The Economics of Cryptographic Trust: Understanding Certificate Authorities

by

Michael Alan Specter
B.A., Computer Science, George Washington University, (2010)

Submitted to the Institute for Data, Systems, and Society &
Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degrees of
Master of Science in Technology and Policy
and
Master of Science in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
February 2016

Signature of Author .....
September 30, 2015

Certified by .....
David D. Clark
Senior Research Scientist, CSAIL
Thesis Supervisor

Certified by .....
Daniel J. Weitzner
Principal Research Scientist, CSAIL
Thesis Supervisor

Accepted by .....
Professor Munther A. Dahleh
Acting Director, Technology and Policy Program

Accepted by .....
Professor Leslie A. Kolodziejski
Chair, Departmental Committee on Graduate Students

The Economics of Cryptographic Trust: Understanding Certificate Authorities

by

Michael Alan Specter

Submitted to the Institute for Data, Systems, and Society &
Department of Electrical Engineering and Computer Science
on September 30, 2015, in partial fulfillment of the
requirements for the degrees of
Master of Science in Technology and Policy
and
Master of Science in Electrical Engineering and Computer Science

Abstract

Certificate Authorities (CAs) play a crucial role in HTTPS, the mechanism that secures all of the web's most important communication; if it has a log-in page, it must use HTTPS. However, recent history is littered with instances of CAs unabashedly undermining the trust model of the web in favor of economic gain, causing catastrophic harm to users in the process. The purpose of this thesis is to understand how well user, domain owner, and browser vendor controls function in order to evaluate methods of realigning CA incentives.

Using a compendium of past incidents of CA failure as a series of natural experiments, along with a large dataset of all publicly available certificate collections, we find that it is possible to causally link a very slight increase in domain owners leaving a CA when a CA acts inappropriately. We further find that the technical architecture of the CA system leaves users without effective control over which CAs they trust, and that browsers face certain difficulty in distrusting larger CAs. The end result is a system where large CAs can unilaterally undermine the trust model of the web without clear repercussion.

Thesis Supervisor: David D. Clark
Title: Senior Research Scientist, CSAIL

Thesis Supervisor: Daniel J. Weitzner
Title: Principal Research Scientist, CSAIL

Acknowledgments

With the notable exception of space travel, no worthwhile work has ever been done in a vacuum, and this thesis is no exception. I owe a debt of thanks to a number of incredible people, both in and outside of MIT, for helping me these last few years. Indeed, without this amazing outpouring of support, this thesis would be far worse and I would be far less sane.

First and foremost, I would like to thank my advisors David D. Clark and Daniel J. Weitzner. Having great advisors, who work well as a team and want to help me work through tough and interesting problems, has been fantastic. This work is proof that having two perspectives on a problem is awesome, but having three can be even better.

Beyond my advisors, I am deeply indebted to those who put up with my constant discussion of cryptography and PKI systems, many of whom were unfairly subjected to earlier drafts of this thesis. Part of the joy of existing between groups and degrees is that I have many such people to thank, including Steven Bauer, Josephine Wolff, and my office mates and neighbors Ben Yuan, Micah Brodsky, professor Gerry Sussman.

I owe many thanks to my friends at MIT's Lincoln Laboratory, for providing both financial and moral support; specifically Tim Sally, Clark Wood, Chris Connelly, Doug Stetson, and John Wilkinson. Lincoln was my home for a long time before coming to MIT's main campus, and has shaped my thinking on security, research, and perseverance.

I would have been hopelessly lost without MIT's wonderful administrative staff; I now know from personal experience that good project managers and course secretaries can move mountains. For this I thank Marisol Diaz, Barbara DeLaBarre, and Susan Perez.

I also received a massive amount of help from those in industry who foolishly took time out of their day to help a lowly grad student. These include David Benjamin and Gary Belvin of Google and Kathleen Wilson of Mozilla, who helped me better understand OpenSSL, provided background on the CA system from a browser's

standpoint, and otherwise saved me an immense amount of time.

Finally, I would like to thank my family and friends. All challenges are made easier by knowing that there are those who want you to succeed, and know how to keep you grounded. Specifically, I am indebted to Cristina Logg, Dan and Linda Goodman, and my parents, Barry and Lainee Specter, for their guidance and support – without them I would have never have gained a healthy passion for both the arts and sciences, or have ever thought I could marry the two.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> "Having an encrypted conversation with a stranger may be like meeting that person in a dark alley. Whatever happens, there are no witnesses."
>
> David D. Clark & Marjory S. Blumenthal

There is a fundamental tension between the way in which humans exhibit trust, and the way in which computer systems are designed. Human trust is dynamic, fickle, and prone to change rapidly over time as individual or institutional reputations change. Conversely, programmatic trust is more than likely defined by the creator of the program, is ossified at the time of authoring, and reflects all of the inherent incentives and pressures that the creator is under. Trust between computer systems is therefore likely to change less dramatically with time, and be far less aligned with users' best interests. It is where computational trust and human trust intersect that progress can stop altogether, and the stakes can get much higher.

There are few better examples of this tension than in the development and use of cryptographic mechanisms, such HTTPS and the Certificate Authority (CA) ecosystem. HTTPS is the set of protocols over which all secure communication happens on the web. This includes most email, social networks, banks, and airlines — if a service has a login page, it most likely uses HTTPS. Worse, if it has a login page and does not use HTTPS, it is almost certainly insecure.

A fascinating part of HTTPS is that it depends on a sociotechnical system of trust that has evolved over a number of years. Beyond trusting browser vendors, users must

trust a list of third party organizations called *Certificate Authorities* (CAs), whose purpose is to vouch to the browser that the web server on the other end of your communication is actually who it says it is; CAs provide cryptographic proof that the connection is *authentic*. Indeed, encrypted communication without authentication is as insufficient as this chapter's quote implies; all the client can know is that it is speaking to someone over an encrypted connection, not who that person is, or if that communication is being actively intercepted and re-encrypted to the intended recipient.

The notion of using trusted third parties to provide authentication is not without real-world similarity. Driver's licences are trusted as a means of authentication because they are issued by the state, which bar tenders can then trust to be accurate when deciding to serve a young-looking person alcohol. Notaries are trusted to issue legal affidavits because they themselves are trusted and licenced by the state. Police officers have a badge that says they are licenced by the state to be trustworthy, and can be trusted to carry and use weapons in defense. Lawyers and doctors undergo licencing through professional organizations, and must maintain a level of decorum to remain members. It is in this exact same way that CAs are trusted to vouch for websites, however the root of their authority to do so comes from browser vendors rather than a government bureaucracy; as will become apparent later sections, CAs are poorly regulated and less well-understood than traditional licencing systems.

1.1 Context and motivation

The focus of this thesis is on the numerous ways in which the web's cryptographic trust model has been broken. Indeed, there have been quite a few instances of CAs mistakenly mis-issuing certificates, failing to protect themselves adequately, or undermining the trust of the system by intentionally issuing malicious certificates.[1] Recent advances in network scanning and general awareness of the importance of such failures has led to better measurement of the CA marketplace. However, while there have

[1]These failures are comprehensively enumerated in Section 3.6.

been a number of papers examining the CA market from a qualitative standpoint, few have been able to provide a thorough quantitative analysis.

Part of what makes the CA market worth studying is that there has been an industry-wide push to encrypt all Internet communication, with various actors creating incentives for web site owners to switch over to HTTPS. For instance, Mozilla, the nonprofit that makes the popular Firefox browser, is currently threatening to "deprecate" unencrypted connections such that any website that wants to use new browser features (such as future versions of HTML) will be forced to do so over HTTPS [54]. Similarly, Google has announced that it is using HTTPS as a signaling mechanism for website quality, the implication being that websites that use encryption will be more highly ranked than others [62]. Finally, HTTP/2, the newest version of the HTTP protocol that implements a myriad of performance enhancements, will work only over HTTPS for at least a few major browsers [48], specifically to act as an incentive for domain owners to switch over to encrypted communications.

These incentives have not been put in place without reason — there are moral, ethical, and practical security concerns that browser and operating system vendors are tackling head on. Even the Internet Engineering Task Force (IETF), the multi-stakeholder standards organization responsible for the development of the Internet, has stated that passive monitoring is an attack that future protocols should be designed to prevent [24]. It is doubtful that a world in which users are ubiquitously spied upon by malicious middlemen is a bright one, and so the computer science community is attempting to make an ubiquitously encrypted web a reality.

However, without a solid understanding of the economic nature of the CA system, it is unknown if the sum of these efforts are too little to encourage large-scale encryption, or, worse, if they are liable to force an evolution in the CA system that leads to less overall security. As we will see later in this thesis, recent history is littered with incidents of CAs actively undermining the security of the system, both through negligence and for outright economic gain, and forcing website administrators to support such a system may not be wholly effective. Quantitative economic analysis could be an invaluable tool for better understanding the incentive structure

of the CA market, which is crucial in providing insight into why efforts to replace the CA system with newer technologies have failed, and to understand which pressures on CAs could prove fruitful in realigning incentives.

1.2 Contributions of this thesis

There are two main objectives to this thesis. The first is to use publicly available market data to determine if social pressures such as reputation effects, in the form of bad press and public shaming, have been effective in keeping CAs in check in lieu of more direct regulatory sanctions. The other objective is to improve upon what little is known or understood about the tools that browser vendors and users can employ to punish CAs for acting inappropriately. In concert, these investigate the economic incentive structure surrounding CAs in order to develop a more complete picture of whether or not there is inherent reason for CAs to uphold the trust model of the web.

As mentioned in previous sections, other studies have attempted to analyze the economics of the CA ecosystem, but fall short of the analysis that is required to take actual action — none quantitatively examine the economic effect of a certificate authority behaving badly. For instance, it is currently unknown in the common literature if, after a certificate authority somehow behaves poorly, it also loses domain owners and users.

Part of the reason for this lack of analysis is a technical one: until recently, there has been very little publicly-attainable information on the changing state of the HTTPS market over time, so no such analysis could be done. Luckily, recent advances in scanning technology (as well as the Internet's general failure to switch to IPv6), have enabled new open datasets to contribute to the public discourse.

The contributions of this thesis are:

- An introduction to the CA system for a nontechnical audience
- A discussion of current and future efforts in fixing the CA ecosystem
- A detailed history of all known certificate authority failures
- A qualitative analysis of the CA system

- A quantitative analysis of the growth of the CA market place

1.3 Roadmap for future sections

The goal of the first few chapters is to provide a full understanding of why HTTPS is important, why the CA system is in such turmoil, and who, exactly, is involved. Chapter 2 begins with a discussion of what can happen in a world without HTTPS, pulling from real-world examples of actual harm that came from failing to encrypt communications. It continues to describe the HTTPS protocol itself, certificate path validation, and concludes with a survey of actors in the CA market. Chapter 3 continues by chronicling a subset of the various and sundry ways that the CA ecosystem has been undermined in recent history, focussing on categories of attacks as well as providing a timeline for the next few sections.

Chapters 2 and 3 exist to provide motivation for chapters 4 and 5, which include a qualitative and quantitative analysis of the CA market, a survey of proposed replacements, their efficacy, and a discussion of possible policy and technical solutions going forward.

Chapter 2

Understanding HTTPS

> "If you're looking for computer security then the Internet is not the place to be. If you think that you're an exception to the norm and you have a secure setup that communicates over the Internet you're probably mistaken... You, the government and consumer, should care and want software products to include security and authentication mechanisms."
>
> Pieter Zatko aka Mudge, Congressional Testimony [31]

This chapter presents a cursory background explanation of HTTPS, the TLS protocol, x.509 certificates, and certificate validation, as well as a survey of the market actors involved in the CA ecosystem. It begins by demonstrating why it is so important that websites communicate over HTTPS via historical examples of failures that would have been prevented, had the website or service used TLS. Next, the chapter dives into HTTPS and TLS protocol, focusing on the method used to validate whether or not a server is a trusted entity. This is done to inform the reader of the tools that each actor can use, how the system is meant to function on a general level, and what tools each actor has when the system goes awry or actors misbehave. Finally, this chapter presents past failed attempts at changing the trust model, as well as descriptions of improvements that appear to be impending. These serve as informative lessons on the difficulty of altering a very much entrenched trust model, which in-turn highlights the need for further economic analysis.

Advanced readers who are familiar with the technical aspects of HTTPS (Man

in the Middle attacks, Certificate Validation, etc.) may wish to skip ahead to the sections on the market actors themselves.

2.1 Understanding man in the middle attacks

It is perhaps easier to describe HTTPS in the context of what kind of harms can occur without it; the many forms of harm that can befall users when a transmission is sent unencrypted demonstrate the need for HTTPS. Broadly categorized as "Man in the Middle" (MITM) attacks, HTTPS attempts to prevent all of the maladies that occur when an adversary sits between a browser and a server, and is therefore able view or modify communication between the two. The bad guys in this situation can be anyone from oppressive governments, to ISPs, to wireless access providers, to anyone with a laptop and knowhow watching an unencrypted wireless network.

HTTPS is the encrypted version of HTTP, which in turn depends on a protocol called Transport Layer Security (TLS) or its predecessor Secure Sockets Layer (SSL); HTTPS can be thought of as HTTP over TLS or SSL.[1] Browsers commonly decorate the address bar with various green logos, locks, and other markings in order to indicate to a user that the website he or she is visiting is "secure." Specifically, these decorations indicate that the communication is encrypted, and that the browser has determined that the website itself is *authentic*; that it is actually the website the user intended to visit and not some malicious middleman.

Pervasive surveillance of unencrypted traffic is an obvious potential malady: Any website sent in the clear will, without a doubt, be readable to those in-between the user and a server. Unfortunately, users accessing even the most inane websites still have aggregate worth to advertisers, which, combined with the knowledge that ISPs already have about their users, becomes incredibly valuable. It is important to note that this is not a theoretical attack – AT&T and Verizon, looking to cash in on mobile advertising, inserted a header value into all outgoing traffic from their users' mobile

[1]Since SSL is effectively deprecated in favor of TLS, the rest of this thesis will refer mainly to TLS.

phones with a unique identifier that advertisers could then use to track users across websites. Unfortunately, this was implemented so poorly that any network actor, not just AT&T or Verizon, between the user and any system the user was accessing could effectively track AT&T and Verizon customers [13, 39].

Worse, unencrypted traffic may directly lead to users losing control of their accounts. Websites use what are called session cookies – unique IDs that identify a users – to authenticate and verify users post-login. When these are passed over the network in the clear, any malicious middleman can steal these identifiers and reuse them to impersonate users. While this was always an attack that was theoretically possible, a Firefox plugin was released in 2010 that made it painfully easy for anyone listening in on an unencrypted wireless network to get into any number of an unsuspecting user's accounts including Facebook, Twitter, Gmail, Amazon, and Reddit [23].

Active attacks — attacks which modify the communication between the user and server, or simply modify the users machine — can be even more insidious. For example, in 2010 it was discovered that many software vendors failed to cryptographically verify the source of software updates. Researchers Francisco Amato and Federico Kirschbaum released a tool called Evilgrade, a plugin framework which enabled attackers who controlled the user's network to exploit this update procedure to remotely infect users' computers. Common software platforms like Java, OSX, Safari, iTunes, and OpenOffice were all found to be vulnerable [25].

