Syllabus:-Unit-IV

- Transport Layer - Design issues,
- Connection management,
- Session Layer· - Design issues,
- Remote procedure call.
- Presentation Layer - Design issues,
- Data compression techniques,
- Cryptography
- TCP - Window Management.

Possible Conceptual Questions:-

Question No-(1): What are various design issues of the Transport Layer?

Answer: - Design issues of Transport Layer:-

Transport layer is responsible for following issues:-

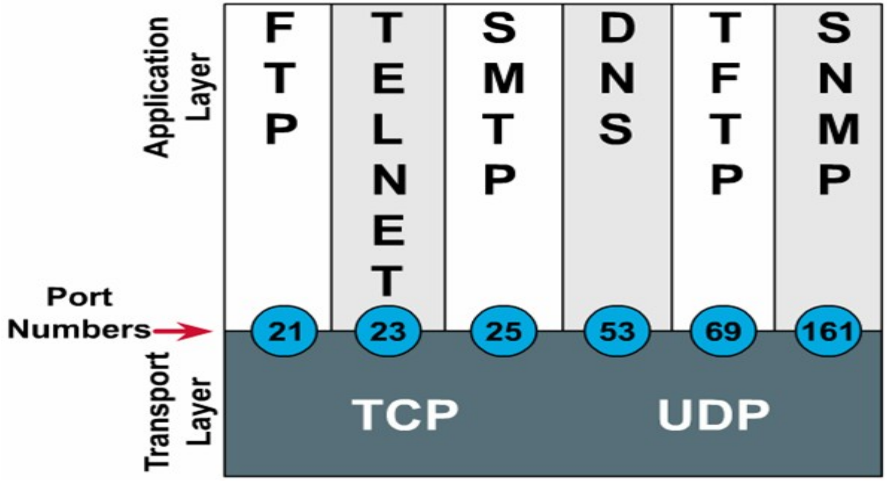
a) Accepting message segments from the application layer and to divide into packets.
b) End-to-End Delivery of the packet
c) Combining packets into message segment at receiver side.
d) Connection management.

In other words transport layer is responsible for two tasks:-

- Transport and regulate the flow of information from source to destination, reliably and accurately.
- The end-to-end control:
  - Sliding windows.
  - Sequencing numbers.
  - Acknowledgments.
  - Segmentation.
  - Multiplexing.

Question No-(2): What are the differences between TCP and UDP? Also discuss various fields in TCP and UDP headers?

Answer:- The TCP/IP protocol model at Layer 4 (transport layer) has two protocols - TCP and UDP.Both TCP and UDP use port (or socket) numbers to pass information to the upper layers.

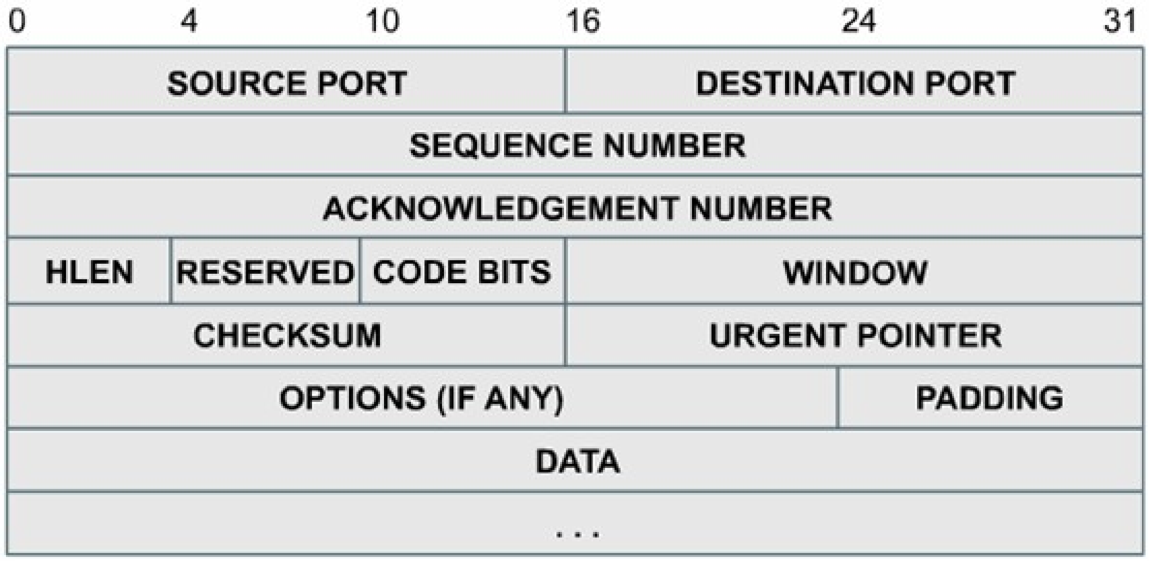


- TCP supplies a virtual circuit between end-user applications.
- These are main characteristics of the TCP:

- Connection-oriented.
- Reliable.
- Divides outgoing messages into segments.
- Reassembles messages at the destination station.
- Re-sends anything not received.
- Reassembles messages from incoming segments.

► In brief TCP is transport layer protocol that provides reliable full-duplex data transmission**.**

TCP-Header:- Following are the fields in TCP-header.



