

Cryptography and Cyberwar

THROUGHOUT HISTORY, military commanders and national leaders have needed to communicate securely. Since ancient times, they have used various techniques to help ensure that enemies could not read intercepted communications. Thus, cryptography was born. Cryptography, or secure communication in the presence of those who should not be able to receive the communication, typically relies on ciphers. These are methods or algorithms that make the information hidden or keep it secret. The processes used have ranged from simple letter substitution like that used in children's secret codes to complex encryption systems that required prior knowledge of a secret key to decode messages.

In modern cyberwar, cryptography is still used for the same purposes as those ancient ciphers, but on a far greater scale and with significantly more resistance to being broken. Modern cryptographic systems are far more complex, involving fast computers, huge prime numbers, advanced algorithms, and techniques that allow individuals and systems that have had no prior contact with each other to create secure connections. The broad use of cryptography in its many forms is a key part of cyberwarfare. Understanding how it works, where it can be used, and how it can be attacked is critical to both computer network attack and computer network defense strategies.

Cyberwarfare now involves both attackers and defenders using cryptographic systems and attacks aimed at defeating them. Attackers in cyberwarfare frequently target data stored on systems and in transit across networks. This means that defenders must try to protect that data from being acquired or modified throughout its life cycle. Defenders also must understand the encryption systems they put in place and how to defend the systems themselves. To do this, they must ensure that they are using secure cryptographic systems, which has proven to be a challenge despite their best efforts.

That challenge exists because attackers now target the underlying cryptographic algorithms, software, and even the hardware used for encryption. Publicized information indicates that organizations at the nation-state level have actively worked to weaken the ways that important parts of widely used encryption algorithms function—to make them easier to break. Other attackers have focused on gaining access to the software used to protect data so they can reverse engineer it and find areas where it may be weak.

Attackers have also begun to use encryption as a weapon in its own right. Encrypting data in a way that prevents legitimate users from accessing it allows attackers to negotiate for the victims to perform an action, or to choose when and where the data is available. The world of connected computers and networks now allows attackers to seize what would have been the contents of file cabinets and vaults and put that data in a virtual, unbreakable safe that only they can provide access to.

This chapter introduces important cryptographic concepts and some important historical cryptosystems as well as modern encryption algorithms that are currently widely used. It then explores how cryptography is used by both attackers and defenders in cyberwar—including how nation-states seek to influence for their own advantage the way in which cryptographic systems are created. Next, the chapter discusses how to defeat attacks on cryptographic systems and how malware writers and cyberwarriors have weaponized them. Finally, it looks at the future of cryptography in cyberwar as combatants attack and defend, and as next-generation technologies and techniques become available.

Chapter 10 Topics

This chapter covers the following topics and concepts:

- What the basics of cryptography are
- How cryptography is used in cyberwar
- How cryptanalysis attacks cryptography
- How to defeat attacks on cryptographic systems
- How cryptography is weaponized
- What the future of cryptography in cyberwarfare is

Chapter 10 Goals

When you complete this chapter, you will be able to:

- Explain and describe common encryption concepts
- Explain how encryption has been used in cyberwarfare
- Describe methods used to attack encryption
- Describe effective encryption practices
- Describe uses of encryption for data and communications security

An Introduction to Cryptography

Cryptography is the study and practice of techniques for secure communication that is protected from adversaries. Cryptography in cyberwarfare often involves **encryption**, which is a process that encodes information in a way that prevents unauthorized parties from reading the data. Of course, data that is encrypted needs to go through the **decryption** process, which removes the encoding, allowing authorized parties to read the original text.

Cryptographic systems have existed for thousands of years. In fact, one of the earliest documented ciphers dates back to Roman times. Julius Caesar used what is now called the Caesar cipher, a simple cipher that encrypted messages by simply shifting letters to the right by the same number of letters every time. Thus, if adversaries captured his communications, they appeared nonsensical. Over time, adversaries figured out the encryption methods in use, so more advanced techniques were developed and used.

Of course, cryptographic techniques include more than encryption. One early technique for hiding messages reportedly required the messenger to shave his head. Once the messenger's head was bald, the message would be tattooed onto his scalp. When his hair regrew, he would be sent with the message. Recovering the message was as simple as shaving the messenger's head and reading it. Obviously, this wasn't a great method to transport messages in a timely fashion—as you would have to wait for the messenger's hair to grow back before sending the message—but it did conceal them well. Few opponents think to shave a captured enemy's head, and messages on the back of your head underneath your hair would be very hard to read.

NOTE

Cryptography includes more than just encryption. Most modern cyberwarfare uses involve encryption, so encryption is a main focus throughout most of this chapter. Despite that, as you read this chapter, remember that encryption is but one cryptographic technique—not the only one.

Advances from simple ciphers and other cryptographic techniques like these have continued since those early developments. The need to send secure messages has been a constant, and the ability to conceal meaning in written messages was necessary for communications in warfare and in trade. In fact, the cryptographic arms race has been a fact of life for military, political, and commercial reasons for thousands of years.

Modern cryptographic systems are far more advanced than the systems used in Julius Caesar's time, or even those that were used in World War II by both the Axis and the Allies. Computer systems and advanced mathematical algorithms use very large prime numbers, fast processors, and combinations of complex techniques to secure data from prying eyes. The Internet has created a need for very fast, open, and secure encryption systems that people can use in their daily lives. In fact, most Internet users use encryption every day without realizing it as they log on to Web sites and as their traffic is carried from service provider to service provider across the Internet.

Both computer network attacks and computer network defense involve cryptographic systems on a regular basis. Attackers may try to break through encryption, weaken encryption systems or inputs into the encryption algorithms that defenders use, or even use encryption itself as a means of attack. Defenders use encryption to protect data at rest and during transit. Further, defenders choose from a host of available options when selecting how strong their encryption is, how and when they use it, and how users interface with it.

Cryptographic Concepts

The term *cryptography* covers a range of techniques for secure communication, all with the same goal: to protect communication from adversaries. It relies on four major concepts:

- *Confidentiality*, or the ability to ensure that data is not exposed to those who should not see it. Encryption systems that cannot be broken by opponents can ensure confidentiality.
- *Data integrity*, which is the ability to ensure that data has not been modified, either by addition or removal of data, or by modifying the data's meaning. Message integrity is enforced by values known as message digests or checksums, which prove that the message has not changed by using algorithms to compare their original and received versions.
- *Authentication*, or the ability to verify the identity of a sender.
- *Nonrepudiation*, which requires that it be possible to prove that the sender did send the file or message. Nonrepudiation means that the sender cannot claim that someone falsely sent the message posing as the sender.

Many modern cryptographic systems offer all of these abilities by providing a means to encrypt a message; the ability to include a digital signature in communications that proves who the sender is; and a means to check that the data is intact, has not been seen by unintended recipients, and that it can be proved that the sender actually sent it.

Ciphers and Encryption

In cryptography, encryption is the process for encoding information such as a message or file so that only authorized parties can read it. This is accomplished by using an encryption algorithm or process to encode the original message, and to decode the encrypted message when it arrives. This section examines encryption concepts and terms, and then explores the basics of the two major types of encryption algorithms: symmetric and asymmetric ciphers. It also examines the concept of cryptographic hashing, a technique used to ensure that data has not been changed by unauthorized parties.

Key Encryption Terms

As you discover how encryption is used in cyberwar, you must become familiar with some common terms. Keep the following terms in mind as you read about encryption systems throughout this chapter:

- Codes are often confused with ciphers, but aren't ciphers. A **code** is, however, a cryptographic system that substitutes values or words for other words. Morse code is an excellent example of a code that substitutes a series of dots and dashes for letters, and most people are familiar with the police use of codes like "10-4" from movies and television. Both codes are commonly available, and don't provide any confidentiality, message integrity, or authentication.
- **Plaintext** is the original data prior to encryption.
- **Ciphers** are encryption algorithms that are applied to plaintext. Unlike codes, ciphers are intended to provide confidentiality and message integrity, and some provide capabilities for authentication and nonrepudiation.
- **Ciphertext** is the output of a cryptographic algorithm.
- A **key** is the secret variable in an encryption algorithm. In modern encryption, keys are typically very large prime numbers. Keys exist in a *key space*, which is the range of all possible values that a key can have and remain valid for the purposes of the encryption algorithm. In modern digital encryption systems, a key space is typically measured in bits, or the number of binary 1s and 0s that it contains. A key doubles in complexity for every additional bit larger it becomes.
- A **cryptosystem** is the hardware and software implementation of a cipher.

NOTE

The goals of cryptography are very similar to the C-I-A triad of confidentiality, integrity, and availability, but they aren't the same. Be careful as you review the goals, as authentication and availability can be very easy to confuse.

