3

Encryption, Digital Signatures, and Certification Authorities

For the exchange of information and commerce to be secure on any network, a system or process must be put in place that satisfies requirements for confidentiality, access control, authentication, integrity, and nonrepudiation. The key to the securing information on a network is cryptography. Cryptography can be used as a tool to provide privacy, to authenticate the identities of communicating parties, and to ensure message integrity. Confidentiality, access control, authentication, integrity, and nonrepudiation are terms that were introduced in Chapter 1, but they are defined again in Table 3.1.

Traditionally, cryptography conjures up thoughts of spies and secret codes. In reality, cryptography and encryption have found broad application in society. Every time you use an ATM machine to get cash or a point-of-sale machine to make a purchase, you are using encryption. Encryption is the process of scrambling the contents of a file or message to make it unintelligible to anyone not in possession of the "key" required to unscramble it.

Civilizations have been using various cryptosystems for at least 4,000 years. A cryptosystem or algorithm is the process or procedure to turn plaintext into cryptotext. A crypto algorithm is also known as a "cipher." There are several key elements that go into making an effective cryptosystem. First and foremost it must be reversible. A crypto algorithm is of no practical use if

Table 3.1
Cryptography Terms

Term	Definition
Confidentiality	The ability to encrypt or encode a message to be transmitted over an insecure network
Access control	The ability to control the level of access that an individual or entity can have to a network or system and how much information they can receive
Authentication	The ability to verify the identity of individuals or entity on the network
Integrity	The ability to ensure that a message or data has not been altered in transit from the sender to the recipient
Nonrepudiation	The ability to prevent individuals or entities from denying that they sent or received a file, when in fact they did

once you have scrambled your information, you cannot unscramble it. The security of the cryptosystem should be dependent on the secrecy and length of the key and not on the details of the algorithm. In other words, knowing the algorithm should not make it significantly easier to crack the code (restricted versus unrestricted). If security is dependent on keeping the algorithm secret, then it is considered a "restricted" algorithm. It is also important that the algorithm has been subjected to substantial cryptoanalysis. Only those algorithms that have been analyzed completely and at length are trustworthy. The algorithm should contain no serious or exploitable weakness. Theoretically, all algorithms can be broken by one method or another. However, an algorithm should not contain an inherent weakness that an attacker can easily exploit.

Below is an example of a cipher; to scramble a message with this cipher, simply match each letter in a message to the first row and convert it into the number or letter in the second row. To unscramble a message, match each letter or number in a message to the corresponding number or letter in the second row and convert it into the letter in the first row.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T

To illustrate how this works, see the following where the cipher is used to scramble the message "Little green apples."

- Cipher text: FCNNF5 AL55H 1JJF5M;
- Clear text: LITTLE GREEN APPLES.

This rudimentary cipher would not be effective at keeping a message secret for long. It does not comply with one of the qualities of a truly effective cipher, where knowing the algorithm should not make it significantly easier to crack the code. This is an example of a restricted algorithm. In this case, the cipher is the code. Once you know the cipher, you can unscramble any message. Ciphers usually fall into one of two categories: block ciphers or stream ciphers.

Stream Ciphers

Stream cipher algorithms process plaintext to produce a stream of ciphertext. The cipher inputs the plaintext in a stream and outputs a stream of ciphertext. Figure 3.1 illustrates the concept of the stream cipher's function.

Stream ciphers have several weaknesses. The most crucial shortcoming of stream ciphers is the fact that patterns in the plaintext can be reflected in the ciphertext. To illustrate this weakness we can use the rudimentary cipher introduced earlier in the chapter. Below, I have scrambled the plaintext message "Let us talk one to one" into ciphertext to compare the two patterns:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T

- Plaintext: Let us talk one to one.
- Ciphertext: F5n om n1fe ih5 ni ih5.

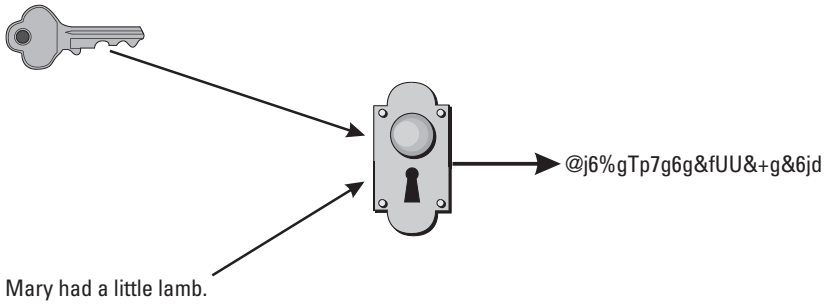

Figure 3.1 Stream cipher.

Patterns in the plaintext are reflected in the ciphertext. Words and letters that are repeated in the plaintext are also repeated in the ciphertext. Knowing that certain words repeat makes breaking the code easier. In addition, certain words in the English language appear with predictable regularity. Letters of the alphabet also appear in predictable regularity. The most commonly used letters of the alphabet in the English language are E, T, A, O, N, and I. The least commonly used letters in the English language are J, K, X, Q, and Z. The most common combination of letters in the English language is "th." As a result, if a code breaker is able to find a "t" in a code, it doesn't take long to find an "h." It is not hard for a trained code breaker to break this type of code.

Another weakness of stream ciphers is that they can be susceptible to a substitution attack even without breaking the code. This is a type of replay attack where someone can simply copy a section of an old message and insert it into a new message. You don't need to break the code to insert the old section into a new message.

Examples of stream ciphers include the Vernam cipher, Rivest cipher #4 (RC4), and one-time pads.