Such attacks may not only harm the infected user, but unlucky third parties as well. For instance, in March 2015 the Great Firewall of China was used to actively intercept and modify requests to Baidu.com, a China-based search engine. Interestingly, Baidu provided a service similar to Google Analytics, which many non-Chinese companies used to track visitors via javascript included from Baidu on their websites. Instead of tracking or otherwise maliciously affecting the user that visited these sites, the Great Firewall was used to intercept requests for Baidu's analytics script and replace it with one that would then DDoS specific anti-censorship GitHub projects [15]. In this case, the Chinese Government was able to bring down large swaths of a US company's service by exploiting unsuspecting users who were doing nothing more than

browsing the web. Had the connection to Baidu been done securely over HTTPS, this would likely have not been possible without direct collusion from Baidu itself.

2.1.1 Why authentication is important

It is tempting to claim that blindly adding more cryptography to the situation will resolve the majority of the above issues. The magic of public key cryptography allows anyone to encrypt a document using a public key, such that only the corresponding private key can decrypt the document. From this point, it is relatively easy to see how one might bootstrap a communications protocol — send the browser a public key over the network, have the browser encrypt a symmetric key or its own public key with the server's public key, and send that back to begin communicating. However removing unencrypted communication is only half the battle; unauthenticated communication is actually equivalent for most forms of attack.

Unfortunately, providing just a public key over the network is not enough to defend against all MITM attacks. Consider Figure 2-1. In this scenario, a malicious middle man intercepts the public key coming from the server, and passes along its own. It then creates a secure connection to the intended server, while creating another encrypted connection to the user's browser. Unfortunately, since the connection was unauthenticated, there is no way for a browser to know that the connection is being snooped upon, and the malicious middleman wins. The next sections, specifically Section 2.2.1, provide an explaination of how browsers avoid this issue by authenticating the connection.

2.2 Protocol Overview

TLS itself was created by the Internet Engineering Task Force (IETF), a multi-stakeholder standards body that has directed the development of Internet protocols, in one form or another, since shortly after the Internet's inception. The formal specification for TLS can be found in the IETF's TLS RFC, which the RFC summarizes in the following steps [20]:

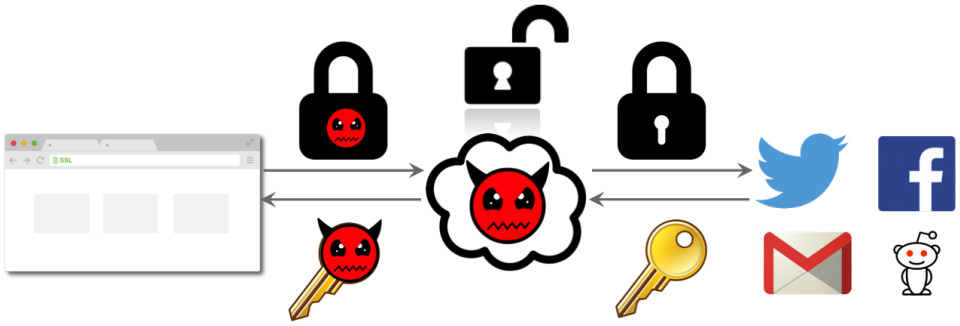


Figure 2-1: Illustration of how a malicious MITM can actively re-encrypt a connection to a user, rendering the naive unauthenticated encryption useless. In this straw-man, a public key is sent from a service's server (right), through an untrusted middleman (center) who then sends his own key to the user (left). Finally, the middleman creates an encrypted connection between himself and the user, and himself and the server, both believing the connection to be secure, even though the middleman can then snoop on all traffic.

1. Exchange hello messages to agree on algorithms, exchange random values, and check for session resumption.
2. Exchange the necessary cryptographic parameters to allow the client and server to agree on a premaster secret.
3. Exchange certificates and cryptographic information to allow the client and server to authenticate themselves.
4. Generate a master secret from the premaster secret and exchanged random values.
5. Provide security parameters to the record layer.
6. Allow the client and server to verify that their peer has calculated the same security parameters and that the handshake occurred without tampering by an attacker.

Much of the above gets into the minutiae of how the server and a browser negotiate which cryptographic algorithm and primitives should be used to secure the communication, as well as various performance enhancements. While important, and the source of much study, these parts of the protocol are not particularly interesting with regard to this thesis.[2] Instead, the focus of the rest of this section will deal

[2]In fact, there have been numerous times when implementation errors and protocol failures have broken HTTPS, including malicious middlemen forcing downgrades to known weak algorithms [14, 50], simple systems problems like memory bounds-checking errors [52], and flaws in the specific mode of cryptography used [27].

mainly with step 3 above, called Certificate Path Validation, which involves the CA system and is explained in detail below.

2.2.1 Certificate Path Validation

The goal of Certificate Path Validation is to prove to the browser that the server on the other end is authentic by creating a chain of trust to some entity that the browser knows a-priori. It does so by sending a list of certificates — x.509-encoded documents containing public keys and metadata for further identification and policy restrictions. The anatomy of an x.509 document is specified in an IETF RFC, and encoded as ASN.1, an exemplar certificate can be found in Figure A-2 as taken from Mac OSX's Keychain utility. Relevant parts of an x.509 certificate are:[3]

- **Issuer:** An ASN.1 encoded structure that can include the organization name (O), Common Name (CN), email address, country, and state of the Issuer of this certificate.
- **Subject:** Another ASN.1 encoded structure, similar to the Issuer, but provides information about this certificate rather than the certificate's issuer. If this is a certificate for a website, the Common Name (CN) must also be the Fully Qualified Domain Name (FQDN) of the website the browser went to.
- **Signature and Signature algorithm used:** Provides information allowing the recipient to cryptographically verify that the certificate has not been tampered with, and that the contents of the certificate were signed by the issuer.
- **Subject public key:** The public key of this certificate.
- **Validity Time:** The time in which this certificate is valid
  - **notBefore:** Time at which this certificate begins to be valid.
  - **notAfter:** Time after which this certificate is no longer valid.
- **Extensions (Optional):**
  - **subjectAltName:** A list of alternative domain names that this certificate represents, as in, alternative to the Subject's CN.

[3]For expediency, a number of irrelevant sections have been omitted, a more detailed explanation can be found in IETF RFC 5280, although I wish to issue a warning to the reader: many of the parts of the x.509 structure are obtuse and effectively ignored in practice [19].

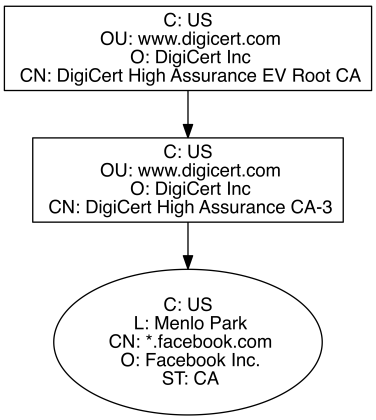


Figure 2-2: Trust chain for *.facebook.com leaf certificate, full subject displayed.

- **BasicConstraints:** Indicates if this certificate can sign other certificates, or if there are path length requirements.
- **nameConstraints:** If this is an intermediate or root certificate, this parameter restricts which domains this certificate can sign for. For example, this could be example.com, meaning that this certificate can only sign leaves whose domain begins with *.example.com.
- **CRL Distribution Points:** A list of URLs to download Certificate Revocation Lists for this CA or leaf.
- **Certificate Authority Information Access:** OCSP query point URLs.

The final certificate in the list, called the *leaf*, includes the domain for which the certificate is providing a public key to be used in communication with the server, and is signed by the corresponding private key of the public key in the second certificate in the list, which is signed by the third, and so on. Known to the browser a-priori, the final certificate, called the *root* is not actually sent; the browser consults a pre-populated database of certificates called the *root store*. The root, unlike the other certificates, is also *self-signed*; the private key that signed the certificate corresponds to the certificate's public key, and the issuer and subject fields are the same. Certificates between the root and leaf are known as *intermediate certificates* [19]. Figure 2-2 is an example trust chain for *.facebook.com in which the bottom node is the leaf, DigiCert High Assurance CA-3 is the intermediate, and the Digicert High Assurance EV RootCA is the root.

If one were to collect all certificate chains in one image, the CA system ideally looks something like a set of trees, with each root certificate signing intermediates

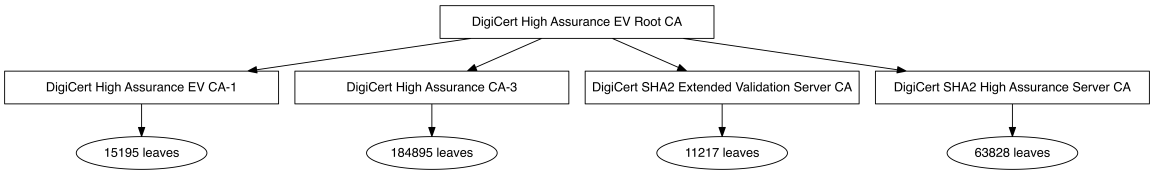


Figure 2-3: Aggregate trust tree for the specific Digicert root that was used to sign for *.facebook.com

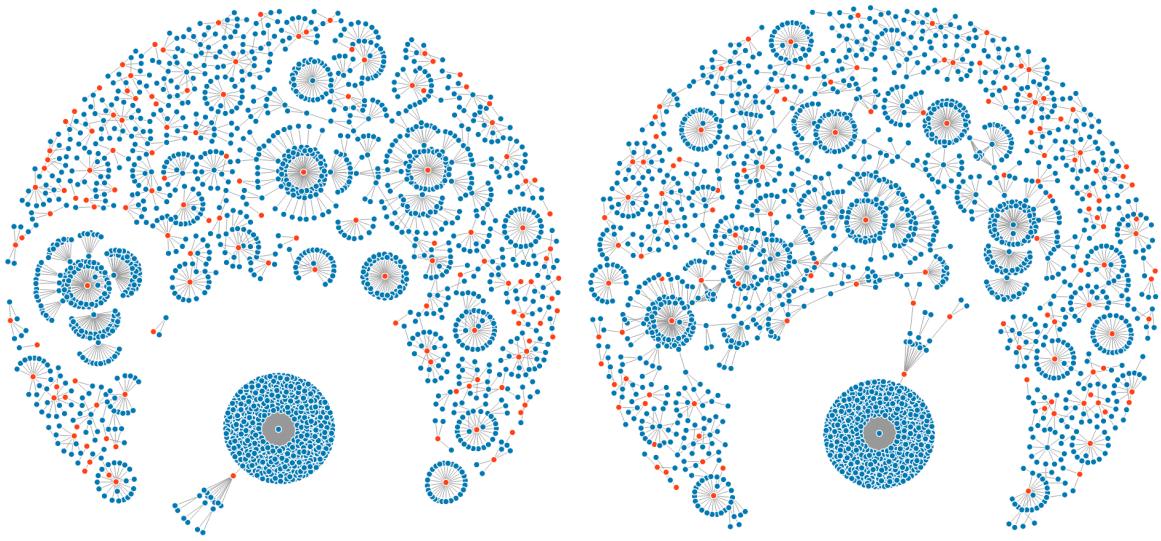
which then sign sets of leaves. Figure 2-3 displays the trust tree for the specific root certificate that signed for facebook.com in figure 2-2.

However, this is actually a very simplified representation of the CA system; companies have multiple roots, often obscurely named, and CAs regularly sign certificates that bridge two certificate authorities. This process is called *cross-signing*, and functionally allows certificate authorities to share trust. If, for instance, some browsers do not explicitly trust root A, but root A was cross-signed by a trusted root B, servers can present a path from a leaf issued by A through to B.

The aggregate result is a bit more complicated. With multiple root certificates, and multiple topologies for individual hierarchies, the result is more of an interconnected mesh. Figure 2-4 is one illustration, created from the dataset generated for this thesis, of the entire CA system, both with and without cross-signing. With cross-signing, there are far more intermediate nodes and far more interconnected graphs. Indeed, of the 170 roots currently trusted by Mozilla's root store, 52 have known cross-signers. It is important to note that this is a conservative number; there may be many more cross-signed certificates that are simply not used in common deployment, or have not been disclosed.

2.3 Trust model failure mitigations

Certificate Authorities and domain administrators are not impervious to attack or system failures. Private keys can be accidentally deleted, domain administrators servers can be broken into, intermediates will accidentally mis-issue certificates to the wrong entities. Such maladies will inevitably occur, and, especially when the system



(a) Entire system without cross-signing links (b) Entire system with cross-signing links

Figure 2-4: The entire CA system in a force directed graph, all orange nodes are directly owned by Certificate Authorities. Graphs (a) and (b) are equivalent except for the inclusion of cross-signing links — graph (b) includes cross-signing certificates, and has significantly more edges and is far more well-connected.

is protecting such valuable data, it crucial that the system have processes in place for fixing them.

The following are a number of the most effective ways in which this system detects and handles such trust model failures. Understanding these mitigations helps in explaining the CA failures discussed in Section 3.6, and in discerning the powers and abilities much of the market actors have in ensuring that certificates are distributed properly.

2.3.1 Revocation

An obvious requirement is to be able to remove trust in a particular certificate. An intermediate private key could be stolen, for instance, allowing a rouge actor to sign whichever leaf certificate she wants, and therefore snoop on any website's traffic. Without the ability to remove trust in a certificate, such an actor could continue doing so until the intermediate expired. There are three common ways in which this process, called revocation, is done: Online Certificate Status Protocol (OCSP),

Certificate Revocation Lists (CRLs), and out of band updates.

CRLs are exactly what they sound like, lists of revoked certificates that browsers should no longer trust. As part of the certificate verification process, browsers download these lists via a link embedded in metadata in yet another X.509 extension, and then check against these lists in order to determine if trust in a certificate is still valid.

OCSP is an inversion of the CRL model — instead of downloading the list of revoked certificates, the browser queries the certificate authority about the status of a particular certificate. Finally, browsers have been known to push updates through their software update mechanism that explicitly distrusts certain certificates.

It should be noted that all of these procedures share a specific flaw: they soft-fail. Should a CA's OCSP query not return, the certificate is accepted as valid. There is little doubt that such tools are near worthless in the event of a MITM attack, as an attacker could easily block access to the OCSP or CRL url, and their malicious certificate would be accepted. Google's Chrome browser has a partial mitigation for this, by using Chrome's update process itself a mechanism for pushing aggregated CRLs to the browser itself, similar to normal out of band updates [2, 5].

2.3.2 Public Key Pinning

Public key pinning is the process of solidifying the chain of trust for a particular leaf certificate by specifying which leaf, intermediate, and/or root certificate may be used to sign for this connection. There are two ways in which pinning can occur, the first is by getting the browser to add your key via an out of band push, as Google has done with their domains via Chrome. The second is via an IETF standard called HTTP Public Key Pinning (HPKP), a trust on first use mechanism that specifies the pinning procedure when a browser first connects to a domain [57].

The upshot of all of this is that pinning restricts the attack surface of a domain's certificate; should an intermediate be compromised or otherwise mis-issue a certificate for your domain, the browser should reject the malicious certificate, notifying the browser vendor in the process. The downside is that it increases the operational risk of the domain as well — should the intermediate a leaf uses be compromised or

invalidated, or, if the leaf itself is pinned, the certificate itself be compromised, the domain might be unreachable. For instance, Cryptocat, an encrypted chat service, "committed pinning-suicide" by switching intermediate CAs between renewals, effectively removing themselves from the web until a new out of band update by browsers removed the pin [7].

