

Explain the Security attacks with its types.

2. Explain the objectives of Computer Security.

3. Write a short note on:

Integrity

Confidentiality

Non-repudiation

Access Control

Availability

4. Explain the passive attacks with its type.

5. Explain Active attacks with its type.
6. Explain the Symmetric Cipher model in brief.
7. Explain DES in brief.
8. Explain in brief Symmetric key and Asymmetric key cryptography.
9. Explain in brief two Algorithm types.
10. Explain modes of operation in block ciphers or algorithmic modes in detail.
11. Explain the different variations in DES.
12. Explain Diffie-Hellman algorithm in network security.

13. Explain the RSA algorithm with suitable example.

Give all these answer for 5 marks each each answer should be in proper format this subject is software testing and quality assurance of bsc cs sem 5 mumbai University give good easy to learn answer for each

Show thinking

Of course! Here are easy-to-learn answers for your questions, tailored for a 5-mark format for a BSc CS course.

1. Security Attacks and Their Types

A **security attack** is any action that compromises the security of information owned by an organization. These attacks are broadly classified into two main categories: **Passive Attacks** and **Active Attacks**.

- **Passive Attacks:** These attacks involve eavesdropping or monitoring transmissions. The attacker's goal is to obtain information without altering the system's resources or data. They are difficult to detect because they don't change anything.
 - **Release of Message Contents:** This is like wiretapping a phone line to hear the conversation. The attacker learns the content of the information being transmitted (e.g., reading an email).
 - **Traffic Analysis:** Even if the message is encrypted, an attacker can observe the pattern of the messages. They can determine the location and identity of communicating hosts, how often they communicate, and the length of messages, which can reveal sensitive information.
- **Active Attacks:** These attacks involve some modification of the data stream or the creation of a false stream. They are easier to detect than passive attacks but can cause more direct damage.
 - **Masquerade (Spoofing):** One entity pretends to be a different entity. For example, an attacker might capture a user's login credentials and use them to log in as that user.
 - **Replay:** The attacker passively captures a data unit (like a login request) and retransmits it later to produce an unauthorized effect.
 - **Modification of Messages:** The attacker alters a legitimate message by deleting, adding, or changing parts of it. For example, changing a message from "Allow John to access files" to "Allow Fred to access files".

- **Denial of Service (DoS):** The attacker prevents or inhibits the normal use or management of communication facilities. For example, flooding a server with so many requests that it crashes and becomes unavailable to legitimate users.
-

2. Objectives of Computer Security

The primary goal of computer security is to protect information and systems from unauthorized access, use, disclosure, disruption, modification, or destruction. This is achieved through a set of core objectives, often called the **pillars of security**.

The main objectives are:

1. **Confidentiality (Secrecy):** This ensures that information is not disclosed to unauthorized individuals, entities, or processes. It's about keeping secrets safe.
 - **Example:** Encrypting your credit card number during an online transaction prevents an eavesdropper from reading it.
 2. **Integrity:** This ensures that data is accurate and has not been altered or destroyed in an unauthorized manner. It guarantees the trustworthiness of data.
 - **Example:** When you download a file, a checksum (hash) is often provided. If the hash of your downloaded file matches the original, its integrity is verified.
 3. **Availability:** This ensures that systems and data are accessible and usable upon demand by an authorized user. It means the services are available when you need them.
 - **Example:** A website being protected from a Denial of Service (DoS) attack ensures that legitimate users can still access it.
 4. **Authenticity:** This is the property of being genuine and being able to be verified and trusted. It confirms that the users are who they say they are and that the data comes from its claimed source.
 - **Example:** Using a password or a fingerprint to log in to your phone verifies your identity.
 5. **Non-repudiation:** This ensures that a party to a transaction cannot deny having received a transaction, nor can the other party deny having sent a transaction. It provides proof of origin and delivery.
 - **Example:** A digital signature on a document ensures that the sender cannot later claim they never signed it.
-

3. Short Note On

Integrity

Integrity is a core security objective that ensures the accuracy and completeness of data. It guarantees that information has not been altered, tampered with, or corrupted in an unauthorized way. It's about maintaining the trustworthiness of information over its entire lifecycle. For

example, if a bank's database shows your account balance as ₹10,000, integrity ensures that an attacker cannot change it to ₹100 without authorization. Mechanisms like hashing algorithms (e.g., SHA-256) and access controls are used to maintain data integrity.

