Answers

Ques 1: What are the services provided by Transport Layer?

Ans : Services provided by Transport Layer :

a) Process to Process Communication :
   Transport layer offers process to process communication, in this type of communication a message is not only exchanged between two hosts (computers), rather messages are transferred to a particular process residing on a host. It is said to be complete delivery of messages.
b) Addressing : Port Numbers
   To accomplish Process to Process Communication, along with IP Address to identify Local Host and Remote Host, it also needs Port Numbers identify Local Process and Remote Process.

   For Process to Process communication, “Socket Address” that is IP Address plus Port Numbers is required.
c) Encapsulation and Decapsulation :
   Encapsulation occurs at Senders site. When a process has to send a message, it passes the message to transport layer with pair of socket addresses. The transport layer add transport layer header.
   Decapsulation occurs at receiver site. When messages arrive at destination, the header is dropped and transport layer deliver the messages to process with sender socket address in case to send reply to sender.
d) Multiplexing and Demultiplexing :
   When a process or entity receives items from more than one source it is referred as multiplexing, when process or entity delivers items to more than one source it is referred as demultiplexing.
e) Flow Control:
   Transport Layer provides the balance between rate of production and consumption of messages
   It is required in case sender sends items at the rate at which is faster than consumption rate, so consumer has to discard some items.
f) Error Control:

TCP is responsible for:

1. Detect and discard corrupted packets.
2. Keep track of lost and discarded packets and resend them.
3. Recognize duplicate packets and discards them.
4. Buffer out of order packets until the missing packets arrive.

Ques 2: Differentiate between User Datagram Protocol (UDP) and Transmission Control Protocol (TCP)?

Ans:

Basis	Transmission Control Protocol	User Datagram Protocol
Reliability	TCP is connection oriented protocol. When a message or file is send, it ensures its delivery to receiver unless connection fails. If connection lost server will request the lost part.	UDP is connection less protocol. When a message or data is send, it is not ensured that message will reach the receiver end; it may get lost on the way.
Ordered	Message or data will be received in the same order in which sender sends them. TCP is ordered protocol.	There is no specific order in which receiver will receive the messages or data. UDP is Non-Ordered Protocol.
Streaming	TCP is stream oriented protocol. Data is read as a “stream” with nothing distinguishing where one packet ends and other begins. A virtual tube is formed between sender and receiver.	Packets are sent individually, only one packet is there per read call.
Speed of Transfer	TCP has lower speed of transfer due to more interaction between client and server and performing functionalities like error control and flow control.	UDP has higher speed of transfer as there is no interaction between sender and receiver to build a connection, and UDP does not perform the functionalities like error control and flow control.
Error	TCP provides error control and flow	UDP does not provide error

Control and Flow Control	control mechanisms to ensure the delivery.	control and flow control mechanisms.
Header Size	TCP header size is of 20 bytes	UDP header size is of 8 bytes

Ques 3: What is process to process communication and how it is different from host to host communication?

Ans: Process: A process can be defined as an entity or running program of Application layer which uses services of Transport Layer.

Process to Process Communication: Transport layer offers process to process communication, in this type of communication a message is not only exchanged between two hosts (computers), rather messages are transferred to a particular process residing on a host. It is said to be complete delivery of messages.

Basis	Host to Host	Process to Process
Delivery	Complete delivery	Incomplete delivery
Procedure	Messages are transferred to a particular process residing on a host.	Messages are only transferred between hosts, no process handling is there.
Service offered by	Transport Layer	Network Layer
Identities required	Local Host, Local Process, Remote Host, Remote process	Local Host, Remote Host
Addressing	It includes “Socket Address”, that is a IP Address (to identify Local Host and Remote Host) plus Port Number (to identify Local Process and Remote Process).	It includes IP Address to identify Local Host and Remote Host.

Ques 4: What are port numbers? Define its ranges.

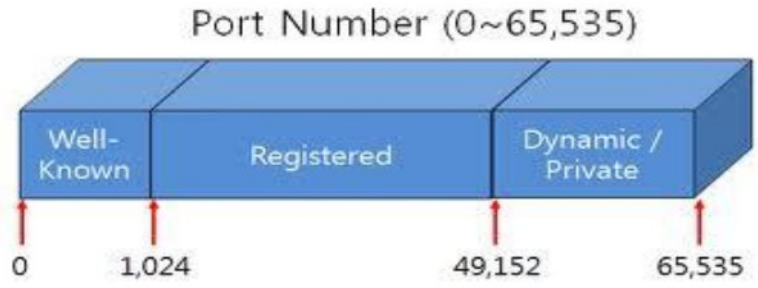
Ans: Port Numbers: In computer networking, port numbers are the part of socket address used to identify “Local Process” and “Remote Process”, to fulfill the “Process to Process Communication”.

IP Address	Port Number

Socket Address

Port Numbers Ranges:Port Numbers lies between the ranges 0-65535.

ICANN (Internet Corporation for Assigned Names and Numbers), has divided the port numbers into three different sub ranges.



a) Wellknown : Port Numbers 0-1023
   Well known port numbers are assigned and controlled by ICANN. These port numbers are generally assigned to Remote Host (Server).
b) Registered : Port Numbers 1024-49151
   Registered port numbers are not assigned or controlled by ICANN. These are registered with ICANN to prevent duplication.
c) Dynamic/Private : Port Numbers 49152-65535
   These port numbers are neither controlled nor registered. These are temporary port numbers. Generally ephemeral (short lived) port numbers to client are assigned from this range.

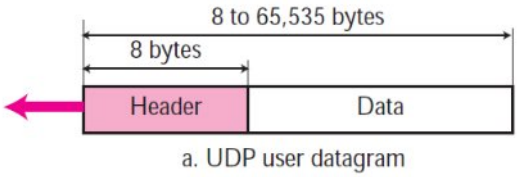
Q5. What is socket address?

Ans.

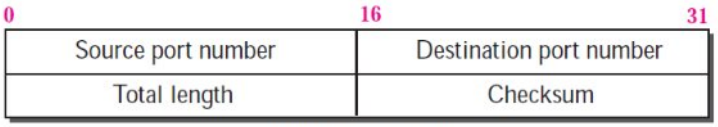
A transport-layer protocol in the TCP suite needs both the IP address and the port number, at each end, to make a connection. The combination of an IP address and a port number is called a **socket address.**

To use the services of transport layer in the Internet, we need a pair of socket addresses:
the client socket address and the server socket address.

Q6. Draw the UDP Datagram and write its uses.
Ans.
A user datagram or UDP packet:-



a. UDP user datagram



b. Header format

Uses/applications of UDP:-

1. UDP is suitable for a process that requires simple request-response communication
   with little concern for flow and error control. It is not usually used for a process such as FTP that needs to send bulk data.
2. UDP is suitable for a process with internal flow and error-control mechanisms. For
   example, the Trivial File Transfer Protocol (TFTP) process
   includes flow and error control. It can easily use UDP.
3. UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software but not in the TCP software.
4. UDP is used for management processes such as SNMP.
5. UDP is used for some route updating protocols such as Routing Information Protocol.
6. UDP is normally used for real-time applications that cannot tolerate uneven delay
   between sections of a received message.

Q7. A TFTP server residing on a host with IP address 130.45.12.7 sends a message to a TFTP client residing on a host with IP address 14.90.90.33. What is the pair of sockets used in this communication?

Ans.

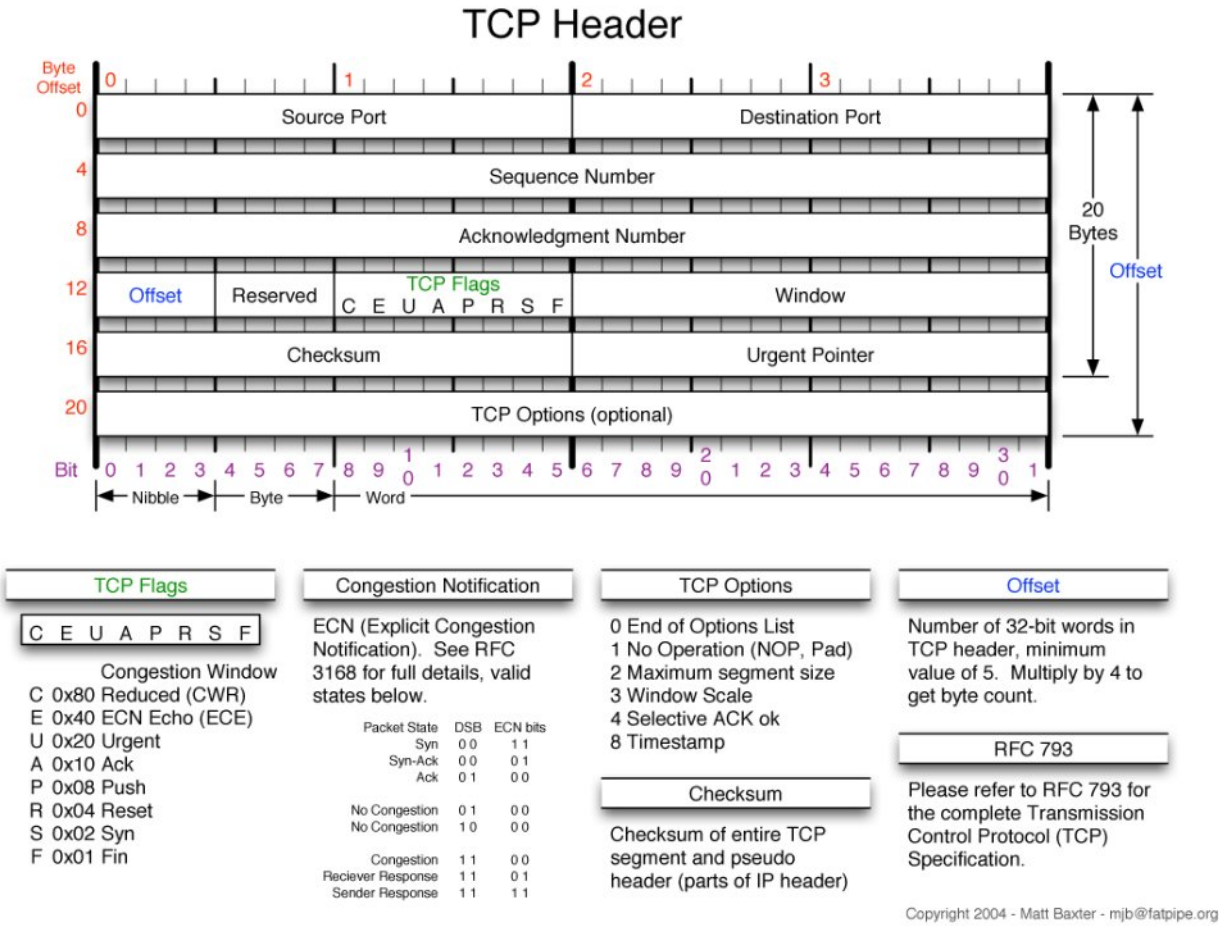
TFTP, Trivial File Transfer Protocol uses the services of UDP on the well-known port 69.
So the pair of sockets used in this communication are:-
Server Socket: 130.45.12.7 – 69
Client Socket: 14.90.90.33 – 69

Q8. What are the services provided by TCP?
Ans.
The Transmission Control Protocol provides the following services:-

1. Process to process communication
   As with UDP, TCP provides process-to-process communication using port numbers.
2. Stream delivery service
   TCP is a stream oriented protocol. It allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes.
   TCP uses buffers for storage at both sender and receiver nodes. It produces segments which are encapsulated in the internet layer into packets.
3. Full Duplex communication
   Data can flow in both directions at the same time.

4. Multiplexing and demultiplexing
   TCP performs multiplexing at the sender and demultiplexing at the receiver.
5. Connection oriented service
   When process at node 1 sends/receive data to/from process at node 2, following happens:
   1. The two TCPs establish a virtual connection between them.
   2. Data are exchanged in both directions.
   3. The connection is terminated.
6. Reliable service
   It uses an acknowledgment mechanism to check the safe and sound arrival of data.

Q9: Draw TCP Segment format and explain its fields?



- **Source port address.** This is a 16-bit field that defines the port number of the application program in the host that is sending the segment. This serves the same purpose as the source port address in the UDP header
- **Destination port address.** This is a 16-bit field that defines the port number of the

application program in the host that is receiving the segment. This serves the same

purpose as the destination port address in the UDP header.

- **Sequence number.** This 32-bit field defines the number assigned to the first byte

of data contained in this segment. As we said before, TCP is a stream transport protocol.

To ensure connectivity, each byte to be transmitted is numbered. The sequence number tells the destination which byte in this sequence is the first byte in the segment. During connection establishment (discussed later) each party uses a random number generator to create an **initial sequence number** (ISN), which is usually different in each direction.

- **Acknowledgment number.** This 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party. If the receiver

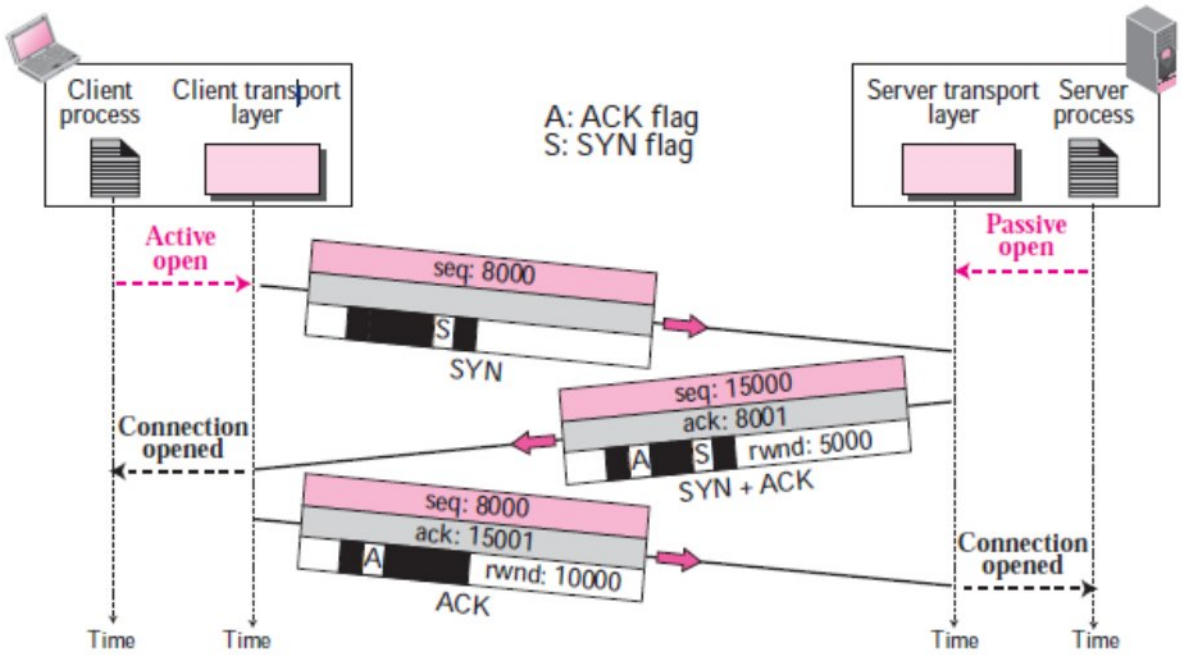
of the segment has successfully received byte number *x* from the other party, it

returns *x* + 1 as the acknowledgment number. Acknowledgment and data can be piggybacked together.