2.4 Planned improvements

While the above sections depict the CA system as it currently stands, there are a few impending changes that are worthwhile to mention. These are improvements that either have been accepted by standards organizations, that have stood up infrastructure to begin existing imminently, and have enough backing to be at least somewhat likely to be used.

2.4.1 Let's Encrypt

Let's Encrypt is a new CA that will provide free DV certificates to any domain that requests it using an automated validation scheme called ACME. The operative word is free — Let's Encrypt will both issue and revoke certificates for no charge, unlike all other known certificate authorities.[4] The project is a collaboration between Akamai, Cisco, the Electronic Frontier Foundation, BBN, University of Michigan, and Stanford [35].

While the goals of the program are certainly laudable, there are questions as to Let's Encrypt's long-term economic viability. The implicit argument behind Let's Encrypt is that the cost of protecting private keys, maintaining security audits, running revocation servers, and responding to requests is less than the costs commonly imposed by CAs, which can be maintained via philanthropic donations. According to Cloudflare, a Content Delivery Network and hosting provider, the mass revocation during the HeartBleed incident alone would have cost one of the less-massive

[4]StartCom and WoSign will both issue certificates for free, but charge a fee to revoke.

CAs over $400,000 in monthly bandwidth costs alone, assuming bandwidth costs of $10Mbps [38], let alone the rest of a CA's responsibilities.

2.4.2 Certificate Transparency

Certificate Transparency (CT) is an accountability measure created by Google that is currently being standardized by the IETF into an RFC. The idea behind is that CAs will issue certificates to publicly auditable logs which are then monitored for mis-issuances. Since the log itself is stored in a data structure that allows for quick lookups, browsers will query these logs to ensure that a certificate exists prior to validating a connection; If a certificate is not in the log, the browser rejects the certificate [1, 40].

Incentives for CAs to begin rolling out Certificate Transparency have been incredibly heavy-handed. Google plans to eventually make CT mandatory for all EV certificates in Chrome, however the Certificate Transparency RFC is still not completed [12]. None of the other browser vendors including Mozilla, Microsoft, and Apple have commented on whether or not they plan on supporting CT in the future.

2.5 Alternative trust models

Over the years, researchers, fed-up with flaws in the CA system, have shaped many proposals for new and interesting trust models that improve upon the existing architecture in some way. Each of these have been popularly proposed, even standardized, but all have failed to gain commercial or widespread traction — indeed despite their individual attractiveness, none of the following have been included by default in the current version of any major browser. These failures stress the importance of understanding how proliferation of standards work, and how the economic aspects of trust must be taken into account.

2.5.1 DNS-based Authentication of Named Entities

DNS-based Authentication of Named Entities (DANE) bootstraps trust using the DNS system as a method of conveying certificate information — essentially making the DNS a trust anchor in conjunction with CAs by presenting a hash of the certificate as a part of the DNS record itself. The upshot is that both a CA and the DNS could then provide two factors of authentication for the owner of a domain.

Unfortunately, DANE is very unlikely to be used in the near future. First, DANE relies on the DNS to be secure, which requires a separate standard called DNSSEC, another set of DNS extensions, to be in widespread use. DNSSEC deployment has been slow [36], leading DANE to go unimplemented in major browsers. The only browser to have natively supported DANE is Chrome, but Google, citing latency issues and weakness in many of the cryptographic primitives used in the current DNSSEC deployment, removed DANE in favor of pursuing Certificate Transparency and working with the current CA system [7].

2.5.2 Perspectives and Convergence

Perspectives and Convergence are two systems that use an alternative method of discovering and providing certificates to the browser called *multipath probing*. In a multipath probing scheme, independent *notaries* fulfill the role of a CA by scanning the Internet and downloading every certificate into a database. The certificate may be self-signed or signed by a CA. The idea is that, when the browser attempts to validate the certificate, it sends a hash of that certificate to a notary which then returns which domain that certificate belongs to, and if that certificate is valid[60].

Convergence builds upon Perspectives to provide further security and privacy guarantees. Specifically, it sends the browser's query through a set of notaries in an encryption scheme that prevents any individual notary from knowing what domain a specific browser is querying for.[5] The result is that no notary can know which user was going to which website, a great privacy improvement.

[5]The scheme itself is done similarly to how TOR routes through a circuit of nodes, decrypting layers of an encryption onion along the way.

The result of either system is one in which CAs no longer have any power, and in which the user has *trust agility*; he or she can distrust a specific notary without invalidating large swaths of the Internet. This represents a huge technical and moral benefit over the current model, in which removing a specific untrusted CA would impact an untold number of websites [37].

However, Convergence's failure highlights the fact that economic factors must be taken into account when designing trust models; it is the economic structure of the CA system that makes Convergence a very unlikely candidate. Adam Langley, a prominent developer of Google Chrome's enhancements on SSL/TLS, argues that the only entity that would have a direct incentive to run notaries would be the browser. So, Convergence, if included, would consist of notaries run mainly by Google, "so the design boils down to Chrome phoning home for certificate validation. That has both unacceptable privacy implications and very high uptime requirements on the notary service [34]."

Chapter 3

Market Actors

> "I hacked Comodo from InstantSSL.it, their CEO's e-mail address mfpenco@mfpenco.com
> Their Comodo username/password was:
> user: gtadmin password: globaltrust
> Their DB name was: globaltrust and instantsslcms"
>
> Comodohacker

Although the structure of the CA system is pretty complex, the relationship between actors in the CA ecosystem are relatively simple to understand. There is little to no government regulation or oversight (excluding government CAs themselves), every CA's certificates are considered equally valid by the browsers that honor them (certificates are, with few exception, perfect substitutes), and the powers that each actor has is well enumerated by the technology involved.

This section provides an introductory analysis of the actors responsible for the CA system, beginning with an enumeration of the actors, their responsibilities, and some descriptive information about the tools and abilities each of them has to align the incentives of other actors.

Finally, a comprehensive list of known CA failures and their resolution is presented and discussed, with an emphasis on the greater community's response as it pertains to the tools and abilities each actor has.

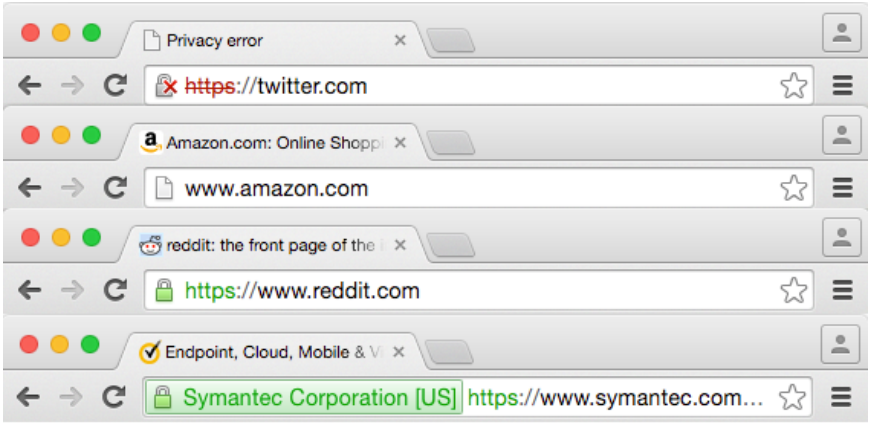


Figure 3-1: All possible browser decorations for Chrome version 45.0.2454.85. In order from top to bottom of intended assurance: path or certificate validation failure, no HTTPS, Domain Validated HTTPS, and Extended Validation HTTPS.

3.1 Browser Vendors

Browsers are the ultimate authority when it comes to trust — they control which CA certificates are allowed in its database of root certificates, called the *root store*, the standards that each CA must follow in order to be accepted into the root store, and the browser decoration that occurs when the communication is considered secure.

Another power browsers have is to decide on browser decoration, which attempts to provide information to the user about exactly how trusted the site is. Certain CAs can sign certificates with an extension known as *Extended Validation* (EV), which enables the green bar seen in Figure 3-1, as opposed to the regular green lock commonly seen in *Domain Validated* (DV) sites. The idea is that DV certificates are only meant to demonstrate that the owner has proven that he or she owns the domain, whereas EV requires further verification.

Each browser vendor has their own application process for inclusion in the root store, with differing levels of openness about what qualifies a CA. Most, with the notable exception of Mozilla's Firefox, elide away this responsibility to the underlying OS. Mozilla, being a community driven project, is very open about the inclusion process and removal process; removals and inclusions are often publicly discussed in their Google Group mozilla.dev.security.policy, all removals and inclusions are further

discussed and voted on in their public bug tracker, and their inclusion policies exist in their publicly accessible wiki [49]. Table 3.1 provides cursory information about the top five most-used browser vendors.

Browser	Marketshare	Browser Developer Type	Certificate Store Type
Internet Explorer	52.17%	OS Vendor	OS Keystore
Chrome	29.49%	Web Services, OS Vendor	OS Keystore
Firefox	11.68%	Nonprofit	Contains its Own
Safari	4.97%	OS Vendor	OS Keystore
Opera	1.27%	Web services	OS Keystore

Table 3.1: List of the five most common browsers, their marketshare, what sort of industry the browser developer is in, and whether or not they use the OS's certificate store [51] as of September 2015.

Removing trust in a CA is a bit of a trade-off for browser vendors. For obvious reasons, distrusting a CA should effectively remove websites signed by these CAs from the set of sites a browser can connect to. Worse, having fewer CAs increases the browser's dependence on other CAs, and similarly limits the field for Domain Owners. The upshot is that trusting fewer CAs yields a lower attack surface, and the action of removing trust in a CA can be punitive retaliation against CAs that invalidate the CA trust model, serving as a warning shot to other CAs.

3.2 Certificate Authorities

Certificate Authorities are organizations that own the private keys of the roots mentioned in the Certificate Path Validation (Section 2.2.1) description. From a systemic point of view, CA responsibilities include maintaining the security of their own private keys, signing leaf certificates, and accurately vetting whose certificate they sign to ensure that they are not malicious middlemen.

In practice, Certificate Authorities rarely sign website (leaf) certificates directly with the root. As implied in Section 2.2.1, roots are usually used to sign an intermediate certificate, which then is used to sign a leaf. There are two main reasons CAs choose to create intermediates.

The first is technical; every time a CA uses their root private key to sign a certificate, it exposes that private key to theft or malicious use. A compromised root would be catastrophic — all leaf certificates under the root would be effectively compromised, browsers would most likely remove the root from their root stores, and, even if the CA were able to get another root though the browser inclusion processes, all of the leaf certificates would have to be re-issued. In contrast, if an intermediate is compromised, the CA can revoke the intermediate, and only a subset of the leaves would need to be reissued under a new intermediate, which the CA can issue almost instantaneously. It is therefore useful for CAs to diversify into intermediates in order to ameliorate risk.

Second, it is common practice for CAs to delegate their authority to allow for organizational compartmentalization. It makes sense to use different intermediates for different purposes — certificates can be used for a laundry list of verification schemes, and having separate intermediates for code signing, email, and web certificates makes organizational sense.

Similarly, many CAs have subsidiary businesses in order to exist in foreign countries, or may enter into a contract with another company or organization which then distributes their own certificates. These CA organizations are known as *Subordinate Authorities (SAs).* Often, subsidiary businesses are not given intermediate certificates, but are instead entrusted with an account to create leaves via out of band coordination with a CA; these entities are called *Reselling Authorities* or RAs. Finally, there are certain companies that CAs will trust to sign certificates for their own domains, called *Enterprise Authorities* or EAs, which might be restricted to a particular subdomain by the nameConstraints extension.

There are a number of different organizations that have gained a certificate in the root store. These include governments, universities, as well as local and international businesses. However, as one might expect, the largest CAs with the most leaves tend to be for-profit businesses, and it is therefore these commercial organizations that are the main focus of this thesis.

3.3 Standards Bodies

There are only a few standards bodies outside of those that decide on cryptographic primitives that have relevancy to the CA system. Specifically, the IETF and the CA/Browser Forum (CA/B) are largely responsible for any sort of CA governance or coordination. Both of these organizations follow a relatively open multistakeholder model in which all affected parties are invited to attend, but differ in explicit purpose.

The IETF, whose RFCs are quoted throughout this thesis, deals mainly with standardizing specifics of protocols, but has increasingly been using its influence to push browsers and web companies toward ubiquitous encryption.

The CA/B acts as a way of aligning the policies of the browser vendors and certificate authorities. They set a list of "baseline requirements" (BRs) — minimum universal requirements agreed to by various CAs that CAs must follow in order to be accepted into root stores. Further, the CA/B sets standards for authentication of EV certificates, and acts as a coordinating body between browser vendors, operating system vendors, and CAs.

3.4 Users

There are many obstacles between the average user and their ability to make decisions about which CAs they actually trust. Many users do not have the background to understand, or wish to understand, the issues of cryptographic trust. Cormac Hurley, a scientist at Microsoft research, argues that this phenomenon is actually a rational economic choice; that users are balancing the added frictional cost of following time-consuming advice against the cost of an actual breach [29]. Others argue that the usability of much encryption software is so poor that it is functionally impossible for the average user to protect themselves.[1]

It therefore unlikely that nontechnical users will take any sort of action in the CA space — in fact, most of the literature surrounding users involved with encryption

[1]See [61], as well as any of the myriad of papers that followed it.

is about herding them into using encryption properly in the first place. For example, there have been numerous studies on attempting to convince users that clicking through browser certificate warnings is a poor choice. It appears that these studies have come to some success, although this does little to explain to users why ignoring such warnings is a bad idea [9].

Even in cases where knowledgeable business and IT centers can curate their users root stores for them, users have only a few powers available to them when dealing with certificate authorities, none of which are ideal. Users may switch which browser or operating system they use in order to change root stores and revocation policies, but, as mentioned in 3.1, the root stores are fairly homogeneous.

Similarly, users can manually remove or explicitly distrust CA certificates from their own browser's root stores. However, trusting fewer CAs will inherently limit the user's ability to connect to particular websites, and, without a-priori knowledge of who this specific CA signed for, there's little way that any user can know which CAs are worth distrusting.

3.5 Domain Owners

Domain owners are the end customers for CAs — they purchase leaf certificates to allow for authenticated communication to their servers. Conceptually, domain owners can decide to purchase their certificate from any CA since CAs are largely identical; outside of EV certificates and punitive action, browsers tend to treat certificates as equally valid, and will accept the connections from any of them interchangeably while displaying the same browser decoration.

It is this autonomy of selection which makes domain owners interesting from an economic standpoint. Since certificates are perfect substitutes, a purely rational domain owner should purchase their certificate from the cheapest possible source, holding customer service factors constant.

