A complete study of P.K.I. (PKI's Known Incidents)

Nicolas Serrano
School of Informatics,
Computing & Engineering
Indiana University
Bloomington
nicserra@iu.edu

Hilda Hadan
School of Informatics,
Computing & Engineering
Indiana University
Bloomington
hhadan@iu.edu

L Jean Camp
School of Informatics,
Computing & Engineering
Indiana University
Bloomington
ljcamp@indiana.edu

Abstract— **In this work, we report on a comprehensive analysis of PKI resulting from Certificate Authorities' (CAs) behavior using over 1300 instances. We found several cases where CAs designed business models that favored the issuance of digital certificates over the guidelines of the CA Forum, root management programs, and other PKI requirements. Examining PKI from the perspective of business practices, we identify a taxonomy of failures and identify systemic vulnerabilities in the governance and practices in PKI. Notorious cases include the "backdating" of digital certificates, the issuance of these for MITM attempts, the lack of verification of a requester's identity, and the unscrupulous issuance of rogue certificates. We performed a detailed study of 379 of these 1300 incidents. Using this sample, we developed a taxonomy of the different types of incidents and their causes. For each incident, we determined if the incident was disclosed by the problematic CA. We also noted the Root CA and the year of the incident. We identify the failures in terms of business practices, geography, and outcomes from CAs.**

We analyzed the role of Root Program Owners (RPOs) and differentiated their policies. We identified serial and chronic offenders in the PKI trusted root programs. Some of these were distrusted by RPOs, while others remain being trusted despite failures. We also identified cases where the concentration of power of RPOs was arguably a contributing factor in the incident. We identify these cases where there is a risk of concentration of power and the resulting conflict of interests.

Our research is the first comprehensive academic study addressing all verified reported incidents. We approach this not from a machine learning or statistical perspective but, rather, we identify each reported public incident with a focus on identifying patterns of individual lapses. Here we also have a specific focus on the role of CAs and RPOs. Building on this study, we identify the issues in incentive structures that are contributors to the problems.

I. INTRODUCTION

The Public Key Infrastructure supports secure connections between clients and servers. Also, it is used to send encrypted and authenticated emails, to sign software and to authenticate users into a secure system. Public Key Infrastructure (PKI) comprises a series of servers, network protocols, hashing and encrypting algorithms, security policies, systems and applications, working together to allow, for example, that a person can check his bank account online without the fear of an account takeover.

But we all trust in PKI. We must trust in PKI. It was conceived to be trusted. Its cryptographic foundation is solid, the role of each participant is defined, the hardware is mature and applications program interfaces are widely used. However, there have been problems with PKI. There are reasons to reconsider this trust. For example, while the mathematical foundations of the cryptography used in PKI have been studied and demonstrated to be complex to crack, advances in hardware have turned computationally secure algorithms into breakable ones. In addition, sometimes the implementation of these cryptographical algorithms introduces flaws or vulnerabilities that are external to the core crypto-mathematical function, and that can be exploited by attackers.

Sometimes, the vulnerabilities are not in the cryptographic protocols, implementing code or hardware, but in the business systems or processes that support the operations of PKI, for example, in the issuance of digital certificates. Certificates above all are a good sold in the PKI world. These miscellaneous but necessary steps that are required to obtain a digital certificate have proven to be sometimes hazardous.

Here we address the business component of PKI, examining the organizations that are the issuers of the certificates. The goal of a business is to be competitive and to make profit. The goal of a digital certificate is to bring security to its user. Therefore, digital certificates are private goods that offer security to its users and that are sold by some companies for a profit. These companies may be interested in ensuring security to people interacting with their customers after the sale, but the goal of a certificate authority (CA) is to profit from selling as many certificates as possible. It would be possible to make a theoretical argument that this is a moral hazard[1], but here we take an empirical approach to document the questionable behaviors of these companies. One common behavior is failing to verify the identity of the requester or issuing certificates that are structurally susceptible to misuse. From a strictly technical perspective, these misbehaviors may seem unexpected from given the key role of CAs in PKI and the trust that they hold. Still, these issues cannot be called particularly surprising. After almost 20 years, questions that arose in the first generation of PKI about vulnerabilities arising from assumptions about organizational behaviors [1] as well as about human trust [2] remain unanswered.

There is no question of the need for the most secure and unbreakable cryptographic algorithms and investment in

[1]A moral hazard occurs when the party decides to take a risk, or how much risk to take, is not the one that bears the harm resulting from the risk.

Electronic copy available at: https://ssrn.com/abstract=3425554

the associated libraries; however, the moment that profit is introduced and risks are allocated more questions must be asked. This empirical evaluation addresses the issues of (as Ross Anderson says) *"Why Information Security is Hard"*, [3].

II. Related Work

The two empirical contributions of this work are the systematic compilation and classification of CA failures as well as a taxonomy of these failures beyond rogue certificates. While rogue CAs are a known problem, other systematic failures are individually studied yet not evaluated as part of a taxonomy. This research builds on work primarily from the security community rather than the business literature.

The research is most aligned with this work on the original objections to the concentration of trust in certificate authorities. Lauded technologist Martin Abadi identified the presence of perverse incentives as PKI was being rolled out [4]. In contrast to the hierarchical model with transitivity of trust, he proposed that individuals begin with little or no trust and then add roots. This was echoed by philosopher Helen Nissenbaum, who joined with peers to point out that trust is not only not transitive, but also that the assumptions of patterns of human interaction in PKI were in direct opposition to the understanding of actual human behavior by social scientists, behavioral researchers, and philosophers [2].

As previously noted, the economic challenges and perverse incentives of the current x.509 system were discussed by Schneier and Ellison in 2000 [1].

In one examination of the socio-technical components of PKI, Park addressed the citizen-facing adoption of PKI in South Korea and concludes that the success was despite technical choices (e.g., ActiveX) not because of technical excellence [5]. Another researcher proposed a socio-technical solution to the challenges of TLS by building a model that embeds the existence of possible warnings and potential user responses [6].

S. Roosa et al. discussed some of the most known incidents in PKI related to CAs problems, including legal and economic aspects of the PKI model [7]. In their conclusions, they share points with this research, for example, stating that the CA's audits have room for improvement. In a paper some years later, authors at INRIA and Microsoft found that despite requirements for improved practices from the CA Browser Forum in 2015 there was a large number of noncompliant certificates, [8]. The size of CAs was found to be indeterminate, yet some months later Symantec was forced to sell their certificate business to DigiCert due to the chronic bad practices at the VeriSign-owned CAs (VeriSign, Thawte, Equifax, GeoTrust, and RapidSSL).

S. Matsumoto et al. proposed a PKI based on the blockchain (Instant Karma PKI, IKP) to offer incentives for CAs if they behave correctly, and for others if they report rogue certificates [9]. Using blockchain, the mechanism can be implanted in a decentralized fashion with automatic alerts generated by the smart-contracts processing in the nodes. The idea is based on the same findings of this study: the lack of incentives that CAs have to behave according to what it is expected of them. However, the blockchain solution does not resolve the incentives issue, obfuscates the trust chain, and makes revocation more problematic.

Khan and coauthors went beyond and proposed a new PKI paradigm, Accountable and Transparent TLS Certificate Management (ATCM) [10]. Using this scheme, the entire life-cycle of digital certificates would be made public and re-engineered to fix existent flaws (like revocation processes) and improve its performance. Once again, one goal of this approach is to make the activity of CAs public and increase accountability.

Wang et al. proposal share points in common with previous ideas [11]. While the goals of implementing certificate and revocation transparency are similar to ATCM, the architecture of the system is based on the blockchain, like in IKP. Both ATCM and this proposal are very similar to the presently used Certificate Transparency mechanism, as discussed in Section VII. [12]

J. Gustafsson et al. surveyed the implementation of Certificate Transparency by different actors and how these differ between them [13].

Q. Scheitle et al. provided a broader consideration of Transparency, identifying the privacy threats and data leakages that this public solution causes to the domains [14]. S. Eskandarian et al. provided possible solutions to maintain the privacy of subdomains in the public logs, and to protect the privacy of users querying these logs [15].

Leaving the Certificate Transparency discussion, Braun et al. demonstrated that users automatically trust in a huge number of CAs, yet they make use of a minimal subset of these [16]. In order to prevent unnecessary risks, an approach where the set of trusted CAs are individualized to each user would arguably improve the network.

Z. Dong et al. identified the problem that rogue certificates bring to the PKI network. They proposed a machine learning method for timely detection, one that combined global and local data. An advantage of that approach is that there was no requirement to change the existent infrastructure and it does not decrease individual privacy [17]. The proposal offered promising results in the laboratory, and immediately identified the rogue in the real *"2014 - India CCA"* incident, which is touched upon in Section VI.

A. Micheloni et al. similarly focused on identifying rogue certificates [18]. Their scheme is similar to previous ideas using notaries, however, they decentralize the method and implement it in a P2P fashion called Laribus. In addition to improving the availability of a centralized notary system, they protect the privacy of the users when querying the "Notary" peers.

D. Kumar et al. developed a tool called ZLint that helped to detect misconfigured digital certificates in the wild, based on the specification of the Baseline Requirements (BR) [19]. On the one hand, they showed that mis-issuance of digital certificates has declined abruptly in recent years; however, they discovered that some small CAs have been

Electronic copy available at: https://ssrn.com/abstract=3425554

issuing erroneous certificates since the beginning of their operations at an alarming rate. O. Gasser et al. measured the presence of misconfigured digital certificates as well, through active scans of the network and by searching the logs of Certificate Transparency [20]. While both approaches discovered numerous misconfigured certificates, the rate of misconfigured certificates found in the transparency logs was lower than that of other collection approaches.

Finally, while this study addresses the moral hazard and resulting negative externalities that errors in CAs bring to those relying on PKI, other major areas of study are in the faulty implementations of the cryptographic model behind PKI. Brubaker et al. developed a methodology for large-scale testing of digital certificates validation in different TLS implementations [21]. The research revealed alarming security vulnerabilities in several TLS implementations that allowed the creation of rogue certificates, and a lack of adequate warnings to the final users by the major web browsers. Another example is the widely known "Goto Fail", where Apple's TLS library was found not to verify correctly that a certificate had any association with the domain it was intended to verify.

We are not aware of current research focusing on misbehaviors and unethical practices of CAs in the last years, or in classifications of PKI incidents by their cause or type. Previous large-scale analysis depended on automated identification of non-compliant certificates, where business relationships or domain of use may have created false positives. Here we use small, individually verifiable instances of certificate failures with a focus on the organization of the market.

III. Certificate Governance

Certificate governance is implemented with three major stakeholders. There are the managers of Root Programs. Root Programs determine which certificate authorities will be included by default in an application distribution, and which will be untrusted. The CA Forum includes representatives from the Root Programs and from the Certificate Authorities. The CA Forum sets standards for certificates themselves and the processes by which these are issued. And, of course, Certificate Authorities actually issue certificates.

To provide an understanding of the PKI ecosystem, we begin by describing the Root Programs, summarizing their differences at the end of the section. Later we describe the CA Forum and their guidelines and requirements.

A. Root Programs Requirements

Root Programs are managed by different entities that create and distribute software platforms. We detail and differentiate the Root Programs of Microsoft, Apple, Google and Mozilla, given their importance in the ecosystem. However, there are other Root Programs, for example: Oracle's Certificate Authority Root Certificates in Java[2], Adobe Approved Trust List[3] and Android[4] (although this is more a store policy than a formal program). There are programs that have not survived; for example, Nokia, Blackberry and Opera had their own programs years ago.

The impact of the root programs varies by scope. For example, the January 2019 market share of Chrome is 68.48% with Edge and IE coming a distant second at 10.4% and Mozilla, Safari, and Opera following at 9.9%, 6.44%, and 2% respectively (for desktop web browsers). Based on these figures, the Root Programs and root certificate choices of the top five browsers were examined, as these cover over 96% of the market.

1) Microsoft Edge and Internet Explorer:

The Microsoft Trusted Root Certificate Program (known as "Program") is used by Microsoft to identify those root certificates which will be included in Microsoft products, most notably in Windows [22].

Applicants who seek to have the root of their Certificate Authorities included must complete a formal application process. This includes:

- Basic contact information about the applicant.
- The detail of the root signing certificate(s).
- Evidence that the inclusion of the CA into the Program results in a benefit to Microsoft's customers.
- A positive result from a third-party auditor[5],

The contract verifies that the CA inclusion in the program is a unilateral decision taken solely by Microsoft that can be ended with Microsoft's discretion. Once accepted and after inclusion in the next code release, the CA must comply with basic operational requirements.

- A positive audit result for each root certificate used in signing.
- Contact information updates.
- The CA must share with Microsoft its complete PKI hierarchy.
- Verification of all requirements to subordinates or cross-signed root certificates not included in the Program.
- Informing Microsoft about transferring the ownership of an enrolled root certificate to another entity.

For example, the sale of StartCom to WoSign violated the last of these requirements.

- General root certificate requirements
  - These are x.509 v3 and the CN field identifies the publisher and is unique.
  - A life range between 8 and 25 years.
  - Private key and Subject Name must be new for each one.
  - Government CAs are restricted to .gov domains and their country of sovereignty.

[2]www.oracle.com/technetwork/java/javase/javasecarootcertsprogram-1876540.html

[3]https://helpx.adobe.com/acrobat/kb/approved-trust-list2.html

[4]https://android.googlesource.com/platform/system/ca-certificates/+/master/files/

[5]The audit requirements can be found in https://technet.microsoft.com/en-us/library/cc751157.aspx

Electronic copy available at: https://ssrn.com/abstract=3425554

  - Intermediate CAs must follow the requirements of the CAB Forum Baseline.
  - A single CA cannot be used to issue server authentication and code signing/timestamping certificates at the same time.
  - Permitted EKUs for the root certificates.
- Strength of keys requirements
  - No 1024-bit RSA used.
  - No SHA-1 used.
  - Specific ECC allowed.
- Certificate revocation requirements
  - The CA must be able to revoke any certificates issued by it.
  - The revocation process must follow a documented revocation policy.
  - All certificates must be associated with a CRL or OSCP.
  - The CA must direct its own timestamp authority.

In case of noncompliance with the Program at any time, Microsoft will remove the root certificate(s) from the program. Again, WoSign and StartCom provide an exemplar [23]. In addition to the revocation of the CA and its subordinates, Microsoft has other options. One of these is automatically blocking access to resources signed by rejected CA certificates to all customers. Another is to demand a third party audit of the CA, its business and technical operations, back to the date that Microsoft determines. Microsoft maintains the right to communicate with any affected parties, including by a public release of information.

In terms of responding to any CA incident and thus protecting customers, the CA agrees to a set of conditions. Microsoft must be informed about any occurrence in no more than 24 hours with full information related to the causes, consequences and miscellaneous aspects of the incident. This notification should include a list of all certificates miss-used due to the incident. Then, as the CA responded, they must provide continuous feedback to Microsoft about the recovery and mitigation of the issue. After the incident has been addressed, the CA must provide a full detailed final report related to the security incident.

As November of 2018, the Program has 388 accepted root certificate participants, with 347 of these active [24].

2) Safari: Apple, the provider of the web browser Safari, uses a similar approach to Microsoft. Root certificates are kept in a store in the operating system and the root certificate authorities must comply with documented requirements to get their root certificates into this store [25]. The program is called Apple Root Certificate Program.

The main requirement is to conduct a series of CA audits. Apple accepts the CA audits carried by WebTrust [26] or equivalent. In addition, in case of issuing EV certificates, the CA must pass the specific WebTrust's audit for EV [27] and, also, follow the CA/Browser Forum guidelines [28]. For simple TLS certificates, the WebTrust's audit requirements are [29] and the CA/Browser Forum guidelines [30]. Also, CAs should notify to Apple any transfer of ownership in the CA operations.

Finally, in the same spirit that Microsoft, Apple states that the inclusion of the root CA certificate must bring value to Apple's users, and that they can remove root certificates at their own discretion.

As can be seen, Apple's Root Certificate Program relies completely on the CA/Browser Forum guidelines and requirements, without special needs as in the case of Microsoft.

At present, the last version of Apple's OSs includes 178 root certificates in their trust store (and 38 explicitly blocked) [31].

3) Google Chrome: The root certificate requirements demanded in Google Chrome are defined by The Chromium Projects [32]. In general, the policy followed by this web browser is to make use of the certificates store of the operating system. For example, if a root certificate is found in the Microsoft Windows Certificate Store, then it is considered trustworthy. In the case of several Linux distributions, due to the lack of a centralized root certificate program Chrome uses Mozilla's Network Security Services (NSS) [33] to perform certificate verification[6].

Also, root CAs that issue Extended Validation certificates are hard-coded in the binary file of the browser, therefore, in that case the CA must contact Chromium to be supported in Chrome. In addition, for these CAs Google demands the adherence to its Certificate Transparency policy[7]. Not adhering to this will make Google cease to show the EV indicator in the browser.

There are other scenarios where Google Chrome will distrust a root certificate despite its existence in the OS certificate store. Chromium is very clear (and dramatic) about this[8]: *"Google Chrome reserves the right to distrust root certificates present in the operating system's root certificate list. At the core of trust in the PKI system is the fact that the operation of Root CAs is beyond reproach. If one of these guardians of trust were to operate in a non-trustworthy way, it would be no different than a police officer who was covering up a crime or protecting the identity of a criminal (because it reflected personally on the officer), or a firefighter who was not responding to fires in which people died. If one of these bastions of public trust (police, fire) were violating the trust we had placed in them, the reaction would be strong and swift. And, it is worth noting that this is so egregious a violation, that there is no consideration as to the collateral damage that might be caused by removing him/her from society - for example, hardship to his family, since he is the sole breadwinner. Our hearts would go out to those who were adversely affected, but it would not alter the effect".*

Google expects the following behavior from a CA:

- Identity verification of the certificate requester, with a level of assurance depending on whether it is a simple

[6]https://wiki.mozilla.org/CA/FAQ

[7]https://docs.google.com/viewer?a=v&pid=sites&srcid=Y2hyb21pdW0ub3JnfGRldnxneDoyNjg1MWJkOWY2MmM4MzA0

[8]https://www.chromium.org/Home/chromium-security/root-ca-policy

Electronic copy available at: https://ssrn.com/abstract=3425554

domain server certificate or an EV.
- Every certificate issued has a record associated with it.
- Every certificate signed by the CA must have an entry in the logs, which must maintain its integrity and be audited frequently to detect unexpected behaviors.
- In case of miss-issuance of a certificate, the CA must notify all affected parties, revoke the problematic certificate and publish the revocation.
- Have adequate controls in place to prevent non-authorized access and issuance of certificates.
- In case of severe compromise, perform a detailed post-mortem analysis, fully disclosing it to the public.

Any deviation to this expected conduct, especially after a compromise in the CA or fraudulent behavior, will end in Google not trusting anymore in the root certificate despite its possible trust by the operating system. Also, for Root CAs that have shown problems in the past, Google can make mandatory their adherence to Certificate Transparency for all issued digital certificates, including not EV ones.

4) Mozilla Firefox: Mozilla has a root certificate program like Microsoft and Apple. The difference with them is that this program, the Mozilla's CA Certificate Program, manages the root certificate inclusion in the Network Security Services (NSS) in contrast to a determinate operating system. This NSS is an open source library whose goal is to help in the security aspects of client-server applications' development [33]. Mozilla Firefox makes use of the NSS root certificate store in the same way that Google Chrome uses it in some Linux distributions [34]- [35].

The policy that governs the Mozilla CA certificate is the Mozilla Root Store Policy[9] [36]. This document is a formal policy with a narrowed scope that effectively applies to the Root CAs or Intermediates CAs that have their certificates distributed by Mozilla. The document contains several sections that the CAs must follow. As a matter of example, CAs must:

- Offer a relevant service to Mozilla's users.
- Provide their services in a safe environment and enforce appropriate access control.
- Perform a detailed validation of a certificate request, and perform corrective actions if the used method of validation contains a security vulnerability.
- Comply with the CA/Browser Forum guidelines and requirements.
- As in other cases, frequently be audited against CA good practices defined by WebTrust or ETSI [37], by authorized auditors.
- Publicly disclose audit results, Certification Practice Statements and other relevant documentation.
- As in Microsoft policy, follow certain constraints related to key strength and hashing algorithms (for example RSA or ECC key size, or only using SHA-1 under specific circumstances), and do not issue certificates with certain forbidden characteristics.

[9]Current version is 2.6.1.

- Satisfy with the requirements in any Intermediate CAs capable of issuing new digital certificates.
- Proceed with promptly certificate revocation.
- Inform Mozilla about any material change in the CA, like ownership transferal.

After the inclusion of a new root certificate, it is provided to the users in the next release of NSS, specifically located in the /usr/lib/firefox/libnssckbi.so file with the other root certificates[10].

Mozilla will determine the inclusion of a Root or Intermediate CA digital certificate in its store through a public process. However, if there is enough evidence of riskily practices by the CA, Mozilla can deny a particular inclusion, or remove the CA certificate from the store, at its sole discretion.

As of February 2019, the program has 57 CA participants [38] and 155 root certificates [39].

Interesting, Mozilla and Microsoft (including Chrome some months ago) have developed a common CA and root certificate database. The goal is to centralize the communication between the CAs and both programs [40]. (Although a CA/Browser Forum report states that the members are Mozilla, Microsoft, Google, Cisco and Apple[11]).

5) Opera: Prior to 2013, Opera had a root certificate program as well, however, it was discontinued. After that date, Opera started to use the root certificate store of the operating system plus Chrome revocation information[12], or the NSS provided by Mozilla[13].

6) Tor and Brave: The Tor Browser is a very specialized web browser focused on security and privacy. With more than 3.8×10^9 internet users[14], the roughly 2.0×10^6 Tor users (Tor clients actually) is orders of magnitude lower than the other browsers[15]. However, due to its renown in the computer security sphere and its commitment to security, it is an appropriate object of study for this analysis.

Tor is built on Mozilla Firefox (almost 95% is Firefox code[16]). Each Tor release is based on a previous Mozilla Firefox ESR (Extended Support Release) release, with some patches, preferences and functionalities specific to Tor added by the developers[17]. Firefox ESR release life cycles are longer than the common version, but these releases are more stable, hence that choice.

Consequently, Tor relies on Mozilla's NSS to manage the root certificates in its web browser. Nevertheless, to avoid

[10]https://wiki.mozilla.org/NSS:Root_certs
[11]https://cabforum.org/wp-content/uploads/CAB-ForumLondon-June-2018-BrowserNews.pdf
[12]https://cabforum.org/wp-content/uploads/CAB-Forum-2018-10-17-Opera-Root-Program-Update.pdf
[13]https://cabforum.org/browser-os-info/
[14]https://www.statista.com/statistics/273018/number-of-internet-users-worldwide/
[15]https://metrics.torproject.org/userstats-relay-country.html
[16]https://blog.torproject.org/tor-heart-firefox
[17]https://www.torproject.org/projects/torbrowser/design/

Electronic copy available at: https://ssrn.com/abstract=3425554

disk usage preventing data leakage, Tor stores intermediate certificates in memory only.

Another privacy-focused, and thus potentially of interest, web browser is Brave. Upon initial evaluation we found that its root program decisions were mostly based on the operating system. On Windows it relies on Microsoft, for Apple on the Apple program, and for Linux on NSS. This is the same behavior as any browser based on The Chromium Project.

7) Overview: The programs examined here have more similarities than differences. In this summation, we distinguish how the programs deal with core PKI challenges.

The summary of the findings is displayed in Table I. For example, Google does not maintain a store of root certificates. Also, Google demands the usage of Certificate Transparency. Microsoft and Mozilla have a more detailed Root Program Policy. In addition, the latter goes through public reviews during the process of inclusion of CAs certificates in its program. On the other hand, all programs share the CA's requirement to comply with the Baseline Requirements and the existence of frequent audit reports.

	Microsoft	**Apple**	**Google**	**Mozilla**
Store	MS Store	Apple Store	Uses OS's store or NSS	NSS
Extensive Root Program Policy	Yes	No	No	Yes
CAs must follow CA/Browser Forum Baseline Requirements	Yes	Yes	Yes[18]	Yes
Audit requirements for CAs	Yes	Yes	For EV	Yes
CA transfer of ownership disclosure	Yes	Yes	Yes[18]	Yes
CA must provide final report after incident	Yes	No	Yes	Yes
Certificate Transparency mandatory requirements	No	No	Yes[19]	No
Public CA review and inclusion decision	No	No	No	Yes[20]
Government CA restrictions	Yes[21]	No	No	No
Accepted root certificates	388[22]	178[23]	N/A	155[24]

TABLE I
COMPARISON BETWEEN MAIN ROOT PROGRAMS.

The three most widely used Programs have similar technical requirements for the root certification inclusion. Also, the same operational guidelines must be followed, and the same audits must be passed by the CA in order to get accepted. Finally, any Program has the right to remove a root certificate from its managed store at its sole discretion if it observes a misbehavior in the CA or misuse of the root certificate. The Programs uniformly and firmly claim that the inclusion of a given root certificate must bring a benefit for the end users of the web browser.

In our research, we mostly relied primarily on the decisions of Mozilla's Root Program for our insights into PKI. Our data compilation is grounded in Bugzilla as described in Section V. In addition, we examined Google's Root Program for additional information and evidence in cases of severe CA misbehaviors.

B. Guidelines for Certificate Authorities

Formed in 2005, the *Certification Authority Browser Forum* (CA/Browser Forum) is an organization that designs and publishes guidelines regarding the issuance and management of digital certificates (specifically X.509 v3). Its members are representatives of CAs, web browser vendors, OSs developers and other software companies that shape the PKI ecosystem. Table II shows the 51 CAs members. Many of these will be studied in Section VI in our incidents' analysis. Additionally, in Table III the 8 internet browser vendors members are included. As it was seen in Section III-A, some of these have their own CA Root Program.

