CHAPTER

Domain 3: Cryptography

4

EXAM OBJECTIVES IN THIS CHAPTER

- Cornerstone Cryptographic Concepts
- History of Cryptography
- Symmetric Encryption
- Asymmetric Encryption
- Hash Functions
- Cryptographic Attacks
- Implementing Cryptography

UNIQUE TERMS AND DEFINITIONS

- Plaintext—an unencrypted message
- Ciphertext—an encrypted message
- Cryptology—the science of secure communications
- Symmetric Encryption—encryption that uses one key to encrypt and decrypt
- Asymmetric Encryption—encryption that uses two keys: if you encrypt with one you may decrypt with the other
- Hash Function—one-way encryption using an algorithm and no key

INTRODUCTION

Cryptography is secret writing: secure communication that may be understood by the intended recipient only. While the fact that data is being transmitted may be known, the content of that data should remain unknown to third parties. Data in motion (moving on a network) and at rest (stored on a device such as a disk) may be encrypted.

The use of cryptography dates back thousands of years, but is very much a part of our modern world. Mathematics and computers play a critical role in modern cryptography.

CORNERSTONE CRYPTOGRAPHIC CONCEPTS

Fundamental cryptographic concepts are embodied by all strong encryption, and must be understood before learning about specific implementations.

CISSP® Study Guide. DOI: 10.1016/B978-1-59749-563-9.00004-4

Key Terms

Cryptology is the science of secure communications. *Cryptography* creates messages whose meaning is hidden; *cryptanalysis* is the science of breaking encrypted messages (recovering their meaning). Many use the term cryptography in place of cryptology: it is important to remember that cryptology encompasses both cryptography and cryptanalysis.

A *cipher* is a cryptographic algorithm. A *plaintext* is an unencrypted message. *Encryption* converts the plaintext to a *ciphertext*. *Decryption* turns a ciphertext back into a plaintext.

Confidentiality, Integrity, Authentication, and Non-Repudiation

Cryptography can provide confidentiality (secrets remain secret) and integrity (data is not altered in an unauthorized manner): it is important to note that it does not directly provide availability. Cryptography can also provide authentication (proving an identity claim).

Additionally, cryptography can provide *nonrepudiation*, which is an assurance that a specific user performed a specific transaction and that the transaction did not change. The two must be tied together. Proving that you signed a contract to buy a car is not useful if the car dealer can increase the cost after you signed the contract. Nonrepudiation means the individual who performed a transaction, such as authenticating to a system and viewing personally identifiable information (PII), cannot repudiate (or deny) having done so afterward.

Confusion, Diffusion, Substitution, and Permutation

Diffusion means the order of the plaintext should be "diffused" (or dispersed) in the ciphertext. *Confusion* means that the relationship between the plaintext and ciphertext should be as confused (or random) as possible. These terms were first defined by Claude Shannon, the father of information security, in his paper Communication Theory of Secrecy Systems, in 1949.[1]

Cryptographic *substitution* replaces one character for another; this provides diffusion. *Permutation* (also called transposition) provides confusion by rearranging the characters of the plaintext, anagram-style. "ATTACKATDAWN" can be rearranged to "CAAKDTANTATW," for example. Substitution and permutation are often combined. While these techniques were used historically (the *Caesar Cipher* is a substitution cipher), they are still used in combination in modern ciphers such as the *Advanced Encryption Standard* (AES).

Strong encryption destroys patterns. If a single bit of plaintext changes, the odds of every bit of resulting ciphertext changing should be 50/50. Any signs of nonrandomness may be used as clues to a cryptanalyst, hinting at the underlying order of the original plaintext or key.

NOTE

The dates and names (such as Claude Shannon) associated with cryptographic breakthroughs are generally not testable, unless the inventor's name appears in the name of the device or cipher. This information is given to flesh out the cryptographic concepts (which are very testable).

Cryptographic Strength

Good encryption is strong: for key-based encryption, it should be very difficult (and ideally impossible) to convert a ciphertext back to a plaintext without the key. The *work factor* describes how long it will take to break a cryptosystem (decrypt a ciphertext without the key).

Secrecy of the cryptographic algorithm does not provide strength: in fact secret algorithms are often proven quite weak. Strong crypto relies on math, not secrecy, to provide strength. Ciphers that have stood the test of time are public algorithms, such as the *Triple Data Encryption Standard* (TDES) and the Advanced Encryption Standard (AES).

Monoalphabetic and Polyalphabetic Ciphers

A *monoalphabetic cipher* uses one alphabet: a specific letter (like "E") is substituted for another (like "X"). A *polyalphabetic cipher* uses multiple alphabets: "E" may be substituted for "X" one round, and then "S" the next round.

Monoalphabetic ciphers are susceptible to frequency analysis. Figure 4.1 shows the frequency of English letters in text. A monoalphabetic cipher that substituted "X" for "E," "C" for "T," etc., would be quickly broken using frequency analysis. Polyalphabetic ciphers attempt to address this issue via the use of multiple alphabets.

Modular Math

Modular math lies behind much of cryptography: simply put, modular math shows you what remains (the remainder) after division. It is sometimes called "clock math" because we use it to tell time: assuming a 12-hour clock, 6 hours past 9:00 PM is 3:00 AM. In other words, 9 + 6 is 15, divided by 12 leaves a remainder of 3.

As we will see later, methods like the running-key cipher use modular math. There are 26 letters in the English alphabet; adding the letter "Y" (the 25th letter) to "C" (the third letter) equals "B" (the 2nd letter). In other words, 25 + 3 equals 28. 28 divided by 26 leaves a remainder of 2. It is like moving in a circle (such as a clock face): once you hit the letter "Z," you wrap around back to "A."

Exclusive Or (XOR)

Exclusive Or (XOR) is the "secret sauce" behind modern encryption. Combining a key with a plaintext via XOR creates a ciphertext. XOR-ing to same key to the ciphertext

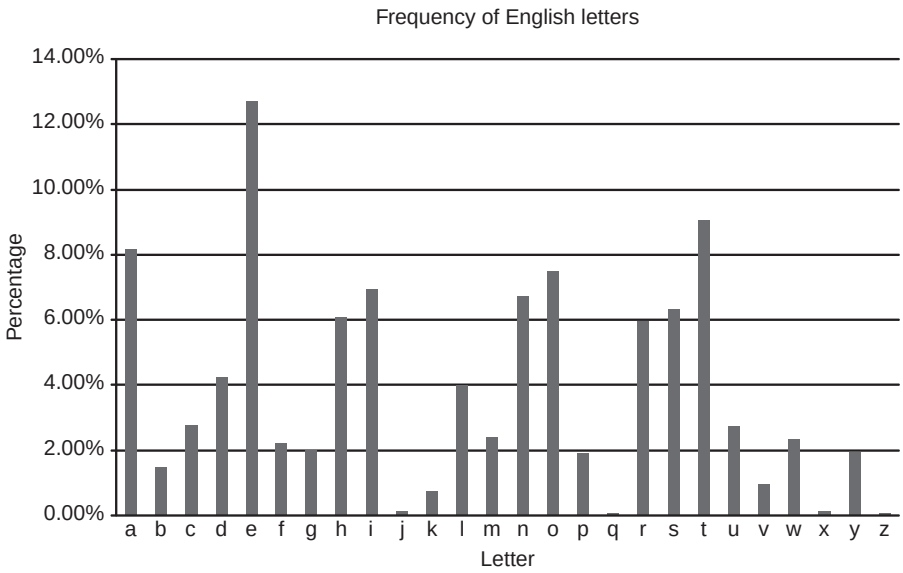


FIGURE 4.1

Frequency of English letters.

restores the original plaintext. XOR math is fast and simple, so simple that it can be implemented with phone relay switches (as we will see with the *Vernam Cipher*).

Two bits are true (or 1) if one or the other (exclusively, not both) is 1. If both bits are 0 or both bits are 1, they XOR to 0. XOR uses a *truth table*, shown in Table 4.1. In the truth table, a 0 is "false" and a 1 is true. This dictates how to combine the bits of a key and plaintext.

If you were to encrypt the plaintext "ATTACK AT DAWN" with a key of "UNICORN," you would XOR the bits of each letter together, letter by letter. We will encrypt and then decrypt the first letter to demonstrate XOR math. "A" is binary 01000001 and "U" is binary 01010101. We then XOR each bit of the plaintext to the key, using the truth table in Table 4.1. This results in a Ciphertext of 00010100, shown in Table 4.2.

Table 4.1 XOR Truth Table

X	Y	X XOR Y
0	0	0
0	1	1
1	0	1
1	1	0

Table 4.2 01000001 XORed to 01010101

Plaintext	0	1	0	0	0	0	0	1
Key	0	1	0	1	0	1	0	1
Ciphertext	0	0	0	1	0	1	0	0

Now let us decrypt the ciphertext 00010100 with a key of "U" (binary 01010101). We XOR each bit of the key (01010101) with the ciphertext (00010100), again using the truth table in Table 4.1. We recover our original plaintext of 01000001 (ASCII "A"), as shown in Table 4.3.

Types of Cryptography

There are three primary types of modern encryption: *symmetric*, *asymmetric*, and *hashing*. Symmetric encryption uses one key: the same key encrypts and decrypts. Asymmetric cryptography uses two keys: if you encrypt with one key, you may decrypt with the other. Hashing is a one-way cryptographic transformation using an algorithm (and no key).

HISTORY OF CRYPTOGRAPHY

Cryptography is the oldest domain in the Common Body of Knowledge: stretching back thousands of years to the days of the Pharos in Egypt. Cryptography has changed the course of human history, playing a role in world wars and political intrigue.

