
CHAPTER 6

The Key Certification Problem: Rethinking PKI

This chapter is an extended look at the prior art, practice, and problems of the Public Key Infrastructure (PKI) approach to public key certification.

6.1 A Tale of Two Protocols

Skeeter and Bubba, FTP Software, 1991

Levy, Kastenholz and Knowles realized that they could improve the security of TCP by putting a Diffie-Hellman key agreement step directly into TCP's three-way handshake. The exchange was implemented with TCP options #16 ("Skeeter") and #17 ("Bubba"). [Kas01] If a TCP implementation supporting these options made a connection to a second TCP implementation that supported the options, the two network stacks would use the protocol to decide upon a key and use that key to encrypt all future communications with the IDEA block cipher. [Lev04]

Because there was no certification of the remote system, the Skeeter/Bubba scheme only provided defense against passive eavesdropping, not against an active attacker who could mount a man-in-the-middle attack. The project was abandoned for two reasons. First, an engineer at FTP thought that it would be wasteful to have computers calculate large prime numbers for every TCP connection (none of those working on the project had any training in cryptography and knew how to optimize the system). Second, people in the company who understood security criticized the solution because it was susceptible to the man-in-the-middle attack. Today the Bubba and Skeeter TCP options with the cryptic reference to "[Knowles]" in the Internet's list of Assigned Number RFCs (e.g., RFC 1700[RP94]) are the only remnants of the project.

SSL, Netscape Communications, 1994

The Netscape Navigator web browser, released in beta form on October 13, 1994, came with built-in support for the company's then-proprietary SSL encryption protocol. [Net94a, Net94b]

The early Netscape browsers came with a single pre-loaded X.509 certificate for the RSA Data

Security Commercial Certification Authority; this certificate was used to authenticate remote SSL servers. These servers were only trusted if they presented the Netscape client with an X.509 certificate that was signed by the RSA CA *and* if the certificate had the server's DNS address in the Common Name ("CN") subfield of the X.509 certificate's subject's Distinguished Name. Without a proper certificate, Navigator 1.0 refused to create a "secure" connection.

By requiring the use of certificates to enable encryption, Netscape kicked off the market for certification services. And by mandating the use of a particular key, Navigator established the RSA CA as the world's preeminent certification authority. Future versions of SSL permitted the use of client-side certificates as well, to positively identify web users. But client-side certificates were not mandatory, and that market never really developed.

Navigator 2.0 shipped with a pluggable PKI that supported new CAs by having the user click on a hyperlink and downloading the CA's certificate. The program came with seven certificates pre-installed: CommerceNet, MCI Mall, Netscape Test, ATT Research, RSA Commercial, ATT, and RSA Secure Server. In 1995 RSA's certification services were transferred to a new company, VeriSign, which was created solely for that purpose. Nevertheless, the name "RSA Certification Authority" continues to appear on many certificates.

6.1.1 Understanding the failure of PKI

This chapter will argue that a series of technical decisions made in the 1980s and 1990s to deploy a broad-based PKI system based on X.500-style naming, the X.509 certificate format, and the approach of having multiple Certificate Authorities that could be loaded into client software was fundamentally flawed. Although some organizations have been able to make some aspects of the PKI model work in some situations, the overall system has not achieved anywhere near the adoption that was widely expected in the 1990s. PKI tried to solve too many separate problems at once and ended up solving none of them particularly well. Instead, the market favored easier-to-use solutions that could be incrementally deployed.

The comparison of the Skeeter/Bubba system and Netscape's early SSL is revealing: Skeeter/Bubba was abandoned because the system couldn't make the kinds of security guarantees that would later be made by proponents of SSL. By using anonymous Diffie-Hellman, Skeeter/Bubba would have provided protection against passive eavesdropping but not against active man-in-the-middle attacks. SSL provided defense against both kinds of attacks, assuming that the PKI was properly administered.

Without training in PKI, the engineers at FTP couldn't imagine how to add identity certification to the system that they had created. In fact, they didn't need to. Skeeter/Bubba could have trivially incorporated a self-signed RSA or El Gamal key as part of the protocol. The Skeeter/Bubba implementation could have maintained a record of keys used by remote TCP stacks and issued an alert if those keys changed, similar to the way that SSH does. [Ylo96] This approach would have caught man-in-the-middle attacks that took place after the first time that two systems made contact. But the FTP engineers didn't understand the technology that they were implementing, and they failed.

The engineers at Netscape designed a certificate-based solution that was technically unassailable.

But by limited certificates to those signed by a single key, it may have also limited the number of organizations that could trivially deploy SSL-based servers. SSL could have supported anonymous Diffie-Hellman key agreement as an alternative strategy for establishing a secure connection between client and server for the cases where the server did not have a valid X.509 certificate. (The SSL 3.0/TLS protocol includes this mode of operation in the standard, although few implementations actually support it.) Such a strategy might have allowed for more rapid deployment of SSL-enabled servers, as servers could have been deployed without the difficulty of obtaining a third-party certificate. Instead, a relatively small number of certificates were issued (as will be shown in Section 6.2.7), and the company issuing those certificates saw spectacular growth in its stock valuation as a result.

It is commonly held that halfway security measures are generally not useful: witness that one of the reasons that the FTP engineers abandoned their system is that it would not have been secure against man-in-the-middle attacks. But while the scheme could not have defended against some active attackers, it would have been more than adequate to defeat the password sniffers that plagued the Internet in the 1990s. [Gar96b, FM97]

Netscape's success appeared at the time to be the result of two primary factors. For consumers, Netscape's easy-to-use interface and the point-and-click navigation offered by the web gave many a compelling reason to go online. But for businesses, it was the promise that Netscape's technology would provide "military-strength cryptography" that opened up the real possibility that the Internet might one day be usable for online banking and commerce.

A decade later, Netscape's dream is largely realized. Yet many of the security promises turned out to have been hollow. To understand why, and how this history can be turned around, a short overview is in order.

6.2 Reinterpreting the History of PKI

From the beginning, those promoting public key technology have subscribed to a view that keys should represent human identities. This immediately presented a problem, because it meant that deploying public key technology required that notions of identity needed in order to create a global PKI.

6.2.1 Diffie and Hellman's "public file"

Diffie and Hellman's seminal paper on public key cryptography introduced the concept of a "public file" that contained the enciphering keys for all of the participants in a public key system. One of the purposes of this public file is "to authenticate user *A* to user *B* [and] vice versa. By making the public file essentially a read memory, one personal appearance allows a user to authenticate his identity many times to many users." [DH76, p.34]

The Public File was essentially a database of triplets consisting of names, addresses and keys:

(name, address, k_{name})

The presence of this triplet in the Public File indicated that the information is certified.

Although [DH76] left the terms “name” and “address” undefined, they are commonly taken to mean “legal name” and “street address.” Today the term “Certificate Authority” is widely used to describe the kind of Public File envisioned by Diffie and Hellman.

A second feature, unmentioned in [DH76], is that there was to be a *single* public file. That is, a theoretical user looking up the identity of a theoretical person would never needed to consider which directory in which to look—because there was only one!

This scheme also failed to consider what would happen if two individuals with the same name resided at the same address.

6.2.2 Certificates [Koh78]

Kohnfelder proposed the concept of certificates in his 1978 undergraduate thesis “as an aid in simplifying the communication problems encountered when implementing the method” of the Diffie-Hellman public file. [Koh78, p.2] The motivation, Kohnfelder wrote, was to create a way by which keys could be securely acquired from the Public File. The solution was for the Public File to sign its transmissions to its users!

Once Kohnfelder made this conceptual breakthrough—the realization that digital signatures could be used to sign public keys in addition to messages—he was quick to realize that certificates could be used to keep a local copy of public keys and eliminate the need for participants to continually refer back to the Public File:

“Continually referencing the Public File is a nuisance. When a communicant is initially contacted he must suspend that communication, get the appropriate key from the Public File, and then resume the original communication. Thus either the communicant must use two communication lines at once or break and then reinitiate a communication link.”[p.39]

With certificates, writes Kohnfelder, it is no longer necessary for the communicants to involve the Public File in their day-to-day communications:

“The use of certificates allows key information to be obtained as reliably as if it were from the Public File without ever making contact with the Public File. There is a certificate for each communicant in the system. Each certificate can only be created by the Public File and contains a name and key information pair. Communicants can check that a certificate was created by the Public File. Communicants convey their key information to others simply by sending their certificates.”[p.40]

Kohnfelder used this notation to describe a certificate:

$$\langle A_U, E_U(\cdot), “U” \rangle$$

Where A_U denotes the public file's authenticator, $E_U(\cdot)$ denotes the "encryption function" of the certificate holder (what we today call the certificate holder's public key), and "U" denotes the "plaintext name" of the holder. [p.41]

Today we might want a certificate using this notation to indicate that the signature is actually signing the Name and Key tuple:

$$\{\text{name}, k_{\text{name}}\}_{\text{sig-pf}}$$

6.2.3 X.400, X.500 and X.509

In 1980s a variety of large telecom interests sought to build a grand unified electronic network for all forms of data communications. They called this project the Open Systems Interconnect (OSI).

The OSI email system was defined by X.400, an international standard developed by CCITT (since renamed the ITU-T). This system standardized all aspects of electronic email, including message creation, transit, delivery, multi-media encapsulation, and security. X.400 also provided for message transit between the X.400 world and Telex, facsimile, and physical mail. [Alv97, Wik] X.400 can be thought of as a single system designed to provide functionality similar to what the SMTP, POP, MIME and S/MIME standards do today.

X.500 was the directory service designed to support the X.400 mail system. The standard's Directory Access Protocol (DAP) can be thought of as a master directory of names, addresses, phone numbers, and other information that was designed to be used both inside and outside organizations. X.500 can be thought of as a combination of the today's Internet Domain Name System (DNS) and the Lightweight Directory Access Protocol (LDAP).

Names in the X.500 system were called "distinguished names," implying that they were unique (that is, *distinguishable*). These names are created from a set of *relative distinguished named* (RDNs) that are normally displayed as a short one-or-two letter abbreviation, an equal sign, and a human-readable string. The RDNs are concatenated together, separated by forward slashes, to form the distinguished name. This model meshed remarkably well with the telecom companies that created it, most of which had monopoly control for a single geographical area.

For example, if John Wilson were a user at the University of Auckland's computer science department, he might have a certificate indicating that his common name was "John Wilson," that he was a member of the Organization Union "Computer Science," which was in turn a member of the Organization "University of Auckland," which in turn was in the Country "NZ." Each of these organizations could in fact maintain their own certifying CAs, as illustrated in Figure 6-1.

Because the directory could contain confidential information such as lists of employee names, X.500 included provisions for authentication and access control lists. One level of access control was needed to view parts of the non-public directory, while another level was required to make changes. (This is similar to today's Dynamic DNS protocols.)

Several mechanisms were specified for access control to the directory: passwords, a simple challenge-response mechanism, and the use of public keys. Thus was born the X.509 certificate standard as a

system for controlling access to and modification of X.500 directories.

No information regarding access control or permissions was stored in the original X.509 certificates, because the only intended use of the certificates was for directory access control. (This limitation was overcome in X.509v3, which created an extension mechanism.) Many people who use them consider the X.500 and X.509 systems to be unwieldily, a classic result of design by committee. “Although no real directories of this type were ever seriously deployed, PKI designers and users have had to live with the legacy of this approach ever since,”[Gut02b, p.2]

6.2.4 X.509 and PEM

Rather than invent its own certificate format for distributing public keys, when the when the IETF PEM committee needed a data format for holding certificates, it adopted X.509. Thus, a system that was designed to use private keys as tools for controlling access to directory information was significantly changed into a system that used possession of those private keys as proof of identity.

Subsequent standards such as S/MIME and SSL followed suit.

6.2.5 Distinguished Names in Practice

One of the key differences between the X.500 standard as envisioned and the way that Distinguished Names have played themselves out is that there has been only a passing effort to ensure that the names are correct and globally unique.

