Private Key Assurances

Authentication
Integrity
Nonrepudiation

Chapter 12

Creating Digital Signatures Using the Private Key

In Chapter 10 Bob used Alice's public key to encrypt a message and gain confidentiality with Alice. In this chapter Alice uses her private key to offer Bob (and her other customers) the other cryptographic assurances: authentication, integrity, and nonrepudiation.

Let's briefly review some public key concepts. Alice's public key provides Bob confidentiality because only Alice can decrypt a message encrypted with her public key. Although Casey knows Alice's public key, he can't decrypt Bob's messages to Alice (see Figure 12-1).

Public key cryptography allows Alice to openly distribute her public key and solves the secret key distribution problem discussed in Chapter 8, arguably the most outstanding cryptographic achievement of the past millennium.

Public key cryptography lets you emulate written signatures.

Amazingly, public key cryptography solves another problem crucial to e-commerce and Internet cyber relationships. The second big win for public key cryptography empowers Alice (or any private key holder) to emulate signed paper documents. This use of public key technology is called a *digital signature*. Because many legal systems and trust are based on signed paper documents, the

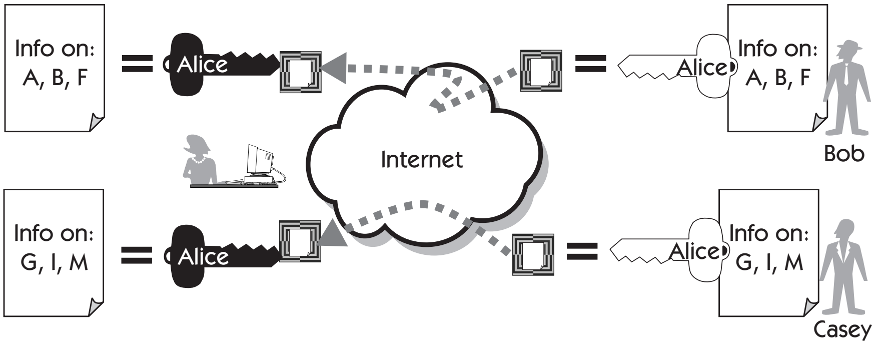


Figure 12-1 Alice's customers send her confidential messages requesting stock reports in her next newsletter. Only Alice can decrypt these messages.

significance of cryptographic digital signatures to the development of e-commerce is difficult to overemphasize.

Written and Digital Signature Assurances

In Figure 12-1, each of Alice's customers sends her requests to include particular stock reports in her next newsletter. Using the confidentiality afforded by Alice's public key, each of her customers encrypts his or her requests so that no one else can discover his or her particular interests. Alice reports on every stock requested in her newsletter. To save money on mailing, Alice does not mail the newsletter in an envelope but rather sends it as an open newsletter at the bulk rate.

Although Alice and her customers don't care who reads her stock reports, her customers want assurances that they receive her genuine, unaltered newsletter. In cryptographic terms, Alice and her customers are concerned with authentication of the newsletter (knowing that it contains only Alice's genuine stock reports) and not with confidentiality (ensuring that the information stays secret).

Written signature assurances

In the paper world, Alice achieves authentication by personally signing each newsletter; her signature assures her customers that the newsletter was created by Alice and was not changed during transit (see Figure 12-2).

Digital signature assurances

In the Internet scenario, as shown in Figure 12-3, Alice and her customers are also concerned with authentication and not with confidentiality. The newsletter is sent by e-mail to many customers and passes through many computers. Alice and her customers don't care that other customers (or anyone else) can read the e-mailed newsletter; Alice's customers want assurances that they receive Alice's genuine newsletter and that the newsletter has not been altered in transit.

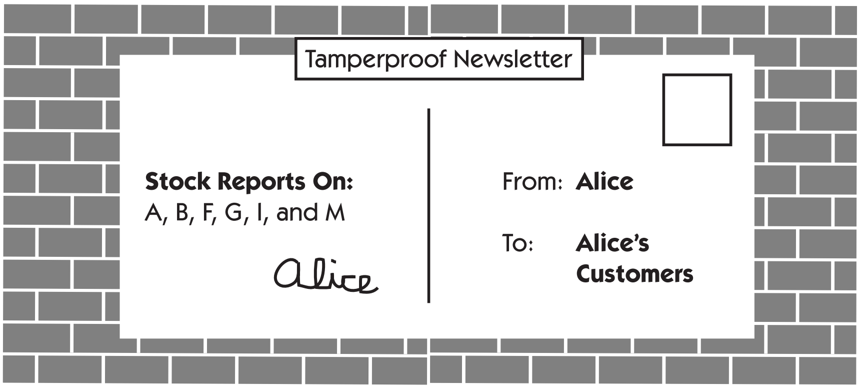


Figure 12-2 A tamperproof, signed newsletter is signed by Alice and can't be replaced by BlackHat without being detected.

