CBCS SCHEME

BCS7

USN 1 C R 2 2 C I 0 2 0

Seventh Semester B.E./B.Tech. Degree Examination, Dec.2025/Jan.2026

Cryptography and Network Security

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

			M	L	C
		Module – 1			
Q.1	a.	Obtain Ciphertext for the given plaintext "HILLCIPHER" by applying the Hill Cipher technique using key $K = \begin{bmatrix} 03 & 02 \\ 08 & 05 \end{bmatrix}$	7	L3	CO1
	b.	Write a short note on Steganography and its advantages and disadvantages.	6	L2	CO1
	c.	With a neat diagram, explain the model for network security.	7	L2	CO1
		OR			
Q.2	a.	State the rules used for encryption in PLAYFAIR cipher and encrypt the message "COMPUTER" using the keyword "ENGINEERING" using PLAYFAIR cipher.	7	L3	CO1
	b.	Describe simple XOR and one – time pad encryption techniques with an example and their difficulties.	7	L2	CO1
	c.	With a block diagram, explain the various steps involved in encryption and key generation of the DES algorithm.	6	L2	CO1
		Module – 2			
Q.3	a.	Demonstrate the Diffie – Hellman key exchange algorithm.	8	L2	CO2
	b.	Perform encryption and decryption using the RSA algorithm given public key is 6 for two prime numbers 17 and 31 with message 3.	7	L3	CO2
	c.	Describe the fundamental requirements that a public key cryptosystem must meet to ensure security.	5	L2	CO2
		OR			
Q.4	a.	Explain briefly the elliptic curve cryptography and mention two applications.	8	L2	CO2
	b.	Let q = 719 and g = 5 , Xa = 157 , Xb = 293. Use the Diffie Hellman Key exchange algorithm to find Ya , Yb and Secret key K.	7	L3	CO2
	c.	Briefly explain the security aspects of the RSA algorithm.	5	L2	CO2

		Module – 3			
5	a.	Explain the symmetric key distribution using Asymmetric Encryption.	7	L2	CO3
	b.	Explain the role of cryptographic hash functions in message authentication with a neat diagram.	8	L2	CO3
	c.	Discuss the general elements of an X.509 certificate.	5	L2	CO3
		OR			
Q.6	a.	What is Key Management? Explain with a neat diagram, how key usage can be controlled in encryption and decryption using control vectors.	7	L2	CO3
	b.	Describe the architecture of the Public Key Infrastructure X.509 (PKIX) model with a neat diagram.	8	L2	CO3
	c.	Write a short note on the various schemes of public key distribution.	5	L2	CO3
		Module – 4			
Q.7	a.	Explain functions and cryptographic algorithms used in S/MIME functionality.	8	L2	CO4
	b.	Define TLS and explain its architecture with a neat diagram.	7	L2	CO4
	c.	Bring out the differences between Kerberos version 4 and version 5.	5	L2	CO4
		OR			
Q.8	a.	Describe remote user authentication using asymmetric encryption.	8	L2	CO4
	b.	Explain Pretty Good Privacy (PGP) message transmission and reception with a neat diagram.	7	L2	CO4
	c.	Elaborate on the various security approaches that address web security threats.	5	L2	CO4
		Module – 5			
Q.9	a.	How does Domain Keys Identified Mail (DKIM) address the threats posed by email attackers and what is its strategy for email authentication?	8	L2	CO5
	b.	Explain Internet Key Exchange (IKE) key determination features.	7	L2	CO5
	c.	Explain Basic combinations of Security Associations.	5	L2	CO5
		OR			
Q.10	a.	Illustrate the key components of the Internet mail architecture with a clear diagram.	8	L2	CO5
	b.	Explain the Encapsulating IP Security Payload.	7	L2	CO5
	c.	Describe the functional flow of Domain Keys Identified Mail (DKIM).	5	L2	CO5

* * * * *

Q1 (a) Hill Cipher

Plaintext: HILLCIPHER
Key matrix (K):

$$K = \begin{bmatrix} 03 & 02 \\ 08 & 05 \end{bmatrix}$$

Step–1: Convert letters to numbers (A=0 … Z=25)

H=7, I=8, L=11, L=11, C=2, I=8, P=15, H=7, E=4, R=17

Group into pairs (2-letter blocks):
HI LL CI PH ER

So vectors are:
HI = [7,8]
LL = [11,11]
CI = [2,8]
PH = [15,7]
ER = [4,17]

Step–2: Encryption formula

$$C = K \cdot P (\text{mod} 26)$$

Block 1: HI

$$\begin{bmatrix} 03 & 02 \\ 08 & 05 \end{bmatrix} \begin{bmatrix} 7 \\ 8 \end{bmatrix} = \begin{bmatrix} 3(7) + 2(8) \\ 8(7) + 5(8) \end{bmatrix} = \begin{bmatrix} 37 \\ 96 \end{bmatrix} (\text{mod} 26) = \begin{bmatrix} 11 \\ 18 \end{bmatrix}$$

11 = L, 18 = S → **LS**

Block 2: LL

$$\begin{bmatrix} 03 & 02 \\ 08 & 05 \end{bmatrix} \begin{bmatrix} 11 \\ 11 \end{bmatrix} = \begin{bmatrix} 55 \\ 143 \end{bmatrix} (\text{mod} 26) = \begin{bmatrix} 3 \\ 13 \end{bmatrix}$$

3=D, 13=N → **DN**

Block 3: CI

$$= \begin{bmatrix} 22 \\ 56 \end{bmatrix} (\text{mod} 26) = \begin{bmatrix} 22 \\ 4 \end{bmatrix}$$

22=W, 4=E → **WE**

Block 4: PH

$$= \begin{bmatrix} 59 \\ 155 \end{bmatrix} (\mathrm{mod} 26) = \begin{bmatrix} 7 \\ 25 \end{bmatrix}$$

7=H, 25=Z → **HZ**

Block 5: ER

$$= \begin{bmatrix} 46 \\ 117 \end{bmatrix} (\mathrm{mod} 26) = \begin{bmatrix} 20 \\ 13 \end{bmatrix}$$

20=U, 13=N → **UN**

Final Ciphertext

LSDNWEHZUN

Q1 (b) Short note on Steganography (Advantages & Disadvantages)

Steganography

Steganography is the technique of **hiding secret data inside another normal-looking file** so that nobody suspects a message exists.