The guidelines of the organization focus on the technical aspects of the digital certificates for TLS server web authentication [41]. In addition, the Forum addresses procedural requirements (Baseline Requirements, BR) for certificate authorities that issue public (non-internal) TLS web server authentication certificates [30]- [42]. Finally, it sets standards for EV certificates for secure connections [28] and code-signing [43].

The Forum has been publishing EV guidelines since 2007, with the last version (number 1.6.8) being effective since 2018. Baseline Requirements date from 2011, with the last version being 1.6.3 from 2019. These were the latest versions and thus the ones included at the time of this study.

Root Programs demand that Certificate Authorities comply with these guidelines and requirements. However, these are only minimal standards of security and good practices, therefore, a CA is expected to have a better infrastructure, procedures, and controls that the stated in these documents.

The level of alignment and compliance with the guidelines and requirements are testified by independent external auditors to the CA, following three alternatives: *WebTrust for Certification Authorities Principles and Criteria* [44], *European Telecommunications Standards Institute (ETSI) standards ETSI TS 102 042 or ETSI TS 101 456* [37], or the *ISO standard 21188:2006 Public key infrastructure for financial services – Practices and policy framework* [45] (although there is a newer version, ISO 21188:2018). These statements must be public and frequent.

Although it is expected the compliance with the CA/Browser Forum's requirements (which are developed and voted by the CAs themselves), in Section VI we will analyze an alarming collection of past incidents, where CAs failed to follow them. Moreover, we discovered issues in auditors' reports and practices, which, given the role they play, is utterly worrying.

[18]In these cases, Google demands it since the policy of any store used in the device will demand it.

[19]All CAs for EV certificates and selected CAs for non-EV certificates.

[20]Mozilla maintains the right to decide for no inclusions anyways.

[21].gov domains and domains under sovereign control.

[22]11-2018.

[23]02-2019.

[24]02-2019.

Electronic copy available at: https://ssrn.com/abstract=3425554

Actalis S.p.A.	ComSign	**GoDaddy**	SecureTrust
Amazon Trust Services	D-TRUST GmbH	**Hellenic ARICA**	Sectigo
ANF Autoridad de Certificación	DigiCert	**Izenpe**	Shanghai Electronic CA Center
AS Sertifitseerimiskeskus	Digidentity	**Kamu Sertifikasyon Merkezi**	Skaitmeninio sertifikavimo centras
Buypass AS	Disig	**KPN Corporate Market**	SSL.com
Camerfirma	DocuSign	**Let's Encrypt**	Swisscom
Certinomis	E-TUGRA	**Logius PKIoverheid**	SwissSign AG
CERTIGNA	eMudhra Technologies	**National Center for Digital Certification**	TAIWAN-CA
certSIGN	Entrust	**Network Solutions**	TrustCor Systems
Certum	ESG de Electronische Signatuur	**Open Access Technology International**	TURKTRUST
China Financial Certification Authority	Firmaprofesional	**Prvni certifikacni autorita**	Visa
Chunghwa Telecom Co	Global Digital Cybersecurity Authority	**QuoVadis**	Wells Fargo
China INIC	GlobalSign	**Secom Trust Systems**	

TABLE II
CERTIFICATION AUTHORITIES MEMBERS OF THE CA/BROWSER FORUM

Qihoo 360 Technology Co Ltd	Google Inc
Apple Inc	Microsoft Corporation
Cisco Systems Inc	Mozilla Foundation
Comodo Security Solutions	Opera Software AS

TABLE III
INTERNET BROWSER SOFTWARE VENDORS MEMBERS OF THE CA/BROWSER FORUM

IV. RESEARCH QUESTIONS

The three inter-related questions in this study are about the types, causes, and nature of failures in certificate issuance.

1) What are the most common types of incidents related to CAs in the PKI network?

We wanted to understand the incidents based on their different types and severity. To be included in our study, these incidents must have been generated by CAs.

2) What are the causes of these incidents? What led or allowed them to happen?

Our research would be incomplete without trying to identify the causes that generated the incidents in our compiled dataset.

3) Are these failures in PKI technical errors in the CAs, or is there evidence of incompetence, misbehavior, or ethical lapses?

We wanted to determine if these incidents are merely software or hardware failures that have repercussions in PKI. Alternatively, are there patterns of decision-making in the CAs that may undermine the present and future trust in PKI?

V. RESEARCH APPROACH

In this section we present the methods used in our research, from the data gathering to the analysis.

A. Governance background

First, we detail the different root programs in the industry and their inclusion requirements, linking that information with the most used web browsers. The goal is to understand the current governance structure and the role operators have in the PKI. Therefore, we also surveyed the *CA/BR Forum* guidelines to understand their requirements. Here we described the data compilation in our investigation of how and why CAs fail to comply with the published standards.

B. Data collection

For the data collection, our main source was *Bugzilla*[25]. From that public source we got 348 of our 379 problematic incidents related to the CAs, after an investigation of over 1300 reported "bugs" that the previous query returned to us. There was no sample taken, and every "bug" was thoroughly studied to check its validity and relevance for our research. For example, we discarded "bugs" that were not related to CAs' incidents or where there was no consensus or enough evidence about the failure or not on the CA side. Also, it is important to note that a "bug" may be related to several problematic certificates, therefore, the number of times a Problematic CA appeared in our incidents' collection was a good proxy to get an idea of the number of incidents that were detected for that CA, however, this quantity is not the same that the actual number of, for example, digital certificates miss-issued by that CA. The time period of these incidents ranges from 2008 to 2019 (the deadline for our incidents collection was the beginning of February/2019), with an additional incident coming from 2001. No incidents were reported via Bugzilla or any other data source used between 2002 and 2007.

We found *Bugzilla* to be well suited for our research goals, as the cause of the vulnerability is usually included in the report. The data are also reliable. In order to collect complementary information for each bug, we also analyzed `https://groups.google.com/forum/#!forum/mozilla.dev.security.policy`, where is more suited to discussions than the proper bug entry, and `crt.sh` where we could find the miss-issued digital certificates (which are taken from "Certificate Transparency" logs).

In addition to that source, we used `https://`

[25]The exact query we used was: `https://bugzilla.mozilla.org/buglist.cgi?component=CA\%20Certificate\%20Compliance\&component=CA\%20Certificate\%20Root\%20Program\&component=CA\%20Certificates\%20Code\&product=NSS\&query_format=advanced\&resolution=---\&resolution=FIXED\&resolution=INVALID\&resolution=WONTFIX\&resolution=INACTIVE\&resolution=DUPLICATE\&resolution=WORKSFORME\&resolution=INCOMPLETE\&resolution=SUPPORT\&resolution=EXPIRED\&resolution=MOVED\&order=priority\%2Cbug_severity\&limit=0`

Electronic copy available at: https://ssrn.com/abstract=3425554

`security.googleblog.com/` and `https://blog.mozilla.org/security/` to collect more incidents or to gather further information about the most notorious failures.

Finally, a search of CA incidents was conducted over the internet using common search engines, but only adding the problems that came from trusted sources as public disclosures from CAs, academic research or presentations by industry. We avoided media articles without proper evidence for rigor.

Having collected 379 incidents, we consider this number big enough to perform correctly a qualitative study of PKI failures, incidents and misbehaviors, and to draw valid conclusions about this problematic based on our findings.

C. Data classification

For each incident, we collected the following information when possible. First, the **year** the incident was reported, which in general equals to the year the incident happened. Sometimes, a common incident in a CA lasted for several years. In that case, we used the year when the incident was reported. Second, the problematic root or intermediate CA, RA, reseller or auditor, *in our analysis they will be known as* ***Problematic CAs***, and the root CA where the problematic entity rooted or it was related, *in our analysis they will be known as problematic* ***Root CAs***. Third, the **kind** of incident for that entry, for example: "1024 bits key", "Erroneous/Misleading Audit report", "Fields in certificates not compliant to BR", "Non-BR-compliant or problematic OCSP responder or CRL" and "Possible issuance of rogue certificates". Also the **cause** of the incident, whenever it was stated, for example: "Software bugs", "Believed to be compliant/Misinterpretation/Unaware", "Business model/CA decision/Testing" and "Operational error". Finally, if it was **a self-report** of an incident from a CA, which in that case showed a healthy behavior and transparency from the CA. In section VI, the full information collected can be appreciated.

D. Data analysis

For the analysis of the data, a qualitative approach fitted perfectly with the kind of data collected. Once we had all the incidents collected and classified based on the different properties and attributes explained before, we created pairs of table-charts in order to highlight our findings. The amount of data collected and our posterior classification provided a very rich source for a qualitative exploration of past and present faults in PKI, and enabled the discovery of new dimensions and perspectives of the phenomenon previously hidden.

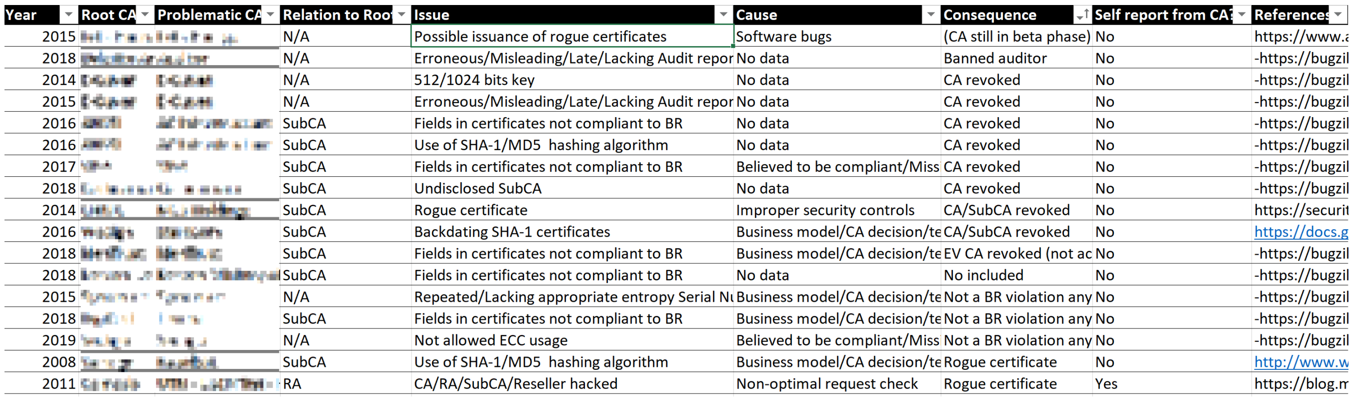
In section VI, we focused on the following topics. First, general numbers about incidents and the main **Problematic CAs** and the **Roots CAs** where they were chained to. Second, information about the different **countries of origin** of these CAs. Third, the most repeated **types** and **causes** of incidents. Following with several perspectives studying **incidents' self-reporting** numbers and a full study **year-per-year** about the Root CAs, type of incidents and their causes, with several charts to illustrate tendencies and repetitions. Finally, **rogue certificates** and their issuing CAs, rooting CAs and causes.

Further tables and charts used in our investigation can be found in Section X, Appendix 1. Some highlighted incidents of mis-practices can be found in Section XI, Appendix 2. In addition, in Section XII, Annex 1, we provide a brief history of PKI for the non-experienced reader.

We included a few Problematic CA's auditors that were investigated and even forbidden by some Root Programs. In more than one occasion, these were a problematic entity in the PKI ecosystem. Therefore, it was worthwhile including them in this analysis.

Once the incidents' nature and main causes are clear and understood, better solutions could be designed to prevent endemic or epidemic incidents.

Electronic copy available at: https://ssrn.com/abstract=3425554



Year	Root CA	Problematic CA	Relation to Root	Issue	Cause	Consequence	Self report from CA?	References
2015	[illegible]		N/A	Possible issuance of rogue certificates	Software bugs	(CA still in beta phase)	No	https://www.a
2018	[illegible]		N/A	Erroneous/Misleading/Late/Lacking Audit repor	No data	Banned auditor	No	-https://bugzil
2014	[illegible]	[illegible]	N/A	512/1024 bits key	No data	CA revoked	No	-https://bugzil
2015	[illegible]	[illegible]	N/A	Erroneous/Misleading/Late/Lacking Audit repor	No data	CA revoked	No	-https://bugzil
2016	[illegible]	[illegible]	SubCA	Fields in certificates not compliant to BR	No data	CA revoked	No	-https://bugzil
2016	[illegible]	[illegible]	SubCA	Use of SHA-1/MD5 hashing algorithm	No data	CA revoked	No	-https://bugzil
2017	[illegible]	[illegible]	SubCA	Fields in certificates not compliant to BR	Believed to be compliant/Miss	CA revoked	No	-https://bugzil
2018	[illegible]	[illegible]	SubCA	Undisclosed SubCA	No data	CA revoked	No	-https://bugzil
2014	[illegible]	[illegible]	SubCA	Rogue certificate	Improper security controls	CA/SubCA revoked	No	https://securit
2016	[illegible]	[illegible]	SubCA	Backdating SHA-1 certificates	Business model/CA decision/te	CA/SubCA revoked	No	https://docs.g
2018	[illegible]	[illegible]	SubCA	Fields in certificates not compliant to BR	Business model/CA decision/te	EV CA revoked (not ac	No	-https://bugzil
2018	[illegible]	[illegible]	SubCA	Fields in certificates not compliant to BR	No data	No included	No	-https://bugzil
2015	[illegible]	[illegible]	N/A	Repeated/Lacking appropriate entropy Serial Nu	Business model/CA decision/te	Not a BR violation any	No	-https://bugzil
2018	[illegible]	[illegible]	SubCA	Fields in certificates not compliant to BR	Business model/CA decision/te	Not a BR violation any	No	-https://bugzil
2019	[illegible]	[illegible]	N/A	Not allowed ECC usage	Believed to be compliant/Miss	Not a BR violation any	No	-https://bugzil
2008	[illegible]	[illegible]	SubCA	Use of SHA-1/MD5 hashing algorithm	Business model/CA decision/te	Rogue certificate	No	http://www.w
2011	[illegible]	[illegible]	RA	CA/RA/SubCA/Reseller hacked	Non-optimal request check	Rogue certificate	Yes	https://blog.m

Fig. 1. A glance at our database of collected incidents. (Although all the information collected in this study is publicly available, here we distorted the CA's names in order to maintain an impartial position with all the studied organizations)

VI. In Depth Incidents Analysis[26]

A. Data analysis

0) (Pre-Analysis) Issued certificates per CA:

Before starting with our analysis, we present quantities extracted from different Certificate Transparency logs, to provide an idea of the numbers of non-expired certificates issued by different CAs, and to offer a grasp of who are the "big players" in the CA market. The source of this data[27] monitors several CT logs from Cloudflare, DigiCert, Google and Sectigo.

Table IV (Fig. 2) shows the CAs that have most issued digital certificates logged in the previously mentioned source (to March/2019). These certificates are non-expired and CT-qualified. At present, *"Let's Encrypt"* is the most used CA (their digital certificates are for free). Behind it come *"Sectigo"*, *"DigiCert"* and *"GoDaddy"*; all of them with more than one million digital certificates in the queried logs. We also include several CAs with fewer certificates because many of them will be studied in the following sections of this work, given the incidents we collected related to them.

CA	#Non-expired CT cert
Let's Encrypt	92,300,644
Sectigo	27,859,495
DigiCert	12,577,372
GoDaddy	2,476,593
GlobalSign	680,249
Amazon	644,901
Certum	306,926
Starfield	258,734
GeoTrust	218,030
Actalis	203,603
StartCom	124,077
Entrust	98,405
VeriSign	71,102
Thawte	65,198
SECOM	63,365
Others	143,239

TABLE IV
CAs with most non-expired issued certificates (to March/2019)

1) General incident numbers per CA:

We begin our analysis by describing the major offenders to the PKI network. In Table V we show the Root CAs with most incidents collected. For the 379 incidents found where a Problematic CA was involved, we listed its corresponding Root CA. In 7 of these 379 incidents, the main offender was an auditor, therefore, we did not link these incidents to a specific root CA, since the auditor is the primarily responsible. In this table, we can see how *"DigiCert"* accumulates more incidents than any other Root CA, well beyond the rest. The second place is for *"Comodo"*, now rebranded as *"Sectigo"*. The third most often problematic is *"Symantec"*, which at this date has been distrusted by the major Root Programs. Its CA business was purchased by *"DigiCert"*, including in the deal other brands like *"Thawte"*, *"GeoTrust"* and *"RapidSSL"*.

In Table VI we compare the top offender Root CAs with the other offender Root CAs, where we can conclude that the top 10 offenders had almost the same number of incidents that the other 67. The reason behind this situation could be that these Root CAs have several intermediate CAs, Resellers and Registration Authorities (RAs) linked to them, and/or their practices are not optimal. Especially their security controls, compliance with the Baseline and Root Programs Requirements and monitoring of related entities (mainly subordinate CAs).

Table VII details the CAs with most incidents studied. These are mostly Root CAs or Intermediate CAs, but in addition to these entities, we found that Resellers and RAs are a source of incidents as well, besides the Auditors. Three of the four most problematic CAs -*"StartCom"*, *"WoSign"* and *"PROCERT"*- had their signing digital certificate revoked from the Root Programs, given the number of incidents related to them and their lack of an adequate response. Also, *"VISA"* was largely investigated by Mozilla, but then this Root CA removed its certificate from Mozilla's Root

[26]During the analysis the reader must remember the distinction explained between the **Problematic CAs** (the CA source of the incident) and the **Root CAs** (the CA where the problematic CA was rooted to).

[27]https://ct.cloudflare.com/ (Last access on 03/20/2019).

Electronic copy available at: https://ssrn.com/abstract=3425554

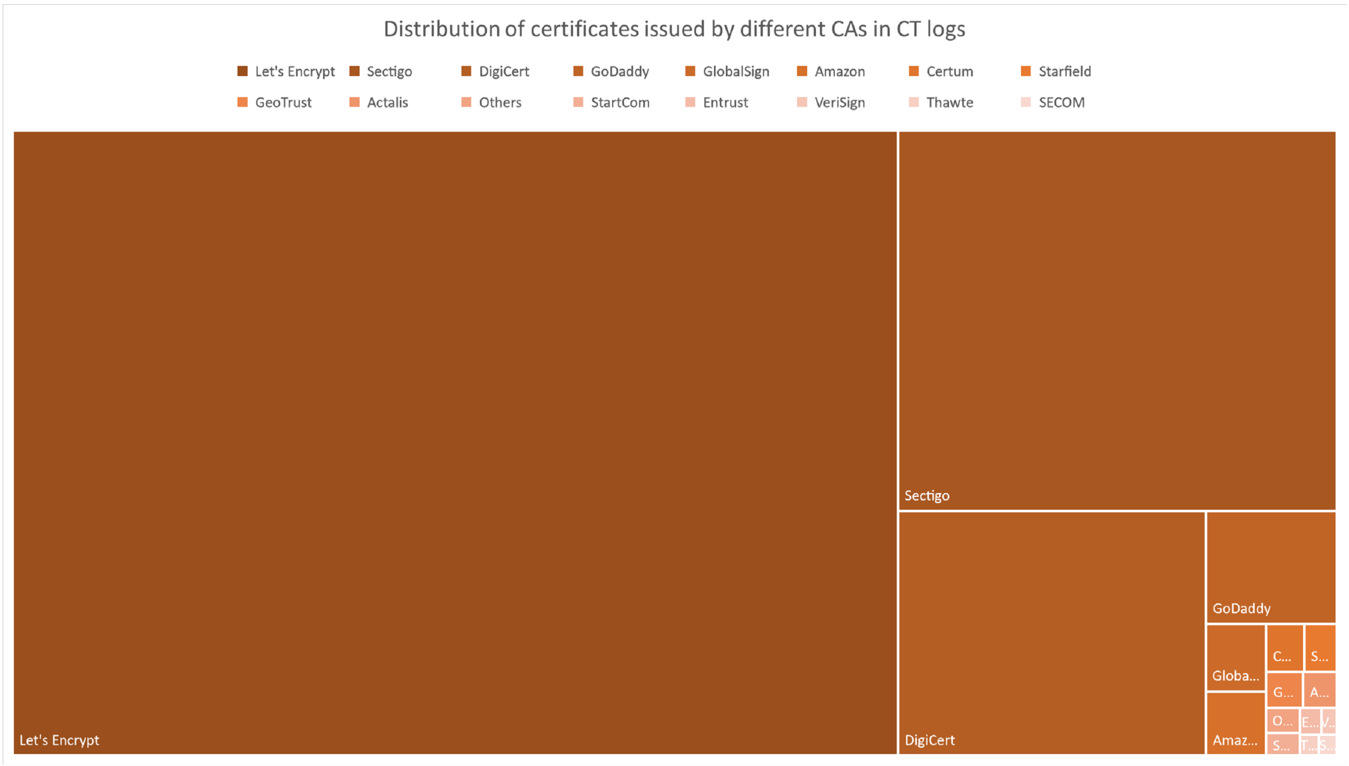


Fig. 2. CAs with most non-expired issued certificates (to March/2019)

Top Root CAs	#Incidents
Certinomis	11
StartCom	11
GlobalSign	11
SNCE Venezuela	12
QuoVadis	13
Camerfirma	17
WoSign	17
Symantec	18
Comodo	21
DigiCert	60

TABLE V
ROOT CAS WITH MOST INCIDENTS COLLECTED

	#Incidents	#CAs
Top Root CAs	191	10
Others	188	67

TABLE VI
TOP ROOT CAS INCIDENT REPORTERS VS THE REST REPORTING CAS

Top Problematic CAs	#Incidents
SwissSign	8
Camerfirma	8
Certum	8
GoDaddy	8
VISA	9
QuoVadis	9
Comodo	11
PROCERT	12
DigiCert	12
WoSign	15
StartCom	16

TABLE VII
PROBLEMATIC CAS WITH MOST INCIDENTS COLLECTED

Program.

If we compare the concentration of incidents linked to Root CAs and the same concentration in Problematic CAs, we can appreciate in Table VIII that in the latter set the incidents are more evenly distributed.

	#Incidents	#CAs
Top Problematic CAs	116	11
Others	263	131

TABLE VIII
TOP PROBLEMATIC CAS INCIDENT REPORTERS VS THE REST REPORTING CAS

Electronic copy available at: https://ssrn.com/abstract=3425554

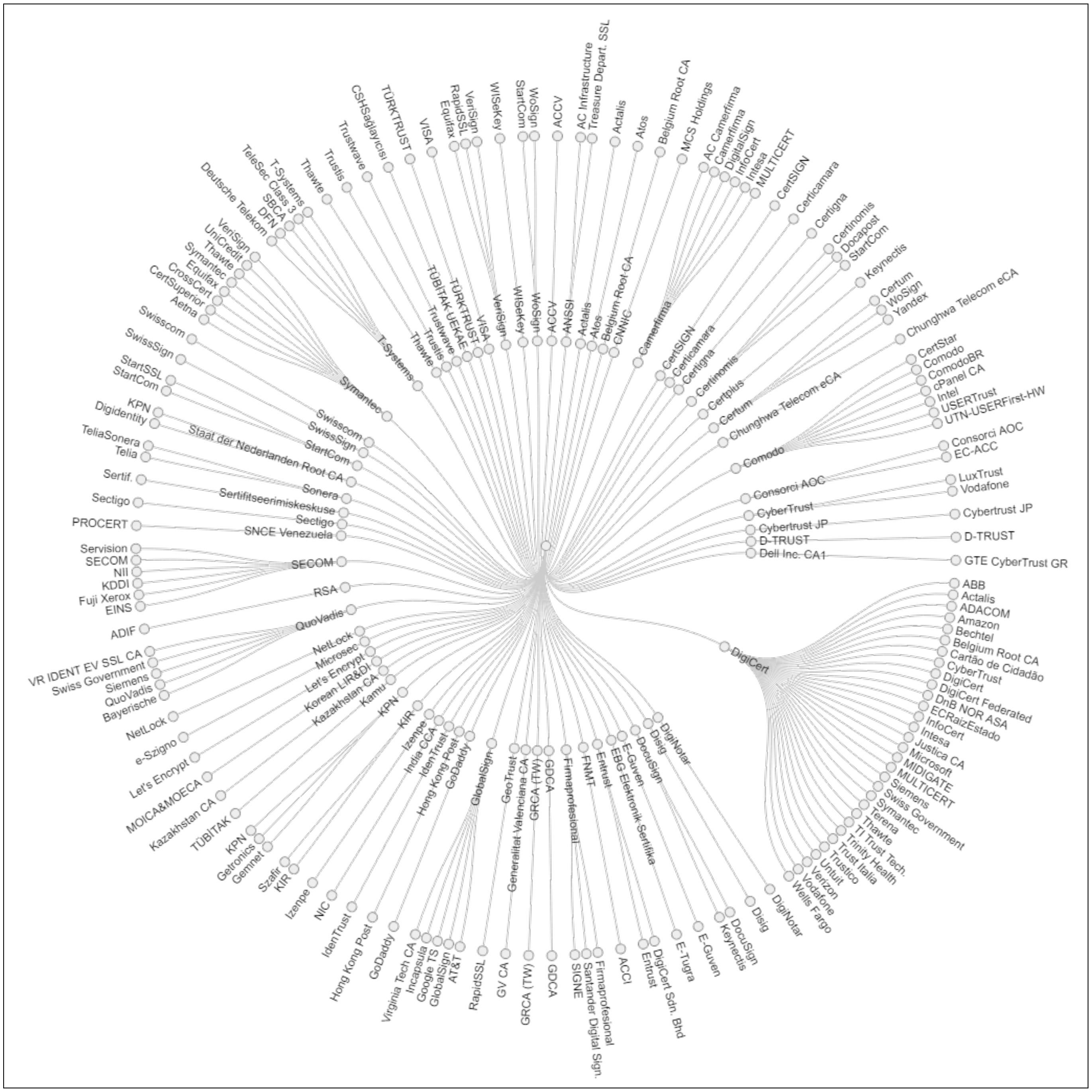


Fig. 3. Relation between the Problematic Root CAs (in the centre) and the Problematic CAs.

