Unit-04

Cryptography

4.1 Cryptography

Cryptography is the science and art of securing information by transforming it into a format that is unreadable to anyone except those possessing the appropriate key. This process ensures the confidentiality, integrity, and authenticity of the information. Cryptography is widely used to protect sensitive data in various fields, including communications, finance, and digital transactions.

Modular Arithmetic and Its Properties

Modular arithmetic is a system of arithmetic for integers, where numbers "wrap around" after reaching a certain value, known as the modulus. It is sometimes referred to as "clock arithmetic" because of its similarity to the way a clock resets after reaching 12.

Basic Properties of Modular Arithmetic

1. **Addition**:

   If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then:

   $$(a + c) \equiv (b + d) \pmod{n}$$

   Example: $(7 + 9) \bmod 5 = 16 \bmod 5 = 1$

2. **Subtraction**:
   If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then:

   $$(a - c) \equiv (b - d) \pmod{n}$$

   Example: $(7 - 9) \bmod 5 = -2 \bmod 5 = 3$

3. **Multiplication**:
   If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then:

   $$(a \cdot c) \equiv (b \cdot d) \pmod{n}$$

   Example: $(7 \cdot 3) \bmod 5 = 21 \bmod 5 = 1$

4. **Exponentiation**:
   If a ≡ b (mod n), then:
   a^k ≡ b^k (mod n)
   Example: 2^3 ≡ 8 (mod 5) ≡ 3 (mod 5).

5. **Division**:
   Division in modular arithmetic is only possible if the divisor has a multiplicative inverse modulo n.
   Example: The inverse of 3 modulo 7 is 5 because $3 \times 5 \equiv 1 \pmod 7$

The Euclidean Algorithm

The **Euclidean algorithm** is an efficient method for computing the greatest common divisor (GCD) of two integers. The GCD of two numbers is the largest number that divides both of them without leaving a remainder. The Euclidean algorithm is based on the principle that the GCD of two numbers also divides their difference.

Steps of the Euclidean Algorithm

Given two integers a and b where a>b

- Divide *aa*a by *b*, and let the remainder be *r* (i.e., $a=b\times q+r$ where *q* is the quotient and *rr*r is the remainder).
- Replace *aa*a with *b* and *b* with *r*.
- Repeat the process until the remainder *r* is 0.
- The GCD is the last non-zero remainder.

Example

Find the GCD of 252 and 105:

1. $252 \div 105 = 2$ remainder 42 (so, $252 = 105 \times 2 + 42$)
2. $105 \div 42 = 2$ remainder 21 (so, $105 = 42 \times 2 + 21$)
3. $42 \div 21 = 2$ remainder 0 (so, $42 = 21 \times 2 + 0$)

The last non-zero remainder is 21, so $\mathrm{GCD}(252, 105) = 21$.

Fermat's Little Theorem

Fermat's Little Theorem is a fundamental result in number theory that provides a useful property of prime numbers in the context of modular arithmetic. It states:

If p is a prime number and a is an integer such that a is not divisible by p, then:

$$a^{p-1} \equiv 1 \pmod{p}$$

Or equivalently:

$$a^p = a \pmod{p}$$

Applications

- **Primality Testing**: Fermat's Little Theorem is used in some primality tests to determine if a number is prime.
- **RSA Cryptography**: The theorem is applied in the RSA algorithm to ensure the correctness of modular inverses.

4.2. Classical cryptography

Classical cryptography refers to the study and use of encryption techniques before the modern era of computers. These methods were primarily manual or mechanical and were used for secure communication. Despite being simpler than modern cryptographic techniques, they laid the foundation for the advanced encryption methods we use today.

Techniques in Classical Cryptography

1. **Substitution Ciphers**:

   In a substitution cipher, each letter or group of letters in the plaintext is replaced by another letter or group of letters. The substitution can be based on a fixed system, such as shifting letters by a certain number in the alphabet.

   - **Monoalphabetic Cipher**:
     Each letter in the plaintext is substituted with another letter, but the mapping is arbitrary rather than shifted.
     Example: A might be replaced with Q, B with W, and so forth.

   - **Polyalphabetic Ciphers**:
     These ciphers use multiple substitution alphabets to encrypt the plaintext, making frequency analysis more difficult.