Egyptian Hieroglyphics

Hieroglyphics are stylized pictorial writing used in ancient Egypt. Some hieroglyphics contained small puzzles, meant to attract the attention of the reader, who would solve the simple pictorial challenge. One type of puzzle featured a serpent-like symbol in place of a letter such as "S." This form of writing was popular from roughly 2000 to 1000 B.C.

The meaning was hidden, albeit weakly, and this became the first known example of secret writing, or cryptography.

Table 4.3 00010100 XORed to 01010101

Ciphertext	0	0	0	1	0	1	0	0
Key	0	1	0	1	0	1	0	1
Plaintext	0	1	0	0	0	0	0	1

Spartan Scytale

The Scytale was used in ancient Sparta around 400 B.C. A strip of parchment was wrapped around a rod (like the tape on a baseball or cricket bat). The plaintext was encrypted by writing lengthwise down the rod (across the wrapped strip). The message was then unwound and sent. When unwound, the words appeared as a meaningless jumble.

The receiver, possessing a rod of the same diameter, wrapped the parchment across the rod, reassembling the message.

Caesar Cipher and other Rotation Ciphers

The Caesar Cipher is a monoalphabetic rotation cipher used by Gaius Julius Caesar. Caesar rotated each letter of the plaintext forward three times to encrypt, so that A became D, B became E, etc., as shown in Table 4.4.

Table 4.5 shows how "ATTACK AT DAWN" encrypts to "DWWDFN DW GDZQ" using the Caesar Cipher. Note that rotating three letters is arbitrary; any number of letters (other than 26, assuming an English alphabet) may be rotated for the same effect.

Another common rotation cipher is Rot-13, frequently used to conceal information on bulletin board systems such as Usenet. For example, details that could "spoil" a movie for someone who had not seen it would be encoded in Rot-13: "Qrpxneq vf n ercyvpnag!" Many Usenet readers had a Rot-13 function to quickly decode any such messages.

Rot-13 rotates 13 characters, so that "A" becomes "N," "B" becomes "O," etc. A nice feature of Rot-13 is one application encrypts (albeit weakly); a second application decrypts (the equivalent of Rot-26, where "A" becomes "A" again).

Table 4.4 Caesar (Rot-3) Cipher

...	X	Y	Z	A	B	C	D	E	F	G	H	...
⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	
...	A	B	C	D	E	F	G	H	I	J	K	...

Table 4.5 Encrypting "ATTACK AT DAWN" with the Caesar Cipher

ROT 0	A	T	T	A	C	K	A	T	D	A	W	N
ROT 1	B	U	U	B	D	L	B	U	E	B	X	O
ROT 2	C	V	V	C	E	M	C	V	F	C	Y	P
ROT 3	D	W	W	D	F	N	D	W	G	D	Z	Q

Vigenère Cipher

The Vigenère cipher is a polyalphabetic cipher named after Blaise de Vigenère, a French cryptographer who lived in the 16th century. The alphabet is repeated 26 times to form a matrix, called the Vigenère Square. Assume a plaintext of "ATTACKATDAWN." A key (such as "NEXUS") is selected and repeated ("NEXUSNEXUS. . ."). The plaintext is then encrypted with the key via lookups to the Vigenère Square. Plaintext "A" becomes ciphertext "N," and Figure 4.2 shows how plaintext "T" becomes ciphertext "X." The full ciphertext is "NXQUUXEQXSJR."

Cipher Disk

Cipher disks have two concentric disks, each with an alphabet around the periphery. They allow both monoalphabetic and polyalphabetic encryption. For monoalphabetic encryption, two parties agree on a fixed offset: "Set 'S' to 'D'." For polyalphabetic encryption, the parties agree on a fixed starting offset, and then turn

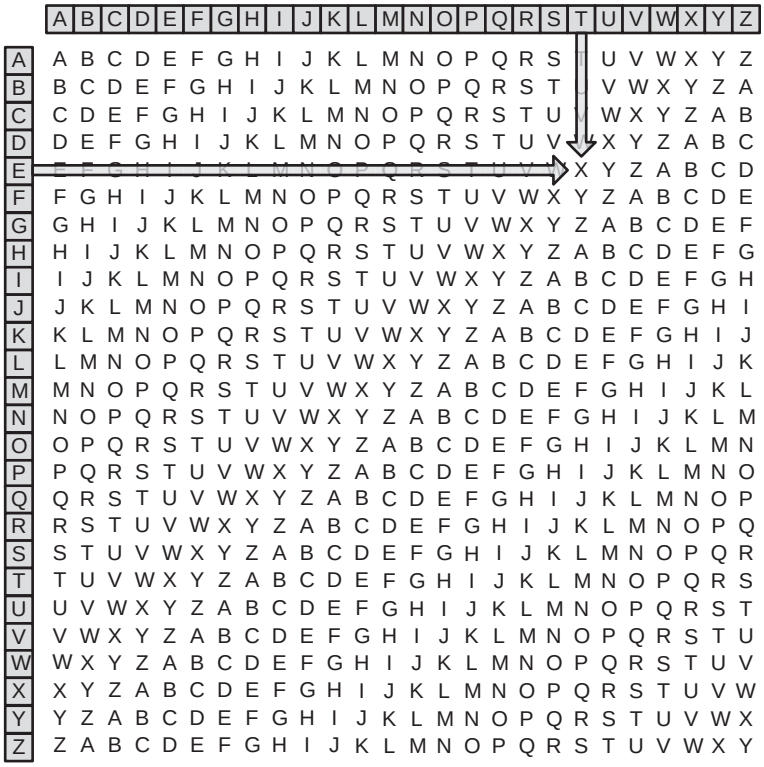


FIGURE 4.2

Vigenère square encrypting plaintext "T" with a key of "E."

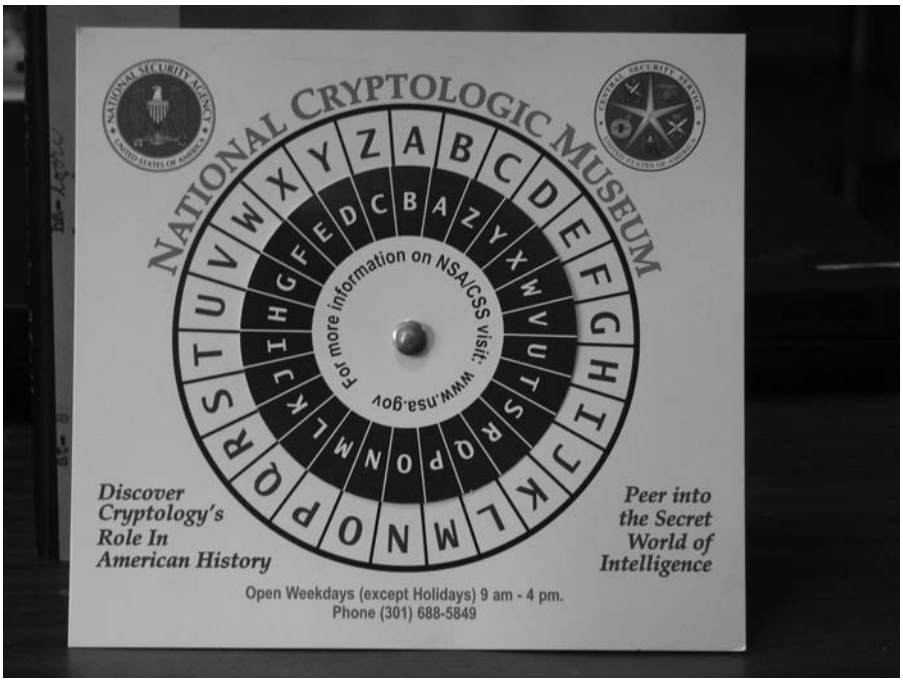


FIGURE 4.3

A modern cipher disk from the National Cryptologic Museum.

(*Courtesy of the National Security Agency*)

the wheel once every X characters: "Set 'S' to 'D,' and then turn the inner disk 1 character to the right after every 10 characters of encryption." Figure 4.3 shows a modern cipher disk.

The cipher disk was invented in 1466 or 1467 by Leon Battista Alberti, an Italian architect and Renaissance man. The disks were made of copper, with two concentric alphabets. In addition to inventing the cipher disk, Alberti is considered the inventor of the polyalphabetic cipher: he began with a static offset, but turned the disks after each few words were encrypted.

Cipher disks were used for hundreds of years, through the U.S. Civil war. Figure 4.4 shows original brass cipher disks used by the Confederate States of America.

Jefferson Disks

Jefferson Disks were created by Thomas Jefferson in the 1790s. Jefferson called his invention the "Wheel Cypher;" it had 36 wooden disks, each with 26 letters in random order ("jumbled and without order," according to Jefferson[2]) along the edge, like the ridges of a coin. The device, shown in Figure 4.5, was used briefly and

FIGURE 4.4

Confederate States of America brass cipher disks.

(*Courtesy of the National Security Agency*)

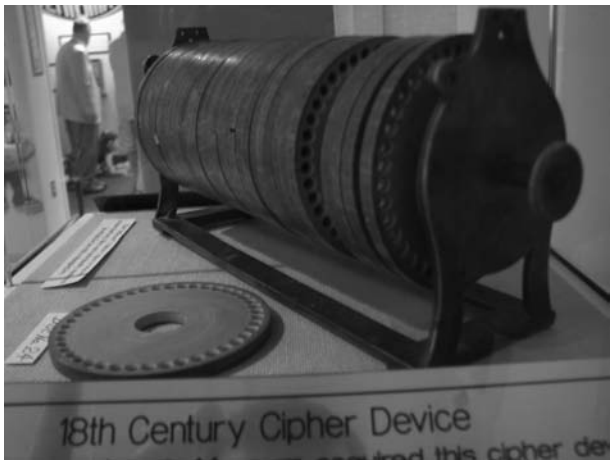


FIGURE 4.5

Jefferson disks.

