Chapter 18

Pretty Good Privacy and the Web of Trust

PGP PKIs don't use a CA.

PKIs based on Pretty Good Privacy (PGP) were invented to serve the individual Alices and Bobs of the world. Compared with an X.509 PKI, a PGP-based PKI presents much less bureaucracy that must be managed. In a classic PGP-based PKI, each user issues and manages his or her own digital certificates; there's no certificate authority (CA). This does not necessarily mean that PGP digital certificates are less trustworthy than X.509 certificates. Instead, it means that users of PGP-based PKIs don't have a central controlling authority that assumes responsibilities. PGP cryptographic methods and keys are as strong as those used with X.509.

The History of PGP

PGP created by Phil Zimmermann

A discomfort with central controlling authorities is one reason PGP was invented. Its founder, Philip Zimmermann, uncomfortable with the federal government's record on individual privacy protection, decided to create a product that would give an individual the power to protect his or her own privacy. But it was not an easy birth.

Why Philip Zimmermann Created PGP

"The government has a track record that does not inspire confidence that they will never abuse our civil liberties," Philip Zimmermann explains in *Why I Wrote PGP,* a document that can be downloaded with his software. He proposes one solution to counter the U.S. government's trend to outlaw cryptography and encroach on individual privacy: To make cryptography harder to criminalize, use it as much as possible while it is legal. He wants to avoid privacy being outlawed because that would mean that "only outlaws will have privacy."

(Continued)

> Zimmermann's approach to digital certificate administration, with its "web of trust," mirrors his concern that people should have the power to control their own privacy. He writes, "PGP empowers people to take their privacy into their own hands. There's a growing social need for it. That's why I created it."

During the late 1980s, Zimmermann designed a user-friendly product for personal computer encryption. In June 1991 he asked a friend to post PGP on a computer bulletin board where it could be downloaded for free around the world.

The story of PGP's birth and turbulent early years is engagingly related by Simon Singh in *The Code Book* (see the Bibliography). According to Singh, Zimmermann used RSA patented technology, and he didn't have a license to use it. RSA did not want to give him a free license and pursued him for patent infringement.[1] A bigger problem came when the FBI visited Zimmermann in 1993. For the next three years he was the subject of a grand jury investigation, which examined the issue of whether Zimmermann was an arms dealer because PGP had been exported over the Internet.

Singh relates that during the investigation, public support for Zimmermann grew and cryptographers and civil libertarians established an international fund to help finance his legal defense. In addition, in 1995 Massachusetts Institute of Technology Press published Zimmermann's *The Official PGP User's Guide*, a 600-page book distributed worldwide. Singh writes that the legal authorities were concerned that bringing Zimmermann to trial would "achieve nothing more than a constitutional debate about the right to privacy, thereby stirring up yet more public sympathy in favour of widespread encryption." In 1996, the case against Zimmermann was dropped. The patent issue was also eventually resolved when RSA granted him a license to use its technology in PGP. Current versions of PGP also support Diffie-Hellman.

PGP is freely available; Network Associates sells a commercial version.

In 1997 Zimmermann sold PGP to Network Associates and became one of the company's senior fellows. Although PGP is now sold commercially, you can still download it for free from www.MIT.edu.

Because of Zimmermann and others who fought for widespread use of commercial encryption products, we now have choices in how to handle the security of our communications. So let's see how these choices compare in regard to digital certificates.

Comparing X.509 and PGP Certificates

Both X.509-based PKI and PGP-based PKI are based on digital certificates. Figure 18-1 shows the major differences.

[1] RSA's patent expired in September 2000.

