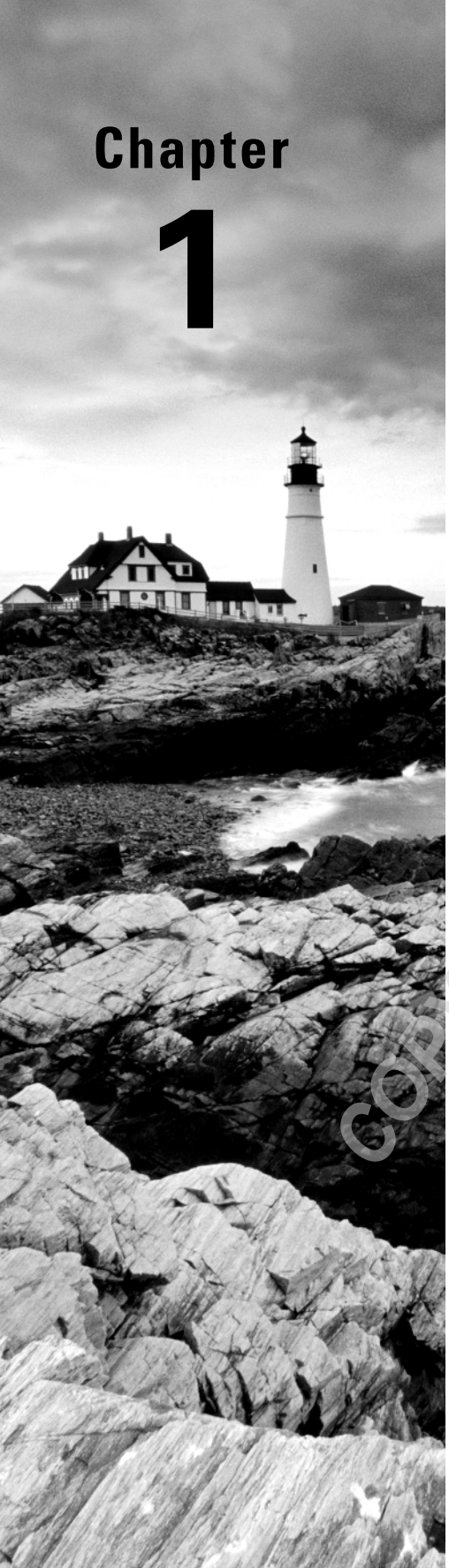


Chapter 1

Cryptographic Tools and Techniques

THE FOLLOWING COMPTIA CASP+ EXAM OBJECTIVES ARE COVERED IN THIS CHAPTER:

✓ **2.1 Analyze a scenario and integrate network and security components, concepts and architectures to meet security requirements.**

✓ **2.1 Analyze a scenario and integrate network and security components, concepts and architectures to meet security requirements.**

- Physical and virtual network and security devices
  - HSM

✓ **2.3 Analyze a scenario to integrate security controls for mobile and small form factor devices to meet security requirements.**

- Security implications/privacy concerns
  - TPM

✓ **4.4 Given a scenario, implement cryptographic techniques.**

- Techniques
  - Key stretching
  - Hashing
  - Digital signature
  - Message authentication
  - Code signing
  - Pseudo-random number generation
  - Perfect forward secrecy

COPYRIGHTED MATERIAL

- Data-at-rest encryption
  - Disk
  - Block
  - File
  - Record
- Steganography

- Implementations
  - DRM
  - Watermarking
  - GPG
  - SSL/TLS
  - SSH
  - S/MIME
  - Cryptographic applications and proper/improper implementations
    - Strength
    - Performance
    - Feasibility to implement
    - Interoperability
  - Stream vs. block
  - PKI
    - Wild card
    - OCSP vs. CRL
    - Issuance to entities
    - Key escrow
    - Certificate
    - Tokens
    - Stapling
    - Pinning
  - Cryptocurrency/blockchain

This chapter discusses *cryptography*, which can be defined as the art of protecting information by transforming it into an unreadable format. Everywhere you turn you see cryptography. It is used to protect sensitive information, prove the identity of a claimant, and verify the integrity of an application or program. As a security professional for your company, which of the following would you consider more critical if you could choose only one?

- Provide a locking cable for every laptop user in the organization.
- Enforce full disk encryption for every mobile device.

Our choice would be full disk encryption. Typically, the data will be worth more than the cost of a replacement laptop. If the data is lost or exposed, you'll incur additional costs such as client notification and reputation loss.

As a security professional, you should have a good basic understanding of cryptographic functions. This chapter begins by reviewing a little of the history of cryptography. Next, we discuss basic cryptographic types, explaining symmetric and asymmetric encryption, hashing, digital signatures, and public key infrastructure. These concepts are important as we move on to more advanced topics and begin to look at cryptographic applications. Understanding them will help you prepare for the CompTIA exam and to implement cryptographic solutions to protect your company's assets better.

The History of Cryptography

Encryption is not a new concept. The desire to keep secrets is as old as civilization. There are two basic ways in which encryption is used: for data at rest and for data in motion. Data at rest might be information on a laptop hard drive or in cloud storage. Data in motion might be data being processed by SQL, a URL requested via HTTP, or information traveling over a VPN at the local coffee shop bound for the corporate network. In each of these cases, protection must be sufficient. The following list includes some examples of early cryptographic systems:

Scytale This system functioned by wrapping a strip of papyrus or leather, on which a message was written, around a rod of fixed diameter. The recipient used a rod of the same diameter to read the message. Although such systems seem basic today, it worked well in the time of the Spartans. Even if someone was to intercept the message, it appeared as a jumble of meaningless letters.

Caesar's Cipher Julius Caesar is known for an early form of encryption, the Caesar cipher, which was used to transmit messages sent between Caesar and his generals. The cipher worked by means of a simple substitution. Before a message was sent, the plain text

was rotated forward by three characters (ROT3). Using Caesar's cipher to encrypt the word *cat* would result in *fdw*. Decrypting required moving back three characters.

Other Examples Substitution ciphers replace one character for another. The best example of a substitution cipher is the Vigenère polyalphabetic cipher. Other historical systems include a running key cipher and the Vernam cipher. The running key cipher is another way to generate the keystream for use with the tabula recta. The Vernam is also known as the *onetime pad*.

Cryptographic Services

As a security professional, you need to understand cryptographic services and how they are applied. You also need to know the goals of cryptography and basic terms. Although your job may not require you to be a cryptographic expert, to pass the CASP+ exam you should be able to explain how specific cryptographic functions work.

Cryptographic Goals

Cryptography includes methods such as symmetric encryption, asymmetric encryption, hashing, and digital signatures. Each provides specific attributes and solutions. These cryptographic services include the following goals:

Privacy Also called confidentiality. What is private (confidential) should stay private, whether at rest or in transit.

Authentication There should be proof that the message is from the person or entity you believe it to be from.

Integrity Information should remain unaltered at the point at which it was produced, while it is in transmission, and during storage.

Non-repudiation The sender of data is provided with proof of delivery, and the recipient is assured of the sender's identity.

An easy way to remember these items for the exam is to think of PAIN. This simple acronym (privacy, authentication, integrity, and non-repudiation) should help you remember the basic cryptographic goals.

Knowing these basic goals can go a long way in helping you to understand that cryptography can be used as a tool to achieve confidentially, integrity, and availability. For example, consider how encryption can protect the *privacy* and confidentiality of information at rest or in transit. What if your CEO has been asked to travel to the Far East for trade negotiations? Think about the CEO's laptop. If it is lost or compromised, how hard would it be for someone to remove unencrypted data? Strong encryption offers

an easy way to protect that information should the equipment be lost, stolen, or accessed by unauthorized individuals. Applications such as CryptoForge and BitLocker offer the ability to encrypt a hard drive. PKWare provides users with enterprise data security that persistently protects and manages data whenever it is used, shared, and stored, both inside and outside the organization. Sookasa transparently protects files across the Dropbox and Google Drive clouds as well as linked mobile devices while preserving the native user experience on the Windows, MacOS, iOS, and Android operating systems.

During a trip to Beijing in December 2007, it was discovered that someone had accessed a laptop used by former Commerce Secretary Carlos Gutierrez and had placed monitoring programs on it designed to secretly remove information. Read more at `http://fortune.com/2016/07/27/ceo-twitter-hack-list/`.

Authentication is another key goal of cryptography. First, *authentication* is associated with digital signatures. Authentication provides a way to ensure that a message is from whom we believe it's from. In its basic form, authentication is used to determine identity. It is also part of the identification and authentication process.

Integrity is another cryptographic goal. Integrity is important while data is in transmission and in storage. *Integrity* means that information remains unaltered. Imagine the situation of needing to download a patch. Although the patch is available on the developer's site, you also have a copy on DVD that was given to you by a colleague. Is the version on the DVD the same as the one on the developer's website? Integrity verification programs that perform hashing such as MD5 or SHA can help you determine this.

Non-repudiation is assurance that an entity in a communication cannot deny authenticity. It is proof of the veracity of a claim. Non-repudiation means that a sender of data receives proof of delivery and the recipient is assured of the sender's identity. Neither party should be able to deny having sent or received the data at a later date. This can be achieved with digital signatures. A *digital signature* provides authenticity, integrity, and non-repudiation. In the days of face-to-face transactions, non-repudiation was not as hard to prove. Today, the Internet makes many transactions faceless. We may never see the people we deal with; therefore, non-repudiation becomes all the more critical. Non-repudiation is achieved through digital signatures, digital certificates, and message authentication codes (MACs).

When implementing a cryptographic system, there has to be consideration of strength versus performance versus feasibility to implement versus interoperability. Stronger systems typically require more process power and longer encryption/decryption times. Basically, you must consider how strong an encryption process should be. The strength of a cryptosystem relies on the strength of an algorithm and the complexity of the key generation process. The strength of the encryption mechanism also rests on the size and complexity of the key. If the cryptosystem uses a weak key generation process, then the entire process is weak. The key size goes a long way in determining the strength of the cryptosystem.

The designer of a cryptographic system must also understand the implications of cryptographic methods and design. As an example, Caesar might have thought his system of encryption was quite strong, but it would be seen as relativity insecure today. You need a sufficiently sized key to deter brute-force and other attacks. In the world of cryptography, key lengths are defined by the number of binary bits. So, a 64-bit key has a keyspace of 2 to the power of 64, or 18,446,744,073,709,551,616.

Cryptographic Terms

As a security professional, you need to understand basic cryptographic terms. You will encounter these terms when examining a vendor's security solution, discussing security controls with colleagues, and implementing a security solution. Here are some basic cryptographic terms:

Plain Text Clear text that is readable

Cipher Text Encrypted text that is unreadable

Encryption Transforming data into an unreadable format. For example, using Caesar's cipher to encrypt the word *dog* would result in *grj*. Encryption here has moved each character forward by three letters.

Cryptanalysis The act of obtaining plain text from cipher text without a cryptographic key. It is used by governments, the military, enterprises, ethical hackers, and malicious hackers to find weaknesses and crack cryptographic systems.