Example:

- Hide text inside **image pixels**
- Hide secret audio inside **sound files**
- Hide message inside **video frames**

Types

1. **Image steganography**
2. **Audio steganography**
3. **Video steganography**
4. **Text steganography**

Advantages

- Secret data is **not noticeable**
- Provides **extra privacy + secrecy**
- Useful for **copyright watermarking**

- More difficult to detect compared to encryption alone

Disadvantages

- If detected, the message is compromised
- Limited capacity (cannot hide huge data)
- Compression/resizing may destroy hidden data
- Needs special tools/algorithms to embed & extract

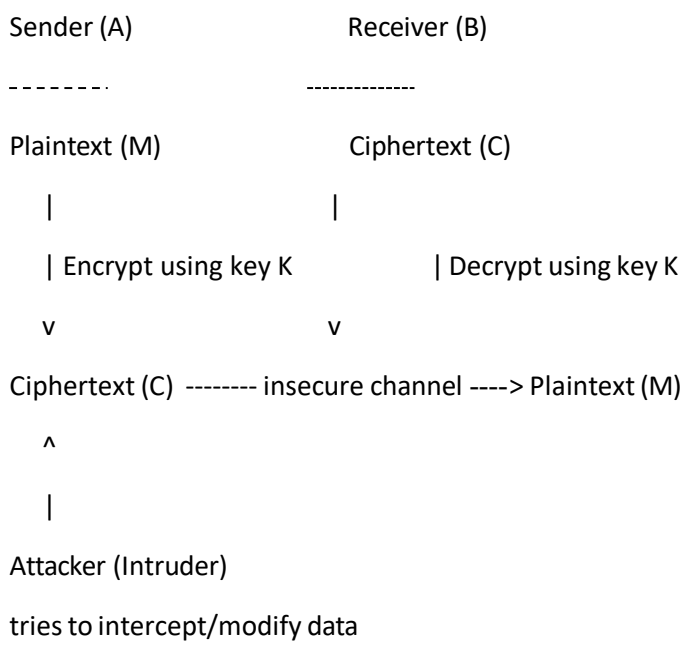
Q1 (c) Model for Network Security (with diagram)

Network Security Model

It ensures **secure communication** over insecure networks like the Internet by using:

- Encryption
- Authentication
- Integrity
- Key management

Neat diagram (standard model)



Main components

1. **Sender**
2. **Receiver**
3. **Encryption algorithm**
4. **Decryption algorithm**

5. **Key (K)**
6. **Insecure communication channel**
7. **Attacker**

Q2 (a) Playfair Cipher

Message: COMPUTER
Keyword: ENGINEERING

Step–1: Construct key matrix (I/J combined)

Keyword without repetition: **E N G I R A B C D F H K L M O P Q S T U V W X Y Z**

Playfair 5×5 matrix:

E N G I R

A B C D F

H K L M O

P Q S T U

V W X Y Z

Step–2: Make digraphs

COMPUTER → CO MP UT ER

Step–3: Encrypt each pair

- CO → **FL**
- MP → **HT**
- UT → **PU**
- ER → **NE**

Final Ciphertext

FLHTPUNE

Q2 (b) Simple XOR and One-Time Pad (OTP) with difficulties

S**imple XOR encryption**

XOR works like:

$$C = P \oplus K$$
$$P = C \oplus K$$

Where:

- P = plaintext bits
- K = key bits
- C = cipher bits

Example

P = 1010
K = 1100
C = P XOR K = 0110

One-**Time Pad (OTP)**

OTP is XOR encryption where:

- key is **truly random**
- key length = plaintext length
- key is used **only once**

$$C = P \oplus K$$

OTP is **perfectly secure** if rules are followed.

Difficulties / Problems

Simple XOR problems

- If key repeats, attacker can break it
- Not secure for long messages

OTP problems

- Key generation must be truly random
- Key length must be as long as message
- Secure key distribution is difficult
- Key storage becomes impractical for big communication

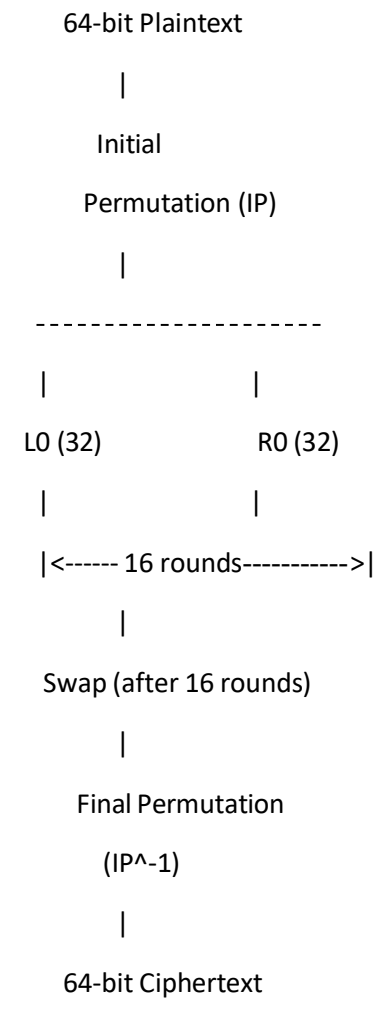
Q2 (c) DES Encryption + Key Generation Steps (Block diagram)