Block Ciphers

Block ciphers differ from stream ciphers in that they encrypt and decrypt information in fixed size blocks rather than encrypting and decrypting each letter or word individually. A block cipher passes a block of data or plaintext through its algorithm to generate a block of ciphertext. Ideally, a block cipher should generate ciphertext roughly equivalent in size (in terms of number of blocks) to the cleartext. A cipher that generates a block of ciphertext that is significantly larger than the information it is trying to protect is of little practical value. Think about it in terms of network bandwidth: If the ciphertext block was twice the size of the plaintext, the net effect is that your bandwidth would be cut in half. This would also have an impact on files stored in an encrypted format. An unencrypted file 10 MB in size would be 20 MB in size when encrypted.

Another requirement of block ciphers is that the ciphertext should contain no detectable pattern. Figure 3.2 illustrates how block ciphers function. Examples of well-known block ciphers include the Data Encryption Standard (DES), the International Data Encryption Algorithm (IDEA), and SKIPJACK.

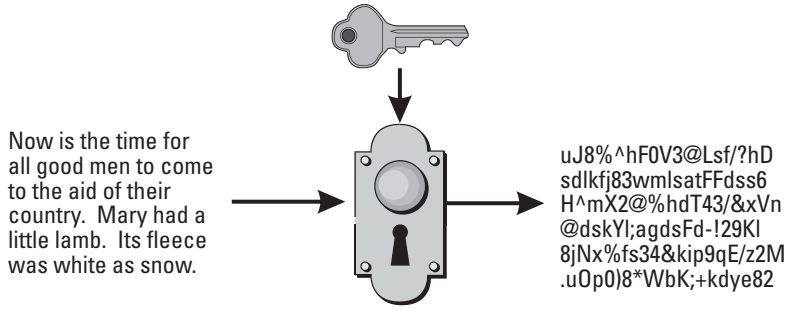

Figure 3.2 Block cipher.

Breaking Ciphers

For as long as ciphers have existed, there have been people trying to break them. There are many methods employed to break cipher. Some methods are ingenious. Some are sophisticated and technical in nature, while others are more crude in nature. The following sections describe some of the more widely used techniques employed in breaking ciphers.

Known Plaintext Attack

This method relies on the code breaker knowing in advance the plaintext content of a ciphertext message. Having both the plaintext and the ciphertext the code breaker reengineers the cipher and the key used to create the ciphertext.

Chosen Plaintext Attack

This method relies on the ability of the code breaker to somehow get a chosen plaintext message encrypted. During World War II the United States used a variation of this method to ascertain the plans of the Japanese navy in the Pacific.

Right after Pearl Harbor the U.S. Pacific Fleet was forced to fight what was primarily a defensive war. The U.S. Pacific Fleet had been devastated by the Japanese surprise attack on Pearl Harbor, and all that was left of the fleet were three aircraft carriers and a handful of supporting ships.

The United States had some success in breaking the Japanese codes. The U.S. Navy had determined that the Japanese were planning to attack a location referred to in their transmissions as “AF.” The United States

suspected that site AF was Midway Island. To determine if AF was, in fact, Midway, the United States ordered that a message be transmitted from Midway stating that the island's water condenser had broken down. The message was to be sent in the clear so that there would be no chance that the Japanese could not intercept it. Sure enough the Japanese took the bait. A few days later, the United States intercepted a Japanese coded message stating that AF's water condenser had failed.

From that message the United States knew that the Japanese were going to attack Midway. As a result, the United States was able to send what was left of the Pacific Fleet to Midway where they ambushed the Japanese carrier task force. The United States sank four of the Japanese' frontline aircraft carriers. It was a strategic victory for the United States in the Pacific from which the Japanese navy never recovered. From that point on, it was the Japanese Navy that was forced to fight a defensive war.

Cryptanalysis

Technically, any method employed to break a cipher or code is cryptanalysis. However, when I refer to cryptanalysis I am specifically talking about employing mathematical analysis to break a code. This method requires a high level of skill and sophistication. It is usually only employed by academics and governments. Today it relies very heavily on the use of ultrafast super computers.

Probably the most active and successful organization in the world, dedicated to breaking codes, is the National Security Agency (NSA). This is the largest and most secret spy agency in the United States. It is sometimes referred to as the Puzzle Palace, because the group spends so much time and energy on codes and cipher. The NSA employs tens of thousands of people. The only comparable organization in the world ever to have existed in terms of size is the former Soviet Union's KGB. But with the breakup of the Soviet Union, the NSA is now left without peers.

Brute Force

The brute force method tries every possible combination of keys or algorithms to break a cipher. Doing so can require tremendous resources. Usually, this type of attack requires computer assistance. If the algorithm is simple or the key is small, then the CPU resources required could be provided by a simple PC. If the algorithm is sophisticated or the key is large, then advanced computing power might be required.

Social Engineering

This method relies on breaking a cipher by getting someone knowledgeable about the cipher to reveal information on how to break it. Bribing someone, tricking him or her into divulging information, or threatening him or her with harm can reveal information. When the threat of harm is employed it is sometimes referred to as rubber-hose cryptanalysis.

Other Types of Attacks

Some other types of attacks are discussed as follows.

- *Substitution:* This is a type of replay attack where a previous message, in part or in whole, is inserted into a legitimate message. An attacker does not need to break the cipher for this type of attack to be effective.
- *Timing attacks:* Some cryptosystems can be broken if an outsider is able to accurately measure the time required to perform the encryption and decryption of a known ciphertext. The known ciphertext and the timing provide enough information to deduce fixed exponents and factors of some systems. This vulnerability is mostly theoretical. If an attacker has enough access to a network to be able to accurately measure the time required to encrypt and decrypt information, then you have other and bigger problems to worry about.

