Stale TLS Certificates

Investigating Precarious Third-Party Access to Valid TLS Keys

Zane Ma
Oregon State University
Georgia Institute of Technology

Aaron Faulkenberry
Georgia Institute of Technology

Thomas Papastergiou
Georgia Institute of Technology

Zakir Durumeric
Stanford University

Michael D. Bailey
Georgia Institute of Technology

Angelos D. Keromytis
Georgia Institute of Technology

Fabian Monrose
Georgia Institute of Technology

Manos Antonakakis
Georgia Institute of Technology

ABSTRACT

Certificate authorities enable TLS server authentication by generating certificates that attest to the mapping between a domain name and a cryptographic keypair, for up to 398 days. This static, name-to-key caching mechanism belies a complex reality: a tangle of dynamic infrastructure involving domains, servers, cryptographic keys, etc. When any of these operations changes, the authentication information in a certificate becomes stale and no longer accurately reflects reality. In this work, we examine the broader phenomenon of *certificate invalidation events* and discover three classes of security-relevant events that enable a third-party to impersonate a domain outside of their control. Longitudinal measurement of these precarious scenarios reveals that they affect over 15K new domains per day, on average. Unfortunately, modern certificate revocation provides little recourse, so we examine the potential impact of reducing certificate lifetimes (cache duration): shortening the current 398-day limit to 90 days yields a 75% decrease in precarious access to valid TLS keys.

CCS CONCEPTS

• **Networks** → *Public Internet*; *Network measurement*; • **Security and privacy** → **Security protocols**; **Web protocol security**; **Authentication**; **Key management**.

KEYWORDS

Web PKI, TLS Certificates, Server Authentication, Stale Records

ACM Reference Format:
Zane Ma, Aaron Faulkenberry, Thomas Papastergiou, Zakir Durumeric, Michael D. Bailey, Angelos D. Keromytis, Fabian Monrose, and Manos Antonakakis. 2023. Stale TLS Certificates: Investigating Precarious Third-Party Access to Valid TLS Keys. In *Proceedings of the 2023 ACM Internet Measurement Conference (IMC '23), October 24–26, 2023, Montreal, QC, Canada.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3618257.3624802

IMC '23, October 24–26, 2023, Montreal, QC, Canada

ACM ISBN 979-8-4007-0382-9/23/10.
https://doi.org/10.1145/3618257.3624802

1 INTRODUCTION

The modern web (e.g., HTTPS, email) relies on Transport Layer Security (TLS) for server authentication, with more than 75% of global website access occurring over HTTPS [36]. To provide server authentication, certificate authorities (CAs) verify the association between a domain name and a cryptographic public key, and then generate a signed attestation known as a TLS certificate. TLS clients (e.g., web browsers) ultimately rely on these certificates to authenticate servers during TLS, making sure that the server's semantic identity (i.e., domain name) matches the cryptographic identity (i.e., public key) found in the certificate.

Certificates are an authentication caching mechanism that temporarily associates a domain name to a cryptographic public key. They are only valid for a limited *validity period* (also called the *certificate lifetime*), which is specified within the certificate itself. Once a certificate has expired, a CA must re-verify the domain's public key and issue a new TLS certificate. Like any caching mechanism, a natural tension exists between performance gains and the consequences of stale records. Certificate lifetimes (cache durations) should ideally balance 1) the security concerns of stale records with 2) the operational burden of certificate issuance on CAs and the broader web public key infrastructure (PKI) ecosystem.

To understand the security consequences of valid-but-stale certificates, we first outline the network dynamics that lead to staleness. In particular, we enumerate the *cache invalidation events* that cause the authentication and authorization information in a TLS certificate to diverge from real-world operations. While prior security discussion has focused primarily on one-off invalidation events (e.g., catastrophic CA compromise [8]), we introduce continuous invalidation events that give rise to a much larger, replenishing population of stale certificates. These continuous invalidation events are the result of an increasingly complex chain of dependencies—domain registrants, content distribution networks (CDNs), virtual web hosting, certificate management software—that underlie the symbolic name-to-key link contained in TLS certificates. Stale certificates occur when these dependencies change during a certificate's lifetime of up to 398 days[1], and in certain cases can enable attackers to impersonate domains they do not control.

We measure the occurrence of stale certificates in the wild by combining longitudinal data for certificates, domain registrations,

[1]Prior to 2020, domain-validated certificates could last up to 825 days [17].

and DNS. We focus on three critical scenarios where stale certificates allow a third-party to impersonate a domain they do not control: key compromise, domain registrant change, and managed TLS departure. Even when taking a lower-bound targeted measurement approach, we detect over 9 million instances of abusable third-party stale certificates from 2016–2023 across 4.5 million effective second-level domains: over 300 domains per day due to key compromise, 1.2K domains per day due to domain registrant change, and 7.7K per day from managed TLS migrations. Additionally, although we lack the data to infer active exploitation of stale certificates, we find evidence of third-party stale certificates under the control of known malicious actors.

Two mechanisms are meant to account for the occurrence of certificate invalidation events and the stale certificates they produce: certificate revocation as a first line of defense and expiration as the final backstop. Unfortunately, certificate revocation remains largely ineffective, so efforts to address stale certificates and limit exposure must focus on modifying certificate lifetimes. We perform a survival analysis and simulate the reduction of all certificate lifetimes to 90 days (the default for many automated CAs such as Let's Encrypt [7]), finding an optimistic 42–70% decrease in stale certificates and 29–67% reduction in overall staleness-days. Similar to expiration policies of non-digital identities (e.g., passports with out-of-date photos, physical attributes, etc.), discussion of certificate validity policies should be informed by these measurements of subscriber information dynamics.

Beyond quantifying the stale certificate phenomenon, this investigation highlights the inharmonious design of the existing web PKI, which bridges the increasingly complex gap between domain names and cryptographic TLS keys. By explicitly identifying the differences between what a certificate ideally attests to and the dynamic reality of today's web, we can better evaluate potential solutions and make progress towards a more precise and secure web PKI.

In summary, this paper makes the following contributions:

- Taxonomy of certificate invalidation events that lead to stale certificates.
- Large-scale measurement of third-party stale certificates from April 2013 – May 2023.
- Empirical estimation of the performance / security tradeoff for informing certificate lifetime policy.

2 BACKGROUND

Stale certificates arise from the contrasting operational dynamics of TLS certificates with the underlying authentication information (e.g., DNS names, CA infrastructure) that they attest to. This section details the relevant aspects of domain name management as well as certificate issuance, management, and revocation that enable stale certificates.

2.1 Domain Management

The Internet Corporation for Assigned Names and Numbers (ICANN) creates top-level domains (TLDs) and delegates their operation to *registries* such as Verisign, which operates the `.com` and `.net` TLDs. Domain *registrars* are intermediaries that handle billing, account management, and customer support for *registrants* who purchase

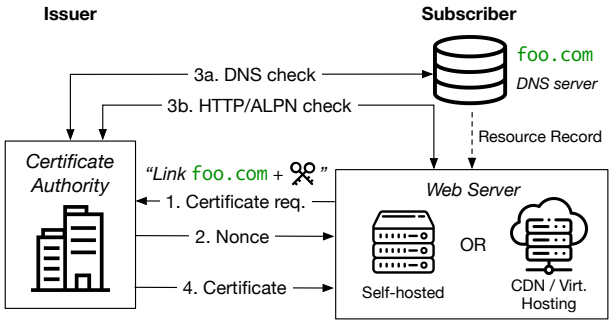


Figure 1: TLS Certificate Issuance—Certificates link a subscriber identity (e.g., DNS name) to a keypair.

and own domains. Most TLD registries allow public registration of second-level domains, however, some only allow registration of higher-level domains such as `*.co.uk`. In these cases, the effective TLD (eTLD) refers to the parent label (e.g., `co.uk`) for publicly registerable domains. The effective 2LD (e2LD) refers to the child label and eTLD (e.g., `foo.co.uk`).

Once a domain has been registered, it can change registrants under three scenarios: 1) the registrant transfers the domain to a different registrant at the same or different registrar; 2) the registrar transfers the domain to a new owner after the domain has expired, but before it is released to the registry; 3) the domain is re-registered by the public, including drop-catch services, after expiration and registry release. We refer readers to [50, 53] for a more detailed explanation of domain registration lifecycles. Detecting registrant changes is difficult for cases 1 & 2, since it relies on inconsistent, user-provided WHOIS data that has been largely anonymized in recent years [58]. Public re-registration is easier to detect because the registry-controlled "Creation Date" only changes upon domain name deletion and subsequent re-registration.

2.2 Certificate Issuance

The web public key infrastructure (PKI) provides a scalable solution to public TLS authentication by delegating identity verification to certification authorities (CAs). After performing identity verification for a *subscriber*, which is any entity (e.g., a web server) that requests a CA's services, CAs generate signed digital certificates that link a subscriber's *semantic identity* (e.g., DNS name) to their *cryptographic identity* (i.e., public key). During TLS authentication, this attestation is accepted if the CA is trusted by the authenticator. We refer the reader to [22] for a broader overview of TLS.

This paper examines the identity properties attested and cached by web PKI certificates and how they change over time to yield stale certificates. To understand the assurances that a web PKI certificate provides, we describe the identity verification processes that CAs perform before issuing a certificate. We focus on Domain Validated (DV) certificates, which constitute >85% of web certificates [23] and provide a baseline set of verifications for other certificate types.