Electronic copy available at: https://ssrn.com/abstract=3425554

2) *CAs' countries of origin:*

Our second focus is on the problematic Root CAs' country of location. In Table IX we observe that our Root CAs belong to exactly 29 different countries. By far, the most common country was US, with 12 cases. In second place comes Spain with 7. Following these, France and Turkey were home to 5 problematic Root CAs each.

Country	#CAs
Belgium, Bermuda, Canada, Colombia, Estonia, Finland, Hong Kong, India, Ireland, Italy, Kazakhstan, Korea, Romania, Slovak Republic, South Africa, Venezuela	1
Hungary, Japan, Poland, Taiwan	2
Germany, Netherlands, Switzerland, UK	3
China	4
France, Turkey	5
Spain	7
USA	12

TABLE IX
NUMBER OF ROOT CAS WITH INCIDENTS PER COUNTRY

Only a handful of countries have problematic Root CAs. However, examining the countries of origin of present Root CAs in Mozilla's Root Store, it can be seen that the countries of origin of the CAs have not changed (details can be found in Section X, Appendix 1). Thus, the past incidents and the resulting revocation of Root CAs from Root Programs have not had a huge impact on this aspect of the problem. For example, no case has been considered sufficiently drastic that a country would be prohibited from having root CAs in Root Programs despite the use of rogue CAs in politically motivated incidents.

Finally, there may be an issue of possible bias in that US CAs may be subject to more scrutiny. It is also possible that the historical distribution of servers and business causes this concentration.

3) *Types of incidents:*

In relation to the type of incidents we found, we detailed our findings in Table X. This table shows the type of incident, the number of incidents that were not reported (disclosed by the CA itself), the total number of incidents for a given type, and the percentage of incidents in this category.

We believe that self-reporting any incident is a healthy practice for a CA, and shows mature internal processes and management. Nevertheless, the figures related to self-reporting are disappointingly low. There are several cases where self-reporting is not feasible; for example, the CA may be unaware of the problem. Conversely, there were many cases in our study where the CAs knew about their issues and preferred to remain silent. The categories that we chose to classify the incidents are, primarily, self explanatory and not new for readers familiar with X.509. By far, the most common problem is that the certificate attests to incorrect information in a certificate field.

Specially, for the category *Other* we included the following less frequent incidents:

- Backdating SHA-1 certificates
- Certificates for malicious domains
- Charging for compromised cert revocation (HeartBleed)
- CPS non-compliance
- Debian OpenSSL Vulnerability
- Delayed certificate revocation
- Digital certificate for non-existent domain/Bad information of requester
- MITM attempt
- No BR self-assessment
- No disclosing another CA purchase
- No test website for CA
- Non-acceptable requester validation
- Not allowed ECC usage
- Registering competitor trademark
- Self revocation
- Timestamp certificate from root
- Un-revoking certificates
- Validity greater than 825 days

After the over one-third of the incidents related to fields in the certificates, revocation failures dominate. OCSP responders or CRLs that are non-compliant with the Baseline Requirements, especially the former, are in this category. While there is no sign that these incidents are related to unethical or dishonest practices by the CAs, it may show that their issuing practices and alignment with the Baseline Requirements have room for improvement. In terms of our original research question, this is evidence of incompetence.

Based on our reading of the data, we believe that the one reason for the number of self-reported incidents related to

Electronic copy available at: https://ssrn.com/abstract=3425554

fields in digital certificates non-compliant with the Baseline Requirements is the introduction of Certificate Transparency. Specifically, if fields result from incompetence or inadvertent issuance, before Certificate Transparency each error was handled privately. In addition to that transparency database, the security community has developed open source tools, called *lints*, that verify compliance (i.e., *ZLint*[28], *CABLint*[29], and *X509Lint*[30]). Using lints, CAs can easily identify their issuance of certificates with incorrect fields, so they can be addressed on time. Besides increasing the ease of detection, the threat of public posting may create an incentive to correct a certificate before researchers locate it. Hopefully, the availability of lints will improve CA practice. Of course, it helps that these incidents (incorrect fields) are not severely penalized by the Root Programs Owners, unless they are a constant problem in a CA. The result is lower cost, positive incentives, and no disincentive for correcting fields.

Incident	#No	Total	Percentage
Fields in certificates not compliant to BR	112	146	38.52%
Non-BR-compliant[31]or problematic OCSP responder or CRL	33	39	10.29%
Erroneous/Misleading/Late/Lacking Audit report	24	25	6.60%
Repeated/Lacking appropriate entropy Serial Numbers	19	22	5.80%
Undisclosed SubCA	15	19	5.01%
512/1024 bits key	16	18	4.75%
Possible issuance of rogue certificates	13	18	4.75%
Use of SHA-1/MD5 hashing algorithm	13	15	3.96%
CAA[32]mis-issuance	12	14	3.69%
Rogue certificate	12	12	3.17%
CA/RA/SubCA/Reseller hacked	8	11	2.90%
Other	35	40	10.55%

TABLE X
MOST REPEATED TYPES OF INCIDENT, NUMBER OF NOT-REPORTED INCIDENTS, AND PERCENTAGE OF EACH TYPE OF INCIDENT COMPARED TO THE TOTAL

[28]https://github.com/zmap/zlint

[29]https://github.com/awslabs/certlint

[30]https://github.com/kroeckx/x509lint

[31]*BR* are the CA/Browser Forum's *Baseline Requirements*.

[32]DNS Certification Authority Authorization (CAA) Resource Record, RFC 6844.

4) The causes of incidents:

After studying the different types of incidents we classified the causes of these incidents. As summarized in Table XI and enumerated below, we found ten repeated categories, some reports with no data, and nine that did not fit into our categories. This table lists the major causes of incidents, the rate of self-reporting for each cause, and the percentage of each cause in relation to the total number of incidents studied. The causes were categorized as follows.

- *Software bugs*: In this case, the cause is a software error or flaw that generated an incident in the PKI ecosystem. In this case, these bugs may produce faulty certificates, problems in OCSP responders, or erroneous checks in a request.
- *Believed to be compliant/Misinterpretation/Unaware*: These causes show a lack of awareness of or failure to comply with updates of the Baseline or the Root Program Requirements, or misunderstanding of the concepts (technical or not) behind these requirements.
- *Business model/CA decision/Testing*: In this group, we classified the incidents that share a common pattern. The CA is aware of the Baseline or Root Program Requirements, but places its own business strategies over compliance with these requirements. In other words, they prefer their near-term benefits to the health of the PKI ecosystem. Examples of these are backdating SHA-1 certificates in order to evade its prohibition, charging for the revocation of compromised digital certificates, selling certificates for Man-in-the-Middle (MITM) attempts, and the potential (or actual) issuance of rogue certificates. It goes without saying that this category presented the most alarming incidents with regarding CAs' misbehaviors or lack of ethics.
- *Human error*: This category is self-explanatory. These correspond to human mistakes in manual entry of data for certificate requests, or forgetting steps in the setup of a new intermediate CA. This is the cause when one specific unique employee makes an error in a process. In one particular case, there was an erroneous self revocation of a Sub CA[33], although this is not a frequent mistake.
- *Operational error*: This included all the incidents that were generated because of an internal faulty procedure in the CA or a related entity. It could be argued that these overlap the incidents included in the "Human error" category. We distinguish these between mistakes in manual one-off processes and errors in the operational processes that could not result from a specific identifiable employee's manual error. Examples of these incidents are mistakes in audit reports, not disclosing subordinate CAs, and delayed revocations of compromised certificates.
- *Non-optimal request check*: This cause refers to cases

[33]2016-GlobalSign, https://downloads.globalsign.com/acton/fs/blocks/showLandingPage/a/2674/p/p-008f/t/page/fm/0

Electronic copy available at: https://ssrn.com/abstract=3425554

where the checks of an applicant for a certificate were not performed correctly. This can, for example, enable the generation of rogue certificates, EV certificates that have not been verified, or, simply, improper digital certificates. This category does not include cases related to "software bugs" or "human error", but only certificate requests that were not performed properly. An example is relying on pre-defined email accounts from the requester like "ssladministrator@...", "ssladmin@...", "info@..." and "admin@...", which are not always secured by the email provider.

- *Improper security controls*: Under this category, we grouped all the incidents regarding CAs or other entities being hacked, with the possible or actual outcome of rogue certificates being generated and used in the wild as a result. While it would be unreasonable to blame a CA that is the victim of an exploitation of a zero-day vulnerability in its platform, evidence shows that, in general, the causes of the hacking incidents were predictable known problems.
- *Change in Baseline Requirements*: In this category, there are a few incidents where a sudden change in the BR made the CAs change their certificate issuance. Here CAs were not compliant until they updated.
- *Infrastructure problem*: Infrastructure problems can be related to unavailable servers, defective networks, or problems in the hardware. This category is for failures of technical components that support the business of the CA.
- *Organizational constraints*: This category includes incidents where the causes were constraints in the environment where the CAs operated, for example, national legal requirements. For instance, some countries require government agencies domains to have digital certificates provided by a national public CA, and this national public CA is obligated to follow governmental requirements (as using a particular non-BR-complain encryption algorithm). Export control regimes requiring weak certificates to non-US entities would be an example of a requirement that weakens PKI. Thus, Root Program owners must be flexible enough with the requirements in these particular cases and accept some exceptions.
- *Other*: In this category, we included less repeated causes, like "Misconfigured software" and "Misisuances not in sample provided to auditors", in addition to three incidents that were not closed at the time of writing this work.
- *No data*: The incidents under this category did not provide enough information to identify their causes.

Table XI shows that almost one-quarter of the incidents are caused by *Software bugs*. The second most frequent cause is the CAs' "lack of awareness" regarding a specific requirement. If it is a repeated source of incidents in a particular CA, it may show a lack of mature internal practices; however, this cause could be used as a "scapegoat" by a CA, trying to justify an incident. It is reasonable to believe that the numbers in this category are a maximum possible estimate as much as a correct count. However, the most alarming information from this table is that in 52 incidents, almost 14% of the total, the CAs were aware of their misalignment with the requirements but they preferred to implement an unaligned, more profitable business model. This dismissal of the requirements can lead to unexpected incidents with a systematic negative impact on the welfare of the PKI users.

Cause	#No	Total	Percentage
Software bugs	67	91	24.01%
Believed to be compliant, Misinterpretation, Unaware	61	69	18.21%
Business model, CA decision, Testing	47	52	13.72%
Human error	28	37	9.76%
Operational error	23	29	7.65%
Non-optimal request check	20	24	6.33%
Improper security controls	13	15	3.96%
Change in Baseline Requirements	1	7	1.85%
Infrastructure problem	6	6	1.58%
Organizational constraints	6	6	1.58%
Other	9	9	2.11%
No data	32	35	9.23%

TABLE XI
MOST REPEATED CAUSE OF INCIDENT, NUMBER OF NOT-REPORTED INCIDENTS, AND PERCENTAGE OF EACH CAUSE OF INCIDENT COMPARED TO THE TOTAL

Regarding self-reported incidents, *Software bugs* and *Change in Baseline Requirements* are more prone to be reported by the faulty CA than incidents caused by CAs' "lack of awareness" (CAs cannot report what they do not know) or "planned lack of alignment to the requirements" (the consequence of this could be too harsh for the CA).

5) *Types of incidents by cause:*

Table XII provides a summary of the primary sources of PKI incidents. It illustrates that the most common errors in certificates result from problems that instantiate as incorrect fields of the issued digital certificates. The primary reported source of these errors is faulty CA software. In addition, bugs have also created flaws in the OCSP responders and CRLs, and in the CAA checks before the issuing of the certificates.

The other major source of flawed certificates is that the CA makes an error in the issuing process. Again, these are often excused by the CA as being a result of simple unawareness of the compliance requirements. Sometimes the lack of compliance is undeniably known by the CA, since incidents of this type happened due to particular business models and commercial decisions in a CA. There are certificates issued erroneously from processes that were intended for testing purposes, where an operational certificate is mistakenly issued. Business decisions are causes of problems for a number of incidents coded into the *"Other"* category as well. The final common cause is human error, where people simply enter the wrong information in a field.

The CAs' unawareness about their lack of compliance results not only in problematic certificate fields. Other results from the lack of compliance have included faulty OCSP

Electronic copy available at: https://ssrn.com/abstract=3425554

responders and incorrect CRLs. In addition, CAs have failed to notify Root Program Owners about the creation of intermediate CAs through the generation of additional signing certificates.

Another inference that we can make from the data is that the incidents related to these organizations being hacked are primarily the result of inadequate security measures. Despite the Root Program Owners' requirements for operational and technical security, CAs have not proven highly resilient.

Finally, we can see that problems related to audit outcomes rarely provide adequate public information to determine their origin. These are categorized as *"No data"* as their cause in our study. As a result, a customer seeking a certificate cannot know which CA has experienced the fewest problems.

All these observations are summarized in Fig. 4.

Electronic copy available at: https://ssrn.com/abstract=3425554

Type \ Cause	Believed to be compliant/ Misinterpretation/ Unaware	Business model/ CA decision/ Testing	Change in BR	Human error	Improper security controls	Infrastructure problem	Non-optimal request check	Operational error	Organizational constraints	Software bugs	Other	No data
512/1024 bits key	0	7	0	1	0	0	0	2	0	2	0	6
CA/RA/SubCA/Reseller hacked	0	0	0	0	10	0	1	0	0	0	0	0
CAA mis-issuance	3	1	0	0	0	0	0	0	0	9	0	1
Erroneous/Misleading/Late/Lacking Audit report	5	2	0	0	0	0	0	1	1	0	3	13
Fields in certificates not compliant to BR	27	14	5	23	0	0	9	7	3	46	2	10
Non-BR-compliant or problematic OCSP responder or CRL	12	2	1	3	0	4	0	3	0	12	1	1
Possible issuance of rogue certificates	1	1	0	0	2	0	7	0	0	7	0	0
Repeated/Lacking appropriate entropy Serial Numbers	3	5	1	3	0	2	0	4	0	4	0	0
Rogue certificate	0	1	0	0	2	0	7	0	0	2	0	0
Undisclosed SubCA	9	0	0	1	0	0	0	8	0	0	0	1
Use of SHA-1/MD5 hashing algorithm	3	3	0	5	1	0	0	0	0	1	0	2
Other	6	16	0	1	0	0	0	4	2	8	2	1

TABLE XII

TYPES OF INCIDENTS BY CAUSE

Electronic copy available at: https://ssrn.com/abstract=3425554

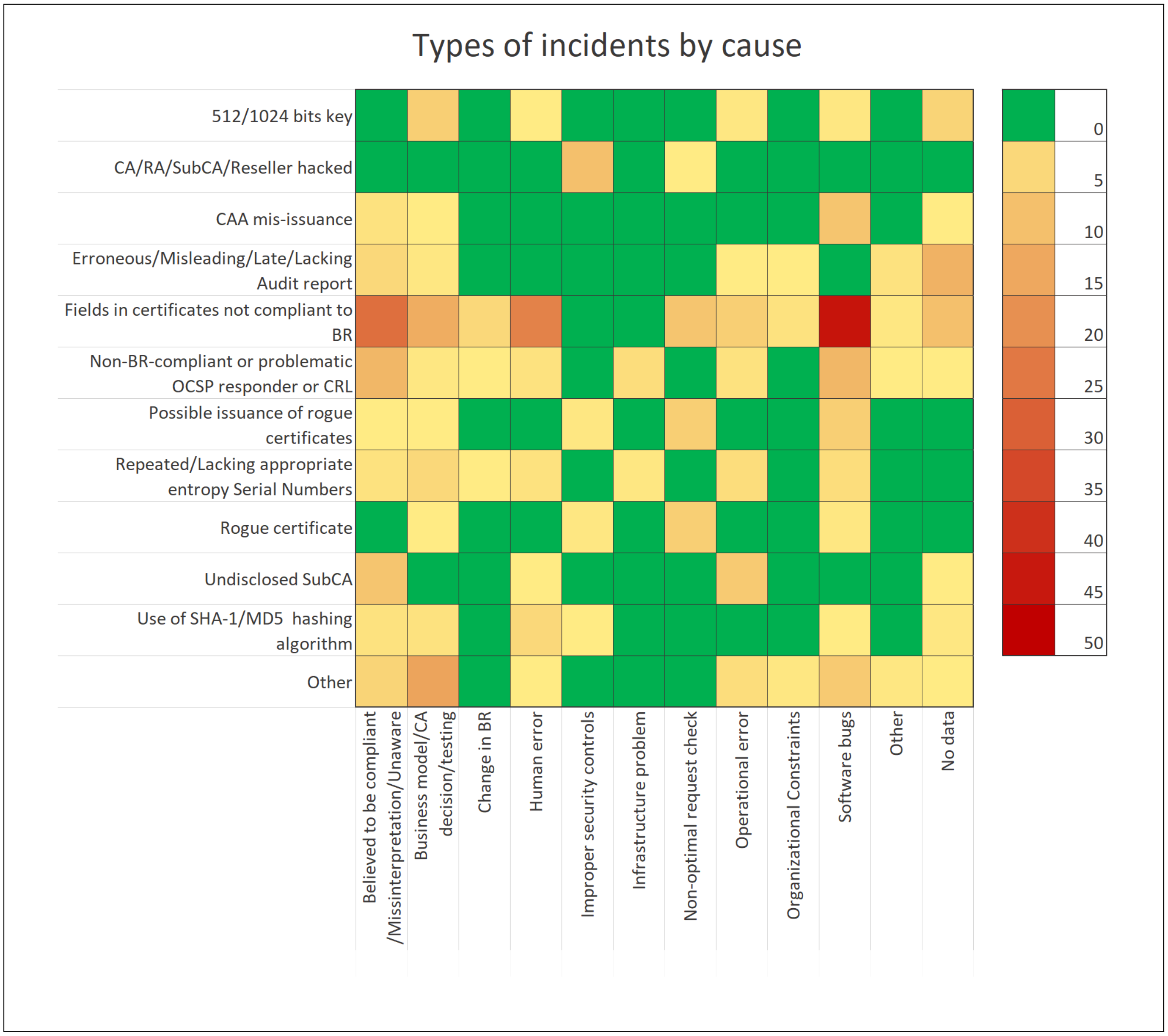


Fig. 4. Frequency of types of incidents by cause, where the type of the incident is the vertical axis and the cause of the incident is on the horizontal axis.

Electronic copy available at: https://ssrn.com/abstract=3425554

6) CAs' reporting practices:

It is not mandatory to provide an incident report after any incident, mis-issuance or problem in general in the CA. Yet this may be seen as a good indicator of the CAs transparency, practices, mature procedures, and, last but not least, its commitment to the global welfare of the PKI network.

We found that providing a public incident report is not a common practice for most of the CAs. However, there are a few CAs that have the good habit of providing them.

These are only 18 out of 77 Root CAs (and auditors) studied, which is a fairly low percentage (23.38%). Interestingly enough, there are 19 Root CAs (24.67%) that had more than 3 incidents in our study that have never submitted an incident report by themselves without being enforced or alerted by a third party.

To better compare the CAs that have reported at least 1 incident in our study, we added Table XIII.

There we can see that *"Entrust"* and *"KIR"*[34] have a 75.00% of self-reporting, 6 in 8 cases and 3 in 4 cases, respectively. Behind them, with percentages slightly above 55.00% come *"Let's Encrypt"* (4 in 7) and *"SwissSign"* (5 in 9). *"Certigna"* has a 50%, but only 2 studied incidents. Finally, above 35% we have *"QuoVadis"* (38.46%) and *"DigiCert"* (36.67%). In the case of *"DigiCert"*, it is the Root CA that has provided more self-reports after incidents, with 22 in 60 cases. Fig 5 highlights this information.

Root CA	#No	#Yes	Total	Percentage
Camerfirma	16	1	17	5.88%
Certigna	1	1	2	50.00%
Certum	7	3	10	30.00%
Comodo	18	3	21	14.29%
Consorci AOC	2	1	3	33.33%
DigiCert	38	22	60	36.67%
D-TRUST	3	1	4	25.00%
Entrust	2	6	8	75.00%
GlobalSign	8	3	11	27.27%
GoDaddy	6	2	8	25.00%
KIR	1	3	4	75.00%
Let's Encrypt	3	4	7	57.14%
QuoVadis	8	5	13	38.46%
Sonera	5	1	6	16.67%
SwissSign	4	5	9	55.56%
Symantec	14	4	18	22.22%
Trustwave	2	1	3	33.33%
T-Systems	7	1	8	12.50%

TABLE XIII
ROOT CAS AND SELF-REPORTING COMPARISON (INCLUDING ONLY CAS THAT HAVE SELF-REPORTED AT LEAST ONE INCIDENT)

To conclude, note that Table XIV offers a quick comparison between the set of self-reporting CAs and the set that has never self-reported an issue. The former, those that self-report, has 18 CAs against the 59 that form the latter. The fact that self-reporting CAs have 212 total incidents versus the 167 in others may result from bias in public data; i.e., some CAs do not self-report and thus may have corrected errors before they were found by third parties. If we subtract the

[34]Krajowa Izba Rozliczeniowa S.A.

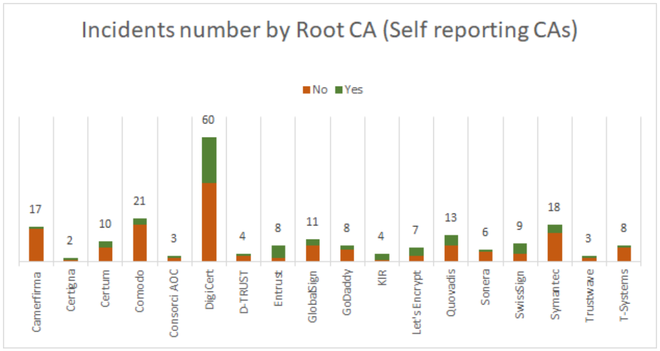


Fig. 5. Root CAs and self-reporting comparison (Including only CAs that have self-reported at least one incident)

67 self-reported incidents of the first set, we have that the non-self-reports set have those 167 not-reported incidents, while the other group has only 145 not-reported ones. If these non-reporting CAs identify incidents at the same rate, then it is reasonable to believe that there are tens of incidents not included in the data-set.

	#Root CA	**#Yes incidents**	**#No incidents**	**Total**
Self-reporting CAs	18	67	145	212
Non-self-reporting CAs	59	0	167	167

TABLE XIV
COMPARISON BETWEEN SELF-REPORTING AND NOT SELF-REPORTING CAS

The complete details can be found in Appendix 1, Table APP1.III[35], where we summarized the results of our survey of self-disclosure. In that table, we detailed each Root CA (or auditor) and if their studied incidents were reported/disclosed by them or by a third entity or person. We also provided the last column to facilitate a quick look at the percentages of self-reporting for each CA.

[35]In this analysis we also included the faulty auditors found in our study.

Electronic copy available at: https://ssrn.com/abstract=3425554

7) *Per year study of faulty Root and Problematic CAs:*

The comparison per year of CA's incidents shows that in the last years the number of problematic CAs issuances has skyrocketed. We detailed for each Root CA (or Auditor), the number of incidents collected for each year, appended in Table APP1.IV[36]. We then related this data directly to the Problematic CAs where the incidents were generated. (Please see Table APP1.V in Appendix 1 as well.) One argument for this phenomenon is that the number of certificates has similarly increased; however, there has been no corresponding growth in the number in CAs. In addition to the lack of a increase in CAs, the code base of the entire infrastructure has been maturing. Mature code is less likely to have vulnerabilities or logic flaws than less tested or newly created software [46]. Thus, increases arguably indicate chronic difficulties in the core function of root CAs.

For each year studied, we show the number of incidents in that year (light orange), and the accumulated number of incidents (including the year shown) for the Root CA (grey). For years where numerous Root CAs presented incidents, we only added figures for the most problematic CAs (taking into consideration the accumulated quantities). The rest of the incidents are grouped into an *"Others"* group. Finally, in the horizontal axis where the names of the Root CAs are displayed, we wrote in bold the companies that belonged to the set of top problematic Root CAs, as detailed in Table V. As expected, in the chart for 2019 (the last one) only these top problematic Root CAs appear in the chart, along with a column for *"Others"*.

The oldest incident to which we could find documentation is from 2001 and it was originated by *"VeriSign"*. The second incident in our study happened seven years after that one, precisely in 2008, adding a total of 5 incidents for that year. The faulty Root CAs for that year were *"Comodo"*, *"StartCom"*, *"Thawte"* and *"VeriSign"* again; all Root CAs that would exhibit more problems in the future. See Fig. 6 for the 2008 graphical representation. Recall Verisign was sold to Symantec because this core root CA had chronic failures in organization, technical, and purposeful business-driven errors.

From 2008 and onward, in every year the CAs presented at least 1 incident. In 2009 there was only 1 incident, again from *"VeriSign"*.

The next year, 2010, a total of 6 incidents were made public, 2 from *"Comodo"* and *"VeriSign"*, and 1 from *"Certum"* and *"GoDaddy"*; CAs that displayed a few more incidents in the last years. Fig. 8 shows the problematic Root CAs for that year and their accumulated number of incidents.

2011 was a unique year due to an epidemic of CAs being hacked, and this is reflected in the reported incident. The number of specific incidents doubled over the previous year (from 6 to 12). In addition to the already named CAs, new brands appear: *"TÜRKTRUST"* (inside *"Others"* in 9), *"KPN"*, *"GlobalSign"* and *"Entrust"*.