Digital Signature A hash value that has been encrypted with the private key of the sender. It is used for authentication and integrity.

Chain of Trust The relationship between subordinate certificate authorities. The concept of chain of trust is critical in the world of public key infrastructure as it provides a means to pass trust from one entity to another. It allows the delegation of certificate duties to a subordinate certificate authority.

Root of Trust Root of trust can be described as the concept of trust in a system, software, or data. It is the most common form of attestation, and it provides a basic set of functions that are always trusted by the operating system. *Attestation* means that you are validating something as true. A root of trust can be designed as hardware-based, software-based, or hybrid. The *Trusted Platform Module (TPM)* is one of the most common.

Think of root of trust as something that has been deemed trustworthy. As an example, if you are asked to serve on the jury of a court case, the lawyers should be seen as trustworthy. That's because the court trusts that the lawyers are licensed to practice law in the state and that a client-to-lawyer relationship has been established by the legal system and because the court uses a well-defined procedural process for evidence to be admitted. Although computer systems don't need lawyers, let's hope, they do need trust, and that is the role that TPM plays. TPM has a root of trust that is defined by the endorsement key (EK) pair. It is a unique RSA key found within all TPM devices.

Cryptographic systems can be broadly classified into symmetric, asymmetric, and hashing:

Symmetric Cryptography This type uses a single private key.

Asymmetric Cryptography This type uses two keys: a public key known to everyone and a private key that only the recipient of messages uses.

Although both concepts are discussed in more detail later in the chapter, at this point it's important to understand that both symmetric and asymmetric cryptography make use of a key. The key is input into the encryption algorithm as data on which to perform mathematical operations such as permutation, substitution, or binary math.

Hash A *hash* is a defined mathematical procedure or function that converts a large amount of data into a fixed small string of data or integer. The output of a hash is known as a hash value, hash code, hash sum, checksum, fingerprint, or message digest.

For the CASP+ exam, more than one term may be used to describe a hash.

Here are some other terms that you will need to know for the exam:

Algorithm An *algorithm* is a set of rules or ordered steps used to encrypt and decrypt data. The algorithm is a set of instructions used with the cryptographic key to encrypt plain text data. Plain text data encrypted with different keys or dissimilar algorithms will produce different cipher text.

Cipher Text *Cipher text* is data that is scrambled and unreadable. When plain text is converted into cipher text, the transformation can be accomplished in basically two ways:

Block Ciphers These function by dividing the message into blocks for processing.

Stream Ciphers These function by dividing the message into bits for processing.

Cryptographic Key The strength of the encryption process is based in part on the cryptographic key. The *cryptographic key,* or simply *key*, is a piece of information that controls how the cryptographic algorithm functions. It can be used to control the transformation of plain text to cipher text or cipher text to plain text. For attackers to brute-force the cryptographic system, they would need to guess the key. That is why the more values or combinations for the key, the longer it will take for an attacker to gain access to your encrypted data. The security of the system rests in the key. If the key generation process is weak, the entire system that is designed around it will also be weak. A good example of this can be seen with Wired Equivalent Privacy (WEP), whose use of RC4 and weak key generation led to many of the attacks against this wireless protection system.

Weak key generation might be caused by repeating values. On wireless networks with high volumes of traffic, keys may be reused in just a few hours. This weakness allows an attacker to collect traffic and capture the weak keys in an attempt to derive the shared key and then gain access to the WEP-protected wireless network.

Entropy Although key size is important, the randomness of the key is also critical. You may have been asked to create a random key before and not have realized what you were actually doing. For example, many security products begin the process of generating a pseudorandom key by having the user tap random keys on a keyboard, randomly move the mouse, or create random network Ethernet traffic. Such activity is known as entropy. *Entropy* is a measure of the randomness of data collected by an application or an operating system and used to create a cryptography key.

Having a random key is a good start, but the key must also remain secret. This is no different than thinking of your password as a key. If everyone knows the password to your computer, anyone can access it at any time they please. High-value data requires strong protection, which typically means longer keys that are exchanged more frequently in order to protect against attacks.

Not all cryptosystems are of the same strength. For example, Caesar's cipher seemed quite strong when it was created, but it is insecure today. As a security professional, always ask how strong an encryption process should be.

Cryptographic systems may also make use of a nonce. A *nonce* is a number used once—that is, as random a number as a cryptosystem can generate. The programs that create these are known as *pseudorandom number generators*. Such systems use algorithms to generate a sequence of numbers that approximates the properties of random numbers. Pseudorandom numbers are unique and different each time one is generated.

If you are interested in seeing programs that can be used to create pseudorandom numbers, take a moment to check out `www.agner.org/random/`.

An initialization vector (IV) is an example of a type of nonce. An IV is used to create a unique cipher text every time the same message is encrypted using the same key.

Table 1.1 highlights some of the strengths and weaknesses of symmetric and asymmetric encryption.

TABLE 1.1 Symmetric and asymmetric encryption

Encryption type	Advantage	Disadvantage
Symmetric	Faster than asymmetric	Key distribution is difficult and must be done out of band; symmetric encryption provides only confidentiality.

Encryption type	Advantage	Disadvantage
Asymmetric	Easy key exchange	Can provide confidentiality and authentication, but it does so more slowly than with symmetric. It is so slow that it's typically used only to move small amounts of data.

Cipher Types and Methods

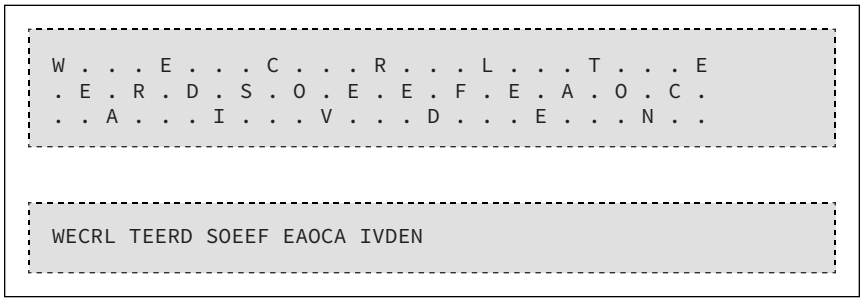
Let's now continue with our discussion of block and stream ciphers.

Block Ciphers

Block ciphers are widely used in software products. Most modern encryption algorithms implement some type of block cipher.

Block ciphers operate on blocks or fixed-size chunks of data; 64-bit blocks are commonly used. One type of block cipher is a *transposition cipher*, which shifts units of plain text in a consistent way so that the cipher text constitutes a permutation of the plain text. An example of this can be seen in a rail-fence cipher. This type of transposition cipher encrypts the message in a downward pattern on successive rails of an imaginary fence; then it moves up toward the top when the bottom is reached. This pattern repeats itself over successive rails. The message is then encrypted by being read off in rows. Figure 1.1 shows how a rail-fence cipher of the message of "WE ARE DISCOVERED. FLEE AT ONCE." would appear.

FIGURE 1.1 A rail-fence cipher (an example of a transposition cipher)



There are various encryption methods used in block ciphers. During the encryption and decryption process, the message is divided into blocks of bits. These blocks are then put through functions such as substitution, transposition, confusion, and diffusion.

Substitution Using this method means to put one thing in the place of another, such as one letter for another, or letters for numbers, and so on.

Transposition This method scrambles a message by reordering the plain text in some definite way.

Confusion This method uses a relationship between the plain text and the key that is so complicated that an attacker can't alter the plain text and determine the key.

Diffusion In this method, a change in the plain text results in multiple changes spread out throughout the cipher text.

The *substitution box (s-box)* is one technique that is used to introduce confusion. When properly implemented, s-boxes are designed to defeat cryptanalysis. An s-box takes a number of input bits, *m*, and transforms them into some number of output bits, *n*. S-boxes can be implemented as a type of lookup table and used with symmetric encryption systems such as the Data Encryption Standard (DES) and the newer Triple DES, discussed later in the chapter.

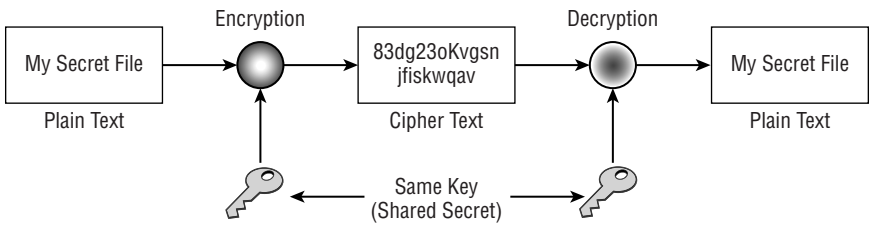
Stream Ciphers

A *stream cipher* inputs digits, bits, or characters and encrypts the stream of data. The onetime pad is an example of a stream cipher. The onetime pad works on each letter of the plain text message independently. A stream cipher combines the plain text bit with a pseudorandom cipher bit stream by means of an exclusive OR (XOR) operation. Stream ciphers operate at a higher speed than block ciphers and in theory are well suited for hardware implementation.

Symmetric Encryption

Symmetric encryption uses a single shared key for encryption and decryption. These are known as dual-use keys, as they can be used to lock and unlock data. Symmetric encryption is the oldest form of encryption. Historical systems such as scytale and Caesar's cipher are types of symmetric encryption. Symmetric encryption offers users privacy by keeping individuals who do not have the key from having access to the true contents of the message. Figure 1.2 shows the symmetric encryption process.

FIGURE 1.2 Symmetric encryption process



Notice how the plain text is encrypted with the single shared key and is then transmitted to the recipient of the message, who goes through the same process to decrypt the message. The dual use of keys is what makes this system so simple, but it also introduces weakness. Symmetric encryption is fast, and with a small key it can be used to encrypt bulk data very quickly. It is also strong and difficult to break if the key is of sufficient size. However, symmetric encryption does have disadvantages.

The problem is key distribution. For symmetric encryption to be effective, there must be a secure method in place to transfer keys. In our modern world, there needs to be some type of out-of-band transmission. *Out-of-band transmission* means using a different means to transmit the key. As an example, if Bob wants to send Alice a secret message but is afraid that Mike can monitor their communication, how can he send the message? If the key is sent in clear text, Mike can intercept it. Bob could deliver the key in person, mail it, or even send a courier. All of these out-of-band methods are highly impractical in the world of e-commerce and electronic communications because they do not scale well.

Even if the problems of key exchange are overcome, there are still other concerns. Another problem is key management. If, for example, 10 people needed to communicate using symmetric encryption, the number of keys needed would be 45. As the number of people using symmetric encryption rises, so does the required number of keys. To determine the numbers of keys needed in symmetric encryption, the following formula is used:

$$n(n - 1)/2$$

which simplifies to

$$n(n - 1) \div 2 \text{ [or } 10(10 - 1) \div 2 = 45 \text{ keys]}$$

The third and final flaw with symmetric encryption is that it only provides confidentiality.

For the CASP+ exam, you should understand the three primary issues involved with the use of symmetric encryption. These include concerns over key exchange and key management and the fact that symmetric encryption offers only confidentiality.