Confidentiality

Confidentiality, also known as secrecy, is the objective of preventing the unauthorized disclosure of sensitive information. It ensures that only authorized users can access and view the data. This is perhaps the most common aspect of security people think of. For example, when you send a private message on a social media app, confidentiality ensures that no one else can read its contents. The most common tool to achieve confidentiality is **encryption**.

Non-repudiation

Non-repudiation provides proof that a specific action was performed by a specific entity, making it impossible for that entity to deny it later. It is crucial for e-commerce, legal documents, and financial transactions. It provides proof of origin and proof of delivery. For instance, when you digitally sign a contract, non-repudiation prevents you from later claiming that you never signed it. This is typically implemented using **digital signatures**.

Access Control

Access Control is the security mechanism that regulates who or what can view or use resources in a computing environment. It is the selective restriction of access to a place or other resource. The process involves three steps:

1. **Identification:** Who are you? (e.g., your username).
 2. **Authentication:** Can you prove it? (e.g., your password or fingerprint).
 3. **Authorization:** What are you allowed to do? (e.g., read-only access vs. read/write access). A simple example is logging into your computer: you provide a username (identification) and a password (authentication), which then grants you access to your files (authorization).
-

Availability

Availability ensures that information and computer systems are accessible and operational for authorized users whenever they are needed. It means preventing service disruptions and ensuring timely and reliable access to resources. Threats to availability include hardware failures, software

bugs, and attacks like Denial of Service (DoS), where an attacker floods a server with traffic to make it crash. Ensuring availability involves implementing redundancy (like backup servers), disaster recovery plans, and protection against DoS attacks.

4. Passive Attacks with Its Types

Passive attacks are security attacks where the attacker's goal is to learn or make use of information from the system but does not affect system resources. The attacker simply eavesdrops on or monitors transmissions. These attacks are very difficult to detect because they do not involve any alteration of the data.

The primary focus of prevention for passive attacks is on **encryption**, which makes the intercepted data unintelligible.

There are two main types of passive attacks:

1. **Release of Message Contents:**

- This attack involves an adversary listening to a communication channel and understanding the content of the messages being exchanged.
- It is like tapping a phone line and listening to the conversation or reading someone's emails without their knowledge.
- The primary defense against this is to encrypt the messages, so even if the attacker intercepts them, they cannot be read.

2. **Traffic Analysis:**

- In this attack, even when the message content is encrypted and cannot be read, the attacker can still gain intelligence by observing the flow of information.
- The attacker observes the source and destination of messages, the frequency and length of the messages, and other patterns.
- For example, a sudden increase in encrypted communication between a company's headquarters and its legal team might suggest a major lawsuit or acquisition is underway. This information can be valuable even without knowing the exact content.

5. Active Attacks with Its Types

Active attacks involve some modification of the data stream or the creation of a false stream. In these attacks, the adversary actively tries to alter system resources or affect their operation. Active attacks are easier to detect than passive attacks, as they create changes in the system. The focus of protection is on **detection** and **recovery**.

The four main types of active attacks are:

1. **Masquerade (Spoofing):**
 - This attack occurs when one entity pretends to be a different entity. The attacker uses someone else's identity to gain unauthorized access to a system.
 - **Example:** An attacker uses a stolen username and password to log into a system as a legitimate user.
 2. **Replay:**
 - A replay attack involves the passive capture of a data unit (e.g., an authenticated login session) and its subsequent retransmission to produce an unauthorized effect.
 - **Example:** An attacker captures the data packets of a user logging into their bank account and later "replays" them to the bank's server to gain access.
 3. **Modification of Messages:**
 - This attack involves an attacker intercepting a message and changing its content before forwarding it to the intended recipient.
 - **Example:** An attacker intercepts a financial transaction message "Pay ₹1,000 to John" and modifies it to "Pay ₹1,000 to Attacker". This is also known as a **Man-in-the-Middle (MITM)** attack.
 4. **Denial of Service (DoS):**
 - This attack aims to make a machine or network resource unavailable to its intended users.
 - **Example:** An attacker floods a web server with an overwhelming number of requests, causing it to crash or become so slow that legitimate users cannot access it.
-

6. Explain the Symmetric Cipher Model in Brief

The **Symmetric Cipher Model**, also known as conventional or single-key encryption, is a cryptographic system where the same key is used for both **encryption** and **decryption**. This shared key must be kept secret between the sender and the receiver.

