MODULE – 4

1.Repeaters

A repeater is a device that operates only in the physical layer. Signals that carry information within a network can travel a fixed distance before attenuation endangers the integrity of the data. A repeater receives a signal and, before it becomes too weak or corrupted, regenerates the original bit pattern. The repeater then sends the refreshed signal. A repeater can extend the physical length of a LAN, as shown in Figure 15.2

A repeater forwards every frame; it has no filtering capability. A repeater is a connecting device that operates in the physical layer of the Internet model. A repeater regenerates a signal, connects segments of a LAN, and has no filtering capability. A repeater is a regenerator, not an amplifier.

2.Active hub

An active hub is actually a multipart repeater. It is normally used to create connections between stations in a physical star topology.

3.BRIDGE

A bridge does not change the physical (MAC) addresses in a frame. A bridge is a connecting device that operates in the physical and data link layers of the Internet model. A bridge can use the spanning tree algorithm to create a loopless topology.

Transparent Bridges

A transparent bridge is a bridge in which the stations are completely unaware of the bridge's existence. A transparent bridge can forward and filter frames and automatically build its forwarding table If a bridge is added or deleted from the system, reconfiguration of the

stations is unnecessary. According to the IEEE 802.1 specification, a system equipped with transparent bridges must meet three criteria:

I. Frames must be forwarded from one station to another.

2. The forwarding table is automatically made by learning frame movements in the network.
3. Loops in the system must be prevented.

4.Routers

A router is a three-layer device that routes packets based on their logical addresses (host-to-host addressing). A router normally connects LANs and WANs in the Internet and has a routing table that is used for making decisions about the route. The routing tables are normally dynamic and are updated using routing protocols.

Three-Layer Switches A three-layer switch is a router, but a faster and more sophisticated. The switching fabric in a three-layer switch allows faster table lookup and forwarding.

5.Gateway:A gateway takes an application message, reads it, and interprets it. This means that it can be used as a connecting device between two internetworks that use different models. For example, a network designed to use the OSI model can be connected to another network using the Internet model. The gateway connecting the two systems can take a frame as it arrives from the first system, move it up to the OSI application layer, and remove the message.

SUMMARY

The five categories contain devices which can be defined as

1.Those which operate below the physical layer such as a passive hub

2. Those which operate at the physical layer (a repeater or an active hub).

3. Those which operate at the physical and data link layers (a bridge or a two-layer switch).

4. Those which operate at the physical, data link, and network layers (a router or a three-layer switch).

5. Those which can operate at all five layers (a gateway).

NETWORK LAYER(Host – to –Host communication)

The network layer is responsible for the delivery of individual packets from the source to the destination host.

Network Layer : Logical Addressing

The network layer at the source is responsible for creating a packet from the data coming from another protocol (such as a transport layer protocol or a routing protocol). The header of the packet contains, among other information, the logical addresses of the source and destination. The network layer is responsible for checking its routing table to find the routing information (such as the outgoing interface of the packet or the physical address the next node). If the packet is too large, the packet is fragmented .

The network layer at the switch or router is responsible for routing the packet. When a packet arrives, the router or switch consults its routing table and finds the interface from which the packet must be sent. The packet, after some changes in the header, with the routing infonnation is passed to the data link layer again. • The network layer at the destination is responsible for address verification; it makes sure that the destination address on the packet is the same as the address of the host. If the packet is a fragment, the network layer waits until all fragments have arrived, and then reassembles them and delivers the reassembled packet to the transport layer.

Internet as a Datagram Network .DISCUSS? 5 MARK

The Internet, at the network layer, is a packet-switched network. Packet switching uses either the virtual circuit approach or the datagram approach. The Internet has chosen the datagram approach to switching in the network layer. It uses the universal addresses defined in the network layer to route packets from the source to the destination.

Switching at the network layer in the Internet uses the datagram approach to packet switching.

Delivery of a packet can be accomplished by using either a connection-oriented or a connectionless network service.