[36]Whenever it was possible we used the reporting date of the incident for this item, no matter if the reporting entity was the CA or another institution or person.

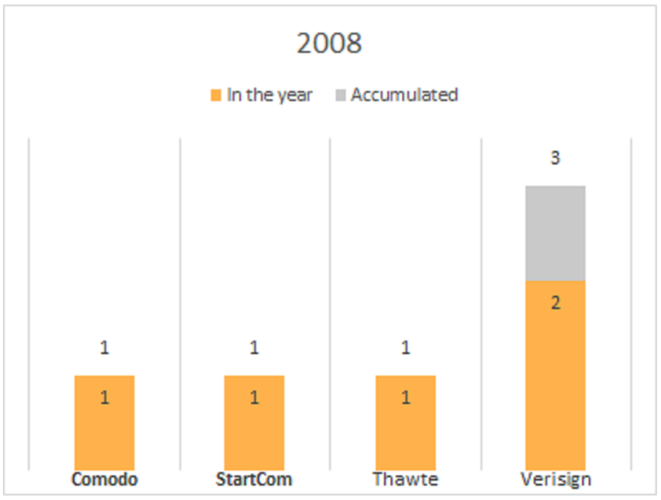


Fig. 6. Distribution of incidents per Root CAs - 2008

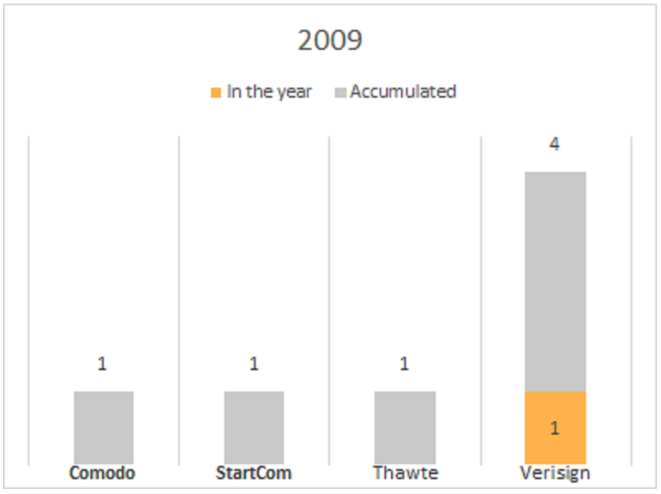


Fig. 7. Distribution of incidents per Root CAs - 2009

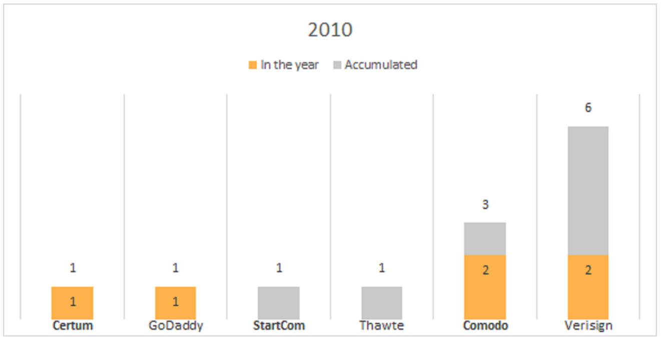


Fig. 8. Distribution of incidents per Root CAs - 2010

Electronic copy available at: https://ssrn.com/abstract=3425554

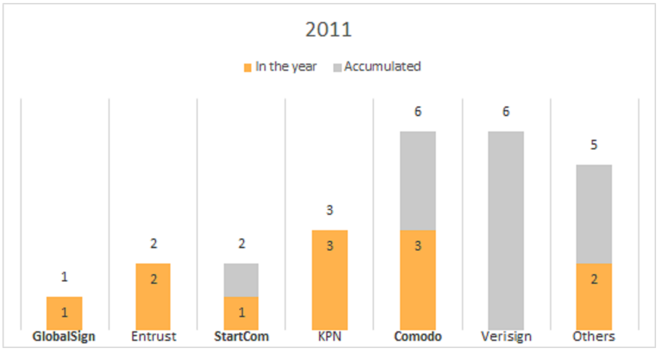


Fig. 9. Distribution of incidents per Root CAs - 2011

Years 2012 and 2013 presented a lower number of incidents compared to 2011, 5 and 2 respectively. For 2012 only Root CAs that had their first incidents in that year added to the total number; they were *"Camerfirma"*, *"Chunghwa" "Telecom"* and *"Trustwave"*. The same happened in 2013, with *"ANSSI"* and *"Atos"*. Fig. 10 and Fig. 11 correspond to these years.

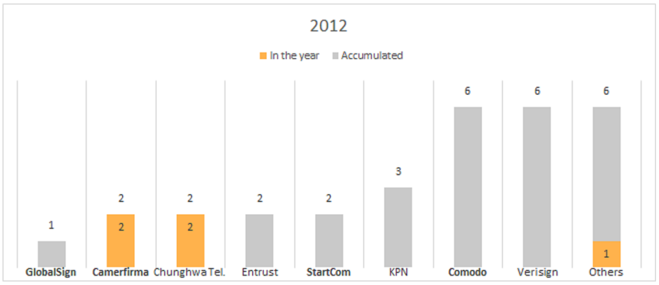


Fig. 10. Distribution of incidents per Root CAs - 2012

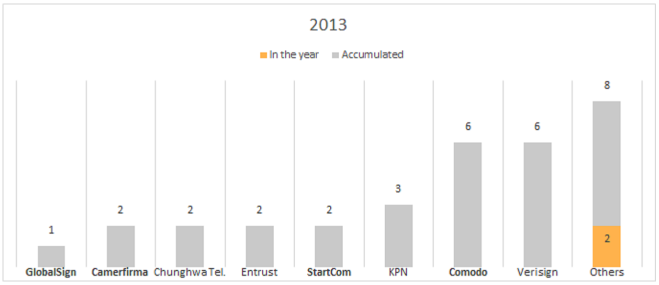


Fig. 11. Distribution of incidents per Root CAs - 2013

In 2014 and 2015 there were approximately 25 incidents each year. Compared to the previous years, it was a major change. Fig. 12 shows *"Symantec"*, which purchased the chronically problematic *"Verisign*, and *"DigiCert"*, two CAs in the top by the number of incidents. For the year 2015, the main source of incidents was *"WoSign"* (another top problematic Root CA), as can be seen in Fig. 13.

Almost 50 incidents were collected for 2016. This was a steep increase from the previous years, (nearly doubling). Much of these issues were caused by *"WoSign"* (triggering its revocation), *"Symantec"* (alerting the Root Program' owners about their unacceptable practices) and *"DigiCert"*. A range of new CAs appeared, with one incident each, as shown in Fig. 14. The number of incidents under *"Others"* represents 14 CAs.

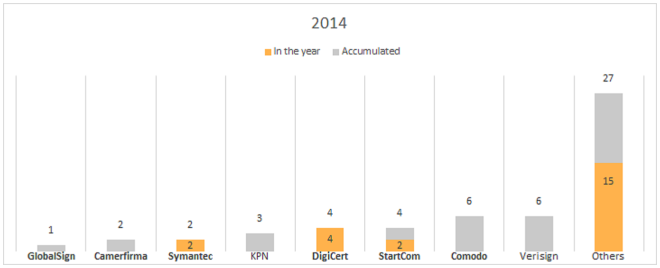


Fig. 12. Distribution of incidents per Root CAs - 2014

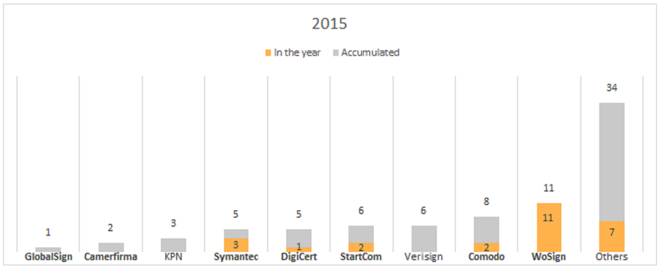


Fig. 13. Distribution of incidents per Root CAs - 2015

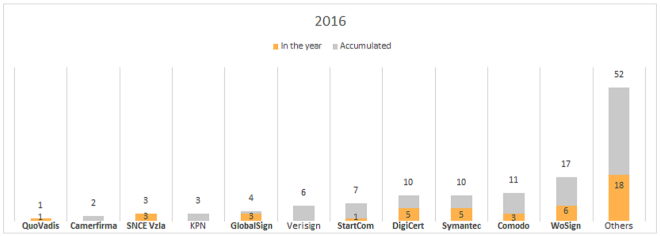


Fig. 14. Distribution of incidents per Root CAs - 2016

The biggest increase from one year to another happened in 2017. For that year, we collected 135 incidents; more than for all the previous years together and exactly three times the sum of incidents from the previous year. *"DigiCert"* was the Root CA that added more incidents for that year by far, followed by *"SNCE Venezuela"*, *"Comomdo"* and *"Symantec"*. This can be checked in Fig. 15. At this point, the only non-top problematic Root CA in the chart was *"VeriSign"*.

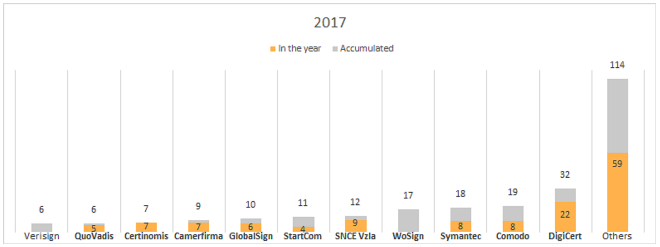


Fig. 15. Distribution of incidents per Root CAs - 2017

Electronic copy available at: https://ssrn.com/abstract=3425554

Something similar can be seen for the year 2018, with *"DigiCert"* leading the incident figures for that year. In this particular year, this Root CA was related to almost a quarter of the incidents. Other problematic Root CAs for that year were *"Camerfirma"* and *"QuoVadis"*; see Fig. 16. While for 2018 we collected 96 incidents, two-thirds than in 2017, in this year we collected 23 incidents for one Root CA, *"DigiCert"*; the highest number we could collect for a Root CA in one year. At this point in time, given the number of incidents accumulated by the top problematic Root CAs, this chart only contains these entities and the *"Others"* aggregation.

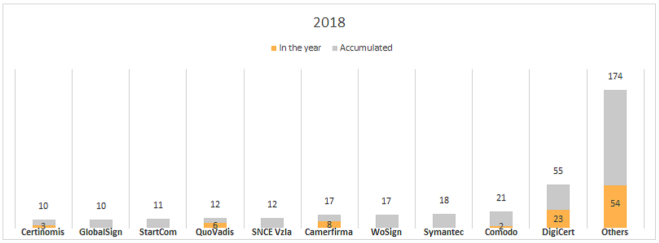


Fig. 16. Distribution of incidents per Root CAs - 2018

Finally, for 2019 we collected 22 incidents, quite a high figure taking into account that only events from January were studied. If the trend in the 12 months for this year keeps this rhythm, 2019 could set a record. Again *"DigiCert"* is the Root CA associated with more incidents (5) for a given period of time.

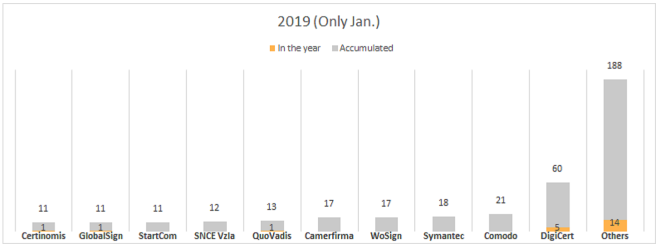


Fig. 17. Distribution of incidents per Root CAs - 2019 (Only January)

Specifically in relation to the Problematic CAs (where the incidents are generated), previously in Tab. VII we showed that the most problematic were: *"DigiCert"*, *"PROCERT"*, *"StartCom"*, *"WoSign"*, *"Comodo"*, *"GoDaddy"*, *"QuoVadis"*, *"VISA"*, *"Camerfirma"*, *"Certum"* and *"SwissSign"*. In Fig. 18 we can appreciate the evolution in the number of their incidents, in contrast with the *"Others"* Problematic CAs. In that chart, the number of incidents per Problematic CA was grouped in two sets for each year. To observe their individual evolution during the studied years, we can look at Figures 19, 20 and 21. In these charts we can note that these CAs have presented an important number of incidents mostly in the last years, being *"Comodo"* and *"StarCom"* the unique CAs that had incidents in more than four different years. Other cases like *"WoSign"* and *"PROCERT"*, their alarming number of incidents happened in a lapse of time no bigger than two years; triggering their revocation from Root Programs as a consequence of their unacceptable practices.[37]

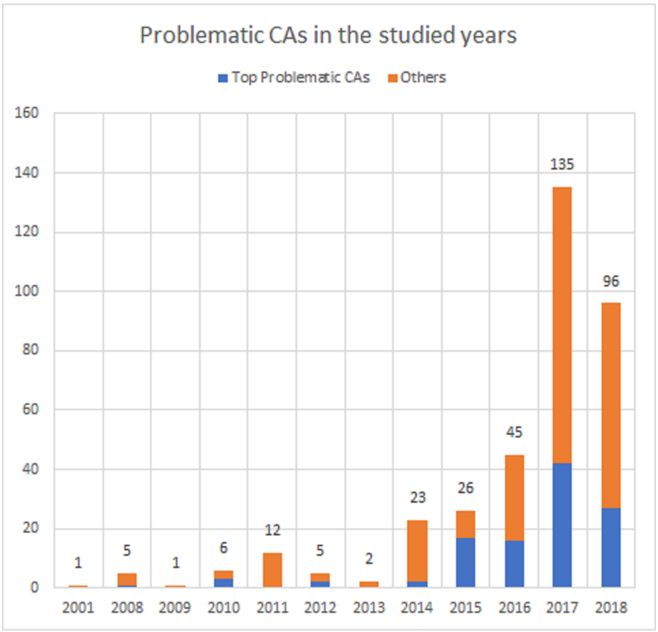


Fig. 18. Problematic CAs in the studied years

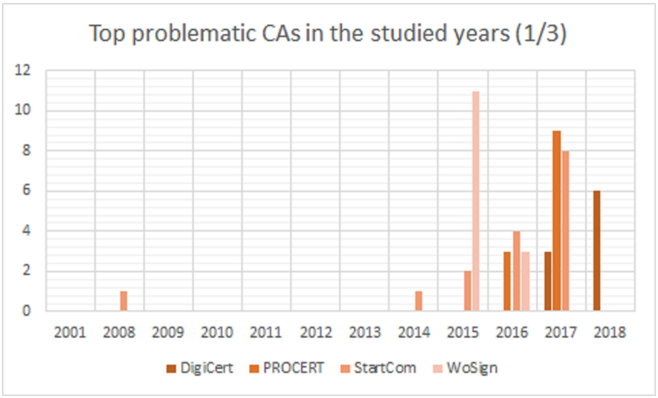


Fig. 19. Problematic CAs in the studied years (DigiCert, PROCERT, StartCom, and WoSign)

[37]The charts mentioned in the paragraph do not include the incidents collected in January/2019.

Electronic copy available at: https://ssrn.com/abstract=3425554

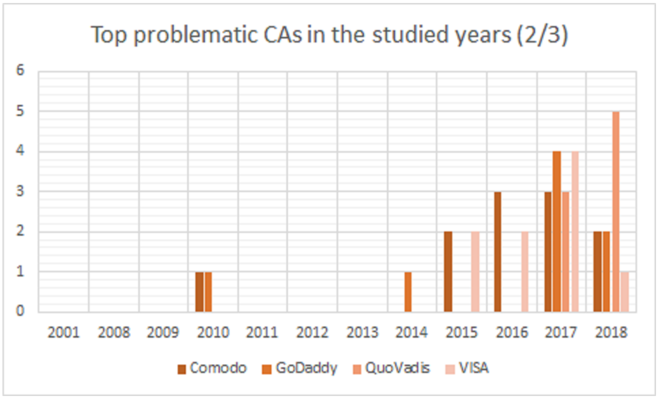


Fig. 20. Problematic CAs in the studied years (Comodo, GoDaddy, QuoVadis and VISA)

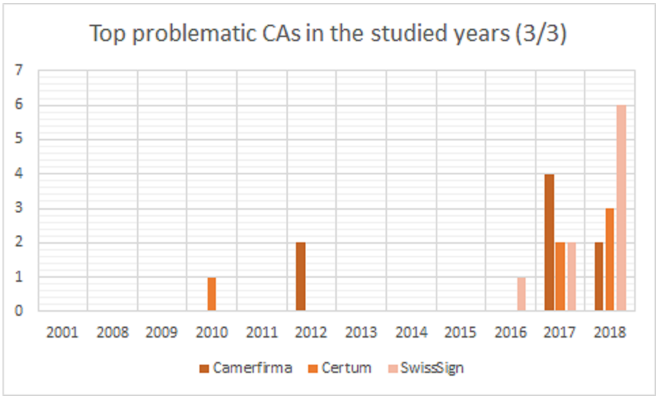


Fig. 21. Problematic CAs in the studied years (Camerfirma, Certum and SwissSign)

8) *Per year comparison of causes and types of incidents:* Another study involving information segmented by year we performed was the following.

We focused on the type of incidents and the causes of these for each year, in addition to taking a brief look at self-reporting figures. First, we can appreciate in Table XV this last element, the disclosing practices. There we can see how the practice of self-reporting incidents is becoming more common, specially if we see the change from 2017 to the beginning of 2019 (when this study collected the last elements), where the percentage increased from 8.15% in 2017 (11 in 135 incidents) to 31.25% in 2018 (30 in 96 incidents) and to 54.54% in January of 2019. Fig. 22 visually displays the total number of incidents per year and this trend in self-reporting. It would be really healthy for the PKI welfare if these figures keep evolving in that direction.

Second, Table XVI shows the different types of incidents per each year, and their evolution in the studied period. There, we can see how from 2001 to 2011 mostly all the incidents collected refer to the possible or actual issuance of *Rogue Certificates* and CAs or other entities being hacked. After this chaotic phase was overcome, with the help of stricter requirements by the Root Program's owners and the emergence of the BR, comes a phase where most of the incidents are related to non-compliance with these requirements:

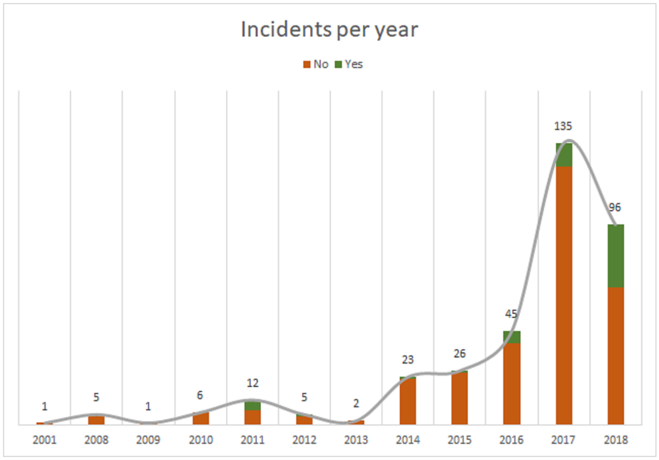


Fig. 22. Incidents non/yes reported per year

Fields in certificates not compliant to BR (specially this category), *512/1024 bits key*, *Repeated/Lacking appropriate entropy Serial Numbers*, *Use of SHA-1/MD5 hashing algorithm*, *Non-BR-compliant or problematic OCSP responder or CRL*, *CAA mis-issuance*, *Erroneous/Misleading/Late/Lacking Audit report* and *Undisclosed SubCA*. However, besides all these new incidents related to non-compliance, the possible or actual issuance of *Rogue Certificates* numbers did not diminish. Fig. 23 illustrates these findings.

We believe that the emergence of more demanding requirements helped to detect problematic practices in the CAs that were common and without control prior to these requirements. Moreover, the usage of Certificate Transparency and the new "lint" tools have enormously helped to detect technical problems in (mis-)issued digital certificates and to identify if these did not follow the defined requirements.

Finally, and in a similar approach than in the previous table, in Table XVII we detailed the data collected for each year regarding the causes of the incidents.

It can be seen that several causes generated incidents mostly in the last years, or, more probably, these incidents and their causes could be detected recently thanks to the new tools and requirements. Examples of these causes are: *Believed to be compliant/Misinterpretation/Unaware*, *Software bugs*, *Human error*, *Infrastructure problem*, *Misconfigured software*, *Miss-issuances not in sample*, *Operational error* and *Organizational constraints*.

In other cases like *Business model/CA decision/Testing* and *Non-optimal request check*, these causes have been generating incidents since the beginning of PKI usage, and, sadly, they have not been eradicated.

Fig. 24 shows these findings.

Electronic copy available at: https://ssrn.com/abstract=3425554

	2001	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
No	1	5	1	6	7	4	2	22	25	39	124	66	10
Yes	0	0	0	0	5	1	0	1	1	6	11	30	12

TABLE XV
SELF-REPORTING INCIDENTS PER YEAR

Type of Incident	**2001**	**2008**	**2009**	**2010**	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**2019**
Backdating SHA-1 certificates	0	0	0	0	0	0	0	0	0	2	0	0	0
CA/RA/SubCA/Reseller hacked	0	0	0	1	9	0	0	0	0	0	0	1	0
CAA mis-issuance	0	0	0	0	0	0	0	0	0	0	12	2	0
Certificates for malicious domains	0	0	0	0	0	0	0	0	0	0	0	0	1
Charging for compromised cert revocation (HeartBleed)	0	0	0	0	0	0	0	1	0	0	0	0	0
CPS non-compliance	0	0	0	0	0	0	0	0	1	1	0	0	0
Digital certificate for non-existent domain/Bad information of requester	0	0	0	1	0	0	0	0	0	0	1	0	0
Fields in certificates not compliant to BR	0	0	0	0	1	2	0	7	3	15	63	44	11
MITM attempt	0	0	0	0	1	1	1	0	1	0	0	0	0
No BR self-assessment	0	0	0	0	0	0	0	0	0	0	0	2	0
No disclosing another CA purchase	0	0	0	0	0	0	0	0	1	0	0	0	0
No test website for CA	0	0	0	0	0	0	0	0	0	0	0	1	0
Non-BR-compliant or problematic OCSP responder or CRL	0	0	0	1	0	1	0	1	1	0	26	6	3
Not acceptable requestor validation	0	0	0	0	0	0	0	0	1	0	1	0	0
Possible issuance of rogue certificates	0	0	1	2	0	0	0	0	2	5	4	4	0
Register competitor trademark	0	0	0	0	0	0	0	0	0	1	0	0	0
Rogue certificate	1	3	0	1	0	0	0	2	5	0	0	0	0
Self revocation	0	0	0	0	0	0	0	0	0	1	0	0	0
Timestamp certificate from root	0	0	0	0	0	0	0	0	0	0	0	1	0
Undisclosed SubCA	0	0	0	0	0	0	0	0	0	0	2	17	0
Un-revoking certificates	0	0	0	0	0	0	0	0	0	0	1	0	0
Validity greater than 825 days	0	0	0	0	0	0	0	0	0	0	0	2	2
Debian OpenSSL Vulnerability	0	0	0	0	0	0	0	0	0	0	0	2	0
512/1024 bits key	0	0	0	0	1	1	0	11	2	1	1	1	0
Delayed certificate revocation	0	0	0	0	0	0	0	0	1	0	4	0	1
Use of SHA-1 /MD5 hashing algorithm	0	1	0	0	0	0	0	0	1	12	1	0	0
Erroneous/Misleading /Late/Lacking Audit report	0	0	0	0	0	0	0	0	2	4	6	12	1
Repeated/Lacking appropriate entropy Serial Numbers	0	1	0	0	0	0	1	1	4	2	12	1	0
Not allowed ECC usage	0	0	0	0	0	0	0	0	1	1	1	0	3
TOTAL	**1**	**5**	**1**	**6**	**12**	**5**	**2**	**23**	**26**	**45**	**135**	**96**	**22**