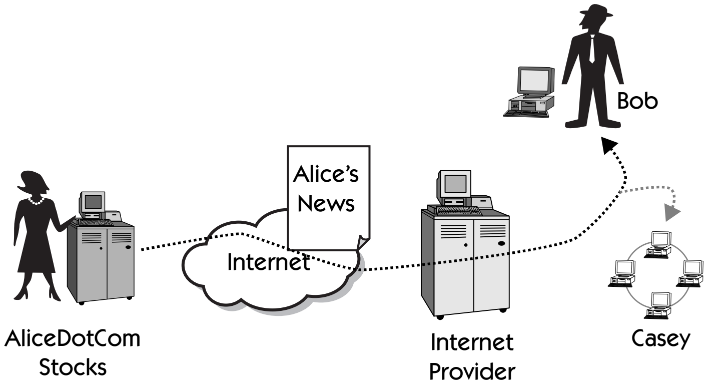


Figure 12-3 Alice e-mails her newsletter to her customers.

Reviewing and Comparing Authentication

Secret key and public key cryptography differ in how they provide authentication.

Secret Key Authentication

For thousands of years, encryption meant secrecy (confidentiality). Authentication (genuineness) is a byproduct of secret key cryptographic systems. Bob trusts the confidentiality and authenticity of the messages sent by Alice because he trusts that only he and Alice share their secret key.

If Alice has only one customer (say, Bob) and a shared secret key with him, it's easy to get those assurances. Alice encrypts the newsletter with the secret key and sends it to Bob with a MAC or a message digest.[1] If BlackHat substitutes his own newsletter for Alice's newsletter, it simply won't decrypt with Alice and Bob's shared secret key.

Private Key Authentication

Public key assurances are implemented differently from secret key assurances.

Because public key cryptography has public and private keys, cryptographic assurances are implemented differently than with secret key cryptography.

[1] A *message digest* is another kind of message fingerprint. It is introduced in Chapter 7 and explained in more detail in Chapter 13.

Public key cryptography allows messages to be encrypted with the private key (as well as the public key).[2]

Understandably, encrypting with the private key can be confusing because, until this chapter, encryption has been associated with confidentiality. But Alice doesn't encrypt with her private key for the purpose of making confidential messages. Instead, Alice encrypts her newsletter with her private key to prove to Bob that the newsletter he receives originated from her.

Many copies of Alice's public key but one copy of her private key

Recall that when Alice makes her public/private key pair, she openly distributes many copies of her public key to her many customers. She never shares her private key; there is only one copy of her private key. So any message that Bob or anyone else decrypts with Alice's public key must have been encrypted with Alice's private key.

Private key encryption provides authentication, integrity, and nonrepudiation.

Private key encrypted message

This means that encrypting with her private key gives Alice a way to prove she authored the message (author authentication) and to assure recipients that the message was not altered in transit (message authentication, also known as integrity). So even though a private key encrypted message does not provide confidentiality, it does provide authentication and integrity (see Figure 12-4). (Notice in Figure 12-4 that a slightly different symbol is used to illustrate a private key encrypted message. We use this symbol in the rest of the book.) You'll also see that private key encrypted messages provide nonrepudiation.

