(Please write your Roll No. Immediately) Roll. No----------

Sample Paper for End Term Examination

B.Tech- VIII Semester **Feb., 2014**
Paper Code: ETIT 412 **Subject: Network Security**
Time: 3 Hours **Max. Marks: 75**

Note: Question No. 1 is compulsory. Attempt any four Questions from rest.

Q.N.1 Attempt all questions. (5*5)

A. What is the OSI security architecture?
B. List and briefly define categories of security services.
C. List and briefly define categories of security mechanisms.
D. Difference between block cipher and stream cipher.
E. Explain any substitution techniques for cryptography.

Q.N.2 A. Explain different type of Operation Modes for Block ciphers. (7.5)

B. What is Cryptanalysis attack? (5)

Q.N.3. A. what is Differential Cryptanalysis and Linear Cryptanalysis of DES? (7.5)

B. What is MAC? Explain in brief. (5)

Q.N.4. A. Give some web security threats and there countermeasures. (7.5)

B. What is Difference between HTTP and HTTPS protocol? (5)

Q.N.5. A. What is key exchange algorithm? Explain. (7.5)

B. write a short note on HTTPS. (5)

Q.N.6. A. Explain PGP for e-mail security. (6)

B. Describe SMIME. (6.5)

Q.N.7 what is IP sec protocol. Explain IN details with operation mode and its application.

Draw the frame format of IPsec also. (12.5)

ANSWERS

Q.N.1 A. For our purposes, the OSI security architecture provides a useful, if abstract, overview of many of the concepts that this book deals with. The OSI security architecture focuses on security attacks, mechanisms, and services. These can be defined briefly as

• **Security attack:** Any action that compromises the security of information owned by an organization.

• **Security mechanism:** A process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack.

• **Security service**: A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

B. The assurance that the communicating entity is the one that it claims to be.

Access control: The prevention of unauthorized use of a resource (i.e., this service controls who can have access to a resource, under what conditions access can occur, and what those accessing the resource are allowed to do).

Data confidentiality: The protection of data from unauthorized disclosure.

Data integrity: The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

Nonrepudiation: Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

Availability service: The property of a system or a system resource being accessible and usable upon demand by an authorized system entity, according to performance specifications for the system (i.e., a system is available if it provides services according to the system design whenever users request them).

C. SPECIFIC SECURITY MECHANISMS

May be incorporated into the appropriate protocol layer in order to provide some of the OSI security services.

Encipherment

The use of mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and zero or more encryption keys.

Digital Signature

Data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient).

Access Control

A variety of mechanisms that enforce access rights to resources.

Data Integrity

A variety of mechanisms used to assure the integrity of a data unit or stream of data units.

Authentication Exchange

A mechanism intended to ensure the identity of an entity by means of information exchange.

Traffic Padding

The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.

Routing Control

Enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.

Notarization

The use of a trusted third party to assure certain properties of a data exchange.

PERVASIVE SECURITY MECHANISMS

Mechanisms that are not specific to any particular OSI security service or protocol layer.

Trusted Functionality

That which is perceived to be correct with respect to some criteria (e.g., as established by a security policy).

Security Label

The marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource.

Event Detection

Detection of security-relevant events.

Security Audit Trail

Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.

Security Recovery

Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

D. Stream Ciphers vs. Block Ciphers

Unlike what we've seen, private-key (aka symmetric) encryption schemes used in practice generally

- are not be based on nice computational problems
- are not proven secure via reductions
- are designed for a particular input length (so can only be treated with concrete security)

Stream Ciphers

-Essentially meant to be pseudorandom generators, used for statefull encryption.

-Examples: linear feedback shift registers (not secure, but used as component in better stream ciphers), RC4, SEAL

-Extremely simple and fast

-Practical issues: can generate pseudorandom bits online and decrypt very quickly without Requires synchronization

Block ciphers

-Examples: DES, AES, IDEA,

-Main tools for private-key encryption in practice.

Have both stateless modes and stateful/stream-like modes

E. SUBSTITUTION TECHNIQUES

Substitution technique is one that the letters in the plaintext will be replaced by other letters or by numbers or symbols.

[Caesar Cipher]

The earliest use of substitution cipher is also the simplest one that is proposed by Julius Caesar, called Caesar Cipher. The Caesar Cipher works with replacing each letter with the letter standing three places further down of the alphabet order. For

Example: plaintext: a b c d e f g h w x y z cipher text: e f g h i j k l z a b c

So if the plaintext is “meet me after the party”.

The cipher text would be “phhw

ph diwhu wkh sduwb”. 5

Plaintext: meet me after the party

Cipher text: phhw ph diwhu wkh sduwb

If we assign each letter a number from 0 to 25(from A to Z). Take the Cipher text as C, Encryption as E, and plaintext as P. Then we can describe the Caesar Cipher as below

$C=E(p)=(p+3)\mathrm{mod}(26)$ (1)

A shift could be any amount, so the general Caesar algorithm is

$C=E(p)=(p+k)\mathrm{mod}(26)$ (2)

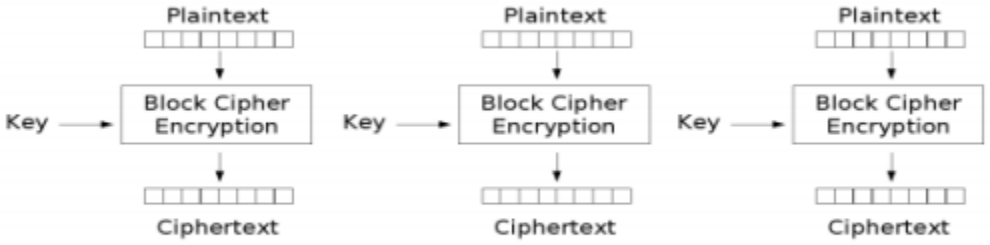
Where k takes on a value in the range from 1 to 25. And the decryption algorithm is simply $p=D(C)=(C-k)\mathrm{mod}(26)$ (3)