0 4 10 16 24 31

<table>
<tr><td colspan="3">SOURCE PORT</td><td colspan="2">DESTINATION PORT</td></tr>
<tr><td colspan="5">SEQUENCE NUMBER</td></tr>
<tr><td colspan="5">ACKNOWLEDGEMENT NUMBER</td></tr>
<tr><td>HLEN</td><td>RESERVED</td><td>CODE BITS</td><td colspan="2">WINDOW</td></tr>
<tr><td colspan="3">CHECKSUM</td><td colspan="2">URGENT POINTER</td></tr>
<tr><td colspan="4">OPTIONS (IF ANY)</td><td>PADDING</td></tr>
<tr><td colspan="5">DATA</td></tr>
<tr><td colspan="5">. . .</td></tr>
</table>

► **Source Port** 16 bits-This is the port through which was packet was generated at source machine**.**

► **Destination Port** 16 bits**-** This is the port through which the packet is supposed to be received by destination machine**.**

► **Sequence Number:** 32 bits – Each TCP packet is assigned an identification number by transport layer. The sequence number field represents the sequence number of the first data octet in this segment (except when SYN is present).This field is used to ensure correct sequencing of the arriving data

► **Acknowledgment Number:** 32 bits- This field contains the value of the next sequence number the sender of the segment is expecting to receive.

► **HLEN:**-Number of 32-bit words in the header.

► **Checksum:-**Ensure that the data has not been damaged during transmission

► **Urgent Pointer**: Indicates the end of the urgent data

► **Control or Flag Bits:** 8 bits

- ► **ACK:** Acknowledgment field significant
- ► **RST:** Reset the connection
- ► **SYN:** Synchronize sequence numbers
- ► **FIN:** No more data from sender

► **Window:** 16 bits

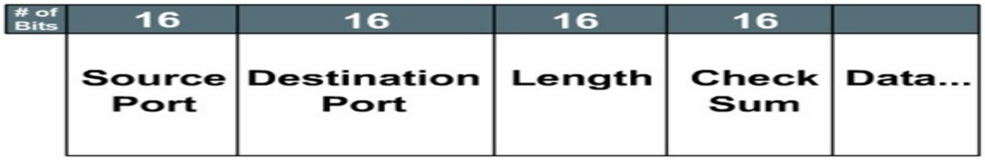
- ► The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept. Number of octets sender is willing to accept

UDP-Protocol: UDP transports data unreliably between hosts.
Following are the characteristics:

- Connectionless.
- Unreliable.
- Transmit messages (called user datagrams).
- Provides no software checking for message delivery (unreliable).
- Does not reassemble incoming messages.
- Uses no acknowledgements**.**

Thus UDP is a simple protocol that exchanges datagrams, without acknowledgments or guaranteed delivery. Applications that do not need to guarantee data delivery use the **faster** UDP. UDP has **no windowing** or **acknowledgments**, so error detection is provided by application layer protocols. UDP is encapsulated within the IP packet. UDP is the transport layer protocol used by **DNS, TFTP, SNMP, and DHCP**.

UDP-Header:- Following are the fields of UDP-Header:-



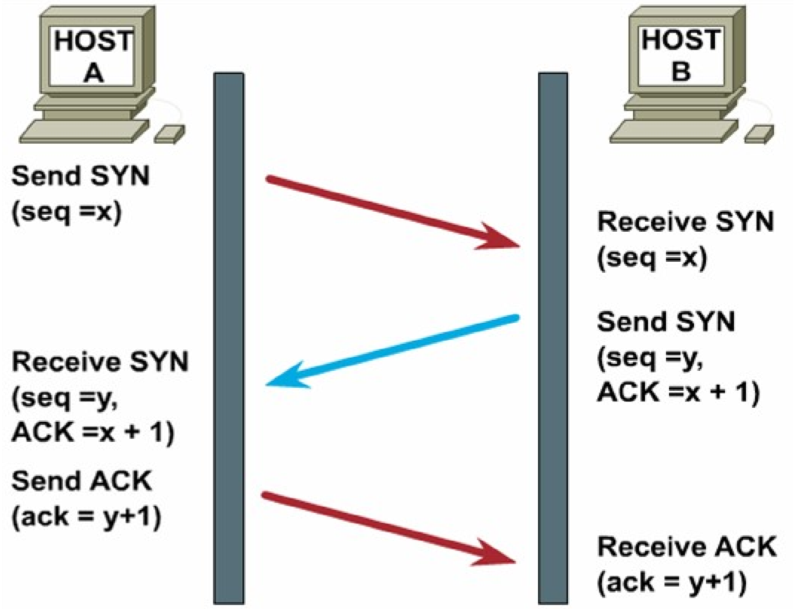
# of Bits	16	16	16	16	
	Source Port	Destination Port	Length	Check Sum	Data...

Source Port: - Optional - used only if reply is expected
Destination Port:-Specifies the application/protocol to which UDP needs to pass the data
Length:-Number of octets in the UDP segment
Checksum:-Ensure that the data has not been damaged during transmission

Question No-(3): What are various steps of connection establishment and connection termination in TCP-connection?

