a complete view of the certificate ecosystem," in *16th IMC*, 2016, pp. 543–549.

[30] J. Gustafsson, G. Overier, M. F. Arlitt, and N. Carlsson, "A first look at the CT landscape: Certificate transparency logs in practice," in *18th PAM*, 2017, pp. 87–99.

[31] C. Nykvist, L. Sjöström, J. Gustafsson, and N. Carlsson, "Server-side adoption of certificate transparency," in *19th PAM*, 2018, pp. 186–199.

[32] Q. Scheitle, O. Gasser, T. Nolte, J. Amann, L. Brent, G. Carle, R. Holz, T. C. Schmidt, and M. Wählisch, "The rise of certificate transparency and its implications on the Internet ecosystem," in *18th Internet Measurement Conference (IMC)*, 2018, pp. 343–349.

[33] B. Dowling, F. Günther, U. Herath, and D. Stebila, "Secure logging schemes and certificate transparency," in *21st ESORICS*, 2016, pp. 140–158.

[34] L. Nordberg, D. K. Gillmor, and T. Ritter, "IETF Draft - Gossiping in CT," 2018, https://datatracker.ietf.org/doc/html/draft-ietf-trans-gossip-05.

[35] L. Chuat, P. Szalachowski, A. Perrig, B. Laurie, and E. Messeri, "Efficient gossip protocols for verifying the consistency of certificate logs," in *3rd CNS*, 2015, pp. 415–423.

[36] Google Inc, "Certificate transparency in Chrome," 2018, https://github.com/chromium/ct-policy/blob/master/ct_policy.md.

[37] ——, "Known logs," 2020, http://www.certificate-transparency.org/known-logs.

[38] G. Edgecombe, "Certificate transparency monitor," https://ct.grahamedgecombe.com/.

[39] Opsmate Inc, "Certificate transparency ecosystem," https://sslmate.com/labs/ct_ecosystem/ecosystem.html.

[40] ——, "Certificate transparency log growth," https://sslmate.com/labs/ct_growth/.

[41] Microsoft Corporation, "Microsoft trusted root certificate program: Participants," 2018, https://gallery.technet.microsoft.com/Trusted-Root-Program-d17011b8.

[42] Mozilla, "Mozilla included CA certificate list," 2018, https://wiki.mozilla.org/CA/Included_Certificates.

[43] Apple Inc, "Lists of available trusted root certificates in macOS," 2018, https://support.apple.com/en-us/HT202858.

[44] Google Inc, "The root CA list accepted by the logs," 2019, https://groups.google.com/a/chromium.org/forum/#!msg/ct-policy/iNX-EmfRLOA/ctBCTYGfAwAJ, https://github.com/chromium/ct-policy/issues/.

[45] Entrust Corporation, "Entrust Certificate transparency search," https://www.entrust.com/resources/certificate-solutions/tools/certificate-transparency-search.

[46] Let's Encrypt, "A free, automated, and open certificate authority (CA)," https://letsencrypt.org/.

[47] Wikipedia, "List of Internet top-level domains," 2018, https://en.wikipedia.org/wiki/List_of_Internet_top-level_domains.

[48] SSL.com, "DV, OV, and EV certificates," 2015, https://www.ssl.com/article/dv-ov-and-ev-certificates/.

[49] Google Inc, "Chromium STHSets," 2019, https://clients2.google.com/service/update2/crx.

[50] ——, "Google gossip," 2019, https://www.gstatic.com/ct/gossip.

[51] E. Rogstad, L. Briand, and R. Torkar, "Test case selection for black-box regression testing of database applications," *Information and Software Technology (IST)*, vol. 55, no. 10, pp. 1781–1795, 2013.

[52] P. Schroeder, P. Faherty, and B. Korel, "Generating expected results for automated black-box testing," in *17th ASE*, 2002, pp. 139–148.