For example, consider the SSL certificates used to certify the Amazon.com web server, shown in Figures 6-2 and 6-3. The issuer’s self-signed root certificate has the country name of “US,” the organizational name “RSA Data Security, Inc.,” and the organizational name “Secure Server Certification Authority.” The distinguished name on the subject’s certificate states that Amazon.com is located in Seattle, Washington, US, and that the common name is `www.amazon.com`, which is the practice with SSL certificates.

The problem here is that the Subject’s certificate was not in fact issued by RSA Data Security: it was issued by VeriSign, Inc., an independent company. As previously noted, RSA sold its certification services to VeriSign in 1995—ten years before the certificate was issued! A footnote on page 16 of the *VeriSign Certification Practice Statement version 3.0* notes that the Organization (O) field of VeriSign’s CA Certificates states should say “VeriSign, Inc.,” but that “An exception to this is the Secure Server CA, which indicates “RSA Data Security, Inc.,” but is now a VeriSign CA.”[Ver05b]

Unfortunately, there is no reference to the CPS in either the Issuer’s Certificate or the Subject’s Certificate, making the authority of the CPS over these certificates questionable at best. A secondary problem is that Amazon.com is a Delaware corporation, not a Washington corporation.

Furthermore, since the CA’s name is not technically part of the Distinguished Name, if another CA were to issue a certificate allegedly for Amazon.com but given to another entity, those two certificates would be indistinguishable in many web browsers. In practice this hasn’t been a problem, but in theory it is a very significant problem. Thus, even though X.509 was built to give assurances of globally meaningful identity, in practice it does not.

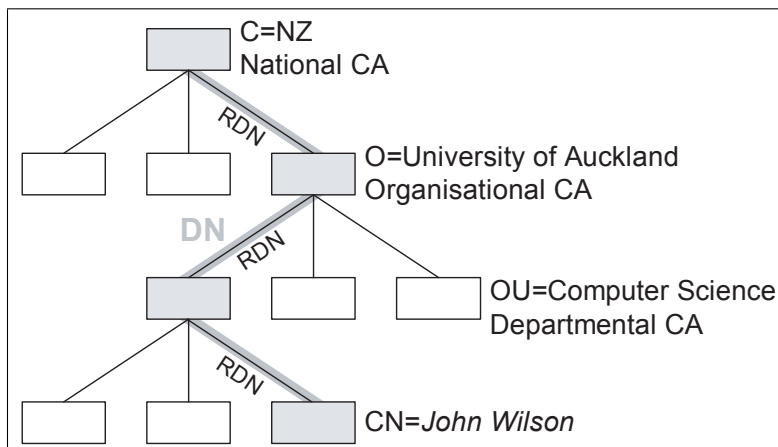


Figure 6-1: X.500 directory and certificate model. A different certificate authority (CA) is attached to each part of the directory to manage access control, while relative distinguished names (RDNs) define a path through the directory and together form a distinguished name. From [Gut02c, Fig. 1], *slightly modified and used with permission*

```
% openssl s_client -connect www.amazon.com:https
...
CONNECTED(00000003)
---
Certificate chain
 0 s:/C=US/ST=Washington/L=Seattle/O=Amazon.com Inc./CN=www.amazon.com
   i:/C=US/O=RSA Data Security, Inc./OU=Secure Server Certification
   Authority
---
```

Figure 6-2: The X.500 Distinguished Name (DN) for the subject (“s:”) and the issuer (“i:”) of the SSL certificate for Amazon.com’s secure web server, read on March 23, 2005 with the OpenSSL command above. (Although not displayed by OpenSSL, the client certificate was issued on 1/5/05 and expires on 1/6/06.)

6.2.6 A taxonomy of PKI trust models

Kaufman, Perlman and Speciner have created a taxonomy of PKI trust models.[KPS02, §15.3] This taxonomy discusses seven trust models which appear to cover all variations present or that have been proposed:

Top-down models:

- **The Monopoly Model**, in which a single organization is universally trusted by all companies, countries, universities, organizations, and individuals in the world. “This is a wonderfully simple model, mathematically,” the authors note dryly. “This is the model favored by organizations hoping to be the monopolist.” There are many disadvantages to this model, not the least of which is that “the entire security of the world rests on that one organization never having an incompetent or corrupt employee who might be bribed or ticked into issuing bogus certificates or divulging the CA’s private key”

This is the model that was proposed by [DH76].

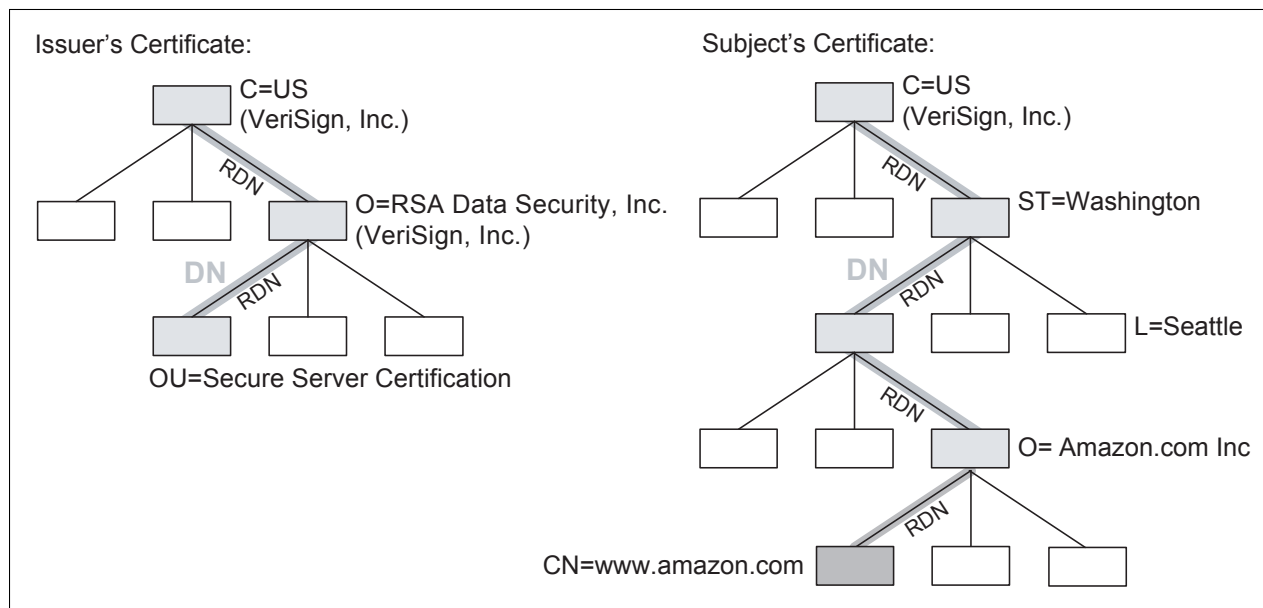


Figure 6-3: A graphical representation of the X.500 distinguished names presented in Figure 6-2

- **The Monopoly plus Registration Authorities Model**, in which the monopolist has delegated the task of registering individual keys. “This model’s advantage over the monopoly model is that it is more convenient and secure to obtain certificates... However, all of the other disadvantages of the monopoly model apply.”
- **The Delegated CAs Model**, in which the monopolist signs the delegating CA’s certificate, and the delegated CA signs the subject’s certificate. The advantage to this approach is that a relying party can see the chain of trust.
- **The Oligarchy Model**, in which there are many trust anchors, each of which can issue certificates for any name in the universe. This model spares users monopoly pricing, but it is less secure than the monopoly model, since a compromise at *any* of the trust anchors can subvert the entire system.

This is the model that has been deployed in today’s browsers and S/MIME clients.

- **The Top-Down Model with Name Constraints**, in which the root CA places constraints on the kinds of names that can be certified by the delegated CAs. This model eliminates some of the problems with both the Monopoly model and the Delegated CAs Model, but it still has the problems that are inherent in a single root.

This is the model that was adopted by PEM. The “name constraints” part of the model is similar to the Domain Name System.

Non-Hierarchical Models:

- **The Anarchy Model**, in which each user is responsible for configuring their own trust anchors. “To get the key of someone whose key is not in your set of trust anchors, you can search through the public database to see if you can find a path from one of your trust an-


```
Public Key Server -- Index ``george bush ``
```

Type	bits	/keyID	Date	User ID
pub	1024D/5FC77CDB		2005/04/12	George W. Bush <gwbush@whitehouse.gov>
pub	1024D/0CB5C0BC		2004/09/21	George Walker Bush (DOD) <president@whitehouse.gov>
pub	1024D/53BB7633		2004/03/04	George Walker Bush (DOD) <president@whitehouse.gov>
pub	1024D/CC9755AF		2003/09/04	George W. Bush (POTUS) <president@whitehouse.gov>
pub	1024D/D53B305C		2003/06/27	George W. Bush (Seal of the President of the U.S.) <george@whitehouse.gov>
pub	1024D/5ED285FC		2003/05/06	Thomas A. Amoroso, MD FACEP (George W Bush Delenda Est) <tamoroso@aq.org>
pub	1024D/8135507E		2003/03/28	George Bush <bush@hell.com>
pub	1024D/35687EDD		2001/12/19	George W Bush <don't_mess_with_Texas@yazhoos.gov>
pub	1024D/9E26D3E6		2001/10/26	George W. Bush <gwbush@whitehouse.gov>
pub	1024D/5E315572		2000/09/15	george w bush <pdcl_2@hushmail.com>

Figure 6-4: A search for the string “george bush” on the PGP public key server on April 29, 2005, shows the kinds of attacks that can be waged against the Anarchy Model. It is highly unlikely that any of these keys actually belong to any individual named “George Bush.”

chors to the name you want. This absolutely eliminates the monopoly pricing, but it is really unworkable on a large scale.”[KPS02]

This is the is the model that is used by PGP.

Kaufman, Perlman and Speciner readily admit that this model works well for small communities where the members are trustworthy, but that it falls apart when there are hostile players who inject many bogus certificates into the system—especially when nothing limits the number of certificates that each player can create. (See Figure 6-4).

- **The Bottom-Up Model with Name Constraints**, in which each organization creates their a private PKI and then allows other CAs to issue names for a particular portion of the global namespace. This is similar to the model adopted by Lotus Notes, and it is becoming the *de facto* model of many PKI “bridges” that are currently being deployed in the United States.

Kaufman, Perlman and Speciner seem to like this model a great deal (not tremendously surprising, considering the team’s history), although they note that there is nothing to prevent a single “carbon-based life form” from obtaining multiple certificates under different identities. That might even be a feature.

6.2.7 The SSL Server PKI today: E-Soft’s survey

So just how successful was Netscape and the rest of the Internet community in establishing the PKI certificate infrastructure on which rests SSL’s resistance to man-in-the-middle attacks?

The E-Soft security consulting firm in Ontario, Canada, has conducted a monthly “Security Space” survey of web servers across the Internet since September 1998. The company’s March 1st, 2005 survey of 209,596 responding SSL-enabled servers found that only 88,690 servers (42.3%) had certificates that were valid. Of the remaining certificates, 49,563 (23.6%) had a certificate-host mismatch; 44,963 (21.4%) were signed by an unknown CA (that is, a CA not distributed with the Microsoft Internet Explorer browser); 17,891 (8.5%) were self-signed; and 16,547 (7.9%) were expired. [Sec05b] More than 10 years after Navigator first shipped, it seems that more than half of

	Count	%
Total Valid Certificates	88,690	42.3%
Invalid Certificates:		
Certificate-Host Mismatch	49,563	23.6%
Unknown Signer	44,963	21.4%
Self-Signed	17,891	8.5%
Expired Certificate	16,547	7.9%
Total Invalid Certificates:	120,906	57.7%

Table 6.1: Results of the Security Space March 2005 survey of invalid SSL certificates. Invalid certificates do not total because some certificates have multiple problems.[Sec05b]

the sites that are using SSL are nevertheless vulnerable to man-in-the-middle attacks because they do not have valid certificates. These results are presented in Table 6.1.

