CRYPTOGRAPHY

This part introduces you to an essential element of modern security, that of cryptography. The importance of cryptography has increased over time to become a key defense in securing data from threat actors. Module 6 defines cryptography, explains cryptographic algorithms, looks at attacks on cryptography, and shows how cryptography is implemented. Module 7 continues with more advanced cryptography topics such as digital certificates, public key infrastructure (PKI), and transport encryption algorithms.

PART 3

MODULE 6

BASIC CRYPTOGRAPHY

After completing this module, you should be able to do the following:

1. Define cryptography
2. Describe hash, symmetric, and asymmetric cryptographic algorithms
3. Explain different cryptographic attacks
4. List the various ways in which cryptography is used

Front-Page Cybersecurity

It has long been recognized that the Allies' ability to read the encrypted transmission codes of Germany during World War II was a decisive key to victory. Yet this does not mean that the Allies could simply listen in and hear the conversations taking place, much like listening to a voice conversation over a telephone. The encryption keys used for encoding the messages were changed daily, so that each day, codebreakers had to work to determine the new key for that day before reading any messages. The transmissions were not voice communications but Morse code (dots and dashes). So, the Allies' attempts to read encrypted transmissions was a daily endeavor that continued throughout the war, not always successfully.

An often-overlooked aid in breaking the encoded messages was the complexity of sending the messages themselves.

At the start of World War II, the Germans used a cipher machine called Enigma, which had rotor settings and wiring schemes that were changed daily. Later Germany implemented an additional scheme called Double Playfair. This was an even more complex system than Enigma: it used a five-by-five grid with two separate keys, and each pair of letters had to be encrypted not once but twice. An overlaying stencil also changed daily as with the Enigma settings. Double Playfair was used on the front lines to send information about immediate military plans, and often by the time the message could be deciphered by the Allies, it was too late to take advantage of it.

The complexity of Double Playfair became its own worst enemy.

Due to the difficulty of coding messages with Double Playfair, the German radio operators often made mistakes in encrypting messages, and they were then forced to immediately send corrected messages. Thus, the same message with only minor corrections was sent multiple times. Repeating the same message with slight variations to correct errors was a tremendous aid to the codebreakers trying to crack the message because it revealed a detectable pattern that made the code much easier to break.

Other seemingly innocent actions by German radio operators also helped the codebreakers. Morse code messages are generally transmitted by a hand-operated device such as a telegraph key. Operators differ in how they tap out the dots and dashes on the key: they may use a slightly shorter or longer dash or gap between each tap, and they may even do so only for particular characters. This tendency is called their unique "fist." Experienced listeners can recognize specific senders based on their fist alone.

The fist of the German radio operators could help identify the sender. Over time, the Allies could pinpoint where the sender was located based on the content of the messages. Information about the transmitted military plans could even be narrowed to the specific region or town where the plans would be carried out. Some German radio operators also sent more gossip than others did, so over time, message content also helped to identify the sender.

Another aid to identify senders was that many of them, like all humans, were creatures of habit. They might use a girlfriend's name each time they sent a test message or end each message with the same phrase. This repeated content helped the Allies detect patterns and crack the codes.

The complexity of Double Playfair and repeating the same words regularly from habit were all helpful in breaking the German's code during World War II. Some of the same principles hold true today. While it may be hard to crack an encryption cipher itself, if the process by the user is poorly implemented, then encryption suddenly becomes very weak.

Consider attorneys who need to protect important documents stored at their office. They may erect a fence surrounding the property, install strong door locks, and place cameras over the doors in order to deter thieves. Yet, as important as physical defenses are, they nevertheless could be breached—and, in some cases, rather easily. For the attorneys to securely safeguard the documents, as a second line of defense, they would likely store the documents in a safe protected by a combination lock. Even if thieves could climb over the fence, break the door locks, and circumvent the cameras to enter the office, the intruders would have to break the code to the combination lock before reaching the documents. The effort would require a much higher level of time and expertise and generally would defeat all but the most sophisticated and determined thieves.

Information security uses the approach to protect data. Physical and technical security, such as motion detection devices and firewalls, help to keep out data thieves. For high-value data that must be fully protected, security professionals use a second level of protection: encryption. Even if attackers penetrate the device and reach the data, they still must uncover the key to unlock the encrypted contents, a virtually impossible task if the encryption is properly applied.

In this module, you will learn how encryption can be used to protect data. You will first look at what cryptography is and how it is used. You will also examine cryptographic attacks and, finally, see how to use cryptography.

DEFINING CRYPTOGRAPHY

CERTIFICATION

2.1 Explain the importance of security concepts in an enterprise environment.

2.8 Summarize the basics of cryptographic concepts.

Defining cryptography involves understanding what it is and how it is used. It also involves knowing the limitations of cryptography.

What Is Cryptography?

As early as 600 BC, the ancient Greeks wrestled with how to keep messages sent by couriers from falling into enemy hands. One early method was to tell the message to a courier so he could later repeat it when he arrived at his destination. However, couriers often paraphrased the message in their own words, which resulted in omissions or variations due to forgetfulness. In some cases, couriers even intentionally altered the message if they had been bribed or blackmailed by the enemy.

Written dispatches, on the other hand, accurately conveyed the message without the courier knowing its contents, thus reducing the risk of a security breach. But enemies could uncover written messages if they found and searched the courier, so the Greeks contrived ingenious ways of disguising and concealing documents. News of an imminent

Persian invasion early in the fifth century BC was sent by writing the message in ink on wooden tablets and then covering them with wax, the normal medium for written messages. An innocuous message was written on the wax, which hid the underlying secret message. Other techniques included messages disguised in the form of a wound dressing, hidden in an earring, written as a tattoo, or concealed in a sandal or a mule's hoof.

Although these strategies made the message more difficult to find, if it were discovered, the contents could be read easily. The ancient Greeks then turned to making their messages intelligible only to the desired recipient. One technique was to substitute one letter for another (in English, $A = M$, $B = N$, $C = O$, etc.), though this was fairly easy to decipher. The Greeks also marked letters of an ordinary text with tiny dots to indicate which letters, when combined, revealed the secret message. The Greeks even wrote poems in which the first letter of each line of the poetry could be extracted to spell out a message.[1]

By hiding the message on tablets covered with wax or as a tattoo, the Greeks were performing **steganography**, a word from Greek meaning *covered writing*. Steganography hides the existence of information. Today, steganography often hides data in a harmless image, audio, or even video file. It typically takes the data, divides it into small pieces, and hides the pieces among invisible portions of the file. A common scheme is to hide data in the file header fields that describe the file, between sections of the *metadata* (data that is used to describe the content or structure of the actual data), or in the areas of a file that contain the content itself. An example of steganography is shown in Figure 6-1.

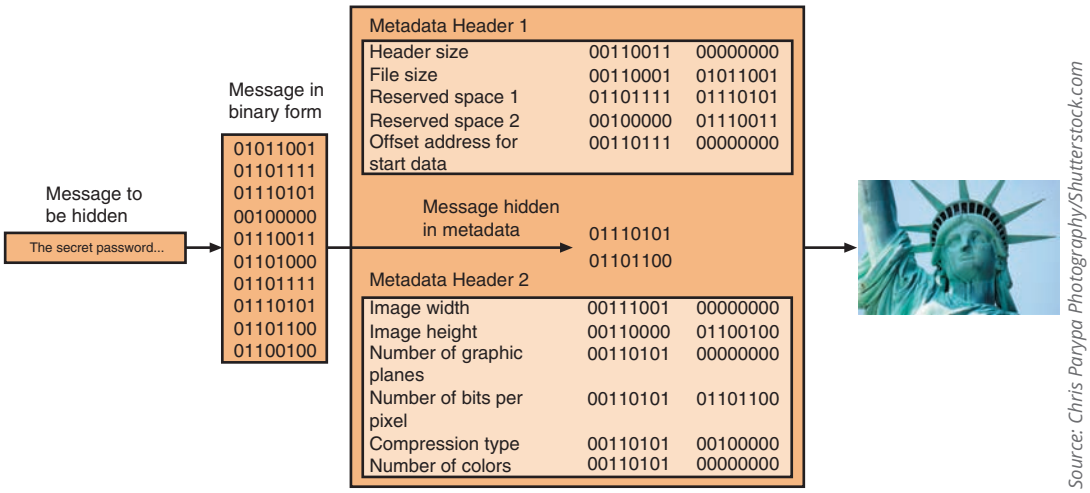


Figure 6-1 Data hidden by steganography

By making the message more difficult to read, such as by substituting letters, the ancient Greeks were also performing one of the early forms of **cryptography**, which is from Greek meaning *hidden writing*. Cryptography is the practice of transforming information so that it cannot be understood by unauthorized parties and, thus, is secure. Cryptography is usually accomplished through "scrambling" the information so that only approved recipients (either human or machine) can understand it.

When using cryptography, the process of changing the original text into a scrambled message is known as **encryption**. (The reverse process is **decryption**, or changing the message back to its original form.) In addition, the following terminology applies to cryptography:

- *Plaintext*. Unencrypted data that is input for encryption or is the output of decryption is called *plaintext*.
- *Ciphertext*. *Ciphertext* is the scrambled and unreadable output of encryption.
- *Cleartext*. Unencrypted data that is not intended to be encrypted is *cleartext* (it is "in the clear").

Plaintext data to be encrypted is input into a cryptographic **algorithm** (also called a *cipher*), which consists of procedures based on a mathematical formula. A *key* is a mathematical value entered into the algorithm to produce the ciphertext. Just as a key is inserted into a door lock to lock the door, in cryptography a unique

NOTE 1

Steganography is sometimes used together with encryption so that the information is doubly protected. First encrypting the data and then hiding it requires someone seeking the information to first find the data and then decrypt it.

NOTE 2

At its essence, cryptography replaces trust with mathematics.

mathematical key is input into the encryption algorithm to "lock" the data by creating the ciphertext. When the ciphertext needs to be returned to plaintext so the recipient can view it, the reverse process occurs with the decryption algorithm and key to "unlock" it. Cryptographic algorithms are public and well known; however, the individualized key for the algorithm that a user possesses must at all costs be kept secret. The cryptographic process is illustrated in Figure 6-2.

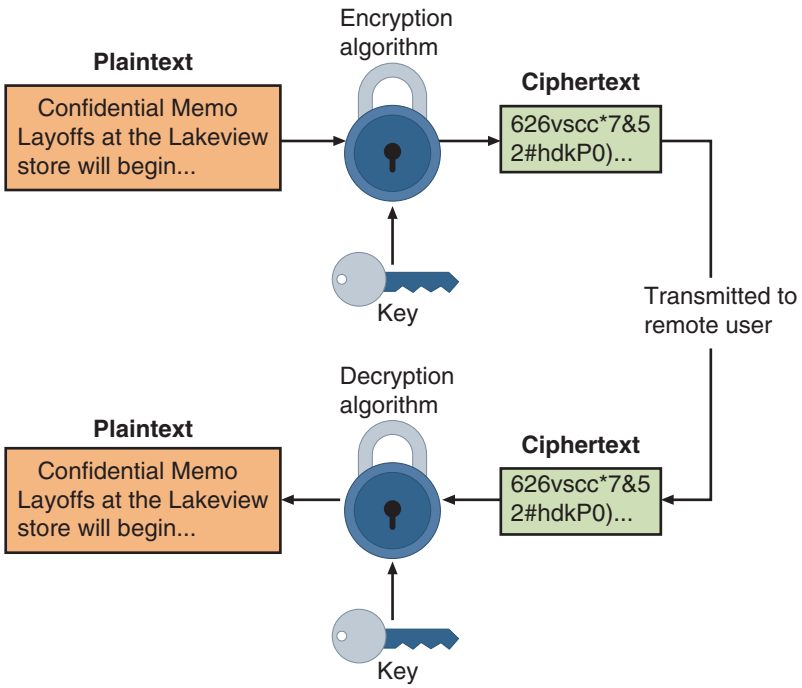


Figure 6-2 Cryptographic process

Ciphers can be separated into several categories, with specific types in each category. One category is a *substitution cipher*, which exchanges one character for another. By substituting *1* for the letter *A*, *2* for the letter *B,* etc., the word *security* becomes *1804022017081924*. One type of substitution cipher is *ROT13*, in which the entire alphabet is rotated 13 steps (*A* = *N*, *B* = *O*, etc.) so that the word *security* becomes *frphevgl*. Another cipher is the *XOR cipher*, which is based on the binary operation e*X*clusive *OR* to compare two bits: if the bits are different, a *1* is returned, but if they are identical, then a *0* is returned. For example, to encrypt the word *security* with the word *flapjack* using the XOR cipher, the binary equivalent of the first letter *s* (*01110011*) is "XOR-ed" with the binary of the letter *f* (*01100110*) to return *00010101*. The substitution continues for each letter (*e* XOR *l*, *c* XOR *a*, etc.), so the entire result of *security* XOR *flapjack* is 00010101 00001001 00000010 00000101 00011000 00001000 00010111 00010010.

NOTE 3

The strength of a cryptographic algorithm depends upon several factors, one of which is the quality of *random numbers*, or numbers that do not follow an identifiable pattern or sequence. Software usually relies upon a *pseudorandom number generator (PRNG)*, which is an algorithm for creating a sequence of numbers whose properties approximate those of a random number. PRNGs attempt to create numbers that are as random as possible.

Cryptography Use Cases

Why use cryptography? Several common use cases (situations) for cryptography can provide a range of security protections. These protections include the following:

- *Confidentiality.* Cryptography can protect the confidentiality of information by ensuring that only authorized parties can view it. When private information, such as a list of employees to be laid off, is transmitted across

the network or stored on a file server, its contents can be encrypted, which allows only authorized individuals who have the key to read it.

- *Integrity.* Cryptography can protect the integrity of information. Integrity ensures that the information is correct and no unauthorized person or malicious software has altered that data. Because ciphertext requires that a key must be used to open the data before it can be changed, cryptography can ensure its integrity. The list of employees to be laid off, for example, can be protected so that no names can be added or deleted by unauthorized personnel.
- *Authentication.* The authentication of the sender can be verified through cryptography. Specific types of cryptography, for example, can prevent a situation such as circulation of a list of employees to be laid off that appears to come from a manager but, in reality, was sent by an imposter.
- *Nonrepudiation.* Cryptography can enforce nonrepudiation. *Repudiation* is defined as denial; nonrepudiation is the inability to deny. In information technology, **nonrepudiation** is the process of proving that a user performed an action, such as sending an email message. Nonrepudiation prevents an individual from fraudulently reneging on an action. The nonrepudiation features of cryptography can prevent managers from claiming they never sent lists of employees to be laid off to an unauthorized third party.

NOTE 4

A practical example of nonrepudiation is Astrid taking her car into a repair shop for service and signing an estimate form of the cost of repairs and authorizing the work. If Astrid later returns and claims she never approved a specific repair, the signed form can be used as nonrepudiation.

- *Obfuscation.* **Obfuscation** is making something obscure or unclear. Cryptography can provide a degree of obfuscation by encrypting a list of employees to be laid off so that an unauthorized user cannot read it.

The security protections afforded by cryptography are summarized in Table 6-1.

Table 6-1 Information protections by cryptography

Characteristic	Description	Protection
Confidentiality	Ensures that only authorized parties can view the information	Encrypted information can only be viewed by those who have been provided the key.
Integrity	Ensures that the information is correct and no unauthorized person or malicious software has altered that data	Encrypted information cannot be changed except by authorized users who have the key.
Authentication	Provides proof of the genuineness of the user	Proof that the sender was legitimate and not an imposter can be obtained.
Nonrepudiation	Proves that a user performed an action	Individuals are prevented from fraudulently denying that they were involved in a transaction.
Obfuscation	Makes something obscure or unclear	By being made obscure, the original information cannot be determined.

Obfuscation is frequently misunderstood and often misapplied. By definition, encrypting data obfuscates that data, and its protection is based on an obfuscated key. However, obfuscation in other areas of cybersecurity cannot provide the same assurance that something is confidential. That is because obfuscation is often used erroneously as an attempt to hide something from outsiders: "If the bad guys don't know about it, that makes it secure." However, this approach (called *security through obscurity*) is flawed since what makes it secure is the fact that it is unknown. Because it is essentially impossible to keep secrets from everyone, eventually the data will be discovered and the security compromised. Thus, obfuscation cannot by itself be used as a general cybersecurity protection.

Unfortunately, some organizations attempt to apply security by obscurity to cryptography. They create proprietary cryptographic algorithms (touted as "military-grade" cryptography) and suggest that because the algorithm is "secret,"

NOTE 5

What makes cryptography secure is the obscurity of the key and not of the algorithm.

it is secure. However, in reality, proprietary algorithms are weak. Modern cryptographic algorithms are based on known mathematical proofs, and the algorithms are selected by competition after having been thoroughly reviewed by the cryptographic community. The algorithms have proven their value over time by their wide adoption and use. Proprietary algorithms, on the other hand, have not been properly vetted and will likely contain flaws and, thus, should not be used.

Cryptography can provide protection to data as that data resides in any of three states:

- *Data in processing*. **Data in processing** (also called *data in use*) is data on which actions are being performed by devices, such as printing a report from a device.
- *Data in transit*. Actions that transmit the data across a network, such as an email sent across the Internet, are called **data in transit** (sometimes called *data in motion*).
- *Data at rest*. **Data at rest** is data stored on electronic media.

Limitations of Cryptography

Despite providing widespread protections, cryptography faces constraints (limitations) that can impact its effectiveness. In recent years, the number of small electronic devices that consume little power (**low-power devices**) has grown significantly. These devices range from tiny sensors that control office heating and lighting to consumer devices such as thermostats and lightbulbs. Increasingly, low-power devices need to be protected from threat actors who could accumulate their data and use it in nefarious ways. In addition, many applications require extremely fast response times, including communication applications (such as collecting car toll road payments), high-speed optical networking links, and secure storage devices such as solid-state disks. Cryptography is viewed as a necessary feature to add to protect low-power devices and applications that require fast response times to make them secure.

NOTE 6

Compared with the average energy requirements of a laptop computer (60 watts), the typical wireless sensor draws only 0.001 watt.

However, adding cryptography to low-power devices or those that have near instantaneous response times can be difficult. To perform their computations, cryptographic algorithms require time and energy, both of which are typically in short supply for low-power devices and applications needing ultra-fast response times. This results in a **resource vs. security constraint**, or a limitation in providing strong cryptography due to the tug-of-war between the available resources (time and energy) and the security provided by cryptography. Ideally, a cryptographic algorithm should have **low latency**, or a small amount of time that occurs between when a byte is input into a cryptographic algorithm and the time the output is obtained. However, some algorithms require multiple (even 10 or higher) "cycles" on sections of the plaintext, each of which draws power and delays the output. One way to decrease latency is to make the cryptographic algorithm run faster—though doing so increases power consumption, which is not available to low-power devices or would slow the normal operations of the device. The resource vs. security constraint is illustrated in Figure 6-3. Table 6-2 lists additional cryptographic constraints.

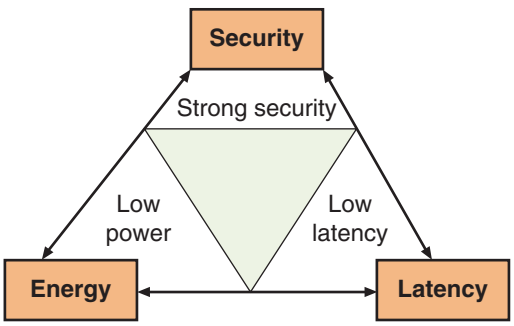


Figure 6-3 Resource vs. security constraint

Table 6-2 Cryptographic constraints

Limitation	Explanation
Speed	The speed at which data can be encrypted or decrypted depends upon several hardware and software factors, and in some instances, a slower speed is unacceptable.
Size	The resulting size of an encrypted file can be as much as one-third larger than the plaintext.
Weak keys	Some ciphers can produce a **weak key** that causes the cipher to behave in unpredictable ways or may compromise overall security.
Key length	Some ciphers have a short **key length**, or the number of bits in a key, which results in weaker security.
Longevity	As computers continue to become more powerful and can "crack" keys, the **longevity** or useful lifetime of service of ciphers may diminish.
Predictability	A weak random number generator or PRNG of the cipher may create predictable output.
Reuse	If someone reuses the same key for each encryption, then it provides a larger data footprint for an attacker to use in attempting to break the encryption.
Entropy	**Entropy** is the measure of randomness of a data-generating function, and ciphers with low entropy give the ability to predict future-generated values.
Computational overhead	Sensors and Internet of Things (IoT) devices often lack the capacity to accommodate the computational overhead for cryptography.

Not all constraints prevent a device from using cryptography; in some instances, encrypting and decrypting may simply slow the device. Therefore, cryptography must have **high resiliency**, or the ability to quickly recover from these constraints. Due to the importance of incorporating cryptography in low-power devices, a new subfield of cryptography called **lightweight cryptography** is being developed. Lightweight cryptography has the goal of providing cryptographic solutions uniquely tailored to low-power devices that need to manage resource vs. security constraints. However, lightweight cryptography is not a weakened cryptography but may simply have fewer features and be less robust than normal cryptography.

NOTE 7

Cryptography tools are also used by threat actors in their attacks. In one recent attack, victims who opened a malicious email attachment triggered a macro that executed a PowerShell script. The script instructed the user's device to visit a legitimate website to download an image. The image contained malicious code that was hidden within the image using steganography and was also encrypted. In a clever move, the script contained an intentional error, and the error code that was returned by the OS was the decryption key for opening the encrypted malware.

TWO RIGHTS & A WRONG

1. Steganography hides the existence of information.
2. Unencrypted data that is input for encryption or is the output of decryption is called cleartext.
3. Entropy is the measure of randomness of a data-generating function.

See Appendix B for the answer.

CRYPTOGRAPHIC ALGORITHMS

CERTIFICATION

2.8 Summarize the basics of cryptographic concepts.

One variation of cryptographic algorithms is based on the device—if any—that is used in the cryptographic process. During the last half of the twentieth century, all cryptography became computer-based, whereas for the first half of that century, people used calculating machines. Before that time, cryptographic algorithms were entirely hand calculated. An example is a *one-time pad (OTP)*, which combines plaintext with a random key. A *pad* is a long sequence of random letters. The letters are combined with the plaintext message to produce the ciphertext. To decipher the message, the recipient must have a copy of the pad to reverse the process.

To encipher a message, the position in the alphabet of the first letter in the plaintext message is added to the position in the alphabet of the first random letter from the pad. For example, to encrypt *SECRET* using the pad *CBYFEA*, the first letter *S* (#19 of the alphabet) is added to the first letter of the pad *C* (#3 of the alphabet), and then 1 is subtracted (19 + 3 − 1 = 21). The result is *U* (letter 21 in the alphabet). Each letter is similarly encrypted, with any number larger than 26 wrapping around to the start of the alphabet. To decipher a message, the recipient takes the first letter of the ciphertext and subtracts the first random letter from the pad (any negative numbers are wrapped around to the end of the alphabet). An OTP is illustrated in Table 6-3.

Table 6-3 OTP

Plaintext	Position in alphabet	Pad	Position in alphabet	Calculation	Result
S	19	C	3	19 + 3 − 1 = 21	U
E	5	B	2	5 + 2 − 1 = 6	F
C	3	Y	25	3 + 25 − 1 = 1	A
R	18	F	6	18 + 6 − 1 = 23	W
E	5	E	5	5 + 5 − 1 = 9	I
T	20	A	1	20 + 1 − 1 = 20	T

CAUTION As its name implies, the one-time pad should be used only one time and then destroyed. Because OTP is hand calculated and is the only known encryption method that cannot be broken mathematically, OTPs were used by special operations teams and resistance groups during World War II as well as by intelligence agencies and spies during the Cold War.

Another variation in cryptographic algorithms is the amount of data that is processed at a time. Some algorithms use a **stream cipher** that takes one character and replaces it with one character. Other algorithms make use of a **block cipher**. Whereas a stream cipher works on one character at a time, a block cipher manipulates an entire block of plaintext at one time. The plaintext message is divided into separate blocks of 8 to 16 bytes, and then each block is encrypted independently. For additional security, the blocks can be randomized. Recently, a third type called a *sponge function* has been introduced. A sponge function takes as input a string of any length and returns a string of any requested variable length. This function repeatedly applies a process on the input that has been *padded* with additional characters until all characters are used (*absorbed* in the *sponge*).

CAUTION Stream ciphers are less secure because the engine that generates the stream does not vary; the only change is the plaintext itself. Block ciphers are considered more secure because the output is more random, as the cipher is reset to its original state after each block is processed.

The three broad categories of cryptographic algorithms are hash algorithms, symmetric cryptographic algorithms, and asymmetric cryptographic algorithms.