DES overview

- Symmetric block cipher

- Block size = **64 bits**
- Key size = **64 bits** (effective 56 bits, 8 parity bits)
- Uses **16 rounds**

Block diagram of DES steps



Swap (after 16 rounds)

Final Permutation

(IP^-1)

64-bit Ciphertext

DES Round Structure

For each round i = 1 to 16:

$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Function f includes:

1. Expansion (32→48)
2. XOR with round key Ki
3. S-Box substitution (48→32)

4. Permutation (P)

Key Generation (Key Schedule)

64-bit Key

|

Parity Drop (PC-1) → 56-bit key

|

Split into C0 and D0 (28-bit each)

|

Left Circular Shifts (1 or 2 bits each round)

|

Compression Permutation (PC-2)

|

Generates 16 round keys (48 bits each)

K1, K2, ... K16

Q3 (a) Demonstrate the Diffie–Hellman key exchange algorithm (8 Marks)

Goal

Diffie–Hellman (DH) allows two users (A and B) to **generate a common secret key over an insecure channel**, without sending the secret key directly.

Algorithm Steps

Step 1: Public parameters

Both users agree on:

- A large prime number **q**
- A primitive root **g** of q

These are **public** (known to attacker also).

Step 2: Private keys

- A chooses private key: **Xa**
- B chooses private key: **Xb**

These are **secret**.

Step 3: Generate public keys

- A computes:

$$Y_a = g^{Xa} \, mod \, q$$

- B computes:

$$Y_b = g^{Xb} \, mod \, q$$

A sends **Ya** to B
B sends **Yb** to A

Step 4: Generate shared secret key

- A computes:

$$K = (Y_b)^{Xa} \, mod \, q$$

- B computes:

$$K = (Y_a)^{Xb} \, mod \, q$$

Both get the **same key K**.

Proof of correctness

$$(Y_b)^{Xa} = (g^{Xb})^{Xa} = g^{XaXb}$$
$$(Y_a)^{Xb} = (g^{Xa})^{Xb} = g^{XaXb}$$

So both sides obtain identical key.

Security

An attacker knows **q, g, Ya, Yb**, but cannot compute K easily because it requires solving **Discrete Logarithm Problem**.

Q3 (b) Perform encryption and decryption using RSA

Given: primes **p=17**, **q=31**, public exponent **e=6**, message **M=3** (7 Marks)

Step 1: Compute n

$$n = p \times q = 17 \times 31 = 527$$

Step 2: Compute ϕ(n)

$$\phi(n) = (p-1)(q-1) = 16 \times 30 = 480$$

Step 3: Check RSA condition

RSA requires:

$$gcd(e, \phi(n)) = 1$$

But:

$$gcd(6{,}480) = 6 \neq 1$$

Therefore, e = 6 is NOT valid for RSA key generation (since modular inverse d will not exist).

Conclusion:

RSA encryption/decryption is NOT possible with $e = 6$ for $p = 17, q = 31$

Correct note to write in exam:

Since gcd(e, ϕ(n)) ≠ 1, the private key d does not exist. Hence RSA cannot be performed with given values.

Q3 (c) Fundamental requirements of a public-key cryptosystem (5 Marks)

A good public-key cryptosystem must satisfy:

1. **Easy key generation**
    - It must be easy to generate a public/private key pair.
2. **Easy encryption**
    - Sender should be able to encrypt using receiver's public key easily.
3. **Easy decryption**
    - Receiver should decrypt ciphertext using private key efficiently.
4. **Infeasible to find private key**

- It should be computationally impossible to derive private key from public key.

5. **Infeasible to recover plaintext without private key**
    - Even if attacker knows public key and ciphertext, finding plaintext should be infeasible.
6. **Keys are inverses**
    - Either key can be used for encryption/decryption (useful for digital signatures).

Q4 (a) Explain ECC and mention two applications (8 Marks)

Elliptic Curve Cryptography (ECC)

ECC is a public-key cryptography method based on mathematical properties of elliptic curves.

General elliptic curve equation:

$$y^2 = x^3 + ax + b$$

(mod p)

ECC works using **point operations** on the curve:

- Point addition
- Point doubling
- Scalar multiplication

Why ECC is used?

ECC provides **same security as RSA but with smaller key sizes**.