To analyze the certification trends, we downloaded six years of the E-Soft’s survey data and performed a meta-analysis to determine the trends of self-signed and expired certificates over that time period. After working with E-Soft to correct a variety of data discrepancies that were observed, the following conclusions were reached:

- The significant drop in number of SSL site certificates between September 1998 and March 1999 is due to a change in the way that E-Soft found the sites that it included in its analysis. At first, sites were found through a combination of link-following from existing sites (“spidering”) and systematic probing of the IP address space and, in one case, a download and probing of an entire top-level-domain. Reinke believes that the probes and zone transfers found many test systems that were not designed for public use. In March 1999 E-Soft ceased probing for web sites and restricted itself to those found through the “spidering” technique.
- E-Soft did not obtain a copy of the GeoTrust CA key until the fall of 2001. As a result, the spike in sites that are signed by an “unknown signer” June 2001 through October 2001 is probably a result of GeoTrust coming online; the sudden drop of sites Security Space reports as being signed by an “unknown signer” in November 2001 is probably the result of E-Soft acquiring the GeoTrust CA key.
- The steadily increase number of sites signed with an “unknown signer” in 2003 and 2004 may be the result of VeriSign using a new key for signing sites that E-Soft cannot use because of license restrictions.[Rei04]

Figure 6-5 shows the total number of SSL certificates included in the E-Soft survey over the six years of data from September 1998 through March 2005. Figure 6-6 shows the percentage of certificates that were valid.

A large number of the “unknown signers” may actually be web sites created with the OpenSSL CA.pl perl script. This script creates both private keys for a fictional “snake oil CA” and for the web site as well. Such certificates would appear as “unknown signers” in the E-Soft data, not as self-signed sites. This hypothesis cannot be verified, as E-Soft has declined our invitation to share its raw data so that an analysis of X.509 Subject and Issuer names can be performed.

Despite these caveats, the E-Soft data is important because it reflects X.509 certification trends in general, rather than those of the most popular sites on the Internet. As such, the data indicates that web site administrators have been steadily increasing *both* the number of systems equipped with SSL. Meanwhile, the percentage of X.509 certificates that were “valid” slowly climbed from roughly 10% to somewhere between 40% and 50%. Finally, while the number of self-signed certificates has remained more-or-less constant, the number of “unknown signers” has steadily increased.

On the other hand, the Netcraft April 2005 web server survey found 62 *million* web sites—counted by hostnames—a gain of roughly 1.7 million sites over the March 2005 survey.[Net05a] Of these, approximately 28 million were labeled by Netcraft as “active.” If these numbers are both to be trusted and comparable with the E-Soft numbers, then there are roughly 100 times as many web sites than implement HTTP alone as those that implement both HTTP and SSL-protected HTTPS. This may be reasonable, as many web sites clearly do not have an SSL component. Likewise, many businesses contract with third-party web sites for services such as shopping carts or credit-card processing; these services are typically provided under the third party’s domain name. Thus, thousands of web sites that claim to be “secure” may, in fact, all use the same third-party SSL domain name to collect a credit card numbers. Fundamentally, such outsourcing is indistinguishable to users from a phishing attack.

Perhaps SSL will see deeper penetration in another 10 years...

6.2.8 The SSL Client PKI today: Counting Thawte Freemail Certificates

Gutmann notes there are few studies available that allow one to quantify the success of client-side PKI deployment. Most studies that consider PKI for authenticating users tend to be post-mortem analyses of failed projects.[Gut03, p.45] But surprisingly, one source of data to gauge the success of client-side PKI is the serial numbers that Thawte Consulting places on its Thawte Freemail certificates.

Thawte Freemail Digital ID certificates are unique in today’s world: these certificates are the only certificates for which the CA key is widely distributed by software vendors, that work with S/MIME, and that are available for free. Thawte’s original business model was to give away the certificates with the Common Name “Thawte Freemail Member” and then to charge the certificate holder a small fee to have the user’s actual name appear in the Common Name field. Thawte refuses to comment on the success of this business strategy, but given the apparent lack of individuals using Freemail Certificates, it has not been very successful.

Every X.509 certificates contains a certificate serial number. According to Section 3.3.13 of the draft X.509v4 standard, certificate serial numbers must be “An integer value, unique within the issuing authority, which is unambiguously associated with a certificate issued by that CA.”[Tel01, p.14] That is, no two certificates from a confirming CA may have the same serial number, but otherwise serial numbers can be whatever the issuer wishes. Thus, a CA could put on its serial number the time-of-day down to the nanosecond that the certificate was issued, a hash of the certificate’s public key, or even an ordinal counting the total number of certificates that the CA has issued.

In the course of working on this thesis, we obtained a Thawte Freemail certificate on September 10, 2004 with serial number 0d 04 d8 (853208₁₀). A second certificate was obtained for an

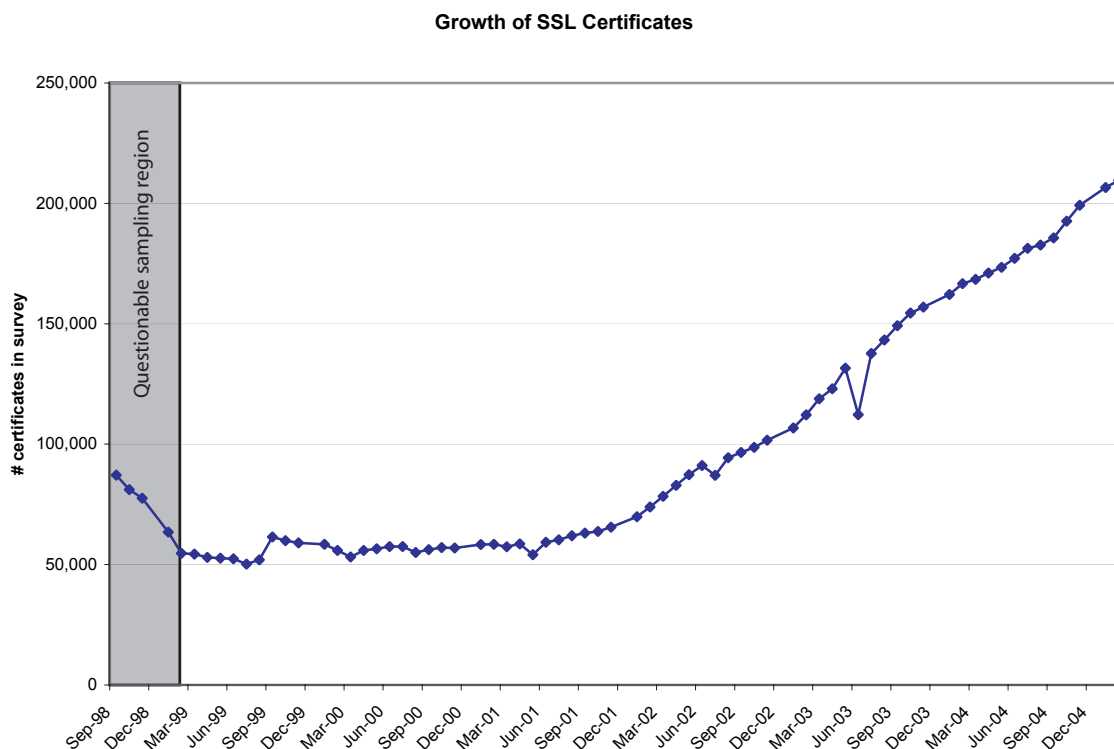


Figure 6-5: Total SSL Certificates in the E-Soft SSL Survey. The full survey data can be found at <http://www.securityspace.com>. (For comparison, the Netcraft April 2005 web server survey found 62 million web sites—counted by hostnames, as opposed to IP addresses. Of these, approximately 28 million were labeled by Netcraft as “active.”[Net05a])

unrelated purpose by Marvin Garfinkel on September 26, 2004, with serial number `d1 1d a9` (`85956110`). These numbers seemed suspiciously close, possibly indicating that Thawte might be issuing certificates in sequential order.

To test this theory, we obtained two Freemail certificates on March 25, 2005. Those certificates were obtained 20 minutes apart and had decimal serial numbers of 940,275 and 940,282, a difference of 7. Once again, this seems to indicate that Thawte was indeed issuing certificates in sequential order, although it wasn’t possible to tell from this sampling if numbers are being skipped. (See Table 6.2.)

A linear regression of all four certificate serial numbers and issuance dates found an average certificate issuance rate of 446 certificates/day ($R^2 = 0.99993443$, $p = .00002$). Since Thawte’s certificates have a lifetime of 1 year, a rate of 446 certificates/day translates into roughly 160,000 certificates that are outstanding at the present time. This is roughly 75% of the number of SSL server certificates in the E-Soft survey.

One way to check the accuracy of this number would be extrapolate back and see if the *lifetime count* of Freemail certificates makes sense. If Thawte really has issued 940,000 Freemail certificates, than the current rate could account for approximately 6 years of issuance. In fact, the Thawte

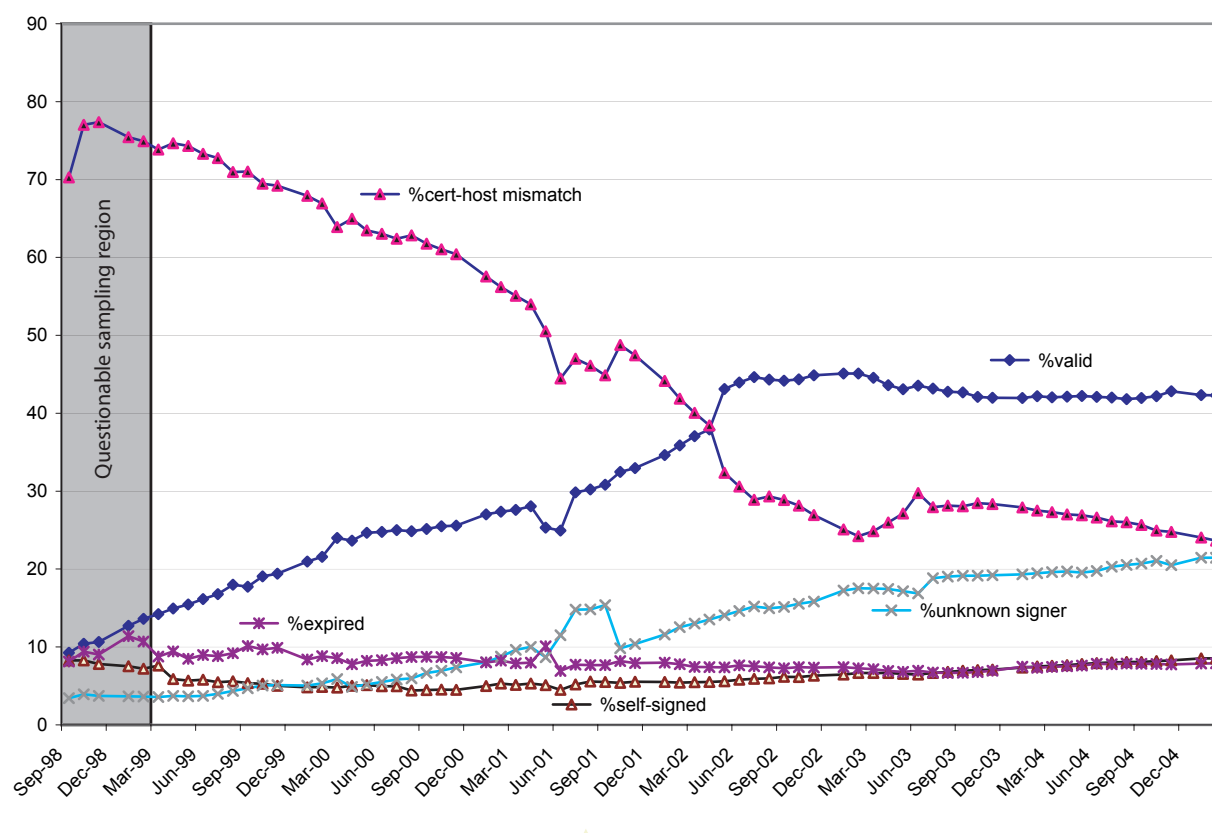


Figure 6-6: Percentage of SSL certificates that are valid or that are invalid because they have a certificate-hostname mismatch, they are self-signed, they are signed by an unknown signer, or because they are expired. This graph based on an analysis of the E-Soft survey at <http://www.securityspace.com>

Freemail CA certificate has validity dates from 12/31/1995 through 12/31/2020—indicating that Thawte has been making certificates for at least 9 years with this key. The regression numbers are consistent if the Freemail service ramped up from a standstill to its current rate of issuance.

