Chapter 20: Cryptography

Introduction

Cryptography is the practice of securing communication and information through the use of mathematical algorithms, transforming readable data into an unreadable format to protect it from unauthorized access. It plays a crucial role in ensuring privacy, data integrity, and authentication in digital communications. Cryptographic systems use various techniques like encryption, decryption, hashing, and digital signatures to safeguard data, making it resistant to cyberattacks and unauthorized tampering. These systems are widely used in fields such as online banking, secure messaging, and digital contracts, forming the backbone of modern cybersecurity.

In this chapter, you will explore:

- Cryptography concepts and different encryption algorithms
- Applications of Cryptography
- Cryptanalysis Methods and cryptography attacks
- Cryptography attacks countermeasures

Cryptography Concepts and Encryption Algorithms

Cryptography is the foundation of secure communication, ensuring confidentiality, integrity, authentication, and non-repudiation of data. It involves techniques like symmetric encryption (e.g., AES, DES) where the same key is used for encryption and decryption, and asymmetric encryption (e.g., RSA, ECC) which uses a public-private key pair. Hashing algorithms (e.g., SHA-256, MD5) provide data integrity by generating fixed-size digests. Cryptographic protocols like TLS, IPSec, and PGP secure data in transit and at rest. Strong encryption algorithms play a crucial role in protecting sensitive information from cyber threats, making cryptography a critical component of modern cybersecurity frameworks.

Cryptography

Cryptography is a technique for encrypting clear text data into scrambled code. The encrypted data is then sent over a public or private network to its destination to ensure confidentiality. This encrypted data, known as "Ciphertext," is decrypted at the destination for processing. Strong encryption keys are used to avoid key cracking. The objective of cryptography is not purely about confidentiality; it also concerns integrity, authentication, and non-repudiation.

Types of Cryptography

Symmetric Cryptography

Symmetric Key Cryptography is the oldest and most widely used cryptography technique in the domain of cryptography. Symmetric ciphers use the same secret key for the encryption and decryption of data. The most widely used symmetric ciphers are AES and DES.

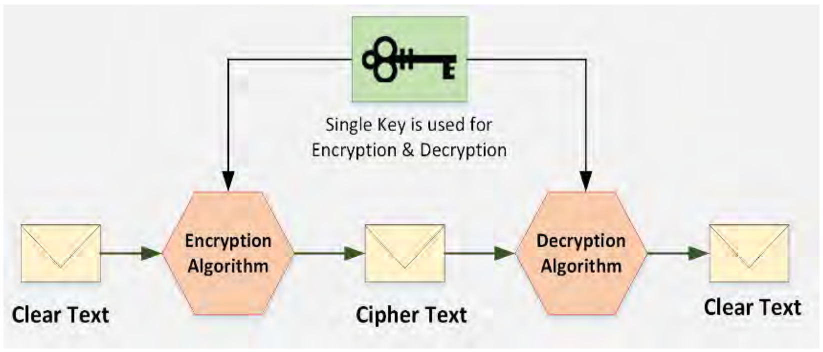


Figure 20-01: Symmetric Cryptography

Asymmetric Cryptography/Public Key Cryptography

Unlike Symmetric Ciphers, in Asymmetric Cryptography, two keys are used. Everyone publicly knows one key, while the other key is kept secret and is used to encrypt data by the sender; hence, it is also called Public Key Cryptography. Each sender uses its secret key (also known as a Private Key) for encrypting its data before sending it. The receiver uses the respective sender's public key to decrypt the data. RSA, DSA, and the Diffie-Hellman Algorithm are popular examples of asymmetric ciphers. Asymmetric key cryptography delivers confidentiality, integrity, authenticity, and non-repudiation by using the public and private key concepts. The private key is only known by the owner itself, whereas the public key is issued by Public Key Infrastructure (PKI), where a trusted Certificate Authority (CA) certifies the ownership of key pairs.

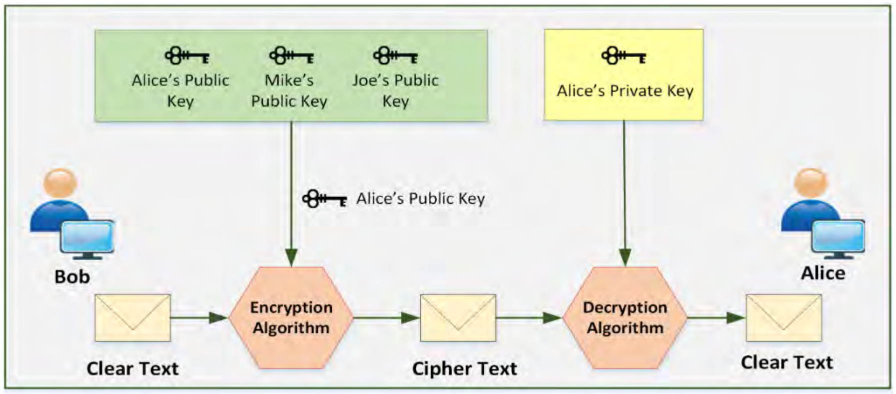


Figure 20-02: Asymmetric Cryptography

Government Access to Keys (GAK)

Government Access to Keys (GAK) is a legislative obligation for individuals and organizations to disclose their cryptographic keys to government agencies. These keys are used by law enforcement to monitor suspicious communication and collect evidence of cybercrimes. Government agencies promise to secure the keys and only use them when a court issues a warrant. Key escrow is used for uninterrupted access to keys, with a third party allowing others to use them under specific circumstances. However, there is growing concern about privacy and security of cryptographic keys and information. Government agencies often use a single key to protect other keys, making it difficult to determine the necessary level of protection.

Ciphers

A cipher is an algorithm used in cryptography for encryption and decryption. Transforming plaintext into a cipher or code is known as encipherment; decipherment is the opposite. If a message is encrypted with a cipher, it cannot be decrypted unless the recipient has the secret key. Cell phones and the Internet are examples of communication technology that depend on ciphers to preserve privacy and security. Cipher algorithms can be classified as either closed-source (the algorithm itself is not in the public domain and the process is created for usage in particular domains, like the military) or open-source (the algorithmic process is in the public domain while the key is private and chosen by the user). Additionally, ciphers can be licensed or freely used by the general public.

Thousands of cipher algorithms are available on the internet. Some of them are proprietary, while others are open source. The following are the common methods by which ciphers replace original data with encrypted data.

Types of Ciphers

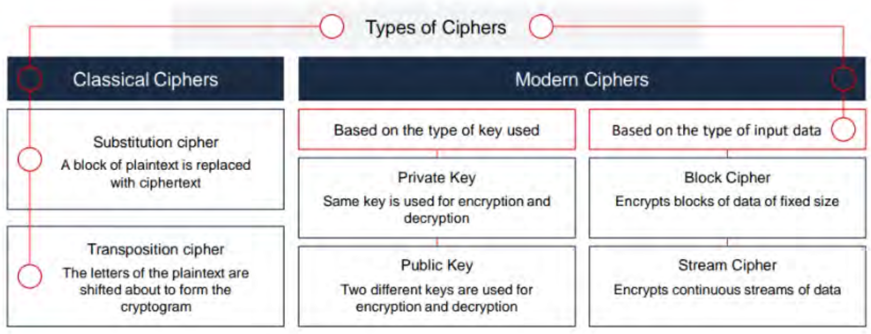

Types of Ciphers
Classical Ciphers
Modern Ciphers
Substitution cipher
A block of plaintext is replaced with ciphertext
Transposition cipher
The letters of the plaintext are shifted about to form the cryptogram
Based on the type of key used
Private Key
Same key is used for encryption and decryption
Public Key
Two different keys are used for encryption and decryption
Based on the type of input data
Block Cipher
Encrypts blocks of data of fixed size
Stream Cipher
Encrypts continuous streams of data

Table 20-01: Types of Ciphers

Substitution

In this method, every single character of data is substituted with another character. A very simple example in this regard would be to replace a character by shifting it three characters along. Here, “D” would replace “A” and so on. To make it more complex, we can select certain letters to be replaced in the whole text. In our example, the value of the key is three, and both nodes should know that value. Otherwise, they would not be able to decrypt the data.

Transposition

This method makes substitution even more difficult to break by using multiple character substitution.

Keys

In the above example of substitution, we used a key of “three”. Keys play the main role in every cipher algorithm. Without knowing the key, data cannot be decrypted.

Types of Modern Ciphers

Based on the type of key used the modern cipher can be divided into:

- Symmetric-key algorithms (Private-key cryptography): Use the same key for encryption and decryption.
- Asymmetric-key algorithms (Public-key cryptography): Use two different keys for encryption and decryption.

Stream Cipher

A Stream Cipher is a type of symmetric-key cipher that encrypts plain text one by one. There are various types of stream ciphers, for example, synchronous, asynchronous, etc. RC4 is the most common type of stream cipher design. The transformation of encrypted output varies during the encryption cycle.

Block Cipher

This is a type of symmetric-key cipher that encrypts plain text by processing the fixed-length blocks. The transformation of encrypted data does not vary in a block cipher. It encrypts the block of data using the same key on each block. DES and AES are common types of block cipher design.

Symmetric Encryption Algorithms

Symmetric encryption algorithms use a single secret key for both encryption and decryption, making them efficient for securing large amounts of data. They are commonly used in confidentiality protection for files, databases, and network communications.

Data Encryption Standard (DES)

Data Encryption Standard (DES) algorithm is a symmetric key algorithm used for encryption that is now considered insecure. However, successors such as Triple-DES and G-DES have replaced DES encryption. DES uses a 56-bit key size that is too small to protect data.

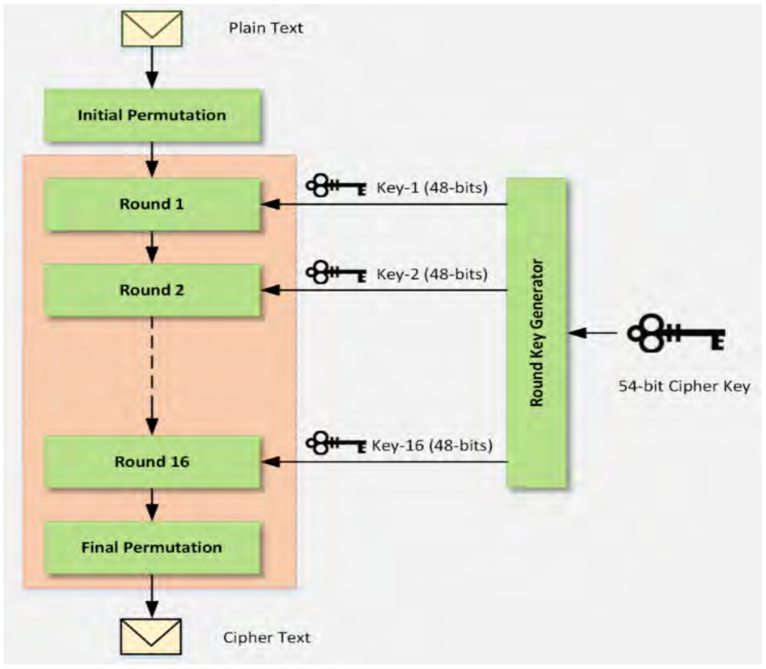


Figure 20-03: DES Algorithm

The DES algorithm consists of 16 rounds, which process the data with 16 intermediary round keys of 48 bits. These intermediary keys are generated from 56-bit cipher keys by a Round Key Generator. Similarly, a DES reverse cipher computes the data in clear text format from ciphertext using the same cipher key.
The following are the major parameters of DES:

DES Algorithm Parameters	Values
Block Size	64 bits
Key Size	56 bits
Number of Rounds	16
16 Intermediary Keys	48 bits

Table 20-02: DES Algorithm Parameters

Triple Data Encryption Standard (3DES)

The U.S. Federal Government initiated a contest to find a replacement cryptography algorithm for DES, which was eventually replaced by 3DES. 3DES uses a "key bundle" consisting of three DES keys, K1, K2, and K3. Each key is a standard 56-bit DES key. The process involves DES encrypting with K1, DES decrypting with K2, and DES encrypting with K3. There are three options for the keys: independent, identical, or the same. The first option is the most secure, while the third option is the least secure.

Advanced Encryption Standard (AES)

When DES becomes insecure and performing DES encryption three times (3-DES or Triple-DES) takes high computation and time, another encryption algorithm is needed that is more secure and effective. Rijndael issued a new algorithm in 2000-200 1 known as the Advanced Encryption Algorithm (AES). AES is also a private key symmetric algorithm, but it is stronger and faster than Triple-DES. AES can encrypt 128-bit data with 128/192/256-bit keys.
The following are the major parameters of AES.

AES Algorithms Parameters	AES- 128	AES- 192	AES-256
Block Size	4 / 16 / 128 bits	6 / 24 / 192 bits	8 / 32 / 256
Key Size	4 / 16 / 128 bits	4 / 16 / 128 bits	4 / 16 / 128 bits
Number of Rounds	10	12	14
Round Key Size	4 / 16 / 128 bits	4 / 16 / 128 bits	4 / 16 / 128 bits
Expanded Key Size	44 / 176 bits	52 / 208	60 / 240

Table 20-03: AES Algorithm Parameters

To understand the AES algorithm, consider an AES 128-bit scenario. In 128-bit AES, there will be 10 rounds. The initial 9 rounds perform the same step, i.e., substitute bytes, shift rows, mix columns, and add round keys. The last round is slightly different, with only substitute bytes, shifting rows, and adding round keys. The figure 20-04 shows the AES algorithm architecture.