► **NOTE**

You will explore hashes and checksums in more depth later in this chapter.

- **Checksums** are mathematical values calculated to check for common errors in data, and are quite fast.
- **Message digests** or **hashes** are cryptographic functions that protect the integrity of a message by allowing you to check that the message has not changed. Hashes are slower than checksums, but provide additional capabilities and features that checksums do not.

► **NOTE**

Because a symmetric cipher may be in use by many users, the security of the system is only as good as the practices of the least careful user. This can be a major issue for symmetric ciphers, as key distribution is difficult, and knowing if a key has been lost can be almost impossible if an attacker can intercept messages but only reads conversations without taking other action.

Symmetric Ciphers

Symmetric ciphers rely on a shared key. When a sender and recipient want to communicate, they use the same key for both encryption and decryption. This means that **symmetric encryption** can't provide authentication or nonrepudiation, as you cannot identify who the sender is based on the encryption method. In addition, symmetric encryption between many users typically uses the same key for all of them to avoid having to encode the message many times. If one of those users were to no longer be trusted, the symmetric key for all users would need to be changed for future communications.

Symmetric encryption can be very hard to break when sender and recipient are using a strong key that is handled responsibly. Perhaps more important, symmetric encryption is relatively fast compared with asymmetric encryption. Thus, symmetric encryption can be very useful in ongoing communications when a strong cipher requiring less processor time is desired.

TABLE 10-1 Symmetric cipher scaling.

NUMBER OF USERS	UNIQUE KEYS
2	1
3	3
4	6
5	10
10	45
100	4,950
1,000	499,500
100,000	4,999,950,000
1,000,000	499,999,500,000

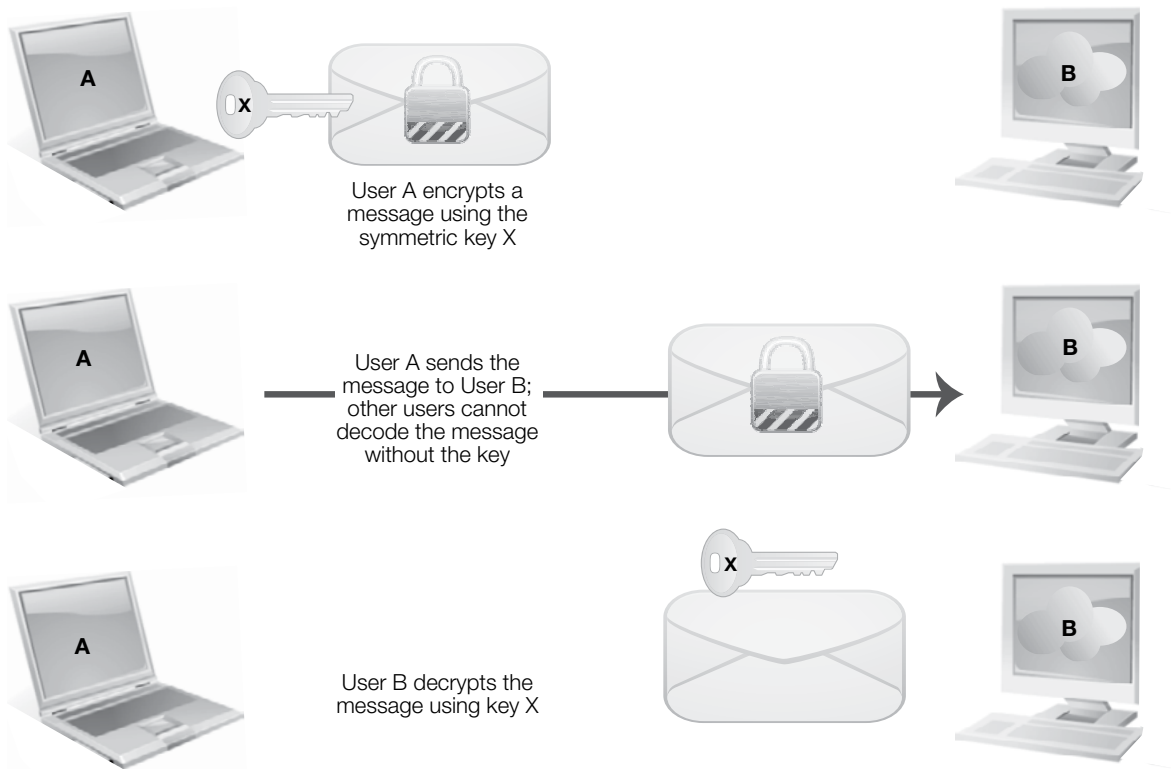
Symmetric cryptography has a number of problems that influence its use:

- The algorithm isn't scalable. Because any user with the key can decrypt any communication sent with it, security between individuals in large groups can only be maintained by having a key for each pair of individuals. As groups grow, this number grows to unsupportable numbers, as shown in Table 10-1. The formula to calculate the number of unique keys is $\text{keys} = n(n-1)/2$ where n is the number of users.
- **Key distribution** is a challenge because keys need to be distributed securely. Thus, if you don't already have a secure way to exchange keys, you can't ensure that the shared key won't be compromised when it's transferred. This is a chicken-and-egg problem, with the inability to create a secure transfer process also meaning that you can't transfer the means to build one safely.
- Symmetric encryption doesn't handle changes in trust relationships well. When one person who has a shared symmetric key leaves a group, that entire group of users must generate a new key. If the key is exposed, the same thing must occur, potentially resulting in the creation of as many new keys as there are users who use them to communicate with the person whose key was exposed.
- Symmetric encryption doesn't provide nonrepudiation. You cannot prove that a symmetric key was used by the expected owner. In fact, every communication has at least two users who could have sent the message. Thus, although symmetric encryption can provide useful advantages, it misses an important use case for many cryptographic exchanges.

Figure 10-1 shows how a typical message is sent between User A and User B. User A encrypts the message using his or her shared secret key and transmits the message to User B. User B then uses the same shared secret to decrypt the message. Bear in mind that User B has no way to prove that User A is actually the person who sent the message, and is forced to rely on User A's secure handling of the shared key.

Enigma: Using and Breaking Wartime Symmetric Encryption

During World War II, Germany used an electromechanical symmetric encryption device called the Enigma device. The Enigma devices originally used by the Germans used three wheels to perform complex substitutions of letters very much like a series of Caesar ciphers. Each day, a codebook was used to set the initial settings of the rotors, and then various changes were made for each message. The Enigma actually used techniques similar to many cryptosystems today, including the concept of an initialization vector, or starting position for the encryption, and basic error detection, which was accomplished by sending the initial message settings twice.

**FIGURE 10-1**

Symmetric ciphers use the same key to encode and decode a message.

Decrypting the ciphertext based on this cipher required the recipient to know how the wheels were set when the message was encrypted. Thus, protecting the codebooks for Enigma devices was a critical part of the security of the Enigma encryption scheme. As the war went on, Germany added wheels and would issue Enigma devices with replacement wheels, which could be inserted into the devices, increasing the number of potential combinations. Some late-war versions of German Navy Enigma devices used as many as eight wheels!

Cracking Enigma was a complex task, as the number of potential combinations for each encrypted message was very large. Fortunately for the Allies, the way in which Enigma was used did have vulnerabilities, including the double sending of the initial setting and the use of a standard initial setting. The Allies leveraged initial breakthroughs in cryptanalysis to learn how to decrypt Enigma, but needed to be able to more quickly crack messages as they came in. Poland attacked this using early electromechanical computing devices called *bomba*. After the invasion of Poland, both Britain and the

FYI

The substitution of one letter for another is known as a *substitution cipher*, which is a very basic symmetric cipher. Substitution ciphers that perform only a single substitution have been included in familiar children's toys for decades in the form of secret decoder rings and other cereal box toys. Enigma's multiround substitution cipher—and additional variables due to other settings—increased the potential substitutions. Each additional sequential wheel resulted in 26 times more possibilities. This increase in possible outputs remains a goal when attempting to make encryption stronger, but modern cryptosystems typically achieve it by increasing key length in bits.

United States developed their own *bombes*, which were faster electromechanical computers that could crack Enigma. The U.S. Enigma-cracking device could run a three-rotor run for an Enigma-encrypted message in as little as 10 minutes.

In the end, the intelligence gathered from decrypted Enigma-encrypted messages allowed the Allies to sink German ships, to monitor German military research developments, and to better understand whether their operations against Germany and the Axis were successful. This use of early electromechanical computers was the predecessor for today's computer-based cracking of cryptosystems. It also demonstrated the power of cracking your enemy's encryption without your enemy being aware of it.