Hash Algorithms

One type of cryptographic algorithm is a one-way hash algorithm. A **hash** algorithm creates a unique "digital fingerprint" of a set of data. This process is called **hashing**, and the resulting fingerprint is a *digest* (sometimes called a *message digest* or *hash*) that represents the contents. Hashing is used primarily for comparison purposes.

Although hashing is a cryptographic algorithm, its purpose is *not* to create ciphertext that can later be decrypted. Instead, hashing is intended to be one-way in that its digest cannot be reversed to reveal the original set of data. For example, when 12 is multiplied by 34, the result is 408. If a user were asked to determine the two numbers used to create the number 408, it would not be possible to work backward and derive the original numbers with absolute certainty because there are too many mathematical possibilities (1×408, 2×204, 3×136, 4×102, etc.). Hashing is similar in that it is not possible to determine the plaintext from the digest.

NOTE 8

Although hashing and checksums are similar in that they both create a value based on the contents of a file, hashing is not the same as creating a checksum. A checksum is intended to verify (*check*) the integrity of data and identify data-transmission errors, while a hash is designed to create a unique digital fingerprint of the data.

A hashing algorithm is considered secure if it has the following characteristics:

- *Fixed size.* A digest of a short set of data should produce the same size as a digest of a long set of data. For example, a digest of the single letter *a* is 86be7afa339d0fc7cfc785e72f578d33, while a digest of one million occurrences of the letter *a* is 4a7f5723f954eba1216c9d8f6320431f, the same length.
- *Unique.* Two different sets of data cannot produce the same digest. Changing a single letter in one data set should produce an entirely different digest. For example, a digest of *Sunday* is 0d716e73a2a7910bd4ae63407056d79b while a digest of *sunday* (lowercase *s*) is 3464eb71bd7a4377967a30da798a1b54.
- *Original.* It should not be possible to produce a data set that has a desired or predefined hash.
- *Secure.* The resulting hash cannot be reversed to determine the original plaintext.

Hashing is often used as a check to verify that the original contents of an item have not been changed. For example, digests are often calculated and then posted on websites for files that can be downloaded, as seen in Figure 6-4. After downloading the file, users can create their own digest on the file and then compare it with the digest value posted on the website. A match indicates the original file did not change while it was being downloaded.

Image Name	Torrent	Version	Size	SHA256Sum
Kali Linux 64-Bit (Installer)	Torrent	2020.2	3.6G	ae9a3b6a1e016cd464ca31ef5055506cecfc55a10f61bf1acb8313eddbe12ad7
Kali Linux 64-Bit (Live)	Torrent	2020.2	2.9G	e90e0cfb4bc8fc640219dba66c9fe4308c9502164e432c47a30af50ce9cb3ba2
Kali Linux 64-Bit (NetInstaller)	Torrent	2020.2	420M	def160159e12ff52fb5f4991240bd760500d7cd5ee38601a8bf35809a20f9450

Source: Kali Linux

Figure 6-4 Verifying downloads with digests

Common hash algorithms include the following:

- *Message Digest (MD)*. One of the earliest hash algorithms is a "family" of algorithms known as Message Digest (MD). Versions of MD hashes were introduced over almost 20 years, from MD2 (1989) to MD6 (2008). The most widely used of these algorithms is *MD5*. This hash algorithm uses four variables of 32 bits each in a round-robin fashion to create a value that is then compressed. Serious weaknesses have been identified in MD5, and it is no longer considered suitable for use.
- *Secure Hash Algorithm (SHA)*. Another family of hashes is the Secure Hash Algorithm (SHA). *SHA-1* was developed in 1993 but is no longer considered suitable for use. *SHA-2* has six variations, the most common are SHA-256, SHA-384, and SHA-512 (the last number indicates the length in bits of the digest that is generated) and is currently considered a secure hash. In 2015, after eight years of competition between 51 original entries, *SHA-3* was announced as a new standard. One design goal of SHA-3 was to make it dissimilar to previous hash algorithms to prevent threat actors from building on earlier work of compromising the algorithms.
- *RIPEMD*. *RIPEMD* stands for RACE Integrity Primitives Evaluation Message Digest. The primary design feature of RIPEMD is two different and independent parallel chains of computation, the results of which are then combined at the end of the process. All versions of RIPEMD are based on the length of the digest created, including RIPEMD-128, RIPEMD-256, and RIPEMD-320.

Table 6-4 illustrates the digests generated from several one-way hash algorithms using the word *Cengage*.

Table 6-4 Digests generated from one-time hash algorithms

Hash	Digest
MD2	b365b3f6ca8b35460782a658f2e82009
MD4	4fe043f7a0cde169a5d069b46bcfc0f7
MD5	7e169c6f44088e315c7b1f6513c1b0f7
RipeMD160	dd52a79bce64a1d145b51ce639e0dadda976516d
SHA-1	963ea98f0af1927e02ed0f13786162a5b8e713c0
SHA-256	f6c8a86bf6a5128cbaf2ad251b0beaa3604c11c51587de518737537800098d76
SHA-512	a2221473f8c4737d97f44265b3731a61127cc9521d4e07d18b1be357df4f04f8367a4a5255 cae956611f71b426bf494f2a68f41ca4aa122c2a07b570880e81fd
SHA3-512	3a82d58e17f3991413c5f4e9811930b69513bba02a860eed82070f892ab381f9fd926a88cf68745 565f51a93b97a1317ae8b84e2dfb798e4a2aa331187dc9e34

Symmetric Cryptographic Algorithms

The original cryptographic algorithms for encrypting and decrypting data are symmetric cryptographic algorithms. **Symmetric cryptographic algorithms** use the same key to encrypt and decrypt the data. Data encrypted by Bob with a key can only be decrypted by Alice using that same key. Because the key must be kept private (confidential), symmetric encryption is also called *private key cryptography*. Symmetric encryption is illustrated in Figure 6-5, where identical keys are used to encrypt and decrypt a document.

Symmetric cryptography can provide strong encryption—if the key is kept secure between the sender and all the recipients. Common symmetric cryptographic algorithms include the following:

- *Data Encryption Standard (DES)*. One of the first widely used symmetric cryptography algorithms was the *Data Encryption Standard (DES)*. The U.S. government officially adopted DES as the standard for encrypting unclassified information. Although DES was once widely implemented, it is no longer considered suitable for use.

NOTE 9

DES effectively catapulted the study of cryptography into the public arena. Until the deployment of DES, cryptography was studied almost exclusively by military personnel. The popularity of DES helped move cryptography implementation and research to academic and commercial organizations.

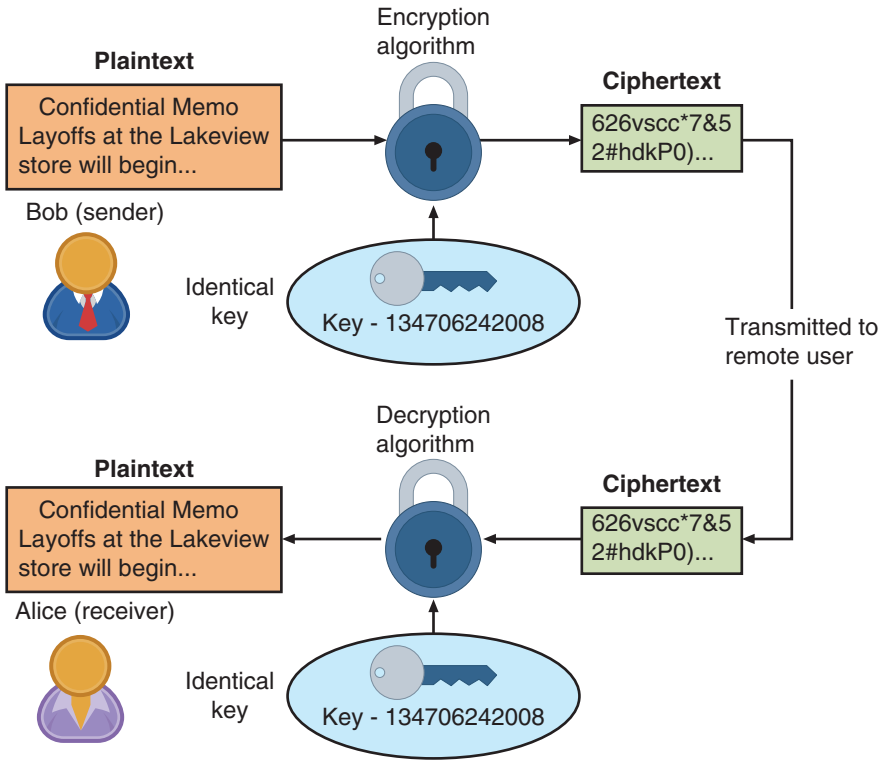


Figure 6-5 Symmetric (private key) cryptography

- *Triple Data Encryption Standard (3DES). Triple Data Encryption Standard (3DES)* was designed to replace DES. As its name implies, 3DES uses three rounds of encryption instead of just one. The ciphertext of one round becomes the entire input for the second iteration. 3DES employs a total of 48 iterations in its encryption (3 iterations × 16 rounds). The most secure versions of 3DES use different keys for each round, as shown in Figure 6-6. Although 3DES addresses several of the key weaknesses of DES, it is no longer considered the most secure symmetric cryptographic algorithm.
- *Advanced Encryption Standard (AES).* The *Advanced Encryption Standard (AES)* is a symmetric algorithm that performs three steps on every block (128 bits) of plaintext. Within step 2, multiple rounds are performed depending upon the key size: a 128-bit key performs nine rounds, a 192-bit key performs 11 rounds, and a 256-bit key, known as AES-256, uses 13 rounds. Within each round, bytes are substituted and rearranged, and then special multiplication is performed based on the new arrangement. To date, no attacks have been successful against AES.
- *Rivest Cipher (RC).* Rivest Cipher (RC) is a family of six algorithms. *RC4*, the most common RC cipher, is a stream cipher that accepts keys up to 128 bits in length.
- *Blowfish. Blowfish* is a block cipher algorithm that operates on 64-bit blocks and can have a key length from 32 to 448 bits. To date, no significant weaknesses have been identified. A later derivation of Blowfish known as Twofish is also considered a strong algorithm, although it has not been used as widely as Blowfish.

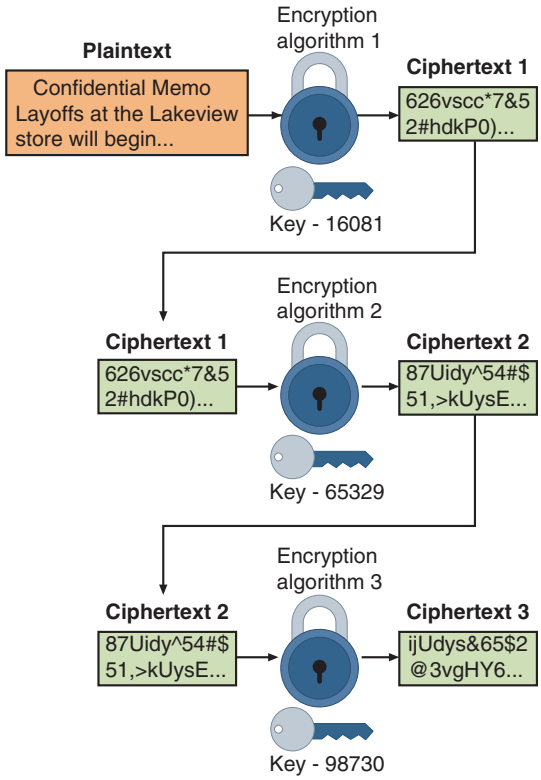


Figure 6-6 3DES

Asymmetric Cryptographic Algorithms

If Bob wants to send an encrypted message to Alice using symmetric encryption, he must be sure that she has the key to decrypt the message. Yet how should Bob get the key to Alice? He cannot send it electronically as an email attachment, because that would make it vulnerable to interception by attackers. Nor can he encrypt the key and send it, because Alice would not have a way to decrypt the encrypted key. This problem illustrates the primary weakness of symmetric encryption algorithms: distributing and maintaining a secure single key among multiple users, who are often scattered geographically, poses significant challenges.

A completely different approach is **asymmetric cryptographic algorithms**, also known as *public key cryptography*. Asymmetric encryption uses two keys instead of only one. The keys are mathematically related and are known as the public key and the private key. The public key is known to everyone and can be freely distributed, while the private key is known only to the individual to whom it belongs. When Bob wants to send a secure message to Alice, he uses Alice's public key to encrypt the message. Alice then uses her private key to decrypt it. Asymmetric cryptography is illustrated in Figure 6-7.

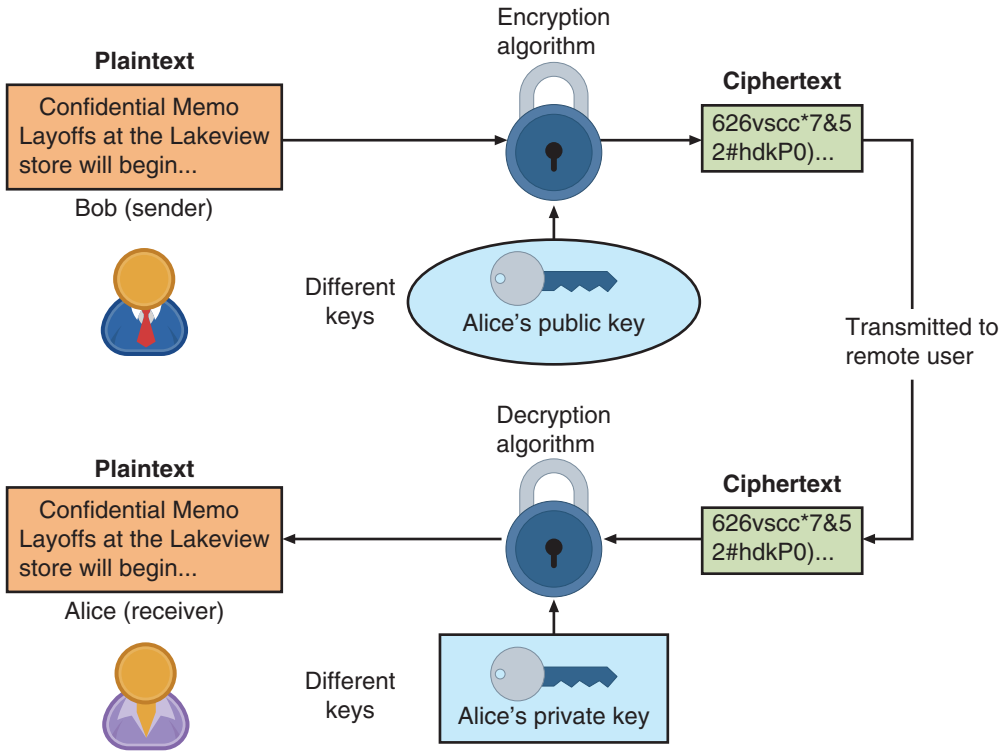


Figure 6-7 Asymmetric (public key) cryptography

NOTE 10

Although different cryptographers were working on the idea of asymmetric encryption in the early 1970s, the development is often credited to Whitfield Diffie and Martin Hellman, based on a publication of their paper *New Directions in Cryptography* in November 1976.

Several important principles regarding asymmetric cryptography are as follows:

- *Key pairs*. Unlike symmetric cryptography that uses only one key, asymmetric cryptography requires a pair of keys.
- *Public key*. Public keys, by their nature, are designed to be public and do not need to be protected. They can be freely given to anyone or even posted on the Internet.
- *Private key*. The private key must be kept confidential and never shared.
- *Both directions*. Asymmetric cryptography keys can work in both directions. A document encrypted with a public key can be decrypted with the corresponding private key. In the same way, a document encrypted with a private key can be decrypted with its public key.

CAUTION No user other than the owner must ever have the private key.

There are different asymmetric algorithms and variations as well as issues surrounding key management.

RSA

The asymmetric algorithm *RSA* was published in 1977 and became the basis for several products. The RSA algorithm multiplies two large prime numbers (a prime number is a number divisible only by itself and 1), p and q, to compute their product ($n = pq$). Next, a number e is chosen that is less than n and a prime factor to $(p - 1)(q - 1)$. Another number d is determined so that $(ed - 1)$ is divisible by $(p - 1)(q - 1)$. The values of e and d are the public and private exponents. The public key is the pair (n,e) while the private key is (n,d). The numbers p and q can be discarded.

An illustration of the RSA algorithm using very small numbers is as follows:

1. Select two prime numbers, p and q (in this example, $p = 7$ and $q = 19$).
2. Multiply p and q together to create n ($7 \times 19 = 133$).
3. Calculate m as $p - 1 \times q - 1$ ($[7 - 1] \times [19 - 1]$ or $6 \times 18 = 108$).
4. Find a number e so that it and m have no common positive divisor other than 1 ($e = 5$).
5. Find a number d so that $d = (1 + n \times m)/e$ or ($[1 + 133 \times 108]/5$ or $14{,}364/5 = 2{,}875$).

For this example, the public key n is 133 and e is 5, while for the private key n is 133 and d is 2873.

Elliptic Curve Cryptography (ECC)

The basis of RSA asymmetric encryption security is factoring, or the prime numbers that make up a value. As computers become faster and more powerful, the ability to "crack" RSA asymmetric encryption by computing the factoring has grown. Instead of using factoring as the basis, other research looked at an obscure (and esoteric) branch of mathematics called elliptic curves. In short, an elliptic curve is a set of points that satisfy a specific mathematical equation. Elliptic curves paved the way for a different form of asymmetric encryption not based on factoring.

Instead of using large prime numbers as with RSA, **elliptic curve cryptography (ECC)** uses sloping curves. An elliptic curve is a function drawn on an X-Y axis as a gently curved line. By adding the values of two points on the curve, a third point on the curve can be derived, of which the inverse is used, as illustrated in Figure 6-8. With ECC, users share one elliptic curve and one point on the curve. One user chooses a secret random number and computes a public key based on a point on the curve; the other user does the same. They can now exchange messages because the shared public keys can generate a private key on an elliptic curve.

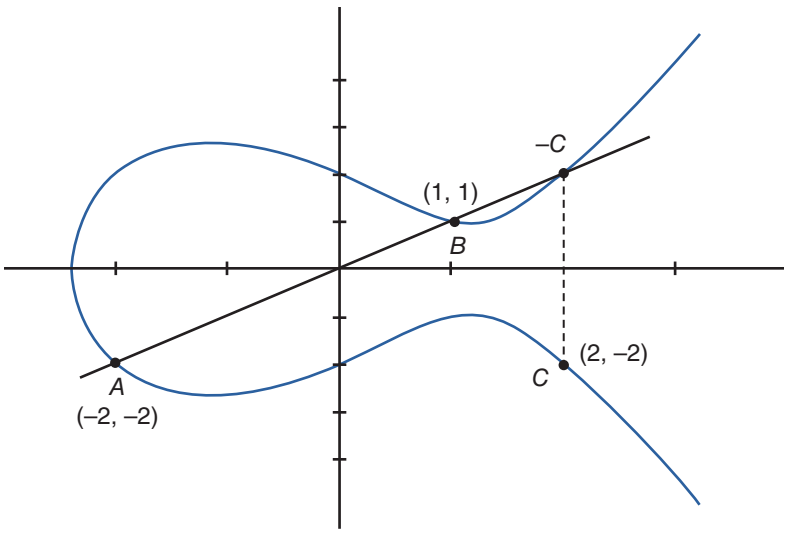


Figure 6-8 Elliptic curve cryptography (ECC)

The difference in size necessary to produce the same level of security between RSA and ECC keys is significant. Table 6-5 compares the key length (in bits) of RSA and ECC keys that have the same level of security. Despite a slow start, ECC has gained wide popularity. It is used by the U.S. government to protect internal communications, by the Tor project to help assure anonymity, and as the mechanism to prove ownership of bitcoins. All modern OSs and web

browsers rely on ECC. Because mobile devices are limited in terms of computing power due to their smaller size, ECC offers security that is comparable to other asymmetric cryptography but with smaller key sizes, resulting in faster computations and lower power consumption.

NOTE 11

According to one study to break a 228-bit RSA key, the amount of energy needed would take less energy than what is required to boil a teaspoon of water. However, breaking a 228-bit ECC key would require more energy than it would take to boil all the water on Earth.[2]

Table 6-5 RSA vs. ECC key length for same security level

RSA key length	ECC key length
1,024	160
2,048	224
3,072	256
7,680	384
15,360	521

Digital Signature Algorithm (DSA)

Asymmetric cryptography also can be used to provide proofs. Suppose Alice receives an encrypted document that says it came from Bob. Although Alice can be sure that the encrypted message was not viewed or altered by someone else while being transmitted, how can she know for certain that Bob was the sender? Because Alice's public key is widely available, anyone could use it to encrypt the document. Another individual could have created a fictitious document, encrypted it with Alice's public key, and then sent it to Alice while pretending to be Bob. Alice's key can verify that no one read or changed the document in transport, but it cannot verify the sender.

Proof can be provided with asymmetric cryptography, however, by creating a *digital signature*, which is an electronic verification of the sender. A handwritten signature on a paper document serves as proof that the signer has read and agreed to the document. A digital signature is much the same but can provide additional benefits. A digital signature can

- *Verify the sender*. A digital signature serves to confirm the identity of the person from whom the electronic message originated.
- *Prevent the sender from disowning the message*. The signer cannot later attempt to disown it by claiming the signature was forged (nonrepudiation).
- *Prove the integrity of the message*. A digital signature can prove that the message has not been altered since it was signed.

The basis for a digital signature rests on the ability of asymmetric keys to work in both directions (a public key can encrypt a document that can be decrypted with a private key, and the private key can encrypt a document that can be decrypted by the public key).

The steps for Bob to send a digitally signed message to Alice are as follows:

1. After creating a memo, Bob generates a digest on it.
2. Bob encrypts the digest with his private key. The encrypted digest is the digital signature for the memo.
3. Bob sends both the memo and the digital signature to Alice.
4. When Alice receives them, she decrypts the digital signature using Bob's public key, revealing the digest. If she cannot decrypt the digital signature, then she knows that it did not come from Bob (because only Bob's public key can decrypt the digest generated with his private key).
5. Alice then hashes the memo with the same hash algorithm Bob used and compares the result to the digest she received from Bob. If they are equal, Alice can be confident that the message has not changed since he signed it. If the digests are not equal, Alice will know the message has changed since it was signed.

These steps are illustrated in Figure 6-9.

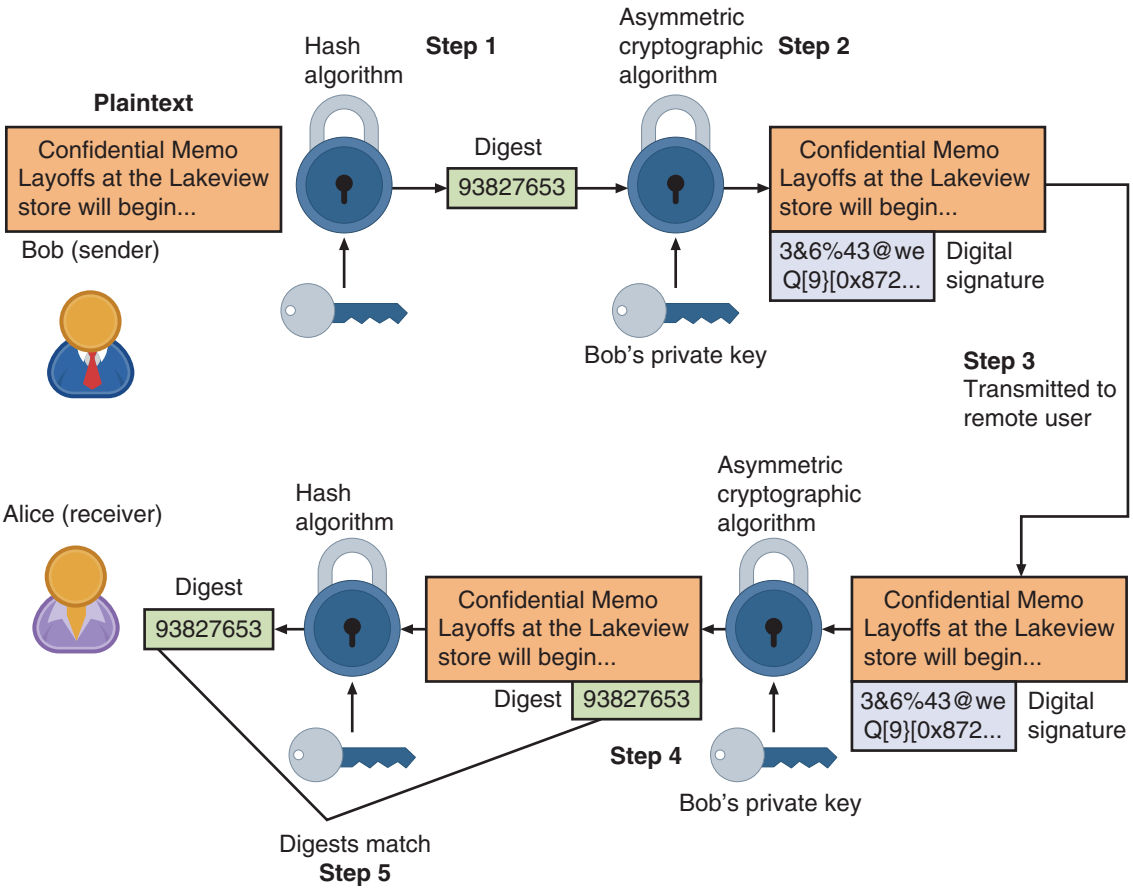


Figure 6-9 Digital signature

CAUTION Using a digital signature does not encrypt the message itself. In the example, if Bob wanted to ensure the privacy of the message, he also would have to encrypt it using Alice's public key.