Example:

Keyword: LEMON

Plaintext: HELLO

Ciphertext: SPWWZ

The keyword is repeated to match the length of the plaintext, and each letter in the plaintext is shifted according to the corresponding letter in the keyword.

2. **Transposition Ciphers**:

   In a transposition cipher, the positions of the characters in the plaintext are shifted according to a regular system, but the actual characters remain unchanged. This type of cipher focuses on rearranging the order of characters.

   Examples:

- **Rail Fence Cipher**: The plaintext is written in a zigzag pattern across multiple "rails" and then read off row by row to create the ciphertext.
  - **Plaintext**: MEETMEATNOON
  - **Ciphertext (using 3 rails)**: MTAEOTEMNENO
- **Columnar Transposition**: The plaintext is written out in rows and then read off by columns based on a predetermined key.
  - **Plaintext**: ATTACKATDAWN
  - **Key**: 3, 1, 4, 2 (column order)
  - **Ciphertext**: TACTAKWADATN
    -

3. **One-Time Pad**

   The one-time pad is a theoretically unbreakable encryption method when used correctly. Each character in the plaintext is encrypted by combining it with a corresponding character from a randomly generated key of the same length using modular arithmetic (usually XOR for binary data). The key is only used once and must remain secret.

   Example:

- **Plaintext**: HELLO
- **Key**: XMCKL

- **Ciphertext**: EQNVZ (Each letter is combined with the key using modular addition)

4.3. Modern Cryptography

Modern cryptography refers to the field of cryptography developed with the advent of computers, focusing on techniques that secure data in the digital age. Unlike classical cryptography, which primarily dealt with securing handwritten messages, modern cryptography is designed to protect digital information and communications. It involves mathematical algorithms, complex keys, and advanced protocols that provide security guarantees like confidentiality, integrity, authentication, and non-repudiation.

Modern cryptography is built upon a few fundamental principles that ensure secure communication, data integrity, and authentication in digital environments. These principles form the foundation of cryptographic systems and guide the development and application of cryptographic algorithms and protocols.

1. Confidentiality

Ensures that information is only accessible to those who are authorized to view it. Confidentiality is achieved through encryption, where data is transformed in such a way that only authorized parties can decrypt and read it. **Example**: Encrypting a message so that only the intended recipient can read it.

2. Integrity

Ensures that the data has not been altered or tampered with during transmission or storage. Integrity is typically verified using cryptographic hash functions, checksums, or message authentication codes (MACs). **Example**: A hash of a file is calculated before and after transmission; if the hashes match, the file has not been altered.

3. Authentication

Verifies the identity of the parties involved in communication or the origin of a message. Authentication can involve passwords, digital certificates, or cryptographic signatures. **Example**: Using a digital signature to verify that a document was sent by a trusted sender.

4. Non-repudiation

Ensures that a party cannot deny the authenticity of their signature on a document or a message they sent. This is often achieved through digital signatures. **Example**: A signed contract cannot be denied by the person who signed it.

Symmetric Cryptography

Symmetric cryptography, also known as **private-key cryptography**, is a method of encryption where the same key is used for both encrypting and decrypting data. This approach is efficient and fast, making it ideal for scenarios where large amounts of data need to be encrypted. The primary challenge lies in securely distributing and managing the encryption key.

Features

- **Single Key**: The same key is used for both encryption and decryption.
- **Speed**: Fast encryption and decryption processes.
- **Confidentiality**: Security depends on keeping the key secret.

Asymmetric Cryptography

Asymmetric cryptography, or **public-key cryptography**, involves the use of two different but mathematically related keys: a public key and a private key. The public key is used for encryption, while the private key is used for decryption. This system addresses the key distribution problem inherent in symmetric cryptography.

Features:

- **Two Keys**: Public key for encryption, private key for decryption.
- **Security**: Enhanced by the difficulty of deriving the private key from the public key.
- **Scalability**: Suitable for secure communication in large networks.