The goal of DV identity verification is for a CA to confirm a subscriber's "ownership or control of the domain"[18]. Two general forms of verification exist: 1) transmission (e.g., phone, email, fax, etc.) of a random nonce to a domain contact specified in WHOIS

or DNS SOA/TXT/CAA records, followed by receipt confirmation, or 2) transmission of a random nonce to the subscriber (typically via HTTP connection that the subscriber initiates for requesting a certificate) and then CA verification that the nonce is present in a custom DNS record, TLS Application-Layer Protocol Negotiation (ALPN) response, or HTTP web page under the domain name (Figure 1). The latter methods have been automated as ACME [11], which has gained wide adoption by CAs such as Let's Encrypt [7]. ACME has enabled automated CAs such as Let's Encrypt to achieve high CA issuance throughput and limit issued certificates to 90-day lifetimes.

2.3 Certificate Management

Certificate management has historically been a difficult task for web server administrators [30, 46], but the development of ACME and its ecosystem of automated tools have made certificate management more usable [69]. This automation and the growing trend towards content delivery networks (CDNs) and shared web hosting have commodified the deployment and management of HTTPS certificates, and domain registrants now have a multitude of methods for acquiring and managing a DV TLS server authentication certificate. We highlight the most common options:

(1) Use manual methods or automated software to receive a certificate that the domain registrant hosts on their own public-facing web server.
(2) Receive a certificate for their domain (similar to above), and then upload their private key and certificate to a CDN that terminates external TLS connections.
(3) Delegate external web traffic to a CDN via canonical name (CNAME) or nameserver (NS) record (effectively setting the CDN as the authoritative nameserver) and have the CDN obtain a TLS certificate and manage all HTTPS connections.
(4) Directly request a TLS certificate from the domain registrar that also provides hosting services or can access external hosting services to manage SSL for the domain registrant [33].
(5) Use a third-party hosting platform (e.g., bluehost WordPress, cPanel) that automatically configures and manages SSL for the domain/website owner [13, 25].

Methods 2–5, which we define as managed TLS, introduce a third-party that has access to the private key for a domain's certificate. The involvement of a third-party increases operational complexity and creates new, precarious stale certificate scenarios.

2.4 Certificate Revocation

Revocation has been a component of the web PKI since its inception. RFC 5280 [14], which standardized today's TLS certificates, also specified how to revoke certificates via certificate revocation lists (CRLs). In order for TLS clients to respond appropriately to potential CRL revocation, a set of *revocation reasons* was also standardized, ranging from "unspecified" to "key compromise." Unfortunately, despite years of research [49, 68] and several attempted solutions—Certificate Revocation Lists (CRLs) [14], Online Certificate Status Protocol (OCSP) [65], and OCSP Must Staple [38]—effective revocation on the modern web remains elusive. Many major browsers, namely Google Chrome [27] and Microsoft Edge [52], do not check subscriber certificate revocation due to privacy and performance concerns [57]. Other non-browser user agents (e.g., `curl`, TLS libraries for API calls) also eschew default revocation checking [32]. Even when revocation checking is supported (e.g., Mozilla Firefox, Apple Safari), TLS clients often operate under a soft-fail policy [48], allowing a TLS interceptor to circumvent revocation by dropping revocation-related traffic[2]. Because certificate revocation is ineffectual under this threat model, its main value in this work is to provide a source of *reported* certificate changes, also known as invalidation events.

3 CERTIFICATE INVALIDATION EVENTS

Stale certificates are the result of *certificate invalidation events*, which are real-world changes that nullify the information contained within a certificate. For instance, if a domain owner changes their website's TLS keys, then the previous keys should no longer be authorized to represent their website. However, because the prior certificate may still be unexpired, the outdated keys remain technically functional. While this example may seem innocuous, stale certificates resulting from key change can extend the window of potential attacks in instances such as key compromise, where a third-party can impersonate domains that they do not control.

We propose a taxonomy of certificate invalidation events to systematically understand how they can occur and what issues they can cause. Although RFC 5280 [14] standardized a set of explicit revocation (i.e., invalidation) reasons, they are a poor basis for a taxonomy of certificate invalidation events. First, they are outdated, having been defined in 2008 and aligning poorly with modern use; fittingly, Mozilla only permits the usage of six out of the ten original reasons [61]. Second, the definitions are imprecise and leave room for ambiguity; as an example, only one reason can be specified and the reasons "superseded" and "cessation of operation" are not mutually exclusive. Lastly, one key purpose of providing revocation reasons is to inform potential security responses by TLS clients, but the existing reason codes do not align well with differing levels of security threats. For instance, "cessation of operation" includes scenarios where a website is no longer in operation, which is a non-urgent invalidation, but it would also include cases where a domain squatter controls a stale certificate for a transferred domain, which raises security concerns due to potential for abuse.

To address these limitations, our taxonomy of certificate invalidation events takes a principled approach based on the types of information that a TLS certificate attests to. Stale certificates only occur when the content of a TLS certificate no longer reflects real-world operations. As shown in Table 1, our taxonomy first categorizes certificate information based on one of four higher-level roles: subscriber authentication, key authorization, issuer information, and certificate data. This abstraction then enables subsequent classification of certificate invalidation events (Table 2), including those that are not immediately obvious based on revocation data or certificate field inspection. These "hidden" invalidation events, discussed in Sections 5.2 & 5.3, arise from changes to implicit, underlying network operations.

[2] One exception: Firefox hard-fails with OCSP Must-Staple.

Category	Description	Related fields
Subscriber authentication	Subscriber identifiers: domain + crypto. keys	Subject Name, SAN, Subj. Public Key, Subj. Key ID
Key authorization	Permissions + constraints on key utilization	Basic Constraints, Key Usage, Extended Key Usage
Issuer information	Details of CA that issued certificate	Issuer Name, Auth. Key ID, Signature, CRL Distribution Points, Auth. Info. Access, Certificate Policy
Certificate metadata	Meta-information about the certificate itself	Serial #, Precert. Poison, Signed Cert. Timestamps

Table 1: Certificate Information Taxonomy

Invalidation Event	Category	Example	Security Implications
Domain control change			
Change in ownership	Sub. Authen.	Domain registrant change (**§ 5.2**)	Third-party. TLS domain impersonation.
Change in use	Sub. Authen.	Domain expiration + no new owner	First-party. Minimal.
Key control change			
Change in ownership	Sub. Authen.	Key compromise (**§ 5.1**)	Third-party. TLS domain impersonation.
Change in use	Sub. Authen.	Key disuse: e.g., rotation	First-party. Minimal.
		*managed TLS departure (**§ 5.3**)	*Third-party. TLS domain impersonation.
Key authorization change	Key Author.	Key scope reduction	First-party. Over-permissioned server/client authentication, email/-code/OCSP signing.
Revocation info. change	Issuer Info.	CA infrastructure change	First-party. Minimal.

Table 2: Certificate Invalidation Events

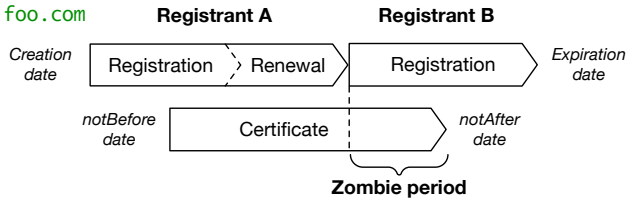


Figure 2: Domain registrant change—Using WHOIS registry creation dates, we identify re-registrations and intersect them with certificate lifespans.

3.1 Subscriber authentication

Subscriber authentication fields are the most important information in a certificate. They identify the entity (or entities) that is being authenticated, which for TLS certificates typically consists of 1) a DNS domain name and 2) a cryptographic public key. Operational changes related to either of these subscriber identifiers can invalidate a previously accurate certificate, and in some cases, enable domain impersonation because the operator of the stale certificate's key may no longer represent the subscriber. For example, if the owner of a domain changes, or a third-party acquires the private key for a certificate, the certificate is invalidated. Below, we detail invalidation events for domain and key control, which are both subject to changes in ownership and changes in usage.

Domain control When a domain changes ownership, any existing valid certificates are invalidated because the public key in the certificate no longer authenticates the domain. As shown in Figure 2, assume that the registrant for `foo.com` obtains a certificate for the domain. If domain ownership for `foo.com` changes to a new registrant, the prior registrant still retains control of the cryptographic keys for the previously issued certificate. If this change occurs before the expiration of the certificate, then the old registrant has the technical ability to impersonate `foo.com` even though it is controlled by the new owner. Although this paper does not seek to measure active exploit of stale certificates, we highlight a few scenarios in which an attacker might realistically abuse domain registrant change. Domain squatters could amass certificates for their squatted domains prior to selling them. Alternatively, malicious actors could abuse registrar return policies [2–4] to purchase a domain, obtain a 13-month certificate for it, and then promptly unregister the domain for a full refund. This form of attack has almost zero cost and does not require the attacker to interact with the victim who subsequently acquires the domain.

In addition to changes in domain registrant, a domain may undergo a change in usage, going from used to unused. This is technically a certificate invalidation event, since the certificate key should no longer authenticate the domain; however, the security implications are minimal because the stale certificate is not controlled by a third-party and does not increase the attack surface for the unused domain. However, unused domains expire and, if purchased by a new owner, could result in domain registrant changes as described above.