[53] R. Dahlberg and T. Pulls, "Verifiable light-weight monitoring for certificate transparency logs," in *23th Nordsec*, 2018, pp. 171–183.

[54] R. Holz, L. Braun, N. Kammenhuber, and G. Carle, "The SSL landscape: A thorough analysis of the X.509 PKI using active and passive measurements," in *11th IMC*, 2011, pp. 427–444.

[55] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman, "Analysis of the HTTPS certificate ecosystem," in *13th IMC*, 2013, pp. 291–304.

[56] B. Amann, R. Sommer, M. Vallentin, and S. Hall, "No attack necessary: The surprising dynamics of SSL trust relationships," in *29th ACSAC*, 2013, pp. 179–188.

[57] H. Perl, S. Fahl, and M. Smith, "You won't be needing these any more: On removing unused certificates from trust stores." in *16th FC*, 2014, pp. 307–315.

[58] L. Huang, A. Rice, E. Ellingsen, and C. Jackson, "Analyzing forged SSL certificates in the wild," in *35th S&P*, 2014, pp. 83–97.

[59] T. Chung, Y. Liu, D. R. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson, "Measuring and applying invalid SSL certificates: The silent majority," in *16th IMC*, 2016, pp. 527–541.

[60] S. Fahl, Y. Acar, H. Perl, and M. Smith, "Why Eve and Mallory (also) love webmasters: A study on the root causes of SSL misconfigurations," in *9th AsiaCCS*, 2014, pp. 507–512.

[61] 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 *39th IEEE Symposium on Security and Privacy (S&P)*, 2018, pp. 785–798.

[62] D. Akhawe, B. Amann, M. Vallentin, and R. Sommer, "Here's my cert, so trust me, maybe? Understanding TLS errors on the web," in *22nd International World Wide Web Conference (WWW)*, 2013, pp. 59–70.

[63] 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 HTTPS certificate errors," in *14th ACM CCS*, 2017, pp. 1407–1420.

[64] M. Aertsen, M. Korczynski, G. Moura, S. Tajalizadehkhoob, and J. van den Berg, "No domain left behind: Is Let's Encrypt democratizing encryption?" in *2nd Applied Networking Research Workshop (ANRW)*, 2017, pp. 48–54.

[65] O. Gasser, B. Hof, M. Helm, M. Korczynski, R. Holz, and G. Carle, "In log we trust: Revealing poor security practices with certificate transparency logs and Internet measurements," in *19th PAM*, 2018, pp. 173–185.

[66] M. Cui, Z. Cao, and G. Xiong, "How is the forged certificates in the wild: Practice on large-scale SSL usage measurement and analysis," in *18th ICCS*, 2018, pp. 654–667.

[67] E. Fasllija, H. F. Enişer, and B. Prünster, "Phish-Hook: Detecting phishing certificates using certificate transparency logs," in *15th SecureComm*, 2019, pp. 320–334.

[68] Y. Sakurai, T. Watanabe, T. Okuda, M. Akiyama, and T. Mori, "Discovering HTTPSified Phishing Websites Using the TLS Certificates Footprints," in *5th IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, 2020, pp. 522–531.

[69] B. Li, D. Chu, J. Lin, Q. Cai, C. Wang, and L. Meng, "The weakest link of certificate transparency: Exploring the TLS/HTTPS configurations of third-party monitors," in *18th TrustCom*, 2019.

[70] N. Korzhitskii and N. Carlsson, "Characterizing the root landscape of certificate transparency logs," in *19th IFIP Networking*, 2020.

[71] A. Sun, B. Li, H. Wan, and Q. Wang, "Polict: Flexible policy in certificate transparency enabling lightweight self-monitor," in *2nd International Workshop on Secure Cryptographic Implementation (SCI), in conjunction with 19th ACNS*, 2021, pp. 358–377.