The *Digital Signature Algorithm (DSA)* is a U.S. federal government standard for digital signatures. DSA was proposed by NIST in 1991 for use in their Digital Signature Standard (DSS). Although patented, NIST has made the patent available worldwide royalty-free. The standard continues to be revised and updated periodically by NIST.

Key Exchange

Public and private keys may result in confusion regarding whose key to use and which key should be used. Table 6-6 lists the practices to follow when using asymmetric cryptography.

Table 6-6 Asymmetric cryptography practices

Action	Whose key to use	Which key to use	Explanation
Bob wants to send Alice an encrypted message.	Alice's key	Public key	When an encrypted message is to be sent, the recipient's, and not the sender's, key is used.
Alice wants to read an encrypted message sent by Bob.	Alice's key	Private key	An encrypted message can be read only by using the recipient's private key.

(continues)

Table 6-6 Asymmetric cryptography practices *(continued)*

Action	Whose key to use	Which key to use	Explanation
Bob wants to send a copy to himself of the encrypted message that he sent to Alice.	Bob's key	Public key to encrypt Private key to decrypt	An encrypted message can be read only by the recipient's private key. Bob would need to encrypt it with his public key and then use his private key to decrypt it.
Bob receives an encrypted reply message from Alice.	Bob's key	Private key	The recipient's private key is used to decrypt received messages.
Bob wants Susan to read Alice's reply message that he received.	Susan's key	Public key	The message should be encrypted with Susan's key for her to decrypt and read with her private key.
Bob wants to send Alice a message with a digital signature.	Bob's key	Private key	Bob's private key is used to encrypt the hash.
Alice wants to see Bob's digital signature.	Bob's key	Public key	Because Bob's public and private keys work in both directions, Alice can use his public key to decrypt the hash.

In addition to confusion regarding which key to use, there are also issues with sending and receiving keys (**key exchange**) such as exchanging a symmetric private key. One solution is to make the exchange outside of the normal communication channels (for example, Alice could hire Charlie to carry a USB flash drive containing the key directly to Bob).

Solutions for a key exchange that occurs within the normal communications channel of cryptography include the following:

- *Diffie-Hellman (DH).* The Diffie-Hellman (DH) key exchange requires Alice and Bob to each agree upon a large prime number and related integer. Those two numbers can be made public, yet Alice and Bob, through mathematical computations and exchanges of intermediate values, can separately create the same key.
- *Diffie-Hellman Ephemeral (DHE).* Whereas DH uses the same keys each time, Diffie-Hellman Ephemeral (DHE) uses different keys. **Ephemeral keys** are temporary keys that are used only once and then discarded.
- *Elliptic Curve Diffie–Hellman (ECDH).* Elliptic Curve Diffie–Hellman (ECDH) uses elliptic curve cryptography instead of prime numbers in its computation.
- *Perfect forward secrecy.* Public key systems that generate random public keys that are different for each session are called **perfect forward secrecy**. The value of perfect forward secrecy is that if the secret key is compromised, it cannot reveal the contents of more than one message.

TWO RIGHTS & A WRONG

1. A digest of a short set of data should produce the same size as a digest of a long set of data.
2. SHA-1 is considered a secure hash algorithm.
3. Asymmetric cryptography keys can work in both directions.

See Appendix B for the answer.

CRYPTOGRAPHIC ATTACKS AND DEFENSES

CERTIFICATION

1.2 Given a scenario, analyze potential indicators to determine the type of attack.

2.8 Summarize the basics of cryptographic concepts.

Because cryptography provides a high degree of protection, it is a defense that remains under attack by threat actors for any vulnerabilities. However, the new field of quantum cryptography defenses can aid in making cryptography more secure.

Attacks on Cryptography

Two of the most common types of attacks on cryptography include algorithm attacks and collision attacks.

Algorithm Attacks

Modern cryptographic algorithms are reviewed, tested, and vetted by specialists in cryptography over several years before they are released to the public for use. Very few threat actors have the advanced skills needed to even attempt to break an algorithm. However, attackers can use other methods to circumvent strong algorithms—including known ciphertext attacks, downgrade attacks, and taking advantage of improperly implemented algorithms.

Known Ciphertext Attacks When properly implemented, cryptography prevents threat actors from knowing the plaintext or the key; the only item they can see is the ciphertext itself. Yet they can use sophisticated statistical tools to analyze the ciphertext and discover a pattern in the ciphertexts, which may be useful in revealing the plaintext text or key. This type of attack is called a *known ciphertext attack* or *ciphertext-only attack*, because all that is known is the ciphertext, though it can still reveal clues that may be mined.

Wireless data networks are particularly susceptible to known ciphertext attacks. Threat actors can capture large sets of ciphertexts to analyze and then inject their own frames into the wireless transmissions.

NOTE 12

Two techniques can be added to a cryptographic algorithm to thwart known ciphertext attacks by making the ciphertext more difficult to analyze. *Diffusion* changes a single character of plaintext into multiple characters of ciphertext, while *confusion* makes each character of the ciphertext based upon several parts of the key.

Downgrade Attack Because new hardware and software are introduced frequently, they often include *backwards compatibility* so that a newer version can still function with the older version. However, in most instances, the newer version must revert to the older and less secure version. In a **downgrade attack**, an attacker forces the system to abandon the current higher security mode of operation and instead "fall back" to implementing an older and less secure mode. The threat actor can then attack the weaker mode.

Attacks Based on Misconfigurations Most breaches of cryptography are the result of incorrect choices or misconfigurations of the cryptography options, known as misconfiguration implementation. Selecting weak algorithms, such as DES or SHA-1, should be avoided since they are no longer secure. Many cryptographic algorithms have several configuration options, and unless careful consideration is given to the options during setup, the cryptography may be improperly implemented. Careless users can also weaken cryptography if they choose SHA-256, for example, when a much stronger SHA3-512 is available through a simple menu choice.

Collision Attacks

One of the foundations of a hash algorithm is that each digest must be unique. If it were not unique, then a threat actor could trick users into performing an action that was assumed to be safe but in reality was not. For example, digests are calculated and then posted on websites for files that can be downloaded. Suppose an attacker could infiltrate a website and post a malicious file for download, but when the digest was generated for the malicious file, it created the same digest as the one posted for the legitimate file. Two files having the same digest is known as a **collision**. A *collision attack* is an attempt to find two input strings of a hash function that produce the same hash result.

NOTE 13

While for hash algorithms that produce long digests, such as SHA3-512, the odds of such a collision are very low, yet for hash algorithms that produce shorter digests, such as MD5, the odds increase. Table 6-4 compares the length of various digests.

Typically, a threat actor would be forced to try all possible combinations until a collision was found. However, a statistical phenomenon called the **birthday attack** makes it easier. It is based on the *birthday paradox*, which says that for someone in a given room to have a 50 percent chance of sharing your birthday, 253 people would need to be in the room. If, however, you are looking for a greater than 50 percent chance that any two people in the room have the same birthday, you only need 23 people. That's because the matches are based on pairs. If you choose yourself as one side of the pair, then you need 253 people to have 253 pairs (in other words, you combined with 253 other people make up all 253 sets). If you are concerned only with matches and not with matching someone to you specifically, then you need only 23 people in the room, because 23 people can form 253 pairs when cross-matched with each other. The same situation applies to hashing collisions. It is much harder to find something that collides with a specific hash than it is to find two inputs that hash to the same value.

NOTE 14

With the birthday paradox, the question is whether each person must link with every other person. If so, only 23 people are needed; if not, comparing only your single birthday to everyone else's, 253 people are needed.

Quantum Cryptographic Defenses

The foundation of modern technology is the bit (binary digit). A bit can either be off (0) or on (1), much like a light bulb. However, for several years, researchers have been developing a revolutionary type of computer called a **quantum computer**. Quantum computing relies on quantum physics using atomic-scale units (*qubits*) that can be both 0 and 1 at the same time. As a result, one qubit can carry out two separate streams of calculations simultaneously, meaning that quantum computers will be much faster and more efficient than today's computers.

NOTE 15

Businesses today are preparing for the move to quantum computers. Companies in the financial services, automotive, and pharmaceutical sectors have already started experimenting with quantum computing. By 2023, 20 percent of organizations, including businesses and governments, are expected to budget for quantum-computing projects, up from less than 1 percent in 2018.

Although quantum computers are potentially much more powerful than traditional computers, they are also more delicate and prone to faults. Because the technology is still in its early stages, no commercial-grade quantum computer has been built yet. However, online cloud quantum computing platforms are available to provide developers with an online platform to create applications for quantum computers. Developers can also experiment with quantum algorithms on traditional computers.

NOTE 16

Microsoft has eight quantum computing labs around the world. The company is also developing its own quantum computer that relies on topology, a branch of mathematics that studies geometric objects that experience physical changes. Microsoft says the topological approach can help a quantum computer run algorithms more reliably, with fewer risks of temperature or noise impacting the accuracy of a calculation or even preventing the calculation from being completed.

Quantum cryptography takes advantage of quantum computing for increasing cybersecurity. One subcategory of quantum cryptography is **quantum communication** or secure telecommunications. Because eavesdropping changes the physical nature of the information, users in a quantum communication exchange can easily detect eavesdroppers.

NOTE 17

The most well-known and developed application of quantum communication is *quantum key distribution (QKD)*. Bob and Alice are linked together with a quantum channel and a regular channel, such as the Internet. Bob and Alice each use single photons (a type of elementary particle) that are randomly polarized to states representing ones and zeroes, and the photons are used to transmit a series of random number sequences that are then used as keys in cryptographic communications. Alice begins by generating a random stream of qubits that are sent over the quantum channel. Upon reception of the stream, Bob and Alice, using the regular channel, perform operations to check if an attacker has tried to extract information on the qubits stream. The presence of an eavesdropper is revealed by the imperfect correlation between the two lists of bits obtained after the transmission of qubits between them.

Quantum computing also has a drawback for cybersecurity. Asymmetric cryptography begins by multiplying two prime numbers, a strong method because it is difficult for today's computers to determine the prime numbers that make up the value (*factoring*). However, a single quantum computer could perform factoring by using hundreds of atoms in parallel to quickly factor huge numbers, rendering virtually all current asymmetric cryptographic algorithms useless.

While some researchers think that quantum computers will create a "cryptographic apocalypse" soon (10 years from now), most security professionals think it is 30 years or more in the future more. Proposals for new "quantum-safe" encryption that could not be broken by quantum computers are currently being developed. Such encryption is called **post-quantum cryptography**, or cryptographic algorithms that are secure against an attack by a quantum computer.

TWO RIGHTS & A WRONG

1. In a downgrade attack, an attacker forces the system to abandon the current higher security mode of operation and instead "fall back" to implementing an older and less secure mode.
2. Post-quantum cryptography is comprised of algorithms that are secure against an attack by a quantum computer.
3. The basis of a quantum computer is a bit.

See Appendix B for the answer.

USING CRYPTOGRAPHY

CERTIFICATION

2.1 Explain the importance of security concepts in an enterprise environment.

2.8 Summarize the basics of cryptographic concepts.

3.2 Given a scenario, implement host or application security solutions.

Cryptography can be applied through either software or hardware. Also, a relatively new technology known as blockchain uses cryptography as its basis.

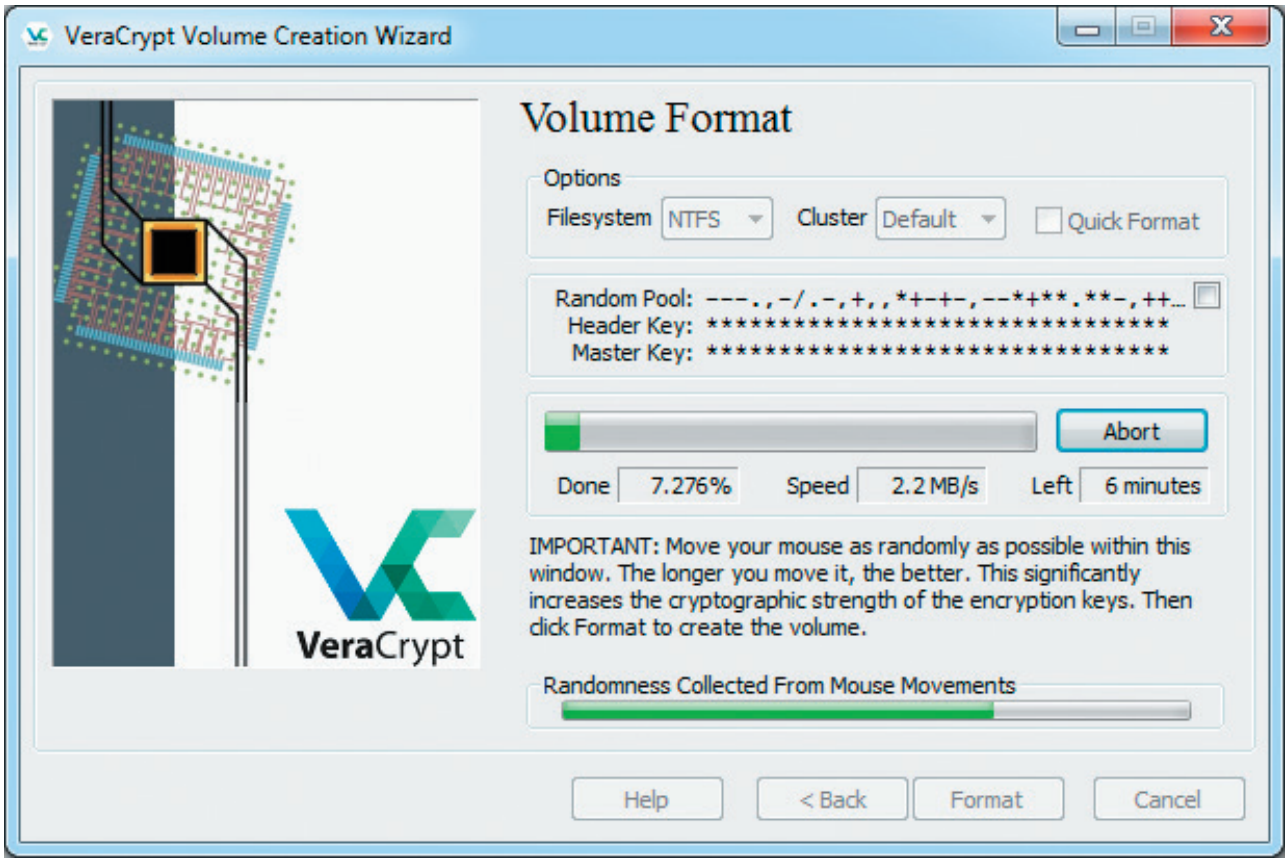
Encryption through Software

Cryptography can be implemented through software running on a device. Encryption can also be performed on a larger scale by encrypting the entire disk drive itself.

File and File System Cryptography

Cryptographic software can be used to encrypt or decrypt files one by one. However, this process can be cumbersome. Instead, protecting groups of files, such as all files in a specific folder, can take advantage of the OS's file system. A *file system* is a method used by an OS to store, retrieve, and organize files. Protecting individual files or multiple files through file system cryptography can be performed using third-party software or OS cryptographic features.

Third-Party Software Third-party software tools available for performing encryption include GNU Privacy Guard (which is abbreviated GNuPG), AxCrypt, Folder Lock, and VeraCrypt, shown in Figure 6-10.



Source: VeraCrypt

Figure 6-10 VeraCrypt

Operating System Encryption Modern OSs provide encryption support natively. Microsoft's *Encrypting File System (EFS)* is a cryptography system for Windows releases that use the Windows NT file system (NTFS), while Apple's *FileVault* performs a similar function. Because the technology is tightly integrated with the file system, file encryption and decryption are transparent to the user. Any file created in an encrypted folder or added to an encrypted folder is automatically encrypted. When an authorized user opens a file, it is decrypted as data is read from a disk; when a file is saved, the OS encrypts the data as it is written to a disk.

Full Disk Encryption

Cryptography can also be applied to entire disks instead of individual files or groups of files. This practice is known as *full disk encryption (FDE)* and protects all data on a hard drive. One example of full disk encryption software is that included in Microsoft Windows known as *BitLocker* drive encryption software. BitLocker encrypts the entire system volume, including the Windows Registry and any temporary files that might hold confidential information. BitLocker prevents attackers from accessing data by booting from another OS or placing the hard drive in another computer.

Hardware Encryption

Software encryption suffers from the same fate as any application program: it can be subject to attacks to exploit its vulnerabilities. As a more secure option, cryptography can be embedded in hardware. Hardware encryption cannot be exploited like software encryption. Hardware encryption can be applied to USB devices and standard hard drives. More sophisticated hardware encryption options include self-encrypting drives, the trusted platform module, and the hardware security model.

USB Device Encryption

Many instances of data leakage are the result of USB flash drives being lost or stolen. Although data can be secured with software-based cryptographic application programs, vulnerabilities in the programs can open the door for attackers to access the data.

As an alternative, encrypted hardware-based USB devices such as flash drives can be used to prevent these types of software-based attacks. The drives resemble standard USB flash drives, with the following significant differences:

- Encrypted hardware-based USB drives will not connect to a computer until the correct password has been provided.
- All data copied to the USB flash drive is automatically encrypted.
- The external cases are designed to be tamper-resistant so attackers cannot disassemble the drives.
- Administrators can remotely control and track activity on the devices.
- Compromised or stolen drives can be remotely disabled.

NOTE 18

One hardware-based USB encrypted drive allows administrators to remotely prohibit accessing the data on a device until it can verify its status, to lock out the user completely the next time the device connects, or even to instruct the drive to initiate a self-destruct sequence to destroy all data.

Self-Encrypting Drives (SED)

Just as an encrypted hardware-based USB flash drive will automatically encrypt any data stored on it, **self-encrypting drives (SEDs)** can protect all files stored on them. When the computer or other device with an SED is initially powered up, the drive and the host device perform an authentication process. If the authentication process fails, the drive can be configured to simply deny any access to the drive or even perform a *cryptographic erase* on specified blocks of data. (A cryptographic erase deletes the decryption keys so that no data can be recovered.) It is also impossible to install the drive on another computer to read its contents.

A set of specifications for SEDs developed by the Trusted Computing Group (TCG) is **Opal**. SEDs that support Opal use hardware encryption technology to secure data stored in them. Opal also ensures the interoperability of SEDs among vendors.

Hardware Security Module (HSM)

A **Hardware Security Module (HSM)** is a removable external cryptographic device. An HSM can be a USB device, an expansion card, a device the connects directly to a computer through a port, or a secure network server. An HSM includes an onboard random number generator and key storage facility, as well as accelerated symmetric and asymmetric encryption, and can even back up sensitive material in encrypted form. Because the security is based on hardware and not through software, malware cannot compromise it.

HSMs are popular consumer-level devices. Figure 6-11 shows a USB consumer HSM from Yubico, a provider of authentication and encryption hardware devices. Some financial banking software comes with a specialized HSM hardware key, also called a "security dongle." The device is paired with a specific financial account and cannot be cloned or compromised.

Trusted Platform Module (TPM)

The **Trusted Platform Module (TPM)** is a chip on the motherboard of the computer that provides cryptographic services. For example, TPM includes a true random number generator instead of a PRNG as well as full support for asymmetric encryption. (TPM can also generate public and private keys.) Also, TPM can measure and test key

Source: Yubico

Figure 6-11 USB HSM

NOTE 19

An HSM is external while a TPM is internal.

components as the computer is starting up. It will prevent the computer from booting if system files or data have been altered. With TPM, if the hard drive is moved to another computer, the user must enter a recovery password before gaining access to the system volume.

Blockchain

Consider the company Serious Scooters (SS), which manufactures scooters. Several suppliers sell raw materials to Serious: tires, metal, plastic, seats, nuts, and washers, just to name a few. Serious sells their scooters to many retailers and then deposits the proceeds into their bank so that they can pay their suppliers, employees, and others. In the Accounting Department, a ledger is maintained of all transactions of raw materials that are bought and scooters that are sold. The ledger serves as the central repository of the accounting information for Serious Scooters.

Serious Scooters is not the only entity to keep a ledger: all of its suppliers, the retailers who buy from Serious, the banks, and everyone else also keep ledgers of their transactions with Serious as well as other customers. See Figure 6-12.

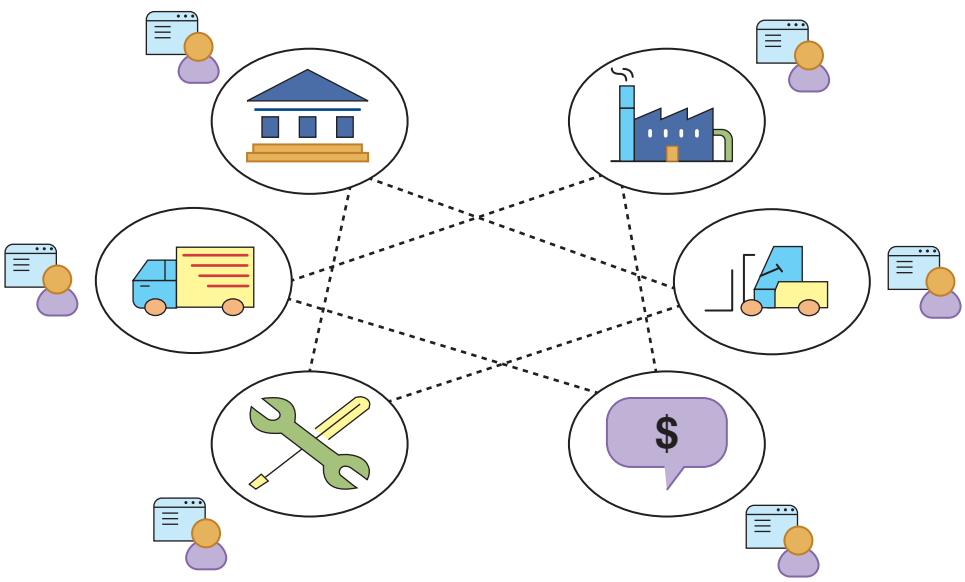


Figure 6-12 Multiple organizations with ledgers

It is inefficient for each entity to maintain its own ledger. Because suppliers, retailers, and banks duplicate parts of transactions, what happens if one supplier to Serious records a transaction in error? It may take a long time to correct the error because the supplier must contact Serious, who may need to contact its bank and even its retailers, to track down the error.

Instead, what if Serious Scooters and everyone else shared a single, tamper-evident ledger? It would eliminate or reduce paper processes, speeding up transaction times and increasing efficiencies. With a shared ledger, any transactions would be recorded only once and could not be altered. All parties must also give consensus before a new transaction is added to the network. Having a single shared ledger is seen in Figure 6-13.

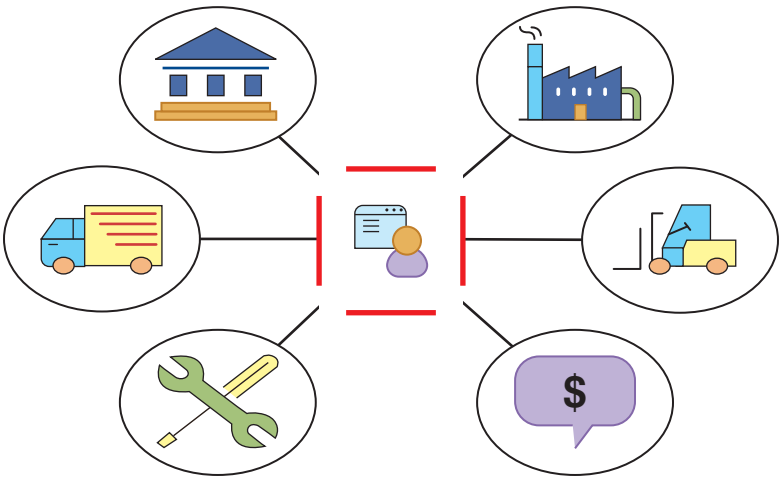


Figure 6-13 Multiple organizations using single ledger

This alternative is a **blockchain**. A blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network. At a high level, blockchain technology allows a network of computers to agree at regular intervals on the true state of a distributed ledger. It is a system in which a record of transactions made is maintained across several computers that are linked in a peer-to-peer network.