Key control Changes in key ownership are invalidation events when an unauthorized third-party gains access to the private key of a valid certificate, also known as key compromise. Key compromise indicates exposure of a private TLS key, allowing a third-party to potentially obtain the key to impersonate any domains connected to the key through a certificate. Similar to domain registrant change invalidation, key compromise enables a well-positioned third-party

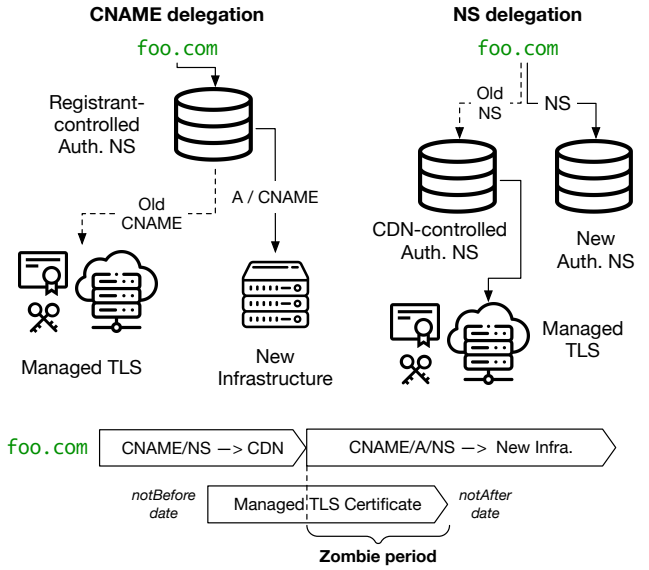


Figure 3: Managed TLS change—We detect changes for both CNAME and NS delegation to CDNs that offer managed TLS.

to perform TLS interception. Key compromise is difficult to identify directly in the wild, so Section 5.1 examines key compromise through the lens of certificate revocation, which are reported invalidations.

Changes in key use can also constitute certificate invalidation events. When a key in a TLS certificate ceases to be used before the certificate expires (e.g., key rotation = disuse + new key), the certificate information becomes stale. In most cases, this has almost no effect on security, since the stale certificate minimally increases the TLS attack surface by permitting an additional public key, which is either difficult to crack or already had security concerns prior to disuse. However, in the case of managed TLS services, key disuse can allow a third-party to impersonate a domain that they are no longer trusted for. For example, when a domain registrant points their traffic to the CDN or web hosting service, the CDN / web hosting service will automatically issue a certificate for that domain and fully control the private key for that certificate. When a domain registrant migrates away from one of these services (Figure 3) to any other infrastructure (e.g., self-hosting or alternate managed TLS service), the prior CDN / web hosting service is left with control of the certificate keys for a domain that they no longer manage. Given the accelerating adoption of CDNs and managed TLS services [44], we expect a growing number of domains to be susceptible to this form of stale certificates, which we empirically evaluate in Section 5.3.

3.2 Key authorization

TLS certificates specify what cryptographic operations (e.g., encryption, signing) and what forms of authentication (e.g., server/client authentication, code-signing, etc.) a given cryptographic key is authorized for. Key authorizations are requested by the subscriber and enforced by the CA, which performs different types of verification during certificate issuance for different authorized uses. If a domain owner replaces or removes an existing key authorization (e.g., switches a key from code signing to server authentication), the prior certificate becomes a stale certificate. This form of stale certificate is rare, since most websites / organizations have independent PKI infrastructure for different use cases and are unlikely to reuse or share keys between them.

3.3 Issuer information

TLS certificates contain a wide range of issuer information, and most of the information is a retrospective attestation of historical facts. For example, the issuer name and certificate policies included in a certificate indicate specific context that was true at the time of issuance. Certificate revocation information (i.e., CRL and AIA/OCSP information), on the other hand, points to active infrastructure that needs to be maintained throughout the lifetime of a certificate. Changes to a CA's revocation infrastructure constitute a certificate invalidation event that can prevent access to accurate revocation information. The security implications are minimal, however, since revocation is unreliable to begin with.

3.4 Security Consequences

First-party stale certificates The majority of certificate invalidation events lead to stale certificates controlled by the domain owner. For example, when a certificate's key becomes unused during the certificate's lifetime, only the domain owner or managed TLS service can still access the unused private key. The security considerations of first-party stale certificates are often minimal and limited to mistaken actions taken by the domain owner.

Third-party stale certificates Three certificate invalidation scenarios—key compromise, domain registrant change, and managed TLS change—result in an untrusted third-party controlling the keying material of a valid certificate for a domain they do not operate. This allows the third-party to impersonate the domain and potentially perform TLS interception attacks, given on-path network positioning between a client and server, which can be achieved through local (e.g., ARP spoofing, ISP) or global methods (e.g., DNS poisoning, nation-state actor). The web PKI community has historically acted assertively in response to other issues which similarly resulted in a third-party having the ability to impersonate another party's domain [62, 63].

We emphasize that stale certificates arising from two scenarios (i.e., domain registrant change and managed TLS departure) occur naturally, without domain owner mistake or active attacker involvement. Our results indicate that third parties have already had control of valid certificate keys for over 3M domains that they do not operate. Unlike interception arising from CA compromise or malicious root injection [26, 29, 42], these stale certificates are stealthy and cannot be easily detected via existing mechanisms such as Certificate Transparency (CT). This is because these certificates start out as legitimately issued certificates and only become abusable by a third-party later on, which CT does not directly detect.

Dataset	Used for	Date range	Size	Details
CT	Revocations Managed TLS Registrant change	2013/03 – 2023/05	5B certs (deduplicated)	Certificate Transparency entries from logs trusted by Apple or Google Chrome.
CRL	Revocations	2022/11 – 2023/05	31M total CRLs from 92 CAs	Daily collection of 5.4K cert revocation lists.
WHOIS	Registrant change	2016/01 – 2021/07	4B records (301M domains)	`.com` and `.net` domain registration info.
aDNS	Managed TLS	2022/08/01 – 2022/10/30	300M A/AAAA, 274M NS, 10M CNAME records per day	Daily DNS scans for all e2LDs in public zones.

Table 3: Datasets—We leverage large-scale certificate management, domain management, and network infrastructure datasets to detect stale certificates.

4 METHODOLOGY

To detect stale certificates and understand their impact on the web PKI, we first measure network operational dynamics that invalidate the subscriber information of TLS certificates. In particular, we focus on three classes of stale certificates that introduce precarious third-party access to valid TLS keys. We want to understand 1) how common these stale TLS certificates are, 2) what types of domains are at risk, and 3) whether we can effectively combat stale certificates by reducing certificate lifetimes.

The primary dataset used for measuring third-party stale certificates is Certificate Transparency (CT), which is a set of log servers that, in aggregate, publishes all TLS certificates trusted by Google Chrome and Apple Safari. We collected these certificates up to May 12, 2023 from 117 CT logs (including sharded logs) that were trusted by Google Chrome [35] or Apple [9] at some point in time. We deduplicate precertificates and issued certificates based on their non-CT components, resulting in a total of 5B certificates analyzed. We also ignore fully qualified domain names (FQDNs) that have more than 3K certificates (< 0.0004% of all domains) since they are either test domains (e.g., `flowers-to-the-world.com` [37]) or represent an anomalous case of certificate issuance.

4.1 Key compromise revocation

Although certificates typically contain embedded revocation information, we utilize a more convenient source of revocation information. Since October 2022, Mozilla has required CRL disclosure for all Mozilla-trusted certificates [72], effectively aggregating all certificate revocation information. We downloaded all disclosed CRLs once a day from November 1, 2022 until May 5, 2023. Some CRL servers had protections against automated scraping, but in total we were able to successfully download and parse over 4,900 (>97%) of daily CRLs (Appendix B), collecting a total of 31.7M revocations. CRLs do not include a full copy of revoked certificates—they only indicate the authority key ID (i.e., issuer key), the certificate serial number, revocation time, and revocation reason—so we cross-referenced the revocations with all certificates found in CT, resulting in a total of 21.39M revoked certificates. We further removed 129 certificates (0.0006%) that were revoked before becoming valid, 7945 certificates (0.037%) that were revoked after expiration, 33,860 (0.16%) certificates that were revoked prior to October 1, 2021 (13 months prior to CRL collection). These filters remove outlier CRL data that do not represent normal certificate revocation behaviors. When assessing the staleness period of revoked certificates, we conservatively assume that the revocation was issued as soon as the invalidation event occurred.

4.2 Domain registrant changes

To detect changes in domain registrant that result in stale certificates, we utilized bulk historical WHOIS data collected by an industry partner from January 1, 2016, to July 8, 2021. WHOIS data is notoriously difficult to rely on due to inconsistent formatting of responses across registrar [56], inaccuracy in registrant fields, and redacted responses in GDPR-protected WHOIS records [58]. We restricted our analysis to the `com` and `net` top-level domains (TLDs) for which Verisign is the registry, since the WHOIS records are consistently structured and generally reliable. Additionally, we only considered the fields contained in a "thin" WHOIS entry as they are controlled by the registry (Verisign) rather than a registrar. We recorded the (Domain, Registry Creation Date) pair in our WHOIS dataset for each new registration, a technique commonly used by prior works [45, 50, 51, 60]. For each domain registration, we identified stale certificates with periods of validity that span the new registration date: `notBefore` < `registryCreationDate` < `notAfter`. For certificates that met these criteria, the stale period was determined by the time beginning at registrant change (`registryCreationDate`) and ending at the `notAfter` date of the certificate. Overall, we employed a conservative registrant change detection methodology that generally prioritizes precision over recall. Thus, our measurements should be viewed as a lower bound on stale certificates arising from domain registrant changes.

