The Boon and Bane of Cross-Signing: Shedding Light on a Common Practice in Public Key Infrastructures

Jens Hiller*
hiller@comsys.rwth-aachen.de
RWTH Aachen University

Johanna Amann
johanna@icir.org
ICSI, Corelight, LBNL

Oliver Hohlfeld
hohlfeld@b-tu.de
Brandenburg University of Technology

ABSTRACT

Public Key Infrastructures (PKIs) with their trusted Certificate Authorities (CAs) provide the trust backbone for the Internet: CAs sign certificates which prove the identity of servers, applications, or users. To be trusted by operating systems and browsers, a CA has to undergo lengthy and costly validation processes. Alternatively, trusted CAs can *cross-sign* other CAs to extend their trust to them. In this paper, we systematically analyze the present and past state of cross-signing in the Web PKI. Our dataset (derived from passive TLS monitors *and* public CT logs) encompasses more than 7 years and 225 million certificates with 9.3 billion trust paths. We show benefits and risks of cross-signing. We discuss the difficulty of revoking trusted CA certificates where, worrisome, cross-signing can result in valid trust paths to remain *after* revocation; a problem for non-browser software that often blindly trusts all CA certificates and ignores revocations. However, cross-signing also enables fast bootstrapping of new CAs, e.g., Let's Encrypt, and achieves a non-disruptive user experience by providing backward compatibility. In this paper, we propose new rules and guidance for cross-signing to preserve its positive potential while mitigating its risks.

CCS CONCEPTS

• **Security and privacy → Network security**.

KEYWORDS

PKI; X.509; SSL; TLS; cross-signing; cross certification

ACM Reference Format:
Jens Hiller, Johanna Amann, and Oliver Hohlfeld. 2020. The Boon and Bane of Cross-Signing: Shedding Light on a Common Practice in Public Key Infrastructures. In *2020 ACM SIGSAC Conference on Computer and Communications Security (CCS '20), November 9–13, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3372297.3423345

For an extended version of the paper see [45].

*Parts of the work conducted during an internship at the International Computer Science Institute (ICSI).

CCS '20, November 9–13, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7089-9/20/11…$15.00
https://doi.org/10.1145/3372297.3423345

1 INTRODUCTION

Public key infrastructures (PKIs) like the Web PKI, provide the trust infrastructure for many applications in today's Internet. They, e.g., enable webbrowsers, or apps on mobile operating systems (OS), to authenticate servers for secure online banking, web shopping, or password entry. Governments use PKIs for authentication in privacy-preserving health systems, remote functionality of administrative offices, or electronic voting [3, 32, 90, 92].

Certificate Authorities (CAs) serve as trust anchors in PKIs and have the ability to issue trusted certificates to companies and individuals. The security of a PKI relies on benign and correct acting of *all* its CAs. Despite audit processes, there have been several cases of severe CA misbehavior or security breaches: In 2011, the DigiNotar CA was compromised [78]. This caused its removal from root stores. All DigiNotar issued certificates became untrusted. In the following years, a range of new security measures were introduced to reduce the impact of future compromises [2]. However, most face small deployment and thus have limited effect [2]. Along this path, the *Certification Authority Browser* (CAB) *Forum* gradually increased the requirements that CAs must fulfill to remain in root stores.

Alternatively, trusted CAs can *cross-sign* other CAs to extend their trust to them—thereby mitigating the lengthy and costly validation process that new CAs need to undergo. Cross-signing describes the approach to obtain signatures from several issuers for one certificate[1]. It enables new CAs to quickly establish trust. A prominent example is the bootstrapping of Let's Encrypt, which issued trusted certificates based on a cross-sign of their CA certificates by the already trusted CA IdenTrust while applying for root store inclusion of their own root certificate [42]. Cross-signing also ensures broad validation of certificates in face of divergent root stores of OSes or applications.

However, cross-signing also bears risks: as cross-signs are not systematically tracked [81], cross-signing can challenge proper revocation of certificates in case of CA misbehavior, erroneous operation, or stolen keys. The complexity added by cross-signs already resulted in too broad application of certificate revocation [31, 95]. In this paper, we show that cross-signs also can lead to certificates remaining valid when their CA was distrusted; that the complexity of existing cross-signs makes revocation difficult; that different software and operating systems do not always thoroughly revoke certificates; and that cross-signing makes it difficult to track revocation of CA certificates, especially for non-browser software.

In this paper, we perform the first systematic study of the use and security effects of *cross-signing* (also known as *cross certification*), which is one major reason for missing transparency in PKIs [81].

[1]Technically, cross-signing creates several certificates that share subject and public key as each certificate has exactly one issuer.

For this, we use a passive TLS dataset that contains information about more than seven years of real-world TLS usage, containing more than 225 million certificates derived from more than 300 billion connections—which provides us with insights on the effect of cross-signs on real user connections. For a broad coverage of CA certificates, we combine this private, user-centered dataset with publicly available data from Certificate Transparency (CT) logs.

The main contributions of our paper are as follows:

- We provide a classification of different cross-sign patterns and analyze them with respect to their benefits, but also their risks of unexpected effects on the trust system.
- We systematically analyze the use of cross-signing in PKI systems, with a focus on the Web PKI. Thereby, we reveal problematic cross-signs that render certificate revocation or root store removals ineffective, leading to unwanted valid trust paths. We also find legit use cases, e.g., cross-signing enabled the quick tremendous success of Let's Encrypt, eases the transition to progressive cryptography while maintaining compatibility for legacy applications, and makes a *single* certificate trusted across different applications and operating systems, achieving a non-disruptive user experience.
- We propose new rules and guidance for cross-signing to preserve its positive potential but mitigate enclosed risks.

2 BACKGROUND

This section gives a brief overview of how CAs establish trust, how trust is anchored in root stores, and how certificate revocation is applied today. For a thorough description of PKIs and their fundamental concepts, we refer readers to [28, 47].

Operating systems and some web browsers maintain root stores. They serve as trust anchors when validating certificates: To be valid, a certificate must be issued—directly or indirectly—by a trusted *root certificate*, i.e., a certificate that is included in the root store. Root certificates mostly issue *intermediate certificates* which can issue further intermediate or *leaf certificates*. Figure 1 shows an example with several root (R_i), intermediate (I_i), and leaf (L_i) certificates.

In case of breaches like stolen private keys, certificates have to be revoked. Revocation information is traditionally distributed by CAs using Certificate Revocation Lists (CRLs) [48] or interactively using OCSP [38]. However, these mechanisms mostly remain unused due to (i) the overhead for distribution and (ii) inherent privacy concerns [54, 86]. Applications thus often solely rely on the current state of the operating system's root store.

Consequently, Browsers and operating systems have started shipping vendor-controlled CRLs. Mozilla uses *OneCRL* [39], Google *CRLSets* [80] and a blacklist [79]; Microsoft and Apple include information on blocked CA certificates in their root stores [51, 60].

3 CROSS-SIGNING

We next introduce cross-signing and provide a classification of the different cross-signing patterns used in the remainder of the paper.

3.1 Cross-signing: Definition

We illustrate cross-signing with a typical use-case: To issue trusted certificates, a CA must be included in the respective root stores of web browsers, operating systems, etc. Inclusion in these root stores

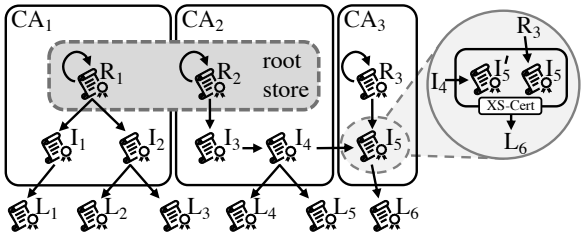

Figure 1: An example PKI including a cross-sign ($I_5 + I_5'$).

requires time demanding audit and certification processes. During the process of being included in root stores, a CA may already want to issue trusted certificates. To this end, another CA_{trusted}, whose certificate is already included in root stores, *cross-signs* the CA's root or intermediate certificate to create a trust path that ends in the already trusted root certificate of CA_{trusted}. In Figure 1, the intermediate certificate I_5 is cross-signed by I_4, providing a trust path to R_2. As real-world example, Let's Encrypt has been using an intermediate certificate cross-signed by IdenTrust to already issue certificates while waiting for root store inclusion of its own root certificate [42]. Similarly, CAs that are included in only some root stores can use cross-signing to extend trust to further root stores.

CAs typically call this *cross-signing* (also *cross-certification*) and the resulting certificates *cross-certificates* [42, 43]. Analogously, RFC 5280 defines a cross-certificate as a CA certificate that has different *entities* as issuer and subject [6]. In this paper, we use a broader definition: (i) To analyze cross-signing for *all* certificate types, i.e., root, intermediate, and leaf certificates, we consider all certificates, not only CA certificates. (ii) To also track effects of signing a certificate with multiple CA certificates of the *same* entity, we only require signatures by two different CA certificates, but not that issuer and subject are controlled by *different* entities.

Specifically, our definition is as follows (cf. Figure 1). To *cross-sign* a certificate (here: I_5) that was originally issued by R_3, a CA certificate (here: I_4) creates and signs a copy I_5' which has the same subject and public key as I_5. This process is necessary as each certificate has exactly one issuer field [6], i.e., $issuer(I_5) = R_3 \neq issuer(I_5') = I_4$. Thus, a cross-sign is a certificate for which another certificate exists that has the same subject and public key, but a different issuer and signature. These certificates form a *cross-sign certificate* (XS-Cert). The certificates of a XS-Cert can be used interchangeably: If I_5 and I_5' are CA certificates, a certificate issued by I_5 will also validate using I_5'. In more detail: when a certificate is validated, the validating software searches a CA certificate whose subject equals the issuer of the current certificate. It then checks that the signature of the current certificate validates against the public key of the CA certificate. As all certificates of a XS-Cert share subject and public key, I_5 and I_5' can be used interchangeably.

Note that there is a difference between cross-signing and certificate *re-issuances without re-keying*. When a certificate reaches the end of its validity, it often is replaced by a certificate with the same subject, key, and a new validity period. We need to distinguish these cases from cross-signing. Telling them apart is complicated since cross-signs often (and legitimately) do not have the exact same validity periods as the original certificate, e.g., because the

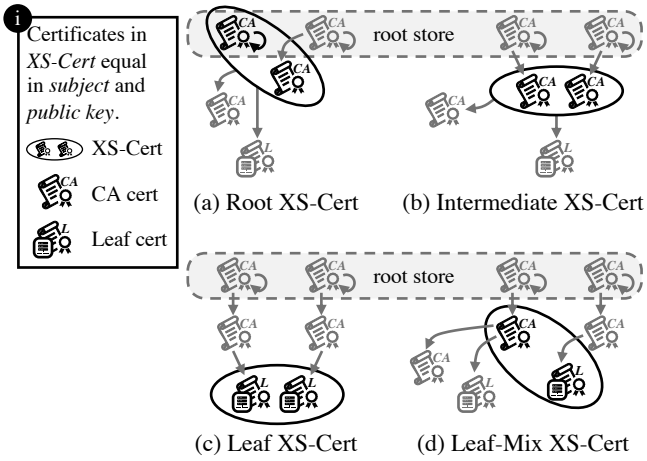

Figure 2: Cross-signing types. We observe many root, intermediate and leaf XS-Certs in our dataset. Luckily, leaf-mix XS-Certs remain a theory: We find no such problematic case.

not before field should correspond to the issuance date [68]. As, e.g., GoDaddy allows for a renewal of certificates up to 120 days prior to expiration, we require certificates to have an overlapping validity period of at least 121 days to be considered cross-signs.

3.2 Cross-signing: Classification

For our analysis of cross-signing and corresponding risks for the PKI, we classify XS-Certs into 4 types. These types depend on the type of certificates in the XS-Cert. We illustrate this in Figure 2.

Root XS-Certs. A root XS-Cert comprises at least two CA certificates of different issuers. At least one of them is part of a root store (cf. Figure 2a). When CAs use the term cross-signing, they typically refer to this type. This approach extends trust to root stores that do not include a CA's root certificate and can extend a CA's trust to more applications or operating systems. Likewise, an already trusted root can bootstrap trust in a new CA certificate by cross-signing it. Moreover, CAs can start to use a new CA certificate without disrupting compatibility with old applications that do not receive root store updates. However, as we will discuss later, such cross-signs increase the complexity of removing trust in certificates that are part of a root store. Trust paths can inadvertently remain valid via paths using other certificates of the XS-Cert (Section 5.1).

Intermediate XS-Certs. Intermediate XS-Certs contain two or more intermediate CA certificates from different issuers, but no root store certificate (cf. Figure 2b). They are similar to root XS-Certs, but no certificate in the XS-Cert is directly trusted by root store maintainers. The same benefits (bootstrapping, large trust coverage, and compatibility with old applications) apply. Additionally, CAs without an own root certificate may employ such cross-signing to root trust in multiple CAs, and thus achieve independence from a single business partner. However, intermediate XS-Certs also are problematic if certificate revocations are not thoroughly applied to all certificates of a XS-Cert (cf. Section 5.1).

Leaf XS-Certs. Leaf XS-Certs contain only leaf certificates (cf. Figure 2c). This could in principle be used in cases where some client applications only trust divergent CAs; Here, the server can send *that* certificate of which it knows that it will be trusted by the client, e.g., using the user agent data in an HTTP request to determine the client's root store. As leaf certificates cannot issue certificates, problematic cases are limited in scope. Still, for a revocation in case of a security incident, all certificates of the XS-Cert have to be revoked – otherwise insecure certificates remain valid.

Leaf-Mix XS-Certs (Theoretical). In theory, a XS-Cert could contain leaf certificates and (root or intermediate) CA certificates (cf. Figure 2d). Private keys of CA certificates have special protection requirements such as the use of sealed hardware. As private keys of leaf certificates are typically present on comparably vulnerable end-systems, a leaf-mix XS-Cert would put a key that can issue certificates at high risk. Fortunately, our dataset does not contain any valid leaf-mix XS-Certs.

4 DATASETS AND METHODOLOGY

The analysis in this paper is based on a dataset of certificates used in the wild. The certificates are passively collected from the outgoing SSL/TLS connections on all ports of several universities and research networks mainly located in North America. Beyond public data (e.g., CT Logs), it also contains certificates that are not publicly available (e.g., user or private certificates) and thereby enables us to take a broad yet unseen perspective on cross-signing. Our dataset spans a period of more than 7 years, starting February 2012 and ending August 2019, covering more than 300 billion TLS connections. For a broad view on CA certificates, we extend this dataset with CA certificates from CT logs[2] (those logs used by crt.sh [58]).

The passive data collection effort was cleared by the respective responsible party at every contributing institution. Our collection effort focuses on machine to machine communication and excludes or anonymizes sensitive information. See Appendix A for details.

We release the toolchain [46] that enables the analysis of the cross-signing relationships of X.509 certificates from any source (e.g., Censys [33] or CT logs [58]). Since that part of our data set derived from passive measurements is subject to NDAs and contains private certificates that are not contained in public repositories, we are unable to release our data set. Our toolchain can, however, be used to reproduce our analysis using other data sources, e.g., CT.