Asymmetric Ciphers

The problems with symmetric ciphers resulted in a desire to create a solution that would allow for a scalable key distribution system that could distribute keys safely and that could prove who senders were. Thus asymmetric key algorithms, also known as *public key algorithms*, were created. **Asymmetric encryption** relies on key pairs with a public and a private key used in the encryption algorithm. Each key is typically a very large prime number. The encryption algorithm's strength relies on the fact that determining which very large prime numbers were used in the algorithm is very difficult.

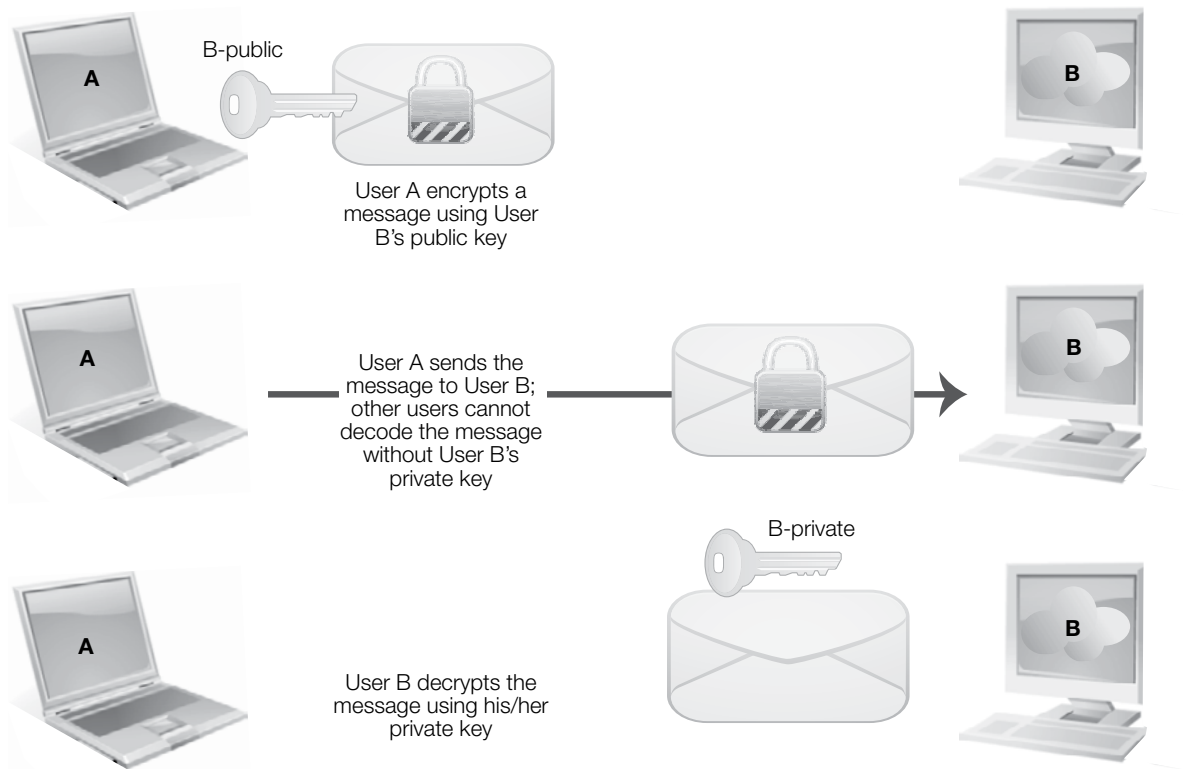
technical TIP

Identifying the two large prime numbers that were multiplied to create a final number is known as *factoring*. Some of the basic strength of current asymmetric encryption schemes comes from the difficulty of factoring the product of two large prime numbers.

If you compare the features of asymmetric cryptography with the issues of symmetric cryptography, you'll see that asymmetric cryptography handles them quite nicely:

- **Public key encryption** can provide all four critical requirements: confidentiality, message integrity, authentication, and nonrepudiation.
- Key distribution can be handled by simply distributing the public key for each user. Because the public key is intended to be distributed, and cannot be used except to encrypt a message to that user, it's safe to send or even to publicly distribute online.
- Unlike symmetric keys, asymmetric keys only require twice the number of keys as the number of users in the system, because each user needs a public and a private key. New users only have to publish their public key. As shown in Table 10-2, asymmetric ciphers scale well.
- Users can be removed easily because they have no knowledge of a shared key. Removing them simply requires revoking their keys.
- Compromise of a private key requires generating a new key pair and distributing the public key again. Because it's safe to distribute, the only problem is getting it into the hands of all users who need it.
- Distribution of keys doesn't require some form of existing secure communication. Because you send a public key, you can exchange keys safely without pre-existing knowledge. This makes public key encryption ideal for online encryption, such as during online shopping, where the participants have likely not encountered each other before and have no pre-existing way to securely communicate.

TABLE 10-2 Asymmetric cipher scaling.	
NUMBER OF USERS	UNIQUE KEYS
2	4
3	6
4	8
5	10
10	20
100	200
1,000	2,000
100,000	200,000
1,000,000	2,000,000

**FIGURE 10-2**

Asymmetric encryption systems use a recipient's public key to encode messages and the recipient's private key to decode messages.

Figure 10-2 shows the basic flow of an asymmetric encryption process. User A, the sender, encrypts a message using User B's public key. User A sends it, confident that only User B can decrypt the message. Upon receiving the message, User B uses his or her private key to decrypt the message. User B can then read the message.

The question of how to prove that a given public key is owned by the individual who claims to own it can appear challenging. Fortunately, solutions to this problem have been designed, too. Users sign a public key using their private key. The signatures are verified using those individuals' public keys, just as with messages. This means that trust is created by verifying that users whose keys are trusted have signed a public key. Therefore, the public key likely does belong to the person whom it appears to belong to.

In Figure 10-3, note the difference in the encryption process used by User A. User A uses his or her private key to digitally sign the message. This is something only User A can do because User A is the holder of the private key. Once User B receives the message, he or she can use User A's public key to verify the signature. This is done using a similar process

as was used to decrypt the message with his or her own private key. This process results in a message that meets all four of the desired capabilities of a cryptographic system: confidentiality, integrity, authentication, and nonrepudiation:

- Confidentiality is ensured because the message is encrypted.
- Integrity is provided by hashing the message and providing a means of proving that the hash hasn't changed.
- Authentication is provided by the digital signature, as only User A has the private key.
- Nonrepudiation is provided by the same method: No other user could have sent a message as User A, as no other user should have User A's private key.

Figure 10-3 shows how nonrepudiation can be provided in an asymmetric cryptosystem by using digital signatures. User A's private key provides the proof that User A was the one who sent the message.