No	Symmetric Key Cryptography	Asymmetric Key Cryptography
1	Single key is used for encryption and decryption.	Two different key is used, one for encryption and other for decryption.
2	It is also called secret key or private key cryptography.	It is also called public key cryptography or conventional cryptographic system.
3	It is faster than asymmetric key cryptography.	It is slower than symmetric key cryptography.
4	It uses less resource in compare to asymmetric key cryptography.	It uses more resource in compare to symmetric key cryptography.
5	For encryption of large message symmetric key cryptography is used.	In asymmetric key cryptography plain text and cipher text treated as integer number.
6	For example: DES, AES and BLOWFISH	For example, RSA and Diffie-Hellman Key exchange.
7	Mathematically it is represented as C = E(K, P) and P = D(K, C) i.e., C = Cipher text, E = Encryption, D = Decryption, P = Plain text, K =Secret Key	Mathematically it is represented as C = E(Pu(R), P) and P = D(Pr(R), C) i.e., C = Cipher text, E = Encryption, D = Decryption, P = Plain text, Pr(R) =Private Key of receiver, Pu(R) = Public key of receiver.

Hash Functions

Hash functions are mathematical algorithms that take an input (or "message") and return a fixed-size string of bytes, typically a digest that is unique to each unique input. Hash functions are designed to be fast and irreversible, meaning that it should be computationally infeasible to reconstruct the original input from the hash value or find two different inputs that produce the same hash value (known as a collision).

Features:

- **Deterministic**: The same input always produces the same output.
- **Fixed Size**: Output (hash) is of fixed size, regardless of the input size.
- **Irreversibility**: Cannot easily determine the original input from the hash.
- **Collision Resistance**: It is difficult to find two different inputs that produce the same hash.

4.4. Private-Key Cryptography

Private-key cryptography, also known as **symmetric-key cryptography**, is a type of encryption where the same key is used for both encryption and decryption of data. This method is simple and efficient, making it well-suited for encrypting large amounts of data quickly. The major challenge with private-key cryptography is securely distributing and managing the keys, especially in large networks.

4.5. Public-Key Cryptography

Public-key cryptography, also known as **asymmetric cryptography**, involves the use of two keys: a public key, which can be shared openly, and a private key, which is kept secret. Unlike private-key cryptography, these keys are mathematically related but are not identical, allowing secure communication without the need to share a secret key in advance.

4.6. The RSA System

The RSA system (Rivest-Shamir-Adleman) is one of the most widely used public-key cryptosystems, primarily for securing sensitive data and establishing secure communications over the internet. It was invented in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT and is based on the mathematical difficulty of factoring large composite numbers, specifically the product of two large prime numbers.

The RSA algorithm

The RSA algorithm is a widely used encryption and decryption system in modern cryptography, especially for secure data transmission. It is based on the mathematical difficulty of factoring large prime numbers. The core of the RSA algorithm involves modular arithmetic, specifically working with a large modulus.

Modulus (n): The modulus is the product of two large prime numbers, typically denoted as p and q. It is a crucial part of both the public and private keys.

- $n = p \times q$
- The modulus n determines the size of the key and thus the security level of the RSA algorithm.

Public Key: Consists of the modulus *nn*n and an exponent *ee*e.

- *(e, n)*

- The public key is used for encryption and can be distributed openly.

Private Key: Consists of the modulus *nn*n and an exponent *dd*d.

- *(d, n)*
- The private key is used for decryption and must be kept secret.

Example

1. **Key Generation**:
   - Choose primes $p = 61$ and $q = 53$.
   - Compute $n = 61 \times 53 = 3233$.
   - Calculate $\phi(n) = (61 - 1) \times (53 - 1) = 3120$.
   - Choose $e = 17$ (which is coprime with 3120).
   - Calculate d such that $17 \times d \equiv 1 \pmod{3120}$. Here, $d = 2753$.

   The public key is $(17, 3233)$ and the private key is $(2753, 3233)$.
2. **Encryption**:
   - Message $M = 65$.
   - Compute $C = 65^{17} \pmod{3233} = 2790$.
3. **Decryption**:
   - Compute $M = 2790^{2753} \pmod{3233} = 65$.

Role of the Modulus (n)

The modulus *n* in RSA plays a crucial role in the encryption and decryption process. Its size directly influences the security of the RSA algorithm. The larger the modulus, the more secure the encryption, but this also increases the computational resources required. Since factoring large numbers (finding *p* and *q* from *n*) is computationally difficult, this makes RSA secure when large enough primes are used.