Answer:- Connection Establishment in TCP:- Communicating hosts go through a synchronization process to establish a virtual connection.

This synchronization process ensures that both sides are ready for data transmission and allows the devices to determine the initial sequence numbers.



Sequence numbers are reference numbers between the two devices. The sequence numbers give each host a way to ACK the SYN, so the receiver knows which connection request the sender is responding to.

Sender initiates the process of connection establishment by sending a TCP packet in which SYN flag must be set. Receiver will acknowledge this SYN Packet. After getting the acknowledgement of SYN packet sender will carry on the normal procedure of packet transmission.

Connection Termination:- At the end of the communication, sender sends last packet with FIN flag as set. It indicates that this is the last packet from sender side. Receiver acknowledges the last packet. After it sender sends a packet with RST flag as set. RST flag indicates that the earlier connection has been terminated and reset successfully.

Question No-(4): What are various design issues of session layer?
Answer:- Design Issues of Session Layer:- The session layer is level five of the seven level OSI model. It responds to service requests from the presentation layer and issues service requests to the transport layer.

The session layer provides the mechanism for opening, closing and managing a session between end-user application processes, i.e. a semi-permanent dialogue. Communication sessions consist of requests and responses that occur between applications.

Session layers are commonly used in application environments that make use of remote procedure calls (RPCs).

An example of a session layer protocol X.225 or ISO 8327.

In case of a connection loss this protocol may try to recover the connection. If a connection is not used for a long period, the session layer protocol may close it and re-open it.

It provides for either full duplex or half-duplex operation and provides synchronization points in the stream of exchanged messages.
.

List of Session layer services

- Authentication
- Permissions
- Session restoration (checkpointng and recovery)

Authentication is the act of establishing or confirming something (or someone) as *authentic*, that is, that claims made by or about the thing are true. This might involve confirming the identity of a person, the origins of an artifact, or assuring that a computer program is a trusted one.

Permissions or Access control

One familiar use of authentication and authorization is access control. A computer system supposed to be used only by those authorized must attempt to detect and exclude the unauthorized. Access to it is therefore usually controlled by insisting on an authentication procedure to establish with some established degree of confidence the identity of the user, thence granting those privileges as may be authorized to that identity.

In some cases, ease of access is balanced against the strictness of access checks. For example, the credit card network does not require a personal identification number, and small transactions usually do not even require a signature. The security of the system is maintained by limiting distribution of credit card numbers, and by the threat of punishment for fraud.

Checkpoints

Session layer is responsible for creating several checkpoints, checkpoints are also treated as recovery points i.e. in case of failure the system rollback to its previous checkpoint configuration or action.

Question No-(5): Discuss the procedure of remote procedure call with the help of suitable diagram?

Answer: **Remote Procedure Call (RPC)**: **-** RPC is a powerful technique for constructing distributed, client-server based applications. It is based on extending the notion of conventional, or local procedure calling, so that the called procedure need not exist in the same address space as the calling procedure. The two processes may be on the same system, or they may be on different systems with a network connecting them. By using RPC, programmers of distributed applications avoid the details of the interface with the network. The transport independence of RPC isolates the application from the physical and logical elements of the data communications mechanism and allows the application to use a variety of transports.

There are 3 components on each side in case of RPC System

- a user program (client or server)
- a set of *stub* procedures
- RPC runtime support

Server program defines the server's interface using an *interface definition language* (IDL).The IDL specifies the names, parameters, and types for all client-callable server procedures.

A *stub compiler* reads the IDL and produces two stub procedures for each server procedure: a client-side stub and a server-side stub

The server writer writes the server and links it with the server-side stubs; the client writes her program and links it with the client-side stubs.

The stubs are responsible for managing all details of the remote communication between client and server.

A client-side stub is a procedure that looks to the client as if it were a callable server procedure.

•A server-side stub looks to the server as if it's a calling client.
•The client program thinks it is calling the server; in fact, it's calling the client stub.
•The server program thinks it's called by the client; in fact, it's called by the server stub.
•The stubs send messages to each other to make the RPC happen.

Binding:-

•Binding is the process of connecting the client and server
•The server, when it starts up, *exports* its interface, identifying itself to a network name server and telling the local runtime its dispatcher address.
•The client, before issuing any calls, *imports* the server, which causes the RPC runtime to lookup the server through the name service and contact the requested server to setup a connection.
•The *import* and *export* are explicit calls in the code.

Marshalling:

•Marshalling is the packing of procedure parameters into a message packet.
•The RPC stubs call type-specific procedures to marshall (or unmarshall) all of the parameters to the call.
•On the client side, the client stub marshalls the parameters into the call packet; on the server side the server stub unmarshalls the parameters in order to call the server's procedure.

•On the return, the server stub marshalls return parameters into the return packet; the client stub unmarshalls return parameters and returns to the client.