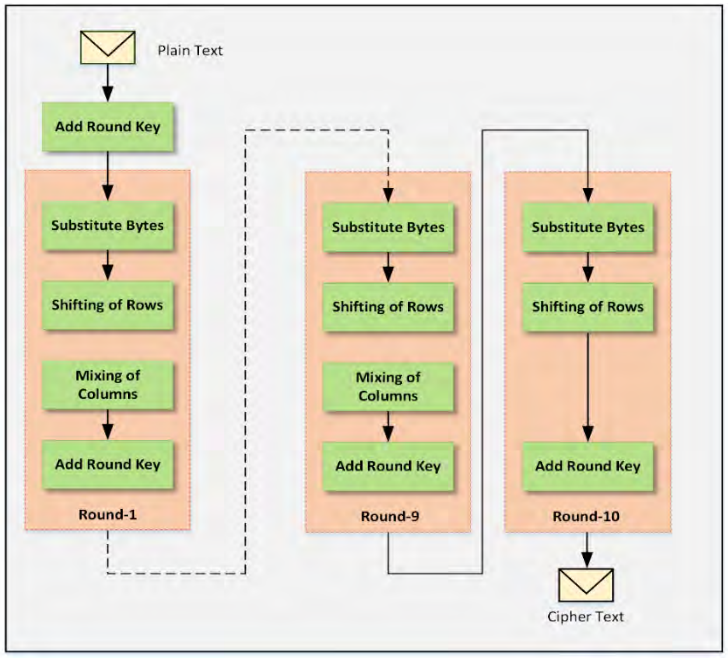


Figure 20-04: AES Algorithm

RC4, RC5, RC6 Algorithms

RC4 is an older encryption technique designed in 1987 by Ron Rivest based on stream cipher. RC4 is used in SSL and WEP protocols. RC4 generates a pseudorandom stream used for encrypting plain text by bit-wise exclusive-or (similar to the Vernam cipher except for the generated pseudorandom bits). Similarly, the process of decryption is performed as it is a symmetric operation. In the RC4 algorithm, a 24-bit Initialization Vector (IV) generates a 40- or 128-bit key.

RC5 is a symmetric key block cipher introduced in 1994. RC5 has variable block sizes (32, 64, or 128 bits) with a key size of 0 to 2040 bits and 0 to 255 rounds. It is suggested that RC5 is used with the 64-bit block size, 128-bit key, and 12 rounds. RC5 also consists of some modular additions and exclusive OR (XOR)s.

RC6 is also a symmetric key block cipher that is derived from RC5 with a block size of 128 bits with 128-,192-,256-, and up to 2040-bit key support. RC6 is very similar to RC5 in structure, using data-dependent rotations, modular addition, and XOR operations. RC6 does use an extra multiplication operation not present in RC5 to make the rotation dependent.

Blowfish

Blowfish is a symmetric block cipher algorithm designed to replace DES or IDEA algorithms. It uses the same secret key to encrypt and decrypt data, splitting data into 64-bit blocks and producing a key ranging from 32 bits to 448 bits. Blowfish is a 16-round Feistel cipher working on 64-bit blocks and is used in software like password protection tools and e-commerce websites for secure payments. The algorithm has two parts: key expansion and data encryption. The first part breaks the original key into subkeys, dividing it into 4,168 bytes. The P-array and S-boxes contain 18 and 256 entries, respectively, ensuring high speed and efficiency.

Twofish

Twofish is a 128-bit block cipher that was one of the U.S. Government's five finalists for replacing DES. It is a Feistel cipher, using a single key for encryption and decryption up to 256 bits. It is fast for CPU or hardware, flexible for network-based applications, and allows various performance trade-offs on parameters like encryption speed and memory usage.

Threefish

Threefish, developed in 2008 as part of the Skein algorithm, is a large, tweakable symmetric-key block cipher with equal block and key sizes i.e., 256, 512, and 1024. It uses ARX (addition-rotation-XOR) operations and 64-bit words, requiring 72, 72, and 80 rounds for security.

Serpent

Serpent, a symmetric-key block cipher, was a finalist in the AES contest. It uses 128-bit symmetric block ciphers. It involves 32 rounds of computational operations and 8-variable S-boxes. Despite being secure, researchers chose Rijndael over Serpent due to its moderate speed and complexity. Rijndael minimizes correlation between encoded images or plaintexts, making it a standout competitor in AES.

TEA

The Tiny Encryption Algorithm (TEA) was developed by David Wheeler and Roger Needham in 1994 and is a Feistel cipher that uses 64 rounds. It is a simple algorithm that can be implemented in code and uses a 128-bit key on a 64-bit block. The key is divided into four 32-bit subkeys, K[0], K[1], K[2], and K[3]. TEA uses addition and subtraction with mod 232 instead of XOR. The block is divided into two halves, R and L, and R is processed through the round function. The round function performs a left shift of 4, adds the result to K[0], delta, and then shifts the result right by 5 and adds it to K[1]. The round function is then XORed with L, and L and R are swapped for the next round. The number of rounds should be even since they are implemented in pairs called cycles.

CAST-128

CAST-128, also known as CAST5, is a symmetric-key block cipher with a classical Feistel network and a block size of 64 bits. It uses a key size ranging from 40 to 128 bits in 8-bit increments. CAST-128's components include large 8x32-bit S-boxes, modular addition and subtraction, key-dependent rotation, and XOR operations. It uses a masking key and rotation key for its functions, and is used as a default cipher in GPG and PGP.
CAST-256 is an extension of CAST-128 that uses the same design procedure. CAST-256 has a 128-bit block size and uses key sizes varying from 128 to 256 bits. Furthermore, it uses zero-correlation cryptanalysis, which can break 28 rounds with time = 2246.9 and data = 298.8.

Ghost Block Cipher

The GOST (Government Standard) block cipher, also known as Magma, is a symmetric-key cipher with a 32-round Feistel network working on 64-bit blocks with a 256-bit key length. It consists of an S-box that can be kept secret and contains around 354 bits of secret information. GOST is an encryption algorithm that uses a 32-bit subkey modulo 2^32 and a rotate left shift operation to shift 11 bits. The key scheduling is performed by breaking the 256-bit key into eight 32-bit subkeys, used in order for the first 24 rounds and reversed for the last 8 rounds.

Camellia

Camellia is a symmetric-key block cipher with a block size of 128 bits and key sizes of 128, 192, and 256 bits. It uses four 8x8-bit S-boxes for affine transformations and logical operations, with a logical transformation layer applied every six rounds. Camellia uses the key whitening technique for increased security and is part of the Transport Layer Security (TLS) protocol. Despite its smaller key size of 128 bits, Camellia cannot be brute-forced, making it a safe cipher. Its processing skills are equivalent to those of AES or Rijndael.

Asymmetric Encryption Algorithms

Asymmetric encryption, also known as public-key cryptography, uses a pair of keys: a public key for encryption and a private key for decryption. This eliminates the need to share secret keys and enhances security, making it ideal for secure communication, digital signatures, and key exchange.

DSA and Related Signature Schemes

A signature, just as it is used in daily life, proves authenticity and proves the actual origin of a document. In computer networking, the Digital Signature Algorithm (DSA) is used to sign a digital document. A Digital Signature can provide three components of network security, i.e., the authenticity of a message, integrity of a message, and non-repudiation. A digital signature cannot provide confidentiality of communication. However, this can be achieved by using encrypted messages and signatures.

A digital signature uses a public key to sign and verify packets. The signing of a document requires a private key, whereas verification requires a public key. The sender of a message signs it with his/her private key and sends it to the receiver. The receiver verifies the authenticity of the message by decrypting the packet with the sender's public key, as the sender's public key only decrypts the message and verifies the sender of that message.

The integrity of a message is preserved by signing the entire message. If any content of the message is changed, it will not get the same signature. In a nutshell, integrity is the process of signing and verifying a message obtained by using Hash Functions.

A Digital Certificate contains various items listed below:

- **Subject:** The certificate holder's name
- **Serial Number:** A unique number for certificate identification
- **Public Key:** A copy of the certificate holder's public key
- **Issuer:** A certificate issuing authority's digital signature to verify that the certificate is real
- **Signature Algorithm:** An algorithm used by the Certificate Authority (CA) to sign a certificate digitally
- **Validity:** Validity of a certificate, or expiry date and time, of the certificate

A Digital Certificate has X.509 version supported format, which is the standard format.

Note: Certificate validation determines whether the certificate and public key it contains are trustworthy. A Certificate Authority completes the verification process.

RSA (Rivest Shamir Adleman)

This algorithm is named after its creators, Ron Rivest, Adi Shamir, and Leonard Adleman. Also known as Public Key Cryptography Standard (PKCS) # 1, the main purpose of its use today is authentication. RSA key length varies from 1024 to 4096 bits. The longer the key, the more secure it is, but also the slower it is to perform cryptographic operations.RSA is one of the de-facto encryption standards.

The RSA Signature Scheme

1. Two very large prime numbers, "p" and "q," are required.
2. Multiply the above two primes to find n, the modulus for encryption and decryption. In other words, n = p * q.
3. Calculate ϕ = (p - 1) * (q - 1).
4. Choose a random integer "e", i.e., Encryption Key. Calculate "d" (Decryption Key) so that d x e = 1 mod ϕ.
5. Announce "e" and "n" to the public while keeping "ϕ" and "d" secret.

Diffie–Hellman

Is a cryptographic protocol that allows two parties to establish a shared key over an insecure channel. It was developed and published by Whitfield Diffie and Martin Hellman in 1976. Actually, it was independently developed a few years earlier by Malcolm J. Williamson of the British Intelligence Service, but it was classified at that time.

Diffie–Hellman Algorithm

The system has two parameters called p and g

- Parameter p is a large prime number
- Parameter g (usually called a generator) is an integer less than p, with the following property: for every number n between 1 and p-1 (both inclusive), there is a power k of g such that n = g kmod p

Many cryptography textbooks use the fictitious characters "Alice" and "Bob" to illustrate cryptography; we will do the same here as well

- Alice generates a random private value a, and Bob generates a random private value b. Both a and b are drawn from the set of integers
- They derive their public values using parameters p and g and their private values. Alice's public value is ga mod p, and Bob's public value is gb mod p
- They exchange their public values
- Alice computes gab = (gb)a mod p, and Bob computes gba = (ga)b mod p
- Since gab = gba = k, Alice and Bob now have a shared secret key k

Elliptic Curve Cryptography (ECC)

ECC is a modern public-key cryptography developed to avoid using larger cryptographic keys. The asymmetric cryptosystem depends on number theory and mathematical elliptic curves (algebraic structure) to generate short, quick, and robust cryptographic keys. RSA is an incumbent public-key algorithm, but its key size is large. The speed of encryption always depends on the key size: a smaller key length allows faster encryption. To minimize the key size, elliptic curve cryptography has been proposed as a replacement for the RSA algorithm.

he operational key sizes of both algorithms to achieve similar goals are listed below:

ECC Key Size (bits)	Equivalent RSA Key Size (bits)
160–223	1024
224-255	2048
256-383	3072
384-511	7680
512+	15360

Table 20-4: Comparison of ECC and RSA key size

While RSA uses a key size of 1024 to encrypt the data, ECC provides equal security with a comparatively smaller key size ranging between 160 to 223. For high-level computing, RSA uses a key size of 7680 to implement security, whereas ECC can provide the same level of security with a key size ranging between 384 to 511.

YAK

YAK is a public-key-based Authenticated Key Exchange (AKE) protocol. The authentication of YAK is based on public key pairs, and it needs PKI to distribute authentic public keys. YAK is a variant of the two-pass Hashed Menezes-Qu-Vanstone (HMQV) protocol using zero-knowledge proofs (ZKP) for proving the knowledge of ephemeral secret keys from both parties. The YAK protocol lacks joint key control and perfect forward secrecy attributes.

The YAK protocol implementation between two parties Alice and Bob is described as follows:

1. Alice chooses a random number x such that x∈ R[0, q – 1], computes X = gx, and generates ZKP of x, denoted by KP{x}. Alice sends X and KP{x} to Bob.
2. Bob chooses a random number y such that y∈ R[0, q – 1], computes Y = gy, and generates ZKP of y, denoted by KP{y}. Bob sends Y and KP{y} to Alice.
3. Alice verifies the received KP{x} and computes the session key after verification as k = H ((Y.PKB)x + a), where H is a hash function.
4. Bob verifies the received KP{y} and computes the session key after verification as k = H ((X.PKA)y + b).
5. They authenticate each other, and both obtain the same session key k = H (g(x + a)(y + b)).

<u>Message Digest (One-Way Hash) Functions</u>

The Message Digest is a cryptographic hashing technique used to ensure the integrity of a message. Message and message digest can be sent together or separately through a communication channel. A receiver recalculates the hash of the message and compares it with the message digest to ensure no changes have been made. One-Way-Hashing of a message digest means the hashing function must be a one-way operation. The original message must not be able to be recreated. The message digest is a unique fixed-size bit string that is calculated in a way that if a single bit is modified, it changes 50% of the message digest value.

Message Digest Function: MD5 and MD6

The MD2, MD4, MD5, and MD6 algorithms are message digest series. MD5 produces a 128-bit hash value used as a checksum to verify integrity. Hashing is the technique for ensuring integrity. The hash value is calculated by computing specific algorithms to verify the integrity of data to ensure it was not modified. Hash values play an important role in proving integrity not only of documents and images but also in protocols to ensure the integrity of a transporting payload. However, MD5 is not collision-resistant; therefore, it is better to use the latest algorithms, such as MD6, SHA-2, and SHA-3

Secure Hashing Algorithm (SHA)

A Message Digest 5 (MD5) is a cryptographic hashing algorithm. Another more popular, secure, and widely used hashing algorithm is the Secure Hashing Algorithm (SHA). SHA- 1 is a secure hashing algorithm producing a 160-bit hashing value compared to MD5, which produces a 128-bit value. However, SHA-2 is now an even more secure, robust, and safer hashing algorithm.

```
Syntax: The password is 12345

SHA- 1: 567c552b6b559eb6373ce55a43326ba3db92dcbf
```

Secure Hash Algorithm 2 (SHA-2)

SHA2 can vary a digest between 224 bits and 512 bits. SHA-2 is a group of different hashes, including SHA-256, SHA-384, and SHA 5 12. The stronger cryptographic algorithm will minimize the chances of compromise.