Example:

- **256-bit ECC ≈ 3072-bit RSA** (approx same strength)

So ECC is:

- faster
- uses less memory
- good for mobile/IoT devices

Applications (any two)

1. **SSL/TLS (HTTPS)**

- Used in secure web communication (ECDHE, ECDSA)

2. **Digital Signatures**
    - ECDSA is used in Bitcoin, blockchain and certificates

Other examples:

- Secure messaging apps
- Smart cards
- IoT device security

Q4 (b) Diffie–Hellman problem (7 Marks)

Given:

- $q = 719$
- $g = 5$
- $X_a = 157$
- $X_b = 293$

Find:

- Y_a, Y_b, and secret key K

Step 1: Compute Y_a

$$Y_a = g^{Xa} \bmod q = 5^{157} \bmod 719 = 28$$

Step 2: Compute Y_b

$$Y_b = g^{Xb} \bmod q = 5^{293} \bmod 719 = 279$$

Step 3: Compute Shared Secret key

A computes:

$$K = (Y_b)^{Xa} \bmod q = 279^{157} \bmod 719 = 618$$

B computes:

$$K = (Y_a)^{Xb} \bmod q = 28^{293} \bmod 719 = 618$$

Final Answers

$$\boxed{Y_a = 28}$$
$$\boxed{Y_b = 279}$$
$$\boxed{K = 618}$$

Q4 (c) Briefly explain security aspects of RSA algorithm (5 Marks)

RSA security depends on:

1. **Factoring difficulty**
    - RSA is secure because it is computationally very hard to factor:

$$n = pq$$

when p and q are very large.

2. **Key size**
    - Larger n gives more security
    - Common sizes: **2048-bit, 3072-bit**
3. **Private key protection**
    - If private key d is leaked → entire RSA breaks.
4. **Padding is necessary**
    - Plain RSA is vulnerable to attacks.
    - Use padding like:
        - **OAEP** (encryption)
        - **PSS** (signatures)
5. **Mathematical attacks**
    - Weak choices of p, q or e can reduce security.
    - p and q must be random and large.

Module – 3

Q5 (a) Explain the symmetric key distribution using Asymmetric Encryption (7 Marks)

Meaning

In this method, **public key (asymmetric) encryption is used only to securely send a symmetric session key**, and then the actual data communication happens using fast symmetric encryption.

Because:

- symmetric encryption is **fast**
- asymmetric encryption is **slow** but good for **secure key transfer**

Steps of symmetric key distribution (using public key cryptography)

Assume:

- A = Sender
- B = Receiver
- PU_B = B's public key
- PR_B = B's private key
- Ks = Session key (symmetric key)

Procedure

1. **B generates public/private key pair**
    - Public key PU_B is shared openly
    - Private key PR_B is kept secret
2. **A generates a random session key Ks**
3. **A encrypts Ks using B's public key**

$$C = E(PU_B, Ks)$$

4. **A sends encrypted session key C to B**
5. **B decrypts using private key**

$$Ks = D(PR_B, C)$$

6. Now both A and B have the same session key Ks.
7. **Actual message transmission using symmetric encryption**

$$Cipher = E(Ks, Message)$$

Neat diagram

A (Sender) B (Receiver)

---------- ----------

Choose session key Ks

|

Encrypt Ks with B's public key PU_B

C = E(PU_B, Ks)

|

| -------- send C --------->

Decrypt using PR_B

Ks = D(PR_B, C)

Now both share Ks (secret symmetric key)

Message Encryption:

Ciphertext = E(Ks, M) -----> Decrypt: M = D(Ks, Ciphertext)

Advantages

- Secure exchange of symmetric key
- Efficient for long communication (fast symmetric encryption)
- Used in SSL/TLS, HTTPS

Q5 (b) Explain the role of cryptographic hash functions in message authentication with neat diagram (8 Marks)

Message Authentication Means

Ensuring:

1. **Integrity** (message not changed)
2. **Authentication** (message is from correct sender)

Cryptographic hash function gives a fixed-size output:

$$H(M) = \text{message digest}$$

Role of hash functions

Hash functions help in message authentication by providing:

1. **Message integrity**
    - If message changes, digest changes
2. **Fast verification**
3. **Compact representation**
4. **Used with secret key (MAC) or digital signature**

Method 1: Hash + Secret Key = Message Authentication Code (MAC)

Sender generates:

$$MAC = H(K \parallel M)$$

where K = shared secret key

Sender sends:

$$(M, MAC)$$

Receiver computes again:

$$MAC' = H(K \parallel M)$$

If:

$$MAC = MAC'$$

Message is authentic and unmodified.

Neat diagram (MAC using Hash)

Sender (A) Receiver (B)

Message M Receive M

Shared secret key K Shared secret key K

MAC = H(K || M) MAC' = H(K || M)

| |

Send (M, MAC) ----------> Compare MAC and MAC'

If MAC = MAC' => Authentic + Integrity OK

Else => Message modified or fake

Properties of secure cryptographic hash function

1. **Preimage resistance** (one-way)
2. **Second preimage resistance**
3. **Collision resistance**
4. Fast computation

Examples: SHA-256, SHA-3

Q5 (c) Discuss the general elements of an X.509 certificate (5 Marks)

X.509 Certificate

It is a **digital certificate** issued by a CA which binds:

- identity of owner + public key

General elements / fields of X.509 certificate

1. **Version**
    - v1, v2, v3
2. **Serial Number**
    - unique number assigned by CA
3. **Signature Algorithm Identifier**

- e.g., SHA256withRSA

4. **Issuer Name**
   - CA details who issued certificate
5. **Validity Period**
   - Not Before, Not After dates
6. **Subject Name**
   - owner/user details
7. **Subject Public Key Information**
   - public key + algorithm info
8. **Extensions (v3)**
   - key usage, subject alternative name, etc.
9. **CA Digital Signature**
   - signature of CA on certificate

Certificate structure summary

$$Certificate = (Data) + Signature$$

Q6 (a) What is Key Management? Explain with diagram how key usage can be controlled using control vectors (7 Marks)

Key Management

Key management refers to:

generating, distributing, storing, updating, and destroying cryptographic keys securely.