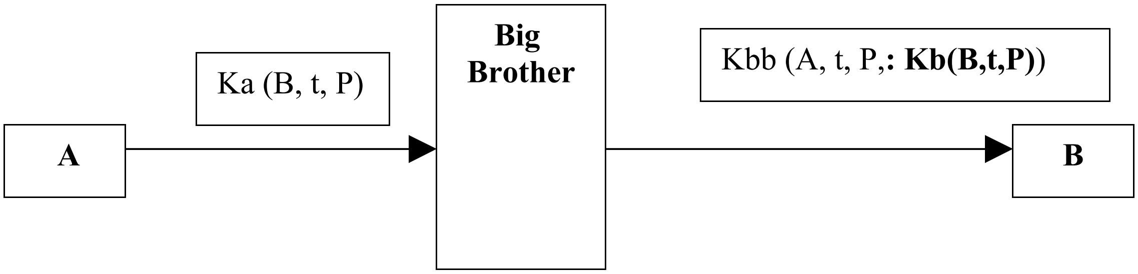
<u>Question No-(6):</u> **What do you understand by Digital Signatures? Explain Symmetric-key signature and public-key signature methods with suitable example?**
<u>Answer</u>: - <u>Digital Signature</u>:- Digital Signature is a mechanism to authenticate the text communicated by one party to other party. In actual physical documents are authorized by signing, but in electronic transmission documents are authorized by digital signature methods.

Following are the three problems that are avoided by the digital signatures:-

- If a text is sent by the sender, later on sender should not able to deny about the transmission.
- Receiver should not able to generate false transmission on behalf of sender.
- Receiver should able to verify that message is coming from right sender.

Following are two methods to implement digital signature mechanism:-

<u>Symmetric-Key Signatures</u>: Symmetric-Key Signature method is based on the concept of Big Brother. In this method it is assumed that all parties faith on one entity that is Big brother. He knows encryption keys of all parties but never shares keys with others.



In this method, sender (say A) sends Message P along with timestamp t, (the time at which it was sent by A). The message is sent to Big brother not to B in fact. The Big brother kept a receipt of it along with the timestamp t then sends this message to B. Big brother also sends a receipt **Kb(B,t,P)** to B along with the message. In case some controversy arises later on, sender may show the receipt issued by the Big brother, receipt issued by the big brother is treated as authenticated.

The main problem with this method is that appropriate and reliable big brother is not available at all. Big brothers may be Govt. Servers basically but All can not faith on one entity completely.

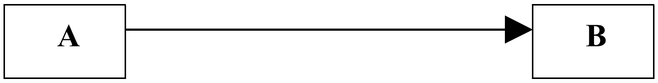
<u>Public-Key Signatures</u>: Public-Key Signature method is based on the concept of public and private keys. In this method each party has two types of keys. One key is public and one key is private. Public keys are known to all whereas private key is known to respective party only.

Two assumptions are made in this method:-

a) Encryption of decrypted text will give original plain text. i.e. Ke(Kd(P)) = P.
b) Decryption of encryption text will give original plain text. i.e. Kd(Ke(P)) = P.

Public Key = Ea
Private Key = Da

Public Key = Eb
Private Key = Db



Sender A will encrypt the message P with B's Public Key and A's Private Key, thus message sent will be Da Eb(P). At Receiver side, B will encrypt the message with B's Private Key and then will encrypt with public key of A. This process will give the original message.

<u>Question No-(7):</u> What do you understand by RSA algorithm? Explain various steps of RSA algorithm with suitable example?

<u>Answer</u>: - <u>RSA Algorithm</u>:- RSA algorithm was developed by **(Rivert, Shameer and adlmen).** This is an important encryption algorithm. The encryption-decryption process is not reversible, that is decryption process is not simply reverse set of actions that of encryption.

Following are the steps of the RSA algorithm:-

1. Select two prime numbers p & q (relatively large).
2. Calculate z = (p-1) **X** (q-1) and n = p **X** q
3. Select a number d that must relatively prime to z.
4. Find out one e such that (e **X** d) % z = 1
5. Now **d** will work as decryption key and **e** as encryption key.
6. To encrypt Plain Text character P into encrypted character C, use formula
   $$C = P^e \% n$$
7. To decrypt encrypted text character C into plain text character P, use formula
   $$P = C^d \% n$$

(Students must also show one example as discussed in the class.)

<u>Question No-(8):</u> What are various compression techniques?

<u>Answer:</u>- <u>Compression</u>:- Compression refers to the reduction of the size. Reduction of the size helps in the transmission of the data in network.
Following are various techniques for compression of the data:-

a) **<u>Lossless Compression:</u>** Lossless compression is the set of compression techniques in which data is compressed in such a way such that no information is lost at the time of decompression. Following are three general techniques under lossless compression.

<u>Character count:</u>- In this method, If same character is being repeated multiple time, instead of repeating the same character special symbol is inserted followed by frequency of continuous repetition. It is assumed that special character consumes less space in comparison to normal characters.