Encryption

Encryption is the process of scrambling the contents of a file or message to make it unintelligible to anyone not in possession of the "key" required to unscramble the file or message. There are two types of encryption: symmetric (private/secret) key and asymmetric (public) key encryption.

Symmetric Key Encryption

When most people think of encryption it is symmetric key cryptosystems that they think of. Symmetric key, also referred to as private key or secret key, is based on a single key and algorithm being shared between the parties who are exchanging encrypted information. The same key both encrypts and decrypts messages. This concept is illustrated in Figure 3.3.

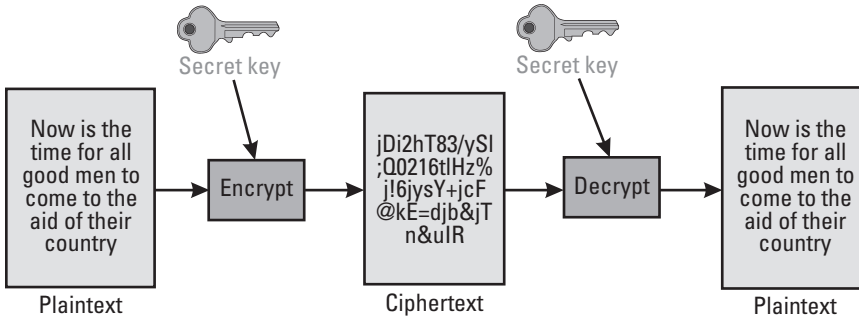

Figure 3.3 Symmetric key encryption.

The strength of the scheme is largely dependent on the size of the key and on keeping it secret. Generally, the larger the key, the more secure the scheme. In addition, symmetric key encryption is relatively fast.

The main weakness of the system is that the key or algorithm has to be shared. You can't share the key information over an unsecured network without compromising the key. As a result, private key cryptosystems are not well suited for spontaneous communication over open and unsecured networks. In addition, symmetric key provides no process for authentication or nonrepudiation. Remember, nonrepudiation is the ability to prevent individuals or entities from denying (repudiating) that a message was sent or received or that a file was accessed or altered, when in fact it was. This ability is particularly important when conducting e-commerce. Table 3.2 lists the advantages and disadvantages of symmetric key cryptosystems. Examples of widely deployed symmetric key cryptosystems include DES, IDEA, Blowfish, RC4, CAST, and SKIPJACK.

Table 3.2
The Advantages and Disadvantages of Symmetric Key Cryptography

Advantages	Disadvantages
Fast	Requires secret sharing
Relatively secure	Complex administration
Widely understood	No authentication
	No nonrepudiation

Data Encryption Standard (DES)

DES is one of the oldest and most widely used algorithms. DES was developed by IBM with the encouragement of the NSA. It was originally deployed in the mid 1970s. DES consists of an algorithm and a key. The key is a sequence of eight bytes, each containing eight bits for a 64-bit key. Since each byte contains one parity bit, the key is actually 56 bits in length. According to author James Bamford in his book *The Puzzle Palace,* IBM originally intended to release the DES algorithm with a 128-bit key, but the NSA convinced IBM to release it with the 56-bit key instead. Supposedly this was done to make it easier for the NSA to decrypt covertly intercepted massages.

DES is widely used in automated teller machine (ATM) and point-of-sale (POS) networks, so if you use an ATM or debit card you are using DES. DES has been enhanced with the development of triple DES. However, DES has been broken. It is gradually being phased out of use.

International Data Encryption Algorithm (IDEA)

IDEA is a symmetric key block cipher developed at the Swiss Federal Institute in the early 1990s. IDEA utilizes a 128-bit key. Supposedly, it is more efficient to implement in software than DES and triple DES. Since it was not developed in the United States, it is not subject to U.S. export restrictions.

CAST

The CAST algorithm supports variable key lengths, anywhere from 40 bits to 256 bits in length. CAST uses a 64-bit block size, which is the same as the DES, making it a suitable drop-in replacement. CAST has been reported to be two to three times faster than a typical implementation of DES and six to nine times faster than a typical implementation of triple DES. The CAST algorithm was developed by Carlisle Adams and Strafford Travares and patented by Entrust Technologies, but a version of the CAST algorithm is available for free commercial and noncommercial use. CAST is employed in Pretty Good Privacy (PGP).

Rivest Cipher #4 (RC4)

Developed by Ron Rivest of RSA fame, RC4 is a stream cipher that uses a variable size key. However, when used with a key of 128 bits it can be very effective. Until recently, the approved export version only used a 40-bit key. RC4 is used in Netscape Navigator and Internet Explorer.

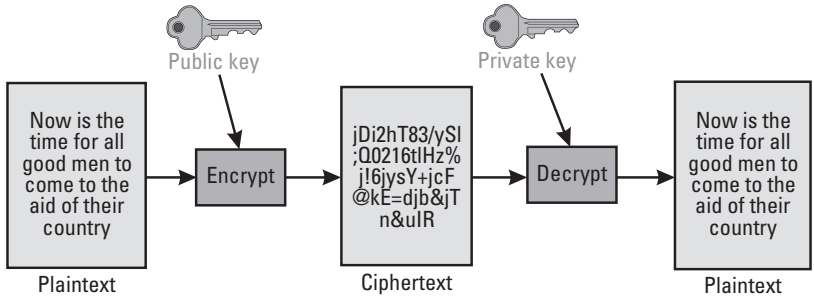

Figure 3.4 Asymmetric key encryption.