We validate certificates using our custom validation logic which closely mirrors the way browsers perform certificate validation. We build potential paths using name-matching between subject and issuer fields. Unlike browsers, we build all possible paths from a certificate to root certificates[3]. We do not just search for the shortest possible path. To this end, we check each path separately by passing the complete path to OpenSSL for validation and using only the specific root certificate for this path as trust anchor. We obtain the earliest and latest possible validity dates based on the *not before* and *not after* dates of the path's certificates and use these for the validation. We also check if a path is valid given the path-length-constraints of its CA certificates. To obtain validity information for

[2]These contain 156,315 CA certificates, however, 147,439 of them are used for *Google Certificate Transparency (Precert Signing)* which we exclude from our analysis. Further 2,635 are already included in our passive dataset. CT thus adds 6,241 CA certificates.
[3]Due to computational complexity, we base our evaluation for some CT certificates on all paths of length 12 or smaller (multiple million paths per certificate); for the vast majority of certificates, also longer paths were validated and used.

a certificate, all found paths and validity period(s) of the certificate are mapped to the corresponding root store versions—based on the trust status of a path's root certificate in root stores over time.

We use the root stores of major operating systems, web browsers, and grid networks and consider certificate additions as well as removals across their versions. Specifically, we use the root stores of Microsoft [30], Apple (iOS, OSX) [50], Google (Android) [40], Mozilla (Firefox, Linux distributions) [63], and the grid computing PKI [36]. We also include the PKIs of the governments of the United States [41], Australia [70], Switzerland [87], Oman [73], Netherlands [74], Japan [72], India [71], and Estonia [5].

We test for revocations based on browser revocation lists, i.e., Mozilla's OneCRL, Google's CRLSets and blacklist, Apple's lists of blocked certificates, and Microsoft's root store program. We also use revocation status information from crt.sh for CA CRLs and Microsoft's *disallowedcert.stl* blacklist to cross-check with the former web browser or operating system revocation mechanisms.

Our dataset includes 225,243,355 certificates of which 23,504,394 are valid[4] (720 only due to inclusion in a root store; 23,503,674 have a valid path to a root store). We focus our analysis on these valid certificates. They split into 9,197 CA certificates, 23 self-signed X509v1 certificates, 3 self-signed X509v3 certificates using legacy extensions to show their CA status, and 23,495,171 leaf certificates. 793 certificates are included in at least one of our used root stores (all 694 self-signed CA certificates, the 23 self-signed X509v1 and 3 X509v3 legacy certificates, and 73 not self-signed CA certificates; All the latter 73 certificates are already valid due to their root store inclusion but additionally provide paths to other trusted roots).

To give a short overview of our dataset: We see 526 of the 793 root certificates being used—meaning that we found another certificate that was signed by them in our dataset. In total, we find more than 9.3 billion valid paths between the certificates in our dataset. The longest path is 17 certificates long (including root and leaf). We find 63 certificates with this path length. In the entirety of our measurement, the root CA against which we can validate the most certificates was the IdenTrust *DST Root CA X3* (with 8,473,760 certificates). This is probably driven by their cross-sign of Let's Encrypt which we will detail later in this paper. The Let's Encrypt *ISRG Root X1* follows as a close second with 8,382,825 certificates. The CA to which we can find most unique paths is the Comodo *AddTrust External CA Root* CA. For this root, we can find 3.7 billion different validation paths for a total of 5,966,846 distinct certificates. The maximum number of different validation paths (as in different paths through the CA ecosystem through which we could validate it) for a CA certificate we found was 45,225,135. These paths provide validity for the intermediate *IdenTrust Global Common Root CA 1*.

Taking a look at cross-signs, we find a total of 47,543 XS-Certs. These split into 86 root, 236 intermediate, and 47,221 leaf XS-Certs[5]. For the remainder of this paper, we focus our analysis on root and intermediate CA certificates as these cross-signs impact the trust of thousands or millions of leaf-certificates. We however note that it is interesting that there is a significant number of leaf XS-Certs which may bring along the same problems as cross-signed CA certificates, e.g., incomplete revocation. We note that CRLite [55] (now being integrated in Firefox) potentially enables large scale revocation for leaf certificates—by reducing the overhead for CRL updates with efficient Bloom filter-based incremental updates.

No clear phenomenon explains the leaf XS-Certs: For 63% of leaf XS-Certs, one CA issued all certificates (multiple CAs: 37%). DigiCert is involved in 67% of leaf XS-Certs (45% of those stem from Symantec before DigiCert acquired it), followed by GoDaddy (10%) and Comodo (8%). Let's Encrypt, however, is involved in only significantly less than 1% of cases. In future work it might be worth trying to explore the motivation behind these cases.

When analyzing the root and intermediate XS-Certs, we find *internal* and *external cross-signs*. Internal XS-Certs comprise only certificates that have been issued within the same CA group, i.e., issuer and certificate owner are controlled by the same entity. For example, we find the two largest CA groups, Comodo[6] and DigiCert, to internally cross-sign certificates within and across their subsidiary CAs, both considered internal cross-signing.

External XS-Certs cross boundaries of CA groups, i.e., issuer and certificate owner of at least one certificate of the XS-Cert are controlled by different entities. These are particularly interesting as issuing CAs take responsibility for the actions of the resulting intermediates [64]; whereas the information flow between organizations may be limited, challenging fast and thorough revocation.

We find internal and external cross-signing for both, root (58 internal / 28 external) and intermediate (70 / 166) XS-Certs. For intermediate CA certificates, we find 124 cases where the organization does not have their own root certificate. These organizations are completely dependent on the cross-signs from their issuers.

In addition to these XS-Certs, we also identify 97,240 cases of *certificate re-issuances*. Certificate re-issuances are similar to cross-signing: there are several certificates using the same subject and key. The difference to XS-Certs is that the validity periods of these certificates overlap only slightly or not at all (note that key-reuse across certificates with *different* subjects clearly distinguishes from cross-signing). We only consider cases as XS-Certs if their certificate's validities overlap for 121 or more days (cf. Section 3.1) and exclude re-issuances from our following discussion. Notably, certificate re-issuing is prevalent for leaf certificates (97,233) and almost non-existent when root (1) or intermediate (6) certificates are involved. We still analyzed the latter, but found them uninteresting.

5 CROSS-SIGNS IN THE WILD

In this section, we systematically analyze the use of cross-signing in existing PKIs. To identify the different categories of cross-signing and their motivation, we start off from its primary goal: extending certificate trust—exploiting all derived trust paths in our dataset. We combine the derived validity status in root stores (over time) with revocation data and enrich this with knowledge on ownership and (governmental-)control over CAs, and algorithmic properties of certificates. We overview the identified categories in Table 1 with their number of occurrence in our dataset(s). We focus on (i) the

[4] 147,439 invalid certificates are for *Google Certificate Transparency Precert Signing* (excluded from analysis). Most other invalid certificates are a side effect of observing real Internet traffic: 70% are used for user authentication in the grid and not trusted by CAs. Auto-generated certificates for Tor or WebRTC account for 17% and 1%, respectively. 9% are self-signed. The remaining 3% comprise IoT and firewall certificates. Earlier studies already found invalid certificates to be common in the web [26].

[5] The number of leaf XS-Certs is solely based on our passive dataset.

[6] By *Comodo*, we refer to the CA business, which was recently renamed to *Sectigo* to avoid naming confusion with the separate company *Comodo Group* [29]

Table 1: Observed XS-Certs by category. Numbers marked with ∗ are based on manual investigation of the passive dataset. A XS-Cert can count to multiple categories.

⚠	Valid after revocation	16	∗
⚠	PKI barrier breaches	7	∗
✓	Bootstrapping	57	
+/−	Expanded trust (new stores / longer time)	64 / 46	
+/−	Alternative paths	155	
+/−	Support of multiple signature algorithms	23	
+/−	Ownership change	35	∗
🐞	Backdating	7	∗
🐞	Missing transparency	2	∗

bad ⚠: potential and exploited security problems (Sections 5.1 - 5.2), (ii) the good ✓: ease bootstrapping of new CAs (Section 5.3), (iii) cases with pros and cons [+/−], e.g., the transition to new cryptographic algorithms (Sections 5.4 - 5.6), and (iv) the ugly 🐞 behavior due to problematic practices and lacking transparency (Section 5.7).

5.1 Valid Paths After Revocations

Cross-signs significantly complicate the revocation of CA certificates. When a CA certificate is revoked, *all* of its cross-signs need to be revoked as well. A single remaining unrevoked cross-sign means that a valid path to a trusted root still exists—and all certificates of the revoked CA can still be validated. As we will show in this section, cross-signing repeatedly caused incomplete revocations, or even added new valid trust paths for already revoked certificates. We next empirically study this problem of *valid paths after revocation* that adds an additional layer of complexity to the already fragile revocation mechanisms in PKIs. We focus our discussion on cross-signs that caused incomplete revocations for WoSign and DigiNotar as well as incomplete revocations in vendor-controlled CRLs (especially Mozillas' OneCRL and Googles' CRLSet).

5.1.1 The WoSign and StartCom Ban. WoSign was distrusted by major root store holders after a series of misbehaviors [65] (see Figure 3 for a timeline of events). WoSign did not announce the acquisition of the StartCom CA in time. It also evaded rules for distrusting SHA1-signed certificates issued after January 1st 2016 by backdating the *not before* date[7] of certificates they issued. As a result, Mozilla set up a special *not before rule*, i.e., for certificates that were issued after October 21, 2016, Mozilla distrusted paths that end in a WoSign or StartCom root [97]. In January 2018, Mozilla completely removed these roots [97, 98]. Google performed similar actions, removing WoSign and StartCom roots around September 2017 [93, 94], as did Apple and Microsoft [4, 89].

By analyzing cross-signs, we observe that WoSign certificates were cross-signed by Comodo subsidiaries, Certplus, Unizeto Certum, and StartCom. Furthermore, in April 2017—after Google and Mozilla set up *not before* rules for certificates issued by WoSign and StartCom—the widely trusted *Certinomis - Root CA* cross-signed the *StartCom EV SSL ICA* (see Figure 3). The *StartCom EV SSL ICA* intermediate was issued a few days before by *StartCom Certification*

[7]The certificate's not before date defines from which time on the certificate is valid, but it is also used as an indicator for the issuance date of the certificate [68].

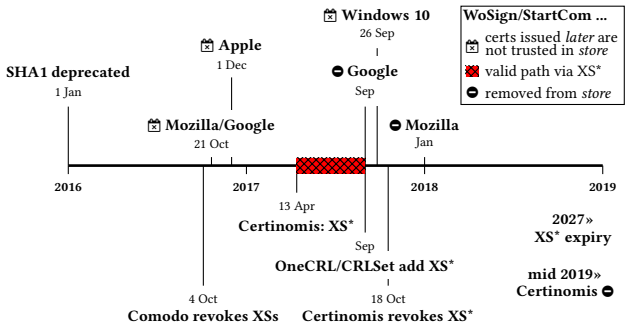

Figure 3: Certinomis' cross-sign of WoSign/StartCom (XS*) bypassed *not before* rules (📅) set up by major root stores.

Authority G3 and thus was affected by the *not before* rules set up by Google and Mozilla, rendering it unable to issue trusted certificates. However, the new Certinomis cross-sign bypassed the *not before* rules as it established a trust path to a root that was not operated by WoSign or StartCom, enabling StartCom to issue valid certificates despite its ban. After discovery, Mozilla and Google revoked the cross-sign. Mozilla added it to OneCRL in September 2017 [22, 67]. One month later, Certinomis also added the certificate to its CRL. In the end, this incident added to a list of issues that led to the distrust of Certinomis in mid 2019. Overall, the cross-sign provided undesired valid trust paths for about 6 months. In our dataset, this affected 11 certificates. We saw a subset of them in a small amount of connections during this 6 month period. This result highlights the complexity of revocation when cross-signing is involved. A cross-signing CA not only has to keep track of the cross-sign (as for any issued intermediate), but must also carefully examine and track actions applied to the cross-signed certificate.

The other cross-signs of WoSign/StartCom did not result in any undesired trust paths. We briefly discuss them to further stress the complexity of CA revocations given cross-signs. WoSign received several cross-signs before its ban. Keynectic's *Certplus Class 1 Primary CA*, which was part of the Microsoft root store, cross-signed three WoSign CA certificates. These cross-signs were never revoked. This only did not cause any undesired trust paths because Microsoft set up a *not before November 16, 2016* rule for the Certplus root in September 2017 and disabled the Certplus root in May 2018.

Mozilla and Google, however, could have been more thorough when revoking WoSign and StartCom. WoSign's roots *Certification Authority of WoSign* and *Certification Authority of WoSign G2* set up four (internal) intermediate XS-Certs: *WoSign Class 3 OV Server CA G2*, *WoSign Class 4 EV Server CA G2*, *WoSign Class 4 EV Pro Server CA G2*, and *WoSign Class 3 OV Pro Server CA G2*. Despite these cross-sign's expiry not before November 2029, none of them was revoked in OneCRL or CRLSet when WoSign was distrusted. While our data shows that this was not necessary in this case as corresponding roots are revoked, an explicit revocation can prevent undesirable trust paths—e.g. if there was another unknown cross-sign, or if WoSign was able to obtain new cross-signs for its intermediates.

Comodo explicitly revoked its cross-signs when Mozilla started to distrust WoSign in 2016. Specifically, cross-signs of the *Certification Authority of WoSign* by Comodo's *UTN-USERFirst-Object* and

UTN-DataCorp SGC were revoked. Similarly, *Unizeto Certum CA* revoked its cross-sign of *Certificate Authority of WoSign G2* in 2016.

We also find that StartCom cross-signed WoSign certificates years before being acquired by them, going back to at least 2006 (details in Appendix B.1). This shows a close business relationship between them years before the (not timely reported) acquisition of StartCom by WoSign in 2015. This cross-signing behavior also continued after the acquisition. In this regard, the analysis of cross-signs can provide a rich source of information for security research.

5.1.2 Incomplete DigiNotar Revocation. After a security incident at DigiNotar in 2011, an attacker was able to obtain certificates for several domains, including Google and Mozilla's add-on domains. As a result, Mozilla distrusted all DigiNotar root certificates [69].

Our data shows that *DigiNotar Root CA* was cross-signed by *Entrust.net Secure Server CA* in 2007, creating an intermediate which was valid up to August 2013. Both, root and cross-sign, were revoked by Microsoft, Google, and Mozilla in 2011: Microsoft removed the root from its root store and blocked the intermediate. Likewise, Google added both certificates to Chrome's blacklist disabling all certificates that use the corresponding public key[8], including existing and possible future cross-signs. Mozilla, as OneCRL was not introduced before 2015 [39], implemented a special revocation mechanism to disable the root cert. However, applications that base their trust on the certificates in the root store of Mozilla, Google, or Microsoft, often do not support their special revocation mechanisms. As the cross-signing root *Entrust.net Secure Server CA* remained in root stores until 2015 (Mozilla, Google) or longer (Microsoft, Apple), such applications still established a valid path until expiry of the DigiNotar cross-sign in August 2013. Hence, applications were at risk to accept certificates issued by *DigiNotar Root CA* up to two years after the distrust of DigiNotar roots because of the cross-sign.

