Chapter 17

X.509 PUBLIC KEY INFRASTRUCTURE

Now that you've seen how digital certificates help with the delivery of public keys, we'll examine two ways a network of digital certificates can be managed. This chapter and Chapter 18 look at digital certificate administrative frameworks, which are called *public key infrastructures* (PKI).[1] The two major PKI frameworks are X.509 and Pretty Good Privacy (PGP).

There's more discussion here on X.509 because it seems to fit the corporate/government model, and, as a result, most of the effort (and money) seems to be coalescing around X.509. Nevertheless, PGP has some attractive attributes.[2] PGP is discussed and contrasted with X.509 in Chapter 18.

Although there are several standardization bodies,[3] most of the work on X.509 and other Internet standards seems to come from an Internet Engineering Task Force (IETF) subcommittee, called the PKIX working group. The PKIX group promotes, enhances, and develops methods to administer X.509 certificates.[4]

[1] PKI is a generic term; perhaps a more descriptive name would have been *digital certificate infrastructures*. Don't confuse PKI with another widely used public key acronym, PKCS (public key cryptographic standards). PKCS are suggested guidelines published and owned by one company, RSA Data Securities. PKCS #1 is discussed at the end of Chapter 22.

[2] Efforts are under way to make X.509 and PGP interoperate.

[3] For example, the International Organization for Standardization (ISO), the International Electrotechnical Commission (IEC), the International Telecommunication Union (ITU), NIST, and the American National Standards Institute (ANSI).

[4] X.509 was originally an ITU recommendation. PKIX's goal is to further specify X.509 for use with Internet applications, such as e-mail, SSL, and IPsec.

About IETF

The Internet Engineering Task Force is a group of people and companies interested in the harmonious operation of the Internet. Anyone can download information about IETF standards from its site, www.ietf.org, without charge.

IETF members (and their publications) are usually very technical, as illustrated by their suggested dress code: "There are those in the IETF who refuse to wear anything other than suits. Fortunately, they are well known (for other reasons) so they are forgiven this particular idiosyncrasy."

Why Use X.509 Certificate Management?

In Chapter 16 we outlined the needs of a typical digital certificate user, and they are reviewed in Table 17-1. Briefly, there are two types of digital certificate users: a subject and a consumer. A subject needs to get his or her own public key certified and posted to Internet repositories. A consumer needs to retrieve and use digital certificates and know whether a digital certificate is still trustworthy.

Most digital certificate users don't want any certificate administrative responsibilities.

But most digital certificate users probably don't want any responsibilities for administering digital certificates and probably don't have the knowledge or resources (such as a valid source of random data) that are required to generate strong public/private key pairs. Most digital certificate users simply want PKI to work with as little effort as possible on their part. Enter the manager.

To give certificate users a transparent digital certificate solution, digital certificate managers want a centrally controlled system or, in X.509-PKI parlance, a *certificate authority* (CA).

Table 17-1 Digital certificate user's needs reviewed.

What a Digital Certificate User Wants
Public key certified by a trusted third party
Retrieve other users' certified digital certificates (public keys)
Notified if a trusted certificate has been prematurely revoked

What Is a Certificate Authority?

A certificate authority manages application, certification, issuance, and revocation.

A CA manages digital certificate application, certification (authentication of the applicant), issuance, and revocation. In this respect a CA is similar to the DMV, a driver's license authority (see Chapter 16). A CA is also similar to a key distribution center (KDC) introduced in Chapter 8. Both KDCs and CAs act as trusted third parties.

A notable difference between a KDC and a CA is the type of key with which each is entrusted. Users of a KDC trust the KDC with their secret keys and trust that the KDC keeps their secret keys secret. Because public keys don't need to be secret, a CA doesn't keep any secrets, except, of course, its own private key.

This chapter examines some of the more important CA responsibilities.[5] Although a CA can outsource most of its responsibilities, for simplicity we assume that it handles all the management functions. When that's not the case, we'll specify.

Application, Certification, and Issuance

CA authenticates the applicant and ensures that the applicant has the matching private key.