Key management functions

1. Key generation
2. Key distribution
3. Key storage/protection
4. Key rotation / renewal
5. Backup & recovery

6. Key revocation / destruction

Control Vector concept

A **Control Vector (CV)** is extra information attached to a key which **restricts its usage**.

Example restrictions:

- key can be used only for encryption not decryption
- key can be used only for MAC
- key cannot be exported
- key valid only in some device/system

Working (Key controlled encryption/decryption)

Instead of directly storing key K,
we store:

$$K' = K \oplus CV$$

To use it, system combines CV again and retrieves correct operational key.

Neat diagram

Key K Control Vector CV

| |

| XOR |

v v

Stored Key = K ⊕ CV (protected form in database)

During usage:

Stored Key (K ⊕ CV) ⊕ CV = K

CV decides permitted usage:

- encryption only

- decryption only

- MAC only

- export not allowed

Advantages

- Prevents misuse of keys
- Controls key purpose
- Adds extra security in key storage

Q6 (b) Describe architecture of Public Key Infrastructure X.509 (PKIX) model with diagram (8 Marks)

PKIX

PKIX = Public Key Infrastructure (X.509)
Used to manage certificates, trust, and secure communication.

Main components

1) Certification Authority (CA)

- issues certificates
- signs certificates

2) Registration Authority (RA)

- verifies user identity before CA issues certificate

3) End Entity

- certificate owner/user (person/server/device)

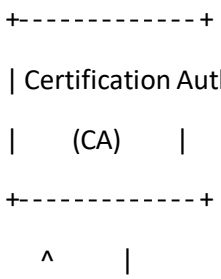
4) Repository / Directory

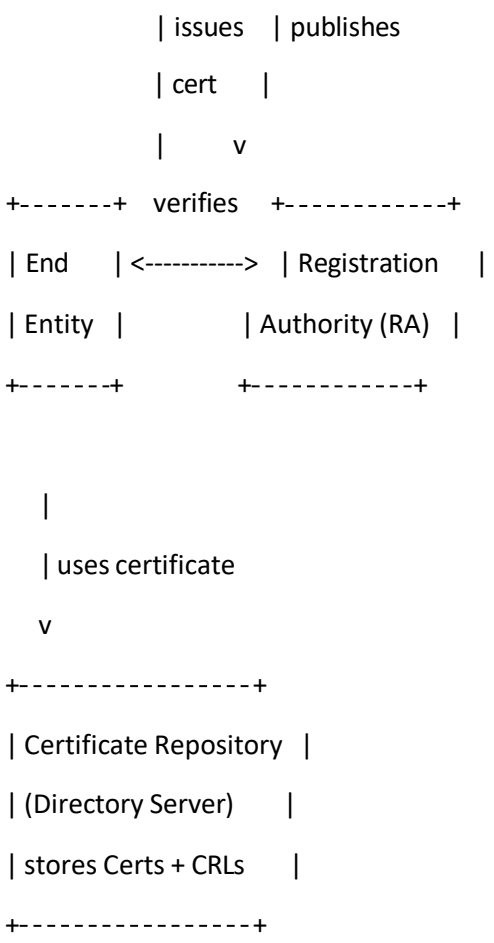
- stores certificates and CRLs publicly

5) Certificate Revocation List (CRL)

- list of revoked certificates

Neat PKIX Architecture diagram





Working steps

1. User requests certificate to RA
2. RA verifies identity
3. CA generates and signs certificate
4. Certificate stored in repository
5. Users validate using CA public key
6. Revoked certificates listed in CRL

Q6 (c) Short note on various schemes of public key distribution (5 Marks)

Public key distribution means:

how users securely obtain each other's public keys.

Schemes

1) Public Announcement

- user broadcasts public key openly
- drawback: attacker can send fake key

2) Publicly available directory

- trusted directory stores keys
- users access directory for keys
- directory must be protected

3) Public Key Authority (online trusted server)

- authority provides public key on request
- provides authentication and prevents fake keys
- requires online availability always

4) Public Key Certificates (X.509)

- CA issues certificate binding identity + public key
- widely used in Internet (SSL/TLS)
- prevents spoofing

Most secure and commonly used:

Public Key Certificates (X.509)

MODULE – 4

Q7 (a) Explain functions and cryptographic algorithms used in S/MIME functionality (8 Marks)

What is S/MIME?

S/MIME (Secure/Multipurpose Internet Mail Extensions) is a standard used to provide **security for e-mails** using **public key cryptography + digital certificates**.

It provides:

- Confidentiality
- Authentication
- Message integrity
- Non-repudiation

Functions of S/MIME

1) Enveloped Data (Confidentiality)

- S/MIME encrypts the e-mail content so that only receiver can read it.

Process:

- Generate session key (symmetric key)
- Encrypt message using symmetric algorithm
- Encrypt session key using receiver public key

---

2) Signed Data (Authentication + Integrity + Non-repudiation)

- Sender signs e-mail using digital signature.
- Receiver verifies signature using sender's public key.

---

3) Clear-signed Data

- Message is readable in plain text
- Signature is attached separately

---

4) Signed and Enveloped Data

- Both encryption and signing together (most secure)

Cryptographic Algorithms used in S/MIME

🔹 **(A) Digital Signature Algorithms**

Used to sign the message:

- **RSA**
- **DSA**
- **ECDSA**

🔹 **(B) Hash Algorithms**

Used to create message digest:

- **SHA-1** (older)
- **SHA-256 / SHA-384 / SHA-512** (preferred)

- **MD5** (not secure now)

🔹 (C) Symmetric Encryption Algorithms

Used to encrypt mail content:

- **AES (128/192/256)**
- **3DES**
- **DES** (not secure now)
- **RC2** (older)

🔹 (D) Key Encryption (Key transport)

Used to encrypt the session key:

- **RSA public key encryption**
- Diffie-Hellman (key agreement)

🔹 (E) Certificates

Uses **X.509 certificates**

- Issued by CA
- binds user identity with public key

Conclusion:
S/MIME combines **digital signature + encryption + certificates** for complete secure email.