However, it is often true in other markets that the moral aspects of doing business with a company may hold sway over consumer action. Consumers often purchase

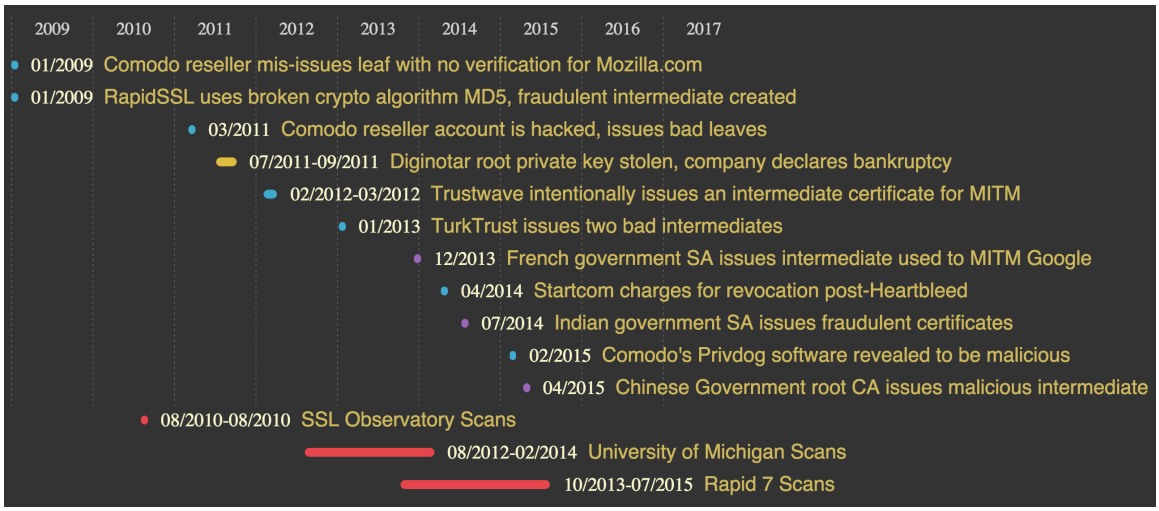


Figure 3-2: A timeline of CA events since late 2008. The top events are CA failures that undermined the CA PKI, while the bottom 3 red lines are time periods for which there are monthly IPv4 certificate collection scans. Blue lines indicate commercial failures, purple indicates a governmental CA failure, and yellow indicates both.

locally grown or responsibly grown foods, for instance, even though they are more expensive. A central question of this thesis is if domain owners, as a group, follow this sort of responsible consumerism. Put another way, should a CA act irresponsibly, will domain owners switch to another trust provider, or do they tend to do what is more economically prudent? Later sections of this thesis attempt to answer this and other questions about Domain Owners.

3.6 Certificate Authority failures and responses

An important part of any discussion surrounding the CA system's trust model must include its failure modes. Enumerating these past failures allows us to understand what tools the actors delineated in the previous sections are actually willing and able to use. It also highlights the fact that CAs have not always been well behaved, often for reasons that appear provide economic or geopolitical benefit. Figure 3-2 is a full timeline of all known unfortunate instances in which CAs have willingly or unwillingly undermined the PKI since 2008.

This section expands on this figure, providing a comprehensive history of cer-

tificate authority failures and community responses to illustrate how market actors interact in the case of CA misbehavior. Instances are listed chronologically, include a description of the incident, the browser community's response, and provides further discussion and analysis where appropriate.

- **01/2009:** Comodo reseller mis-issues leaf for Mozilla.com
  - **Description:**

    A Comodo reseller known as CertStar issued a fraudulent certificate for *.mozilla.com, without any sort of authentication or verification that the requester actually owned the domain. Luckily, the requester was Eddy Nigg, a security researcher from Startcom, another CA [22].
  - **Community Response:**

    Nigg publicly alerted Mozilla, who then began an investigation into the Comodo's reseller. Representatives from Comodo then fired CertStar as an RA, and issued a revocation for all certificates issued via CertStar's account [43].

- **01/2009:** Many CAs continue to use MD5 after shown to be broken
  - **Description:**

    MD5, a cryptographic hash algorithm, had been known to be broken for a number of years prior to 2009. Still, many companies, including Verisign, Thawte, and RSA, continued to use MD5 as a part of their signature algorithm for trusted roots and intermediates. Security researchers Alexander Sotirov and Jacob Appelbaum created a proof of concept intermediate certificate, giving them unconstrained power to sign MITM certificates.
  - **Community Response:**

    Verisign ceased using the vulnerable roots immediately after the vulnerability was disclosed, and offered free revocation and re-issuance under another root from that point forward. MD5 certificates are no longer considered valid in modern browsers [58].

- **03/2011:** Comodo reseller instantssl.it hacked
  - **Description:**

    An Iranian calling himself ComodoHacker broke into instantssl.it, a Comodo RA. The attacker managed to steal the credentials necessary to self-issue certificates for Google, Yahoo, Skype, and Mozilla.org.
  - **Community Response:**

    Google discovered the attack, blocked the fraudulent certificate for their domain, and alerted Comodo. Google owed their discovery to their newly implemented certificate pinning, which, instead of allowing for malicious MITM, refused the connection and notified Google. Comodo investigated the RA, found that their account had been violated, and issued a report stating that the attack was executed by a government entity in Iran [18]. There has been little proof that the attacker was actually from the government, instead appearing to be an inexperienced lone actor [16].
- **07/2012 - 09/2012:** Diginotar hacked, root key stolen
  - **Description:**

    Diginotar, a dutch subsidiary owned by an American corporation, was broken into by ComodoHacker, their root private key stolen. The scale of the resulting hack was immense; fraudulent certificates were issued for Google, comodo.com, android.com, digicert.com, logmein.com, microsoft.com, and more.
  - **Community Response:**

    Google, again, was the first to notice the attack, after again receiving notice from their user's browsers that public key pins were not matching for Google domains. After alerting Mozilla and blocking a few of the known bad leaves, a number of days passed before further action was taken. Diginotar happened to be the CA responsible for a large number of Dutch government websites.

This represented a conundrum for browsers — removing trust in Diginotar's known-compromised roots immediately would result in removing the Dutch government's sites from the web. Further, Diginotar had been cross-signed with a number of other CAs, resulting in the need to explicitly block a number of intermediates and cross-signing certificates. Trust was finally removed, and Diginotar filed for bankruptcy [28, 44].

- **02/2012** Trustwave intentionally issues an intermediate certificate for MITM
  - **Description:**

    Trustwave, an American CA, issued an intermediate certificate to a company expressly so that the business could perform MITM attacks on its own employees. Realizing that the practice itself was wrong, Trustwave took the unusual step of admitting to the problem and revoking the intermediate [59].
  - **Community Response:**

    Other than explicitly revoking trust in the malicious intermediate, there was much debate at Mozilla about the correct action to take against Trustwave. On one hand, revoking trust in the root could provide a disincentive for other CAs who had done the same to follow suit, but allowing Trustwave without repercussion might similarly send a too passive message. Eventually, Mozilla and others decided to allow Trustwave to remain a part of the root store [45].

- **01/2013:** TurkTrust issues two bad intermediates
  - **Description:**

    A Turkish CA named TurkTrust accidentally issued two intermediate certificates to companies wishing to purchase normal leaf certificates. All of this was fine until said businesses began to use them to perform MITM, again on their own employees.

- **Community Response:**

  TurkTrust's mistake was caught, again, by Google's use of public key pinning on Google domains. The malicious intermediates were revoked, but no other punitive action was taken against TurkTrust[41, 46, 3].

- **12/2013:** French Government SA, ANSSI, issues fraudulent certs

  - **Description:**

    A French government SA for the ANSSI was found to have issued an intermediate which then signed a certificate for google.com.

  - **Community Response:**

    The malicious certificate was again found via the public key pinning done by Google. Rather than completely distrusting the ANSSI root, Google took the step to restrict the CA such that leaves would only be accepted if they were for French country-code TLD's (.fr, .gp, .gf, etc.) [4, 32]

- **04/2014:** Startcom charges for Revocation after Heartbleed

  - **Description:**

    Startcom has a unique business model in which they do not charge for certificate issuance, but do charge in the event of revocation. The result is that companies have a disincentive to revoke lost certificates. In 2014, a protocol bug in OpenSSL caused a vast number of websites to accidentally disclose their private keys, resulting in the immediate need for them to revoke their certificates. Startcom refused to waive their fee, which many viewed as a betrayal of trust and violation of CA policies [47, 42]

  - **Community Response:** None, outside of some mild public shaming; Startcom remains part of the CA Ecosystem and no action was taken.

- **07/2014:** Indian governmental SA issues bad intermediates

  - **Description:**

An Indian governmental SA called the National Informatics Center (NIC) was caught issuing certificates for Google and Yahoo domains. The organization's certificates were actually intermediates of an Indian company, the Indian Controller of Certifying Authorities (India CCA). Uniquely, India CCA's root certificate was only installed in Microsoft's root store, so Firefox was unaffected.

  - **Community Response:**

    The certificate was discovered, again, because of Google's certificate pinning in Chrome. India CCA revoked all of NIC's intermediate certificates, and Google limited India CCA's root to a very restrictive set of domains [6].

- **04/2015:** Comodo's Privdog discovered to accept all certificates

  - **Description:**

    Privdog is an application written by Comodo that has been marketed as a tool to block advertisements and enhance user privacy. Far from protecting users, the application injects "trusted partner" advertisements into the user's browser and was written so poorly that it managed to break TLS in the process; any system with PrivDog would effectively trust any certificate presented to it. The malware functioned by installing a malicious root certificate in the OS's root store, and man in the middling every TLS connection by resigning the incoming certificate with its root. The end result was that *any* incoming certificate chain was considered valid, malicious or not.

  - **Community Response:**

    Rampant public shaming resulted in PrivDog updating in such a way that certificates were validated prior to it performing a MITM, however no direct action has been taken against Comodo. [30]

- **04/2015:** Chinese government root CA CNNIC issues bad SA

  - **Description:**

The China Internet Network Information Center (CNNIC), a DNS registrar and CA in China with dubious links to the Chinese government, issued an intermediate certificate to MCS Holdings, an Egyptian company without notifying relevant browsers. MCS Holdings then issued a malicious certificate for *.google.com.

– **Community Response:**

Both Google and Mozilla removed trust in the CNNIC's root certificates, but did so with a caveat — all currently known signed certificates will continue to be trusted via a whitelist pushed down by the browser's update process [8, 33]

3.6.1 Discussion

There are a few lessons that can be gleaned from these incidents, their repercussions, and the progress that has been made technologically since 2008. First, it is evident that public key pinning works, and out of bound notification of pinning failures seems to provide browsers with actionable indication that something is dreadfully wrong with a CA. However, since pinning is often not done by many domain owners, it is questionable if attacks against anything other than Google's domains should be found out; this does not help those who do not have the time or capability to implement public key pinning, and it is doubtful that responses to attacks on non-major companies will cause Google or Mozilla to take action.

Another noticeable trend is that a disproportionately large number of these events are from one company, Comodo, and two thirds of those are from their resellers, which do not have an intermediate or unique root that a user can easily distrust.

About a third of these incidents come from government CAs — a trend that has not gone unnoticed. Indeed, the US House of Representatives recently sent a letter to the CEOs of Apple, Google, Microsoft, and Mozilla asking if such CAs could be restricted to their respective Country Code Top Level Domains (ccTLDs) [26].

Intentionally left out of the community response sections in this section are reac-

tions from domain buyers themselves. Many of these failures, such as Comodo's use of PrivDog, and the rent seeking attitude of StartCom post-HeartBleed, are overtly malicious, with CAs knowingly putting users in harm's way for economic gain. Given the press coverage for each of these events, it is not unreasonable to assume that domain owners could easily be made aware of these failings. An analysis of these effects is focus of the next few chapters.

Chapter 4

Analysis of the CA Market

> "I think there's an interesting question here, after all of this, what happened to Comodo? ... Nothing. Their business didn't suffer, they didn't lose customers, they didn't get sued. Really, the only thing that happened was that their CEO was named entrepreneur of the year!"
>
> Moxie Marlinspike [37]

This chapter presents an attempt at measuring the effect of publicly known failures by CAs and the efficacy of potential browser and user reaction to those events. This is done by asking two main questions about the structure of the CA system. First, did an event cause those domain owners who used a malicious CA to leave that CA? Second, can users and browsers effectively remove offending CAs from the CA trust model as a putative response? These questions can be answered by borrowing techniques from economics and through structural and technical analysis of the CA system itself.

There is some evidence that the technical community is in need of guidance for these issues. Less than a year after Comodo's reseller was breached by Comodohacker, a prolific security researcher and self-proclaimed anarchist going by the name Moxie Marlinspike[1] proposed an alternative to the CA system. Called Convergence, this system effectively obviated the need for Certificate Authorities, but at a cost to

[1]His moniker is likely a pseudonym. Indeed, his website is littered with posts about sailing and the sea.

current Domain Owners and browser vendors.[2]

His justification for the replacing the old system can be boiled down to the logic from the quote in this chapter's epigraph — although Comodo has suffered a number of massive and embarrassing violations of their security, often to the determent of users, Comodo's business has in fact prospered [37]. To put it another way, Moxie's implication is that big certificate companies like Comodo represent a cartel or other oligopoly that must be made irrelevant by the development of new cryptographic technologies.

Moxie may not be wrong (as we will see in this chapter there is indeed an unbalanced cartel of oversized CAs), however his argument is flawed enough to be unconvincing to an economist or policy maker. Aside from completely disregarding the potential use of legal or regulatory tools in realigning incentives, Moxie has omitted the fact that the entire market for certificates has drastically increased since Comodo was hacked. Indeed, figure 4-1, a graph taken from the dataset created for this thesis, shows that the entire market for certificates approximately doubled between 2010 and 2011, and has increased over an order of magnitude since then.

It turns out that the explosive growth in the market for certificates is essential in understanding the effect of bad press on CAs. Specifically, it makes comparing a CA's current and past performance effectively worthless — since the entire market has surged, there is little doubt that any functioning CA will see a growth in revenue over time. The question that must be asked is therefore not *are bad CAs doing more business?*, instead, we must ask if a CA would have done more business had they not failed to protect themselves or users.

In addition to market effects, Moxie's argument ignores the potential for users and browsers to take punitive action against misbehaving CAs. As mentioned in Sections 3.4 and 3.1, it is plausible that users and browsers can collectively remove Certificate Authorities from their root stores, effectively neutering a CA's ability to cause harm. The shape of the CA system's market, as well as the technical structure of the PKI hierarchy, is informative in assessing what tools users and browser vendors

[2]Convergence's design and failure to gain popular use are fully discussed in Section 2.5.2.