The model has five main components:

1. **Plaintext:** The original, readable message or data that is fed into the algorithm as input.
2. **Encryption Algorithm:** The algorithm that performs various substitutions and transformations on the plaintext to convert it into ciphertext.
3. **Secret Key:** A value independent of the plaintext and algorithm, which is also an input to the encryption algorithm. The key controls the specific transformations used.
4. **Ciphertext:** The scrambled, unreadable message produced as output. It depends on the plaintext and the secret key.
5. **Decryption Algorithm:** Essentially the reverse of the encryption algorithm. It takes the ciphertext and the same secret key to produce the original plaintext.

How it works:

- **Encryption:** $C = EK(P)$

- Where C is the Ciphertext, E is the Encryption Algorithm, K is the Secret Key, and P is the Plaintext.
- **Decryption:** $P = D_K(C)$
 - Where P is the Plaintext, D is the Decryption Algorithm, K is the same Secret Key, and C is the Ciphertext.

The main challenge with symmetric encryption is the **key distribution problem**: how to securely share the secret key between the sender and receiver without a third party intercepting it. Examples of symmetric algorithms include **DES**, **AES**, and **Blowfish**.

7. Explain DES in Brief

DES (Data Encryption Standard) is a symmetric-key block cipher that was once a federal standard in the United States. Although now considered insecure for many modern applications, it was highly influential in the advancement of cryptography.

Here are its key characteristics:

- **Algorithm Type:** Symmetric-key block cipher.
- **Block Size:** It encrypts data in **64-bit blocks**. This means it takes a 64-bit chunk of plaintext as input and produces a 64-bit chunk of ciphertext as output.
- **Key Size:** It uses a **56-bit key**. The original key provided is 64 bits, but 8 of those bits are used for parity checking and are discarded, leaving an effective key length of 56 bits.
- **Structure:** DES is based on a **Feistel Network**. This structure ensures that the encryption and decryption processes are very similar, which makes implementation easier.
- **Rounds:** The algorithm performs **16 rounds** of complex operations. In each round, a different 48-bit subkey (derived from the main 56-bit key) is used to perform substitutions and permutations on the data block.

General Process:

1. The 64-bit plaintext block undergoes an **Initial Permutation (IP)**.
2. The block is then put through **16 rounds** of encryption. Each round involves splitting the block into two 32-bit halves, applying a combination of substitution (using S-boxes) and permutation (P-boxes) functions, and XORing the result with the other half.
3. Finally, the block goes through a **Final Permutation (FP)**, which is the inverse of the initial permutation.

Weakness: The primary weakness of DES is its **small key size (56 bits)**. With modern computing power, a 56-bit key can be cracked using a brute-force attack in a relatively short amount of time. This led to its replacement by algorithms like **3DES** and ultimately **AES (Advanced Encryption Standard)**.

8. Explain in Brief Symmetric Key and Asymmetric Key Cryptography

Symmetric and Asymmetric are the two fundamental types of cryptography, distinguished by the keys they use.

Symmetric Key Cryptography

Also known as **secret-key** or **private-key cryptography**.

- **Keys:** It uses a **single, shared secret key** for both encryption and decryption. Both the sender and the receiver must have a copy of this same key.
- **Analogy:** It's like using a physical key to lock and unlock a box. The same key that locks the box is needed to unlock it.
- **Speed:** It is very **fast** and efficient, making it suitable for encrypting large amounts of data (e.g., entire files or database backups).
- **Challenge:** The main problem is **key distribution**. How do you securely share the secret key with the recipient without an attacker intercepting it?
- **Examples:** DES, 3DES, AES, Blowfish.