[72] B. Li, F. Li, Z. Ma, and Q. Wu, "Exploring the security of certificate transparency in the wild," in *1st International Workshop on Secure Cryptographic Implementation (SCI), in conjunction with 18th ACNS*, 2020, pp. 453–470.

[73] R. Roberts and D. Levin, "When certificate transparency is too transparent: Analyzing information leakage in HTTPS domain names," in *18th ACM WPES*, 2019, pp. 87–92.

[74] S. Eskandarian, E. Messeri, J. Bonneau, and D. Boneh, "Certificate transparency with privacy," in *17th PETS*, 2017, pp. 329–344.

[75] V.-A. Phan, "Private queries on public certificate transparency data," University of California at Berkeley, Tech. Rep., 2019, https://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-27.pdf.

[76] M. Oxford, D. Parker, and M. Ryan, "Quantitative verification of certificate transparency gossip protocols," in *8th IEEE Conference on Communications and Network Security (CNS)*, 2020, pp. 1–9.

[77] S. Matsumoto, P. Szalachowski, and A. Perrig, "Deployment challenges in log-based PKI enhancements," in *8th European Workshop on System Security (EuroSec)*, 2015, pp. 1–7.

[78] M. D. Ryan, "Enhanced certificate transparency and end-to-end encrypted mail," in *21st NDSS*, 2014.

[79] A. Singh, B. Sengupta, and S. Ruj, "Certificate transparency with enhancements and short proofs," in *22nd ACISP*, 2017, pp. 381–389.

[80] A. Tomescu, V. Bhupatiraju, D. Papadopoulos, C. Papamanthou, N. Triandopoulos, and S. Devadas, "Transparency logs via append-only authenticated dictionaries," in *26th ACM Conference on Computer and Communications Security (CCS)*, 2019, pp. 1299–1316.

[81] P. Szalachowski, L. Chuat, and A. Perrig, "PKI safety net (PKISN): Addressing the too-big-to-be-revoked problem of the TLS ecosystem," in *1st IEEE EuroS&P*, 2016, pp. 407–422.

[82] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman, "CONIKS: Bringing key transparency to end users," in *24th USENIX Security Symposium*, 2015, pp. 383–398.

[83] B. Hof and G. Carle, "Software distribution transparency and auditability," arXiv ePrint, 2017, https://arxiv.org/abs/1711.07278.

[84] D. Chu, J. Lin, F. Li, X. Zhang, Q. Wang, and G. Liu, "Ticket transparency: Accountable single sign-on with privacy-preserving public logs," in *15th SecureComm*, 2019, pp. 511–531.

[85] M. Chase and S. Meiklejohn, "Transparency overlays and applications," in *13th ACM CCS*, 2016, pp. 168–179.

[86] Z. Wang, J. Lin, Q. Cai, Q. Wang, D. Zha, and J. Jing, "Blockchain-based certificate transparency and revocation transparency," *IEEE TDSC*, 2020.

[87] M. Y. Kubilay, M. S. Kiraz, and H. A. Mantar, "CertLedger: A new PKI model with certificate transparency based on blockchain," *Computers & Security*, vol. 85, pp. 333–352, 2019.

Bingyu Li received the B.S. degree from Jilin University in 2013, and the Ph.D. degree from the University of Chinese Academy of Sciences in 2020. He is a postdoctoral fellow at School of Cyber Science and Technology, Beihang University. His research interest includes public key infrastructure, identity management, applied cryptography and system security.

Jingqiang Lin received his MS and PhD degrees from Graduate University of Chinese Academy of Sciences in 2004 and 2009, respectively. He is a full professor at School of Cyber Security, University of Science and Technology of China. His research interest includes applied cryptography and system security.

Fengjun Li received the B.E. degree from the University of Science and Technology of China, the M.Phil. degree from the Chinese University of Hong Kong, and the Ph.D. degree from the Pennsylvania State University. She is currently an associate professor with the department of Electrical Engineering and Computer Science at the University of Kansas. Her research interests include a broad area of computer and network security, with a recent focus on adversarial machine learning and IoT security. She has published more than 50 refereed technical papers, served on several technical program committees, and reviewed papers for numerous journals. Her research has been supported by NSF, DoD, NSA, the Ripple University Initiative, etc.