NOTE 20

Walmart has started using blockchain to manage its supply-chain data for produce. Instead of using bar codes, scanners, paper forms, and individual databases, Walmart uses a digital app to record in blockchain when fruit is picked, packed, and shipped to the store. Before implementing blockchain, it took Walmart 6 days, 18 hours, and 26 minutes to track a single bag of sliced mangoes using the manual system. After implementing blockchain, that same bag was tracked it in 2.2 seconds.

Blockchain relies on cryptographic hash algorithms, most notably the SHA-256, to record its transactions. This makes it computationally infeasible to try to replace a block or insert a new block of information without the approval of all entities involved.

TWO RIGHTS & A WRONG

1. Modern OSs provide encryption support natively.
2. Opal is a standard for FEDs.
3. An HSM is external while a TMP is internal.

See Appendix B for the answer.

VM LAB You're now ready to complete the live, virtual machine labs for this module. The labs can be found in each module in the MindTap.

SUMMARY

- Cryptography is the practice of transforming information into a secure form so that unauthorized persons cannot access it. Unlike steganography, which hides the existence of data, cryptography masks the content of data so that it cannot be read. The original data, called plaintext, is input into a cryptographic encryption algorithm that has a mathematical value (a key) used to create ciphertext. Of the categories and types of algorithms, a substitution cipher exchanges one character for another, as in ROT13, in which the entire alphabet is rotated 13 steps. Another common algorithm is the XOR cipher, which uses the binary operation e*X*clusive *OR* to compare two bits. The strength of a cryptographic algorithm depends upon several factors.
- Cryptography can provide confidentiality, integrity, authentication, nonrepudiation, and obfuscation. It can also protect data as it resides in any of three states: data in processing, data in transit, and data at rest. Yet despite providing these protections, cryptography faces constraints that can impact its effectiveness. Adding cryptography to low-power devices or those that have near-instantaneous response times can be a problem because the algorithms require both time and energy, which are typically in short supply for low-power devices and applications needing ultra-fast response times. This results in a resource vs. security constraint. Other constraints are speed, size, weak keys, key length, longevity, predictability, reuse, entropy, and computational overhead. Due to the importance of incorporating cryptography in low-power devices, a new subfield of cryptography called lightweight cryptography is being developed.
- One variation of a cryptographic algorithm is based on the device (if any) that is used in the cryptographic process. Another variation is the amount of data that is processed at a time. A stream cipher takes one character and replaces it with one character while a block cipher manipulates an entire block of plaintext at one time.
- Hashing creates a unique digital fingerprint called a digest, which represents the contents of the original material. Hashing is not designed for encrypting material that will be later decrypted. If a hash algorithm produces a unique fixed-size hash and the original contents of the material cannot be determined from the hash, the hash is considered secure. Common hashing algorithms are Message Digest, Secure Hash Algorithm, and RACE Integrity Primitives Evaluation Message Digest.
- Symmetric cryptography, also called private key cryptography, uses a single key to encrypt and decrypt a message. Symmetric cryptography can provide strong protections against attacks if the key is kept secure. Common symmetric cryptographic algorithms include Data Encryption Standard, Triple Data Encryption Standard, Advanced Encryption Standard, Rivest Cipher, and Blowfish.
- Asymmetric cryptography, also known as public key cryptography, uses two keys instead of one. The keys are mathematically related and are known as the public key and the private key. The public key is widely available and can be freely distributed, while the private key is known only to the recipient of the message and must be kept secure. Common asymmetric cryptographic algorithms include RSA, Elliptic Curve Cryptography, Digital Signature Algorithm, and those relating to Key Exchange.
- Because cryptography provides a high degree of protection, it remains under attack. A known ciphertext attack uses statistical tools to attempt to discover a pattern in the ciphertexts, which then may be useful in revealing the plaintext text or key. In a downgrade attack, a threat actor forces the system to abandon the current higher security mode of operation and instead fall back to implementing an older and less secure mode. Many breaches of cryptography are the result not of weak algorithms but instead of incorrect configuration or uses of the cryptography, known as misconfiguration implementation. When two files have the same digest, this is known as a collision. A collision attack is an attempt to find two input strings of a hash function that produce the same hash result.
- Quantum computing relies on quantum physics using atomic-scale units (*qubits*) that can be both 0 and 1 at the same time. As a result, it is possible for one qubit to carry out two separate streams of calculations simultaneously, so that quantum computers will be much faster and more efficient than today's computers. Quantum cryptography takes advantage of quantum computing for increasing cybersecurity. One subcategory of quantum cryptography is quantum communication or secure telecommunications. Because eavesdropping changes the physical nature of the information, users in a quantum communication exchange can easily detect eavesdroppers.
- Cryptography can be applied through either software or hardware. Software-based cryptography can protect large numbers of files on a system or an entire disk. Several third-party software tools are available. Modern OSs provide encryption support natively. Cryptography also can be applied to entire disks, known as full disk encryption (FDE).

- Hardware encryption cannot be exploited like software cryptography. Hardware encryption devices can protect USB devices and standard hard drives. More sophisticated hardware encryption options include self-encrypting drives, the Hardware Security Model, and the Trusted Platform Module.
- A blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network. At a high level, blockchain technology allows a network of computers to agree at regular intervals on the true state of a distributed ledger, and it is a system in which a record of transactions made are maintained across several computers that are linked in a peer-to-peer network.

Key Terms

algorithm
asymmetric cryptographic algorithm
birthday attack
block cipher
blockchain
collision
cryptography
data at rest
data in processing
data in transit
decryption
downgrade attack
elliptic curve cryptography (ECC)
encryption
entropy
ephemeral key
Hardware Security Module (HSM)
hash
hashing
high resiliency
key exchange
key length
lightweight cryptography
longevity
low latency
low-power devices
nonrepudiation
obfuscation
Opal
perfect forward secrecy
post-quantum cryptography
quantum communication
quantum computer
resource vs. security constraint
self-encrypting drives (SEDs)
steganography
stream cipher
symmetric cryptographic algorithm
Trusted Platform Module (TPM)
weak key

Review Questions

1. Which of the following hides the existence of information?
   a. Encryption
   b. Decryption
   c. Steganography
   d. Ciphering

2. Cryptography can prevent an individual from fraudulently reneging on an action. What is this known as?
   a. Repudiation
   b. Nonrepudiation
   c. Obfuscation
   d. Integrity

3. Brielle is researching substitution ciphers. She came across a cipher in which the entire alphabet was rotated 13 steps. What type of cipher is this?
   a. XOR
   b. XAND13
   c. ROT13
   d. Alphabetic

4. Which of the following is FALSE about "security through obscurity"?
   a. It attempts to hide its existence from outsiders.
   b. It can only provide limited security.
   c. It is essentially impossible.
   d. Proprietary cryptographic algorithms are an example.

5. What is low latency?
   a. A low-power source requirement of a sensor.
   b. The time between when a byte is input into a cryptographic cipher and when the output is obtained.
   c. The requirements for an IoT device that is using a specific network.
   d. The delay between when a substitution cipher decrypts the first block and when it finishes with the last block.

6. What are public key systems that generate different random public keys for each session?
   a. Public Key Exchange (PKE)
   b. perfect forward secrecy
   c. Elliptic Curve Diffie-Hellman (ECDH)
   d. Diffie-Hellman (DH)

7. What is data called that is to be encrypted by inputting it into a cryptographic algorithm?
   a. Plaintext
   b. Byte-text
   c. Cleartext
   d. Ciphertext

8. Which of these is NOT a basic security protection for information that cryptography can provide?
   a. Integrity
   b. Authenticity
   c. Risk
   d. Confidentiality

9. Cicero is researching hash algorithms. Which algorithm would produce the longest and most secure digest?
   a. SHA-256
   b. MD5
   c. SHA3-512
   d. SHA6-6

10. Which of the following is NOT a symmetric cryptographic algorithm?
    a. DES
    b. SHA
    c. Blowfish
    d. 3DES

11. Which of the following is not to be decrypted but is only used for comparison purposes?
    a. Digest
    b. Key
    c. Stream
    d. Algorithm

12. Which of these is NOT a characteristic of a secure hash algorithm?
    a. The results of a hash function should not be reversed.
    b. Collisions should occur no more than 15 percent of the time.
    c. A message cannot be produced from a predefined hash.
    d. The hash should always be the same fixed size.

13. Deo has been asked to explain RSA to his colleague. After his explanation, Deo is asked what, if any, weaknesses RSA has. How would Deo respond?
    a. RSA has no known weaknesses.
    b. As computers become more powerful, the ability to compute factoring has increased.
    c. RSA weaknesses are based on ECC.
    d. The digest produced by the RSA algorithm is too short to be secure.

14. Which of these is the strongest symmetric cryptographic algorithm?
    a. Data Encryption Standard
    b. Advanced Encryption Standard
    c. Triple Data Encryption Standard
    d. RC 1

15. If Bob wants to send a secure message to Alice using an asymmetric cryptographic algorithm, which key does he use to encrypt the message?
    a. Alice's private key
    b. Alice's public key
    c. Bob's public key
    d. Bob's private key

16. Egor wanted to use a digital signature. Which of the following benefits will the digital signature NOT provide?
    a. Verify the sender
    b. Verify the receiver
    c. Prove the integrity of the message
    d. Enforce nonrepudiation

17. Basil was reading about a new attack that forces the system to abandon a higher cryptographic security mode of operation and instead fall back to an older and less secure mode. What type of attack is this?
    a. Deprecation attack
    b. Pullback attack
    c. Downgrade attack
    d. Obfuscation attack

18. What is a collision?
    a. Two files produce the same digest.
    b. Two ciphertexts have the same length.
    c. Two algorithms have the same key.
    d. Two keys are the same length.

19. Which of the following is NOT a characteristic of the Trusted Platform Module (TPM)?
    a. It provides cryptographic services in hardware instead of software.
    b. It can generate asymmetric cryptographic public and private keys.
    c. It can easily be transported to another computer.
    d. It includes a pseudorandom number generator (PRNG).

20. Which of these provides cryptographic services and is external to the device?
    a. Trusted Platform Module (TPM)
    b. Hardware Security Module (HSM)
    c. Self-encrypting hard disk drives (SED)
    d. Encrypted hardware-based USB devices

Hands-On Projects

CAUTION

If you are concerned about installing any of the software in these projects on your regular computer, you can instead install the software in the Windows virtual machine created in the Module 1 Hands-On Projects. Software installed within the virtual machine will not impact the host computer.

Project 6-1: Using OpenPuff Steganography

Time Required: 25 minutes
Objective: Summarize the basics of cryptographic concepts.
Description: Unlike cryptography that scrambles a message so that it cannot be viewed, steganography hides the existence of the data. In this project, you will use OpenPuff to create a hidden message.

1. Use your web browser to go to **embeddedsw.net/OpenPuff_Steganography_Home.html**.

NOTE 21

It is not unusual for websites to change the location of where files are stored. If the URL no longer functions, open a search engine and search for "OpenPuff."

2. Click **Manual** to open the OpenPuff manual. Save this file to your computer. Read through the manual to see the features available. Return to the home page when finished.
3. Click **Download binary for Windows/Linux** to download the program. When a page appears asking for payments, click the **.ZIP** link to download the program for evaluation without submitting a payment.
4. Click **Screenshot** to view a screen capture of OpenPuff. Right-click this image and save it as **OpenPuff_Screenshot.jpg** on your computer. This will be the carrier file that will contain the secret message.

NOTE 22

For added security, OpenPuff allows a message to be spread across several carrier files.

5. Navigate to the location of the download and uncompress the OpenPuff zip file on your computer.
6. Now create the secret message to be hidden. Open Notepad and enter **This is a secret message**.
7. Save this file as **Message.txt** and close Notepad.
8. Create a zip file from the **Message** file. Navigate to the location of this file through File Explorer and right-click it.
9. Click **Send to** and select **Compressed (zipped) folder** to create the zip file.
10. Navigate to the OpenPuff directory and double-click **OpenPuff.exe**.
11. Click **Hide**.

NOTE 23

Under Bit selection options, note the wide variety of file types that can be used to hide a message.

12. Under **(1)**, create three unrelated passwords and enter them into **Cryptography (A)**, **(B)**, and **(C)**. Be sure that the **Scrambling (C)** password is long enough to turn the **Password check** bar from red to green.
13. Under **(2)**, locate the message to be hidden. Click **Browse** and navigate to the file **Message.zip**. Click **Open**.
14. Under **(3)**, select the carrier file. Click **Add** and navigate to **OpenPuff_Screenshot.jpg.**
15. Click **Hide Data!**.
16. Navigate to a different location from that of the carrier files and click **OK**.
17. After the processing has completed, navigate to the location of the carrier file that contains the message, and open the file. Can you detect anything different with the file now that it contains the message?

18. Now uncover the message. Close the OpenPuff Data Hiding screen to return to the main menu.
19. Click **Unhide!**.
20. Enter the three passwords.
21. Click **Add Carriers** and navigate to the location of **Carrier1** that contains the hidden message.
22. Click **Unhide!** and navigate to a location to deposit the hidden message. When it has finished processing, click **OK**.
23. Click **Done** after reading the report.
24. Go to that location and you will see **Message.zip**.
25. Close OpenPuff and close all windows.

Project 6-2: Running an RSA Cipher Demonstration

Time Required: 20 minutes
Objective: Summarize the basics of cryptographic concepts.
Description: The steps for encryption using RSA can be illustrated in a Java applet on a website. In this project, you will observe how RSA encrypts and decrypts.

NOTE 24

It is recommended that you review the section earlier in this module regarding the steps in the RSA function.

1. Use your web browser to go to **people.cs.pitt.edu/~kirk/cs1501/notes/rsademo/**.

NOTE 25

It is not unusual for websites to change the location of where files are stored. If the URL no longer functions, open a search engine and search for "RSA Cipher Demonstration."

2. Read the information about the demonstration.
3. Click **key generation page**.
4. Change the first prime number (P) to **7**.
5. Change the second prime number (Q) to **5**.
6. Click **Proceed**.
7. Read the information in the popup screen and record the necessary numbers. Close the screen when finished.
8. Click **Encryption Page**.
9. Next to **Enter Alice's Exponent key, E:** enter **5** as the key value from the previous screen.
10. Next to **Enter Alice's N Value:** enter **35**.
11. Click **Encrypt**. Read the message and record the values. Close the screen when finished.
12. Click **Decryption Page**.
13. Next to **Enter the encrypted message** enter **1**.
14. Next to **Enter your N value:** enter **35**.
15. Next to **Enter your private key, D:** enter **5**.
16. Click **Proceed**. Note that **1** has been decrypted to **A**.
17. Close all windows.

Project 6-3: Installing GUI Hash Generator and Comparing Digests

Time Required: 20 minutes
Objective: Summarize the basics of cryptographic concepts.
Description: In this project, you will download a GUI hash generator and compare the results of various hash algorithms.

1. Create a Microsoft Word document with the contents **Now is the time for all good men to come to the aid of their country.**
2. Save the document as **Country1.docx** on the desktop or in a directory specified by your instructor.
3. Now make a single change to **Country1.docx** by removing the period at the end of the sentence so it says **Now is the time for all good men to come to the aid of their country** and then save the document as **Country2.docx** in the same directory.

4. Close the document and Microsoft Word.
5. Use your web browser to go to **hashcalc.soft112.com**.

NOTE 26

It is not unusual for websites to change the location of where files are stored. If the URL no longer functions, open a search engine and search for "HashCalc."

6. Scroll down and then click **Download HashCalc**.
7. Click **Download**.
8. Click **Download Link**.
9. Follow the default instructions to install HashCalc.
10. Launch HashCalc to display the HashCalc window.
11. In addition to the hash algorithms selected by default, check the box next to the following hash algorithms to add them: **MD5**, **SHA256**, **SHA384**, **SHA512**, and **MD2**.
12. Click the file explore button next to **Data:**.
13. Navigate to the document **Country1.docx**.
14. Click **Open**.
15. In the HashCalc window, click **Calculate**.
16. Review the digests generated. If necessary, expand the size of the window. What can you say about these digests? Compare MD2 with SHA512. What makes SHA512 better than MD2? Why?
17. Click the file explore button next to **Data:**.
18. Navigate to the document **Country2.docx**.
19. Click **Open**.
20. In the HashCalc window, click **Calculate**.
21. This file is the same as the previous except a single period was removed. Are the digests different? What does this tell you about hashing digests?
22. Close all windows.

Project 6-4: Using Microsoft's Encrypting File System (EFS)

Time Required: 20 minutes
Objective: Given a scenario, implement host or application security solutions.
Description: Microsoft's Encrypting File System (EFS) is a cryptography system for Windows releases that use the Windows NT file system (NTFS). Because EFS is tightly integrated with the file system, file encryption and decryption are transparent to the user. In this project, you will turn on and use EFS.

1. Create a Word document with the contents of the first two paragraphs under **Today's Attacks and Defenses** on the first page of this module.
2. Save the document as **Encrypted.docx**.
3. Save the document again as **Not Encrypted.docx**.
4. Right-click the **Start** button, and then click **File Explorer**.
5. Navigate to the location of **Encrypted.docx**.
6. Right-click **Encrypted.docx**.
7. Click **Properties**.
8. Click the **Advanced** button.
9. Check the box **Encrypt contents to secure data**. This document is now protected with EFS. All actions regarding encrypting and decrypting the file are transparent to the user and should not noticeably affect any computer operations. Click **OK**.
10. Click **OK** to close the Encrypted Properties dialog box.
11. Launch Microsoft Word and then open **Encrypted.docx**. Was there any delay in the operation?
12. Now open **Not Encrypted.docx**. Was it any faster or slower?
13. Retain these two documents for use in the next project. Close Word.

Project 6-5: Using BestCrypt

Time Required: 25 minutes
Objective: Given a scenario, implement host or application security solutions.
Description: Third-party software applications can be downloaded to protect files with cryptography. In this project, you will download and install Jetico BestCrypt.

1. Use your web browser to go to **www.jetico.com**.

NOTE 27

It is not unusual for websites to change the location of where files are stored. If the URL no longer functions, open a search engine and search for "Jetico BestCrypt."

2. Under **ENCRYPT FILES**, click **LEARN MORE**.
3. Click **FREE TRIAL**.
4. Click **DOWNLOAD**.
5. Click **STANDARD EDITION**.
6. Follow the default installation procedures to install BestCrypt. A computer restart may be necessary.

NOTE 28

Note that this is a limited-time evaluation copy. Any files that are encrypted will only be available as read-only after the time limit expires.

7. Launch BestCrypt to display the BestCrypt control panel.
8. Files to be automatically encrypted are placed in a BestCrypt container. To create a container in the left pane, right-click the drive in which you want the container to be created, and then click **Container** and **New**.
9. Note the default file path for this container. Click **Show Advanced Settings**.
10. In the **Security Options** tab, click the arrow next to **Algorithm:** to display the cryptographic algorithms. Change to **Blowfish-448**.
11. Click **Create**.
12. The **Enter password** dialog box opens. Enter a strong password and confirm it. Click **OK**.
13. The **Seed value generation** window opens. Carefully read the instructions. What is the purpose of this window? Follow the instructions by pressing random keys or moving your cursor.
14. The **Format Local Disk** dialog box opens. This is to format the virtual drive that will contain your files. Click **Start** and then **OK**. When completed, click **Close**.
15. Note that you now have a new drive letter added to your computer, which is where you will place the files you want to encrypt. This container is entirely encrypted, including file names and free space, and functions like a real disk. You can copy, save, or move files to this container disk, and they will be encrypted as they are being written.
16. Right-click **Start** and then click **File Explorer**.
17. Click the drive letter of the drive that BestCrypt created.
18. Now drag a file into this drive (BestCrypt container). The file is automatically encrypted.
19. Open the document from your BestCrypt container. Did it take any longer to open now that it is encrypted? Close the document again.
20. Maximize the BestCrypt window and then click **Container** and **Dismount** to stop your container. A container will also be unmounted when you log off.
21. Based on your experiences with BestCrypt and EFS, which do you prefer? Why? What advantages and disadvantages do you see for both applications?
22. Close all windows.

Project 6-6: Blockchain Tutorial

Time Required: 25 minutes
Objective: Summarize the basics of cryptographic concepts.
Description: Understanding how blockchain functions can best be accomplished by performing a hands-on tutorial. In this project, you will use an online tutorial to learn about blockchain.

1. Use your web browser to go to **blockchain.mit.edu/how-blockchain-works**.

NOTE 29

It is not unusual for websites to change the location of where files are stored. If the URL no longer functions, open a search engine and search for "Andersbrown How Blockchain Works."

2. Click **Blockchain 101 – A Visual Demo** and watch the video.
3. When the video has completed click **2. Hash**.
4. In the **Data:** box, enter **This is data set 1** and note how the hash changes as you enter each letter.
5. Now change the **1** to **2**. What happens to the hash?
6. Click **3. Block**.
7. What new fields have been added?
8. Click **4. Blockchain**. Scroll to the right to see all the blocks in the chain. Look at the **Prev:** for **Block #5**. Compare that with the **Hash:** of **Block #4**. Are they identical? Why?
9. Compare the **Prev:** and **Hash:** of each block with the former block.
10. Return to **Block #5**. Enter **This is data set 5**. What happens to the color of the block? Why is this block now nvalid?
11. Go to **Block #4**. Enter **This is data set 4**. What happens to the color of Blocks #4 and #5? Why?
12. Return to **Block #5**. Click **Mine** to correct the information in the block.
13. Return to **Block #3**. Enter **This is data set 3**. What happens to Blocks #3, #4, and #5? Why? How does this illustrate that the blockchain resists change?
14. In **Block #3** click **Mine**. What happens to the color of this block?
15. Go to **Block #4** and click **Mine**.
16. Go to **Block #5** and click **Mine**.
17. If you were to make a change to Block #5, on which block must you click Mine to correct it? If you were to make a change to Block #3, on which blocks must you click Mine? Why the difference?
18. Close all windows.

Case Projects

Case Project 6-1: Broken SHA-1

Since 2004, security researchers theorized that SHA-1 would be vulnerable to a collision attack in which the same digest from two different plaintexts could be created. In early 2017, security researchers decisively demonstrated that SHA-1 could create a collision from two separate documents. However, this attack was limited, and it was estimated it would cost attackers from $110,000 to $560,000 on Amazon's Web Services (AWS) to carry it out. In early 2020, researchers unveiled a new attack on SHA-1 that was even more powerful. The new collision attack gives attackers more options and flexibility to produce the same digest for two or more data sets simply by appending data to each of sets. This attack cost as little as $45,000 to carry out. This compromise of SHA-1 has rendered it no longer suitable for use. How did the researchers do it? Visit the website Shattered (**shattered.io**) to find information about how SHA-1 was breached in 2017. Read the Q&A section and view the infographic. Try dragging one of your files to the File Tester to see if it is part of the collision attack. What did you learn? How serious is a collision? What is the impact? Now do conduct research on the 2020 SHA-1 attack. How was it carried out? Write a one-page paper of what you learned.

Case Project 6-2: Compare Cipher Tools

A variety of online cipher tools demonstrate different cryptographic algorithms. Visit the website Cipher Tools (**http://rumkin.com/tools/cipher/**) and explore the different tools. Select three tools, one of which is mentioned in this module (ROT13, one-time pad, etc.). Experiment with the three tools. Which is easy to use? Which is more difficult? Which tool would you justify as more secure than the others? Why? Write a one-page paper on your analysis of the tools.

Case Project 6-3: Lightweight Cryptography

Due to the importance of incorporating cryptography in low-power devices, a new "subfield" of cryptography called lightweight cryptography is being developed. This has the goal of providing cryptographic solutions that are uniquely tailored for low-power devices that need to manage resource vs. security constraints. Research lightweight cryptography. What are its goals? How will it work? What are its limitations? What are its advantages? Write a one-page paper on your findings.

Case Project 6-4: Twofish and Blowfish

Twofish and Blowfish are considered strong symmetric cryptographic algorithms. For example, Blowfish can accommodate key lengths of up to 448 bits (56 bytes). Use the Internet to research both Twofish and Blowfish. How secure are they? What are their features? What are their strengths and weaknesses? How are they currently being used? How would you compare them? Write a one-page paper on your findings.