Asymmetric Key Cryptography

Also known as **public-key cryptography**.

- **Keys:** It uses a **pair of keys** for each user: a **public key** and a **private key**. The public key can be shared with anyone, but the private key must be kept secret.
- **Analogy:** It's like having a mailbox with a slot (the public key) and a locked door. Anyone can drop a message in the slot, but only you, with the private key, can open the door to read the messages.
- **Speed:** It is mathematically complex and much **slower** than symmetric cryptography. It is generally used for encrypting small amounts of data, like sharing symmetric keys or creating digital signatures.
- **Challenge:** It solves the key distribution problem. To send an encrypted message, you just need the recipient's public key, which is freely available.
- **Examples:** RSA, Diffie-Hellman, ECC (Elliptic Curve Cryptography).

Feature	Symmetric Cryptography	Asymmetric Cryptography
Key Used	Single shared key	Pair of keys (public & private)
Speed	Fast	Slow
Key Management	Difficult (Key Distribution Problem)	Easy (Public key can be shared)
Use Cases	Encrypting large data volumes	Key exchange, digital signatures
Examples	AES, DES, 3DES	RSA, ECC, Diffie-Hellman

Export to Sheets

9. Explain in Brief Two Algorithm Types

In symmetric cryptography, ciphers can be categorized into two main types based on how they process plaintext data: **Block Ciphers** and **Stream Ciphers**.

1. Block Ciphers

A **block cipher** is an algorithm that operates on fixed-length groups of bits, called **blocks**. It takes a block of plaintext (e.g., 64 or 128 bits) and a key as input and produces a corresponding block of ciphertext of the same size.

- **Operation:** It encrypts data one block at a time. If a message is larger than the block size, it's broken into multiple blocks. If the final part of the message is smaller than a full block, it must be padded to the correct size.
- **Characteristics:**
 - Works on fixed-size chunks of data.
 - The same key is used for each block (though modes of operation can add variation).
 - Widely used and well-analyzed.
- **Modes of Operation:** Because encrypting identical plaintext blocks with the same key produces identical ciphertext blocks (a security risk), block ciphers are used with different "modes of operation" (like CBC, ECB, CTR) to introduce randomness.
- **Examples:** DES (64-bit block), AES (128-bit block), 3DES.

2. Stream Ciphers

A **stream cipher** is an algorithm that encrypts plaintext one bit or one byte at a time. It works by generating a pseudorandom stream of bits, called the **keystream**, which is then combined with the plaintext (usually via an XOR operation) to produce the ciphertext.

- **Operation:** Plaintext bits are encrypted continuously as they become available. Decryption is the same process: the same keystream is generated and XORed with the ciphertext to recover the plaintext.
- **Characteristics:**
 - Encrypts data as a continuous stream.
 - Generally faster than block ciphers.
 - Does not require padding.
 - Good for real-time applications where data arrives in a stream, such as live audio/video feeds.
- **Important Note:** The same keystream should never be used twice with two different plaintexts, as this can break the encryption.
- **Examples:** RC4, ChaCha20, Salsa20.

10. Explain Modes of Operation in Block Ciphers

A **mode of operation** describes how to repeatedly apply a block cipher's single-block operation to securely transform amounts of data larger than a block. Using a block cipher in a naive, repetitive way is insecure. Modes are designed to add randomization and prevent patterns in the plaintext from being visible in the ciphertext.

Here are some common modes of operation:

1. Electronic Codebook (ECB) Mode

This is the simplest mode. The message is divided into blocks, and each block is encrypted independently using the same key.

- **Formula:** $C_i = E_K(P_i)$
- **Advantage:** Simple and allows for parallel encryption of blocks.
- **Disadvantage: Highly insecure.** Identical plaintext blocks will result in identical ciphertext blocks. This means patterns in the plaintext (like a repeated word or a solid color in an image) remain visible in the ciphertext. It is not recommended for most uses.

2. Cipher Block Chaining (CBC) Mode

In CBC mode, each block of plaintext is XORed with the *previous* ciphertext block before being encrypted. For the first block, an **Initialization Vector (IV)** is used in place of a previous ciphertext block. The IV must be random but does not need to be secret.