(*Courtesy of the National Security Agency*)

then forgotten. Cipher wheels were later independently invented. Jefferson's papers describing his "cypher" were rediscovered in 1922.

To encrypt a message with Jefferson Disks, you must first create an identical set of disks and securely send one to the party you wish to communicate with. Then arrange the first 36 letters of plaintext along one line of letters on the disks. Then pick any other line of "jumbled" letters: this is the ciphertext. Continue this process for each 36 letters of plaintext.

To decrypt, the recipient arranges the ciphertext along one line of the disks. Then the recipient scans the other 25 lines, looking for one that makes sense (the rest will be a jumble of letters, in all likelihood).

David Kahn, in his seminal history of cryptography called *The Codebreakers*, stated that the Jefferson Disk was the most advanced cryptographic device of its time and called Thomas Jefferson "the Father of American Cryptography."[2]

Book Cipher and Running-Key Cipher

The *book cipher* and *running-key* cipher both use well-known texts as the basis for keys.

A book cipher uses whole words from a well-known text such as a dictionary. To encode, agree on a text source, and note the page number, line, and word offset of each word you would like to encode. Benedict Arnold used a book cipher to communicate with British conspirators.

Arnold and British army officer John André agreed to use Nathan Bailey's Universal Etymological English Dictionary to encode and decode messages. Here is a sample of ciphertext sent from Arnold to André on July 12, 1780: "As 158.9.25 and 115.9.12 are 226.9.3′d by./236.8.20ing 131.9.21, 163.9.6. . ." The ciphertext means "As <word on page 158, column 9, offset 25> and <word on page 115, column 9, offset 12>. . ." etc. This translates into "As Life and fortune are risked by serving His Majesty. . ."[3]

Running-key ciphers also use well-known texts as the basis for their keys: instead of using whole words, they use modulus math to "add" letters to each other. Assume a conspirator wishes to send the message "ATTACK AT DAWN" to a fellow conspirator. They have agreed to use the Preamble of the United States Constitution ("We the People of the United States, in Order to form a more perfect Union. . .") as their running key. Table 4.6 shows the resulting ciphertext.

Codebooks

Codebooks assign a codeword for important people, locations, and terms. One example is the *Cipher for Telegraphic Correspondence*, which was used by Union General Joseph Hooker during the United States Civil War. Each word in the codebook has two codenames. As shown in Figure 4.6, the president was "Adam" or "Asia," the Secretary of State was "Abel" or "Austria," etc.

One-Time Pad

A one-time pad uses identical paired pads of random characters, with a set amount of characters per page. Assume a pair of identical 100-page one-time pads with

Table 4.6 Running-Key Ciphertext of "ATTACK AT DAWN"

A	T	T	A	C	K	A	T	D	A	W	N
+	+	+	+	+	+	+	+	+	+	+	+
W	E	T	H	E	P	E	O	P	L	E	I
=	=	=	=	=	=	=	=	=	=	=	=
X	Y	N	I	H	A	F	I	T	M	B	W

FIGURE 4.6

Cipher for Telegraphic Correspondence.

(*Courtesy of the National Security Agency*)

1000 random characters per page. Once the identical pair of pads is created, they are securely given to two groups or individuals who wish to communicate securely.

Once the pads are securely distributed, either side may communicate by using the first page of the pad to encrypt up to 1000 characters of plaintext. The encryption is done with modular addition (as we saw previously, "Y" + "C" = "B"). The message is then sent to the receiver, who references the same page of the pad to decrypt via modular subtraction ("B" − "C" = "Y"). Once a page of the pad is used, it is discarded and never used again.

The one-time pad is the only encryption method that is mathematically proven to be secure, if the following three conditions are met: the characters on the pad are truly random, the pads are kept secure, and no page is ever reused.

Vernam Cipher

The first known use of a one-time pad was the Vernam Cipher, named after Gilbert Vernam, an employee of AT&T Bell Laboratories. In 1917 he invented a teletypewriter (capable of transmitting teletypes via phone lines) that encrypted and decrypted using paper rolls of tape containing the encryption key. Originally the keys were reused; the system began using a one-time pad (pairs of identical tapes with random keys that were never reused) in the 1920s.

The Vernam cipher used bits (before the dawn of computers, as other teletypes also did). The one-time pad bits were XORed to the plaintext bits.

Project VENONA

VENONA was the project undertaken by United States and United Kingdom cryptanalysts to break the KGB's (the Soviet Union's national security agency) encryption in the 1940s.

The KGB used one-time pads for sensitive transmissions, which should have rendered the ciphertext unbreakable. The KGB violated one of the three rules of one-time pads: they reused the pads. This allowed the U.S. and U.K. cryptanalysts to break many of the transmissions, providing critical intelligence. Many famous names were decrypted, including details on the nuclear espionage committed by Ethel and Julius Rosenberg.

NOTE

Project VENONA itself is not testable; it is described to show the dangers of reusing the pages of a one-time pad.

Hebern Machines and Purple

Hebern Machines are a class of cryptographic devices known as rotor machines, named after Edward Hebern. Figure 4.7 shows an original Hebern Electric Code Machine. They look like large manual typewriters, electrified with rotors (rotating motors). These devices were used after World War I, through World War II, and in some cases into the 1950s.

FIGURE 4.7

Hebern electric code machine.

(*Courtesy of the National Security Agency*)

Enigma

Enigma was used by German Axis powers during World War II. The initial cryptanalysis of Enigma was performed by French and Polish cryptanalysts; the work was continued by the British, led by Alan Turing in Bletchley Park, England. The intelligence provided by the cryptanalysis of Enigma (called Ultra) proved critical in the European theater of World War II. British cryptanalyst Sir Harry Hinsley said, "the war, instead of finishing in 1945, would have ended in 1948 had the Government Code and Cypher School not been able to read the Enigma ciphers and produce the Ultra intelligence."[4]

Enigma, shown in Figure 4.8, looks like a large typewriter with lamps and finger wheels added. The military version of Enigma (commercial versions also existed) had three finger wheels which could be set to any number from 1 to 26 (the finger wheels provide the key). As you type on the keyboard, the finger wheels turn, and a lamp for the corresponding ciphertext illuminates. To decrypt, set the finger wheels back to their original position, and type the ciphertext into the keyboard. The lamps illuminate to show the corresponding plaintext.

SIGABA

SIGABA was a rotor machine used by the United States through World War II into the 1950s. While similar to other rotor machines such as Enigma, it was more complex, based on analysis of weaknesses in Enigma by American cryptanalysts including William Friedman. SIGABA was also called ECM (Electronic Code Machine) Mark II.

SIGABA, shown in Figure 4.9, was large, complex, and heavy: far heavier and cumbersome than Enigma. As a result, it saw limited field use. SIGABA was never known to be broken.

FIGURE 4.8

A young cryptographer using Enigma at the National Cryptologic Museum.

(*Courtesy of the National Security Agency*)

FIGURE 4.9

SIGABA.

(*Courtesy of the National Security Agency*)

Purple

Purple is the Allied name for the encryption device used by Japanese Axis powers during World War II. While many sources describe Purple as a rotor machine from the same era, such as Enigma and American SIGABA, Purple is actually a stepping-switch device, primarily built with phone switch hardware. Other models included Red and Jade. Figure 4.10 shows a fragment of a Purple machine recovered from the Japanese Embassy in Berlin at the end of World War II.

While Alan Turing led the British cryptanalysis of Enigma, senior cryptanalyst William Friedman led the United States effort against Purple. The Japanese Axis powers took Japanese plaintext, added code words, and then encrypted with Purple. The U.S. challenge was threefold: decrypt, translate the code words, and then translate Japanese to English.

In 1942, the Allies decoded Purple transmissions referencing a planned sneak attack on "AF." The Allies believed AF was a code word for Midway Island, but they wanted to be sure. They sent a bogus message, weakly encoded, stating there was a water problem on Midway Island. Two days later the Allies decrypted a Purple transmission stating there was a water problem on AF.

The Allies knew where and when the "sneak" attack would be launched, and they were ready. The Battle of Midway Island provided a decisive victory for the Allies, turning the tide of war in the Pacific theater.

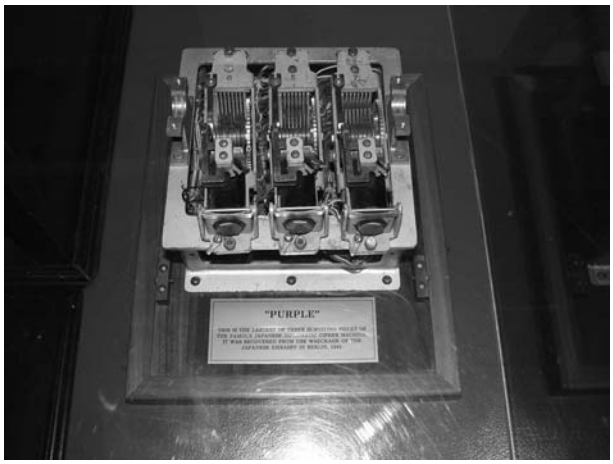


FIGURE 4.10

Fragment of Japanese Purple machine.

(*Courtesy of the National Security Agency*)

Cryptography Laws

The importance of crypto was not lost on many governments, especially the United States. Intelligence derived from cryptanalysis was arguably as powerful as any bomb. This lead to attempts to control cryptography through the same laws used to control bombs: munitions laws.

COCOM

COCOM is the Coordinating Committee for Multilateral Export Controls, which was in effect from 1947 to 1994. It was designed to control the export of critical technologies (including cryptography) to "Iron Curtain" countries during the cold war.

Charter COCOM members included the United States and a number of European countries. Later Japan, Australia, Turkey, and much of the rest of the non-Soviet-controlled countries in Europe joined. Export of encryption by members to non-COCOM countries was heavily restricted.