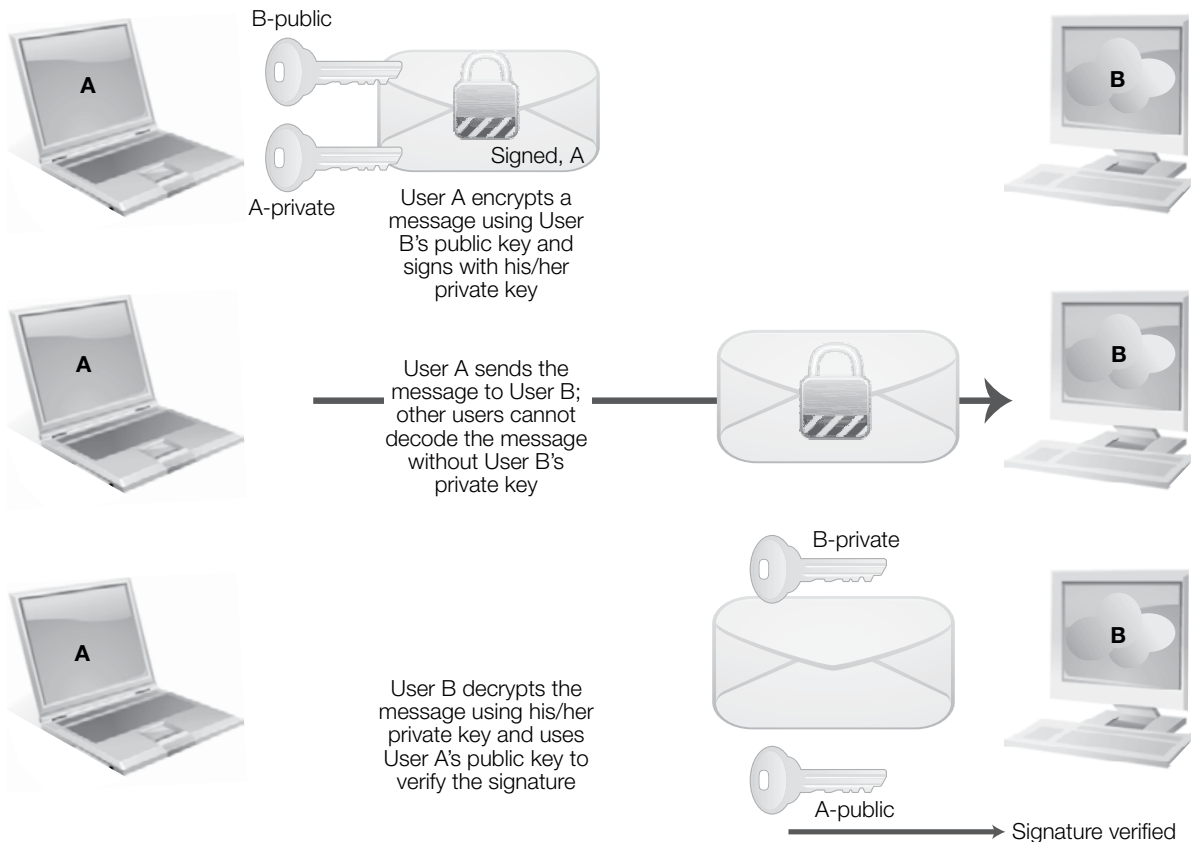


FIGURE 10-3

Users can use their private key to digitally sign a message, allowing recipients to verify their identity using the sender's public key.

FYI

Key-handling practices for asymmetric encryption systems are critically important. Often users who are unfamiliar with asymmetric key systems will accidentally send the private key, or both keys, to someone they want to communicate with. Once that happens, the private key should be considered compromised, and a new **key pair** must be generated. This means that the way you store a private key is really important, too. If attackers get access to private keys, they can send whatever messages they want as that user or service.

This entire system relies on the safety of the private keys of the individuals involved. It's normal in cryptographic systems like this to treat private keys with great care, and to require a strong password to use the private key. If keys are lost or are exposed, trust in the system is immediately lost. In cyberwar, the obvious strategy is to acquire private keys if possible, and to ensure that the target does not know that it has lost control of the private keys.

Steganography

Steganography, or the practice of concealing a message inside another message, picture, or file, is a topic that consistently comes up when discussing terrorists and insurrectionists who are attempting covert communication. Encrypted messages themselves are difficult to conceal, as ciphertext can be easily distinguished from normal plaintext. Thus, the ability to hide a message inside normally innocuous traffic is very desirable.

Much of the focus on modern digital steganographic techniques has involved concealing data inside images and other digital data. Specialized steganographic tools have been created that can add data to images, and, of course, detection tools known as *steganalysis tools* were created that can detect that same data when it's added. For years, rumors abounded that terrorist groups were hiding messages and attack plans in images posted to their Web sites and sent via e-mail.

Steganography isn't limited to just hiding text in images, however. Techniques for hiding encrypted messages in comments on Web sites, inside sound files and voice-over-IP conversations, and even simple ideas like hiding single frames in video or using text the same color as the background can all be considered a form of steganography.

Identifying files with steganographic data concealed in them can be difficult without access to the original file. Well-done steganographic techniques can appear like random noise or corruption in a file, and because the data is encrypted, it can appear truly random.

Steganographic techniques aren't limited to cyberwarriors and information security practitioners. In 2009, *World of Warcraft* (WoW) players discovered that screenshots created by the WoW client embedded data, including the user's account name, server IP address, and a timestamp. Players were concerned, and rumors spread that Blizzard, the company that runs WoW, would use the data hidden in screenshots to punish players who took and published screenshots of in-game exploits.

Modern Cryptosystems

A number of cryptosystems are in widespread use today, and each of them has a role in modern cyberwarfare. Among these are:

- DES, one of the earliest broadly adopted electronic encryption systems
- 3DES, its later upgrade
- AES
- RSA
- A number of other competitors that are available for programmers and system builders

Although the technical details of these algorithms' implementation are beyond the scope of this text, a basic understanding of the algorithms that are commonly used is helpful when exploring their use in cyberwar.

Data Encryption Standard

The Data Encryption Standard (DES) is a symmetric key cipher that was submitted in response to a U.S. government search for an algorithm to protect sensitive but unclassified electronic data. It was adopted in 1977 with U.S. National Security Agency (NSA) approval, and was quickly adopted for systems worldwide. Unfortunately, DES used a relatively short 56-bit key, and contained classified design components, leading to suspicion that the NSA was involved in its creation.

DES was broken in 1999 when groups working together were able to break a DES-encrypted message's encryption in less than 24 hours. DES has been considered outmoded for years, but remains widely available, and continues to be used in some software and systems.

Triple DES

Triple DES (3DES) was created using the pre-existing DES algorithm as a basis. 3DES works around the short key length of DES by using multiple rounds of encryption using the DES algorithm. This increases its key length to a maximum of 168 bits if run with

FYI

The U.S. federal government defines how cryptographic modules are accredited for modern cryptosystems in the Federal Information Processing Standard (FIPS) 140-2. FIPS defines four levels of security from Level 1, which provides basic encryption capabilities like an encryption chip in a personal PC, up to Level 4, which has enhanced security features and strong security mechanisms that detect and prevent physical and other attacks. FIPS testing is provided by the U.S. National Institute of Standards and Technology (NIST) and the Canadian Communications Security Establishment.

technical TIP

You might expect that running DES three times would result in an effective 168 bits of security, but an attack known as the *meet-in-the-middle attack* means that attacks require less work to complete an attack against the full list of potential keys. Attacks like these can mean that the expected protection provided by a cryptosystem is far less than expected. For 3DES, this means that it effectively has 2^{56} fewer possible keys for an attacker to try.

three rounds using different keys. Unfortunately, due to the manner in which attacks against it are conducted, it's effectively only a 112-bit key.

Because DES was already commonly available and broadly implemented on a variety of systems, 3DES was a relatively simple upgrade for most software and hardware developers. Obviously, using 3DES instead of DES requires three times the amount of computational work, but DES is a relatively fast algorithm, which allowed 3DES to be a practical solution to the issues with DES at the time.

Advanced Encryption Standard

In October 2000, with the highly publicized cracking of DES still fresh in the minds of security practitioners, the NIST announced approval of a replacement for DES. The replacement, known as the Advanced Encryption Standard (AES), uses the Rijndael cipher and was approved for use for all sensitive but unclassified data, much like its predecessor DES had been.

AES allows three key strengths, 128 bits, 192 bits, and 256 bits, allowing a balance between the amount of processor time required to encrypt a file and the security desired from the cipher. Like DES and 3DES, AES is also a symmetric cipher. AES is now commonly used in many areas, and has largely but not completely supplanted DES and 3DES.

RSA

Unlike DES, 3DES, and AES, the RSA algorithm created by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977 is an asymmetric encryption algorithm. As you learned earlier, asymmetric algorithms provide a number of distinct advantages over symmetric algorithms, including authentication, nonrepudiation, and the ability to safely distribute keys without a pre-existing secure channel. RSA also has the advantage of allowing arbitrarily large key lengths, thus allowing users of RSA encryption to increase key length to keep encrypted data secure for longer against possible brute-force attacks. RSA, much like DES, 3DES, and AES, has been implemented on a wide variety of platforms and systems. RSA is the most commonly used asymmetric encryption algorithm in use at the time of writing.

FYI

One important concept for cryptographic systems is Kerckhoffs's principle. Kerckhoffs's principle is that a cryptographic system should remain secure even if the enemy knows everything about the system except the key. Although not all cryptographers agree, exposing a system to third-party review can help ensure that a malicious actor hasn't compromised your cryptosystem by inserting a critical flaw in its design. All four of these algorithms have been broadly studied, although as noted earlier, DES was criticized for its NSA ties and secret elements when it was released.

Hashing and Message Digests

Earlier in this chapter, you learned about the need to ensure that messages haven't been modified when using a cryptosystem. The function typically used to do that verification is a *cryptographic hash*. Hashes take a message and generate a unique output value based on the message. The output of a hash is often known as a *message digest*.

Senders use message digests when they want to prove that their message has not been modified. To do so, they hash their original message, and then send the hash with the message. If the hash sent with the message matches the hash calculated by the recipient, the message has not been modified.

Hashes have five basic requirements:

1. They must accept inputs of any length.
2. Their output must be a fixed length.
3. They should be easy to compute for any input data.
4. They should not be reversible. Thus, you should not be able to calculate an original message from a hash.
5. They should not have collisions, meaning that they should not generate the same hash for different inputs.