Asymmetric Key Encryption

For centuries, all cryptography was based on the symmetric key cryptosystems. Then in 1976, two computer scientists, Whitfield Diffe and Martin Hellman of Stanford University, introduced the concept of asymmetric cryptography. Asymmetric cryptography is also known as public key cryptography. Public key cryptography uses two keys as opposed to one key for a symmetric system. With public key cryptography there is a public key and a private key. The keys' names describe their function. One key is kept private, and the other key is made public. Knowing the public key does not reveal the private key. A message encrypted by the private key can only be decrypted by the corresponding public key. Conversely, a message encrypted by the public key can only be decrypted by the private key. This process is illustrated in Figure 3.4.

With the aid of public key cryptography, it is possible to establish secure communications with any individual or entity when using a compatible software or hardware device. For example, if Alice wishes to communicate in a secure manner with Bob, a stranger with whom she has never communicated before, Alice can give Bob her public key. Bob can encrypt his outgoing transmissions to Alice with Alice's public key. Alice can then decrypt the transmissions using her private key when she receives them. Only Alice's private key can decrypt a message encrypted with her public key. If Bob transmits to Alice his public key, then Alice can transmit secure encrypted data back to Bob that only Bob can decrypt. It does not matter that they exchanged public keys on an unsecured network. Knowing an individual's public key tells you nothing about his or her private key. Only an individual's private key can decrypt a message encrypted with his or her public key. The security breaks down if either of the parties' private keys is compromised.

While symmetric key cryptosystems are limited to securing the privacy of information, asymmetric or public key cryptography is much more versatile. Public key cryptosystems can provide a means of authentication and can support digital certificates. With digital certificates, public key cryptosystems can provide enforcement of nonrepudiation. Unlike symmetric key cryptosystems, public key allows for secure spontaneous communication over an open network. In addition, it is more scalable for very large systems (tens of millions) than symmetric key cryptosystems. With symmetric key cryptosystems, the key administration for large networks is very complex. Table 3.3 summarizes the advantages and disadvantages of the public key cryptosystems.

Public Key Cryptosystems

There are three public key algorithms in wide use today—Diffie-Hellman; RSA; and the Digital Signature Algorithm (DSA). They are described in the following sections.

Diffie-Hellman

The Diffie-Hellman algorithm was developed by Whitfield Diffie and Martin Hellman at Stanford University. It was the first usable public key algorithm. Diffie-Hellman is based on the difficulty of computing discrete logarithms. It can be used to establish a shared secret key that can be used by two parties for symmetric encryption. Diffie-Hellman is often used for IPSEC key management protocols.

For spontaneous communications with Diffie-Hellman, two communicating entities would each generate a random number that is used as their private keys. They exchange public keys. They each apply their private keys

Table 3.3
The Advantages and Disadvantages of Public Key Cryptography

Advantages	Disadvantages
No secret sharing necessary	Slower or computationally intensive
Authentication supported	Certificate authority required
Provides nonrepudiation	
Scalable	

to the other's public key to compute identical values (shared secret key). They then use the shared secret key to encrypt and exchange information.

Rivest, Shamir, Adelman (RSA)

The RSA public key algorithm was developed by Ron Rivest, Adi Shamir, and Len Adelman at MIT. RSA multiplies large prime numbers together to generate keys. Its strength lies in the fact that it is extremely difficult to factor the product of large prime numbers. This algorithm is the one most often associated with public key encryption. The RSA algorithm also provides digital signature capabilities. I will discuss digital signatures later in this chapter. They are used in SSL to set up sessions and with privacy-enhanced mail (PEM) and PGP. SSL is discussed in Chapter 5. PEM and PGP are discussed in Chapter 6.

Digital Signature Algorithm

DSA was developed as part of the Digital Signature Standard (DSS). (A more detailed discussion of DSS and DSA is provided later in this chapter.) Unlike the Diffie-Hellman and RSA algorithms, DSA is not used for encryption but for digital signatures.

A Slight Digression

For many years it was believed that Whitfield Diffie and Martin Hellman were the first to conceive of asymmetric cryptography and that Ron Rivest, Adi Shamir, and Len Adelman were the first to develop the RSA algorithm. However, it is now claimed that neither collaborative was the first and that the concept of asymmetric cryptography, the Diffie-Hellman algorithm, and the RSA algorithm were all discovered years earlier in England by the Government Communications Headquarters (GCHQ), which is the British equivalent of the NSA. The GCHQ claims that it conceived of the concept years before anyone else but never released information on the work for national security reasons.

Message Integrity

To attain a high level of confidence in the integrity of a message or data, a process must be put in place to prevent or detect alteration during transit. One technique employed is called a hash function. A hash function takes a message of any length and computes a product value of fixed length. The

product is referred to as a "hash value." The length of the original message does not alter the length of the hash value. Hash functions are used to ensure the integrity of a message or file. Using the actual message or file, a hash function computes a hash value, which is a cryptographic checksum of the message. This checksum can be thought of as a fingerprint for that message. The hash value can be used to determine if the message or file has been altered since the value was originally computed.

Using e-mail as an example, the hash value for a message is computed at both the sending and receiving ends. If the message is modified in anyway during transit, the hash value computed at the receiving end will not match the value computed at the sending end. Hash functions must be one way only. In other words, there should be no way to reverse the hash value to obtain information on the message. Obviously, this would represent a risk.

Another requirement of an effective one-way hash function is that the possibility of "collisions" is very limited, if nonexistent. A collision occurs when the same hash value is computed for two or more unique messages. If the messages are different the hash values should be different. No two unique messages should compute the same hash value. Table 3.4 lists some of the more widely implemented hashing algorithms.