TABLE XVI
TYPE OF INCIDENTS PER YEAR

Electronic copy available at: https://ssrn.com/abstract=3425554

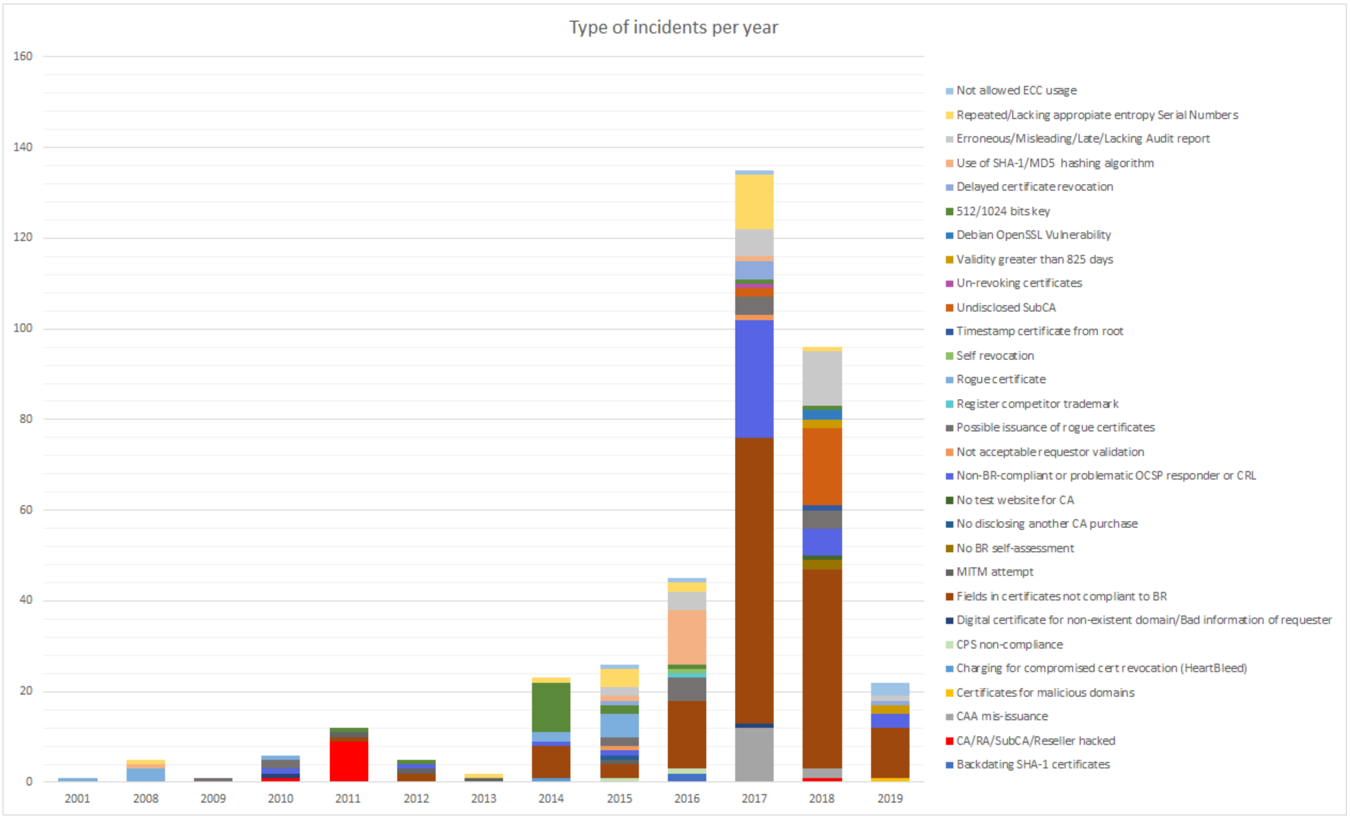


Fig. 23. Type of incidents per year

Cause of Incident	**2001**	**2008**	**2009**	**2010**	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**2019**
Believed to be compliant/ Misinterpretation/Unaware	0	0	0	0	0	1	0	2	2	5	33	22	4
Software bugs	0	1	0	0	1	1	0	1	5	10	45	24	3
Business model/CA decision /Testing	0	2	0	1	0	2	1	6	8	11	12	5	4
Change in BR	0	0	0	0	0	0	0	0	0	0	1	3	3
Human error	0	0	0	0	0	0	0	3	1	8	13	11	1
Improper security controls	0	0	0	1	8	0	0	2	0	2	1	1	0
Infrastructure problem	0	0	0	0	0	0	0	1	2	0	2	1	0
Misconfigured software	0	0	0	0	0	0	0	0	0	0	0	2	0
Miss-issuances not in sample	0	0	0	0	0	0	0	0	0	0	0	2	0
New Issue/No data yet	0	0	0	0	0	0	0	0	0	0	0	0	3
No data	0	0	0	1	2	0	0	6	3	7	5	11	0
Non-optimal request check	1	2	1	3	1	0	0	0	4	1	6	3	2
Operational error	0	0	0	0	0	0	1	2	0	0	15	9	2
Organizational constraints	0	0	0	0	0	1	0	0	1	1	1	2	0
Several	0	0	0	0	0	0	0	0	0	0	1	0	0
TOTAL	**1**	**5**	**1**	**6**	**12**	**5**	**2**	**23**	**26**	**45**	**135**	**96**	**22**

TABLE XVII
CAUSE OF INCIDENTS PER YEAR

Electronic copy available at: https://ssrn.com/abstract=3425554

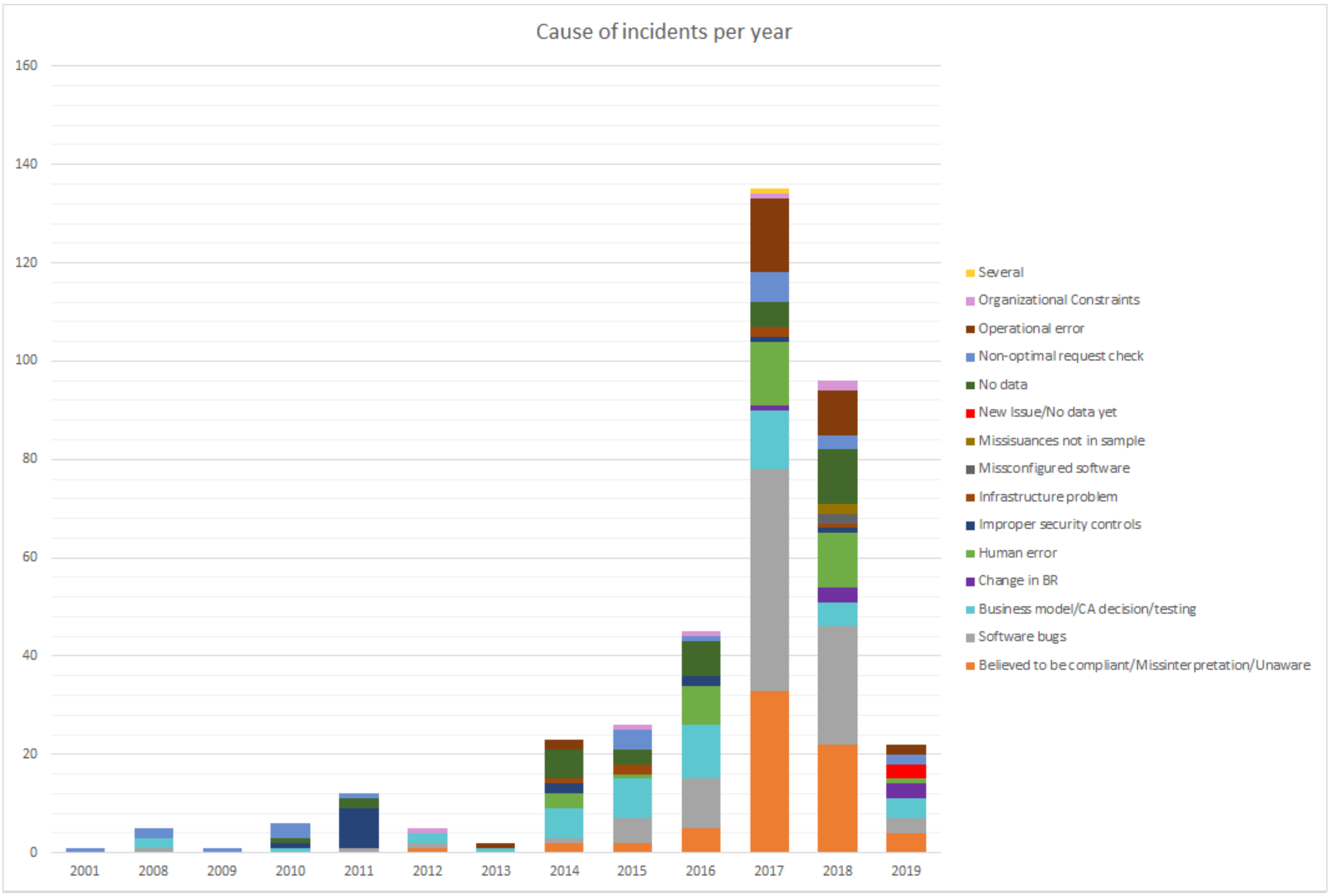


Fig. 24. Cause of incidents per year

9) Rogue certificates incidents in detail:

Our final analysis is focused on incidents related to the issuance of *Rogue Certificates*. In Table XVIII we detailed the Root CAs that have presented at least one incident where one or more *Rogue Certificates* were potentially[38] or actually issued. There we can appreciate the 12 Root CAs with these incidents. In decreasing order they are: *"Comodo"* with 8; *"WoSign"* with 5; *"GoDaddy"*, *"Symantec"* and *"VeriSign"* with 4; *"DigiCert"* with 2; and *"CNNIC"*, *"DigiNotar"*, *"India CCA"*, *"Let's Encrypt"*, *"StartCom"* and *"Thawte"* with 1. In Section X - Appendix 1, charts about rogue certificates are provided.

Under our study we found that in several of these cases, after the incidents were made public CAs were distrusted by Root Program's Owners and, as a consequence, in some opportunities companies closed or sold their CA business. *"CNNIC"* had its certificate and the certificate of its SubCA revoked. *"Comodo"* had its CA business sold to "Francisco Partners" and rebranded to *"Sectigo"* (although there is no connection in this decision with the issuance of *Rogue Certificates* or other incidents). *"DigiNotar"* was revoked and investigated by the Dutch government, and then closed. *"India CCA"* had its subCA revoked. *"StartCom"* was revoked and sold to *"Certinomis"*. *"Symantec"* was distrusted and its CA business was sold to *"DigiCert"*, along with its subCAs. *"Thawte"* was purchased by *"VeriSign"*, but the decision did not seem to be as a result of any incident. Identically the same happened to *"VeriSign"* when it was sold to *"Symantec"*. Finally, *"WoSign"* was distrusted and later purchased and rebranded.

Another thing that we can see in Table XVIII is that disclosing this kind of incidents is not a common practice, which seems evident due to the high negative impact that this may have in the CA. Only *"Comodo"* (2 in 8 cases), *"DigiCert"* (1 in 2 cases), *"GoDaddy"* (2 in 4 cases) and *"Symantec"* (1 in 4 cases) have ever publicly disclosed an incident of this type.

Table XIX crosses the Root CAs with the problematic CAs that were directly involved in the *Rogue Certificate* incidents and that were rooted in the former. In some cases the company may be the same. We can see that except in a few cases with direct Root CAs involvement like *"Comodo"*, *"GoDaddy"* and *"WoSign"*, these incidents were not concentrated in a particular CA. What is more, in some cases, only one incident related to *Rogue Certificates* was enough to distrust a problematic (non-Root) CA.

In Table XX we described the different causes that gener-

[38]We added all the incidents that potentially may have enabled the issuance of Rogue Certificates, but, clearly, there were incidents more probable than others.

Electronic copy available at: https://ssrn.com/abstract=3425554

Root CA	#No	#Yes	Total
CNNIC	1	0	1
Comodo	6	2	8
DigiCert	1	1	2
DigiNotar	1	0	1
GoDaddy	2	2	4
India CCA	1	0	1
Let's Encrypt	1	0	1
StartCom	1	0	1
Symantec	3	1	4
Thawte	1	0	1
VeriSign	4	0	4
WoSign	5	0	5

TABLE XVIII
ROGUE CERTIFICATES, SELF-REPORTING NUMBERS

Root CA	Problematic CAs	Total
CNNIC	MCS Holdings	1
Comodo	CertStar	1
	Comodo	6
	UTN-USERFirst-HW	1
DigiCert	DigiCert	1
	Thawte	1
DigiNotar	DigiNotar	1
GoDaddy	GoDaddy	4
India CCA	NIC	1
Let's Encrypt	Let's Encrypt	1
StartCom	StartCom	1
Symantec	Thawte	1
	CertSuperior	1
	CrossCert	1
	Symantec	1
Thawte	Thawte	1
VeriSign	Equifax	1
	RapidSSL	2
	VeriSign	1
WoSign	StartCom	2
	WoSign	3

TABLE XIX
ROOT CAs AND PROBLEMATIC CAs RELATION IN ROGUE CERTIFICATE INCIDENTS

ated the potential or actual issuances of *Rogue Certificates*. These are the causes previously studied in this work. Based on the data, we can say that 4 incidents were caused by "Inappropriate Management" (*Believed to be compliant/Misinterpretation/Unaware* and *Business model/CA decision/Testing*), 20 incidents were caused by "Inappropriate Practices" (*Improper security controls* and *Non-optimal request check*) and 9 incidents were caused by "Technical Flaws" (*Software bugs*). We believe that "Technical Flaws" may happen to every technological component in production environment, even to the most tested element, however, causes like "Inappropriate Practices" and "Inappropriate Management" should be eradicated, specially if we consider that CAs must be aligned to the Baseline and Root Program requirements and be frequently audited in order to prevent this to happen, or at least to detect these incidents.

Finally, Table XXI shows the number of incidents related to *Rogue Certificates* that were studied for each year. It can

Cause	Total
Believed to be compliant/Misinterpretation/Unaware	1
Software bugs	9
Business model/CA decision/Testing	3
Improper security controls	5
Non-optimal request check	15

TABLE XX
CAUSES OF ROGUE CERTIFICATES

be appreciated that this issue has not been solved, since in the last years the numbers have been fairly high compared to the historical records, with 2015 and 2016 being the most problematic years. In addition, we can see how this kind of incidents seem to happen to a particular CA in a restricted set of continuous years if it does not happen in only one year. This may be explained in two ways: CAs solve the cause of the incident, or they are distrusted from Root Programs. An exception to that specific problematic range of time is the case of *"Comodo"*, which was affected in six of the twelve years studied, spanning incidents from 2008 to 2018. These findings can be graphically appreciated in Fig. 25.

Electronic copy available at: https://ssrn.com/abstract=3425554

Root CA	2001	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
CNNIC	0	0	0	0	0	0	0	1	0	0	0	0
Comodo	0	1	0	1	1	0	0	0	2	2	0	1
DigiCert	0	0	0	0	0	0	0	0	0	0	0	2
DigiNotar	0	0	0	0	1	0	0	0	0	0	0	0
GoDaddy	0	0	0	1	0	0	0	0	0	0	2	1
India CCA	0	0	0	0	0	0	0	1	0	0	0	0
Let's Encrypt	0	0	0	0	0	0	0	0	1	0	0	0
StartCom	0	1	0	0	0	0	0	0	0	0	0	0
Symantec	0	0	0	0	0	0	0	0	2	0	2	0
Thawte	0	1	0	0	0	0	0	0	0	0	0	0
VeriSign	1	1	1	1	0	0	0	0	0	0	0	0
WoSign	0	0	0	0	0	0	0	0	2	3	0	0
TOTAL	**1**	**4**	**1**	**3**	**2**	**0**	**0**	**2**	**7**	**5**	**4**	**4**

TABLE XXI
ROGUE CERTIFICATES PER YEAR

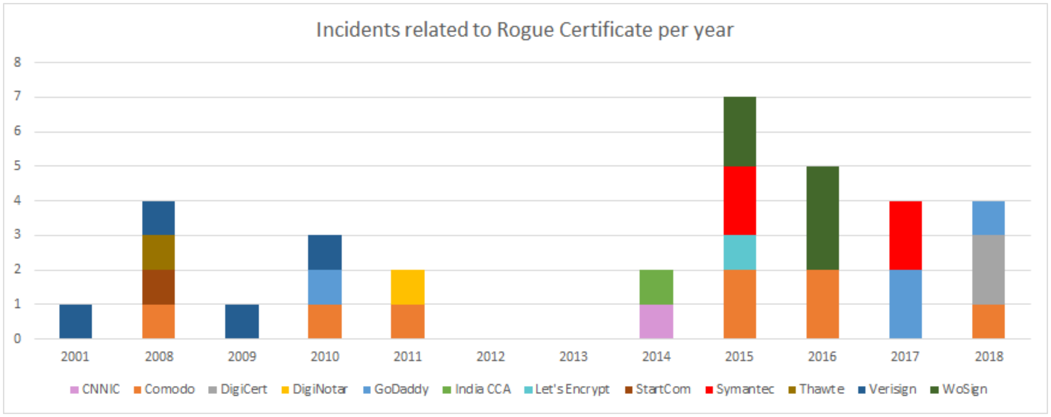


Fig. 25. Rogue Certificates per year

VII. Previous and Future Solutions

Regarding Root Programs' requirements and the CA/Browser's Baseline Requirements, it was observed that they are in continual improvements to fix the found holes, or to align to current times and vulnerabilities. No set of requirements can be complete, extensive and applicable to every CA, therefore, it is totally understandable that these requirements are updated reactively after negative circumstances. Also, the continual improvements and hardening of these requirements is a healthy display of involvement with the network's security, given the constant arms-race against malicious parties.

To prevent the effective use of rogue certificates several improvements to the PKI were designed. We briefly detailed the most popular options, although not all of them were finally implemented. In Section II, we also detailed some of the last proposals to improve the PKI network that have plenty of value to offer in the near future if they are adopted.

One of the first solutions was the use of **HTTP Public Key Pinning (HPKP)** or **Certificate Pinning** (RFC 7469 [47]). With this mechanism, a web host would send one or more cryptographic identities (public key or certificate) to a user agent (web browser) in an HTTP header (for example it can be the public key of its domain certificate, or a public key in its chain of trust). The user agent would remember that identity ("pin it") for some time, comparing the provided identity offered from the host each time the user visits it with the pinned identity. This scheme made possible the detection of rogue certificates; one of the most notable cases was in *"DigiNotar-2011"*. However, it was deprecated by Google and it is losing favor given that its usage is complex and prone to errors that trigger the loss of availability in misconfigured web sites[39].

The defense mechanism that substituted Certificate Pinning is **Certificate Transparency (CT)** (RFC 6962 [12]). This alternative attacks miss-issuances as well. It comprises making public a set of logs with the details of certificates issued by CAs. Domain owners should check these logs in order to detect rogue certificates for their sites, therefore, it will not proactively prevent the generation of new rogue certificates. It is a mitigating, deterrent and reactive solution, that transfers the burden to the CAs to log every issued certificate, and to the domain owners to check rogue certificates for their domains. With this solution, web browsers are no more a central part in the control, but indeed they have the power to enforce this logging policy, since they own the Root

[39]https://groups.google.com/a/chromium.org/forum/#!msg/blink-dev/he9tr7p3rZ8/eNMwKPmUBAAJ

Electronic copy available at: https://ssrn.com/abstract=3425554

Programs. Although it is not mandatory to CAs the logging of newly issued certificates in the Certificate Transparency's logs, some Root Programs demand it in order to trust in the digital certificate, for example Google's Root Program. *Signed Certificate Timestamp* (SCTs) are issued each time a CA submits a new certificate to a CT log, and, subsequently, each SCT is added to its related certificate as evidence that it was logged in a CT log. It is expected that in the future, clients (for example web browsers) will be able to check the certificates' legitimate SCT with the use of the *Expect-CT* HTTP header[40]. Recently, it has been very useful to detect issued digital certificates with erroneous fields, and CAs are using it in the pre-issuance phase to detect these problems beforehand. **In conclusion, while CT is a great improvement to the general welfare of the PKI network, in special with the adding of the different "lints" lately, we do not consider that this solution is a silver bullet against all the problems related to Problematic CAs and it may not prevent the generation and usage of rogue certificates, especially if the CA or the hacker does not log a particular malicious issuance; at least in the current state of the technology**.

Microsoft has included in its Internet Explorer and Edge's **Smart Screen** some rules to detect the usage of rogue certificates in domains visited by the users. Smart Screen collects the certificate information of a visited website, and then analyzes it in its servers to detect problems in the certificate, differences with previously collected information or differences with its Root Program data [48].

DNS Certification Authority Authorization (CAA), RFC 6844 [49], is an approach that requires that domain owners register in the DNS record of its domain the CA (or CAs) that can issue digital certificates for that domain. Since 2017, CAs should follow that field in the DNS record, otherwise, they would be disobeying the Baseline Requirements. As was seen in our analysis, CAs had issued digital certificates violating this requirement. Besides, it does not prevent or detect miss-issued rogue certificates.

Another proposal is **DNS based Authentication of Named Entities (DANE)**. A similar approach to HPKP, but instead of sending the key information in an HTTP header, it is stored in the domain's DNS registry. This piece of information stored in a record in the DNS is also signed with DNSSEC (Domain Name System Security Extensions) to bind it to the domain. The formal definition is in the RFC 6698 ([50]), however, it is not directly supported by the web browsers; only through independent extensions to them.

Trust Assertions for Certificate Keys (TACK) was a proposal that never went into production, [51]. At its draft time, it was competing with Google's pinning mechanism (HPKP). These two had several characteristics in common, however, TACK did not rely on anything related to the CA. For example, the server would send to the client a specific "TACK key", instead of a public key. Besides, this "TACK key" was the same in all the different servers that a domain may have. Finally, it was proposed as an extension to TLS, in contrast to HPKP that involved a new HTTP header.

[40]https://tools.ietf.org/html/draft-ietf-httpbis-expect-ct-07

In a crowdsourcing fashion, there were other proposals that involved the use of *"notaries"* to authenticate the digital certificates provided by the servers, like **Perspectives** and **Converge**. These "notaries" would collect information about certificates from different users accessing a specific domain, therefore, if the certificate that a site offers is the same that the one stored in the "notary" by distributed sources, probably the user is not being a victim of an MITM attack. But none of these ideas seems to have reached much adoption or even reached the production phase. A variant to these "notaries" mechanisms was **MECAI (Mutually Endorsing CA Infrastructure)**, were the "notaries" were also the CAs, with the constraint that an issuing CA could not be its own "notary" for its certificates, or *"Voucher Authority"*, as the proposal referred to these entities [52].

Finally, there were/are other approaches, but, if these ideas are not obsolete yet, most of them are through web browsers extensions with limited adoption, instead of being important changes to PKI that focus on the source of the problems. Some examples are: **Sovereign Keys, Certificate Patrol, EFF SLL Observatory, MonkeySphere, AKI, Crossbear, DoubleCheck, S-Links, DNSChain, PoliCert, DetecTor and ICSI Certificate Notary**, and a brief description can be found in [53].

VIII. Conclusion and Future Work

Based on the findings in our study, we could answer our three research questions:

1) What are the most common types of incidents related to CAs in the PKI network?
2) What are the causes of these incidents? What led or allowed them to happen?
3) Are these failures in PKI technical errors in the CAs, or is there enough evidence of misbehavior or lack of ethics in their practices?

The main conclusion of the study is that **Certificate Authorities are a source of negative incidents in PKI**. Incidents originating in these entities have been increasing in the studied lapse of time. Surprisingly, the ten Root CAs with most incidents related to them hoarded almost half of these incidents. Unethical or unacceptable behavior, unawareness or business decisions, unacceptable practices or levels of security, bugged software or human errors, and bad or flawed verification of the certificate's requestors are some of the causes. While some of these origins may be understood and accepted by the Root Programs, for example bugs in software are hard to eradicate, cases where the problems' source is in a business model or a CA decision that violates the Baseline Requirements or goes against the welfare of the PKI users should be severely penalized in order to deter them, since we found that it is a pervasive behavior in the CAs. It was also noted that after an important change in the BR or in the Root Program requirements, several CAs fail to comply with these new requirements. The reasons behind these failures may be

Electronic copy available at: https://ssrn.com/abstract=3425554

technical errors or misbehavior and unethical practices from the CAs.

The consequences of this phenomenon are numerous; however, the most common type of incidents are related to ***Fields in certificates not compliant with the Baseline Requirements*****, caused mostly by *****Software bugs*****, *****Unawareness in the CAs***** and *****Human errors***. These incidents became easier to be detected after the inclusion of *Certificate Transparency*. The conjunction of the logging practices, the different checks and "lint" tools, and the monitoring of the logs have allowed an easier detection of these miss-issued digital certificates. However, the impact in these incidents is not very critical compared to others studied in this work. For example, we detected that mostly the same causes that generated the integrity problems in the certificates were also linked to incidents in the *"OCSP responders and CRLs"*. These incidents can have a worse impact on the PKI network than a simple error in a certificate field.

From the perspective of the welfare in the PKI ecosystem, the most critical problem that we found is the miss-issuing of **rogue certificates**, especially used in MITM attacks. In addition, we found numerous instances where they could have been issued, based on particular problems found in the CAs. Rogue certificate incidents were found almost every year in our studied lapse of time. The major causes of these incidents were *Software bugs* and *Non-optimal request check*. It is interesting to note that, in these cases, the impact is not only on the final users; after the detection of (in general) non-disclosed incidents, Certificate Authorities can face the revocation of their signing certificate and exclusion of Root Programs, and consequently the possible end of their business. Root CAs, subordinate CAs, Registration Authorities and Resellers that do not meet the expected behavior and security controls and practices are constant sources of incidents.

Furthermore, we found that another entity that takes part in the PKI network that has brought issues as well is the figure of the auditor. **Auditors** may write erroneous audit documents due to flaws in their audit procedures. It happens in any kind of audit. However, non-accurate audit documents can come since auditors are business too, and if they make a statement claiming severe errors and unethical behaviors in a CA, this CA may be removed from the Root Programs, thus, the outcome would be one client less for this auditor. We are not stating that this is a common practice in the auditors, but indeed there is a direct conflict of interest in their role, especially when **decisions about CAs after a disclosure tend to be binary**: the CA must correct the source of the problem, or they are removed from the programs.

We also detected that **disclosing failures**, no matter if they are unintentional or product of misbehaviors, in the CAs is not a common activity. It should be encouraged and enforced in order to bring more transparency and security to the PKI network. Nevertheless, it is pleasing to note that at least a few CAs have shown an exemplary display of incident disclosure.

Another outcome of the study is the **tremendous power that Root Program's owners have** in the PKI network. With just their independent decision they can end the business of a CA, especially if several Root Program's owners are aligned with the revocation's posture. Given that they are also the owners of the web browsers, they are **judge, jury and executioner** in the network. On the other hand, if a misbehavior is detected in a CA but not all the program's owners agree with a mass removal of it, a removal by a sole owner may have a negative impact on this owner given the potential loss of customers that that decision may carry; therefore, **if there is no consensus between Root Program's owners, CAs may keep with their miss-practices**.