There are a number of commonly used hashing algorithms. Two of the most common are MD5 and SHA. Both are in common use; however, MD5 has been shown to have a number of significant flaws, including the potential to create arbitrary collisions, or inputs that hash to the same value as other inputs. SHA-1, the original version of the SHA hash, has also been shown to have problems. Due to these vulnerabilities, a newer version of SHA known as SHA-2 is the preferred solution for most secure purposes today.

In addition to their uses for verifying message integrity, hashes are also commonly used for digital signatures in asymmetric cryptosystems. To digitally sign a message, the sender simply hashes the message, and then encrypts the message digest with his or her private key. The sender then adds the encrypted message digest to the plaintext message and sends it. This proves that the sender truly is the sender and that the message has not been modified, as it will hash to the correct value once it's decrypted with the sender's public key.

technical TIP

A common technique to make applications that use hashes to store passwords more secure is *salting*. A *salt* is a value added to the data that is hashed. The salt is kept secret and stored securely. This means that if the hashed password list is stolen, attackers can't simply compare the hashes with a list of known password hashes. The salted hashes will be different than those of the same text simply run through the algorithm.

The Hash Life Cycle

Hashing algorithms are vulnerable to attacks that create collisions. A collision occurs when two strings of text hash to the same hash value. If attackers can manage to do this with arbitrary strings they have chosen, the hash is broken and cannot be trusted. Because hashes are often used to verify that files haven't changed, this means that attackers who can create an arbitrary hash can replace trusted files with malware without any warning to users who responsibly check the file's hash. Obviously, if a hash can be replaced with data that an attacker chooses, the hashing algorithm should be replaced.

Research into how hashing algorithms can be defeated is often made public, and thus you typically have a reasonable idea when a hashing algorithm is first starting to be vulnerable to collisions or other attacks. Once you know that an algorithm has problems, it would make sense to start to retire the algorithm and replace it with a new, secure algorithm. That doesn't always happen due to the cost of replacing infrastructure, or simply because of bad habits or lack of knowledge on developers' part.

Even with problems like long-term replacement costs, you might think that organizations would replace an algorithm that has had known problems for almost 20 years, like MD5. Unfortunately, not every organization makes wise choices about which algorithms they choose to use or how they use them. The MD5 hashing algorithm has known flaws that allow the creation of arbitrary collisions. Perhaps worse, tools called *rainbow tables* exist that allow a speedy lookup of the original value of text that generates many hashes. Unfortunately, because MD5 is widely available, fast, and easy to implement, many organizations have continued to use MD5 to store hashed passwords. When attackers capture their databases and discover MD5-hashed passwords, they are able to quickly determine the original text of those passwords and then use the e-mail addresses and passwords they have stolen to attempt to log on to other sites.

Cryptography in Cyberwar

Now that you have a basic understanding of the underlying concepts and technologies used in cryptographic systems, you can look at their uses in computer network defense and computer network attack scenarios.

Computer Network Defense and Cryptographic Systems

The strategies and technologies used in computer network defense are typically the same types of technologies used in a traditional information security defense-in-depth plan. Layers of defensive encryption-based technologies are deployed to ensure that data is exposed as little as possible, and to make sure that keys are properly protected. Among these technologies are:

- **Drive encryption**, which is encryption either for full drives or for volumes (drive partitions). Drive encryption has become increasingly important, as it helps to protect against attackers who manage to access the machine while it's out of its owner's control. A number of strategies are available to implement drive encryption—including those that are based on hardware devices built in to the drive or the PC, as well as software-based encryption capabilities.
- **File encryption**, which is encrypting single files or groups of files. This common solution is used particularly when files need to be transferred or stored on unencrypted media. File encryption is built in to a variety of products, including AES encryption in modern versions of Microsoft Office.
- **Digital certificates**, or electronic documents that use a digital signature to link a public key to an individual, organization, or system. Digital certificates are widely used and trusted based on the processes used to verify the certificate holder. The authenticity of this relationship is checked by a **registration authority (RA)**, the part of a public key infrastructure that registers new users.
- A **public key infrastructure (PKI)**, which is a set of hardware, software, practices, and people that creates, distributes, stores, manages, and revokes digital certificates. PKI is used to ensure that individuals and systems who request certificates are who they claim to be. PKI systems then provide a chain of trust between a certificate issuer known as a **certificate authority (CA)**, a registration authority, certificate requestors, and those who must accept the certificates. PKI also includes the ability to revoke a certificate, which ensures that a stolen or compromised certificate can have its trust removed.
- **Transport Layer Security (TLS)**, which is the replacement for Secure Sockets Layer (SSL), the encryption that is typically used for secure Web transactions. TLS uses asymmetric cryptography to perform a secure exchange of symmetric keys, combining the abilities of an asymmetric cryptosystem with the speed of a symmetric cryptosystem. It also relies on PKI and x.509 certificates to verify trust relationships between certificates used for Web sites and their owners.

CAC Cards

The U.S. Department of Defense (DoD) uses a Common Access Card, or CAC card, for identification of both military and civilian staff. The card combines a picture ID with a chip that provides an encrypted certificate, which is issued by a federal public key infrastructure.

When a CAC card user wants to log on to a system or access a resource that supports use of the card, the user inserts the CAC card and then enters a PIN number. The PIN information is stored on the CAC card, allowing it to be verified. If the PIN matches, it checks the user's identification number against a central directory, which matches the user with his or her rights.

CAC cards combine encryption technology with other authentication and user identification capabilities. This means they're an important part of the DoD's security design. It also means that stealing a CAC card and acquiring the PIN that goes with it could provide access to attackers.

- Virtual private networks (VPNs), which rely on encryption when they want to protect the data they are sending in addition to simply creating a private network inside of another network. VPNs allow users on untrusted networks to become part of a trusted network, and can allow users and systems to safely transfer data through those untrusted networks in a secure way.
- Hashing, which is commonly used to verify that files remain unchanged, as part of cryptographic systems for digital signatures, and to ensure integrity.

All of these technologies rely on the algorithms, software, and hardware they are based on not being compromised. If attackers can successfully insert themselves earlier in the design process, they can attack in ways that most defenders cannot reasonably defend against, because they don't have the resources or technology to assess underlying systems and algorithms.

Computer Network Attack and Cryptographic Systems

Many of the techniques used in computer network attacks focus on defeating the cryptographic systems deployed by defenders. Because of this, you can review the same technologies used for defense when looking at how to conduct attacks:

- Drive encryption is vulnerable to a number of attacks. They include social engineering to persuade users to provide their passwords, acquiring passwords through the use of hardware keyboard capture tools, and attacking while the user is logged on with data unencrypted. In addition to these attacks, more-advanced attacks require physical access to the PC, but can retrieve passwords from live memory even if the machine is encrypted.

- File encryption is vulnerable to the same attacks as drive encryption, but files can be easier to acquire as they are often sent via e-mail or other file transfer services, and can be accessed from unencrypted drives.
- Digital certificates can be attacked by falsifying credentials or persuading a certificate issuer that the attacker is another person or organization. This has been leveraged multiple times by attackers, including well-publicized attacks that have created Google.com and Microsoft certificates. In each instance, the certificates were real certificates, but were issued to parties who were not the actual claimed owners.
- PKI has many parts that must work together successfully to remain secure. One successful attack against a Dutch certificate authority called DigiNotar offers an excellent example of a successful attack against PKI. In 2011, DigiNotar discovered that hackers had been able to issue over 500 fraudulent certificates through its service. This scale of compromise resulted in DigiNotar being removed from the list of trusted certificate authorities by major browser vendors. As a result, DigiNotar went bankrupt as it was no longer able to function as a business. The threat was especially severe for the Dutch government, as DigiNotar was trusted with the ability to issue certificates for the Dutch government's PKI.
- TLS attacks are typically conducted by using fake digital certificates. The DigiNotar attack was used to create a rogue Google.com certificate, issued under false pretenses, that was trusted by users. That certificate was used for a large-scale attack in which the intruder pretended to be Google, allowing the intruder to intercept secure communications meant for Google.
- Virtual private networks are most often attacked by intercepting data before it's encrypted, or by compromising the systems that host the VPN service. As with most encryption, getting to the data before it's encrypted is often the attacker's best solution.
- Hashing attacks are typically done through attacks on the hashing algorithm. In 2005, attackers launched attacks that allowed fake certificates to be created that had the same MD5 hash as the original certificates. Defenders must ensure that they don't use outdated cryptosystems that allow attackers to easily defeat their defenses.