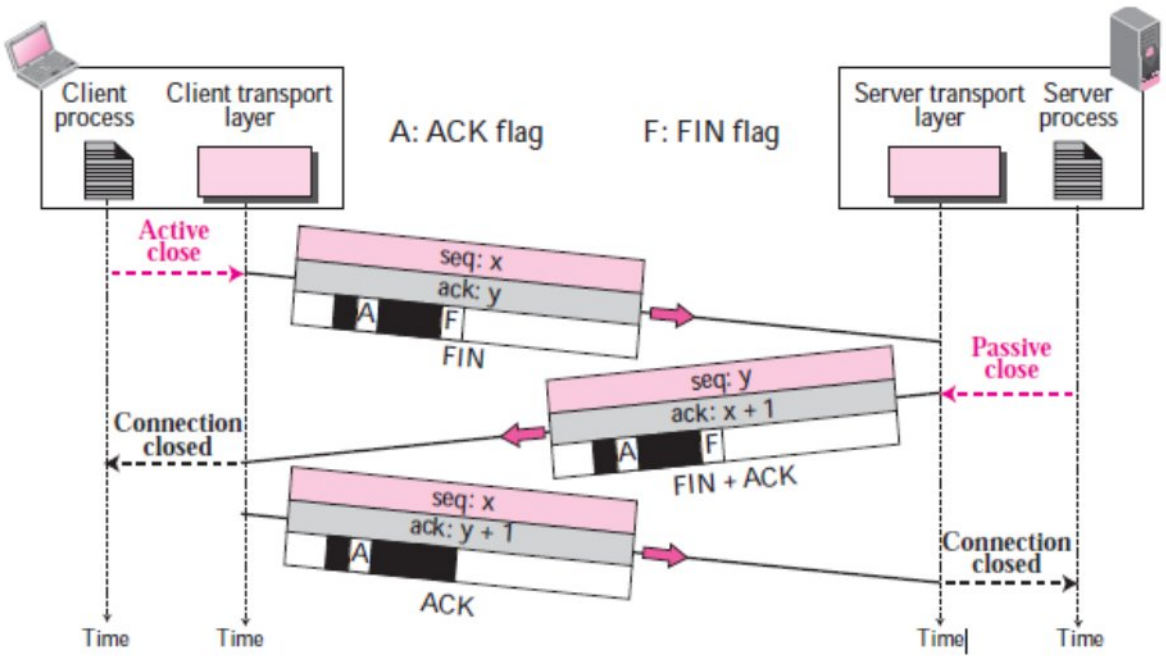
- **Header length.** This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes. Therefore, the value of this field is always between 5 (5× 4= 20) and 15 (15× 4= 60).
- **Reserved.** This is a 6-bit field reserved for future use.
- **Control.** This field defines 6 different control bits or flags. One or more of these bits can be set at a time. These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.
- **Window size.** This field defines the window size of the sending TCP in bytes. Note that the length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes. This value is normally referred to as the receiving window (*rwnd*) and is determined by the receiver. The sender must obey the dictation of the receiver in this case.
- **Checksum.** This 16-bit field contains the checksum. The calculation of the checksum for TCP follows the same procedure as the one described for UDP
- **Urgent pointer.** This 16-bit field, which is valid only if the urgent flag is set, is used when the segment contains urgent data. It defines a value that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.
- **Options.** There can be up to 40 bytes of optional information in the TCP header.

Q10: Draw timing diagram for

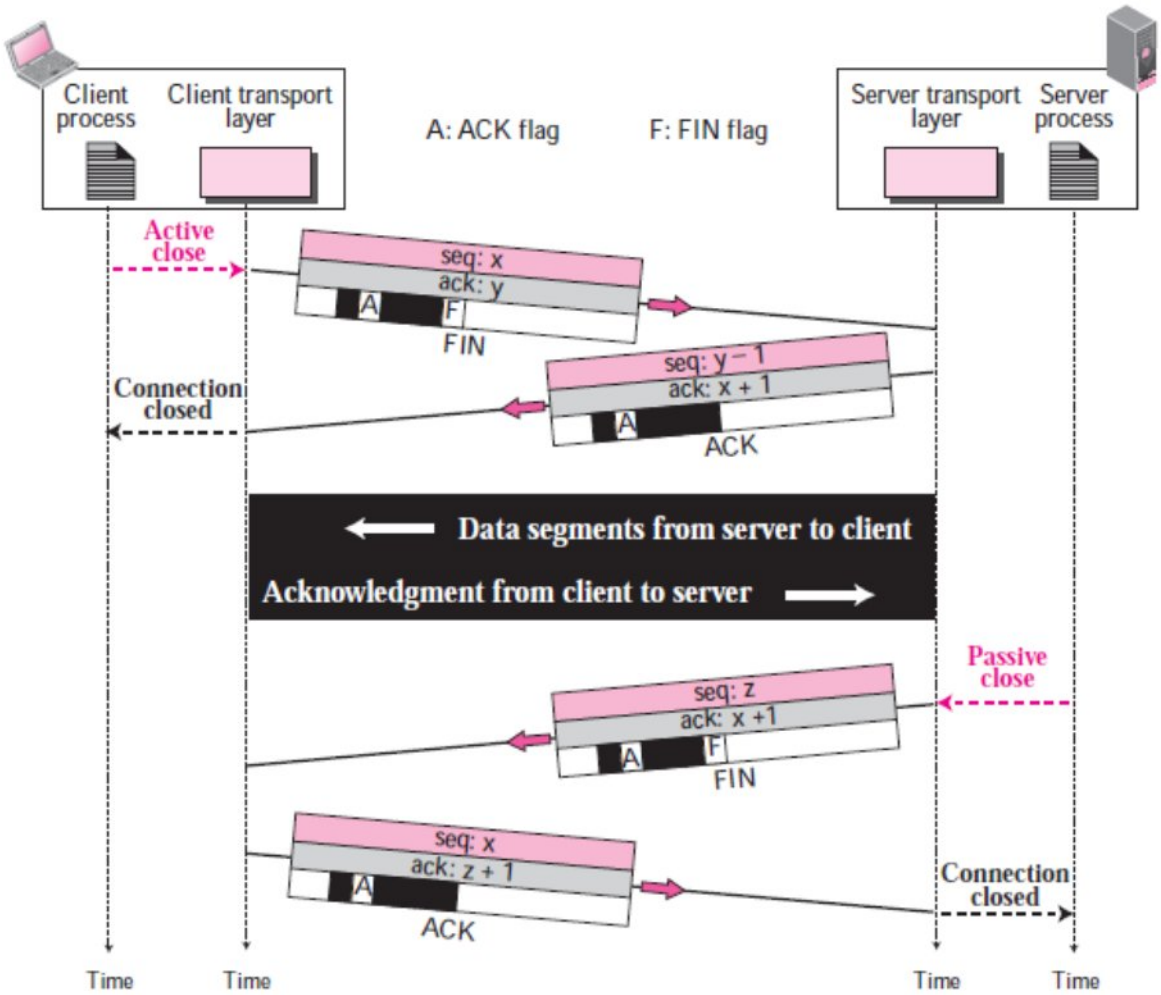
(a) Connection establishment using 3 way handshaking in TCP


Client process
Client transport layer
A: ACK flag
S: SYN flag
Server transport layer
Server process
Active open
Passive open
seq: 8000
S
SYN
seq: 15000
ack: 8001
A
S
rwnd: 5000
SYN + ACK
Connection opened
seq: 8000
ack: 15001
A
rwnd: 10000
ACK
Connection opened
Time
Time
Time
Time

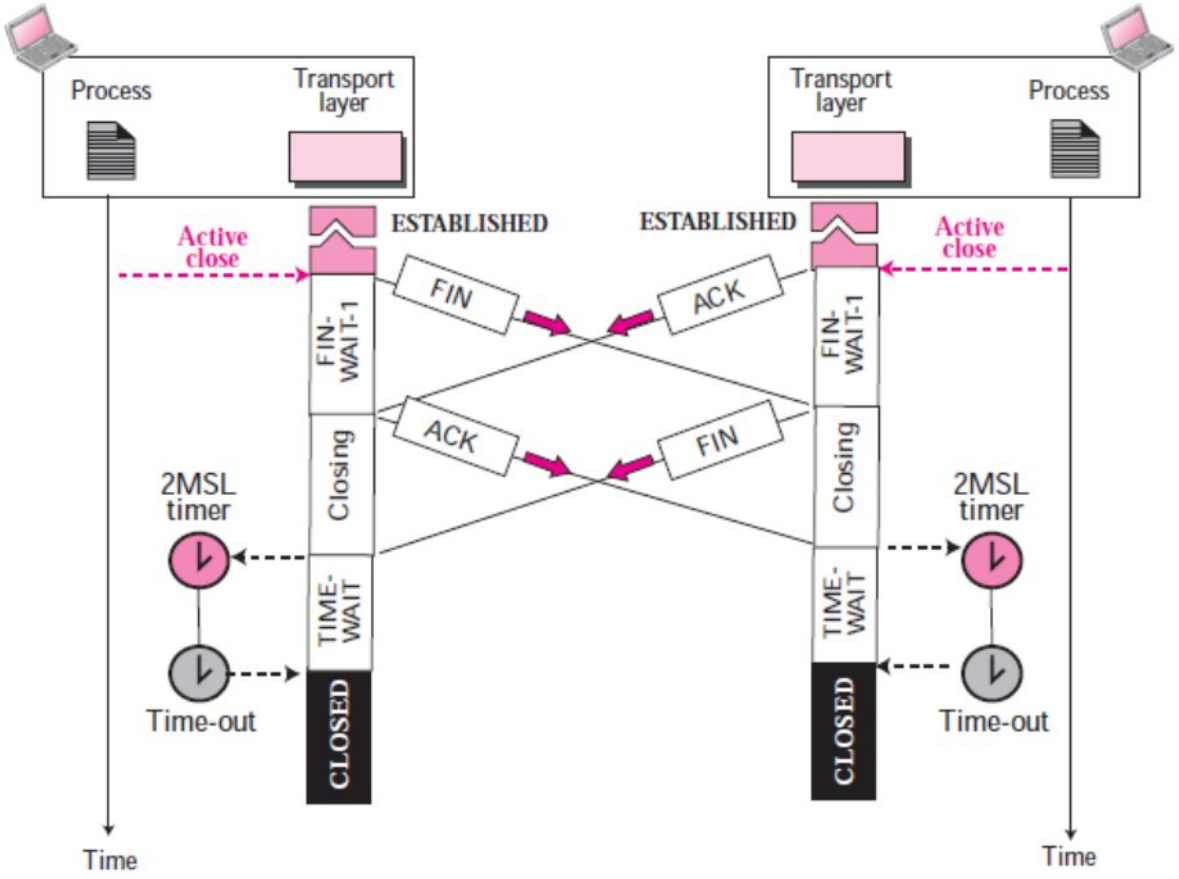
(b) Connection Termination using 3 way handshaking



(c) Half close in TCP

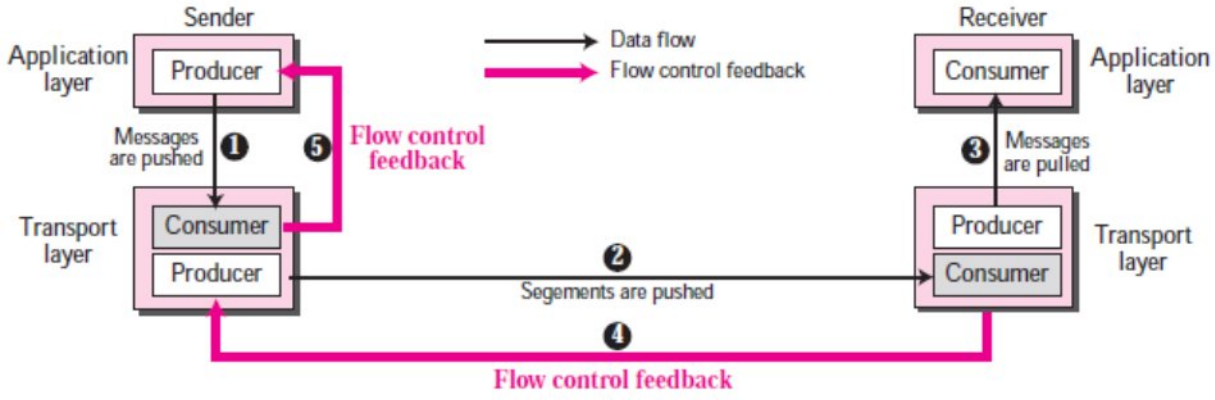


(d) Simultaneous close



Q11: How is flow control regulated in TCP?

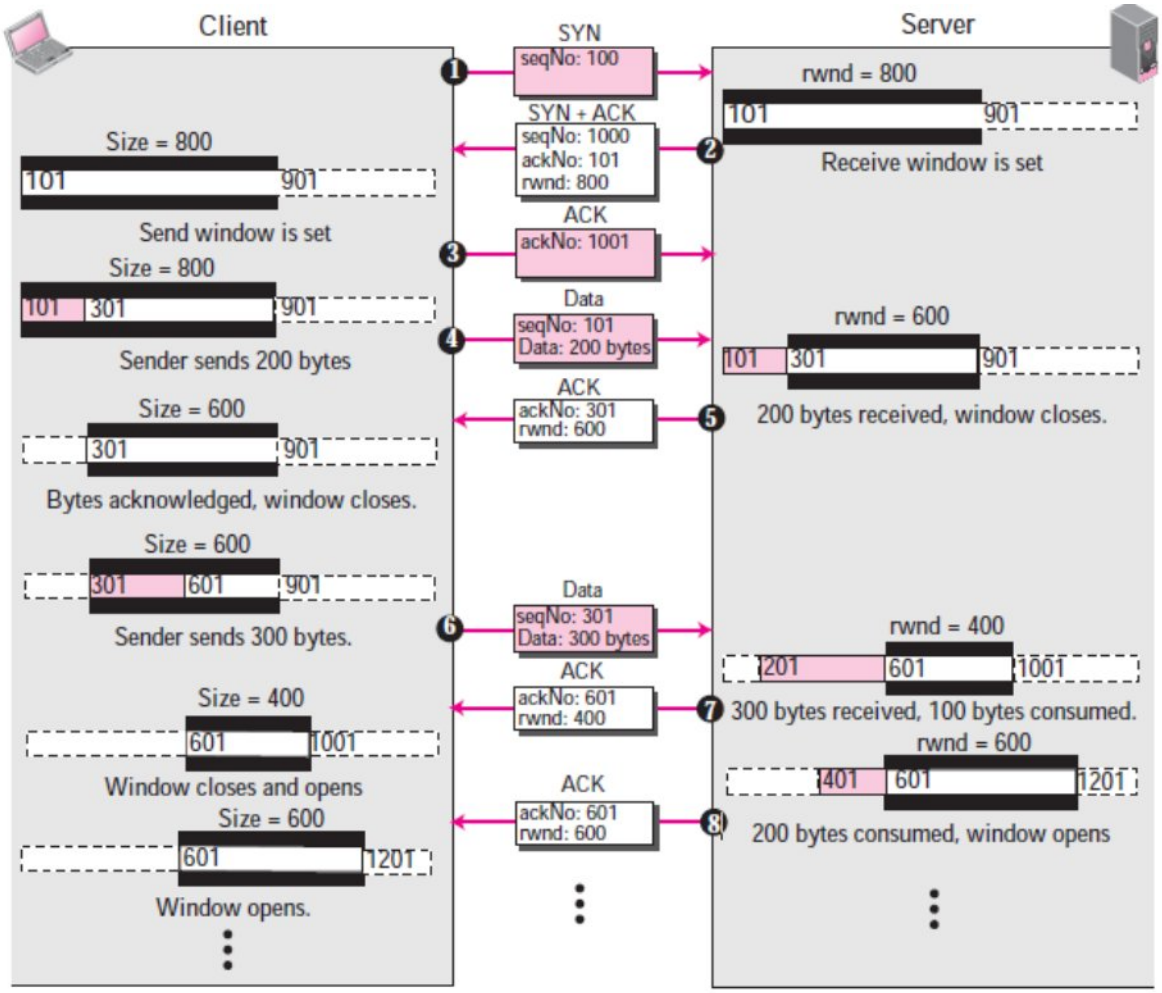
Flow control balances the rate a producer creates data with the rate a consumer can use the data. TCP separates flow control from error control. In this section we discuss flow control, ignoring error control. We temporarily assume that the logical channel between the sending and receiving TCP is error-free.



The figure shows that data travel from the sending process down to the sending TCP, from the sending TCP to the receiving TCP, and from receiving TCP up to the receiving process (paths 1, 2, and 3). Flow control feedbacks, however, are traveling from the receiving TCP to the sending TCP and from the sending TCP up to the sending process (paths 4 and 5). Most

implementations of TCP do not provide flow control feedback from the receiving process to the receiving TCP; they let the receiving process pull data from the receiving TCP whenever it is ready to do so. In other words, the receiving TCP controls the sending TCP; the sending TCP controls the sending process.
Flow control feedback from the sending TCP to the sending process (path 5) is achieved through simple rejection of data by sending TCP when its window is full. This means that our discussion of flow control concentrates on the feedback sent from the receiving TCP to the sending TCP (path 4).

Example of Flow Control



Q12: What is Silly Window syndrome? Explain Nagle's algorithm and Clark's solution?