Although it is true that symmetric encryption is not perfect, it does offer some great features that make it an excellent choice for securing data and providing confidentiality. Symmetric encryption is fast. It can encrypt and decrypt very quickly, and it is considered strong. Symmetric encryption is very hard to break if a large key is used. Here are some well-known symmetric algorithms:

Data Encryption Standard The *Data Encryption Standard (DES)* was once the most common symmetric algorithm used. It has now been officially retired by the National Institute of Standards and Technology (NIST). Its short-term replacement was 3DES. Today, all versions of DES have been replaced by the Advanced Encryption Standard (AES).

Advanced Encryption Standard The *Advanced Encryption Standard (AES)* is the symmetric algorithm chosen as a replacement for DES. It was adopted from the Rijndael algorithm, and it is used for sensitive and secret data. Its key sizes are 128-, 192-, and 256-bit.

Blowfish *Blowfish* is a general-purpose symmetric algorithm intended as a replacement for DES. Blowfish has a variable block size and up to a 448-bit key.

Carlisle Adams/Stafford Tavares *Carlisle Adams/Stafford Tavares (CAST)* is a 128- or 256-bit block cipher that was a candidate for AES.

International Data Encryption Algorithm The *International Data Encryption Algorithm (IDEA)* is a block cipher that uses a 128-bit key to encrypt 64-bit blocks of plain text. It is used by Pretty Good Privacy (PGP).

Rivest Cipher 4 *Rivest Cipher 4 (RC4)* is a stream-based cipher. Stream ciphers treat the data as a stream of bits.

Rivest Cipher 5 *Rivest Cipher 5 (RC5)* is a fast-block cipher. It is different from other symmetric algorithms in that it supports a variable block size, a variable key size, and a variable number of rounds. A *round* is a sequential repetition of a series of math functions. Allowable choices for the block size are 32, 64, and 128 bits. The key can range up to 2,040 bits.

Secure and Fast Encryption Routine *Secure and Fast Encryption Routine (SAFER)* is a block-based cipher that processes data in blocks of 64 and 128 bits.

Skipjack Promoted by the US National Security Agency (NSA), *Skipjack* uses an 80-bit key and operates on 64-bit blocks of text. Skipjack faced opposition because the government would maintain a portion of the information required to reconstruct a Skipjack key so that legal authorities could decrypt communications between the affected parties when approved by a court.

Twofish *Twofish* is a block cipher that operates on 128-bit blocks of data and is capable of using cryptographic keys up to 256 bits in length.

Now let's look at some of the popular symmetric encryption standards in more depth.

Data Encryption Standard

DES was originally developed by IBM and then modified by NIST. The NSA endorsed the revised standard. It was published in 1977, and it was released by the American National Standards Institute (ANSI) in 1981.

DES is a symmetric encryption standard that is based on a 64-bit block that processes 64 bits of plain text at a time. DES outputs 64-bit blocks of cipher text. The DES key size is 56 bits, and DES has four primary modes of operation:

- Electronic codebook (ECB) mode
- Cipher block chaining (CBC) mode
- Output feedback (OFB) mode
- Cipher feedback (CFB) mode

All four modes use the 56-bit key, and though the standard lists the key as 64 bits, 8 bits are used for parity checking, so the true key size is actually 56 bits. *Parity checking* is a simple form of error detection. Each 64-bit, plain text block is separated into two 32-bit blocks and then processed by the 56-bit key. The plain text is processed by the key through 16 rounds of transposition and substitution.

Examine closely any CASP+ exam question that mentions DES. Remember that although DES operates on 64-bit blocks, the effective key length is only 56 bits.

Electronic Codebook Mode

Electronic codebook (ECB) mode is the default mode of encryption used by DES. If the last block is not a full 64 bits, padding is added. ECB produces the greatest throughput, but it is also the easiest implementation of DES encryption to crack. If used with large amounts of data, it is easily broken because the same plain text encrypted with the same key always produces the same cipher text. This is why if you use ECB, you should do so only on small amounts of data.

When you're using ECB, keep in mind that a fixed key and a known repeating plain text message will always produce the same cipher text.

Cipher Block Chaining Mode

When DES is operating in *cipher block chaining (CBC) mode*, it is somewhat similar to ECB except that CBC inserts some of the cipher text created from the previous block into the next one. This process is called *XORing*. It makes the cipher text more secure and less susceptible to cracking. CBC is aptly named because data from one block is used in the next, and the blocks are chained together. This chaining produces dependency, but it also results in more random cipher text.

Output Feedback Mode

Output feedback (OFB) mode is implemented as a stream cipher and uses plain text to feed back into the stream of cipher text. Transmission errors do not propagate throughout the encryption process. An initialization vector is used to create the seed value for the first encrypted block. DES XORs the plain text with a seed value to be applied with subsequent data.

Cipher Feedback Mode

Cipher feedback (CFB) mode can be implemented as a stream cipher and used to encrypt individual characters. CFB is similar to OFB in that previously generated cipher text is

added to subsequent streams. Because the cipher text is streamed together, errors and corruption can propagate through the encryption process.

How secure is DES? Not as secure as it once was. Computing power has increased over the years, and that has decreased the time required to brute-force DES. In 1998, the Electronic Frontier Foundation was able to crack DES in about 23 hours, and that was over 20 years ago. Today's computing power could brute-force DES in minutes.

Triple DES

Triple DES (3DES) was designed to be a stopgap solution. DES was initially certified for five years and was required to be recertified every five years. While easily passing these recertifications in the early years, DES began to encounter problems around the 1987 recertification. By 1993, NIST stated that DES was beginning to outlive its usefulness. They began looking for candidates to replace it. This new standard was to be referred to as the *Advanced Encryption Standard (AES)*.

AES was to be the long-term replacement, but something else was needed to fill the gap before AES was ready to be deployed. Therefore, to extend the usefulness of the DES encryption standard, 3DES was adopted. It can use two or three keys to encrypt data, depending on how it is implemented. It has an effective key length of 112 or 168 bits, and it performs 48 rounds of transpositions and substitutions. Although it is much more secure, it is as slow as one-third the speed of 56-bit DES.

Rijndael

Rijndael is a block cipher adopted by NIST as the AES to replace DES. In 2002, NIST chose *Rijndael* to replace DES. Its name is derived from its two developers, Vincent Rijmen and Joan Daemen. It is a fast, simple, robust encryption mechanism. Rijndael is also known to resist various types of attacks.

Advanced Encryption Standard

The Rijndael algorithm uses three layers of transformations to encrypt and decrypt blocks of message text:

- Linear mix transform
- Nonlinear transform
- Key addition transform

Rijndael uses a four-step, parallel series of rounds. Rijndael is an iterated block cipher that supports variable key and block lengths of 128, 192, or 256 bits:

- If both key and block size are 128 bits, there are 10 rounds.
- If both key and block size are 192 bits, there are 12 rounds.
- If both key and block size are 256 bits, there are 14 rounds.

Each of the following steps is performed during each round:

1. **Byte substitution**: Each byte is replaced by an s-box substitution.
2. **Shift row**: Bytes are arranged in a rectangle and shifted.
3. **Mix column**: Matrix multiplication is performed based on the arranged rectangle.
4. **Add round key**: Each byte of the state is combined with the round key.

On the last round, the fourth step is bypassed and the first step is repeated.

International Data Encryption Algorithm

The *International Data Encryption Algorithm (IDEA)* is a 64-bit block cipher that uses a 128-bit key. It is different from others, as it avoids the use of s-boxes or lookup tables. Although IDEA is patented by a Swiss company, it is freely available for noncommercial use. It is considered a secure encryption standard, and there have been no known attacks against it. Like DES, it operates in four distinct modes. At one time, it was thought that IDEA might replace DES, but patent royalties made that impractical.

Rivest Cipher Algorithms

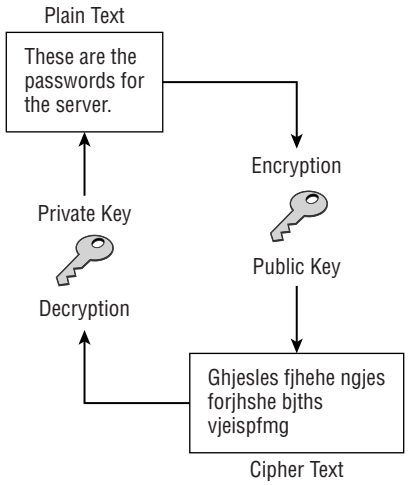
The *RC cipher algorithm series* is part of a family of ciphers designed by Ron Rivest. Rivest ciphers include RC2, RC3, RC4, RC5, and RC6. RC2 is an older algorithm that maintains a variable key size, 64-bit block cipher that can be used as a substitute for DES. RC3 was broken as a cipher as it was being developed. So, RC3 was never actually used. RC4 was implemented as a stream cipher. The 40-bit version is what was originally available in WEP. It is most commonly found in the 128-bit key version. RC5 is a block cipher in which the number of rounds can range from 0 to 255, and the key can range from 0 bits to 2,048 bits in size. Finally, there is RC6. It features variable key size and rounds, and it added two features not found in RC5: integer multiplication and 4-bit working registers.

Symmetric encryption does offer speed, but if you're looking for a cryptographic system that provides easy key exchange, you will have to consider asymmetric encryption.

Asymmetric Encryption

Asymmetric encryption, or *public key cryptography*, is different from symmetric encryption. It overcomes one of the big barriers of symmetric encryption: key distribution. Asymmetric encryption uses two unique keys, as shown in Figure 1.3. What one key does, the other key undoes.

FIGURE 1.3 Asymmetric encryption



Here's how asymmetric encryption works: Imagine that you want to send a coworker a message. You use your coworker's public key to encrypt the message. Your coworker receives the message and uses a private key to decrypt it.

Public key cryptography is made possible by the use of one-way functions. A *one-way function*, or trapdoor, is a math operation that is easy to compute in one direction, yet it is almost impossible to compute in the other direction. Depending on the type of asymmetric encryption used, this difficulty is based on either the discrete logarithm problem or the factoring of a large number into its prime factors. Although the math behind the encryption process is not needed to pass the CASP+ exam, in algebra, *discrete logarithms* are group-theoretic analogs of ordinary logarithms. For example, if you are given two large prime numbers, it is easy to multiply them. However, if you are given only their product, it is difficult or impossible to find the factors with today's processing power. Asymmetric systems may also make use of a *zero-knowledge proof*. This concept allows you to prove your knowledge without revealing the fact to a third party.

If the message is encrypted with the public key, only the matching private key will decrypt it. The private key is kept secret, whereas the public key can be given to anyone. If the algorithm is properly designed, it should not be possible for someone to easily deduce the private key of a pair if that person has only the public key.

Consider the following example of asymmetric encryption: Given the prime numbers 397 and 823, it is easy to multiply them together and get 326,731. However, if you

are given the number 326,731, it's quite difficult to extract the two prime numbers, 397 and 823. Anyone who knows the trapdoor can perform the function in both directions, but if you are lacking the trapdoor, you can perform the function in only one direction. Trapdoor functions can be used in the forward direction for encryption and signature verification, whereas the inverse direction is used for decryption and signature generation.