If it is known that a given cipher text is a Caesar cipher, then a brute-force cryptanalysis will be easily performed. Just try all the 25 for the possible value of k. In this example, there are three reasons for us to use the brute-force cryptanalysis. First is that the encryption and the decryption algorithms are known. Second is that there are only 25 keys to try. Third is that the language of the plaintext is known and easily recognizable. For general cases, we always assume that the first condition is held, that is the algorithms of encryption and decryption are always known by the enemy who want to break the cipher. What really makes the brute-force attack impractical is that most of the algorithms use a large number of keys, that is, the second condition. For example, the triple DES algorithm uses a 168-bit key which makes people who choose to use the brute-force attacking way wasting resources or time. And the third condition is also important. If the language of the plaintext is unknown, we do not have any idea to recognize that if the key we try is right even in the trial that is right.

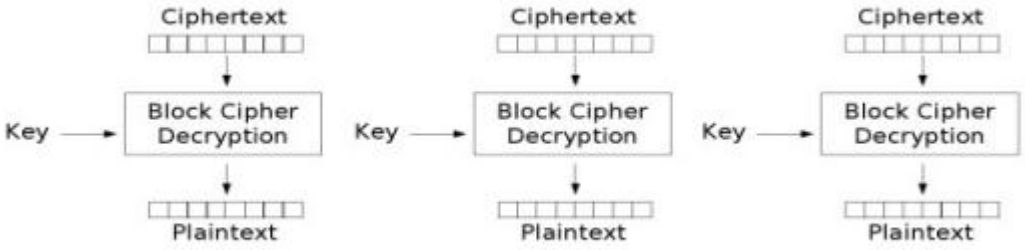
Q.N.2 A. Here we simply introduce some modes to implement block ciphers. These different modes we call them”Operation Modes”. We choose one of them to implement the block cipher by considering the different kind of outstanding threatens.

ECB (electronic codebook mode) :

The simplest sense of Block cipher is ECB mode. In ECB mode, each encryption and decryption of the data blocks are independent from one another. It means that the speed of ECB mode is very fast because the parallel inputs and parallel outputs could be used. And the transmission errors will be confined inside the single block, and will not influence on the other blocks. The drawback of ECB mode is that the same plaintext input will have the same cipher text output. It would be an advantage that the attackers could take on.



Electronic Codebook (ECB) mode encryption

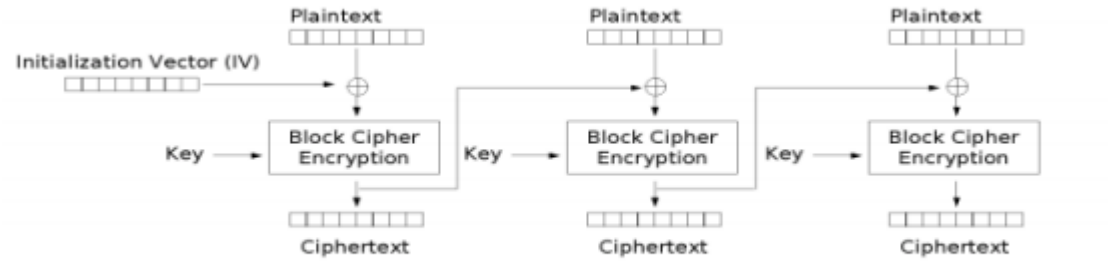


Electronic Codebook (ECB) mode decryption

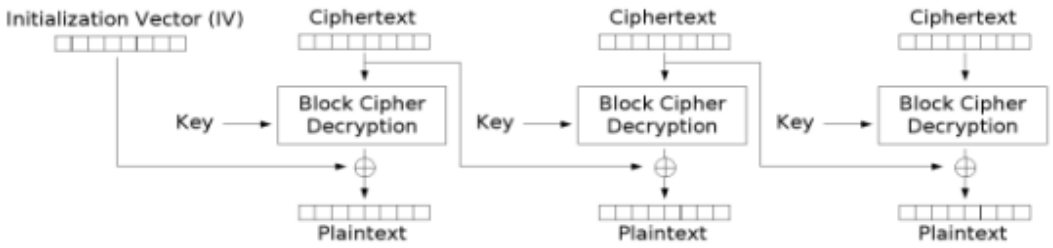
CBC (cipher block chaining mode) :

CBC mode efficiently solves the security problem of ECB. The encryption of CBC is to do XOR between the current plaintext and the former plaintext, then deal the result from the above with the key. And the output is the current cipher. Decryption is quite simple that we could use the specification of the XOR. We could realize the detail by checking out Fig.8 and Fig.9. (Notice that there is a IV, initialization vector in the first step where there is no former ciphertext.)The disadvantage of CBC is that the processing speed in CBC is slower than ECB because the parrell

inputs cannot be used here.



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Q.N.2. B. Cryptanalysis is the methods to attack cryptographic protection. There are several ways to achieve this goal. A cipher is breakable if it is possible to determine the plaintext or key from the ciphertext, or to determine the key from the plaintext-ciphertext pair. There is a kind of attack called “Brute-force attack” which means that the attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried before succeeding. Besides Brute-force attack, there are three typical attacking situations we describe as followed.