A serious problem can arise in the sliding window operation when either the sending application program creates data slowly or the receiving application program consumes
data slowly, or both. Any of these situations results in the sending of data in very small
segments, which reduces the efficiency of the operation. For example, if TCP sends

segments containing only 1 byte of data, it means that a 41-byte datagram (20 bytes of
TCP header and 20 bytes of IP header) transfers only 1 byte of user data. Here the overhead
is 41/1, which indicates that we are using the capacity of the network very inefficiently.
The inefficiency is even worse after accounting for the data link layer and physical layer overhead. This problem is called the **silly window syndrome.** For each site, we first describe how the problem is created and then give a proposed solution.

Syndrome Created by the Sender
The sending TCP may create a silly window syndrome if it is serving an application program that creates data slowly, for example, 1 byte at a time. The application program writes 1 byte at a time into the buffer of the sending TCP. If the sending TCP does not have any specific instructions, it may create segments containing 1 byte of data. The result is a lot of 41-byte segments that are traveling through an internet. The solution is to prevent the sending TCP from sending the data byte by byte. The sending TCP must be forced to wait and collect data to send in a larger block. How long should the sending TCP wait? If it waits too long, it may delay the process. If it does not wait long enough, it may end up sending small segments. Nagle found an elegant solution.

Nagle's algorithm is simple:

- The sending TCP sends the first piece of data it receives from the sending application

program even if it is only 1 byte.

- After sending the first segment, the sending TCP accumulates data in the output

buffer and waits until either the receiving TCP sends an acknowledgment or until enough data has accumulated to fill a maximum-size segment. At this time, the sending TCP can send the segment.

- Step 2 is repeated for the rest of the transmission. Segment 3 is sent immediately if

an acknowledgment is received for segment 2, or if enough data have accumulated to fill a maximum-size segment.

The elegance of Nagle's algorithm is in its simplicity and in the fact that it takes into account the speed of the application program that creates the data and the speed of the
network that transports the data. If the application program is faster than the network,
the segments are larger (maximum-size segments). If the application program is slower
than the network, the segments are smaller (less than the maximum segment size).

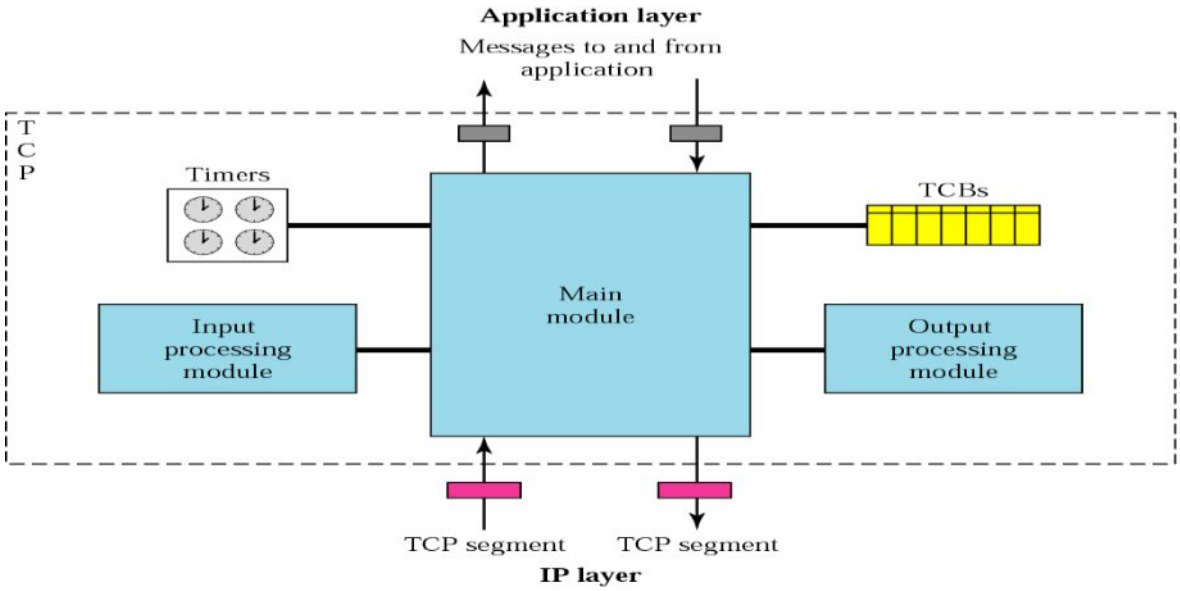
Syndrome Created by the Receiver
The receiving TCP may create a silly window syndrome if it is serving an application program that consumes data slowly, for example, 1 byte at a time. Suppose that the sending application program creates data in blocks of 1 kilobyte, but the receiving application program consumes data 1 byte at a time. Also suppose that the input

buffer of the receiving TCP is 4 kilobytes. The sender sends the first 4 kilobytes of data. The receiver stores it in its buffer. Now its buffer is full. It advertises a window size of zero, which means the sender should stop sending data. The receiving application reads the first byte of data from the input buffer of the receiving TCP. Now there is 1 byte of space in the incoming buffer. The receiving TCP announces a window size of 1 byte, which means that the sending TCP, which is eagerly waiting to send data, takes this
advertisement as good news and sends a segment carrying only 1 byte of data. The procedure will continue. One byte of data is consumed and a segment carrying 1 byte of data is sent. Again we have an efficiency problem and the silly window syndrome. Two solutions have been proposed to prevent the silly window syndrome created by an application program that consumes data slower than they arrive.

Clark's solution is to send an acknowledgment as soon as the data arrive, but to announce a window size of zero until either there is enough space to accommodate a segment of maximum size or until at least half of the receive buffer is empty.

Delayed Acknowledgment The second solution is to delay sending the acknowledgment. This means that when a segment arrives, it is not acknowledged immediately. The receiver waits until there is a decent amount of space in its incoming buffer before acknowledging the arrived segments. The delayed acknowledgment prevents the sending TCP from sliding its window. After the sending TCP has sent the data in the window, it stops. This kills the syndrome. Delayed acknowledgment also has another advantage: it reduces traffic. The receiver does not have to acknowledge each segment. However, there also is a disadvantage in that the delayed acknowledgment may result in the sender unnecessarily retransmitting the unacknowledged segments. TCP balances the advantages and disadvantages. It now defines that the acknowledgment should not be delayed by more than 500 ms.

Q17 . TCP package
Ans.



*** TCBs:**
Transmission Control Blocks

Q 18. Compare IPV4 and IPv6.
Ans.

IPv4	IPv6
Addresses are 32 bits (4 bytes) in length.	Addresses are 128 bits (16 bytes) in length
Address (A) resource records in DNS to map host names to IPv4 addresses.	Address (AAAA) resource records in DNS to map host names to IPv6 addresses.
Pointer (PTR) resource records in the IN-ADDR.ARPA DNS domain to map IPv4 addresses to host names.	Pointer (PTR) resource records in the IP6.ARPA DNS domain to map IPv6 addresses to host names.
IPSec is optional and should be supported externally	IPSec support is not optional
Header does not identify packet flow for QoS handling by routers	Header contains Flow Label field, which Identifies packet flow for QoS handling by router.
Both routers and the sending host fragment packets.	Routers do not support packet fragmentation. Sending host fragments packets

Header includes a checksum.	Header does not include a checksum.
Header includes options.	Optional data is supported as extension headers.
ARP uses broadcast ARP request to resolve IP to MAC/Hardware address.	Multicast Neighbor Solicitation messages resolve IP addresses to MAC addresses.
Internet Group Management Protocol (IGMP) manages membership in local subnet groups.	Multicast Listener Discovery (MLD) messages manage membership in local subnet groups.
Broadcast addresses are used to send traffic to all nodes on a subnet.	IPv6 uses a link-local scope all-nodes multicast address.
Configured either manually or through DHCP.	Does not require manual configuration or DHCP.
Must support a 576-byte packet size (possibly fragmented).	Must support a 1280-byte packet size (without fragmentation).

Feature improvements of ipv6 over ipv4:

-Client-side IP address assignment, no need for DHCP
One interesting feature of IPv6 is that addresses can be assigned automatically and dynamically by the client device, meaning that there is no need for a DHCP server like there is with IPv4.

This means that you can plug a laptop to your local network, it will get an address prefix from any router it finds and then generate an IP for itself for that network, based on the hardware MAC address.

In practice, this means simpler routers and less overhead for managing IP attribution. It will also mean even less configuration requirements. Of course, you can still assign addresses to devices via DHCPv6.

-Multiple network addresses assigned to the same device
A second interesting feature is the ability to assign two or more addresses to the same device. You can thus stay connected to several networks at the same time, which is a great boon for flexibility. Apps will be able to choose the network they need or have access to and you won't have to choose between networks anymore.

-Encryption and IPsec built in
IPv6 also comes with greater security since it has an encryption option built in. This means that the communications will be secured at the lowest level possible, via IPsec, each packet that gets sent has to be decrypted for it to be interpreted.

-No more checksums, better performance
Also on the technical side, IPv6 gets rid of checksum verification. Since everything or almost everything sent via IPv6 networks has its own error-control mechanism, there was no need for one at the IP-level. This clears the overhead added by the need to do checksums at every step, leading to a more responsive and faster connection. There are plenty of other improvements and optimizations that should, in the end, lead to more efficient and faster communications. Still, there's a long way before most of the internet traffic is carried over IPv6.

Q19. Short note on SCTP
Ans. The Stream Control Transmission Protocol (SCTP) is designed to transport PSTN signalling messages over IP networks, but is capable of broader applications. SCTP is an application-level datagram transfer protocol operating on top of an unreliable datagram service such as UDP. It offers the following services:

- Acknowledged error-free non-duplicated transfer of user data.
- Application-level segmentation to conform to discovered MTU size.
- Sequenced delivery of user datagrams within multiple streams, with an option for order-of-arrival delivery of individual datagrams.
- Optional multiplexing of user datagrams into SCTP datagrams, subject to MTU size restrictions.
- Enhanced reliability through support of multi-homing at either or both ends of the association.

The design of SCTP includes appropriate congestion avoidance behaviour and resistance to flooding and masquerade attacks. The SCTP datagram is comprised of a common header and chunks. The chunks contain either control information or user data.The following is the format of the SCTP header.

Source Port Number	Destination Port Number
Verification Tag	
Adler 32 Checksum	

Source Port Number
This is the SCTP sender's port number. It can be used by the receiver, in combination with the source IP Address, to identify the association to which this datagram belongs.

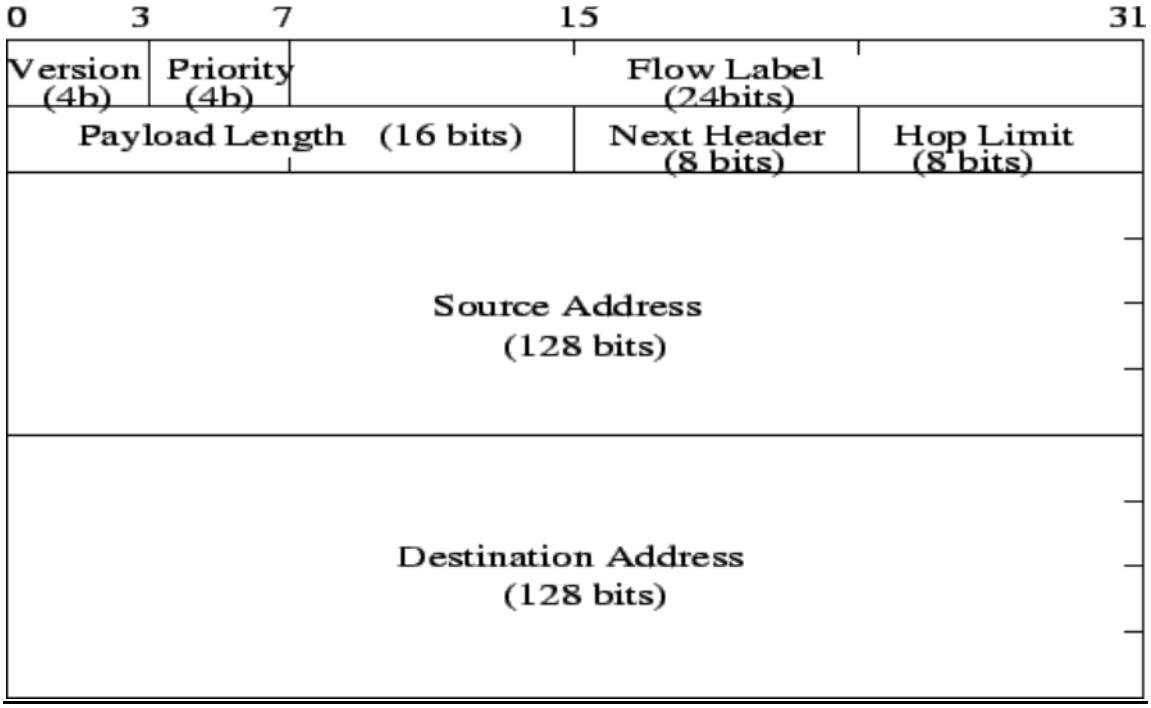
Destination Port Number
This is the SCTP port number to which this datagram is destined. The receiving host will use this port number to de-multiplex the SCTP datagram to the correct receiving endpoint/application.

Verification Tag
The receiver of this 32 bit datagram uses the Verification tag to identify the association. On transmit, the value of this Verification tag must be set to the value of the Initiate tag received from the peer endpoint during the association initialization.

Adler 32 Checksum

This field contains an Adler-32 checksum on this SCTP datagram.

Q20. Header format of IPV6



Fields defined in the IPv6 header are: [1]
_ *Version*. This 4-bit field identifies the IP version number. For IPv6, it is 6.
_ *Traffic class*. This 8-bit field is similar in spirit to the ToS field in IPv4.
_ *Flow label*. This 20-bit field is used to identify a "flow" of datagrams.
_ *Payload length.* This 16-bit value is treated as an unsigned integer giving the number of bytes in the IPv6 datagram following the 40-byte packet header.
_ *Next header*. 8-bit selector. Identifies the type of header immediately following the IPv6 header. Uses the same values as the IPv4 Protocol field.
_ *Hop limit*. 8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.
_ *Source Address*. 128-bit address of the originator of the packet.
_ *Destination Address*. 128-bit address of the intended recipient of the packet.

Several fields appearing in the IPv4 header are no longer present in the IPv6 header:
_ *Fragmentation/Reassembly*. IPv6 does not allow for fragmentation and reassembly at intermediate routers; these operations can be performed only by the source and destination. Fragmentation and reassembly is a time-consuming operation; removing this functionality from the routers and placing it squarely in the end systems considerably speeds up IP forwarding within the network.
_ *Checksum*. Because the transport layer and data link protocols in the Internet layers perform checksumming, the designers of IPv6 felt that this functionality was sufficiently redundant in the network layer that it could be removed. Furthermore, since the IPv4 header contains a TTL field (similar to the hop limit field in IPv6), the IPv4 header checksum needed to be recomputed at every router. As with fragmentation and reassembly, this too was a costly operation in IPv4.
_ *Options*. An options field is no longer a part of the standard IP header. However, it has not gone away. Instead, the options field is one of the possible "next headers"

pointed to from within the IPv6 header. That is, just as TCP or UDP protocol headers can be the next header within an IP packet, so too can an options field. The removal of the options field results in a fixed length, 40-byte IP header.

IPv6 Extensions

IPv6 includes an improved option mechanism over IPv4. IPv6 options are placed in separate extension headers that are located between the IPv6 header and the transport-layer header in a packet. Most IPv6 extension headers are not examined or processed by any router along a packet's delivery path until it arrives at its final destination. This facilitates a major improvement in router performance for packets containing options.

The other improvement is that unlike IPv4 options, IPv6 extension headers can be of arbitrary length and the total amount of options carried in a packet is not limited to 40 bytes. This feature plus the manner in which they are processed, permits IPv6 options to be used for functions that were not practical in IPv4.