MD4

MD4 was developed by Ron Rivest of RSA. MD4 is a one-way hash function that takes a message of variable length and produces a 128-bit hash value or message digest. MD4 has been proven to have weaknesses. Analysis has shown that at least the first two rounds of MD4 are not one-way (there are three rounds in MD4) and that the algorithm is subject to collisions.

MD5

MD5 was also created by Ron Rivest as an improvement on MD4. Like MD4, MD5 creates a unique 128-bit message digest value derived from the

Table 3.4
Widely Used Hashing Algorithms

Message digest #4 (MD4) from RSA
Message digest #5 (MD5) from RSA
Secure hash algorithm-1 (SHA-1)
RACE Integrity Primitives Evaluation (RIPE) MD-160 (RIPEMD-160)

contents of a message or file. This value, which is a fingerprint of the message or file content, is used to verify the integrity of the message's or file's contents. If a message or file is modified in any way, even a single bit, the MD5 cryptographic checksum for the message or file will be different. It is considered very difficult to alter a message or file in a way that will cause MD5 to generate the same result as was obtained for the original file.

While MD5 is more secure than MD4, it too has been found to have some weaknesses: Analysis has found a collision in the compression function of MD5, although not for MD5 itself. Nevertheless, this attack casts doubts on the whether MD5 is truly a collision-resistant hash algorithm.

The MD5 algorithm is intended for digital signature applications, where a large file must be "compressed" in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as RSA.

Secure Hash Algorithm-1 (SHA-1)

SHA-1 is a one-way hash algorithm used to create digital signatures. SHA-1 is derived from SHA, which was developed in 1994 by the NIST. SHA-1 is similar to the MD4 and MD5 algorithms developed by Ron Rivest. SHA-1 is slightly slower than MD4 and MD5, but it is reported to be more secure.

The SHA-1 hash function produces a 160-bit hash value or message digest. I am aware of no known cryptographic attacks against SHA-1 that have been successful. Since it produces a 160-bit message digest it is more resistant to brute force attacks than MD4 and MD5, which produce a 128-bit message digest.

RIPEMD

RIPEMD is a hash function that was developed through the European Community's project RIPE. There are several extensions to RIPEMD1RIPEMD-128, RIPEMD-160, and RIPEMD-256. Each extension is a reference to the length of the hash value or message digest. For example, RIPEMD-160 is a 160-bit cryptographic hash function, designed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel.

Authentication

To have a high level of confidence and trust in the integrity of information received over a network, the transacting parties need to be able to

authenticate each other's identity. In the example involving Alice and Bob, it was demonstrated how they could transmit secure information between each party using encryption by exchanging public keys. While confidentiality was ensured with the use of public key cryptography, there was no authentication of the parties' identities. Bob may not really have been Bob. For that matter, Bob doesn't really know if Alice was Alice. In addition, how does Alice know that when she was sending her public key to Bob, that Jack did not intercept it and use it to send his public key to her and masquerade as Bob. To ensure secure business transactions on unsecured networks like the Internet, both parties need to be able to authenticate their identities. Authentication in a digital setting is a process whereby the receiver of a message can be confident of the identity of the sender. The lack of secure authentication has been a major obstacle in achieving widespread use of the Internet for commerce. One process used to authenticate the identity of an individual or entity involves digital signatures.

Digital Signatures

A digital signature allows a receiver to authenticate (to a limited extent) the identity of the sender and to verify the integrity of the message. For the authentication process, you must already know the sender's public key, either from prior knowledge or from some trusted third party. Digital signatures are used to ensure message integrity and authentication. In its simplest form, a digital signature is created by using the sender's private key to hash the entire contents of the message being sent to create a message digest. The recipient uses the sender's public key to verify the integrity of the message by recreating the message digest. By this process you ensure the integrity of the message and authenticate the sender. Figure 3.5 illustrates the process.

To sign a message, senders usually append their digital signature to the end of a message and encrypt it using the recipient's public key. Recipients decrypt the message using their own private key and verify the sender's identity and the message integrity by decrypting the sender's digital signature using the sender's public key.

Once again we will use Alice and Bob to illustrate how digital signatures work. Alice has a pair of keys, her private key and her public key. She sends a message to Bob that includes both a plaintext message and a version of the plaintext message that has been encrypted using her private key. The encrypted version of her text message is her digital signature. Bob receives the message from Alice and decrypts it using her public key. He then compares the decrypted message to the plaintext message. If they are identical, then

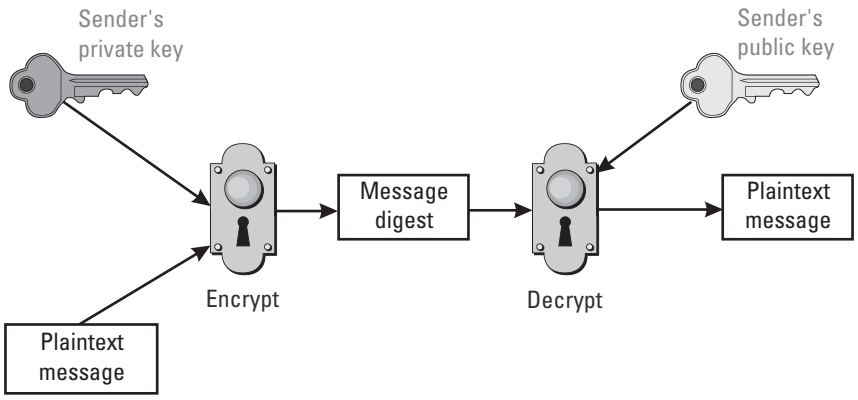

Figure 3.5 Digital signature.

he has verified that the message has not been altered and that it came from Alice. He can authenticate that the message came from Alice because he decrypted it with Alice's public key, so it could only have been encrypted with Alice's private key, to which only Alice has access.