Q7 (b) Define TLS and explain its architecture with neat diagram (7 Marks)

Definition of TLS

TLS (Transport Layer Security) is a protocol that provides security for data transfer over networks (Internet).
It ensures:

- Confidentiality (encryption)
- Integrity (hash/MAC)
- Authentication (certificates)

TLS is used in:

- HTTPS
- Secure email transfer
- VPNs

TLS Architecture

TLS has **two main layers**:

1) TLS Record Protocol (Lower layer)

Provides:

- Fragmentation
- Compression (optional)
- MAC (integrity)
- Encryption (confidentiality)

---

2) TLS Handshake Protocol (Upper layer)

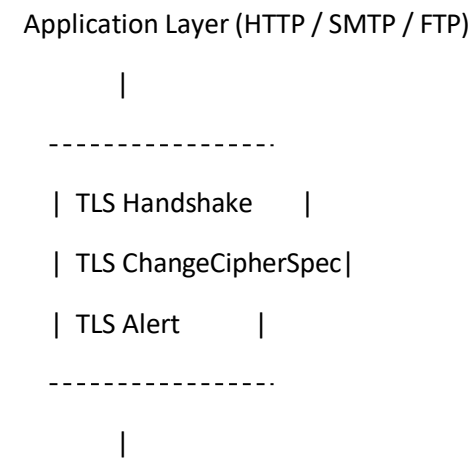
Used to:

- Authenticate server/client
- Negotiate cipher suites
- Generate session keys

Other upper protocols:

- Change Cipher Spec
- Alert protocol

Neat architecture diagram



TLS Record Protocol

(Fragment + MAC + Encrypt + Transmit)

|

TCP Transport Layer

|

IP Network

Working briefly

1. Handshake → chooses algorithms and keys
2. Authentication using certificates
3. Record protocol encrypts and sends data securely

Q7 (c) Differences between Kerberos version 4 and version 5 (5 Marks)

Feature	**Kerberos V4**	**Kerberos V5**
Encryption	DES only	Supports multiple algorithms (AES, 3DES, etc.)
Ticket lifetime	fixed format	flexible lifetime
Network addressing	only IPv4	supports multiple address types
Ticket format	limited	extensible + new fields
Inter-realm authentication	weak	improved & scalable
Message encoding	proprietary	ASN.1 encoding (standard)
Security	less strong	stronger, supports better encryption

■ Conclusion:

Kerberos V5 is more secure and flexible than V4

Q8 (a) Describe remote user authentication using asymmetric encryption (8 Marks)

Concept

Remote user authentication means verifying the identity of a user who is logging in over a network.

Using **asymmetric encryption**:

- user has private key
- server verifies using public key
- prevents password theft attacks

Steps (Public key authentication)

Assume:

- User = U
- Server = S
- PU_U = user public key
- PR_U = user private key

Step 1: Registration

User registers public key at server:

- Server stores PU_U

Step 2: Login process

1. User sends request: “I am U”
2. Server generates random number (nonce) R
3. Server sends R to user
4. User signs/encrypts R using PR_U:

$$C = E(PR_U, R)$$

5. User sends C back to server
6. Server decrypts using PU_U:

$$R = D(PU_U, C)$$

7. If decrypted R matches original → user authenticated

Neat diagram

User (U) Server (S)

------ -------

Login request ----------> send nonce R

<---------- R

Encrypt/sign nonce:

C = E(PRu, R)

Send C ----------> Decrypt using PUu

R' = D(PUu, C)

If R' = R => Authentication success

Advantages

- No password transmission
- Works even on insecure networks
- Strong authentication

Q8 (b) Explain PGP message transmission and reception with neat diagram (7 Marks)

What is PGP?

PGP (Pretty Good Privacy) is used for securing email and files.

It provides:

- Confidentiality
- Authentication
- Integrity
- Compression
- Email compatibility

PGP Transmission (Sender side)

1. Create message M
2. Create hash:

$$H(M)$$

3. Sign hash using sender private key:

$$Sig = E(PR_s, H(M))$$

4. Attach signature to message
5. Compress data
6. Generate session key Ks
7. Encrypt message with Ks (symmetric)
8. Encrypt session key with receiver public key:

$$E(PU_r, Ks)$$

9. Send encrypted session key + encrypted message

Reception (Receiver side)

1. Decrypt session key using receiver private key
2. Decrypt message using Ks
3. Decompress
4. Verify signature using sender public key

Neat diagram

Sender (PGP) Receiver (PGP)

---------- ----------

Message M

| hash: H(M)

| sign: Sig = E(PRs, H(M))

| attach Sig

| compress

| encrypt with session key Ks

|----------> decrypt Ks using PRr

Encrypt Ks using receiver public key PU_r decrypt msg using Ks

E(PU_r, Ks) decompress

verify:

H(M) == D(PUs, Sig)

Q8 (c) Elaborate on various security approaches that address web security threats (5 Marks)

Web security threats include:

- phishing
- SQL injection
- XSS
- session hijacking
- malware
- DoS attacks

Security approaches

1) Use HTTPS (TLS/SSL)

- encrypts client-server communication
- prevents MITM attacks

2) Authentication & Access control

- strong passwords, MFA
- role-based access control (RBAC)

3) Input Validation

- prevent SQL injection and XSS
- use parameterized queries

4) Firewalls & IDS/IPS

- firewall blocks malicious traffic
- IDS detects attacks, IPS prevents them

5) Secure cookies & session management

- HttpOnly, Secure cookies
- session timeout and regeneration

6) Regular patching and updates

- fixes known vulnerabilities

Final statement:

Combination of encryption, authentication, secure coding, and monitoring provides web security

---

MODULE – 5

Q9 (a) How does DKIM address threats posed by email attackers and what is its strategy for email authentication? (8 Marks)

What is DKIM?