Wassenaar Arrangement

After COCOM ended, the *Wassenaar Arrangement* was created in 1996. It features many more countries, including former Soviet Union countries such as Estonia, the Russian Federation, Ukraine, and others. The Wassenaar Arrangement also relaxed many of the restrictions on exporting cryptography.

SYMMETRIC ENCRYPTION

Symmetric encryption uses one key to encrypt and decrypt. If you encrypt a zip file, and then decrypt with the same key, you are using symmetric encryption. Symmetric encryption is also called "Secret key" encryption: the key must be kept

secret from third parties. Strengths include speed and cryptographic strength per bit of key. The major weakness is that the key must be securely shared before two parties may communicate securely. Symmetric keys are often shared via an out-of-band method, such as via face-to-face discussion.

The key is usually converted into a subkey, which changes for each block of data that is encrypted.

Stream and Block Ciphers

Symmetric encryption may have stream and block modes. Stream mode means each bit is independently encrypted in a "stream." Block mode ciphers encrypt blocks of data each round: 56 bits for the Data Encryption Standard (DES), and 128, 192, or 256 bits for AES, for example. Some block ciphers can emulate stream ciphers by setting the block size to 1 bit; they are still considered block ciphers.

Initialization Vectors and Chaining

An initialization vector is used in some symmetric ciphers to ensure that the first encrypted block of data is random. This ensures that identical plaintexts encrypt to different ciphertexts. Also, as Bruce Schneier notes in *Applied Cryptography*, "Even worse, two messages that begin the same will encrypt the same way up to the first difference. Some messages have a common header: a letterhead, or a 'From' line, or whatever."[5] Initialization vectors solve this problem.

Chaining (called *feedback* in stream modes) seeds the previous encrypted block into the next block to be encrypted. This destroys patterns in the resulting ciphertext. DES *Electronic Code Book* mode (see below) does not use an initialization vector or chaining and patterns can be clearly visible in the resulting ciphertext.

Data Encryption Standard

DES is the Data Encryption Standard, which describes the *Data Encryption Algorithm* (DEA). DES was made a United States federal standard symmetric cipher in 1976. It was created due to a lack of cryptographic standards: vendors used proprietary ciphers of unknown strengths which did not interoperate with other vendor's ciphers. DES was designed by IBM, based on their older Lucifer symmetric cipher. It uses a 64-bit block size (meaning it encrypts 64 bits each round) and a 56-bit key.

EXAM WARNING

Even though "DES" is commonly-referred to as an algorithm, DES is technically the name of the published standard that describes DEA. It may sound like splitting hairs, but that is an important distinction to keep in mind on the exam. "DEA" may be the best answer for a question regarding the algorithm itself.

Modes of DES

DES can use five different modes to encrypt data. The modes' primary difference is block versus (emulated) stream, the use of initialization vectors, and whether errors in encryption will propagate to subsequent blocks.

The five modes of DES are:

- Electronic Code Book (ECB)
- Cipher Block Chaining (CBC)
- Cipher Feedback (CFB)
- Output Feedback (OFB)
- Counter Mode (CTR)

ECB is the original mode of DES. CBC, CFB, and OFB were later added in FIPS Publication 81 (see http://www.itl.nist.gov/fipspubs/fip81.htm). CTR mode is the newest mode, described in NIST Special Publication 800-38a (see: http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf).

Electronic Code Book (ECB)

Electronic Code Book (ECB) is the simplest and weakest form of DES. It uses no initialization vector or chaining. Identical plaintexts with identical keys encrypt to identical ciphertexts. Two plaintexts with partial identical portions (such as the header of a letter) encrypted with the same key will have partial identical ciphertext portions.

> **NOTE**
>
> The term "Code Book" in Electronic Code Book derives from cryptographic code books such as those used during the United States Civil war. This is also a hint to remind you of ECB's simplicity (and weakness).

ECB may also leave plaintext patterns evident in the resulting ciphertext. Bitmap image data (see Figure 4.11a) encrypted with a key of "Kowalski" using 56-bit DES ECB mode (see Figure 4.11b) shows obvious patterns.

Cipher Block Chaining (CBC)

Cipher Block Chaining (CBC) mode is a block mode of DES that XORs the previous encrypted block of ciphertext to the next block of plaintext to be encrypted. The first encrypted block is an initialization vector that contains random data. This "chaining" destroys patterns. One limitation of CBC mode is that encryption errors will propagate: an encryption error in one block will cascade through subsequent blocks due to the chaining, destroying their integrity.

Cipher Feedback (CFB)

Cipher Feedback (CFB) mode is very similar to CBC; the primary difference is CFB is a stream mode. It uses feedback (the name for chaining when used in stream modes) to destroy patterns. Like CBC, CFB uses an initialization vector and destroys patterns, and errors propagate.

FIGURE 4.11

(a) Plaintext 8-bit bitmap (BMP) image (*Courtesy of the National Security Agency*). (b) 56-bit DES ECB-encrypted ciphertext bitmap.

Output Feedback (OFB)

Output Feedback (OFB) mode differs from CFB in the way feedback is accomplished. CFB uses the previous ciphertext for feedback. The previous ciphertext is the subkey XORed to the plaintext. OFB uses the subkey *before* it is XORed to the plaintext. Since the subkey is not affected by encryption errors, errors will not propagate.

Counter (CTR)

Counter (CTR) mode is like OFB; the difference again is the feedback: CTRmode uses a counter. This mode shares the same advantages as OFB (patterns are destroyed and errors do not propagate) with an additional advantage: since the feedback can be as

Table 4.7 Modes of DES Summary

	Type	Initialization Vector	Error Propagation?
Electronic Code Book (ECB)	Block	No	No
Cipher Block Chaining (CBC)	Block	Yes	Yes
Cipher Feedback (CFB)	Stream	Yes	Yes
Output Feedback (OFB)	Stream	Yes	No
Counter Mode (CTR)	Stream	Yes	No

simple as an ascending number, CTR mode encryption can be done in parallel. A simple example would be the first block is XORed to the number 1, the second to the number 2, etc. Any number of rounds can be combined in parallel this way.

Table 4.7 summarizes the five modes of DES.

Single DES

Single DES is the original implementation of DES, encrypting 64-bit blocks of data with a 56-bit key, using 16 rounds of encryption. The work factor required to break DES was reasonable in 1976, but advances in CPU speed and parallel architecture have made DES weak to a *brute-force* key attack today, where every possible key is generated and attempted. Massively parallel computers such as COPACOBANA (Cost-Optimized Parallel COde Breaker, given as a nontestable example, see: http://www.copacobana.org for more information), which uses over 100 CPUs in parallel, can break 56-bit DES in a week or so (and faster with more CPUs), at a cost of under $10,000.

Triple DES

Triple DES applies single DES encryption three times per block. Formally called the "Triple Data Encryption Algorithm (TDEA) and commonly called TDES," it became a recommended standard in 1999 by the United States Federal Information Processing Standard (FIPS) Publication 46-3 (see: http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf). FIPS 46-3 recommended single DES for legacy use only, due to the ever-lowering work factor required to break single DES.

Triple DES has held up well after years of cryptanalysis; the primary weakness is that it is slow and complex compared to newer symmetric algorithms such as AES or Twofish. Note that "double DES" (applying DES encryption twice using two keys) is not used due to a *meet-in-the-middle attack*: see the "Cryptographic Attacks" section for more information.

Triple DES encryption order and keying options

Triple DES applies DES encryption three times per block. FIPS 46-3 describes "Encrypt, Decrypt, Encrypt" (EDE) order using three keying options: one, two,

or three unique keys (called 1TDES EDE, 2TDES EDE, and 3TDES EDE, respectively).

This order may seem confusing: why not encrypt, encrypt, encrypt, or EEE? And why use one through three keys? If you "decrypt" with a different key than the one used to encrypt, you are really encrypting further. Also, EDE with one key allows backwards compatibility with single DES.

Table 4.8 shows a single DES ECB encryption of "ATTACK AT DAWN" with the key "Hannibal" results in ciphertext of "•ÁGPÚÂ ¦qŸÝ«¦-" (this is the actual ciphertext; some bytes contain nonprintable characters).

Applying triple DES EDE with the same key each time results in the same ciphertext as single DES. Round 3 is identical to round 1, as shown in Table 4.9.

2TDES EDE uses key 1 to encrypt, key 2 to "decrypt," and key 1 to encrypt. This results in 112 bits of key length. It is commonly used for legacy hardware applications with limited memory.

3TDES EDE (three different keys) is the strongest form, with 168 bits of key length. The effective strength is 112 bits due to a partial meet-in-the-middle attack; see the Cryptographic Attacks section of this chapter for more information.

International Data Encryption Algorithm (IDEA)

The International Data Encryption Algorithm is a symmetric block cipher designed as an international replacement to DES. The IDEA algorithm is patented in many countries. It uses a 128-bit key and 64-bit block size. IDEA has held up to cryptanalysis; the primary drawbacks are patent encumbrance and its slow speed compared to newer symmetric ciphers such as AES.

Advanced Encryption Standard (AES)

The Advanced Encryption Standard is the current United States standard symmetric block cipher. It was published in Federal Information Processing Standard (FIPS) 197 (see: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf). AES uses 128-bit (with

Table 4.8 Single DES encryption

Operation	Key	Input	Output
Encrypt	Hannibal	ATTACK AT DAWN	•ÁGPÚÂ ¦qŸÝ«¦

Table 4.9 Triple DES Encryption with One Key

Operation	Key	Input	Output
Encrypt	Hannibal	ATTACK AT DAWN	•ÁGPÚÂ ¦qŸÝ«¦
Decrypt	Hannibal	•ÁGPÚÂ ¦qŸÝ«¦	ATTACK AT DAWN
Encrypt	Hannibal	ATTACK AT DAWN	•ÁGPÚÂ ¦qŸÝ«¦

10 rounds of encryption), 192-bit (12 rounds of encryption), or 256-bit (14 rounds of encryption) keys to encrypt 128-bit blocks of data. AES is an open algorithm, free to use, and free of any intellectual property restrictions.