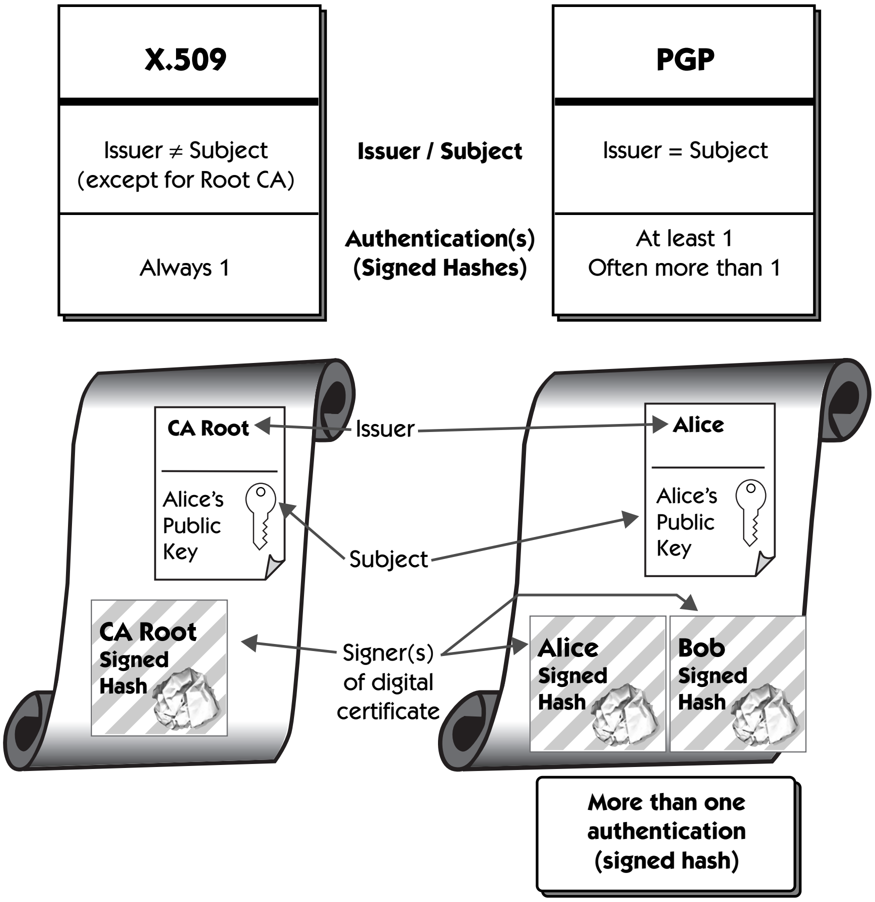


Figure 18-1 Comparing X.509 and PGP digital certificates.

X.509 user certificates are created by a trusted CA.

Almost all X.509 digital certificates have a separate issuer and subject (in Figure 18-1, Root CA issued a certificate for Alice). Only a root CA issues its own certificate; that is, only a root CA certificate is self-signed.

Self-signed certificates are easy to forge. That's why, as you saw in Chapter 17, self-signed certificates are only as trustworthy as the delivery source. For example, Netscape delivers an ATT digital certificate with the Netscape Internet browser.

All PGP users create and sign their own digital certificate.

PGP doesn't use the CA concept. Instead, each user signs his or her own digital certificate; the issuer and subject are identical (in Figure 18-1, Alice issues a certificate for herself). This means that all PGP certificates are initially self-signed; they're similar to X.509 root certificates except that PGP certificates

X.509 certificates have one signer (the issuer); PGP certificates allow more than one signer.

are seldom, if ever, included with Internet browsers. PGP certificates obtain trustworthy status in other ways.

A classic X.509 certificate format allows only one signer per certificate. PGP's certificate format allows more than one person to sign any particular certificate; in theory, each additional signer adds trustworthiness to the certificate. Let's explain with an example.

Building Trust Networks

In Figure 18-2 Alice makes her own certificate and delivers it to Bob.

But BlackHat can intercept Alice's e-mail to Bob (or Casey), substitute a forgery, and sign Alice's name. How does Bob trust a certificate he receives from Alice? The most secure way is for Alice to put a copy of her public key on a disk and hand-deliver it to Bob. Personal hand delivery is secure, but it is not usually convenient in a world of global communications.

Bob Validates Alice's Key

Public keys have fingerprints (HxMel's is at end of book).

Here's an easier way Bob can validate that Alice's certificate (and public key) was not corrupted during transit. The current version of PGP freeware, distributed by MIT, displays a unique public key fingerprint, shown in Figure 18-3. Bob should call Alice and say, "I got your public key; its fingerprint [expressed as