DKIM (DomainKeys Identified Mail) is an email authentication mechanism that allows the receiver to verify that:

1. the email really came from the claimed domain, and
2. the message was not altered during transmission.

It uses **digital signatures + DNS public key publishing**.

Threats posed by email attackers

Attackers generally do:

1. **Email spoofing**
    - forging "From" address to appear as legitimate domain
2. **Phishing**
    - fake mails to steal passwords/OTP/cards
3. **Message modification**
    - altering mail content in transit
4. **Spam / fake domain impersonation**
5. **Man-in-the-middle mail tampering**

How DKIM addresses these threats

DKIM adds **cryptographic protection**:

1) Protects against spoofing (domain impersonation)

- Only the real domain has access to **private key**
- Attacker can't generate valid DKIM signature

2) Protects against content modification

- DKIM signature is computed over message headers and body.
- If body is changed → signature verification fails.

3) Strengthens sender authentication

- Confirms that mail was authorized by the sending domain.

4) Supports anti-spam policies (with DMARC)

- DKIM results help DMARC decide:
    - accept / quarantine / reject

DKIM Strategy for Email Authentication

DKIM follows these steps:

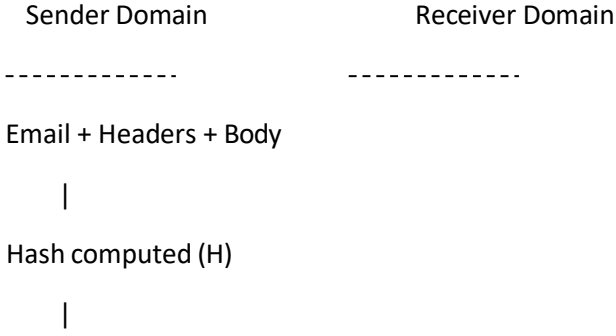
(A) Sender side

1. Sender's mail server generates a **hash** of email contents
2. Encrypts hash using **domain private key**
3. Adds **DKIM-Signature header** to email
4. Sends email normally

(B) Receiver side

1. Receiver extracts DKIM signature header
2. Finds public key using **DNS record** (TXT record)
3. Recomputes hash of received message
4. Decrypts signature using public key
5. Compares hashes
    - If matched → email is authentic & unchanged

Neat DKIM Diagram



Sign hash using private key

Sig = E(PRdomain, H)

|

Add "DKIM-Signature" header

|

Email sent over Internet ----------> Receiver extracts DKIM header

|

Get public key from DNS (TXT record)

|

Verify: H' == D(PUdomain, Sig)

|

Match => Authentic & unchanged

Final conclusion:

DKIM prevents spoofing and message tampering using signature verification with DNS public keys.

Q9 (b) Explain Internet Key Exchange (IKE) key determination features (7 Marks)

What is IKE?

IKE (Internet Key Exchange) is a protocol used in **IPsec** to:

- authenticate peers
- negotiate Security Associations (SA)
- generate & manage session keys securely

Key determination features in IKE

1) Automatic key generation

- IKE automatically generates shared secret keys
- avoids manual key distribution

---

2) Diffie–Hellman (DH) key exchange

- Used to create shared secret over insecure network

$$K = g^{ab} \, mod \, p$$

- Provides secure key agreement.

---

3) Perfect Forward Secrecy (PFS)

- Even if long-term key is compromised later,
  old session keys remain secure.
- Achieved by using new DH exchange for each session.

---

4) Key freshness

- IKE ensures each session uses **new keys**
- prevents replay attacks

---

5) Protection against replay

- Uses **nonces** and sequence numbers
- prevents attacker from replaying old messages

---

6) Efficient rekeying

- IKE supports periodic key renewal (rekeying)
- keys are refreshed automatically after time/traffic limit.

---

7) Secure authentication for keying

IKE supports:

- pre-shared keys
- digital signatures
- public key certificates

So only legitimate users can establish keys.

Summary line:

IKE securely determines fresh session keys using DH, supports PFS, replay protection and automatic rekeying.

Q9 (c) Explain basic combinations of Security Associations (SA) (5 Marks)

Security Association (SA)

SA is a **one-way logical connection** used in IPsec which defines:

- protocol (AH or ESP)
- encryption algorithm
- key
- lifetime
- mode (transport/tunnel)

SA Combinations

SAs can be combined for more security.

1) Transport Adjacency

- multiple security protocols applied in **transport mode**
- Example: AH + ESP together for host-to-host

IP Header | AH | ESP | TCP/UDP | Data

---

2) Iterated Tunnel

- more than one tunnel applied (nested tunnels)
- used for VPN through multiple gateways

Outer IP | ESP | Inner IP | ESP | TCP | Data

---

3) Transport inside Tunnel

- tunnel mode + transport mode combination
- common in VPN setups

Example: ESP tunnel between gateways + AH transport between hosts.

Conclusion:

SAs can be combined as Transport Adjacency, Iterated Tunnel, and Transport inside Tunnel for stronger security.