- **Formula:** $C_i = E_K(P_i \oplus C_{i-1})$ (with $C_0 = IV$)
- **Advantage:** Hides patterns in plaintext because each ciphertext block depends on all preceding plaintext blocks. It is much more secure than ECB.
- **Disadvantage:** Encryption is sequential and cannot be parallelized. An error in one ciphertext block affects the decryption of the next block.

3. Cipher Feedback (CFB) Mode

This mode essentially turns a block cipher into a self-synchronizing stream cipher. It encrypts the *previous ciphertext block*, and the result is XORed with the current plaintext block to create the current ciphertext block. An IV is used for the first step.

- **Formula:** $C_i = P_i \oplus E_K(C_{i-1})$ (with $C_0 = IV$)
- **Advantage:** Can be used as a stream cipher to encrypt data smaller than a full block.
- **Disadvantage:** Encryption is sequential. Transmission errors in one ciphertext block propagate to the next.

4. Output Feedback (OFB) Mode

This mode also turns a block cipher into a stream cipher. It works by creating a keystream that is independent of the plaintext. It does this by repeatedly encrypting an IV. This keystream is then XORed with the plaintext blocks.

- **Formula:** $C_i = P_i \oplus O_i$, where $O_i = EK(O_{i-1})$ (with $O_0 = IV$)
- **Advantage:** Transmission errors in a ciphertext block do not propagate to subsequent blocks (only the corresponding plaintext block is affected). The keystream can be pre-computed.
- **Disadvantage:** If the same IV is used for two different messages with the same key, the encryption is broken.

5. Counter (CTR) Mode

In CTR mode, a "counter" value (a nonce combined with a block number) is encrypted for each block. This encrypted result is then XORed with the plaintext block. The counter is different for each block.

- **Formula:** $C_i = P_i \oplus EK(\text{counter}_i)$
- **Advantage:** Highly efficient. Both encryption and decryption can be done in parallel. It is simple and widely used in modern protocols like TLS.

11. Explain the Different Variations in DES

The primary weakness of the original **Data Encryption Standard (DES)** is its small 56-bit key size, which makes it vulnerable to brute-force attacks. To address this without designing a completely new algorithm, several variations of DES were created.

The main variations are **Double DES (2DES)** and **Triple DES (3DES)**.

1. Double DES (2DES)

This was a straightforward attempt to improve security by applying DES twice with two different keys, K_1 and K_2 .

- **Process:** The plaintext is first encrypted with K_1 , and the resulting ciphertext is then encrypted again with K_2 .
 - **Encryption:** $C = EK_2(EK_1(P))$
 - **Decryption:** $P = DK_1(DK_2(C))$
- **Security Flaw:** One might assume this doubles the key length to 112 bits (56+56), making it much stronger. However, 2DES is vulnerable to a **Meet-in-the-Middle Attack**. An attacker can encrypt the plaintext with all possible K_1 values and decrypt the ciphertext with all possible K_2 values, and then look for a match in the middle. This reduces the effective security to only 257, which is only slightly better than single DES. Because of this vulnerability, 2DES is not used.

2. Triple DES (3DES or TDES)

This variation applies the DES algorithm three times in a row and is significantly more secure. It became a widely used standard to replace DES. There are two main versions:

- **3DES with 3 Keys (Keying Option 1):**
 - This version uses three independent keys: K_1 , K_2 , and K_3 .
 - **Process:** It follows an **Encrypt-Decrypt-Encrypt (EDE)** sequence.
 - **Encryption:** $C = E_{K_3}(D_{K_2}(E_{K_1}(P)))$
 - **Effective Key Length:** $56 \times 3 = 168$ bits. This provides a very high level of security against brute-force attacks.
- **3DES with 2 Keys (Keying Option 2):**
 - This is the most widely used version. It uses only two keys, where the first and third keys are the same ($K_1 = K_3$).
 - **Process:** It also follows an EDE sequence.
 - **Encryption:** $C = E_{K_1}(D_{K_2}(E_{K_1}(P)))$
 - **Effective Key Length:** $56 \times 2 = 112$ bits. This is still considered strong enough for most commercial applications.

