20CB603/20CS603/20DS603/20IT603

Hall Ticket Number:

III/IV B. Tech (Regular/Supplementary) DEGREE EXAMINATION

May, 2025 — **Common to CB, CS, DS, & IT**

Sixth Semester — **Cryptography & Network Security**

Time: Three Hours — **Maximum:** 70 Marks

Answer question 1 compulsorily. **(14X1 = 14 Marks)**

Answer one question from each unit. **(4X14 = 56 Marks)**

			CO	BL	M
1	a)	Define the three security goals.	CO1	L1	1M
	b)	Define a symmetric-key cipher	CO	L1	1M
	c)	What is the block size in DES? What is the cipher key size in DES? What is the round-key size in DES?	CO	L1	1M
	d)	List five modes of operation in modern block ciphers.	CO	L1	1M
	e)	Define a state in AES.	CO	L1	1M
	f)	Distinguish between symmetric-key and asymmetric-key cryptosystems.	CO	L2	1M
	g)	Define the first criterion for a cryptographic hash function.	CO	L1	1M
	h)	Define a cryptographic hash function.	CO	L1	1M
	i)	List the security services provided by a digital signature.	CO	L1	1M
	j)	List the duties of a KDC.	CO	L1	1M
	k)	What is an Authentication Server (AS).	CO	L2	1M
	l)	List services provided by SSL or TLS.	CO	L1	1M
	m)	Distinguish between two modes of IPSec.	CO	L2	1M
	n)	Difference between transport mode and tunnel mode.	CO	L1	1M
		Unit-I			
2	a)	Distinguish between passive and active security attacks.	CO1	L2	4M
	b)	Explain security services and mechanisms in detail.	CO1	L3	10M
		(OR)			
3	a)	With an example, distinguish between a substitution cipher and a transposition cipher.	CO1	L3	7M
	b)	Explain Electronic Codebook (ECB) Mode in detail.	CO1	L2	7M
		Unit-II			
4		Explain Advanced Encryption Standard (AES) in detail.	CO2	L3	14M
		(OR)			
5	a)	In RSA Given p = 19, q = 23, and e = 3, find n, φ(n), and d.	CO2	L3	7M
	b)	Explain elgamal crypto system with an example.	CO2	L3	7M
		Unit-III			
6	a)	Explain X.509 certification.	CO3	L2	7M
	b)	Explain Diffie Hellman key exchange algorithm with an example.	CO3	L3	7M
		(OR)			
7		Explain Kerberos in detail.	CO3	L2	14M
		Unit-IV			
8		Explain Secure Socket Layer (SSL) protocols.	CO4	L3	14M
		(OR)			
9	a)	Define Security Association (SA) and explain its purpose.	CO4	L3	7M
	b)	List ISAKMP payload types and the purpose of each type.	CO4	L2	7M

			CO	BL	M
1	a)	Define the three security goals. Confidentiality, Integrity, Availability	CO1	L1	1M
	b)	Define a symmetric-key cipher A symmetric-key cipher is an encryption algorithm in which the same key is used for both encryption and decryption.	CO	L1	1M
	c)	What is the block size in DES? What is the cipher key size in DES? What is the round-key size in DES? Block size in DES is 64 bits, Cipher key size in DES is 64 bits (but only 56 bits are used for encryption; 8 bits are for parity), Round-key size in DES is 48 bits.	CO	L1	1M
	d)	List five modes of operation in modern block ciphers. Five common modes of operation in modern block ciphers are: Electronic Codebook (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), Counter (CTR).	CO	L1	1M
	e)	Define a state in AES. A state in AES is a 4×4 matrix of bytes (16 bytes total) that holds the data being encrypted or decrypted.	CO	L1	1M
	f)	Distinguish between symmetric-key and asymmetric-key cryptosystems. Symmetric-key cryptosystem uses the same key for both encryption and decryption. Asymmetric-key cryptosystem uses two keys: a public key for encryption and a private key for decryption.	CO	L2	1M
	g)	Define the first criterion for a cryptographic hash function. It should be pre-image resistant — given a hash value h, it should be computationally infeasible to find any message m such that hash(m) = h.	CO	L1	1M
	h)	Define a cryptographic hash function. A cryptographic hash function is a mathematical algorithm that takes an input (or message) and produces a fixed-size hash value or digest, which uniquely represents the input data.	CO	L1	1M
	i)	List the security services provided by a digital signature. Authentication, Integrity, Non-repudiation	CO	L1	1M
	j)	List the duties of a KDC. Generate and distribute secret keys, Authenticate users, Establish secure communication between users, Maintain a database of user credentials, Prevent unauthorized access to keys.	CO	L1	1M
	k)	What is an Authentication Server (AS). An Authentication Server (AS) is a trusted entity in a network that verifies the identity of users or devices before granting access to network resources or services.	CO	L2	1M
	l)	List services provided by SSL or TLS. Authentication, Confidentiality, Data integrity, Secure session management	CO	L1	1M
	m)	Distinguish between two modes of IPSec. Transport Mode encrypts only the payload of the IP packet, leaving the header intact. Tunnel Mode encrypts the entire IP packet, and adds a new IP header.	CO	L2	1M
	n)	Difference between transport mode and tunnel mode. Transport Mode encrypts only the payload of the IP packet, leaving the header intact. Tunnel Mode encrypts the entire IP packet, and adds a new IP header.	CO	L1	1M

Unit-I

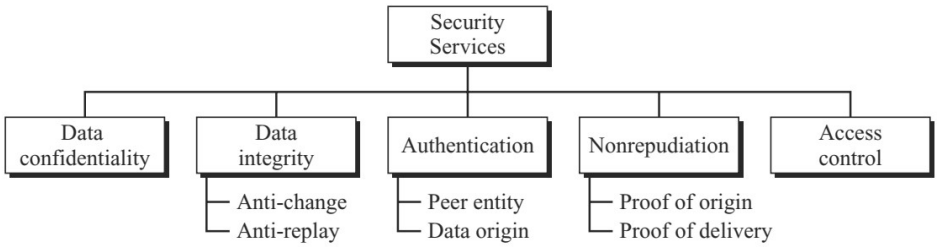
2 a) Distinguish between passive and active security attacks. CO1 L2 4M

Feature	Passive Attack	Active Attack
Definition	Attempts to observe or monitor communication	Attempts to alter or disrupt communication
Goal	Gain unauthorized access to information	Cause modification, deletion, or insertion of data
Impact	Typically does not affect system operations	Can damage system integrity and operations
Detection	Hard to detect, since it does not alter data	Easier to detect due to changes in data or behavior
Examples	Eavesdropping, Traffic analysis	Message modification, Replay attacks, Denial of Service (DoS)
Countermeasures	Use of encryption and secure protocols	Use of authentication, integrity checks, and intrusion detection systems

CO1 L3 10M

b) Explain security services and mechanisms in detail.

Security Services:



Data Confidentiality: Data confidentiality is designed to protect data from disclosure attack. It is designed to prevent snooping and traffic analysis attack.

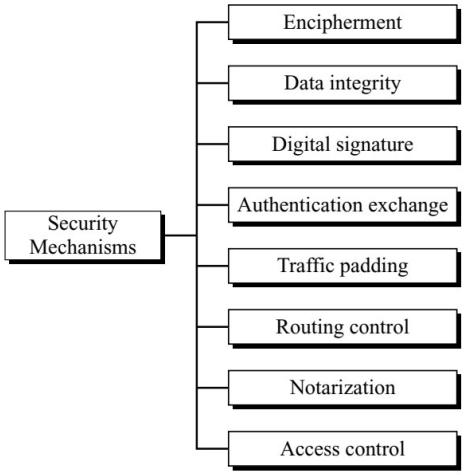
Data Integrity: Data integrity is designed to protect data from modification, insertion, deletion, and replaying by an adversary. It may protect the whole message or part of the message.