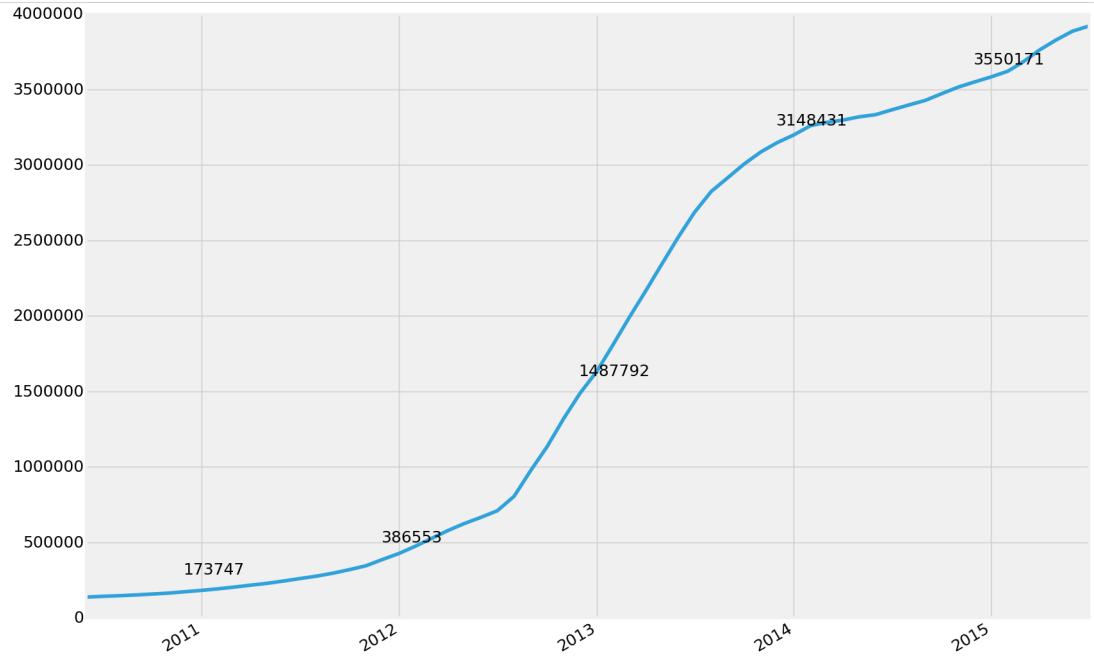


Figure 4-1: The total number of unique, unexpired, cryptographically valid, CA-signed certificates over time since August 2010

can realistically use to punish certificate authorities.

4.1 Methodology and approach

This study's approach can be summarized as a series of steps:

1. Downloading and parsing a market-representitive collection of certificates over time.

2. Cryptographically verifying the certificate hierarchy to provide parent-child relationships.

3. Removal and otherwise abatement of confounding factors such as duplicate certificates, enterprise CAs, and free temporary certificates.

4. Calculating average treatment effect of negative events on Certificate Authorities with respect to domain retention and market growth.

5. Analysis of the certificate authority PKI hierarchy structure to determine what options users and browsers have to restrict CAs.

Each of the above steps are fully eplained in the following sections, with careful attention given to understanding the methodology involved.

4.1.1 Data sources & collection

The first publicly available collection of market data used IPv4 address space scanning as a mechanism. The study was performed by the EFF in August 2010, under the banner of the EFF SSL Observatory [53]. Since then, there have been a number of services that have collected CA certificates. The Perspectives project, as well as Berkeley's ICSI, use passive monitoring of TLS connections to analyze traffic [10, 60]. Recently, the University of Michigan's Zmap project has made available regular scans of the IPv4 address space in their SSL Ecosystem study, and Rapid 7, a security company in Massachusetts, began a similar initiative called Project Sonar, continuing to provide scans since the SSL Ecosystem study finished in early 2014 [21, 55].[3]

The dataset used for this study is a superset of three of the above collections. In order to avoid biasing the sample set, we only used studies that followed the same methodology — repeated IPv4 address space scanning. In other words, this study's dataset is the union of the EFF SSL Observatory, Rapid 7's Project Sonar, and the University of Michigan datasets, providing comprehensive IPv4 address space scans over the past 5 years. Some descriptive statistics about the database:

- 9,538,051 Unique certificates signed by some root
- 59,985,064 Self-signed certificates, including roots
- 83,857,693 Certificates in total, including garbled certificates and certificates signed by non-web roots.[4]

[3]It should be noted that both the University of Michigan and the Rapid 7 teams have graciously put their entire dataset online. They can be found at `https://scans.io/`.

[4]It turns out that many of these are routers or other network appliances with certificates signed by their company, such as Cisco or Juniper Networks.

4.1.2 Cryptographic verification of parent-child relationships

All certificates in the database were cryptographically verified in order to determine which CA's root certificate was ultimately used to sign which leaf certificate, and that the certificate itself was not garbled. The roots in question were taken from Mozilla's Firefox root store.

Mozilla's root store has a number of properties which are beneficial for this exploration. Unlike Chrome or Internet Explorer, the Firefox root store has good provenance — we know exactly when each certificate was added to the root store since the database is kept in certdata.txt, a file in Firefox's source tree that has been backed up in revision control since before 2008. It was therefore trivial to see how the CA list changed over time by continually reverting to older versions of certdata.txt, reading the file, and dumping the included certificates. Luckily, the need for interoperability between browsers incentivizes browsers to include similar roots, so there should be little difference between root stores, and swapping between root stores should not drastically affect any statistical outcome.

The path building exercise followed the IETF's RFC on path validation in reverse. Relationships were first inferred by exploiting the fact that certificate path validation requires child certificate's Issuer field to match the parent's Subject field. This matching set of certificates were then cryptographically verified using OpenSSL to prove that the issuer actually signed the subject, and that the certificate itself was not garbled.

Pseudocode for the algorithm[5] used to construct cryptographically verifiable trees is represented below:

```
1 for certificate in Database.find(self-signed=False):
2     for parent in Database.find(subject=certificate.issuer):
3         if is_cryptographically_signed_by(certificate, parent):
4             certificate.parents += parent
```

A side benefit of this methodology is that it reveals the complexity of the structure of the CA system. For instance, a surprising result of this study is that it is not

[5]Which, admittedly, looks suspiciously like the actual Python code used.

unusual for a CA to have multiple intermediates that share the same public key, or a root that shares its public key with intermediates. This yields a more complex structure than examination of certificate chains alone will reveal, and drives much of the discussion in Section 4.4.

4.1.3 Data cleanup methodology

Apart from cryptographically verifying the paths from a certificate to various roots, there are a number of cleanup operations that must must be completed before any sort of market analysis could be done. First, all known Enterprise Authorities were removed from the analysis pool since it is impossible to reason about the gain per certificate from such entities — it is likely not tied to the number of leaves that intermediate has signed.

Second, we define the canonical market source of a certificate is defined as the first intermediate parent from a leaf owned by a CA. CAs will often cross-sign intermediates of other certificate authorities, likely causing errors in market analyses that rely solely on the chain presented by specific servers. This may sound odd, but a full explanation (and proof of the phenomenon) should be understandable from Section 4.4.

It is important to note that this methodology will result in major discrepancies between this and all known CA market analyses, with good reason. Many surveys naively assume that the root of the chain presented by the server is the market source. Unfortunately, there are many instances in which the certificate chain is misleading due to cross-signing, server misconfiguration, or other PKI anomalies. This study ameliorates this issue entirely by eliding the chains presented as a source of confusion, instead looking toward the owners of intermediates as a canonical market source.

4.1.4 Calculating the average treatment effect of negative events

Determining whether or not an event has had an effect on a company can be done using a venerable economic technique known as a difference in difference measurement.

First popularized in Card and Krueger's 1993 study of the effect of minimum wage on businesses in New Jersey and Pennsylvania [17], this technique is now commonly used in economic studies to analyze the effectiveness of treatments — a change in policy, regulation, or, in this case, reputation — on an effected population.

This study co-opts difference in difference to analyze the effect of lowered reputation due to undermining the trust model of the web (a treatment) on the domain owners who use that particular CA (the population). Instead of measuring an individual CA, growth of competing CA's are measured, holding constant the CA's size at time of treatment, business model, etc., which controls for the overall growth in the CA market. The main difference between the CA's is the treatment effect to be measured — if the CA had somehow received bad press for some sort of bad behavior. We focus on the problems associated specifically with undermining the trust model of the CA system, as presented in Section 3.6.

4.1.5 Analysis of the certificate authority hierarchy structure

Understanding the technical structure of the PKI provides answers to a number of technical and policy questions, particularly informing what options users and browsers have to restrict damage from CAs.

More information is required to determine what powers users actually have to work against misbehaving CAs. Assuming a sufficiently capable user exists to take advantage of technical controls on root certificates, he or she must first be able to weigh the advantages and disadvantages those restrictions. For instance, does a user blocking a CA actually remove enough trust such that that root can no longer perform MITM attacks? Can users effectively triage roots such that only roots with worthy sites still remain in the browser?

Browsers must work through a similar calculus when handling a misbehaving root CA, albeit with a larger number of varying and subtle technical controls. However, there is still the question of what actions browser vendors can take against larger CAs. Do the certificate hierarchies of major CAs allow browser vendors to apply pressure when it is needed?

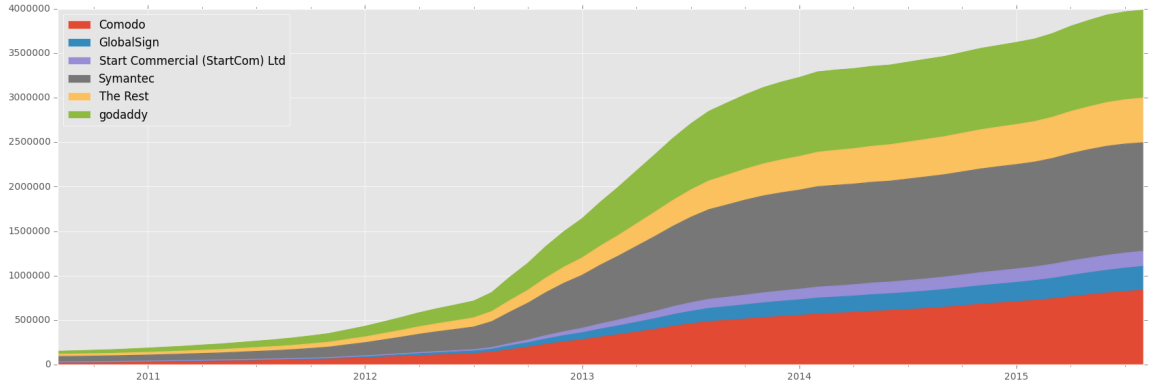


Figure 4-2: Area plot of certificates signed over time by CA organization.

These questions can be answered by exploring the PKI from a qualitative technical perspective. For instance, knowing which CAs have cross-signed for which other CAs, as well as understanding how explicitly distrusting CA maps to impact of web usability, provides insight into how punishable a CA actually is. If a CA has many intermediates, each with a few leaves, it will be easier for a browser or a user to explicitly restrict an individual intermediate's public key rather than the entire root. Sections 4.4 and 4.5 try to answer these questions though empirical analysis of the CA ecosystem.

4.2 Introductory quantitative information about the CA marketplace

This section details a number of important descriptive statistics about the shape of the market, and how the market has changed over time. Much of this information can be found in previous studies (especially [21, 55]), but it was worth reproducing their results in order to re-establish basic facts about the ecosystem itself.[6]

The CA ecosystem is dominated by five companies: Comodo, GlobalSign, StartCom, Symantec, and GoDaddy, with GoDaddy, Symantec, and Comodo being responsible for over half of the market at any one time. Figure 4-2 depicts how the state

[6]Besides, the following graphs are quite aesthetically pleasing.

of the market has changed since 2011 as an area plot, emphasizing the imbalanced nature of the market.

There are a few items of interest that can be gleaned from this image. Most striking is the fact that Comodo, Symantec, and GoDaddy have maintained a sizable lead over the rest of the market, and are responsible for the majority of signed certificates. Further, this lead has been maintained for a number of years, even though the market itself has expanded dramatically.

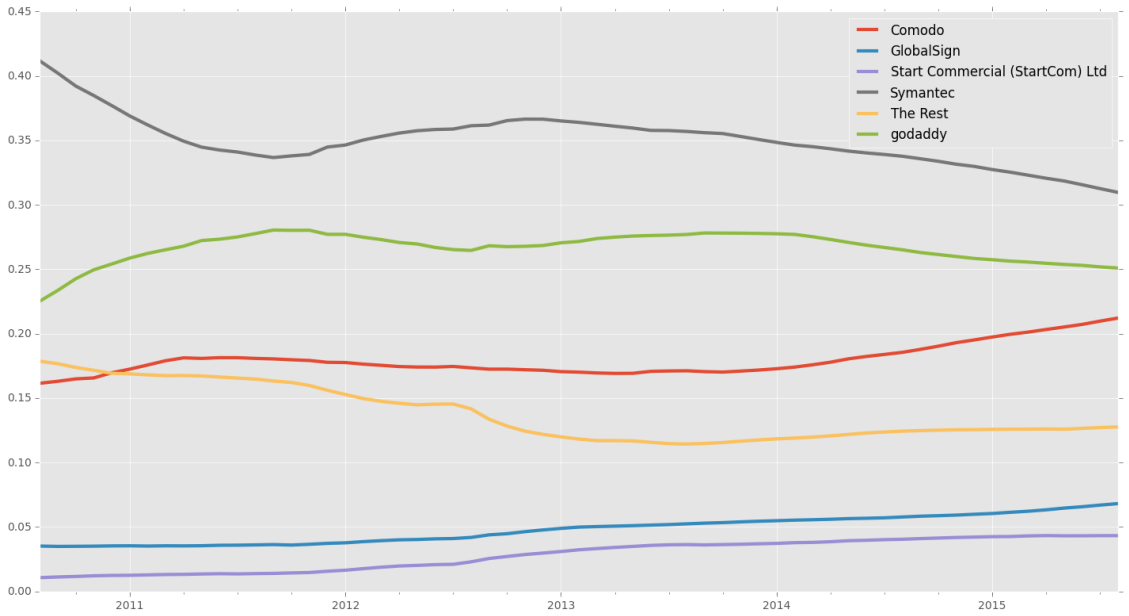


Figure 4-3: Plot of percentage market-share over time

Domain owners appear to purchase certificates on a yearly or biannual basis. Figure 4-4 is a histogram of the number of years every CA-signed leaf certificate remained valid, the vast majority of which are for one year. This tells us that most domain owners have the option of switching between CAs every one or two years without incurring penalty of a sunk cost. This becomes important in later sections when discussing reputation effects — any effect likely to happen will occur within two years of the CA's shaming.

Finally, there seems to be a slight shift in marketshare between the top five CAs over time. Figure 4-3 depicts the percentage of marketshare in number of domains

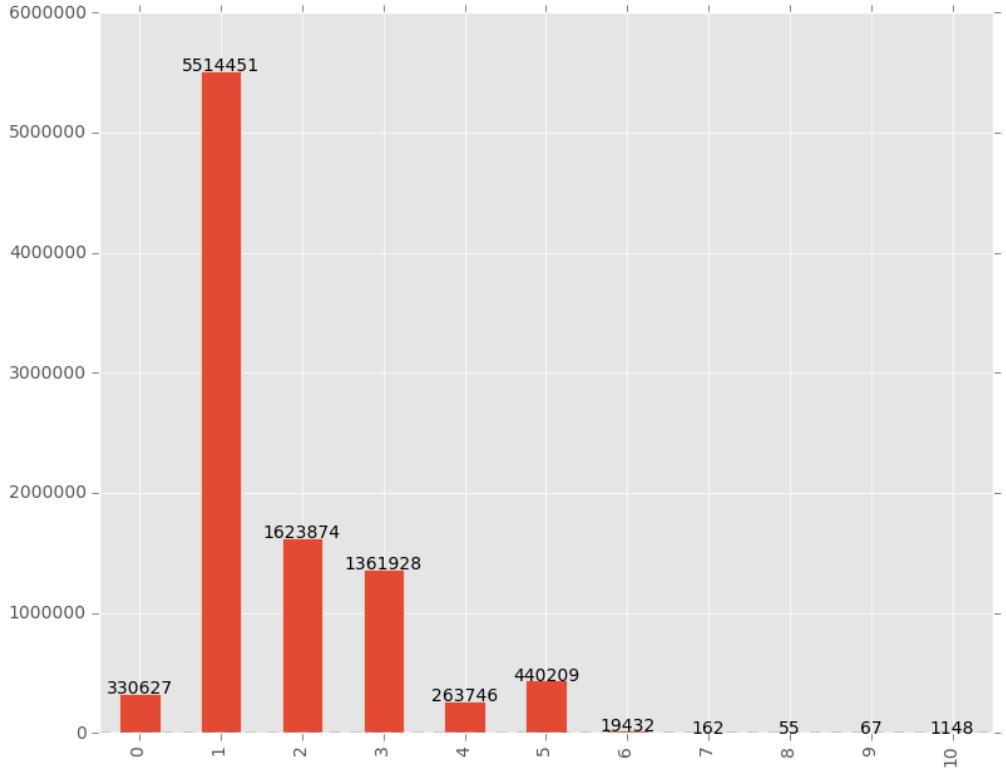


Figure 4-4: A histogram of the number of years every root-signed certificate in the dataset was valid, ignoring revocation.

that each CA signed for over time. Each of the CAs appear to be converging, with Symantec slowly losing market dominance in favor of Comodo and GoDaddy. Although Startcom has made some impressive gains, it appears that the behemoths of the CA market will not be budging from their top seats any time soon.