1 ciphertext-only attack: the attackers only know the information of ciphertext and the detail of how encryption works, that is they know the encryption algorithm. (The Kirchhoff principle).

2 known-plaintext attack: the attackers know the encryption algorithm and the ciphertext just as cipher-only attackers know. And additional, they know one or more ciphertext – plaintext pairs generated by the secret key.

3 chosen-plaintext attack: the attackers not only know the encryption algorithm but also have the ability to modify the input plaintext and observe the corresponding output ciphertext. Among the

three situations, no doubts that the chosen-plaintext attack is the most threaten one in the point of view of the original data protectors. A cipher is called “unconditional secure” if no matter how much the ciphertext is intercepted, there is not enough information to determine the corresponding plaintext uniquely. By the way, we have to realize that all ciphers are breakable if given unlimited resources. So generally speaking, the “computationally secure” is sometimes more meaningful, which means if it can be broken by systematic analysis with available limited resources.
Computationally secure is established with the two criteria meet at the same time:
1. The cost of breaking the cipher exceeds the value of the encrypted information.
2. The time required to break the cipher exceeds the useful lifetime of the information.

Q.N.3. A. Differential Cryptanalysis (Biham-Shamir)
Main idea:
•This is a chosen plaintext attack, assumes than an attacker knows (Plaintext, Ciphertext) pairs
• Diff. Cryptanalysis involves comparing the XOR of 2 plaintexts to the XOR of the 2 corresponding cipher texts
•Difference $\Delta P = P1 \oplus P2$, $\Delta C = C1 \oplus C2$
•Distribution of ΔC’s given ΔP may reveal information about the key (certain key bits)
•After finding several bits, use brute-force for the rest of the bits to find the key.
•Another attack described in 1993 by M. Matsui
•Instead of looking for isolated points at which a block cipher behaves like something simpler it involves trying to create a simpler approximation to the block cipher as a whole.
•It is an attack that can be applied to an iterated cipher.

Linear Cryptanalysis of DES
•Another attack described in 1993 by M. Matsui
•Instead of looking for isolated points at which a block cipher behaves like something simpler, it involves trying to create a simpler approximation to the block cipher as a whole.
•It is an attack that can be applied to an iterated cipher.
•M. Matsui showed (1993/1994) that DES can be broken:
–8 rounds: 2^{21} known plaintext

–16 rounds: 2^{43} known plaintext, 40 days to generate the pairs (plaintext, ciphertext) and 10 days to find the key

•The attack has no practical implication, requires too many pairs.

•Exhaustive search remains the most effective attack.

DES Strength Against Various Attacks

Attack Method	Known	Chosen	Storage Complexity	Processing Complexity
Exhaustive precomputation	-	1	2^{56}	1
Exhaustive search	1	-	Negligible	2^{55}
Linear cryptanalysis	2^{43} 2^{38}	- -	For texts	2^{43} 2^{50}
Differential cryptanalysis	- 2^{55}	2^{47} -	For texts	2^{47} 2^{55}

Q.N.3. B. Meet-in-the–middle attack on double encryption

This attack requires knowing some plaintext/ciphertext pairs. Let's assume that we have a plaintext/ciphertext pair; i.e., we know the plaintext p and the corresponding (double DES enciphered) ciphertext C. Attacks on DES have typically been brute force attacks (see "Breaking DES"); so, we will use brute force here.

Here is the double encryption:

p→ → E k , p E k , E k , p =C ( ) 1 21 (( ))

Encrypt p using 2^{56} all possible keys, and store the results. (Storage could be a problem.)
The stored results will include all possible encryptions p→E k, p (1).

Then decrypt C using all possible keys. 2^{56}

D k, C =D k , E k , E k , p E k , p ( ) 2 221 1 ( ) (( )) → ( )

After decrypting with each key, check for a match with the stored outputs of the 2^{56} possible encryptions. When we have a match, we have located a possibly correct pair of keys. Now, perhaps more than one pair of keys will result in a match, but the number of pairs of keys that return matches should be small. We could try each possible pair of keys. If more than one plaintext/ciphertext correspondence is known (for the key pair), then other correspondences could be used to check which of the keys is correct

So, it only takes twice as long to break double DES using brute force. Because DES has 56-bit security, double DES has $2x2^{56} = 2^{57}$ security.

Q.N.3. A. RSA Algo.

One of the first public-key schemes was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978 [RIVE78]. The RSA scheme has since that time reigned supreme as the most widely accepted and implemented approach to public-key encryption. **RSA** is a block cipher in which the plaintext and ciphertext are integers between 0 and *n* 1 for some *n.* Encryption and decryption are of the following form period for some plaintext block *M* and ciphertext block *C*:

C Me mod *n*

M Cd mod *n* (*Me*)*d* mod *n Med* mod *n*

Both sender and receiver must know the values of *n* and *e*, and only the receiver knows the value of *d.* This is a public-key encryption algorithm with a public key of *KU* {*e*, *n*} and a private key of *KR* {*d*, *n*}. For this algorithm to be satisfactoryfor public-key encryption, the following requirements must be met.