Authentication: This service provides the authentication of the party at the other end of the line. In connection-oriented communication, it provides authentication of the sender or receiver during the connection establishment (peer entity authentication). In connectionless communication, it authenticates the source of the data (data origin authentication).

Nonrepudiation: Nonrepudiation service protects against repudiation by either the sender or the receiver of the data. In nonrepudiation with proof of the origin, the receiver of the data can later prove the identity of the sender if denied. In nonrepudiation with proof of delivery, the sender of data can later prove that data were delivered to the intended recipient.

Access Control: Access control provides protection against unauthorized access to data. The term access can involve reading, writing, modifying, executing programs, and so on.

Security Mechanisms:



Encipherment: Encipherment, hiding or covering data, can provide confidentiality. Today two techniques cryptography and steganography are used for enciphering.

Data Integrity: The data integrity mechanism appends to the data a short checkvalue that has been created by a specific process from the data itself. The receiver receives the data and the checkvalue. He creates a new checkvalue from the received data and compares the newly created checkvalue with the one received. If the two checkvalues are the same, the integrity of data has been preserved.

Digital Signature: A digital signature is a means by which the sender can electronically sign the data and the receiver can electronically verify the signature.

Authentication Exchange: In authentication exchange, two entities exchange some messages to prove their identity to each other.

Traffic Padding: Traffic padding means inserting some bogus data into the data traffic to thwart the adversary's attempt to use the traffic analysis.

Routing Control: Routing control means selecting and continuously changing different available routes between the sender and the receiver to prevent the opponent from eavesdropping on a particular route.

Notarization: Notarization means selecting a third trusted party to control the communication between two entities. This can be done, for example, to prevent repudiation. The receiver can involve a trusted party to store the sender request in order to prevent the sender from later denying that she has made such a request.

Access Control: Access control uses methods to prove that a user has access right to the data or resources owned by a system. Examples of proofs are passwords and PINs.

(OR)

3 a) With an example, distinguish between a substitution cipher and a transposition cipher. CO1 L3 7M

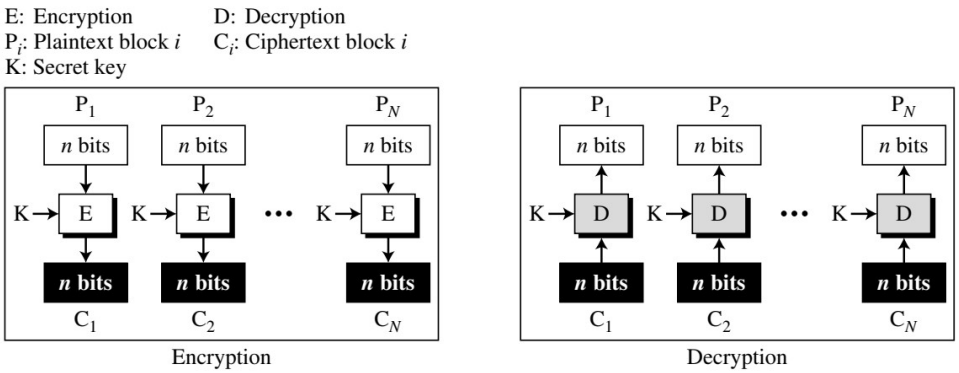
Feature	Substitution Cipher	Transposition Cipher
Definition	Replaces each character with another character	Rearranges the positions of characters
Operation	Substitutes letters or symbols	Does not alter characters, only their order
Security Basis	Based on character substitution	Based on character position permutation
Example	Plaintext: HELLO → Ciphertext: KHOOR (Caesar cipher, shift by 3)	Plaintext: HELLO → Ciphertext: OLLEH (simple reversal)

CO1 L2 7M

b) Explain Electronic Codebook (ECB) Mode in detail.

Electronic Codebook (ECB) Mode

The simplest mode of operation is called the electronic codebook (ECB) mode. The plaintext is divided into N blocks. The block size is n bits. If the plaintext size is not a multiple of the block size, the text is padded to make the last block the same size as the other blocks. The same key is used to encrypt and decrypt each block.



The relation between plaintext and ciphertext block is shown below:

Encryption: $C_i = E_K (P_i)$ Decryption: $P_i = D_K (C_i)$

Advantages:

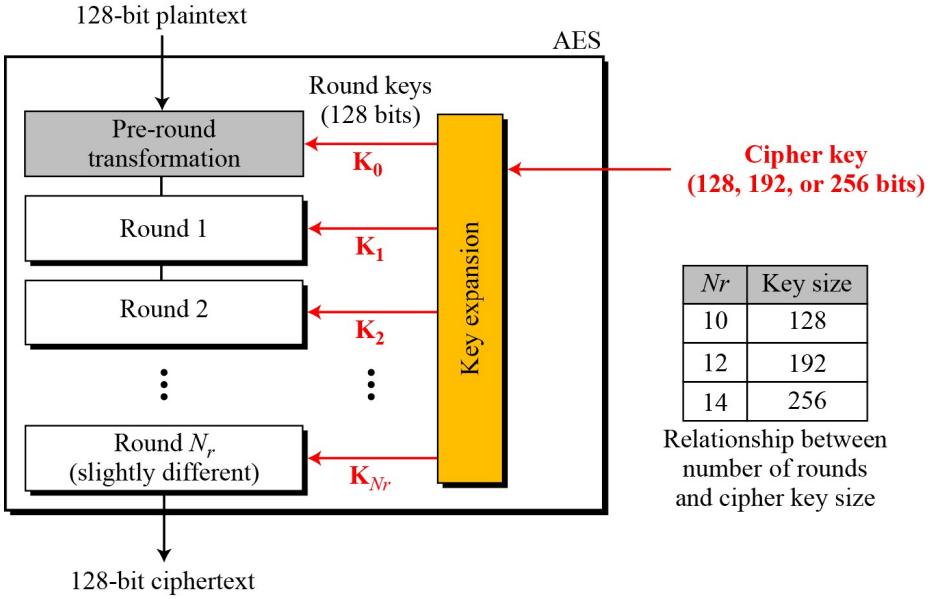
- Parallel encryption of blocks of bits is possible, thus it is a faster way of encryption.
- Simple way of the block cipher.

Disadvantages:

Prone to cryptanalysis since there is a direct relationship between plaintext and ciphertext.

4 Explain Advanced Encryption Standard (AES) in detail. CO2 L3 14M

AES is a non-Feistel cipher that encrypts and decrypts a data block of 128 bits. It uses 10, 12, or 14 rounds. The key size, which can be 128, 192, or 256 bits, depends on the number of rounds. We can have three different AES versions; they are referred as AES-128, AES-192, and AES-256. However, the round keys, which are created by the key-expansion algorithm are always 128 bits, the same size as the plaintext or ciphertext block.

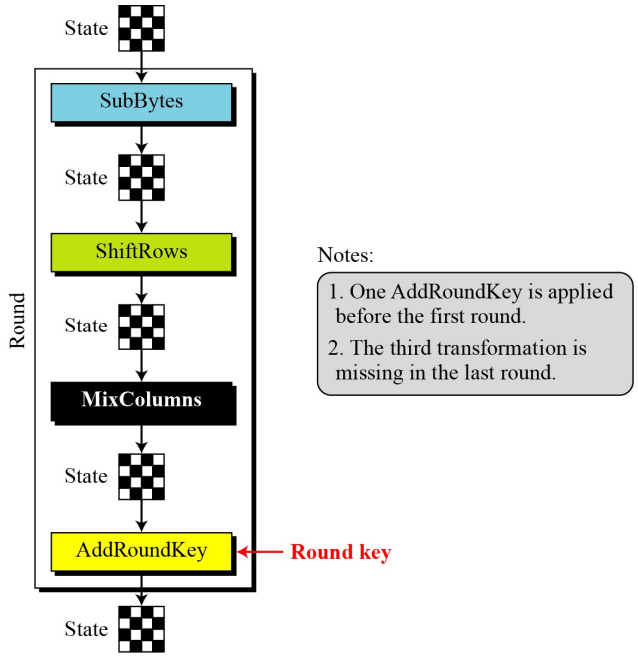


Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Structure of Each Round

Each round, except the last, uses four transformations that are invertible. The last round has only three transformations. Each transformation takes a state and creates another state to be used for the next transformation or the next round. The pre-round section uses only one transformation (AddRoundKey); the last round uses only three transformations (MixColumns transformation is missing).