e.g. 333333355555577777AAAAAA3222222
will be compressed as:- #37#56#75#A63#26

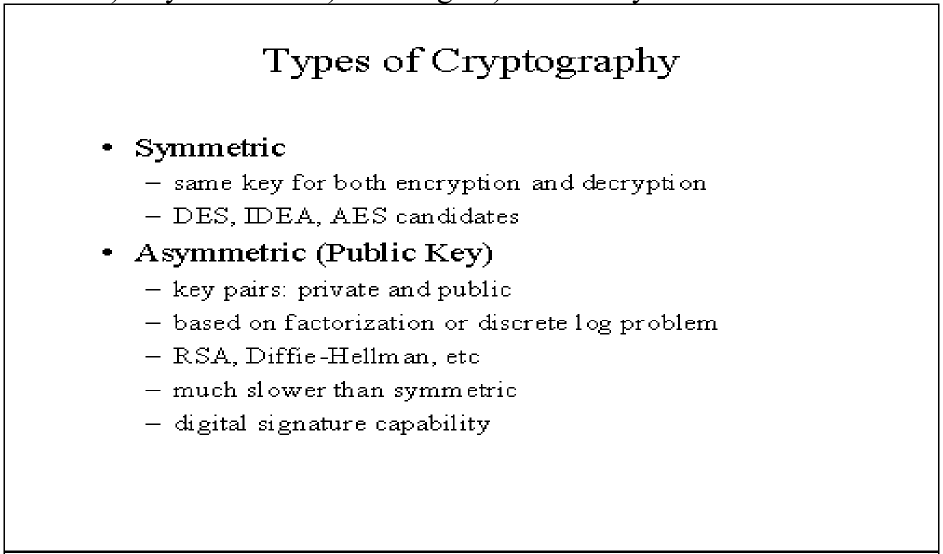
<u>Relative Compression:-</u> In this method repeated words or repeated statements(preferably longer) are stored at one memory address. When same word of statement is repeated than instead of writing the word the memory address in referenced. This approach may also compress the data up to the great extent.

LZW Compression techniques:- In this approaches the coding style is selected based on the frequency of the characters. e.g. if A is coming in the text again and again will be given small code, If Z is coming rarely may be given longer code.

b) **Lossy Compression:-** This is special kind of compression technique used for video and image compression. In this method ignorable information (e.g. corner pixels in video) are lost at the time of compression. Therefore at the time of de-compression exact size is not obtained as it was before compression.
This method drastically can reduce the size, but obviously we need to compromise with some unrelavent information.
In this method graphical objects are re-drawn as 3D-vectors and these vectors are transmitted.

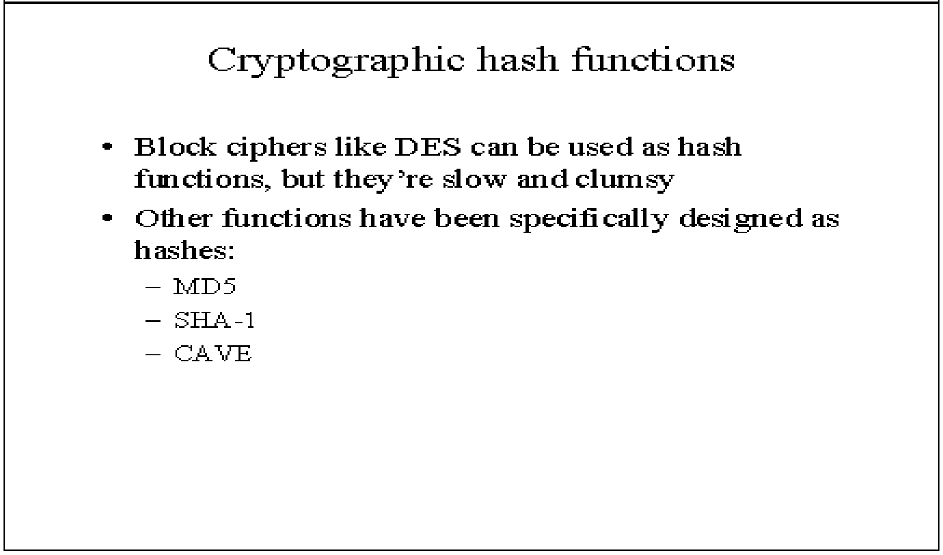
Question No-(9): What are various cryptography techniques?
Answer:- Crytography can be categorited into fours categories basically**:-**