Why EDE sequence? The Encrypt-Decrypt-Encrypt sequence was chosen for backward compatibility. If you set $K_1 = K_2 = K_3$, the 3DES operation becomes a standard single DES encryption, allowing systems with 3DES to communicate with older DES-only systems.

While 3DES is much slower than single DES, its security made it a popular choice until it was eventually superseded by the more efficient and secure **AES (Advanced Encryption Standard)**.

12. Explain Diffie-Hellman Algorithm in Network Security

The **Diffie-Hellman algorithm** is a **key exchange protocol**. Its purpose is not to encrypt messages itself, but to allow two parties who have never met before (e.g., Alice and Bob) to securely establish a **shared secret key** over an insecure public channel. This shared secret can then be used as the key for a symmetric encryption algorithm like AES.

The security of the algorithm relies on the fact that it is computationally difficult to calculate discrete logarithms.

Here is the step-by-step process:

1. **Public Agreement:**
 - Alice and Bob publicly agree on two numbers:
 - A large prime number, p .
 - A primitive root modulo p , called g .
 - These two numbers, p and g , are not secret and can be known by an eavesdropper (Eve).
2. **Private Key Generation:**
 - Alice chooses a secret private integer, a . She keeps this secret.

- Bob chooses a secret private integer, b . He keeps this secret.
- 3. **Public Key Calculation and Exchange:**
 - Alice calculates her public key, A , using the formula: $A = g \text{ mod } p$. She sends A to Bob over the public channel.
 - Bob calculates his public key, B , using the formula: $B = g \text{ mod } p$. He sends B to Alice over the public channel.
 - Eve, the eavesdropper, can see p, g, A , and B , but cannot easily figure out the secret integers a or b .
- 4. **Shared Secret Calculation:**
 - Alice receives Bob's public key B and calculates the shared secret key, S :
 $S = B \text{ mod } p = (g^b) \text{ mod } p = g^{ab} \text{ mod } p$
 - Bob receives Alice's public key A and calculates the shared secret key, S :
 $S = A \text{ mod } p = (g^a) \text{ mod } p = g^{ab} \text{ mod } p$

Result: Both Alice and Bob have independently computed the **exact same shared secret key, S** , without ever transmitting it directly. They can now use this key S to encrypt their subsequent communications.

13. Explain the RSA Algorithm with Suitable Example

RSA (Rivest-Shamir-Adleman) is one of the first and most widely used **asymmetric (public-key) cryptography** algorithms. It is used for both secure data transmission (encryption) and digital signatures. The security of RSA relies on the practical difficulty of factoring the product of two large prime numbers.

The algorithm involves three main stages: Key Generation, Encryption, and Decryption.

1. Key Generation

1. Choose two large, distinct prime numbers, p and q .
2. Calculate the modulus, $n = p \times q$.
3. Calculate Euler's totient function: $\phi(n) = (p-1) \times (q-1)$.
4. Choose an integer e (the public exponent) such that $1 < e < \phi(n)$ and $\text{gcd}(e, \phi(n)) = 1$.
5. Calculate d (the private exponent) such that $d \times e \equiv 1 \pmod{\phi(n)}$. d is the modular multiplicative inverse of e .
6. The **Public Key** is the pair (e, n) .
7. The **Private Key** is the pair (d, n) .

2. Encryption

To encrypt a plaintext message M (represented as a number), the sender uses the recipient's public key (e, n) :

- $C = M \text{ mod } n$ The resulting ciphertext is C .

3. Decryption

To decrypt the ciphertext C , the recipient uses their own private key (d,n) :

- $M = C d \bmod n$ This recovers the original plaintext message M .
-

Example (using small numbers for simplicity)

1. Key Generation

- **Step 1:** Choose primes $p=7$ and $q=11$.
- **Step 2:** Calculate $n=p \times q = 7 \times 11 = 77$.
- **Step 3:** Calculate $\phi(n) = (p-1)(q-1) = (6)(10) = 60$.
- **Step 4:** Choose e such that $\gcd(e, 60) = 1$. Let's choose $e=13$.
- **Step 5:** Find d such that $13 \times d \equiv 1 \pmod{60}$.
 - Using the Extended Euclidean Algorithm, we find that $d=37$ (because $13 \times 37 = 481$, and $481 \bmod 60 = 1$).
- **Keys:**
 - **Public Key:** $(e,n) = (13,77)$.
 - **Private Key:** $(d,n) = (37,77)$.