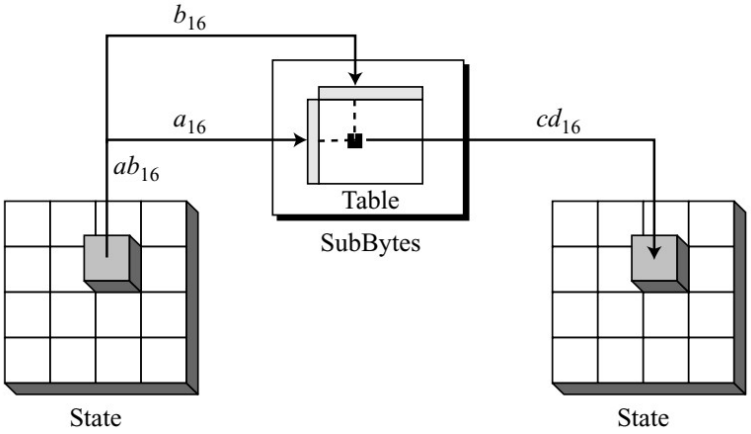
Transformations

Substitution:

AES, like DES, uses substitution. However, the mechanism is different. First, the substitution is done for each byte. Second, only one table is used for transformation of every byte, which means that if two bytes are the same, the transformation is also the same.

SubBytes

The first transformation, SubBytes, is used at the encryption site. To substitute a byte, we interpret the byte as two hexadecimal digits. The left digit defines the row and the right digit defines the column of the substitution table. The two hexadecimal digits at the junction of the row and the column are the new byte.



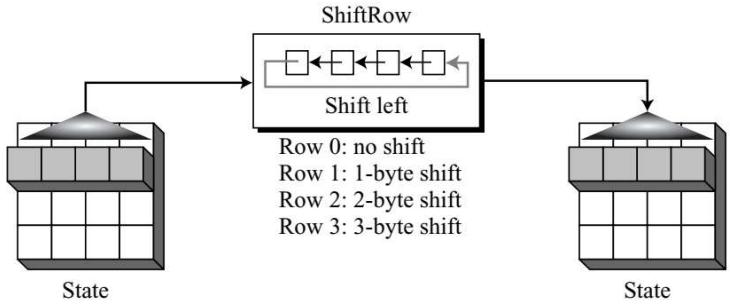
In the SubBytes transformation, the state is treated as a 4 × 4 matrix of bytes. Transformation is done one byte at a time. The contents of each byte is changed, but the arrangement of the bytes in the matrix remains the same. In the process, each byte is transformed independently. There are sixteen distinct byte-to-byte transformations.

Permutation:

Another transformation found in a round is shifting, which permutes the bytes. Unlike DES, in which permutation is done at the bit level, shifting transformation in AES is done at the byte level; the order of the bits in the byte is not changed.

ShiftRows

In the encryption, the transformation is called ShiftRows and the shifting is to the left.The number of shifts depends on the row number (0, 1, 2, or 3) of the state matrix. This means the row 0 is not shifted at all and the last row is shifted three bytes.



Note that the ShiftRows transformation operates one row at a time.

InvShiftRows

In the decryption, the transformation is called InvShiftRows and the shifting is to the right. The number of shifts is the same as the row number (0, 1, 2, and 3) of the state matrix.

Mixing:

The substitution provided by the SubBytes transformation changes the value of the byte based only on original value and an entry in the table; the process does not include the neighboring bytes. We can say that SubBytes is an intrabyte transformation. The permutation provided by the ShiftRows transformation exchanges bytes without permuting the bits inside the bytes. We can say that ShiftRows is a byte-exchange

transformation. We also need an interbyte transformation that changes the bits inside a byte, based on the bits inside the neighboring bytes. We need to mix bytes to provide diffusion at the bit level.

The mixing transformation changes the contents of each byte by taking four bytes at a time and combining them to recreate four new bytes. To guarantee that each new byte is different (even if all four bytes are the same), the combination process first multiplies each byte with a different constant and then mixes them. The mixing can be provided by matrix multiplication.

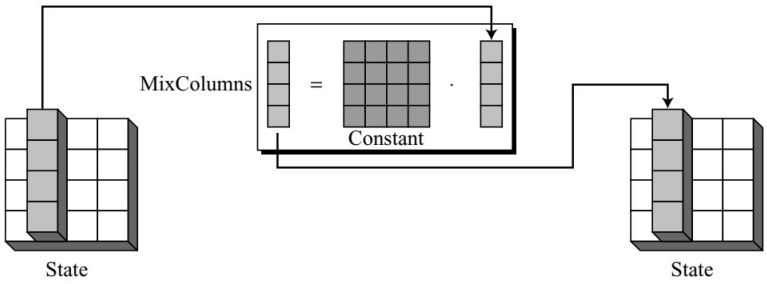
$$\begin{bmatrix} ax+by+cz+dt \\ ex+fy+gz+ht \\ ix+jy+kz+lt \\ mx+ny+oz+pt \end{bmatrix} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix}$$

New matrix Constant matrix Old matrix

AES defines a transformation, called MixColumns, to achieve this goal. There is also an inverse transformation, called InvMixColumns.

MixColumns

The MixColumns transformation operates at the column level; it transforms each column of the state to a new column. The transformation is actually the matrix multiplication of a state column by a constant square matrix.



Key Adding:

Probably the most important transformation is the one that includes the cipher key. All previous transformations use known algorithms that are invertible. If the cipher key is not added to the state at each round, it is very easy for the adversary to find the plaintext, given the ciphertext. The cipher key is the only secret between Alice and Bob in this case.

AES uses a process called key expansion that creates N_r +1 round keys from the cipher key. Each round key is 128 bits long it is treated as four 32-bit words. For the purpose of adding the key to the state, each word is considered as a column matrix.

AddRoundKey

AddRoundKey also proceeds one column at a time. It is similar to MixColumns in this respect. MixColumns multiplies a constant square matrix by each state column; AddRoundKey adds a round key word with each state column matrix. The operation in MixColumns is matrix multiplication; the operation in AddRoundKey is matrix addition. Since addition and subtraction in this field are the same, the AddRoundKey transformation is the inverse of itself.

(OR)

5 a) In RSA Given p = 19, q = 23, and e = 3, find n, φ(n), and d. CO2 L3 7M

Step 1: Compute n

$n = p \times q = 19 \times 23 = 437$

Step 2: Compute $\phi(n)$

$\phi(n) = (p - 1)(q - 1) = (19 - 1)(23 - 1) = 18 \times 22 = 396$

Step 3: Compute d

d is the modular inverse of e modulo $\phi(n)$, i.e.,

$d \equiv e^{-1} \bmod \phi(n)$

$d \equiv 3^{-1} \bmod 396$

We need to find d using Extended Euclidean Algorithm:

Given:

- e=3
- $\phi(n) = 396$

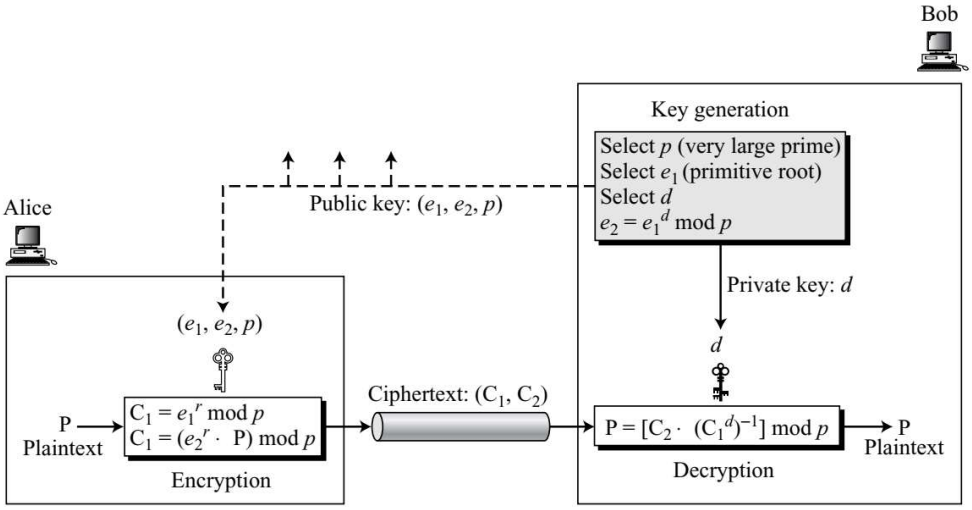
So:

- gcd(396,3) = 3

But this tells us gcd(3,396) = 3, not 1.
That means 3 and 396 are not coprime, and no modular inverse exists.
We see that gcd(3, 396) = 3 ≠ 1, so 3 is not valid as a public exponent.

b) Explain elgamal crypto system with an example. CO2 L3 7M

if p is a very large prime, e1 is a primitive root in the group G = <Z p*, × > and r is an integer, then e2 = e1r mod p is easy to compute using the fast exponential algorithm (square-and-multiply method), but given e2, e1, and p, it is infeasible to calculate r = $\log_{e1} e2 \bmod p$ (discrete logarithm problem).



Key Generation:

ElGamal_Key_Generation

{

Select a large prime p

Select d to be a member of the group $\mathbf{G} = < \mathbf{Z}_p^*, \times >$ such that $1 \le d \le p - 2$

Select e_1 to be a primitive root in the group $\mathbf{G} = < \mathbf{Z}_p^*, \times >$

$e_2 \leftarrow e_1^d \bmod p$

Public_key $\leftarrow (e_1, e_2, p)$ // To be announced publicly

Private_key $\leftarrow d$ // To be kept secret

return Public_key and Private_key

}

Encryption:

Anyone can send a message to Bob using his public key.

ElGamal_Encryption (e_1, e_2, p, P) // P is the plaintext

{

Select a random integer r in the group $\mathbf{G} = < \mathbf{Z}_p^*, \times >$

$C_1 \leftarrow e_1^r \bmod p$

$C_2 \leftarrow (P \times e_2^r) \bmod p$ // C_1 and C_2 are the ciphertexts

return C_1 and C_2

}

Decryption:

ElGamal_Decryption (d, p, C_1, C_2) // C_1 and C_2 are the ciphertexts

{

$P \leftarrow [C_2 (C_1^d)^{-1}] \bmod p$ // P is the plaintext

return P

}

Note: any relevant example is acceptable.

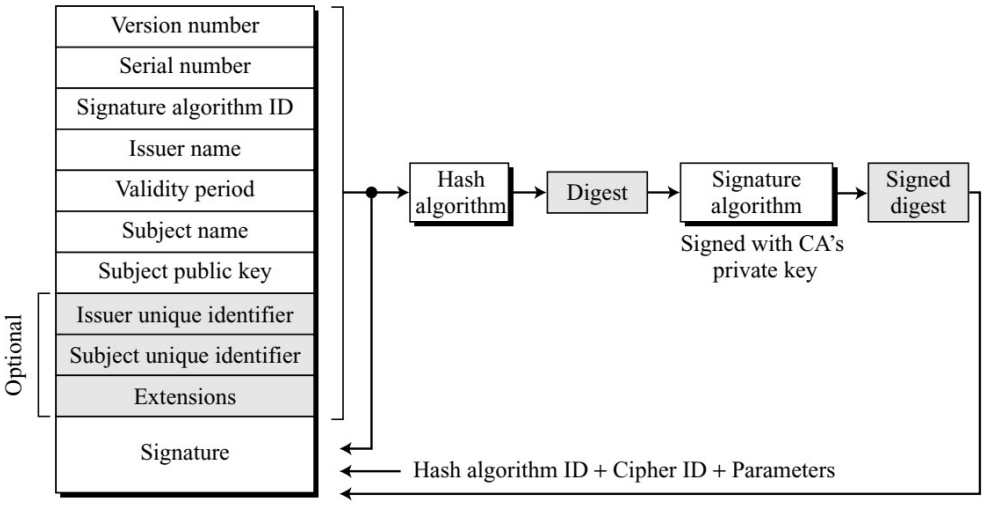
Unit-III

6 a) Explain X.509 certification. CO3 L2 7M

the use of a CA has solved the problem of public-key fraud, it has created a side-effect. Each certificate may have a different format. If Alice wants to use a program to automatically download different certificates and digests belonging to different people, the program may not be able to do this. One certificate may have the public key in one format and another in a different format. The public key may be on the first line in one certificate, and on the third line in another. Anything that needs to be used universally must have a universal format.

To remove this side effect, the ITU has designed X.509, a recommendation that has been accepted by the Internet with some changes. X.509 is a way to describe the certificate in a structured way.

Certificate:



A certificate has the following fields:

❑ Version number. This field defines the version of X.509 of the certificate. The version number started at 0; the current version (third version) is 2.

❑ Serial number. This field defines a number assigned to each certificate. The value is unique for each certificate issuer.

❑ Signature algorithm ID. This field identifies the algorithm used to sign the certificate. Any parameter that is needed for the signature is also defined in this field.

❑ Issuer name. This field identifies the certification authority that issued the

certificate. The name is normally a hierarchy of strings that defines a country, a state, organization, department, and so on.

❑ Validity Period. This field defines the earliest time (not before) and the latest time (not after) the certificate is valid.

❑ Subject name. This field defines the entity to which the public key belongs. It is also a hierarchy of strings. Part of the field defines what is called the common name, which is the actual name of the beholder of the key.

❑ Subject public key. This field defines the owner's public key, the heart of the certificate. The field also defines the corresponding public-key algorithm (RSA, for example) and its parameters.

❑ Issuer unique identifier. This optional field allows two issuers to have the same issuer field value, if the issuer unique identifiers are different.

❑ Subject unique identifier. This optional field allows two different subjects to have the same subject field value, if the subject unique identifiers are different.

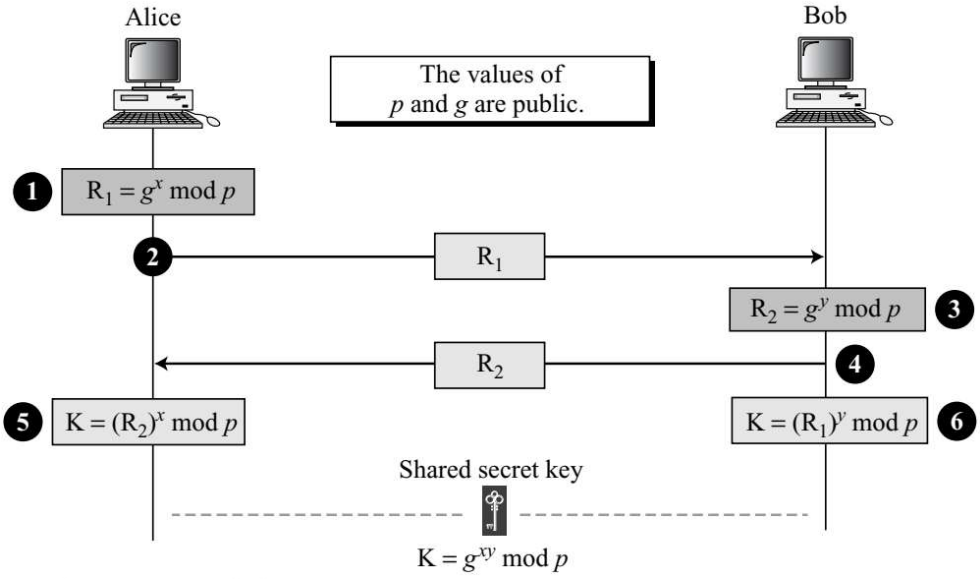
❑ Extensions. This optional field allows issuers to add more private information to the certificate.