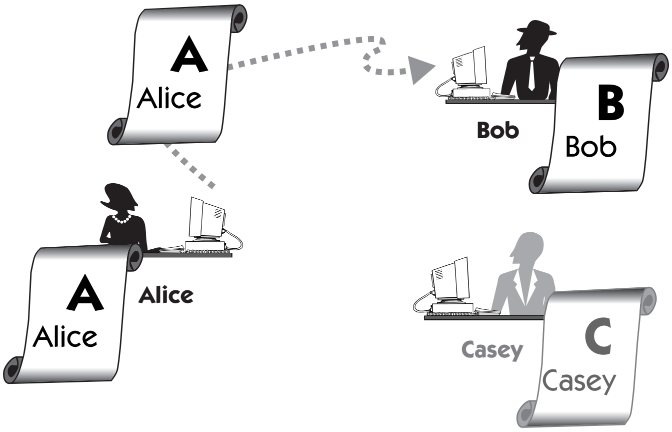
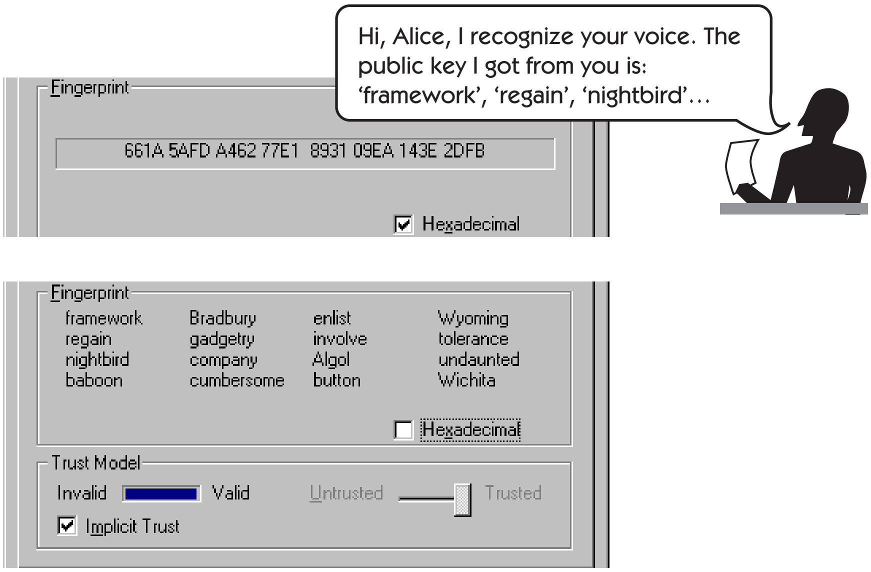


Figure 18-2 In the PGP model, each user creates and distributes his or her keys. Here, Alice sends her digital certificate to Bob.


Hi, Alice, I recognize your voice. The public key I got from you is: 'framework', 'regain', 'nightbird'…

Fingerprint

661A 5AFD A462 77E1 8931 09EA 143E 2DFB

☑ Hexadecimal

Fingerprint

framework	Bradbury	enlist	Wyoming
regain	gadgetry	involve	tolerance
nightbird	company	Algol	undaunted
baboon	cumbersome	button	Wichita

☐ Hexadecimal

Trust Model

Invalid ▬ Valid   Untrusted ▭ Trusted

☑ Implicit Trust

Figure 18-3 Public key fingerprint displayed by PGP.

hexadecimal numbers] is 661A 5AFD.…" If it hasn't been corrupted, the public key fingerprint Bob received will match the fingerprint Alice reads from her computer. Because Bob trusts his knowledge of Alice's voice and her confirmation of her correct fingerprint, Bob trusts that he has Alice's public key (unless someone intercepts Bob's phone call and imitates Alice's voice). HxMel's PGP key is at the end of the Epilogue.

PGP transforms the public key fingerprint into distinct words.

As a convenience, PGP includes a word representation of the hexadecimal numbers, such as "framework, regain, nightbird. . . ." The 20-word fingerprint is made from a list of words carefully chosen so that each word has a distinguishing sound. In that way, Bob is unlikely to misunderstand what Alice says. For example, because the PGP word list includes *regain*, the word list doesn't include *remain*, *remainder*, and *retainer*. Figure 18-3 also shows Alice's fingerprint expressed as words (as well as numbers). Figure 18-4 shows Alice sending her certificate to Bob and Bob's confirming phone call.

Casey Validates Alice's Key Sent by Bob

PGP also allows a user to validate the trustworthiness of another user's digital certificate (public key); that is, PGP empowers Bob so that he can formally corroborate the trustworthiness of Alice's digital certificate.

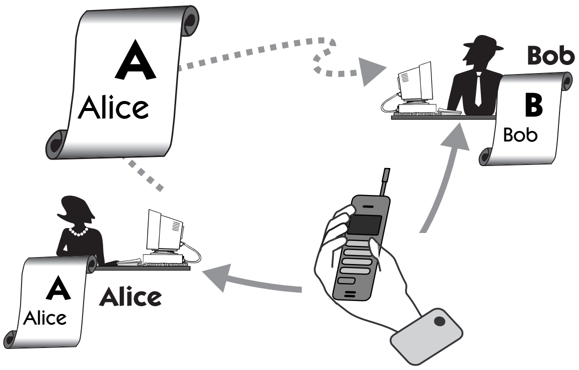


Figure 18-4 Bob confirms that he received Alice's correct public key.