4.3 Managed TLS change

Based on a list of the top CDNs [54], we found that all CDNs supported customer-uploaded HTTPS certificates and twelve out of twenty-six provided CDN-managed certificates (i.e., issuance, renewal). However, only one CDN Cloudflare had easily discernible managed TLS certificates, which all include the `sni*.cloudflaressl.com` domain along with the customer domain in the Subject Alternate Name (SAN) extension. The inclusion of a Cloudflare domain in the certificate provides the added benefit of distinguishing between Cloudflare managed and user uploaded certificates. Other managed TLS services utilize popular public CAs (e.g., DigiCert, Let's Encrypt) and we cannot easily distinguish their managed TLS certificates from self-managed TLS certificates. Although we only examine domains that utilize Cloudflare's managed TLS services,

Cloudflare is the largest CDN by number of customers [44], and this is reinforced by the fact that Cloudflare's CA is a top five CA by issuance volume.

For this investigation, we employed a dataset of active DNS scans. We first extracted the domains from all publicly available zone files from the Centralized Zone Data Service (CZDS) [1], which includes the popular `org`, `com`, and `net` zones. We resolved each domain on a daily basis, and collected records containing information about a domain's IPs (A/AAAA), nameservers (NS), and canonical names (CNAME). Given the sheer size of our DNS collection, we had to limit the time window of the experiment to three months (Table 3). For each Cloudflare managed TLS domain, we compared each day's NS and CNAME records with neighboring days. We detected departure from the Cloudflare managed TLS if any of the Cloudflare nameservers or CNAMEs (`*.<ns,cdn>.cloudflare.com`) appeared on the first day and were absent on the next day.

4.4 Limitations

Generally, our methods for detecting key compromise, domain registrant change, and managed TLS departure err on the side of accuracy over complete coverage. Our results are lower bound estimates on the overall population of certificate invalidation events and third-party stale certificates.

Domain registrant tracking Our detection method misses intra- and inter-registrar domain transfers, as well as pre-release re-registration, both of which update existing registrations rather than creating a new registration with a new creation date. First, due to a lack of ground truth data, domain transfer is difficult to infer based on public WHOIS information whether the domain recipient is the same entity as the transfer initiator. This would require tracking individual registrants across multiple accounts within a single registrar and also across multiple registrars. Furthermore, the rapid growth of privacy-preserving WHOIS in recent years [58] makes registrant tracking exceedingly difficult. Second, as expressed in prior work [50], pre-release re-registration is an opaque process that cannot be easily measured. Third, we make the assumption that a new registry created date signals a new registrant; however, it is possible for the prior domain registrant to re-acquire their own domain. We argue that this is unlikely because a registrant that wants to keep their domain would renew their domain during the 45 day grace period or 30 day redemption period, and not risk losing the domain when it becomes available to the general public.

Domain validation reuse CAs can skip domain-validation certificate issuance checks for a subscriber account if they have received evidence of the subscriber's domain control within the last 398 days. This "domain validation reuse" practice can result in a certificate that is stale from the moment that it is issued. This study does not examine this form of staleness and assumes that certificates are actively validated just before issuance.

IP routing changes We do not look at the extent to which BGP hijacks interfere with the routing of connections between a web client and web server. While other stale certificates occur naturally by design, BGP hijacks violate expected design (even though BGP has minimal security). We also do not measure legitimate IP acquisitions, which occur on a longer timescale and require dangling DNS records, which are beyond the scope of this work.

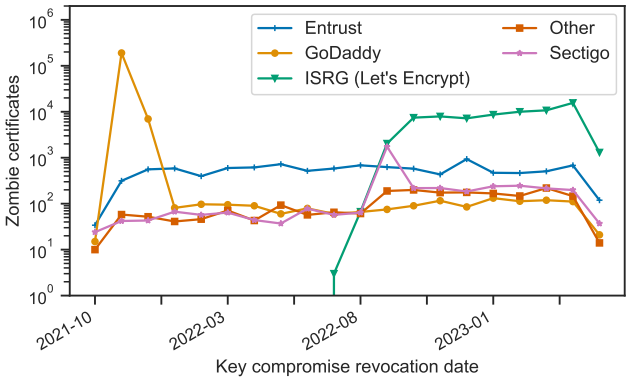


Figure 4: Monthly key compromise volumes show the concentration of reporting within a few CAs.

5 STALE CERTIFICATES

Stale certificates arise from the mismatch between the static information contained within a certificate and the dynamic domain/key/CA infrastructure that it represents. We focus on third-party stale certificates and quantify the occurrence of the three instances where a third-party can utilize a stale certificate to impersonate a domain that they do not control.

5.1 Key compromise

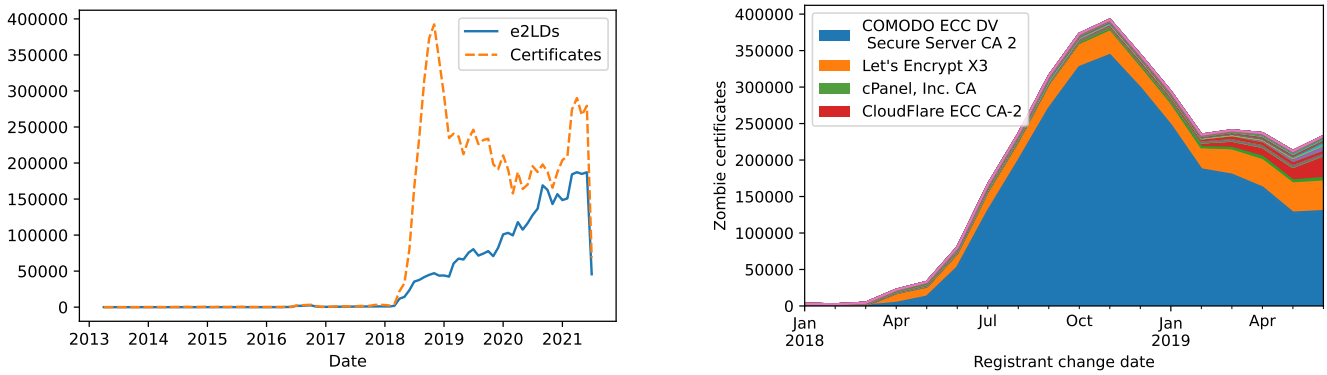
The most pressing revocation reason is key compromise, which implies that an unauthorized third-party has actively or passively (e.g., logged by network appliance) obtained a copy of the private key used to authenticate a subscriber's domain. 2.42% (286K) of revocations are labeled as key compromise, which accounts for 202K effective second-level domains. Over 65% of these revocations were issued by GoDaddy in November and December 2021 (Figure 4) as a result of a major breach that exposed TLS private keys [6] for a subset of its 1.2M impacted customers. Excluding this acute event, we see that key compromise revocation increased between the end of 2021 and early 2023. Let's Encrypt began publishing key compromise revocation in July 2022, but baseline key compromise revocation has increased gradually even without this influx.

5.2 DNS registrant change

In total, we found 7.7M stale certificates (3.6M e2LDs) resulting from a domain registrant change that intersects a valid certificate. Nearly all of these stale certificates occurred after the introduction of Let's Encrypt in 2018 which multiplied the number of domains utilizing TLS certificates (Figure 5a). The number of e2LDs with stale certificates increases gradually, while the total number of stale certificates spiked in November 2018, implying a large number of certificates per e2LD. To better understand this growth, we examined the issuers for stale certificates from domain registrant change from 2018–2019 and found that the majority of the stale certificates during this period were issued by COMODO (Figure 5b). Nearly

Method	Date range	# Stale certs		# Stale FQDNs		# Stale e2LDs	
		Daily	Total	Daily	Total	Daily	Total
Revoked: all	2021/10/01 – 2023/05/05	20,327	11.8M	28,0352	16.4M	7,125	4.1M
Revoked: key compromise	2021/10/01 – 2023/05/05	493	286K	787	457K	347	202K
Domain registrant change	2013/04/16 – 2021/07/09	2,593	7.7M	2,807	8.4M	1,214	3.6M
Cloudflare managed TLS departure	2022/08/01 – 2022/10/30	9,495	854K	18,833	1.69M	7,722	695K

Table 4: Stale certificate detection—The average daily rates of new stale certificates, domains, and e2LDs show differing magnitudes of third-party staleness.



(a) New monthly stale certs **(b) Spike investigation**

Figure 5: Domain registrant change—Stale certificates have grown drastically since 2018.