To help ensure your success on the CASP+ exam, Table 1.2 compares symmetric and asymmetric cryptographic systems.

TABLE 1.2 Attributes of symmetric and asymmetric encryption

Symmetric	Asymmetric
Confidentiality	Confidentiality, integrity, authentication, and non-repudiation
One single shared key	Two keys: public and private
Requires an out-of-band exchange	Useful for in-band exchange
Not scalable; too many keys needed	Scalable; works for e-commerce
Small key size and fast	Larger key size required, and slower to process
Useful for bulk encryption	Best for small amounts of data, digital signatures, digital envelopes, and digital certificates

Diffie–Hellman

Dr. W. Diffie and Dr. M.E. Hellman released the first public key-exchange protocol in 1976. They developed it specifically for key exchange and not for data encryption or digital signatures. The *Diffie–Hellman protocol* was designed to allow two users to exchange a secret key over an insecure channel without any prior communication. The protocol functions with two system parameters: p and g. Both parameters are public and can be used by all of the system's users. Parameter p is a prime number, and parameter g, which is usually called a *generator*, is an integer less than p that has the following property: For every number n between 1 and $p - 1$ inclusive, there is a power k of g such that $g^k = n \bmod p$. Diffie–Hellman is used in conjunction with several authentication methods, including the Internet Key Exchange (IKE) component of IPsec.

$$g^k = n \bmod p$$

Diffie–Hellman was groundbreaking in its ability to allow two parties to exchange encryption keys securely, but it is not without its problems. It is vulnerable to *man-in-the-middle (MitM)* attacks because the key exchange process does not authenticate the participants. You should use digital signatures to alleviate this vulnerability.

RSA

The *RSA algorithm* is named after its inventors. Ron Rivest, Adi Shamir, and Len Adleman developed RSA in 1977. Although RSA, like other asymmetric algorithms, is slower than symmetric encryption systems, it offers secure key exchange and is considered very secure. RSA supports a key size up to 3,072 bits. The design of RSA is such that it has to use prime numbers whose product is much larger than 129 digits for security; 129-digit decimal numbers are factored using a number field sieve algorithm. RSA public and private keys are generated as follows:

1. Choose two large prime numbers, p and q, of equal length and compute $p \times q = n$, which is the public modulus.
2. Choose a random public key, e, so that e and $(p - 1)(q - 1)$ are relatively prime.
3. Compute $e \times d = 1 \bmod [(p - 1)(q - 1)]$, where d is the private key.
4. Thus, $d = e^{-1} \bmod [(p - 1)(q - 1)]$.

From these calculations, (d, n) is the private key and (e, n) is the public key. The plain text, P, is encrypted to generate cipher text, C, as follows:

$$C = P^e \bmod n$$

and is decrypted to recover the plain text, P, as follows:

$$P = C^d \bmod n$$

RSA functions by breaking the plain text into equal-length blocks, with each block having fewer digits than n. Each block is encrypted and decrypted separately. Anyone attempting to crack RSA would be left with a tough challenge because of the difficulty of factoring a large integer into its two factors. Cracking an RSA key would require an extraordinary amount of computer processing power and time. The RSA algorithm has become the de facto standard for industrial-strength encryption, especially since the patent expired in 2000. It is built into many protocols, such as PGP; software products; and systems such as Mozilla Firefox, Google Chrome, and Microsoft Edge.

Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) can be found in smaller, less powerful devices such as smartphones and handheld devices. ECC is considered more secure than some of the other asymmetric algorithms because elliptic curve systems are harder to crack than those based on discrete log problems. Elliptic curves are usually defined over finite fields such as real and rational numbers, and they implement an analog to the discrete logarithm problem.

ElGamal

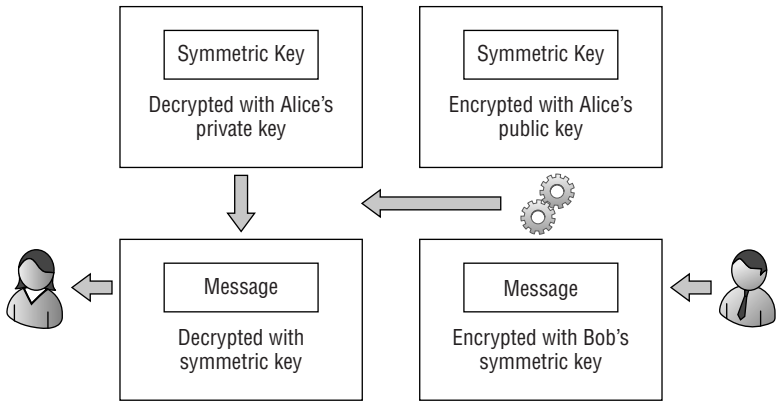
ElGamal was released in 1985, and its security rests in part on the difficulty of solving discrete logarithm problems. It is an extension of the Diffie–Hellman key exchange. ElGamal consists of three discrete components: a key generator, an encryption algorithm, and a decryption algorithm. It can be used for digital signatures, key exchange, and encryption.

Hybrid Encryption

Sometimes mixing two things together makes good sense. Do you remember the commercial, "You got your chocolate in my peanut butter?" While you may not consider cryptography as tasty as chocolate, there is a real benefit to combining both symmetric and asymmetric encryption. Symmetric encryption is fast, but key distribution is a problem. Asymmetric encryption offers easy key distribution, but it's not suited for large amounts of data. Combining the two into hybrid encryption uses the advantages of each and results in a truly powerful system. Public key cryptography is used as a key encapsulation scheme, and the private key cryptography is used as a data encapsulation scheme. Here is how the system works. If Bob wants to send a message to Alice, the following occurs:

1. Bob generates a random private key for a data encapsulation scheme. This session key is a symmetric key.
2. The data encapsulation happens when Bob encrypts the message using the symmetric key that was generated in step 1.
3. The key encapsulation happens when Bob encrypts the symmetric key using Alice's public key.
4. Bob sends both of these items, the encrypted message and the encrypted key, to Alice.
5. Alice uses her private key to decrypt the symmetric key and then uses the symmetric key to decrypt the message. This process is shown in Figure 1.4.

FIGURE 1.4 Hybrid encryption



Almost all modern cryptographic systems make use of hybrid encryption. This method works well because it uses the strength of symmetric encryption and the key exchange capabilities of asymmetric encryption. Some good examples of hybrid cryptographic

systems are IPsec, Secure Shell, Secure Electronic Transaction, Secure Sockets Layer, PGP, and Transport Layer Security. With hybrid systems, can we achieve perfect secrecy? This depends on items such as the algorithm, how the key is used, and how well keys are protected. The concept of *perfect forward secrecy (PFS)* refers to the goal of ensuring that the exposure of a single key will permit an attacker access only to data protected by a single key. To achieve PFS, the key used to protect transmission of data cannot be used to create any additional keys. Also, if the key being used to protect transmission of data is derived from some other keying material, that material cannot be used to create any additional keys.

Hashing

Hashing refers to a broad category of algorithms that are useful for their ability to provide integrity and authentication. Integrity ensures that the information remains unchanged and is in its true original form. Authentication provides the capability to ensure that messages were sent from those you believed sent them and that those messages are sent to their intended recipients.

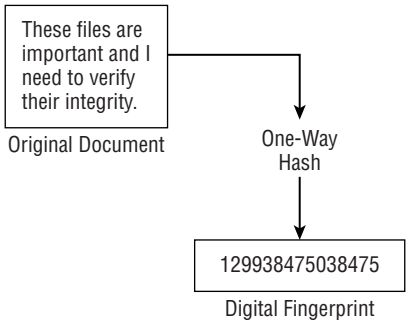
Hashing and Message Digests

Hashing algorithms operate by taking a variable amount of data and compressing it into a fixed-length value referred to as a *hash value*. Hashing provides a fingerprint or message digest of the data. A well-designed hashing algorithm will not typically produce the same hash value or output for two different inputs. When this does occur, it is referred to as a *collision*.

Collisions can be a problem in the world of hashing. A *collision* occurs when two different files create the same hashed output. One way to deal with collisions is to increase the size of the hashing algorithm output, such as, for example, moving from SHA 160 to SHA 256 so that a larger hash is created.

Hashing can be used to meet the goals of integrity and non-repudiation, depending on how the algorithm is implemented. Hashing is one of the primary means used to perform change monitoring. As an example, you might use a program such as Tripwire, a well-known change monitoring program, to verify that the contents of a specific folder remain unchanged over time. One of the advantages of hashing is its ability to verify that information has remained unchanged, but it is also used in authentication systems and digital signatures. Figure 1.5 gives an overview of the hashing process.

FIGURE 1.5 Hashing process



A *hash* is a one-way process and is not intended to be used to reproduce data. When a message or data file is hashed, the hashing algorithm examines every bit of the data while it is being processed. This means that if two files are close yet not exactly the same, their hashes will be different. For example, if we gave you a copy of a software program that had CASP+ exam study questions and you went to the Sybex website and downloaded the same software, hashing both files should result in the same value. An example of a cryptographic hash is shown in Figure 1.6. It can be seen in the sha256sum column.

FIGURE 1.6 An example of a cryptographic hash on a software product

Download Kali Linux Images

We generate fresh Kali Linux image files every few months, which we make available for download. This page provides the links to **download Kali Linux** in its latest official release. For a release history, check our Kali Linux Releases page. Please note: You can find unofficial, untested weekly releases at `http://cdimage.kali.org/kali-weekly/`.

Image Name	Download	Size	Version	sha256sum
Kali Linux 64 Bit	HTTP | Torrent	2.8G	2018.2	56f677e2edfb2efcd0b08662ddde824e254c3d53567ebbbcdbbf5c03efd9bc0f
Kali Linux Light 64 Bit	HTTP | Torrent	865M	2018.2	554f020b0c89d5978928d31b8635a7eeddf0a3900abcacdbc39616f80d247f86
Kali Linux E17 64 Bit	HTTP | Torrent	2.6G	2018.2	be0a858c4a1862eb5d7b8875852e7d38ef852c335c3c23852a8b08807b4c3be8
Kali Linux Lxde 64 Bit	HTTP | Torrent	2.6G	2018.2	449ecca86b0f49a52f95a51acdde94745821020b7fc0bd2129628c56bc2d145d
Kali Linux Xfce 64 Bit	HTTP | Torrent	2.6G	2018.2	0e94035a0a56fccc49961b0da56b9243ed3da6a3f8d696884e6f0b936f74dbfb

If there were even a slight change between the two files, the hashed values would be different. Comparing the hashes for the two files would indicate that the software we gave you had been altered. This same process is how programs such as Tripwire, MD5sum, and Windows System File Checker (`sfc.exe`) work. These kinds of programs can be used to monitor a file, a folder, or an entire hard drive for unauthorized changes. You also see this process used for functions such as code signing. *Code signing* is the process of digitally signing executables and scripts to confirm the software author. Code signing also guarantees that the code has not been altered or corrupted since it was signed by use of a hash. Listed here are some examples of hashing algorithms:

- Message Digest Algorithm (MD) series
- Secure Hash Algorithm (SHA) series
- HAVAL
- RIPEMD
- Tiger
- MAC
- HMAC

MD Series

The *MD algorithms* are a series of cryptographic algorithms that were developed by Ron Rivest. These have progressed throughout the years as technology has advanced. The first algorithm was MD2, which is considered outdated. One reason for its demise is that it was prone to collisions. MD4 was the next algorithm in the series. MD4 processes data in 512-bit blocks. As with MD2, MD4 was found to be subject to collisions and could potentially be vulnerable to forced collisions. These issues helped lead to the development of MD5, which processes a variable-size input and produces a fixed 128-bit output. A common implementation of MD5 is MD5sum. It's widely used to verify the integrity of a program or file. Consider the following example: If we received a copy of `snort.exe` from a friend, we could hash it and verify that the MD5sum matches what is found on the Sourcefire website:

```
C:\temp>md5sum snort.exe
d1bd4c6f099c4f0f26ea19e70f768d7f *snort.exe
```

Thus, a hash acts to prove the integrity of a file. Like MD4, MD5 processes the data in blocks of 512 bits. However, MD5 has also fallen from favor as it too has been shown to be vulnerable to collisions.