1) Symmetric 2) Asymmetric 3) Hashing 4) Public key



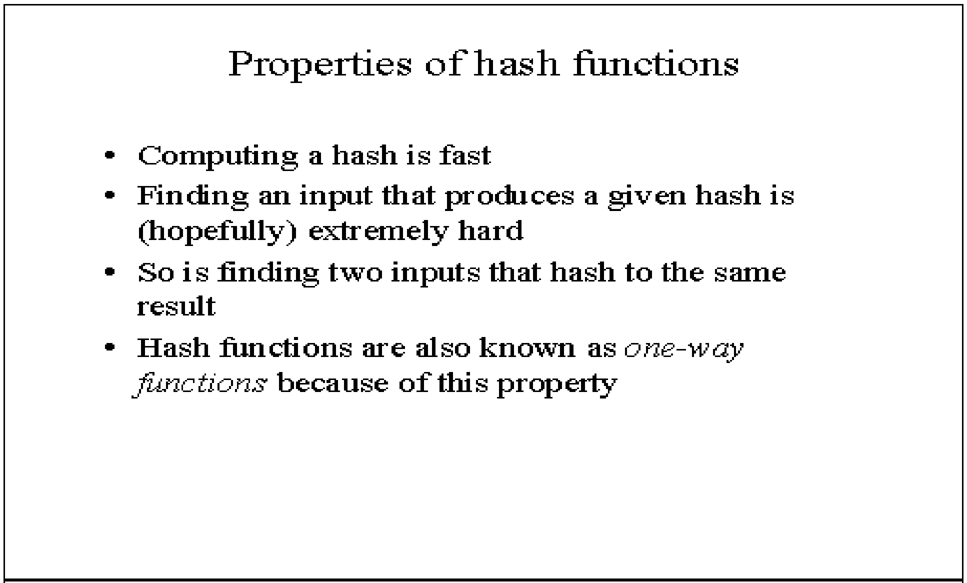
Types of Cryptography

- Symmetric
  - same key for both encryption and decryption
  - DES, IDEA, AES candidates
- Asymmetric (Public Key)
  - key pairs: private and public
  - based on factorization or discrete log problem
  - RSA, Diffie-Hellman, etc
  - much slower than symmetric
  - digital signature capability



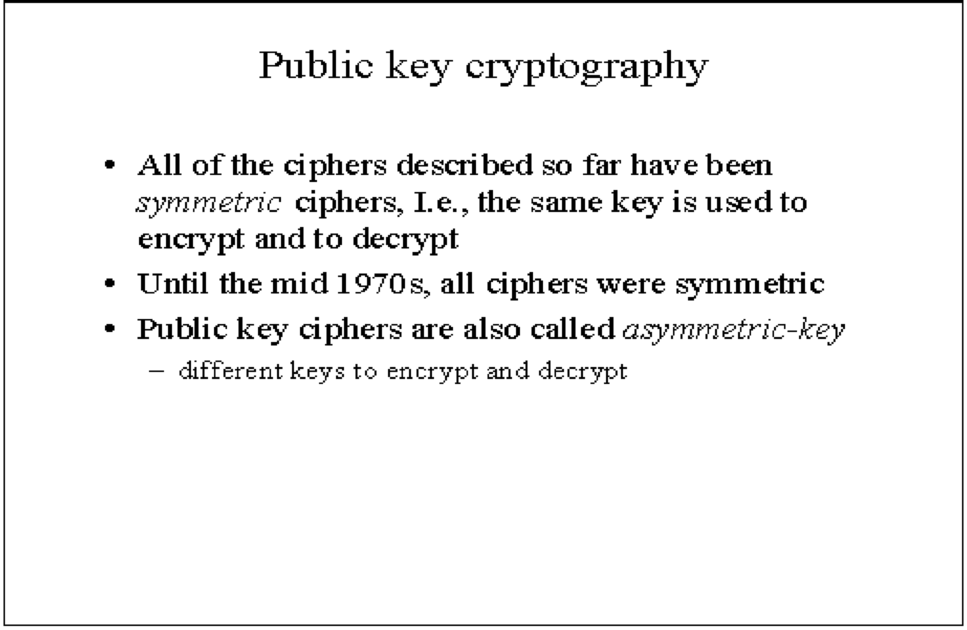
Cryptographic hash functions

- Block ciphers like DES can be used as hash functions, but they're slow and clumsy
- Other functions have been specifically designed as hashes:
  - MD5
  - SHA-1
  - CAVE



Properties of hash functions

- Computing a hash is fast
- Finding an input that produces a given hash is (hopefully) extremely hard
- So is finding two inputs that hash to the same result
- Hash functions are also known as *one-way functions* because of this property



Public key cryptography

- All of the ciphers described so far have been *symmetric* ciphers, I.e., the same key is used to encrypt and to decrypt
- Until the mid 1970s, all ciphers were symmetric
- Public key ciphers are also called *asymmetric-key*
  - different keys to encrypt and decrypt

Following are the general techniques of cryptography.

<u>Transposition cipher</u>

In classical cryptography, a **transposition cipher** changes one character from the plaintext to another (to decrypt the reverse is done). That is, the order of the characters is changed. Mathematically a bijective function is used on the characters' positions to encrypt and an inverse function to decrypt.

<u>Substitution cipher</u>

In cryptography, a **substitution cipher** is a method of encryption by which units of plaintext are substituted with ciphertext according to a regular system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing an inverse substitution.

Substitution ciphers can be compared with transposition ciphers. In a transposition cipher, the units of the plaintext are rearranged in a different and usually quite complex order, but the units themselves are left unchanged. By contrast, in a substitution cipher, the units of the plaintext are retained in the same sequence in the ciphertext, but the units themselves are altered.

The one-time pad

In its most common implementation, the one-time pad can be called a substitution cipher only from an unusual perspective; typically, the plaintext letter is combined (not substituted) in some manner (eg, XOR) with the key material character at that position.