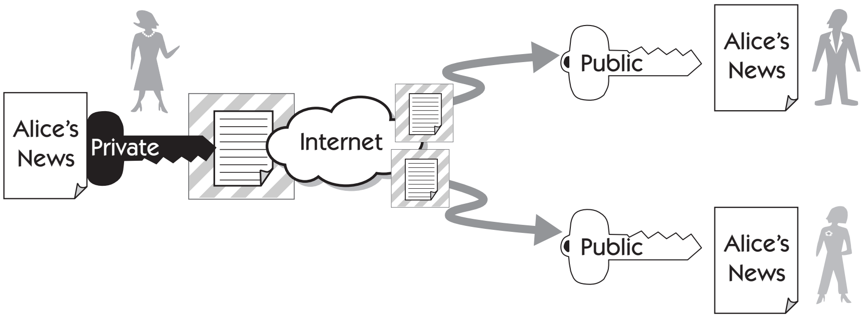


Figure 12-4 Alice encrypts her message using her RSA private key. Any owner of her public key can decrypt the message.

[2] Although the phrase "encrypted with the private key" is widely used and accepted, purists prefer the terms that are discussed in the section "Signing Terminology" later in this chapter.

Authentication and Integrity Using Private and Secret Keys

Public key cryptography supports authentication and integrity more efficiently than secret key cryptography

As has been explained, secret key cryptography also supports authentication and integrity. But Alice must share a separate secret key with each customer, something that's too difficult to administer (see Chapter 8). Public key cryptography supports authentication and integrity more efficiently. Alice needs only encrypt one newsletter, which she then copies and sends to each public key holder. *Any* public key holder decrypting the newsletter is assured that Alice encrypted the newsletter (ensuring authenticity) and that the newsletter has not been altered since Alice encrypted it (ensuring integrity). Of course, all assurances are null and void if someone else has Alice's private key. But the key to your privacy isn't something you intend to share.

Definition: digital signature

Private key encryption is just like signing your name, and that's why it's called a *digital signature*. This is another important reason to keep your private key private.

But what if BlackHat intercepts Alice's encrypted newsletter and substitutes his own private key encrypted newsletter (see Figure 12-5)?

Decrypting BlackHat's spoofed newsletter with Alice's public key makes gibberish. You might think that because BlackHat knows Alice's public key, he could figure out how to make encrypted text that decrypts to an intelligible message. So far there's no known successful attack of this kind.

If BlackHat wants to convince Bob that his forged newsletter is from Alice, BlackHat must somehow dupe Bob into using BlackHat's public key instead of Alice's public key. For example, BlackHat could intercept Alice's e-mail, which

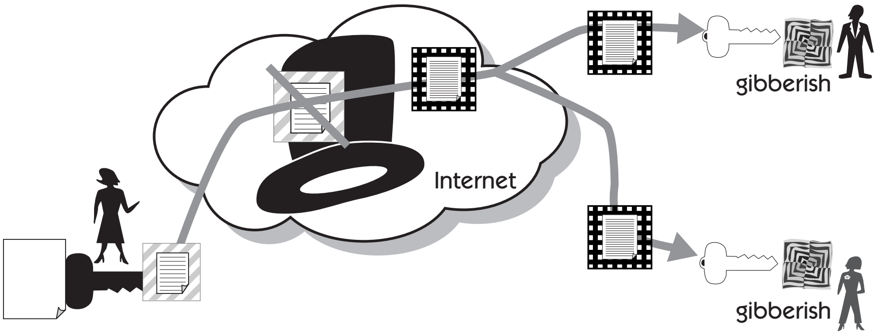


Figure 12-5 BlackHat intercepts Alice's newsletter and substitutes his own, which he sends to Alice's customers who end up with gibberish after decrypting with Alice's public key.

distributes her public key to Bob, and substitute his public key for Alice's. In Chapter 22, you'll see what happens when BlackHat successfully substitutes his public key for Alice's. Digital certificates help Bob prevent BlackHat from successfully completing that switch; see Chapters 16 through 18.

Private Key Authentication Methods

Now let's take a look at the private key authentication methods: RSA and Digital Signature Algorithm (DSA).

RSA

Since its invention in the late 1970s, RSA using a sufficiently long key has withstood all known attacks. Interestingly, a recent plausible attack against RSA was by Adi Shamir, the *S* in RSA.[3]

RSA offers all cryptographic assurances.

RSA is the only widely used public key cryptographic system that enables its public *and* private keys to encrypt messages. The math behind RSA makes both public and private key encryption equally secure. This means that RSA provides both confidentiality (encrypting with public key and decrypting with private key) and digital signing (encrypting with private key and decrypting with public key). Almost all other popular cryptographic methods support one or the other but not both. RSA's versatility seems almost too good to be true. Surprisingly, because RSA is so versatile, if Alice uses the same RSA key pair for both confidentiality and digital signatures, BlackHat can mount an attack to recover encrypted plaintext (see Chapter 22). Because of these kinds of attacks and for other reasons, many cryptographers believe that anyone using RSA should keep two RSA key pairs: one pair used exclusively for signatures and a second pair for everything else.