Case Project 6-5: SHA-3

The hash algorithms SHA-1 and SHA-2 were not created by publicly sourced contests but instead were created by the National Security Agency (NSA) and then released as public-use patents. Although they are not identical, they share some of the same underlying mathematics, which has been proven to contain some cryptographic flaws. SHA-2 is a safer hash largely because of its increased digest length. SHA-3 is a completely different type of hash algorithm. Research SHA-3. What were its design goals? How is it different from SHA-1 and SHA-2? What are its advantages? How does its performance in hardware and software compare? When will it be widely implemented? What a one-page paper on your research.

Case Project 6-6: One-Time Pad (OTP) Research

Use the Internet to research OTPs: who was behind the initial idea, when they were first used, in what applications they were found, how they are used today, and other relevant information. Then visit an online OTP creation site such as **www.braingle.com/brainteasers/codes/onetimepad.php** and practice creating your own ciphertext with OTP. If possible, exchange your OTPs with other students to see how you might try to break them. Would it be practical to use OTPs? Why or why not? Write a one-page paper on your findings.

Case Project 6-7: Blockchain

Use the Internet to research how blockchain is currently being used. What are its advantages? What are its disadvantages? How widespread is its acceptance? What are future applications of blockchain? Write a one-page paper on your research.

Case Project 6-8: Information Security Community Site Activity

The Information Security Community Site is an online companion to this textbook. It contains a wide variety of tools, information, discussion boards, and other features to assist learners. Go to **community.cengage.com/infosec2** and click the *Join or Sign in* icon to login, using your login name and password that you created in Module 1. Click **Forums (Discussion)** and click on **Security+ Case Projects (7th edition)**. Read the following case study.

This is a true story (with minor details changed). Microsoft had uncovered several licensing discrepancies in its software. Clients were using the software while claiming they had purchased it from an authorized software retailer. The sale of one software package to a company in Tampa was traced back to a retailer in Pennsylvania, and yet the retailer had no record of any sales to the Tampa company. A private security consulting agency was called in, and they discovered that the network system administrator "Ed" in Pennsylvania was downloading pirated software from the Internet and selling it to customers as legitimate software behind the company's back. Ed had sold almost a half-million dollars in illegal software. The security firm

also noticed a high network bandwidth usage. Upon further investigation, they found that Ed was using one of the company's servers as a pornographic website with more than 50,000 images and 2,500 videos. In addition, a search of Ed's desktop computer uncovered a spreadsheet with hundreds of credit card numbers from the company's e-commerce site. The security firm speculated that Ed was either selling the card numbers to attackers or using them himself.

The situation was complicated by the fact that Ed was the only person who knew certain administrative passwords for the core network router and firewall, network switches, the corporate virtual private network (VPN), the entire human resources system, the email server, and the Windows Active Directory. In addition, the company had recently installed a Hardware Security Module (HSM) to which only Ed had the password. The security consultant and the Pennsylvania company were worried about what Ed might do if he were confronted with the evidence, since he could hold the entire organization hostage or destroy every piece of useful information.

A plan was devised. The company invented a fictitious emergency at one of their offices in California that required Ed to fly there overnight. The long flight gave the security team a window of about five and a half hours during which Ed could not access the system (the flight that was booked for Ed did not have wireless access). Working as fast as they could, the team mapped out the network and reset all the passwords. When Ed landed in California, the chief operating officer was there to meet him and fire Ed on the spot.

Now it's your turn to think outside of the box. What would you have done to keep Ed away so you could reconfigure the network? Or how could you have tricked Ed into giving up the passwords without revealing to him that he was under suspicion? Record your answers on the Community Site discussion board.

Case Project 6-9: North Ridge Security

North Ridge Security provides security consulting and assurance services. You have recently been hired as an intern to assist them.

A new North Ridge client wants to provide encryption for any data that leaves their premises. You are asked to provide an overview of the different ways in which encryption can be used.

1. Create a PowerPoint presentation about encryption through software (third-party software and OS), FDE, SED, HSM, and TPM. Include the advantages and disadvantages of each. Your presentation should contain at least 10 slides.
2. After the presentation, the client asks for your recommendation regarding meeting their needs for encryption when taking data off-site. Create a memo communicating the actions you believe would be best for the company to take.

References

1. Russell, Frank. *Information Gathering in Classical Greece*. Ann Arbor: University of Michigan Press, 1999.
2. Lenstra, Arjen; Kleinjung, Thorsten; Thome, Emmanuel. *Universal Security from Bits and Mips to Pools, Lakes—And Beyond.* Accessed Jun. 3, 2020. https://eprint.iacr.org/2013/635.pdf.

MODULE 7

PUBLIC KEY INFRASTRUCTURE AND CRYPTOGRAPHIC PROTOCOLS

After completing this module, you should be able to do the following:

1. Define digital certificates
2. Describe the components of Public Key Infrastructure (PKI)
3. Describe the different cryptographic protocols
4. Explain how to implement cryptography

Front-Page Cybersecurity

With today's super-fast computers and the advancements in cryptography, it would seem that an encrypted message dating back almost 80 years could easily be broken. However, that proved not to be the case in this fascinating incident.

In 1982, David and Anne Martin were renovating a fireplace that had been sealed off for many years in their 17th-century house in the village of Bletchingley, England. In the chimney, the Martins discovered the remains of a carrier pigeon with a small scarlet capsule attached to its leg. The red color of the capsule marked the bird as a military carrier pigeon for the Allied Forces in World War II. Inside the capsule was a message written in code containing 27 groups of five letters or numbers on thin paper the size of a cigarette paper. The message read

AOAKN HVPKD FNFJW YIDDC RQXSR DJHFP GOVFN MIAPX PABUZ WYYNP CMPNW HJRZH NLXKG MEMKK ONOIB AKEEQ WAOTA RBQRH DJOFM TPZEH LKXGH RGGHT JRZCQ FNKTQ KLDTS FQIRW AOAKN 27 1525/6

At the bottom of the coded message were two items that were not in code: "Number of Copies Sent: Two" and "Sender: Serjeant [*sic*] W. Stot." Additional sets of numbers (NURP 40 TW194 and NURP 37 DK 76) probably indicated the military number of the two birds who carried the message.

The Martins contacted several British government authorities about their find but initially there was no interest in the bird's message. However, in 2012, Bletchley Park, which served as the headquarters of British Intelligence codebreakers during World War II and is now a museum, took an interest in the message. It turns out the message may have been top secret. First, although Bletchley Park—only five miles from the Martin's house—used carrier pigeons during World War II, none of its official messages were sent in code; they were all written in longhand. Second, messages were never carried by more than one bird. Evidently the bird's message may have been part of a classified program.

In late 2012, the British Government Communications Headquarters, which is responsible for code breaking, examined the encrypted message. After top government codebreakers spent months using super-fast computers to attempt to break the code, they finally announced that the code could not be cracked. (A few amateur sleuths have claimed to have deciphered the message, but these claims have proved false.)

Why is it so tough to break this code? The code was written using a one-time pad, or OTP. (OTP is covered in Module 6.) As a key, an OTP uses a random set of letters that only the sender and recipient know. If an OTP is truly random, is used only one time, and is kept secret by the sender and receiver, it can be virtually impossible to crack. That seems to be the case in this incident. Thus, it is a matter of "key management": because the key to the code is lost, the message cannot be deciphered.

We may never know what message that pigeon 40TW194 was carrying. Yet, as a Government Communications Headquarters spokesperson said, "It is a tribute to the skills of the wartime code makers that, despite working under severe pressure, they devised a code that was undecipherable both then and now."[1]

Despite the clear benefits of cryptography in protecting data, few users have chosen to use it (other than what occurs by default for them). Yet for enterprises, the encrypting and decrypting of data is not only considered essential, but in many instances, it is required. However, when cryptography is used in the enterprise, a much higher degree of complexity is involved, particularly regarding keys. What happens if an employee has encrypted an important proposal but suddenly falls ill and cannot return to work? Does only the employee know the key? Or could a copy of the key be retrieved? Where is the copy of the key stored? Who can have access to it? How can the keys of hundreds or even thousands of employees be managed?

The issue of managing cryptography, particularly in the enterprise, is the topic of this module. First, you will learn about the authentication and distribution of public keys through digital certificates. Next, you will study the management of keys through public key infrastructure. Finally, you will look at cryptographic protocols to see the role of cryptography on data in transit/motion across a network data and how to implement cryptography.

DIGITAL CERTIFICATES

CERTIFICATION

3.2 Given a scenario, implement host or application security solutions.

3.9 Given a scenario, implement public key infrastructure.

One of the common applications of public key cryptography is digital certificates. Using digital certificates involves understanding their purpose, knowing how they are managed, and determining which type of digital certificate is appropriate for different situations.

Defining Digital Certificates

Asymmetric cryptography (also called public key cryptography) uses a pair of related keys. The public key can be distributed and shared with anyone, while the corresponding private key must be kept confidential by the owner. Asymmetric cryptography has two uses: it can encrypt or decrypt a set of data, and it can be used as a proof to verify a "signature" of the sender.

Suppose that Alice receives an encrypted document that claims it is from Bob. Because it is encrypted, Alice knows that the document was not viewed by someone else. But how can she know for certain that Bob was the sender? Because Alice's public key is widely available, anyone could use it to encrypt a document. An imposter of Bob could have created a fictitious document, encrypted it with Alice's public key, and then sent it to Alice pretending to be Bob. While Alice can use her private key to decrypt the document to read it, she does not know with absolute certainty who sent it.

A degree of proof can be provided with asymmetric cryptography by creating a *digital signature*, which is an electronic verification of the sender. After creating a document, Bob generates a hash digest on it and then

encrypts the digest with his private key, which serves as the digital signature. Bob sends both the encrypted document and the digital signature to Alice, who decrypts the digital signature using Bob's public key, revealing the digest. If she cannot decrypt the digital signature, then she knows that it did not come from Bob (because only Bob's public key can decrypt the digest generated with his private key).

NOTE 1

Digital signatures are covered in Module 6.

However, a digital signature has a weakness: it can only prove the *owner* of the private key and does not necessarily confirm the true identity of the sender. That is, a digital signature only shows that the private key of the sender was used to encrypt the digital signature, but it does not definitively prove *who* was the sender of that key. If Alice receives a message with a digital signature claiming to be from Bob, she cannot know for certain that it is the *real* Bob whose public key she is retrieving.

For example, suppose Bob created a message along with a digital signature and sent it to Alice. However, Mallory intercepted the message. Mallory then created her own set of public and private keys using Bob's identity. Mallory could then create a new message and digital signature (with the imposter private key) and send them to Alice. Upon receiving the message and digital signature, Alice would unknowingly retrieve the imposter public key (thinking it belonged to Bob) and decrypt it. Alice would be tricked into thinking Bob had sent it when, in reality, it came from Mallory. This interception and imposter public key are illustrated in Figure 7-1.

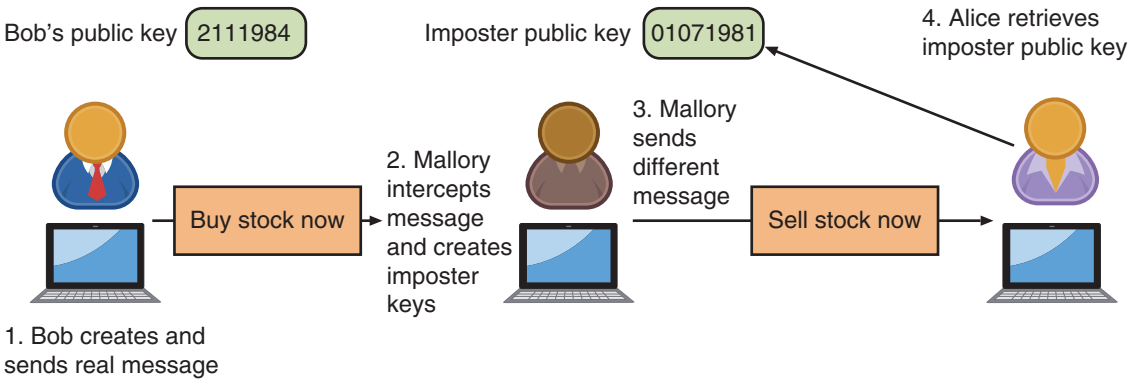


Figure 7-1 Imposter public key

Suppose that Bob wants to ensure that Alice receives his real public key and not an imposter public key. He could travel to Alice's city, knock on her front door, and say, "I'm Bob and here's my key."

Yet how would Alice even know *this* was the real Bob and not Mallory in disguise? For verification, she could ask to see Bob's passport. A passport is a document that is provided by a *trusted third party*. Although Alice may not initially trust Bob because she does not know him, she will trust the government agency that required Bob to provide proof of his identity when he applied for the passport. Using a trusted third party who has verified Bob—and who Alice also trusts—would solve the problem.

This scenario illustrates the concept behind a digital certificate. A **digital certificate** is a technology used to associate a user's identity to a public key and that has been digitally signed by a trusted third party. The third party verifies the owner and that the public key belongs to that owner.

A digital certificate is basically a container for a public key. It can be used to identify objects other than users, such as servers and applications. Typically, a digital certificate contains information such as the owner's name or alias, the owner's public key, the name of the issuer, the digital signature of the issuer, the serial number of the digital certificate, and the expiration date of the public key. It can contain other user-supplied information, such as an email address, postal address, and basic registration information.

When Bob sends a message to Alice, he does not ask her to retrieve his public key from a central site. Instead, Bob attaches the digital certificate to the message. When Alice receives the message with the digital certificate, she can check the signature of the trusted third party on the certificate. If the signature was signed by a party that she trusts, then Alice can safely assume that the public key—contained in the digital certificate—is actually from Bob. Digital certificates make it possible for Alice to verify Bob's claim that the key belongs to him and prevent an attack that impersonates the owner of the public key.

Managing Digital Certificates

Several entities and technologies manage digital certificates, including the certificate authorities and tools for managing certificates.

Certificate Authorities

Alice purchases a new car and visits the local county courthouse to fill out the car title application paperwork to register her car. After signing the application and verifying her identity, the information is forwarded to the state capital, where the state's department of motor vehicles (DMV) issues an official car title that is sent to her as the new owner.

This scenario illustrates some of the entities involved with digital certificates. A user who wants a digital certificate must generate the public and private keys to use and then complete a request with information such as name, address, and email address, known as a **Certificate Signing Request (CSR)**. The user electronically signs the CSR by affixing a public key and then sends it to a **registration authority** responsible for verifying the credentials of the applicant. Once verified, the CSR is transferred to an **intermediate certificate authority (CA)**. The intermediate CA, of which there are many, processes the CSR and issues the digital certificates. The intermediate CAs perform functions on behalf of a **certificate authority (CA)** that is responsible for digital certificates. A comparison between the earlier car title scenario and the elements of a digital certificate are shown in Table 7-1.

NOTE 2

Just as a state has many county courthouses, a CA has many intermediate CAs.

Table 7-1 Digital certificate elements

Car title scenario	Digital certificate element	Explanation
Car title application	Certificate Signing Request (CSR)	Formal request for digital certificate
Sign car title application	Create and affix public key to certificate	Added to digital certificate for security
Visit county courthouse	Intermediate certificate authority	Party that can process CSR on behalf of CA
Title sent from state DMV	Certificate authority (CA)	Party responsible for digital certificates

Intermediate CAs are subordinate entities designed to handle specific CA tasks such as processing certificate requests and verifying the identity of the individual. Depending on the type of digital certificate, the person requesting a digital certificate can be authenticated by:

- *Email.* In the simplest form, the owner might be identified only by an email address. Although this type of digital certificate might be sufficient for basic email communication, it is insufficient for most other activities.
- *Documents.* A registration authority can confirm the authenticity of the person requesting the digital certificate by requiring specific documentation such as a birth certificate or a copy of an employee badge that contains a photograph.
- *In person*. In some instances, the registration authority might require the applicant to apply in person to prove his existence and identity by providing a government-issued passport or driver's license.

NOTE 3

Although the registration function could be implemented directly with the CA, using separate intermediate CAs offers advantages. If many entities require a digital certificate, or if these are spread out across geographical areas, using a single centralized CA could create bottlenecks or inconveniences. Using multiple intermediate CAs, who can "off-load" the registration functions, can create an improved workflow. The process works because the CAs trust the intermediate CAs.

Just as a breach at a state DMV could result in many fraudulent car titles being distributed, so too could the consequences of a compromised CA taint its intermediate CAs and the digital certificates that they issued. CAs must therefore be kept safe from unauthorized access. A common method to ensure the security and integrity of a CA is to keep it in an offline state from the network (**offline CA**). It is only brought online (**online CA**) when needed for specific and infrequent tasks, typically limited to the issuance or reissuance of certificates authorizing intermediate CAs.

Certificate Management

Multiple entities make up strong certificate management, including a certificate repository and a means for certificate revocation.

Certificate Repository (CR) A *certificate repository (CR)* is a publicly accessible centralized directory of digital certificates that can be used to view the status of a digital certificate. The directory can be managed locally by setting it up as a storage area that is connected to the CA server.

Certificate Revocation Digital certificates normally have an expiration date. However, in some circumstances, the certificates are revoked before they expire. Some reasons might be benign, such as when a certificate is no longer used or the details of the certificate—such as the user's address—have changed. Other circumstances could be more dangerous. For example, if attackers steal a user's private key, they could impersonate the victim by using digital certificates without other users being aware of the impersonation. In addition, what would happen if digital certificates were stolen from a CA? The thieves could issue certificates to themselves that would be trusted by unsuspecting users. The CA must publish lists of approved certificates as well as revoked certificates in a timely fashion so that security is not compromised.

CAUTION

Digital certificates have been stolen from CAs or intermediate CAs. With a stolen digital certificate, thieves can trick unsuspecting users into connecting with an imposter site, thinking it is a legitimate site. State actors have also been charged with stealing digital certificates to trick their own citizens into connecting with a fraudulent email site to monitor their messages and to locate and crackdown on dissidents.

The status of a certificate can be checked in two ways to see if it has been revoked. The first is to use a **Certificate Revocation List (CRL),** which is a list of certificate serial numbers that have been revoked. Many CAs maintain an online CRL that can be queried by entering the certificate's serial number. In addition, a local computer receives updates on the status of certificates and maintains a local CRL, as illustrated in Figure 7-2.

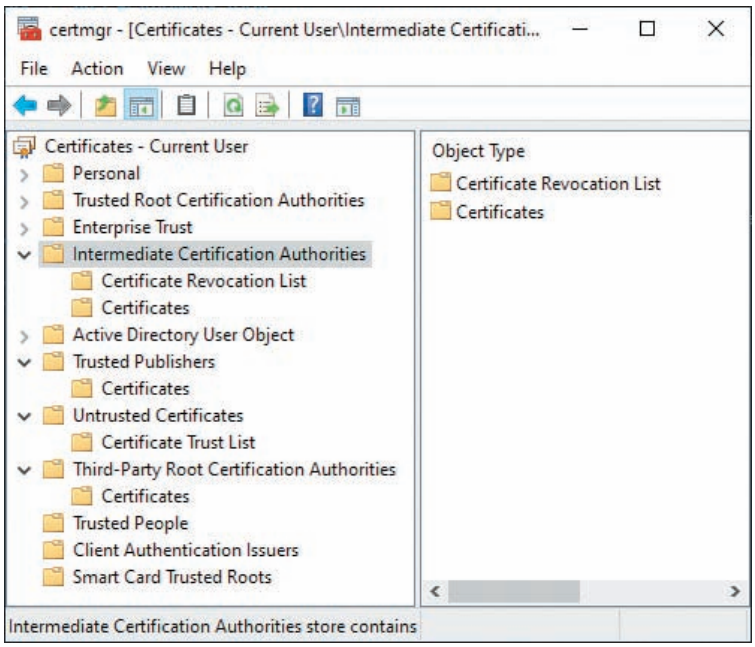


Figure 7-2 Certificate Revocation List (CRL)

The second method is **Online Certificate Status Protocol (OCSP)**, which performs a real-time lookup of a certificate's status. OCSP is called a request-response protocol. The browser sends the certificate's information to a trusted entity like the CA, known as an *OCSP Responder*. The OCSP Responder then provides revocation information on that one specific certificate.

A variation of OCSP is called OCSP **stapling**. OCSP requires the OCSP Responder to provide responses to every web client of a certificate in real time, which may create a high volume of traffic. With OCSP stapling, web servers send queries to the Responder OCSP server at regular intervals to receive a signed time-stamped OCSP response. When a client's web browser attempts to connect to the web server, the server can include (*staple*) in the handshake with the web browser the previously received OCSP response. The browser then can evaluate the OCSP response to determine if it is trustworthy. OCSP stapling is illustrated in Figure 7-3.

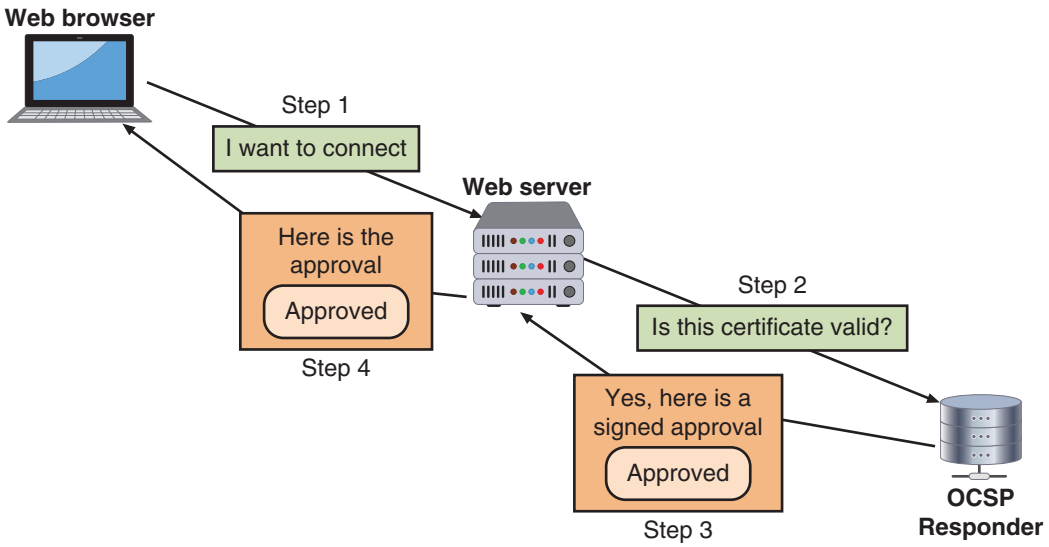


Figure 7-3 OCSP stapling

Determining the revocation status of certificates presented by websites is an ongoing problem in web security. Initially, modern web browsers (Chrome, Firefox, Internet Explorer, Safari, and Opera) used OCSP. However, if the web browser cannot reach the OCSP Responder server, such as when the server is down, then the browser receives a network error message (called a *soft fail*), and the revocation check is simply ignored. Also, online revocation checking by web browsers can be slow. For these reasons, web browsers have implemented a range of solutions to reduce or eliminate the need for online revocation checking by instead "harvesting" lists of revoked certificates from CAs and then pushing them to the user's browser. Table 7-2 lists the solutions used by selected web browsers for determining the revocation status of certificates.

Table 7-2 Web browser certificate revocation procedures

Browser	Procedure name	Description	Resources
Google Chrome	CRLSet	CRLSet is a list of revoked certificates determined by searching CRLs published by CAs that is pushed to the browser as a software update.	Users can view the CRLSet version and manually check for updates at chrome://components.
Mozilla Firefox	OneCRL	OneCRL is a list of intermediate certificates that have been revoked by CAs and then pushed to Firefox in application updates; the browser can also be configured to query OCSP responders.	Lists of OneCRL can be viewed as webpages at https://crt.sh/mozilla-onecrl.
Apple Safari	No name	Collects revoked certificates from CAs that are then periodically retrieved by Apple devices; when revoked certificate request occurs, Safari performs an OCSP check to confirm.	View blocked certificates at https://support.apple.com/en-us/HT209144#blocked.
Microsoft Edge	CRLSet	Windows OS checks for server certificate revocation and Edge relies on this listing.	Can be viewed through Windows OS.

Types of Digital Certificates

Digital certificates can be grouped into the broad categories of root certificates, domain certificates, and hardware and software certificates. In addition there are standardized certificate formats and attributes.

Root Digital Certificates

Suppose that Alice is making a purchase at an online ecommerce site and needs to enter her credit card number. How can she be certain that she is at the authentic website of the retailer and not an imposter's look-alike site that will steal her credit card number? The solution is a digital certificate. The online retailer's web server issues to Alice's web browser a digital certificate that has been signed by a trusted third-party. In this way, Alice can rest assured that she is at the authentic online retailer's site.