Attacking Cryptography

Attacks against cryptographic systems are known as *cryptanalysis*. Cryptanalysis techniques include analyzing encrypted data to attempt to find patterns, finding vulnerabilities in the underlying cryptographic systems, and even simply trying every possible combination in a cryptosystem to decrypt the message.

The attack type chosen is often based on how much information is available to the attacker. That information is normally classified in one of five ways:

- Ciphertext only, in which the attacker has only one or more encrypted messages
- Known plaintext, where the attacker knows what the encrypted message was, and knows which ciphertext it matches to
- Chosen plaintext, where the attacker is able to choose the message that is encrypted, and thus can see the effect on the output in the ciphertext
- Adaptive chosen plaintext, in which the attacker can choose each successive plaintext, allowing the attacker to study changes in plaintext as messages are modified
- Related key attacks, which rely on analyzing messages sent with different, but related keys. This is similar to those used in Enigma intercepts, where the daily setting remained the same, but the operator setting changed for each message

The availability of strong encryption using publicly available algorithms like RSA and AES that have few, if any, currently known critical flaws means that attacks against them tend to occur before encryption is used. Or, attackers have to target the keys used to encrypt the data to recover it. However, that doesn't mean that attackers don't attempt assaults against encryption. Both attackers and defenders in cyberwar must consider the common attack techniques that are used against cryptosystems. The following sections look at some of the most common attacks against encryption when only ciphertext is available.

Brute Force

A brute-force attack is in many ways the simplest attack that can be conducted against an encrypted message. Brute-force attacks simply attempt to use every possible key against an encrypted message until the original plaintext is recovered. Thus, brute-force attacks are sometimes called *exhaustive key searches*.

The problem with brute-force attacks is that brute forcing strong encryption requires an incredibly large number of tries to be successful. They can be made faster by using faster computers or specialized techniques, but a brute-force attack against a 4,096-bit, RSA-encrypted message could take years to succeed, even with supercomputers available to attack it.

NOTE

In World War II, the Japanese Purple Machine, which was very similar to the Enigma device, was targeted by the United States. The attack on the Purple Machine was made significantly easier by the Japanese custom of using the same message format consistently, allowing a known plaintext attack.

Acquiring the Keys

One of the best ways to attack encryption is to acquire the keys. You can acquire keys in a number of ways, but the most common are:

- Social engineering the key owner into providing the key or using force or threats to convince the holder of the key to surrender it.
- Attacking the system that holds the keys.
- Using attacks against the key-generation system. The attacks either produce less-random keys or reveal the keys as they are generated.
- Exploiting a flaw in the encryption software.

Once a system is compromised, any keys that it contains should be considered compromised. Many organizations make the mistake of storing keys on systems that are connected to the Internet, or leaving keys in locations that are more accessible than intended.

Revealed in April 2014, the Heartbleed bug provided an excellent example of a way to acquire keys by exploiting a software flaw. Researchers at Codenomicon and Neel Mehta of Google Security discovered the flaw that came to be known as the Heartbleed bug in OpenSSL, a widely used open source implementation of TLS. When exploited, the bug allowed attackers to ask the server for a “heartbeat” response and permitted the attackers to ask for a larger response than the query they sent. The response that was sent back included the contents of the memory allocated to the OpenSSL application. In addition, the response could include the secret keys used for encryption between systems, as well as the decrypted traffic sent between users and servers.

The flaw itself was inadvertently added to the underlying code that makes up OpenSSL two years prior to its discovery, and OpenSSL was used by many Web servers around the world, including by Google, Amazon, and other major cloud service providers. Any organization that was able to identify and exploit the Heartbleed bug before it became public knowledge could have gained access to those sites using captured usernames and passwords, or could have pretended to be the sites themselves by using its secret keys.

Attacking the Algorithm

Encryption algorithms can be seriously weakened if parts of the cryptosystem have flaws. In a typical modern cryptosystem, you will often find a random number generator and a number of distinct parts of the algorithm. In addition, the encryption system is typically implemented as either hardware, software, or a combination of both. This provides attackers with a number of places to insert flaws. This type of attack isn’t just theoretical, as you will read in the following section.

NSA and RSA

Attacking the cryptosystems after they already exist isn't the only way to succeed. If an attacker can ensure that the cryptosystem is flawed before it enters general use, the attacker can exploit it without the huge amounts of effort that it usually takes to find flaws in a cipher. This is a significant danger for cryptosystems, and it's one reason that the most trusted encryption algorithms are often those that have had broad public review to help ensure that they don't have flaws lurking inside them. Unfortunately, cryptographic systems are also incredibly complex, and the number of individuals who can fully understand them is relatively small. Minor changes in how an algorithm is constructed or how it's implemented can have a major impact on how secure it is.

In 2013, Reuters reported that the NSA had paid RSA, a security company that specializes in encryption technologies, \$10 million to create a cryptographic system that included a deliberate flaw. The flawed Dual Elliptic Curve cryptographic system was part of RSA's BSafe security kit, and the Dual Elliptic Curve system's vulnerability is reported to have made it possible for the NSA to crack it. Later reports suggested that that vulnerability was not the only one inserted into the software. In fact, in 2014, researchers alleged that the Extended Random extension, which was proposed by the NSA, would have simplified attacks on Dual Elliptic Curve. The NSA didn't find success with Extended Random, however, as it was removed from RSA's software.

Of course, ensuring that the cipher itself has a weakness isn't the only way to defeat a cryptosystem, and the NSA isn't the only organization to have successfully provided a weak cryptosystem to organizations it wanted to monitor. After World War II, the Allies sold captured Enigma machines to developing countries. The fact that the Allies had the ability to decrypt Enigma messages was not public knowledge, and thus customers in those countries, including national governments, unwittingly used a broken cipher to protect their secrets.

NOTE

There are many other ways to attack cryptosystems. This section has explored just a few to provide a basic understanding of the threats defenders face in computer network defense, and which attackers may try in computer network attacks.

Defeating Attacks on Cryptographic Systems

Attacks on cryptographic systems occur during the creation of the systems, during their implementation, and most often, when the system is in broad use. There are as many ways to defend cryptographic systems as there are possible attacks, but most defenses focus on ensuring that the system is well designed, and that its component parts themselves aren't vulnerable. A well-designed, strong cryptosystem that has been carefully tested and whose supporting infrastructure has not been compromised can usually be considered safe.

Defenses

Defending against attacks on cryptographic systems requires that defenders think like the attackers they might face. Attackers will attempt to make the cryptosystem less effective, they will search it for flaws that will make it easier to crack, and if they have no other options, they may simply attack using a brute-force attack.

Organizations that rely on widely available ciphers and hashes have to watch for news about problems with them. They must have plans in place to replace or upgrade them when they begin to have problems. The issues that MD5 and DES have encountered over the past 20 years have shown that many organizations don't handle the transition from one technology to another well, and that they often leave themselves vulnerable.

In addition to building a plan to handle the ciphers themselves, organizations that engage in computer network defense need to carefully consider where they use encryption.

Defense in Depth Using Cryptographic Systems

Cryptographic systems are often deployed as part of a defense-in-depth strategy. Figure 10-4 shows common uses of encryption in a typical computer network defense scenario. Note that the workstation uses full disk encryption, files sent via e-mail are encrypted, and Web traffic sent from the workstation is encrypted in transit over the wire by TLS.

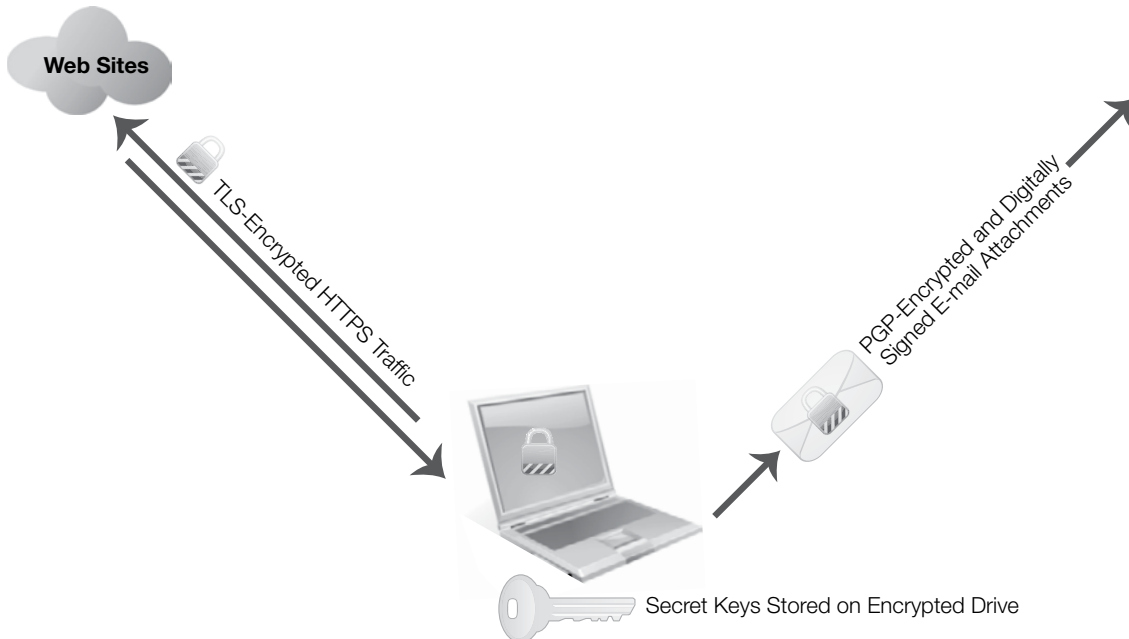


FIGURE 10-4

Encryption uses for a typical modern laptop, including secure Web access, e-mail, and full drive encryption.