A spokesperson for Thawte said that she “can neither confirm nor deny the e-mail certificate volumes we produce due to competitive reasons.”[Bax05]

For comparison, on March 20, 2005 the PGP key server `keyserver.kjssl.com` was found to have 305,884 keys in its key database—a collection rate of roughly 30 thousand keys per year, as the key servers first became generally available in 1994. As keys cannot be deleted from the PGP key server, it is unknown how many of these keys are active and how many are abandoned.[Har05]

The E-Soft report shows that the use of PKI for certifying *servers* has been reasonably successful: both the number of SSL servers *and* the number of servers with valid certificates has steadily increased. It’s only the percentage of servers with valid certificates that has leveled off.

Principal	Certificate SN		Date	Elapsed Days
	Hex	Decimal		
Simson Garfinkel	0d 04 d8	853,208	September 10, 2004	0
Marvin Garfinkel	0d 1d a9	859,561	September 26, 2004	16
Jared Garfinkel	0e 58 fe	940,275	March 25, 2005	196
Draken Garfinkel	0e 58 fa	940,282	March 25, 2005	196

Table 6.2: Thawte Freemail certificates obtained for this dissertation

On the other hand, the Thawte analysis makes it clear that the use of PKI for certifying the identity of *individuals* on the public Internet has been a failure. Although there are many organizations that use PKI technology internally, this technology is invariably used to certify a pre-existing relationship, not to convey identity to unrelated third parties. That is, these are *private PKIs*, not *Public PKIs*. Some people call this use of PKI technology “PKI with a little-p.”

6.2.9 On the difficulty of obtaining a Thawte Freemail certificate

One reason that might explain the relative paucity of user certificates issued by Thawte is that Internet users simply don’t think that it is important to obtain these certificates.

To test this possibility, we documented the step-by-step procedure required to obtain a Freemail certificate. In all, 26 steps were required:

1. Go to <http://www.thawte.com/> and select “Personal E-mail certificates” from a hidden menu.
2. Click a button labeled “join.”
3. Agree to Thawte’s 2000-word Certificate Practice Statement.
4. Specify the “Charset For Text Input,” name, date of birth, and nationality. (Thawte does not issue certificates to children under 13, possibly to avoid problems with the US Children’s Online Privacy Protection Act. Thawte also does not issue certificates to people over 95 years old.)
5. Provide a National Identification Number (e.g., a driver’s license number, passport number, or social security number).
6. Provide an email address.
7. Choose Language Preference and Charset Preference settings for the certificate.
8. Pick a personal password. According to Thawte, “This is the most important page in the entire enrollment process.”
9. Enter a preferred contact phone number and answers to at least 5 challenge questions. The questions include “What is your mother’s maiden name? What is your father’s middle name? What do you enjoy doing the most? What is your partner’s nickname? What is the make of your fridge? What is the location of your dream holiday? Who is your favorite author? Who is your favorite actor? What is your car registration? How many pets do you have?”
10. Confirm enrollment information by clicking on a link.
11. Receive an email message sent to the email address that was provided in Step 6.

12. Click on the link that is sent in the email and enter the “Probe” and “Ping” values that were sent. (e.g. “6LCUcsXpsarvnRTxy4yf9EU” and “K4nUHbt9TdsJ7T79w8pXUE”)
13. Check the box “I enrolled because an ESO requested I obtain Name Validation or Strong Extranet Certificates” if “you are enrolling because an PKI Security Officer (SO) has requested that you obtain a Name Validation Certificate or a Strong Extranet Certificate.” (Notice that the ESO acronym is never defined.)
14. Enter the email address provided in Step 6 as a username, and enter the password provided in Step 8 into a pop-up browser box.
15. Request an “X.509 Format Certificate” by clicking the “request” button that contains a shopping cart. (The other option is to click on the button labeled “test” but users are told “Please do not request these certificates unless you know what you are doing.”)
16. Select the program you are using: Netscape Communicator or Messenger, Microsoft Internet Explorer, Lotus Notes R5, OperaSoftware Browser, or C2Net SafePassage Web Proxy.
17. Confirm your employer. In the case of Freemail certificates, the only employer that can be selected is “No Employment Information Available.”
18. Select the email address to put on the certificate. (If the box isn’t checked, the certificate won’t work.)
19. Select which of the “Strong Extranet” certifications that should appear on the certificate.
20. Configure X.509v3 certificate extensions, either by accepting the default extensions, or by explicitly configuring them. Users are advised to avoid clicking on configure “unless you know what you are doing.”
21. Select either a 2048 (High Grade) bit key or a 1024 (Medium Grade) bit key.
22. Wait for the Key Generation to complete.
23. Confirm the certificate request.
At this point Thawte sends email #2 which says “Thank you for requesting a certificate from us. We will issue it as soon as possible and notify you by email when it is done.”
By the time the email was received, the certificate was in fact already issued.
24. Click on the easy-to-miss link labeled “here” to go to the personal e-mail certificates management page.
25. Click on the word “Navigator” (if you are using Mozilla Firefox.)
26. Click on the button with an icon of a dog labeled “fetch.”

When this process was finished, there is no visual indication that anything has happened, although the certificate had been installed in the web browser. Using the certificate with Mozilla Thunderbird required exporting the certificate and private key from Firefox and importing the certificate into Thunderbird—a complicated process. But perhaps the most infuriating part of this entire process is the fact that only certifies the holder’s *email address* appears on the Freemail certificate, yet the registration process requires that the user provide a full legal name, national identification number (such as a social security number or passport number), date of birth, and other confidential information. This is a clear violation of the European Union’s data protection rule, which requires

that the collection of personal information to be minimized. What's even worse, it was evident that the only piece of information that Thawte actually verified was the email address. This verification could have been trivially performed simply by mailing the certificate to the user's email address, or by mailing a link that could be used to download the certificate.

Based on this analysis, we concluded that it is likely that only the most highly motivated individuals will go through this process and provide such personal information to Thawte.

As an aside, it is also worth noting that neither the mail that the Thawte certificate server sends out nor the mail sent by the company's press officer is digitally signed. One could also question a CA certificate that has a validity period through 12/31/2020 but only a 1024-bit RSA public key.

6.3 Alternatives to X.509

X.509 is important because it is the basis of the Internet's PKI. But X.509 is not the only PKI standard currently in use. This section briefly reviews four other PKI systems and explores how they perform key certification.

6.3.1 Key certification and distribution in PGP

PGP is an email security program that was specifically written to enable anti-government activists in the US and Central America to have electronic communications that could not be intercepted by their government.¹

Because of this design goal, PGP does not require trusted third parties to either make or certify keys. Instead, PGP users can make their own keys and immediately use them to sign messages. Users can send their keys out over networks or place them on floppy disks and distribute them using "sneaker-net."

Keys can also be downloaded from an untrusted location (e.g., a key server or a web page) and authenticated through the use of the key's hash, or "fingerprint," that is itself obtained in a trusted manner. For example, it is common practice in some communities for people to place hashes of their PGP fingerprints on their business cards.

PGP provides for third-party key certification using a mechanism that Zimmermann calls "the web of trust." Briefly, PGP allows any PGP user to "sign" a key belonging to any other PGP user. By signing a key one is making an assertion that the public key really belongs to the individual whose name is on it. PGP then allows users to decide which individuals that they trust and which they do not. In this manner, the set of trusted keys for each PGP user is a graph that places the user at the graph's center, surrounded by edges that connect the user to a set of trusted individuals, and finally connected through a second set of edges to all individuals that are certified by all individuals that are trusted by the PGP user. Although in principle the web could extend to higher orders, the initial PGP implementation limited the span of the trust graph to two, and that limitation has been preserved ever since.

¹Although in 1991 Zimmermann said that he released PGP because he feared that legislation pending in the US Senate would have made the use of strong cryptography illegal in the United States, he has since said that he specifically intended for PGP to be used by democracy activists in Central and South America. [Zim00]

PGP appears to work well in small trusted groups, but its promoters have not been successful in scaling the PGP key certificate model to a large user base.

6.3.2 SPKI/SDSI

The simple public key infrastructure (SPKI) / Simple Distributed Security Infrastructure (SDSI) is an effort to overcome the complexity and scalability problems of X.509-based PKI systems. In SDSI public keys are valid globally, but identifiers (“names”) bound to public keys are only valid locally to the entity that creates the binding. Multiple names per key are allowed. In SPKI, authorizations are made locally. SPKI/SDSI is specified by RFC 2692[Ell99] and RFC 2693[EFL⁺99].

Although there has been some academic work with SPKI/SDSI technology, there have been no significant deployments.

6.3.3 PKI and Key certification and distribution in Lotus Notes

With its integrated PKI and more than 100 million users, Lotus Notes is one of the most successful deployments of PKI technology to date.[Zur05b] One of the reasons that Notes has been so successful is that each organization is an independent PKI island: there is no attempt to place all of the various Notes installations under the umbrella of a single parent PKI. As a result, each organization deploying Notes is free to establish its own identification and certification policies.

Notes uses a proprietary PKI system in which the system administrator makes each user’s keys in advance, certifies them, and distributes the keys to Notes users in a so-called “user file.” The public keys are stored in the Notes directory. Since every Notes user in the directory has a key pair, it is a simple matter to encrypt mail for any recipient that a Notes user can look up.

Instead of using X.500-style Distinguished Names, the Notes identification system has the appearance of the individual’s name, a slash, and the name chosen by Notes administrator for their Notes installation. Thus, Zurko’s Notes name is:

Mary Ellen Zurko/Westford/IBM

It is easy to see that Zurko’s affiliation is with some kind of facility that IBM has in Westford. The name doesn’t make clear whether Westford is a geographic location or an area of specialization, but it ultimately isn’t very important.

Organizations that wish to exchange secure information with each other can cross-certify. When cross-certification occurs, the installation name naturally appears when identities are displayed in the Notes interface. So if IBM chooses to cross-certify with Philips and Zurko receives a signed message from Jan Brands, the name she sees in her Notes client might look like this:

Jan R. Brands/EHV/SC/PHILIPS

In the case of Brands, not only is it easy to see that his affiliation is with Philips, but it is also evident that he is with a group called EHV that’s part of some group called SC. We don’t need

to know that EHV means “Eindhoven” and that SC means “Semiconductors” in order for this Jan R. Brands readily distinguishable from another Jan R. Brands who might be in a different Philips group. (The John Wilson problem would rear its ugly head if Philips had two Jan R. Brands working at the Eindhoven office.)

This is *not* simply an X.509-style certificate without the individual components (O, OU, C and L) being labeled: In this hypothetical example, IBM’s management has made an affirmative decision to sign the root key of the Phillips Corporation and give it the name /PHILIPS. IBM’s manager could just as easily given the Philips Corporation it the name /YUMMY.

Any organization can purchase a copy of Notes and call itself IBM. But unless that organization cross-certifies itself with IBM, the copy of the Notes client running on Zurko’s computer won’t display the other organization’s certificates with the /IBM suffix.

The Notes S/MIME implementation allows individual Notes users to make their own personal cross-certification decisions based on received S/MIME certificates. This certification path is shown when signed messages are displayed in the notes interface; the interface even allows the user to specify alternative names that should be displayed, as shown in Figure 6-7.

Using the notation introduced earlier in this chapter, Notes certified identities could be represented like this:

$$\{\text{name}, k_{\text{name}}\}_{\text{sig-pf}}^{\text{name-pf}}$$

If we expand our notation so that information displayed on the screen is printed in **CAPITALIZED BOLDFACE**, then Notes certified identities actually appear like this:

$$\{\mathbf{NAME}, k_{\text{name}}\}_{\text{sig-pf}}^{\mathbf{NAME-PF}}$$

Thus, Notes uses *global identifiers with locally meaningful names*. This approach makes it rather easy for Notes users to visually distinguish identities that have been certified by different Notes administrators—even in the case where Notes domains have cross-certified.