Malware	*352 domains*	URL	*685 domains*
grayware	82	phishing	367
backdoor	74	malicious	190
Unknown	53	malware	128
downloader	51		
virus	29		
spyware	27		
ransomware	18		
Other	18		
MW only (328)		MW + URL (24)	URL only (661)

Table 5: Domain reputation—1% of 100K randomly sampled domains have malicious activity that temporally coincides with stale certificate control.

all the COMODO-issued stale certificates are Cloudflare "cruise-liner" certificates [19], which contain dozens of distinct Cloudflare customers in a single certificate. For a single Cloudflare customer domain, we observe hundreds of temporally-overlapping certificates, which only differ by a handful of inserted or removed domains. This suggests that Cloudflare issues new certificates whenever a new domain enrolls in managed TLS, or when a domain leaves. By the middle of 2019, we see decreasing usage of cruise-liner certificates by Cloudflare, and increasing issuance of per-domain certificates issued by Cloudflare's own CA.

Domain reputation Stale certificates from domain registration are especially worrisome if the prior domain owner is malicious. We used VirusTotal (VT) [5] to determine the reputation of domains found on stale certificates. We analyzed a random sample of 100K domains with stale certificates from domain owner change . For each domain, we queried VT to find associated malicious URLs and malicious files (flagged by at least five vendors). To correlate the period of malicious domain activity with stale certificates, we identified the minimum `first_submission` date across all domains associated with malicious files, or the first date that five or more vendors labeled a malicious URL. We used AVClass2 [66] to identify malware families and manually inspected the resulting malware family labels and resolved aliases using Malpedia [64]. For each domain with malicious URLs, we tallied the number of unique security vendors labeling the domain as either `malware`, `phishing`, or `malicious`.

Table 5 illustrates the types of malware and malicious URLs associated with the 1,013 domains that exceed the VT detection threshold described above. While the overall percentage of malicious stale certificate domains is small, we find evidence of malicious operators in control of stale certificates. Our existing datasets do not provide insight into stale certificate usage, which future work should address.

5.3 Managed TLS departure

Over the course of three months from August 1, 2022 to October 30, 2022, we observed 854K stale certificates representing 695K e2LDs (Table 4). While we cannot directly compare this sum with the longer period of stale certificates from key compromise revocation

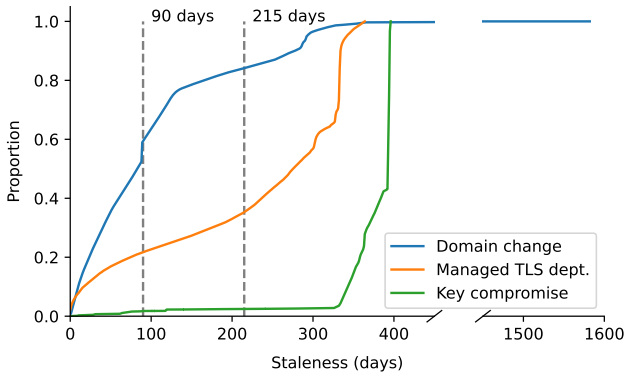


Figure 6: Third-party staleness—Over 50% of third-party stale certificates have staleness periods exceeding 90 days, across all types.

Alexa Rank	Domain Reg. change	Managed TLS dept.	Key compromise
Top 1K	8	12	41
Top 10K	307	127	217
Top 100K	5,839	1,742	928
Top 1M	84,319	14,776	6,771
Total domains	3,649,526	695,064	201,662
% of total	2.5%	2.4%	3.9%

Table 6: Domain popularity—A small percentage of domains in stale certificates appear in biannual samples of the Alexa Top 1M between 2014–2022.

or domain registrant changes, we find that on a daily basis, managed TLS change accounts for more stale certificates per day, even though we only examine a single managed TLS provider, Cloudflare. From a security threat perspective, this means that a single CDN Cloudflare may have the cryptographic material to intercept the TLS connections of over 695K former customer domains that have actively migrated away from Cloudflare's services. While we cannot infer the intent of domains that move away from Cloudflare, and there are benign reasons for moving (e.g., better competitor product / pricing), we must consider the worst-case scenario satisfactorily in order to ensure security. As seen in the prior example of GoDaddy's managed Wordpress (with TLS) breach [6], which affected over a million customers, there are cases where the prior managed TLS service can be justifiably untrusted and poses a lingering threat to domain owners who are attempting to migrate away.

5.4 Third-party Staleness Overview

One critical aspect of third-party stale certificates is how long they endure and how long a third-party can retain questionable access to another party's TLS keys. Figure 6 depicts the distribution of staleness periods across all three forms of third-party stale certificates. A certificate's staleness period is a function of two variables: the certificate's total lifetime and when within that lifetime an invalidation event occurs. Both managed TLS departure (300 days) and key compromise (398 days) have longer median staleness periods than domain registrant change (90 days), indicating that the harms introduced by the former two scenarios extend three or four times longer than domain registrant change.

Although we consider all three cases to be precarious access to TLS keys, the number of daily third-party stale certificates and e2LDs (Table 4) appears to correlate with the approximate operational proximity of the third-party: key compromise can result from an active adversary, domain ownership change empowers the previous domain owner, and managed TLS departure overextends the privileged capabilities of the prior CDN/virtual hosting service used by the domain. Due to measurement incompleteness, we caution over-interpretation of these results, which could result from observing a large proportion of one staleness class than others. However, because we have a global revocation perspective, (compared to one CDN and only post-expiration domain change), we can infer that the true volumes of stale certificates arising from domain registrant and managed TLS change are likely much larger than key compromise. Although these non-compromise stale certificates may be more difficult for an attacker to successfully weaponize, their natural abundance makes them concerning.

Domain popularity We measure the popularity of domains found in stale certificates in order to understand the types of websites that suffer from third-party certificates. We take a biannual (every six months) sample of Alexa Top 1M domains from 2014–2022 and examine the most popular (lowest) rank that a domain in a stale certificate has appeared. Because Alexa popularity lists only contain e2LDs, we match certificate domain names based on their e2LD. Table 6 shows that the overwhelming majority of domains found in stale certificates fall into the long tail of low popularity domains. However, it also demonstrates that popular domains (or formerly popular) domains are not immune to stale certificates. Top domains are unlikely to undergo registrant change during the height of their popularity, but managed TLS departure can realistically occur as part of an infrastructure migration. Similarly, key compromise can occur to any popularity domain. Most instances of popular key compromise domains are due to certificates for subdomains; for example, we observe compromises for the domains `cp8.cloudflare.com` and `mqasmartaudit.apple.com`.

Takeaways After accounting for major outliers, we find that the number of stale certificates is growing steadily. Third-party stale certificates affect domains of varying popularity including several top 1K domains, and in some cases, are associated with malware and malicious URLs. Stale certificates from key compromise invalidation tend to last many times longer than other forms of third-party staleness.

6 REDUCING CERTIFICATE LIFETIMES

Due to the ineffectiveness of certificate revocation, certificate expiration is an appealing defense mechanism against third-party stale certificates. The CA/Browser Forum has repeatedly discussed and adjusted the maximum validity period, with some arguing for shorter periods to increase PKI agility (e.g., phasing out SHA-1 certificates took three years [73]) and some resisting reduction in order to reduce operational burden [12]. In 2017, the CA/Browser

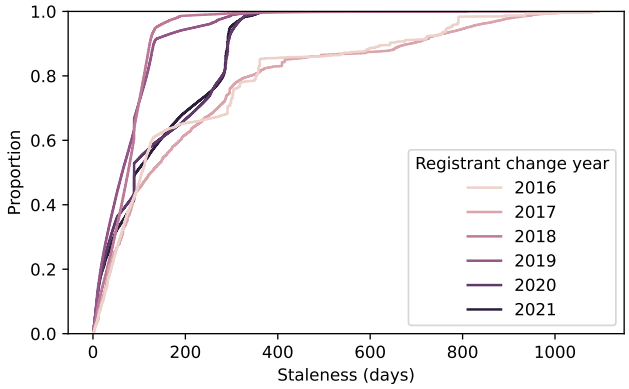


Figure 7: Domain owner staleness from 2016–2021 shows mixed results: decreased maximum staleness and fluctuating average staleness.

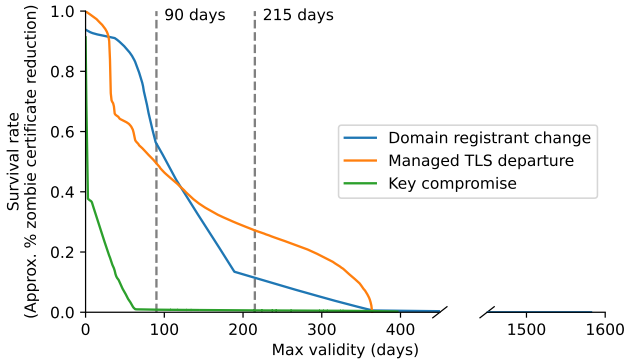
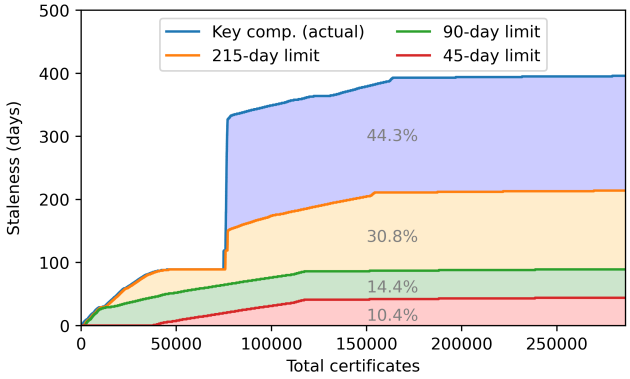


Figure 8: Certificate survival rate—56% of registrant change, 49.5% of managed TLS departure, and 1% of key compromise occur after 90 days of issuance.