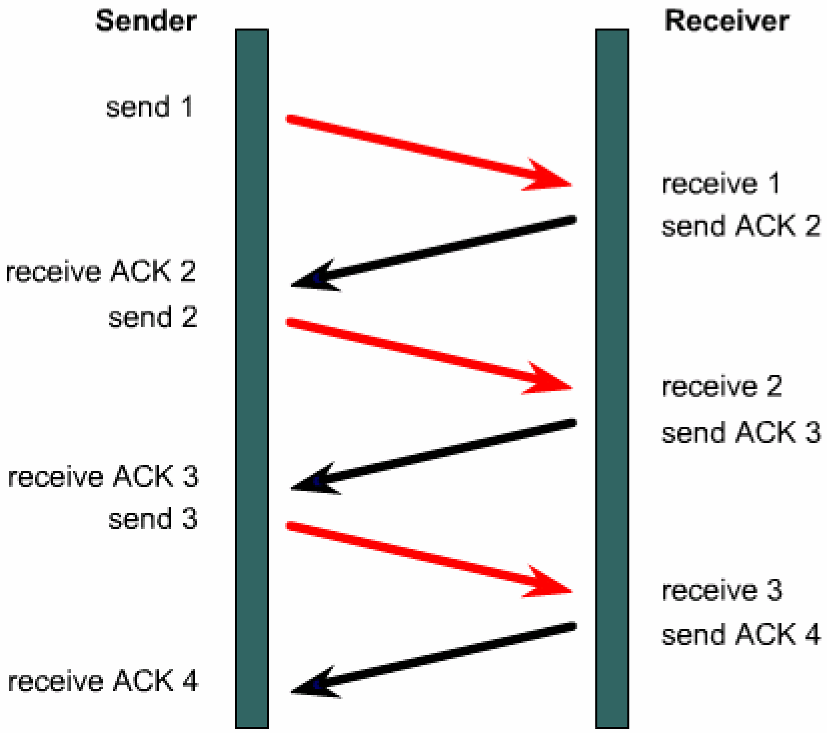
The one-time pad is, in most cases, impractical as it requires that the key material be as long as the plaintext, *actually* random, used once and *only* once, and kept entirely secret from all except the sender and intended receiver. When these conditions are violated, even marginally, the one-time pad is no longer unbreakable.

Question No-(10): Discuss TCP-Window Management System?
Answer:- **Windowing and Window Size:** Window management in TCP is an important concept that ensures reliability in packet delivery as well as reduce the wastage of time in waiting for the acknowledge after each packet.

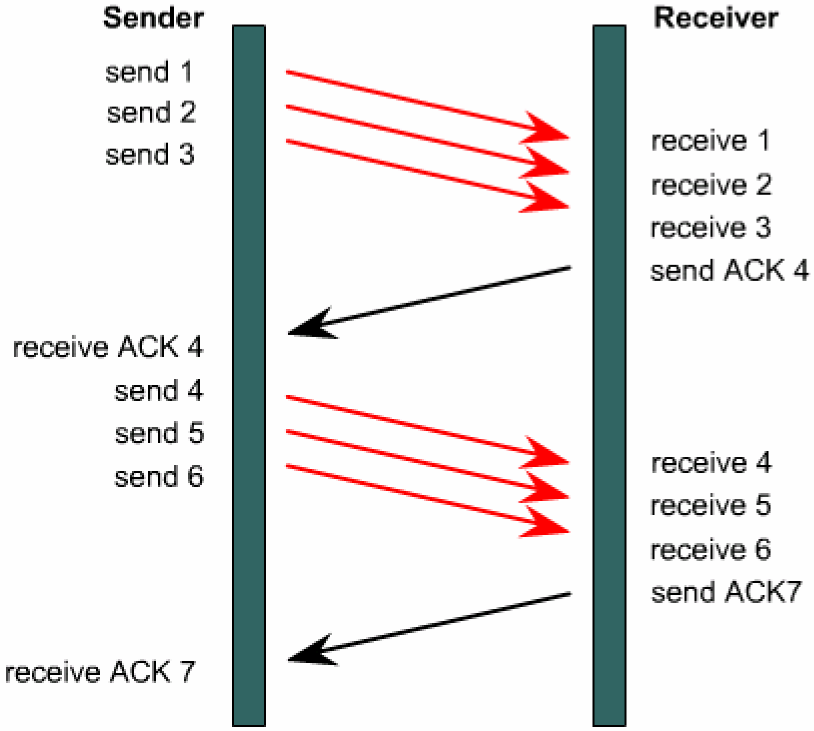
Window size: window size determines the amount of data that you can transmit before receiving an acknowledgment. Sliding window refers to the fact that the window size is negotiated dynamically during the TCP session.

1. Expectational acknowledgment means that the acknowledgment number refers to the octet that is next expected
2. If the source receives no acknowledgment, it knows to retransmit at a slower rate.