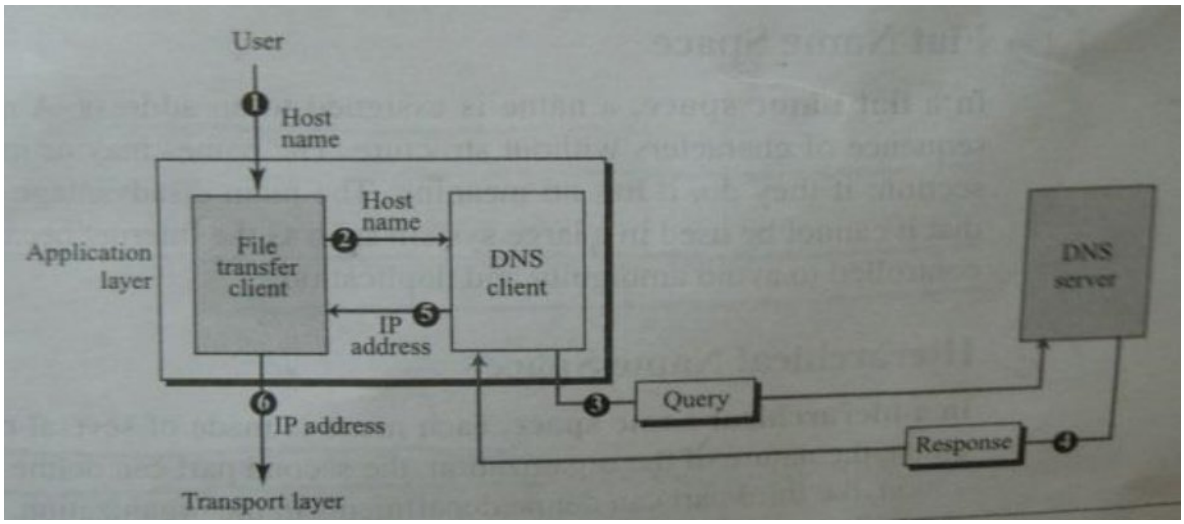
In order to improve the performance when handling subsequent option headers and the transport protocol which follows, IPv6 options are always an integer multiple of 8 octets long, in order to retain this alignment for subsequent headers.

The IPv6 extension headers that are currently defined are: [3]

_ *Routing*: Extended Routing (like IPv4 loose source route).
_ *Fragmentation*: Fragmentation and Reassembly.
_ *Authentication*: Integrity and Authentication. Security
_ *Encapsulation*: Confidentiality.
_ *Hop-by-Hop Options*: Special options which require hop-by-hop processing.
_ *Destination Options*: Optional information to be examined by the destination node.

Q 21 Write a short note on Domain Name System.

- DNS is a client server application program used to help other application programs. DNS is used to map a host name (application layer) to an IP address (network layer).In short it is a naming system for the internet.
- To identify an entity, the Internet uses the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name.
- When the Internet was small, mapping was done using a host file. The host file had only two columns: one for the name and one for the address. When a program or user wanted to map a name to an address, the host consulted the host file and found the mapping.
- Today, it is impossible to have one single host file relate every address to a name, and vice versa. The host file would be too large to store in every host.
- The concept used is to divide this huge amount of information into smaller parts and store each part on a different computer. In this method, the host that needs mapping can contact the closest computer holding the needed information. This method is used by the Domain Name System (DNS)



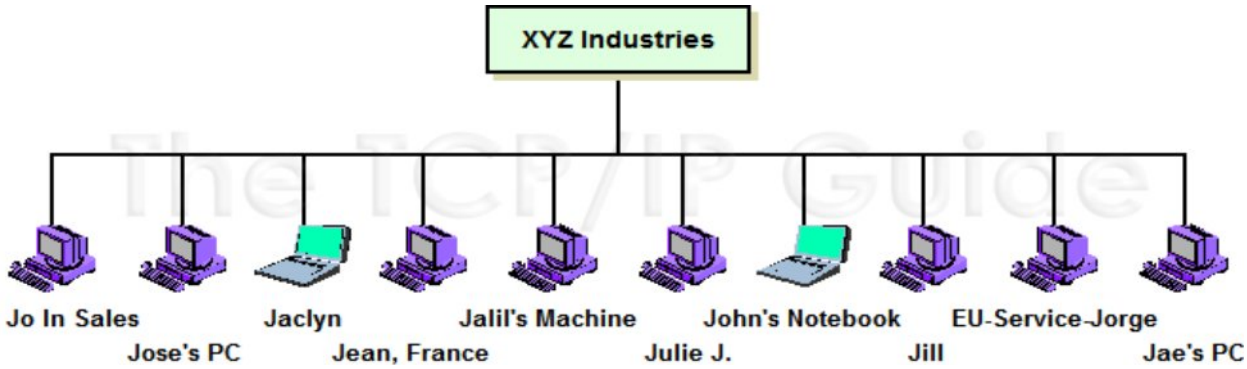
Q22. What is name space? Differentiate and explain flat namespace and hierarchial namespace

Ans 22:

Namespace for networks allow computers to continue to use simple, efficient numeric addresses, while letting humans specify easier-to-remember names that identify them.

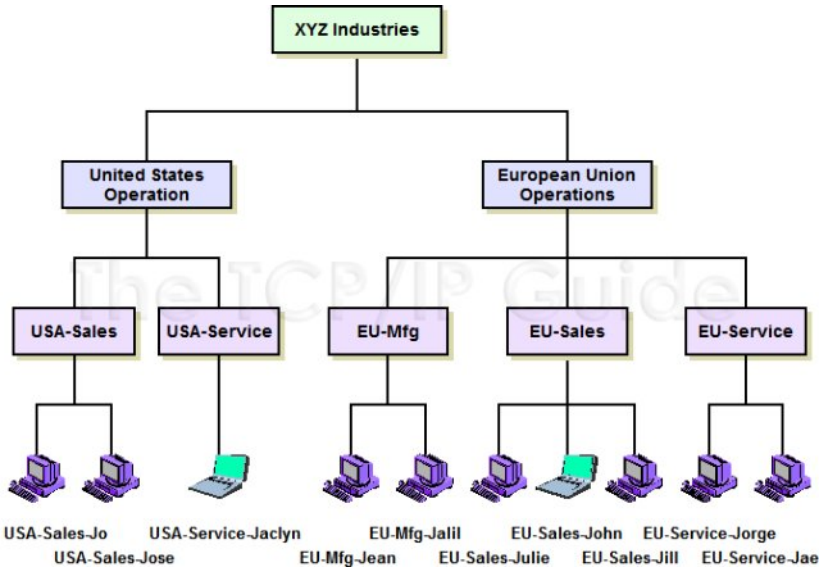
Flat Namespace:-

- A flat name space requires a central authority of some sort to assign names to all devices in the system to ensure uniqueness.
- Flat name spaces have the advantage of simplicity and the ability to create short and easily remembered names.
- They do not scale well to name systems containing hundreds or thousands of machines, due to the difficulties in ensuring each name is unique.
- Another issue is the overhead needed to centrally manage these names.



Hierarchical Namespace:-

- Hierarchical name architecture is ideally suited to a more distributed registration scheme that allows many authorities to share in the registration and administrative process.
- Hierarchical name spaces are more sophisticated and flexible, because they allow names to be assigned using a logical structure.
- We can name our machines using a hierarchy that reflects our organization's structure, for example, and give authority to different parts of the organization to manage parts of the name space. As long as each department is named uniquely and that unique department name is part of each machine name, we don't need to worry about each assigned name being unique across the entire organization, only within the department.
- The price of this flexibility is the need for longer names and more complexity in name registration and resolution.



	FLAT NAME SPACE	HEIRARCHICAL NAME SPACE
INTRO	• In flat name space a name is a sequence of characters without structure. • Names may/may not have a common section ;if they do it has no meaning	• In this type each name is made up of several parts • Names have common section ; meaningful ..can specify nature, name departments of the organisation
CONTROL	Centralized control for assigning names	Decentralized control.
ADVANTAGE	• Avoid ambiguity Duplication • Generate short easily remembered names	• Names are unique without need of any central authority • Flexible • Can be used for large

		systems
DISADVANTAGE	Cannot be used for large system such as INTERNET	Generate longer names comparatively

Q 23. Explain FQDN and PQDN with examples

Ans23:FQDN

A fully qualified domain name (FQDN) is the complete domain name for a specific computer, or host, on the Internet. The FQDN consists of two parts: the hostname and the domain name. For example, an FQDN for a hypothetical mail server might bemymail.somecollege.edu. The hostname is mymail, and the host is located within the domainsomecollege.edu.

PQDN

If a label is not terminated by a null string, it is called a partially qualified domain name (PQDN). A PQDN starts from a node, but it does not reach the root. It is used when the name to be resolved belongs to the same site as the client. Here the resolver can supply the missing part, called suffix, to create an FQDN.For example, suppose you are in charge of the computer science department at the University of Widgetopia. The domain name for the department as a whole is “cs.widgetopia.edu.” and the individual hosts you manage are named after fruit. In the DNS files you maintain you could refer to each device by its FQDN every time; for example, “apple.cs.widgetopia.edu.”, “banana.cs.widgetopia.edu.” and so on. But it's easier to tell the software “if you see a name that is not fully qualified, assume it is in the ‘cs.widgetopia.edu’ domain”. Then you can just call the machines “apple”, “banana”, etc. Whenever the DNS software sees a PQDN such as “kiwi” it will treat it as “kiwi.cs.widgetopia.edu”.

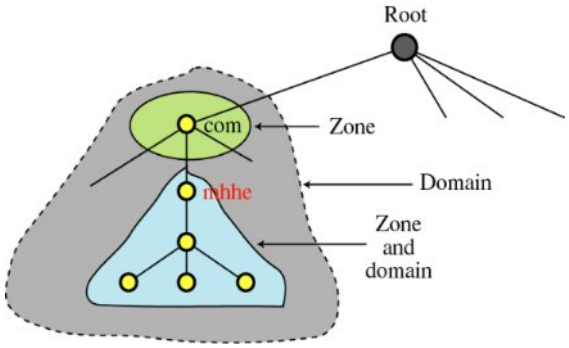
Q24. define and differentiate zone and domain

Ans 24:

- A domain is a portion of the overall DNS namespace.
- A zone, however, can contain multiple contiguous domains.
  Domain name servers store information about part of the domain name space called a zone. The name server is authoritative for a particular zone. A single name server can be authoritative for many zones.
- A zone is simply a portion of a domain. For example, the Domain Microsoft.com may contain all of the data for Microsoft.com, Marketing.microsoft.com and Development.microsoft.com. However, the zone Microsoft.com contains only information for Microsoft.com and references to the authoritative name servers for the sub-domains.
- The zone Microsoft.com can contain the data for sub-domains of Microsoft.com if they have not been delegated to another server. For example,

Marketing.microsoft.com may manage its own delegated zone. Development.microsoft.com may be managed by the parent, Microsoft.com.

- If there are no sub-domains, then the zone and domain are essentially the same. In this case the zone contains all data for the domain.



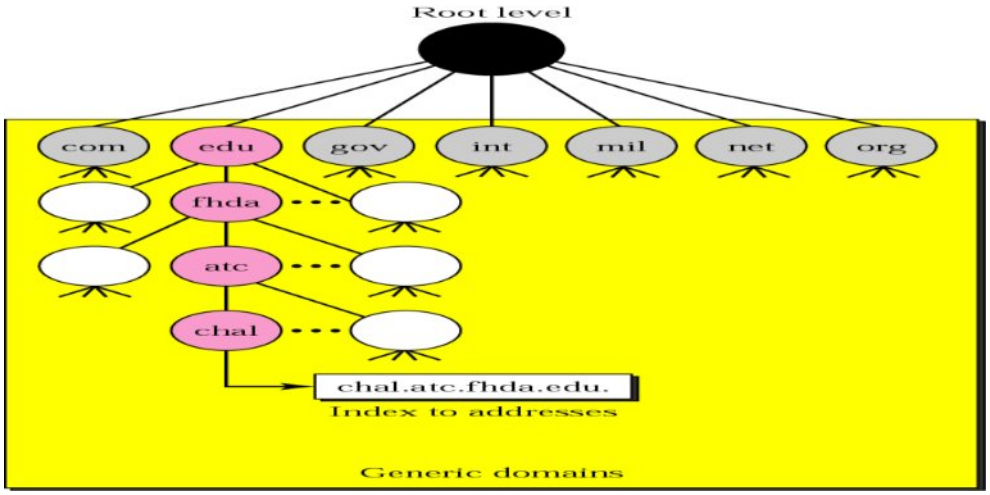
Q25. Define and explain domains of DNS used in internet.

Ans. DNS is a protocol that can be used in different platforms. In the internet, the domain is divided into three different sections.

- -Generic Domains
- -Country Domains
- -Inverse Domains

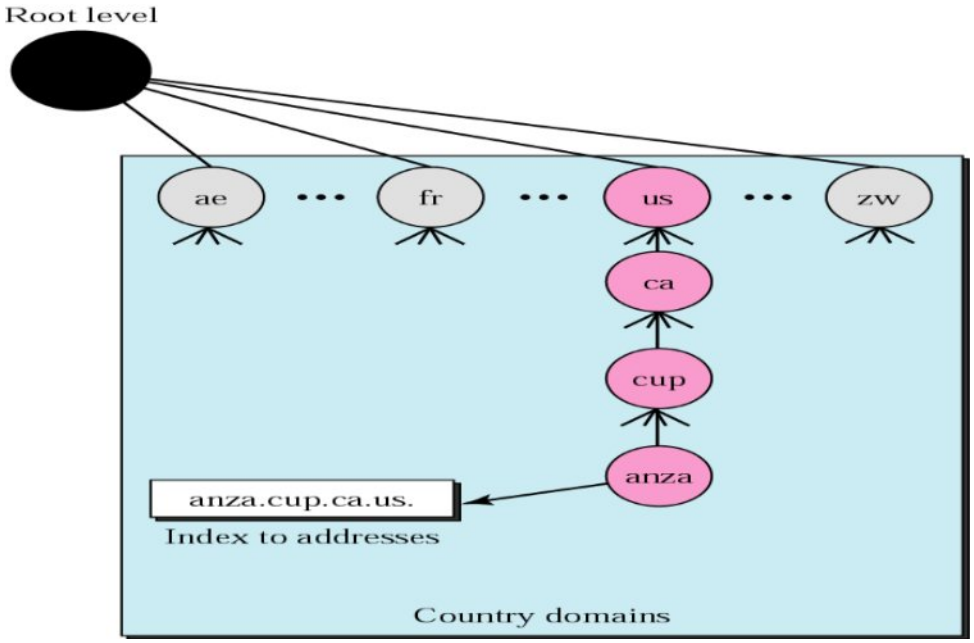
Generic Domains

- The generic domains define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database. The first level in the generic domains section allows seven possible three-character labels describing the organization types



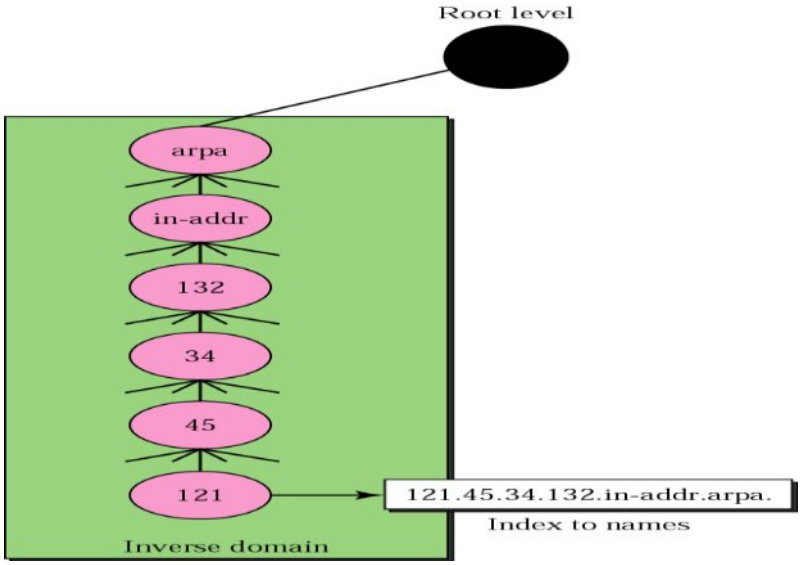
Country Domains

- The country domains section uses two-character country abbreviations.



Inverse Domain

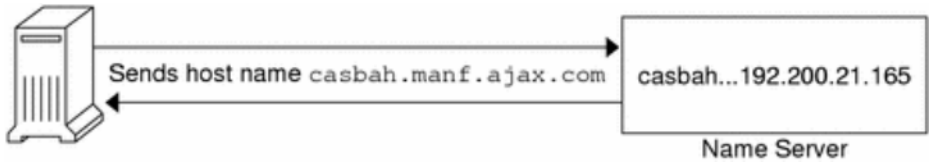
- The inverse domain is used to map an address to a name.



Q26. Explain name-address resolution in DNS.