Beginning a PGP web of trust.

Definition: trusted introducer

Let's look at an example. Figure 18-5 assumes that Alice sent Bob her digital certificate and that Bob verified it using the key fingerprint as in Figure 18-4. Bob then sends Casey his self-signed digital certificate. In addition, Bob sends Casey Alice's digital certificate after adding his signature (verification), shown here as a seal with the letter *B* (for Bob).[3] Casey calls Bob and validates Bob's public key fingerprint. If Casey trusts Bob, Casey can also trust Alice's digital certificate (Bob added his signature to Alice's certificate in Figure 18-5). Casey doesn't need to call Alice to verify her fingerprint. Bob is acting as what PGP calls a *trusted introducer*, which is similar to an X.509 CA.

Dawn Validates Alice's Key Sent by Casey via Bob

Casey adds his signature (and more trustworthiness) to Alice's certificate.

Let's look at Figure 18-6, where Dawn enters this PGP community and wants Casey's and Alice's public keys. Casey sends Dawn his digital certificate; then Dawn calls him and validates Casey's digital fingerprint (not shown in Figure 18-6). Dawn also asks Casey to add his signature to Alice's digital certificate, which Casey received from Bob. Casey does so and sends it to Dawn. Then Dawn uses her validated copy of Casey's public key to verify Casey's signature on Alice's certificate.[4]

Maybe Dawn trusts Casey enough that she doesn't need any additional verification. But because each additional signature increases Dawn's trust in Alice's digital certificate, she might also want to verify Bob's validation of Alice. In that case, Dawn will need a verified copy of Bob's public key. She can ask Bob to send it to her directly, or, given that she already has a verified copy of Casey's public key, she can ask Casey to sign and send Bob's certificate.

[3] This may seem different because classic X.509 certificates allow only one signature.

[4] Alternatively, Dawn can request and validate Alice's certificate from Alice directly.

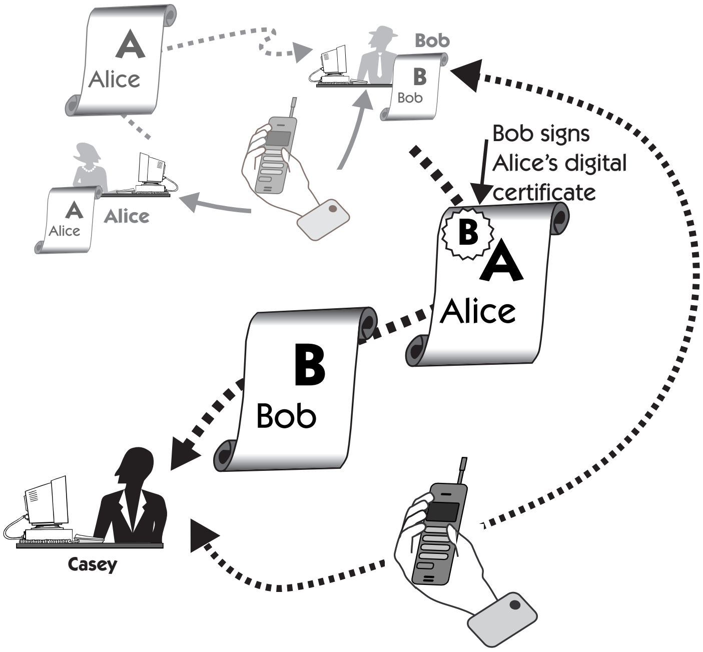


Figure 18-5 After Bob verifies Alice's digital certificate with a phone call to Alice, he sends Casey his (Bob's) certificate along with Alice's certificate validated by him (Bob).

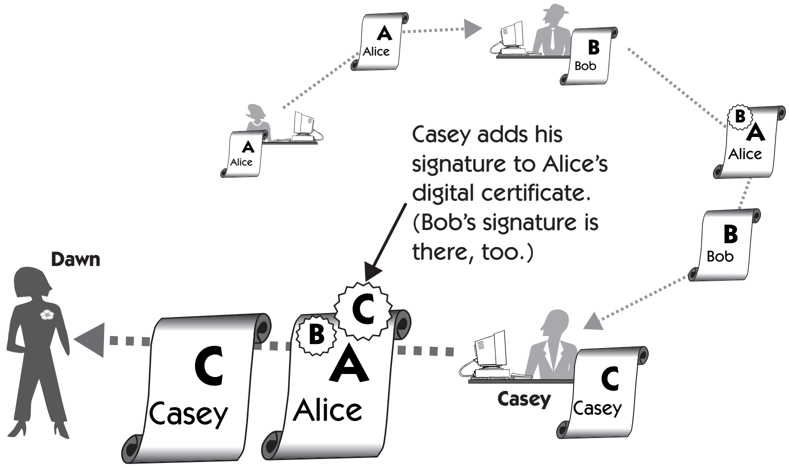


Figure 18-6 Casey sends Dawn his digital certificate and Alice's digital certificate signed by Bob and Casey. Then Dawn verifies Casey's signature on Alice's certificate.