However, an estimated 24 million ecommerce sites are in operation.[2] In addition, consider the number of websites for banks, credit card companies, schools, and workplaces, to name a few of the other websites that need to provide protection to its customers and clients. How can each digital certificate be verified as being authentic and not expired or revoked, and how should they be organized?

NOTE 4

In 2016, the nonprofit Internet Security Research Group (ISRG) created *Let's Encrypt*, which is advertised as a free, automated, and open CA run for the public's benefit. Their goal is to provide free digital certificates to any website with the stated reason "because we want to create a more secure and privacy-respecting Web." Let's Encrypt issues up to one million digital certificates daily, and by early 2020, it had issued more than one billion digital certificates.

Grouping and verifying digital certificates relies on **certificate chaining**. Certificate chaining creates a path between the trusted root CAs (of which there are a few) and intermediate CAs (of which there are many) with the digital certificates that have been issued. Every certificate is signed by the entity that is identified by the next-higher certified entity in the chain. In this way, the trust of a certificate can be traced back to the highest level of CA.

The beginning point of the chain is a specific type of digital certificate known as a **root digital certificate**. A root digital certificate is created and verified by a CA. Because a CA has no higher-level authority, root digital certificates are **self-signed** and do not depend upon a higher-level authority for authentication. The next level down from the root digital certificate is one or more *intermediate certificates* that have been issued by intermediate CAs. The root digital certificate (verified by a CA) trusts the intermediate certificate (verified by an intermediate CA), which may in turn validate more lower-level intermediate CAs until it reaches the end of the chain. The endpoint of the chain is the **user digital certificate** itself. Certificate chaining is illustrated in Figure 7-4.

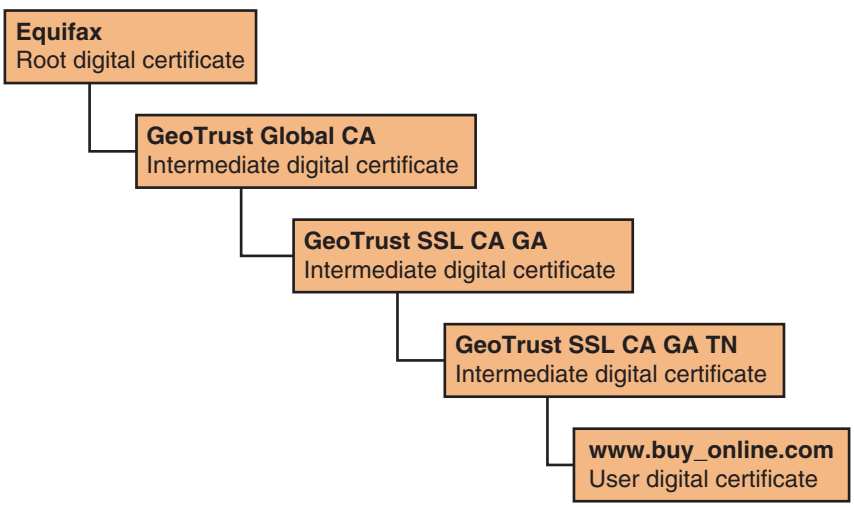


Figure 7-4 Certificate chaining

NOTE 5

How is the Queen of England like a root digital certificate? All British passports are issued in the name of Her Majesty. When the queen travels, she does not need her own passport because there is no higher authority to authorize one. In a similar fashion, a root digital certificate does not depend on any higher-level authority. However, all other members of the queen's royal family must still carry their own passport—even if they are traveling with her.

Approved root digital certificates and intermediate certificates are distributed in one of three ways. First, they can be distributed through updates to the OS. Second, they can be distributed through updates to the web browser. At one time, browsers relied on the underlying OS-approved list, but today they rely on their own browser updates. Web browser certificate chaining and root digital certificates can be seen in the browsers and are illustrated in Figure 7-5. A third option is **pinning**, in which a digital certificate is hard-coded (*pinned*) within the app (program) that is using the certificate. Pinning is common for securing mobile messaging apps and for certain web-based services and browsers.

NOTE 6

As of mid-2020, Microsoft Windows recognized 82 trusted root CAs while Apple recognized 115 for all of its OS versions and one "always ask" root CA, which is untrusted but not blocked.

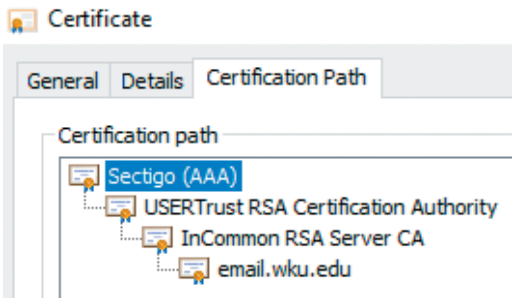


Figure 7-5 Certificate chaining in web browser

Domain Digital Certificates

Most digital certificates are web server digital certificates that are issued from a web server to an endpoint. Web server digital certificates perform two primary functions: they ensure the authenticity of the web server to the client and the authenticity of the cryptographic connection to the web server. Web servers can set up secure cryptographic connections by providing the server's public key with a digital certificate to the client. The handshake setup between web browser and web server, also called a *key exchange,* is illustrated in Figure 7-6:

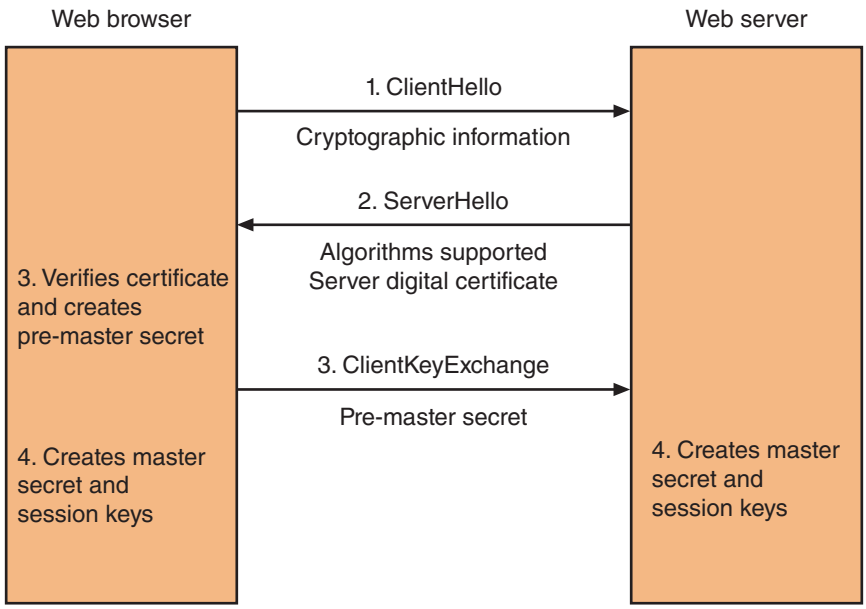


Figure 7-6 Key exchange

1. The web browser sends a message ("ClientHello") to the server that contains information including the list of cryptographic algorithms that the client supports.
2. The web server responds ("ServerHello") by indicating which cryptographic algorithm will be used. It then sends the server digital certificate to the browser.
3. The web browser verifies the server certificate (such as making sure it has not expired) and extracts the server's public key. The browser generates a random value (called the *pre-master secret*), encrypts it with the server's public key, and sends it back to the server ("ClientKeyExchange").
4. The server decrypts the message and obtains the browser's pre-master secret. Because both the browser and server now have the same pre-master secret, they can each create the same *master secret*. The master secret is used to create *session keys*, which are symmetric keys to encrypt and decrypt information exchanged during the session and to verify its integrity.

To address the security of web server digital certificates, there are several types of domain digital certificates. These include domain validation digital certificates, extended validation digital certificates, wildcard digital certificates, and subject alternative names digital certificates.

Domain Validation Some entry-level certificates provide domain-only validation to authenticate that only a specific organization has the right to use a particular domain name. A **domain validation digital certificate** verifies the identity of the entity that has control over the domain name. The certificates indicate nothing regarding the trustworthiness of the individuals behind the site; they simply verify who has control of that domain.

NOTE 7

Because domain validation digital certificates are not verifying the identity of a person but only the control over a site, they often can be generated automatically and are very inexpensive or even free.

Extended Validation (EV) An enhanced type of domain digital certificate is the **Extended Validation (EV) certificate**. This type of certificate requires more extensive verification of the legitimacy of the business. Requirements include the following:

- The intermediate CA must pass an independent audit verifying that it follows the EV standards.
- The existence and identity of the website owner, including its legal existence, physical address, and operational presence, must be verified by the intermediate CA.
- The intermediate CA must verify that the website is the registered holder and has exclusive control of the domain name.
- The authorization of the individual(s) applying for the certificate must be verified by the intermediate CA, and a valid signature from an officer of the company must accompany the application.

Wildcard A **wildcard digital certificate** is used to validate a main domain along with all subdomains. For example, a domain validation digital certificate for *www.example.com* would only cover that specific site. A wildcard digital certificate for **.example.com* would cover *www.example.com*, *mail.example.com*, *ftp.example.com*, and any other subdomains.

Subject Alternative Name (SAN) A **Subject Alternative Name (SAN)** digital certificate, also known as a *Unified Communications Certificate* (*UCC*), is primarily used for Microsoft Exchange servers or unified communications (the integration of different types of electronic communication such as email, SMS text messaging, and fax). The certificate allows multiple server or domain names to use the same secure certificate by associating different values with the certificate.

Hardware and Software Digital Certificates

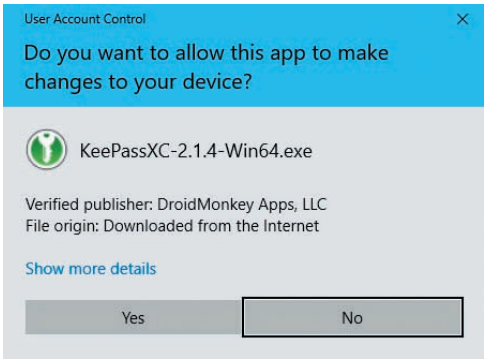
In addition to root digital certificates and domain digital certificates, more specific digital certificates relate to hardware and software. These include the following:

- *Machine/computer digital certificate*. A **machine/computer digital certificate** is used to verify the identity of a device in a network transaction. For example, a printer may use a machine digital certificate to verify to the endpoint that it is an authentic and authorized device on the network.

NOTE 8

Many network devices can create their own self-signed machine digital certificates.

- *Code signing digital certificate.* Digital certificates are used by software developers to digitally sign a program to prove that the software comes from the entity that signed it and no unauthorized third party has altered or compromised it. This is known as a **code signing digital certificate**. When the installation program is launched that contains a code signing digital certificate, a pop-up window appears that says *Verified publisher,* as shown in Figure 7-7. An installation program that lacks a code digital certificate will display a window with the warning *Publisher:Unknown*.



Source: KeePassXC Team

Figure 7-7 Verified publisher message

- *Email digital certificate*. An **email digital certificate** allows a user to digitally sign and encrypt mail messages. Typically, only the user's name and email address are required to receive the certificate.

NOTE 9

In addition to email messages, digital certificates also can be used to authenticate the authors of documents. For example, a user can create a Microsoft Word or Adobe Portable Document Format (PDF) document and then use a digital certificate to create a digital signature.

Digital Certificate Attributes and Formats

Hardware devices require that digital certificates contain specific attributes (fields) and are presented in a specific format. Doing so allows the device to read and process the digital certificate.

The standard format for digital certificates is *X.509*. The format was first introduced more than 20 years ago and was adapted for Internet use. The current version is Version 3. Digital certificates following this standard can be read or written by any hardware device or application that follows the X.509 format.

Several **certificate attributes** make up an X.509 digital certificate; these are used when parties negotiate a secure connection. Attributes that must be included are the certificate validity period, end-host identity information, encryption keys that will be used for secure communications, the signature of the issuing CA, and the **common name (CN)**. CN is the name of the device protected by the digital certificate. The CN can reference a single device (*www.example.com*) or multiple devices with a wildcard certificate (**.example.com*) but is not the URL (*https://example.com*). Other optional attributes may also be included. Figure 7-8 illustrates several attributes in a digital certificate.

X.509 certificates can either be contained in a binary file with a **.cer** extension or in a *Base64* file, which is a binary-to-text encoding scheme that presents binary data in ASCII string format. X.509 certificates have three encoding formats (layouts): *Basic Encoding Rules (BER)*, **Canonical Encoding Rules (CER)**, and **Distinguished Encoding Rules (DER)**. The X.509 certificates themselves can be contained within different file formats. Table 7-3 shows several of the formats.

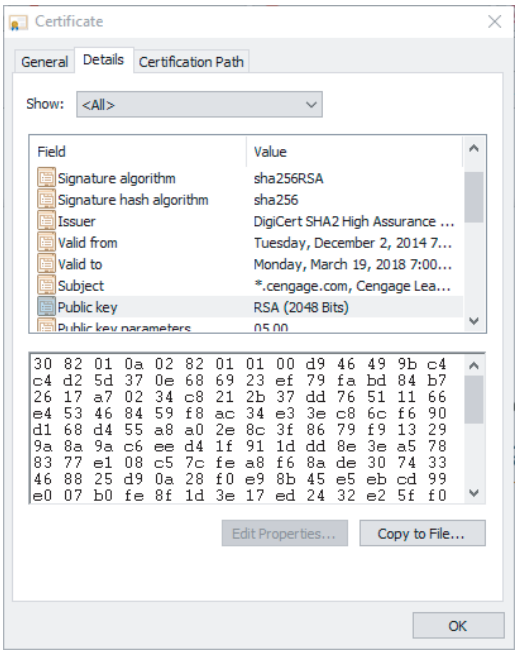


Figure 7-8 Digital certificate attributes

Table 7-3 X.509 file formats

Name	File extension	File type	Comments
Privacy Enhancement Mail (PEM)	.pem	Base64	Designed to provide confidentiality and integrity to emails, it uses DER encoding and can have multiple certificates.
Personal Information Exchange (PFX)	.pfx	Binary	The preferred file format for creating certificates to authenticate applications or websites, PFX is password protected because it contains both private and public keys.
PKCS#7	**.P7B**	Base64	Cryptographic Message Syntax Standard that defines a generic syntax for defining digital signature and encryption.
PKCS#12	**.P12**	Binary	Personal Information Exchange Syntax Standard defines the file format for storing and transporting a user's private keys with a public key certificate.

TWO RIGHTS & A WRONG

1. A digital certificate is a technology used to associate a user's identity to a public key and that has been digitally signed by the owner of the private key.
2. A certificate repository (CR) is a publicly accessible centralized directory of digital certificates that can be used to view the status of a digital certificate.
3. Root digital certificates are self-signed.

See Appendix B for the answer.

PUBLIC KEY INFRASTRUCTURE (PKI)

CERTIFICATION

3.9 Given a scenario, implement public key infrastructure.

One of the important management tools for the use of digital certificates and asymmetric cryptography is public key infrastructure. You should understand public key infrastructure, know the PKI trust models, how PKI is managed, and the features of key management.

What Is Public Key Infrastructure (PKI)?

One single digital certificate between Alice and Bob involves multiple entities and technologies. Asymmetric cryptography must be used to create the public and private keys, a registration authority must verify the CSR, an intermediate CA must process the CSR, the digital certificate must be placed in a CR and moved to a CRL when it expires, and so on. In an organization where multiple users have digital certificates (and sometimes multiple digital certificates), it can quickly become overwhelming for someone to manage all of these entities. In short, organizations need a consistent means to manage digital certificates.

Public key infrastructure (PKI) is what you might expect from its name: it is the underlying infrastructure for **key management** of public keys and digital certificates. PKI is a framework for the administration of all the elements involved in digital certificates for digital certificate management—including hardware, software, people, policies, and procedures—to create, store, distribute, and revoke digital certificates. In short, PKI is digital certificate management.

Trust Models

Trust is defined as confidence in or reliance on another person or entity. One of the principal foundations of PKI is that of trust: Alice must trust that the public key in Bob's digital certificate belongs to him.

A **trust model** refers to the type of trust relationship that can exist between individuals or entities. In one type of trust model, *direct trust*, a relationship exists between two individuals because one person knows the other person. Because Alice knows Bob—she has seen him, she can recognize him in a crowd, she has spoken with him—she can trust that the digital certificate that Bob personally gives her contains his public key. A *third-party trust* refers to a situation in which two individuals trust each other because each trusts a common third party. An example of a third-party trust is a courtroom. Although the defendant and prosecutor may not trust one another, they both can trust the judge (a third party) as fair and impartial. In that case, they implicitly trust each other because they share a common relationship with the judge. In terms of PKI, if Alice does not know Bob, it does not mean that she can never trust his digital certificate. Instead, if she trusts a third-party entity who knows Bob, then she can trust that the digital certificate with the public key is Bob's.

A less secure trust model that uses no third party is called the *web of trust* model and is based on direct trust. Each user signs a digital certificate and then exchanges certificates with all other users. Because all users trust each other, each user can sign the certificate of all other users.

Three PKI trust models use a CA: the hierarchical trust model, the distributed trust model, and the bridge trust model.

Hierarchical Trust Model

The *hierarchical trust model* assigns a single hierarchy with one master CA called the *root*. The root signs all digital certificate authorities with a single key. A hierarchical trust model is illustrated in Figure 7-9.

A hierarchical trust model can be used in an organization where one CA is responsible for only the digital certificates for that organization. However, on a larger scale, a hierarchical trust model has several limitations. First, if the CA's single private key were compromised, then all digital certificates would be worthless. Also, having a single CA who must verify and sign all digital certificates may create a significant backlog.

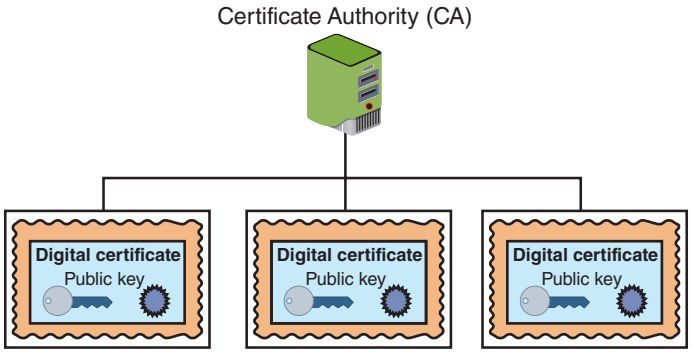


Figure 7-9 Hierarchical trust model

Distributed Trust Model

Instead of having a single CA, as in the hierarchical trust model, the *distributed trust model* has multiple CAs that sign digital certificates. This essentially eliminates the limitations of a hierarchical trust model. The loss of a CA's private key would compromise only those digital certificates it had signed, and the workload of verifying and signing digital certificates can be distributed. In addition, CAs can delegate authority to other intermediate CAs to sign digital certificates. The distributed trust model is the basis for most digital certificates used on the Internet. A distributed trust model is illustrated in Figure 7-10.

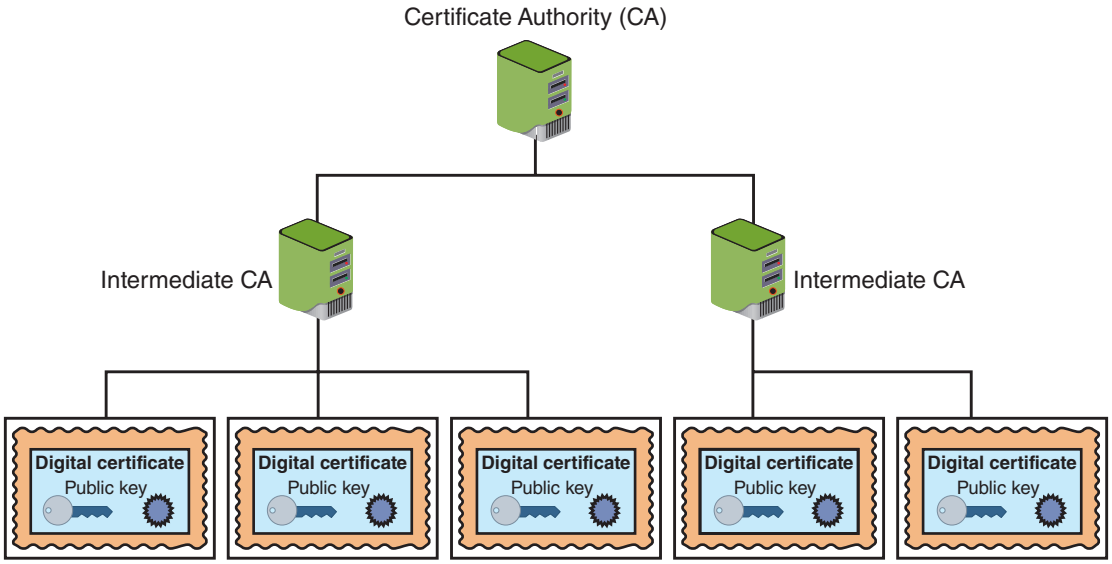


Figure 7-10 Distributed trust model

Bridge Trust Model

The *bridge trust model* is similar to the distributed trust model in that no single CA signs digital certificates. However, with the bridge trust model, one CA acts as a *facilitator* to interconnect all other CAs. The facilitator CA does not issue digital certificates; instead, it acts as the hub between hierarchical trust models and distributed trust models, linking the models together. The bridge trust model is shown in Figure 7-11.

NOTE 10

One application of the bridge trust model involves linking federal and state governments. The U.S. Department of Defense (DOD) has issued millions of identification cards known as Common Access Cards (CACs) to military personnel; these cards are based on the Personal Identity Verification (PIV) standard and are linked to a digital certificate. Some states have begun issuing IDs compatible with the CACs to emergency service personnel, and one state has cross-certified with the federal PKI through a trust bridge for authenticating digital certificates.

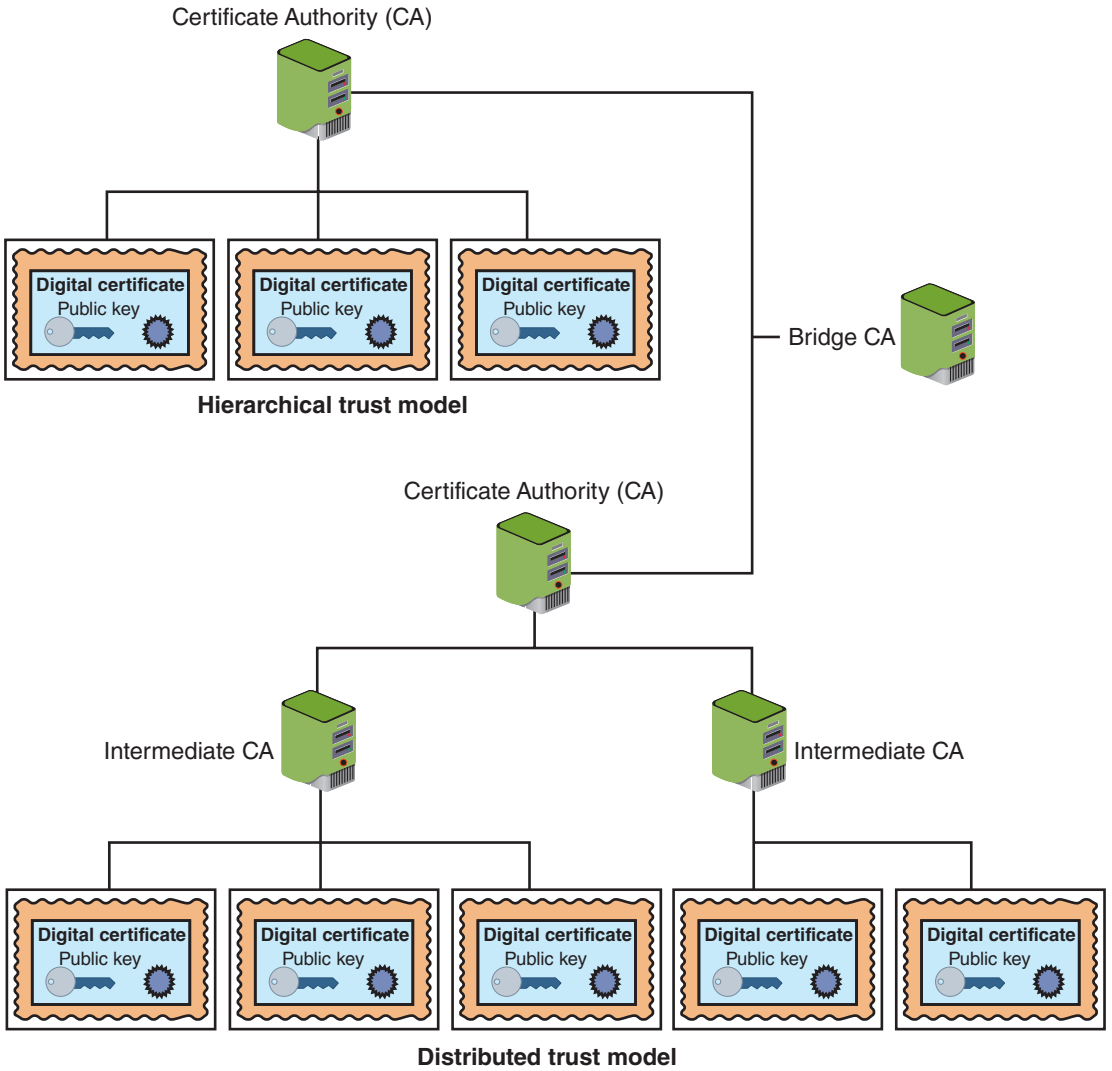


Figure 7-11 Bridge trust model

Managing PKI

An organization that uses multiple digital certificates on a regular basis needs to properly manage those digital certificates. Effective management includes establishing policies and practices and determining the life cycle of a digital certificate.