A CA should validate an applicant's identity before issuing a digital certificate. Suppose our applicant is Bob. Once the CA has identified Bob, the CA must verify that Bob has the matching private key before issuing him a digital certificate. The digital certificate is like a traditional letter of introduction; it effectively says, "I, the CA, hereby identify Bob and assert that this is really Bob's public key." Most CAs offer various classes of digital certificates depending on the certifying documents the applicant submits and the fee paid to the CA.

Low Assurance Certificates

In Figure 17-1, Bob applies for a digital certificate over the Internet under the applicant name bob@hxMel.com. The CA issues and sends a digital certificate to the e-mail address bob@hxMel.com.

Low assurance means that the CA has not validated the applicant in person.

Because the CA doesn't know whether the owner of the e-mail account bob@hxMel.com is really Bob, this certificate is a *low assurance* digital certificate. It is so named because, for example, Bob may have stepped away from the computer long enough for BlackHat to apply for and get the digital certificate using Bob's e-mail account. Or perhaps Bob uses a system (such as Windows 98) that doesn't restrict logon and BlackHat applied to the CA without Bob's knowledge.

[5] The IETF has suggested CA standards. See www.ietf.org.

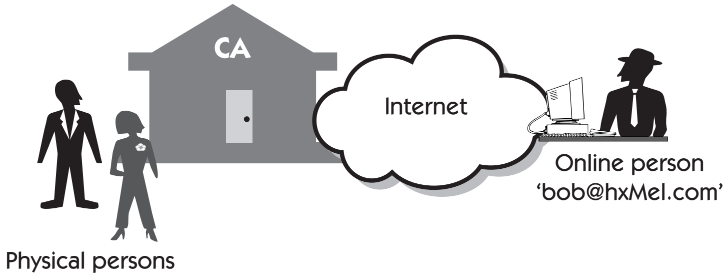


Figure 17-1 Bob applies to a CA over the Internet for a low assurance certificate. A high assurance certificate requires that Bob apply in person.

Most CAs ask for a credit card and issue the low assurance certificate to the name on the card.

For further assurance, the CA can request that the applicant pay for the certificate with a credit card and then issue the certificate to the exact name that appears on the card. If BlackHat also knows Bob's credit card number, he could successfully impersonate Bob.

For a low assurance certificate the CA may only validate whoever can get the e-mail at bob@hxMel.com. The CA is confident that's true because the certificate is sent to bob@hxMel.com. If Alice gets the e-mailed certificate, it's OK with the CA because the CA has validated only a subject who can receive mail at bob@hxMel.com. The CA is assured that the applicant at bob@hxMel.com has the matching private key after successfully completing something like a challenge response.[6] (A short review of a challenge and response protocol is in Figure 17-10 at the end of this chapter.)

Low assurance digital certificates from a commercial CA such as VeriSign are called a "Class 1 Individual Subscriber Persona Not Validated" and give only "assurances that communications originate from a particular source. Class 1 Certificates **do not provide proof of identity**."[7] The boldface type, added by VeriSign, emphasizes that the company did not authenticate the certificate owner in person.

Low assurance certificates may use the same high-quality cryptography as high assurance.

The use of low assurance certificates does not mean that the CA uses less secure cryptography. Both low and high assurance certificates can use the same cryptographic methods and key lengths.

High Assurance Certificates

For a *high assurance* digital certificate, the CA most likely will require the applicant to appear in person and present more than one form of identification,

[6] A few CAs issue digital certificates only if they can create the public/private key pair.

[7] PKI Disclosure Statement from www.verisign.com.

High assurance means that the CA should validate the applicant in person and verify the applicant in other ways. High assurance certificates are more costly.

such as a driver's license or a passport, so that the CA can match a certified picture to the applicant. Of course, this procedure doesn't prevent forgery, but it does make it more difficult. CAs can implement other user authentication practices to preclude almost any imagined forgery. For example, before a CA issues a certificate for an e-commerce company, the CA may verify the company phone numbers with independent commercial databases, phone the company executive offices, and confirm the application. High assurance certificates cost more than low assurance certificates.