4.3 Does reputation matter to Domain Owners?

This section presents an attempt at measuring the effect of bad press on misbehaving CAs in two ways. First, we begin by using information from this dataset to determine if an event caused domain owners who used a malicious CA to leave a CA. Second, we provide descriptive statistics that try to answer the related question of whether

or not a negative event correlates with a cessation of growth for that CA.

Domain owners collectively have quite a bit of power in the CA market for one simple reason: They make purchasing decisions. They therefore decide if a CA becomes increases in size, if a CA deserves to be punished for acting out, and, most importantly, how much money a CA makes over time. Should they naturally be swayed by the moral and ethical issues surrounding the CA they work with, it could be an indicator that informal collective action policy tools might be effective in incentivizing CAs to behave properly.

For the purposes of this paper, we define two different types of losses for CAs, *net loss*, or the overall loss of a domain to a CA, and *defection* in which a CA loses a domain to a competitor. There are a number of reasons why certificate authorities might lose a domain that are unrelated to their performance. A business may decide not to provide TLS, or go bankrupt entirely and lose the domain as a whole.

Below we measure comparative CA loss with respect to both overall net loss, and defection loss. CAs are compared with direct competitors by controlling for size, business model, and country of origin.

Below we examine two events from the CA history presented in Section 3.6, focusing on events that have had enough of a time lapse to measure effect, and that were the result of a CA undermining the CA trust model.

There were a number of other events that could not have been used in the study for one reason or another. Many were perpetrated by organizations without profit motive (such as governmental CA breaches), and could therefore not be examined comparatively. Others were from CA's with far too few certificates to be worth studying, including Trustwave and TurkTrust issuing bad intermediates. Finally, events post-2015 have not had enough time lapse for worthwhile analysis — the majority of the affected certificate holders have not been up for renewal.

This leaves us with the two measurable events: Comodo's reseller getting broken into (03/2011), and StartCom refusing to revoke stolen certificates after HeartBleed (04/2014). Both events were the result of rent-seeking by the companies involved, and both have enough data to be worth study.

4.3.1 Comodo (03/2011)

	08/01/2010 Pre-Event	08/01/2012 Post-Event	08/01/2012 Defection	Pre and Post Difference
Comodo (Treated Group)	23,623	10,340	1,162 (4.92%)	13,283
GoDaddy (control)	32,943	17,822	849 (2.57%)	15,121
Difference (Treated vs Control)	-9320	-7482		-1838

Table 4.1: Difference in difference between Comodo and its closest competitor, GoDaddy.

Table 4.1 reports the measurements of Comodo's pre and post treatments, as well as a control group (GoDaddy's) pre and post treatment. There are a few interesting bits of information that can be gleaned from this chart.

First, we note that GoDaddy appears to have a higher net loss than Comodo — a cursory analysis suggests that Comodo actually comparatively benefited from its failing. There are a few extenuating circumstances here though; GoDaddy suffered a reputation blow and boycott in March of 2012 due to their support of the SOPA, a controversial (and, frankly, ill-conceived) anti-piracy bill [11].[7]

More interesting is the comparable defection rate. Even though both had negative reputation effects, Comodo lost more domains than GoDaddy to domains who wished to continue using TLS, and, more importantly, who wished to continue paying for service. The picture is even more interesting when taken as a percentage of the original number of certificates in 2010: Comodo lost roughly 5% of their domains in 2010 to defections, whereas GoDaddy lost only 2.6%.

Indeed, this loss is backed up by the information in Figure 4-3; Comodo appears to begin losing comparative marketshare from late 2011 until 2014. However, the graph is a bit misleading, as in raw numbers the entire known market grew from 146,241 unique domains in 08/2010 to 804,486 in 08/2012, a growth of about 550%.

[7]It is worth mentioning that the boycott mainly targeted their domain name registration service, not their CA service, and ended quickly after GoDaddy reversed their stance.

Conversely Comodo grew from 23,623 domains to 139,575, or roughly 590%.

However, Comodo's gains may have been due to the falling marketshare of the rest. The data presented in Table 4.2 demonstrates that, compared to its closest competitor GoDaddy, Comodo managed to lose a significant amount of marketshare.

This result is actually good news for the health of the industry. These numbers imply that Comodo's break-in caused their business to lose nontrivial amount of money even compared to GoDaddy, who did not undermine the CA trust model. This is even more surprising given the fact that GoDaddy suffered a non-CA related PR disaster during the same years.

	08/01/2010 Pre-Event	08/01/2012 Post-Event	Pre and Post Difference
Comodo (Treated Group)	23,623	139,575	-115,952
GoDaddy (control)	32,943	212,874	-179,931
Difference (Treated vs Control)	-9320	-73299	63,979

Table 4.2: Difference in difference between Comodo and its closest competitor, GoDaddy.

4.3.2 Startcom and HeartBleed (04/2014)

It appears that the case of HeartBleed actually had some effect on Startcom's ability to sell certificates. Normalized to the size of each certificate authority's leaf-base, defection rates between the Startcom and Globalsign appear comparable, with Startcom having had roughly 1% more defections than Globalsign. The difference in difference indicates as well that there is an effect of net loss as well, as Startcom lost significantly more than Globalsign over the same period.

Keeping in mind that certificates are perfect substitutes, this result is made even more interesting by the fact Startcom certificates cost much more than Globalsign's. Startcom provides domain-validated certificates for zero issuance cost, while Global-

	04/01/2014 Pre-Event	04/01/2015 Post-Event	04/01/2015 Defection	Pre and Post Difference
Startcom (Treated Group)	127,072	55,121	5,167 (4.1%)	71,951
Globalsign (Control Group)	184,510	129,384	6,095 (3.3%)	55,126
Difference (Treatment - Control)	-57,438	-74,263		16,825

Table 4.3: Difference in difference as well as defection between StartCom and Globalsign.

	04/01/2014 Pre-Event	04/01/2015 Post-Event	Pre and Post Difference
Startcom (Treated Group)	127,072	163,012	-35,940
Globalsign (Control)	184,510	238,157	-53,647
Difference (Treatment - Control)	-57,438	-75,145	17,707

Table 4.4: Difference in difference of the growth rate between Startcom and Globalsign.

sign currently charges $240 per year for the same certificate.[8]

The difference in difference measurement between Startcom and Globalsign's growth rate further confirms that Startcom's malfeasance cost them members. Startcom appears to have gained roughly 75 thousand fewer domains than Globalsign over the same period of time, and the difference in difference indicates that, holding their pre-sizes constant, Startcom had been affected by their unwillingness to help those affected by HeartBleed. This evidence corroborates the result found in the previous section on Comodo: it does appear that there is some sort of effect for CAs behaving badly.

[8]See `https://www.globalsign.com/en/ssl/buy/?type=dv`

4.4 Users are structurally barred from making trust decisions

This section expands upon the discussion on users in Section 3.4, in which it was noted that many users are unable to to make informed, rational choices about which CAs to distrust. The reality of the Mesh PKI, however, makes this discussion much more interesting; it is unlikely that a user can rationally disable trust in a CA, once that CA has become part of the global PKI, due to the evolving landscape of the CA system and subtleties involved in the system's architecture.

As described in the section on users, as well as in the description of the system in general (Section 3.4 and Chapter 2, respectively), users only have access to roots certificates a-priori. The result is that a user can explicitly distrust a root, should that root's CA prove to be untrustworthy, but intermediates and the number of leaves each intermediate owns are often not known. The result is that the CA system has become so interdependent that it is functionally impossible for a user, however knowledgeable, to distrust a specific certificate authority.

This is partially due to a combination of phenomena that confuses the structure of the PKI — public key sharing and intermediate cross-signing. Cross signing works by having one root sign a cross-signing certificate. This cross-signing certificate has as its issuer the cross-signer, as its subject the signee's subject, and contains signee's public key. The result is a bridge between the two.

The result is that public keys and subject names are shared between intermediates and root certificates, creating paths that do not include the original root. It is the confluence of two odd effects — public key sharing and non-root cross signing — that has made excluding a CA unintuitive and unlikely for a subset of the CA system.

For the sake of argument, let us say that some user wanted to remove trust in any CA whose business operates mainly in China. Indeed, there is a lower probability that for users in the US would need to trust a Chinese company; a Chinese CA might be more likely to sign for Chinese domains and users, lessening the likelihood that a US user would be negatively affected by removing that CA. To pick an example,

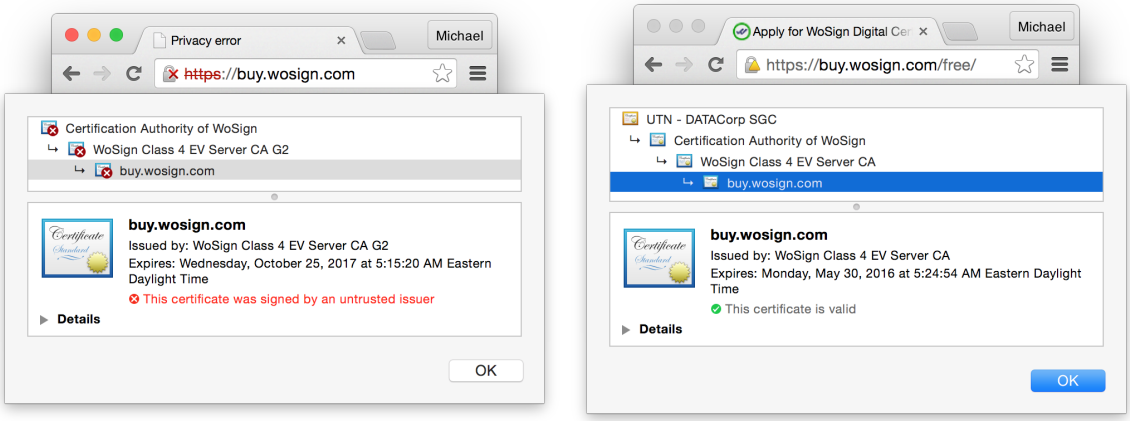


Figure 4-5: The above is an example of cross-signing bypassing user exclusion. The left image shows that the WoSign root was untrusted, while the right image shows a still-trusted chain that websites could use that includes, due to cross-signing, the WoSign root public key.

WoSign is a Chinese CA which, at the time of writing, has signed for 20,000 unique domains in the last 4 years, the majority of which are from the .cn TLD space, and almost none of which are in the Alexa top million. Further, given the geopolitical nature of US relations with China, removing them from the root store makes sense, as it has the potential to decrease a business's attack surface.

A naive option would be to delete a CA's certificates from the root store, which would allow for cross-signed chains to continue to function. Indeed, any path through the root to another root will still be valid. Instead, users must change settings in their root store management software to explicitly distrust the root so that any chain including its public key is considered invalid. In other words, there are actually two settings — forgetting a root exists and explicitly distrusting the root.

We again return to WoSign as an example. As a proof of concept, a chain was manually crafted from a WoSign leaf to another root certificate owned by Comodo, which had been used to cross-sign for WoSign (the node in Figure 4-6 with a Subject Name of UTN-Datacorp).

The result is in Figure 4-5 — even though WoSign's root certificates have been removed from the OS's root store, and the root in one image shares the same public key as the cross-signed intermediate in the other, the WoSign signed leaf is still trusted.

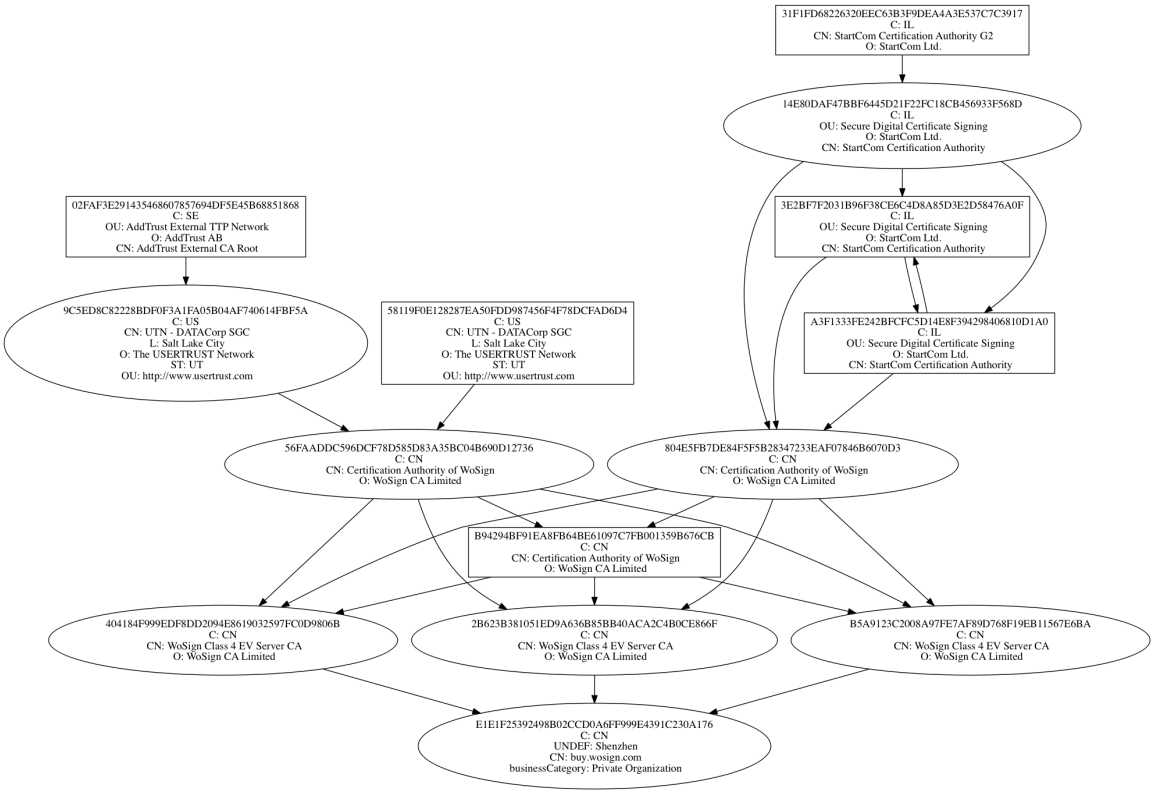


Figure 4-6: The above image is a directed graph from the "buy.wosign.com" leaf certificate to a number root certificates (Comodo, StartCom, and WoSign respectively) included in the Mozilla root store. Roots are denoted by squares, intermediates and leaves are denoted by circles, and arrows indicate cryptographic signing relationships. Note that multiple certificates signed the same intermediates, as well as the leaf — this is due to *public key sharing* between these certificates. Any chain leading from the leaf to a root in this graph would be considered browser-valid.