SHA

A *Secure Hash Algorithm (SHA)* is similar to MD5. Some consider it a successor to MD5 because it produces a larger cryptographic hash. SHA outputs a 160-bit message digest. SHA-1 processes messages in 512-bit blocks and adds padding, if needed, to get the data to add up to the right number of bits. SHA-1 has only 111-bit effectiveness. SHA-1 is part

of a family of SHA algorithms, including SHA-0, SHA-1, SHA-2, and SHA-3. SHA-0 is no longer considered secure, and SHA-1 is also now considered vulnerable to attacks. SHA-2 is, by the U.S. government, the recommended replacement for the collision-vulnerable MD5. Some of the strongest versions currently available are SHA-256 and SHA-512. SHA-3 was released in 2012 and uses the Keccak algorithm.

HAVAL

HAVAL is another example of a one-way hashing algorithm that is similar to MD5. Unlike MD5, HAVAL is not tied to a fixed message-digest value. HAVAL-3-128 makes three passes and outputs a 128-bit fingerprint, and HAVAL-4-256 makes four passes and produces a fingerprint that is 256 bits in length. In late 2004, it was determined that HAVAL-3-128 yielded collisions.

Message Authentication Code

A *message authentication code (MAC)* is similar to a digital signature except that it uses symmetric encryption. MACs are created and verified with the same secret (symmetric) key. There are four types of MACs that you may come across in your career as a security professional: unconditionally secure, hash function-based, stream cipher-based, and block cipher-based.

HMAC

Sometimes hashing by itself is not enough, and in such situations a *hashed message authentication code (HMAC)* may be needed. HMAC was designed to be immune to the multicollision attack. This functionality was added by including a shared secret key. Basically, HMAC functions by using a hashing algorithm such as MD5 or SHA-1 and then alters the initial state by adding a password. Even if someone can intercept and modify the data, it's of little use if that person does not possess the secret key. There is no easy way for the person to re-create the hashed value without it.

Digital Signatures

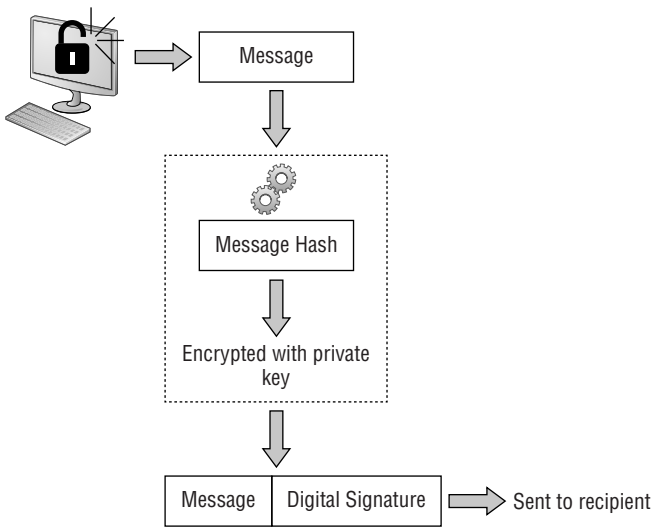
Digital signatures are a category of algorithms based on public key cryptography. They are used for verifying the authenticity and integrity of a message. To create a digital signature, the message is passed through a hashing algorithm. The resulting hashed value is then encrypted with the sender's private key. Upon receiving the message, the recipient decrypts the encrypted sum and then recalculates the expected message hash using the sender's public key. The values must match to prove the validity of the message and verify that it was sent by the party believed to have sent it. Digital signatures work because only that party has access to the private key. Let's break this process out step-by-step to help detail the operation:

1. Bob produces a message digest by passing a message through a hashing algorithm.
2. The message digest is then encrypted using Bob's private key.

3. The message is forwarded to the recipient, Alice.
4. Alice creates a message digest from the message with the same hashing algorithm that Bob used. Alice then decrypts Bob's signature digest by using Bob's public key.
5. Finally, Alice compares the two message digests: the one originally created by Bob and the other one that she created. If the two values match, Alice can rest assured that the message is unaltered.

Figure 1.7 illustrates the creation process. It shows how the hashing function ensures integrity and how the signing of the hash value provides authentication and non-repudiation.

FIGURE 1.7 Digital signature creation



The digital signature is hashed with the sender's private key. This helps prove that only the sender could have completed the signing process.

To help ensure your success on the CASP+ exam, integrity verification methods are reviewed in Table 1.3.

TABLE 1.3 Attributes of symmetric and asymmetric encryption

Method	Description
Parity	Simple error detection code
Hashing	Integrity

Method	Description
Digital signature	Integrity, authentication, and non-repudiation
Hashed MAC	Integrity and data origin authentication
CBC MAC	Integrity and data origin authentication
Checksum	Redundancy check, weak integrity

Digital signatures are typically used within the Digital Signature Standard. The Digital Signature Standard makes use of the Digital Signature Algorithm, and it also makes use of SHA-1 and public key encryption.

Public Key Infrastructure

Public key infrastructure (PKI) allows two parties to communicate even if they were previously unknown to each other. PKI makes use of users, systems, and applications. It allows users that are previously unknown to each other to communicate over an insecure medium such as the Internet. The most common system of using PKI is that of a centralized certificate authority. Applications that make use of PKI commonly use X.509 certificates.

PKI facilitates e-commerce. Consider how different dealing with brick-and-mortar businesses is from transactions over the Internet. Dealing with brick-and-mortar businesses gives you plenty of opportunity to develop trust. After all, you can see who you are dealing with, talk to the employees, and get a good look at how they do business.

In the modern world of e-commerce, transactions are much less transparent. You may not be able to see with whom you are dealing, yet you might have full trust in them. PKI addresses these concerns and brings trust, integrity, and security to electronic transactions.

One nontechnical issue with key distribution is controlling access to keys. Any PKI system has to be carefully controlled to ensure that the wrong individuals don't get access to secret keys.

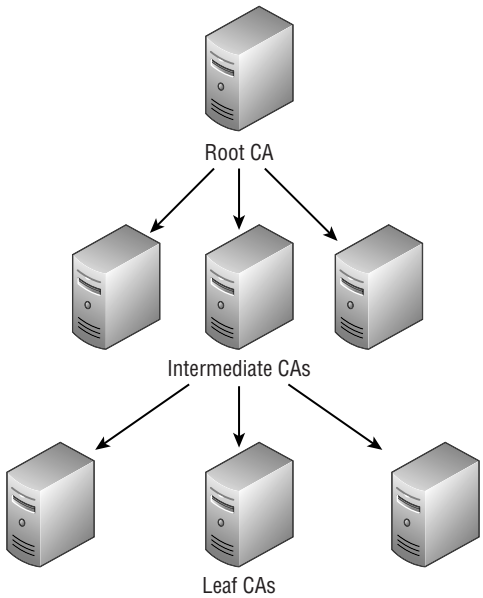
From a user's perspective, PKI may look seamless, yet in reality it is made up of many components. PKI consists of hardware, software, and policies that manage, create, store, and distribute keys and digital certificates. The basic components of PKI are as follows:

- The certificate authority (CA)
- The registration authority (RA)
- The certificate revocation list (CRL)
- Digital certificates
- A certificate distribution system

Certificate Authority

The *certificate authority (CA)* is like a passport office. The passport office is responsible for issuing passports, and passports are a standard for identification for anyone wanting to leave the country. Like passport offices, CAs vouch for your identity in a digital world. Verisign, Thawte, and Entrust are some of the companies that perform CA services. The most commonly used model is the *hierarchical trust model*. An example of this model is shown in Figure 1.8. In small organizations, a single trust model may be used. Its advantage is that it's not as complex and has less overhead than the hierarchical trust model.

FIGURE 1.8 Hierarchical trust model



Although the companies mentioned above are external CAs, some organizations may also decide to tackle these responsibilities by themselves. Regardless of who performs the services, the following steps are required:

1. The CA verifies the request for certificate with the help of the registration authority.
2. The individual's identification is validated.
3. A certificate is created by the CA, which verifies that the person matches the public key that is being offered.

Registration Authority

If the CA is like a passport authority, the *registration authority (RA)* is like a middleman. Think of it as one of the rush services that you can use when you need to get your passport

right away. The RA is positioned between the client and the CA. Although the RA cannot generate a certificate, it can accept requests, verify a person's identity, and pass along the information to the CA for certificate generation.

RAs play a key role when certificate services are expanded to cover large geographic areas. One central private or corporate CA can delegate its responsibilities to regional RAs; for example, there might be one RA in the United States, another in Canada, another in Europe, and another in India.

Certificate Revocation List

As with passports, digital certificates do not stay valid for a lifetime. Certificates become invalid for many reasons, such as someone leaving the company, information changing, or a private key being compromised. For these reasons, the *certificate revocation list (CRL)* must be maintained.

The CRL is maintained by the CA, which signs the list to maintain its accuracy. Whenever problems are reported with digital certificates, the digital certificates are considered invalid and the CA has the serial number added to the CRL. Anyone requesting a digital certificate can check the CRL to verify the certificate's integrity. There are many reasons a certificate may become corrupted, including the following:

- The certificate expired.
- The DNS name or the IP address of the server changed.
- The server crashed and corrupted the certificate.

Digital Certificates

Digital certificates are critical to the PKI process. The digital certificate serves two roles. First, it ensures the integrity of the public key and makes sure that the key remains unchanged and in a valid state. Second, it validates that the public key is tied to the stated owner and that all associated information is true and correct. The information needed to accomplish these goals is added to the digital certificate.

Digital signatures play a vital role in proving your identity when performing electronic transactions.

Digital certificates are formatted to the X.509 standard. The most current version of X.509 is version 3. One of the main developments in version 3 was the addition of extensions. This version includes the flexibility to support other topologies such as *bridges* and *meshes*. It can operate as a web of trust, much like PGP. An X.509 certificate includes the following elements:

- Version
- Serial number

- Algorithm ID
- Issuer
- Validity
- Not before (a specified date)
- Not after (a specified date)
- Subject
- Subject public key information
- Public key algorithm
- Subject public key
- Issuer-unique identifier (optional)
- Subject-unique identifier (optional)
- Extensions (optional)

Figure 1.9 shows some of these elements.