A CA may outsource the registration and authentication tasks to a local registration authority (RA).[8] For example, an RA with many offices can save HxMel.com the considerable expense of sending each of its 1,000 geographically dispersed employees to a CA. Instead, a CA-approved RA can verify and issue each of HxMel's employees high assurance digital certificates. The CA might even empower the RA with other responsibilities, such as creating an employee's public/private key pair.

Distribution

Because nothing on a certificate is secret, the CA can store it in a certificate repository and/or send it to the subject (applicant); certificates can be e-mailed.

A CA can decide to outsource digital certificate storage to a dedicated server. Proposed PKIX standards for publishing and retrieving certificates use the Lightweight Directory Access Protocol (LDAP).[9] The CA might also exchange digital certificates with other CAs and even certify other CAs. CA cross certification is still in its infancy, and there are still many logistical problems.

Other CA paying customers

Who might want to pay the CA for a copy of a digital certificate? CAs (and their venture capitalists) believe that anyone who wants someone else's certified public key—for example, users of digital signature schemes that use public key cryptography—will pay a minimal amount for each certificate.

Certificate Revocation

Although a digital certificate is normally expected to be valid during the dates listed on the certificate, a CA can revoke a certificate prematurely.

Revoking a single digital certificate

Suppose that HxMel's CA issues a digital certificate to a temporary contractor, Temp3. When Temp3 completes his work, HxMel's business partners need to know that Temp3's digital certificate has been withdrawn. (Temp3's public key no longer represents HxMel, and his private key cannot be used to sign contracts.)

[8] Sometimes this is referred to as an LRA, or local registration authority.

[9] See the LDAP working group on www.ietf.org.

If Temp3 returns to work, a new digital certificate and public key must be issued. In other words, revoked digital certificates cannot be renewed; that would entail too much administration, potential error, and possible fraud.

What if the CA's private key is compromised?

If a CA's private key is compromised, all certificates issued *after* the private key was compromised should be revoked because a certificate user can't be sure whether the genuine CA or BlackHat used the CA's private key to make and issue certificates.

Definition: certificate revocation list

Revoked certificates are put on a certificate revocation list (CRL). A certificate user or holder should check the most recent CRL just as a merchant validates a credit card before completing a transaction.

Polling and Pushing: Two CRL Delivery Models

Definition: polling

There are two basic CRL delivery models. One model, known as *polling*, requires the certificate user to request the current CRL whenever the user needs a public key on a digital certificate. One problem with polling is the time delay between CA certificate revocation and CA publication of a new CRL. For some applications, even a delay of a few hours is too long.

Definition: pushing

In the other model, known as *pushing*, the CA delivers users new CRLs as soon as it revokes a certificate. A problem with pushing is that the user must store each new pushed CRL even if it doesn't contain any relevant revoked certificates. Also, because the CRL is pushed, the CA and the user must ensure that BlackHat doesn't intercept and delete the pushed CRL before it reaches the user.

Building X.509 Trust Networks

Definition: root CA

Just as each state has its own DMV, each particular X.509 PKI implementation has a *root* CA. For example, any company may have its own CA or, if finer granularity is needed, even a CA for each company division or department. In Figure 17-2, a root CA[10] we'll call Root CA issues Alice a digital certificate. After Alice has been issued a certificate, Bob can get it from either Root CA or Alice. Bob uses his copy of Root CA's public key to verify Alice's certificate and public key (see Chapter 16 for information about the verification process).

Definition: self-signed certificate

Why does Bob trust his copy of Root CA's public key? Root CA has a distinctive digital certificate, issued *by* Root CA and *to* Root CA. In other words, Root CA is both issuer and subject; it's called a *self-signed* digital certificate.

[10] Some authors refer to root CA as "top CA" or "most-trusted CA."