AES was designed to replace DES. Two- and three-key TDES EDE remain a FIPS-approved standard until 2030, to allow transition to AES. Single DES is not a current standard, and not recommended.

Choosing AES

The United States National Institute of Standards and Technology (NIST) solicited input on a replacement for DES in the *Federal Register* in January 1997. They sought a public symmetric block cipher algorithm that was more secure than DES, open, and fast and efficient in both hardware and software. Fifteen AES candidates were announced in August 1998, and the list was reduced to five in August 1999. Table 4.10 lists the five AES finalists.

Rijndael was chosen and became AES. The name, pronounced "Rhine Dahl" in English, is a combination of the Belgian authors' names: Vincent Rijmen and Joan Daemen. Rijndael was chosen "because it had the best combination of security, performance, efficiency, and flexibility."[6]

Table 4.11 shows the "State," which is the block of data that is being encrypted via AES. Each smaller box in the State is a byte (8 bits), and there are 16 bytes (128 bits) in each block. Data is encrypted and visualized in literal blocks. The algorithm that AES is based on was called "Square" for this reason.

Table 4.10 Five AES Finalists

Name	Author
MARS	IBM (11 authors)
RC6	RSA (Rivest, Robshaw, Sidney, Yin)
Rijndael	Daemen, Rijmen
Serpent	Anderson, Biham, Knudsen
Twofish	Schneier, Kelsey, Hall, Ferguson, Whiting, Wagner

Table 4.11 One 128-bit Block of AES Data, Called the State

0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
3,0	3,1	3,2	3,3

AES functions

AES has four functions: SubBytes, ShiftRows, MixColumns, and AddRoundKey. These functions provide confusion, diffusion, and XOR encryption to the State.

ShiftRows

ShiftRows provides diffusion by shifting rows of the State. It treats each row like a row of blocks, shifting each a different amount:

- Row 0 is unchanged
- Row 1 is shifted 1 to the left
- Row 2 is shifted 2 to the left
- Row 3 is shifted 3 to the left.

Table 4.12 shows the transformation to the State.

MixColumns

MixColumns also provides diffusion by "mixing" the columns of the State via finite field mathematics, as shown in Table 4.13.

SubBytes

The SubBytes function provides confusion by substituting the bytes of the State. The bytes are substituted according to a substitution table (also called an S-Box).

To use the table, take the byte of the State to be substituted (assume the byte is the letter "T"). ASCII "T" is hexadecimal byte "53." Look up 5 on the X row and 3 on the Y column, resulting in hexadecimal byte "ed;" this replaces "53" in the State. Figure 4.12 shows the AES substitution table directly from FIPS-97, with the byte 53 lookup overlaid on top:

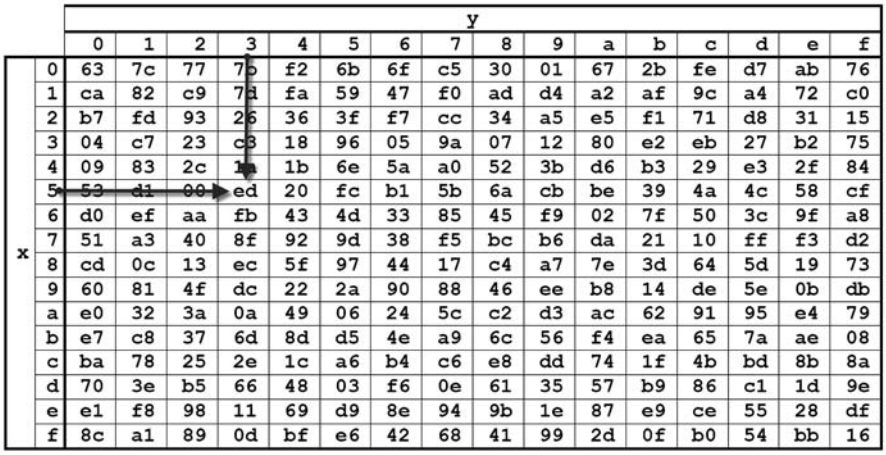
Table 4.12 ShiftRows, Before and After

0,0	0,1	0,2	0,3
1,0	1,1	1,2	1,3
2,0	2,1	2,2	2,3
3,0	3,1	3,2	3,3

0,0	0,1	0,2	0,3
1,1	1,2	1,3	1,0
2,2	2,3	2,0	2,1
3,3	3,0	3,1	3,2

Table 4.13 MixColumns

Mix Column 0	Mix Column 1	Mix Column 2	Mix Column 3



x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

FIGURE 4.12

AES substitution table converting byte "53" to "eb."[7]

AddRoundKey

AddRoundKey is the final function applied in each round. It XORS the State with the subkey. The subkey is derived from the key, and is different for each round of AES.

Blowfish and Twofish

Blowfish and Twofish are symmetric block ciphers created by teams lead by Bruce Schneier, author of *Applied Cryptography*. Blowfish uses from 32 through 448 bit (the default is 128) keys to encrypt 64 bits of data. Twofish was an AES finalist, encrypting 128-bit blocks using 128 through 256 bit keys. Both are open algorithms, unpatented and freely available.

RC5 and RC6

RC5 and RC6 are symmetric block ciphers by RSA Laboratories. RC5 uses 32 (testing purposes), 64 (replacement for DES), or 128-bit blocks. The key size ranges from zero to 2040 bits.

RC6 was an AES finalist. It is based on RC5, altered to meet the AES requirements. It is also stronger than RC5, encrypting 128-bit blocks using 128-, 192-, or 256-bit keys.

ASYMMETRIC ENCRYPTION

For thousands of years, cryptographic ciphers suffered from a chicken-and-egg problem: in order to securely communicate with someone, you had to first (securely) share a key or device. Asymmetric encryption was a mathematical

breakthrough of the 1970s, finally solving the age-old challenge of preshared keys. Asymmetric pioneers include Whitfield Diffie and Martin Hellman, who created the Diffie-Hellman key exchange in 1976. The RSA algorithm was invented in 1977 (RSA stands for "Rivest, Shamir, and Adleman," the authors' names).

Asymmetric encryption uses two keys: if you encrypt with one key, you may decrypt with the other. One key may be made public (called the *public key*); asymmetric encryption is also called public key encryption for this reason. Anyone who wants to communicate with you may simply download your publicly-posted public key and use it to encrypt their plaintext. Once encrypted, your public key cannot decrypt the plaintext: only your *private key* can do so. As the name implies, your private key must be kept private and secure.

Additionally, any message encrypted with the private key may be decrypted with the public key. This is typically used for digital signatures, as we will see shortly.

Asymmetric Methods

Math lies behind the asymmetric breakthrough. These methods use "one-way functions," which are easy to compute "one way," and difficult to compute in the reverse direction.

Factoring Prime Numbers

An example of a one-way function is factoring a composite number into its primes. A prime number is a number evenly divisible only by one and itself; a composite number is evenly divisible by numbers other than 1 and itself.

Multiplying the prime number 6269 by the prime number 7883 results in the composite number 49,418,527. That "way" is quite easy to compute, taking milliseconds on a calculator. Answering the question "which prime number times which prime number equals 49,418,527" is *much* more difficult. That problem is called factoring, and no shortcut has been found for hundreds of years. This is the basis of the RSA algorithm.

Factoring a large composite number (one thousands of bits long) is so difficult that the composite number can be safely publicly posted (this is the public key). The primes that are multiplied to create the public key must be kept private (they are the private key).

EXAM WARNING

Do not confuse "one way function" with "one way hash." The former describes asymmetric algorithms, the latter describes hash algorithms.

Discrete Logarithm

A logarithm is the opposite of exponentiation. Computing 7 to the 13th power (exponentiation) is easy on a modern calculator: 96,889,010,407. Asking the question "96,889,010,407 is 7 to what power" (finding the logarithm) is more difficult.

Discrete logarithms apply logarithms to groups, which is a much harder problem to solve. This one-way function is the basis of the *Diffie-Helmann* and *ElGamal* asymmetric algorithms.

Diffie-Hellman key agreement protocol

Key agreement allows two parties to securely agree on a symmetric key via a public channel, such as the Internet, with no prior key exchange. An attacker who is able to sniff the entire conversation is unable to derive the exchanged key. The Diffie-Hellman Key Agreement Protocol (also called the Diffie-Hellman Key Exchange) was created in 1976 by Whitfield Diffie and Martin Hellman in 1976. Diffie-Hellman uses discrete logarithms to provide security.

Elliptic Curve Cryptography (ECC)

ECC leverages a one-way function that uses discrete logarithms as applied to elliptic curves. Solving this problem is harder than solving discrete logarithms, so algorithms based on Elliptic Curve Cryptography (ECC) are much stronger per bit than systems using discrete logarithms (and also stronger than factoring prime numbers). ECC requires less computational resources because shorter keys can be used compared to other asymmetric methods. ECC is often used in lower power devices for this reason.

Asymmetric and Symmetric Tradeoffs

Asymmetric encryption is far slower than symmetric encryption, and is also weaker per bit of key length. The strength of asymmetric encryption is the ability to securely communicate without presharing a key.

Table 4.14 compares symmetric and asymmetric algorithms based on key length. Note that systems based on discrete logarithms and factoring prime numbers are far weaker per bit of key length than symmetric systems such as Triple DES and AES. Elliptic Curve fares much better in comparison, but is still twice as weak per bit compared to AES.