The strengths of digital signatures are that they are almost impossible to counterfeit and they are easily verified. However, if Alice and Bob are strangers who have never communicated to each other before, and Bob received Alice's public key, but had no other means to verify who Alice was, other than Alice's assertion that she was who she claimed to be, then the digital signature is useless for authentication. It will still verify that a message has arrived unaltered from the sender, but it cannot be used to authenticate the identity of the sender. In cases where the parties have no prior knowledge of one another, a trusted third party is required to authenticate the identity of the transacting parties.

Competing Standards

There are two competing standards for digital signature technology. Both systems are based on the International Telecommunications Union's X.509 standard for public key certification. The one that has been around the longest is the RSA Data Security's public key encryption standard, which has become a de facto standard in the industry. RSA Data Security uses the RSA public key algorithm, for both encryption and authentication, invented by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977. The more recently developed standard is the U.S. government's DSS, which specifies a DSA. It

was selected by the National Institute of Standards and Technology (NIST) in 1994.

Many have questioned the wisdom of the NIST's decision to select DSS. Not surprisingly, one of the most vocal opponents has been RSA Data Security and companies associated with RSA. However, many others have questioned the choice of DSS. The DSS cryptosystem is relatively new and has not been fully tested. For that reason alone, many believe that it is not as secure as the RSA standard, which has been subjected to rigorous testing for the past 19 years. Some have even questioned the NIST's motives for selecting DSS. The decision was made in cooperation with the NSA. The process was secretive and conducted with very little public participation or debate. Some have gone so far as to suggest that DSS was selected because the NSA has a back door into the system. While the competing standards do not represent an obstacle to implementing digital signatures within a large multinational organization, they can result in the inability to exchange digital signatures between organizations.

Digital Certificate

Digital signatures can be used to verify that a message has been delivered unaltered and to verify the identity of the sender by public key. The problem with authenticating a digital signature, however, is that you must be able to verify that a public key does in fact belong to the individual or entity that claims to have sent it and that the individual or entity is in fact who or what it claims to be.

A digital certificate issued by a certification authority (CA) utilizing a hierarchical public key infrastructure (PKI) can be used to authenticate a sender's identity for spontaneous, first-time contacts. Digital certificates provide a means for secure first-time spontaneous communication. A digital certificate provides a high level of confidence in the identity of the individual or entity with which you are communicating. A digital certificate is a means to authenticate identity.

A digital certificate is usually issued by a trusted/known third party (CA) to bind an individual or entity to a public key. The digital certificate is digitally signed by the CA with the CA's private key. This provides independent confirmation that an individual or entity is in fact who it claims to be. The CA issues digital certificates that vouch for the identities of those to whom the certificates were issued.

Using Alice and Bob as our example, Alice can send Bob her public key. Bob will be able to verify her digital signature using Alice's public key.

Given such a key, how does he verify that it actually belongs to Alice and does not really belong to Jack who is masquerading as Alice? If he has no other means available to him, he cannot. However, if Alice's public key is presented as part of a digital certificate signed by a known CA, Bob can have a high level of confidence that Alice is who and what she claims to be.

A digital certificate is a method of binding an individual or entity to a public key. The certificate is digitally signed by a CA providing independent confirmation that individuals or entities are in fact who they claim to be and that the public key provided by them does in fact belong to that party. The CA and the CA's public key must be widely known for the digital certificate to be of practical value. The CA's public key must be widely known so that there is no need to authenticate the CA's digital signature. You are relying on the CA's digital signature to authenticate the certificate owner's identity and to bind that identity to their public key.

Table 3.5 illustrates possible contents of a digital certificate. In its simplest form, a digital certificate would include the name and address of the individual/entity, certificate expiration date, serial number for the certificate, and the individuals' or entities' public key. Most importantly, it would be digitally signed by the issuing CA using the CA's private key. A digital certificate could contain other information as well. Depending on the type of certificate, it could include information on access privileges, geographic location of the owner, or the age of the owner. When fully implemented, digital certificates will most likely take several forms.

Each person's digital certificate could contain a mini-database on the owner, which includes the authorizations, access privileges, and the owner's public key. Digital certificates cannot be forged and are expected to be legally acceptable as handwritten notarized signatures. The International Chamber of Commerce is exploring the creation of a "cybernotary," a lawyer able

Table 3.5
Digital Certificate Example

Name: Individual, organization, entity
Owner's public key
Certificate expiration date
Certificate's serial number
Name of issuing CA
Issuing CA's digital signature

to demonstrate that he or she can issue certificates from a secure computer environment.

A digital signature used in concert with a digital certificate potentially possesses greater legal authority than a handwritten signature. The inability to forge a digital signature, the fact that the digital signature can verify that the document has not been altered since it was signed, and the certificate verifying the identity of the signer make a digitally signed document irrefutable. The signer cannot repudiate his or her signature at a later date.

Limitations of Digital Certificates

There are still a number of issues that need to be addressed, such as how to handle expired certificates; there is the risk that a long-term document could be signed with a digital certificate with a two-year expiration date. What is the legality of the document once the digital certificate expires? Another issue that needs to be addressed is how to handle revocation of certificates.