❑ Signature. This field is made of three sections. The first section contains all other fields in the certificate. The second section contains the digest of the first section encrypted with the CA's public key. The third section contains the algorithm identifier used to create the second section.

Certificate Renewal: Each certificate has a period of validity. If there is no problem with the certificate, the CA issues a new certificate before the old one expires. The process is like the renewal of credit cards by a credit card company; the credit card holder normally receives a renewed credit card before the one expires.

b) Explain Diffie Hellman key exchange algorithm with an example. CO3 L3 7M

In the Diffie-Hellman protocol two parties create a symmetric session key without the need of a KDC. Before establishing a symmetric key, the two parties need to choose two numbers p and g. The first number, p, is a large prime number on the order of 300 decimal digits (1024 bits). The second number, g, is a generator of order p - 1 in the group <Zp*, ×>. These two (group and generator) do not need to be confidential. They can be sent through the Internet; they can be public.



The steps are as follows:

1. Alice chooses a large random number x such that $0 \le x \le p - 1$ and calculates $R_1 = g^x \bmod p$.
2. Bob chooses another large random number y such that $0 \le y \le p - 1$ and calculates $R_2 = g^y \bmod p$.
3. Alice sends R_1 to Bob. Note that Alice does not send the value of x; she sends only R_1.

4. Bob sends R_2 to Alice. Again, note that Bob does not send the value of *y*, he sends only R_2.
5. Alice calculates $K = (R_2)^x \bmod p$.
6. Bob also calculates $K = (R_1)^y \bmod p$.

K is the symmetric key for the session.

$K = (g^x \bmod p)^y \bmod p = (g^y \bmod p)^x \bmod p = g^{xy} \bmod p$

Bob has calculated $K = (R_1)^y \bmod p = (g^x \bmod p)^y \bmod p = g^{xy} \bmod p$. Alice has calculated $K = (R_2)^x \bmod p = (g^y \bmod p)^x \bmod = g^{xy} \bmod p$. Both have reached the same value without Bob knowing the value of x and without Alice knowing the value of y.

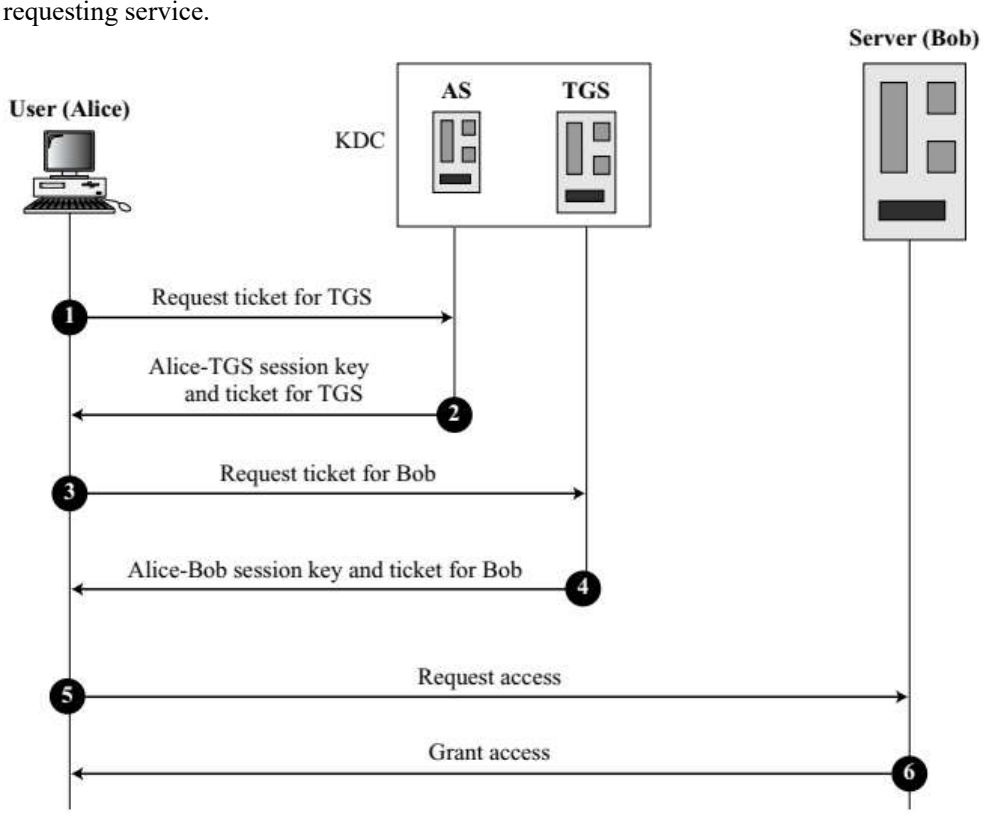
Note: any relevant example is acceptable.

(OR)

7 Explain Kerberos in detail. CO3 L2 14M

Kerberos is an authentication protocol, and at the same time a KDC, that has become very popular. Several systems, including Windows 2000, use Kerberos. It is named after the three-headed dog in Greek mythology that guards the gates of Hades. Originally designed at MIT, it has gone through several versions.

Servers: Three servers are involved in the Kerberos protocol: an authentication server (AS), a ticket-granting server (TGS), and a real (data) server that provides services to others. In our examples and figures, Bob is the real server and Alice is the user requesting service.



Authentication Server (AS)

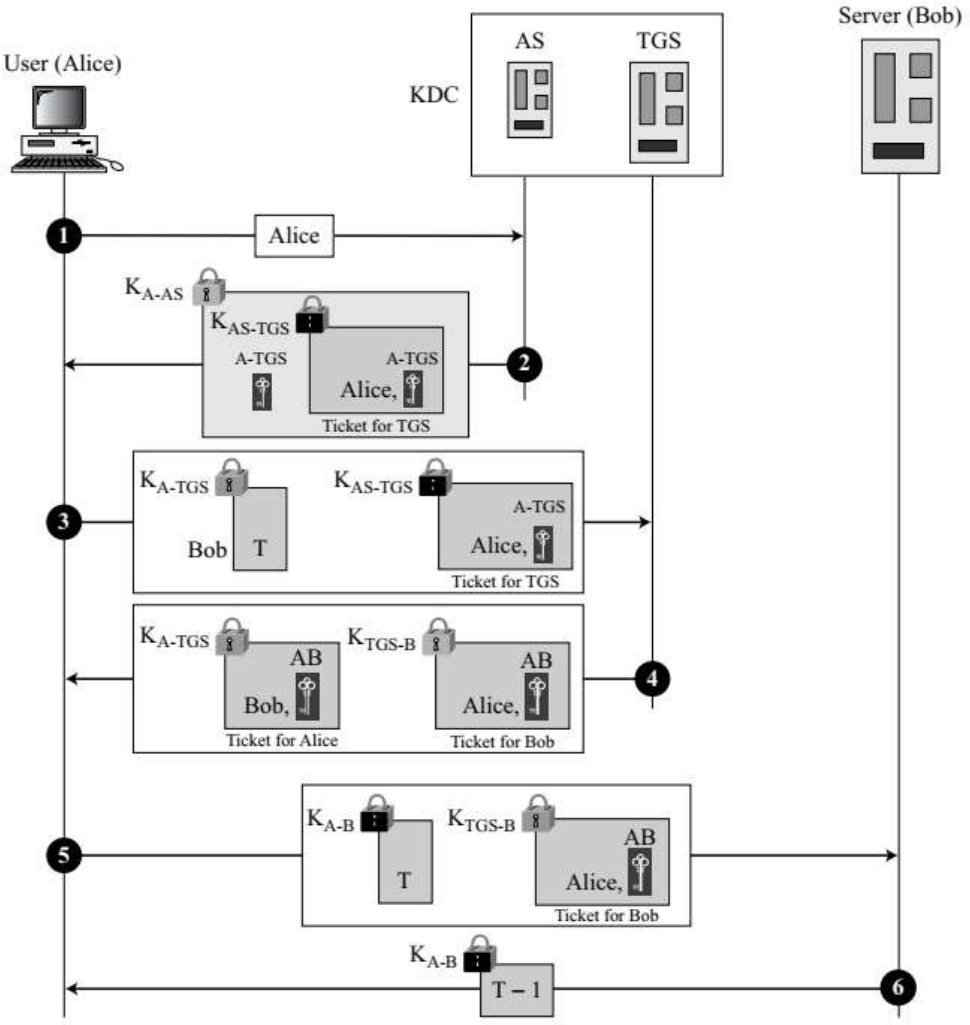
The authentication server (AS) is the KDC in the Kerberos protocol. Each user registers with the AS and is granted a user identity and a password. The AS has a database with these identities and the corresponding passwords. The AS verifies the user, issues a session key to be used between Alice and the TGS, and sends a ticket for the TGS.