SHA-256

```
Syntax: The password is 12345

SHA-256: 5da923a6598f034d9 1f375f73 143b2b2f58be8a 1c94 17886d5966968b7f79674
```

SHA-384

```
Syntax: The password is 12345

SHA-384: 929f4c 12885cb73d05b90dc825f70c2de64ea72 1e 15587deb3430999 1f6d57 1 14500465243ba08a554f8fe7c8dbbca04
```

SHA-5 12

```
Syntax: The password is 12345

SHA-5 12:

1d967a52ceb7383 16e85d94439dbb 1 12dbcb8b7277885b76c849a80905ab370dc 1 1d2b84dcc88d6 1393 1 17de483a950ee253fba0d26b5b 168744b94af2958 145
```

IPEMD-160

RACE Integrity Primitives Evaluation Message Digest (RIPEMD) is a 160-bit hash algorithm developed by Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. There exist 128-, 256-, and 320-bit versions of this algorithm, called RIPEMD-128, RIPEMD-256, and RIPEMD-320, respectively. These algorithms replace the original RIPEMD, which was found to have a collision issue. They do not follow any standard security policies or guidelines. RIPEMD-160 is a more secure version of the RIPEMED algorithm. In this algorithm, the compression function consists of 80 stages, i.e., 5 blocks that execute 16 times each. This process repeats twice by combining the results at the bottom using modulo 32 addition.

Hashed Message Authentication Code (HMAC)

HMAC uses the mechanism of hashing but adds the further feature of using a secret key in its operation. Both peers only know this secret key. Therefore, in this case, only parties with secret keys can calculate and verify the hash. By using HMAC, if there is an attacker eavesdropping, he/she will not be able to inject or modify the data and recalculate the correct hash because he/she will not know the correct key used by HMAC.

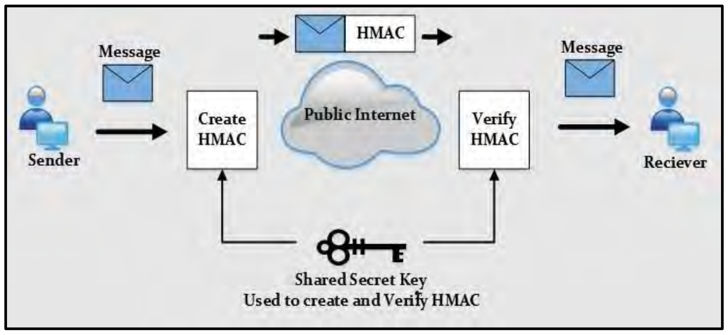


Figure 20-05: HMAC Working Conceptual Diagram

GOST – Hash Function

This hash algorithm was initially defined in the Russian national standard GOST R 34.11-94 "Information Technology - Cryptographic Information Security - Hash Function." It produces a fixed-length output of 256 bits. The input message is broken up into chunks of 256-bit blocks. If a block is less than 256 bits, then the message is padded by appending as many zeros to it as are required to make the length of the message 256 bits. The remaining bits are filled with a 256-bit integer arithmetic sum of all previously hashed blocks. Then, a 256-bit integer representing the length of the original message, in bits, is produced.

Message Digest Function Calculators

Several MD5 calculating tools are available that can directly calculate the hash value of text as well as offers to upload the desired file. Some of the most popular tools are:

1. HashCalc
2. MD5 Calculator
3. HashMyFiles

Multilayer Hashing Calculators

Multilayer hashing, also known as nested hashing or recursive hashing, is a technique that applies a hash function multiple times to the input or the output of a previous hashing operation. This method enhances security and creates more complex hash structures. Tools like CyberChef can facilitate multilayer hashing. By adding additional layers of complexity, this process makes it more difficult for attackers to reverse-engineer the original input data from the hash value, thereby increasing the difficulty of brute-force attacks.

Working of Multilayer Hashing

- **Initial Hashing:** The original input data (such as a message or file) is hashed using a cryptographic hash function, such as SHA-3 or MD5. The result is a fixed-size hash value, typically represented as a string of characters.
- **Subsequent Hashing:** The hash value obtained from the initial hashing step is then hashed again using the same or a different hash function. This process can be repeated multiple times to create several layers of hashing.
- **Final Hash Value:** After a predetermined number of hashing iterations, the final hash value is produced. This value serves as the final output of the multilayer hashing process.

Hardware-Based Encryption

Hardware-based encryption is a method that utilizes computer hardware to assist or replace software during the data encryption process. Devices that implement encryption techniques can be classified as hardware-based encryption devices. In this approach, the workload of cryptographic processes is shifted to hardware processors, freeing up system resources for other functions.

These devices can also securely store encryption keys and sensitive information in protected areas of RAM or non-volatile storage such as flash memory. Hardware encryption devices minimize instruction sets, allowing only authorized code to be executed. By not supporting third-party software, they help prevent the execution of malicious programs.

Hardware-based encryption offers several advantages over software encryption, including faster processing of algorithms, tamper-resistant key storage, and protection against unauthorized code execution. Examples of hardware-based encryption devices include wireless access points, Nitrokey, credit card terminals, and network bulk encryptors.

Quantum Cryptography

As the world increasingly embraces online information sharing, the number of security attacks on cryptosystems is rising sharply. Traditional mathematical encryption relies on binary digits (0 and 1), making it vulnerable to eavesdropping and manipulation through various techniques. To address these vulnerabilities, quantum cryptography has been introduced to protect data against midpoint thefts, such as man-in-the-middle (MITM) attacks.

Quantum cryptography is based on the principles of quantum mechanics and employs quantum key distribution (QKD) using photons instead of mathematical algorithms for encryption. In this approach, data elements are encrypted using a sequence of photons that possess a spinning trait as they travel from one point to another. These photons change their orientations while passing through filters, which can be vertical, horizontal, forward slash, or backslash. In this system, vertical and backslash spins represent "ones," while horizontal and forward slash spins represent "zeros."

- Horizontal (–): 0
- Vertical (|): 1
- Backslash (/): 1
- Forward slash (\): 0

Attackers can eavesdrop on the data but cannot manipulate it because photons are transmitted through arbitrary filters. To bypass this protection, attackers must know the exact characteristics of the photons; if they fail to select the correct transmission, the polarization of the photons becomes distorted. This distortion alerts the receiver to an error, indicating that eavesdropping has occurred.

<u>Other Encryption Techniques</u>

Homomorphic Encryption

Homomorphic encryption is distinct from conventional encryption methods, as it allows mathematical operations to be performed on encrypted data without needing to decrypt it first. This technique enables users to keep their data secure and in an encrypted format, even while it is being processed or manipulated. The same key holder manages both the encryption and decryption processes. Homomorphic encryption allows users to encrypt their confidential data and outsource it to a cloud service provider for processing while maintaining its security.

Post-quantum Cryptography

Post-quantum cryptography, also known as quantum-resistant or quantum-proof cryptography, refers to advanced algorithms—primarily public-key based—that are designed to protect security systems against attacks from both conventional and quantum computers. This type of cryptography can work alongside existing communication protocols and network operations. Additionally, post-quantum cryptography can function as a stand-alone encryption method, capable of replacing current vulnerable cryptosystems while adhering to standard security policies.

The goal of post-quantum cryptography is to ensure secure communication across a wide range of applications, including secure secret-key processing, public-key-based signatures, and public-key encryption for high-security activities such as secure e-voting. It encompasses several low-cost, secure systems typically used for online communications. Ultimately, post-quantum cryptography aims to prepare for the era of quantum computing by updating specific algorithms and standards.

Lightweight Cryptography

One of the major challenges in current cryptography techniques is their application in low-powered devices. Researchers are working to develop a compact, quantum-safe algorithm that can function effectively on these devices. While most existing cryptographic algorithms are designed for servers and desktops, lightweight cryptographic algorithms are tailored for low-complexity applications such as RFID tags, sensor-based systems, and various IoT applications. The primary goal of developing lightweight cryptography is to reduce power consumption and resource usage without compromising security.

Cipher Modes of Operations

Cipher modes of operation, also known as block cipher modes, are techniques used to encrypt a fixed block of plaintext by utilizing a secret key. In some modes, an initialization vector is also employed. These modes play a crucial role in ensuring the confidentiality and authenticity of data. To facilitate encryption and decryption, the client and server exchange a securely encrypted symmetric key. Below, we discuss four block cipher modes of operation that illustrate how encryption occurs on the source side and decryption on the destination side.

Electronic Code Book (ECB) Mode

The ECB (Electronic Codebook) mode is a simple process for encrypting and decrypting data that involves plaintext, a secret key, and a block cipher encryption algorithm. In this method, the plaintext is divided into fixed-length blocks that match the size of the secret key.

The encryption begins by taking the first block of plaintext and using the secret key as input for the block cipher encryption algorithm, which produces the first block of ciphertext. This process is repeated for all blocks of plaintext.

On the receiving end, decryption is carried out in the same way. The secret key is used as input for the block cipher decryption algorithm, which yields the first block of plaintext. This decryption process is then repeated for all ciphertext blocks.

However, ECB mode has a notable drawback: if two or more plaintext blocks are identical, their corresponding ciphertext blocks will also be identical. This consistency can give analysts clues about the plaintext, making it easier to predict the original data.

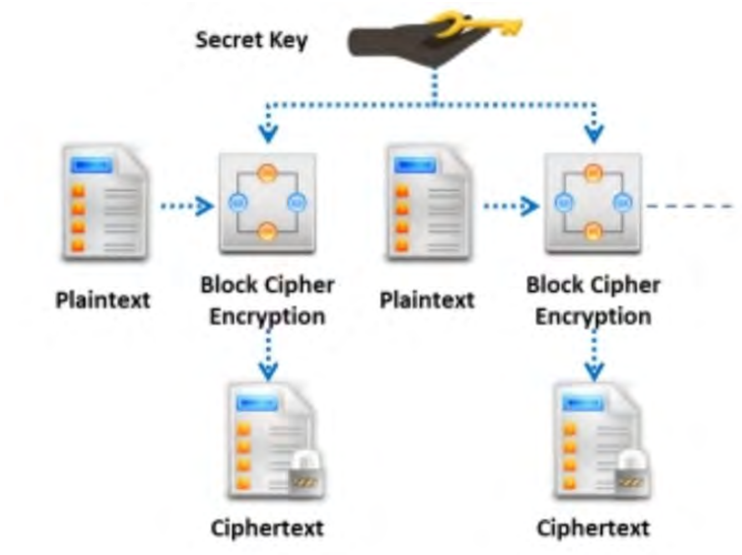


Figure 20-06: Electronic Code Book (ECB) Mode Encryption

Cipher Block Chaining (CBC) Mode

The Cipher Block Chaining (CBC) mode of encryption utilizes an initialization vector (IV) and a secret key for the encryption process. Initially, the plaintext is divided into blocks of the same size. The first block is XORed with the initialization vector (IV), and the result is fed into the block cipher encryption algorithm along with the secret key. This produces the first block of ciphertext.

Next, this ciphertext block is used to perform XOR with the following plaintext block, and this chaining process continues until all plaintext blocks have been processed.

On the receiving end, the first block of ciphertext and the secret key are sent to the block cipher decryption algorithm. The output is then XORed with the same IV to retrieve the first block of plaintext. For the subsequent ciphertext blocks, the previously generated ciphertext block is used in place of the IV for the XOR operation. This process continues for all the remaining ciphertext blocks.

However, this mode has a drawback: if any generated ciphertext block contains an error, that error will propagate to the subsequent ciphertext blocks.

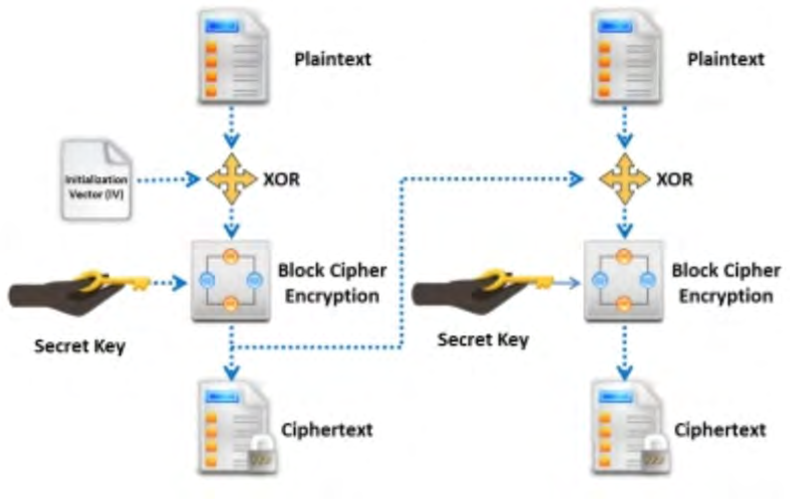


Figure 20-07: Cipher Block Chaining (CBC) Mode Encryption

Cipher Feedback (CFB) Mode

In the CFB (Cipher Feedback) mode, previously generated ciphertext is used as feedback for the encryption algorithm to encrypt the next plaintext block. The process begins by storing the initialization vector (IV) in a shift register, which is then combined with a secret key and sent to the encryption algorithm. From the result of this encryption, the first S bits are selected, and an XOR operation is performed with a plaintext block of size S to produce the ciphertext block.

For the next encryption block, the most recently generated ciphertext is used as input for the shift register, which shifts S bits to the left. This process is repeated until the entire plaintext has been processed.

On the receiving end, the decryption process mirrors this method up to the XOR operation. The first S bits from the output of the encryption algorithm are XORed with the first ciphertext block, resulting in the first plaintext block. For subsequent blocks, the previously used ciphertext is fed into the shift register, and the process continues until the last ciphertext block is decrypted.

One advantage of CFB mode is that it complicates cryptanalysis due to data loss that occurs with the use of shift registers.

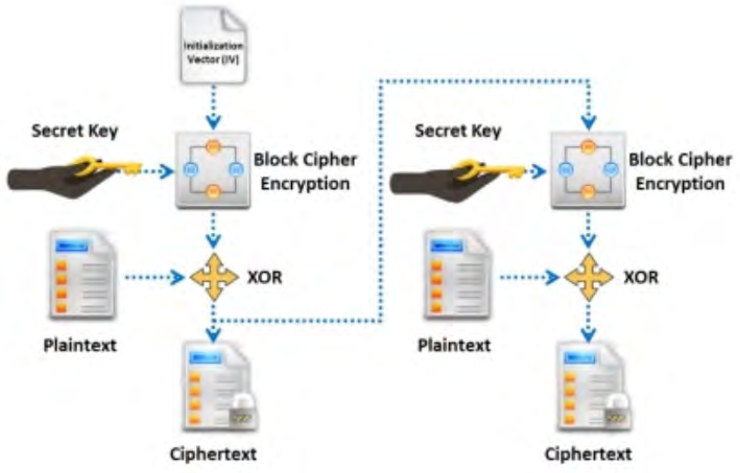


Figure 20-08: Cipher Feedback (CFB) Mode Encryption

Counter Mode

The counter mode is a block cipher operation that uses a counter value for encryption and decryption. A counter is initialized and input along with a secret key into the block cipher, and the result is XORed with a block of plaintext to produce ciphertext. This process is repeated for all plaintext blocks.