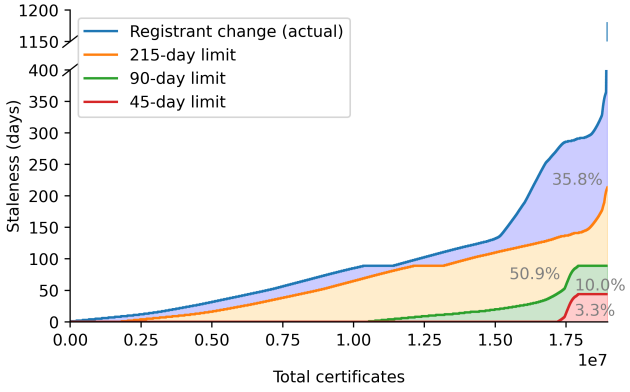
Forum passed a restriction to limit the maximum validity of DV certificates to 825 days [17]. Then, in September 2020, browsers began to enforce a 398-day maximum limit [10, 34, 71], which is based upon annual (up to 366 days) CA customer certificate renewal, with a one maximum-month (31 days) period to perform renewal, plus a one day additional buffer.

Despite these reductions, the number of third-party stale certificates continues to increase (Figure 4, 5a). Furthermore, based on our domain registrant change data from 2016 through July 2021 (Figure 7), we observe mixed improvements in average staleness duration after maximum lifetime reduction beginning September 2020. The tail of high-duration staleness in 2016/2017 is curtailed after 2018, but the average staleness periods for domain registrant change increased between 2019 and 2020, while remaining the same between 2020 and 2021. To prevent third-party stale certificate growth from outpacing certificate lifetime reductions, we explore how further lifetime restrictions impact both the quantity and quality (staleness time) of third-party stale certificates.

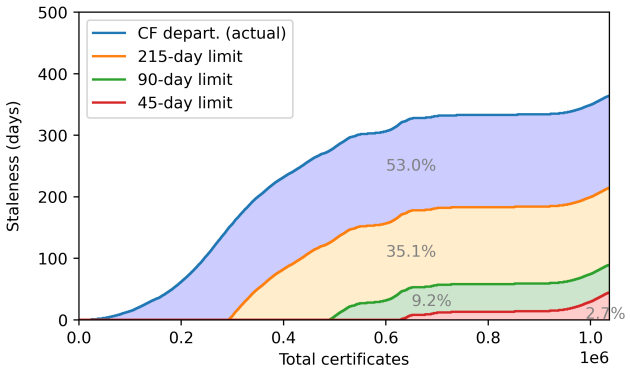
To estimate the effects of further shortening certificate validity periods, we examine the impact of a 90-day maximum lifetime,



(a) Key Compromise



(b) Domain Registrant Change



(c) Managed TLS Departure

Figure 9: Simulated staleness—Max lifetime reduction to 215 days could eliminate 36–53% of third-party staleness days.

which some CAs already self-enforce on all certificates that they issue [16]: Let's Encrypt, Google Trust Services (GTS), and cPanel. We also consider 45-day and 215-day max lifetime, which is the longest stretch of six months plus an extra month of padding for operational considerations, similar to the current 398-day limit. To quantify the improvement of each hypothetical lifetime of n days, we perform an experiment in which we take all stale certificates with lifetime greater than n and decrease their certificate expiration date to achieve a total lifetime of n. We do not modify certificates

with lifetimes less than n. Using this method, we see the following reductions in relative staleness-days: 96.7%, 86.7%, and 35.8% reduction for domain registrant change reflecting 45-, 90-, and 215-day maximum lifetimes, respectively; 97.7%, 75.3%, and 45.3% decrease for Cloudflare manage TLS departure (Figure 9c); and 89.6%, 75.2%, and 44.3% reduction for key compromise (Figure 9a).

Next, we examine how quickly certificates become stale. Figure 8 indicates the proportion of certificates that have not yet become stale after a certain number of days. Based on this survival analysis, we naively estimate that with a 90-day maximum certificate lifetime, we could eliminate up to 49.5% of stale managed TLS change certificates and up to 56% reduction for domain registrant change. For a 215-day limit, the theoretical reduction would be 29.5% and 14.5%, respectively. This is an upper-bound estimate, since it is assumes that existing stale certificates would not be renewed if they expired earlier within 90 or 215 days.

Takeaways Certificate expiration is the last line of defense against all forms of problematic certificates. Taking a data-backed approach, we study the impact of shortening the maximum certificate lifetime to 45-, 90-, or 215- days. We estimate a 1–56% reduction in stale certificates, depending on the type of certificate invalidation, and 75% decrease in overall stale days if 90-day certificate lifetimes are enforced.

7 DISCUSSION

This work exposes stale certificates as a prevalent, growing, and worrisome phenomenon in the web PKI. In this section, we 1) discuss the root causes of stale certificate growth in light of our findings and 2) explore the potential mitigations and future implications of stale certificates.

7.1 Stale certificate growth

Our measurements of domain registrant change, globally increasing CDN adoption [44], and widespread growth in HTTPS deployment [36] suggest that stale certificates, both third-party and first-party, are increasing over time. Third-party stale certificate growth, which leaves more and more valid TLS credentials in the wrong hands, stems from several trends. First, the rising tide of HTTPS means that as more websites adopt TLS, the absolute number of stale certificates will increase given all else equal. Second, assuming relatively fixed customer attrition rates, the growing usage of CDNs and shared web hosting services will further inflate the number of stale certificates arising from managed TLS departure. Third, *automated* certificate issuance has been a boon to the first two growth factors; however, it can lead to *automatic* issuance [20] that independently exacerbates the stale certificate issue. A domain registrant that is intending to sell or stop using their domain will likely halt manual certificate issuance. Similarly, a domain registrant intending to leave a managed TLS provider would not continue requesting new certificates. In these scenarios, unattended automatic certificate issuance can inadvertently extend a soon-to-be-broken name-to-key mapping and increase the number of third-party stale certificates. Issuance automation is a double-edged sword since it is also the only proven path towards further reductions in certificate lifetimes.

7.2 Mitigation and future implications

As shown in Section 6, one promising mitigation to stale certificates is shortening their lifetime, which is a perennial discussion topic in the PKI community [17, 67, 70]. However, certificate lifetime policies must also consider the operational burden on the web PKI. Several major CAs (e.g., Let's Encrypt, cPanel) have employed automated certificate issuance and already enforce self-imposed 90-day limits. Increased issuance also places additional load on Certificate Transparency logs, which have introduced temporal log sharding and tighter upload criteria to handle TLS certificate growth. Further PKI operational cost reduction (e.g., more automation, protocol optimization) can tip the tradeoff between operational costs and third-party stale certificates.

Certificate revocation is another promising mitigation, but unfortunately, it is absent in many browsers or does not protect against active TLS interception (Section 2.4). If new proposals such as CRLite [49] gain adoption and overcome hard-fail hurdles (avoid becoming a denial-of-service vector), revocation could be an effective defense against third-party certificate staleness. Additionally, targeted mitigations against individual forms of staleness such as keyless CDN protocols [39] and Cloudflare's Keyless SSL service [24] would add layers of defense and drastically reduce manged TLS staleness, which is the largest contributor.

Finally, certificate staleness due to domain registrant and managed TLS change is rooted in the current design of the web PKI, which attempts to bridge independent domain, web server, and certificate lifecycles. A more systemic solution is to decrease the network dependency chain between DNS names and their corresponding cryptographic keys, thus reducing the opportunities for unintentional or malicious dependency fracturing. Proposals such as DNS-Based Authentication of Named Entities (DANE) [40] and Named Data Networking (NDN) [74] align cryptographic keys with the authoritative source for name information, thus empowering name operators and likely reducing authentication cache durations (hours-scale TTLs for DANE) [43]. From a trust standpoint, these proposals condense intermediate, third-party network dependencies onto a registrar or nameserver operator, which are already trusted today as the entrypoint to most internet connections.

8 RELATED WORK

Managed TLS Liang et al. were the first to examine the rise of managed TLS providers [54]. They found that CDN customers could not revoke TLS certificates issued on their behalf, the first anecdotal instance of a stale certificate. Subsequent analysis of the HTTPS key sharing ecosystem in 2016 [19] reinforced the prevalence of managed TLS services, even prior to the introduction of automated issuance.

Problematic certificates Researchers have previously identified several large-scale forms of problematic certificates: certificates with weak cryptography [28, 41] or non-compliant formatting [47], invalid certificates [21] that may be accepted by poorly implemented TLS clients [32], long-lived non-leaf certificates [59], and forged certificates [26, 29, 42]. All of these problematic certificates directly involve CA error, TLS client implementation bugs, or malware exploitation. In contrast, stale certificates do not result from

obvious errors or exploits by CAs or other web PKI participants—they are a direct consequence of the design of the web PKI which separates semantic and cryptographic identities across many layers of network indirection. Our work builds off BygoneSSL [31] and expands on the more general phenomenon of certificate invalidation events leading to abusable, third-party stale certificates.