In a connection-oriented service, the source first makes a connection with the destination before sending a packet. When the connection is established, a sequence of packets from the same source to the same destination can be sent one after another. In this case, there is a relationship between packets. They are sent on the same path in sequential order. A packet is logically connected to the packet traveling before it and to the packet traveling after it. When all packets of a message have been delivered, the connection is terminated. In a connection-oriented protocol, the decision about the route of a sequence of packets with the same source and destination addresses can be made only once, when the connection is established. Switches do not recalculate the route for each individual packet. This type of service is used in a virtual-circuit approacH. to packet switching such as in Frame Relay and ATM.

In conneetionless service, the network layer protocol treats each packet independently, with each packet having no relationship to any other packet. The packets in a message mayor may not travel the same path to their destination. This type of service is used in the datagram approach to packet switching. The Internet has chosen this type of service at the network layer. The reason for this decision is that the Internet is made of so many heterogeneous networks that it is almost impossible to create a connection from the source to the destination without knowing the nature of the networks in advance.

Communication at the network layer in the Internet is connectionless.

EXPLAIN IPv4 ADDRESSES ?(15 MARKS)

An IPv4 Address is a 32- bit address that uniquely and universally defines the connection of a device(for eg:a computer or a router) to the internet.

❖ The IPv4 addresses are unique and universal

They are unique in the sense that each address definesone, and only one, connection to the Internet. Universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

❖ An IPv4 address is 32 bits long.ie,the address space of IPv4 is 2^{32}.

Address Space

An address spaceis the total number of addresses used by the protocol. If a protocol uses *N* bits to define an address, the address space is 2^N.

Notations

There are two notations to show an IPv4 address:

- binary notation
- dotted decimal notation.

Binary Notation

In binary notation, the IPv4 address is displayed as 32 bits. Each octet is often referred to as a byte. So it is common to hear an IPv4 address referred to as a 32-bit address or a 4-byte address.

Eg: 01110101 10010101 00011101 00000010

Dotted-Decimal Notation

Internet addresses are written in decimal form with a decimal point (dot) separating the bytes to make the IPv4 address more

compact and easier to read.

Eg: 117.149.29.2

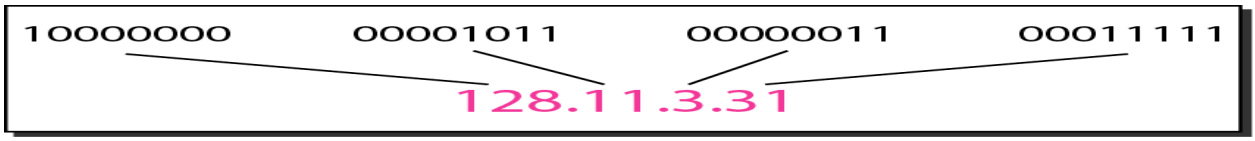
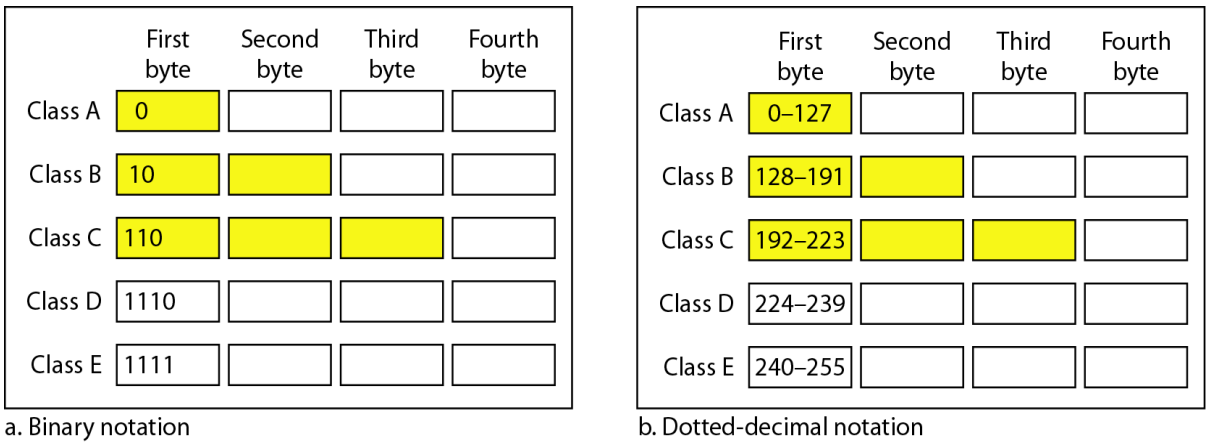


Fig: Dotted-decimal notation and binary notation for an IPv4 address

Classful Addressing

In classful addressing, the address space is divided into five classes: A, B, C, D,and E. Each class occupies some part of the address space.