5.1.3 Incomplete Vendor-Controlled CRLs. Most applications do not perform certificate revocation checks—especially not for CA certificates (see Section 2). Instead, many browsers and operating systems ship their own revocation lists, which they have to keep updated. The security of the web PKI thus also depends on how consistently cross-signed certificates are revoked. In this section, we examine these lists and show that cross-signs are *not* consistently revoked. This further highlights the complexities of cross-signs.

In 2011, the *Actalis Authentication Root CA* created the intermediate *Actalis Authentication CA G2*. A cross-sign by *Baltimore CyberTrust Root* enabled broad trust. The *Actalis Authentication CA G2* was later revoked by Actalis with "cessation" given as the reason in November 2016. The same is true for the cross-sign by CyberTrust. OneCRL included these revocations, but Google lists only the Actalis intermediate in its CRLSet. Hence, the cross-sign still works for devices using Google's root store—like Android phones. This affected 13 certificates we saw in traffic; they were recognized as valid for two years until expiry of the cross-sign.

Another revocation inconsistency affects the internal intermediate XS-Cert *GlobalSign Extended Validation CA - SHA256 - G2*. The first certificate issued by *GlobalSign Root CA - R2* was revoked in September 2019 due to cessation of operation, but GlobalSign did not revoke the cross-sign by *GlobalSign Root CA - R3*. OneCRL (Mozilla) and CRLSet (Google) mimic this inconsistency [23]. Lacking any explanation, we believe that the cross-sign should have been revoked, too. In this case, we did not find any non-expired certificates affected by this.

The intermediate XS-Cert *Entrust Certification Authority - L1E* shows a similar revocation inconsistency: While *Entrust Root Certification Authority* revoked the corresponding intermediates in July 2018, the cross-sign by *Entrust.net Certification Authority (2048)* was not revoked before February 2019, i.e., seven months later. Google's CRLSet even missed revoking one of the earlier CRL-revoked cross-sign. Although no severe security problem evolved (the intermediate was intentionally superseded and all issued certificates expired before the initial revocation), both cases show that already cross-signing within a CA can inadvertently prolong the validity of certificates, let alone cross-signing across CAs.

Three intermediate XS-Certs of the US Federal PKI show inconsistent revocation states. For all three (*DoD Interoperability Root CA 2*, *NASA Operational CA*, and *DHS CA4*), one intermediate was revoked via CRL, marking them as superseded. However, corresponding cross-sings were not revoked; without an obvious explanation. Vendor-controlled CRLs did not even add the revoked certificates. A total of 10, 5,725 and 4 certificates were affected, respectively.

Finally, special cross-signing-related requests of CAs can further complicate the maintenance of vendor-controlled CRLs. Specifically, the root XS-Cert *Belgium Root CA2* was cross-signed by *GlobalSign Root CA* and *CyberTrust Global Root*. While the former cross-sign expired in 2014, Belgium requested an inclusion of the CyberTrust-issued cross-sign in OneCRL in October 2017 because the intermediate is no longer used to issue TLS certificates [18]. This request created a very narrow-band revocation as it affects only applications that use OneCRL, i.e., mostly Firefox: First, neither Google nor Microsoft block the cross-sign in their vendor-controlled CRLs. Second, the root is still included in Apple's root store today (and was never included in any other) until its expiry in 2021. Third, the cross-sign is not included in the CA's CRL, probably as CRLs lack a mechanism to revoke certificates for specific key usages (notably, the only allowed key usage for the root are *Certificate Sign* and *CRL Sign*). On the positive side, we can not provide evidence for a real world relevance of this inconsistency: All observed issued certificates already expired before the (incomplete) revocation.

Takeaway: *Cross-Signing resulted in incomplete revocations or even added new valid trust paths for already revoked certificates. Thus, coping with cross-signing consequences adds a further burden to the important but fragile revocation mechanisms in PKIs. Consequently, we propose adapted mechanisms that ease the revocation in face of cross-signing. Furthermore, CAs should be required to publicly disclose and explain inconsistencies even if they serve a benign purpose.*

5.2 PKI Barrier Breaches

Next, we show how cross-signing can enable a valid trust path between otherwise isolated PKIs—a potentially undesired effect. We discuss how cross-signs intentionally or inadvertently cause breaches of such PKI boundaries, e.g., by making state-controlled PKI systems trusted by the Web PKI. We show this effect on two examples of the Federal PKI and the Swiss Government PKI.

[8]Technically, Chrome's blacklist lists hashes of the certificate's *SubjectPublicKeyInfo* (SPKI), i.e., the encoded public key.

5.2.1 Undesired Global Trust in the FPKI. The Federal PKI (FPKI) provides PKI functionality for US government services. While the FPKI was part of the Apple and Microsoft root programs till 2018, it was never accepted by Mozilla [8, 66]. However, we find several cross-signs of FPKI CAs by widely trusted Web-PKI CAs. As the FPKI extensively uses internal cross-signs among its CA certificates, this trust in single CA certificates expanded to a large part of the FPKI. As a result, many applications that use the Mozilla root store unknowingly trusted certificates issued by the FPKI, even though it did never fulfill the necessary root store policies [8, 66]. As the FPKI applied for trust by Mozilla since 2009 [8] (aborted in 2018), the cross-signs could be seen as attempt by the FPKI to *bootstrap* (cf. Section 5.3) trust in Mozilla's rootstore in parallel. However, at the time of cross-signing, the public discussion on the FKPI's application for trust already called out concerns regarding the eligibility of affected certificates [8]. These concerns should also have prevented the cross-signs. This is especially worrisome since state-controlled PKI systems can suffer from potential political influence [85].

We start our analysis with the intermediate XS-Cert *Federal Bridge CA 2013*. This intermediate was issued in 2013 by the *Federal Bridge CA* and two years later cross-signed by the *Federal Common Policy CA*. More importantly, also in 2015, it was cross-signed by widely trusted CA certificates of IdenTrust and VeriSign: First, it was cross-signed by the intermediate *IdenTrust ACES CA 1* which roots back to *DST ACES CA X6*. Second, VeriSign cross-signed the FPKI CA with its *VeriSign Class 3 SSP Intermediate CA - G2*, an intermediate issued by VeriSigns internal XS-Cert *VeriSign Universal Root*. These cross-signs provided the FPKI with broad trust coverage until the cross-signs expired or were revoked. VeriSigns cross-sign expired in July 2016, shortly before the issuing *VeriSign Class 3 SSP Intermediate CA - G2* was revoked by CA and vendor-specific CRLs in 2017. IdenTrust revoked its cross-sign in February 2016, but OneCRL did not inherit the CRL entry before November 2017. Until revocation in 2015, a further trust path was provided via a cross-sign by the *Federal Bridge CA* which was itself issued by the already aforementioned *DST ACES CA X6* [8].

The thus trusted *Federal Bridge CA* and *Federal Bridge CA 2013* distributed the trust further by cross-signing the *FPKI Federal Common Policy* root. Between 2008 and 2011, a further trust path to *DST ACES CA X6* for this FPKI certificate was established by a cross-sign from the root XS-Cert *FBCA Common Policy*. *FBCA Common Policy* offers this path to *DST ACES CA X6* via a cross-sign by *FBCA Entrust*; additionally it is included in recent Apple root stores and trusted by Microsoft (between 2012 and 2017). These trust paths are also available for *Federal Bridge CA 2013* which was (mutually) cross-signed by *FPKI Federal Common Policy*. Further cross-signs by *Federal Bridge CA 2016* and *U.S Department of State AD Root* do not provide new trust paths, however, as these have in return been signed by *FPKI Federal Common Policy*, too, the broad trust also expands to these intermediates.

Further cross-signs by *FPKI Federal Common Policy* expand the IdenTrust and VeriSign trust path deeper in the FPKI: It cross-signed the intermediate *SHA-1 Federal Root CA*; as likewise done by *FBCA Common Policy* (providing an alternative path to *DST ACES CA X6*) and, again, *VeriSign Class 3 SSP Intermediate CA - G2*. *SHA-1 Federal Root CA* further signed the *DoD Interoperability Root CA 1* which repeatedly cross-signed the XS-Cert *DoD Root CA2*. However, in contrast to the former XS-Certs, *name constraint* extensions limit these cross-signs to issue for the U.S. Government only.

5.2.2 Cross-Signs of the Swiss Government. In 2016, the Swiss Government created the *Swiss Government Public Trust Standard CA 02* intermediate which was cross-signed by *QuoVadis Enterprise Trust CA 2 G3* in 2017. Positively, QuoVadis used *X509v3 Name Constraints* to white-list domains for which these cross-signs are allowed to issue certificates. However, QuoVadis did not set the *critical* flag for this extension, i.e., implementations are allowed to ignore it. Consequently, the cross-signs could yield undesired trust paths for software that does not implement X509v3 Name Constraints. This especially could affect applications that derive trusted roots from root stores of Mozilla, Google, or Apple as these do not establish a valid path for the original intermediate (contrarily to Microsoft's root store). QuoVadis used its CRL to revoke the cross-signs in mid 2019, but only CRLSet adopted this revocation. The Swiss Government did not misuse this opportunity: All 1,039 certificates observed in our measurements are part of the white-listed domains.

Contrarily, *Baltimore CyberTrust Root* did not set up name constraints when it cross-signed *Swiss Government SSL CA 01* (an intermediate issued in 2014 by *Swiss Government Root CA II* which is included in Microsoft's root store since 2016). Until its expiry in 2017, this cross-sign allowed the Swiss CA to issue certificates with valid trust paths to all major root stores. In total, 756 certificates validate under these circumstances; 9 of them are not part of the white-listed domains, but still part of the Swiss top-level domain.

We describe three more cross-signs of state-controlled CAs when discussing DigiCert in the light of ownership changes (Section 5.6.2).

Takeaway: *Due to the extensive cross-signing in the FPKI, only few trust anchors to the Web-PKI added many new trust paths. This highlights the need for mechanisms that provide CAs with better insight on the effect on trust paths before they cross-sign a certificate. Furthermore, enforcing short validity periods for intermediates could limit the impact of unexpected trust paths.*

5.3 The Good: Bootstrapping of new CAs

In contrast to the previous cases which show security problems of cross-signing, we now focus on the benefits of cross-signing. Especially for new CAs, inclusion into root stores is a lengthy process. Obtaining a cross-sign from a broadly trusted root or intermediate enables a CA to start its business while pursuing the process to include certificates into root stores. In our dataset, we can identify such bootstrapping help for Let's Encrypt, the China Internet Network Information Center (CNNIC), and GoDaddy. Furthermore, CyberTrust bootstrapped trust in *Actalis Authentication CA G2* when Actalis' root had not yet been trusted (cf. Section 5.1.3). We provide a full list of involved CAs, e.g., COMODO, Digicert, WoSign, Globalsign, AffirmTrust, and government CAs, in Appendix B.2.

5.3.1 Bootstrapping Let's Encrypt. Let's Encrypt launched as nonprofit Certificate Authority in 2015. It significantly increased the amount of secured Internet communication by automating the certificate issuing process and providing certificates for free.

However, to provide this service early on, it had to depend on a cross-sign by a widely trusted root for more than 5 years. Specifically, Let's Encrypt will first start to default to its own ISRG Root

starting September 29, 2020 [43]. Before, IdenTrust helped to bootstrap trust for Let's Encrypt. To this end, its *DST Root CA X3* cross-signed four intermediates originally issued by the *ISRG Root X1*, a root of the Internet Security Research Group (ISRG) which manages Let's Encrypt. As *ISRG Root X1* was initially not included in root stores, IdenTrust was the sole trust anchor for Let's Encrypt certificates enabling a fast ramp up of the service [42].

Looking at the details, Let's Encrypt uses four intermediate XS-Certs. *Let's Encrypt Authority X1* and *X2* are no more actively used [42] (but not revoked and still valid until the end of 2020). In contrast to newer cross-signs, IdenTrust prevents the X1 and X2 cross-signs from issuing certificates for the top-level domain *.mil*. The more recent *Let's Encrypt Authority X3* is currently used by Let's Encrypt to issue (leaf) certificates and remains valid until 2021-10 (original) and 2021-03 (cross-sign). Finally, *Let's Encrypt Authority X4* serves as backup, but will expire at the same time as X3.

Even after the switch to the own ISRG roots, the cross-signs will be beneficial: Legacy clients that do not include the ISRG roots in their root store can fall back to the IdenTrust trust path [42].

5.3.2 Entrust Helped CNNIC. Similar to Let's Encrypt, the Chinese CNNIC obtained bootstrapping help from an established CA; *Entrust.net Secure Server Certification Authority* issued a *CNNIC SSL* intermediate in 2007. Shortly after the inclusion of *CNNIC ROOT* into root stores (e.g. 2009 for Mozilla [9]), the *CNNIC ROOT* cross-signed this intermediate, creating an alternate trust path.

5.3.3 GoDaddy – Internal Bootstrapping via Subsidiary. Similarly, GoDaddy used cross-signing to bootstrap trust into its CA certificates [7] when it entered the certificate business in 2004. In contrast to the previous cases, however, it used an *internal* cross-sign. Particularly, GoDaddy started off with the root certificates *GoDaddy Class 2 CA* and *Starfield Class 2 CA* and cross-signed them with ValiCert's root *ValiCert Class 2 Policy Validation CA* (created 1999). Interestingly, GoDaddy acquired ValiCert just the year before, probably already preparing for its new business. Until its removal from root stores around 2014, the ValiCert root thus bootstrapped trust to Mozilla, Android, and Apple for the new certificates.

GoDaddy later bootstrapped a root for Amazon: When Amazon created its *Amazon Root CA 1* in 2015, an immediate cross-sign by *Starfield Services Root CA - G2* established trust right away whereas it took years for Amazon's root to arrive in root stores. E.g., Mozilla and Google included it in 2017 and Apple even later in 2018.

Takeaway: *Cross-Signing enables new CAs to already start their business during the process of including their roots into root stores. Without such a cross-sign, the long periods for inclusion and sufficient propagation of root stores of several years could be prohibitive for new companies in this business.*

5.4 Expanding Trust and Alternative Paths

Any cross-sign either (i) *expands the trust to more root stores*, i.e., provides trust paths to not yet covered root stores, (ii) *extends the validity period* in covered root stores, or (iii) *adds alternative paths* to root stores already covered by other certificates in the XS-Cert. A XS-Cert often causes several of (i)–(iii); Table 1 only counts them to their primary case (i > ii > iii). We briefly describe some XS-Certs in the following and refer to Appendix B.3 for further examples.