On the side of the more positive findings are the **continual updates of the Baseline Requirements and the Root Programs** requirements, that help to maintain a healthy PKI network. Also, the **commitment that Root Program's owners have** shown with the PKI network; although they may have more power than in an optimal solution, they are the only ones that are supervising and controlling the CAs' misbehaviors and equivocations.

Regarding the present solutions, as we stated, **Certificate Transparency** is a great effort to sanitize the PKI network and its good qualities have been observed since its deployment. But inasmuch as its approach is reactive and it needs continuous monitoring to detect CAs deviations or anomalies in issued digital certificates, a more preventive and proactive solution may be more desirable. New "lints" for the logs, Root Programs making mandatory the logging of new issued certificates and browsers demanding Signed Certificate Timestamps in the digital certificates are improvements in that direction.

A particular finding regarding security solutions in PKI is that, given the previous failures of other proposals, it seems that **the fewer changes to the PKI network the solution brings, the more probability of acceptance and adhesion the solution will have**. In the related work section, several promising ideas were presented.

In short, while the theoretical cryptography basis of PKI may seem flawless, bad technical implementations, erroneous or unacceptable operational procedures, and business interests undermine this scheme. In special, **Certificate Authorities may present a risk to the entire network that is not known by all the participants**. Perhaps a cleaner scheme without so many interested parties, especially the ones whose decisions balance between bringing trust to the network or to be more profitable, could be a better mechanism. After all, today PKI's network relies on these entities, and we as end users are obliged to trust in them without any viable alternative.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. CNS 1565375 CNS 1814518; support from a Cisco Research Award No. 1377239, and funds from the Comcast Innovation Fund. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF, Cisco, or

Electronic copy available at: https://ssrn.com/abstract=3425554

Comcast. The reporting mechanism enforced and the work done by Mozilla, the professionals working there, and the community in general were helpful in our research.

IX. References

References

[1] Carl Ellison and Bruce Schneier. "Ten risks of PKI: What you'‘re not being told about Public Key Infrastructure". In: *Computer Security Journal* 16 (Dec. 2000).

[2] L Jean Camp, Helen Nissenbaum, and Cathleen McGrath. "Trust: A collision of paradigms". In: *International Conference on Financial Cryptography*. Springer. 2001, pp. 91–105.

[3] R. Anderson. "Why information security is hard - an economic perspective". In: *Seventeenth Annual Computer Security Applications Conference*. IEEE Comput. Soc.

[4] Martin Abadi et al. "Global Authentication in an Untrustworthy World". In: *Proceedings of the 14th USENIX Conference on Hot Topics in Operating Systems*. HotOS'13. Santa Ana Pueblo, New Mexico: USENIX Association, 2013, pp. 19–19. URL: http://dl.acm.org/citation.cfm?id=2490483.2490502.

[5] D. Park. "Social Life of PKI: Sociotechnical Development of Korean Public-Key Infrastructure". In: *IEEE Annals of the History of Computing* 37.2 (Apr. 2015), pp. 59–71. ISSN: 1058-6180. DOI: 10.1109/MAHC.2015.22.

[6] A. Ferreira et al. "Studies in Socio-technical Security Analysis: Authentication of Identities with TLS Certificates". In: *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. July 2013, pp. 1553–1558. DOI: 10.1109/TrustCom.2013.190.

[7] S. B. Roosa and S. Schultze. "Trust Darknet: Control and Compromise in the Internet's Certificate Authority Model". In: *IEEE Internet Computing* 17.3 (May 2013), pp. 18–25.

[8] Antoine Delignat-Lavaud et al. "Web PKI: Closing the Gap between Guidelines and Practices". In: *Proceedings of the ISOC Network and Distributed System Security Symposium (NDSS '14)*. Feb. 2014. URL: http://antoine.delignat-lavaud.fr/doc/ndss14.pdf.

[9] Stephanos Matsumoto and Raphael M. Reischuk. "IKP: Turning a PKI Around with Decentralized Automated Incentives". In: *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2017.

[10] Salabat Khan et al. "Accountable and Transparent TLS Certificate Management: An Alternate Public-Key Infrastructure with Verifiable Trusted Parties". In: *Security and Communication Networks* 2018 (July 2018), pp. 1–16.

[11] Ze Wang et al. "Blockchain-Based Certificate Transparency and Revocation Transparency". In: *Financial Cryptography and Data Security*. Springer Berlin Heidelberg, 2019, pp. 144–162.

[12] B. Laurie, A. Langley, and E. Kasper. *Certificate Transparency*. RFC 6962. June 2013.

[13] Josef Gustafsson et al. "A First Look at the CT Landscape: Certificate Transparency Logs in Practice". In: *Passive and Active Measurement*. Springer International Publishing, 2017, pp. 87–99.

[14] Quirin Scheitle et al. "The Rise of Certificate Transparency and Its Implications on the Internet Ecosystem". In: *ACM SIGCOMM Internet Measurement Conference (IMC)*. ACM, Sept. 2018.

[15] Saba Eskandarian et al. "Certificate Transparency with Privacy". In: *Proceedings on Privacy Enhancing Technologies* 2017.4 (Oct. 2017), pp. 329–344.

[16] Johannes Braun and Gregor Rynkowski. "The Potential of an Individualized Set of Trusted CAs: Defending against CA Failures in the Web PKI". In: *2013 International Conference on Social Computing*. IEEE, Sept. 2013.

[17] Zheng Dong, Kevin Kane, and L. Jean Camp. "Detection of Rogue Certificates from Trusted Certificate Authorities Using Deep Neural Networks". In: *ACM Transactions on Privacy and Security* 19.2 (Sept. 2016), pp. 1–31.

[18] Andrea Micheloni et al. "Laribus: Privacy-Preserving Detection of Fake SSL Certificates with a Social P2P Notary Network". In: *2013 International Conference on Availability, Reliability and Security*. IEEE, Sept. 2013.

[19] Deepak Kumar et al. "Tracking Certificate Misissuance in the Wild". In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, May 2018.

[20] Oliver Gasser et al. "In Log We Trust: Revealing Poor Security Practices with Certificate Transparency Logs and Internet Measurements". In: *Passive and Active Measurement*. Springer International Publishing, 2018, pp. 173–185.

[21] Chad Brubaker et al. "Using Frankencerts for Automated Adversarial Testing of Certificate Validation in SSL/TLS Implementations". In: *2014 IEEE Symposium on Security and Privacy*. IEEE, May 2014.

[22] Microsoft Corporation. *Microsoft Trusted Root Certificate: Program Requirements*. URL: https://technet.microsoft.com/en-us/library/cc751157.aspx (visited on 11/30/2018).

[23] Microsoft Corporation. *Microsoft to remove WoSign and StartCom certificates in Windows 10*. URL: https://cloudblogs.microsoft.com/microsoftsecure/2017/08/08/microsoft-to-remove-wosign-and-startcom-certificates-in-windows-10/ (visited on 11/30/2018).

[24] Microsoft Corporation. *Trusted Root Program Participants (as of November 27 2018)*. URL: https:

Electronic copy available at: https://ssrn.com/abstract=3425554

//gallery.technet.microsoft.com/Trusted-Root-Program-831324c6 (visited on 02/15/2019).

[25] Apple Inc. *Apple Root Certificate Program*. URL: www.apple.com/certificateauthority/ca_program.html (visited on 11/30/2018).

[26] CPA Canada. *WEBTRUST PRINCIPLES AND CRITERIA FOR CERTIFICATION AUTHORITIES*. URL: www.webtrust.org/principles-and-criteria/docs/item85228.pdf (visited on 11/30/2018).

[27] CPA Canada. *WEBTRUST® FOR CERTIFICATION AUTHORITIES: WEBTRUST PRINCIPLES AND CRITERIA FOR CERTIFICATION AUTHORITIES – EXTENDED VALIDATION SSL*. URL: www.webtrust.org/principles-and-criteria/docs/item85117.pdf (visited on 11/30/2018).

[28] CA/Browser Forum. *Guidelines For The Issuance And Management Of Extended Validation Certificates*. URL: https://cabforum.org/wp-content/uploads/CA-Browser-Forum-EV-Guidelines-v1.6.8.pdf (visited on 11/30/2018).

[29] CPA Canada. *WEBTRUST PRINCIPLES AND CRITERIA FOR CERTIFICATION AUTHORITIES – SSLBASELINE WITH NETWORK SECURITY*. URL: www.webtrust.org/principles-and-criteria/docs/item85437.pdf (visited on 11/30/2018).

[30] CA/Browser Forum. *Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates*. URL: https://cabforum.org/wp-content/uploads/CA-Browser-Forum-BR-1.6.0.pdf (visited on 11/30/2018).

[31] Apple Inc. *List of available trusted root certificates in iOS 12.1.3, macOS 10.14.3, watchOS 5.1.3, and tvOS 12.1.2*. URL: https://support.apple.com/en-us/HT209501 (visited on 02/15/2019).

[32] The Chromium Projects. *Root Certificate Policy*. URL: www.chromium.org/Home/chromium-security/root-ca-policy (visited on 11/30/2018).

[33] Mozilla Foundation. *Network Security Services*. URL: https://developer.mozilla.org/en-US/docs/Mozilla/Projects/NSS (visited on 11/30/2018).

[34] Mozilla Foundation. *Mozilla's CA Certificate Program*. URL: https://wiki.mozilla.org/CA (visited on 11/30/2018).

[35] Mozilla Foundation. *Mozilla CA Certificate Store*. URL: www.mozilla.org/en-US/about/governance/policies/security-group/certs/ (visited on 11/30/2018).

[36] Mozilla Foundation. *Mozilla Root Store Policy*. URL: www.mozilla.org/en-US/about/governance/policies/security-group/certs/policy/ (visited on 11/30/2018).

[37] ETSI. *Standards*. URL: www.etsi.org/standards (visited on 11/30/2018).

[38] Mozilla Foundation. *Mozilla CA Information Report*. URL: https://ccadb-public.secure.force.com/mozilla/CAInformationReport (visited on 02/15/2019).

[39] Mozilla Foundation. *Mozilla Included CA Certificate List*. URL: https://ccadb-public.secure.force.com/mozilla/IncludedCACertificateReport (visited on 02/15/2019).

[40] ccadb. *Common CA Database*. URL: https://ccadb.org/ (visited on 11/30/2018).

[41] CA/Browser Forum. *Certificate Contents for Baseline SSL*. URL: https://cabforum.org/baseline-requirements-certificate-contents (visited on 11/30/2018).

[42] CA/Browser Forum. *About the Baseline Requirements*. URL: https://cabforum.org/about-the-baseline-requirements (visited on 11/30/2018).

[43] CA/Browser Forum. *Guidelines For The Issuance And Management Of Extended Validation Code Signing Certificates*. URL: https://cabforum.org/wp-content/uploads/EV-Code-Signing-v.1.4.pdf (visited on 11/30/2018).

[44] CPA Canada. *Principles and Criteria*. URL: www.webtrust.org/principles-and-criteria/item83172.aspx (visited on 11/30/2018).

[45] ISO. *ISO 21188:2006, Public key infrastructure for financial services – Practices and policy framework*. URL: www.iso.org/standard/35707.html (visited on 11/30/2018).

[46] Leyla Bilge and Tudor Dumitraș. "Before we knew it: an empirical study of zero-day attacks in the real world". In: *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM. 2012, pp. 833–844.

[47] C. Evans, C. Palmer, and R. Sleevi. *Public Key Pinning Extension for HTTP*. RFC 7469. Apr. 2015.

[48] Microsoft Corporation. *A novel method in IE11 for dealing with fraudulent digital certificates*. URL: https://blogs.technet.microsoft.com/pki/2014/02/21/a-novel-method-in-ie11-for-dealing-with-fraudulent-digital-certificates/ (visited on 01/30/2019).

[49] P. Hallam-Baker and R. Stradling. *DNS Certification Authority Authorization (CAA) Resource Record*. RFC 6844. Jan. 2013.

[50] P. Hoffman and J. Schlyter. *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA*. RFC 6698. Aug. 2012.

Electronic copy available at: https://ssrn.com/abstract=3425554

[51] M. Marlinspike and T. Perrin. *Trust Assertions for Certificate Keys*. URL: `https://tools.ietf.org/html/draft-perrin-tls-tack-02` (visited on 01/30/2019).

[52] Kai Engert. *MECAI - Mutually Endorsing CA Infrastructure*. URL: `https://kuix.de/mecai/mecai-proposal-v2.pdf` (visited on 01/30/2019).

[53] H. Tschofenig and E. Lear. *Evolving the Web Public Key Infrastructure*. URL: `https://www.ietf.org/archive/id/draft-tschofenig-iab-webpki-evolution-01.txt` (visited on 01/30/2019).

X. APPENDIX 1

In Section X, Appendix 1, we detail the tables and charts we created (in addition to the ones found in the document's body) to help us in our research.

A. Root program usage by web browser

Web Browser Root Store			Mozilla's CA Cert. Program
OS level Root Store	Microsoft Trusted Root Cert. Program	Apple Root Cert. Program	
Uses it	Edge/IE Chrome Opera Brave	Safari Chrome Opera Brave	Firefox Chrome Opera Tor Brave

TABLE APP1.I
THE ROOT CERTIFICATE PROGRAMS DISCUSSED IN THIS WORK. THE PROGRAMS EXIST AT DIFFERENT LEVELS AND APPLY TO DISTINCT PRODUCT LINES.

B. Root CAs and Problematic CAs (not roots of the chain) with most incidents collected

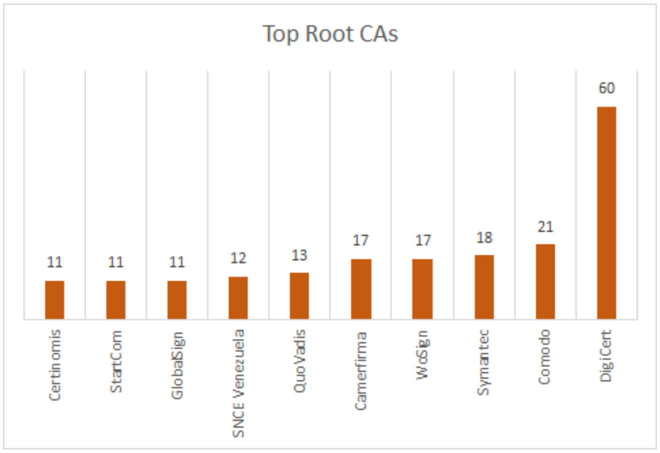


Fig. APP1.1. Root CAs with most incidents collected

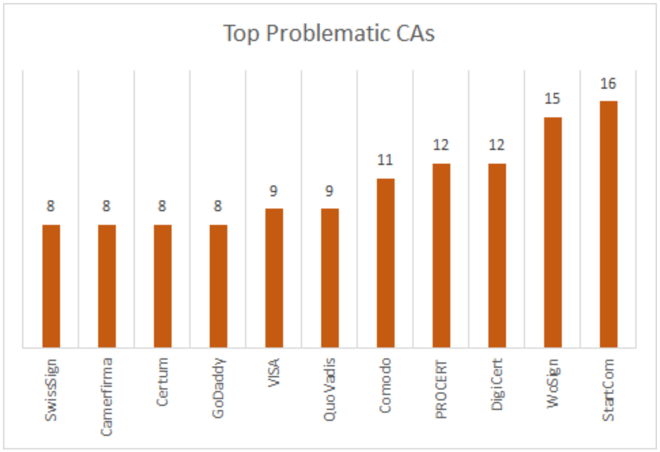


Fig. APP1.2. Problematic CAs with most incidents collected

Electronic copy available at: https://ssrn.com/abstract=3425554

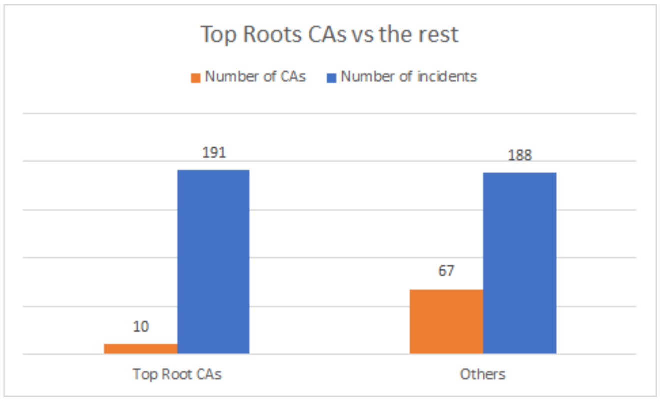


Fig. APP1.3. Top Root CAs incident reporters vs the rest reporting CAs

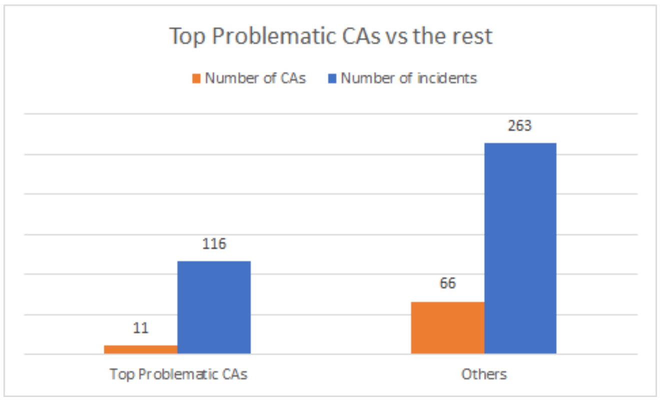


Fig. APP1.4. Top Problematic CAs incident reporters vs the rest reporting CAs

C. *Country of origin of the studied Root CAs*

ACCV	Spain	**IdenTrust**	USA
Actalis	Italy	**India CCA**	India
ANSSI	France	**Izenpe**	Spain
Atos	Germany	**Kamu**	Turkey
Belgium Root CA	Belgium	**Kazakhstan CA**	Kazakhstan
Camerfirma	Spain	**KIR**	Poland
Certicamara	Colombia	**Korean LIRDI**	Korea
Certigna	France	**KPN**	Netherlands
Certinomis	France	**Let's Encrypt**	USA
Certplus	France	**Microsec**	Hungary
CertSIGN	Romania	**NetLock**	Hungary
Certum	Poland	**QuoVadis**	Bermuda
Chunghwa Telecom eCA	Taiwan	**RSA**	USA
CNNIC	China	**SECOM**	Japan
Comodo	UK	**Sectigo**	UK
Consorci AOC	Spain	**Sertifitseerimiskeskuse**	Estonia
Cybertrust	Ireland	**SNCE Venezuela**	Venezuela
Cybertrust JP	Japan	**Sonera**	Finland
Dell Inc. CA1	USA	**Staat der Nederlanden Root CA**	Netherlands
DigiCert	USA	**StartCom**	China
DigiNotar	Netherlands	**Swisscom**	Switzerland
Disig	Slovak Republic	**SwissSign**	Switzerland
DocuSign	France	**Symantec**	USA
D-TRUST	Germany	**Thawte**	South Africa
EBG Elektronik Sertifika	Turkey	**Trustis**	UK
E-Guven	Turkey	**Trustwave**	USA
Entrust	Canada	**T-Systems**	Germany
Firmaprofesional	Spain	**TÜBİTAK UEKAE**	Turkey
FNMT	Spain	**TÜRKTRUST**	Turkey
GDCA	China	**VeriSign**	USA
Generalitat Valenciana CA	Spain	**VISA**	USA
GeoTrust	USA	**WISeKey**	Switzerland
GlobalSign	USA	**WoSign**	China
GoDaddy	USA		
GRCA (TW)	Taiwan		
Hong Kong Post	Hong Kong		

TABLE APP1.II
ROOT CAS AND COUNTRY OF ORIGIN

Electronic copy available at: https://ssrn.com/abstract=3425554

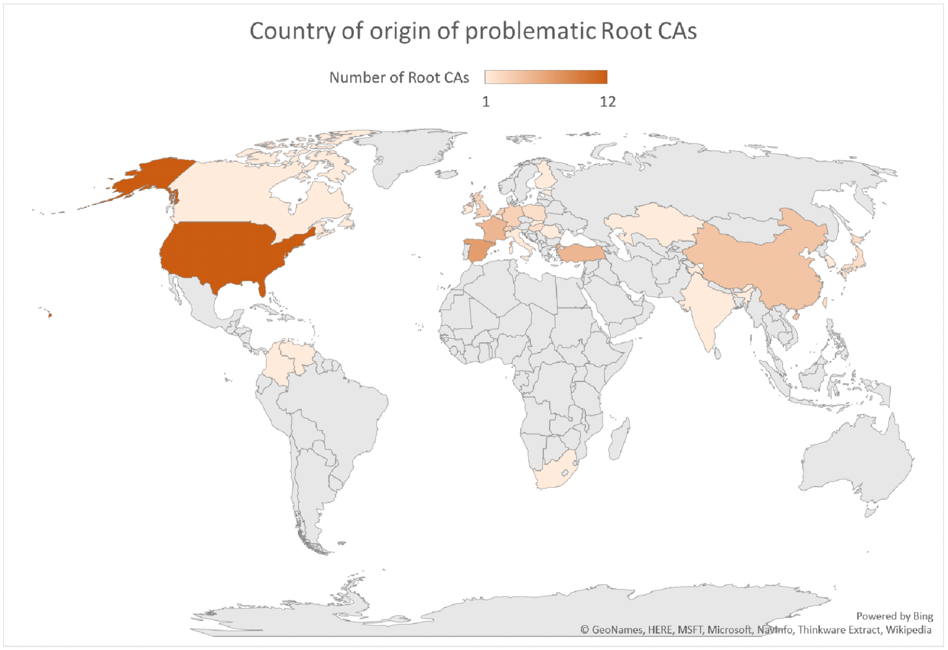

Country of origin of problematic Root CAs
Number of Root CAs
1
12
Powered by Bing
© GeoNames, HERE, MSFT, Microsoft, NavInfo, Thinkware Extract, Wikipedia

Fig. APP1.5. Country of origin of problematic Root CAs. Map version

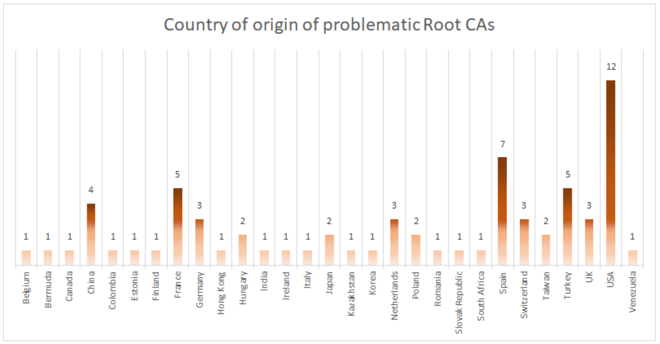


Fig. APP1.6. Country of origin of problematic Root CAs

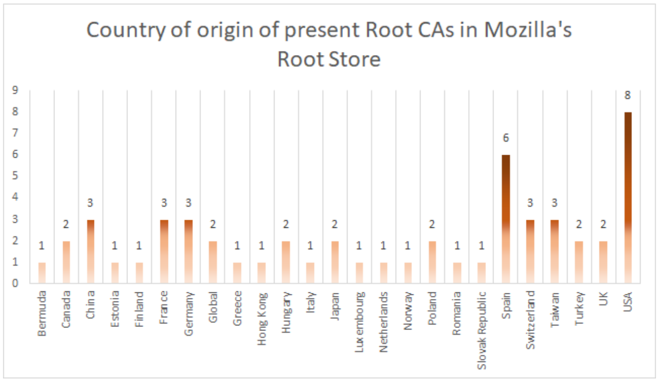


Fig. APP1.7. Country of origin of present Root CAs in Mozilla's Root Store

Electronic copy available at: https://ssrn.com/abstract=3425554

D. *Most repeated types and causes of incidents*

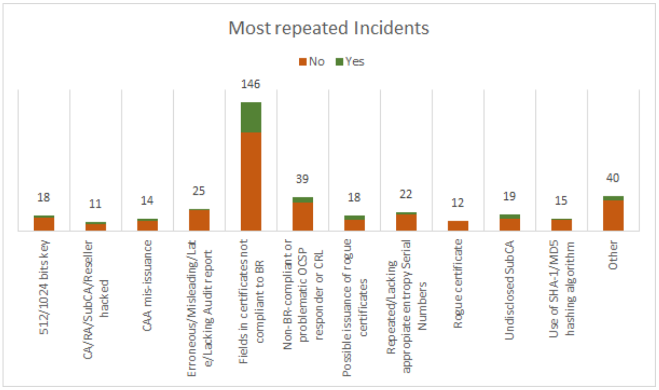


Fig. APP1.8. Most repeated types of incident and discrimination between self-reporting or not

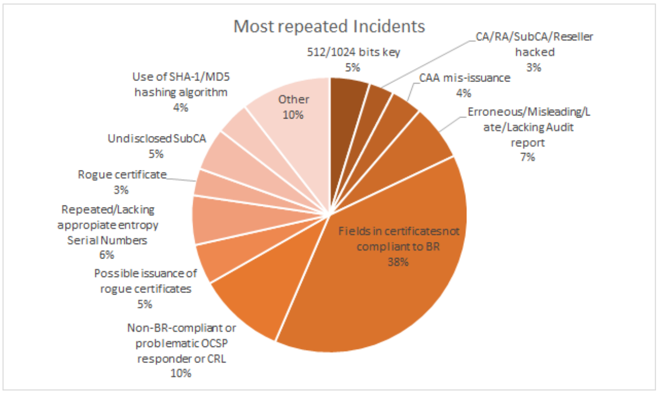


Fig. APP1.9. Distribution of the most repeated incidents

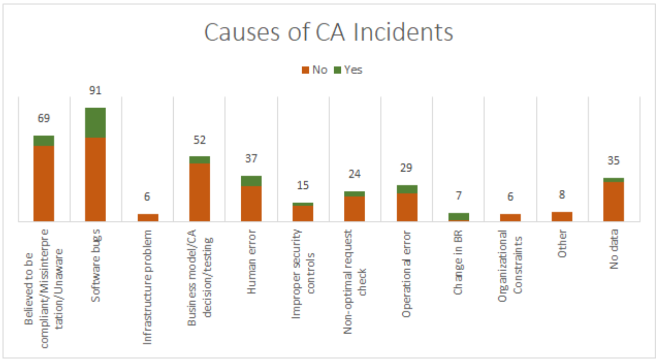


Fig. APP1.10. Most repeated cause of incident and discrimination between self-reporting or not