Certificate Policy (CP)

A *certificate policy* (*CP*) is a published set of rules that govern the operation of a PKI. The CP provides recommended baseline security requirements for the use and operation of CA, intermediate CA, and other PKI components. A CP should cover such topics as CA or intermediate CA obligations, user obligations, confidentiality, operational requirements, and training.

Certificate Practice Statement (CPS)

A *certificate practice statement* (*CPS*) is a more technical document than a CP. A CPS describes in detail how the CA uses and manages certificates. Additional topics for a CPS include how users register for a digital certificate, how to issue digital certificates, when to revoke digital certificates, procedural controls, key pair generation and installation, and private key protection.

X.509 certificates contain a specific field that can link to the associated CP. In addition, extensions may be added to X.509 certificates that indicate how the certificate should be used. One such field, the *Extended Key Usage* field, can

contain an *object identifier (OID)*, which names an object or entity. OIDs are made up of a series of numbers separated with a dot, such as *1.2.840.113585*, and correspond to a node in a hierarchy tree structure. OIDs can name every object type in an X.509 certificate, including the CPS. A large standardized set of OIDs exists, or an enterprise can have a root OID assigned to it and then create its own sub-OIDs, much like creating subdomains beneath a domain.

Certificate Life Cycle

Digital certificates do not last forever: employees leave, new hardware is installed, applications are updated, and cryptographic standards evolve. Each change affects the usefulness of a digital certificate. The life cycle of a certificate is typically divided into four parts:

1. *Creation.* At this stage, the certificate is created and issued to the user. Before the digital certificate is generated, the user must be positively identified. The extent to which the user's identification must be confirmed can vary, depending upon the type of certificate and any existing security policies. Once the user's identification has been verified, the request is sent to the CA for a digital certificate. The CA can then apply its appropriate signing key to the certificate, effectively signing the public key. The relevant fields can be updated by the CA, and the certificate is then forwarded to the registration authority. The CA also can keep a local copy of the certificate it generated. A certificate, once issued, can be published to a public directory if necessary.
2. *Suspension.* This stage could occur once or multiple times throughout the life of a digital certificate if the certificate's validity must be temporarily suspended. Suspension may occur, for example, when employees are on a leave of absence and their digital certificates may not be used for any reason until they return. Upon a user's return, the suspension can be withdrawn or the certificate can be revoked.
3. *Revocation.* At this stage, the certificate is no longer valid. Under certain situations, a certificate may be revoked before its normal expiration date, such as when a user's private key is lost or compromised. When a digital certificate is revoked, the CA updates its internal records, and any CRL with the required certificate information and time stamp (a revoked certificate is identified in a CRL by its certificate serial number). The CA signs the CRL and places it in a public repository so that other applications using certificates can access the repository to determine the status of a certificate.
4. *Expiration.* At the expiration stage, the certificate can no longer be used. Every certificate issued by a CA must have an expiration date. Once it has expired, the certificate may not be used for any type of authentication. The user will be required to follow a process to receive a new certificate with a new expiration date.

Key Management

One common vulnerability that allows threat actors to compromise a PKI is improper certificate and key management. Because keys form the foundation of PKI systems, they must be carefully managed. Proper key management includes key storage, key usage, and key handling procedures.

Key Storage

The means of storing keys in a PKI system is important. Public keys can be stored by embedding them within digital certificates, while private keys can be stored on the user's local system. The drawback to software-based storage is that it can leave keys open to attacks: vulnerabilities in the client operating system, for example, can expose keys to attackers.

Storing keys in hardware is an alternative to software-based storage. For storing public keys, special CA root and intermediate CA hardware devices can be used. Private keys can be stored on smart cards or in tokens.

CAUTION

Whether private keys are stored in hardware or software, they must be adequately protected. To ensure basic protection, never share the key in plaintext, always store keys in files or folders that are themselves password protected or encrypted, do not make copies of keys, and destroy expired keys.

Key Usage

If more security is needed than a single set of public and private keys, multiple pairs of dual keys can be created. One pair of keys may be used to encrypt information, and the public key can be backed up to another location. The second pair would be used only for digital signatures, and the public key in that pair would never be backed up.

Key Handling Procedures

Certain procedures can help ensure that keys are properly handled, including the following:

- *Escrow.* **Key escrow** refers to a process in which keys are managed by a third party, such as a trusted CA. In key escrow, the private key is split, and each half is encrypted. The two halves are registered and sent to the third party, which stores each half in a separate location. A user can then retrieve the two halves, combine them, and use the new copy of the private key for decryption. Key escrow relieves users from worrying about losing their private keys. The drawback to this system is that after a user has retrieved the two halves of the key and combined them to create a copy of the key, that copy of the key can be vulnerable to attacks.
- *Expiration.* Keys have an **expiration** date after which they cease to function. This prevents an attacker, who may have stolen a private key, from being able to decrypt messages for an indefinite period. Some systems set keys to expire after a set period by default.
- *Renewal.* Instead of letting a key expire and then creating a new key, an existing key can be renewed. With renewal, the original public and private keys can continue to be used and new keys do not have to be generated. However, continually renewing keys makes them more vulnerable to theft or misuse.
- *Revocation.* Whereas all keys should expire after a set period, a key may need to be revoked prior to its expiration date. For example, the need for revoking a key may be the result of an employee being terminated from his position. Revoked keys cannot be reinstated. The CA should be immediately notified when a key is revoked, and then the status of that key should be entered on the CRL.
- *Recovery.* What happens if employees are hospitalized, and their organization needs to transact business using their keys? Some CA systems have an embedded key recovery system that designates a *key recovery agent (KRA)*, a highly trusted person responsible for recovering lost or damaged digital certificates. Digital certificates can then be archived along with a user's private key. If the user is unavailable or if the certificate is lost, the certificate with the private key can be recovered. Another technique is known as *M-of-N control*. A user's private key is encrypted and divided into a specific number of parts, such as three. The parts are distributed to other individuals with an overlap, so multiple individuals have the same part. For example, the three parts could be distributed to six people, with two people each having the same part. This is known as the N group. If it is necessary to recover the key, a smaller subset of the N group, known as the M group, must meet and agree that the key should be recovered. If a majority of the M group can agree, they can then piece the key together. M-of-N control is illustrated in Figure 7-12.

NOTE 11

The reason for distributing parts of the key to multiple users is that the absence of one member would not prevent the key from being recovered.

- *Suspension.* The revocation of a key is permanent; key suspension is for a set period. For example, if employees are on an extended medical leave, it may be necessary to suspend the use of their keys for security reasons. A suspended key can be later reinstated. As with revocation, the CA should be immediately notified when a key is suspended, and the status of that key should be checked on the CRL to verify that it is no longer valid.
- *Destruction.* Key destruction removes all private and public keys along with the user's identification information in the CA. When a key is revoked or expires, the user's information remains on the CA for audit purposes.

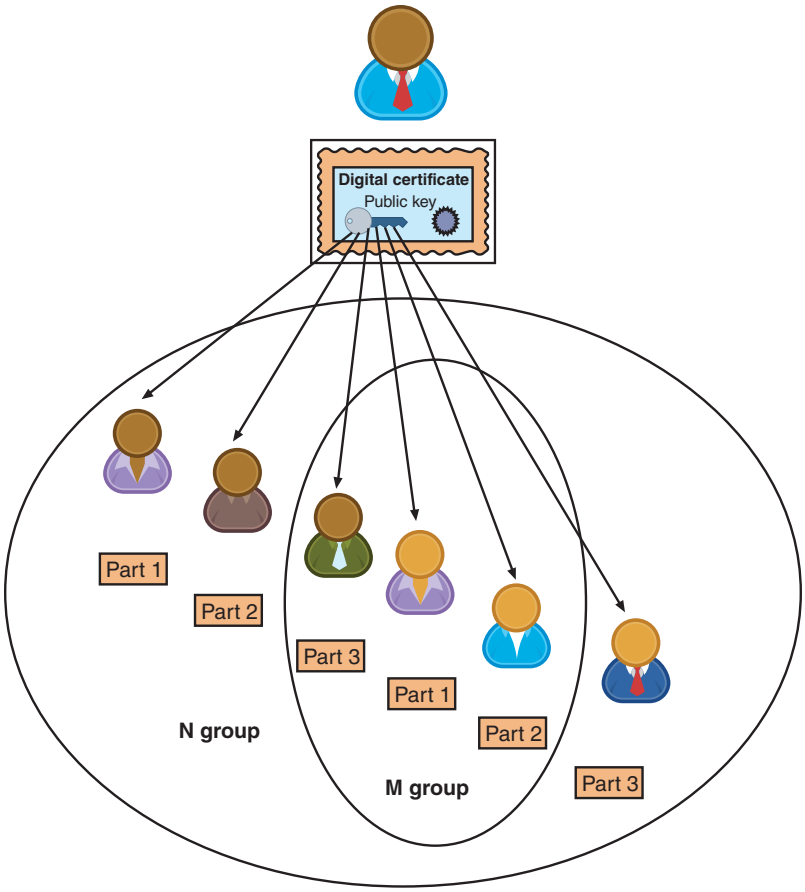


Figure 7-12 M-of-N control

TWO RIGHTS & A WRONG

1. The hierarchical trust model assigns a single hierarchy with one master CA called the root.
2. An OID, which names an object or entity, corresponds to a node in a hierarchy tree structure. OIDs can name every object type in an X.509 certificate.
3. When a digital certificate is revoked, the user must update internal records and any CRL with the required certificate information and timestamp.

See Appendix B for the answer.

CRYPTOGRAPHIC PROTOCOLS

CERTIFICATION

1.3 Given a scenario, analyze potential indicators associated with application attacks.

2.1 Explain the importance of security concepts in an enterprise environment.

2.8 Summarize the basics of cryptographic concepts.

3.1 Given a scenario, implement secure protocols.

In addition to protecting data in processing and at rest, cryptographic algorithms are most often used to protect data in transit or motion across a network. When cryptographic algorithms are used in networks, they are sometimes called *cryptographic protocols*. The most common cryptographic protocols include Secure Sockets Layer, Transport Layer Security, Secure Shell, Hypertext Transport Protocol Secure, Secure/Multipurpose Internet Mail Extensions, Secure Real-time Transport Protocol, and IP security. In addition, cryptographic protocols have weaknesses.

Secure Sockets Layer (SSL)

One of the early and most widespread cryptographic protocols is **Secure Sockets Layer (SSL)**. The protocol was developed by Netscape in 1994 in response to the growing concern over Internet security. The design goal of SSL was to create an encrypted data path between a client and a server that could be used on any platform or operating system. The current version of SSL is Version 3.0.

However, the way SSL functions has left it vulnerable to attack. When the user arrives on a secure webpage via a link from a non-secure site, the following sequence of events take place between a user's browser and a web server: (1) the user's browser sends an unsecured HTTP request to the web server; (2) the server responds via HTTP and redirects the browser (via a *301 redirect* or a temporary *302 redirect*) to a secure page instructing it to use the secure protocol HTTPS; (3) the user's browser then sends a secure HTTPS request, and the secure session begins.

A threat actor can intercept the new request from the user to the server. Attackers can then establish an HTTPS connection between themselves and the server while having an unsecured HTTP connection with the user, giving the threat actors complete control over the secure webpage while the user's responses are sent to the attacker in plaintext. This practice is called **SSL stripping**.

Transport Layer Security (TLS)

Transport Layer Security (TLS) is a widespread cryptographic protocol that is the replacement for SSL. Although the algorithms SSL and TLS use are sometimes listed as being interchangeable or even linked to each other (*TLS/SSL*), they are not. Although TLS v1.0 was considered marginally more secure than SSL v3.0, subsequent versions of TLS are significantly more secure and address several vulnerabilities in SSL v3.0. The current version of TLS is v1.3.

NOTE 12

TLS v1.3 was a significant upgrade over TLS v1.2. It removes support for MD5 and SHA-224, requires use of Perfect Forward Secrecy in case of public-key-based key exchange, and encrypts handshake messages after the *ServerHello* exchange.

A **cipher suite** is a named combination of the encryption, authentication, and message authentication code (MAC) algorithms that are used with TLS and SSL. They are negotiated between the web browser and web server during the initial connection handshake. Cipher suites typically use descriptive names to indicate their components. For example, *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256* specifies that TLS is the protocol, during the handshake keys will be exchanged via the ephemeral Elliptic Curve Diffie Hellman (ECDHE), AES running the Galois Counter Mode with 128-bit key size is the encryption algorithm, and SHA-256 is the hashing algorithm.

Secure Shell (SSH)

Secure Shell (SSH) is an encrypted alternative to the Telnet protocol used to access remote computers. SSH is a Linux/UNIX-based command interface and protocol for securely accessing a remote computer. SSH is actually a suite of the three utilities *slogin* (secure login to a remote computer), *ssh* (execute commands on a remote host without logging in), and *scp* (copy files between remote computers) that are secure versions of the unsecure UNIX counterpart utilities. Both the client and server ends of the connection are authenticated using a digital certificate, and passwords are protected by being encrypted. SSH can even be used as a tool for secure network backups.

Hypertext Transport Protocol Secure (HTTPS)

One common use of TLS and the older SSL is to secure Hypertext Transport Protocol (HTTP) communications between a browser and a web server. The secure version is "plain" HTTP sent over TLS or SSL and is called **Hypertext Transport Protocol Secure (HTTPS)**. HTTPS uses port 443 instead of HTTP's port 80 and users must enter URLs with *https://* instead of *http://*.

NOTE 13

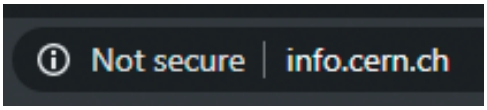
Another cryptographic protocol for HTTP was Secure Hypertext Transport Protocol (SHTTP). However, it was not as secure as HTTPS and is now obsolete.

At one time, web browsers prominently displayed a visual indicator to alert users that the connection between the browser and the web server was using HTTPS. As shown in Figure 7-13, most browsers displayed a green padlock to indicate the connection was encrypted and secure.

Source: Google Chrome web browser

Figure 7-13 Domain validation padlock

However, as more websites transitioned to HTTPS, some web browsers changed from displaying an indicator that the connection *was* secure to only a warning that the connection was *not* secure. In 2018, Google Chrome started displaying a warning if the connection was not secure, as seen in Figure 7-14. Chrome and other browsers, such as Microsoft Edge, still display a padlock, though it is gray, not green.



Source: Google Chrome web browser

Figure 7-14 Google Chrome HTTP warning

NOTE 14

The change away from the green padlock works on the same principle as the lights on a car dashboard. Lights warn drivers of low tire air pressure, low oil pressure, and low gasoline, among other conditions. The lights are designed to turn on only when the vehicle detects a problem, such as the tire air pressure falling below a certain point. The approach of keeping the lights off when "all is OK" and turning on the warning light only when something demands attention is the principle behind migrating away from web browser indicators when a site does not pose a problem.

Secure/Multipurpose Internet Mail Extensions (S/MIME)

Secure/Multipurpose Internet Mail Extensions (S/MIME) is a protocol for securing email messages. MIME is a standard for how an electronic message will be organized, so S/MIME describes how encryption information and a digital certificate can be included as part of the message body. It allows users to send encrypted messages that are also digitally signed.

Secure Real-time Transport Protocol (SRTP)

The **Secure Real-time Transport Protocol (SRTP)** has several similarities to S/MIME. Just as S/MIME is intended to protect MIME communications, SRTP is a secure extension protecting transmissions using the *Real-time Transport Protocol (RTP)*. Also, as S/MIME is designed to protect only email communications, SRTP provides protection for Voice over IP (VoIP) communications. SRTP adds security features, such as message authentication and confidentiality, for VoIP communications.

IP Security (IPsec)

Internet Protocol Security (IPsec) is a protocol suite for securing Internet Protocol (IP) communications. IPsec encrypts and authenticates each IP packet of a session between hosts or networks. IPsec can provide protection to a much wider range of applications than TLS or the older SSL.

IPsec is considered a *transparent* security protocol. It is transparent to the following entities:

- *Applications.* Programs do not have to be modified to run under IPsec.
- *Users.* Unlike some security tools, users do not need to be trained on specific security procedures (such as encrypting with PGP).
- *Software.* Because IPsec is implemented in a device such as a firewall or router, no software changes must be made on the local client.

Unlike TLS, which is implemented as a part of the user application, IPsec is in the operating system or the communication hardware. IPsec is more likely to operate at a faster speed because it can cooperate closely with other system programs and the hardware.

IPsec provides three areas of protection that correspond to three IPsec protocols:

- *Authentication.* IPsec authenticates that packets received were sent from the source. The authentication is identified in the header of the packet to ensure that no specific attacks took place to alter the contents of the packet. This is accomplished by the **Authentication Header (AH)** protocol.
- *Confidentiality.* By encrypting the packets, IPsec ensures that no other parties could view the contents. Confidentiality is achieved through the **Encapsulating Security Payload (ESP)** protocol. ESP supports authentication of the sender and encryption of data.
- *Key management.* IPsec manages the keys to ensure that they are not intercepted or used by unauthorized parties. For IPsec to work, the sending and receiving devices must share a key. This is accomplished through a protocol known as *Internet Security Association and Key Management Protocol/Oakley* (*ISAKMP/Oakley*), which generates the key and authenticates the user using techniques such as digital certificates.

IPsec supports two encryption modes: transport and tunnel. **Transport mode** encrypts only the data portion (payload) of each packet yet leaves the header unencrypted. The more secure **tunnel mode** encrypts both the header and the data portion. IPsec accomplishes transport and tunnel modes by adding new headers to the IP packet. The entire original packet (header and payload) is then treated as the data portion of the new packet.

NOTE 15

Because tunnel mode protects the entire packet, it is generally used in a network-to-network communication, while transport mode is used when a device must see the source and destination addresses to route the packet.

Weaknesses of Cryptographic Protocols

At first glance, it seems cryptographic protocols should have few, if any, weaknesses since they are built upon well-known algorithms. Any security issues with cryptographic protocols would also be equal to those issues within the cryptographic algorithms.

However, that is not the case for the following reasons:

- Due to the inherent complexity of networking, cryptographic protocols are notoriously difficult to design.
- While the mathematics and related security of basic cryptographic algorithms have been extensively studied and are well understood, the same cannot always be said of cryptographic protocols. The critical analysis of these protocols has often not reached the same levels of the cryptographic algorithms. Thus, cryptographic protocol errors and implementation errors in cryptographic protocols that have been in use for decades, are still being uncovered.
- Several older cryptographic protocols still in use today were designed by networking experts and not by cryptographic protocol experts. As a result, cryptographic protocols suffer more from legacy issues than underlying cryptographic algorithm weaknesses.
- The associated security proofs to guarantee the correctness of cryptographic protocols are much more complicated than those for cryptographic algorithms.

TWO RIGHTS & A WRONG

1. SSL is a replacement cryptographic protocol for TLS.
2. A cipher suite is a named combination of the encryption, authentication, and message authentication code (MAC) algorithms that are used with TLS.
3. S/MIME is a protocol for securing email messages.

See Appendix B for the answer.

IMPLEMENTING CRYPTOGRAPHY

CERTIFICATION

2.8 Summarize the basics of cryptographic concepts.

Cryptography that is improperly applied can lead to vulnerabilities that threat actors will exploit. Thus, you should understand the options that relate to cryptography so that it can be implemented correctly. Implementing cryptography includes understanding key strength, secret algorithms, block cipher modes of operation, and cryptographic service providers.

Key Strength

A cryptographic key is a value that serves as input to an algorithm, which then transforms plaintext into ciphertext (and vice versa for decryption). A key, which is essentially a random string of bits, serves as an input parameter for hash, symmetric encryption, and asymmetric cryptographic algorithms.

CAUTION A key is different from a password. Passwords are designed to have people create and remember them so that the passwords can be reproduced when necessary. A key is used by hardware or software that is running the cryptographic algorithm; as such, human readability is not required.

Three primary characteristics determine the resiliency of the key to attacks (called *key strength*). The first is its randomness. A key is considered strong when it is random with no predictable pattern. A random key thwarts an attacker from attempting to uncover the key.

A second characteristic that determines key strength is its *cryptoperiod*, or the length of time for which a key is authorized for use. Having a limited cryptoperiod helps protect the ciphertext from extended cryptanalysis and limits the exposure time if a key is compromised.

NOTE 16

Different cryptoperiods are recommended for different types of keys.

The final characteristic is the length of the key. Shorter keys can be more easily broken than longer keys. All the possible values for a specific key make up its *key space*. The formula for determining a given key space for symmetric algorithms is $character\text{-}set^{key\text{-}length}$. For example, suppose a key has a length of 3 and is using a 26-character alphabet. The list of possible keys (*aaa, aab, aac,* etc.) would be 26^3 or 17,576 possible outcomes. Thus, the key length in this example is *3* and the key space is *17,576*.

On average, half the key space must be searched to discover the key. In the example, a key with a length of only 3 that has a key space of 17,576 requires searching only 8,788 keys (on average) until the correct key is discovered. The number of searches is low, creating a risk that threat actors will compromise the key.

However, if the key length of 3 was increased by just one character to 4, the key space increases to *456,976* requiring on average *228,488* attempts. Table 7-4 illustrates the key strength for varying key lengths, the key space, and average attempts necessary to break the key for a 26-character alphabet.

Table 7-4 Key strength

Key length	Key space	Average number of attempts needed to break
3	17,576	8,788
4	456,976	228,488
5	11,881,376	5,940,688
6	308,915,776	154,457,888
7	8,031,810,176	4,015,905,088
8	208,827,064,576	104,413,532,288

Secret Algorithms

Keys must be kept secret (except for public keys). Does the same apply to algorithms? That is, should an enterprise invest in hiring a cryptographer to create a new cryptographic algorithm and then hide the existence of that algorithm from everyone? Wouldn't such a secret algorithm enhance security in the same way as keeping a key or password secret?

The answer is no. In the past, cryptographers have often attempted to keep their algorithms or the workings of devices that encrypted and decrypted documents a secret. However, the approach has always failed. One reason is that cryptography is the most useful when it is widespread: a military force that uses cryptography must by nature allow many users to know of its existence to use it. The more users who know about it, the more difficult it is to keep it a secret. In contrast, a password only requires one person—the user—to keep it confidential.

NOTE 17

In 1883, Auguste Kerckhoffs, a Dutch linguist and cryptographer, published what is known as *Kerckhoffs's principle*, a set of six design principles for military ciphers. One principle stated that systems should not require secrecy so that it should not be a problem if it falls into enemy hands. This principle is still applied today by splitting algorithms from keys: algorithms are public while keys are private.

Block Cipher Modes of Operation

One variation in cryptographic algorithms is the amount of data that is processed at a time. Some algorithms use a *stream cipher*, while other algorithms make use of a *block cipher*. Whereas a stream cipher works on one character at a time, a block cipher manipulates an entire block of plaintext at one time. Because the size of the plaintext is usually larger than the block size itself, the plaintext is divided into separate blocks of specific lengths, and then each block is encrypted independently.

NOTE 18

Stream and block ciphers are covered in Module 6.

A **block cipher mode of operation** specifies how block ciphers should handle the blocks. It uses a symmetric key block cipher algorithm to provide an information service. The service could be **authentication mode of operation** that provides a credentialing service or **unauthentication mode of operation** that provides a service such as confidentiality. The following are some of the most common modes:

- *Electronic Code Book (ECB)*. The *Electronic Code Book (ECB)* mode is the most basic approach: the plaintext is divided into blocks, and each block is then encrypted separately. However, this can result in two identical plaintext blocks being encrypted into two identical ciphertext blocks. Attackers can use the repetition to their advantage. They could modify the encrypted message by modifying a block or even reshuffling the order of the blocks of ciphertext. ECB is not considered suitable for use.