On the receiving end, the same counter values and secret keys are used. The counter value is encrypted again, and the resulting output is XORed with the ciphertext block to retrieve the original plaintext.

A key advantage of the counter mode is that it avoids error propagation, as it does not depend on previously generated ciphertext. However, synchronized counter values must be maintained on both the source and destination sides.

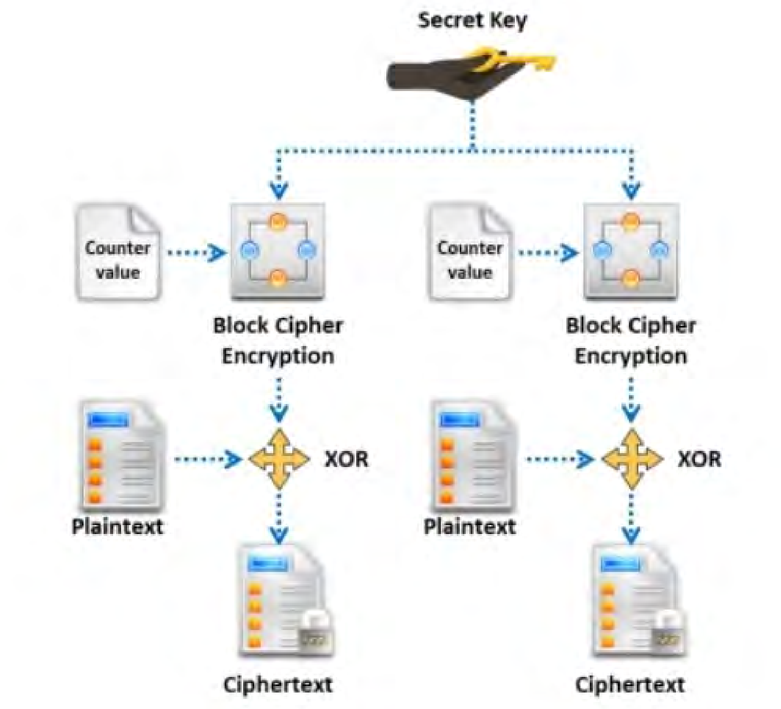


Figure 20-09: Counter Mode Encryption

Modes of Authenticated Encryption

Authenticated encryption (AE) modes provide both integrity and confidentiality for transmitted messages using a shared secret key, which helps prevent man-in-the-middle (MITM) attacks. To counter chosen ciphertext attacks, AE integrates a message authentication code (MAC) with the ciphertext, making it impossible to alter the ciphertext without detection. This integration also enhances security by rejecting improper ciphertexts during decryption.

Authenticated encryption with Message Authentication Code (MAC)

A MAC (Message Authentication Code) is a value generated by hashing a plaintext message with a shared secret key. It ensures the integrity of the message, allowing the receiver to verify its authenticity using the attached hash value. There are three different ways to utilize a MAC when encrypting a message.

1. Encrypt-then-MAC (EtM)

In this method, the plaintext is first encrypted using a secret key. Once the ciphertext is obtained, a hash value known as the message authentication code (MAC) is generated. This MAC is then attached to the ciphertext before transmission. This approach offers greater security for the transmitted message compared to other authenticated encryption (AE) methods.

2. Encrypt-and-MAC (E&M)

In the E&M approach, a Message Authentication Code (MAC) is first generated for the plaintext. After that, the plaintext is encrypted using a secret key. Finally, both the ciphertext and the MAC are combined and transmitted.

3. MAC-then-encrypt (MtE)

In the MtE approach, a Message Authentication Code (MAC) is first created for the plaintext using a hash function. This MAC is then combined with the plaintext. Subsequently, the combination of the plaintext and the MAC is encrypted with a secret key to produce ciphertext, which includes the encrypted MAC.

Authenticated Encryption with Associated Data (AEAD)

AEAD (Authenticated Encryption with Associated Data) is a method used to ensure the integrity and authenticity of a message containing a mix of encrypted and unencrypted data. This technique adds extra data to the ciphertext at specific locations to protect against chosen ciphertext attacks. The message header remains unencrypted, allowing the receiver to verify the source of the message, while the payload is encrypted to maintain confidentiality.

Cryptography Tools

You can use various cryptographic tools to encrypt and decrypt your information, files, etc. These tools implement different types of encryption algorithms.

- BCTextEncoder Source: https://www.jetico.com
- CryptoForge (https://www.cryptoforge.com)
- AxCrypt (https://axcrypt.net)
- Microsoft Cryptography Tools (https://www.microsoft.com)
- Concealer (https://www.belightsoft.com)
- SensiGuard (https://www.sensiguard.com)
- Cypherix (https://www.cypherix.com)

Applications of Cryptography

Cryptography is widely used to secure digital communication, protect sensitive data, and ensure authentication in various domains. In network security, cryptographic protocols like SSL/TLS secure web traffic, while VPNs use encryption to protect data in transit. Data protection relies on encryption to safeguard files, databases, and cloud storage. Cryptography also enables secure authentication through password hashing, multi-factor authentication (MFA), and digital signatures. In blockchain and cryptocurrency, cryptographic hashing ensures data integrity and transaction security. Additionally, cryptographic techniques are essential in email security (PGP, S/MIME), secure messaging (Signal, WhatsApp), and digital rights management (DRM) to prevent unauthorized access.

Public Key Infrastructure (PKI)

PKI is the combination of policies, procedures, hardware, software, and people required to create, manage, and revoke digital certificates. A Public Key Infrastructure (PKI) allows users of the internet and other public networks to engage in secure communication, data exchange, and money exchange through public and private cryptographic key pairs provided by a certificate authority.

Components of PKI

- Certificate Management System: Generates, distributes, stores, and verifies certificates
- Digital Certificates: Establishes credentials of a person when performing online transactions Validation Authority (VA): Stores certificates (with their public keys)
- Certification Authority (CA): Issues and verifies digital certificates
- End User: Requests, manages, and uses certificates
- Registration Authority (RA): Acts as the verifier for

Public and Private Key Pair

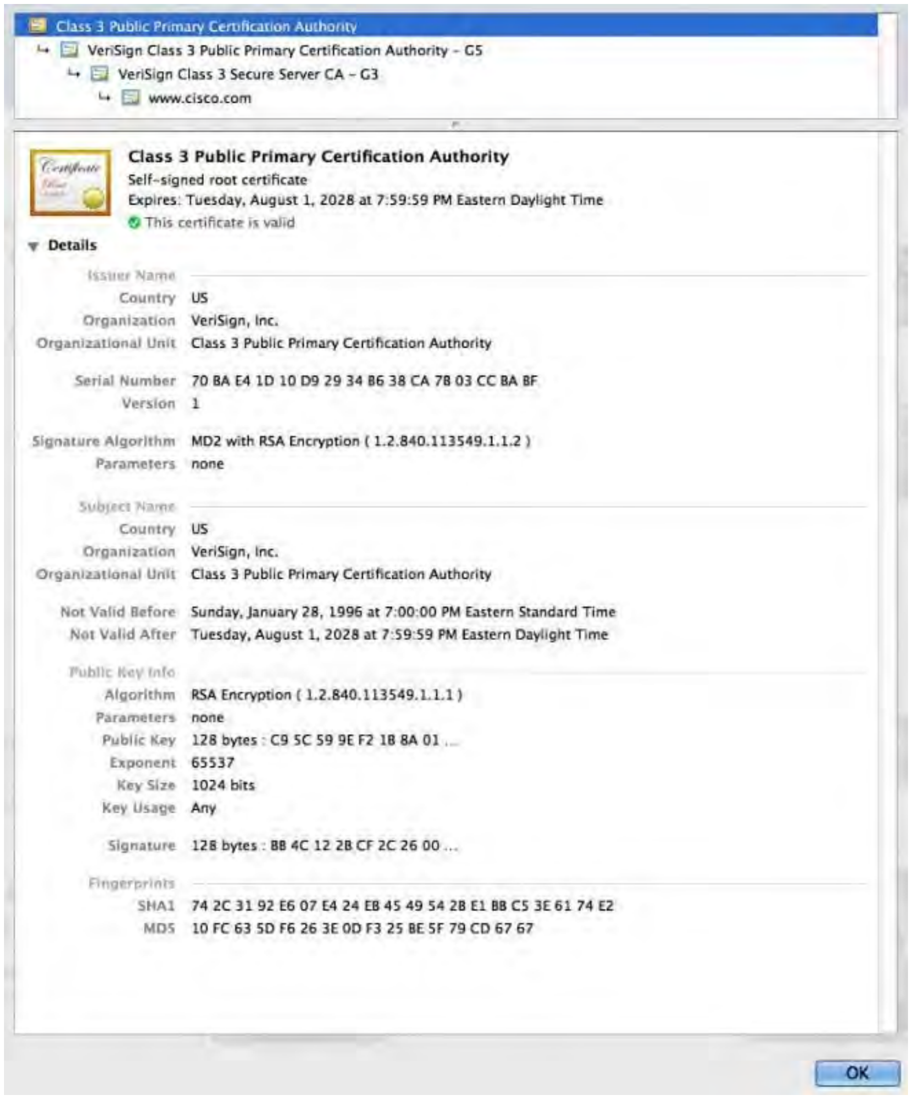
The Public and Private Key Pairs work like a team in the encryption/decryption process. The public key is provided to everyone, and the private key is secret. No one has a device's private key. We encrypt data sent to a particular node by using its public key. Similarly, the private key is used to decrypt the data. This is also true in the opposite case. If a node encrypts data with its private key, the public key is used for decryption.

Certificate Authorities (CA)

A Certificate Authority (CA) is a computer or entity that creates and issues digital certificates. A number of things such as IP address, fully qualified domain name, and the public key of a particular device are present in the digital certificate. CA also assigns a serial number to the digital certificate and signs the certificate with its digital signature.

Root Certificate

A Root Certificate provides the public key and other details of CA. An example of a Root certificate is:



Class 3 Public Primary Certification Authority
VeriSign Class 3 Public Primary Certification Authority - G5
VeriSign Class 3 Secure Server CA - G3
www.cisco.com

Class 3 Public Primary Certification Authority
Self-signed root certificate
Expires: Tuesday, August 1, 2028 at 7:59:59 PM Eastern Daylight Time
This certificate is valid

Details

Issuer Name	
Country	US
Organization	VeriSign, Inc.
Organizational Unit	Class 3 Public Primary Certification Authority
Serial Number	70 BA E4 1D 10 D9 29 34 B6 38 CA 7B 03 CC BA BF
Version	1
Signature Algorithm	MD2 with RSA Encryption (1.2.840.113549.1.1.2)
Parameters	none
Subject Name	
Country	US
Organization	VeriSign, Inc.
Organizational Unit	Class 3 Public Primary Certification Authority
Not Valid Before	Sunday, January 28, 1996 at 7:00:00 PM Eastern Standard Time
Not Valid After	Tuesday, August 1, 2028 at 7:59:59 PM Eastern Daylight Time
Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	none
Public Key	128 bytes : C9 5C 59 9E F2 1B 8A 01 ...
Exponent	65537
Key Size	1024 bits
Key Usage	Any
Signature	128 bytes : BB 4C 12 2B CF 2C 26 00 ...
Fingerprints	
SHA1	74 2C 31 92 E6 07 E4 24 EB 45 49 54 2B E1 BB C5 3E 61 74 E2
MD5	10 FC 63 5D F6 26 3E 0D F3 25 BE 5F 79 CD 67 67

Figure 20-10: Example Root Certificate

There are multiple informative sections in the figure 20-10, including serial number, issuer, country and organization names, validity date, and the public key itself. Every OS has its placement procedure regarding certificates. A certificate container for a specific OS can be searched on the internet to get to the certificates stored on the local computer.

Identity Certificate

The purpose of an Identity Certificate is similar to a root certificate except that it provides the public key and identity of a client computer or device. A good example of this is a client router or web server that wishes to make SSL connections with other peers.

Signed Certificate vs. Self-signed Certificate

Self-signed Certificates and Signed Certificates from a Certificate Authority (CA) provide security in the same way. Communication using these types of certificates is protected and encrypted by high-level security. The presence of a Certificate Authority implies that a trusted source has certified the communication. Signed Security Certificates are purchased, whereas Self-signed Certificates can be configured to optimize cost. A third-party Certificate Authority (CA) requires verification of domain ownership and other verification to issue a certificate.

Note: Cross certification enables entities in one Public Key Infrastructure (PKI) to trust entities in another PKI. This mutual trust relationship is typically supported by a cross-certification agreement between Certificate Authorities (CAs) in each PKI.

Signed Certificate (CA) vs. Self-Signed Certificate

The difference between a CA certificate and a self-signed certificate is the issuer of the certificate. A self-signed certificate is created, signed, and issued by the subject of the certificate (the entity it is issued to), while a CA certificate is created, signed, and issued by a third party called a certificate authority (CA) that is authorized to validate the identity of the applicant. A CA certificate signed by a publicly trusted CA can build trust among the website visitors, and therefore, it is used to validate public websites. A self-signed certificate is used in private networks.

Digital Signature

A Digital Signature is a technique to evaluate the authenticity of digital documents as the signature authenticates the authenticity of a document. A digital signature confirms the author of the document, date, and time of signing and authenticates the content of the message.

There are two categories of digital signature:

1. Direct Digital Signature
2. Arbitrated Digital Signature

Direct Digital Signature

Direct Digital Signatures involves only the sender and receiver of a message, assuming that the receiver has the sender's public key. The sender may sign the entire message or hash it with the private key and send it to the destination. The receiver decrypts it using the public key.