Qiongxiao Wang received the B.E. degree from Jilin University and the Ph.D. degree from Graduate University of Chinese Academy of Sciences. She is currently an Associate Professor at Institute of Information Engineering, Chinese Academy of Sciences. Her research interest includes applied cryptography and identity management.

Wei Wang received his B.E. degree from Tsinghua University in 2010, and the Ph.D. degree from University of Chinese Academy of Sciences in 2016. Now he is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences as an assistant professor. His research interests include system security and authentication techniques.

Qi Li received the PhD degree from Tsinghua University. Now he is an associate professor of Institute for Network Sciences and Cyberspace, Tsinghua University. He has ever worked with ETH Zurich and the University of Texas at San Antonio. His research interests include network and system security, particularly in Internet and cloud security, mobile security, and big data security. He is currently an editorial board member of IEEE TDSC and ACM DTRAP.

Guangshen Cheng is a Master student at University of Science and Technology of China, major in Cyber Space Security. He received his bachelor degree from Central University of Finance and Economics in 2016. His research interest includes applied cryptography and system security.

Jiwu Jing received the bachelor degree from the Department of Electronics Engineering, Tsinghua University in 1987, and the M.Sc. and Ph.D. degrees from the Graduate School, University of Science and Technology of China in 1990 and 2003, respectively. He is a full professor at School of Computer Science and Technology, University of Chinese Academy of Sciences. He was the convenor of the Expert Group on Information Security of the National High Technology Research and Development Program of China (863 Program). His main research interest includes public key infrastructure, identity management, fault tolerance, mobile security, information system, and network protection. He received the National Science and Technology Progress Award, the Science and Technology Progress Award of the Chinese Ministry of Electronics Industry, and the Science and Technology Progress Award of CAS.

Congli Wang received the M.S. degree from the University of Chinese Academy of Sciences in 2019. She is currently an engineer with Research Institute of China Telecom Co., Ltd. Her research interests include public key infrastructure, applied cryptography and network security.

APPENDIX A
DOMAIN NAMES WITHOUT CERTIFICATES

We investigated the 18 domains on the certificates configured by the websites and why they were not returned by monitors. In summary, 10 websites support HTTPS but configure certificates in a problematic means. (*a*) Two websites (nextlnk1.com, nextlnk2.com) use certificates whose CNs do not match the domain names. (*b*) Two websites (blog.jp, pop.bid) use expired certificates (expired before our experiments on October 27, 2018), not covered in our experiments. (*c*) A self-signed certificate is configured in freejobalert.com, which has not been submitted to any logs. And (*d*) five websites (lun.com, dytt8.net, igmatik.com, vseigru.net, seasonvar.ru) configure certificates which are issued after our experiments on October 27, 2018. The remaining 8 websites (haber7.com, bancodevenezuela.com, hdrezka.ag, livedoor.biz, mahresult.nic.in, doorblog.jp, iz682noju02ye5.com, hitpromoit.com) do not support HTTPS.

APPENDIX B
MORE DETAILED EXPERIMENT RESULTS

Table XIII lists the mainstream CAs that are not supported by the public logs, and Table XIV gives the detailed certificate search results of 6,000 selected Alexa Top-1M websites. Table XV lists the number of domains with incomplete results of 6,000 selected Alexa Top-1M websites. Table XVI lists the missing result in each type of certificates of 6,000 selected Alexa Top-1M websites.

TABLE XIII
THE MAINSTREAM CAS UNACCEPTED BY PUBLIC LOGS.