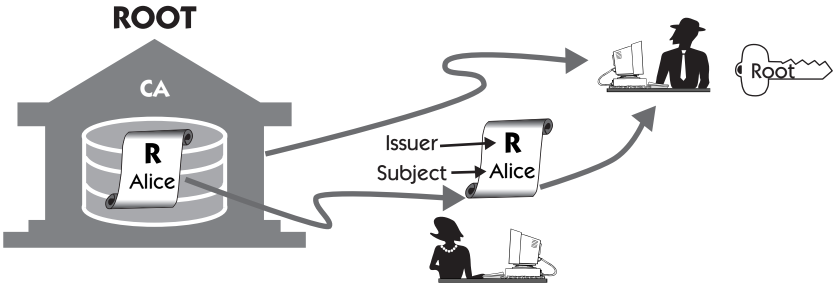


Figure 17-2 Bob uses his copy of Root CA's public key to verify that he received Alice's genuine public key.

Root Certificates

Self-signed (root) digital certificates are accepted without additional verification.

A self-signed certificate is also called a *root certificate*, and it's the foundation[11] of every X.509 PKI implementation (see Figure 17-3). X.509 PKI software looks for self-signed (root) certificates, extracts the attached public key, and assumes it is trusted. This means that the root CA digital certificate and its associated public key are often accepted without additional verification.[12] Chapter 16 shows some examples of self-signed certificates in an Internet browser.

A root CA distributes its public key in a self-signed certificate with Internet browsers and on other public Internet sites. Figure 17-4 shows a screen shot of a GTE CyberTrust self-signed certificate shipped with Microsoft's browser.

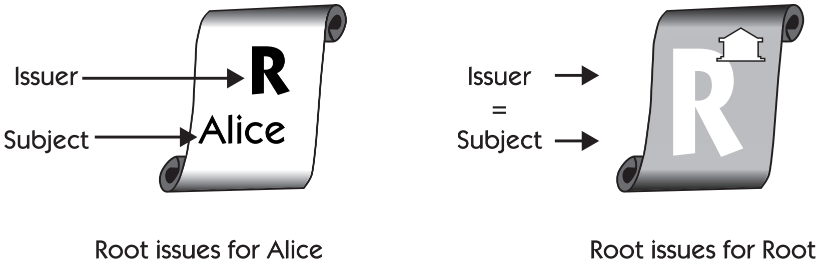


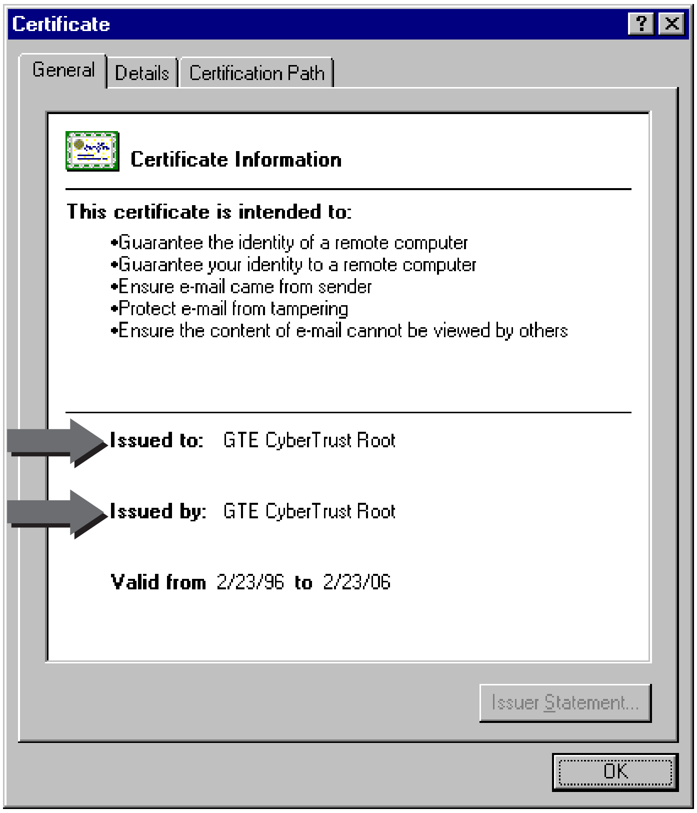
Figure 17-3 Root CA's certificate is a self-signed certificate and is explicitly trusted. We use a certificate symbol with a gray background to emphasize its unique characteristics.

[11] A synonym for *foundation* is *base*. *Base* means "bottom part," as in the base of a mountain. Nevertheless, a root CA is most often shown on top, so we show it that way, too.