5.4.1 Expanding Trust. Some root certificates are not included in all major root stores. We find that CAs broadly use cross-signs to close these holes in the root-store coverage, enhancing their trust. E.g., a cross-sign provides *Entrust Root Certification Authority - G2* with trust paths for Mozilla. Similarly, USERTrust cross-signed *AddTrust Qualified CA Root* and *AddTrust Class 1 CA Root* to provide trust for Microsoft and the grid PKI. On the downside, as aforementioned, these additional trust paths complicate certificate revocations. Bootstrapping (cf. Section 5.3) is a special case of this trust-expanding cross-signing. The count in Table 1 excludes bootstrapping cases. Furthermore, we distinguish cases that extend only the validity period. E.g., *GlobalSign Domain Validation CA - SHA256 - G2* originally used trust paths via *GlobalSign Root CA - R3*, while a cross-sign by *GlobalSign Root CA* extends the trust by several years.

5.4.2 Alternative Paths. Often, multiple certificates of a XS-Cert provide valid paths to the same root store. This approach provides fall-back trust: it proactively maintains alternative paths to deal with unexpected revocations or removals. E.g., the intermediate XS-Certs for *Servision Inc.*, *XiPS*, and *KAGOYA JAPAN Inc.* – originally issued by GoDaddy's *ValiCert Class 1 Policy Validation Authority* – were cross-signed by SECOM's *Security Communication RootCA1* in 2012. The cross-signs provided readily usable fall-back paths when the originally issuing ValiCert certificate was removed from Mozilla and Google root stores due to its 1024 bit RSA key [11, 12].

Note that most cross-signs establish alternative paths as issuing CA certificates are often trusted in many root stores. The count in Table 1 only lists XS-Certs whose sole outcome are alternative paths, i.e., it includes only XS-Certs that do *not* expand the trust.

Takeaway: *Cross-Signing enables large root store coverage if issuing CAs span only a subset of root stores. Likewise, cross-signing can provide alternative trust paths to proactively deal with CA revocations and removals. Both ensures a non-disruptive user experience. However, multiple trust paths for a certificate can also lead to incomplete revocations and thus challenge the security of PKIs (cf. Section 5.1). Using cross-signing for these purposes thus necessitates better mechanisms to mitigate these security problems. We discuss possible solutions such as better logging and limited lifetimes for XS-Certs in Section 6.*

5.5 Cross-Signing Eases the Transition to New Cryptographic Algorithms

Security guidelines by entities like the CAB Forum motivate CAs to support advancements in cryptography early on, e.g., new signature algorithms. To maintain backward compatibility for legacy implementations, CAs use cross-signs to establish alternative trust paths that support new algorithms. Legacy clients that do not support the new algorithms can still use the old trust paths. We find this happening commonly with intermediate XS-Certs. For root certificates, this approach is not required - their self-signed signatures are typically not checked by clients [10]. In these cases, the CA can just issue a new intermediate that uses the new algorithms.

To give some examples, the *Virginia Tech Global Qualified Server CA* intermediate was issued with a SHA1 signature by *Trusted Root CA G2* (GlobalSign) in 2012. In December 2014, i.e., approaching the deadline for deprecating issuance based on SHA1 [37], GlobalSign cross-signed the intermediate from the *Trusted Root CA SHA256 G2*. The SHA1 intermediate was revoked in January 2017.

Other intermediates switched to SHA2 long before the official deprecation: The US Government (*Common Policy*) issued a SHA1 intermediate *Betrusted Production SSP CA A1* in 2008 and created a SHA256 cross-sign in 2010 using *Federal Common Policy CA*. The SHA1 intermediate was revoked in 2011.

Actalis obtained a SHA1 cross-sign from *Baltimore CyberTrust Root* for its SHA256 intermediate *Actalis Authentication Root CA* at time of original issuance. Likewise, the Japanese Government issued a cross-sign for *ApplicationCA2 Sub* to provide parallel support for SHA1 and SHA256 on creation of the intermediate in 2013.

Keynectis cross-signed its intermediate *KEYNECTIS Extended Validation CA* with roots of its subsidiaries Certplus (*Class 2 Primary CA*) and OpenTrust (*OpenTrust CA for AATL G1*). The cross-signs offer SHA1, SHA256, SHA512 and ECDSA-with-SHA384 signatures.

Takeaway: *Cross-signing can establish new trust paths that fully support new cryptographic algorithms. This enables state-of-the-art clients to achieve better security but at the same time maintains backward compatibility for legacy clients which validate the same certificate with the help of older paths. We expect CAs to put similar backward compatibility procedures in place once the use of a successor for SHA2, like SHA3, becomes popular. However, we posit that backward compatibility should be kept for a limited time only—using short certificate validity periods—to push application developers to support state-of-the-art cryptography.*

5.6 Effect of Ownership-Changes on XS-Certs

In this section, we analyze the effect of CA acquisitions on existing cross-signs. Cross-signs, especially across organizations, add contractual obligations—and necessitate trust granting CAs to check the actions of CAs they cross-signed. Thus, we are interested if new owners revoke cross-signs that were created by acquired CAs. Similarly, we are interested if cross-signs are revoked once the owner of a cross-signed CA changes.

5.6.1 Cross-Signs Outlive Ownership-Changes. The *Network Solutions Certificate Authority* root has been repeatedly cross-signed by members of Comodo's trust network. This process spanned several ownership changes. The original root was created 2006. At this time, Network Solutions had been owned by Pivotal Equity for three years. In the same year, Comodo's *UTN-USERFirst-Hardware* cross-signed the root twice. After Network Solutions was sold to General Atlantic, Comodo's *AddTrust External CA Root* cross-signed the certificate in 2010. Furthermore, we find a cross-sign by AddTrust: this intermediate has been valid since 2000—when Network Solutions was owned by VeriSign—but it likely was backdated (cf. Section 5.7.1). All these cross-sings outlived the acquisition of Network Solutions by web.com in 2011 and remain valid until 2020, thus spanning at least two ownership changes.

We also find cases in which the CAs that issue cross-sign certificates changed owners. When Digicert created the *DigiCert High Assurance EV Root CA* in 2006, it was cross-signed by *Entrust.net Secure Server Certificate Authority* and *Entrust.net Certificate Authority*. 3 years later, Thoma Bravo acquired Entrust and sold it to Datacard in 2013, which rebranded it to Entrust Datacard. Despite these ownership changes, the cross-signs of Digicert's root remained valid (until 2014 and 2015), making the new owners responsible for trust paths of certificates issued by DigiCert[9]. This raises the question if the new owners were aware of the cross-signs - and decided to keep them, or if they simply were not aware of their existence.

5.6.2 DigiCert – Internal Islands and External Legacy Cross-Signing. In the DigiCert group, most XS-Certs were created before DigiCert acquired the corresponding CAs. Thus, we predominantly find internal cross-signs within each subsidiary and only few cross-signs across DigiCert subsidiaries. Most cross-signs originate from times of VeriSign, Verizon, and QuoVadis. DigiCert only occasionally used the acquired certificates to cross-sign its own *DigiCert* roots.

When acquiring Verizon in 2015, DigiCert also became responsible for external XS-Certs. First, Verizon cross-signed root certificates of WellsFargo in 2013 and 2015. All corresponding intermediates were revoked by the CA's CRL and Mozilla's OneCRL in 2017, when the roots were removed from all root stores (after request by WellsFargo [15]), except for Apple. Similarly, Verizon cross-signed *Certipost E-Trust Primary Normalized CA* providing this former only Microsoft-trusted root a broad trust coverage.

Verizon also cross-signed state-controlled CAs which is potentially problematic (cf. Section 5.2). In 2010 and 2013, it cross-signed the *Swiss Government* root, increasing its trust beyond Apple and Microsoft. Similarly, it cross-signed the *Belgium Root CA*, *CA2*, and Portugal's *SCEE ECRaizEstado*–some of them were later revoked in 2018 due to a series of misissuances [21]. The acquisition of Verizon made Digicert responsible for these cross-signs of state-controlled CAs. Thus, cross-signs did not only provide state-controlled CAs with large trust coverage, but these cross-signs also faced ownership changes which increase the risk for unnoticed problems.

We describe further XS-Certs with ownership changes and provide details on Digicert's cross-signs in Appendix B.6.

Takeaway: *Considering the frequent cross-signs across CAs, potential new owners must—before acquiring a CA—review existing issued and received cross-signs and corresponding obligations. Similarly, cross-signing CAs must be informed when a cross-signed CA changes its owner. Both requires an easily accessible and verifiable store of XS-Certs as we suggest in Section 6.*

5.7 The Ugly: Potentially Problematic Practices and Missing Transparency

In this section, we highlight practices of CAs that—while not explicitly forbidden—are frowned upon by root store maintainers [68] or make it hard to assess the legitimacy of existing trust.

5.7.1 Backdating of Cross-signs. Comodo's *AddTrust External CA Root* backdates several cross-signs, i.e., it sets the *not before* field to a date several years before the actual issuance. We used crt.sh to verify that the early *not before* dates are not caused by re-issued root certificates. Backdating is explicitly forbidden if it bypasses a root store policy that relies on information in the *not before* field—like the deprecation of SHA-1 signatures for new certificates [37]. Mozilla lists backdating as problematic practice [68] as backdating can interfere with future policies (cf. not before rules, Section 5.1.1).

The *USERTrust ECC Certification Authority* root (valid since 2010) was cross-signed by *AddTrust External CA Root* in 2013 [20] using

[9]Note that Thoma Bravo was not in control of Digicert and Entrust at the same time. Thoma Bravo acquired Digicert in 2015, two years after selling Entrust to Datacard.

a *not before* date of 2000. Other cross-signs of the root use more accurate times, e.g., a cross-sign by *AAA Certificate Services* in 2019.

We find further cross-signs by AddTrust which we believe to be backdated as their *not before* date precedes that of the cross-signed root by several years. We lack clear evidence, but signs for a reoccurring backdating behavior of the CA are present: All presumably backdated cross-signs are issued by *AddTrust External CA Root* using *not before* dates from 2000 or 2010. Specifically, the cross-signs of *COMODO Certification Authority*, *COMODO ECC Certification Authority COMODO RSA Certification Authority*, and *USERTrust RSA Certification Authority* list *not before* dates from 2000 whereas the corresponding roots list dates from 2006, 2008 and 2010. Cross-signs of the *Network Solutions Certificate Authority*, a root created in 2006, list dates from 2000 and 2010.

Finally, we find StartCom to set *not before* to 2006 (corresponding to the original root) when cross-signing *StartCom Certification Authority* whereas the cross-signing *StartCom Certificate Authority G2* is valid only since 2010. This case is only a problematic practice and no strict violation of the policy. However, it is noteworthy that StartCom already backdated certificates long before the incidents that led to the removal of trust in WoSign and StartCom. That decision was partly based on their violation of policies by backdating certificates to circumvent SHA1 deprecation (cf. Section 5.1).

5.7.2 Missing Insight for Partly Disabled Comodo Root XS-Certs. Comodo requested the exclusion of some root certificates from root stores, but kept their cross-signs valid. Specifically, the root certificates of *USERTrust UTN-USERFirst-Hardware* and *AddTrust Class 1 CA Root* were removed from the root stores of Firefox and Android in 2017 as they are no longer in use [16]. However, corresponding cross-signs by *UTN-USERFirst-Client Authentication and Email* (also trusted by Firefox and Android) have not been revoked as they are still in use. Especially the *USERTrust UTN-USERFirst-Hardware* cross-sign is still heavily used after the withdrawal of its root: We find valid trust paths for 4,244,104 leaf and 276 CA certificates until 2020. We note that, from everything we can tell, this is—in this case—desired behavior and not malicious. It shows, however, that it is hard to tell for an outside observer if cross-signs just have been forgotten - or are purposefully kept active.

The removal (and eventual revocation) of *USERTrust UTN - DATACorp SGC* is a counterexample. It was removed from Firefox and Android root stores in 2015 due to a planned removal of public trust by the owner [13, 14]. After additional request by the owner, the intermediates were revoked by adding them to Mozilla's OneCRL [14]. The need for an additional request shows the complexity of revocations in face of cross-signs. In our opinion, the original removal request should already have raised questions on the revocation of the cross-signs. Instead, this required an explicit additional request by the owner—which only happened about 3 months later.

Takeaway: *The backdating of certificates might be tempting, e.g., to match the validity periods of the original certificate. However, backdating is a problematic behavior that hides the issuance date of a certificate without providing benefits. On top of this, cross-signing adds a new problem: The legitimacy of an incomplete revocation is hard to assess as corresponding information is typically unavailable. To achieve transparency, evaluating cross-signs and requesting information about them should be an integral part of revocation processing.*

XS-Certs in the Wild – Summary: *While cross-signing complicates the PKI, CAs also use it to adopt more secure cryptography while maintaining compatibility with legacy software. In the following, we propose several changes with the goal of establishing secure cross-signing practices and making cross-signs comprehensible.*

6 DISCUSSION AND RECOMMENDATIONS

CAs depend on cross-signing. For example, the fast tremendous success of Let's Encrypt would not have been possible without bootstrapping the CA through cross-signing, and also GoDaddy started with a cross-sign. Cross-signing also enables CAs to issue certificates with new, more secure cryptographic algorithms while maintaining compatibility with legacy applications—as it happened with the SHA1 to SHA256 transition. Cross-signing can be used to increase root-store coverage by establishing paths to different root stores—even in face of differing root store policies and resulting diverging root store setups. Cross-signing can pro-actively set up alternative trust paths based on other CA certificates such that end-user certificates still validate. Thus, cross-signing is a beneficial mechanism for the PKI ecosystem.

Yet, cross-signing complicates certificate handling and introduces new problems, some concerning the management of cross-signs. For an observer it is hard to tell if cross-signs intentionally or unknowingly outlive CA owner-ship changes (see Section 5.6.1). Problematic practices like backdating of certificates [68] further complicate the assessment of existing cross-signs. The by most important problem are undesired trust-paths caused by cross-signs. While revocation of CA certificates already is a complicated process, necessitating software updates and/or updates of vendor-specific CRLs, cross-signs make this process even more complex—and led to numerous incomplete revocations as shown in Section 5.1. New cross-signs need careful examination to make sure that they do not accidentally extend the trust of a CA, or span trust across PKIs.

Cross-sign revocations are a hard to solve problem for application software. For example, all major browsers and Microsoft use their own systems to propagate CA revocation information. This revocation information is, however, not easily accessible for many applications. A typical Unix program that uses OpenSSL or NSS has no easy way to get revocation information. It will typically blindly trust all CA certificates that it encounters—no matter if it was revoked. The same is true for mobile applications on iOS or Android. Even on Microsoft Windows one will only get access to some revocation information—and only when using the operating system's APIs. Many applications, however, chose to use OpenSSL even on Windows—e.g., through using the popular libcurl [53]. Programming languages like Go and Java also have their own validation logic and will thus not get revocation information [76].