1. It is possible to find values of *e*, *d*, *n* such that *Med* mod *n M*for all *M* < *n.*

2. It is relatively easy to calculate *Me* and *Cd* for all values of *M*< *n.*

3. It is infeasible to determine *d* given *e* and *n.*

The first two requirements are easily met. The third requirement can be met for large values of *e* and *n.*

Key Generation	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	gcd $(\phi(n), e) = 1$; $1 < e < \phi(n)$
Calculate d	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption	
Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

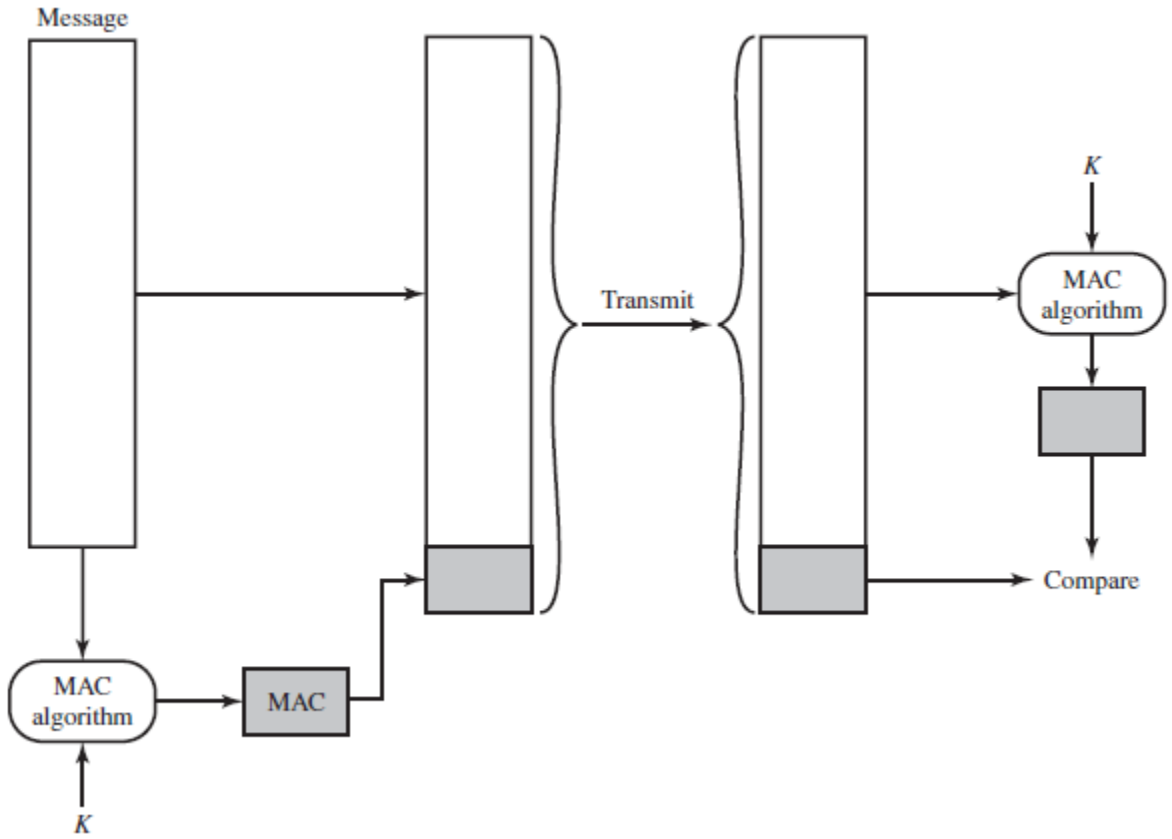
Q.N.3.B. ***MESSAGE AUTHENTICATION CODE*** One authentication technique involves the use of a secret key to generate a small block of data, known as a **message authentication code (MAC)**, that is appended to the message. This technique assumes that two communicating parties, say A and B, share a common secret key *KAB.* When A has a message to send to B, it calculates the message authentication code as a function of the message and the key: MAC*M* F(*KAB*,*M*).The message plus code are transmitted to the intended recipient. The recipient performs the same calculation on the received message, using the same secret key, to generate a new message authentication code. The received code is compared to the calculated code. If we assume that only the receiver and the sender know the identity of the secret key, and if the received code matches the calculated code, then the following statements apply:

1. The receiver is assured that the message has not been altered. If an attacker alters the message but does not alter the code, then the receiver's calculation of the code will differ from the received

code. Because the attacker is assumed not to know the secret key, the attacker cannot alter the code to correspond to the alterations in the message.

2. The receiver is assured that the message is from the alleged sender. Because no one else knows the secret key, no one else could prepare a message with a proper code.

3. If the message includes a sequence number (such as is used with HDLC and TCP), then the receiver can be assured of the proper sequence, because an attacker cannot successfully alter the sequence number.



A number of algorithms could be used to generate the code. The NIST specification, FIPS PUB 113, recommends the use of DES. DES is used to generate an encrypted version of the message, and the last number of bits of ciphertext are used as the code. A 16- or 32-bit code is typical. The process just described is similar to encryption. One difference is that the authentication algorithm need not be reversible, as it must for decryption. Because of the mathematical properties of the authentication function, it is less vulnerable to being broken than encryption.

Q.N.4. A.

	Threats	Consequences	Countermeasures
Integrity	• Modification of user data • Trojan horse browser • Modification of memory • Modification of message traffic in transit	• Loss of information • Compromise of machine • Vulnerabilty to all other threats	Cryptographic checksums
Confidentiality	• Eavesdropping on the net • Theft of info from server • Theft of data from client • Info about network configuration • Info about which client talks to server	• Loss of information • Loss of privacy	Encryption, Web proxies
Denial of Service	• Killing of user threads • Flooding machine with bogus requests • Filling up disk or memory • Isolating machine by DNS attacks	• Disruptive • Annoying • Prevent user from getting work done	Difficult to prevent
Authentication	• Impersonation of legitimate users • Data forgery	• Misrepresentation of user • Belief that false information is valid	Cryptographic techniques

Q.N.4. B. **Hypertext Transfer Protocol** (**HTTP**) is a protocol used in networking. When you type any web address in your web browser, your browser acts as a client, and the computer having the requested information acts as a server. When client requests for any information from the server, it uses HTTP protocol to do so. The server responds back to the client after the request completes. The response comes in the form of web page which you see just after typing the web address and press "Enter".