No.†	Subject DN	Validity Period	Platform
1	C=CN, O=China Financial Certification Authority, CN=CFCA Identity CA	2015.06.30–2040.06.30	Microsoft
2	C=US, O=Microsoft Corporation, CN=Microsoft Time Stamp Root Certificate Authority 2014	2014.10.22–2039.10.22	Microsoft
3	C=US, O=Symantec Corporation, CN=Symantec Enterprise Mobile Root for Microsoft	2012.03.15–2032.03.14	Microsoft
4	L=Internet, O=VeriSign, Inc., OU=VeriSign Commercial Software Publishers CA	1996.04.09–2004.01.07	Microsoft
5	O=VeriSign Trust Network, OU=VeriSign, Inc., OU=VeriSign Time Stamping Service Root	1997.05.12–2004.01.07	Microsoft
6	C=AT, O=e-commerce monitoring GmbH, CN=GLOBALTRUST 2015	2015.06.11–2040.06.10	Microsoft
7	C=Microsoft ECC Product Root Certificate Authority 2018, O=Microsoft Corporation, CN=US	2018.02.27–2043.02.27	Microsoft
8	C=ZA, O=Thawte, OU=Thawte Certification, CN=Thawte Timestamping CA	1997.01.01–2020.12.31	Microsoft
9	C=US, O=The USERTRUST Network, CN=UTN-USERFirst-Object	1999.07.09–2019.07.09	All

†: These 9 mainstream CAs are not accepted by any Google-operated log, and the first 5 are not accepted by any regular log.

TABLE XIV
THE NUMBER OF DOMAINS WITH INCOMPLETE RESULTS, IN EACH GROUP OF WEBSITES WITH DIFFERENT NUMBERS OF CERTIFICATES.

	Φ	$= 0$	$= 1$	(1, 10]	(10, 100]	(100, $+\infty$)	[0, $+\infty$)
Top-1K	D_Φ	18	50	235	471	226	1000
	crt.sh	-	0	8	28	68	104
	SSLMate	-	0	7	53	104	164
	Censys	-	0	3	33	84	120
	Google Monitor	-	0	59	269	218	546
	Facebook Monitor	-	0	14	102	173	289
Top-(1K, 5K]	D_Φ	32	82	309	452	125	1000
	crt.sh	-	0	12	35	31	78
	SSLMate	-	0	11	39	50	100
	Censys	-	0	3	16	33	52
	Google Monitor	-	0	57	245	119	421
	Facebook Monitor	-	0	21	190	96	307
Top-(5K, 20K]	D_Φ	62	134	310	423	71	1000
	crt.sh	-	0	4	17	25	46
	SSLMate	-	0	5	26	30	61
	Censys	-	0	1	9	17	27
	Google Monitor	-	1	55	175	63	294
	Facebook Monitor	-	4	69	257	63	393
Top-(20K, 100K]	D_Φ	122	179	377	297	25	1000
	crt.sh	-	0	10	15	4	29
	SSLMate	-	0	9	18	6	33
	Censys	-	0	5	5	4	14
	Google Monitor	-	1	62	113	22	198
	Facebook Monitor	-	8	68	164	19	259
Top-(100K, 500K]	D_Φ	166	218	390	222	4	1000
	crt.sh	-	0	6	10	0	16
	SSLMate	-	0	6	11	2	19
	Censys	-	0	4	2	1	7
	Google Monitor	-	2	51	61	3	117
	Facebook Monitor	-	12	85	125	4	226
Top-(500K, 1M]	D_Φ	228	233	392	143	4	1000
	crt.sh	-	0	3	8	0	11
	SSLMate	-	0	4	10	1	15
	Censys	-	0	3	3	1	7
	Google Monitor	-	1	33	36	3	73
	Facebook Monitor	-	9	70	77	4	160

Φ: the range of the number of certificates for a domain.
D_Φ: the number of domains in a group of websites. For every domain in this group, the sum of unique certificates returned is within the range Φ.