The classes in binary and dotted-decimal notation



Classes and Blocks

In classful addressing is that each class is divided into a fixed number of blocks with each block having a fixed size .

- Class A addresses were designed for large organizations with a large number of attached hosts or routers.
- Class B addresses were designed for midsize organizations with tens of thousands of attached hosts or routers.

- Class C addresses were designed for small organizations with a small number of attached hosts or routers.
- Class D addresses were designed for multicasting.

- The class E addresses were reserved for future use.

Netid and Hostid

In classful addressing, an IP address in class A, B, or C is divided into netid and hostid.These parts are of varying lengths, depending on the class of the address.
In class A, one byte defines the netid and three bytes define the hostid. In class B, two bytes define the netid and two bytes define the hostid. In class C, three bytes define the netid and one byte defines the hostid.

Subnetting

It is the process of divide a large address block into smaller contiguous subgroups and assign each group to smaller networks(subnet).

Supernetting

In supernetting,several networks are combined to create a supernetwork or a supernet.ie,To combine several contiguous address spaces into a larger single address space.

Classless Addressing To overcome address depletion and give more organizations access to the Internet,classless addressing implemented. In this scheme, there are noclasses, but the addresses are still granted in blocks.

Restriction -To simplify the handling of addresses, the Internet authorities impose three restrictions on classless address blocks:

1. The addresses in a block must be contiguous, one after another.
2. The number of addresses in a block must be a power of 2 (I, 2, 4, 8, ...).
3. The first address must be evenly divisible by the number of addresses.

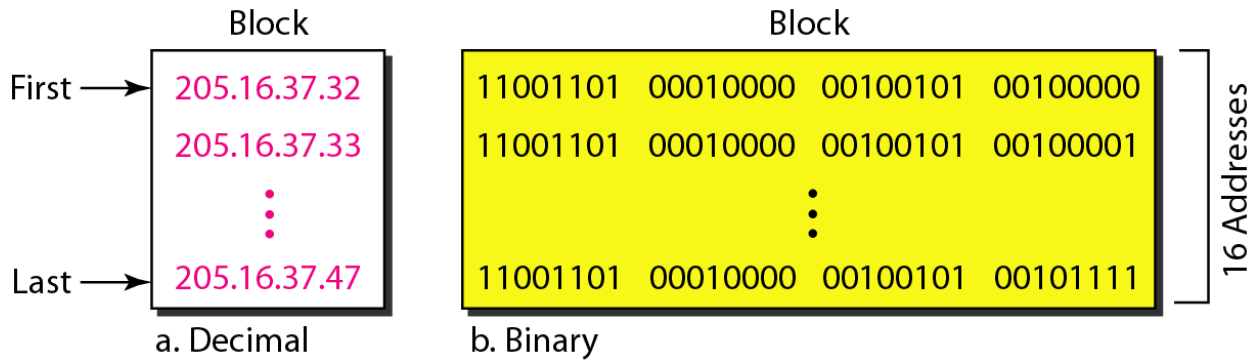


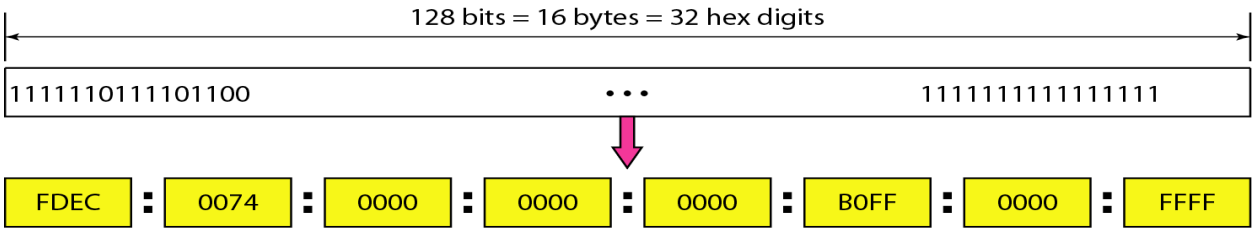
Fig: A block of 16 addresses granted to a small organization