[12] Although commercial CAs often publish their certificates (e.g., on Web sites) for verification, few users validate the CA root certificate.

In I.E. 5, click on:
Tools
Internet Options
Content
Certificates
Trusted Root CA
View

In Netscape 4.7, click on:
Communicator,
Tools
Security Info
Certificates
Signers
Edit



Certificate Information

This certificate is intended to:

- Guarantee the identity of a remote computer
- Guarantee your identity to a remote computer
- Ensure e-mail came from sender
- Protect e-mail from tampering
- Ensure the content of e-mail cannot be viewed by others

Issued to: GTE CyberTrust Root

Issued by: GTE CyberTrust Root

Valid from 2/23/96 **to** 2/23/06

Figure 17-4 A self-signed digital certificate in Internet Explorer.

Note that the issuer and subject is GTE CyberTrust. Most popular browsers (such as Netscape) also include many different root CA certificates. Internet e-commerce systems such as Secure Socket Layer rely on self-signed certificates as the beginning point of trust. A root CA that serves only a particular company has other ways to distribute root certificates, such as personal installation, company e-mail, and so on.

Trust flows from the CA. A CA root certificate is trusted by everyone in the CA's PKI.

In a simple X.509 PKI, the root CA signs every certificate. In Figure 17-5, Root CA issues Alice and Bob certificates. Alice verifies Bob's certificate with Root CA's public key; Alice now has a trusted copy of Bob's public key. Similarly, Bob verifies Alice's digital certificate. Alice and Bob have trusted copies of each other's public keys only because they trust Root CA and trust that they have Root CA's public key (genuine digital certificate).

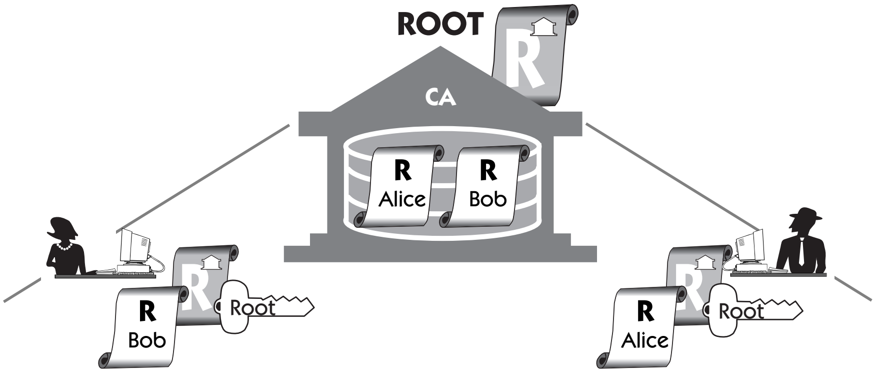


Figure 17-5 Root CA issues certificates for Alice and Bob. They trust each other's certificates and public keys because they trust the issuer (Root CA) and trust that they have Root CA's genuine public key.

All X.509 certificates trust the validity of some root CA certificate

There are many public and private CAs, and each is independent. For example, GTE Certificate Company CA issues certificates without consulting other certificate companies. As we said, as of this writing, there are interoperability issues between CAs. Also, whenever Bob trusts a particular digital certificate, Bob implicitly trusts the issuer (say, xYz Certificate Company) of that particular certificate.

Popularity of X.509 certificates

One reason that companies like the CA model is that it makes it easy to bring someone new into the trusted network of digital certificates. In Figure 17-6, Root CA issues Casey a digital certificate and gives him a copy of Root CA's certificate (public key). That's all that Root CA has to do to bring Casey into the trusted network. Casey can retrieve a copy of Alice's and/or Bob's digital certificate from Root CA, or he can retrieve a copy of those certificates from either Alice or Bob. Alice and Bob can, likewise, get Casey's digital certificate. All the participants—Alice, Bob, and Casey—trust each other's public keys because they trust Root CA, which issued the certificates. But note that if the Root CA self-signed certificate is untrustworthy, so is every certificate that Root CA signed.