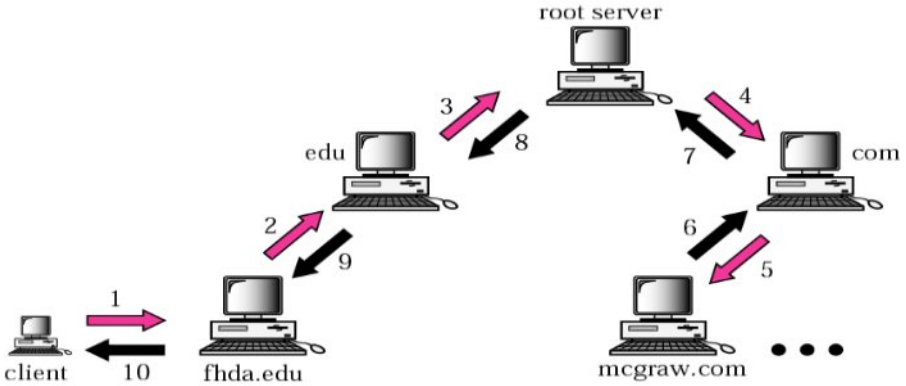
Ans. Though it supports the complex, worldwide hierarchy of computers on the Internet, the basic function of DNS is actually very simple: providing name-to-address resolution for TCP/IP-based networks. Name-to-address resolution, also referred to as mapping, is the process of finding the IP address of a computer in a database by using its host name as an index.

- Name-to-address mapping occurs when a program running on your local machine needs to contact a remote computer. The program most likely will know the host name of the remote computer but might not know how to locate it, particularly if the remote machine is in another company, miles from your site. To get the remote machine's address, the program requests assistance from the DNS software running on your local machine, which is considered a DNS client.
- Your machine sends a request to a DNS name server, which maintains the distributed DNS database. The files in the DNS database bear little resemblance to the NIS+ host or ipnodes Table or even the local /etc/hosts or /etc/inet/ipnodes file, though they maintain similar information: the host names, the ipnode names, IPv4 and IPv6 addresses, and other information about a particular group of computers. The name server uses the host name your machine sent as part of its request to find or “resolve” the IP address of the remote machine. It then returns this IP address to your local machine if the host name is in its DNS database.
- The following figure shows name-to-address mapping as it occurs between a DNS client and a name server, probably on the client's local network.
- Name to Address Resolution

- 
Sends host name casbah.manf.ajax.com
casbah...192.200.21.165
Name Server

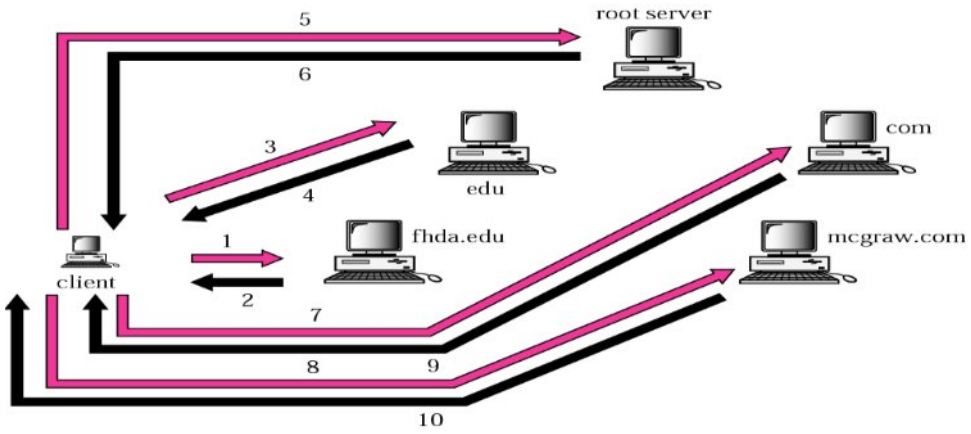
Q27. Draw and explain iterative and recursive resolution in DNS.

Ans. With a *recursive name query* , the DNS client requires that the DNS server respond to the client with either the requested resource record or an error message stating that the record or domain name does not exist. The DNS server cannot just refer the DNS client to a different DNS server.Thus, if a DNS server does not have the requested information when it receives a recursive query, it queries other servers until it gets the information, or until the name query fails.Recursive name queries are generally made by a DNS client to a DNS server, or by a DNS server that is configured to *query* is one in which a DNS client allows the DNS server to return the best answer it can give based on its cache or zone data. If the queried DNS server does not have an exact match for the queried name, the best possible information it can return is a *referral* (that is, a pointer to a DNS server authoritative for a lower level of the domain namespace). The DNS client can then query the DNS server for which it obtained a referral. It continues this process until it locates a DNS server that is authoritative for the queried name, or until an error or time-out condition is met.



Recursive resolution

In an iterative resolution: the name server, will not go and fetch the complete answer for your query, but will give back a referral to other DNS server's, which might have the answer



Iterative resolution

Q28. Explain DNS messages along with their header format.

Ans. DNS has two types of messages: query and response. Both types have the same format. The query message consists of a header and the question records; the response message consists of a header, question records, answer records, authoritative records, and additional records.

Both query and response messages have the same header format with some bits set to zero for the query messages.

Header Format

Identification	Flags
Number of question records	No. of answer records(all set to 0 in query messages)
No. of authoritative records (all 0s in query messages)	No. of additional records (all set to 0 in query messages)

Q29. Explain the situation in which BOOTP is required?

Before a device on a TCP/IP network can effectively communicate, it needs to know its IP address. While a conventional network host can read this information from its internal disk, some devices have no storage, and so do not have this luxury. They need help from another device on the network to provide them with an IP address and other information and/or software they need to become active IP hosts. This problem of getting a new machine up and running is commonly called *bootstrapping*, and to provide this capability to IP hosts, the TCP/IP *Bootstrap Protocol (BOOTP)* was created.

Q30. Explain BOOTP operation on same and different networks with the help of diagram.

Ans. The DHCP client and server can either be on the same network or on different networks.

Let us discuss each situation separately.

SAME NETWORK

Although the practice is not very common, the administrator may put the client and the server on the same network. In this case, the operation can be described as follows:

1. The DHCP server issues a passive open command on UDP port number 67 and waits for a client.

2. A booted client issues an active open command on port number 68 (this number will be explained later). The message is encapsulated in a UDP user datagram, using the destination port number 67 and the source port number 68. The UDP user datagram, in turn, is encapsulated in an IP datagram. The reader may ask how a

client can send an IP datagram when it knows neither its own IP address (the source address) nor the server's IP address (the destination address). The client uses all 0s as the source address and all 1s as the destination address.
3. The server responds with either a broadcast or a unicast message using UDP source port number 67 and destination port number 68. The response can be unicast because the server knows the IP address of the client. It also knows the physical address of the client, which means it does not need the services of ARP for logical to physical address mapping. However, some systems do not allow the bypassing of ARP, resulting in the use of the broadcast address.

DIFFERENT NETWORK
As in other application-layer processes, a client can be in one network and the server in another, separated by several other networks.
However, there is one problem that must be solved. The DHCP request is broadcast because the client does not know the IP address of the server. A broadcast IP datagram cannot pass through any router. A router receiving such a packet discards it. Recall that an IP address of all 1s is a limited broadcast address.
To solve the problem, there is a need for an intermediary. One of the hosts (or a router that can be configured to operate at the application layer) can be used as a relay.
The host in this case is called a **relay agent.** The relay agent knows the unicast address of a DHCP server and listens for broadcast messages on port 67. When it receives this type of packet, it encapsulates the message in a unicast datagram and sends the request to the DHCP server. The packet, carrying a unicast destination address, is routed by any router and reaches the DHCP server. The DHCP server knows the message comes from a relay agent because one of the fields in the request message defines the IP address of the relay agent. The relay agent, after receiving the reply, sends it to the DHCP client.

Q30. Draw the BOOTP message format and explain all the fields.

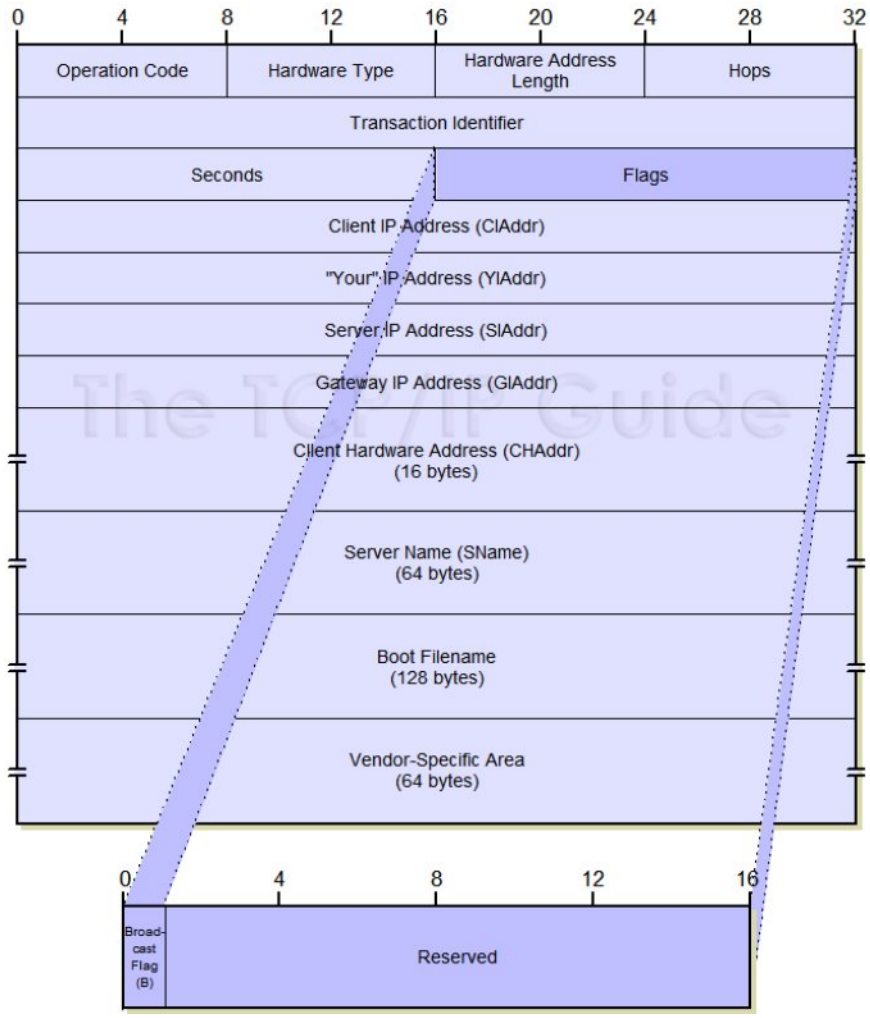


Figure 256: BOOTP Message Format

Table 186: BOOTP Message Format		
Field Name	**Size (bytes)**	**Description**
Op	1	***Operation Code:*** Specifies the type of message. A value of 1 indicates a request (*BOOTREQUEST* message) while a value of 2 is a reply (*BOOTREPLY* message).

<table>
<tr><td>HType</td><td>1</td><td>Hardware Type: This field specifies the type of hardware used for the local network, and is used in exactly the same way as the equivalent field (HRD) in the Address Resolution Protocol (ARP) message format. Some of the most common values for this field:
<table><tr><th>HType Field Value</th><th>Hardware Type</th></tr><tr><td>1</td><td>Ethernet (10 Mb)</td></tr><tr><td>6</td><td>IEEE 802 Networks</td></tr><tr><td>7</td><td>ARCNET</td></tr><tr><td>15</td><td>Frame Relay</td></tr><tr><td>16</td><td>Asynchronous Transfer Mode (ATM)</td></tr><tr><td>17</td><td>HDLC</td></tr><tr><td>18</td><td>Fibre Channel</td></tr><tr><td>19</td><td>Asynchronous Transfer Mode (ATM)</td></tr><tr><td>20</td><td>Serial Line</td></tr></table></td></tr>
<tr><td>HLen</td><td>1</td><td>Hardware Address Length: Specifies how long hardware addresses are in this message. For Ethernet or other networks using IEEE 802 MAC addresses, the value is 6. This too is the same as the field with a similar name (HLN) in the ARP field format.</td></tr>
<tr><td>Hops</td><td>1</td><td>Hops: Set to 0 by a client before transmitting a request and used by BOOTP relay agents to control the forwarding of BOOTP messages.</td></tr>
<tr><td>XID</td><td>4</td><td>Transaction Identifier: A 32-bit identification field generated by the client, to allow it to match up the request with replies received from BOOTP servers.</td></tr>
<tr><td>Secs</td><td>2</td><td>Seconds: According to RFC 951, the client enters into this field the number of seconds “elapsed since [the] client started trying to boot”. This is supposed to provide information to BOOTP servers to help them decide which requests to respond to first.

Unfortunately, this definition was somewhat vague; it wasn't clear if this meant the amount of time since the machine was powered on, or since the first BOOTREQUEST message was sent. In addition, some devices incorrectly implemented this field. As a result, it is not always used.</td></tr>
</table>

<table>
<tr><td>Flags</td><td>2</td><td>Flags: In the original BOOTP standard (RFC 951), this was an empty two-byte field. RFC 1542 changed this to a Flags field, which at present contains only one flag. The structure of the field is thus as follows:
<table><tr><th>Subfield Name</th><th>Size (bytes)</th><th>Description</th></tr><tr><td>B</td><td>1/8 (1 bit)</td><td>Broadcast Flag: A client that doesn't know its own IP address at the time it sends its BOOTP request sets this flag to 1. This serves as an immediate indicator to the BOOTP server or relay agent that receives the request that it definitely should send its reply by broadcast.</td></tr><tr><td>Reserved</td><td>1 7/8 (15 bits)</td><td>Reserved: Set to zero and not used.</td></tr></table></td></tr>
<tr><td>CIAddr</td><td>4</td><td>Client IP Address: If the client has a current IP address that it plans to keep using, it puts it in this field. By filling in this field, the client is committing to responding to unicast IP datagrams sent to this address. Otherwise, it sets this field to all zero to tell the server it wants an address assigned. See the end of the detailed operation topic for important information on this field.</td></tr>
<tr><td>YIAddr</td><td>4</td><td>“Your” IP Address: The IP address that the server is assigning to the client. This may be different than the IP address currently used by the client. See the topic describing BOOTP operation in detail for an explanation of what happens in that case.</td></tr>
<tr><td>SIAddr</td><td>4</td><td>Server IP Address: The IP address of the BOOTP server sending a BOOTREPLY message.</td></tr>
<tr><td>GIAddr</td><td>4</td><td>Gateway IP Address: This field is used to route BOOTP messages when BOOTP relay agents facilitate the communication of BOOTP requests and replies between a client and a server on different subnets or networks. To understand the name, remember that the old TCP/IP term for “router” is “gateway”; BOOTP relay agents are typically routers.

Note that this field is set to 0 by the client and should be ignored by the client when processing a BOOTREPLY. It specifically does not represent the server giving the client the address of a default router address to be used for general IP routing purposes.</td></tr>
<tr><td>CHAddr</td><td>16</td><td>Client Hardware Address: The hardware (layer two) address of the client sending a BOOTREPLY. It is used both to look up a device's assigned IP address and also possibly in delivery of a reply message.</td></tr>
</table>

SName	64	***Server Name:*** The server sending a *BOOTREPLY* may optionally put its name in this field. This can be a simple text "nickname" or a fully-qualified DNS domain name (such as "myserver.organization.org"). Note that a client may specify a name in this field when it creates its request. If it does so, it is saying that it wants to get a reply only from the BOOTP server with this name. This may be done to ensure that the client is able to access a particular boot file stored on only one server
File	128	***Boot Filename:*** Contains the full directory path and file name of a boot file that can be downloaded by the client to complete its bootstrapping process. The client may request a particular type of file by entering a text description here, or may leave the field blank and the server will supply the filename of the default file.
Vend	64	***Vendor-Specific Area:*** Originally created to allow vendors to customize BOOTP to the needs of different types of hardware, this field is now also used to hold additional vendor-independent configuration information. See the next topic, on BOOTP vendor information extensions. It may be used by the client and/or the server.

Q31. Write a short note on DHCP.

D*ynamic* H*ost* C*onfiguration* P*rotocol,* a protocol for assigning dynamic IP addresses to devices on a network. With dynamic addressing, a device can have a different IP address every time it connects to the network. In some systems, the device's IP address can even change while it is still connected. DHCP also supports a mix of static and dynamic IP addresses.
Dynamic addressing simplifies network administration because the software keeps track of IP addresses rather than requiring an administrator to manage the task. This means that a new computer can be added to a network without the hassle of manually assigning it a unique IP address. Many ISPs use dynamic IP addressing for dial-up users.

The **Dynamic Host Configuration Protocol (DHCP)** is a client/server protocol designed to provide the four pieces of information for a diskless computer or a computer that is booted for the first time. DHCP is a successor to BOOTP and is backward compatible with it. Although BOOTP is considered deprecated, there may be some systems that may still use BOOTP for host configuration.