2. Encryption

- Let's say the plaintext message is $M=5$.
- The sender uses the public key $(13,77)$ to encrypt it:
 - $C = 5^{13} \bmod 77$
 - $5^2 = 25$
 - $5^4 = 625 \equiv 9 \pmod{77}$
 - $5^8 \equiv 9^2 = 81 \equiv 4 \pmod{77}$
 - $5^{13} = 5^8 \times 5^4 \times 5^1 \equiv 4 \times 9 \times 5 \pmod{77}$
 - $180 \pmod{77} = 26$
- So, the **Ciphertext $C=26$** .

3. Decryption

- The recipient receives the ciphertext $C=26$ and uses their private key $(37,77)$ to decrypt it:
 - $M = 26^{37} \bmod 77$
- This calculation is long, but it correctly yields:
 - $M=5$
- The original message is successfully recovered.

Sources

1) Security Attacks with types

Soch le koi tere ghar me ghusne ki koshish kar raha hai. Yehi computer world me bhi hota hai.

- **Passive attack:** Chor bas jhankta hai, sunta hai (jaise network sniffing). Data badalta nahi.
 - **Active attack:** Chor ghus ke cheez todta hai, change karta hai (data modify, fake messages bhejna).
-

2) Objectives of Computer Security

Computer security ke 5 pillars hote hawa:

- **Integrity** – Data asli hi rahe, koi change na kare.
 - **Confidentiality** – Sirf allowed bande hi dekh sake.
 - **Non-repudiation** – Banda apna action deny na kar sake (jaise mail bhej ke bole “maine bheja hi nahi”).
 - **Access control** – Har kisi ko andar aane ka permission nahi milta.
 - **Availability** – System hamesha chalna chahiye, band na ho jaye.
-

3) Short note wale

Yeh upar ke 5 objectives hi chhote me likhne hote.

4) Passive attack types

- **Release of message contents** – Chor bas message padh raha.
 - **Traffic analysis** – Message ka pattern dekh raha (kaun kis se baat kar raha, kitni der tak).
-

5) Active attack types

- **Masquerade** – Koi dusre ban ke act kar raha (fake ID).
 - **Replay** – Purana valid message dubara bhejna.
 - **Modification** – Data me changes karna.
 - **Denial of Service (DoS)** – System down kar dena.
-

6) Symmetric Cipher Model

Isme ek **same key** hoti hai encryption aur decryption dono ke liye. Jaise tu aur mai ek hi taala aur chaabi use karen.

7) DES (Data Encryption Standard)

Purana encryption algorithm h jo 56-bit key use karta. Data ko block (64-bit) me tod ke encrypt karta.

8) Symmetric vs Asymmetric key cryptography

- **Symmetric** – Ek hi key dono ke liye (fast hai but key sharing problem hai).
 - **Asymmetric** – Do key hote: Public + Private. (Slow hai but secure hai).
-

9) Two Algorithm types

- **Symmetric algorithm** – Ek hi key.
 - **Asymmetric algorithm** – Do key (public-private).
-

10) Modes of operation in block ciphers

Block cipher ko alag style me use karte:

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

11) Variations in DES

DES ke kuch versions bane:

- Double DES
 - Triple DES (3DES) – secure zyada hai.
-

12) Diffie-Hellman Algorithm

Ye ek tareeka h secure key exchange ka. Dono log ek **common secret key** generate karte bina openly share kiye.

13) RSA Algorithm with example

Asymmetric encryption ka bada baap.

- Do key bani: Public key (sabko de do), Private key (apne paas rakho).
- Example:
 - Public key se data encrypt karo → sirf private key se hi decrypt hoga.
 - Private key se sign karo → sab log public key se verify karenge.