6.3.4 Key certification and distribution in Groove Virtual Office

Built by many people who had worked on the original Notes system, the Groove Virtual Office is designed to overcome many of the problems that were discovered through the use of Lotus. For example, many people participate in different organizations using different kinds of identities: A consultant might have the organization that she belongs to, where she is an employee, the organization where she is consulting, where she is a contractor, and a parent/teacher association, where she is a parent. Groove recognizes that the same individual can have multiple roles by allowing a single user to work with multiple identities without having to log out and log back in to the Groove system.

Each Groove identity consists of a key pair that is used for signing and another that is used for

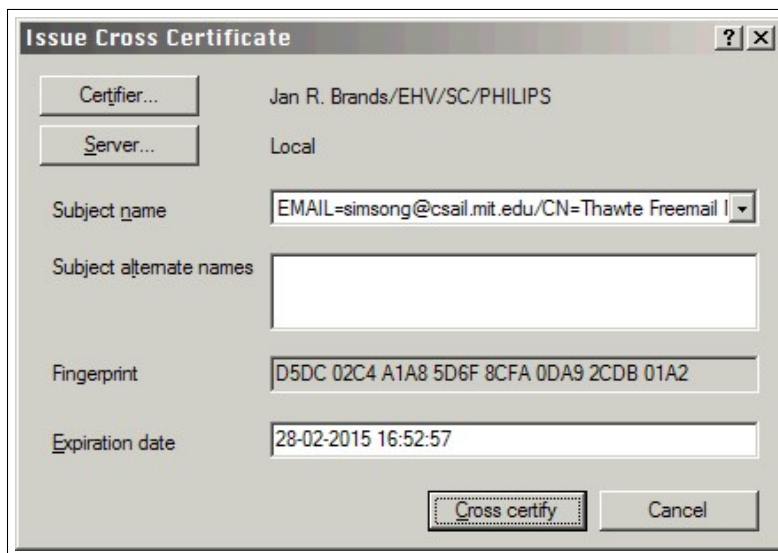


Figure 6-7: Notes allows individual users to certify S/MIME keys that they receive. In this panel, Jan Brands at Philips is given the option of certifying a particular Thawte Freemail certificate. Image courtesy of Jan Brands, *used with permission*.

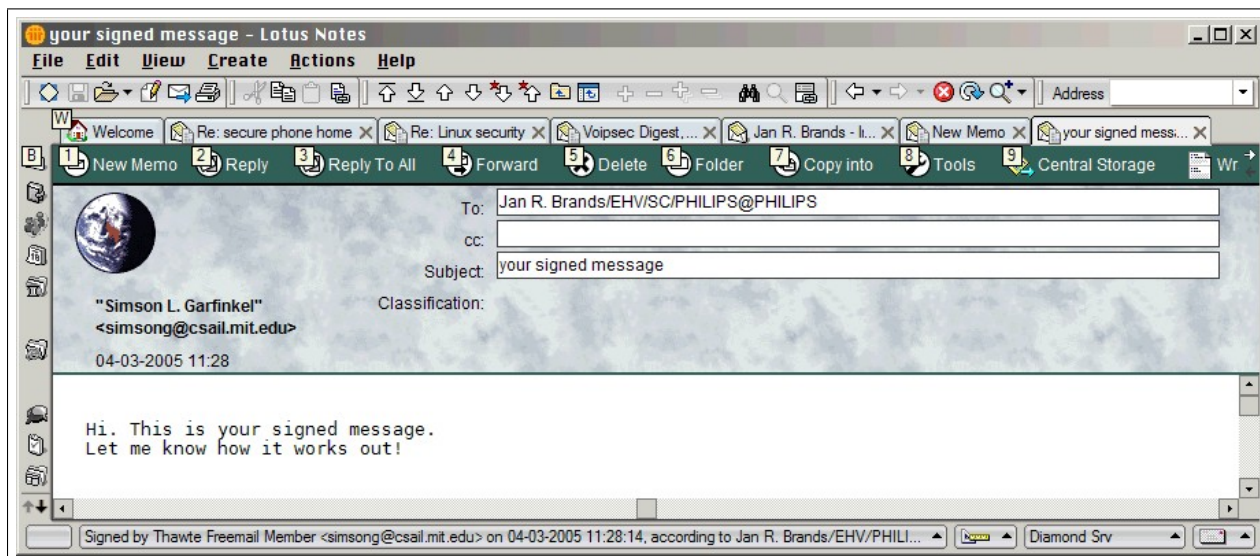


Figure 6-8: When Notes displays a digitally signed message, the distinguished name on the certificate and the certifier are displayed at the bottom in the status. In this case, the display indicates that the identity was certified as a result of the trust dialogue shown in Figure 6-7. Image courtesy of Jan Brands, *used with permission*.

sealing.² Each user's sealing key pair are certified by the signing keys. Groove supports three key certification models. When identification names for the keys are displayed in the Groove user interface (Figure 6-9), they are displayed in one of five colors, the color choice depending upon the certification model that is in use and the level of certification that the key has received:

- A single Groove administrator can certify all of the keys used in the organization. This is the traditional PKI model. Groove shows such identities in *teal*.
- Two organizations choose to cross-certify—that is, to trust each other's identity assertions. Groove displays these identities in *blue*.
- Individuals can trade keys (easily done through the Groove interface) and then directly certify each other's keys—for example, by reading a key fingerprint over a telephone. This is similar to the PGP direct certification model. Groove displays these identities in *green*.
- Identities that are unauthenticated are displayed in *black*.
- If Groove discovers that there are two identities in a user's address book that have the same name, it will display those identities in *red*. The user is then given a chance to disambiguate the two identities.[MBA05] (Figure 6-10)

Using the notation introduced earlier, the visual displayed Groove certification would appear like this:

$$\left[\{ \text{NAME}, k_{\text{name}} \}_{\text{sig-pf}} / \text{NAME-PF} \right] \text{CERTIFICATION-TYPE}$$

According to Groove's designers, this certification model has worked well in circumstances that are not amenable to traditional certification practices:

“For example, negotiators in the talks between the Sri Lankan government and the Tamil Tiger rebels used Groove Virtual Office. Neither side wanted the other to run a certifying server. Even cross-certification was unacceptable because of intense political sensitivities... With direct authentication, two parties who want to communicate securely can authenticate each other without having to trust a third party or even each other. In the case of the Sri Lanka peace talks, this allowed the two sides to communicate in a neutral space that no one controlled. In a corporate world, direct authentication allows you to securely communicate with an external party without having to wait for your IT department to issue a certificate.” [MBA05]

6.3.5 Gutmann's survey recommendations

Finally, Gutmann has performed a “PKI Technology Survey and Blueprint”[Gut01], which looks at the question: “If you asked experienced programmers, sysadmins, and technical project managers how they would implement certificate management, what would the technology framework for

²Different keys are used so that an organization can escrow sealing keys without the need to escrow signature keys. This allows the organization to unseal the contents of any message without giving the organization's management the ability to sign messages so that they appear to be signed by the employee.



Figure 6-9: the groove launch bar shows the identities that are known to the Groove user. Where the identities have been certified by an administrator through the use of a PKI, Groove uses the Lotus Notes syntax of a slash followed by the organization's name. Different colors are used for different kinds of authentication. Illustration courtesy Groove Networks. *Used with permission.*

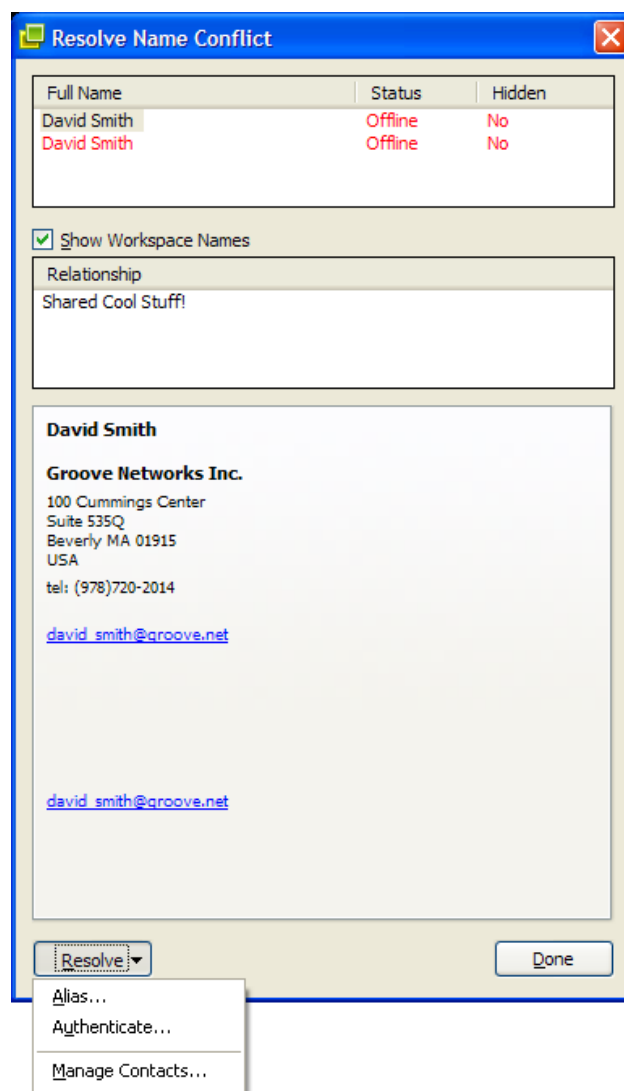


Figure 6-10: If two names in the Groove Launchbar have the same visual name but a different public key, the user is invited to resolve the conflict with the Resolve Name Conflict panel. To resolve the conflict, the names are given different visual appearances. Illustration courtesy Groove Networks. *Used with permission.*

your PKI look like?” The survey then creates a blueprint for deploying PKI based on the responses from his participants. The survey makes the following recommendations:

- For an **identifier**, replace the X.509 Distinguished Name with either an email address, a DNS name, or an IP address.
- For a **unique identifier**, replace the X.509 Distinguished Name with a globally unique identifier—for example, a randomly chosen 128-bit number.
- For **storage of certificates**, replace the X.500 directory with a standard database.
- For **access to certificates**, replace the OSI DAP or the revised LDAP with an HTTP-based protocol.
- For **validity checking**, replace the X.509 certificate revocation lists with a repository presence check: if the certificate is in the repository, it is valid, otherwise it isn't.
- For **historical queries**, replace the timestamp services that have evolved with some kind of system based on authority records.

It's important to note that all of these changes can be implemented without changing the basics of X.509 certificate formats, the S/MIME email encryption standard, or a multitude of other technology. Most of the changes can be implemented simply by changing the way that certificate Subjects are represented, how the certificates are stored, and by abolishing the existing revocation services.

6.4 Fundamental Problems with PKI

In 1996, Davis and others asserted that PKI was simply too complex or too under developed to be deployed to end users.[Dav96] Davis argued that the problem of certifying root keys for CAs would be a fundamental stumbling block in deploying PKI. Even if the root keys (aka “root anchors”) were distributed with software, he argued, users would still need to manually verify those roots and verify them as being correct, so that they could ensure that the security chain was unbroken. Otherwise there would be no way to trust the trust infrastructure.

In fact, the reverse seems to have happened. Roots were distributed with consumer software, but consumers did not verify the roots. In fact, consumers seem to be generally uninformed that the roots even exist! Much of the entire trust infrastructure distributed with software is simply invisible to most users. This is fine as long as things work properly. Frequently, it does not. As evidenced by the E-Soft survey (Section 6.2.7), it is a common experience on the Internet today to reach an SSL-protected web site that does not have a valid certificate. Users sensibly respond by ignoring these messages. (e.g., Figure 6-11) Today it is a relatively rare happenstance that the PKI that we have deployed actually protects a user from a spoofing attack.