The result of removing a root is oddly the worst of both worlds: many WoSign-signed websites will fail validation, but WoSign and its intermediates still have the power to issue fraudulent certificates and, therefore, execute MITM attacks.

Figure 4-6 is a full aggregate tree-level depiction of the phenomena found in Figure 4-5. Any path from a root certificate to the leaf certificate on the bottom will result in a valid chain. So, due to the heavy amount of public key sharing between the intermediates and the root, removing WoSign's root CAs does not restrict WoSign from signing browser-valid certificates. Worse still, removing additional roots leads to further impact to the user — removing Comodo's and Startcom's roots would result in the user being unable to access an unacceptable number of websites.

Although WoSign is being discussed in particular here, it is only meant to act as exemplar organization for discussion. Indeed, of the 170 roots currently trusted by Mozilla's root store, 52 have known cross-signers. It is important to note that this is a conservative number; there may be many more cross-signed certificates that are simply not used in common deployment, or have not been disclosed.

Such subtleties, which would likely be lost on a large number of users including corporate IT departments, are compounded by the dynamic state of CAs. New CAs are added to the trust store yearly, and are pushed via OS updates from OS and browser vendors automatically in security patches. Hundreds of thousands of leaves are added yearly as well, meaning that any removal of trust in a CA could negatively impact user productivity if not monitored or updated continuously. The result is that even the most security concious users are unlikely to modify the trusted list of certificates installed by default, instead shirking the responsibility to OS and browser vendors.

4.5 Browsers face difficulty in removing or punishing larger CA roots

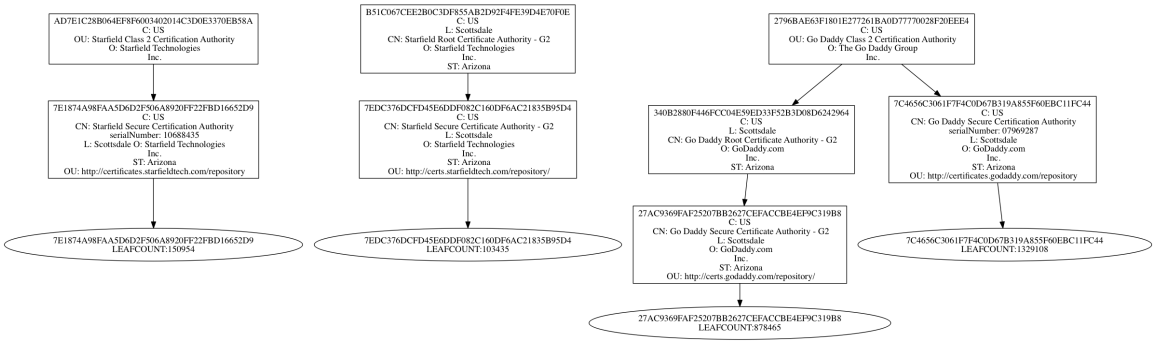


Figure 4-7: All GoDaddy-owned roots and intermediates that have signed certificates in the dataset.

The calculus that users must do in order to remove a CA from the root store is similar to the work that a browser must complete for doing the same thing, albeit

with more subtle and varied tools. A particular difference between browsers and users is that the browser has the ability to block by public key — indeed, Chrome's CRLset revocation mechanism does exactly that.[9]

Throughout the history presented in Section 3.6, there have been a number of times in which Google has manually restricted or punished poorly behaving CAs. In 2013 and 2014, Google manually restricted two different governmental CAs (India and France, respectively) by only allowing them to issue certificates for specific Top Level Domains (TLD) associated with their countries. Further, in 2015, Google and Mozilla removed the Chinese Government's root, the CNNIC, entirely for a similar transgression.

It is worth dissecting these events to understand the limits of such browser actions. In particular, analysis of the draconian distrust of CNNIC's roots provide much insight. For instance, at the time the CNNIC was removed, the CNNIC had roughly 1500 known unexpired leaf certificates in the wild, a relatively minuscule number. Of those, only a fraction actually still responded to HTTPS connection attempts today. This lack of impact factor allowed Google and Mozilla to push an update which whitelisted exactly which certificates were allowed to be verified, effectively ossifying the number of leaves the CNNIC could sign to those already known to Google and Mozilla.[10]

Although browsers can effectively punish CAs like the CNNIC for bad behavior, it remains unclear what, if anything, they can do to larger CAs. GoDaddy, for example, has a far more diverse TLD set, has a very small hierarchical structure for its intermediates, and has a ton of leaves. The impact of restricting GoDaddy in any way would be catastrophic, causing users to be unable to access a number of commonly used websites. Finally, manually pushing a whitelist as was done in the CNNIC's case would be impractical; GoDaddy easily has over 500,000 valid leaf certificates, not all of which could be known by the browser vendors a-priori.

[9]For more technical information about how CRLSets work, the CRLSet tools are a fantastic place to start: `https://github.com/agl/crlset-tools`

[10]Indeed, Mozilla's tactic was to add into its validation procedure a checklist of known-good hashes for CNNIC leaf certificates. These hashes can be found in Firefox's source tree by the name of "CNNICHashWhitelist.inc."

Chapter 5

Discussion and Conclusions

> "Mathematics is perfect; reality is subjective. Mathematics is defined; computers are ornery. Mathematics is logical; people are erratic, capricious, and barely comprehensible."
>
> Bruce Schneier [56]

This thesis set out to understand how well user, domain owner, and browser vendor controls functioned in order to draw conclusions about the best way to realign CA incentives in favor of user security and privacy. What was discovered was much less than ideal — CAs often blatantly reject their duties in favor of economic gain, browser vendors are seemingly unable to coerce large CAs, and users remain both oblivious of and unable to effectively control who they are trusting.

The final chapter ends on a hopeful note by discussing the results obtained in Chapter 4 in light of the structural information presented in Chapters 2 and 3 to draw conclusions about what policies and technologies could be useful in bettering the CA system, and to provide an evaluation of impending improvements.

5.1 Reputation effects are not currently impactful

Although the results in Section 4.3 did indicate that bad behavior has caused an increase in defection and decrease in growth rate in as compared to a similar CA, it is at best unlikely that such effects will cause a shift in CA behavior. In the case of

Comodo, the increased defection rate of a malicious CA was at most approximately 3% more than a comparable CA, while the growth rate actually slightly *increased*. It is therefore very plausible that CAs could be coming down on the malicious side of the risk calculus. To put it another way, this study proves that there exists a slight negative incentive for behaving poorly, but CAs likely still benefit enough from misbehaving that the increased attrition is negligible.

Indeed, the repeated and continuing malfeasance by CAs supports the hypothesis that the CA incentive structure is still broken. Comodo, for instance, recently put users in harms way yet again by releasing a malicious application called PrivDog that completely undermined the CA trust model.[1] Similarly, despite much public shaming and pressure from the community, StartCom continues to require domain owners to pay fees on revocation, and never recapitulated on the fees it collected after Heartbleed.

Worse, it is unclear exactly how much damage is required to act as disincentive; it is incredibly hard to quantify how much revenue a business gains from acting as a CA, and therefore equally difficult to discern the impact on a CA's risk calculation. Corporations that operate CAs often have side-businesses as other sources of revenue such as DNS registrars or hosting companies, so public statements of earnings are not particularly helpful. Resellers and variable rate certificates cause further issues — differences between otherwise indistinguishable certificates (caused by variable reseller rates, bulk purchases by domain owners, etc.) make measuring the monetary impact of losing domain owners quite difficult.

In any case, damaging a CA's reputation does appear to impact at least some set of a CA's user-base enough for a noticeable number of domain owners to defect at a higher rate, which suggests that attempting to further this impact is a worthwhile goal. There are numerous ways in which such a thing might be done, incorporating both policy and technological approaches. Some potential examples include increasing the visibility of past failures by CAs, promoting awareness and general understanding of the CA ecosystem to domain owners, and by empowering users to make trust

[1] A full description of PrivDog is included in 3.6.

decisions.

5.2 Increasing domain owner awareness

If bad press can have some impact, then more bad press could be similarly useful, in part because it further educates domain owners and users. The CA system is incredibly complex, arcane, and poorly understood, which likely leads to further to misunderstandings among domain owners about what they are purchasing and what a CA's actual role is in providing security.[2] Standards bodies and privacy watchdog groups such as the Electronic Frontier Foundation (EFF) have the potential to punish corporate CAs by providing prospective buyers with enough updated information to compare CAs.

The EFF in particular has been very good about providing public feedback to industry about their performance on security and privacy metrics. For instance, their annual "Who has your back?" survey provides a pithy explanation of which business behaves well with respect to various privacy factors, including opposition to backdoors, data retention, disclosure of data demands, etc.[3]

Organizations like the EFF could do something similar for CAs to great effect. Providing publicly accessible information about how badly behaving CAs undermine their privacy and security could lead to drastic improvements in CA attitudes and procedures, as the incidents of specific users removing CAs as a reaction to faults could similarly increase.

[2]This misunderstanding can be seen in the oddities surrounding CA marketing, in which CAs differentiate themselves using ancillary information such as the time it takes to crack the certificate (which is definitionally the same as every other certificate that conforms to modern cryptographic standards), number of free re-issuances, and other customer-service oriented information.

[3]An exemplar survey can be found here `https://www.eff.org/who-has-your-back-government-data-requests-2015`.

5.2.1 Certificate Transparency could lead to major improvement

One of the more hopeful results of this thesis is that it is likely that Certificate Transparency (CT), if implemented, will have a large impact on the CA market. This was not a forgone conclusion — Without any reputation effect, there is a chance that the added transparency of the Certificate Transparency log could be functionally worthless. Indeed, if reputation had no impact on a CA's business, no CA would have reason to change its behavior since a history of mis-issued certificates would have little effect on the company's ability to continue to market their brand.

It is important to remember that CT, as discussed in section 2.4.2, is a purely reactive tool. CT only forces CAs to disclose that they have mis-issued a certificate, which can then be remedied later by a domain owner who discovers the mis-issuance by monitoring the logs for his own domain. None of this stops the mis-issuance from occurring in the first place, and an unashamed CA will continue to behave poorly in favor of other economic benefits. Worse, some number of domain owners will likely fail to remediate mis-issued certificates.

However, since reputation seems to be at least weakly tied to how well a CA maintains the trust model, CT could be a very important tool. CT's log makes CA failure data open — researchers can catalog every time that a CA has mis-issued a certificate and generate reports to guide domain owners in purchasing decisions. In other words, a domain owner would be given information about when, how often, and how brazenly specific CAs failed to maintain security, allowing domain owners to compare CA track records and purchase from those who behave well, thus better aligning CA incentives.

Enthusiasm for CT should be couched by the fact that adoption of CT is still in very early stages, and that CAs have a fairly obvious disincentive to adopt CT. CT is currently only required by one browser, and only for a small subset of certificates (specifically, Extended Validation (EV) certificates). There is no requirement for the widely-used Domain Validated (DV) certificates to be entered into a CT log, and

doing so would require a very robust set of CT auditors to function.

5.3 The CA PKI maps poorly to human trust relationships

Human trust is very contextual, and trust in a PKI should be as well.[4] For instance, a person may place varying levels of trust in an individual depending on where they got their degree,[5] or simply not care depending on the task or context of the licencing involved. For instance, a doctor might not be trusted by an individual to perform open heart surgery if few had ever heard of the medical school the doctor attended.

Similarly, different CAs can be trusted for different things. One might, for instance, trust a government CA to sign for government services websites, but become suspicious when they happen to have signed for an email provider.

None of this is currently possible in the CA system. Certificate hierarchies and domain names lack semantic constructs for browsers: with the exception of Extended Validation certificates, a certificate and its trust chain that is used to verify a bank may be nearly identical to one used for social media sites. Browsers (and consequently users) therefore cannot automatically make value judgements about the certificate chain proffered based on context.

5.3.1 Users must be able to distrust organization's certificates without penalty

Human trust requires the ability to contextually change their trust boundaries; different users should have the ability to distrust a unique set of CAs given their personal preferences. For example, US Government employees may wish to distrust Chinese CAs, and Chinese users may wish to distrust the US Government's intermediates for

[4]That is, of course, unless something like Convergence that is designed to be trust-oblivious is functionally deployed (described in Section 2.5.2).

[5]And potentially none, e.g. Harvard.

equally valid reasons. Conversely, users in deeply oppressive countries may wish to distrust CAs in their own countries for similar reasons.

The unfortunate discovery in Section 4.4 effectively makes this sort of user distrust impossible. There is an outside chance that a user could successfully blacklist an undeserving subset of CAs, but the process of doing so is nontrivial for an expert and effectively impossible for an average user.

An ideal case would be for users to be able to distrust misbehaving CAs, for that action to somehow cause those CAs direct financial harm without causing users discomfort. More research is needed in this niche to design a system with these properties, and for methods of moving the global PKI to these systems over the current entrenched interests.

What is known for sure is that the shocking lack of user control in this system must change. Currently, users are left without the tools necessary to distrust specific CAs, have little nontechnical information about the system, and have even fewer resources about who they are trusting and why. This must be fixed — in order for a market for trust to exist, users must be empowered to make their own decisions, and neither browsers nor CAs appear to be effectively aligned to act as protectors of user interest in all cases.

5.4 Conclusion

As the epigraph of this chapter implies, much of the problems involved in the CA system and the HTTPS protocol exist between the interaction of users and various institutions, incorporating all of the inherent issues of human involvement; A CA's competing interests, both geopolitically and economically, muddy its ability to function as a trust anchor, failures of proposed alternative systems that lack economic incentives have been shown to be implausible, users are unequipped to truly make trust decisions, browser control is coarse at best, and domain owners do not appear to have much sway.

Although much of this paper points to worrying failures in the CA market, there

are some hopeful signs for the future — new technical mechanisms and open market data lend themselves to the (likely unattainable) goal of a perfect trust model. More light on these issues will breed more discussion, which will hopefully lead to better CA models or better regulation.

It is the hope of the author that this thesis provide a starting point, an initial source of clarity to those deciding what to do next regarding regulation and creation of tools for the betterment of the CA market. At the very least, all evidence indicates that whatever system or regulatory technique is proposed must be both technically and economically aligned, or it will likely fail.