TABLE XV
THE NUMBER OF DOMAINS WITH DIFFERENT NUMBERS OF MISSING CERTIFICATES, FOR EACH SEGMENT OF RANDOMLY-SELECTED ALEXA TOP-1M WEBSITES.

	θ	$= 0$	$= 1$	(1, 10]	(10, 100]	(100, $+\infty$)	(0, $+\infty$)
Top-1K	crt.sh	878	42	36	19	7	104
	SSLMate	818	53	65	31	15	164
	Censys	862	47	49	19	5	120
	Google Monitor	436	80	211	179	76	546
	Facebook Monitor	693	86	133	47	23	289
Top-(1K, 5K]	crt.sh	890	24	34	18	2	78
	SSLMate	868	33	43	20	4	100
	Censys	916	19	26	7	0	52
	Google Monitor	547	93	173	125	30	421
	Facebook Monitor	661	0	222	66	19	307
Top-(5K, 20K]	crt.sh	892	23	20	3	0	46
	SSLMate	877	26	23	10	2	61
	Censys	911	10	9	6	2	27
	Google Monitor	644	81	136	64	13	294
	Facebook Monitor	545	155	196	40	2	393
Top-(20K, 100K]	crt.sh	849	12	16	1	0	29
	SSLMate	845	11	19	3	0	33
	Censys	864	5	6	3	0	14
	Google Monitor	680	66	101	26	5	198
	Facebook Monitor	619	122	125	11	1	259
Top-(100K, 500K]	crt.sh	818	8	7	1	0	16
	SSLMate	815	10	8	1	0	19
	Censys	827	3	4	0	0	7
	Google Monitor	717	58	47	11	1	117
	Facebook Monitor	608	124	96	6	0	226
Top-(500K, 1M]	crt.sh	761	5	5	1	0	11
	SSLMate	757	7	7	1	0	15
	Censys	765	5	2	0	0	7
	Google Monitor	699	36	32	4	1	73
	Facebook Monitor	612	96	59	5	0	160

θ: the range of the number of missing certificates for a domain.

TABLE XVI
THE MISSING RESULTS OF DIFFERENT CERTIFICATE TYPES, FOR EACH SEGMENT OF ALEXA TOP-1M WEBSITES.