32. Write a short note on DHCP.
Ans: Each computer that uses the TCP/IP protocol suite needs to know the following 4 pieces of information :
1. The IP address of the computer
2. The subnet mask of the computer
3. The IP address of a router

4. The IP address of a name server

These four pieces of information can be stored in a configuration file and accessed by the computer during the bootstrap process but in case of a diskless computer, the operating system and the networking software could not be stored in read-only memory (ROM) as it is not known to the manufacturer itself. The information is dependent on the individual configuration of the machine and defines the network to which the machine is connected. The **Dynamic Host Configuration Protocol (DHCP)** is a client/server protocol designed to provide the four pieces of information for a diskless computer or a computer that is booted for the first time. DHCP is a successor to BOOTP and is backward compatible with it. The DHCP client and server can either be on the same network or on different networks.

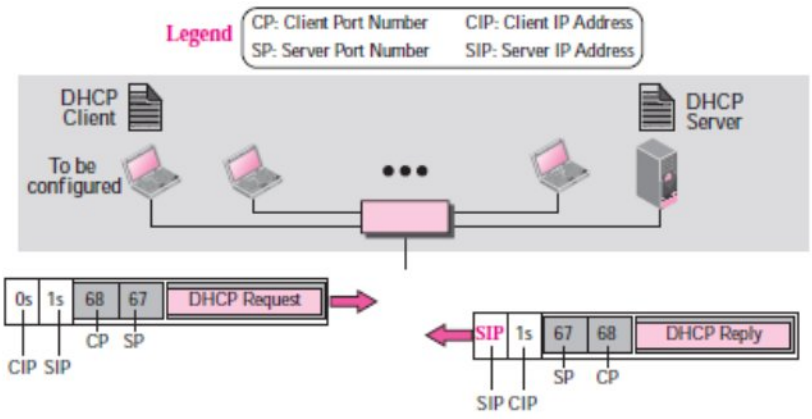
Operation in Same Network

When both the client and the server are in the same network the operation takes place as follows:

1. The DHCP server issues a passive open command on UDP port number 67 and waits for a client.

2. A booted client issues an active open command on port number 68. The message is encapsulated in a UDP user datagram, using the destination port number 67 and the source port number 68. The UDP user datagram, in turn, is encapsulated in an IP datagram. The client uses all 0s as the source address and all 1s as the destination address.

3. The server responds with either a broadcast or a unicast message using UDP source port number 67 and destination port number 68. The response can be unicast because the server knows the IP address of the client and also the physical address of the client, thus bypassing the services of ARP for logical to physical address mapping. The broadcast address is used for systems that donot allow bypassing of the ARP.

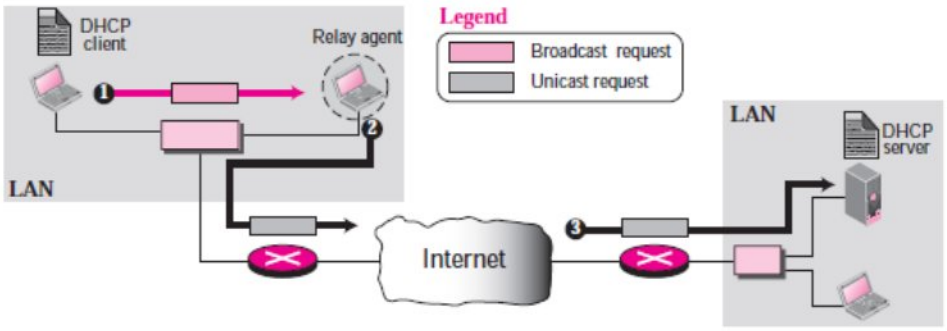


Operation in Different Network

When the client and the server are placed in different networks DHCP works as follows:

The DHCP request is broadcast because the client does not know the IP address of the server. But a broadcast IP datagram cannot pass through any router. A router receiving such a packet discards it as the IP address of all 1s is a limited broadcast address. To solve the problem, one of the hosts (or a router that can be configured to operate at the application layer) is used as a relay and is called a **relay agent.**

The relay agent knows the unicast address of a DHCP server and listens for broadcast messages on port 67. When it receives this type of packet, it encapsulates the message in a unicast datagram and sends the request to the DHCP server. The packet, carrying a unicast destination address, is routed by any router and reaches the DHCP server. The DHCP server knows the message comes from a relay agent because one of the fields in the request message defines the IP address of the relay agent. The relay agent, after receiving the reply, sends it to the DHCP client.



33. Explain Static and Dynamic Address Allocation in DHCP.

Ans : The DHCP has been devised to provide static and dynamic address allocation.

Static Address Allocation

In this capacity, a DHCP server has a database that statically binds physical addresses to IP addresses. When working in this way, DHCP is backward compatible with the deprecated protocol BOOTP.

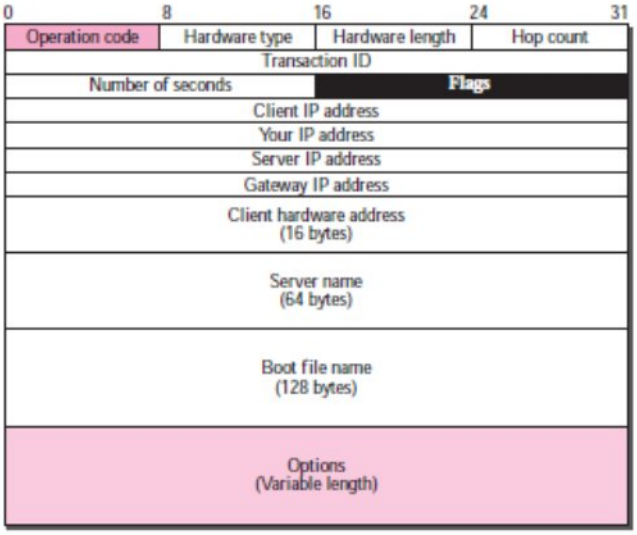
Dynamic Address Allocation

DHCP has a second database with a pool of available IP addresses. This second database makes DHCP dynamic. When a DHCP client requests a temporary IP address, the DHCP server goes to the pool of available (unused) IP addresses and assigns an IP address(temporary address) for a negotiable period of time. When a DHCP client sends a request to a DHCP server, the server first checks its static database. If an entry with the requested physical address exists in the static database, the permanent IP address of the client is returned. On the other hand, if the entry does not exist in the static database, the server selects an IP address from the available pool, assigns the address to the client, and adds the entry to the dynamic database.

The dynamic aspect of DHCP is needed when a host moves from network to network or is connected and disconnected from a network (for example, a subscriber to a service provider). DHCP provides temporary IP addresses for a limited period of time for which it issues a lease. When the lease expires, the client must either stop using the IP address or renew the lease. The server has the choice to agree or disagree with the renewal. If the server disagrees, the client stops using the address.

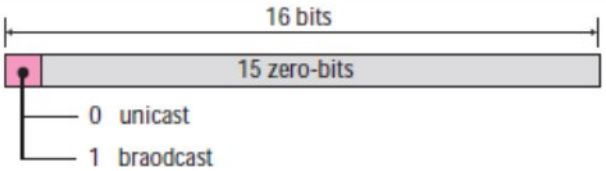
34. Draw DHCP packet format and explain the fields.

Ans :



- **Operation code.** This 8-bit field defines the type of DHCP packet: request (1) or reply (2).
- **Hardware type.** This is an 8-bit field defining the type of physical network. Each type of network has been assigned an integer. Eg : Ethernet = 1.
- **Hardware length.** This is an 8-bit field defining the length of the physical address in bytes. Eg: Ethernet = 6.
- **Hop count.** This is an 8-bit field defining the maximum number of hops the packet can travel.
- **Transaction ID.** This is a 4-byte field carrying an integer whose value is set by the client and is used to match a reply with the request. The server returns the same value in its reply.
- **Number of seconds.** This is a 16-bit field that indicates the number of seconds elapsed since the time the client started to boot.
- **Flag.** This is a 16-bit field in which only the leftmost bit is used and the rest of the bits should be set to 0s. A leftmost bit specifies a forced broadcast reply (instead of unicast) from the server. If the reply were to be unicast to the client, the destination IP address of the IP packet is the address assigned to the client. Since the client does not know its IP address, it may discard the packet. However, if the IP datagram is broadcast, every host will receive and process the broadcast message.

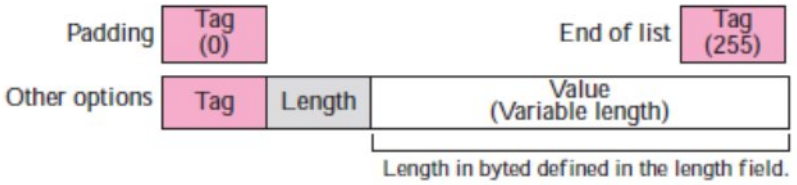
Figure 18.5 *Flag format*



- **Client IP address.** This is a 4-byte field that contains the client IP address. If the client does not have this information, this field has a value of 0.
- **Your IP address.** This is a 4-byte field that contains the client IP address. It is filled by the server (in the reply message) at the request of the client.
- **Server IP address.** This is a 4-byte field containing the server IP address. It is filled by the server in a reply message.

- **Gateway IP address.** This is a 4-byte field containing the IP address of a router. It is filled by the server in a reply message.
- **Client hardware address.** This is the physical address of the client. Although the server can retrieve this address from the frame sent by the client, it is more efficient if the address is supplied explicitly by the client in the request message.
- **Server name.** This is a 64-byte field that is optionally filled by the server in a reply packet. It contains a null-terminated string consisting of the domain name of the server. If the server does not want to fill this field with data, the server must fill it with all 0s.
- **Boot filename.** This is a 128-byte field that can be optionally filled by the server in a reply packet. It contains a null-terminated string consisting of the full pathname of the boot file. The client can use this path to retrieve other booting information. If the server does not want to fill this field with data, the server must fill it with all 0s.
- **Options.** This is a 64-byte field with a dual purpose. It can carry either additional information (such as the network mask or default router address) or some specific vendor information. The field is used only in a reply message. The server uses a number,called a **magic cookie,** in the format of an IP address with the value of 99.130.83.99. When the client finishes reading the message, it looks for this magic cookie. If present, the next 60 bytes are options. An option is composed of three fields: a 1-byte tag field, a 1-byte length field, and a variable-length value field. The length field defines the length of the value field, not the whole option.

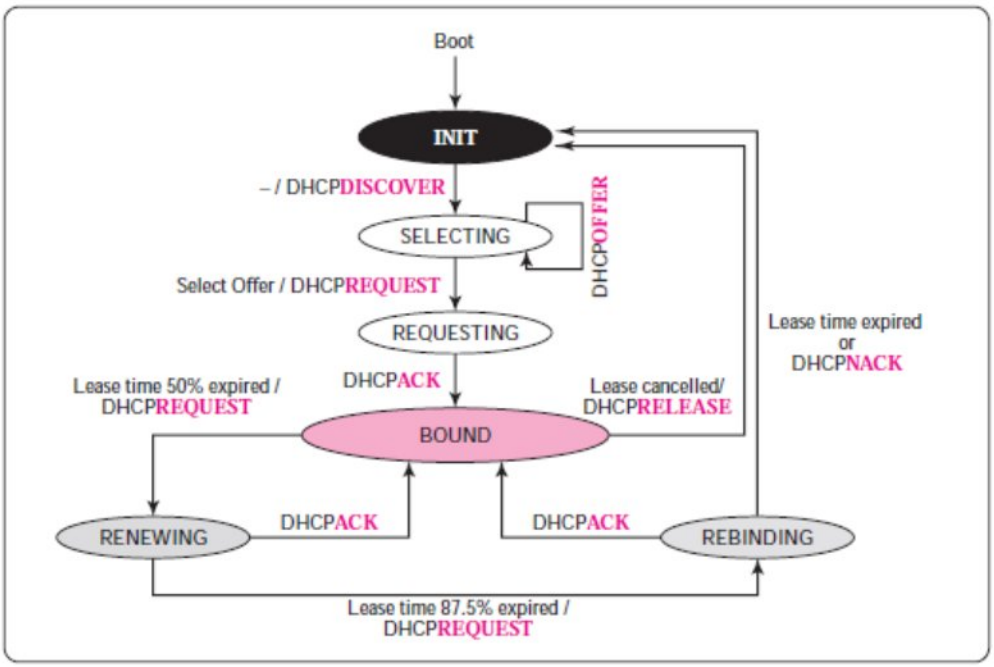
Figure 18.6 *Option format*



35. Draw DHCP transition Diagram. Also define and explain the transition states.

Ans:

Figure 18.8 *DHCP client transition diagram*



INIT State

When the DHCP client first starts, it is in the INIT state (initializing state). The client broadcasts a request message with the DHCPDISCOVER option, using port 67.

SELECTING State

After sending the DHCPDISCOVER message, the client goes to the selecting state. Those servers that can provide this type of service respond with a DHCPOFFER message. In these messages, the servers offer an IP address. They can also offer the lease duration (Default is 1 hour). The server that sends a DHCPOFFER locks the offered IP address so that it is not available to any other clients. The client chooses one of the offers and sends a CPREQUEST message to the selected server. It then goes to the requesting state. However, if the client receives no DHCPOFFER message, it tries four more times, each with a span of 2 seconds. If there is no reply to any of these DHCPDISCOVERs, the client sleeps for 5 minutes before trying again.

REQUESTING State

The client remains in the requesting state until it receives a DHCPACK message from the server that creates the binding between the client physical address and its IP address. After receipt of the DHCPACK, the client goes to the bound state.

BOUND State

In this state, the client can use the IP address until the lease expires. When 50 percent of the lease period is reached, the client sends another DHCPREQUEST to ask for renewal. It then goes to the renewing state. When in the bound state, the client can also cancel the lease and go to the initializing state.

RENEWING State

The client remains in the renewing state until one of two events happens. It can receive a DHCPACK, which renews the lease agreement. In this case, the client

resets its timer and goes back to the bound state. Or, if a DHCPACK is not received, and 87.5 percent of the lease time expires, the client goes to the rebinding state.

REBINDING State

The client remains in the rebinding state until one of three events happens. If the client receives a DHCPNACK or the lease expires, it goes back to the initializing state and tries to get another IP address. If the client receives a DHCPACK, it goes to the bound state and resets the timer.

36. Explain DDNS.

Ans: DNS was designed, when no one predicted that there would be so many address changes. In DNS, when there is a change, such as adding a new host, removing a host,or changing an IP address, the change must be made to the DNS master file. These types of changes involve a lot of manual updating that is unfeasable with the sieze of Internet. The DNS master file must be updated dynamically. The **Dynamic Domain Name System (DDNS)** was devised to respond to this need. In DDNS, when a binding between a name and an address is determined, the information is sent, usually by DHCP to a primary DNS server. The primary server updates the zone. The secondary servers are notified either actively or passively.

In active notification, the primary server sends a message to the secondary servers about the change in the zone,

whereas in passive notification, the secondary servers periodically check for any changes. In either case, after being notified about the change, the secondary requests information about the entire zone (zone transfer).

To provide security and prevent unauthorized changes in the DNS records, DDNS can use an authentication mechanism.

Q37. Explain usage of TELNET.

ANS : Telnet is an old computer protocol (set of programmatic rules). Telnet is famous for being the original Internet when the Net first launched in 1969. Telnet stands for 'telecommunications network', and was built to be form of remote control to manage mainframe computers from distant terminals. In those original days of large mainframe computers, telnet enabled research students and professors to 'log in' to the university mainframe from any terminal in the building. This remote login saved researchers hours of walking each semester. While telnet pales in comparison to modern networking technology, it was revolutionary in 1969, and telnet helped pave the way for the eventual World Wide Web in 1989. While telnet technology is very old, it is still in some use today by purists. Telnet has evolved into a new modern version of remote control called 'SSH', something that many modern network administrators use today to manage linux and unix computers from a distance.

Telnet is a text-based computer protocol. Unlike Firefox or Google Chrome screens, telnet screens are very dull to look at. Very different from Web pages that sport fancy images, animation, and hyperlinks, telnet is about typing on a keyboard. Telnet commands can be rather cryptic commands, with example commands being 'z' and 'prompt% fg'. Most modern users would find telnet screens to be very archaic and slow.

Q 38) Explain and differentiate between Remote Login and Local Login.