RSA's math puzzle is based on factorization.

As shown in Chapter 11, the knapsack method is based on the difficulty of finding numbers that exactly sum to a given total. RSA's security is based on the difficulty of factoring the product of two (large) prime numbers. RSA is still secure against attack, but larger and larger prime numbers are needed because of the ever-increasing power of microprocessors.

RSA, like all good commercial encryption formulas, is openly published. Allowing anyone to attack the method helps to prove or disprove its strength. (See Appendix A for a more in-depth mathematical look at RSA security and information about how to make RSA key pairs.)

[3] Recently, Shamir theorized a way to cryptanalyze a 540-bit RSA key. Most public key systems use at least a 1,024-bit key.

DSA

Digital Signature Algorithm is used only for digital signatures. As with RSA, DSA's private key encryption provides authentication, integrity, and nonrepudiation. But unlike RSA, DSA cannot be used for confidentiality.[4]

DSS is the U.S. government standard that mostly addresses DSA.

DSA was proposed to NIST and was adopted as a U.S. Federal Information Processing Standard (FIPS) in the early 1990s. At the time, several controversial issues surrounded DSA. For example, some believed that because RSA was already in wide use, it should be adopted as the standard; and some believed that NSA had too much development oversight over DSA. DSA has proven to be as secure as RSA. DSA is described in FIPS Pub 186-1 (revised in 1998) titled "Digital Signature Standard (DSS)."[5]

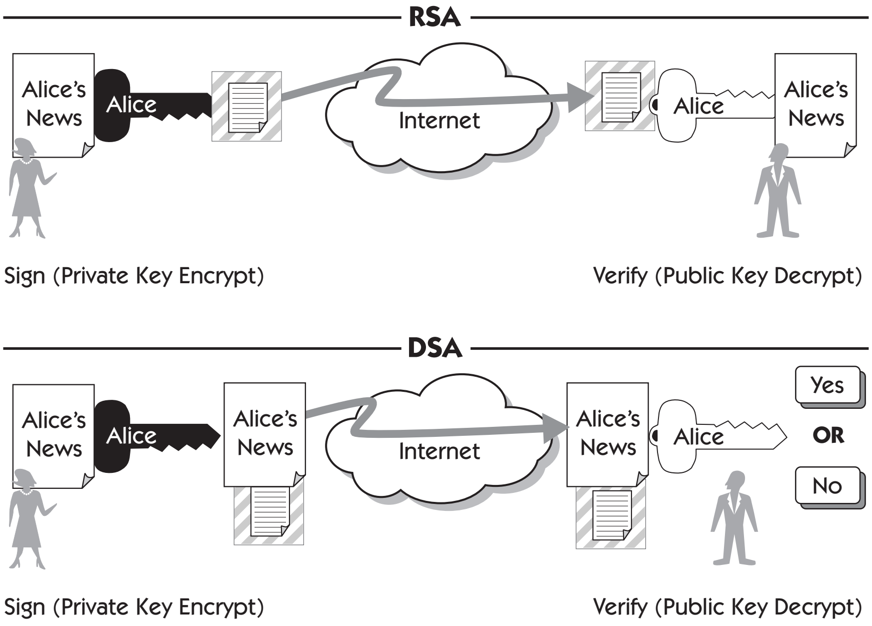


Figure 12-6 Comparing RSA and DSA signing and verification. Note that RSA verification (decryption) recovers the original signed (encrypted) plaintext. DSA verification is "yes" (accept) or "no" (reject).

[4] Although a modified DSA can support public key encryption, this feature is complex and seldom used.

[5] Although DSS is mostly about DSA, it also approves RSA as a signature method.

RSA signature verification recovers plaintext, but DSA does not.

A major difference between RSA and DSA, shown in Figure 12-6, is that RSA digital signature verification (public key decryption) does not need the original plaintext. DSA, like most other popular digital signature methods, does need the original plaintext to verify the digital signature. In other words, RSA digital signature verification recovers plaintext; DSA does not.