Asymmetric and symmetric encryption are typically used together: use an asymmetric algorithm such as RSA to securely send someone an AES (symmetric)

Table 4.14 Symmetric Versus Asymmetric Strength[8]

Symmetric Key Length	Symmetric Algorithm	Discrete Logarithm Equivalent Key Length	Factoring Prime Numbers Equivalent Key Length	Elliptic Curve Equivalent Key Length
112	3TDES	2048	2048	224-255
128	AES	3072	3072	256-283
192	AES	7860	7860	384-511
256	AES	15,360	15,360	512+

key. The symmetric key is called the session key; a new session key may be retransmitted periodically via RSA.

This approach leverages the strengths of both cryptosystems. Use the slower and weaker asymmetric system for the one part that symmetric encryption cannot do: securely preshare keys. Once shared, leverage the fast and strong symmetric encryption to encrypt all further traffic.

HASH FUNCTIONS

A hash function provides encryption using an algorithm and no key. They are called "one-way hash functions" because there is no way to reverse the encryption. A variable-length plaintext is "hashed" into a fixed-length hash value (often called a "message digest" or simply a "hash"). Hash functions are primarily used to provide integrity: if the hash of a plaintext changes, the plaintext itself has changed. Common older hash functions include *Secure Hash Algorithm 1* (SHA-1), which creates a 160-bit hash and *Message Digest 5* (MD5), which creates a 128-bit hash. Weaknesses have been found in both MD5 and SHA-1; newer alternatives such as SHA-2 are recommended.

Collisions

Hashes are not unique, because the number of possible plaintexts is far larger than the number of possible hashes. Assume you are hashing documents that are a mega-bit long with MD5. Think of the documents as strings 1,000,000 bits long, and the MD5 hash as a string 128 bits long. The universe of potential 1,000,000-bit strings is clearly larger than the universe of 128-bit strings. Therefore more than one document could have the same hash: this is called a *collision*.

While collisions are always possible (assuming the plaintext is longer than the hash), they should be very difficult to find. Searching for a collision to match a specific plaintext should not be possible to accomplish in a reasonable amount of time.

MD5

MD5 is the Message Digest algorithm 5, created by Ronald Rivest. It is the most widely used of the MD family of hash algorithms. MD5 creates a 128-bit hash value based on any input length. MD5 has been quite popular over the years, but weaknesses have been discovered where collisions could be found in a practical amount of time. MD6 is the newest version of the MD family of hash algorithms, first published in 2008.

Secure Hash Algorithm

Secure Hash Algorithm is the name of a series of hash algorithms; SHA-1 was announced in 1993 in the United States Federal Information Processing Standard 180 (see http://www.itl.nist.gov/fipspubs/fip180-1.htm). SHA-1 creates a 160-bit hash value.

Like MD5, SHA-1 was also found to have weak collision avoidance. SHA-2 was announced in 2001 (see http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf). SHA-2 includes SHA-224, SHA-256, SHA-384, and SHA-512, named after the length of the message digest each creates.

While SHA-2 is recommended over SHA-1 or MD5, it is still less common due to its relative newness. The search for the next-generation hashing algorithm is already underway: the SHA-3 competition was announced in the *Federal Register* in 2007, similar to the AES competition.

HAVAL

HAVAL (Hash of Variable Length) is a hash algorithm that creates message digests of 128, 160, 192, 224, or 256 bits in length, using 3, 4, or 5 rounds. HAVAL uses some of the design principles behind the MD family of hash algorithms, and is faster than MD5.

CRYPTOGRAPHIC ATTACKS

Cryptographic attacks are used by cryptanalysts to recover the plaintext without the key. Please remember that recovering the key (sometimes called "steal the key") is usually easier than breaking modern encryption. This is what law enforcement typically does when faced with a suspect using cryptography: they obtain a search warrant and attempt to recover the key.

Brute Force

A brute-force attack generates the entire keyspace, which is every possible key. Given enough time, the plaintext will be recovered. This is an effective attack against all key-based ciphers, except for the one-time pad. Since the key of a one-time pad is the same length as the plaintext, brute forcing every possible key will eventually recover the plaintext, but it will also produce vast quantities of other potential plaintexts, including all the works of Shakespeare. A cryptanalyst would have no way of knowing which potential plaintext is real. This is why the one-time pad is the only provably unbreakable form of crypto.

Known Plaintext

A known plaintext attack relies on recovering and analyzing a matching plaintext and ciphertext pair: the goal is to derive the key which was used. You may be wondering why you would need the key if you already have the plaintext: recovering the key would allow you to decrypt other ciphertexts encrypted with the same key.

Chosen Plaintext and Adaptive Chosen Plaintext

A cryptanalyst chooses the plaintext to be encrypted in a chosen plaintext attack; the goal is to derive the key. Encrypting without knowing the key is done via an "encryption oracle," or a device that encrypts without revealing the key. This may sound far-fetched, but it is quite practical: a VPN concentrator encrypts plaintext to ciphertext without revealing the key (only users authorized to manage the device may see the key).

Adaptive-chosen plaintext begins with a chosen plaintext attack in round 1. The cryptanalyst then "adapts" further rounds of encryption based on the previous round.

Chosen Ciphertext and Adaptive Chosen Ciphertext

Chosen ciphertext attacks mirror chosen plaintext attacks: the difference is that the cryptanalyst chooses the ciphertext to be decrypted. This attack is usually launched against asymmetric cryptosystems, where the cryptanalyst may choose public documents to decrypt which are signed (encrypted) with a user's public key.

Adaptive-chosen ciphertext also mirrors its plaintext cousin: it begins with a chosen ciphertext attack in round 1. The cryptanalyst then "adapts" further rounds of decryption based on the previous round.

Meet-in-the-middle Attack

A meet-in-the-middle attack encrypts on one side, decrypts on the other side, and meets in the middle. The most common attack is against "double DES," which encrypts with two keys in "encrypt, encrypt" order. The attack is a known plaintext attack: the attacker has a copy of a matching plaintext and ciphertext, and seeks to recover the two keys used to encrypt.

The attacker generates every possible value for key 1 and uses each to encrypt the plaintext, saving the intermediate (half-encrypted) ciphertext results. DES has a 56-bit key, so this will take 2^{56} encryptions.

The attacker then generates every possible value for key 2, and uses each to decrypt the ciphertext. Once decrypted, the attacker looks up the intermediate ciphertext, looking for a match. If there is a match, the attacker has found both key 1 and key 2. The decryption step will take 2^{56} attempts at most, for a total of 2^{57} attempts (2^{56} encryptions + up to 2^{56} decryptions = 2^{57}).

In other words, despite 112 bits of key length, breaking double DES only twice as hard as breaking 56-bit single DES. This is far too easy, so double DES is not recommended. 3TDES has a key length of 168 bits, but an effective strength of 112 bits due to the meet-in-the-middle attack: 3TDES has three keys and two "middles," one can be used for a meet-in-the-middle attack, bypassing roughly one-third of the work.

Known Key

The term "known key attack" is misleading: if the cryptanalyst knows the key, the attack is over. Known key means the cryptanalyst knows something about the key, to reduce the efforts used to attack it. If the cryptanalyst knows that the key is an uppercase letter and a number only, other characters may be omitted in the attack.

Differential Cryptanalysis

Differential cryptanalysis seeks to find the "difference" between related plaintexts that are encrypted. The plaintexts may differ by a few bits. It is usually launched as an adaptive chosen plaintext attack: the attacker chooses the plaintext to be encrypted (but does not know the key), and then encrypts related plaintexts.

The cryptanalyst then uses statistical analysis to search for signs of nonrandomness in the ciphertexts, zeroing in on areas where the plaintexts differed. Every bit of the related ciphertexts should have a 50/50 chance of flipping: the cryptanalyst searches for areas where this is not true. Any such underlying order is a clue to recover the key.

Linear Cryptanalysis

Linear cryptanalysis is a known plaintext attack where the cryptanalyst finds large amounts of plaintext/ciphertext pairs created with the same key. The pairs are studied to derive information about the key used to create them.

Both differential and linear analysis can be combined as *differential linear analysis*.

Side-channel Attacks

Side-channel attacks use physical data to break a cryptosystem, such as monitoring CPU cycles or power consumption used while encrypting or decrypting. Some purists may claim this is breaking some type of rule, but as Bruce Schneier said, "Some researchers have claimed that this is cheating. True, but in real-world systems, attackers cheat. Their job is to recover the key, not to follow some rules of conduct. Prudent engineers of secure systems anticipate this and adapt to it."[9]

Birthday Attack

The birthday attack is named after the birthday paradox. The name is based on fact that in a room with 23 people or more, the odds are greater than 50% that two will share the same birthday. Many find this counterintuitive, and the birthday paradox illustrates why many people's instinct on probability (and risk) is wrong. You are not trying to match a specific birthday (such as yours); you are trying to match any birthday.

If you are in a room full of 23 people, you have a 1 in 365 chance of sharing a birthday with each of the 22 other people in the room, for a total of 22/365 chances. If you fail to match, you leave the room and Joe has a 21/365 chance of sharing a birthday with the remaining people. If Joe fails to match, he leaves the room and Morgan has a 20/365 chance, and so on. If you add 22/365 + 21/365 + 20/365 + 19/365 . . . + 1/365, you pass 50% probability.

The birthday attack is used to create hash collisions. Just as matching *your* birthday is difficult, finding a specific input with a hash that collides with another input is difficult. However, just like matching *any* birthday is easier, finding *any* input that creates a colliding hash with any other input is easier due to the birthday attack.

Key Clustering

A goal of any cryptographic cipher is that only one key can derive the plaintext from the ciphertext. Key Clustering occurs when two symmetric keys applied to the same plaintext produce the same ciphertext. This allows two different keys to decrypt the ciphertext.