Telnet is known as TERMINAL NETWORK which is a client-server application program. It enables the establishment of a connection to a remote system in such a way that the local terminal appears to be a terminal at the remote system.
We have two types of login here. They are
*LOCAL LOGIN
*REMOTE LOGIN

LOCAL LOGIN:
When a user logs into a local time-sharing program, it is local login. As a user types at a terminal running a terminal emulator, the keystrokes are accepted by the terminal driver. The terminal driver passes the characters to the operating system. The operating system may assign special meanings to characters. These situations do not create any problem in local login because the terminal emulator and the terminal driver know the exact meaning of each character, they may create problems in remote login.

REMOTE LOGIN:
when a user wants to access an application program, he or she performs remote login. The user sends keystrokes to the terminal driver where the local operating system accepts the characters but they won't interpret them. They are sent to TELNET CLIENT, which in turn transforms the characters to a universal character set called "Network Virtual terminal characters" and delivers them to local TCP/IP stack. However, characters can't be passed directly to the operating system because remote operating system is not designed to receive characters from TELNET SERVER. It is designed such that it receives characters from TERMINAL DRIVERS. The solution for this is to add a piece of software called a Pseudo terminal driver, which pretends that the characters are coming from a terminal. The operating system passes the characters to appropriate application programs.

Q 39) Explain NVT and NVT character set.

Communication is established using the TCP/IP protocols and communication is based on a set of facilities known as a Network Virtual Terminal (NVT). At the user or client end the telnet client program is responsible for mapping incoming NVT codes to the actual codes needed to operate the user's display device and is also responsible for mapping user generated keyboard sequences into NVT sequences.

Telnet uses an approach similar to the analogy described above for dealing with its problem of hardware and software compatibility. Rather than having terminals and hosts communicate using their various native "languages", all Telnet clients and servers agree to send data and commands that adhere to a fictional, "virtual" terminal type call the Network Virtual Terminal (NVT). The NVT defines a set of rules for how information is formatted and sent, such as character set, line termination, and how information about the Telnet session itself is sent.

Each Telnet client running on a terminal understands both its native language and NVT. When information is entered by the user on his or her local terminal, it is converted to NVT for transmission over the network in NVT form. When the Telnet server receives this information, it translates it from NVT to the format that the remote host expects to receive it. The identical process is performed for transmissions from the server to the client, in reverse.

The NVT uses 7 bit codes for characters, the display device, referred to as a printer, is only required to display the "standard" printing ASCII characters represented by 7 bit codes and to recognise and process certain control codes. The 7 bit characters are transmitted as 8 bit bytes with most significant bit set to zero. An end-of-line is transmitted as the character sequence CR (carriage return) followed by LF (line feed). If it is desired to transmit an actual carriage return this is transmitted as a carriage return followed by a NUL (all bits zero) character.

The NVT is defined to consist of a logical “keyboard” for input and a logical “printer” for output (the age of the protocol is reflected in these terms; decades ago there were no monitors, all output was on paper). NVT uses the 7-bit *United States ASCII* (*USASCII*) character set. Each character is encoded using one 8-bit byte.

Name	**code**	**Decimal Value**	**Function**
NULL	NUL		No operation
Line Feed	LF	10	Moves the printer to the next print line,
			keeping the same horizontal position
Carriage Return	CR	13	Moves the printer to the left margin
			of the current line

Q 40) Explain embedding in TELNET.

TELNET uses only one TCP connection. The server uses the well-known port 23 and the client uses an ephemeral port. The same connection is used for sending both data and control characters. TELNET accomplishes this by embedding the control characters in the data stream. However, to distinguish data from control characters, each sequence of control characters is preceded by a special control character called interpret as control (IAC).

Q41.Explain option negotiation and sub-option negation in TELNET?

Ans. TELNET lets the client and server negotiate options before or during the use of the service.
Options are extra features available to a user with a more sophisticated terminal. Users with simpler terminals can use default features.

Code Option Meaning

0	Binary	Interpret as 8-bit binary transmission
1	Echo	Echo the data received on one side to the other
3	Suppress go-ahead	Suppress go-ahead signals after data
5	Status	Request the status of TELNET
6	Timing mark	Define the timing marks
24	Terminal type	Set the terminal type
32	Terminal speed	Set the terminal speed
34	Line mode	Change to line mode

To use any of the options mentioned in the previous section first requires option negotiation
between the client and the server.

Character	Code	Meaning 1	Meaning 2	Meaning 3
WILL	251	Offering to enable	Accepting to enable	
WONT	252	Rejecting to enable	Offering to disable	Accepting to disable
DO	253	Approving to enable	Requesting to enable	
DON'T	254	Disapproving to enable	Approving to disable	Requesting to disable

Suboption Negotiation

Some options require additional information. For example, to define the type or speed

of a terminal, the negotiation includes a string or a number to define the type or speed.

Q42.Write and explain control characters used in Telnet to control the remote server?

Ans. Some control characters can be used to control the remote server. When an application
program is running on the local computer, special characters are used to interrupt

(abort) the program (for example, Ctrl+c), or erase the last character typed (for example,
delete key or backspace key), and so on. However, when a program is running on a remote computer, these control characters are sent to the remote machine. The user
still types the same sequences, but they are changed to special characters and sent to the server.

IP (interrupt process). When a program is being run locally, the user can interrupt (abort) the program if, for example, the program has gone into an infinite loop. The user can type the Ctrl+c combination, the operating system calls a function, and the function aborts the program. However, if the program is running on a remote machine, the appropriate function should be called by the operating system of the remote machine. TELNET defines the IP (interrupt process) control character that is read and interpreted as the appropriate command for invoking the interrupting function in the remote machine.

AO (abort output). This is the same as IP, but it allows the process to continue without creating output. This is useful if the process has another effect in addition to creating output. The user wants this effect but not the output. For example, most commands in UNIX generate output and have an exit status. The user may want the exit status for future use but is not interested in the output data.

AYT (are you there?). This control character is used to determine if the remote machine is still up and running, especially after a long silence from the server.When this character is received, the server usually sends an audible or visual signal to confirm that it is running.

EC (erase character). When a user sends data from the keyboard to the local machine, the delete or backspace character can erase the last character typed. To do the same in a remote machine, TELNET defines the EC control character.

EL (erase line). This is used to erase the current line in the remote host.

Q43.Explain out of band signaling in Telnet?

Ans. To make control characters effective in special situations, TELNET uses out-of-band signaling. In out-of-band signaling, the control characters are preceded by IAC and are sent to the remote process.
Imagine a situation in which an application program running at the server site has gone into an infinite loop and does not accept any more input data. The user wants to interrupt the application program, but the program does not read data from the buffer.
The TCP at the server site has found that the buffer is full and has sent a segment specifying that the client window size should be zero. In other words, the TCP at the server site is announcing that no more regular traffic is accepted. To remedy such a

situation, an urgent TCP segment should be sent from the client to the server. The urgent segment overrides the regular flow-control mechanism. Although TCP is not accepting normal segments, it must accept an urgent segment. When a TELNET process (client or server) wants to send an out-of-band sequence of characters to the other process (client or server), it embeds the sequence in the data stream and inserts a special character called a DM (data mark). However, to inform the other party, it creates a TCP segment with the urgent bit set and the urgent pointer pointing to the DM character. When the receiving process receives the data, it reads the data and discards any data preceding the control characters (IAC and IP, for example). When it reaches the DM character, the remaining data are handled normally. In other
words, the DM character is used as a synchronization character that switches the receiving process from the urgent mode to the normal mode and resynchronizes the two ends In this way, the control character (IP) is delivered out of band to the operating system, which uses the appropriate function to interrupt the running application program.

Q44.Explain usage of escape character for usage of interrupt the application program and interrupting the clients?

Ans. A character typed by the user is normally sent to the server. However, sometimes the
user wants characters interpreted by the client instead of the server. In this case, the user can use an escape character, normally compares the interruption of an application program at the remote site with the interruption of the client process at the local site using the escape character. The TELNET prompt is displayed after this escape character.

Q45 . Modes of Operation in Telnet

Ans. Most TELNET implementations operate in one of three modes: default mode, character
mode, or line mode.

Default Mode

The default mode is used if no other modes are invoked through option negotiation. In
this mode, the echoing is done by the client. The user types a character and the client
echoes the character on the screen (or printer) but does not send it until a whole line is
completed. After sending the whole line to the server, the client waits for the GA (go ahead) command from the server before accepting a new line from the user. The operation
is half-duplex. Half-duplex operation is not efficient when the TCP connection itself is full-duplex, and so this mode is becoming obsolete

Character Mode

In the character mode, each character typed is sent by the client to the server. The server normally echoes the character back to be displayed on the client screen. In this
mode the echoing of the character can be delayed if the transmission time is long (such
as in a satellite connection). It also creates overhead (traffic) for the network because
three TCP segments must be sent for each character of data:

1. The user enters a character that is sent to the server.
2. The server acknowledges the received character and echoes the character back (in
one segment).
3. The client acknowledges the receipt of the echoed character.

Line Mode

A new mode has been proposed to compensate for the deficiencies of the default mode
and the character mode. In this mode, called the line mode, line editing (echoing, character erasing, line erasing, and so on) is done by the client. The client then sends the whole line to the server. Although the line mode looks like the default mode, it is
not. The default mode operates in the half-duplex mode; the line mode is full-duplex with the client sending one line after another, without the need for an intervening GA (go ahead) character from the serve

46. For using telnet, operating systems defines an interface with user friendly commands. List example of such commands and explain its usage.

Ans : Usually, the operating system (UNIX, for example) defines an interface with user-friendly commands.the interface is responsible for translating the user-friendly commands to the previously defined commands in the protocol.

Command	*Meaning*	*Command*	*Meaning*
open	Connect to a remote computer	set	Set the operating parameters
close	Close the connection	status	Display the status information
display	Show the operating parameters	send	Send special characters
mode	Change to line or character mode	quit	Exit TELNET

47.What are the security issues of using telnet.

Ans: Security issues faced by Telnet are:

1.Telnet, by default, does not encrypt any data sent over the connection (including passwords), and so it is often practical to eavesdrop on the communications and use the password later for malicious purposes

Eg : A microcomputer connected to a broadcast LAN can easily eavesdrop using snooper software and capture the login name and corresponding password

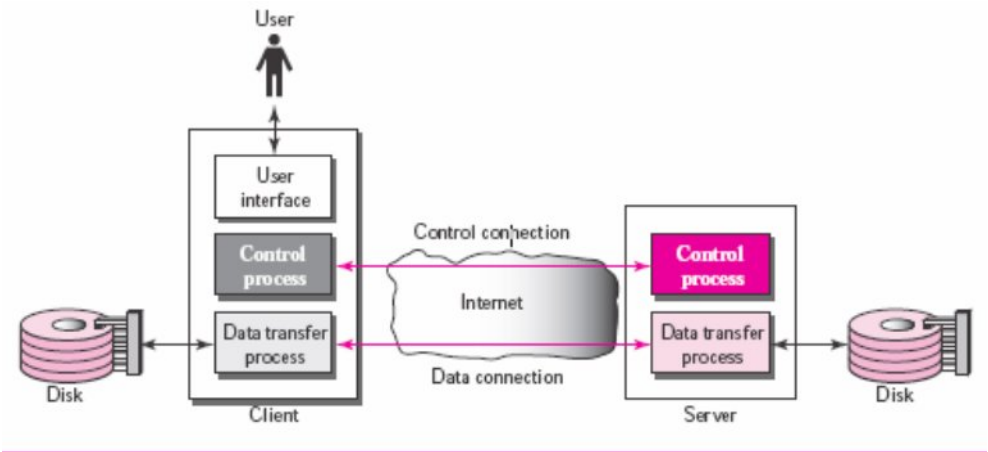
2. Most implementations of Telnet have no authentication that would ensure communication is carried out between the two desired hosts and not intercepted in middle.
3.Several vulnerabilities have been discovered over the years in commonly used Telnet daemons ie background processes.

48.Write a short note on FTP along with their known ports.
Ans: File Transfer protocol is the standard mechansim provided by TCP/IP for copying a file from one host to another.

.FTP provides an efficient solution to problems like naming conventions of file and different representations used for text and data.
. FTP differs from other client-server applications in that it establishes two connections
between the hosts. One connection is used for data transfer, the other for control
information (commands and responses)
1.Control Connection: It uses simple rules of communication as we need to transfer a single line of comand or response only.It stays connected during entire FTP session .
2.Data connnection :It is opened and closed for each file transferred.It uses complex rules of transmission due to variety of data types.

FTP uses two well-known TCP ports: Port 21 is used for the control connection, and port 20 is used for the data connection.The server issues a passive open on port 21 and active open on port 20

Basic Model of FTP



The client has three components: user, interface, client control process, and the client data transfer process.
The server has two components: the server control process and the server data transfer process.
The control connection is made between the control processes. The data connection is made
between the data transfer processes.

49. Explain two FTP connections.

Ans : FTP differs from other client-server applications in that it establishes two connections between the hosts. One connection is used for data transfer, the other for control information (commands and responses)

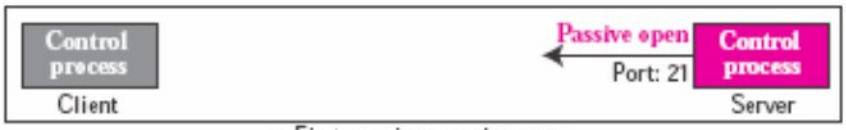
1**.Control Connection**: It uses simple rules of communication as we need to transfer a single line of comand or response only.It stays connected during entire FTP session .

.

The control connection is created in the same way as other application programs described so far. There are two steps:

1. The server issues a passive open on the well-known port 21 and waits for a client.

2. The client uses an ephemeral port and issues an active open



a. First, passive open by server

b. Later, active open by client

This service type, used by the IP protocol, is *minimize delay* because this is an interactive connection between a user (human) and a server.

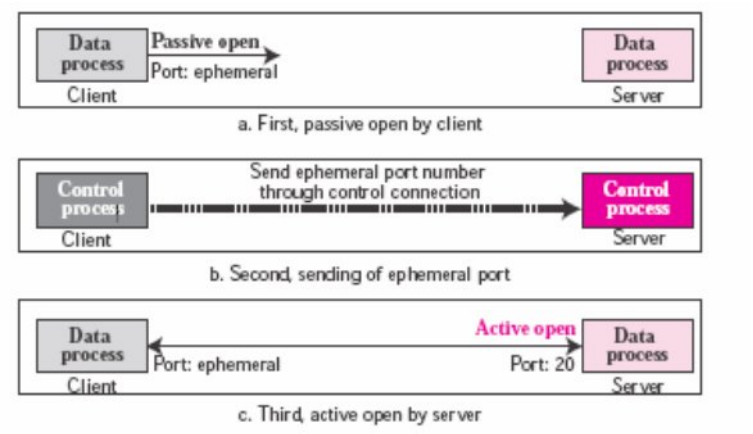
2**.Data connnection** :It is opened and closed for each file transferred.It uses complex rules of transmission due to variety of data types

The following shows how FTP creates a data connection:

1. The client, not the server, issues a passive open using an ephemeral port. This must be done by the client because it is the client that issues the commands for transferring files.

2. The client sends this port number to the server using the PORT command

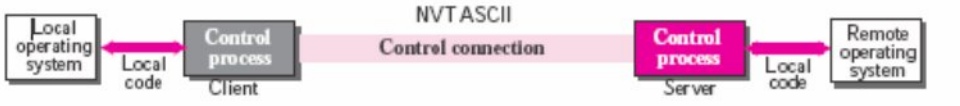
3. The server receives the port number and issues an active open using the well known port 20 and the received ephemeral port number.



50. Explain how communication takes place in FTP with the help of a diagram.
Ans : FTP has two different approaches, one for the control connection and one for the
data connection

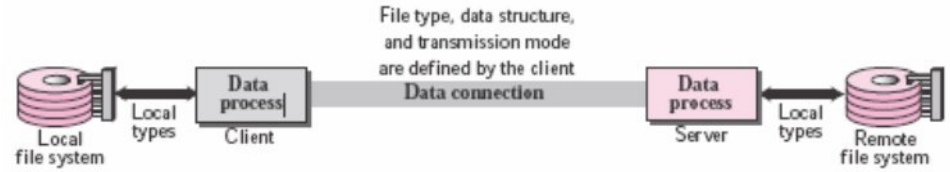
Communication over Control Connection

FTP uses the same approach as TELNET or SMTP to communicate across the control
connection. It uses the NVT ASCII (8 bit) character set . Communication is achieved through commands and responses.
Each command or response is only one short line so there is no concern regarding file format or file structure. Each line is terminated with a two-character (carriage return and line feed)
end-of-line token.