DSA, as it's most often used, gives Bob a simple "yes" or "no"—that is, it accepts or rejects the validity of the digital signature. In practice, this distinction doesn't matter because, as you'll see in Chapter 13, both signing methods compress (digest) the plaintext before signing.

RSA and DSA produce signatures in about the same amount of time, but RSA verifies signatures much faster than does DSA. But DSA can precompute some of its values and thereby produce signatures faster than RSA. Even with precomputed values, however, RSA verifies much faster than DSA.[6]

Signing Terminology

Formal terminology: Only with RSA is private key encryption equivalent to signing and public key decryption equivalent to verification.

Although you'll see the terms *private key encryption* and *public key decryption* used for RSA, DSA, and so on, technically speaking, they should be used only for RSA (because, as seen in Figure 12-6, only RSA keys encrypt and decrypt plaintext). When you're using digital signature methods, the technically correct terms are *signing* (instead of private key encrypting) and *verifying* (instead of public key decrypting). Although *private key encrypting* and *public key decrypting* are widely accepted and used, we use *signing* and *verifying* in the remainder of the book. To accustom you to this nuance in the terminology, from time to time we'll write *signing* with *private key encryption* in parentheses and *verifying* with *public key decryption* in parentheses.

Nonrepudiation

After Alice digitally signs (encrypts with her private key) and sends her newsletter, the recipient is assured that it was made by Alice (or someone who knew her private key). Alice can't deny or repudiate her newsletter recommendations.

[6] Because DSA can generate signatures faster than RSA, perhaps DSA will be used to generate digital signatures on smart cards (see Chapter 23). Then the slower signature verification process can be handled on more powerful network server computers.

This cryptographic assurance, called *nonrepudiation*, is an essential component of Internet commerce.

For example, Bob requests that Alice buy Widget Corporation for his account, and Alice responds with a digitally signed confirmation (a message encrypted with her private key). Alice can't later deny that she sent the confirmation because, as mentioned previously, no one can forge a signed (encrypted) confirmation that will verify (decrypt) with Alice's public key.[7] Of course, someone who stole Alice's private key could forge the signed confirmation.

Assurances in Both Directions

In real-world systems every person has his or her own public/private key pair.

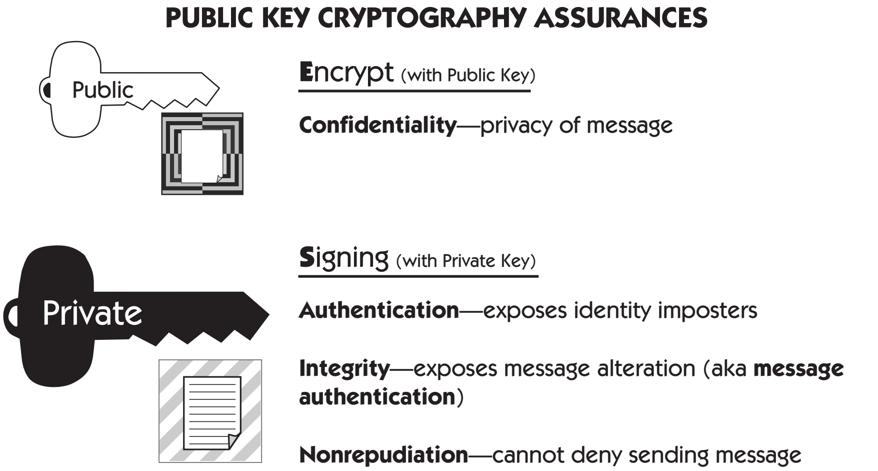
In the simple system just shown, we assumed that only Alice created a public/private key pair and that she has shared her public key with Bob and her other customers. Alice uses her private key to assure Bob of her authenticity, message integrity, and nonrepudiation. But because Bob only has Alice's public key, he can't offer Alice the same assurances. To assure Alice that Bob's communications to her are from him, Bob must create his own public/private key pair and share his public key with Alice. In Part IV we show cryptographic assurances in both directions.

Summary of Public Key Assurances

Figure 12-7 briefly summarizes public key cryptographic assurances and shows the symbols we use in the remainder of the book.