Thus, cross-signing puts the security and privacy of users at risk: It can undermine the security of one of the most critical infrastructures in a digitally interconnected world, which—for better or worse—is the current trust backbone for nearly all secure communication on the Internet.

In the following, we suggest best practices preserving the benefits of cross-signing while limiting future security problems (Sections 6.1 and 6.6) and ensuring *awareness* of CAs as well as giving more *transparency* to the user (Sections 6.2–6.5). Automated checks

can hint at violations of these practices, but—like for many rules set up by root store maintainers—manual inspection is often necessary.

6.1 Limit Problems using Short Validities

Typically, cross-signs have validity periods ranging from several years up to several decades. As a result, security issues often remain problematic for years—especially for applications that do not use revocation information.

To limit the impact of security incidents related to cross-signing, we suggest CAs to limit the maximum validity period of cross-signs. The benefits of shorter validity periods have been discussed, e.g., in [25, 91]. Applications that lack access to revocation information will benefit greatly from this change—there is a much smaller window of opportunity for exploitation after a CA certificate should have been revoked. Continuous renewals of CA certificates will also limit the validity of a cross-sign to the term during which a business relationship between two entities is ongoing—such cross-signs cannot just be forgotten after an ownership change. Short validity periods for cross-signs should not pose an insurmountable problem for CAs: Increasing automation of certificate issuance and renewal—especially advanced by Let's Encrypt in the recent years—already enables the deployment and update of short-living leaf certificates on end-user systems like webservers. Let's Encrypt already defaults to a 3-month validity period for leaf certificates. Our proposal also matches current industry trends: Mozilla, Google, and Apple already announced to limit the maximum validity period of leaf certificates to 398 days (1 year + grace period) [52, 84, 96]. To lessen the impact of problems even more, validity periods should be reduced to about 4 days [25, 91] and—as underlined by our findings—also apply to CA certificates [91]. This would necessitate all CAs to adopt automation similarly to Let's Encrypt.

A more invasive and less preferable approach could be to couple a long-living cross-sign with short-lived proofs of freshness provided by the cross-signing CA. Whenever the certificate is used, the receiver checks if the proof of freshness is recent. These freshness proofs can be distributed by the webserver in the TLS handshake using OCSP stapling—which can be mandated by the certificate using the OCSP Must-Staple extension [44, 77]. However, OCSP Must-Staple does not provide the necessary deployment at CAs, servers, or clients [25, 27]. Thus, short validity periods—which do not require specific client support [25, 91]—are preferable.

6.2 Shedding Light on Cross-Sign Motivations

Our analysis also shows good and positive use cases for cross-signing. However, CAs are not required to state their motivation for a particular cross-sign, although this motivation is important for assessing if a cross-sign (still) serves a benign purpose or should be revoked. For example, a bootstrapping cross-sign can be revoked when the original root has reached the desired trust coverage. Similarly, a cross-sign that improves root store coverage should be revoked when the original CA certificate is no longer used. To shed light on the motivation and enable corresponding checks, we suggest requiring CAs to encode the motivation(s) for a cross-sign as a new *XS extension* in the resulting certificate as follows.

Bootstrapping. The XS extension should encode a reference to the bootstrapped certificate as well as the targeted root stores of the bootstrapping. A corresponding pending request for inclusion of the certificate into the root store must exist—root store maintainers can provide a proof. The cross-sign must not be renewed any longer when the bootstrapped certificate is trusted by the targeted stores.

Expanding Trust. A cross-sign that expands the trust should encode the corresponding stores in its XS extension. Such a certificate must only exist if no other certificate of the XS-Cert provides a valid path to the targeted root store (except for paths that are flagged as alternative, see below). To limit the impact of coincidentally resulting alternative paths by the cross-sign, applications may ignore cross-signs that intend to expand the trust for only other root stores than the store used by the application.

Fall-Back paths. While fall-back paths increase the complexity of the PKI, they can be necessary to, e.g., support legacy devices using old versions of trust-stores with old root certificates. Fall-back cross-signs should encode the targeted root stores and the current certificate they are a fall-back for. We posit that software that ships with up-to-date root stores that get updated regularly should ignore all fall-back certificates. As we describe, other cross-sign use cases must prove that they do *not* serve the sole purpose to create fall-back paths—via verifiable information in the XS extension.

Multiple Algorithms. When providing support for new (signature) algorithms, the cross-sign must—upon issuance—establish a valid path to a root store using the desired algorithms only. No other certificate should provide a valid path to the root store using these algorithms, except for fall-back paths (see above). To enable checks of this condition, the XS extension should encode the set of algorithms which are used to validate the new path; it should also encode the other CA certificates for this path.

Creation time. In addition to the XS-extension, all newly issued CA certificates should include signed timestamps of several CT logs. These can serve as indicator for the time of issuance (rather than the easily manipulatable *not before* field).

6.3 Ownership Change: Report on Cross-Signs

As our analysis in Section 5.6 shows, cross-signs typically outlive ownership changes of involved CAs. However, the public remains unaware if this is an explicit decision. We suggest providing transparency and raising the awareness for cross-signs when owners change. To this end, when the owner of a cross-sign changes, the issuers of the cross-sign should be required to publicly declare if the cross-sign should remain valid. Likewise, the new owner should publicly acknowledge that it will adhere to the obligations that arise from the possession of the cross-sign. We envision such statements to become a mandatory part of the report on CA ownership change that is already required by root stores [64]. Incomplete statements on affected cross-signs would justify revocations and challenge the trustworthiness of the misbehaving CAs.

6.4 Explain Revocation Inconsistencies

We uncover several revocation inconsistencies for XS-Certs (cf. Sections 5.1.3 and 5.7.2). However, it is sometimes unclear if these inconsistencies are accidental—or desired. To enable root store maintainers and researchers to correctly classify these cases, CAs should be required to pro-actively explain revocation inconsistencies of XS-Certs—and explain their purpose to allow for the setup

of checks (cf. Section 6.2). This information can nicely extend the existing best practice to explain the reason for a revocation in CRLs.

6.5 Enable Easy Access to Cross-Signs

Easy access to cross-signing information would significantly help to ensure that cross-signs are (still) legitimate. Certificate Transparency (CT) already aims at making certificate information available to everyone [56]. CT provides semi-trusted append-only public log-servers with the goal to contain all currently trusted certificates in the Internet—and it is well on the way to satisfying this goal [83]. CT is, however, not easily searchable and fragmented in lots of servers. Due to scalability problems, this fragmentation will likely increase in the future [83]. Thus, current CT practices do not ensure a great fit of CT for the evaluation of cross-signs.

Fragmentation of certificates to different CT logs complicates the search for cross-signs since they can be distributed across different logs. We suggest requiring CAs to report all certificates of a XS-Cert to the same CT log(s). This ensures that a single CT log can provide full information on the cross-signing properties of a certificate. To account for the high impact of root and intermediate XS-Certs, CAs should be required to report all CA certificates to a set of CT logs that only log CA certificates. These CT logs can provide fast and complete statements on the cross-signing properties of CA certificates. Due to the limited amount of CA certificates, they should remain small—and not require a huge infrastructure commitment.

6.6 Use Vendor-Controlled CRLs by Default

In Section 5.1, we show the importance of revocation information for intermediate certificates. We also highlight that, for the most part, only large webbrowsers get this revocation information—via vendor-specific CRLs. This, however, opens a potential window of attack on all other applications which lack a sufficiently simple way to get this revocation information [75].

We propose that a new standard system for the revocation of certificates should be created—which could be equivalent to either OneCRL or CRLSets. This system should be supported out-of-the-box by typical operating systems and SSL/TLS libraries (like OpenSSL). It should not necessitate any changes in applications using TLS. TrustBase [75] could be a candidate. Further, recent research shows the applicability of local revocation lists like OneCRL even for an efficient revocation of leaf certificates [55]. Making such approaches the default would significantly improve the safety and robustness of the whole Web PKI.

7 RELATED WORK

While there is a large body of work that examines different facets of the PKI [2, 47, 57, 82, 88] as well as the TLS and the HTTPS ecosystem [34, 49], most studies do not mention or examine cross-signing. In their 2013 Systematization of Knowledge paper, Clark and Oorschot give a thorough review of the issues of the Web PKI [28]. In this paper, the authors already note the problem of revoking CA certificates. They also mention that CA certificates may not even contain the necessary information for revocation checking. One of the first studies of the HTTPS ecosystems performed by Durumeric et al. already noted a high occurrence of cross-signing among CA certificates in 2013 [35]. However, their analysis is limited to an overview on the occurrence of general cross-signing and a very brief statement on effect on root store coverage without any further analysis. Acer et al. [1] analyzed the causes for certificate validation errors encountered by Chrome users during web browsing. They find that missing cross-sign certificates in a presented certificate path cause some errors and thus acknowledge the importance of cross-signing as means to root trust in widely trusted stores. Roosa et al. [81] identified cross-signing as one of the mechanisms that cause intransparency in PKI systems as CAs do not disclose cross-sign relationships when applying for inclusion in a root store. Only since April 2018, CAs must report all intermediates that provide a path to a root in Mozilla's root store and are not constrained to specific domain subtrees [61, 62]. Casola et al. [24] enable CAs to automatically check the compatibility of their policies *before* cross-signing. We highlight problems *beyond policies* and provide guidelines for a safe *operation* of cross-signs.

There is a large amount of work that examines different aspects of certificate revocation. Most studies, however, concentrate on the revocation of end-host certificates and do not detail the impact of revocation of intermediates or the interaction with cross-signing. Liu et al. measure certificate revocation in the Web PKI [59]. While they mention the importance of revoking intermediate certificates, their study only measures leaf-certificate revocations.

8 CONCLUSION

Our longitudinal study shows that cross-signing is a common practice in the Web PKI. We provide a classification of possible cross-sign patterns and analyze their use in and their effect on real world communications. We show that cross-signs are used to bootstrap trust in new CAs—which played a significant role in the tremendous success of Let's Encrypt. Cross-signs also lead to better trust-store coverage, giving CAs trust in stores they are not directly trusted in. Finally, cross-signing allows for a graceful transition to newer cryptographic algorithms like done in the case of SHA-2.

However, our work also highlights the problems that cross-signing introduces to PKIs, the largest being unwanted trust paths. Cross-signing complicates the already error-prone revocation processes of certificates. It led to numerous incomplete revocations. We also show that introducing new cross-signs requires extensive checks to prevent accidental PKI boundary breaches. Cross-signing also makes it hard for observers to determine if trust paths are legitimate—or just forgotten artifacts of past contractual relationships between CAs.

Based on the gathered insights, we propose new cross-signing best practices. It is our hope to initiate and steer the discussion for new rules that preserve the beneficial potential of cross-signing but mitigate its risks.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable feedback. This work was supported by the US National Science Foundation under grant CNS-1528156. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors or originators, and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Mustafa Emre Acer, Emily Stark, Adrienne Porter Felt, Sascha Fahl, Radhika Bhargava, Bhanu Dev, Matt Braithwaite, Ryan Sleevi, and Parisa Tabriz. 2017. Where the Wild Warnings Are: Root Causes of Chrome HTTPS Certificate Errors. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. ACM, New York, NY, USA, 1407–1420. https://doi.org/10.1145/3133956.3134007

[2] Johanna Amann, Oliver Gasser, Quirin Scheitle, Lexi Brent, Georg Carle, and Ralph Holz. 2017. Mission Accomplished?: HTTPS Security After Diginotar. In *Proceedings of the 2017 Internet Measurement Conference* (London, United Kingdom) *(IMC '17)*. ACM, New York, NY, USA, 325–340. https://doi.org/10.1145/3131365.3131401

[3] A. Andersen, K. Y. Yigzaw, and R. Karlsen. 2014. Privacy preserving health data processing. In *2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom)*. 225–230.

[4] Apple. 2018. Blocking Trust for WoSign CA Free SSL Certificate G2. https://support.apple.com/en-in/HT204132. Last accessed on May 05, 2020.

[5] SK ID Solutions AS. 2018. Certificates. https://www.sk.ee/en/repository/certs/. Last accessed on May 05, 2020.

[6] Sharon Boeyen, Stefan Santesson, Tim Polk, Russ Housley, Stephen Farrell, and Dave Cooper. 2008. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280. https://doi.org/10.17487/RFC5280

[7] Bugzilla. 2007. Bug 403437 – Request Valicert/Starfield/GoDaddy Root Certificates be enabled for EV. https://bugzilla.mozilla.org/show_bug.cgi?id=403437. Last accessed on May 05, 2020.

[8] Bugzilla. 2009. Bug 478418 – Please add US FPKI Common Policy CA certificate. https://bugzilla.mozilla.org/show_bug.cgi?id=478418. Last accessed on August 15, 2020.

[9] Bugzilla. 2009. Bug 527759 – Add multiple roots to NSS (single patch). https://bugzilla.mozilla.org/show_bug.cgi?id=527759. Last accessed on May 05, 2020.

[10] Bugzilla. 2011. Bug 650355 – Stop accepting MD5 as a hash algorithm in signatures (toggle security.enable_md5_signatures to false). https://bugzilla.mozilla.org/show_bug.cgi?id=650355. Last accessed on May 05, 2020.

[11] Bugzilla. 2013. Bug 881553 – Remove or turn off trust bits for 1024-bit root certs after December 31, 2013. https://bugzilla.mozilla.org/show_bug.cgi?id=881553. Last accessed on May 05, 2020.

[12] Bugzilla. 2013. Bug 936304 – Remove Entrust.net, GTE CyberTrust, and ValiCert 1024-bit root certificates from NSS. https://bugzilla.mozilla.org/show_bug.cgi?id=936304. Last accessed on May 05, 2020.

[13] Bugzilla. 2015. Bug 1208461 – Remove "UTN - DATACorp SGC" root certificate from NSS. https://bugzilla.mozilla.org/show_bug.cgi?id=1208461. Last accessed on May 05, 2020.

[14] Bugzilla. 2015. Bug 1233408 – Cross-certificates issued to "UTN - DATACorp SGC" revoked by COMODO. https://bugzilla.mozilla.org/show_bug.cgi?id=1233408. Last accessed on May 05, 2020.

[15] Bugzilla. 2017. Bug 1332059 – Remove WellsSecure Public Root Certificate Authority from NSS. https://bugzilla.mozilla.org/show_bug.cgi?id=1332059. Last accessed on May 05, 2020.

[16] Bugzilla. 2017. Bug 1378334 – Disable/Remove some legacy Comodo root certificates. https://bugzilla.mozilla.org/show_bug.cgi?id=1378334. Last accessed on May 05, 2020.

[17] Bugzilla. 2017. Bug 1404403 – SwissSign: Two certs issued with same issuer and serial number. https://bugzilla.mozilla.org/show_bug.cgi?id=1404403. Last accessed on May 05, 2020.

[18] Bugzilla. 2017. Bug 1404501 – Add DigiCert non-TLS Intermediate Certs to OneCRL. https://bugzilla.mozilla.org/show_bug.cgi?id=1404501. Last accessed on May 05, 2020.