If PKI ever successful, then the day will come that PKI warnings have largely been eliminated except for the occasional attack. Otherwise there is no reason to have the warnings at all. But even if all of the bugs could be magically worked out, PKI itself would still not “work” as one might wish. This is because there are fundamental problems with the X.500/X.509 approach of having names that on public key certificates that represent the legal names of specific individuals. Those problems are:

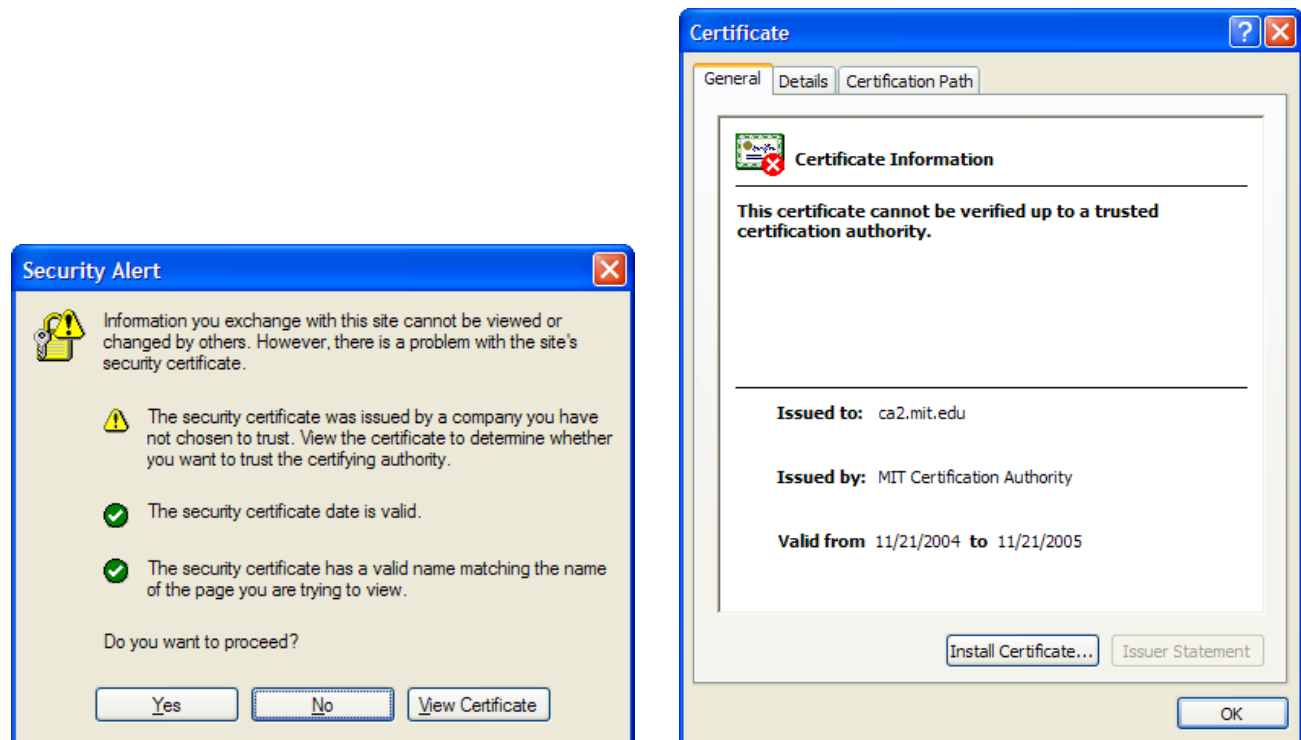


Figure 6-11: On the Internet today, when an SSL warning appears in a web browser, almost invariably the errors are the result of a configuration problem or the failure to have a registered CA. Spoofing attacks are almost never the source of these errors. This figure shows the error generated by Microsoft Internet Explorer at MIT when the user attempts to visit the MIT Certification Authority web site.

- Names are not unique (the John Wilson Problem).
- Revocation appears to be an insolvable problem.
- The correct number of Public Files is not determinable.
- X.509 Distinguished Names are too complicated for people to understand.
- X.509 Distinguished names aren't distinguished.
- The IETF Standard for the format of ARPA Internet text messages (RFC 822) allows comments in email addresses that can conflict with X.509 Distinguished Names.

To understanding these failings, this section will analyze each in turn with respect to three possible usage scenarios:

 
Alice

Alice wants to contact the web site of Kovagis, a company that she has never done business with before, which has obtained a shipment of super-cool motorcycles. Alice wants to reserve a bike but Kovagis will only accept payment by bank transfer. Alice wants to be sure that is at the right company before she gives them her bank information.



Bob



Bob wants to send a message to Alice that is digitally signed with his key and sealed with her's, so that nobody else can read it. Bob met Alice at CHI2005 but he neglected to get her business card; he knows that she works at MIT and he wants to propose to her, so it is very important that no one else intercept the message or know that it is from him. Unknown to Bob, Alice has co-workers who are both malicious and nosy.



Catherine



Catherine was 8 when Grandmother Goldenbucks died in June 2006. Grandma, an early employee at Microsoft, used a digital signature to sign her will a will that which left a ton of money to Catherine. In 2016 Catherine turns 18 and is surprised to learn that the terms of Grandma's will are now being challenged by other members of the Goldenbucks family.

PKI comes with significant costs to the users and to society—costs both in the fees that must be paid to the CAs, the cost of running the CAs, and the inability of users to create and manage their own keys without a CA's blessing. If PKI can't protect against these kinds of targeted attacks, then it might be time to reevaluate the decision to use PKI in the first place.

6.4.1 Names are not unique (the John Wilson Problem)

Although cryptographically secure keys are unique by definition, names are not. When looking up a name in the Public File, there is no guarantee that there will only be one key with that name in the database. Ellison identified this failing as the “John Wilson Problem,” named after a co-worker at Intel who had the same name as seven other Intel employees.[Ell02]

Intel's IT department attempted to disambiguate the John Wilsons by forcing the individuals to use their middle initials in their email addresses and on their certificates (each of the John Wilsons fortunately had different initials). Ellison writes that this attempt to solve the John Wilson Problem was not successful because those middle initials were not significant to people outside of Intel. As a result, each John Wilson routinely received paper and electronic mail destined for another.

On one occasion, two John Wilsons were ticketed on the same flight from the Seattle International Airport to Portland. As it turns out, they were given each other's tickets and boarding passes! This was a significant problem because one John Wilson was continuing on a second flight to Eugene while the other had Portland as his terminal destination. “The solution was for John to go to the gate and have them page John Wilson—and then, when the other John Wilson appeared, trade boarding passes.” [Ell02, p.168] When Ellison tells this story in person, he stresses the fact that the incident happened after the heightened security measures following the 9/11 terrorist attacks were implemented. Ellison notes that the airline simply did not have enough information in its reservation database to disambiguate between the two individuals.

Gutmann observes that the practical impact of the John Wilson problem is that X.509-based sys-

tems generally “turn a key distribution problem into an equally intractable name distribution problem.”[Gut02b, p.4]

Let’s see how Alice, Bob and Catherine handle the John Wilson Problem:



Alice

Alice is very concerned about the John Wilson Problem: how does she know that the Kovagis web site that she has reached is the correct Kovagis Corporation? She knows that Kovagis is the name of the company, but what if some group of hackers filed a “Doing Business As” form with the City of Cambridge and obtained their own certificate for Kovagis. (Mazières reports that a DBA from Cambridge is all that was required for obtaining a certificate from VeriSign.[Maz00]) Since Alice has never done business with Kovagis, nor have any of her friends, she is out of luck. Of course, if they had done business with Kovagis, Alice could rely on her friends’ certification—she wouldn’t need the certification by an allegedly trusted third party.



Bob

If Alice is the only Alice at MIT, then Bob is fine: he can just download her certificate from the MIT certificate server and send her an email. On the other hand, if there is more than one Alice in the directory, then Bob has a problem, for has no obvious way to distinguish between them. He could of course send email to each one and ask if she is the Alice that he met at CHI2005, but if Alice’s evil co-worker Alice B. gets the email, she will probably bluff back in an attempt to get Bob to spill the beans.



Catherine

Catherine’s evil relatives are claiming that the Grandmother Goldenbuck who signed the digital will wasn’t really the Goldenbuck who worked at Microsoft, but the woman’s sister who lived in Kirkland and never really amounted to much. (They had a wicked father who gave all of his daughters the same first name but different middle initials. Alas, both daughters resented their middle initials and stopped using them.) At this point, Catherine and her evil relatives all need to go to court and resolve the validity of the will through traditional fact-finding techniques such as testimony and the examination of other contemporaneous documents.

An alternative solution, notes Gutmann, is to use public keys themselves as global identifiers, since they are unique, and to allow users or local administrators to create their own “local identifiers” to describe who the keys actually belong to. This is the approach used by both PGP and SPKI/SDSI. Sadly, this solution doesn’t work for Alice, Bob or Catherine. They all need the ability to a PKI-based solution to provide high assurance for an initial, unmediated transaction. PKI lets them

down. Indeed, the problem may not be solvable, and may instead represent a risk that is best managed through the use of insurance, rather than cryptography.

6.4.2 Proper revocation appears to be unsolvable

A certificate can be thought of as a local copy of a remote datum. By replicating and distributing certificates throughout a system, it is possible to reduce the load on the Public File and to achieve high availability in the event of a network partition.

Kohnfelder himself noted that the mere existence of certificates creates the need for some kind of revocation procedure—for example, when the private key is lost or compromised. “The problem is similar to that of preventing misuse of lost credit cards,” he notes. [Koh78, p.42] He proposes three solutions to the problem: flooding the system with a list of certificates that are no longer valid—what we now call a Certificate Revocation List (CRL); putting an expiration date on certificates, so that old certificates eventually expire out; and real-time verification of a certificate’s freshness should a communicant suspect that an enemy has stolen the certificate holder’s private key.

But revocation cannot be made reliable in all cases. For example, if there are multiple revocation servers, then there is a chance for them to be out-of-sync with each other, making it possible for one server to say that a certificate is valid while another says that it is not. On the other hand, if there is only a single revocation server, then an attacker who has compromised a certificate can continue to use the compromised certificate by mounting a denial-of-service attack against the revocation server and taking it offline. Relying parties are then forced to choose between accepting a certificate that might possibly be bad, and rejecting all other certificates—at least some of which would almost certainly be good. This is an example of Fox and Brewer’s CAP Principle [FB99, Bre00], which holds that it is not possible to build network systems that are simultaneously are *Consistent*, *Available*, and *Partition-tolerant*. Ellison notes that the Public File achieves Consistency and Partition-tolerance at a loss of Availability, while certificates achieve Availability and Partition-tolerance at the cost of Consistency. [Ell02]

VeriSign, one of the world’s leading Public Files in 2005, lists four circumstances in which users should revoke their keys:

- if the customer loses their private key;
- if the private key is compromised;
- if incorrect information appears on the certificate;
- if the certificate is not working properly. [Ver05c]

By contrast, the X.509v4 draft standard specifies 10 reasons that can be given for revoking a key, as shown in Figure 6.3.

Depending on the use of the certificates and the threat model, a revocation service may need to devote considerable attention to the issue of time and sequence management. If a certificate used for so-called “non-repudiation” purposes (e.g., signing contracts) needs to be revoked because of key compromise, then the following information might need to be recorded:

T_1 the time of the compromise;

T_2 the time that the compromise was detected;

T_3 the time that the compromise was reported to the CA; and

T_4 the time that the CA made the information available on its CRL.

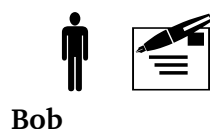
Each of these times is critical because they could conceivably have different implications for a relying party that had been given a contract signed by the attacker. The owner's insurance company might be liable for abuse that occurred between T_1 and T_2 because no one was at fault (other than the attacker); the owner might be liable for abuse that occurred between T_2 and T_3 because he or she had not reported the compromise in a timely manner; and the CA might be liable for a compromise that occurred between T_3 and T_4 . Certainly, if these times are not recorded in the CRL itself, they would almost certainly need to be reconstructed in the event of litigation.

Let's see how the revocation issue affects Alice, Bob and Catherine:



Alice

Under most circumstances, revocation doesn't impact on Alice. Since Kovagis supplies its public key as part of the SSL protocol, Alice won't ever experience the old, revoked certificate. The only potential for harm would be if an attacker both compromised the Kovagis private key *and* was able to divert the `kovagis.com` DNS entry to the hacker's own web site. Most of today's web browsers do not have revocation checking enabled, perhaps evidence that today's e-commerce community does not consider proper revocation to be very important.