Hypertext Transfer Protocol Secure (HTTPS) is a combination of two different protocols. It is more secure way to access the web. It is combination of Hypertext Transfer Protocol (HTTPS) and SSL/TLS protocol. It is more secure way to sending request to server from a client, also the communication is purely encrypted which means no one can know what you are looking for. This kind of communication is used for accessing those websites where security is required. Banking websites, payment gateway, emails (Gmail offers HTTPS by default in Chrome browser), and corporate sector websites are some great examples where HTTPS protocols are used.

For HTTPS connection, public key trusted and signed certificate is required for the server. These certificate comes either free or it costs few dollars depends on the signing authority. There is one other method for distributing certificates. Site admin creates certificates and loads in the browser of users. Now when user requests information to the web server, his identity can be verified easily.

HTTP	HTTPS
URL begins with "http://"	URL begins with "https://"
It uses port 80 for communication	It uses port 443 for communication
Unsecured	Secured
Operates at Application Layer	Operates at Transport Layer
No encryption	Encryption is present
No certificates required	Certificates required

Q.N.5. A. Diffie {Hellman key exchange

The Diffie {Hellman key exchange algorithm solves the following dilemma. Alice and Bob want to share a secret key for use in a symmetric cipher, but their only means of communication is insecure. Every piece of information that they exchange is observed by their adversary Eve. How is it possible for Alice and Bob to share a key without making it available to Eve? At first glance it appears that Alice and Bob face an impossible task. It was a brilliant insight of Diffie and Hellman that the difficulty of the discrete logarithm problem for $_{Fp}$ provides a possible solution.

The first step is for Alice and Bob to agree on a large prime p and a nonzero integer g modulo p. Alice and Bob make the values of p and g public knowledge; for example, they might post the values on their web sites, so Eve knows them, too. For various reasons to be discussed later, it is best if they choose g such that its order in F¤p is a large prime. (See Exercise 1.31 for a way of finding such a g.)

The next step is for Alice to pick a secret integer a she does not reveal to anyone, while at the same time Bob picks an integer b that he keeps secret. Bob and Alice use their secret integers to compute

$$\underbrace{A \equiv g^a \pmod{p}}_{\text{Alice computes this}} \quad \text{and} \quad \underbrace{B \equiv g^b \pmod{p}}_{\text{Bob computes this}}.$$

They next exchange these computed values, Alice sends A to Bob and Bob sends B to Alice. Note that Eve gets to see the values of A and B, since they are sent over the insecure communication channel.

Finally, Bob and Alice again use their secret integers to compute

$$\underbrace{A' \equiv B^a \pmod{p}}_{\text{Alice computes this}} \quad \text{and} \quad \underbrace{B' \equiv A^b \pmod{p}}_{\text{Bob computes this}}.$$

The values that they compute, A' and B' respectively, are actually the same, since

$$A' \equiv B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \equiv B' \pmod{p}.$$

This common value is their exchanged key. The Diffie–Hellman key exchange algorithm is summarized in Table 2.2.

Public Parameter Creation	
A trusted party chooses and publishes a (large) prime p and an integer g having large prime order in $\mathbb{F}_p^*$.	
Private Computations	
Alice	**Bob**
Choose a secret integer a.	Choose a secret integer b.
Compute $A \equiv g^a \pmod{p}$.	Compute $B \equiv g^b \pmod{p}$.
Public Exchange of Values	
Alice sends A to Bob $\longrightarrow$	A
B $\longleftarrow$	Bob sends B to Alice
Further Private Computations	
Alice	**Bob**
Compute the number $B^a \pmod{p}$.	Compute the number $A^b \pmod{p}$.
The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$.	

Q.N.5 B. HTTPS (HTTP over SSL) refers to the combination of HTTP and SSL to implement secure communication between a Web browser and a Web server. The HTTPS capability is built into all modern Web browsers. Its use depends on the Web server supporting HTTPS communication. For example, search engines do not supportHTTPS.

The principal difference seen by a user of a Web browser is that URL (uniform resource locator) addresses begin with https:// rather than http://. A normal HTTP connection uses port 80. If HTTPS is specified, port 443 is used, which invokes SSL.

When HTTPS is used, the following elements of the communication areencrypted:

- URL of the requested document
- Contents of the document
- Contents of browser forms (filled in by browser user)
- Cookies sent from browser to server and from server to browser

• Contents of HTTP header

HTTPS is documented in RFC 2818, *HTTP over TLS*. There is no fundamental change in using HTTP over either SSL or TLS, and both implementations are referred to as HTTPS.

Connection Initiation

For HTTPS, the agent acting as the HTTP client also acts as the TLS client. The client initiates a connection to the server on the appropriate port and then sends the TLS ClientHello to begin the TLS handshake. When the TLS handshake has finished, the client may then initiate the first HTTP request. All HTTP data is to be sent as TLS application data. Normal HTTP behavior, including retained connections, should be followed. We need to be clear that there are three levels of awareness of a connection in HTTPS. At the HTTP level, an HTTP client requests a connection to an HTTP server by sending a connection request to the next lowest layer. Typically, the next lowest layer is TCP, but it also may be TLS/SSL. At the level of TLS, a session is established between a TLS client and a TLS server. This session can support one or more connections at any time. As we have seen, a TLS request to establish a connection begins with the establishment of a TCP connection between the TCP entity on the client side and the TCP entity on the server side.