[19] Bugzilla. 2017. Bug 1407559 – CCADB entries generated 2017-10-11T10:26:59Z. https://bugzilla.mozilla.org/show_bug.cgi?id=1407559. Last accessed on May 05, 2020.

[20] Bugzilla. 2017. Bug 1418148 – Comodo: USERTrust, USERTrust ECC Certification Authority, Certificate signature algorithm does not match TBS signature algorithm. https://bugzilla.mozilla.org/show_bug.cgi?id=1418148. Last accessed on May 05, 2020.

[21] Bugzilla. 2018. Bug 1432608 – Add EC Raiz Estado Cross Certificates to OneCRL. https://bugzilla.mozilla.org/show_bug.cgi?id=1432608. Last accessed on May 05, 2020.

[22] Bugzilla. 2019. Bug 1552374 – Remove Certinomis - Root CA. https://bugzilla.mozilla.org/show_bug.cgi?id=1552374. Last accessed on May 05, 2020.

[23] Bugzilla. 2019. Bug 1602265 – CCADB entries generated 2019-12-07T17:20:17Z. https://bugzilla.mozilla.org/show_bug.cgi?id=1602265. Last accessed on May 05, 2020.

[24] Valentina Casola, Antonino Mazzeo, Nicola Mazzocca, and Massimiliano Rak. 2005. An Innovative Policy-Based Cross Certification Methodology for Public Key Infrastructures. In *Public Key Infrastructure*, David Chadwick and Gansen Zhao (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 100–117.

[25] Laurent Chuat, AbdelRahman Abdou, Ralf Sasse, Christoph Sprenger, David Basin, and Adrian Perrig. 2020. SoK: Delegation and Revocation, the Missing Links in the Web's Chain of Trust. In *2020 IEEE European Symposium on Security and Privacy (EuroS P)*.

[26] Taejoong Chung, Yabing Liu, David Choffnes, Dave Levin, Bruce MacDowell Maggs, Alan Mislove, and Christo Wilson. 2016. Measuring and Applying Invalid SSL Certificates: The Silent Majority. In *Proceedings of the 2016 Internet Measurement Conference* (Santa Monica, California, USA) *(IMC '16)*. Association for Computing Machinery, New York, NY, USA, 527–541. https://doi.org/10.1145/2987443.2987454

[27] Taejoong Chung, Jay Lok, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M. Maggs, Alan Mislove, John Rula, Nick Sullivan, and Christo Wilson. 2018. Is the Web Ready for OCSP Must-Staple?. In *Proceedings of the Internet Measurement Conference 2018* (Boston, MA, USA) *(IMC '18)*. Association for Computing Machinery, New York, NY, USA, 105–118. https://doi.org/10.1145/3278532.3278543

[28] J. Clark and P. C. van Oorschot. 2013. SoK: SSL and HTTPS: Revisiting Past Challenges and Evaluating Certificate Trust Model Enhancements. In *2013 IEEE Symposium on Security and Privacy*. 511–525. https://doi.org/10.1109/SP.2013.41

[29] Comodo. 2018. Comodo CA is now Sectigo. https://comodosslstore.com/sectigo. Last accessed on May 05, 2020.

[30] Microsoft Corporation. 2018. Microsoft Trusted Root Certificate Program: Participants. http://social.technet.microsoft.com/wiki/contents/articles/31634.microsoft-trusted-root-certificate-program-participants.aspx. Last accessed on May 05, 2020.

[31] Comodo Cybersecurity. 2016. GlobalSign SSL Certificate Revocation Error Causes Issues for Users. https://blog.comodo.com/it-security/globalsign-ssl-certificate-revocation-error-causes-issues-users/. Last accessed on May 05, 2020.

[32] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos. 2012. Enabling data protection through PKI encryption in IoT m-Health devices. In *2012 IEEE 12th International Conference on Bioinformatics Bioengineering (BIBE)*. 25–29.

[33] Zakir Durumeric, David Adrian, Ariana Mirian, Michael Bailey, and J. Alex Halderman. 2015. A Search Engine Backed by Internet-Wide Scanning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (Denver, Colorado, USA) *(CCS '15)*. Association for Computing Machinery, New York, NY, USA, 542–553. https://doi.org/10.1145/2810103.2813703

[34] Zakir Durumeric, David Adrian, Ariana Mirian, James Kasten, Elie Bursztein, Nicolas Lidzborski, Kurt Thomas, Vijay Eranti, Michael Bailey, and J. Alex Halderman. 2015. Neither Snow Nor Rain Nor MITM...: An Empirical Analysis of Email Delivery Security. In *Proceedings of the 2015 Internet Measurement Conference* (Tokyo, Japan) *(IMC '15)*. Association for Computing Machinery, New York, NY, USA, 27–39. https://doi.org/10.1145/2815675.2815695

[35] Zakir Durumeric, James Kasten, Michael Bailey, and J. Alex Halderman. 2013. Analysis of the HTTPS Certificate Ecosystem. In *Proceedings of the 2013 Conference on Internet Measurement Conference* (Barcelona, Spain) *(IMC '13)*. ACM, New York, NY, USA, 291–304. https://doi.org/10.1145/2504730.2504755

[36] EUGridPMA. 2018. EUGridPMA and IGTF Distribution Site. https://dist.igtf.net. Last accessed on May 05, 2020.

[37] CA/Browser Forum. 2014. Ballot 118 – SHA-1 Sunset (passed). https://cabforum.org/2014/10/16/ballot-118-sha-1-sunset/. Last accessed on May 05, 2020.

[38] Slava Galperin, Dr. Carlisle Adams, Michael Myers, Rich Ankney, and Ambarish N. Malpani. 1999. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 2560. https://doi.org/10.17487/RFC2560

[39] Mark Goodwin. 2015. Mozilla Security Blog – Revoking Intermediate Certificates: Introducing OneCRL. https://blog.mozilla.org/security/2015/03/03/revoking-intermediate-certificates-introducing-onecrl/. Last accessed on May 05, 2020.

[40] Google. 2018. Android Source Code. https://android.googlesource.com/platform/system/ca-certificates. Last accessed on May 05, 2020.

[41] United States Government. 2018. Federal Public Key Infrastructure (FPKI). https://www.idmanagement.gov/fpki. Last accessed on May 05, 2020.

[42] Internet Security Research Group. 2018. Chain of Trust. https://letsencrypt.org/certificates. Last accessed on May 05, 2020.

[43] Internet Security Research Group. 2020. Transitioning to ISRG's Root. https://letsencrypt.org/2019/04/15/transitioning-to-isrg-root.html. Last accessed on July 21, 2020.

[44] Phillip Hallam-Baker. 2015. X.509v3 Transport Layer Security (TLS) Feature Extension. RFC 7633. https://doi.org/10.17487/RFC7633

[45] Jens Hiller, Johanna Amann, and Oliver Hohlfeld. 2020. Extended Paper Version. ArXiv inclusion pending. ArXiv identifier will be made available at https://github.com/pki-xs-analysis/cross-signing-analysis/blob/master/README_extended_paper_version.md.

[46] Jens Hiller, Johanna Amann, and Oliver Hohlfeld. 2020. Toolchain for Cross-Signing Analysis. https://github.com/pki-xs-analysis/cross-signing-analysis.

[47] 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 *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference* (Berlin, Germany) *(IMC '11)*. ACM, New York, NY, USA,

427–444. https://doi.org/10.1145/2068816.2068856
[48] Russ Housley, Tim Polk, Warwick S. Ford, and David Solo. 2002. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 3280. https://doi.org/10.17487/RFC3280
[49] L. S. Huang, A. Rice, E. Ellingsen, and C. Jackson. 2014. Analyzing Forged SSL Certificates in the Wild. In *2014 IEEE Symposium on Security and Privacy*. 83–97.
[50] Apple Inc. 2018. Lists of available trusted root certificates in iOS. https://support.apple.com/en-us/ht204132. Last accessed on May 05, 2020.
[51] Apple Inc. 2019. Available trusted root certificates for Apple operating systems. https://support.apple.com/en-bh/HT209143. Last accessed on May 05, 2020.
[52] Apple Inc. 2020. About upcoming limits on trusted certificates. https://support.apple.com/en-us/HT211025. Last accessed on May 05, 2020.
[53] Helge Klein. 2018. Windows: native SSL (Secure Channel) instead of OpenSSL? https://github.com/edenhill/librdkafka/issues/2023. Last accessed on May 05, 2020.
[54] Adam Langley. 2012. Revocation checking and Chrome's CRL. https://www.imperialviolet.org/2012/02/05/crlsets.html. Last accessed on Sept 12, 2020.
[55] J. Larisch, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. 2017. CRLite: A Scalable System for Pushing All TLS Revocations to All Browsers. In *2017 IEEE Symposium on Security and Privacy (SP)*. 539–556.
[56] Ben Laurie, Adam Langley, and Emilia Kasper. 2013. Certificate Transparency. RFC 6962. https://doi.org/10.17487/RFC6962
[57] Taeho Lee, Christos Pappas, Pawel Szalachowski, and Adrian Perrig. 2018. Towards Sustainable Evolution for the TLS Public-Key Infrastructure. In *Proceedings of the 2018 on Asia Conference on Computer and Communications Security* (Incheon, Republic of Korea) *(ASIACCS '18)*. ACM, New York, NY, USA, 637–649. https://doi.org/10.1145/3196494.3196520
[58] Sectigo Limited. 2020. crt.sh Monitored Logs. https://crt.sh/monitored-logs. Last accessed on Aug 24, 2020.
[59] Yabing Liu, Will Tome, Liang Zhang, David Choffnes, Dave Levin, Bruce Maggs, Alan Mislove, Aaron Schulman, and Christo Wilson. 2015. An End-to-End Measurement of Certificate Revocation in the Web's PKI. In *Proceedings of the 2015 Internet Measurement Conference* (Tokyo, Japan) *(IMC '15)*. ACM, New York, NY, USA, 183–196. https://doi.org/10.1145/2815675.2815685
[60] Microsoft. 2020. Configure Trusted Roots and Disallowed Certificates. https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/dn265983(v=ws.11). Last accessed on May 05, 2020.
[61] Mozilla. 2017. Mozilla Root Store Policy Version 2.5. https://github.com/mozilla/pkipolicy/blob/2.5/rootstore/policy.md. Last accessed on May 05, 2020.
[62] Mozilla. 2017. November 2017 CA Communication. https://ccadb-public.secure.force.com/mozillacommunications/CACommunicationSurveySample?CACommunicationId=a051J00003mogw7. Last accessed on May 05, 2020.
[63] Mozilla. 2018. NSS Source Code. https://hg.mozilla.org/releases/mozilla-release/raw-file/default/security/nss/lib/ckfw/builtins/certdata.txt. Last accessed on May 05, 2020.
[64] Mozilla. 2020. Mozilla Root Store Policy Version 2.7. https://www.mozilla.org/en-US/about/governance/policies/security-group/certs/policy/. Last accessed on May 05, 2020.
[65] MozillaWiki. 2016. CA:WoSign Issues. https://wiki.mozilla.org/CA:WoSign_Issues. Last accessed on May 05, 2020.
[66] MozillaWiki. 2018. CA:Symantec Issues. https://wiki.mozilla.org/CA:Symantec_Issues. Last accessed on May 05, 2020.
[67] MozillaWiki. 2019. CA/Certinomis Issues. https://wiki.mozilla.org/CA/Certinomis_Issues. Last accessed on May 05, 2020.
[68] MozillaWiki. 2019. CA/Forbidden or Problematic Practices. https://wiki.mozilla.org/CA/Forbidden_or_Problematic_Practices. Last accessed on May 05, 2020.
[69] Johnathan Nightingale. 2011. Mozilla Security Blog – DigiNotar Removal Follow Up. https://blog.mozilla.org/security/2011/09/02/diginotar-removal-follow-up/. Last accessed on May 05, 2020.
[70] Commonwealth of Australia. 2018. Defense PKI. http://www.defence.gov.au/pki/. Last accessed on May 05, 2020.
[71] Government of India Ministry of Electronics and Information Technology. 2018. Controller of Certifying Authorities. http://www.cca.gov.in/cca/. Last accessed on November 08, 2018.
[72] Japan Ministry of Internal Affairs and Communications. 2018. Government Certification Infrastructure (GPKI). https://www.gpki.go.jp/. Last accessed on May 05, 2020.
[73] Sultanate of Oman Information Technology Authority. 2018. Oman PKI Initiative. https://www.ita.gov.om/ITAPortal/Pages/Page.aspx?NID=965&PID=4147&LID=193. Last accessed on November 08, 2018.
[74] Government of the Netherlands Ministry of the Interior and Kingdom Relations. 2018. PKIoverheid Certificaten. https://cert.pkioverheid.nl/cert-pkioverheid-nl.htm. Last accessed on May 05, 2020.
[75] Mark O'Neill, Scott Heidbrink, Scott Ruoti, Jordan Whitehead, Dan Bunker, Luke Dickinson, Travis Hendershot, Joshua Reynolds, Kent Seamons, and Daniel Zappala. 2017. TrustBase: An Architecture to Repair and Strengthen Certificate-based Authentication. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 609–624. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/oneill
[76] Oracle. 2018. Java Secure Socket Extension (JSSE) Reference Guide. https://docs.oracle.com/en/java/javase/11/security/java-secure-socket-extension-jsse-reference-guide.html. Last accessed on Aug 21, 2020.
[77] Yngve N. Pettersen. 2013. The Transport Layer Security (TLS) Multiple Certificate Status Request Extension. RFC 6961. https://doi.org/10.17487/RFC6961
[78] Ronald Prins. 2012. DigiNotar Certificate Authority Breach "Operation Black Tulip". https://doi.org/10.13140/2.1.2456.7364
[79] Chromium Project. 2020. Certificate Blocklist. https://chromium.googlesource.com/chromium/src/+/master/net/data/ssl/blocklist/README.md. Last accessed on May 05, 2020.
[80] Chromium Project. 2020. CRLSets. https://dev.chromium.org/Home/chromium-security/crlsets. Last accessed on May 05, 2020.
[81] S. B. Roosa and S. Schultze. 2013. Trust Darknet: Control and Compromise in the Internet's Certificate Authority Model. *IEEE Internet Computing* 17, 3 (May 2013), 18–25. https://doi.org/10.1109/MIC.2013.27
[82] Quirin Scheitle, Taejoong Chung, Jens Hiller, Oliver Gasser, Johannes Naab, Roland van Rijswijk-Deij, Oliver Hohlfeld, Ralph Holz, Dave Choffnes, Alan Mislove, and Georg Carle. 2018. A First Look at Certification Authority Authorization (CAA). *SIGCOMM Comput. Commun. Rev.* 48, 2 (May 2018), 10–23. https://doi.org/10.1145/3213232.3213235
[83] Quirin Scheitle, Oliver Gasser, Theodor Nolte, Johanna Amann, Lexi Brent, Georg Carle, Ralph Holz, Thomas C. Schmidt, and Matthias Wählisch. 2018. The Rise of Certificate Transparency and Its Implications on the Internet Ecosystem. In *Proceedings of the Internet Measurement Conference 2018* (Boston, MA, USA) *(IMC '18)*. Association for Computing Machinery, New York, NY, USA, 343–349. https://doi.org/10.1145/3278532.3278562
[84] Ryan Sleevi. 2020. Enforce 398-day validity for certificates issued on-or-after 2020-09-01. https://chromium-review.googlesource.com/c/chromium/src/+/2258372. Last accessed on July 31, 2020.
[85] Christopher Soghoian and Sid Stamm. 2012. Certified Lies: Detecting and Defeating Government Interception Attacks against SSL (Short Paper). In *Financial Cryptography and Data Security*, George Danezis (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 250–259.
[86] Emily Stark, Lin-Shung Huang, Dinesh Israni, Collin Jackson, and Dan Boneh. 2012. The Case for Prefetching and Prevalidating TLS Server Certificates. In *Proceedings of the 19th Network and Distributed System Security Symposium (NDSS)*, Vol. 12. http://crypto.stanford.edu/~dabo/pubs/papers/ssl-prefetch.pdf
[87] Systems Swiss Federal Office of Information Technology and Telecommunication FOITT. 2018. Rootzertifikate. https://www.bit.admin.ch/bit/de/home/subsites/allgemeines-zur-swiss-government-pki/rootzertifikate.html. Last accessed on May 05, 2020.
[88] P. Szalachowski, L. Chuat, and A. Perrig. 2016. PKI Safety Net (PKISN): Addressing the Too-Big-to-Be-Revoked Problem of the TLS Ecosystem. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*. 407–422. https://doi.org/10.1109/EuroSP.2016.38
[89] Microsoft Defender ATP Research Team. 2017. Microsoft to remove WoSign and StartCom certificates in Windows 10. https://www.microsoft.com/security/blog/2017/08/08/microsoft-to-remove-wosign-and-startcom-certificates-in-windows-10/. Last accessed on May 05, 2020.
[90] J. Tepandi, I. Tahhirov, and S. Vassiljev. 2010. Wireless PKI Security and Mobile Voting. *Computer* 43, 6 (2010), 54–60.
[91] Emin Topalovic, Brennan Saeta, Lin-Shung Huang, Collin Jackson, and Dan Boneh. 2012. Towards Short-Lived Certificates. In *Proceedings of IEEE Oakland Web 2.0 Security and Privacy (W2SP 2012)*. http://www.ieee-security.org/TC/W2SP/2012/papers/w2sp12-final9.pdf
[92] N. Vatra. 2010. Public Key Infrastructure for public administration in Romania. In *2010 8th International Conference on Communications*. 481–484.
[93] Andrew Whalley. 2016. Google Security Blog – Distrusting WoSign and StartCom Certificates. https://security.googleblog.com/2016/10/distrusting-wosign-and-startcom.html. Last accessed on May 05, 2020.
[94] Andrew Whalley and Devon O'Brien. 2017. Google Security Blog – Final removal of trust in WoSign and StartCom Certificates. https://security.googleblog.com/2017/07/final-removal-of-trust-in-wosign-and.html. Last accessed on May 05, 2020.
[95] Chris Williams. 2016. How a chunk of the web disappeared this week: GlobalSign's global HTTPS snafu explained. https://www.theregister.com/2016/10/15/globalsign_incident_report/. Last accessed on July 24, 2020.
[96] Ben Wilson. 2020. Reducing TLS Certificate Lifespans to 398 Days. https://blog.mozilla.org/security/2020/07/09/reducing-tls-certificate-lifespans-to-398-days/. Last accessed on July 31, 2020.
[97] Kathleen Wilson. 2016. Mozilla Security Blog – Distrusting New WoSign and StartCom Certificates. https://blog.mozilla.org/security/2016/10/24/distrusting-new-wosign-and-startcom-certificates/. Last accessed on May 05, 2020.
[98] Kathleen Wilson. 2017. Mozilla Security Blog – Removing Disabled WoSign and StartCom Certificates from Firefox 58. https://blog.mozilla.org/security/2017/08/30/removing-disabled-wosign-startcom-certificates-firefox-58/. Last accessed on May 05, 2020.