Definition: subordinate CA

As we mentioned, a root CA can outsource registration and distribution of certificates. It can also give Alice authority to issue digital certificates, making her a *subordinate* (*sub*) CA.[13] Usually, the CA directs the sub CA to sign certificates with the sub CA's private key and not with the CA's private key.

[13] This is most often abbreviated as sub. In this context, sub means "subordinate" and not "substitute."

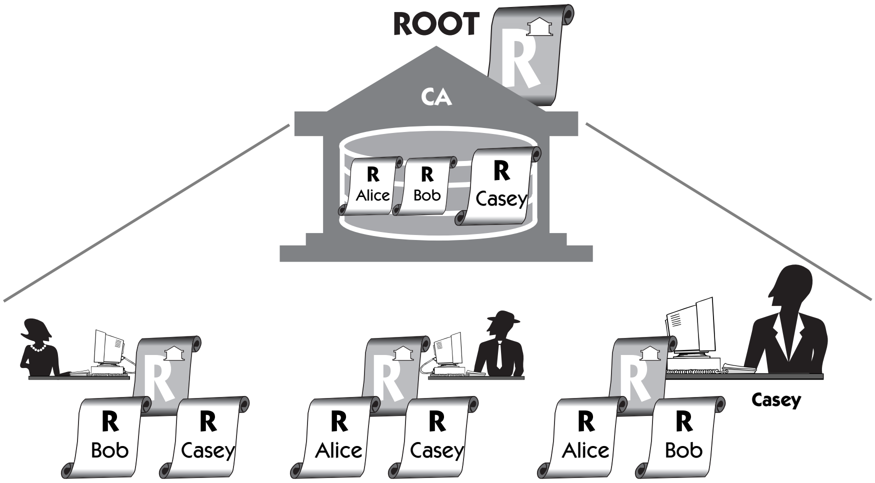


Figure 17-6 Root CA issues Casey a digital certificate and brings Casey into this particular root CA's trusted network.

This management strategy might be reasonable if the root CA issued certificates to full-time employees and the sub CA issued different certificates to contractors and visitors. Sub CA Alice might have different registration and authentication processes for her group of certificate holders. (Or sub CAs can be used for different registration tasks. For example, VeriSign has a sub CA for each of its low, middle, and high assurance certificates.)

Chaining trust from the root CA

In Figure 17-7, Alice issues a certificate to Temp3. Bob can retrieve Temp3's certificate from Alice's public repository, and because Bob has Alice's certificate, he can verify Temp3's certificate issued by Alice, as shown in Figure 17-8.

Bob verifies Temp3's certificate as follows:

1. Bob uses his trusted copy of Root CA's digital certificate to extract Root CA's pubic key (Root CA's public key is accepted without verification).
2. Using Root CA's public key, he verifies Alice's digital certificate and extracts a copy of Alice's public key.
3. Using Alice's public key, he verifies Temp3's digital certificate and extracts a copy of Temp3's public key.[14]

Bob trusts Temp3's digital certificate because he trusts Root CA and Alice, the certificate chain from Root CA to Temp3. If either Root CA or Alice is not

[14] If Temp3 is empowered to issue certificates (e.g., for other temp workers), the certificate chain would grow another link.

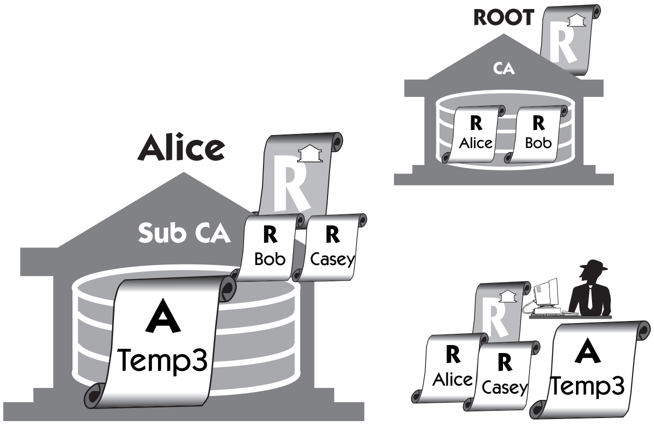


Figure 17-7 Alice, a sub CA, issues a digital certificate to Temp3. Bob gets a copy of Temp3's certificate from Alice's public repository.