Connection Closure

An HTTP client or server can indicate the closing of a connection by including the following line in an HTTP record: Connection: close. This indicates that the connection will be closed after this record is delivered. The closure of an HTTPS connection requires that TLS close the connection with the peer TLS entity on the remote side, which will involve closing the underlying TCP connection. At the TLS level, the proper way to close a connection is for each side to use the TLS alert protocol to send a close_notify alert. TLS implementations must initiate an exchange of closure alerts before closing a connection. A TLS implementation may, after sending a closure alert, close the connection without waiting for the peer to send its closure alert, generating an "incomplete close". Note that an implementation that does this may choose to reuse the session. This should only be done when the application knows (typically through detecting HTTP message boundaries) that it has received all the message data that it cares about. HTTP clients also must be able to cope with a situation in which the underlying TCP connection is terminated without a prior close_notify alert and without a Connection: close indicator. Such a situation could be due to a programming error on the server or a communication error that causes the TCP connection to drop.

However, the unannounced TCP closure could be evidence of some sort of attack. So the HTTPS client should issue some sort of security warning when this occurs.

Q.N.6. A.

PGP is a remarkable phenomenon. Largely the effort of a single person, Phil Zimmermann, PGP provides a confidentiality and authentication service that can be used for electronic mail and file storage applications. In essence, Zimmermann has done the following:

1. Selected the best available cryptographic algorithms as building blocks.

2. Integrated these algorithms into a general-purpose application that is independent of operating system and processor and that is based on a small set of easy-to-use commands.

3. Made the package and its documentation, including the source code, freely available via the Internet, bulletin boards, and commercial networks such as AOL (America On Line).

4. Entered into an agreement with a company (Via crypt, now Network Associates) to provide a fully compatible, low-cost commercial version of PGP.

PGP has grown explosively and is now widely used. A number of reasons can be cited for this growth.

1. It is available free worldwide in versions that run on a variety of platforms, including Windows, UNIX, Macintosh, and many more. In addition, the commercial version satisfies users who want a product that comes with vendor support.

2. It is based on algorithms that have survived extensive public review and are considered extremely secure. Specifically, the package includes RSA, DSS, and Diffie-Hellman for public-key encryption;CAST-128, IDEA, and 3DES for symmetric encryption; and SHA-1 for hash coding.

3. It has a wide range of applicability, from corporations that wish to select and enforce a standardized scheme for encrypting files and messages to individuals who wish to communicate securely with others worldwide over the Internet and other networks.

4. It was not developed by, nor is it controlled by, any governmental or standards organization. For those with an instinctive distrust of "the establishment," this makes PGP attractive.

5. PGP is now on an Internet standards track (RFC 3156; *MIME Security with OpenPGP*). Nevertheless, PGP still has an aura of an antiestablishment endeavor. We begin with an overall look at the operation of PGP. Next, we examine how cryptographic keys are created and stored. Then, we address the vital issue of public-key management.

Summary of PGP

Function	Algorithms Used	Description
Digital signature	DSS/SHA or RSA/SHA	A hash code of a message is created using SHA-1. This message digest is encrypted using DSS or RSA with the sender's private key and included with the message.
Message encryption	CAST or IDEA or Three-key Triple DES with Diffie-Hellman or RSA	A message is encrypted using CAST-128 or IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and included with the message.
Compression	ZIP	A message may be compressed for storage or transmission using ZIP.
E-mail compatibility	Radix-64 conversion	To provide transparency for e-mail applications, an encrypted message may be converted to an ASCII string using radix-64 conversion.

Q.N.6. B.

Secure/Multipurpose Internet Mail Extension (S/MIME) is a security enhancement to the MIME Internet e-mail format standard based on technology from RSA Data Security. Although both PGP and S/MIME are on an IETF standards track, it appears likely that S/MIME will emerge as the industry standard for commercial and organizational use, while PGP will remain the choice for personal e-mail security for many users. S/MIME is defined in a number of documents—most importantly RFCs 3370, 3850, 3851, and 3852. To understand S/MIME, we need first to have a general understanding of the underlying e-mail format that it uses, namely MIME. But to understand the significance of MIME, we need to go back to the traditional e-mail format standard, RFC 822, which is still in common use. The most recent version of this format specification is RFC 5322 (*Internet Message Format*). Accordingly, this section first provides an introduction to these two earlier standards and then moves on to a discussion of S/MIME.

MIME Content types

Type	Subtype	Description
Text	Plain	Unformatted text; may be ASCII or ISO 8859.
	Enriched	Provides greater format flexibility.
Multipart	Mixed	The different parts are independent but are to be transmitted together. They should be presented to the receiver in the order that they appear in the mail message.
	Parallel	Differs from Mixed only in that no order is defined for delivering the parts to the receiver.
	Alternative	The different parts are alternative versions of the same information. They are ordered in increasing faithfulness to the original, and the recipient's mail system should display the "best" version to the user.
	Digest	Similar to Mixed, but the default type/subtype of each part is message/rfc822.
Message	rfc822	The body is itself an encapsulated message that conforms to RFC 822.
	Partial	Used to allow fragmentation of large mail items, in a way that is transparent to the recipient.
	External-body	Contains a pointer to an object that exists elsewhere.
Image	jpeg	The image is in JPEG format, JFIF encoding.
	gif	The image is in GIF format.
Video	mpeg	MPEG format.
Audio	Basic	Single-channel 8-bit ISDN mu-law encoding at a sample rate of 8 kHz.
Application	PostScript	Adobe Postscript format.
	octet-stream	General binary data consisting of 8-bit bytes.