FIGURE 1.9 An example of an X.509 certificate

Different entities can use a certificate. *Issuance to entities* identifies to whom the CA issues certificates. The certificate might be issued to a user, a system, or an application. The CA not only issues the certificate, but it also vouches for the authenticity of entities. It is not mandatory that you use an external CA to issue certificates, they are widely used. An organization may decide to have itself act as a CA. Regardless of whether a third party handles the duties or your company performs them, digital certificates will typically contain the following critical pieces of information:

- Identification information that includes username, serial number, and validity dates of the certificates
- The public key of the certificate holder
- The digital signature of the signature authority. This piece is critical since it validates the entire package.

If you decide to use a third party to issue a certificate, there is a cost. These organizations are generally for-profit and will charge fees for you to maintain your certificate in good standing. Some organizations may choose to use wildcard certificates to cut costs. A *wildcard certificate* allows the purchaser to secure an unlimited number of subdomain certificates on a domain name. The advantage is that you buy and maintain only one certificate. The drawback, however, is that you are using just one certificate and private key on multiple websites and private servers. If just one of these servers or websites is compromised, all of the others under the wildcard certificate will be exposed.

Wildcard certificates allow you to specify a wildcard character in the name. For example, a wildcard certificate for `*.thesolutionfirm.com` will allow you to use `mail.thesolutionfirm.com`, `ftp.thesolutionfirm.com`, `mail.china.thesolutionfirm.com`, and so on.

If a private key is exposed or another situation arises where a certificate must be revoked, PKI has a way to deal with such situations; that is, when a CRL is used. These lists can be checked via the *Online Certificate Status Protocol (OCSP)*, an Internet protocol used for obtaining the revocation status of an X.509 digital certificate. This process is much the same as maintaining a driver's license. Mike may have a driver's license, yet if he gets stopped by a police officer, the officer may still decide to run a check on Mike's license; he's checking on the status of Mike's license in the same way that the OCSP is used to check on the status of an X.509 certificate.

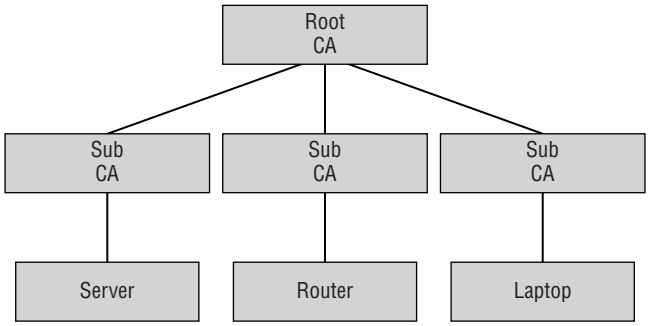
If the topic of OCSP and certificates interests you, be sure to check out Request for Comments (RFC) 6960. This RFC details CRL and OCSP.

Certificate Distribution

Certificates can be distributed by a centralized service or by means of a public authority. The use of a CA is an example of centralized distribution: A trusted CA distributes a public

key to another party. The certificate is signed by means of a digital signature of the CA to prove it is valid. The certificates can be passed from one CA to another by using a chain of trust. A chain of trust provides a trust relationship between each entity. See Figure 1.10 for an example.

FIGURE 1.10 An example of a chain of trust



A second way to distribute keys is directly through a third party. This is called a *web of trust*. For example, if you email us with a question about the book, our return emails will include our public key. It's an easy way to distribute keys, but it does not offer the level of trust that would be obtained from a third-party CA such as Verisign or Thawte. PGP and GPG are examples of systems that provide encryption and can use web-of-trust certificate distribution.

Tokens PKI tokens provide secure storage for digital certificates and private keys. They allow public-key cryptography and digital signatures to be leveraged securely without the risk of leaking the private key information. PKI tokens are hardware devices that store digital certificates and private keys securely. When you need to encrypt, decrypt, or sign something, the token does this internally in a secure chip, meaning that the keys are never at risk of being stolen.

Stapling *Online Certificate Status Protocol (OCSP) stapling* enhances performance of the website and privacy of the client. Prior to OCSP stapling, the OCSP request originates from a client to the CA server in order to validate an SSL certificate. OCSP stapling allows the certificate presenter to query the OCSP responder directly and then let them cache the response. This allows for a securely cached response, which is then delivered ("stapled") with a TLS/SSL handshake via the Certificate Status Request extension response. This helps to ensure that the browser gets the same response performance for the certificate status as it does for the website content.

OCSP stapling also addresses a privacy concern with OCSP that the certificate authority no longer receives the revocation requests directly from the client (browser). OCSP stapling further addresses the concerns about OCSP SSL negotiation delays by removing the need for a separate network connection to a certification authority's responder.

Pinning *Pinning* is the technique of telling your browser of choice that only certificates with a specific public key somewhere in the certificate chain are to be trusted. Current implementations are based on *Trust on First Use (TOFU)*, which means that your browser of choice will have to trust the connection the first time you use a site. This is because the pinning info is sent via an HTTP header by the web server. In the future, this can hopefully be retrieved via DNSSEC-signed DNS records.

You pin the public key of a certificate. There are at least three certificates per site (site cert, intermediate cert, and root cert) that give you a few options on what to pin. Your site cert will be replaced in the near future, commonly within three years, but it will most likely be replaced even before that due to reissues. You have no control over what intermediate cert the CA will use on a reissue, so don't pin that. Certificate authorities have multiple root certs in trust stores, and you simply can't control which one they will use on reissues. You must rely on what you can control—your own public key in the certificate chain—so that is what you want to pin.

HTTP Public Key Pinning (HPKP) doesn't just let you provide a backup public key, it requires you to do so. This is useful when your private key gets compromised. Your backup key will then be used to generate a new certificate. The backup key should, of course, never be stored on the same infrastructure as your primary key, and it will never be used for anything else.

The Client's Role in PKI

Although the CA is responsible for a large portion of the work, in the world of PKI the client also has some duties. Clients are responsible for requesting digital certificates and for maintaining the security of their private key. Loss, compromise, or exposure of the private key would mean that communications are no longer secure. Protecting the private key is an important issue because for the attacker, it may be easier to target the key rather than to try to brute-force or crack the certificate service. Organizations should concern themselves with eight key management issues:

- Generation
- Distribution
- Installation
- Storage
- Recovery
- Change
- Control
- Disposal

Key recovery and control is an important issue that must be addressed. One basic recovery and control method is the *m* of *n* control method of access. This method is designed to ensure that no one person can have total control; it is closely related to dual control. If *n* administrators have the ability to perform a process, *m* of those administrators must authenticate for access to occur. *M* of *n* control should require physical presence for access.

Here is an example: Let's say that a typical *m* of *n* control method requires that four people have access to the archive server and that at least two of them must be present to accomplish access. In this situation, $m = 2$ and $n = 4$. This would ensure that no one person could compromise the security system or gain access.

Real World Scenario

Trust in the World of PKI

Trust isn't a problem in small organizations, but the need to communicate within large organizations or with external clients and third parties requires developing a working trust model. Organizations typically follow one of several well-known trust models, such as single-authority trust, hierarchical trust, or web of trust.

Each model has its advantages and disadvantages, and as a CASP+ you may be asked to recommend a method to your organization. You should keep in mind that although a single authority model is simple, it's not well suited for large organizations; if it is managed by the company, cross-certification to other entities can be an issue. A hierarchical model is typically provided by a commercial entity. While much more robust, there are associated ongoing fees.

Finally, there is the web of trust, the least complex of all models. It may work well for an individual or small groups, but it has a low level of trust. Which model will you choose for your company?

Implementation of Cryptographic Solutions

Has this chapter got you thinking about all of the ways cryptography can be used and how valuable it is to a security professional? We hope that it has. The real question is, now that you're armed with some specific cryptographic solutions, how strong should the encryption be and where might you apply cryptographic solutions?

Encryption can be applied at the disk, block, file, record, and port:

Disk Encryption *Disk encryption* can use either hardware or software to encrypt an entire hard drive or volume. Such technology is incredibly important today. Just consider how much sensitive information individuals have stored on mobile devices and tablets. Such items are easily lost or stolen. Common disk encryption products include BitLocker and AxCrypt.

Block Encryption *Block encryption* secures data in fixed-size groups of bits. An example of a block cipher that we have previously discussed is DES ECB. DES encrypts data in 64-bit blocks.

File Encryption You don't have to encrypt an entire hard drive or volume. In some situations, you may simply need to encrypt specific files. Examples of products that can be used for *file encryption* include AxCrypt and PGP.

Record Encryption Databases are a common area of attack. If you are storing sensitive information in a database, you may want to encrypt the entire database or just specific records. As an example of *record encryption*, in a medical facility you may want to protect records that hold social security numbers or other personal information, leaving only medical IDs and medical records open to the hospital staff.

Port Encryption Some services are just more secure than others. As an example of *port encryption*, Telnet, TCP port 23, sends data in the clear whereas Secure Shell, port 22, uses encryption. Another example is HTTP, as port 80 is clear text whereas HTTPS uses port 443.

These examples demonstrate that cryptography is one of the most valuable tools that a security professional can use, but the trade-offs between strength, performance, and usability must be considered. Each cryptographic solution has strengths and limitations. Organizations must perform a proper risk assessment to determine the level of threat and the amount of protection that each asset requires. That assessment will go a long way in determining the type of technology used. Is the data something that is only useful for the next few minutes, like orders to buy or sell stock? Is the information top secret data on the next generation of fighter jets that have yet to start production? Where is the data being stored? How valuable is it to someone else? How long is it likely to remain valuable?

Even if the information does not require cryptographic solutions to provide privacy, you may still need controls that can help safeguard the information. One such technology is *digital rights management (DRM)*. DRM is an entire suite of technology designed to protect digital content. As an example, you may be reading a copy of this book on your tablet, yet that does not mean the publisher wants to provide free copies to one hundred of your closest friends! That is the situation for which DRM is designed: It helps prevent copyright infringement online and thus helps the copyright holder maintain control of the information.

Next, you need to consider where to build in the protection. Cryptography can be used in many different situations to build a true defense in depth. If you think of cryptography in reference to the TCP/IP model, you can see where cryptographic solutions can be applied, from the application layer all the way down to the physical frame. Let's start at the top of the TCP/IP stack and work down through the layers, highlighting a few cryptographic solutions, and then look at the concept of steganography.

Application Layer Encryption

The following application layer protocols are just a few examples that can be used to add confidentiality, integrity, or non-repudiation:

Secure Shell *Secure Shell (SSH)* is an Internet application that provides secure remote access. It serves as a replacement for FTP, Telnet, and the Berkeley "r" utilities. SSH defaults to TCP port 22.