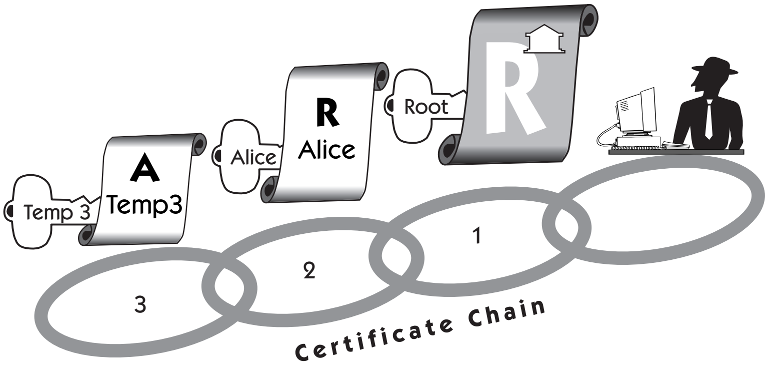


Figure 17-8 A certificate chain. Bob verifies Temp3's digital certificate.

totally trustworthy, Bob is taking a risk in trusting that he has a valid copy of Temp3's public key.

More Risks and Precautions

As previously stated, the CA must ensure the absolute security of its private signing key. Because every CA-issued digital certificate is ultimately signed with the CA's private key, a compromised CA private key means that every digital certificate under this particular CA is suspect. That includes any digital certificate signed by a sub CA (under this particular root CA).

In addition to the need to explicitly trust the root CA, PKIs carry other associated risks. For example, suppose that BlackHat can compromise a single digital certificate in a certificate chain and block access to the current CRL.[15] If BlackHat's attack is successful and users can't get the current CRL, they may incorrectly validate a digital certificate at the end of the chain.

Digital fraud is a budding, burgeoning business. The phrase *caveat emptor* ("let the buyer beware") is as relevant now as it was when secret keys were invented thousands of years ago. But even the best distribution system requires you to trust someone and to take acceptable levels of risk.

Distinguished Names

Definition: distinguished names

Distinguished names are used to give the subject of every digital certificate unique, identifiable characteristics. Suppose that Root CA issues Alice a certificate and places it in a certificate repository (database). How is Alice, wife of Bob, differentiated from the other Alices in the world? The answer is that the CA issues the certificate to a distinguished name. As shown in Figure 17-9, Bob trusts that the CA's distinguished name for Alice has enough data elements (country, business, business unit, e-mail address, etc.) to ensure that there's only one Alice, wife of Bob, mother of Casey and Dawn, and so on.

A Rosenblum by Many Other Names

One of the greatest spies of modern times had many identities. Although, as befitting a spy, his true origins and demise aren't certainly known, he supposedly began life as Sigmund Rosenblum around 1875. During his life he was often known as Sidney Reilly. He also posed as Comrade Relinsky, agent of the Bolshevik secret police.

Reilly, code named ST1 by British intelligence, concocted other identities. For example a brief list includes: a welder in Germany, a merchant near a major Russian naval base, and agent for a German shipbuilder. Reilly is also credited with a plot that seriously wounded Lenin in 1918. He was allegedly executed by the Soviet secret police in 1925.

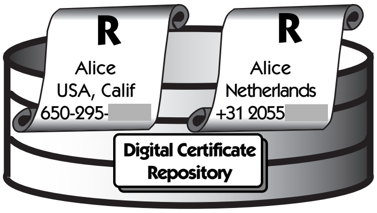


Figure 17-9 Distinguished names are used to differentiate digital certificates.

[15] By using, for example, a denial of service attack.

Certification Practice Statement

Definition: certification practice statement

A CA operates under internally generated legal guidelines called a *certification practice statement* (CPS). A good CA will publish its CPS for prospective customers to read. The CPS details how the CA authenticates its clients, issues certificates, and so on. The CPS doesn't protect Alice or Bob as much as it protects the CA, whose lawyers wrote it.