The use of encryption in defense-in-depth systems provides defenders with the ability to pick appropriate levels of encryption technology to match the sensitivity of data, the resources required to provide the encryption, and the risk to the data and systems. Thus, a highly sensitive file might be kept encrypted, stored on an encrypted drive, and only transferred via a trusted, encrypted virtual private network connection. A lower-sensitivity file might be left unencrypted on the drive, might only be protected at rest by full disk encryption, and might be protected in transit by normal TLS used via a Web browser.

Weaponizing Cryptography

The ways in which encryption is used in modern malware is a useful model to examine when considering how encryption techniques can be used in both offensive and defensive cyberwar activities. The following sections look at three major examples of malware that uses encryption. First, you'll examine the long history of using encryption to evade detection by defensive technologies. Second, you'll learn about the Zeus banking Trojan, which uses encryption to avoid detection and to keep its configurations secret. Finally, you'll read about Cryptolocker, an infection that encrypts data using modern asymmetric encryption, and then extorts a ransom from users who need their data back.

Defensive Cryptography: Malware Encryption

Malware is one of the most prevalent threats in modern cyberwarfare. Stuxnet, Flame, and a host of other targeted threats have infected systems and provided backdoor Trojan access to systems. Malware has provided intelligence by leaking data, allowed infrastructure attacks, and acted as a remote-control system to take over other more critical systems.

Of course, malware wouldn't be effective if it could be easily detected and removed. For years, viruses and other malware were relatively easily detected by antivirus systems that could keep a known fingerprint of the malware and then remove it. Malware authors responded by making malware that changes its fingerprint every time it lands on a new system, but they have also made use of encryption to protect their malware.

Typically, encryption is used in three ways in modern malware:

- To encrypt the malware itself when on disk and running on systems. This helps to ensure that defenders cannot easily reverse engineer the malware to determine what it's doing or who built it by looking at its code.
- To protect the configuration files that the malware uses. This means that defenders cannot verify which targets, control systems, and other settings the malware is configured with. When defenders get access to that information, it can help them determine who the attacker is, or which version of the malware they are fighting against.

- To protect the network communication of the malware. This provides concealment, as defenders theoretically cannot read encrypted traffic. This allows the malware to appear like any other encrypted Web traffic. It also makes sure that defenders don't know what the malware is doing. If they could view the traffic, they could see command-and-control data in real time, possibly letting them know of attacks or other activities.

Cryptography isn't used only for defensive purposes. It can also be used as a weapon by applying the same technologies in ways that cause harm.

Offensive Cryptography

Encryption is increasingly used as part of the weapons packages deployed in cyberwarfare: viruses, Trojans, and other malware. Advanced malware packages now commonly include strong, openly available encryption capabilities that allow them to conceal their communications, protect their configurations, and avoid malware scanners by encrypting their internal workings. The following sections examine two malware packages. First, you'll look at Zeus, a malware package originally aimed at banking and other sensitive data, but which has been updated and made broadly available as a configurable malware package. Next, you'll look at how Cryptolocker uses strong RSA encryption to extort money from infected users, using encryption as a weapon and possibly foreshadowing how next-generation cyberwar tools could disable or hold hostage both military and civilian infrastructure.

Zeus

Cyberwarfare technologies and techniques often cross over into use by criminals, and the techniques developed by malware writers are often of interest to computer network attack (CNA) developers. The Zeus (or Zbot) Trojan is an excellent example of malware demonstrating new techniques that are of interest to both sides. In fact, Zeus is of interest to governments due to their need to provide computer network defense. In 2007, when Zeus was first discovered, it was used to target data at the U.S. Department of Transportation as well as multiple commercial entities.

FYI

Viruses and other malware often have a variety of different names, as each antivirus vendor tends to name them differently based on information they discover about the malware or based on their own naming conventions. Thus, Zeus is also known as Trojan.zbot, ZeuS, and Zbot. If you encounter malware, remember that simply searching for one name may not provide all of the information you need. Amusingly, even criminal malware uses aliases, even if inadvertently.

technical TIP

In addition to the Zeus Trojan, the Citadel, Carberp, and SpyEye Trojans have been widely deployed. Each of these packages targeted user data, such as credentials, credit card information, or other similarly important information. Each of these packages has evolved over time, with features like plug-in architectures, which allowed the malware to adapt to the system it's on, built-in anti-detection technologies to avoid being noticed by antivirus software, and the ability to attack multiple operating systems and platforms. Carberp's claim to fame is that it was the first malware to use a randomly generated key rather than a static key that the creators of the software kept centrally.

Zeus is primarily a sensitive data theft tool specializing in the gathering of stored usernames and passwords, as well as those that users type into systems on which it resides. In fact, Zeus and other banking Trojans like it are specially designed to recognize high-security sites that require multiple factors to log on and to use plug-ins designed for that site to capture the logon information. Zeus is highly modular, and has evolved significantly since its release. In fact, the creator of the Zeus Trojan treated it like commercial software, making licenses available for purchase for a few thousand dollars with a recurring subscription rate for updates.

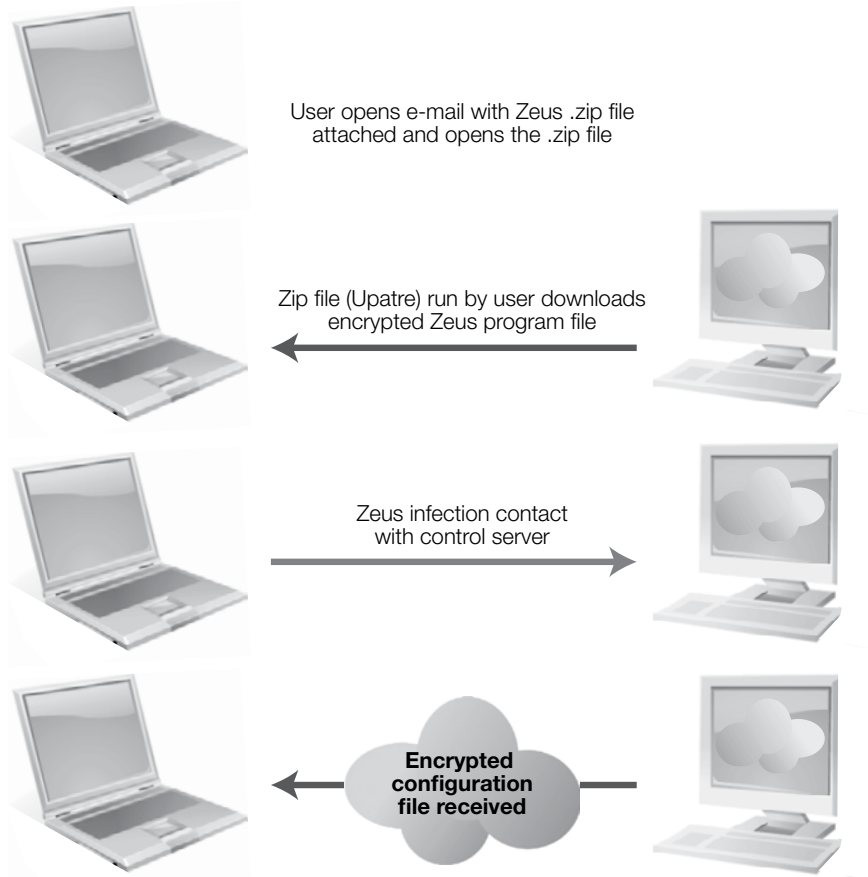
All of this isn't atypical for malware. However, the ways in which Trojans and other malware have begun to use encryption as both a defensive and an offensive tool means that the techniques previously used to protect data are now being used to exploit systems and to protect malware. Because cyberwar tools are often similar to those used for network security exploit by non-cyberwarfare participants, it's reasonable to presume that the same techniques will show up in cyberwarfare attack. Obviously, defenders must attempt to defend against these techniques.