Ticket-Granting Server (TGS)

The ticket-granting server (TGS) issues a ticket for the real server (Bob). It also provides the session key (KAB) between Alice and Bob. Kerberos has separated user verification from the issuing of tickets. In this way, though Alice verifies her ID just once with the AS, she can contact the TGS multiple times to obtain tickets for different real servers.

Real Server

The real server (Bob) provides services for the user (Alice). Kerberos is designed for a client-server program, such as FTP, in which a user uses the client process to access the server process. Kerberos is not used for person-to-person authentication.



Operation:

A client process (Alice) can access a process running on the real server (Bob) in six steps, as shown in Figure 15.8.

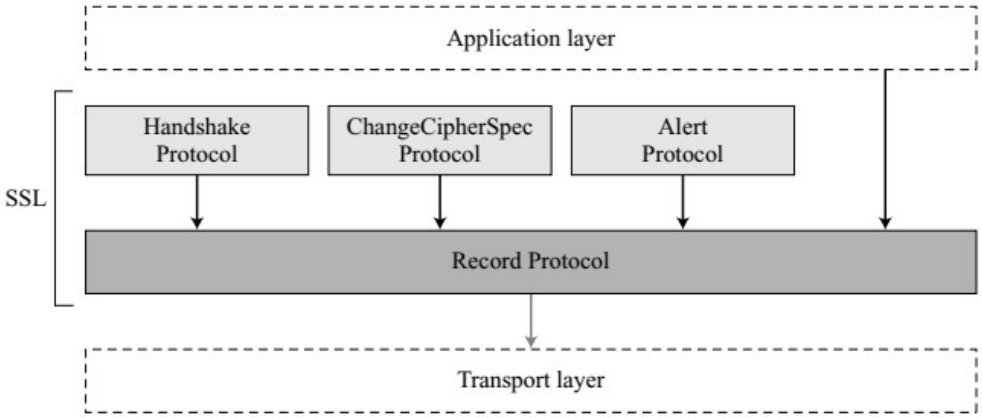
1. Alice sends her request to the AS in plain text using her registered identity.
2. The AS sends a message encrypted with Alice's permanent symmetric key, KA-AS. The message contains two items: a session key, KA-TGS, that is used by Alice to contact the TGS, and a ticket for the TGS that is encrypted with the TGS symmetric key, KAS-TGS. Alice does not know KA-AS, but when the message arrives, she types her symmetric password. The password and the appropriate algorithm together create KA-AS if the password is correct. The password is then immediately destroyed; it is not sent to the network and it does not stay in the terminal. It is used only for a moment to create KA-AS. The

process now uses KA-AS to decrypt the message sent. KA-TGS and the ticket are extracted.
3. Alice now sends three items to the TGS. The first is the ticket received from the AS. The second is the name of the real server (Bob), the third is a timestamp that is encrypted by KA-TGS. The timestamp prevents a replay by Eve.
4. Now, the TGS sends two tickets, each containing the session key between Alice and Bob, KA-B. The ticket for Alice is encrypted with KA-TGS; the ticket for Bob is encrypted with Bob's key, KTGS-B. Note that Eve cannot extract KAB because Eve does not know KA-TGS or KTGS-B. She cannot replay step 3 because she cannot replace the timestamp with a new one (she does not know KA-TGS). Even if she is very quick and sends the step 3 message before the timestamp has expired, she still receives the same two tickets that she cannot decipher.
5. Alice sends Bob's ticket with the timestamp encrypted by KA-B.
6. Bob confirms the receipt by adding 1 to the timestamp. The message is encrypted with KA-B and sent to Alice.

Unit-IV

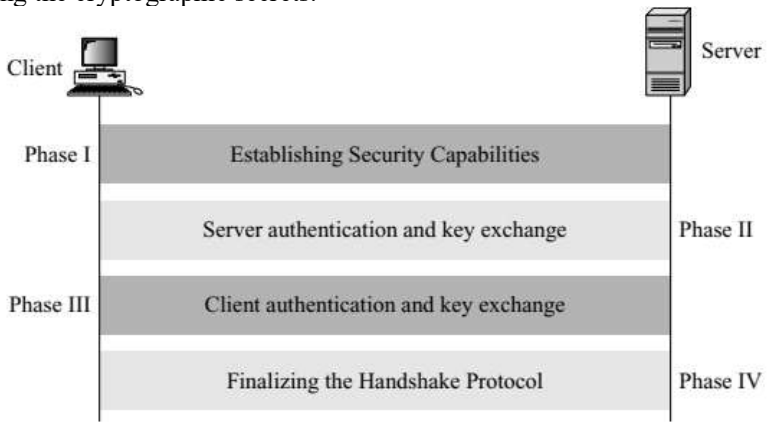
8 Explain Secure Socket Layer (SSL) protocols. CO4 L3 14M

SSL defines four protocols in two layers. The Record Protocol is the carrier. It carries messages from three other protocols as well as the data coming from the application layer. Messages from the Record Protocol are payloads to the transport layer, normally TCP. The Handshake Protocol provides security parameters for the Record Protocol. It establishes a cipher set and provides keys and security parameters. It also authenticates the server to the client and the client to the server if needed. The ChangeCipherSpec Protocol is used for signalling the readiness of cryptographic secrets. The Alert Protocol is used to report abnormal conditions.



Handshake Protocol

The Handshake Protocol uses messages to negotiate the cipher suite, to authenticate the server to the client and the client to the server if needed, and to exchange information for building the cryptographic secrets.

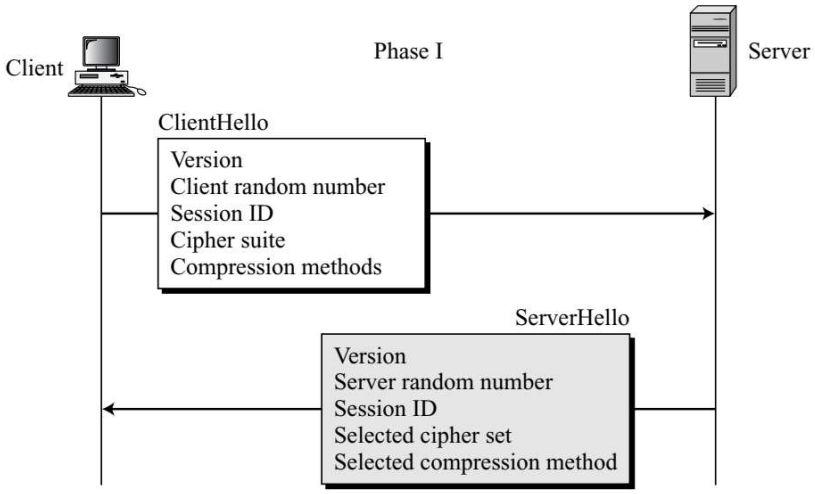


Phase I: Establishing Security Capability

In Phase I, the client and the server announce their security capabilities and choose tho that are convenient for both. In this phase, a session ID is established and the cipher su is chosen. The parties agree upon a particular compression method. Finally, two randc numbers are selected, one by the client and one by the server, to be used for creating master secret as we saw before. Two messages are exchanged in this phase: ClientHel and ServerHello messages.