MIME transfer encoding

7bit	The data are all represented by short lines of ASCII characters.
8bit	The lines are short, but there may be non-ASCII characters (octets with the high-order bit set).
binary	Not only may non-ASCII characters be present, but the lines are not necessarily short enough for SMTP transport.
quoted-printable	Encodes the data in such a way that if the data being encoded are mostly ASCII text, the encoded form of the data remains largely recognizable by humans.
base64	Encodes data by mapping 6-bit blocks of input to 8-bit blocks of output, all of which are printable ASCII characters.
x-token	A named nonstandard encoding.

Cryptography use in SMIME

Function	Requirement
Create a message digest to be used in forming a digital signature.	MUST support SHA-1. Receiver SHOULD support MD5 for backward compatibility.
Encrypt message digest to form a digital signature.	Sending and receiving agents MUST support DSS. Sending agents SHOULD support RSA encryption. Receiving agents SHOULD support verification of RSA signatures with key sizes 512 bits to 1024 bits.
Encrypt session key for transmission with a message.	Sending and receiving agents SHOULD support Diffie-Hellman. Sending and receiving agents MUST support RSA encryption with key sizes 512 bits to 1024 bits.
Encrypt message for transmission with a one-time session key.	Sending and receiving agents MUST support encryption with tripleDES. Sending agents SHOULD support encryption with AES. Sending agents SHOULD support encryption with RC2/40.
Create a message authentication code.	Receiving agents MUST support HMAC with SHA-1. Sending agents SHOULD support HMAC with SHA-1.

Content type of SMIME

Type	Subtype	smime Parameter	Description
Multipart	Signed		A clear-signed message in two parts: one is the message and the other is the signature.
Application	pkcs7-mime	signedData	A signed S/MIME entity.
	pkcs7-mime	envelopedData	An encrypted S/MIME entity.
	pkcs7-mime	degenerate signedData	An entity containing only public-key certificates.
	pkcs7-mime	CompressedData	A compressed S/MIME entity.
	pkcs7-signature	signedData	The content type of the signature subpart of a multipart/signed message.

Q.N.7.

In 1994, the Internet Architecture Board (IAB) issued a report titled “Security in the Internet Architecture” (RFC 1636). The report identified key areas for security mechanisms. Among these were the need to secure the network infrastructure from unauthorized monitoring and control of network traffic and the need to secure end-user-to-end-user traffic using authentication and encryption mechanisms. To provide security, the IAB included authentication and encryption as necessary security features in the next-generation IP, which has been issued as IPv6. Fortunately, these security capabilities were designed to be usable both with the current IPv4 and the future

IPv6. This means that vendors can begin offering these features now, and many vendors now do have some IPsec capability in their products. The IPsec specification now exists as a set of Internet standards.

Applications of IPsec

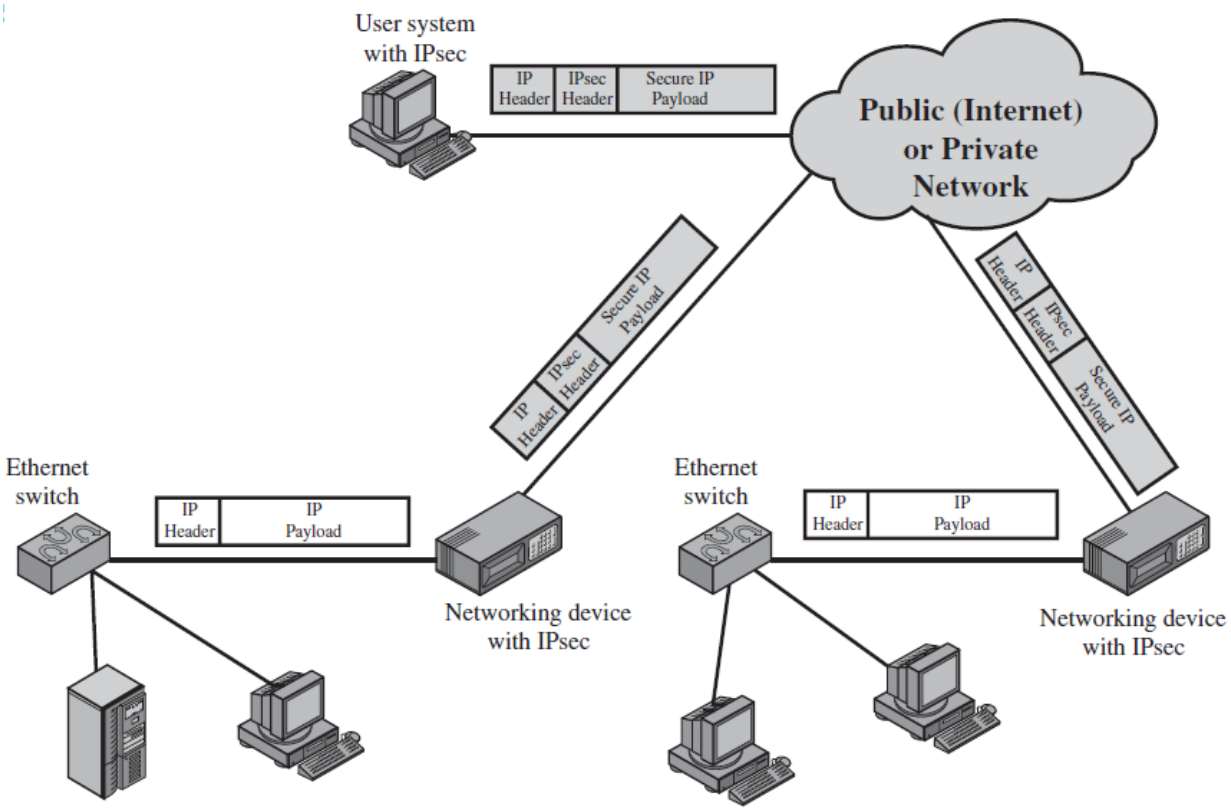
IPsec provides the capability to secure communications across a LAN, across private and public WANs, and across the Internet. Examples of its use include:

• **Secure branch office connectivity over the Internet:** A company can build a secure virtual private network over the Internet or over a public WAN. This enables a business to rely heavily on the Internet and reduce its need for private networks, saving costs and network management overhead.