IMPLEMENTING CRYPTOGRAPHY

Symmetric, asymmetric, and hash-based cryptography do not exist in a vacuum: they are applied in the real world, often in combination, to provide confidentiality, integrity, authentication, and nonrepudiation.

Digital Signatures

Digital signatures are used to cryptographically sign documents. Digital signatures provide nonrepudiation, which includes authentication of the identity of the signer, and proof of the document's integrity (proving the document did not change). This means the sender cannot later deny (or repudiate) signing the document.

Roy wants to send a digitally signed email to Rick. Roy writes the email, which is the plaintext. He then uses the SHA-1 hash function to generate a hash value of the plaintext. He then creates the digital signature by encrypting the hash with his RSA private key. Figure 4.13 shows this process. Roy then attaches the signature to his plaintext email and hits send.

FIGURE 4.13

Creating a digital signature.[10]

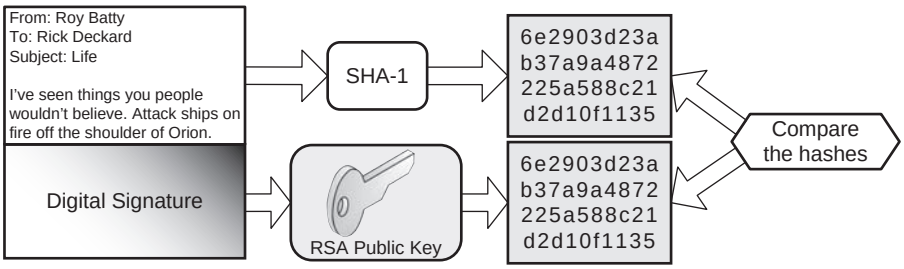


FIGURE 4.14

Verifying a digital signature.

Rick receives Roy's email and generates his own SHA-1 hash value of the plaintext email. Rick then decrypts the digital signature with Roy's RSA public key, recovering the SHA-1 hash Roy generated. Rick then compares his SHA-1 hash with Roy's. Figure 4.14 shows this process.

If the two hashes match, Rick knows a number of things:

1. Roy must have sent the email (only Roy knows his private key). This authenticates Roy as the sender.
2. The email did not change. This proves the integrity of the email.

If the hashes match, Roy cannot later deny having signed the email. This is nonrepudiation. If the hashes do not match, Rick knows either Roy did not send it, or that the email's integrity was violated.

> **NOTE**
>
> Digital signatures provide authentication and integrity, which forms nonrepudiation. They do not provide confidentiality: the plaintext remains unencrypted.

HMAC

A *Hashed Message Authentication Code* (HMAC) combines symmetric encryption with hashing. The approach is similar to a digital signature, except that it uses symmetric encryption instead of asymmetric. HMACs are used by IPsec (see below).

Two parties must preshare a secret key (such as a DES key). Once shared, the sender may generate a HMAC by hashing the message with an algorithm such as MD5 or SHA-1, and then encrypting the hash with the preshared key via symmetric cipher such as DES.

The receiver hashes the plaintext locally and also decrypts the HMAC with his/her copy of the private key, recovering the sender's hash. If the two hashes match, the sender is authenticated, and the message's integrity is assured.

CBC-MAC

Cipher Block Chaining Message Authentication Code (CBC-MAC) uses CBC mode of a symmetric block cipher such as DES to create a message authentication code (MAC). A CBC-MAC provides integrity. CBC-MAC differs from HMAC because it uses one algorithm; HMACS use two: a hash algorithm such as SHA-1, followed by a symmetric block cipher such as DES or AES.

Public Key Infrastructure

Public Key Infrastructure (PKI) leverages all three forms of encryption to provide and manage *digital certificates*. A digital certificate is a public key signed with a digital signature. Digital certificates may be server-based (used for SSL Web sites such as https://www.ebay.com, for example) or client-based (bound to a person). If the two are used together, they provide mutual authentication and encryption. The standard digital certificate format is X.509.

Certificate Authorities

Digital certificates are issued by *Certificate Authorities* (CAs), which authenticate the identity of a person or organization before issuing a certificate to them. CAs may be private (run internally) or public (such as Verisign or Thawte). Anyone off the street cannot simply request and receive a certificate for www.ebay.com, for example; they must prove that they have the authority to do so. This authentication is done by the CA, and can include business records research, emails sent to domain contacts, and similar methods.

Certificate Revocation Lists

The Certification Authorities maintain *Certificate Revocation Lists* (CRL), which, as the name implies, list certificates that have been revoked. A certificate may be revoked if the private key has been stolen, an employee is terminated, etc.

IPsec

IPsec (Internet Protocol Security) is a suite of protocols that provide a cryptographic layer to both IPv4 and IPv6. It is one of the methods used to provide *Virtual Private Networks* (VPN), which allow you to send private data over an insecure network, such as the Internet (the data crosses a public network, but is "virtually private"). IPsec includes two primary protocols: *Authentication Header* (AH) and *Encapsulating Security Payload* (ESP). AH and ESP provide different, and sometimes overlapping functionality.

Supporting IPsec protocols include *Internet Security Association and Key Management Protocol* (ISAKMP) and *Internet Key Exchange* (IKE).

NOTE

This chapter describes the cryptographic aspects of IPsec: see Chapter 8 (Domain 7: Telecommunications and Network Security) for the network-related aspects of IPsec.

AH and ESP

Authentication Header provides authentication and integrity for each packet of network data. AH provides no confidentiality; it acts as a digital signature for the data. AH also protects against *replay attacks,* where data is sniffed off a network and resent, often in an attempt to fraudulently reuse encrypted authentication credentials.

Encapsulating Security Payload primarily provides confidentiality by encrypting packet data. It may also optionally provide authentication and integrity.

Security Association and ISAKMP

AH and ESP may be used separately or in combination. An IPsec Security Association (SA) is a simplex (one-way) connection which may be used to negotiate ESP or AH parameters. If two systems communicate via ESP, they use two SAs (one for each direction). If the systems leverage AH in addition to ESP, they use two more SAs, for a total of four. Each simplex SA connection is identified by a unique 32-bit number called the Security Parameter Index (SPI). The SA process is managed by the Internet Security Association and Key Management Protocol (ISAKMP).

Tunnel and Transport Mode

IPsec can be used in tunnel mode or transport mode. Tunnel mode is used by security gateways (which can provide point-to-point IPsec tunnels). ESP Tunnel mode encrypts the entire packet, including the original packet headers. ESP Transport mode only encrypts the data (and not the original headers); this is commonly used when the sending and receiving system can "speak" IPsec natively.

AH authenticates the original IP headers, so it is often used (along with ESP) in transport mode, because the original headers are not encrypted. Tunnel mode typically uses ESP alone (the original headers and encrypted, and thus protected, by ESP).

NOTE

IPsec is an example of a protocol built by committee, and that is not a compliment. It is overly complex, with multiple overlapping parts. Complexity is the enemy of security. See Bruce Schneier and Niels Ferguson's *A Cryptographic Evaluation of IPsec*, where they argue that AH mode and transport mode should be removed entirely: "Our main criticism of IPsec is its complexity. IPsec contains too many options and too much flexibility; there are often several ways of doing the same or similar things."[11] See: http://www.schneier.com/paper-ipsec.pdf

IKE

IPsec can use a variety of encryption algorithms, such as MD5 or SHA-1 for integrity, and triple DES or AES for confidentiality. The algorithm selection process is negotiated by the Internet Key Exchange. Two sides of an IPsec tunnel will typically use IKE to negotiate to the highest and fastest level of security, selecting AES over single DES for confidentiality if both sides support AES, for example.

SSL and TLS

Secure Sockets Layer (SSL) brought the power of PKI to the Web. SSL authenticates and provides confidentiality to Web traffic. *Transport Layer Security* (TLS) is the successor to SSL. They are commonly used as part of HTTPS (*Hypertext Transfer Protocol Secure*).

When you connect to a Web site such as https://www.isc2.org/, the data is encrypted. This is true even if you have not preshared a key: the data is encrypted out of the gate. This is done via asymmetric encryption: your browser downloads the digital certificate of www.isc2.org, which includes the site's public key, signed by the Certificate Authority's private key. If your browser trusts the CA (such as Verisign), then this signature authenticates the site: you know its isc2.org and not a rogue site. Your browser then uses that public key to securely exchange a symmetric session key. The private key is stored on the isc2.org Web server, which allows it to decrypt anything encrypted with the public key. The symmetric key is then used to encrypt the rest of the session.

The ciphers used for authentication, key exchange, and symmetric encryption are flexible: your browser will negotiate each with the server. Supported algorithms include (but are not limited to) RSA and Diffie-Hellman for key exchange, RSA and Digital Signature Algorithm (DSA) for authentication, and AES and triple DES for confidentiality.

SSL was developed for the Netscape Web browser in the 1990s. SSL 2.0 was the first released version; SSL 3.0 fixed a number of security issues with version 2. TLS was based on SSL 3.0. TLS is very similar to that version, with some security improvements. Although typically used for HTTPS to secure Web traffic, TLS may be used for other applications such as Internet chat and email client access.

PGP

Pretty Good Privacy (PGP) brought asymmetric encryption to the masses. It created a controversy when it was released by Phil Zimmerman in 1991. For the first time, an average computer user could easily leverage the power of asymmetric encryption, which allows strangers (including criminals) to securely communicate without presharing a key.

Zimmerman was investigated for munitions export violations by the United States government after the PGP source code was posted to the Usenet bulletin board system in 1991. The case was dropped by prosecutors in 1996. RSA complained to Zimmerman for including the (then) patented RSA algorithm in PGP. Zimmerman had encouraged users to pay RSA for a license if they used the algorithm. Zimmerman agreed to stop publishing PGP to address the patent issue (though copies were freely available from other sources).