A ETHICS OF DATA COLLECTION

The passive data collection effort that we perform has been cleared by the respective responsible parties at each contributing institution before they begin contributing. Our data-collection effort generally leverages the already existing network security monitoring infrastructure at the contributing institutions: the data collection effort is generally run by the network security teams of the contributing institutions on their already existing network monitoring platforms. We just provide them with a short script (which they can examine) that performs the TLS data extraction.

This script only extracts data that is not privacy sensitive - or anonymizes it *before* the data is sent to us. For example, it does *not* include the client IP address. Instead, the data that we are sent contains a seeded hash of the concatenation of the client and the server IP addresses. The seed is site-specific and unknown to us.

This approach allows us to determine when the same client connects to the same server repeatedly (e.g. to evaluate the effectiveness of session resumption), without enabling us to track which sites a single client accesses.

Moreover, note that we also do not include client-certificates in our data collection; we only collect certificates that Internet servers send to clients.

B DETAILS AND FURTHER EXAMPLES

In the following, we detail some already presented XS-Certs and provide further examples for some cross-signing categories. We describe even more examples in the extended paper version [45].

B.1 WoSign & StartCom: Early Cross-Signing

As briefly noted in Section 5.1.1, StartCom started cross-signing WoSign certificates years before being acquired by them and cross-signing continued after the acquisition. Specifically, *StartCom Certificate Authority* cross-signed the *Certification Authority of WoSign* in 2006, but revoked the cross-signs in March 2017. However, a further cross-sign of *WoSign CA Limited* was not revoked before 2018. StartCom also issued the intermediate *WoSign eCommerce Services Limited*, which was cross-signed by Certplus later on, in 2011 (cf. Section 5.1.1). These cross-signs do not yield undesired trust paths since the StartCom roots were constrained by Mozilla's and Google's *not before* constraints. However, they show a close business relationship between StartCom and WoSign years before WoSign's (not timely reported) acquisition of StartCom in 2015.

This behavior continued after the acquisition: *StartCom Certification Authority* issued and *Certification Authority of WoSign G2* cross-signed the new intermediate *StartCom Class 3 OV Server CA*.

B.2 Bootstrapping: Further Examples

Cross-signing enables CAs to bootstrap trust in certificates while waiting for their inclusion in root stores. We discussed several cases in Section 5.3. Overall, we find bootstrapping cases for Comodo and its subsidiaries (specifically for Comodo and USERTrust; we discuss these cases with other Comodo cross-signs in Appendix B.4), the DigiCert group (Digicert, VeriSign, Verizon, and Cybertrust), Let's Encrypt, IdenTrust, WoSign, AffirmTrust, GlobalSign, Actalis, Starfield Services, Amazon, T-Systems, WellsSecure, Dell Inc., SECOM, TeliaSonera, Unizeto, CertiPath, Certipost, SCEE, Chunghwa Telecom, Carillon Information Security Inc., ORC PKI, TAIWAN-CA.COM Inc., Hongkong Post ARGE DATEN - Austrian Society for Data Protection, CNNIC as well as CAs controlled by the U.S., Swiss, or Belgium governments.

To give more concrete examples, the itself bootstrapped *Starfield Class 2 CA* (cf. Section 5.3.3) used its trust to further bootstrap *Starfield Services Root Certificate Authority - G2*. With another cross-sign of this root, *Starfield Services Root Certificate Authority* added fall-back paths to Microsoft's root store from 2014 to 2019.

Considering state-controlled CAs, specifically the FPKI, *Federal Bridge CA* was effectively externally bootstrapped by the root XS-Cert *DST ACES CA X6*, together with the intermediate XS-Certs *VeriSign Class 3 SSP Intermediate CA - G2* and *IdenTrust ACES CA 1*. The FPKI further used internal bootstrapping for *DoD Root CA 2*, *Federal Common Policy CA*, and *Common Policy*. Furthermore, the Digicert group bootstrapped *Swiss Government SSL CA 01*, *Belgium Root CA* and *Belgium Root CA2* (cf. Sections 5.2.2 and 5.6.2).

B.3 Expanding Trust and Alternative Paths: Further Examples

As noted in Section 5.4, some root certificates are included in only some root stores. Consequently, a single issuer may provide only limited trust. Cross-signs by selected CA certificates can close these holes by providing trust paths to uncovered root stores. Other XS-Certs merely extend the validity period for already covered stores or establish fall-back trust paths. We list further examples in the following; for Comodo and its subsidiaries, we refer to Appendix B.4.

B.3.1 Increasing Trust Store Coverage. Examples for XS-Certs that use cross-signing to increase their root store coverage are the intermediate XS-Certs *Servision Inc.*, *XiPS* and *KAGOYA JAPAN Inc.*. They were issued by GoDaddy's *ValiCert Class 1 Policy Validation Authority* in 2007, 2008, and 2009, respectively. The ValiCert certificate provides trust paths for Mozilla, Google, and Apple, but not for Microsoft's root store. To expand the root store coverage, all three intermediates were cross-signed by the broadly trusted *Security Communication RootCA1* of SECOM end of 2012, which validated issued certificates for the Microsoft ecosystem, too.

Unizeto did not only cross-sign WoSign roots (cf. Section 5.1.1), but also provided *SSL.com* with trust on Apple devices. Specifically, *Certum Trusted Network CA* cross-signed the roots *SSL.com Root CA RSA* and *SSL.com RV Root CA RSA R2*. Both roots, while created 2016 and 2017, respectively, were not included in root stores before 2018. The cross-signs by Certum in 2018 extended trust to Apple's root store which did not include the roots before 2019.

B.3.2 Fall-Back Trust Paths. Cross-signs often also establish fall-back paths which proactively provide alternatives for unexpected revocations. E.g., as noted in Section 5.4.2, the SECOM cross-sign provided trust paths for certificates of Servision, XiPS, and KAGOYA JAPAN after the originally issuing ValiCert certificate was removed from Mozilla and Google root stores due to its 1024 bit RSA key [11, 12]. The same holds for Apple's root store when it removed the ValiCert root in 2018. Thus, cross-signing also keeps intermediates operable in the face of trust-store removals of their issuers.

Also Entrust maintains a small internal cross-signing interconnection to establish fall-back trust paths for already broadly trusted

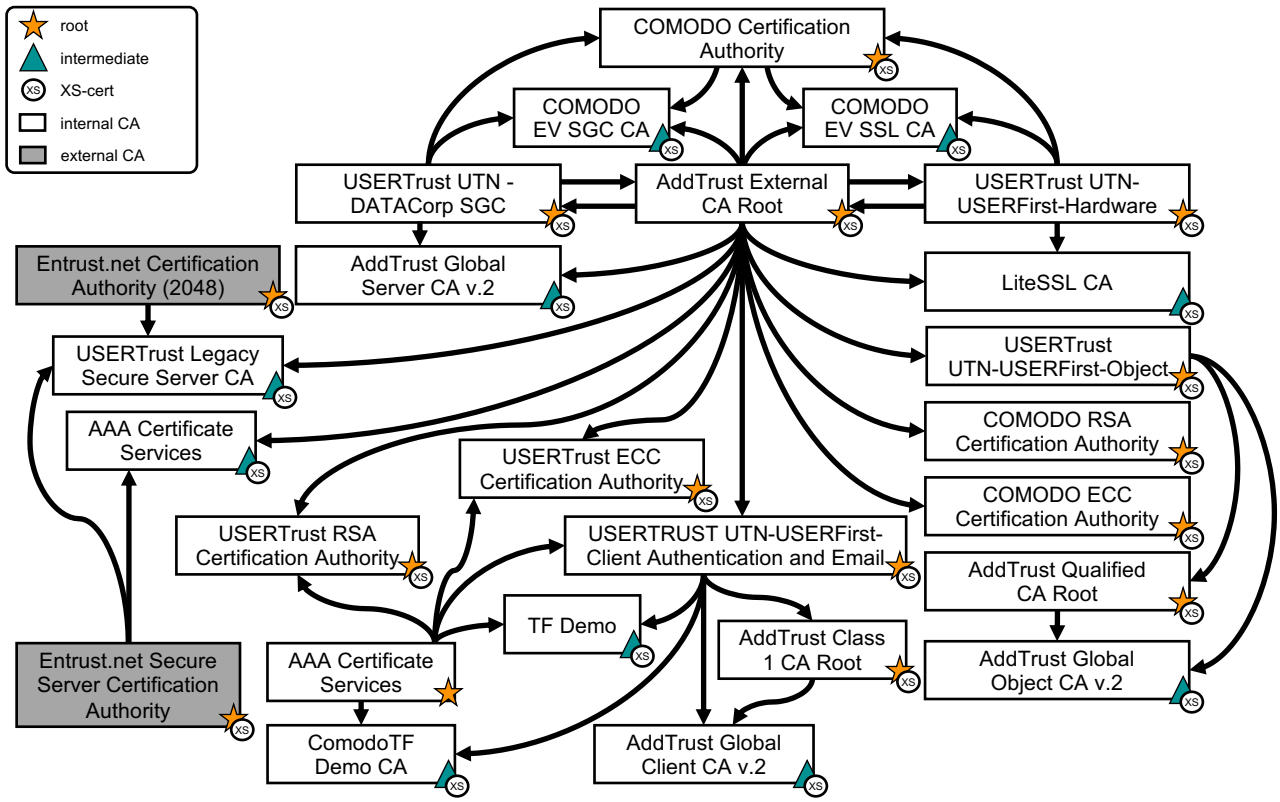

Figure 4: Mostly internal cross-signing in the Comodo/Sectigo group.

roots: *Entrust.net Secure Server Certificate Authority* cross-signed *Entrust.net Certificate Authority* and *Entrust Root Certificate Authority*. The latter furthermore cross-signed *Entrust Root Certificate Authority - G2* and *Entrust Root Certificate Authority - EC1*, expanding their root store coverage. Moreover, Entrust cross-signed its *Entrust.net Secure Server Certification Authority L1B*, *L1C*, *L1K*, and *L1M* establishing fall-back paths. Beyond these internal cross-signs, Entrust also received external cross-signs by today's members of the Digicert group (cf. Sections B.6.2). Moreover, *Entrust.net Secure Server CA* established fall-back paths for Trustwave's *Secure Trust CA* and the *TDC Internet Root* of the danish ISP TDC in 2006.

Apart from its cross-signing to bootstrap new roots (cf. Section 5.3.3), GoDaddy also maintains internal XS-Certs which establish alternative trust paths for already well trusted roots. In 2014, *Starfield Class 2 CA* also cross-signed *Starfield Root Certificate Authority - G2*. Similarly in 2014, *Go Daddy Class 2 Certification Authority* cross-signed *Go Daddy Root Certificate Authority - G2*, establishing alternative trust paths. Beyond these internal cross-signs, Go Daddy's *ValiCert Class 1 Policy Validation* cross-signed *SECOM Security Communication Root CA1*, again establishing fall-back trust paths for an already well-trusted root.