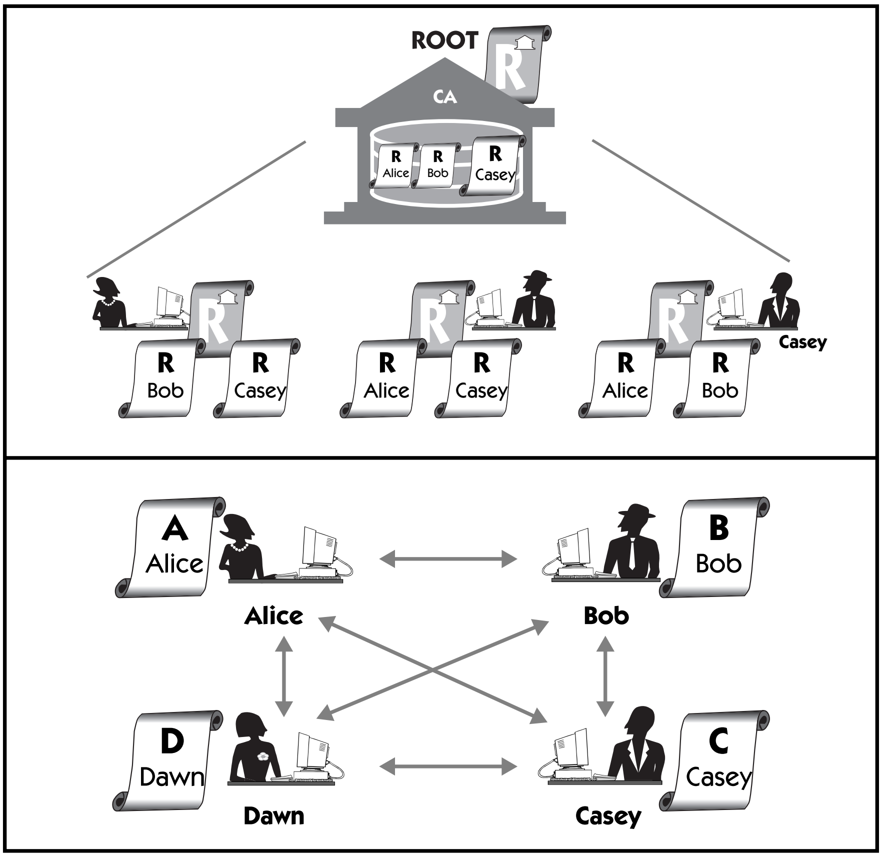


Figure 18-7 Centralized versus decentralized control: comparing an X.509 PKI to a PGP PKI.

Web of Trust

Definition: web of trust

From this limited example, it's apparent that the PGP model rapidly forms an intricate web (of trust). A visual comparison of the PGP and the X.509 models (see Figure 18-7) shows the difference in how trust is established. PGP's trust model is based on a *web of trust*, as opposed to the X.509 model, in which all trust emanates from the certificate authority.

PGP Certificate Repositories and Revocation

PGP users have an option to automatically store copies of their certificates in centralized databases where other users can retrieve them. Each individual certificate owner or some surrogate the owner designates handles certificate revocation.

Compatibility of X.509 and PGP

As of this writing (late 2000), most PKI systems don't support both X.509 and PGP certificates. This unfortunate incompatibility—which means that those using one model can't securely communicate with those using the other model—is also evident in secure e-mail systems (discussed in Chapter 19). However, there is some indication that the two models are moving toward interoperability. X.509 is adopting some PGP features, such as certificates with multiple signatures, and PGP is adopting some X.509 features, such as centralized control.

Review

Philip Zimmermann developed Pretty Good Privacy (PGP), a strong encryption system designed for the masses and based on RSA public key cryptography. PGP is available for free from several Internet servers; probably the most well known source is MIT.

PGP's digital certificates are similar to X.509 self-signed (root) certificates except that PGP certificates can contain more than one signature. In contrast to X.509's centralized control (certificate authority) trust model, PGP uses a distributed trust (web of trust) model.