Arbitrated Digital Signature

Arbitrated Digital Signatures involves a third party called "Trusted Arbiter". The role of this arbiter is to validate the signed messages, insert the date, and then send it to the recipient. It requires a suitable level of trust and can be implemented with either public or private keys.

Secure Sockets Layer (SSL)

In a corporate environment, we can implement the security of corporate traffic over the public cloud by using site-to-site or a remote VPN. In the public cloud, there is no IPsec software running. Normal users also need to do encryption in some cases, such as online banking and electronic shopping. In such situations, SSL comes into play. The good thing about Secure Socket Layer (SSL) is that almost every single web browser in use today supports SSL. By using SSL, a web browser makes an HTTPS-based session with the server instead of HTTP. Whenever a browser tries to make an HTTPS-based session with a server, a certificate request is sent to the server in the background. The server, in return, replies with its digital certificate containing its public key. The web browser checks the authenticity of this certificate with a Certificate Authority (CA). Let's assume that the certificate is valid. Now, the server and the web browser have a secure session between them.

Transport Layer Security (TLS)

The Transport Layer Security (TLS) protocol is used to establish a secure connection between a client and a server and ensure the privacy and integrity of information during transmission. It uses a symmetric key for bulk encryption, asymmetric key for authentication and key exchange, and message authentication codes for message integrity. It uses the RSA algorithm with strengths of 1024 and 2048 bits. Using TLS, one can reduce security risks such as message tampering, message forgery, and message interception. An advantage of TLS is that it is independent of the application protocol. Higher-level protocols can lie on top of TLS transparen

SSL and TLS for Secure Communication

The terms SSL (Secure Socket Layer) and TLS (Transport Layer Security), often used interchangeably, provide data encryption and authentication in motion. These protocols are intended for a scenario where users want secure communication over an unsecured network, for example, the public internet. The most common applications of such protocols are web browsing, Voice over IP (VOIP), and electronic mail.

Consider a scenario where a user wants to send an email to someone or wants to purchase something from an online store where credit card credentials are required. SSL only spills the data after a process known as a 'handshake'. If a hacker bypasses the encryption process, everything from the bank account information to any secret conversation is visible, and malicious users can get hold of it to use for personal gain.

SSL was developed by Netscape in 1994 with the intention of protecting web transactions. The last version of SSL was version 3.0. In 1999, IETF created Transport Layer Security, which is also known as SSL 3. 1 as TLS is, in fact, an adapted version of SSL.

The following are some of the important functionalities SSL/TLS has been designed to do:

- Server authentication to client and vice versa
- Select common cryptographic algorithm
- Generate shared secrets between peers
- Protect normal TCP/UDP connections

Working

The working of SSL and TSL is divided into two phases:

Phase 1 (Session Establishment)

In this phase, common cryptographic protocol and peer authentication take place. There are three sub-phases within the overall phase 1 of SSL/TLS, as explained below:

- **Sub-phase 1:** In this phase, hello messages are exchanged to negotiate common parameters of SSL/TLS, such as authentication and encryption of algorithms
- **Sub-phase 2:** This phase includes one-way or two-way authentication between client and server end.
- **Sub-phase 3:** The last phase calculates a session key, and a cipher suite is finally activated. HMAC provides data integrity features by using either SHA-1 or MD5.

Similarly, using DES-40, DES-CBC, 3DEC-EDE, 3DES-CBC, RC4-40, or RC4- 128 provides confidentiality features

- **Session Key Creation:** Methods for generating session keys are as follows:
    - ***RSA Based:*** Using the public key of a peer encrypts a shared secret string
    - ***A fixed DH Key Exchange:*** Fixed Diffie-Hellman-based key exchanged in a certificate creating a session key
    - ***An ephemeral DH Key Exchange:*** This is considered the best protection option as an actual DH value is signed with the sender's private key, and hence, each session has a different set of keys
    - ***An anonymous DH Key Exchange without any Certificate or Signature:*** Avoiding this option is advised, as it cannot prevent man-in-the-middle attacks.

Phase 2 (Secure Data Transfer)

In this phase, secure data transfer takes place between encapsulating endpoints. Each SSL session has a unique session ID, which is exchanged during the authentication process. The session ID is used to differentiate between an old and a new session. The client can request the server resume the session based on this ID (in this event, the server has a session ID in its cache).
TLS 1.0 is considered a bit more secure than the last version of SSL (SSL v3.0). Even the U.S. government has declared it will not use SSL v3.0 for highly sensitive communications due to the latest vulnerability named POODLE. After the POODLE vulnerability, most web browsers disabled SSL v3.0 for most communication and services. Current browsers (Google Chrome, Firefox, and others) support TLS 1.0 by default and the latest versions of TLS (TLS 1.1 and TLS 1.2) optionally. TLS 1.0 is considered equivalent to SSL3.0. However, newer versions of TLS are considered far more secure than SSL. Keep in mind that SSL v3.0 and TLS 1.0 are incompatible as TLS uses Diffie-Hellman and Data Security Standard (DSS) while SSL uses RSA.

Apart from secure web browsing, HTTPS and SSL/TLS can also be used for securing other protocols such as FTP, SMTP, and SNTP.

Note: OPPORTUNISTICTLS STARTTLS is a protocol command issued by an email client. It indicates that a client wants to upgrade an existing insecure connection to a secure one using the SSL/TLS protocol.

Cryptography Toolkit

Cryptography toolkits include cryptographic primitives, algorithms, and schemes used to provide security for various applications. Some cryptography toolkits are shown below:

- OpenSSL Source: https://www.openssl.org
- wolfSSL (https://www.wolfssl.com)
- AES Crypto Toolkit (https://www.ni.com)
- Libsodium (https://github.com)
- Crypto++ (https://cryptopp.com)
- PyCryptodome (https://github.com)

Pretty Good Privacy (PGP)

OpenPGP is the most widely used email encryption standard. It is defined by the OpenPGP Working Group of the Internet Engineering Task Force (IETF) as a Proposed Standard in RFC 4880. The primary purpose of OpenPGP is to ensure end-to-end encryption for email communications. Additionally, it provides message encryption and decryption, a password manager, data compression, and digital signing capabilities.
OpenPGP is commonly used for encrypting and decrypting messages, emails, files, and directories, thereby enhancing the privacy of email communications. For message encryption, it employs RSA for key transport and IDEA for bulk message encryption. OpenPGP utilizes RSA to compute digital signatures and MD5 for generating message digests. This system combines the advantages of both conventional cryptography (which is about 1,000 times faster than public-key encryption) and public-key cryptography (which addresses key distribution and data transmission issues), making it a hybrid cryptosystem.

GNU Privacy Guard (GPG)

GNU Privacy Guard (GPG) is a software alternative to PGP and a free implementation of the OpenPGP standard used for encrypting and decrypting data. GPG is considered a hybrid encryption program because it utilizes both symmetric-key and asymmetric-key cryptography, which enhances speed and secures key exchange. This is accomplished by using the receiver's public key to encrypt the session key. Additionally, GPG supports S/MIME and Secure Shell (SSH). The latest version of GPG is compatible with most cryptographic functions, including elliptic curve cryptography (ECDSA, ECDH, and EdDSA), and it also integrates the Libgcrypt cryptography library.

Web of Trust (WOT)

The Web of Trust (WoT) is a trust model used in systems like PGP, OpenPGP, and GnuPG. Its primary concept is to decentralize key distribution among PGP users. In a Public Key Infrastructure (PKI), a centralized authority, known as a Certificate Authority (CA), is responsible for signing certificates, thereby ensuring the authenticity of the public key and its owner. In contrast, the WoT empowers everyone in the network to act as a CA, allowing users to sign for other trusted entities.

In the Web of Trust, individuals validate each other's certificates through a network of signatures. These signatures confirm the ownership of keys across different levels of trust. The WoT framework includes various levels of trust, established through both direct and indirect references among users.

Working of WOT

In WOT (Web of Trust), every PGP (Pretty Good Privacy) user in the network maintains a ring of public keys to encrypt data. Users can introduce other individuals whom they trust. In this trust model, a user encodes data using the receiver's public key, which can only be decrypted with the receiver's private key. Furthermore, each user digitally signs the data with their private key. When the recipient validates the signature against the user's public key, they can confirm the user's authenticity. This process ensures that the data is received from a valid user without any alterations, and only the intended recipient can access the information since they are the only one holding the corresponding private key.

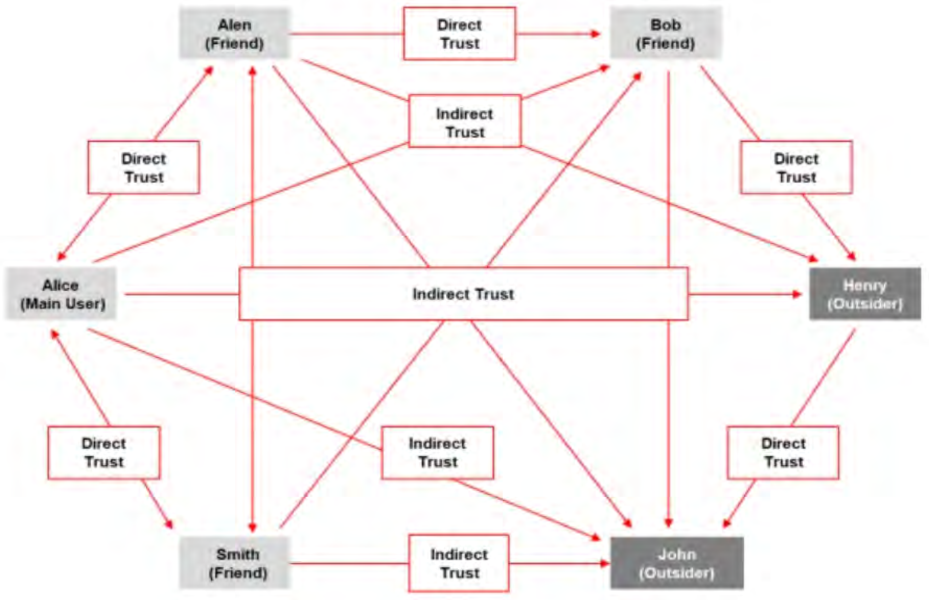


Figure 20-11: Working of WOT

Encrypting Email Messages in Outlook

Encryption using S/MIME certification is a technique that enables users to encrypt their email messages. This method is particularly used for securing Outlook email messages, allowing senders and designated recipients to access the content without compromising its integrity. Below are the steps to encrypt email messages using S/MIME encryption. To complete these steps, a signing certificate must be attached to the keychain.

Signing/Encrypting Email Messages on Mac

The email security features in Apple Mail can be improved by using the encryption options known as digital signatures and message encryption, which are offered by Apple Mail service providers. Users can send emails that are both digitally signed and encrypted from their Apple devices.

Encrypting/Decrypting Email Messages Using OpenPGP

While PGP offers good security, it is still vulnerable to online attacks. However, using OpenPGP—a hybrid version of PGP—across multiple platforms such as Windows, macOS, Android, iOS, Linux, and various browser plugins significantly enhances security. When OpenPGP is installed and properly configured with compatible browser extensions, it can further strengthen security for email communication in browser environments. Users can utilize browser extension tools like FlowCrypt, which work in conjunction with OpenPGP, to ensure secure email communication.

Email Encryption Tools

Some important email encryption tools used to secure email messages are as follows:

- RMail Source: https://rmail.com
- Mailvelope (https://mailvelope.com)
- Virtru (https://www.virtru.com)
- Webroot™ (https://www.webroot.com)
- Secure Email (S/MIME) Certificates (https://www.ssl.com)
- Proofpoint Email Protection (https://www.proofpoint.com)
- Paubox (https://www.paubox.com)

Disk Encryption

Disk Encryption refers to the encryption of a disk to secure files and directories by converting the data into an encrypted format. Disk encryption encrypts every bit on the disk to prevent unauthorized access to data storage. There are several disk encryption tools available to secure disk volume, for example:

- Symantec Drive Encryption
- GiliSoft Full Disk Encryption

Disk Encryption Tools

The common goal of disk encryption tools is to encrypt a disk partition to provide confidentiality to the information stored on it. Some disk encryption tools are discussed below.

- VeraCrypt Source: https://veracrypt.fr
- Rohos Disk Encryption Source: https://rohos.com
- BitLocker Drive Encryption Source: https://www.microsoft.com
- Symantec Encryption (https://www.broadcom.com)
- SafeGuard Enterprise Encryption (https://www.sophos.com)
- GiliSoft Full Disk Encryption (https://www.gilisoft.com)
- Check Point Full Disk Encryption (https://www.checkpoint.com)

- DiskCryptor (https://diskcryptor.org)

Disk Encryption Tools for Linux

The following are disk encryption tools for Linux:

- Cryptsetup Source: https://gitlab.com
- Cryptmount (https://cryptmount.sourceforge.net)
- Tomb (https://dyne.org)
- CryFS (https://www.cryfs.org)
- GnuPG (https://www.gnupg.org)
- Harmony Endpoint (https://www.checkpoint.com)

Disk Encryption Tools for macOS

The following are disk encryption tools for macOS:

- FileVault Source: https://support.apple.com)
- VeraCrypt (https://www.veracrypt.fr)
- BestCrypt Volume Encryption (https://www.jetico.com)
- Dell Full Disk Encryption (https://www.dell.com)
- Comodo Disk Encryption (https://www.comodo.com)
- GravityZone Full Disk Encryption (https://www.bitdefender.com)

Blockchain

A type of distributed ledger technology (DLT) called blockchain is used to safely record and retain transaction history in the form of blocks. Cryptographic methods preserve account transparency, and data stored in blockchains is impervious to unauthorized changes. Multiple blocks are generated for each transaction, and these blocks are cryptographically connected to build a "blockchain." Ledgers are a chain of records or blocks that are exchanged around the network to inform other participants of all transaction data and the number of bits that each member owns.

After the network's participants authenticate blocks using their hash values, crypto miners use complex cryptographic algorithms to validate the hashes further. Only then are the blocks authorized to join the blockchain mechanism.

Two technologies are typically used to create blockchains: asymmetric key algorithms and hash functions, most commonly SHA-256. The process of adding blocks to a blockchain after completing "proof of work" is called "crypto mining," and the process of validating blocks is called "proof of work," for which cryptocurrency miners get paid. Three components make up each block in a blockchain: the hash of the previous block, the data (transaction information), and the hash itself. The hash value is shared with the following block each time a new block is constructed with a different hash value. A blockchain's initial block, known as the genesis, is denoted by zeros. A block can enter the blockchain after confirming the hash of its predecessor.