The certificate revocation process is cumbersome: How do you revoke a certificate once it has been issued? Once a digital certificate is issued, it is valid until it expires. That is usually at least a year. No process exists for immediate revocation of a certificate should it be compromised or should the CA withdraw its certification. CAs will have to periodically issue certificate revocation lists (CRL). All participants utilizing the PKI will have to maintain up-to-date CRLs. CRLs will eventually become very large. In addition, there are a number of issues concerning the legal responsibilities and liabilities of CAs and their issuing of digital certificates that still need to be addressed.

What is most crucial to the success of the digital certificate is the role of the CA. With the CA, the trust is no longer dependent on the individual's or entity's digital signature. Instead, the trust is transferred to the CA.

Certificate Authorities

As stated previously, a CA is a body, either public or private, that seeks to fill the need for a trusted third party in e-commerce. The CA issues digital certificates that vouch for the identities of those to whom the certificates were issued. For this process to be secure, the CA's public key must be trustworthy and well-known. When I say it must be trustworthy, I am referring to the reputation and reliability of the CA as an entity. A digital certificate issued by "Sam's Digital Certificates and Deli" would lack trustworthiness to another

party on the Internet. A CA must also perform the necessary due diligence to verify that individuals or entities are in fact who they say they are, before a digital certificate is issued to an individual or entity. The CA public key must be widely known to be effective. A digital certificate signed by a CA is worthless if you do not know the CA's public key or if you have no independent means of verifying that the public key provided is in fact bound to the CA. For that reason, a CA's public keys need to be easily accessible and verifiable.

There will be a number of entities that issue digital certificates. VeriSign, Inc., which was formed by RSA Data Security and several other major corporations, is the one main issuers. Other companies that issue digital certificates include GTE, AT&T, and Microsoft. There are many others. The process of obtaining a digital certificate is relatively simple for any legitimate individual or entity.

Once again, using Alice and Bob for our example, Alice generates her own key pair from her X.509-compliant software or device. She then sends the public key to a CA with proof of who and what she is. In our example, Alice sends her public key to a CA. If the digital certificate is for her company, the CA might request a copy of the articles of incorporation, copies of the latest financial statements, and other items that establish that the company is what it claims to be and is in good standing. If the certificate is for Alice personally, the CA could request a birth certificate and perhaps take her fingerprints. The verification process is largely dependent on the level of the certificate. Once the CA has done its due diligence and is satisfied that Alice is who she claims to be, the CA sends her a digital certificate to load in her software or device. This certificate will be signed by the CA with its private key. The digital certificate will attest to the fact the CA has determined that Alice is who she says she is and binds to Alice her public key. Alice can now present that certificate to Bob to authenticate her identity and her public key.

When Bob receives Alice's signed message, he will need Alice's public key to verify her digital signature and to ensure that the message has arrived unaltered. Since he already knows the CA's public key (it will be published everywhere), he can decrypt the digital certificate or certify that the digital certificate is signed by the CA, verify the integrity of the certificate, and obtain Alice's public key and then decrypt her signed message.

The need for CAs is clear, but the duties and responsibilities of the CAs are not so clear. There are still many issues that need to be addressed with CAs. Many of these are legal, not technical, in nature: What are the CA's responsibilities when issuing digital certificates? What if the CA makes a mistake and issues one to the wrong individual or entity? CAs may be open to tremendous liability should that mistake result in fraud or some financial loss.

As we move closer to paperless commerce and a paperless society, the concept of CAs becomes increasingly important. They will have a major impact on the future of e-commerce. That impact will affect our day-to-day lives: It means the development of a whole new set of business relationships that will be necessary to function daily. Perhaps, one day, without a digital certificate you may not be able to purchase milk at the corner store. Will CAs become the future's credit agencies, rating everyone as a good or bad "risk"?

Public Key Infrastructure

As part of the future implementation of digital certificates, a movement is under way to develop a PKI. The infrastructure will be necessary to authenticate digital certificates and CAs. A PKI is a hierarchical network of CAs. A "root certificate" authority certifies subordinate CAs. The hierarchy is recognized as trusted by all entities that trust the hierarchical CA. Not every entity needs to trust the other, just the hierarchy. Some plans envision a hierarchy of CAs, where one CA certifies the identity of the previous CA. The top-level root CA in the United States could be the U.S. government. Others envision a more horizontal scheme of cross-certification with only a few layers. In either case, a certificate-based PKI can provide a process to establish trust relationships. Figure 3.6 illustrates how a theoretical PKI might be structured.

The difficult part will be developing the standards and infrastructure for certifying digital signatures and certificates between organizations using different schemes. At the same time, the NIST is working on the development of a federal PKI. While there are many challenges to developing a national PKI, the most daunting task will be the development of the global infrastructure. When we discuss a global or international PKI we open a Pandora's box of "national security" issues.

The Future

Introduction

It is stating the obious to say that developments in the field of cryptography are occurring all the time. Two of the more important changes involve the Advanced Encryption Standard and developments in the area of Elliptic Curve Cryptography.