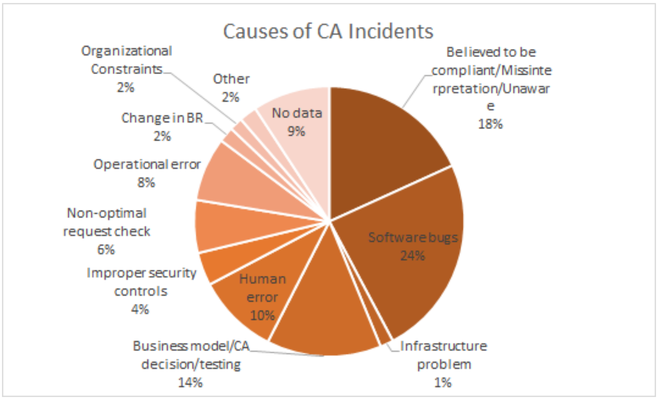


Fig. APP1.11. Distribution of the most repeated causes

E. *Self-reporting*

Electronic copy available at: https://ssrn.com/abstract=3425554

Root CA	#No	#Yes	Percentage
ACCV	2		0.00%
Actalis	3		0.00%
ANSSI	3		0.00%
Atos	1		0.00%
Belgium Root CA	1		0.00%
Camerfirma	16	1	5.88%
Certicamara	1		0.00%
Certigna	1	1	50.00%
Certinomis	11		0.00%
Certplus	1		0.00%
CertSIGN	2		0.00%
Certum	7	3	30.00%
Chunghwa Telecom eCA	2		0.00%
CNNIC	1		0.00%
Comodo	18	3	14.29%
Consorci AOC	2	1	33.33%
CPA Canada (Auditor)	1		0.00%
Cybertrust	2		0.00%
Cybertrust JP	1		0.00%
Dell Inc. Enterprise Issuing CA1	1		0.00%
Deloitte Anjin (Auditor)	1		0.00%
DigiCert	38	22	36.67%
DigiNotar	1		0.00%
Disig	2		0.00%
DocuSign	5		0.00%
D-TRUST	3	1	25.00%
EBG Elektronik Sertifika	3		0.00%
E-Guven	2		0.00%
Entrust	2	6	75.00%
EY HanYoung (Auditor)	1		0.00%
EY Hong Kong (Auditor)	1		0.00%
EY Poland (Auditor)	1		0.00%
Firmaprofesional	5		0.00%
FNMT	1		0.00%
GDCA	1		0.00%
Generalitat Valenciana CA	1		0.00%
GeoTrust	1		0.00%
GlobalSign	8	3	27.27%
GoDaddy	6	2	25.00%
GRCA (TW)	3		0.00%
Hong Kong Post	3		0.00%
IdenTrust	3		0.00%
India CCA	1		0.00%
Izenpe	5		0.00%
Kamu	1		0.00%
Kazakhstan CA	1		0.00%
KIR	1	3	75.00%
Korean LIRDI	1		0.00%
KPMG Samjong (Auditor)	1		0.00%
KPN	3		0.00%
Let's Encrypt	3	4	57.14%
Microsec	2		0.00%
NetLock	2		0.00%
QuoVadis	8	5	38.46%
RSA	1		0.00%
SECOM	7		0.00%
Sectigo	1		0.00%
Sertifitseerimiskeskuse	1		0.00%
SNCE Venezuela	12		0.00%
Sonera	5	1	16.67%
Staat der Nederlanden Root CA	2		0.00%
StartCom	11		0.00%
Swisscom	3		0.00%
SwissSign	4	5	55.56%
Symantec	14	4	22.22%
Thawte	1		0.00%
Trustis	2		0.00%
Trustwave	2	1	33.33%
T-Systems	7	1	12.50%
TÜBİTAK UEKAE	1		0.00%
TÜRKTRUST	2		0.00%
TUViT (Auditor)	1		0.00%
VeriSign	6		0.00%
VISA	9		0.00%
WISeKey	3		0.00%
WoSign	17		0.00%
Several CAs	2		0.00%

TABLE APP1.III

LISTING OF ALL CAS IDENTIFIED AS HAVING INCORRECT CERTIFICATES WITH AN INDICATOR IF THESE WERE SELF-REPORTED, AS WELL AS PERCENTAGES OF EVENTS DISCLOSED

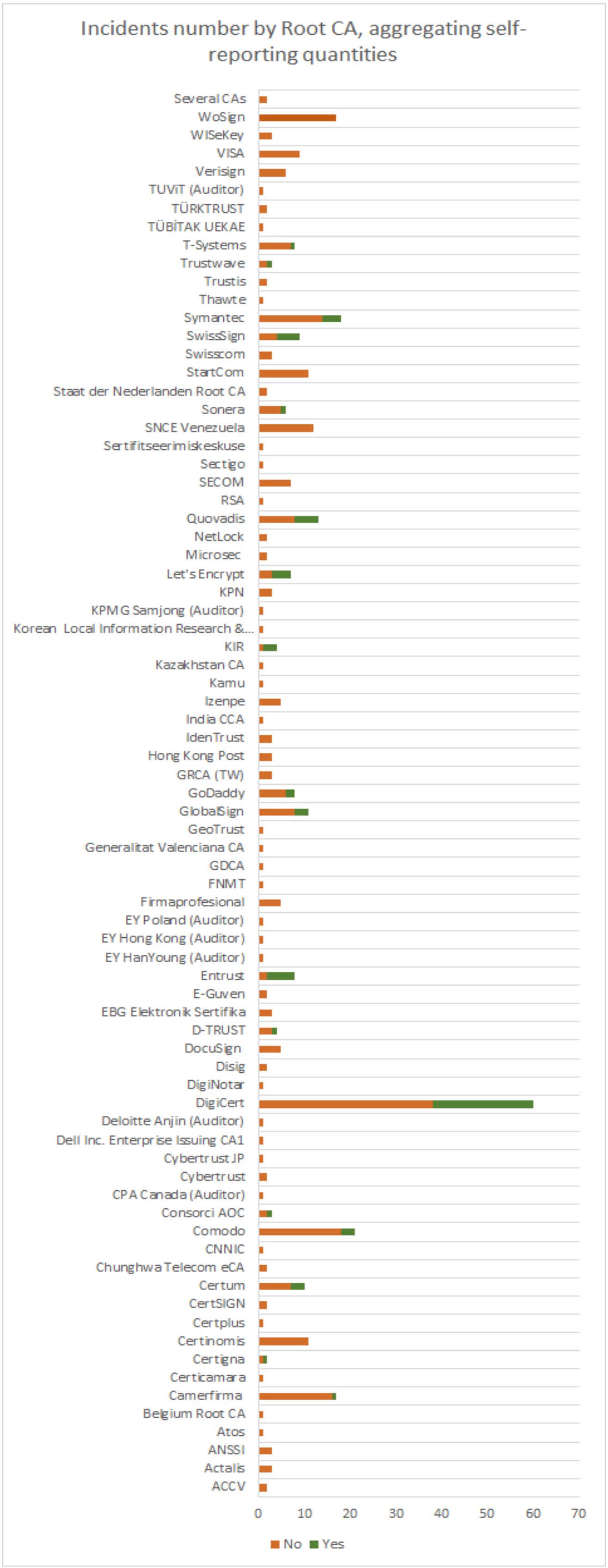


Fig. APP1.12. Root CAs and self-reporting comparison

Electronic copy available at: https://ssrn.com/abstract=3425554

F. Incidents per year per Root/Problematic CA

Electronic copy available at: https://ssrn.com/abstract=3425554

Root CA	2001	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
ACCV	0	0	0	0	0	0	0	0	0	0	0	2	0
Actalis	0	0	0	0	0	0	0	0	0	0	3	0	0
ANSSI	0	0	0	0	0	0	1	0	0	2	0	0	0
Atos	0	0	0	0	0	0	1	0	0	0	0	0	0
Belgium Root CA	0	0	0	0	0	0	0	0	0	0	1	0	0
Camerfirma	0	0	0	0	0	2	0	0	0	0	7	8	0
Certicamara	0	0	0	0	0	0	0	0	0	0	0	1	0
Certigna	0	0	0	0	0	0	0	0	0	1	0	1	0
Certinomis	0	0	0	0	0	0	0	0	0	0	7	3	1
Certplus	0	0	0	0	0	0	0	1	0	0	0	0	0
CertSIGN	0	0	0	0	0	0	0	0	0	0	2	0	0
Certum	0	0	0	1	0	0	0	0	0	0	2	4	3
Chunghwa Telecom eCA	0	0	0	0	0	2	0	0	0	0	0	0	0
CNNIC	0	0	0	0	0	0	0	1	0	0	0	0	0
Comodo	0	1	0	2	3	0	0	0	2	3	8	2	0
Consorci AOC	0	0	0	0	0	0	0	0	0	0	3	0	0
CPA Canada (Auditor)	0	0	0	0	0	0	0	0	1	0	0	0	0
Cybertrust	0	0	0	0	0	0	0	2	0	0	0	0	0
Cybertrust JP	0	0	0	0	0	0	0	0	0	0	0	1	0
Dell Inc. CA1	0	0	0	0	0	0	0	1	0	0	0	0	0
Deloitte Anjin (Auditor)	0	0	0	0	0	0	0	0	0	0	0	1	0
DigiCert	0	0	0	0	0	0	0	4	1	5	22	23	5
DigiNotar	0	0	0	0	1	0	0	0	0	0	0	0	0
Disig	0	0	0	0	0	0	0	0	0	0	2	0	0
DocuSign	0	0	0	0	0	0	0	0	0	1	2	2	0
D-TRUST	0	0	0	0	0	0	0	1	0	0	2	1	0
EBG Elektronik Sertifika	0	0	0	0	0	0	0	0	0	0	0	3	0
E-Guven	0	0	0	0	0	0	0	1	1	0	0	0	0
Entrust	0	0	0	0	2	0	0	0	0	0	1	3	2
EY HanYoung (Auditor)	0	0	0	0	0	0	0	0	0	0	0	1	0
EY Hong Kong (Auditor)	0	0	0	0	0	0	0	0	0	1	0	0	0
EY Poland (Auditor)	0	0	0	0	0	0	0	0	0	0	0	0	1
Firmaprofesional	0	0	0	0	0	0	0	1	0	0	2	2	0
FNMT	0	0	0	0	0	0	0	0	0	0	0	1	0
GDCA	0	0	0	0	0	0	0	0	0	0	0	1	0
Generalitat Valenciana CA	0	0	0	0	0	0	0	0	0	1	0	0	0
GeoTrust	0	0	0	0	0	0	0	0	0	0	1	0	0
GlobalSign	0	0	0	0	1	0	0	0	0	3	6	0	1
GoDaddy	0	0	0	1	0	0	0	1	0	0	4	2	0
GRCA (TW)	0	0	0	0	0	0	0	0	0	0	1	1	1
Hong Kong Post	0	0	0	0	0	0	0	0	0	2	0	0	1
IdenTrust	0	0	0	0	0	0	0	0	0	0	2	1	0
India CCA	0	0	0	0	0	0	0	1	0	0	0	0	0
Izenpe	0	0	0	0	0	0	0	0	0	1	4	0	0
Kamu	0	0	0	0	0	0	0	0	0	0	1	0	0
Kazakhstan CA	0	0	0	0	0	0	0	0	1	0	0	0	0
KIR	0	0	0	0	0	0	0	0	0	0	0	1	3
Korean LIRDI	0	0	0	0	0	0	0	0	0	0	0	1	0
KPMG Samjong (Auditor)	0	0	0	0	0	0	0	0	0	0	0	1	0
KPN	0	0	0	0	3	0	0	0	0	0	0	0	0
Let's Encrypt	0	0	0	0	0	0	0	0	1	1	2	3	0
Microsec	0	0	0	0	0	0	0	0	0	1	1	0	0
NetLock	0	0	0	0	0	0	0	0	0	0	2	0	0
QuoVadis	0	0	0	0	0	0	0	0	0	1	5	6	1
RSA	0	0	0	0	0	0	0	1	0	0	0	0	0
SECOM	0	0	0	0	0	0	0	2	0	0	2	3	0
Sectigo	0	0	0	0	0	0	0	0	0	0	0	0	1
Sertifitseerimiskeskuse	0	0	0	0	0	0	0	0	0	0	1	0	0
Several CAs	0	0	0	0	0	0	0	0	0	0	1	0	1
SNCE Venezuela	0	0	0	0	0	0	0	0	0	3	9	0	0
Sonera	0	0	0	0	0	0	0	0	0	1	1	3	1
Staat der Nederlanden Root CA	0	0	0	0	0	0	0	0	0	0	2	0	0
StartCom	0	1	0	0	1	0	0	2	2	1	4	0	0
Swisscom	0	0	0	0	0	0	0	1	1	0	0	1	0
SwissSign	0	0	0	0	0	0	0	0	0	1	2	6	0
Symantec	0	0	0	0	0	0	0	2	3	5	8	0	0
Thawte	0	1	0	0	0	0	0	0	0	0	0	0	0
Trustis	0	0	0	0	0	0	0	0	0	0	2	0	0
Trustwave	0	0	0	0	0	1	0	1	0	0	1	0	0
T-Systems	0	0	0	0	0	0	0	0	0	2	2	4	0
TÜBİTAK UEKAE	0	0	0	0	0	0	0	0	0	1	0	0	0
TÜRKTRUST	0	0	0	0	1	0	0	0	0	0	1	0	0
TUViT (Auditor)	0	0	0	0	0	0	0	0	0	0	0	1	0
VeriSign	1	2	1	2	0	0	0	0	0	0	0	0	0
VISA	0	0	0	0	0	0	0	0	2	2	4	1	0
WISeKey	0	0	0	0	0	0	0	0	0	0	2	1	0
WoSign	0	0	0	0	0	0	0	0	11	6	0	0	0
TOTAL	**1**	**5**	**1**	**6**	**12**	**5**	**2**	**23**	**26**	**45**	**135**	**96**	**22**

TABLE APP1.IV

INCIDENTS PER ROOT CA PER YEAR

Electronic copy available at: https://ssrn.com/abstract=3425554

Problematic CA	2001	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019
ABB	0	0	0	0	0	0	0	0	0	0	1	2	0
AC Camerfirma	0	0	0	0	0	0	0	0	0	0	1	0	0
AC Infrastructure	0	0	0	0	0	0	0	0	0	2	0	0	0
ACCI	0	0	0	0	0	0	0	0	0	0	0	1	0
ACCV	0	0	0	0	0	0	0	0	0	0	0	2	0
Actalis	0	0	0	0	0	0	0	1	0	0	3	0	0
ADACOM	0	0	0	0	0	0	0	0	0	0	0	1	0
ADIF	0	0	0	0	0	0	0	1	0	0	0	0	0
Aetna	0	0	0	0	0	0	0	0	0	1	0	0	0
Amazon	0	0	0	0	0	0	0	0	0	0	1	0	0
AT&T	0	0	0	0	0	0	0	0	0	0	3	0	0
Atos	0	0	0	0	0	0	1	0	0	0	0	0	0
Auditor	0	0	0	0	0	0	0	0	1	1	0	4	1
Bayerische	0	0	0	0	0	0	0	0	0	0	0	1	0
Bechtel	0	0	0	0	0	0	0	0	0	0	0	1	0
Belgium Root CA	0	0	0	0	0	0	0	1	0	0	1	0	0
Camerfirma	0	0	0	0	0	2	0	0	0	0	4	2	0
Cartão de Cidadão	0	0	0	0	0	0	0	0	0	0	1	0	0
Certicamara	0	0	0	0	0	0	0	0	0	0	0	1	0
Certigna	0	0	0	0	0	0	0	0	0	1	0	1	0
Certinomis	0	0	0	0	0	0	0	0	0	0	2	3	1
CertSIGN	0	0	0	0	0	0	0	0	0	0	2	0	0
CertStar	0	1	0	0	0	0	0	0	0	0	0	0	0
CertSuperior	0	0	0	0	0	0	0	0	0	0	1	0	0
Certum	0	0	0	1	0	0	0	0	0	0	2	3	2
Chunghwa Tel	0	0	0	0	0	2	0	0	0	0	0	0	0
Cihaz Sertifikası Hizmet Sağlayıcısı	0	0	0	0	0	0	0	0	0	1	0	0	0
Comodo	0	0	0	1	0	0	0	0	2	3	3	2	0
ComodoBR	0	0	0	0	1	0	0	0	0	0	0	0	0
Consorci AOC	0	0	0	0	0	0	0	0	0	0	2	0	0
cPanel CA	0	0	0	0	0	0	0	0	0	0	1	0	0
CrossCert	0	0	0	0	0	0	0	0	0	0	2	0	0
Cybertrust	0	0	0	0	0	0	0	1	0	0	0	1	1
Cybertrust JP	0	0	0	0	0	0	0	0	0	0	0	1	0
Deutsche Telekom	0	0	0	0	0	0	0	0	0	0	0	1	0
DFN	0	0	0	0	0	0	0	0	0	0	1	1	0
DigiCert	0	0	0	0	0	0	0	0	0	0	3	6	3
DigiCert Federated	0	0	0	0	0	0	0	0	0	0	0	1	0
DigiCert Sdn. Bhd	0	0	0	0	2	0	0	0	0	0	0	0	0
Digidentity	0	0	0	0	0	0	0	0	0	0	1	0	0
DigiNotar	0	0	0	0	1	0	0	0	0	0	0	0	0
DigitalSign	0	0	0	0	0	0	0	0	0	0	0	1	0
Disig	0	0	0	0	0	0	0	0	0	0	2	0	0
DnB NOR ASA	0	0	0	0	0	0	0	0	0	1	0	0	0
Docapost	0	0	0	0	0	0	0	0	0	0	1	0	0
DocuSign	0	0	0	0	0	0	0	0	0	0	0	1	0
D-TRUST	0	0	0	0	0	0	0	1	0	0	2	1	0
EC-ACC	0	0	0	0	0	0	0	0	0	0	1	0	0
ECRaizEstado	0	0	0	0	0	0	0	0	0	1	1	1	0
E-Guven	0	0	0	0	0	0	0	1	1	0	0	0	0
EINS	0	0	0	0	0	0	0	0	0	0	0	1	0
Entrust	0	0	0	0	0	0	0	0	0	0	1	3	2
Equifax	0	0	1	0	0	0	0	1	0	0	0	0	0
e-Szigno	0	0	0	0	0	0	0	0	0	1	1	0	0
E-Tugra	0	0	0	0	0	0	0	0	0	0	0	3	0
Firmaprofesional	0	0	0	0	0	0	0	1	0	0	2	0	0
Fuji Xerox	0	0	0	0	0	0	0	1	0	0	0	0	0
GDCA	0	0	0	0	0	0	0	0	0	0	0	1	0
Gemnet	0	0	0	0	1	0	0	0	0	0	0	0	0
Generalitat Valenciana	0	0	0	0	0	0	0	0	0	1	0	0	0
Getronics	0	0	0	0	1	0	0	0	0	0	0	0	0
GlobalSign	0	0	0	0	1	0	0	0	0	3	1	0	0
GoDaddy	0	0	0	1	0	0	0	1	0	0	4	2	0
Google TS	0	0	0	0	0	0	0	0	0	0	0	0	1
GRCA (TW)	0	0	0	0	0	0	0	0	0	0	1	1	1
GTE CyberTrust Global Root	0	0	0	0	0	0	0	1	0	0	0	0	0
Hong Kong Post	0	0	0	0	0	0	0	0	0	2	0	0	1
IdenTrust	0	0	0	0	0	0	0	0	0	0	2	1	0
Incapsula	0	0	0	0	0	0	0	0	0	0	1	0	0

Electronic copy available at: https://ssrn.com/abstract=3425554

Problematic CA	**2001**	**2008**	**2009**	**2010**	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**	**2019**
InfoCert	0	0	0	0	0	0	0	0	0	0	2	0	0
Intel	0	0	0	0	0	0	0	0	0	0	3	0	0
Intesa	0	0	0	0	0	0	0	0	0	0	3	0	0
Izenpe	0	0	0	0	0	0	0	0	0	1	4	0	0
Justica CA	0	0	0	0	0	0	0	0	0	0	1	1	0
Kazakhstan CA	0	0	0	0	0	0	0	0	1	0	0	0	0
KDDI	0	0	0	0	0	0	0	0	0	0	0	1	0
Keynectis	0	0	0	0	0	0	0	1	0	1	2	1	0
KIR	0	0	0	0	0	0	0	0	0	0	0	0	3
Korean MOI & MOE CAs	0	0	0	0	0	0	0	0	0	0	0	1	0
KPN	0	0	0	0	1	0	0	0	0	0	1	0	0
Let's Encrypt	0	0	0	0	0	0	0	0	1	1	2	3	0
LuxTrust	0	0	0	0	0	0	0	1	0	0	0	0	0
MCS Holdings	0	0	0	0	0	0	0	1	0	0	0	0	0
Microsoft	0	0	0	0	0	0	0	0	0	0	2	1	0
MIDIGATE	0	0	0	0	0	0	0	0	0	0	0	1	0
MULTICERT	0	0	0	0	0	0	0	0	0	0	1	5	0
NetLock	0	0	0	0	0	0	0	0	0	0	2	0	0
NIC	0	0	0	0	0	0	0	1	0	0	0	0	0
NII	0	0	0	0	0	0	0	0	0	0	1	0	0
No Data (undisclosed RA)	0	0	0	0	1	0	0	0	0	0	0	0	0
PROCERT	0	0	0	0	0	0	0	0	0	3	9	0	0
QuoVadis	0	0	0	0	0	0	0	0	0	0	3	5	1
RapidSSL	0	2	0	1	0	0	0	0	0	0	1	0	0
Santander Digital Signature	0	0	0	0	0	0	0	0	0	0	0	1	0
SBCA	0	0	0	0	0	0	0	0	0	2	0	0	0
SECOM	0	0	0	0	0	0	0	0	0	0	1	1	0
Sectigo	0	0	0	0	0	0	0	0	0	0	0	0	1
Sertifitseerimiskeskuse	0	0	0	0	0	0	0	0	0	0	1	0	0
Servision	0	0	0	0	0	0	0	1	0	0	0	0	0
Several (related to Symantec)	0	0	0	0	0	0	0	0	0	0	3	0	0
Several CAs	0	0	0	0	0	0	0	0	0	0	1	0	1
Siemens	0	0	0	0	0	0	0	1	0	1	2	0	0
SIGNE	0	0	0	0	0	0	0	0	0	0	0	1	0
StartCom	0	1	0	0	0	0	0	1	2	4	8	0	0
StartSSL	0	0	0	0	1	0	0	1	0	0	0	0	0
Swiss Government	0	0	0	0	0	0	0	0	0	0	2	0	0
Swisscom	0	0	0	0	0	0	0	1	1	0	0	1	0
SwissSign	0	0	0	0	0	0	0	0	0	1	2	6	0
Symantec	0	0	0	0	0	0	0	0	2	2	3	0	0
Szafir	0	0	0	0	0	0	0	0	0	0	0	1	0
TeleSec ServerPass EV	0	0	0	0	0	0	0	0	0	0	0	1	0
Telia	0	0	0	0	0	0	0	0	0	0	0	1	0
TeliaSonera	0	0	0	0	0	0	0	0	0	1	1	2	1
Terena	0	0	0	0	0	0	0	0	0	0	0	1	0
Thawte	0	1	0	0	0	0	0	0	1	0	0	1	0
TI Trust Technologies	0	0	0	0	0	0	0	0	0	1	1	0	0
Treasure Department SSL	0	0	0	0	0	0	1	0	0	0	0	0	0
Trinity Health	0	0	0	0	0	0	0	0	0	0	0	1	0
Trust Italia	0	0	0	0	0	0	0	0	0	0	0	1	0
Trustico	0	0	0	0	0	0	0	0	0	0	0	2	0
Trustis	0	0	0	0	0	0	0	0	0	0	2	0	0
Trustwave	0	0	0	0	0	1	0	1	0	0	1	0	0
T-Systems	0	0	0	0	0	0	0	0	0	0	1	1	0
TÜBİTAK	0	0	0	0	0	0	0	0	0	0	1	0	0
TÜRKTRUST	0	0	0	0	1	0	0	0	0	0	1	0	0
UniCredit	0	0	0	0	0	0	0	0	0	2	0	0	0
Untuit	0	0	0	0	0	0	0	0	0	0	0	0	1
USERTrust	0	0	0	1	0	0	0	0	0	0	1	0	0
UTN - USERFirst - HW	0	0	0	0	1	0	0	0	0	0	0	0	0
Verisign	1	0	0	1	0	0	0	1	0	0	1	0	0
Verizon	0	0	0	0	0	0	0	0	1	0	1	1	0
Virginia Tech	0	0	0	0	0	0	0	0	0	0	1	0	0
VISA	0	0	0	0	0	0	0	0	2	2	4	1	0
Vodafone	0	0	0	0	0	0	0	1	0	1	0	0	0
VR IDENT EV	0	0	0	0	0	0	0	0	0	1	0	0	0
Wells Fargo	0	0	0	0	0	0	0	0	0	0	2	0	0
WISeKey	0	0	0	0	0	0	0	0	0	0	2	1	0
WoSign	0	0	0	0	0	0	0	0	11	3	0	0	1
Yandex	0	0	0	0	0	0	0	0	0	0	0	1	0
TOTAL	**1**	**5**	**1**	**6**	**12**	**5**	**2**	**23**	**26**	**45**	**135**	**96**	**22**