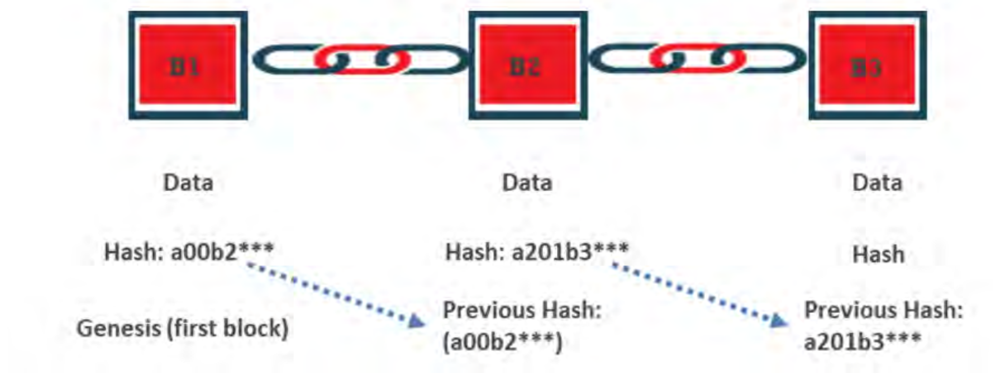


Figure 20-12: Blockchain

If a block is tampered with, the next block in the chain invalidates it because it does not match the previous hash value in the current block. However, a blockchain is not completely protected by merely generating hashes and comparing them with other blocks. Attackers can generate valid hashes for each block using many cryptographic techniques. The "proof of work" mechanism is used, as described above, to mitigate such risks. The security of the blockchain is ensured by both the effective usage of hashes and the "proof of work" by miners (on public ledgers).

Cryptanalysis

The process of analyzing cryptographic systems to find flaws or data leaks is known as cryptanalysis. However, looking for implementation flaws like side-channel attacks or low entropy inputs is also part of cryptanalysis, which is often thought of as studying the weaknesses of a cryptographic system's underlying mathematics.

A Cryptanalyst is someone who carries out Cryptanalysis. Identifying any weak points in the cryptosystem aids in our understanding of them and our ability to strengthen them and work on the algorithm to produce more secure secret codes. For example, the plaintext of a ciphertext, might be extracted by a cryptanalyst. It can assist us in figuring out the encryption key or the plaintext.

Cryptanalysis Methods

1. Linear Cryptanalysis

Linear cryptanalysis is based on finding linear approximations to the action of a cipher, particularly block ciphers. This technique was invented by Mitsuru Matsui. It is a known plaintext attack that uses a linear approximation to describe the behavior of the block cipher. Given sufficient pairs of plaintext and corresponding ciphertext, bits of information about the key can be obtained. Obviously, the more pairs of plaintext and ciphertext one has, the greater the chances of success. Remember that cryptanalysis is an attempt to break cryptography. For example, with the 56-bit Data Encryption Standard (DES), brute-forcing the key could take up to 256 attempts. Linear cryptanalysis, on the other hand, requires approximately 243 known plaintexts to succeed. While this is better than brute-forcing, it is still impractical in most situations without sufficient data. The math may be complex for novice cryptographers, but let's look at the basics. In linear cryptanalysis, a linear equation expresses the equality of two expressions consisting of XORed binary variables.

For example, the following equation XORs the sum of the first and third plaintext bits, and the first ciphertext bit is equal to the second bit of the key:

$$P_1 \oplus P_3 \oplus C_1 = K_2$$

You can use this method to gradually recover the key that was used. After deriving such equations for each bit, you will have an equation of the form:

$$P_{i1} \oplus P_{i2} \oplus \ldots \oplus C_{j1} \oplus C_{j2} \oplus \ldots = K_{k1} \oplus K_{k2} \oplus \ldots$$

2. Differential Cryptanalysis

Differential cryptanalysis is a method used to analyze symmetric-key algorithms. It was invented by Eli Biham and Adi Shamir. This technique involves examining the differences in input data and how those differences influence the resulting output. Initially, differential cryptanalysis was applicable only with chosen plaintext, but it can also be effective with known plaintext and ciphertext.

3. Integral Cryptanalysis

Lars Knudsen first described integral cryptanalysis. This attack is particularly useful against block ciphers based on substitution-permutation networks as an extension of differential cryptanalysis. The differential analysis looks at pairs of inputs that differ in only one bit position, with all other bits being identical. Integral analysis for block size b holds b-k bits constant and runs the other k bits through all 2k possibilities. For $k = 1$, this is just differential cryptanalysis, but with $k > 1$, it is a new technique.

4. Quantum Cryptanalysis

Quantum cryptanalysis refers to the method of breaking cryptographic algorithms with the help of a quantum computer. Attackers can employ Shor's quantum factoring algorithm to target public-key cryptographic systems like RSA and Elliptic Curve Diffie-Hellman (ECDH). This algorithm allows them to factor large numbers in polynomial time. Additionally, they can utilize Grover's quantum search algorithm, which accelerates brute-force key searches for block ciphers such as AES and hash functions like SHA.

Who Uses Cryptanalysis

Governments seeking to decipher the private communications of other countries, businesses creating security products that use cryptanalysts to verify their security features, hackers, crackers, independent researchers, and academics all engage in cryptanalysis, as do many other types of organizations.

The ongoing conflict between cryptographers seeking to protect data and cryptanalysts seeking to crack cryptosystems is what advances our understanding of cryptology as a whole.

Cryptanalysis Tools

The following are a few of the numerous instruments used in cryptanalysis:

- **Cryptol:** The Nation Security Organization (NSA), a US intelligence agency, originally created this open-source programme to target encryption methods. Users of Cryptol are able to observe how algorithms function in programmes that define the cyphers or algorithms.
- **CrypTool:** CrypTool is an additional open-source product that develops e-learning courses and a web page to assist users in learning about cryptographic algorithms and cryptanalysis.
- **Ganzua:** A skeleton key or lockpick is referred to as a "ganzua" in Spanish. It is a Java-based, open-source application that lets researchers construct almost completely arbitrary encryption and plain alphabets. Additionally, users will be able to decipher non-English cryptograms using this tool.

Cryptography Attacks

Cryptography Attacks are intended to recover an encryption key. Once an attacker has the encryption key, he/she can decrypt all messages. Weak encryption algorithms are not resistant enough for cryptographic attacks. The process of finding vulnerabilities in a code, encryption algorithm, or key management scheme is called Cryptanalysis. It may be used to strengthen a cryptographic algorithm or to decrypt the encryption.

Known Plaintext Attack

A Known Plaintext Attack is a cryptographic attack type where a cryptanalyst has access to plaintext and the corresponding ciphertext and seeks to discover a correlation between them.

Ciphertext-only Attack

A Ciphertext-only Attack is a cryptographic attack type where a cryptanalyst has access to a ciphertext but does not have access to the corresponding plaintext. The attacker attempts to extract the plain text or key by recovering as many plain text messages as possible to guess the key. Once the attacker has the encryption key, he/she can decrypt all messages.

Chosen Plaintext Attack

A Chosen Plaintext Attack is a cryptographic attack type where a cryptanalyst can encrypt a plaintext of his choosing and observe the resulting ciphertext. It is the most common attack against asymmetric cryptography. To attempt a chosen-plaintext attack, the attacker has information about the encryption algorithm or may have access to the workstation encrypting the messages. The attacker sends chosen plaintexts through the encryption algorithm to extract ciphertexts and then uses the encryption key. A chosen plaintext attack is vulnerable in a scenario where public-key cryptography is in use, and the public key is used to encrypt the message. In the worst cases, an attacker can expose sensitive information.

Chosen Ciphertext Attack

A Chosen Ciphertext Attack is a cryptographic attack type where a cryptanalyst chooses a ciphertext and attempts to find the corresponding plaintext.

Adaptive Chosen Ciphertext Attack

An Adaptive Chosen Ciphertext Attack is an interactive type of chosen-plaintext attack where an attacker sends some ciphertexts to be decrypted and observes the results of decryption. An adaptive chosen ciphertext attack gradually reveals the information about the encryption.

Adaptive Chosen Plaintext Attack

An Adaptive Chosen Plaintext Attack is a form of chosen plaintext cryptographic attack where the cryptanalyst issues a series of interactive queries, choosing subsequent plaintexts based on information from previous encryptions.

Rubber Hose Attack

A Rubber Hose Attack is the technique of obtaining information about cryptographic secrets such as passwords, keys, or encrypted files by torturing a person.

Collision

Collision refers to a hash collision, which means two different plaintexts have the same hash value. This rare condition is not supposed to exist in a hash algorithm. The hashing process accepts an infinite input length and produces a finite output. Consider a scenario where an attacker finds a hash collision among legitimate and altered documents. Now, being undetected, the attacker can easily fool the target.

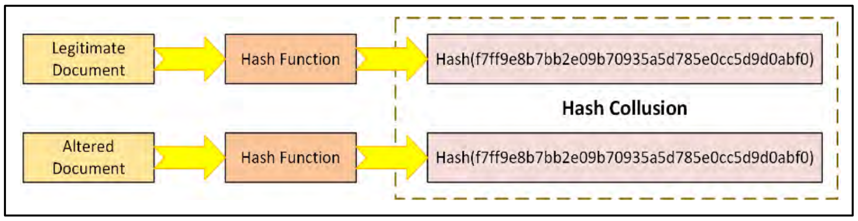

Legitimate Document
Hash Function
Hash(f7ff9e8b7bb2e09b70935a5d785e0cc5d9d0abf0)
Hash Collusion
Altered Document
Hash Function
Hash(f7ff9e8b7bb2e09b70935a5d785e0cc5d9d0abf0)

Figure 20-13: Hash Collision

Code Breaking Methodologies

Code Breaking Methodology includes several tricks and techniques, for example, using social engineering that is helpful to break encryption and expose the information in it, such as cryptographic keys and messages. The following are some effective techniques and methodologies:

- Brute Force
- One-Time Pad
- Frequency Analysis

Brute-Force Attack

Cryptographic systems are difficult to crack due to their lack of practical weaknesses. They use cryptographic algorithms to encrypt or decrypt messages, with a key being the key that specifies the transformation of plaintext to ciphertext and vice versa. If a key is found, messages can be decrypted and read in clear text. 128-bit keys are common and considered strong. To avoid guessing the key, cryptographic systems use randomly generated keys. However, a brute-force attack can be used to determine the key used for encryption. This method requires significant processing power and is time-consuming. The average time needed to find the key depends on the key's length, with short keys taking less time and long keys taking more time. A successful brute-force attack depends on the attacker's time to discover the key, which is relative to the key's length.

Power/Cost	40 bits (5 char)	56 bits (7 char)	64 bits (8 char)	128 bits (16 char)
$2K (1 PC. Can be achieved by an individual)	1.4 min	73 days	50 years	10^20 years
$100K (this can be achieved by a company)	2 sec	35 hours	1 year	10^19 years
$1M (Achieved by a huge organization or a state)	0.2 sec	3.5 hours	37 days	10^18 years

Table 20-05: Estimate Time for a Successful Brute-Force Attack

Birthday Attack

A birthday attack is a type of brute-force attack aimed at cryptographic hashes that simplifies the process of brute-forcing. This method is based on the birthday paradox, which states that in a group of 23 people, the probability of two or more individuals sharing the same birthday is greater than 0.5.

Meet-in-the-Middle Attack on Digital Signature Schemes

A meet-in-the-middle attack is an effective method for attacking cryptographic algorithms that use multiple keys for encryption. This approach reduces the number of brute-force permutations needed to decode text encrypted with more than one key.

The meet-in-the-middle attack leverages a space-time trade-off and is akin to a birthday attack, as it takes advantage of the mathematics behind the birthday paradox. This type of attack is more time-efficient compared to a complete exhaustive search.

The name "meet-in-the-middle" comes from the way the attack works: it encrypts data from one end while simultaneously decrypting from the other end, ultimately converging "in the middle." In a meet-in-the-middle attack, the attacker has access to a known plaintext message as well as the corresponding encrypted text. This technique is often utilized by attackers to forge messages that employ multiple encryption schemes.

Side-Channel Attack

A side-channel attack is a physical attack performed on a cryptographic device/cryptosystem to gain sensitive information. Cryptography is generally part of the hardware or software that runs on physical devices such as semi-conductors (resistor, transistor, and so on) that interact with and affect various environmental factors.

Hash Collision Attack

A hash collision attack is performed by finding two different input messages that result in the same hash output. For example, in a hash collision attack, "hash(a1) = hash(a2)", where a1 and a2 represent some random messages. Since the algorithm itself randomly selects these messages, attackers have no role in the content of these messages. This allows the attacker to perform cryptanalysis by exploiting the digital signature used to generate a different message with the same hash value.

One of the most popular hash functions is SHA-1, which is widely used as a digital signature algorithm. SHA-1 converts an input message into a constant length of unstructured strings of numbers and alphabets, which act as a fingerprint for the sent file. Therefore, the attacker tries to identify similar hashed output to get the digital signatures of the victim. This allows the attacker to forge the victim's digital signature of message a1 on message a2.
Once the attacker detects a collision in the hash, he/she can identify more collisions by concatenating the data to matching messages.

DUHK Attack

The Don't Use Hard-Coded Keys (DUHK) vulnerability is a cryptographic flaw that allows attackers to obtain encryption keys used to secure VPNs and web sessions. This issue primarily affects any hardware or software utilizing the ANSI X9.31 Random Number Generator (RNG).

Pseudorandom number generators (PRNGs) create random sequences of bits based on an initial secret value known as a seed and the current state. The PRNG algorithm is responsible for generating cryptographic keys that establish a secure communication channel over the VPN. In some instances, the seed key is hardcoded into the implementation. Both of these factors contribute to the DUHK attack vulnerability, as an attacker can combine the ANSI X9.31 RNG with the hardcoded seed key to decrypt encrypted data sent or received by that device.