NOTE 19

Using ECB is like assigning code words from a codebook to create an encrypted message and was the basis for naming the process Electronic Code Book.

- *Cipher Block Chaining (CBC)*. *Cipher Block Chaining (CBC)* is a common cipher mode. After being encrypted, each ciphertext block gets "fed back" into the encryption process to encrypt the next plaintext block. Using CBC, each block of plaintext is XORed with the previous block of ciphertext before being encrypted. Unlike ECB, in which the ciphertext depends only upon the plaintext and the key, CBC is also dependent on the previous ciphertext block, making it much more difficult to break.

NOTE 20

XOR ciphers are covered in Module 6.

- *Counter (CTR)*. **Counter (CTR)** mode requires that both the message sender and receiver access a counter, which computes a new value each time a ciphertext block is exchanged. The weakness of CTR is that it requires a synchronous counter for both the sender and receiver.
- *Galois/Counter (GCM)*. The *Galois/Counter (GCM)* mode both encrypts plaintext and computes a message authentication code (MAC) to ensure that the message was created by the sender and that it was not tampered with during transmission. Like CTR, GCM uses a counter. It adds a plaintext string called *additional authentication data (AAD)* to the transmission. The AAD may contain the addresses and parameters of a network protocol that is being used.

NOTE 21

Block cipher modes can specialize in encryption, data integrity, privacy and integrity, and hard drive encryption. Some specialized modes can even gracefully recover from errors in transmission, while other modes are designed to stop upon encountering transmission errors.

Crypto Service Providers

A *crypto service provider* allows an application to implement an encryption algorithm for execution. Typically, crypto service providers implement cryptographic algorithms, generate keys, provide key storage, and authenticate users by calling various crypto modules to perform the specific tasks. Crypto service providers can be implemented in software, hardware, or both and are often part of the operating system. OSs have cryptographic services enabled and providers may also be created and distributed by third parties, allowing for a broader algorithm selection.

NOTE 22

Applications cannot manipulate the keys created by crypto service providers or alter the cryptographic algorithm itself.

TWO RIGHTS & A WRONG

1. Three primary characteristics determine the resiliency of the key to attacks (called key strength).
2. Counter (CTR) mode requires that both the message sender and receiver access a counter, which computes a new value each time a ciphertext block is exchanged.
3. A block cipher mode of operation specifies how block ciphers should handle streams.

See Appendix B for the answer.

VM LAB You're now ready to complete the live, virtual machine labs for this module. The labs can be found in each module in the MindTap.

SUMMARY

- A digital certificate is the user's public key that has been digitally signed by a trusted third party who verifies the owner and that the public key belongs to that owner. It also binds the public key to the certificate. A user who wants a digital certificate must generate the public and private keys to use and then complete a request known as a Certificate Signing Request (CSR). The user electronically signs the CSR by affixing the public key and then sending it to a registration authority, who verifies the authenticity of the user. The CSR is then sent to an intermediate certificate authority (CA), who processes the CSR. The intermediate CAs perform functions on behalf of a certificate authority (CA) that is responsible for digital certificates. A common method to ensure the security and integrity of a root CA is to keep it in an offline state from the network (offline CA) rather than having it directly connected to a network (online CA).
- A Certificate Repository (CR) is a list of approved digital certificates. Revoked digital certificates are listed in a Certificate Revocation List (CRL), which can be accessed to check the certificate status of other users. The status also can be checked through the Online Certificate Status Protocol (OCSP). When using OCSP stapling, web servers send queries to the Responder OCSP server at regular intervals to receive a signed time-stamped OCSP response. Because digital certificates are used extensively on the Internet, all modern web browsers are configured with a default list of CAs and the ability to automatically update certificate information.
- The process of verifying that a digital certificate is genuine depends upon certificate chaining, or linking several certificates together to establish trust between all the certificates involved. The beginning point of the chain is a specific type of digital certificate known as a root digital certificate, which is created and verified by a CA and also self-signed. Between the root digital certificate and the user certificate can be one or more intermediate certificates that have been issued by intermediate CAs. Root digital certificates and intermediate certificates can be packaged as part of modern OSs, part of web browser software, or hard coded within the app (program) that is using the certificate. The endpoint of the chain is the user digital certificate itself.
- Domain validation digital certificates verify the identity of the entity that has control over the domain name but indicate nothing regarding the trustworthiness of the individuals behind the site. Extended Validation (EV) certificates require more extensive verification of the legitimacy of the business. A wildcard digital certificate is used to validate a main domain along with all subdomains. A Subject Alternative Name (SAN) digital certificate, also known as a Unified Communications Certificate (UCC), is primarily used for Microsoft Exchange servers or unified communications. A machine/computer digital certificate is used to verify the identity of a device in a network transaction. Code signing digital certificates are used by software developers to digitally sign a program and prove that the software comes from the entity that signed it and no unauthorized third party has altered or compromised it. The most widely accepted format for digital certificates is the X.509 standard. Several certificate attributes make up an X.509 digital certificate.

- A public key infrastructure (PKI) is the underlying infrastructure for key management of public keys and digital certificates. It is a framework for all the entities involved in digital certificates—including hardware, software, people, policies, and procedures—to create, store, distribute, and revoke digital certificates. One of the principal foundations of PKI is that of trust. Three basic PKI trust models use a CA. The hierarchical trust model assigns a single hierarchy with one master CA called the root, who signs all digital certificate authorities with a single key. The bridge trust model is similar to the distributed trust model. No single CA signs digital certificates, and yet the CA acts as a facilitator to interconnect all other CAs. The distributed trust model has multiple CAs that sign digital certificates.
- An organization that uses multiple digital certificates on a regular basis needs to properly manage those digital certificates. Such management includes establishing policies and practices and determining the life cycle of a digital certificate. Because keys form the very foundation of PKI systems, they must be carefully stored and handled.
- Cryptography is commonly used to protect data in transit/motion. When cryptographic algorithms are used in networks, they are sometimes called cryptographic protocols. Secure Sockets Layer (SSL) was an early cryptographic transport protocol but was replaced with the more secure Transport Layer Security (TLS). Secure Shell (SSH) is a Linux/UNIX-based command interface and protocol for securely accessing a remote computer communicating over the Internet. Hypertext Transport Protocol Secure (HTTPS), a secure version for web communications, is HTTP sent over TLS or SSL. Secure/Multipurpose Internet Mail Extensions (S/MIME) is a protocol for securing email messages. The Secure Real-time Transport Protocol (SRTP) provides protection for Voice over IP (VoIP) communications. IP security (IPsec) is a set of protocols developed to support the secure exchange of packets. Security weaknesses are associated with cryptographic protocols.
- Cryptography that is improperly applied can lead to vulnerabilities that will be exploited; thus, it is necessary to understand the options that relate to cryptography so that it can be implemented correctly. A key must be strong to resist attacks. A strong key must be random with no predictable pattern. Keys should also be long, and the length of time for which a key is authorized for use should be limited. Any attempt to keep an algorithm secret will not result in strong security. A block cipher mode of operation specifies how block ciphers should handle blocks of plaintext. A crypto service provider allows an application to implement an encryption algorithm for execution.

Key Terms

.cer
.P12
.P7B
Authentication Header (AH)
authentication mode of operation
block cipher mode of operation
Canonical Encoding Rules (CER)
certificate attributes
certificate authority (CA)
certificate chaining
Certificate Revocation List (CRL)
Certificate Signing Request (CSR)
cipher suite
code signing digital certificate
common name (CN)
counter (CTR)
digital certificate
Distinguished Encoding Rules (DER)
domain validation digital certificate
email digital certificate
Encapsulating Security Payload (ESP)
expiration
Extended Validation (EV) certificate
Hypertext Transport Protocol Secure (HTTPS)
intermediate certificate authority (CA)
Internet Protocol Security (IPsec)
key escrow
key management
machine/computer digital certificate
offline CA
online CA
Online Certificate Status Protocol (OCSP)
Personal Information Exchange (PFX)
pinning
Privacy Enhancement Mail (PEM)
public key infrastructure (PKI)
registration authority
root digital certificate
Secure Real-time Transport Protocol (SRTP)
Secure Shell (SSH)
Secure Sockets Layer (SSL)
Secure/Multipurpose Internet Mail Extensions (S/MIME)
self-signed
SSL stripping
stapling
Subject Alternative Name (SAN)
Transport Layer Security (TLS)
Transport mode
trust model
tunnel mode
unauthentication mode of operation
user digital certificate
wildcard digital certificate

Review Questions

1. Which is an IPsec protocol that authenticates that packets received were sent from the source?
   - **a.** PXP
   - **b.** DER
   - **c.** CER
   - **d.** AH

2. What is the name of the fields in an X.509 digital certificate that are used when the parties negotiate a secure connection?
   - **a.** Electronic Code Book (ECB) repositories
   - **b.** Certificate attributes
   - **c.** CTR
   - **d.** PFX

3. What entity calls in crypto modules to perform cryptographic tasks?
   - **a.** Certificate Authority (CA)
   - **b.** Crypto service provider
   - **c.** Intermediate CA
   - **d.** OCSP

4. _____ are symmetric keys to encrypt and decrypt information exchanged during the session and to verify its integrity.
   - **a.** Digital digests
   - **b.** Encrypted signatures
   - **c.** Session keys
   - **d.** Digital certificates

5. What is the name of the device protected by a digital certificate?
   - **a.** CN
   - **b.** TLXS
   - **c.** RCR
   - **d.** V2X2

6. What is the strongest technology that would assure Alice that Bob is the sender of a message?
   - **a.** Digital signature
   - **b.** Encrypted signature
   - **c.** Digest
   - **d.** Digital certificate

7. Olivia is explaining to a friend about digital certificates. Her friend asks what two entities a digital certificate associates or binds together. What would Olivia say?
   - **a.** The users' symmetric key with the public key
   - **b.** The users' public key with their private key
   - **c.** The users' identity with their public key
   - **d.** A private key with a digital signature

8. Which of the following can a digital certificate NOT be used for?
   - **a.** To encrypt messages for secure email communications
   - **b.** To encrypt channels to provide secure communication between clients and servers
   - **c.** To verify the authenticity of the CA
   - **d.** To verify the identity of clients and servers on the web

9. Who verifies the authenticity of a CSR?
   - **a.** Certificate signatory
   - **b.** Registration authority
   - **c.** Certificate authority
   - **d.** Signature authority

10. A centralized directory of digital certificates is called a(n) _____.
    - **a.** Digital signature permitted authorization (DSPA)
    - **b.** Authorized digital signature (ADS)
    - **c.** Digital signature approval list (DSAP)
    - **d.** Certificate repository (CR)

11. Elton needs his application to perform a real-time lookup of a digital certificate's status. Which technology would he use?
    - **a.** Certificate Revocation List (CRL)
    - **b.** Real-Time CA Verification (RTCAV)
    - **c.** Online Certificate Status Protocol (OCSP)
    - **d.** Staple

12. What is the purpose of certificate chaining?
    - **a.** To ensure that a web browser has the latest root certificate updates
    - **b.** To look up the name of intermediate RA
    - **c.** To group and verify digital certificates
    - **d.** To hash the private key

13. Which of the following is NOT a means by which a newly approved root digital certificate is distributed?
    - **a.** Pinning
    - **b.** OS updates
    - **c.** Application updates
    - **d.** Web browser updates

14. Which block cipher mode of operating requires that both the message sender and receiver access a counter that computes a new value whenever a ciphertext block is exchanged?
    - **a.** CTR
    - **b.** CN
    - **c.** CD
    - **d.** CXL

15. Which is the first step in a key exchange?
- **a.** The browser generates a random value ("pre-master secret").
- **b.** The web server sends a message ("ServerHello") to the client.
- **c.** The web browser verifies the server certificate.
- **d.** The web browser sends a message ("ClientHello") to the server.

16. What is the file extension for a Cryptographic Message Syntax Standard based on PKCS#7 that defines a generic syntax for defining digital signature and encryption?
- **a.** .P7B
- **b.** .cer
- **c.** .P12
- **d.** .xdr

17. Juan needs a certificate that must only authenticate that a specific organization has the right to use a particular domain name. What type of certificate does he need?
- **a.** Website validation
- **b.** Root
- **c.** Extended validation
- **d.** Domain validation

18. How is confidentiality achieved through IPsec?
- **a.** ESP
- **b.** AHA
- **c.** ISAKMP
- **d.** AuthX

19. Which refers to a situation in which keys are managed by a third party, such as a trusted CA?
- **a.** Key authorization
- **b.** Key escrow
- **c.** Remote key administration
- **d.** Trusted key authority

20. Which is a protocol for securely accessing a remote computer in order to issue a command?
- **a.** Transport Layer Security (TLS)
- **b.** Secure Shell (SSH)
- **c.** Secure Sockets Layer (SSL)
- **d.** Secure Hypertext Transport Protocol (SHTTP)

Hands-On Projects

CAUTION

If you are concerned about installing any of the software in these projects on your regular computer, you can instead install the software in the Windows virtual machine created in the Module 1 Hands-On Projects. Software installed within a sandbox or the virtual machine will not impact the host computer.

Project 7-1: SSL Server and Client Tests

Time Required: 20 minutes
Objective: Explain the importance of security concepts in an enterprise environment.
Description: In this project, you will use online tests to determine the security of web servers and your local web browser.

NOTE 23

It is not unusual for websites to change the location of their stored files. If the URL no longer works, open a search engine and search for "Qualys SSL Server Test."

1. Go to **www.ssllabs.com**.
2. Click **Test your server >>**.
3. Click the first website listed under **Recent Best**.
4. Note the grade given for this site. Under **Summary**, note the **Overall Rating** along with the scores for **Certificate**, **Protocol Support**, **Key Exchange**, and **Cipher Strength**, which make up the cipher suite.
5. If this site did not receive an Overall Rating of *A* under **Summary**, you will see the reasons listed. Read through these. Would you agree? Why?
6. Scroll through the document and read through the **Certificate #1** information. Note the information supplied regarding the digital certificates. Under **Certification Paths**, click **Click here to expand**, if necessary, to view the certificate chaining. What can you tell about it?
7. Scroll down to **Configuration**. Note the list of protocols supported and not supported. If this site were to increase its security, which protocols should it no longer support? Why?

8. Under **Cipher Suites**, interpret the suites listed. Notice that they are given in server-preferred order. To increase its security, which cipher suite should be listed first? Why?
9. Under **Handshake Simulation**, select the web browser and operating system that you are using or that are similar to what you are using. Read through the capabilities of this client interacting with this web server. Note particularly the order of preference of the cipher suites. Click the browser's back button when finished.
10. Scroll to the top of the page, and then click **Scan Another >>**.
11. Select one of the **Recent Worst** sites. Review the **Summary**, **Authentication**, **Configuration**, **Cipher Suites**, and **Handshake Simulation**. Would you agree with this site's score?
12. If necessary, return to the **SSL Report** page, and then click **Scan Another >>**.
13. Enter the name of your school or work URL and generate a report. What score did it receive?
14. Review the **Summary**, **Authentication**, **Configuration**, **Cipher Suites**, and **Handshake Simulation**. Would you agree with this site's score?
15. Make a list of the top five vulnerabilities that you believe should be addressed in order of priority. If possible, share this list with any IT personnel who may be able to take action.
16. Click **Projects**.
17. Now test the capabilities of your web browser. Click **SSL Client Test**. Review the capabilities of your web browser. Print or take a screen capture of this page.
18. Close this web browser.
19. Open a different web browser on this computer or on another computer.
20. Return to **www.ssllabs.com**, click **Projects**, and then click **SSL Client Test** to compare the two scores. From a security perspective, which browser is better? Why?
21. Close all windows.

Project 7-2: Viewing Digital Certificates

Time Required: 20 minutes
Objective: Given a scenario, implement public key infrastructure.
Description: In this project, you will view digital certificate information using the Google Chrome web browser.

1. Use the Google Chrome web browser to go to **www.google.com**.
2. Note the padlock in the address bar. Although you did not enter *https://*, Google created a secure HTTPS connection. Why would it do that?
3. Click the three vertical buttons at the far edge of the address bar.
4. Click **More tools**.
5. Click **Developer tools**.
6. Click the **Security** tab, if necessary. (If the tab does not appear, click the **>>** button to display more tabs.)
7. Read the information under **Security Overview**.
8. Click **View certificate**.
9. Note the general information displayed on the **General** tab.
10. Now click the **Details** tab. The fields are displayed for this X.509 digital certificate.
11. Click **Valid to** to view the expiration date of this certificate.
12. Click **Public key** to view the public key associated with this digital certificate. Why is this site not concerned with distributing this key? How does embedding the public key in a digital certificate protect it from impersonators?
13. Click the **Certification Path** tab. Because web certificates are based on the distributed trust model, there is a *path* to the root certificate. Click the root certificate, and then click the **View Certificate** button. Click the **Details tab**, and then click **Valid to**. Why is the expiration date of this root certificate longer than that of the website certificate? Click **OK** and then click **OK** again to close the Certificate window.
14. Click **Copy to File**.
15. Click **Next**.
16. Note the different file formats that are available. What do you know about each of these formats?
17. Click **Cancel** to close this window.
18. Close all windows.

Project 7-3: Viewing Digital Certificate Revocation Lists (CRL) and Untrusted Certificates

Time Required: 20 minutes
Objective: Given a scenario, implement public key infrastructure.
Description: Revoked digital certificates are listed in a Certificate Revocation List (CRL), which can be accessed to check the certificate status of other users. In this project, you will view the CRL and any untrusted certificates on your Microsoft Windows computer.

1. Press the **Windows+X** keys.
2. Click **Windows PowerShell (Admin)**.
3. Type **certmgr.msc** and then press **Enter**.
4. In the left pane, expand **Trusted Root Certification Authorities**.
5. In the right pane, double-click **Certificates** to display the CAs approved for this computer. Scroll through this list. How many of these CAs have your heard of before?
6. In the left pane, expand **Intermediate Certification Authorities**.
7. Double-click **Certificates** to view the intermediate CAs. Scroll through this list.
8. In the left pane, click **Certificate Revocation List**. The right pane displays all revoked certificates.
9. In the right pane, select a revoked certificate and double-click it.
10. Read the information about the revoked certificate, and click fields for more detail, if necessary. Why do you think this certificate has been revoked? Close the Certificate Revocation List by clicking the **OK** button.
11. In the left pane, expand **Untrusted Certificates**.
12. Click **Certificate Trust List**. The right pane displays certificates that are no longer trusted.
13. In the right pane, double-click an untrusted certificate. Read the information about it, and click fields for more detail, if necessary. Why do you think this certificate is no longer trusted?
14. Click **OK** to close the Certificate dialog box.
15. Close all windows.

Project 4-4: Downloading and Installing a Digital Certificate

Time Required: 25 minutes
Objective: Given a scenario, implement public key infrastructure.
Description: In this project, you will download and install a digital certificate within the Adobe Acrobat Reader DC.

1. Check to determine if Adobe Acrobat Reader DC or Adobe Acrobat Professional is installed on your computer. If so, you may skip these download and installation steps and go directly to Step 5.

NOTE 24

It is not unusual for websites to change the location of stored files. If the URL no longer works, open a search engine and search for "Adobe Acrobat Reader DC Download."

2. Go to **get.adobe.com/reader/**.
3. Click **Download Acrobat Reader**.
4. Follow the instructions to install Reader.
5. Launch Reader.
6. Click **Edit**.
7. Click **Preferences**.
8. Click **Signatures**.
9. Under **Identities & Trusted Certificates**, click **More**.
10. In the left pane, click **Digital IDs** to display the menu choices, if necessary.
11. At the menu at the top of the main pane, click the **Add ID** icon (it is the first icon and has a plus sign).
12. Click **A new digital ID I want to create now**. Click **Next**.
13. If necessary, click **New PKCS#12 digital ID file**. What is a PKCS#12? What type of file extension will it have? Click **Next**.

14. Enter the requested information. Under **Key Algorithm**, click the down arrow to display two options. The default is **2048-bit RSA**, which provides more security, while 1024-bit RSA provides less security but is more universally compatible. Accept the 2048-bit RSA.
15. Under **Use digital ID for**, click the down arrow to display three options. Select the default **Digital Signatures and Data Encryption**. Click **Next**.
16. Create and enter a strong password, and then confirm that password. Click **Finish**.
17. Your file is now created. Click **Export**.
18. If necessary, click **Save the data to a file** and then click **Next**.
19. Save the file to your computer.
20. Close the windows associated with configuring your certificate. You can use this certificate by sending it to anyone who needs to validate your identity.
21. Close all windows.

Case Projects

Case Project 7-1: Transport Layer Security (TLS)

Use the Internet to research TLS. Who was responsible for developing it? When was the first version released? What was the relationship between TLS and SSL? What are its strengths? What are its weaknesses? When will the next version be released? What improvements are projected? Write a one-page paper on your research.

Case Project 7-2: Recommended Cryptoperiods

How long should a key be used before it is replaced? Search the Internet for information regarding cryptoperiods for hash, symmetric, and asymmetric algorithms. Find at least three sources for each of the algorithms. Draw a table that lists the algorithms and the recommended time, and then calculate the average for each. Do you agree or disagree? What would be your recommendation on cryptoperiods for each? Why?

Case Project 7-3: Certificate Authorities (CAs)

OSs come packaged with many digital certificates from certificate authorities (CAs). Use the Internet to determine how to view the CAs for the type and version of OS that you are using and view the list. How many have you heard of? How many are unknown? Select three of the publishers and research their organizations on the Internet. Write a one-paragraph summary of each CA.

Case Project 7-4: Root Certificate Breaches

Use the Internet to research breaches of CAs. What CAs were involved? Were they root or intermediate CAs? Who was behind the theft? How did the thefts occur? How were the stolen certificates then used? Are certificates from these CAs still accepted? Write a one-page paper of your research.

Case Project 7-5: Block Cipher Modes of Operation

Research block cipher modes of operation. Find information regarding how ECB can be compromised and write a detailed description of that. Then research one of the other modes (CBC, CTR, or GCM) in detail. Draw a picture of how this mode functions by turning plaintext into ciphertext. Write a detailed description of your research.

Case Project 7-6: Digital Certificate Costs

Use the Internet to research the costs of the different types of digital certificates: domain validation, EV, wildcard, SAN, machine/computer, code signing, and email. Look up at least three providers of each, and create a table listing the type of certificate, the costs, and the length of time the certificate is valid.

Case Project 7-7: Community Site Activity

The Information Security Community Site is an online companion to this textbook. It contains a wide variety of tools, information, discussion boards, and other features to assist learners. Go to **community.cengage.com/infosec2** and click the *Join or Sign in* icon to login, using your login name and password that you created in Module 1. Click **Forums (Discussion)** and click on **Security+ Case Projects (7th edition)**. Read the following case study.

Read again the section in the module on *Weaknesses of Cryptographic Protocols.* A weak cryptographic protocol can have serious implications for tens or even hundreds of millions of Internet users in a single day. What protections should be implemented over the development and deployment of cryptographic protocols to prevent this? What type of testing should be used? Should there be penalties for vulnerabilities that are uncovered in cryptographic protocols? Record your answers on the Community Site discussion board.

Case Project 7-8: North Ridge Security

North Ridge Security provides security consulting and assurance services. You have recently been hired as an intern to assist them.

Family Spas and Pools (FS&P) is a new North Ridge client that had been the recent victim of different security breaches. A vice president has recently heard about digital certificates for the first time and now wants all users to have and use digital certificates for every function related to FS&P. You are asked to provide information about digital certificates to help FS&P create and use digital certificates in the most meaningful way.

1. Create a PowerPoint presentation about digital certificates, including what they are, what they can protect, how they should be used, and the various types of certificates. Include the advantages and disadvantages of each. Your presentation should contain at least 10 slides.
2. After the presentation, the vice president of FS&P asks which certificates they should use and how they should be managed. Create a memo communicating the actions you believe would be best for the company to take.

References

1. Cowell, Alan, "Code found on pigeon baffles British cryptographers," *New York Times,* Nov. 24, 2012, accessed Feb. 5, 2014, www.nytimes.com/2012/11/24/world/europe/code-found-on-pigeon-baffles-british-cryptographers.html?_r=0.
2. Gennaro, Lisa, "68 useful ecommerce statistics you must know in 2020," *Wpforms*, Feb. 15, 2020, accessed Jun. 9, 2020, https://wpforms.com/ecommerce-statistics/.