Q10 (a) Illustrate key components of Internet mail architecture with clear diagram (8 Marks)

Internet mail architecture

It is the system that supports sending/receiving emails using:

- MUA
- MTA
- MDA
- Mailboxes
- SMTP/POP3/IMAP

Key Components

1) MUA (Mail User Agent)

- email client used by user
- ex: Gmail UI, Outlook, Thunderbird

2) MSA (Mail Submission Agent)

- accepts outgoing mail from MUA

3) MTA (Mail Transfer Agent)

- transfers mail between servers
- uses SMTP

4) MDA (Mail Delivery Agent)

- delivers mail into recipient mailbox

5) Mailbox / Message store

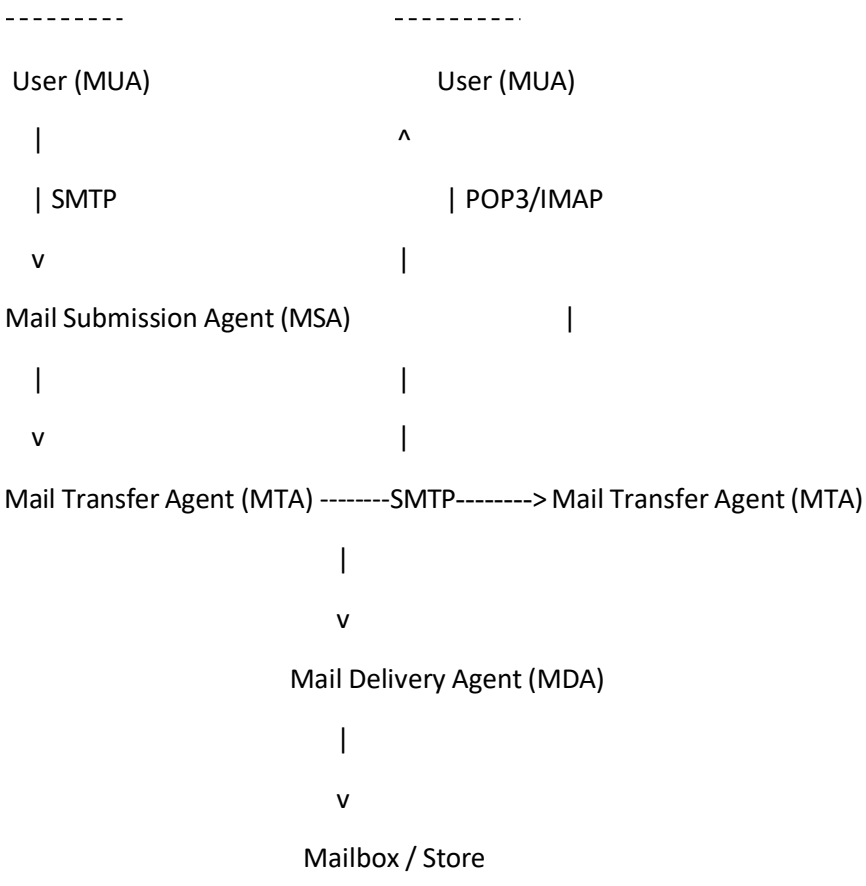
- where mails are stored

6) Protocols

- SMTP: sending mail
- POP3/IMAP: receiving mail

Neat diagram (Internet Mail)

Sender Side Receiver Side



Explanation:

- Mail is created in MUA
- sent using SMTP through MTAs
- stored in mailbox
- receiver reads using IMAP/POP3

Q10 (b) Explain the Encapsulating Security Payload (ESP) (7 Marks)

What is ESP?

ESP (Encapsulating Security Payload) is an IPsec protocol that provides:

1. **Confidentiality** (encryption)
2. **Integrity** (optional)
3. **Authentication** (optional)
4. **Anti-replay protection**

ESP works in:

- transport mode
- tunnel mode

ESP Packet Format

IP Header | ESP Header | Encrypted Payload | ESP Trailer | ESP Auth

ESP Header

- **SPI** (Security Parameter Index)
- **Sequence Number** (anti replay)

Encrypted Payload

- TCP/UDP + data (transport mode)
- entire IP packet (tunnel mode)

ESP Trailer

- padding
- pad length
- next header

ESP Auth (optional)

- integrity check value (ICV)

Neat ESP diagram with fields

```
+--------------+
| IP Header    |
+--------------+
| ESP Header   | --> SPI, Seq No.
+--------------+
| Payload Data | --> encrypted
+--------------+
| Padding + Pad Len |
| Next Header  |
+--------------+
| Authentication (ICV)| --> optional
+--------------+
```

Modes:

Transport mode:

IP | ESP | TCP/UDP | Data

Tunnel mode:

Outer IP | ESP | Inner IP | TCP/UDP | Data

Q10 (c) Describe functional flow of DKIM (5 Marks)

DKIM Functional Flow

1) Email creation

- Sender writes email (headers + body)

2) Canonicalization

- standard formatting applied on headers/body

3) Hash generation

$$H = Hash(message)$$

4) Signature creation

- Hash encrypted using domain private key:

$$Sig = E(PRdomain, H)$$

5) DKIM header added

- DKIM-Signature header includes:
    - signing domain
    - selector
    - hash algo
    - signature value

6) Receiver verification

- receiver gets public key from DNS
- verifies signature and integrity

DKIM flow diagram

Sender -> Hash -> Sign -> Add DKIM header -> Send

Receiver -> Fetch public key from DNS -> Verify -> Pass/Fail