ClientHello The client sends the ClientHello message. It contains the following:

a. The highest SSL version number the client can support.
b. A 32-byte random number (from the client) that will be used for master secı generation.
c. A session ID that defines the session.
d. A cipher suite that defines the list of algorithms that the client can support.
e. A list of compression methods that the client can support.

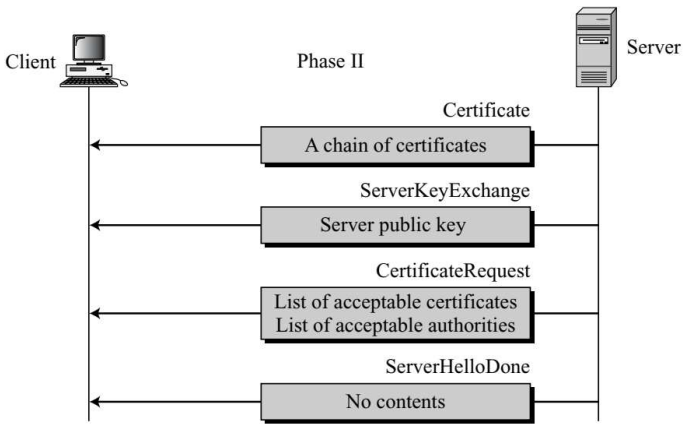


ServerHello The server responds to the client with a ServerHello message. It cc tains the following:

a. An SSL version number. This number is the lower of two version numbers: t highest supported by the client and the highest supported by the server.
b. A 32-byte random number (from the server) that will be used for master sec generation.
c. A session ID that defines the session.
d. The selected cipher set from the client list.
e. The selected compression method from the client list.

Phase II: Server Key Exchange and Authentication

In phase II, the server authenticates itself if needed. The sender may send its certifica its public key, and may also request certificates from the client. At the end, the serv announces that the serverHello process is done.



Certificate If it is required, the server sends a Certificate message to authentica itself. The message includes a list of certificates of type X.509. The certificate is r needed if the key-exchange algorithm is anonymous Diffie-Hellman.

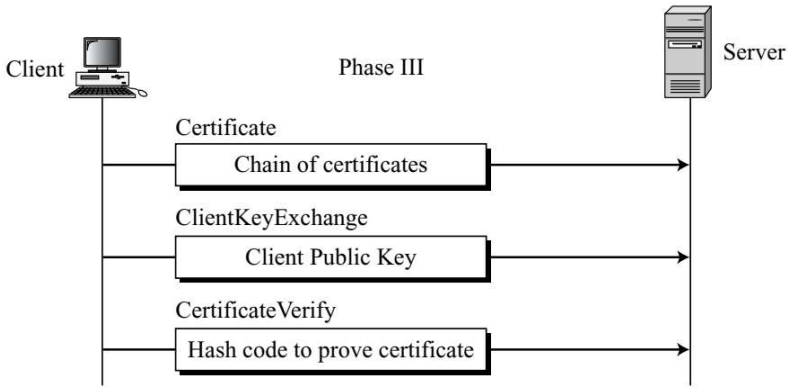
ServerKeyExchange After the Certificate message, the server sends a ServerKe Exchange message that includes its contribution to the pre-master secret. This messa is not required if the key-exchange method is RSA or fixed Diffie-Hellman.

CertificateRequest The server may require the client to authenticate itself. In tl case, the server sends a CertificateRequest message in Phase II that asks for certific tion in Phase III from the client. The server cannot request a certificate from the client it is using anonymous Diffie-Hellman.

ServerHelloDone The last message in Phase II is the ServerHelloDone messag which is a signal to the client that Phase II is over and that the client needs to st Phase III.

Phase III: Client Key Exchange and Authentication

Phase III is designed to authenticate the client. Up to three messages can be sent fr the client to the server, as shown in Figure 17.17.



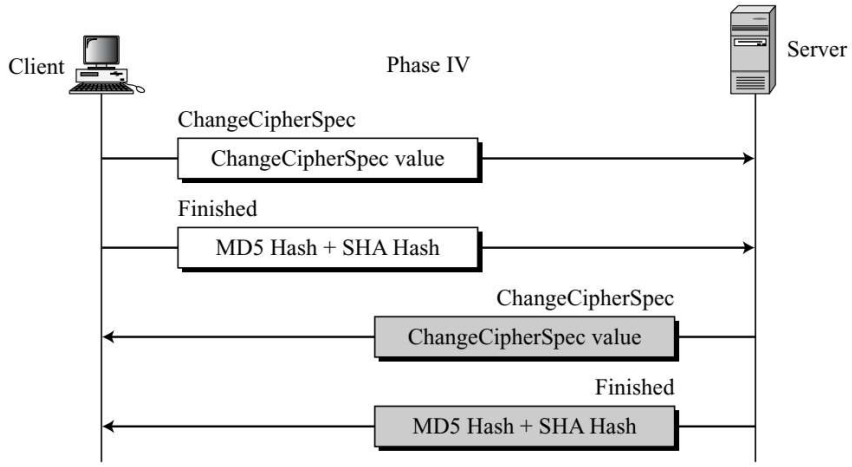
Certificate To certify itself to the server, the client sends a Certificate message. N that the format is the same as the Certificate message sent by the server in Phase II, b the contents are different. It includes the chain of certificates that certify the client.T message is sent only if the server has requested a certificate in Phase II. If there i request and the client has no certificate to send, it sends an Alert message (part of t Alert Protocol to be discussed later) with a warning that there is no certificate. T server may continue with the session or may decide to abort.

ClientKeyExchange After sending the Certificate message, the client sends a Clie KeyExchange message, which includes its contribution to the pre-master secret. T contents of this message are based on the key-exchange algorithm used. If the meth is RSA, the client creates the entire pre-master secret and encrypts it with the RS public key of the server. If the method is anonymous or ephemeral Diffie-Hellman, t client sends its Diffie-Hellman half-key. If the method is Fortezza, the client sends t Fortezza parameters. The contents of this message are empty if the method is fix Diffie-Hellman.

CertificateVerify If the client has sent a certificate declaring that it owns the pub key in the certificate, it needs to prove that it knows the corresponding private key. T is needed to thwart an impostor who sends the certificate and claims that it comes fr the client. The proof of private-key possession is done by creating a message and sig ing it with the private key. The server can verify the message with the public k already sent to ensure that the certificate actually belongs to the client. Note that this possible if the certificate has a signing capability; a pair of keys, public and private, involved. The certificate for fixed Diffie-Hellman cannot be verified this way.

Phase IV: Finalizing and Finishing

In Phase IV, the client and server send messages to change cipher specification and finish the handshaking protocol. Four messages are exchanged in this phase.