		$\#_{dn}$			$\#_{unicert}$		
		EV	OV	DV	EV	OV	DV
Top-1K	Ref	251	962	934	20,078	284,272	77,701
	crt.sh	5 (2.0%)	27 (2.8%)	33 (3.5%)	18 (0.1%)	52,539 (18.5%)	268 (0.3%)
	SSLMate	14 (5.6%)	81 (8.4%)	57 (6.1%)	886 (4.4%)	133,618 (47.0%)	43,896 (56.5%)
	Censys	13 (5.2%)	85 (8.8%)	59 (6.3%)	145 (0.7%)	30,866 (10.9%)	18,385 (23.7%)
	Google Monitor	91 (36.3%)	443 (46.0%)	269 (28.8%)	9,406 (46.8%)	157,920 (55.6%)	38,718 (49.8%)
	Facebook Monitor	38 (15.1%)	173 (18.0%)	133 (14.2%)	994 (5.0%)	101,799 (35.8%)	24,587 (31.6%)
Top-(1K, 5K]	Ref	216	845	953	4,506	66,719	42,836
	crt.sh	0 (0.0%)	8 (0.9%)	19 (2.0%)	0 (0.0%)	67 (0.1%)	56 (0.1%)
	SSLMate	7 (3.2%)	33 (3.9%)	22 (2.3%)	27 (0.6%)	1,302 (2.0%)	6,376 (14.9%)
	Censys	10 (4.6%)	45 (5.3%)	27 (2.8%)	87 (1.9%)	429 (0.6%)	87 (0.2%)
	Google Monitor	69 (31.9%)	297 (35.1%)	200 (21.0%)	793 (17.6%)	9,015 (13.5%)	11,282 (26.3%)
	Facebook Monitor	32 (14.8%)	191 (22.6%)	332 (34.8%)	119 (2.6%)	5,355 (8.0%)	5,558 (13.0%)
Top-(5K, 20K]	Ref	158	600	879	1,829	31,214	15,826
	crt.sh	0 (0.0%)	4 (0.7%)	7 (0.8%)	0 (0.0%)	5 (0.0%)	20 (0.2%)
	SSLMate	6 (3.8%)	27 (4.5%)	23 (2.6%)	24 (1.3%)	6,866 (22.0%)	1,383 (8.7%)
	Censys	6 (3.8%)	31 (5.2%)	14 (1.6%)	27 (1.5%)	769 (2.5%)	31 (0.2%)
	Google Monitor	47 (29.7%)	206 (34.3%)	120 (13.7%)	486 (26.6%)	6,294 (20.2%)	1,001 (6.3%)
	Facebook Monitor	14 (8.9%)	114 (19.0%)	324 (36.9%)	53 (2.9%)	890 (2.9%)	1,342 (8.5%)
Top-(20K, 100K]	Ref	87	384	811	435	9,896	9,227
	crt.sh	0 (0.0%)	8 (2.1%)	7 (0.9%)	0 (0.0%)	90 (0.9%)	16 (0.2%)
	SSLMate	1 (1.1%)	9 (2.3%)	5 (0.6%)	1 (0.2%)	94 (0.9%)	11 (0.1%)
	Censys	1 (1.1%)	21 (5.5%)	13 (1.6%)	1 (0.2%)	173 (1.7%)	27 (0.3%)
	Google Monitor	25 (28.7%)	121 (31.5%)	88 (10.9%)	97 (22.3%)	1,649 (16.7%)	277 (3.0%)
	Facebook Monitor	7 (8.0%)	63 (16.4%)	221 (27.3%)	10 (2.3%)	421 (4.3%)	754 (8.2%)
Top- (100K, 500K]	Ref	42	266	791	188	3,925	5,556
	crt.sh	0 (0.0%)	13 (4.9%)	7 (0.9%)	0 (0.0%)	249 (6.3%)	10 (0.2%)
	SSLMate	2 (4.8%)	5 (1.9%)	14 (1.8%)	2 (1.1%)	8 (0.2%)	118 (2.1%)
	Censys	3 (7.1%)	16 (6.0%)	14 (1.8%)	3 (1.6%)	273 (7.0%)	20 (0.4%)
	Google Monitor	7 (16.7%)	82 (30.8%)	44 (5.6%)	11 (5.9%)	920 (23.4%)	118 (2.1%)
	Facebook Monitor	4 (9.5%)	45 (16.9%)	206 (26.0%)	4 (2.1%)	368 (9.4%)	402 (7.2%)
Top-(500K, 1M]	Ref	20	164	688	27	2,439	3,996
	crt.sh	0 (0.0%)	38 (23.2%)	33 (4.8%)	0 (0.0%)	822 (33.7%)	54 (1.4%)
	SSLMate	0 (0.0%)	1 (0.6%)	7 (1.0%)	0 (0.0%)	1 (0.0%)	21 (0.5%)
	Censys	0 (0.0%)	41 (25.0%)	37 (5.4%)	0 (0.0%)	823 (33.7%)	71 (1.8%)
	Google Monitor	3 (15.0%)	72 (43.9%)	65 (9.4%)	4 (14.8%)	922 (37.8%)	343 (8.6%)
	Facebook Monitor	0 (0.0%)	56 (34.1%)	182 (26.5%)	0 (0.0%)	843 (34.6%)	383 (9.6%)

Ref: the number of domains with certificates of a certain type, or the number of certificates of a certain type, in each segment of websites;
$\#_{dn}$: the number of domains for which at least one monitor returns *incomplete* results, compared with the reference sets;
$\#_{unicert}$: the number of *unique* missing certificates which are *not* returned from at least one monitor.