X.509 Certificate Data

X.509 has been around since 1988 and currently has three versions, titled, not surprisingly, v1, v2, and v3. V3, released in 1996, is the latest and requires certain fields, or information that must be included, as shown in Table 17-2. Following is a brief description of the required fields on an X.509 certificate. References to Alice or Dawn are from Figure 16-2.

Signature method field

The signature method identifies the method the CA used to sign (private key encrypt) the digital certificate. Although we've used RSA to sign, other signature methods, such as DSA, are considered just as secure.

Issuer field: Certificate users trust the certificate issuer.

The issuer can be anyone, but it should be someone who acts responsibly. Recall that a digital certificate is only a private-key-signed (encrypted) message that also contains someone else's public key. Your nephew can sign a certificate proclaiming it contains the public key of Jerry Garcia, late leader of perennial road band the Grateful Dead. Grateful as Garcia might be, he's not going to be

Table 17-2 X.509 v3 mandatory fields.

Field Name	Description
Version	v1, v2, or v3
Serial #	Must be unique.
Signature method	The method used to sign the digital certificate (for example, RSA or DSA).
Issuer name (e.g., Alice)	The signer or entity (person, company, etc.) whose private key signed (encrypted) this certificate. In Figure 16-2, Alice issued the certificate.
Valid time period	Begin and end time the issuer keeps records for this particular certificate.
Subject name (e.g., Dawn)	The person or company whose public key material is included in the next field. In Figure 16-2, we used Dawn.
Subject's public key	Public key and public key method. For example, RSA, DSA, and Diffie-Hellman are three popular options.

Mini-Review
Challenge/Reponse
CA challenges Bob:
Using Bob's public key, CA encrypts challenge phrase ("FiveAM" -> "abcde") and sends it to Bob.

Bob responds to CA:
Bob decrypts the challenge with his private key ("abcde" -> "FiveAM") and returns it to CA.
(See Chapter 7 for more.)

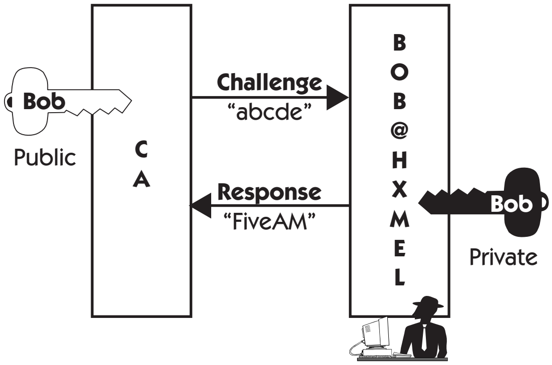


Figure 17-10 A challenge response.

able to tell you from the grave that your nephew is not acting responsibly as an issuer. Did your nephew issue this certificate as a macabre joke, or did he fail to properly check out the person who requested the certificate?

Valid time period field

The time period field contains the issuance and expiration dates for which the issuer certifies the subject's public key; this field is also called the validity interval. The issuer must keep records on the subject until expiration. Most issuers keep records past the expiration date.

You might think that the issuer would be liable if it incorrectly certifies a public key or doesn't quickly inform you when a digital certificate is prematurely revoked (i.e., the issuer has withdrawn the certification of the subject's public key). Not so; see the earlier section, "Certification Practice Statement."

Challenge Response Protocol

Figure 17-10 shows a brief version of a challenge response (introduced in Chapter 7). This is one way that the CA can verify Bob has the matching private key before issuing Bob a digital certificate.

Review

The root certificate authority (CA) is the single focal point of X.509 certificate policies. Digital certificate users trust the accuracy of the public keys the CA issues. Like a DMV, the CA controls certificate application, certification, issuance, expiration, and revocation.

The CA can outsource most of these functions to subcontractors. For example, a PKI defines terms and definitions for registration authorities (RAs), who act on behalf of the CA. The CA can also outsource the distribution of the certificate revocation list (CRL).

A CA root certificate is self-signed by the CA and is usually accepted as valid without additional verification. Commercial CA root certificates are often distributed through another trusted source, such as Netscape's Internet browser. Company CA root certificates can be distributed in other ways.