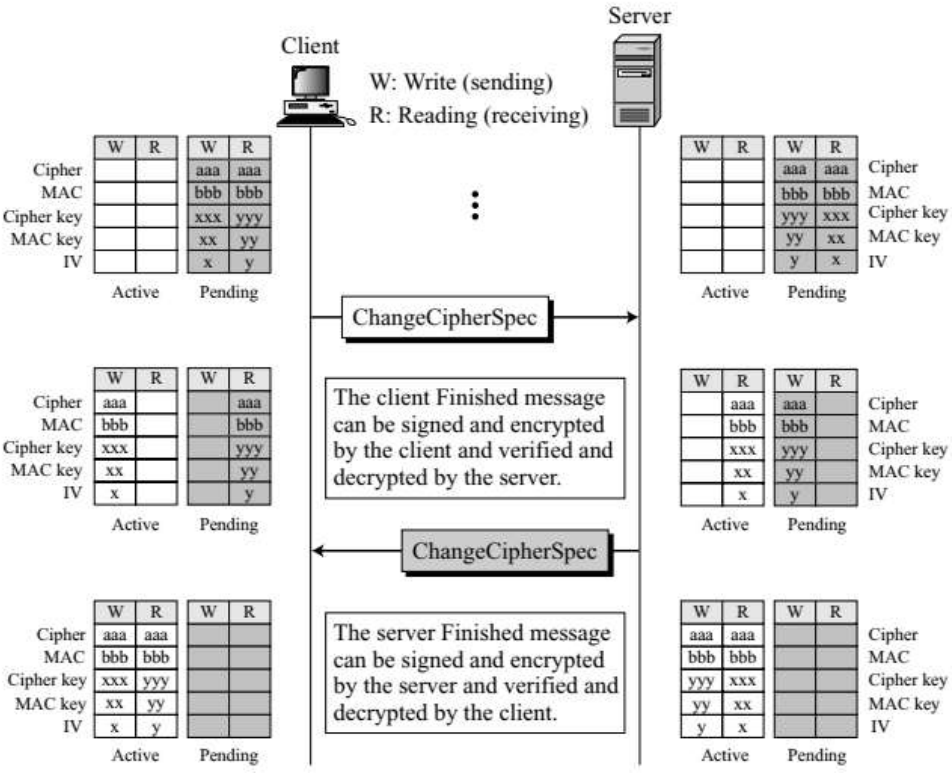
ChangeCipherSpec The client sends a ChangeCipherSpec message to show tha has moved all of the cipher suite set and the parameters from the pending state to t active state. This message is actually part of the ChangeCipherSpec Protocol that will discuss later.

Finished The next message is also sent by the client. It is a Finished message tl announces the end of the handshaking protocol by the client.

ChangeCipherSpec The server sends a ChangeCipherSpec message to show that it l also moved all of the cipher suite set and parameters from the pending state to the act state. This message is part of the ChangeCipherSpec Protocol, which will be discussed la

Finished Finally, the server sends a Finished message to show that handshaking totally completed.

ChangeCipherSpec Protocol:
The ChangeCipherSpec Protocol defines the process of moving values between the pending and active states.



Alert Protocol

SSL uses the Alert Protocol for reporting errors and abnormal conditions. It has only one message type, the Alert message, that describes the problem and its level (warning or fatal).

Record Protocol

The Record Protocol carries messages from the upper layer (Handshake Protocol, ChangeCipherSpec Protocol, Alert Protocol, or application layer). The message is fragmented and optionally compressed; a MAC is added to the compressed message using the negotiated hash algorithm. The compressed fragment and the MAC are encrypted using the negotiated encryption algorithm. Finally, the SSL header is added to the encrypted message.

(OR)

9 a) Define Security Association (SA) and explain its purpose. CO4 L3 7M

Security Association is a very important aspect of IPSec. IPSec requires a logical relationship, called a Security Association (SA), between two hosts.

Idea of Security Association

A Security Association is a contract between two parties; it creates a secure channel between them. Let us assume that Alice needs to unidirectionally communicate with Bob. If Alice and Bob are interested only in the confidentiality aspect of security, they can get a shared secret key between themselves. We can say that there are two Security Associations (SAs) between Alice and Bob; one outbound SA and one inbound SA. Each of them stores the value of the key in a variable and the name of the encryption/ decryption algorithm in another. Alice uses the algorithm and the key to encrypt a message to Bob; Bob uses the algorithm and the key when he needs to decrypt the message received from Alice.

Security Association Database (SAD)

A Security Association can be very complex. This is particularly true if Alice wants to send messages to many people and Bob needs to receive messages from many people. In addition, each site needs to have both inbound and outbound SAs to allow bidirectional communication. In other words, we need a set of SAs that can be collected into a database. This database is called the Security Association Database (SAD). The database can be thought of as a two-dimensional table with each row defining a single SA. Normally, there are two SADs, one inbound and one outbound.

< SPI, DA, P >							
< SPI, DA, P >							
< SPI, DA, P >							
< SPI, DA, P >							

Security Association Database

❑ Security Parameter Index. The security parameter index (SPI) is a 32-bit number that defines the SA at the destination. As we will see later, the SPI is determined during the SA negotiation. The same SPI is included in all IPSec packets belonging to the same inbound SA.

❑ Destination Address. The second index is the destination address of the host. We need to remember that a host in the Internet normally has one unicast destination address, but it may have several multicast addresses. IPSec requires that the SAs be unique for each destination address.

❑ Protocol. IPSec has two different security protocols: AH and ESP. To separate the parameters and information used for each protocol, IPSec requires that a destination define a different SA for each protocol.

Parameters	*Descriptions*
Sequence Number Counter	This is a 32-bit value that is used to generate sequence numbers for the AH or ESP header.
Sequence Number Overflow	This is a flag that defines a station's options in the event of a sequence number overflow.
Anti-Replay Window	This detects an inbound replayed AH or ESP packet.
AH Information	This section contains information for the AH protocol: 1. Authentication algorithm 2. Keys 3. Key lifetime 4. Other related parameters
ESP Information	This section contains information for the ESP protocol: 1. Encryption algorithm 2. Authentication algorithm 3. Keys 4. Key lifetime 5. Initiator vectors 6. Other related parameters
SA Lifetime	This defines the lifetime for the SA.
IPSec Mode	This defines the mode, transport or tunnel.
Path MTU	This defines the path MTU (fragmentation).

b) List ISAKMP payload types and the purpose of each type. CO4 L2 7M

Types	*Name*	*Brief Description*
0	None	Used to show the end of the payloads
1	SA	Used for starting the negotiation
2	Proposal	Contains information used during SA negotiation
3	Transform	Defines a security transform to create a secure channel
4	Key Exchange	Carries data used for generating keys
5	Identification	Carries the identification of communication peers
6	Certification	Carries a public-key certificate
7	Certification Request	Used to request a certificate from the other party
8	Hash	Carries data generated by a hash function
9	Signature	Carries data generated by a signature function
10	Nonce	Carries randomly generated data as a nonce
11	Notification	Carries error message or status associated with an SA
12	Delete	Carries one more SA that the sender has deleted
13	Vendor	Defines vendor-specification extensions

SA Payload

The SA payload is used to negotiate security parameters. However, these parameters are not included in the SA payload; they are included in two other payloads (proposal and transform). The SA payload just defines the domain of interpretation field and the situation field.

Proposal Payload

The proposal payload initiates the mechanism of negotiation. Although by itself it does not propose any parameters, it does define the protocol identification and the SPI. The parameters for negotiation are sent in the transform payload that follows. Each proposal payload is followed by one or more transform payloads that give alternative sets of parameters.

Transform Payload

The transform payload actually carries attributes of the SA negotiation.

Key-Exchange Payload

The key exchange payload is used in those exchanges that need to send preliminary keys that are used for creating session keys. For example, it can be used to send a Diffie-Hellman half-key.

Identification Payload

The identification payload allows entities to send their identifications to each other.

Certification Payload

Anytime during the exchange, an entity can send its certification (for public-encryption/ decryption keys or signature keys). Although the inclusion of the certification payload in an exchange is normally optional, it needs to be included if there is no secure directory available to distribute the certificates.

Certificate Request Payload

Each entity can explicitly request a certificate from the other entity using the certificate request payload.

Hash Payload

The hash payload contains data generated by the hash function as described in the IKE exchanges. The hash data guarantee the integrity of the message or part of the ISAKMP states.

Signature Payload

The signature payload contains data generated by applying the digital signature procedure over some part of the message or ISAKMP state.

Nonce Payload

The nonce payload contains random data used as a nonce to assure liveliness of the message and to prevent a replay attack.

Notification Payload

During the negotiation process, sometimes a party needs to inform the other party of the status or errors.

Delete Payload

The delete payload is used by an entity that has deleted one or more SAs and needs to inform the peer that these SAs are no longer supported.

Vendor Payload

ISAKMP allows the exchange of information particular to a specific vendor.