Communication over Data Connection

The purpose and implementation of the data connection are different from that of the control
connection. We want to transfer files through the data connection. The client must define the type of file to be transferred, the structure of the data, and the transmission mode.



File Type FTP can transfer one of the following file types across the data connection:

- **ASCII file.** This is the default format for transferring text files. Each character is encoded using NVT ASCII. Both sender and receiver is having its own representation of this format.

❑ **EBCDIC file.** If one or both ends of the connection use EBCDIC encoding, the file can be transferred using EBCDIC encoding.

❑ **Image file.** This is the default format for transferring binary files. The file is sent as continuous streams of bits without any interpretation or encoding.

If the file is encoded in ASCII or EBCDIC, another attribute must be added to define the printability of the file.

a. Nonprint. This is the default format for transferring a text file having no vertical specifications for printing.
b. TELNET. In this format the file contains NVT ASCII vertical characters such as CR (carriage return), LF (line feed), NL (new line), and VT (vertical tab). The file is printable after transfer.

Data Structure FTP can transfer a file across the data connection using one of the following interpretations about the structure of the data:
❑ **File structure (default).** The file has no structure. It is a continuous stream of bytes.
Record structure. The file is divided into records. This can be used only with text files.
❑ **Page structure.** The file is divided into pages, with each page having a page number and a page header. The pages can be stored and accessed randomly or sequentially.

Transmission Mode FTP can transfer a file across the data connection using one of the following three transmission modes:
❑ **Stream mode.** This is the default mode. Data are delivered from FTP to TCP as a continuous stream of bytes. TCP is responsible for chopping data into segments of appropriate size.
❑ **Block mode.** Data can be delivered from FTP to TCP in blocks. In this case, each block is preceded by a 3-byte header. The first byte is called the *block descriptor;* the next two bytes define the size of the block in bytes.
❑ **Compressed mode.** If the file is big, the data can be compressed. The compression method normally used is run-length encoding

Q. 51. Explain and categorize commands used in FTP.
Ans. List of all the commands used in FTP:

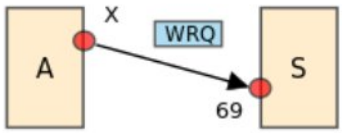
Command	Description
ABOR	Abort an active file transfer.
ACCT	Account information.
ADAT	Authentication/Security Data
ALLO	Allocate sufficient disk space to receive a file.
APPE	Append.

Command	Description
AUTH	Authentication/Security Mechanism
CCC	Clear Command Channel
CDUP	Change to Parent Directory.
CONF	Confidentiality Protection Command
CWD	Change working directory.
DELE	Delete file.
ENC	Privacy Protected Channel
EPRT	Specifies an extended address and port to which the server should connect.
EPSV	Enter extended passive mode.
FEAT	Get the feature list implemented by the server.
HELP	Returns usage documentation on a command if specified, else a general help document is returned.
LANG	Language Negotiation
LIST	Returns information of a file or directory if specified, else information of the current working directory is returned.
LPRT	Specifies a long address and port to which the server should connect.
LPSV	Enter long passive mode.
MDTM	Return the last-modified time of a specified file.
MIC	Integrity Protected Command
MKD	Make directory.
MLSD	Lists the contents of a directory if a directory is named.
MLST	Provides data about exactly the object named on its command line, and no others.
MODE	Sets the transfer mode (Stream, Block, or Compressed).
NLST	Returns a list of file names in a specified directory.
NOOP	No operation (dummy packet; used mostly on keep-alives).
OPTS	Select options for a feature.
PASS	Authentication password.
PASV	Enter passive mode.
PBSZ	Protection Buffer Size
PORT	Specifies an address and port to which the server should connect.
PROT	Data Channel Protection Level.
PWD	Print working directory. Returns the current directory of the host.
QUIT	Disconnect.

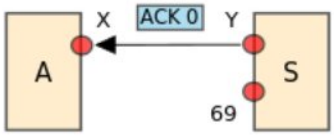
Command	Description
REIN	Re initializes the connection.
REST	Restart transfer from the specified point.
RETR	Transfer a copy of the file
RMD	Remove a directory.
RNFR	Rename from.
RNTO	Rename to.
SITE	Sends site specific commands to remote server.
SIZE	Return the size of a file.
SMNT	Mount file structure.
STAT	Returns the current status.
STOR	Accept the data and to store the data as a file at the server site
STOU	Store file uniquely.
STRU	Set file transfer structure.
SYST	Return system type.
TYPE	Sets the transfer mode (ASCII/Binary).
USER	Authentication username.
XCUP	Change to the parent of the current working directory
XMKD	Make a directory
XPWD	Print the current working directory
XRCP	
XRMD	Remove the directory
XRSQ	
XSEM	Send, mail if cannot
XSEN	Send to terminal

Q. 52. Write a short note on TFTP along with its most common port numbers.
Ans. Trivial File Transfer Protocol (**TFTP**) is a file transfer protocol notable for its simplicity. It is generally used for automated transfer of configuration or boot files between machines in a local environment. Compared to FTP, TFTP is extremely limited, providing no authentication, and is rarely used interactively by a user.
Due to its simple design, TFTP could be implemented using a very small amount of memory. It is therefore useful for booting computers such as routers which may not have any data storage devices.
Trivial File Transfer Protocol (TFTP) is a simple protocol to transfer files. It has been implemented on top of the User Datagram Protocol (UDP) using port number 69. TFTP is designed to be small and easy to implement, and therefore it lacks most of the features of a regular FTP. TFTP only reads and writes files (or mail) from/to a

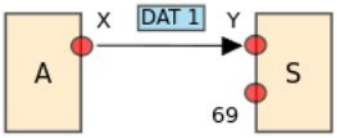
remote server. It cannot list directories, and currently has no provisions for user authentication.
TFTP defines three modes of transfer: netascii, octet, and mail. Netascii is a modified form of ASCII, defined in RFC 764. It consists of an 8-bit extension of the 7-bit ASCII character space from 0x20 to 0x7F (the printable characters and the space) and eight of the control characters. The allowed control characters include the null (0x00), the line feed (LF, 0x0A), and the carriage return (CR, 0x0D). Netascii also requires that the end of line marker on a host be translated to the character pair CR LF for transmission, and that any CR must be followed by either a LF or the null.
Octet allows for the transfer of arbitrary 8-bit bytes, with the received file identical to the sent file. More correctly, if a host receives an octet file and then returns it, the returned file must be identical to the original.[4] The Mail transfer mode uses Netascii transfer, but the file is sent to an email recipient by specifying that recipient's email address as the file name. RFC 1350 declared this mode of transfer obsolete.
No security or authentication is provided by the protocol specification. Unix implementations often restrict file transfers to a single configured directory, and only to read from files with world readability, and only write to already existing files that have world writability.
Protocol walkthrough



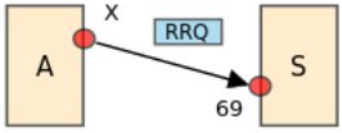
(W1) Host A requests to write



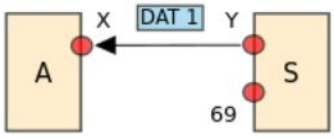
(W2) Server S acknowledges request



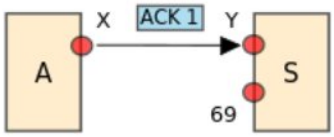
(W3) Host A sends numbered data packets



(R1) Host A requests to read



(R2) Server S sends data packet 1



(R3) Host A acknowledges data packet 1

The initiating host A sends an RRQ (read request) or WRQ (write request) packet to host S at port number 69, containing the filename and transfer mode.

S replies with an ACK (acknowledgement) packet to WRQ and directly with a DATA packet to RRQ. Packet is sent from a freshly allocated ephemeral port, and all future packets to host S should be to this port.

The source host sends numbered DATA packets to the destination host, all but the last containing a full-sized block of data (512 bytes). The destination host replies with numbered ACK packets for all DATA packets.

The final DATA packet must contain less than a full-sized block of data to signal that it is the last. If the size of the transferred file is an exact multiple of the block-size, the source sends a final DATA packet containing 0 bytes of data.

Receiver responds to each DATA with associated numbered ACK. Sender responds to the first received ACK of a block with DATA of the next block.

If an ACK is not eventually received, a retransmit timer resends DATA packet.

TFTP uses UDP port number 69 to establish connections.

Q. 53. Explain 5 types of TFTP messages.

Ans. Five different Operation (Op) Codes are defined:

Read Request (RRQ): When a system needs to Read something from another system in a LAN, it sends an RRQ message to the desired system's port number 69. After sending the RRQ message, it waits for an ACK message after it can start reading.

Write Request (WRQ): When a system needs to Write something to another system in a LAN, it sends a WRQ message to the desired system's port number 69. After sending the WRQ message, it waits for an ACK message after which it can start writing.

Data: Data message is the message by which the systems Read or Write files on the other system. All Data packets, but the last one, are equal in size. The last Data packet must be smaller than the previous ones to indicate that it is the last one.

ACK: Acknowledge message is sent whenever an RRQ, a WRQ or a Data message is accepted.

Error: Error message is sent to indicate any error that has occurred during the transmission.

Q. 54. Explain Connection Establishment and Connection Termination in TFTP.

Ans. TFTP communication is client/server based. The process of transferring a file consists of three main phases. In highly generalized terms, these are:

Initial Connection: The TFTP client establishes the connection by sending an initial request to the server. The server responds back to the client and the connection is effectively opened.

Data Transfer: Once the connection is established, the client and server exchange TFTP messages. One device sends data and the other sends acknowledgments.

Connection Termination: When the last TFTP message containing data has been sent and acknowledged, the connection is terminated.

Initial Connection Protocol
A transfer is established by sending a request (WRQ to write onto a foreign file system, or RRQ to read from it), and receiving a positive reply, an acknowledgment packet for write, or the first data packet for read. In general an acknowledgment packet will contain the block number of the data packet being acknowledged. Each data packet has associated with it a block number; block numbers are consecutive and begin with one. Since the positive response to a write request is an acknowledgment packet, in this special case the block number will be zero. (Normally, since an acknowledgment packet is acknowledging a data packet, the acknowledgment packet will contain the block number of the data packet being acknowledged.) If the reply is an error packet, then the request is denied for the reason stated in the error packet.

Normal Termination
The end of a transfer is marked by a DATA packet that contains between 0 and 511 bytes of data (i.e. Datagram length < 516). This packet is acknowledged by an ACK packet like all other DATA packets. The final ACK packet is never retransmitted; the host acknowledging the final DATA packet may terminate its side of the connection on sending the final ACK. On the other hand, the host sending the last DATA must retransmit it until the packet is acknowledged or the sending host times out. If the response is an ACK, the transmission was completed successfully. If it is an ERROR (unknown transfer ID), or the sender of the data times out and is not prepared to retransmit any more, the transfer may still have been completed successfully, after which the acknowledger may have experienced a problem. It is also possible in this case that the transfer was unsuccessful. In any case, the connection has been closed.

Premature Termination
If a request cannot be granted, or some error occurs during the transfer, then an ERROR packet (opcode 5) is sent. This is only a courtesy since it will not be retransmitted or acknowledged, so it may never be received. Timeouts must also be used to detect errors.

Question55 : Explain Flow Control in TFTP.
Answer : TFTP sends a block of data using the DATA message and waits for an ACK message. If the sender receives an acknowledgment before the time-out, it sends the next block. Thus, flow control is achieved by numbering the data blocks and waiting for an ACK before the next data block is sent.

Question56 : Explain Error Control in TFTP.
Answer :The TFTP error-control mechanism is different from those of other protocols. It is symmetric,which means that the sender and the receiver both use time-outs. The sender uses a time-out for data messages; the receiver uses a time-out for acknowledgment messages. If a data message is lost, the sender retransmits it after time-out expiration. If an acknowledgment is lost, the receiver retransmits it after time-out expiration. This guarantees a smooth operation.Error control is needed in four situations: a damaged message, a lost message, a lost acknowledgment, or a

duplicated message.

Damaged Message

There is no negative acknowledgment. If a block of data is damaged, it is detected by the receiver and the block is discarded. The sender waits for the acknowledgment and does not receive it within the time-out period. The block is then sent again. Note that there is no checksum field in the DATA message of TFTP. The only way the receiver can detect data corruption is through the checksum field of the UDP user datagram.

Lost Message

If a block is lost, it never reaches the receiver and no acknowledgment is sent. The sender resends the block after the time-out.

Lost Acknowledgment

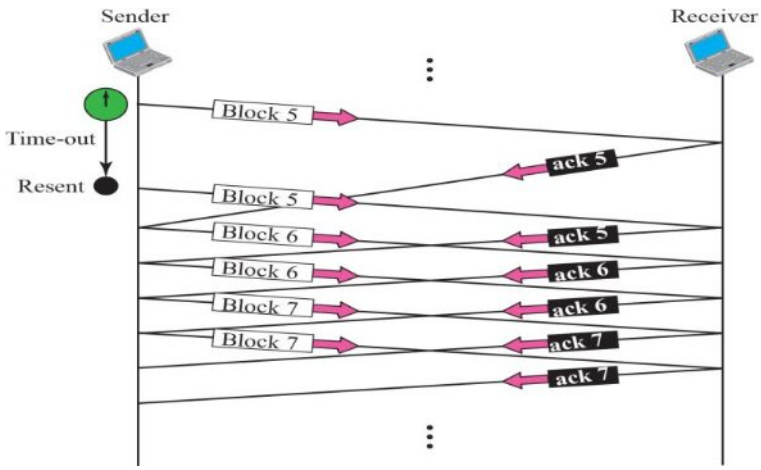
If an acknowledgment is lost, we can have two situations. If
the timer of the receiver matures before the timer of the sender, the receiver retransmits the acknowledgment; otherwise, the sender retransmits the data.

Duplicate Message

Duplication of blocks can be detected by the receiver through
block number. If a block is duplicated, it is simply discarded by the receiver.

Q 57. **Elaborate on Sorcerer's Apprenctice Bug.**

Ans. The Symmetric error and flow control mechanism in TFTP leads to this problem. It happens if ACK (Acknowledgement) message for a packet is not lost but delayed. In this case every succeeding block is sent twice and every succeeding ACK is received twice.Eg.



In the given Example ACK for Block 5 is delayed. After the time out expiration, the sender retransmits Block 5, which will be acknowledged by the receiver again. The sender receives 2 ACK for Block 5, which triggers it to send Block 6 twice. Reciever receives Block 6 twice and again sends 2 ACK, which results in sending Block 7 twice. And so on.

Q 58. Explain Security issues in TFTP.

Ans. 1) TFTP client is not asked to enter any username/password to download or upload a file from a TFTP server.

2) The TFTP protocol does not support any authentication or encryption mechanism, and as such can introduce a security risk when present. Installing the TFTP client is not recommended for systems that access the Internet.
Solution: Limited access of TFTP to non critical files which is achieved by implementing security in router close to TFTP server, which allows only certain hosts to access the server.

Q 59. Write applications of TFTP.

Ans. 1. TFTP is very useful for basic file transfer where security is not a big issue.
2. It can be usedto initialize devices such as bridges or router
3. Its main application is in conjunction with the DHCP. TFTP requires only a small amount of memory and uses only the services of UDP and IP. It can easily be configured in ROM (or PROM). When the station is powered on, TFTP will be connected to a server and can download the configuration files from there. Figure below shows the idea. The powered-on station uses the DHCP client to get the name of the configuration file from the DHCP server. The station then passes the name of the file to the TFTP client to get the contents of the configuration filefrom the TFTP server.

Q 60. Why do we need RRQ or WRQ message in TFTP but not in FTP ?
Ans. TFTP relies on UDP whereas FTP relies on TCP because there is no provision for connection establishment and termination in UDP, UDP transfers each block of data encapsulated in an independent user datagram. In TFTP, however, we do not want to transfer only one block of data; we do not want to transfer the file as independent blocks either. We need connections for the blocks of data being transferred if they all belong to the same file. TFTP uses RRQ, WRQ, ACK, and ERROR messages to establish connection. However FTP does not need it because it relies on TCP which itself is connection oriented.

Reading : To establish a connection for reading, the TFTP client sends the RRQ message. The name of the file and the transmission mode is defined in this message. If the server can transfer the file, it responds positively with a DATA message containing the first block of data. If there is a problem, such as difficulty in opening the file or permission restriction, the server responds negatively by sending an ERROR message.

Writing : To establish a connection for writing, the TFTP client uses the WRQ message. The name of the file and the transmission mode is defined in this message. If the server can accept a copy of the file, it responds positively with an ACK message using a value of 0 for the block number. If there is any problem, the server responds negatively by sending an ERROR message.