Also GlobalSign creates fall-back trust paths via cross-signing. Specifically, it establishes fall-back trust for *AlphaSSL CA - SHA256 - G2* and *GlobalSign CloudSSL CA - SHA256 - G3* by issuing cross-signs with the broadly trusted roots *GlobalSign Root CA* and *GlobalSign*. Further examples with fall-back paths are the FPKI XS-Certs *Federal Bridge CA G4*, *Federal Bridge CA 2016*, and *SHA-1 Federal Root CA*.

B.3.3 Extending Validity Periods. When a cross-signed certificate is already trusted in the root stores that are covered by the cross-signing CA, it typically establishes alternative paths. However, if the issuing certificate and the new cross-sign provide longer validity periods, it also extends the validity period of the XS-Cert.

Beyond extending trust for *GlobalSign Domain Validation CA - SHA256 - G2* (cf. Section 5.4.1), GlobalSign also extended the validity of *GlobalSign Organization Validation CA - SHA256 - G2* and *GlobalSign Root CA - R2* by several years.

Likewise, Symantec extends the validity via cross-signs for its *Symantec Trust Services Private SHA1 Root CA* and *SHA256 Root CA*, but also for intermediates that it manages for its customers, specifically *DAIICHI SANKYO COMPANY Public CA - G2*, *Astellas Global SMIME CA - G2*, and *TTP Services ABZ Nederland CA - G2*.

B.4 Comodo Bootstraps, Expands, and Backups

Comodo extensively uses cross-signing to interlink root certificates within and between its subsidiary CAs as we visualize in Figure 4. Still, this interlinking is established by only a few certificates which cross-sign several others. Especially the broadly trusted *AddTrust External CA Root* cross-signed various root certificates of Comodo and its subsidiaries. E.g., it bootstrapped trust for *USERTrust RSA Certification Authority* and *USERTrust UTN-USERFirst-Object*, and

provides fall-back trust paths for *USERTRUST UTN-USERFirst-Client Authentication and Email*. It also bootstrapped *COMODO ECC Certification Authority* and *COMODO RSA Certification Authority*, which nowadays provides them with trust in the grid PKI and fall-back paths. Before *USERTrust UTN-USERFirst-Hardware* and *USERTrust UTN - DATACorp SGC* ceased their operation, their mutual cross-signs with *AddTrust External CA Root* also provided the latter with fall-back trust paths — which naturally expanded to all its cross-signs. Similarly, the broadly trusted *AAA Certificate Services* root[10] cross-signed three USERTrust roots. For *USERTrust RSA Certification Authority*, it merely extended the validity periods; however, it bootstrapped initial trust for the other two.

USERTrust Legacy Secure Server CA was externally bootstrapped by *Entrust.net Certification Authority (2048)* and *Entrust.net Secure Server Certification Authority* in 2009. About one year before expiry, *AddTrust External CA Root* issued a broadly trusted replacement.

Also, Comodo's intermediate XS-Certs *COMODO EV SSL CA*, *COMODO EV SGC CA*, *AddTrust Global Object CA v.2*, *AddTrust Global Client CA v.2*, *TF Demo*, and *ComodoTF Demo CA* profit from fall-back trust paths established through cross-signing. One of Comodo's earliest intermediate XS-Certs is *LiteSSL CA* which Comodo created for *Positive Software Corporation* after acquiring it in 2005. This cross-sign likewise added fall-back trust paths.

B.5 Switching Trust Anchor

The expiry of or changing trust in certificates requires CAs to establish new trust paths to keep affected but still trustworthy certificates valid. For example, if the issuing certificate will soon expire, the owner of an intermediate may request a different CA to issue a longer lasting intermediate with the same subject and public key. Remember that we require certificates of a cross-sign to overlap by at least 121 days (cf. Section 3.1) and otherwise consider it as re-issuance. However, some trust changes happen years before expiry of the original certificates:

Symantec's *GeoTrust Global CA* issued an intermediate for *UniCredit* in 2012 and renewed it in 2015. In October 2016, Actalis cross-signed this certificate. However, the older GeoTrust issued certs were revoked the day after this issuance, rendering this to a switch to a new root rather than a typical cross-sign.

Similarly, the Spanish *Autoridad de Certificacion Firmaprofesional CIF A62634068* moved from an expiring to a more recent root.

Also Sertifitseerimiskeskus (SK), which is the partner of Estonia for id products, used cross-signing to switch its *AS Sertifitseerimiskeskus* root (expiring in 2016) to the new *EE Certification Centre Root CA*. Interestingly, the cross-sign does include a SHA3 intermediate since 2015 but no cross-sign with SHA2, limiting clients without SHA3 capabilities to the use of a SHA1 trust path element.

B.6 Ownership Changes: Further Examples and Digicert Details

In Section 5.6, we showed that cross-signs frequently outlive ownership changes which may lead to missing awareness of the cross-signs and overseen problems. We provide further examples and a detailed view on the Digicert cases in the following.

[10] The subject *AAA Certificate Services* is used for a root certificate but also for an unrelated intermediate XS-Cert, i.e., the latter uses a different private key.

B.6.1 Entrust Datacard. Entrust Datacard is another example with cross-signs that span an ownership change: In 2016, Entrust Datacard obtained control of *Affirmtrust Networking* and *Affirmtrust Commercial* which were cross-signed by *SwissSign Gold CA - G2* in 2009. The cross-signs for *Affirmtrust Networking* (which was also cross-signed by *SwissSign Silver CA - G2*) were revoked more than a year later when a double-use of the serial number for two certificates of the issuing *SwissSign Gold CA - G2* became known [17]. In contrast, the *Affirmtrust Commercial* cross-sign was revoked only several months before it's expiry in 2019. The corresponding root certificates, however, remain trusted.

B.6.2 Digicert Details. In Section 5.6.2, we outlined that Digicert inherited most of its cross-signs when acquiring CAs. We provide the details on these cross-signs in the following.

In the Digicert group, most XS-Certs were created before the corresponding CAs were acquired by Digicert. Thus, we predominantly find cross-signs between certificates of the same subsidiary and only very few cross-signs across Digicert subsidiaries as we illustrate in Figure 5. Most cross-signs originate from times of VeriSign, Verizon, and QuoVadis with Digicert adding cross-signs among its own *Digicert* roots only.

VeriSign cross-signed four of its own roots (*VeriSign Class 3 Public Primary Certification Authority - G3*, *G4*, *G5*, and *VeriSign Universal Root Certification Authority*), but also accounts for the cross-sign of *thawte Primary Root CA*. Similarly, *GeoTrust Primary Certification Authority*, which started out of Equifax' security business, was cross-signed during the VeriSign ownership. Only *GeoTrust Global CA* was already cross-signed when still owned by Equifax. After acquiring VeriSign as part of Symantec in 2017, Digicert only used its control over VeriSign roots to cross-sign *DigiCert Global Root G2* with *VeriSign Class 3 Public Primary Certification Authority - G5*. Otherwise, it only retained the existing cross-signs. We also find an external cross-sign that was created by VeriSign, but it did not last until the owner change to Symantec: When Thawte was owned by VeriSign, *Thawte Server* cross-signed *Entrust.net Secure Server Certificate Authority*. After their expiry in 2003, Thawte did not renew the cross-sign, however, Entrust obtained a cross-sign by the Digicert-controlled *GTE CyberTrust Global Root* in 2004.

Also most intermediate XS-Certs in the Digicert group were created by VeriSign or Symantec before Digicert obtained control over these CAs. Specifically, VeriSign created four intermediate XS-Certs for Thawte as well as three for itself. After acquiring VeriSign, Symantec, which did not cross-sign any of its roots, further created 3 intermediate XS-Certs for its own CA name. Only one further Symantec intermediate XS-Cert was created by Digicert after its acquisition. Only two (three) months after the cross-signing (acquisition), Digicert revoked all certificates of this latter XS-Cert, but otherwise kept the old intermediate XS-Certs active.

Also, a cross-sign of QuoVadis which already existed when it was acquired by WiSeKey in 2017 survived as an isolated island.

Beyond these scope-wise limited cross-signing, Verizon had numerous internal and external cross-signs which became part of the Digicert group when it acquired the Verizon and CyberTrust roots from Verizon. Internally, Verizon established multiple cross-signs across CyberTrust roots, and a Verizon root cross-sign (cf. Figure 5). Digicert only used *Baltimore CyberTrust Root* to cross-sign *DigiCert*

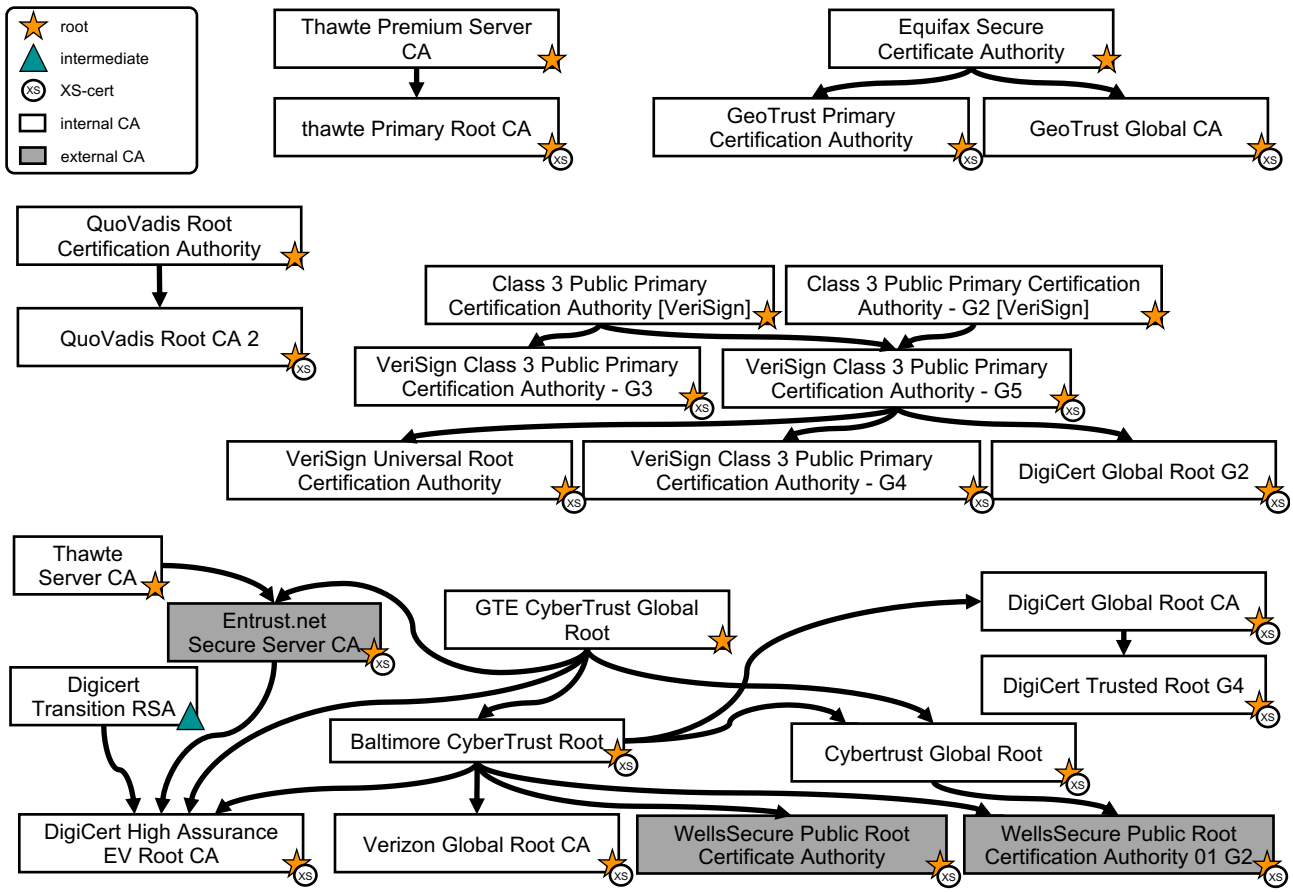

Figure 5: Root XS-Certs of Digicert and its subsidiaries: Cross-signs created prior to acquisition by Digicert form islands.

Global Root CA and *DigiCert High Assurance EV Root CA*. The latter was additionally cross-signed by *GTE CyberTrust Global Root* and *DigiCert Transition RSA Root* (and *Entrust.net Secure Server Certification Authority*; cf. Section 5.6.1). However, more prevalent are the cross-signs of external organizations. First, Verizon started cross-signing root certificates of WellsFargo since 2013. Specifically the *Baltimore CyberTrust Root* cross-signed WellsFargo's *WellsSecure Public Root Certificate Authority* and *WellsSecure Public Root Certification Authority 01 G2* in 2013 and 2015, respectively. The latter was already cross-signed by *Verizon Global Root CA* in 2013. All corresponding intermediates were revoked by the CA's CRL and Mozilla's OneCRL in 2017, when the roots were removed from almost all root stores (after request by WellsFargo [15]), except for Apple's store which still includes *WellsSecure Public Root Certificate Authority*. Similarly, the Verizon-controlled *GTE CyberTrust Global Root* cross-signed *Certipost E-Trust Primary Normalized CA* expanding the Microsoft-only trust to a broad trust coverage and creating another cross-sign that survived the acquisition by Digicert.

Verizon also cross-signed several state-controlled CAs which is potentially problematic (cf. Section 5.2). In 2010 and 2013, *Baltimore CyberTrust Root* cross-signed the *Swiss Government* root which increased the formerly limited trust (Apple and Microsoft only). One of the cross-signs expired in 2014, the other stayed active till after the Digicert acquisition. In 2013, *Belgium Root CA2* replaced a cross-sign by (non-Digicert) *GlobalSign Root* from 2007 with a further Verizon cross-sign by *CyberTrust Global Root*. This cross-sign also stayed active after the Digicert acquisition until October 2017 when it was revoked in OneCRL [19]. Verizon also used *GTE CyberTrust Global Root* and *Baltimore CyberTrust Root* to cross-sign Portugal's *SCEE ECRaizEstado* (formerly trusted by Microsoft only). This time, Digicert actively repeated the latter cross-sign shortly after obtaining control over the CyberTrust roots. However, when the cross-sign by *GTE CyberTrust Global Root* expired in August 2018, the other cross-signs were revoked due to a series of misissuances [21]. Thus, several CAs provided state-controlled CAs with a larger trust coverage than provided by the original roots.

Finally, we find a cross-sign between Digicert's very own roots, i.e., *DigiCert Global Root CA* cross-signed *DigiCert Trusted Root G4*.

B.7 Cross-Signing in the Grid PKI

In the grid-PKI, we do not find a real XS-Cert, but an interesting reissuance. Originally, *ESnet Root CA 1* issued an intermediate named *NERSC Online CA*. A month before its expiry, NERSC created a corresponding root certificate instead of requesting a new intermediate. One could argue that ESnet helped NERSC to bootstrap the trust in its later root.