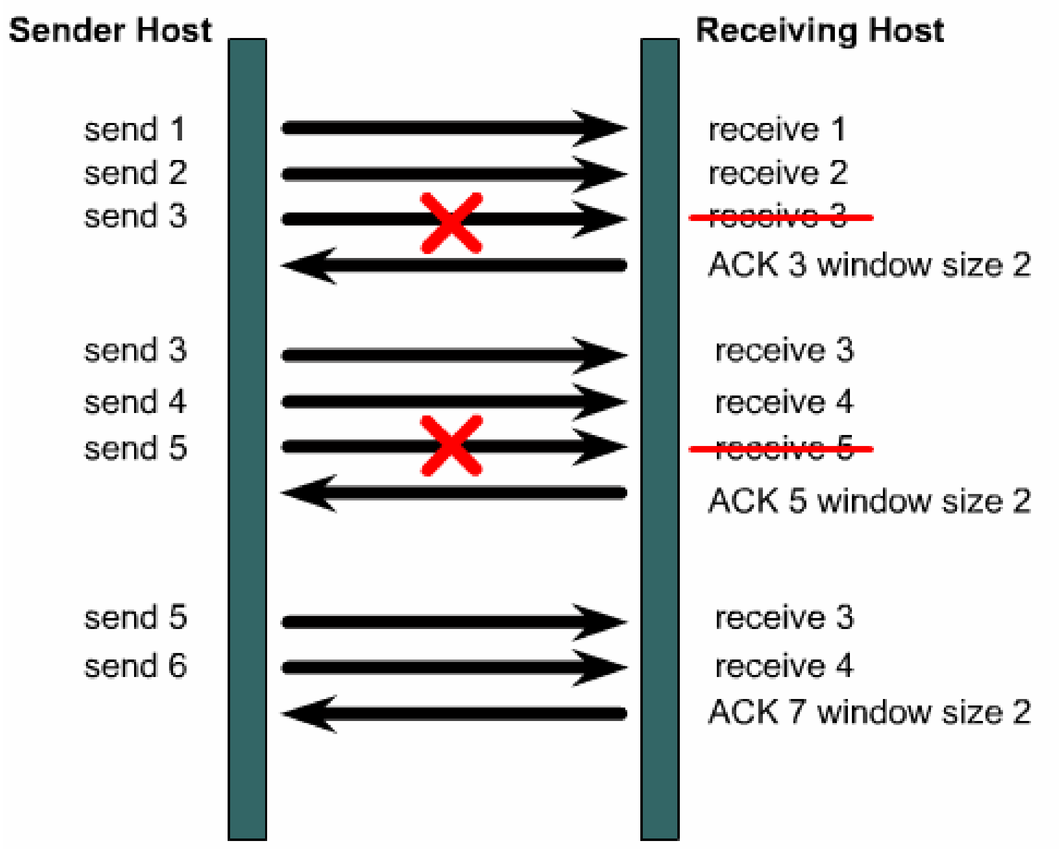
The mechanism of the sliding window style may be understood easily with the help of below given diagrams:



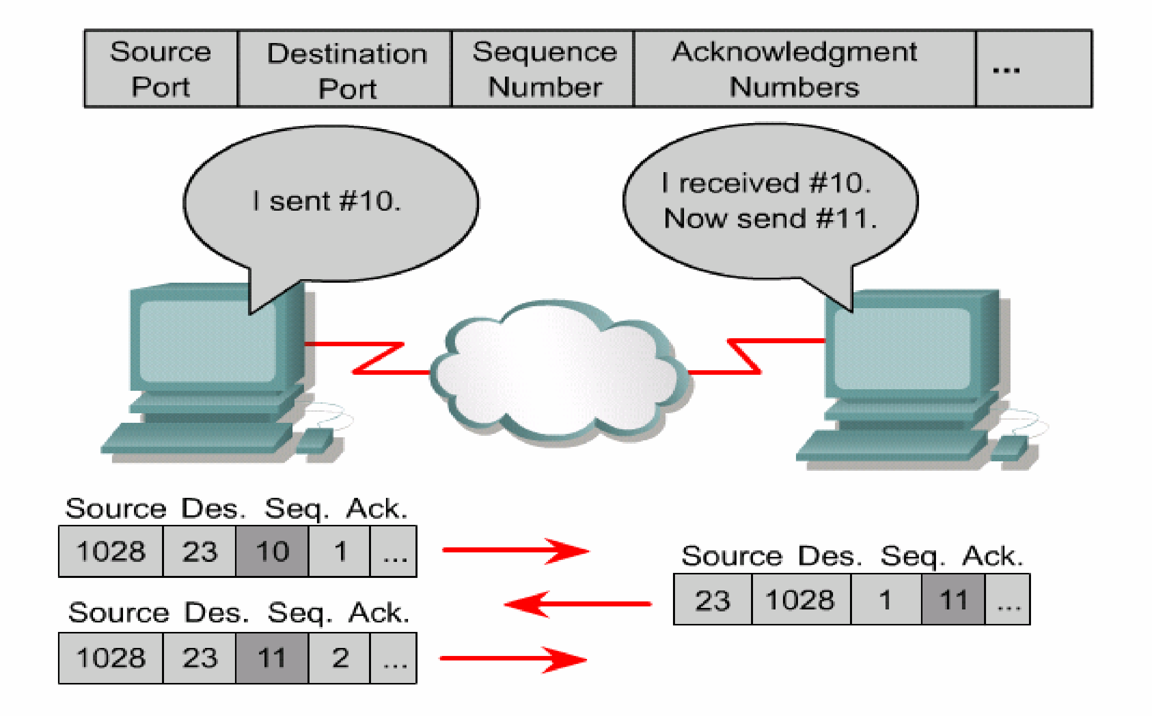
Window size = 1



Window size = 3



TCP Sequence and Acknowledgement



Source Port	Destination Port	Sequence Number	Acknowledgment Numbers	...

Source	Des.	Seq.	Ack.	
1028	23	10	1	...

Source	Des.	Seq.	Ack.	
23	1028	1	11	...

Source	Des.	Seq.	Ack.	
1028	23	11	2	...