PGP provides the modern suite of cryptography: confidentiality, integrity, authentication, and nonrepudiation. It can be used to encrypt emails, documents, or an entire disk drive. PGP uses a *Web of trust* model to authenticate digital

certificates, instead of relying on a central certificate authority (CA). If you trust that my digital certificate authenticates my identity, the Web of trust means you trust all the digital certificates that I trust. In other words, if you trust me, you trust everyone I trust.

S/MIME

MIME (Multipurpose Internet Mail Extensions) provides a standard way to format email, including characters, sets, and attachments. S/MIME (Secure/MIME) leverages PKI to encrypt and authenticate MIME-encoded email. The encryption may be done by the client or client's email server (called an S/MIME gateway).

Escrowed Encryption

Escrowed encryption takes a private key and divides it into two or more parts. The parts are held in escrow by different trusted third-party organizations, which will only release their portion of the key with proper authorization, such as a court order. One goal of escrowed encryption is to offer a balance between an individual's privacy, and the needs of law enforcement.

Clipper Chip

The Clipper Chip was the name the technology used in the Escrowed Encryption Standard (EES), an effort announced in 1993 by the United States government to deploy escrowed encryption in telecommunications devices. The effort created a media firestorm, and was abandoned by 1996.

The Clipper Chip used the Skipjack algorithm, a symmetric cipher that uses an 80-bit key. The algorithm was originally classified as secret. The secrecy of the algorithm was another controversial issue: secrecy of an algorithm does not provide cryptographic strength, and secret ciphers are often found to be quite insecure. Skipjack was later declassified in 1998 (after the Clipper Chip effort had been abandoned).

Steganography

Steganography is the science of hidden communication. The name is based on the Greek words steganos and graphein, which mean covered and write, or concealed writing. Encryption may provide confidentiality to a radio transmission, for example, but the communication itself is not hidden; only the meaning is concealed. Steganography hides the fact that communication is taking place.

The first use of steganography was documented by the ancient Greek historian Herodotus in the *Histories of Herodotus*. Herodotus described shaving a slave's head, tattooing instructions on it, waiting for the hair to grow back, and sending the slave across enemy lines. Another method hid a message inside a rabbit's stomach.

Modern steganography hides information inside data files, such as images. An 8-bit bitmap has 256 colors, for example. Say two different white pixels (called

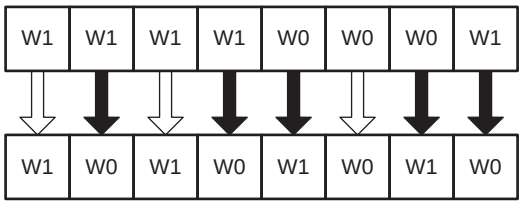


FIGURE 4.15

Steganographic substitution of bitmap pixels.

W0 and W1) in the image appear identical to the naked eye. You may encode a message by treating W0 and W1 as a bitstream.

Assume the file has a sequence of pixels in this order: W1, W1, W1, W1, W0, W0, W0, W1. You would like to encode "10101010" in the image. Treat W0 as binary 0, and W1 as binary 1. Then flip the pixels accordingly, resulting in W1, W0, W1, W0, W1, W0, W1, W0. Figure 4.15 shows the process. A white arrow means the pixel was unchanged; black arrows represent changed pixels.

The image now contains the hidden message "10101010," though it appears the same to the naked eye (and the size has not changed). The integrity of the image has changed. This method is called Substitution. Other methods include injection (add data to the file, creating a larger file) and new file creation. Substitution and Injection require a host file, new file creation creates a new file, as the name implies.

Messages that are hidden via steganography are often encrypted first, providing both confidentiality of the data and secrecy of the communication.

Digital Watermarks

Digital Watermarks encode data into a file. The watermark may be hidden, using steganography. Watermarks are often used to fingerprint files (tying a copy of a file to its owner).

LEARN BY EXAMPLE: *ACADEMY AWARD WATERMARKS*

An example of real-world digital watermark use is the watermarking of DVDs by the Academy of Motion Picture Arts and Sciences. Members of the academy (who decide the recipients of Oscar awards) receive DVD "screeners" of nominated films. The films are often still being shown in movie theaters and not yet available on DVD (publicly).

When the DVD system was first implemented, illegal copies of the screeners would appear on peer-to-peer filesharing networks. These copies were "ripped" (digitally copied) from the screeners.

In response, the Academy of Motion Picture Arts and Sciences began watermarking each screener. Each DVD is customized for the recipient: every frame of every DVD contains a hidden watermark, tying the DVD to the recipient. Should the DVD appear on a P2P network, the academy can track the copy down the source DVD (and member who received it).

In 2007, Salvador Nunez Jr. was arrested for posting the movie *Flushed Away* online, copied from an academy screener. Investigators used the watermark to track the copy to a screener received by his sister, who was a member of the academy.[12]

SUMMARY OF EXAM OBJECTIVES

Cryptography dates to ancient times, but is very much a part of our modern world, providing security for data in motion and at rest. Modern systems such as Public Key Infrastructure put all the cryptographic pieces into play via the use of symmetric, asymmetric, and hash-based encryption to provide confidentiality, integrity, authentication, and nonrepudiation. You have learned how the pieces fit together: slower and weaker asymmetric ciphers such as RSA and Diffie-Hellman are used to exchange faster and stronger symmetric keys such as AES and DES. The symmetric keys are used as session keys to encrypt short-term sessions, such as Web connections via HTTPS. Digital signatures employ public key encryption and hash algorithms such as MD5 and SHA-1 to provide nonrepudiation, authentication of the sender, and integrity of the message. Understanding these concepts and others discussed in this chapter and applying them together is critical for success on the exam.

SELF TEST

1. The RSA algorithm is based on which one-way function?
   - **A.** Elliptic curves
   - **B.** Discrete logarithm
   - **C.** Frequency distribution
   - **D.** Factoring composite numbers into their primes

2. Which of the following cryptographic methods is a monoalphabetic cipher?
   - **A.** Caesar Cipher
   - **B.** Vernam Cipher
   - **C.** Vigenère Cipher
   - **D.** Jefferson Disks

3. What type of encryption is proven to be unbreakable?
   - **A.** AES
   - **B.** ECC
   - **C.** One-time pad
   - **D.** RSA

4. Which AES function provides confusion by replacing one byte of the State with another?
   - **A.** AddRoundKey
   - **B.** MixColumns
   - **C.** ShiftRows
   - **D.** SubBytes

5. Which mode of DES is the weakest?
   - **A.** CFB
   - **B.** OFB

C. ECB
D. CDC

6. Which of the following cryptographic methods is primarily used to assure integrity?
A. One-way function
B. One-way hash
C. Diffusion
D. Confusion

7. Cryptography does not directly provide what?
A. Authentication
B. Confidentiality
C. Integrity
D. Availability

8. The Wassenaar Arrangement replaced what?
A. CPCOM
B. COCOM
C. CUCOM
D. CACOM

9. Nonrepudiation is best described as what?
A. Proving a user performed a transaction
B. Proving a transaction did not change
C. Authenticating a transaction
D. Proving a user performed a transaction that did not change

10. Which of the following is true for digital signatures?
A. The sender encrypts the hash with a public key
B. The sender encrypts the hash with a private key
C. The sender encrypts the plaintext with a public key
D. The sender encrypts the plaintext with a private key

11. Which algorithm should you use for a low-power device that must employ digital signatures?
A. AES
B. RSA
C. ECC
D. ElGamal

12. Which of the following is not required for a one-time pad to be unbreakable?
A. The characters must be truly random
B. The pads must be secure
C. The pads must not be reused
D. Each pad must be unique

13. Which of the following attacks analyzes large amounts of plaintext/ciphertext pairs created with the same key?
    - **A.** Known plaintext attack
    - **B.** Differential cryptanalysis
    - **C.** Linear cryptanalysis
    - **D.** Chosen plaintext attack
14. What is a Hashed Message Authentication Code (HMAC)?
    - **A.** Encrypting a hash with a symmetric cipher
    - **B.** Encrypting a hash with an asymmetric cipher
    - **C.** A message digest
    - **D.** A checksum
15. Which of the following was not an AES finalist?
    - **A.** MARS
    - **B.** RC6
    - **C.** Serpent
    - **D.** Blowfish

SELF TEST QUICK ANSWER KEY

1. **D**
2. **A**
3. **C**
4. **D**
5. **C**
6. **B**
7. **D**
8. **B**
9. **D**
10. **B**
11. **C**
12. **D**
13. **C**
14. **A**
15. **D**

References

1. http://netlab.cs.ucla.edu/wiki/files/shannon1949.pdf [accessed November 19, 2009].
2. Kahn D. *The Codebreakers*. New York, NY: McMillan Company; 1967.
3. http://pmchallenge.gsfc.nasa.gov/docs/2009/presentations/Emond.John.pdf [accessed November 19, 2009].
4. Singh S. *The Code Book*. New York, NY: Anchor Books; 2000.

5. http://www.nist.gov/public_affairs/releases/g01-111.htm [accessed November 19, 2009].
6. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf [accessed November 19, 2009].
7. http://csrc.nist.gov/groups/SMA/ispab/documents/minutes/2006-03/E_Barker-March2006-ISPAB.pdf [accessed November 19, 2009].
8. Schneier B. *Applied Cryptography*. New York, NY: Wiley; 1996.
9. http://www.schneier.com/crypto-gram-9806.html [accessed November 19, 2009].
10. Scott R. *Bladerunner*. Warner Bros; 1982.
11. http://www.schneier.com/paper-ipsec.pdf [accessed November 19, 2009].
12. http://news.cnet.com/8301-10784_3-6161586-7.html [accessed November 19, 2009].