Stale and dangling records Stale certificates are one instance of the broader phenomenon of stale/dangling network records. Dangling records represent disuse by the record owner, and unauthorized access to abandoned resources typically has limited direct impact. However, stale records are the result of a performance and accuracy (and often security) trade-off that can impact records in active use. Instead of mapping domains to cryptographic keys, DNS records map domains to IP-based locations, and prior work [15, 55] has abused stale/dangling DNS records to acquire TLS certificates for third-party domain names. Stale certificates are generally abusable for much longer than stale DNS records, due to differences in typical DNS TTLs (hours/days) and certificate validity lifetimes (months/years).

9 CONCLUSION

The design of today's web PKI leads to certificates that contain stale information. We showed three scenarios in which a third-party gains access to a valid TLS key, enabling potential domain impersonation and TLS interception. Utilizing multiple network datasets, we found that third-party stale certificates currently exist for hundreds of thousands of domains. Unfortunately, the most obvious mitigation, certificate revocation, is absent or easily circumvented in modern browsers. Thus, we explored the alternative solution of shortening certificate lifetimes and found evidence that broadly enforcing a maximum 90-day certificate lifetime would lead to 75–86% reduction in overall staleness. Ultimately, we consider PKI design changes that can address stale certificates at large.

ACKNOWLEDGEMENTS

We would like to thank Dana Keeler for providing insight into certificate revocation.

REFERENCES

[1] [n.d.]. Centralized Zone Data Service. https://czds.icann.org/home.
[2] [n.d.]. GoDaddy - REFUND POLICY. https://www.godaddy.com/legal/agreements/refund-policy.
[3] [n.d.]. Google Domains refund policy. https://support.google.com/domains/answer/6000754?hl=en.
[4] [n.d.]. Namecheap refund policy. https://www.namecheap.com/legal/general/refund-policy/.
[5] [n.d.]. VirusTotal. https://www.virustotal.com.
[6] 2021. GoDaddy Announces Security Incident Affecting Managed WordPress Service. https://aboutus.godaddy.net/newsroom/company-news/news-details/2021/GoDaddy-Announces-Security-Incident-Affecting-Managed-WordPress-Service/default.aspx.
[7] Josh Aas, Richard Barnes, Benton Case, Zakir Durumeric, Peter Eckersley, Alan Flores-López, J Alex Halderman, Jacob Hoffman-Andrews, James Kasten, Eric Rescorla, Seth Schoen, and Brad Warren. 2019. Let's Encrypt: An Automated Certificate Authority to Encrypt the Entire Web. In *ACM Conference on Computer and Communications Security (CCS)*.
[8] Heather Adkins. 2011. An update on attempted man-in-the-middle attacks. https://security.googleblog.com/2011/08/update-on-attempted-man-in-middle.html.
[9] Apple. [n.d.]. Current Apple CT logs. https://valid.apple.com/ct/log_list/current_log_list.json.
[10] Apple Support. [n.d.]. About upcoming limits on trusted certificates. https://support.apple.com/en-us/HT211025.
[11] Richard Barnes, Jacob Hoffman-Andrews, Daniel McCarney, and James Kasten. 2019. *Automatic Certificate Management Environment (ACME)*. RFC 8555.
[12] Doug Beattie. [n.d.]. Response to: About upcoming limits on trusted certificates. https://groups.google.com/g/mozilla.dev.security.policy/c/mz1buYdIy-I/m/789e7if5AQAJ.
[13] bluehost knowledge base. [n.d.]. How To Activate Free SSL Certificate - Free WordPress SSL Setup. https://www.bluehost.com/help/article/how-to-activate-a-free-wordpress-ssl.
[14] Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, Stephen Farrell, and David Cooper. 2008. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280. https://www.rfc-editor.org/info/rfc5280
[15] Kevin Borgolte, Tobias Fiebig, Shuang Hao, Christopher Kruegel, and Giovanni Vigna. 2018. Cloud Strife: Mitigating the Security Risks of Domain-Validated Certificates. In *Network & Distributed System Security Symposium (NDSS)*.
[16] Carl Magnus Bruhner, Oscar Linnarsson, Matus Nemec, Martin Arlitt, and Niklas Carlsson. 2022. Changing of the Guards: Certificate and Public Key Management on the Internet. In *International Conference on Passive and Active Network Measurement (PAM)*.
[17] CA/Browser Forum. [n.d.]. Ballot 193 – 825-day Certificate Lifetimes. https://cabforum.org/2017/03/17/ballot-193-825-day-certificate-lifetimes/.
[18] CA/Browser Forum. [n.d.]. Baseline Requirements Version 1.8.4. https://cabforum.org/wp-content/uploads/CA-Browser-Forum-BR-1.8.4.pdf.
[19] Frank Cangialosi, Taejoong Chung, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, and Christo Wilson. 2016. Measurement and Analysis of Private Key Sharing in the HTTPS Ecosystem. In *ACM Conference on Computer and Communications Security (CCS)*.
[20] Taejoong Chung, Dave Levin, and Alan Mislove. 2021. Measurement and Analysis of Automated Certificate Reissuance. In *International Conference on Passive and Active Network Measurement (PAM)*.
[21] Taejoong Chung, Yabing Liu, David Choffnes, Dave Levin, Bruce MacDowell Maggs, Alan Mislove, and Christo Wilson. 2016. Measuring and Applying Invalid SSL Certificates: The Silent Majority. In *ACM Internet Measurement Conference (IMC)*.
[22] Jeremy Clark and Paul C Van Oorschot. 2013. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *IEEE Symposium on Security and Privacy (S&P)*.
[23] Cloudflare. [n.d.]. Merkle Town. https://ct.cloudflare.com/.
[24] Cloudflare Docs. 2016. Keyless SSL. https://developers.cloudflare.com/ssl/keyless-ssl/.
[25] cPanel Docs. [n.d.]. Guide to SSL: AutoSSL. https://docs.cpanel.net/knowledge-base/security/guide-to-ssl/#autossl.
[26] X de Carné de Carnavalet and Mohammad Mannan. 2016. Killed by proxy: Analyzing client-end TLS interception software. In *Network & Distributed System Security Symposium (NDSS)*.
[27] Ryan Dickson. 2022. Revocation checking for EV server certificates in Chrome. https://groups.google.com/a/mozilla.org/g/dev-security-policy/c/S6A14e_X-T0/m/T4WxWgajAAAJ.
[28] Zakir Durumeric, James Kasten, Michael Bailey, and J Alex Halderman. 2013. Analysis of the HTTPS Certificate Ecosystem. In *ACM Internet Measurement Conference (IMC)*.
[29] Zakir Durumeric, Zane Ma, Drew Springall, Richard Barnes, Nick Sullivan, Elie Bursztein, Michael Bailey, J. Alex Halderman, and Vern Paxson. 2017. The Security Impact of HTTPS Interception. In *Network & Distributed System Security Symposium (NDSS)*.
[30] Sascha Fahl, Yasemin Acar, Henning Perl, and Matthew Smith. 2014. Why Eve and Mallory (Also) Love Webmasters: A Study on the Root Causes of SSL Misconfigurations. In *ACM ASIA Conference on Computer and Communications Security (ASIACCS)*.
[31] Ian Foster and Dylan Ayrey. 2018. Lost and Found Certificates. https://insecure.design/BygoneSSL_DEFCON.pdf.
[32] Martin Georgiev, Subodh Iyengar, Suman Jana, Rishita Anubhai, Dan Boneh, and Vitaly Shmatikov. 2012. The Most Dangerous Code in the World: Validating SSL Certificates in Non-browser Software. In *ACM Conference on Computer and Communications Security (CCS)*.
[33] GoDaddy Help Center. [n.d.]. Set up my Managed SSraiL Certificate. https://my.godaddy.com/help/set-up-my-managed-ssl-certificate-32212.
[34] Google. [n.d.]. Certificate Lifetimes. https://chromium.googlesource.com/chromium/src/+/master/net/docs/certificate_lifetimes.md.
[35] Google. [n.d.]. Chrome Compliant CT logs. https://www.certificate-transparency.org/known-logs.
[36] Google Transparency Report. [n.d.]. HTTPS encryption on the web. https://transparencyreport.google.com/https/overview.
[37] Paul Hadfield. 2017. Response to Medium post. https://medium.com/@hadfieldp/hey-ryan-c0fee84b5c39.
[38] Phillip Hallam-Baker. 2015. X.509v3 Transport Layer Security (TLS) Feature Extension. RFC 7633. https://www.rfc-editor.org/info/rfc7633

[39] Stephen Herwig, Christina Garman, and Dave Levin. 2020. Achieving Keyless CDNs with Conclaves. In *USENIX Security Symposium (Sec)*.

[40] Paul E. Hoffman and Jakob Schlyter. 2012. The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA. RFC 6698. https://www.rfc-editor.org/info/rfc6698

[41] Ralph Holz, Lothar Braun, Nils Kammenhuber, and Georg Carle. 2011. The SSL Landscape: A Thorough Analysis of the X.509 PKI Using Active and Passive Measurements. In *ACM Internet Measurement Conference (IMC)*.

[42] Lin Shung Huang, Alex Rice, Erling Ellingsen, and Collin Jackson. 2014. Analyzing Forged SSL Certificates in the Wild. In *IEEE Symposium on Security and Privacy (S&P)*.

[43] Geoff Huston. 2022. What's going on with certificate revocation? https://blog.apnic.net/2022/03/22/whats-going-on-with-certificate-revocation/.