Appendix A

Figures

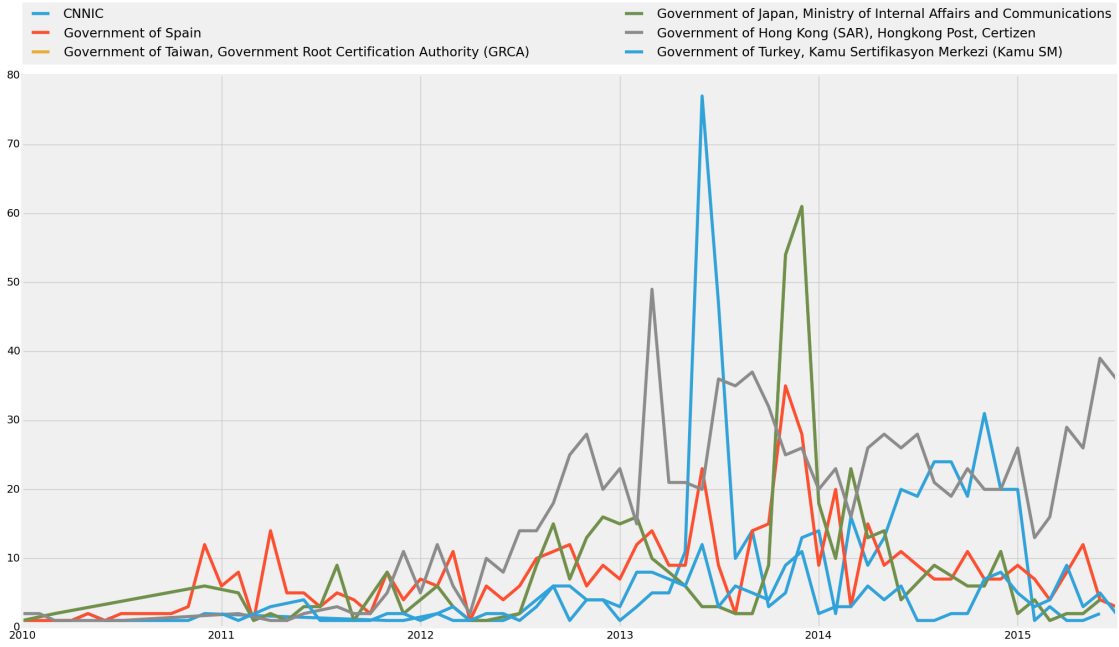


Figure A-1: Number of certificates issued by government-owned CAs over time

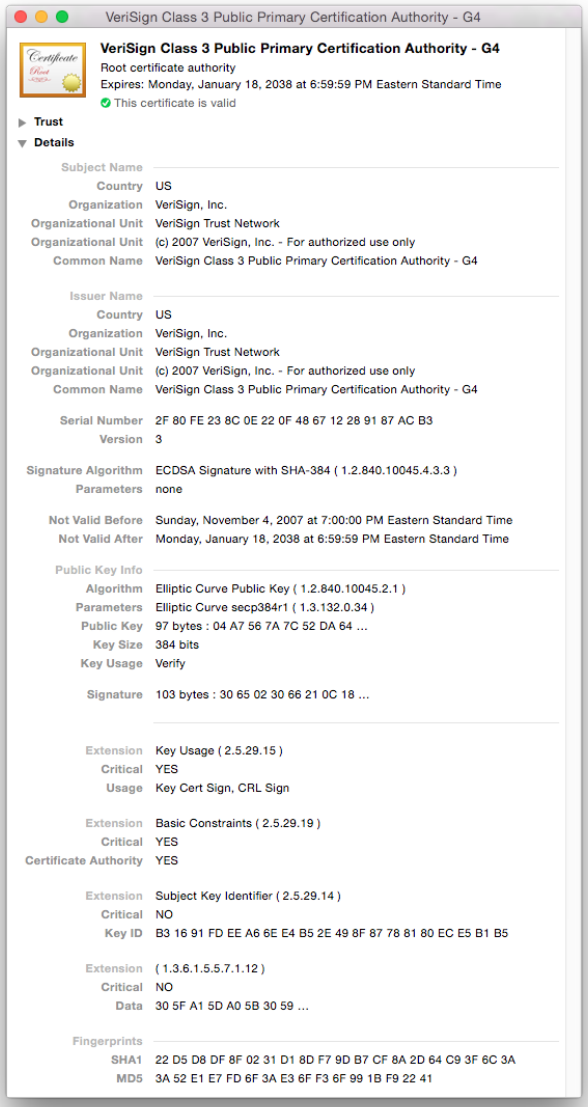


Figure A-2: A VeriSign root certificate as displayed by Mac OSX's Keychain utility

Bibliography

[1] Adam Langley. ImperialViolet - Certificate Transparency. https://www.imperialviolet.org/2011/11/29/certtransparency.html, November 2011.

[2] Adam Langley. ImperialViolet - Revocation checking and Chrome's CRL. https://www.imperialviolet.org/2012/02/05/crlsets.html, February 2012.

[3] Adam Langley. Enhancing digital certificate security. http://googleonlinesecurity.blogspot.com/2013/01/enhancing-digital-certificate-security.html, January 2013.

[4] Adam Langley. Further improving digital certificate security. http://googleonlinesecurity.blogspot.com/2013/12/further-improving-digital-certificate.html, December 2013.

[5] Adam Langley. ImperialViolet - Revocation still doesn't work. https://www.imperialviolet.org/2014/04/29/revocationagain.html, April 2014.

[6] Adam Langley. Maintaining digital certificate security. http://googleonlinesecurity.blogspot.com/2014/07/maintaining-digital-certificate-security.html, July 2014.

[7] Adam Langley. ImperialViolet - Why not DANE in browsers. https://www.imperialviolet.org/2015/01/17/notdane.html, January 2015.

[8] Adam Langley. Maintaining digital certificate security. http://googleonlinesecurity.blogspot.com/2015/03/maintaining-digital-certificate-security.html, March 2015.

[9] Devdatta Akhawe and Adrienne Porter Felt. Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness. In *Usenix Security*, pages 257–272, 2013.

[10] Bernhard Amann, Matthias Vallentin, Seth Hall, and Robin Sommer. Revisiting SSL: A large-scale study of the internet's most trusted protocol. Technical report, Technical report, ICSI, 2012.

[11] BBC News. GoDaddy faces boycott over SOPA anti-piracy law support. http://www.bbc.com/news/technology-16320149, March 2012.

[12] Ben Laurie. Extended Validation in Chrome - Certificate Transparency. http://www.certificate-transparency.org/ev-ct-plan, March 2014.

[13] Mark Bergen. Verizon Looks to Target Its Mobile Subscribers With Ads | Digital - Advertising Age. http://adage.com/article/digital/verizon-target-mobile-subscribers-ads/293356/, May 2014.

[14] Benjamin Beurdouche, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, Alfredo Pironti, Pierre-Yves Strub, and Jean Karim Zinzindohoue. A messy state of the union: Taming the composite state machines of TLS. In *IEEE Symposium on Security and Privacy*, 2015.

[15] Bill Marczak, Jakub Dalek, John Scott-Railton, Nicholas Weaver, Roya Ensafi, David Fifield, Sarah McKune, Arn Rey, Ronald Deibert, and Vern Paxson. China's Great Cannon. https://citizenlab.org/2015/04/chinas-great-cannon/, April 2015.

[16] Peter Bright. Independent Iranian Hacker Claims Responsibility for Comodo Hack. http://www.wired.com/2011/03/comodo_hack/, March 2011.

[17] David Card and Alan B. Krueger. Minimum wages and employment: A case study of the fast food industry in New Jersey and Pennsylvania. Technical report, National Bureau of Economic Research, 1993.

[18] Comodo. Comodo Report of Incident - Comodo detected and thwarted an intrusion on 26-MAR-2011. https://www.comodo.com/Comodo-Fraud-Incident-2011-03-23.html, March 2011.

[19] Dave Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. http://tools.ietf.org/html/rfc5280#section-3.2, May 2008.

[20] Tim Dierks and Eric Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. https://tools.ietf.org/html/rfc5246, August 2008.

[21] Zakir Durumeric, James Kasten, Michael Bailey, and J. Alex Halderman. Analysis of the HTTPS certificate ecosystem. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 291–304. ACM, 2013.

[22] Eddy Nigg. Join The Revolution! » Glitch or Negligence? https://blog.startcom.org/?p=152, January 2009.

[23] Eric Butler. Firesheep. http://codebutler.com/firesheep/, October 2010.

[24] Stephen Farrell and Hannes Tschofenig. Pervasive Monitoring Is an Attack. https://tools.ietf.org/html/rfc7258, May 2014.

[25] Francisco Amato and Federico Kirschbaum. evilgrade, "You STILL have pending upgrades!". https://www.defcon.org/images/defcon-18/dc-18-presentations/Amato-Kirschabum/DEFCON-18-Amato-Kirschabum-Evilgrade.pdf, July 2010.

[26] Frank Pallone, Diana DeGette, Anna G. Eshoo, and Jan Schakowsky. Letters to Browsers Regarding Government Certificate Authorities | Energy & Commerce Committee. http://energycommerce.house.gov/letter/letters-browsers-regarding-government-certificate-authorities, June 2015.

[27] Sean Gallagher. New JavaScript hacking tool can intercept PayPal, other secure sessions. http://arstechnica.com/business/news/2011/09/new-javascript-hacking-tool-can-intercept-paypal-other-secure-sessions.ars, September 2011.

[28] Hans Hoogstraaten. Black Tulip: Report of the investigation into the DigiNotar Certificate Authority breach. https://www.rijksoverheid.nl/binaries/rijksoverheid/documenten/rapporten/2012/08/13/black-tulip-update/black-tulip-update.pdf, August 2012.

[29] Cormac Herley. So long, and no thanks for the externalities: the rational rejection of security advice by users. In *Proceedings of the 2009 workshop on New security paradigms workshop*, pages 133–144. ACM, 2009.

[30] Jane Wakefield. Ad-blocking software is 'worse than Superfish'. http://www.bbc.com/news/technology-31586610, February 2015.

[31] Joe Grand. Hackers Testifying at the United States Senate, May 19, 1998 (L0pht Heavy Industries). https://www.youtube.com/watch?v=VVJldn_MmMY, May 1998.

[32] Kathleen Wilson. Revoking Trust in one ANSSI Certificate. https://blog.mozilla.org/security/2013/12/09/revoking-trust-in-one-anssi-certificate/, December 2013.

[33] Kathleen Wilson. Distrusting New CNNIC Certificates. https://blog.mozilla.org/security/2015/04/02/distrusting-new-cnnic-certificates/, April 2015.

[34] Adam Langley. ImperialViolet - Why not Convergence? https://www.imperialviolet.org/2011/09/07/convergence.html, September 2011.

[35] Let's Encrypt. About. https://letsencrypt.org/about/, 2015.

[36] Wilson Lian, Eric Rescorla, Hovav Shacham, and Stefan Savage. Measuring the Practical Impact of DNSSEC Deployment. In *USENIX Security*, pages 573–588, 2013.

[37] Moxie Marlinspike. BlackHat USA 2011: SSL And The Future Of Authenticity. https://www.youtube.com/watch?v=Z7Wl2FW2TcA&feature=youtube_gdata_player, August 2011.

[38] Matthew Prince. The Hidden Costs of Heartbleed. http://blog.cloudflare.com/the-hard-costs-of-heartbleed/, April 2014.

[39] Robert McMillan. Verizon's 'Perma-Cookie' Is a Privacy-Killing Machine. http://www.wired.com/2014/10/verizons-perma-cookie/, October 2014.

[40] Eran Messeri, Adam Langley, Emilia Kasper, Ben Laurie, and Rob Stradling. Certificate Transparency. https://tools.ietf.org/html/draft-ietf-trans-rfc6962-bis-08, July 2015.

[41] Michael Coates. Revoking Trust in Two TurkTrust Certificates. https://blog.mozilla.org/security/2013/01/03/revoking-trust-in-two-turktrust-certficates/, January 2013.

[42] Mike Masnick. Shameful Security: StartCom Charges People To Revoke SSL Certs Vulnerable To Heartbleed | Techdirt. https://www.techdirt.com/articles/20140409/11442426859/shameful-security-startcom-charges-people-to-revoke-ssl-certs-vulnerable-to-heartbleed.shtml, April 2014.

[43] Mozilla. 470897 – Investigate incident with CA that allegedly issued bogus cert for www.mozilla.com. https://bugzilla.mozilla.org/show_bug.cgi?id=470897, December 2008.

[44] Mozilla. 682927 – Dis-trust DigiNotar root certificate. https://bugzilla.mozilla.org/show_bug.cgi?id=682927, August 2011.

[45] Mozilla. 724929 – Remove Trustwave Certificate(s) from trusted root certificates. https://bugzilla.mozilla.org/show_bug.cgi?id=724929, February 2012.

[46] Mozilla. 825022 – Deal with TURKTRUST mis-issued *.google.com certificate. https://bugzilla.mozilla.org/show_bug.cgi?id=825022, December 2012.

[47] Mozilla. 994478 – Remove StartCom CA from Mozilla for violation of CA policy. https://bugzilla.mozilla.org/show_bug.cgi?id=994478, April 2014.

[48] Mozilla. Networking/http2 - MozillaWiki. https://wiki.mozilla.org/Networking/http2, August 2014.

[49] Mozilla. CA:How to apply - MozillaWiki. https://wiki.mozilla.org/CA:How_to_apply, July 2015.

[50] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. *This POODLE Bites: Exploiting The SSL 3.0 Fallback*. 2014.

[51] NetMarketshare. Browser market share. https://www.netmarketshare.com/browser-market-share.aspx?qprid=0&qpcustomd=0, September 2015.

[52] OpenSSL. TLS heartbeat read overrun (CVE-2014-0160). https://www.openssl.org/news/secadv/20140407.txt, April 2014.

[53] Peter Eckersley and Jesse Burns. The EFF SSL Observatory. https://www.eff.org/observatory, July 2010.

[54] Richard Barnes. Deprecating Non-Secure HTTP. https://blog.mozilla.org/security/2015/04/30/deprecating-non-secure-http/, April 2015.

[55] Mark Schloesser. Project Sonar: Scanning All The Things | Rapid7 Community. https://community.rapid7.com/community/infosec/sonar/blog/2013/09/26/internet-wide-probing–rapid7-sonar, September 2013.

[56] Bruce Schneier. *Secrets and lies: digital security in a networked world*. John Wiley & Sons, 2011.

[57] Ryan Sleevi, Chris Evans, and Chris Palmer. Public Key Pinning Extension for HTTP. https://tools.ietf.org/html/rfc7469, April 2015.

[58] Alexander Sotirov, Marc Stevens, Jacob Appelbaum, Arjen K. Lenstra, David Molnar, Dag Arne Osvik, and Benne de Weger. MD5 considered harmful today, creating a rogue CA certificate. In *25th Annual Chaos Communication Congress*, 2008.

[59] SpiderLabs Anterior. Clarifying The Trustwave CA Policy Update - SpiderLabs Anterior. http://blog.spiderlabs.com/2012/02/clarifying-the-trustwave-ca-policy-update.html, February 2012.

[60] Dan Wendlandt, David G. Andersen, and Adrian Perrig. Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing. In *USENIX Annual Technical Conference*, pages 321–334, 2008.

[61] Alma Whitten and J. Doug Tygar. Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In *Usenix Security*, volume 1999, 1999.

[62] Zineb Ait Bahajji and Gary Illyes. HTTPS as a ranking signal. http://googlewebmastercentral.blogspot.com/2014/08/https-as-ranking-signal.html, August 2014.