Man-in-the-middle attackers exploit the DUHK attack to discover the seed value, monitor the current session, and obtain the current state value. Through this method, they can identify encryption keys and steal confidential information, including critical business data, user credentials, and credit card details.

DROWN Attack

Decrypting RSA with Obsolete and Weakened eNcryption (DROWN) is a grave vulnerability that can affect important cryptographic protocols such as HTTPS and other cryptographic services that depend on SSL and TSL. The DROWN attack is a cross-protocol weakness that can communicate and initiate an attack on servers supporting recent SSLv3/TLS protocol suites. It is a new form of cross-protocol Bleichenbacher padding oracle attack.

Rainbow Table Attack

A rainbow table attack is a type of cryptography attack in which an attacker uses a rainbow table to reverse cryptographic hash functions. It uses the cryptanalytic time-memory trade-off technique, which is less time-consuming than other techniques. It uses already calculated information stored in memory for encryption. In the rainbow table attack, the attacker creates a table of all the possible passwords and their respective hash values, called a rainbow table, in advance.

A rainbow table contains word lists such as dictionary files and brute-force lists and their hash values. It is a lookup table particularly used for recovering a plaintext password from a ciphertext. The attacker uses this table to look for the password and tries to recover it from password hashes. An attacker computes the hash for a list of possible passwords and compares it with the pre-computed hash table (rainbow table). If a match is found, then he/she can crack the password. It is easy to recover passwords by comparing the captured password hashes with pre-computed tables.

Related-Key Attack

Attackers can launch a related-key attack by exploiting the mathematical relationships between the keys used in a cipher. This allows them to gain access to both encryption and decryption functions. The main motive behind this attack is to uncover the related private or secret keys.

To carry out this type of attack, the attacker monitors the cipher's operations when the key values are initially unknown. Through careful analysis, the attacker captures the relationships between those keys. For example, the attacker may observe that the last 80 bits of the keys are consistently the same, even though they are unaware of these bits at the start.

Padding Oracle Attack

In a padding oracle attack, attackers exploit the validation process of message padding in encrypted communications to decipher the ciphertext. This type of attack is also referred to as a Vaudenay attack. Many cryptographic algorithms that use block ciphers require messages to be padded with additional bits so that the last block meets the necessary size requirements. The padding oracle is a function of such encryption that checks whether a message is correctly padded.

This attack primarily targets algorithms operating in Cipher Block Chaining (CBC) mode. During the attack, the server, referred to as the oracle, reveals information about the correctness of an encrypted message's padding. In some cases, this information allows attackers to decrypt messages and even encrypt new messages using the oracle's key without needing access to the original encryption key.

Attacks on Blockchain

Attacks on blockchain networks involve various malicious activities aimed at exploiting vulnerabilities within these systems. These attacks can take several forms, one of the most notable being a 51% attack. In such an attack, a single entity gains majority control over the network's computing power, enabling it to double-spend coins and reverse transactions. Attackers may carry out these actions to steal funds, disrupt services, or undermine trust in the network.

The consequences of blockchain attacks can be severe, resulting in substantial financial losses, diminished confidence in blockchain technology, compromised data integrity, and potential legal and regulatory repercussions for the affected organizations.

Quantum Computing Attacks

Potential quantum-computing attacks and their implications for the security of modern cryptographic systems are discussed below.

Quantum Cryptanalysis Attack

Quantum cryptanalysis leverages quantum computing to identify vulnerabilities in traditionally secure cryptographic systems. Powerful algorithms like Shor's and Grover's suggest a shift in protecting sensitive information. Staying informed about these advancements helps enhance digital security, ensuring the integrity and authenticity of online communications and transactions.

Quantum Side-Channel Attack

Quantum side-channel attacks exploit vulnerabilities in the physical implementation of quantum cryptographic systems to gather information without attacking the algorithms directly. By analyzing factors like quantum noise, error rates, timing, power consumption, and electromagnetic emissions during computations or key exchanges, attackers can infer sensitive information such as cryptographic keys.

Classical-to-Quantum Transition Attack

Classical-to-quantum transition attacks exploit vulnerabilities during the shift from classical to quantum-resistant cryptographic systems. Hybrid systems may have weaknesses, allowing attackers to compromise security and intercept communications before robust quantum protocols are fully implemented.

Cryptanalysis Tools

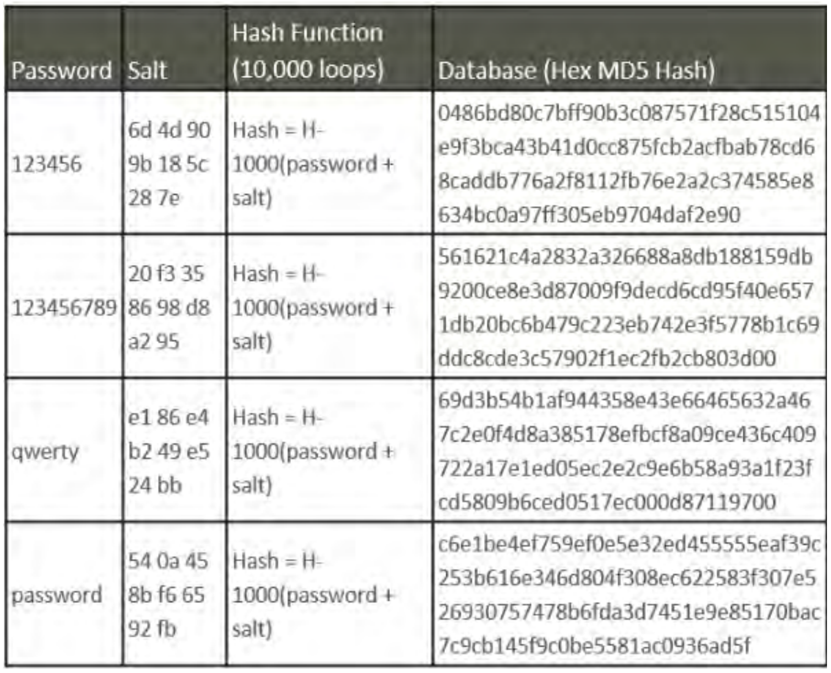
Attackers use cryptanalysis tools to analyze and break ciphers. Some cryptanalysis tools are discussed as follows.

- CrypTool Source: https://www.cryptool.org
- RsaCtfTool (https://github.com)
- Msieve (https://sourceforge.net)
- Cryptol (http://cryptol.net)
- CryptoSMT (https://github.com)
- MTP (https://github.com)

Key Stretching

Key stretching techniques make a potentially weak key, usually a password or passphrase, more secure against brute-force attacks by increasing the resources (time and possibly space) required to test each possible key. Key stretching can be done in various ways.

The essential strength of a password's bits is how strong it is when stretched. The hash function is the key to methods for increasing a password's bit length. Typically, hash functions are looped a huge number of times, imitating randomness and increasing the complexity of a password sent to the database by a few bits at a time.



Password	Salt	Hash Function (10,000 loops)	Database (Hex MD5 Hash)
123456	6d 4d 90 9b 18 5c 28 7e	Hash = H-1000(password + salt)	0486bd80c7bff90b3c087571f28c515104 e9f3bca43b41d0cc875fcb2acfbab78cd6 8caddb776a2f8112fb76e2a2c374585e8 634bc0a97ff305eb9704daf2e90
123456789	20 f3 35 86 98 d8 a2 95	Hash = H-1000(password + salt)	561621c4a2832a326688a8db188159db 9200ce8e3d87009f9decd6cd95f40e657 1db20bc6b479c223eb742e3f5778b1c69 ddc8cde3c57902f1ec2fb2cb803d00
qwerty	e1 86 e4 b2 49 e5 24 bb	Hash = H-1000(password + salt)	69d3b54b1af944358e43e66465632a46 7c2e0f4d8a385178efbcf8a09ce436c409 722a17e1ed05ec2e2c9e6b58a93a1f23f cd5809b6ced0517ec000d87119700
password	54 0a 45 8b f6 65 92 fb	Hash = H-1000(password + salt)	c6e1be4ef759ef0e5e32ed455555eaf39c 253b616e346d804f308ec622583f307e5 26930757478b6fda3d7451e9e85170bac 7c9cb145f9c0be5581ac0936ad5f

Table 20-06: Key Stretching

Key Stretching Algorithms

Bcrypt, Scrypt, and Password-Based Key Derivation Function 2 (PBKDF2) are three common key stretching techniques:

To increase the size of the password hash and make it more difficult for a brute-force attack, the idea behind key stretching is to inject a random sequence of characters:

- **BCRYPT -** The password-hashing algorithm BCRYPT is based on the Blowfish cipher. It salts passwords to extend their length and thwart rainbow table attacks. Additionally, it contains an adaptive function that allows for slowing repetition rates to make it more resistant to attacks, even with a rise in computing power.
- **PBKDF2 -** A cryptographic key is often derived from a password using PBKDF2 (Password Based Key Derivation Function 2). It can also be used to store keys, however, alternative key storage KDFs like Scrypt are typically seen as being preferable. This class complies with the interface defined by KeyDerivationFunction.

- **Scrypt -** It is a key-derivation function that uses passwords (KDF). A KDF is a hash function used in cryptography that uses a pseudorandom function to derive one or more secret keys from a secret value, such as a master key, a password, or a passphrase. In general, KDFs are effective at thwarting brute-force password guessing attempts.

Cryptography Attack Countermeasures

To protect against cryptographic attacks, several countermeasures can be implemented, which are discussed below.

Defend Against Cryptographic Attacks

To prevent cryptographic attacks, several countermeasures can be implemented:

Key Management and Security

Key management and security best practices include restricting cryptographic key access to authorized applications or users, encrypting stored keys with passphrases or passwords, and avoiding embedding keys in source code or binaries. Regular key rotation, enforcing hardware-backed security like HSMs, and limiting key reuse further enhance security.

Encryption Standards and Best Practices

Encryption standards and best practices recommend using strong encryption algorithms such as AES, TLS, and GCM while preferring 256-bit keys for symmetric encryption and at least 2048-bit keys for asymmetric encryption. Strong key derivation functions (KDFs) and secure key schedules should be implemented to prevent simple encryption key relationships.

Integrity and Authentication

To ensure integrity and authentication, digital signatures should be used for message verification, and message authentication must be implemented in encryption protocols. Secure hashing algorithms like SHA-256 and SHA-3 should replace weaker alternatives such as MD5 and SHA-1. Additionally, techniques like salting and key stretching (PBKDF2, bcrypt, Argon2) help prevent brute-force attacks.

Quantum and Future-Proof Security

For quantum and future-proof security, organizations should adopt quantum-resistant cryptographic algorithms, such as lattice-based and hash-based cryptography, and prepare for post-quantum cryptography by testing NIST-recommended algorithms. Zero-knowledge proofs like zk-SNARKs can be utilized for secure authentication and data integrity.

Secure Communication and Implementation

Secure communication and implementation practices include deploying IDS to monitor key exchanges, encrypting communications with protocols like TLS, ensuring encryption schemes do not produce predictable outputs, and utilizing hardware-based random number generators (RNGs) for key generation. These measures collectively help strengthen cryptographic security and mitigate potential vulnerabilities.

<u>Defend Against Blockchain Attacks</u>

To prevent blockchain attacks, several countermeasures can be implemented:

Identity and Transaction Security

Security and identity protection can be enhanced using decentralized identifiers (DIDs) and zero-knowledge proofs, ensuring transactions and identities are verified without exposing sensitive data. Cryptographic keys should be securely stored in hardware security modules (HSMs) and multi-signature wallets should be used to require multiple keys for transaction authorization.

Network and Transaction Monitoring

Network and transaction security measures include real-time monitoring with machine learning to detect double-spending, combining proof-of-work (PoW) and proof-of-stake (PoS) for efficiency, and implementing advanced DDoS protection. Smart contracts should undergo formal verification and regular audits to ensure their security. Secure interoperability protocols and atomic swaps should be used to protect cross-chain transactions and minimize risks.

Node and Peer Security

For node and peer security, randomized peer selection algorithms, periodic connection timeouts, and reputation-based peer scoring should be adopted to prevent targeted attacks. Nodes should use out-of-band verification and trusted bootstrapping nodes for secure network connections. Secondary trusted communication channels can help detect and respond to network anomalies.

Transaction Integrity

To enhance transaction integrity, multiple confirmations should be required before accepting transactions, and pending transaction details should be hidden to prevent front-running attacks. Batch processing and fair sequencing can prevent transaction reordering, while randomized submission times make order manipulation more difficult. Strengthening mining pool surveillance and increasing transaction propagation speed further reduce attack windows, ensuring a more secure blockchain ecosystem.

<u>Defend Against Quantum Computing Attacks</u>

To prevent quantum attacks, several countermeasures can be implemented:

Quantum-Resistant Cryptography

To safeguard against quantum computing attacks, quantum-resistant cryptographic algorithms like lattice-based, hash-based, and code-based cryptography should be adopted. These algorithms are specifically designed to withstand the challenges posed by quantum computers. Quantum mechanics can be used to securely distribute cryptographic keys, ensuring that even in a post-quantum world, sensitive information remains secure. During the transition period, it is important to combine classical cryptographic methods with quantum-resistant algorithms to maintain security. Additionally, using larger symmetric keys helps mitigate the reduced security due to quantum computing, ensuring a higher level of protection against potential attacks.

Key Management and Security

Regularly changing cryptographic keys limits the time they are exposed to potential quantum attacks. To safeguard keys from side-channel attacks, such as power analysis or electromagnetic emissions, protection mechanisms should be implemented. Cloud-based key management services that utilize quantum-resistant algorithms offer another layer of security. Trusted platform modules (TPMs) and hardware security modules (HSMs) should be used for the secure storage of quantum-resistant keys, while ensuring that these systems are regularly updated with quantum-safe firmware to maintain their protection against future threats.

Authentication and Multi-Factor Security

Authentication protocols must be updated to be resistant to quantum computing threats. Developing quantum-resistant authentication methods, including digital certificates, ensures secure user verification. Enhancing multi-factor authentication (MFA) with quantum-resistant techniques adds an additional layer of security, ensuring that even if one factor is compromised, the system remains secure. Quantum-resistant zero-knowledge proofs should be employed to authenticate users without exposing sensitive information, maintaining privacy while verifying identities.