Secure Hypertext Transfer Protocol *Secure Hypertext Transfer Protocol (S-HTTP)* is a superset of HTTP that was developed to provide secure communication with a web server. S-HTTP is a connectionless protocol designed to send individual messages securely.

Pretty Good Privacy *Pretty Good Privacy (PGP)* was developed in 1991 by Phil Zimmermann to provide privacy and authentication. Over time, it evolved into open standards such as OpenPGP and GnuPGP. PGP builds a web of trust that is developed as users sign and issue their own keys. The goal of PGP was for it to become the "everyman's encryption." Popular programs such as HushMail, CounterMail, and K-9 Mail are based on PGP, providing end-to-end encryption.

GNU Privacy Guard Does free sound good? If you are like many of us, the answer is yes, and that is where *GNU Privacy Guard (GPG)* comes into the equation. It is a licensed, free version of PGP. The idea was to provide a free version of PGP that everyone can use. Like PGP, GPG makes use of hybrid encryption and uses the best of both symmetric and asymmetric encryption. The symmetric portion is used for encryption and decryption, and the asymmetric portion is used for key exchange.

S/MIME For those who prefer not to use PGP or GPG, there is another option for the security of email. That solution is *S/MIME (Secure/Multipurpose Internet Mail Extensions)*. S/MIME is a standard for public key encryption and signing of MIME data. S/MIME provides two basic services: digital signatures and message encryption. S/MIME is a popular solution for securing email, and it is built into most email software programs, such as Microsoft Outlook and Mozilla Thunderbird.

Secure Remote Access A variety of applications can be used for *secure remote access*, such as SSH, Remote Desktop Protocol (RDP), and Virtual Network Computing (VNC). RDP is a proprietary protocol developed by Microsoft. It provides the remote user with a graphical interface to the remote computer. VNC is like RDP in that it allows graphic access to a remote computer. VNC makes use of the Remote Frame Buffer (RFB) protocol to control another computer remotely.

Remote technologies are a concept emphasized on the exam because so much of today's access is remote and many times it is over an open network such as the Internet.

Transport Layer Encryption

The transport layer of the TCP/IP stack can also be used to add cryptographic solutions to data communications. Some common examples follow:

Secure Sockets Layer Netscape developed *Secure Sockets Layer (SSL)* for transmitting private documents over the Internet. SSL is application independent and cryptographically independent since the protocol itself is merely a framework for communicating certificates, encrypted keys, and data.

Transport Layer Security *Transport Layer Security (TLS)* encrypts the communication between a host and a client. TLS consists of two layers: the Record Protocol and the TLS Handshake Protocol. Although TLS and SSL are functionally different, they provide the same services, and the terms are sometimes used interchangeably.

Wireless Transport Layer Security *Wireless Transport Layer Security (WTLS)* encrypts the communication between a wireless host and a client. WTLS is a security protocol, and it is part of the Wireless Application Protocol (WAP) stack. WTLS was developed to address the problems, specifically the relatively low bandwidth and processing power, of mobile network devices. These issues will become increasingly important in the next few years as mobile banking is being widely used on smartphones.

Transport layer encryption is not the same as transport encryption. The latter is associated with IPsec.

Internet Layer Controls

The *Internet layer* is home to IPsec, a well-known cryptographic solution. IPsec was developed to address the shortcomings of IPv4. It is an add-on for IPv4. IPsec can be used to encrypt just the data or the data and the header. With the depletion of IPv4 addresses, look for more attention to be paid to IPsec as it is built into IPv6. IPsec includes the following components:

Encapsulated Secure Payload *Encapsulated Secure Payload (ESP)* provides confidentiality by encrypting the data packet. The encrypted data is hidden, so its confidentiality is ensured.

Authentication Header The *Authentication Header (AH)* provides integrity and authentication. The AH uses a hashing algorithm and symmetric key to calculate a message authentication code. This message authentication code is known as the *integrity check value (ICV)*. When the AH is received, an ICV is calculated and checked against the received value to verify integrity.

Security Association For AH and ESP to work, some information must be exchanged to set up the secure session. This job is the responsibility of the *Security Association (SA)*. The SA is a one-way connection between the two parties. If both AH and ESP are used, a total of four connections are required. SAs use a symmetric key to encrypt communication. The Diffie–Hellman algorithm is used to generate this shared key.

Transport and Tunnel Mode AH and ESP can work in one of two modes: transport mode or tunnel mode. *Transport mode* encrypts the data that is sent between peers. *Tunnel mode* encapsulates the entire packet and adds a new IP header. Tunnel mode is widely used with VPNs. The AH and the ESP can be used together or independently of each other.

Physical Layer Controls

Now we have worked our way down to the bottom of the TCP/IP stack. As you've learned, there are many places to encrypt data. Encryption can happen at any one of many different layers. The question a CASP+ must ask is, "What is actually getting encrypted?" Is the data itself secured or the data and all headers? The following list includes some physical layer security solutions:

Full Disk Encryption As previously discussed, disk encryption is a useful tool for the security professional. *Full disk encryption* offers an easy way to protect information should equipment be lost, stolen, or accessed by unauthorized individuals. Some examples of full disk encryption include Microsoft BitLocker, Apple FileVault, and McAfee endpoint encryption. The real benefit of these programs is that everything on the drive is encrypted, including files, directories, and swap space. Full disk encryption can be used in conjunction with technologies such as the Trusted Platform Module (TPM), a protection feature designed to be added as a microchip on the motherboard of a computer. TPM acts as a secure cryptoprocessor, and it can store cryptographic keys that protect information.

Hardware Security Module Many organizations use *Hardware Security Modules (HSMs)* to store and retrieve escrowed keys securely. *Escrowed keys* allow another trusted party to hold a copy of a key. They need to be managed at the same security level as the original key. HSM systems can be used to protect enterprise storage and data, and they can detect and prevent tampering by destroying the key material if unauthorized access is detected.

Password Authentication Protocol We have included *Password Authentication Protocol (PAP)* here, but it should not be used. It is weak at best. PAP is not secure because the username and password are transmitted in clear text.

Challenge Handshake Authentication Protocol *Challenge Handshake Authentication Protocol (CHAP)* is a more suitable option than PAP because it sends the client a random value that is used only once. Both the client and the server know the predefined secret password. The client uses the random value, nonce, and the secret password, and it calculates a one-way hash. The handshake process for CHAP is as follows:

1. The user sends a logon request from the client to the server.
2. The server sends a challenge back to the client.
3. The challenge is encrypted and then sent back to the server.
4. The server compares the value from the client and, if the information matches, grants authorization.

Point-to-Point Tunneling Protocol *Point-to-Point Tunneling Protocol (PPTP)* consists of two components: the transport that maintains the virtual connection and the encryption that ensures confidentiality. It can operate at a 40-bit or a 128-bit length.

Layer 2 Tunneling Protocol *Layer 2 Tunneling Protocol (L2TP)* was created by Cisco and Microsoft to replace Layer 2 Forwarding (L2F) and PPTP. L2TP merged the capabilities of both L2F and PPTP into one tunneling protocol.

The Internet Assigned Number Authority (IANA) maintains a list of security protocols and their relationship to TCP/IP at: `https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml`.

Cryptocurrency

Cryptocurrency is the medium of exchange similar to the United States dollar (USD) but designed for the purpose of exchanging digital information through a process made possible by certain principles of cryptography. Cryptocurrency uses cryptography to secure the digital transactions along with the ability to control the creations of new cryptocurrency coins. Cryptocurrency was first introduced as *Bitcoin* in 2009. Currently, there are hundreds of other known alternative cryptocurrencies, which are referred to as digital *altcoins*.

Cryptocurrency security is broken into two parts. The first part comes from the difficulty in finding the hash set intersection. This task is conducted by miners. The second and most common of the two cases is known as the 51% attack. Within this scenario, a miner has the power to mine more than 51% of the network. The miner can take control of the global blockchain ledger (see the explanation of blockchain in the next section) along with generating an alternative blockchain. At this point, the attacker is limited by the capabilities of the attacker's limited availability or resources. The attack can also reverse the attacker's own transaction or even block other transactions.

The advantage of cryptocurrencies is that they are less susceptible to seizure by law enforcement or even the transaction holds placed on them from acquirers such as PayPal. Cryptocurrencies are all pseudo-anonymous, and some coins have added features to assist in creating true anonymity.

Blockchain

Blockchain is a digitized, decentralized, publicly used ledger of all cryptocurrency transactions. Blockchains are always growing as completed blocks, which is the most recent transaction. They are then recorded and added to the block in chronological order, allowing the market participants to keep track of all digital currency transactions without centralized recordkeeping. Each node gets a copy of the blockchain, and these copies are downloaded automatically.

Blockchain was originally developed as the accounting method for the well-known virtual currency *Bitcoin*. Blockchain is a form of *distributed ledger technology (DLT)*. Think of a DLT as one database spread across multiple locations, and blockchain is one implementation of a DLT. DLTs appear in a variety of commercial sectors at Oracle along with applications within today's market. The technology is primarily used to verify transactions within digital currencies. It is possible to digitize, code, and insert any document into the blockchain. This allows for an indelible record that cannot be altered. The record's authenticity can also be verified by the entire community using the blockchain instead of a single centralized authority.

Blockchain is the main technological innovation of Bitcoin. Bitcoin is not regulated by a specific central authority. Users of blockchain can validate transactions when one person pays another for good or services. This helps eliminate the need for a third party to process or store payments. The newly completed transaction is then publicly recorded within blocks and eventually into the blockchain. Then it is verified and relayed by the other Bitcoin users. On average, a new block is appended to the blockchain every 10 minutes through a process that is called *mining*.

Steganography

Steganography is the science of hiding information in plain sight. It is similar to cryptography in that it can be used to achieve confidentiality. Although both are intended to protect information, they achieve this goal in different ways. With cryptography, you can see the information but you should not be able to discern what is there. As an example, you might sniff an SSL connection between a client and a server, but the data would be scrambled and of little use. However, with steganography the data is hidden. That is why steganography is described as the practice of concealing information within a container such as a message, image, or file.

Steganography requires two items to be hidden: the container and the data. The *container* is the medium into which you will embed the data. Choosing an appropriate container is an important decision, because it is a large part of what determines the effectiveness of the steganographic technique. Therefore, a sender might use an image file of the CASP+ logo and adjust the color by changing the *least significant bit (LSB)* of each byte of data. Such a change would be so subtle that someone who is not specifically looking for it would be unlikely to notice it. Using a graphic is one of the most popular ways to hide data, but other media files offer an advantage. The following containers are common:

- Images
- Audio files
- Video files
- Office documents

Obviously, audio and video files offer more storage than most images simply because the file size can be much larger. A video file of 800 MB is not uncommon. Changing only a fraction of the LSB in an 800 MB file allows for embedding several MB of data. Often, stego tools will tell you the capacity for the to-be-embedded file. The variety of websites on which to post images and other media make it an easy task to exfiltrate data. Finding or recovering steganographic data is known as *steganalysis*. Some common steganalysis techniques are listed here:

Stego-Only Only the steganographic content is available for analysis.

Known-Stego Both the original and the steganographic content are available for review.