Alice's public/private matched pair is shown as white and black, respectively. A white (public) key encrypted message makes a confidential message and is shown against a black and dark gray background to illustrate concealment or secrecy. A black (private) key signed (encrypted) message is indisputably made by Alice (the sole holder of the private key) and is shown against a white and light gray background to illustrate openness or easy visibility. Anyone can verify that the private key signed (encrypted) message is from Alice and then read it because anyone can get and use Alice's public key.

[7] Recall that this assurance can't be made with secret keys alone; Untrusty denied that he sent Alice and Bob a secret key encrypted confirmation (Chapter 7).



PUBLIC KEY CRYPTOGRAPHY ASSURANCES

Encrypt (with Public Key)

Confidentiality—privacy of message

Signing (with Private Key)

Authentication—exposes identity imposters

Integrity—exposes message alteration (aka **message authentication**)

Nonrepudiation—cannot deny sending message

Figure 12-7 Public key cryptography provides the same assurances as secret key cryptography, but those assurances are provided differently. Public key cryptography also prevents the sender from denying sending a message (that is, it assures nonrepudiation), something that is not supported in secret key cryptography without a trusted third party.

Public Key Means Public / Private Key

Private keys are kept secret but never shared.

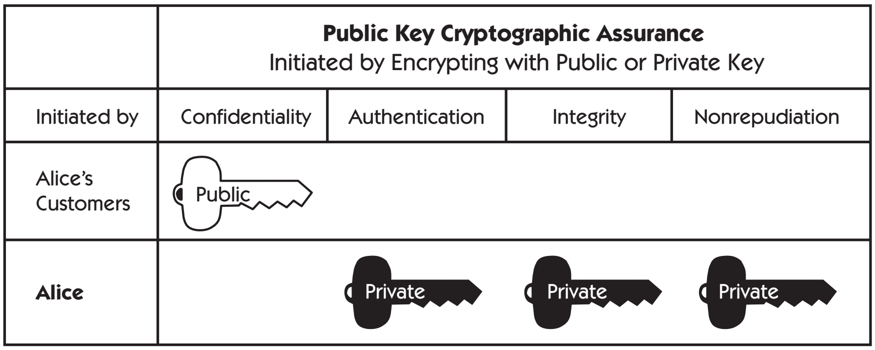
It's also instructive to clarify some public key terminology. All public key cryptographic methods have both pubic and private keys. Nevertheless, public key cryptography is seldom referred to as public/private key cryptography. A private key is always kept secret (concealed or undisclosed), just as secret keys are kept secret. However, one important difference is that private keys are never shared; secret keys are often shared between two (or more) people.

Assurance Initiated

Figure 12-8 shows who initiates a particular assurance. For example, Bob, a customer who has a copy of Alice's public key, initiates confidentiality by encrypting a message with Alice's public key.

Compressing before Signing

Signing (encrypting with a private key) is extremely slow, so you usually add a time-saving (and space-saving) step before you encrypt the message. It's called



	Public Key Cryptographic Assurance Initiated by Encrypting with Public or Private Key			
Initiated by	Confidentiality	Authentication	Integrity	Nonrepudiation
Alice's Customers	Public			
Alice		Private	Private	Private

Figure 12-8 Cryptographic assurances with public and private keys.

message *digesting* or *hashing*. "Digesting" might conjure an image of someone eating a message—yummy, but that's not quite what it means. Chapter 7 briefly introduces message digests, and Chapters 13 and 14 discuss them in more depth.

Review

Public key confidentiality is one-way—from public key holder to private key holder. Bob, Casey, and BlackHat have copies of Alice's public key and can send Alice confidential messages. Alice cannot send them confidential messages encrypted with her private key.

But Alice can provide her public key holders some assurances through the use of signing (encryption) with her private key. Alice can assure her public key holders that her messages are from the authentic Alice and haven't been altered in transit. She signs (encrypts) the messages with her private key (see Figure 12-4). Private key encryption is called digitally signing. Digital signatures need a public key cryptographic system; each person has a private key, which is not shared.

Authentication, integrity, and nonrepudiation extend from the private key holder to public key holders (note the plural).

In general, after Alice creates her RSA or DSA private/public key pair, only she uses her private key. Her customers know only Alice's public key; that's all they use. In fact, if someone other than Alice ever learns her private key, her entire public key cryptographic system is insecure. All future messages lack any assurances.