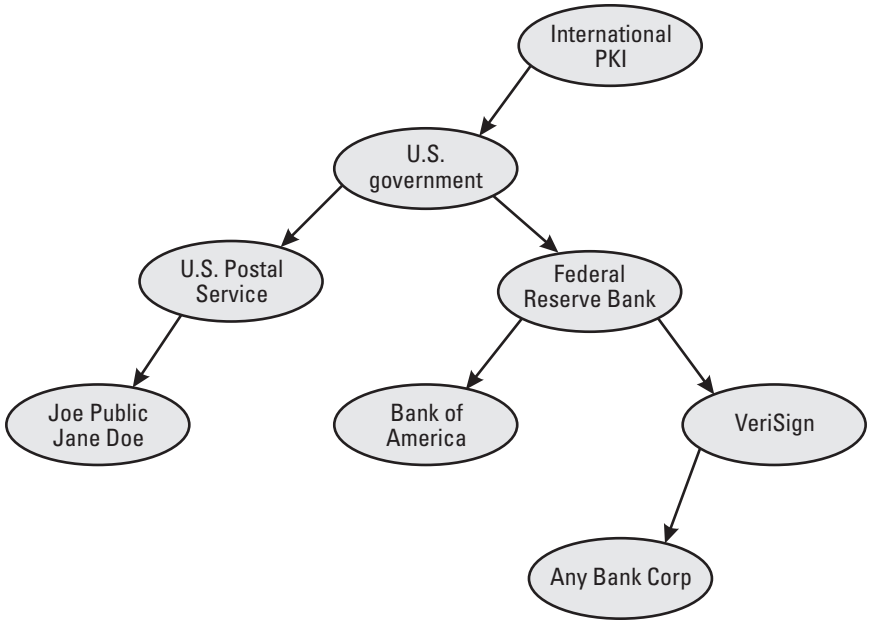

Figure 3.6 Theoretical PKI.

Advanced Encryption Standard (AES)

For decades the encryption standard in the United States has been DES. However, the DES algorithm is no longer as secure as it once was and needs to be replaced. As a result, the NIST is in the process of selecting a new algorithm to use as the new standard into the next century. This new standard is being called the Advanced Encryption Standard (AES). The goal of AES is to select an unclassified, block algorithm that will be available worldwide free of royalty fees.

As of this writing, there are five algorithms that have been selected as finalists for the AES. The five finalists are listed as follows.

- MARS developed by IBM;
- RC6, developed by RSA, which also developed RC4 and RC5;
- Rijndael developed by Vincent Rijmen and Joan Daemen;
- Serpent developed by Ross Anderson, Eli Buham, and Lars Knudsen;

- Twofish developed by Bruce Schneier, Niels Ferguson, Chris Hall, John Kelsey, Doug Whiting, and David Wagner. Incidentally, Bruce Schneier is the author of the book *Applied Cryptography* and developer of the Blowfish block algorithm.

One or more of these algorithms will eventually be selected as the AES. How long it will remain secure will largely depend on the developments in the fields of computer technology and cryptanalysis.

Elliptic-Curve Cryptography (ECC)

Another up and coming development in cryptography appears to be elliptic-curve cryptography (ECC). ECC, which is widely expected to be the next-generation algorithm, has been proposed for use as a public key cryptosystem. EEC's strength comes from the fact that it is computational very difficulty to solve the elliptic curve discrete logarithm problem.

The appeal of ECC algorithms is the fact that they hold the possibility of offering security comparable to the RSA algorithms using smaller keys. Smaller keys mean that less computation is required. Less time and CPU resources would be required to implement this technology on the network. Less time and CPU translates into less cost associated with using these algorithms. As a result, interest in these algorithms is keen.

It has also been said that ECC is more difficult to break than RSA. While both RSA with a 512-bit key and ECC with a 97-bit key have been broken, it has been stated that the ECC algorithm is more difficult to break. In 1999 a team of 195 volunteers in 20 countries using 740 computers took 40 days to recover the 97-bit ECC private key.

Although ECC holds great promise, I am not aware of any practical implementation of the technology in any product now on the market. No matter what algorithm you employ, it is important to be cognizant of the fact that as computing power increases and becomes less expensive, the cryptographic key sizes will have to increase to ensure security. Not too far in the future, a 2,024-bit key will not be sufficient to ensure security.

The Limitations of Encryption

Communications are not necessarily secure simply because they are encrypted. It is important to remember that useful information can even be discerned from encrypted communications. I like to use an example from the book *Blind Man's Bluff.* In the book, authors Sherry Sontag and Chistopher

and Annette Drew tell the story of U.S. submarine espionage during the Cold War. In the 1970s and 1980s, Soviet missile subs were using effective cryptosystems in conjunction with sophisticated transmitters that compressed their encrypted communications into microsecond bursts. While the United States was not able to break the Soviet transmission code, America was able to gather a great deal of information from the transmissions themselves. U.S. analysis of the transmission patterns revealed almost as much information as the actual content of the transmissions would have revealed. For example, the United States was able to determine that the messages were coming from Soviet subs on their way to and from patrol. They were also able to distinguish one sub from another by slight variations in the frequencies of the transmissions and that the Soviet subs sent transmissions at regular points or milestones in their patrols. Consequently, the United States was able to determine Soviet subs' location, when they reached their patrol sector, the halfway point or a particular landmark. The analysis of the transmission patterns enabled the United States to track Soviet subs on patrol without ever breaking the transmissions' code. It is important to understand that simply using encryption is no guarantee of confidentiality or secrecy.

In addition, studies have shown that the randomness of the data for encrypted files stored on media can be used to distinguish the files from other stored data. Generally, operating systems do not store data in a random manner. Data is normally stored in a manner that optimizes retrieval, space, or speed. Encrypted files and algorithm keys by their nature must be random data. As a result, when large encrypted files and public/private key sets are stored on a disk drive their randomness stands out against the normally organized data on the drive. There are programs available that purport to be able to find keys and encrypted files on a disk drive. If true, this could potentially mean that someone could steal key pairs if he or she had access to the drive on which the keys were stored.

Meanwhile, however, it is also important to understand that developments in the field of cryptography and digital signature technology are the enabling force behind the recent explosion in e-commerce on the Internet. Without these technologies, Internet e-commerce would not be possible. As a result, those who want to participate in this new world of e-commerce, either as an entrepreneur or a consumer, need to understand the essential technology that is enabling its development.