TABLE APP1.V

INCIDENTS PER PROBLEMATIC CA PER YEAR

Electronic copy available at: https://ssrn.com/abstract=3425554

G. Rogue Certificates

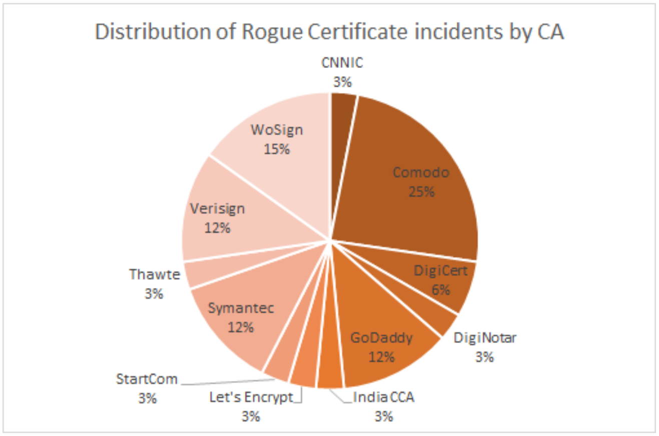


Fig. APP1.13. Rogue Certificates, distribution

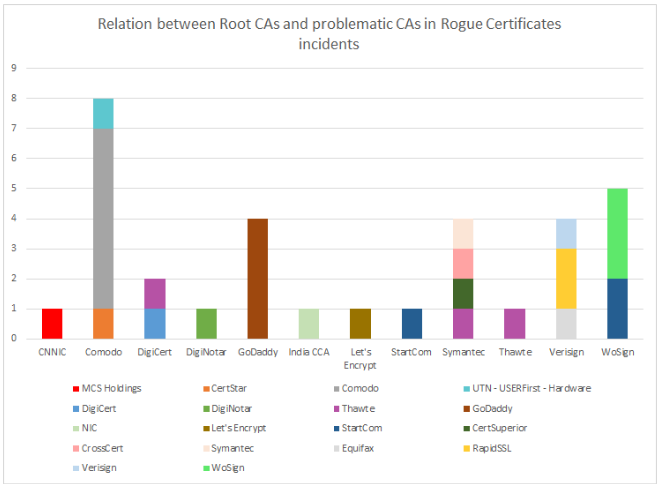


Fig. APP1.15. Root CAs and Problematic CAs relation in Rogue Certificate incidents

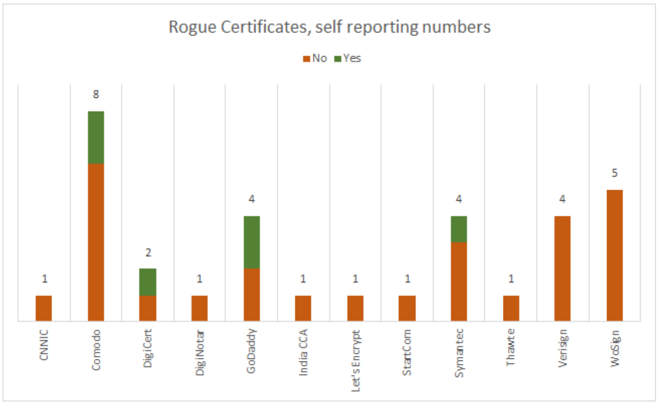


Fig. APP1.14. Rogue Certificates, self-reporting numbers

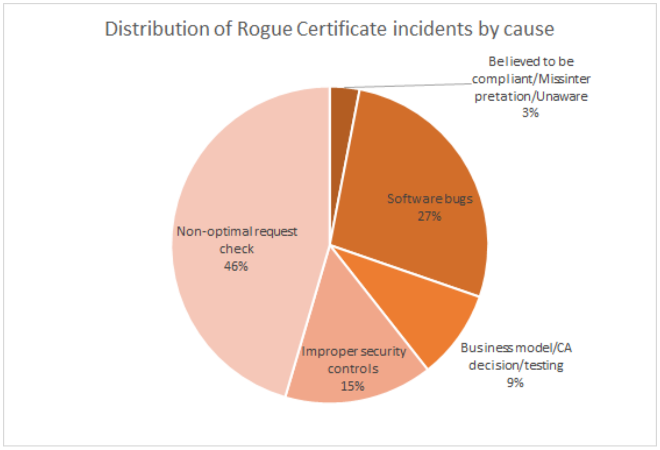


Fig. APP1.16. Causes of Rogue Certificates

Electronic copy available at: https://ssrn.com/abstract=3425554

XI. Appendix 2 - Notorious cases of CA misbehavior

In this Appendix of our work, we describe some of the most alarming misbehavior acts of our studied CAs.

1) 2008 – Comodo: An employee (actually the owner) from *StartCom* (another CA) was able to get a domain certificate for "mozilla.com" from *CertStar*, a Registration Authority (RA) of *Comodo*. There was no validation at all at the RA in the certificate request.

2) 2011 – DigiNotar: This is one of the most known CA's incidents. *DigiNotar* was in control of various CAs and was the provider of the Dutch government PKI. *DigiNotar's* servers were compromised and several rogue certificates were issued (531 known to be exact). Evidence found tied the hacker of this breach to others in the same year, like *Comodo's*, *StartCom's*, *GlobalSign's* and possibly *KPN's*. The attacks began on the first days of June and ended at the end of July. In the middle, *DigiNotar* detected the intrusion but could not stop it. Besides, the company did not disclose the breach to the web browsers or other interested parties, and when asked their reports were vague or misleading. The first rogue certificate detected was for a Google domain. Almost every OSCP request for the rogue certificates came from Iran, so there were suspicions about an MITM attack performed in that country. In general, the cases that came not from Iranian IPs were Tor endpoints or VPNs. *DigiNotar* presented several security vulnerabilities described in a post-mortem analysis, for example: no antivirus in servers, no domain separation between CAs, possible access to them from the management LAN, weak admin passwords, software outdated and not patched. It is strange how the company passed the required audits and complied with the BR requirements. As a consequence of the breach, *DigiNotar's* root certificates were revoked by the web browsers given that they could not know the totality of the rogue certificates issued, the company was investigated by the Dutch government, and, finally, it declared bankruptcy.

3) 2011 – Entrust and GTE CyberTrust (Verizon): DigiCert Sdn. Bhd, a Malayan subordinate CA of these two, issued 22 certificates with weak keys (512 bits). These certificates also lacked a definition for the key usage. Therefore, they were duplicated and used to sign malware. Quickly after the finding, Microsoft, Google and Mozilla stopped trusting in *DigiCert Sdn. Bhd*.

4) 2011 – TÜRKTRUST: An Intermediate Certificate Authority of this Turkish company "mistakenly" issued two intermediate CA certificates instead of simple ones. Over one year after that, one of these CA certificates was used to create a rogue certificate for google.com in order to be installed in a reverse proxy firewall to inspect TLS traffic inside a Turkish government office, a case of MITM. While *TÜRKTRUST* stated that the issuing of that certificate was due to an error in the production environment and that there was no evidence of unethical use of it, there were speculations about the possibility that the government office was sniffing the Gmail accounts and google.com searches of its employees. The forge was detected by Google itself, which stopped to trust in that intermediate CA and in *TÜRKTRUST* EV certificates. Similar measures were taken by Microsoft and Mozilla.

5) 2012 – Trustwave: Trustwave issued a subordinated CA certificate to an organization in order to implant a Data Loss Prevention (DLP) system (a similar scenario to *2011 - TÜRKTRUST*, in which certificates were used for MITM inside an organization). The certificate would live in an HSM so *Trustwave* explained that there was no risk of the certificate being stolen. However, there were suspicious about the practice, especially in Mozilla public boards, that the certificate could be used to issue new certificates to any domain, given that it was a subordinated CA one. Besides, there are other mechanisms to implant a DLP without incurring in such a risk. Right after the issue took public notice, Trustwave made a statement saying that there was enough evidence to trust in the HSM and in the DLP system, therefore, the subordinated CA certificate would not be miss-used or stolen. Nevertheless, they said they would not continue to provide this kind of certificate to this special use anymore. There was a lengthy discussion in Mozilla to decide whether the *Trustwave* root certificate should be revoked or not, given that their root certificate program policy may have been violated. Finally, it was not, but it was another display of practices of dubious ethics from a CA.

6) 2013 – ANSSI: Again, an MITM use for a digital certificate. In this case, certificates were issued to Google's domains (like in *2011-TÜRKTRUST*) by an intermediate CA linking back to *ANSSI*, the French certificate authority and national agency of computer security. Like previous cases, Google blocked the intermediate CA certificate, restricted the CA capabilities (*ANSSI* in this case) and notified other browsers.

7) 2014 – India CCA: Another forging of Google's certificates, maybe with MITM goals, issued by an intermediate CA. A rogue certificate for a Yahoo!'s domain was issued as well. In this case, the root CA was the *Indian CCA*, and the intermediate CA used belonged to the *NIC*, National Informatics Centre of India. The former stated that the issuing process in the latter was compromised. Anew, Google blocked the intermediate CA certificate, restricted the CA capabilities (*NIC* in this case) and notified other browsers.

8) 2014 – CNNIC: Another MITM scenario, like in previous cases. In this opportunity, the root certificate belonged to the China certificate authority and domain name registry, *CNNIC*, and the intermediate CA was *MCS Holdings*. Google identified that the private key of the intermediate CA was misused to create rogue certificates for Google's domains. The private key was placed in a proxy server instead of in a secure place. After Google detection of the rogue certificates, it notified *CNNIC* and other browsers and revoked the trust in the intermediate CA. Some days after the incident, Google and Mozilla also stopped trusting in *CNNIC* certificates (the root CA), until it followed the Certificate Transparency project.

9) 2014 – StartCom: StartCom had a business plan where it was issuing digital certificates for free, but charging for their revocation. In 2014 "Heartbleed" was discovered,

Electronic copy available at: https://ssrn.com/abstract=3425554

and a massive revocation of digital certificates was needed. There was a big discussion about the practices of *StartCom*, charging for the revocation of digital certificates that may be considered as "compromised". While *StartCom* had a valid point in its business model, it may not have provided optimal welfare for the PKI network in these cases, where domains may have decided to keep using faulty digital certificates in order not to pay for their revocation.

10) 2014 – Kazakhstan Gov. Root CA: The government of this country tried to deploy an MITM scheme for all its citizens. For that, it tried to force the citizens to install a national root certificate in their devices, while at the same time it applied to root programs like Mozilla's one. The MITM would have been done in the national ISP, however, due to the massive online spread and rejection of the plan, and the denial of Root Program's owners to include the Root certificate in their store, the idea was canceled.

11) 2015 – WoSign: In 2015 several issues were detected in the root CA *WoSign*. Two of them were related to their free certificates' validation process. First, this process included testing on a port greater than 50,000 in the domain server. This was not against the CA operation guidelines, however, Mozilla's root program did not allow the use of ports greater than 1,024 for the validation. The researcher that discovered the problem reported it to Google and it contacted *WoSign*, which fixed the issue. However, the problem was never disclosed to Mozilla as it was required by its program policy and it did not appear in the subsequent BR audit document. Second, if a certificate requester was in control of a subdomain, then he could have a certificate for the entire domain. A researcher tested this scenario with a university's subdomain and obtained a certificate for the university's main domain, and with his GitHub subdomain (user account, account_name.github.[com—io]) and got certificates for github.com, github.io, and www.github.io. The researcher notified the issue related to GitHub to *WoSign*, and they revoked the certificate. However, nothing happened with the university certificate, therefore, the company did not check for previous occurrences of the vulnerability. This was not the behavior expected for a Mozilla's CA program participant. Besides, Mozilla was not notified of the issue and it did not appear in the subsequent BR audit document. Another important problem presented in *WoSign* was the issuing of digital certificates with the same serial number, due to applications' bugs or server miss-functioning. In other cases, duplicated certificates were issued. Even they were contacted again by Google because the latter identified several cases where *WoSign* was not following its own Certification Practice Statement, which violates the Baseline Requirements. Other minor problems were the use of not recommended hashing algorithms (SHA-1 and SM2). Finally, the last dubious practice of *WoSign* in 2015 was the silently and obscure purchase of *StartCom*, another CA company. The different Root Programs demand from the CAs notifications of any transferal of ownership of CA operations.

12) 2015 – Symantec: This issue was discovered by Google. First, they detected an erroneous EV certificate issued by *Thawte*, owned by *Symantec*. After notifying *Symantec*, the latter responded that not only that certificate but 23 certificates for domains were issued for testing purposes, without the consent of the owners. Google investigated the problem and found that more than 23 certificates were issued. After that, *Symantec* stated that 164 certificates were issued for 76 valid domains, and 2,458 for non-registered domains. Consequently, Google demanded a detailed post-mortem analysis of the issue from *Symantec*, the action plan to correct it, and adequate audit reports. Besides, *Symantec* new certificates issued should support Certificate Transparency after these incidents.

13) 2016 – StartCom/WoSign: WoSign and its subsidiary *StartCom* (after silently purchasing it) were untrusted and removed from root CA programs by all the major browsers. Several incidents led to this decision. First, there was some evidence that the patch management in WoSign servers was far from the expected. Second, they backdated some certificates to avoid the prohibition of using SHA-1 after 01/01/2016. Finally, *StartCom* was able to issue certificates for its *StartEncrypt* brand, rooted in *WoSign* and using the same trick of backdating to use SHA-1 after it was forbidden. This situation also made clear that *StartCom* was using the same infrastructure that *WoSign*, despite *WoSign's* negation about the purchasing. Therefore, after the issues of 2015 and now these of 2016, Mozilla carried out an investigation that concluded with the revocation of the root certificates of both CAs and their expulsion from the browsers CA programs. In addition, *WoSign* auditor (*Ernst & Young* - Hong Kong) was not accepted anymore by Mozilla. Nonetheless, this ban for *WoSign* was not forever and they reapplied to be part of the programs again (with the name *WoTrust*). On the other hand, *StartCom* closed its business. In Mozilla's decision there were not considered some other vulnerabilities presented in *StartCom*. First, *StartEncrypt* had several bugs in the validation process, which may have allowed hackers to obtain certificates for domains they did not control. Second, *StartSSL* email validation was flawed as in various other cases.

14) 2016 – Comodo: Comodo intended to register the trademark "Let's Encrypt" in order to block the expansion of the free digital certificate provider with that name.

15) 2016 – SNCE Venezuela: PROCERT, a Venezuelan CA had several issues in 2016. These issues ranged from certificates with weak keys to wrong information in the certificates' fields. Even though some of the issues were solved in that year, much of these continued to exist in 2017. This led to Mozilla performing an investigation. Other issues that were discovered subsequently were: the use of internal private names or IP addresses in the certificate fields, lack of completeness in the certificates' fields, noncompliance with RFCs, bugged OCSP servers, usage of SHA-1 and non-random serial numbers. These and other unacceptable events related to this CA raised a final alert in Mozilla, that started to question the appropriateness of its practices and including the CA's certificate in its Root Program.

Electronic copy available at: https://ssrn.com/abstract=3425554

16) 2017 – Symantec: In 2017, the accumulation of several issues led to the revocation of *Symantec's* root certificate from all root CA programs. In addition to the problems previously commented and some other minor issues, there was a general lack of alignment to the BR and missing audit reports for various *Symantec's* subordinate CAs. The final event that derived in the sanction by the Root Programs' owners was the identification (by Google) of nearly 30,000 miss-issued certificates rooted to *Symantec* (*Symantec* denied Google's claim, though). The total distrust in *Symantec* was led by Google. This revocation affected not only *Symantec*, but also its subordinates: *Thawte*, *GeoTrust* and *RapidSSL*. The distrust was planned in different phases, and the final revocation was scheduled for October 2018. However, some months after the planned total revocation was set by the root programs, *Symantec* sold its CA business to *DigiCert*.

17) 2018 – Trustico: Due to *Symantec's* CA problem (or scandal), *Trustico*, a former reseller of it that was starting to partner with *Comodo* asked *DigiCert* (which purchased *Symantec's* CA business) for the revocation of 50,000 digital certificates issued by *Trustico* and rooted to *Symantec*. After *DigiCert* negative arguing that there was no sign of compromise in these certificates (and only the owner of the certificate could ask for its revocation), *Trustico* sent to *DigiCert* an email with the private keys of 23,000 of these certificates. After this "forced" compromise of the certificates, *DigiCert* revoked them and contacted the certificates' owners. This fiasco revealed that *Trustico* was storing the private keys of the certificates that they created and issued and, in addition, it was a display of lack of ethical behavior from the company the sending through an insecure channel as the email such a piece of confidential information as private keys are, just to achieve business goals. Besides, a security researcher found a severe vulnerability in *Trustico's* website right after the issue was made public, which led to a momentary closure of its service. Finally, there were suspicions about the mass mail sent by *DigiCert* to these certificates' owners after the revocation. Some said that the "Baseline Requirements" state that only the reseller can contact its customers.

18) 2018 – Several CAs: Finally, we close this list with examples of validated and legitimate certificates issued by CAs that are used in malicious sites, with the goal to perform phishing attacks. This is not a new trend, phishing sites with legal digital certificates have existed since the beginning of the miss-conception that the padlock near the URL and the HTTPS protocol mean that a website is the legitimate one and also the one expected by the user. With the use of *https://crt.sh* (using the wildcard '%' symbol and interesting names), we look up digital certificates issued to suspicious sites. When we tested them, most were blocked by the web browser (Google Chrome Version 69.0.3497.100) or by the antivirus (Avast Free Antivirus) for they were already reported phishing sites. Our educated guess is that the non-blocked were because they were very new (created the same day that we tested them, or instead for phishing they were used for scams). The Table APP2.I is an example of our findings.

Domain	**Cert ID**	**CA**	**Issued**
cei8.stage.paypal.com	875687124	DigiCert	2018-10-19
go-paypal.com	857703714	Amazon	2018-10-13
exchange-paypal.com	877958274	Let's Encrypt	2018-10-20
mysecurespaypal.com	857811997	Comodo	2018-10-13
iduser25653xxxxmobile-paypal.com	853166095	Comodo	2018-10-12
www.offers-paypal.com	790166353	CloudFlare	2018-09-16
github.paypal.com	871470742	DigiCert	2018-10-18

TABLE APP2.I
DIGITAL CERTIFICATES IN SUSPICIOUS SITES

Again, these are just examples, not a complete list. We tried to include as many CAs as possible, but not all the cases we found for a given CA. These examples are related to the legitimate site "paypal.com", however, we tested other promising sites and got the same CAs names that are listed in Table APP2.I. Most of the cases involved *Let's Encrypt* for it gives digital certificates for free. We did not search for sites using Unicode characters and punycode encoding names for "IDN homograph attacks". Nonetheless, we guess that there are phishing websites using this approach with valid digital certificates as well. In conclusion, for this last item of the list, as long as domain registrars continue accepting new web domains with dubious names, CAs continue issuing digital certificates to these sites, and the general user (and some savvy users too) continue to believe that the padlock near the URL or the use of the HTTPS protocol mean that the site is legitimate and secure; phishing sites will continue to exist even with digital certificates; and as a consequence, credentials from web users will continue to be stolen.

Electronic copy available at: https://ssrn.com/abstract=3425554

XII. Annex 1 - Timeline of main events in the PKI history

In Fig. ANNEX.1 we summed up the main events related to PKI (some of these studied in the content of the paper in addition to other milestones) and displayed them as a "timeline". The time-lapse ranges from the seventies, rapidly jumping to the nineties, and ending with the last changes in brands, distrusts, and guidelines and requirements versions. We believe it can be helpful for the non-savvy reader of this work to gain a good grasp about PKI, its main milestones and some of its issues.

In a nutshell, we can see that the fundamental elements that support PKI were initially developed in the nineties, above the concept of asymmetric cryptography established in the seventies. From 2000 to 2010, we can appreciate an expansion in the services and goods offered in PKI and, at the same time, further improvements in the technologies used; some of these as responses to actual failures or visible future problems. From 2010 to 2015, given that rogue certificates, MITM attempts and general miss-issued digital certificates became a constant problem and a serious undermining to the trust in PKI, Root Program Owners and web browser vendors started to implement and use technical solutions to cope with these incidents. The former, web browsers, also became stricter with the Root CAs in their programs, and more prone to distrust them after continuous incidents and bad practices. In recent years, we can see how these technical countermeasures have been improved with new features, while guidelines and requirements have continued to be upgraded.

Given that Root Program Owners have become stricter and that they have more tools at their disposal to detect PKI failures from CAs, we may appreciate more frequent Root CAs revocations from programs in the near future. As a consequence of this, the companies behind the web browser development and root programs may experience an increment in the power of their decisions within the PKI network; even more than the one that some of them may have now.

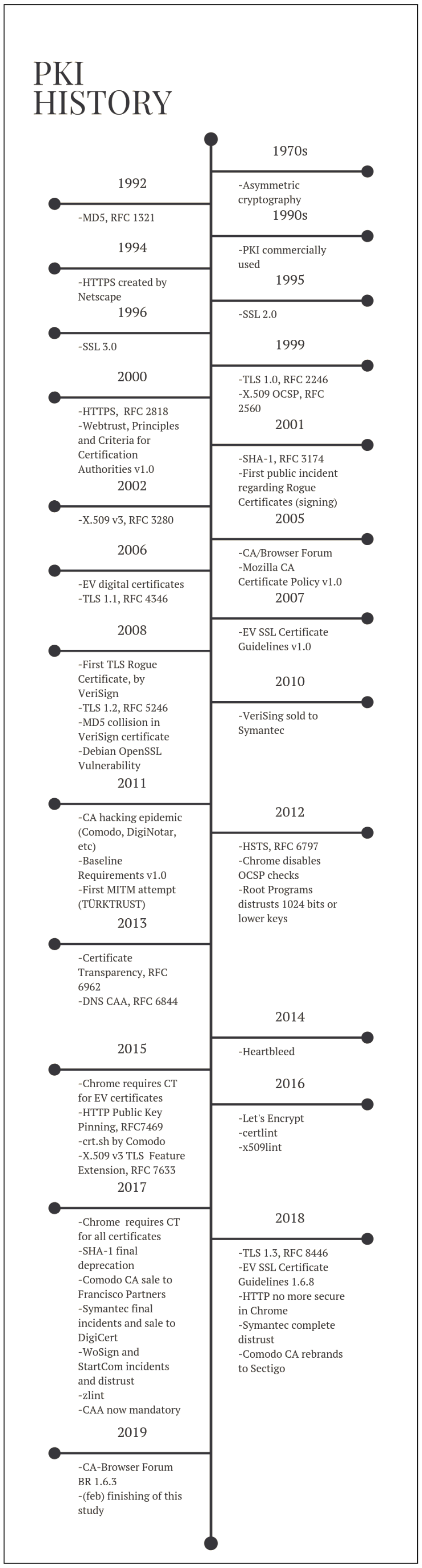

PKI HISTORY
1970s
-Asymmetric cryptography
1992
-MD5, RFC 1321
1990s
-PKI commercially used
1994
-HTTPS created by Netscape
1995
-SSL 2.0
1996
-SSL 3.0
1999
-TLS 1.0, RFC 2246
-X.509 OCSP, RFC 2560
2000
-HTTPS, RFC 2818
-Webtrust, Principles and Criteria for Certification Authorities v1.0
2001
-SHA-1, RFC 3174
-First public incident regarding Rogue Certificates (signing)
2002
-X.509 v3, RFC 3280
2005
-CA/Browser Forum
-Mozilla CA Certificate Policy v1.0
2006
-EV digital certificates
-TLS 1.1, RFC 4346
2007
-EV SSL Certificate Guidelines v1.0
2008
-First TLS Rogue Certificate, by VeriSign
-TLS 1.2, RFC 5246
-MD5 collision in VeriSign certificate
-Debian OpenSSL Vulnerability
2010
-VeriSing sold to Symantec
2011
-CA hacking epidemic (Comodo, DigiNotar, etc)
-Baseline Requirements v1.0
-First MITM attempt (TÜRKTRUST)
2012
-HSTS, RFC 6797
-Chrome disables OCSP checks
-Root Programs distrusts 1024 bits or lower keys
2013
-Certificate Transparency, RFC 6962
-DNS CAA, RFC 6844
2014
-Heartbleed
2015
-Chrome requires CT for EV certificates
-HTTP Public Key Pinning, RFC7469
-crt.sh by Comodo
-X.509 v3 TLS Feature Extension, RFC 7633
2016
-Let's Encrypt
-certlint
-x509lint
2017
-Chrome requires CT for all certificates
-SHA-1 final deprecation
-Comodo CA sale to Francisco Partners
-Symantec final incidents and sale to DigiCert
-WoSign and StartCom incidents and distrust
-zlint
-CAA now mandatory
2018
-TLS 1.3, RFC 8446
-EV SSL Certificate Guidelines 1.6.8
-HTTP no more secure in Chrome
-Symantec complete distrust
-Comodo CA rebrands to Sectigo
2019
-CA-Browser Forum BR 1.6.3
-(feb) finishing of this study

Fig. ANNEX.1. Timeline of main events in the PKI history

Electronic copy available at: https://ssrn.com/abstract=3425554