Known-Message The hidden message and the corresponding steganographic image are known.

Disabling or Active Analysis During the communication process, active attackers change the cover.

To make things even more difficult, the person using steganography might not only hide the data but they may also encrypt it. In these situations, recovery of the data can be quite difficult because you must not only find the data but also crack the encryption routine.

Watermarking

Although the term *steganography* is typically used to describe illicit activities, *watermarking* is used for legal purposes. It is typically used to identify ownership or the copyright of material such as videos or images. If any of these items are copies, the digital copy would be the same as the original; therefore, watermarking is a passive protection tool. It flags the data's ownership but does not degrade it in any way. It is an example of digital rights management.

Cryptographic Attacks

As long as there have been secrets, there have been people trying to uncover them. Attacks on cryptographic systems are nothing new. The formal name for this activity is *cryptanalysis*, which is the study of analyzing cryptography and attempting to determine the key value of a cryptographic system. Depending on which key is targeted by the attacker, it's possible that success may mean that someone could gain access to confidential information or pretend to be an authenticated party to a communication.

There are many ways an attacker can target a system for attack, such as a brute-force attack. A brute-force attack tries all possible solutions. One technique to make the attacker work longer and harder to perform a brute-force attack is *key stretching*. Key stretching refers to cryptographic techniques used to make a possibly weak cryptographic system, such as password generation, more secure. This technique hashes a password along with a random value known as a *salt*. This process can be repeated many times to produce a derived key. Typically, this might be a thousand or more iterations. This approach makes brute-force attacks time-consuming for an attacker. However, remember the previous discussion about the trade-off between strength, performance, and usability? Although it is more secure, the increased number of iterations will require more CPU power and time.

Key stretching refers to cryptographic techniques used to make a brute-force attack slower and harder for the attacker to recover information such as passwords.

Many countries seek to control cryptographic algorithms and place controls on their use. These controls fall under the Wassenaar Arrangement on Export Controls for Conventional

Arms and Dual-Use Goods and Technologies (see www.wassenaar.org). The goal of the agreement is to promote transparency in transfers of conventional arms and dual-use goods and technologies while also promoting greater responsibility in such transfers. The idea is to keep strong cryptography out of the hands of criminals and terrorists.

Real World Scenario

How Strong Is Your Password?

As a security administrator, you've no doubt heard many stories about how some people do very little to protect their passwords. Sometimes, people write their passwords down on sticky notes, place them under their keyboards, or even leave them on a scrap of paper taped to the monitor. As a security professional, you should not only help formulate good password policy but also help users understand why and how to protect passwords. One solution might be to offer password manager programs that can be used to secure passwords. Another approach is migration to biometric or token-based authentication systems.

For this scenario, you'll need to put yourself in the position of an attacker wanting to examine the strength of your password. From this perspective, you will test passwords with the following attributes:

- Create a password that is seven lowercase characters.
- Create a password that is seven upper- and lowercase characters.
- Create a password that is 14 upper- and lowercase characters and that includes at least one special character.

Submit each of the examples on the following site, and test the strength. What are your conclusions?: www.passwordmeter.com.

Summary

This chapter focused on cryptography. Cryptography is one of the most powerful tools of a security professional. It offers you the ability to protect sensitive information through the use of encryption. It can also offer you the ability to verify the integrity of patches, files, and important data. In addition, cryptography makes e-commerce possible. With cryptographic solutions such as PKI, you can trust that a third party is who they claim to be. These are but a few of the solutions cryptography offers.

As a security professional, you need to able to communicate with others about cryptographic solutions and services. You don't have to be able to write your own cryptographic algorithm. You do need to be able to offer solutions to real problems.

There is not a week that goes by without a news report that lists stolen or lost media that contained personal information. As a security professional, you may be in a position to suggest that your company use full disk encryption for all laptops. You may also have the opportunity to promote PGP as a standard to encrypt all email being used to discuss sensitive business matters. You may even be on a team preparing to roll out a new e-commerce site and be asked to offer your opinion on PKI. These are the types of solutions that security professionals offer every day.

Exam Essentials

Be able to describe which cryptographic solution is appropriate for a given situation. Cryptographic solutions can be broadly divided into symmetric encryption, asymmetric encryption, hybrid encryption, and hashing. Each offers specific solutions such as privacy, authentication, integrity, and non-repudiation.

Be able to describe the basic operation of PKI and understand advanced PKI concepts. PKI allows two parties that are previously unknown to each other to communicate over an insecure public network. Such communications can then be used to exchange data securely and privately or for e-commerce. PKI systems make use of public and private keys. Keys are shared through a trusted certificate authority.

Know what terms such as *wildcard* mean when applied to PKI. A wildcard certificate allows the purchaser to secure an unlimited number of subdomain certificates on a domain name. The advantage is that you buy and maintain only one certificate. The drawback, however, is that you are using just one certificate and private key on multiple websites and private servers.

Be able to describe transport encryption. Transport encryption is one of the two modes in which IPsec can operate. When IPsec transport encryption is used, only the data portion or payload of each IP packet is encrypted. This leaves the IP header untouched and sent in the clear.

Be able to describe a digital signature. A digital signature is a hash value that has been encrypted with the private key of the sender. It is used for authentication and integrity.

Be able to describe hashing. Hashing refers to a broad category of algorithms that are useful for their ability to provide integrity and authentication. Hashing algorithms operate by taking a variable amount of data and compressing it into a fixed-length value referred to as a hash value.

Be able to describe code signing. Code signing is the process of digitally signing executables and scripts to confirm the software author. Code signing also guarantees that the code has not been altered or corrupted since it was signed by use of a hash.

Know how non-repudiation works. Non-repudiation is the ability to verify proof of identity. It is used to ensure that a sender of data is provided with proof of delivery and that the recipient is assured of the sender's identity.

Be able to define the concept of pseudorandom number generation. Pseudorandom number generators are algorithms that generate a sequence of numbers that approximates the properties of random numbers.

Be able to explain perfect forward secrecy. Perfect forward secrecy is based on the concept that the exposure of a single key will permit an attacker access only to data protected by a single key.

Know the purpose and use of steganography. Steganography is a form of hidden writing. Steganography allows the user to hide a message inside a container. Common containers include images, music files, videos, and even Microsoft Office documents.

Define the terms *confusion* and *diffusion*. *Confusion* occurs when the relationship between the plain text and the key is so complicated that an attacker can't alter the plain text and determine the key. *Diffusion* is the process that occurs when a change in the plain text results in multiple changes spread throughout the cipher text.

Review Questions

You can find the answers in Appendix A.

1. You have been asked by a member of senior management to explain the importance of encryption and define what symmetric encryption offers. Which of the following offers the best explanation?
   - **A.** Non-repudiation
   - **B.** Confidentiality
   - **C.** Hashing
   - **D.** Privacy and authentication

2. As the security administrator for your organization, you must be aware of all types of hashing algorithms. Which algorithm was developed by Ron Rivest and offers a 128-bit output?
   - **A.** AES
   - **B.** DES
   - **C.** MD5
   - **D.** RC4

3. A coworker is concerned about the veracity of a claim because the sender of an email denies sending it. The coworker wants a way to prove the authenticity of an email. Which would you recommend?
   - **A.** Hashing
   - **B.** Digital signature
   - **C.** Symmetric encryption
   - **D.** Asymmetric encryption

4. A junior administrator at a sister company called to report a possible exposed private key that is used for PKI transactions. The administrator would like to know the easiest way to check whether the lost key has been flagged by the system. What are you going to recommend to the administrator?
   - **A.** Hashing
   - **B.** Issuance to entities
   - **C.** Online Certificate Status Protocol
   - **D.** Wildcard verification

5. You've discovered that an expired certificate is being used repeatedly to gain logon privileges. To what list should the certificate have been added?
   - **A.** Wildcard verification
   - **B.** Expired key revocation list
   - **C.** Online Certificate Status Protocol
   - **D.** Certificate revocation list (CRL)

6. A junior administrator comes to you in a panic after seeing the cost for certificates. She would like to know if there is a way to get one certificate to cover all domains and subdomains for the organization. What solution can you offer?
   - **A.** Wildcards
   - **B.** Blanket certificates
   - **C.** Distributed certificates
   - **D.** No such solution exists

7. Which of the following is not an advantage of symmetric encryption?
   - **A.** It's powerful.
   - **B.** A small key works well for bulk encryption.
   - **C.** It offers confidentiality.
   - **D.** Key exchange is easy.

8. Most authentication systems make use of a one-way encryption process. Which of the following best offers an example of one-way encryption?
   - **A.** Asymmetric encryption
   - **B.** Symmetric encryption
   - **C.** Hashing
   - **D.** PKI

9. Which of the following is an early form of encryption also known as ROT3?
   - **A.** Transposition cipher
   - **B.** Substitution cipher
   - **C.** Scytale
   - **D.** Caesar's cipher

10. Which type of encryption best offers easy key exchange and key management?
    - **A.** Symmetric
    - **B.** Asymmetric
    - **C.** Hashing
    - **D.** Digital signatures

11. SSL and TLS can best be categorized as which of the following?
    - **A.** Symmetric encryption systems
    - **B.** Asymmetric encryption systems
    - **C.** Hashing systems
    - **D.** Hybrid encryption systems

12. You're explaining the basics of cryptography to management in an attempt to obtain an increase in the budget. Which of the following is not symmetric encryption?
    A. DES
    B. RSA
    C. Blowfish
    D. Twofish

13. Which of the following is not a hashing algorithm?
    A. SHA
    B. HAVAL
    C. MD5
    D. IDEA

14. A mobile user calls you from the road and informs you that he has been asked to travel to China on business. He wants suggestions for securing his hard drive. What do you recommend he use?
    A. S/MIME
    B. BitLocker
    C. Secure SMTP
    D. PKI

15. You were given a disk full of applications by a friend but are unsure about installing a couple of the applications on your company laptop. Is there an easy way to verify if the programs are original or if they have been tampered with?
    A. Verify with a hashing algorithm.
    B. Submit to a certificate authority.
    C. Scan with symmetric encryption.
    D. Check the programs against the CRL.

16. What is the correct term for when two different files are hashed and produce the same hashed output?
    A. Session key
    B. Digital signature
    C. Message digest
    D. Collision

17. You have been asked to suggest a simple trust system for distribution of encryption keys. Your client is a three-person company and wants a low-cost or free solution. Which of the following would you suggest?
    A. Single authority trust
    B. Hierarchical trust
    C. Spoke/hub trust
    D. Web of trust

18. Which of the following would properly describe a system that uses a symmetric key distributed by an asymmetric process?

A. Digital signature

B. Hybrid encryption

C. HMAC

D. Message digest

19. A CASP+ must understand the importance of encryption and cryptography. It is one of the key concepts used for the protection of data in transit, while being processed, or while at rest. With that in mind, DES ECB is an example of which of the following?

A. Disk encryption

B. Block encryption

C. Port encryption

D. Record encryption

20. Which of the following can be used to describe a physical security component that is used for cryptoprocessing and can be used to store digital keys securely?

A. HSM

B. TPM

C. HMAC

D. OCSP