Data Protection and Privacy

Quantum-resistant encryption algorithms should be used to encrypt stored data, ensuring its security against the future capabilities of quantum computers. Additionally, breaking data into fragments and distributing them across multiple locations minimizes the risk of reconstruction, even if some fragments are compromised. Random number generation methods used in cryptographic systems must be secure against quantum threats, as they play a vital role in generating keys and ensuring the integrity of encrypted data.

Blockchain and Distributed Ledger Technology

For blockchain and distributed ledger systems, integrating quantum-resistant digital signatures ensures the integrity and authenticity of transactions, protecting against quantum adversaries. It is also important to implement quantum-resistant distributed ledger technology (DLT) to guarantee secure and decentralized transaction records. Threshold cryptography can be used in blockchain applications to require multiple parties to approve transactions, adding another layer of security against potential quantum attacks.

System and Network Security

Quantum-resistant encryption methods should be applied in the design of virtual private networks (VPNs) to protect data in transit against quantum threats. Critical systems should be isolated from less secure networks, and multiple security layers should be implemented to minimize the impact of any potential quantum attack. Developing quantum-specific firewalls can help filter and protect quantum communication channels, ensuring that unauthorized quantum-based attacks are blocked. Additionally, secure multi-party computation (MPC) protocols can be used in cloud environments, providing an extra layer of protection for sensitive data.

System Integration and Updates

Cryptographic systems should be designed in a modular fashion to allow for quick updates and replacement of cryptographic algorithms as quantum computing evolves. Software frameworks that support multiple cryptographic algorithms allow for seamless transitions between older and newer security protocols with minimal disruption. Regular quantum-resistance checks should be incorporated into the software development lifecycle (SDLC) and code review processes to ensure security remains intact. Furthermore, quantum-safe security measures should be integrated into continuous integration/continuous deployment (CI/CD) pipelines, ensuring that updates are rolled out efficiently and securely.

Access Control and Security Governance

Role-based access control (RBAC) and attribute-based access control (ABAC) should be implemented with quantum-safe cryptographic protection to secure access to sensitive information. These access control models help ensure that only authorized users have access to critical systems, preventing unauthorized access in a quantum computing era.

Summary

This module provides an overview of fundamental cryptography concepts used to secure confidential data, along with an exploration of different types of cryptographic techniques. It covers ciphers and various encryption algorithms in detail, explaining how they are used for encryption and decryption. The module also introduces a range of cryptographic tools and emphasizes the significance of Public Key Infrastructure (PKI) in encryption. Additionally, it delves into email encryption protocols and tools, as well as disk encryption and the associated tools. The module further explains various cryptanalysis methods, including code-breaking techniques, and highlights different types of cryptanalysis attacks and the corresponding tools. Lastly, it concludes with a discussion of countermeasures to protect against cryptographic attacks.

Mind Map

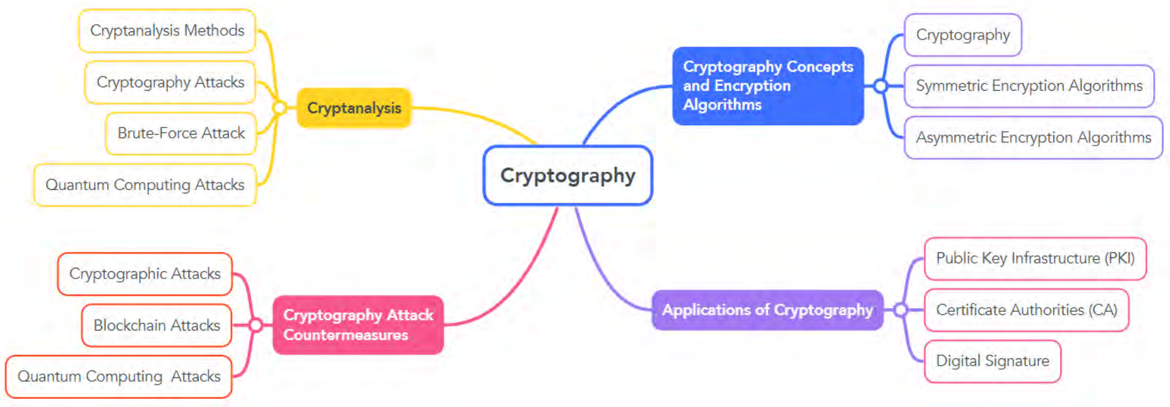

Cryptography
Cryptanalysis
Cryptanalysis Methods
Cryptography Attacks
Brute-Force Attack
Quantum Computing Attacks
Cryptography Concepts and Encryption Algorithms
Cryptography
Symmetric Encryption Algorithms
Asymmetric Encryption Algorithms
Cryptography Attack Countermeasures
Cryptographic Attacks
Blockchain Attacks
Quantum Computing Attacks
Applications of Cryptography
Public Key Infrastructure (PKI)
Certificate Authorities (CA)
Digital Signature

Figure 20-14: Mind Map

Practice Questions

1. Which of the following is not a primary objective of cryptography?

A. Confidentiality
B. Integrity
C. Compression
D. Authentication

2. Which of the following statements about symmetric cryptography is true?
A. It uses a pair of keys for encryption and decryption.
B. AES and DES are common symmetric algorithms.
C. It is less efficient than asymmetric cryptography for large data encryption.
D. It is also known as public key cryptography.

3. What is the primary function of Government Access to Keys (GAK)?
A. To generate encryption keys for public use.
B. To allow government agencies access to cryptographic keys under legal conditions.
C. To ensure encryption is used only for secure transactions.
D. To enhance the speed of encryption algorithms.

4. Which of the following correctly differentiates a block cipher from a stream cipher?
A. Block ciphers encrypt data one bit at a time, while stream ciphers encrypt data in blocks.
B. Block ciphers are always faster than stream ciphers.
C. Block ciphers process data in fixed-size chunks, while stream ciphers encrypt one bit or byte at a time.
D. Stream ciphers use asymmetric encryption, while block ciphers use symmetric encryption.

5. Why was AES developed to replace DES?
A. DES was too slow for modern encryption needs.
B. DES had a small key size that made it vulnerable to brute-force attacks.
C. AES was designed to work only with asymmetric cryptography.
D. DES was never a secure encryption method.

6. Which of the following is NOT an asymmetric encryption algorithm?
A. RSA
B. AES
C. Diffie-Hellman
D. ECC

7. What is the primary advantage of using asymmetric encryption over symmetric encryption?
A. Faster encryption speed
B. Requires only one key for encryption and decryption

C. Eliminates the need for secure key exchange
D. Uses shorter key lengths for better security

8. Which asymmetric encryption algorithm is primarily used for secure key exchange rather than direct encryption?
A. RSA
B. Diffie-Hellman
C. ECC
D. DSA

9. Which cryptographic technique is used in digital signatures to ensure message integrity and authenticity?
A. Public key encryption
B. Hash functions
C. Symmetric encryption
D. Stream ciphers

10. Elliptic Curve Cryptography (ECC) is considered an improvement over RSA because:
A. ECC keys are shorter while providing the same security level
B. ECC encryption is faster than RSA for all key sizes
C. ECC uses a symmetric key instead of a public-private key pair
D. ECC does not require a key exchange process

11. Which of the following is NOT a common application of cryptography?
A. Data encryption for protecting files and cloud storage
B. Email message encryption using PGP
C. Online gaming multiplayer session management
D. Secure authentication through multi-factor authentication (MFA)

12. Which cryptographic protocol is commonly used to secure web traffic and provide encryption over HTTPS?
A. AES
B. SSL/TLS
C. RSA
D. SHA-256

13. What is the primary purpose of a Public Key Infrastructure (PKI)?
A. To protect blockchain data
B. To manage digital certificates and enable secure communication
C. To store cryptographic keys
D. To verify email signatures

14. Which of the following is a key difference between a signed certificate and a self-signed certificate?
A. A signed certificate is created by the certificate holder, while a self-signed certificate is issued by a third-party CA.
B. A signed certificate is issued by a third-party Certificate Authority, while a self-signed certificate is issued by the entity itself.
C. A signed certificate is used for disk encryption, while a self-signed certificate is used for email encryption.
D. A signed certificate does not use public key encryption, while a self-signed certificate does.

15. Which cryptographic method is used in blockchain technology to ensure the integrity of data and transactions?
A. Hash functions
B. Symmetric encryption
C. Public key encryption
D. Digital signatures

16. Which protocol provides end-to-end encryption for email communications, ensuring confidentiality and integrity?
A. SSL/TLS
B. OpenPGP
C. HTTP
D. FTP

17. What is the primary function of a Certificate Authority (CA) in Public Key Infrastructure (PKI)?
A. To store encrypted data
B. To verify digital signatures
C. To issue and verify digital certificates
D. To encrypt and decrypt data

18. Which of the following disk encryption tools is commonly used to encrypt entire disk partitions and protect sensitive data?
A. VeraCrypt
B. S/MIME
C. PGP
D. SSL/TLS

19. Which of the following cryptanalysis techniques is based on finding linear approximations to the action of a cipher, particularly block ciphers?
A. Differential Cryptanalysis
B. Linear Cryptanalysis

C. Quantum Cryptanalysis
D. Integral Cryptanalysis

20. What type of cryptographic attack involves a cryptanalyst having access to a ciphertext but not the corresponding plaintext, and attempting to recover the plaintext or key?
A. Known Plaintext Attack
B. Ciphertext-only Attack
C. Chosen Plaintext Attack
D. Chosen Ciphertext Attack21.

21. Which cryptanalysis technique involves exploiting the mathematical relationships between keys used in a cipher to gain access to both encryption and decryption functions?
A. Related-Key Attack
B. Birthday Attack
C. Padding Oracle Attack
D. Side-Channel Attack

22. What is a major vulnerability in cryptographic systems where attackers exploit the quantum computing algorithms to break traditionally secure systems?
A. Rainbow Table Attack
B. Quantum Cryptanalysis Attack
C. DUHK Attack
D. Meet-in-the-Middle Attack

23. Which of the following is NOT a key practice in managing cryptographic key security? A. Encrypting stored keys with passphrases
B. Embedding keys in source code
C. Regularly rotating cryptographic keys
D. Using hardware-backed security like HSMs

24. Which of the following encryption algorithms is recommended for symmetric encryption according to best practices? A. MD5
B. AES
C. RSA
D. SHA-1

25. Which key stretching technique is based on the Blowfish cipher and helps protect against rainbow table attacks?

A. PBKDF2
B. Scrypt
C. Bcrypt
D. SHA-256

Answers

1. Answer: C
Explanation: Cryptography focuses on confidentiality, integrity, authentication, and non-repudiation, but not data compression.

2. Answer: B
Explanation: Symmetric cryptography uses a single key for encryption and decryption, and AES and DES are widely used symmetric encryption algorithms.

3. Answer: B

Explanation: GAK is a legislative requirement where individuals and organizations must disclose cryptographic keys to government agencies for lawful monitoring.

4. Answer: C

Explanation: Block ciphers operate on fixed-length blocks of data, whereas stream ciphers encrypt data bit-by-bit or byte-by-byte.

5. Answer: B

Explanation: DES has a 56-bit key size, which is now considered too small and vulnerable to brute-force attacks, leading to the development of AES with stronger key sizes.

6. Answer: B

Explanation: Advanced Encryption Standard (AES) is a symmetric encryption algorithm, while RSA, Diffie-Hellman, and ECC are asymmetric encryption algorithms.

7. Answer: C

Explanation: Asymmetric encryption eliminates the need for a shared secret key, as it uses a public-private key pair for encryption and decryption.

8. Answer: B

Explanation: Diffie-Hellman is primarily used for secure key exchange rather than direct encryption or signing.

9. Answer: B

Explanation: Hash functions generate a fixed-length hash value from the message, which is then signed using a private key to ensure integrity and authenticity.

10. Answer: A

Explanation: ECC provides the same level of security as RSA but with much shorter key lengths, making it more efficient in terms of performance and resource usage.

11. Answer: C

Explanation: Cryptography is primarily used for securing data, authenticating users, and encrypting communications. Online gaming multiplayer sessions do not typically require cryptography for security.

12. Answer: B

Explanation: Secure Socket Layer (SSL) and Transport Layer Security (TLS) are protocols used to secure web traffic by encrypting the communication between clients and servers over HTTPS.

13. Answer: B

Explanation: PKI is designed to manage digital certificates, enabling secure communication over networks by using public and private key pairs issued by trusted Certificate Authorities.

14. Answer: B

Explanation: A signed certificate is issued by a trusted third-party Certificate Authority (CA), whereas a self-signed certificate is created and issued by the entity that owns it.

15. Answer: A

Explanation: Hash functions, such as SHA-256, are used in blockchain technology to create a secure, immutable record of transactions and ensure data integrity.

16. Answer: B

Explanation: OpenPGP is a widely used email encryption standard that provides end-to-end encryption for email messages, ensuring their confidentiality and integrity.

17. Answer: C

Explanation: The Certificate Authority (CA) issues and verifies digital certificates, which are essential for ensuring the authenticity of users and their public keys in secure communications**.**

18. Answer: A

Explanation: VeraCrypt is a popular disk encryption tool used to encrypt entire disk partitions to protect sensitive data stored on disks.

19. Answer: B

Explanation: Linear cryptanalysis is based on finding linear approximations to the action of a cipher, especially block ciphers.

20. Answer: B

Explanation: In a Ciphertext-only Attack, the attacker only has the ciphertext and attempts to recover the plaintext or key.

21. Answer: A

Explanation: A Related-Key Attack exploits the mathematical relationships between keys used in a cipher to break the cryptosystem.

22. Answer: B

Explanation: Quantum Cryptanalysis attacks use quantum computing algorithms like Shor's and Grover's algorithms to break traditional cryptographic systems.

23. Answer: B

Explanation: Embedding keys in source code is a practice to avoid, as it exposes keys to potential attackers.

24. Answer: B

Explanation: Advanced Encryption Standard (AES) is recommended for symmetric encryption due to its strength and efficiency.

25. Answer: C

Explanation: Bcrypt is based on the Blowfish cipher and is specifically designed to protect against rainbow table attacks by salting passwords and slowing down brute-force attempts.