• **Secure remote access over the Internet:** An end user whose system is equipped with IP security protocols can make a local call to an Internet Service Provider (ISP) and gain secure access to a company network. This reduces the cost of toll charges for traveling employees and telecommuters.

• **Establishing extranet and intranet connectivity with partners:** IPsec can be used to secure communication with other organizations, ensuring authentication and confidentiality and providing a key exchange mechanism.

• **Enhancing electronic commerce security:** Even though some Web and electronic commerce applications have built-in security protocols, the use of IPsec enhances that security. IPsec guarantees that all traffic designated by the network administrator is both encrypted and authenticated, adding an additional layer of security to whatever is provided at the application layer. Figure is a typical scenario of IPsec usage. An organization maintains LANs at dispersed locations. Non secure IP traffic is conducted on each LAN. For traffic offsite, through some sort of private or public WAN, IPsec protocols are used. These protocols operate in networking devices, such as a router or firewall, which connect each LAN to the outside world. The IPsec networking device will typically encrypt and compress all traffic going into the WAN and decrypt and decompress traffic coming from the WAN; these operations are transparent to workstations and servers on the LAN. Secure transmission is also possible with individual users who dial into the WAN. Such user workstations must implement the IPsec protocols to provide security.



Benefits of IPsec

Some of the benefits of IPsec:

• When IPsec is implemented in a firewall or router, it provides strong security that can be applied to all traffic crossing the perimeter. Traffic within a company or workgroup does not incur the overhead of security-related processing.

• IPsec in a firewall is resistant to bypass if all traffic from the outside must use IP and the firewall is the only means of entrance from the Internet into the organization.

• IPsec is below the transport layer (TCP, UDP) and so is transparent to applications. There is no need to change software on a user or server system when IPsec is implemented in the firewall or router. Even if IPsec is implemented in end systems, upper-layer software, including applications, is not affected.

• IPsec can be transparent to end users. There is no need to train users on security mechanisms, issue keying material on a per-user basis, or revoke keying material when users leave the organization.

• IPsec can provide security for individual users if needed. This is useful for offsite workers and for setting up a secure virtual subnet work within an organization for sensitive applications

IPsec Services

IPsec provides security services at the IP layer by enabling a system to select required security protocols, determine the algorithm(s) to use for the service(s), and put in place any cryptographic keys required to provide the requested services. Two protocols are used to provide security: an authentication protocol designated by the header of the protocol, Authentication Header (AH); and a combined encryption/ authentication protocol designated by the format of the packet for that protocol,

Encapsulating Security Payload (ESP). RFC 4301 lists the following services:

- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets (a form of partial sequence integrity)
- Confidentiality (encryption)
- Limited traffic flow confidentiality

Transport and Tunnel Modes

Both AH and ESP support two modes of use: transport and tunnel mode. The operation of these two modes is best understood in the context of a description of ESP, which is covered in Section 8.3. Here we provide a brief overview.

TRANSPORT MODE Transport mode provides protection primarily for upper-layer protocols. That is, transport mode protection extends to the payload of an IP packet.1 Examples include a TCP or UDP segment or an ICMP packet, all of which operate directly above IP in a host protocol stack. Typically, transport mode is used for end-to-end communication between two hosts (e.g., a client and a server, or two workstations).When a host runs AH or ESP over IPv4, the payload is the data that normally follow the IP header. For IPv6, the payload is the data that normally follow both the IP header and any IPv6 extensions headers that are present, with the possible exception of the destination options header, which may be included in the protection. ESP in transport mode encrypts and optionally authenticates the IP payload but not the IP header. AH in transport mode authenticates the IP payload and selected portions of the IP header.

TUNNEL MODE Tunnel mode provides protection to the entire IP packet. To achieve this, after the AH or ESP fields are added to the IP packet, the entire packet plus security fields is treated as the payload of new outer IP packet with a new outer IP header. The entire original, inner, packet travels through a tunnel from one point of an IP network to another; no routers along the way are able to examine the inner IP header. Because the original packet is encapsulated, the new, larger packet may have totally different source and destination addresses, adding to the security. Tunnel mode is used when one or both ends of a security association (SA) are a security gateway, such as a firewall or router that implements IPsec. With tunnel mode, a number of hosts on networks behind firewalls may engage in secure communications without implementing IPsec. The unprotected packets generated by such hosts are tunneled through external networks by tunnel mode SAs set up by the IPsec software in the firewall or secure router at the boundary of the local network.

	Transport Mode SA	Tunnel Mode SA
AH	Authenticates IP payload and selected portions of IP header and IPv6 extension headers.	Authenticates entire inner IP packet (inner header plus IP payload) plus selected portions of outer IP header and outer IPv6 extension headers.
ESP	Encrypts IP payload and any IPv6 extension headers following the ESP header.	Encrypts entire inner IP packet.
ESP with Authentication	Encrypts IP payload and any IPv6 extension headers following the ESP header. Authenticates IP payload but not IP header.	Encrypts entire inner IP packet. Authenticates inner IP packet.