Figure 10-5 shows how a typical Zeus infection uses encryption during the infection process and ensures that its configuration information remains secure from defenders. Zeus infections often start with an e-mail with an attached .zip file. Users click on the .zip file, unwittingly activating a downloader, which downloads and decrypts the Zeus executable. Once Zeus is live, it then reaches out to command-and-control servers for an encrypted configuration file.

Perhaps one of the most interesting turns in the story of Zeus is the alleged retirement of its author in 2010. After the announcement, the author released the Zeus source code and transferred the rights to sell Zeus to the creator of the SpyEye Trojan. Now, Zeus source code is available from GitHub and other sites, allowing would-be attackers to gain easy access to one of the most capable Trojans found in recent history. In cyberwarfare terms, this means that the Zeus can be weaponized by anybody with the ability to download and configure it from an easily accessible software repository.

FIGURE 10-5

The Zeus.Gameover version of the Zeus Trojan infection process starts when an unsuspecting user opens a .zip file sent by an attacker, infecting his or her workstation. The Trojan uses encryption to hide its installer and its configuration files.

**NOTE**

Cryptolocker relies on 2,048-bit RSA encryption as well as AES-256 encryption keys for each file. Further, Cryptolocker actually informs the infected user that it uses RSA, ensuring that the user knows he or she cannot reasonably expect to decrypt the files on the drive. Even users who contact an expert will find out that the only way to get their information back without backups is to pay the ransom.

Cryptolocker

Malware that encrypts files on a victim's hard drive isn't new, but the Cryptolocker malware was likely the most successful encryption-based malware thus far. Cryptolocker uses asymmetric encryption, generating a public/private key pair and sending the private key to a central command-and-control server where it's stored. It then encrypts all of the important documents, such as Microsoft Word, Excel, and other files on the victim's PC, and presents the user with a message demanding payment of 300 U.S. dollars or 300 Euro within 72 hours.

Cryptolocker and Bitcoin

The Cryptolocker virus typically asked its victims to pay via common international money transfer methods within 72 hours.

As the price of Bitcoin went up, Cryptolocker's authors changed their payment methods to allow payment with Bitcoin. They relied on the relative anonymity of Bitcoin, and demanded a half a Bitcoin in payment. At the time, Bitcoin was worth between \$200 and \$400, a significant increase from the previous value of less than \$100, making it an attractive currency to request.

If victims did not pay up within 72 hours, Cryptolocker deleted itself and the public key, leaving the encrypted files unusable. In fact, Cryptolocker's infection methods meant that when Cryptolocker servers were taken offline, victims could no longer recover their files. Reinfection of the system generated a new public/private key pair, meaning that the data was unrecoverable.

Cryptolocker-style attacks in cyberwar could be used to make critical systems unusable, or as leverage in a negotiation. The same techniques used to encrypt sensitive data could be used for any valuable data, research, or communications. Once again, the techniques used in criminal malware attacks are likely to inform future cyberwar activities.

The Future of Cryptography in Cyberwar

Cryptography will continue to be a critical part of cyberwar. Encryption is the preferred way to protect live data at rest and in transit, and encryption capabilities are now built in to many products from the ground up. The increased use of encryption throughout data's life cycle means that attacks against encryption are necessary for attackers in cyberwar to succeed. It also means that defenders must account for changes in their use of encryption throughout data's life cycle. Physically securing a filing cabinet is no longer sufficient to preserve the security of data, and strong encryption placed on a file when it's created may be easily broken a decade or two later.

Attacks

The modes of attack on cryptosystems have changed as encryption has become increasingly strong. Modern encryption algorithms like RSA with a 2,048-bit key can reasonably be expected to remain secure for years, despite high-speed cracking efforts with massive supercomputers. Regardless of advances in technology, such as faster processors and graphics card-based cryptographic attacks, these algorithms remain strong.

Because strong and secure encryption algorithms exist, attacks continue to move to other locations in the encryption process. If data can be obtained while it's unencrypted, that is far easier than trying brute force to break strong encryption. There are two major places where future attacks are likely to occur. Those areas are already the focus of attacks by both nation-state actors and others engaging in both cyberwar and traditional cyberattacks.

The first place that attacks occur against the modern encryption system is while the data is in use and unencrypted. Modern consumer-grade operating systems and devices often encrypt user data by default, making the data less likely to be accessed if the device is stolen, but they cannot leave the data encrypted while the user is accessing it. This means that attacks must target the user's processes and programs and active use of the device when the data must be unencrypted and accessible to the user. This drives a growth in targeted malware that obtains and sends unencrypted data using the user's rights.

The same large-scale adoption of encryption at rest and in transit makes attacks on the underlying algorithms and hardware that perform encryption more attractive. If a government or other organization can introduce a weakness into an algorithm or a hardware device or chip, it can use that vulnerability to more easily crack or possibly entirely bypass the encryption. Nation-state actors that do this create the potential for a widely adopted technology to have a critical flaw that they introduced. If that flaw becomes public knowledge, the impact is likely to be significant, as known flaws often result in broad exploit of the issue.

Defenses

Future defenses against attacks on encryption will continue to involve a few common solutions:

- Increased key length for existing strong encryption algorithms will be important. Algorithms without known vulnerabilities that can simply increase their key length can continue to outpace the speed of new cracking tools until a vulnerability is found. Even if a vulnerability is found, sometimes a modification of the system can be used to extend its life span, much like the evolution of DES to 3DES discussed earlier in the chapter.
- Open access to encryption systems for review will continue to be important for those that subscribe to Kerckhoffs's principle. Organizations are likely to continue to use internally developed encryption algorithms, but publicly available strong encryption will continue to benefit from exposure to the community.

- Increased oversight of the underlying design of hardware and software will be important as the NSA's alleged influence over the design of various RSA products, as well as accepted cryptosystems demonstrates. Countries across the world will likely require greater oversight of new encryption standards to avoid a single country from having access to a known flaw in an encryption system or one of its component parts.
- More integration of end-to-end encryption that makes accessing data more difficult for untrusted applications will be necessary. Malware that targets data that is unencrypted when viewed by the user of a system will mean that more data will be encrypted even when the user is logged on. There is a fine line between usability and security that will continue to move in sensitive systems.

Overall defense will remain a challenge due to the need to use data. Usability will almost always mean that there will be a moment when data is unencrypted, and thus can be attacked.

Quantum Cryptography

One of the most anticipated and one of the most feared changes in encryption technology is quantum cryptography. **Quantum cryptography** uses the theories of quantum mechanics to perform cryptographic tasks like decryption and encryption. In theory, this allows a variety of possible advantages:

- Decrypting traditional encryption methods may no longer require huge time periods to accomplish using a quantum computer.
- Quantum encryption may be nearly impossible to break.
- Eavesdropping on key distribution would be detectable.
- Location-based cryptography may be possible, with the physical position being the only required key.

No quantum computers currently exist that can perform these functions, but the ideas that quantum cryptography relies on would completely change cryptography as we know it if they become a reality. That change would mean that the role of encryption and cryptanalysis in cyberwar would also completely change, and traditional cryptosystems would become essentially useless.



CHAPTER SUMMARY

In this chapter, you learned about cryptography, the study of how to conceal messages from unwanted readers. You read about symmetric ciphers that use the same key for both encryption and decryption; asymmetric ciphers that allow safe exchange of public keys, nonrepudiation, and authentication; and signatures using hashes. You examined commonly used encryption and hashing algorithms and learned about how they are attacked and what drives their replacement.

You then used your understanding of cryptosystems to examine current cyberwarfare techniques for attacking and defending networks, systems, and data using encryption. The Zeus malware introduced you to the use of encryption to conceal malware downloads, communication, and configurations, and Cryptolocker provided a glimpse into how encrypting files can be used as an attack. With these attacks in mind, you looked at the future of cryptography and its likely effects on both computer network attack and defense.



KEY CONCEPTS AND TERMS

Asymmetric encryption	Digital certificates	Message digests
Certificate authority (CA)	Drive encryption	Plaintext
Checksums	Encryption	Public key encryption
Ciphers	File encryption	Public key infrastructure (PKI)
Ciphertext	Hashes	Quantum cryptography
Code	Key	Registration authority (RA)
Cryptosystem	Key distribution	Symmetric encryption
Decryption	Key pair	