Bob

Revocation is somewhat important for Bob. If Bob's private key was compromised, it will only matter if that attacker is sending out messages that claim to be from Bob. If Alice's old key was compromised, the attacker will still need to be wiretapping her email in order to be able to intercept and decipher Bob's message. (Of course, if the attacker is not wiretapping, then there is considerably less reason for Bob to seal his message in the first place.) More likely, Alice lost her key. If Bob uses the old certificate, then Alice will send back an email that she can't understand it and Bob will send a new message sealed with Alice's new certificate.

CRLReason	note
unspecified	
keyCompromise	Used in revoking an end-entity certificate; it indicates that it is known or suspected that the subject's private key, or other aspects of the subject validated in the certificate, have been compromised;
cACompromise	Used in revoking a CA-certificate; it indicates that it is known or suspected that the subject's private key, or other aspects of the subject validated in the certificate, have been compromised;
affiliationChanged	Indicates that the subject's name or other information in the certificate has been modified but there is no cause to suspect that the private key has been compromised;
superseded	superseded indicates that the certificate has been superseded but there is no cause to suspect that the private key has been compromised;
cessationOfOperation	Indicates that the certificate is no longer needed for the purpose for which it was issued but there is no cause to suspect that the private key has been compromised;
certificateHold	<i>not specified in the standard.</i>
removeFromCRL	<i>not specified in the standard.</i>
privilegeWithdrawn	Indicates that a certificate (public-key or attribute certificate) was revoked because a privilege contained within that certificate has been withdrawn;
aaCompromise	Indicates that it is known or suspected that aspects of the AA validated in the attribute certificate have been compromised.

Table 6.3: The draft X.509v4 allows CAs to specify 10 reasons why a certificate might be compromised.



Catherine



Revocation is a big deal for Catherine. Trying to make a legal claim 10 years after the will was signed, Catherine needs to be able to prove in court that Grandmother's certificate was valid when the document was signed. But she can't, because VeriSign and Thawte (and possibly other CAs) are removing certificates from their CRLs when those certificates expire. Thus, there is no way to know if the certificate had or had not been revoked when it was used. An added problem for Catherine is that the 1024-bit CA key that was used to sign Grandma's certificate may itself be compromised by the year 2016, given faster processing speed of the computers available then. So even if the will is deemed valid today, at some point in the future it will not be possible to rely on the will's cryptographic protection to prove its validity. Future courts will instead need to rely on the determination of older courts.

6.4.3 A single "public file" cannot be both logically consistent and correct; multiple public files cannot be authoritative.

In perhaps the single most prescient paragraph of his entire thesis, Kohnfelder wrote:

“The Public File solves many problems, but it is also a great potential threat to system security. An enemy that had broken the Public File encryption function could authoritatively pass out bogus encryption functions and thereby impersonate any communicant in the system. Even without breaking the Public File encryption function such impersonation is possible by tampering with the records kept by the Public File. The Public File could not work in high security applications since it would not be trusted. Consider a Public File coordinating all diplomatic communications in the world; who could reliably operate such an authority?” [Koh78, p.16]

To put this in the language of today, one of the fundamental problems with certification authorities is that a single root would be unacceptable to the multitude of nations and other political entities that inhabit our planet.

But if there is more than one CA, the result is the Gutmann’s “Which Directory?” problem[Gut04a]: when there are multiple CAs or directories, how do you know which is the correct directory to consult? What if the CAs contain inconsistent or contradictory information? If a person’s key doesn’t exist in any of the directories that you have consulted, how do you know that you have not inadvertently failed to consult the correct directories?

Kaufman *et al.* argue that is logically impermissible to use a list of multiple directories in an attempt to get around the policy problems inherent in a single directory. The problem is that this comprehensive list of directories and the directories themselves can now be viewed as a single directory—a directory that either requires coordination between the subdirectories, or else allows there to be contradictory mappings.[KPS02]

For example, it is completely possible using today’s X.509 technology for an organization such as Dun & Bradstreet to issue certificates to corporations that certify the corporation’s identity using a combination of their stock symbol, their state and their country. For example:

$$\text{cert}_1 = \{\text{MSFT/WA/US}\}_{\mathbf{D\&B}_1}$$

(Once again, what is in **bold** is visible to the user, while the information that is not in bold is included merely for the convenience of the reader.)

But unless there is some way for a user to be sure that they have the correct Dun & Bradstreet root certificate, an attacker who wanted to impersonate Microsoft could trivially create a new D&B certificate and use that certificate to sign a fraudulent MSFT certificate:

$$\text{cert}_2 = \{\text{MSFT/WA/US}\}_{\mathbf{D\&B}_2}$$

Just as the certificates **cert**₁ and **cert**₂ are visually indistinguishable unless the user looks at the certificate fingerprints, so too are the signing certificates **D&B**₁ and **D&B**₂. In today’s world, the way that **D&B**₁ and **D&B**₂ would be distinguished is that one of these certificates would be present on D&B’s web site, <http://www.dnb.com/>, while the other one—the attacker’s—would not be. Es-

entially, this means that the Domain Name System has become the official list of CAs: it arbitrates which is the official Dun & Bradstreet, and which is the imposter/attacker.

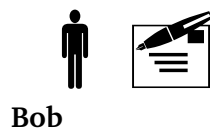
All hierarchies with globally meaningful names share this problem. Logically, such an entity cannot be both comprehensive and correct. The legal battles over Domain Name System names in the 1990s were largely the result of this fact. Today organizations have come to accept the fact that they may not be able to have globally unique DNS names that are both meaningful and that reflect the names that the corporations normally use to refer to themselves—because the domain name may, in fact, be owned by another organization. The result is that today’s DNS is neither comprehensive nor correct: MIT was clever and secured the domain name `mit.com` in addition to `mit.edu` and `mit.net`, but Harvard didn’t grab `harvard.com`, and as a result `harvard.com` now belongs to the Harvard Book Store—and `harvard.net` belongs to Allegiance Telecom in Dallas, Texas.

Some businesses have decided to adapt to these inconsistencies in the Domain Name System by choosing the names of new companies based on the availability of appropriate domain names. They then publicize these names, teaching potential customers the association between the domain name and the product or service being sold. This is similar to using a public key as a global identifier and then giving it local meaning, as is the case with SPKI/SDSI.

So how does the fact that a public file cannot be consistent and correct affect Alice, Bob and Catherine?



Alice doesn’t have a problem with the zero/one/many CA problem, since she knows that the company’s name is Kovagis, and that company name has been explicitly chosen because the word does not appear in Google and the `kovagis.com` domain was available. Even if an attacker manages to get a certificate name that says Kovagis on it, Alice’s transaction will be protected by the DNS. Indeed, the DNS is the main thing that is protecting Alice, not the X.509 certificate on the SSL-enabled web site.



Bob is not sure where to go to get Alice’s certificate. He might search the VeriSign and Thawte web sites, to no avail. It turns out that Alice is using an MIT CSAIL S/MIME key, which means that Bob may be out of luck, since there is no easy way to look up these certificates.



Because there were many CAs when Catherine’s Grandmother obtained her Digital ID, Catherine hopes that Grandma picked a CA that’s still respected—and that the CA’s key can be independently obtained. Otherwise there is no way that the court-appointed cryptographer is going to be able to argue that the will was signed in 2006, and not last week in March 2016.

Certificate Issuer	
Certificate Issuer:	/O=Notesdev/CN=Westford Internet CA
Subject:	/O=IBM/OU=Westford/CN=Mary Ellen Zurko/E=Mary_Ellen_Zurko@notesdev.ibm.com

Figure 6-12: The X.500 Distinguished Name (DN) for Mary Ellen Zurko at IBM, received in an X.509 certificate used to sign an email message on March 22, 2005.



6.4.4 X.509 distinguished names are hard for people to understand



On the surface, there appear to be significant usability problems with X.509 Distinguished Names (DN). Because they are inherently rooted in geography, it is not immediately clear how a DN should be structured for an organization that operates in more than one state or country. One approach is to omit the Country and Locale fields entirely, as shown in Figure 6-12.

A second problem with DNs is that there appears to be no software available for desktop computer users that actually displays certificates and the DNs that they contain in a manner that makes any sense to users. For example, many programs that display certificates don't even bother to explain what the initials like "O" and "OU" actually mean. Whether this is a fundamental result of the X.500 syntax or simply a result of programmer disinterest remains to be seen. (Problems with the display of DNs on X.509 certificates are discussed in Chapter 5.)

Perlman asserts that the distinguished names present on X.509 certificates were never intended to be shown directly to human beings, but that this display was supposed to be mediated by some kind of application program. [Per05b]

As it turns out, this problem is something of a red herring for our three fictional characters:

  Alice is using Mozilla Firefox to view the Kovagis web site. Firefox decodes the web site's certificate and displays the name in the browser's status bar in a form that's easy to read. Alice doesn't see this, because she's looking at the motorcycle photos.

  Bob's copy of Outlook Express shows him the distinguished name on Alice's certificate. It's ugly but he's able to figure it out. He's a bit confused why it says that Alice's Organization is the MIT Computer Science & Artificial Intelligence Laboratory, yet the certificate itself was issued by the MIT Laboratory for Computer Science (as is the case with the client-side certificates that CSAIL is currently issuing).



Catherine



Catherine's copy of Word 2016 decodes the certificate on her grandmother's signed will. Ten years in the future, Microsoft's engineers have finally figured out how to display distinguished names in a manner that makes sense, so Catherine is happy.

6.4.5 Distinguished names aren't distinguished

Another problem with Distinguished Names is that they do not appear to be distinguished. In many cases, they aren't even valid! Gutmann discusses this issue at length in his *X.509 Style Guide*[Gut00], in which he gives dozens of examples showing the misuse of distinguished names. Entrust, for example, placed a liability statement in the issuer DN of its certificates, making that field unusable with X.500 or LDAP directories. Other certificates have DNs with zero length. Another CA chose a non-standard character set for names, and then placed characters in the name field that were illegal for that non-standard character set.

ISO character sets can cause problems for another reason as well: font encoding makes it is possible for different DNs to appear visually indistinguishable on the computer's screen. For example, it is possible to construct names that use accents to create normal-looking unaccented characters. A dotless “ı” and a dot “i” can be merged together appear to resemble a lower-case “i,” but in fact they are not, as a close examination of the third quoted glyph in this paragraph reveals. (The dot and the ı in the previous sentence were intentionally offset by a point to make it easier to see that they are in fact different glyphs. The offset could have been made invisibly small, had the author wished.)

This problem, termed a *homograph attack*, has also been observed in relation to the Internet's recent adoption of international domain names. [The05a] As a result of this attack, The Mozilla Foundation decided to disable support for IDNs in Firefox 1.0.1 and Mozilla 1.8 beta [Mar05a]; Apple issued a security update to remove the display of look-alike characters from URLs displayed in the Safari web browser. [Com05a] Microsoft did not have to issue a patch for Internet Explorer because it did not support international domain names.

Do the problems with Distinguished Names affect our fictional characters? Not really:





Alice





Alice cares a lot about the homographic attack. Has she updated her copy of Firefox? Is that an “ı” in the kovagis.com domain, or is there something funny happening with the Unicode? Alice doesn't remember if she typed the domain name herself, or if she clicked on a link that was in an email message she saw. Flustered, she closes her browser. Then she realizes that she doesn't remember how Kovagis.com spelled its name...

_____ RFC822-style mailbox comments from RFC822: _____
Alfred Neuman <Neuman@BBN-TENEXA> "George, Ted" <Shared@Group.Arpanet> Wilt . (the Stilt) Chamberlain@NBA.US
_____ Misleading RFC822 mailbox comments (not from RFC822): _____
George Washington <Neuman@BBN-TENEXA> "Chamberlain@NBA.US" <attacker@nowhere.org>

Figure 6-13: The top box shows examples of RFC822 “comments” in mailboxes, taken from [Cro82a]. Many X.509 systems allow mailbox names with comments to be included in fields that expect distinguished names. The bottom box shows how these comments can be used to create misleading addresses. The RFC822 mailbox name comment is typically not certified by the Certificate Authority but is nevertheless displayed to the user, creating a potential for spoofing the user with uncertified information.