[44] Intricately Market Analysts. 2020. CDN Industry: Trends, Size, And Market Share. https://blog.intricately.com/cdn-industry-trends-market-share-customer-size.

[45] Maciej Korczynski, Maarten Wullink, Samaneh Tajalizadehkhoob, Giovane CM Moura, Arman Noroozian, Drew Bagley, and Cristian Hesselman. 2018. Cybercrime After the Sunrise: A Statistical Analysis of DNS Abuse in New gTLDs. In *ACM ASIA Conference on Computer and Communications Security (ASIACCS)*.

[46] Katharina Krombholz, Wilfried Mayer, Martin Schmiedecker, and Edgar Weippl. 2017. "I Have No Idea What I'm Doing" - On the Usability of Deploying HTTPS. In *USENIX Security Symposium (Sec)*.

[47] Deepak Kumar, Zhengping Wang, Matthew Hyder, Joseph Dickinson, Gabrielle Beck, David Adrian, Joshua Mason, Zakir Durumeric, J Alex Halderman, and Michael Bailey. 2018. Tracking Certificate Misissuance in the Wild. In *IEEE Symposium on Security and Privacy (S&P)*.

[48] Adam Langley. [n.d.]. No, don't enable revocation checking. https://www.imperialviolet.org/2014/04/19/revchecking.html.

[49] James Larisch, David Choffnes, Dave Levin, Bruce M Maggs, Alan Mislove, and Christo Wilson. 2017. CRLite: A scalable system for pushing all TLS revocations to all browsers. In *IEEE Symposium on Security and Privacy (S&P)*.

[50] Tobias Lauinger, Abdelberi Chaabane, Ahmet Salih Buyukkayhan, Kaan Onarlioglu, and William Robertson. 2017. Game of Registrars: An Empirical Analysis of Post-Expiration Domain Name Takeovers. In *USENIX Security Symposium (Sec)*.

[51] Tobias Lauinger, Kaan Onarlioglu, Abdelberi Chaabane, William Robertson, and Engin Kirda. 2016. WHOIS Lost in Translation: (Mis) Understanding Domain Name Expiration and Re-Registration. In *ACM Internet Measurement Conference (IMC)*.

[52] Eric Law. 2022. Certificate Revocation in Microsoft Edge. https://textslashplain.com/2022/08/01/certificate-revocation-in-microsoft-edge/.

[53] Chaz Lever, Robert Walls, Yacin Nadji, David Dagon, Patrick McDaniel, and Manos Antonakakis. 2016. Domain-Z: 28 Registrations Later; Measuring the Exploitation of Residual Trust in Domains. In *IEEE Symposium on Security and Privacy (S&P)*.

[54] Jinjin Liang, Jian Jiang, Haixin Duan, Kang Li, Tao Wan, and Jianping Wu. 2014. When HTTPS Meets CDN: A Case of Authentication in Delegated Service. In *IEEE Symposium on Security and Privacy (S&P)*.

[55] Daiping Liu, Shuai Hao, and Haining Wang. 2016. All Your DNS Records Point to Us: Understanding the Security Threats of Dangling DNS Records. In *ACM Conference on Computer and Communications Security (CCS)*.

[56] Suqi Liu, Ian Foster, Stefan Savage, Geoffrey M Voelker, and Lawrence K Saul. 2015. Who is .com? Learning to parse WHOIS records. In *ACM Internet Measurement Conference (IMC)*.

[57] Yabing Liu, Will Tome, Liang Zhang, David Choffnes, Dave Levin, Bruce Maggs, Alan Mislove, Aaron Schulman, and Christo Wilson. 2015. An End-to-end Measurement of Certificate Revocation in the Web's PKI. In *ACM Internet Measurement Conference (IMC)*.

[58] Chaoyi Lu, Baojun Liu, Yiming Zhang, Zhou Li, Fenglu Zhang, Haixin Duan, Ying Liu, Joann Qiongna Chen, Jinjin Liang, Zaifeng Zhang, Shuang Hao, and Min Yang. 2021. From WHOIS to WHOWAS: A Large-Scale Measurement Study of Domain Registration Privacy under the GDPR. In *Network & Distributed System Security Symposium (NDSS)*.

[59] Zane Ma, Joshua Mason, Manos Antonakakis, Zakir Durumeric, and Michael Bailey. 2021. What's in a Name? Exploring CA Certificate Control. In *USENIX Security Symposium (Sec)*.

[60] Najmeh Miramirkhani, Timothy Barron, Michael Ferdman, and Nick Nikiforakis. 2018. Panning for gold.com: Understanding the Dynamics of Domain Dropcatching. In *The Web Conference (WWW)*. 257–266.

[61] Mozilla Wiki. [n.d.]. CA/Revocation Reasons. https://wiki.mozilla.org/CA/Revocation_Reasons.

[62] Mozilla Wiki. [n.d.]. CA:Symantec Issues. https://wiki.mozilla.org/CA:Symantec_Issues.

[63] Mozilla Wiki. [n.d.]. CA/WoSign Issues. https://wiki.mozilla.org/CA/WoSign_Issues.

[64] Daniel Plohmann, Martin Clauss, Steffen Enders, and Elmar Padilla. 2017. Malpedia: a collaborative effort to inventorize the malware landscape. *Proceedings of the Botconf* (2017).

[65] Stefan Santesson, Michael Myers, Rich Ankney, Ambarish Malpani, Slava Galperin, and Dr. Carlisle Adams. 2013. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 6960. https://www.rfc-editor.org/info/rfc6960

[66] Silvia Sebastián and Juan Caballero. 2020. AVclass2: Massive Malware Tag Extraction from AV Labels. In *Annual Computer Security Applications Conference (ACSAC)*.

[67] Y. Sheffer, D. Lopez, O. Gonzalez de Dios, A. Pastor Perales, and T. Fossati. 2020. *Support for Short-Term, Automatically Renewed (STAR) Certificates in the Automated Certificate Management Environment (ACME)*. RFC 8739.

[68] Trevor Smith, Luke Dickinson, and Kent Seamons. 2020. Let's Revoke: Scalable Global Certificate Revocation. In *Network & Distributed System Security Symposium (NDSS)*.

[69] Christian Tiefenau, Emanuel von Zezschwitz, Maximilian Häring, Katharina Krombholz, and Matthew Smith. 2019. A Usability Evaluation of Let's Encrypt and Certbot: Usable Security Done Right. In *ACM Conference on Computer and Communications Security (CCS)*.

[70] Emin Topalovic, Brennan Saeta, Lin-Shung Huang, Collin Jackson, and Dan Boneh. 2012. Towards Short-lived Certificates. In *IEEE Oakland Web 2.0 Security and Privacy (W2SP)*.

[71] Ben Wilson. [n.d.]. Reducing TLS Certificate Lifespans to 398 Days. https://blog.mozilla.org/security/2020/07/09/reducing-tls-certificate-lifespans-to-398-days/.

[72] Ben Wilson. 2021. Policy 2.8: MRSP Issue #235: Require CCADB Disclosure of Full CRLs (or equivalent JSON array) for CRLite. https://groups.google.com/a/mozilla.org/g/dev-security-policy/c/oqpGt9LudnI/m/Aww3jUFkBAAJ.

[73] Kathleen Wilson. [n.d.]. Phasing Out Certificates with SHA-1 based Signature Algorithms . https://blog.mozilla.org/security/2014/09/23/phasing-out-certificates-with-sha-1-based-signature-algorithms/.

[74] Lixia Zhang, Deborah Estrin, Jeffrey Burke, Van Jacobson, James D Thornton, Diana K Smetters, Beichuan Zhang, Gene Tsudik, Dan Massey, Christos Papadopoulos, Tarek Abdelzaher, Lan Wang, Patrick Crowley, and Edmund Yeh. 2010. *Named Data Networking (NDN) Project.* .

A ETHICS

This work does not raise ethical issues. All data used in this study is from publicly available sources.

B CRL DATA

CA Name	CRL coverage (%)
Microsoft	0 / 18 (0%)
Carillon Info Sec	0 / 1 (0%)
Visa	0 / 6 (0%)
TrustFactory	0 / 3 (0%)
TunTrust	1 / 4 (25.00%)
Gov. of Saudia Arabia	5 / 7 (71.43%)
Gov. of Brazil	6 / 8 (75.00%)
DTrust	27 / 31 (87.10%)
GLOBALTRUST	44 / 49 (89.80%)
SECOM Trust Systems	47 / 52 (90.38%)
ANCE (Algeria)	12 / 13 (92.31%)
GDCA	13 / 14 (92.86%)
Firmaprofesional	18 / 19 (94.74%)
AC Camerfirma	37 / 39 (94.87%)
Oiste	26 / 27 (96.30%)
NETLOCK	27 / 28 (96.43%)
GlobalSign	263 / 270 (97.41%)
Telia	19 / 24 (97.98%)
Entrust	64 / 65 (98.46%)
DigiCert	621 / 629 (98.73%)
SSL.com	163 / 164 (99.39%)
Sectigo	839 / 842 (99.64%)
All 70 other CAs	*2731 / 2731 (100%)*
Total Coverage	4963 / 5044 (98.40%)

Table 7: CRL coverage—We download and successfully parse 98% of CRLs from Mozilla's mandatory disclosure list.