 
Bob Bob isn't worried about the homographic attacks because he knows that MIT CSAIL, being a small organization, probably has better control over the certificates that it is issuing than a large organization like Thawte does.

 
Catherine The homographic attack doesn't affect Catherine: the court-appointed expert can look at the raw Unicode bytecode and verify that it's plain ASCII. On the other hand, the expert does notice that Grandma put a URL in the DN field that looks like a web page that no longer exists. This is noted in the report, but deemed to be insignificant.

6.4.6 RFC822 comments can conflict with X.509 Distinguished Names

Yet another problem with the X.509 system has been the expansion of distinguished names to incorporate RFC822-style email addresses. The problem here is that RFC822 allows email addresses to include a so-called comment—usually the human-readable name associated with an email address. According to the standard, this comment is to be “ignored by the mailer.”[Cro82a, §A.1.2, p.36] Nevertheless, most mail clients commonly display this comment to the user.

The problem here is that the comments in the RFC822 names may contain information that conflicts with the distinguished name on the certificate. Examples of comments from the standard and how these comments can be used to create misleading names are shown in Table 6-13.

-  
Alice
- The email address issue does not affect SSL certificates, so Alice isn't affected.
-  
Bob
- After Bob and Alice exchange email for several days, Alice's co-worker Evelyn gets a hotmail account `alice-at-home32@hotmail.com` and obtains a digitally signed certificate with the RFC822 name `"alice@csail.mit.edu" <alice-at-home32@hotmail.com>`. Evelyn sends email to Bob with the new certificate and he replies. Thinking that the mail is going to `alice@csail.mit.edu` because that is the name on the certificate, the highly personal message actually goes to Evelyn's Hotmail account.
-  
Catherine
- Grandma's certificate was used for signing, not S/MIME, so this attack isn't relevant.

6.4.7 Summary: which attacks matter to whom?

Table 6.4 shows that each persona is affected by two or more of the problems and resulting possible attacks with the X.509-based system, but that no single attack effects all of the scenarios. This is more evidence that a one-size fits all PKI model is probably not the right choice for deploying this technology on a massive scale.




	 Alice	 Bob	 Catherine
Names are not unique	big problem	big problem	big problem
Proper revocation is unsolvable	not important	not important	big problem
Can't have one public file; can't have many	not important	important	very important
X.509 Distinguished Names are hard to understand	no problem	no problem	no problem
Distinguished names aren't distinguished	a problem	no problem	no problem
RFC822 comments can conflict with X.509 Distinguished Names	not applicable	big problem	not applicable

Table 6.4: How each potential problem with X.509-based PKIs affects each scenario.

6.5 Making PKI Usable

This section explores the three approaches that have been tried so far to make PKI systems usable: complete mediation, education, and mathematical innovation.

6.5.1 Complete mediation: HushMail and the “signing fool”

As discussed in Chapter 5, one approach to making PKI easier to use is to provide the user with a completely mediated environment. This is the approach taken by HushMail, a web-based secure mail provider. All of HushMail’s encryption and PKI operations take place inside an Java applet that is downloaded into the user’s web browser. Although the user’s private and public keys are both kept on the HushMail server, they are kept encrypted, and only decrypted inside the Java applet. Indeed, Hush Communications tells users that if they irretrievably lose their passwords, their only alternative is to open a new account. While HushMail doesn’t provide a comprehensive PKI solution, it does provide an easy-to-use secure email system that can interoperate with people using PGP on the open Internet.

Another example of the complete mediation approach is the so-called “Signing Fool”—a program that automatically signs all outgoing messages with the user’s private key, without regard to the message content. (The Stream proxy is an example of such a program.)

Even though automatic signatures do not demonstrate intentionality—and thus may fall-short of standards required for non-repudiation under certain legal regimes—they still have value. For example, a “Signing Fool” makes it easier to track down the source of a computer that is infected with a worm, since the messages sent with forged `from:` address are signed with the key of the sending machine, not with the key of the person whose address is being forged.

6.5.2 Education: Whitten’s “Lime”

Whitten has developed a system called Lime to demonstrate and test her theories regarding safe staging and metaphor tailoring. Lime only uses safe staging for key certification, and not for the basic functions of sending and receiving cryptographically protected mail. The reason, says Whitten, is that interfaces for signing and sealing can be presented in a clear and usable way without staging. “Key certification, by contrast, is extremely confusing and difficult to present clearly, and does not need to be visible in the main window since we expect that it will be used only when public keys are traded.” [Whi04a, p.30]

Thus, if Whitten were to answer the question posed in her 1998 paper with the research from her 2004 dissertation, it would seem that Johnny can’t encrypt because he doesn’t know how to properly certify keys. Lime overcomes this barrier by presenting its user with a series of six help screens that teach Johnny how to do it. (The title of Whitten’s 2004 DIMACS talk on Lime was actually titled “Giving Johnny the Keys.”) Table 6-14 in this thesis presents my summary of Whitten’s screens; Figure 6-15 shows Whitten’s sixth help screen. In addition to these help screens, additional help appears on the program’s windows, as shown in Figure 6-16.

Although Lime was a “Wizard-of-Oz” prototype, constructed solely for the purpose of conducting a user test, Whitten reports that the safe staging in Lime was able to increase participant success from 0% and 10% to 45% in her key certification experiment.[Whi04a, p.iv]

Help Screen	Purpose
1	Initialization sequence
2	Generate a key pair
3	Store the key pair securely with a pass phrase
4 & 5	Introduces “digital signatures” and “encryption”
6	A “quick briefing on the need to trade public keys.”

Figure 6-14: The six help screens presented in Whitten’s “Lyme” prototype. [Whi04a]

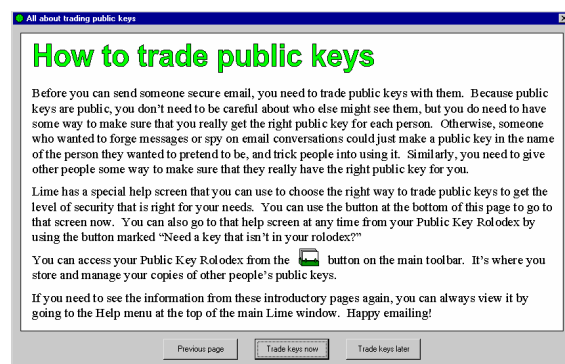


Figure 6-15: Whitten help screen #6 explains to the user how how public keys are traded. Used with permission.

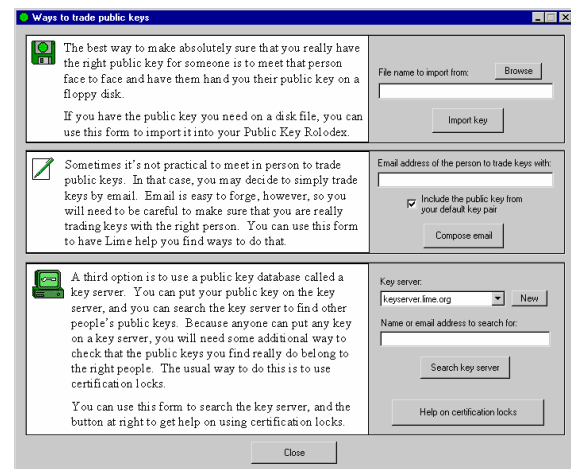


Figure 6-16: Whitten's panel for trading keys reiterates much of the information on the help screen and provides users with the necessary controls to accomplish their tasks.

6.5.3 Change the math: Identity Based Encryption

A third approach to making PKI easier to use is to change the underlying mathematics on which today's public key systems are based.

For example, Identity Based Encryption schemes, proposed by Shamir [Sha85] and recently summarized by Bellare *et al.* [BNN04a], overcome the fundamental usability problem of public key cryptography that requires a user to create a public key before they can receive an encrypted message. These schemes typically rely on a Trusted Third Party which creates a set of parameters for a particular instantiation of the cryptosystem. The TTP maintains a set of private parameters, which must remain secret, and a set of public parameters, which need to be published to all system users. Public keys for participants in the IBE system can be created using the cryptographic hash of a unique identifier, such as an email address, and the public parameters. Private keys can be created using the same hash and the private parameters. Thus, any user in the system can send cryptographically sealed email to any participant in the system, even if the recipient hasn't yet registered! (Security of the message is not compromised because the TTP never receives the encrypted message: it goes straight to the intended recipient.) Once a recipient has received the sealed message, that recipient contacts the Trusted Third Party to obtain their private key necessary to unseal the message. Adida have proposed that the centralization of trust can be mitigated by using different Trusted Third Parties for each mail domain and using the DNS to distribute the public parameters

for each domain.[AHR05a, AHR05b]

The primary advantage of IBE is that a system participant can make a public key without having to contact the Trusted Third Party. This allows, for example, Alice to create Bob's public key while she is on an ocean liner and unable to communicate with either Bob or the TTP (provided that Alice has previously obtained the public system parameters.) Whether this one neat trick is sufficient to justify the deployment of completely new algorithms, new data formats, and new standards remains to be seen. A hybrid system might use the Boneh-Franklin scheme to distribute conventional S/MIME certificates made by the TTP.

The Boneh-Franklin Identity Based Encryption [BF01] system is being commercialized by Voltage, a Palo Alto, California-based company created by the inventors. Although Voltage originally developed and deployed a system for sending and receiving encrypted mail, the company's current direction appears to be the development of secure messaging systems that exist apart from the email infrastructure—for example, systems for providing document-level cryptographic protection.

6.5.4 Missing: a sense of certificate history

Missing from *all* of the implementations is a sense of certificate history. In each of the systems considered above, certificates have the same level of trust the first time that they are received and the thousandth time that they are received: they are either trusted or they are not trusted.

This is not the way that other kinds of human relationships work. Although the path of introduction matters in most relationships, as relationships progress humans tend to invest more credibility in a person's actions than in their pedigree. This is especially true in modern societies, as evidenced by the fact that more people choose their mates as a result of several person-to-person interactions—dating—rather than through arranged marriages.

Certainly it is possible for an individual to misrepresent herself in relationships, gain trust, and then to betray that trust in a high-valued attack. Mitnick outlines many such attacks. [MS02] The high divorce rate in the US is perhaps indicative of others. But it is unlikely that interfaces which do not present history information make users less susceptible to these attacks than interfaces that do: even if the interface does not present history information directly, that information is available indirectly in the computer system by consulting sources such as logfiles or old email messages. Human beings also retain information inside their brains. It would seem reasonable, then, to design a certification model that uses historical information, rather than one that ignores it.

6.5.5 Conclusion

Technical innovation favors solutions that can be deployed incrementally or “bottom-up.” By keeping costs low, small-scale initial deployments make it possible for engineers and early adopters to tinker with systems and make subtle modifications, allowing the technology to evolve from a laboratory curio into a tool that performs a valuable function. Many people attribute the success of personal computers in the 1980s to their “bottom-up” appeal: the computers cost so little that they could be purchased with discretionary funding, effectively avoiding the centralized control of information processing that gripped American businesses and many universities in the 1970s.

A hallmark of successful incremental solutions is that they tend to follow the 80-20 Principle,[Jur05]

solving some problems well and leaving others unaddressed. This is sometimes seen as a feature, as the partial success creates an incentive to invest more time, energy and money into completing the solution. (Of course, the 80-20 Principle also implies that finishing the last 20% of the work requires 80% of the funding.)

The data from E-Soft shows that when merchants are sufficiently motivated, some of them will try to obtain the certificates and some of these certificates may even end up being properly installed. But likewise, the relatively small number of Thawte S/MIME certificates that have been issued implies that without a compelling motivation, most users will simply not go to similar trouble of obtaining a certificate—even if the certificate is monetarily free and can be acquired with less than an hour's effort.

In this chapter we have explored how the deployment of PKI systematically avoided opportunities for incremental deployment, resulting in a system that today is used by some corporations and not very many individuals. In Chapter 7 we will see an alternative to this conundrum: self-made and self-signed certificates that are certified not through third parties, but through their continued use in ongoing relationships.