EXPLAIN IPv6 ADDRESSES(15 MARKS)

An IPv6 address consists of 16 bytes (octets); it is 128 bits long.To make addresses more readable, IPv6 specifies hexadecimal colon notation. In this notation,

128 bits is divided into eight sections, each 2 bytes in length. The address consists of 32 hexadecimal digits, with every four digits separated by a colon.

IPv6 address in binary and hexadecimal colon notation



Address Space

IPv6 has a much larger address space; 2^{128} addresses are available. The designers of IPv6 divided the address into several categories.

- **Unicast Addresses**

A **unicast address** defines a single computer. The packet sent to a unicast address must be delivered to that specific computer. IPv6 defines two types of unicast addresses: geographically based and provider-based.

- **Multicast Addresses**

Multicast addresses are used to define a group of hosts instead of just one. A packet sent to a multicast address must be delivered to each member of the group.

- **Anycast Addresses**

An anycast address, defines a group of nodes.A packet destined for an anycast address is delivered to only one of the members of the anycast group.

❖ **Reserved Addresses**

Another category in the address space is the reserved address. These addresses start with eight O's (type prefix is 00000000). A few subcategories are:

An ***unspecified address*** is used when a host does not know its own address and sends an inquiry to find its address. A ***loopback address*** is used by a host to test itself without going into the network. A ***compatible address*** is used during the transition from IPv4 to IPv6.A ***mapped address*** is used when a computer that has migrated to IPv6 wants to send a packet to a computer still using IPv4.

❖ **Logical Addresses:**

These addresses are used when an organization wants to use IPv6 protocol without being connected to the global Internet. In other words, they provide addressing for private networks.Nobody outside the organization can send a message to the nodes using these addresses.

NETWORK LAYER : INTERNET PROTOCOL

The Internet Protocol version 4 (IPv4) is the delivery mechanism used by the TCP/IP protocols. IPv4 is an unreliable and connectionless datagram protocol-a best-effort delivery service. The term *best-effort* means that IPv4 provides no error control or flow control.

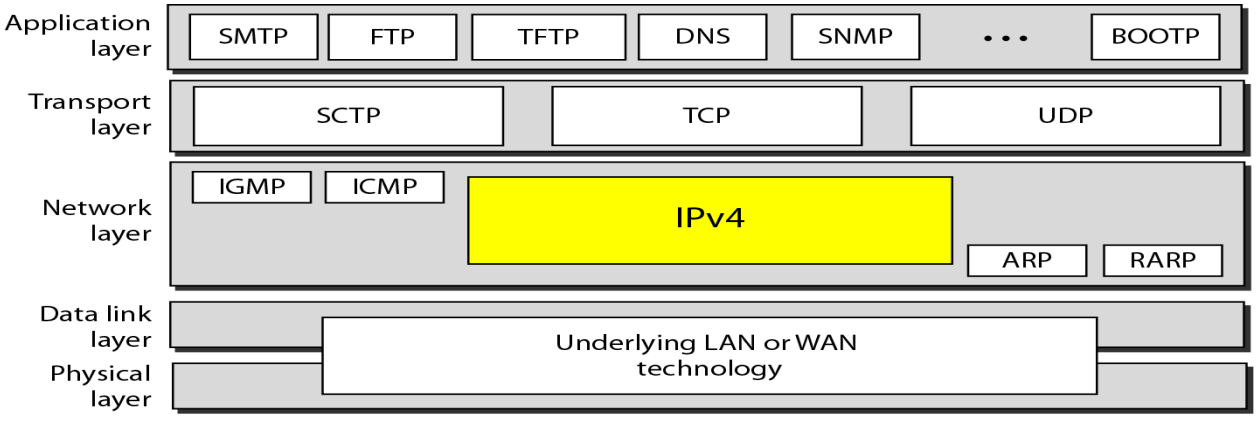
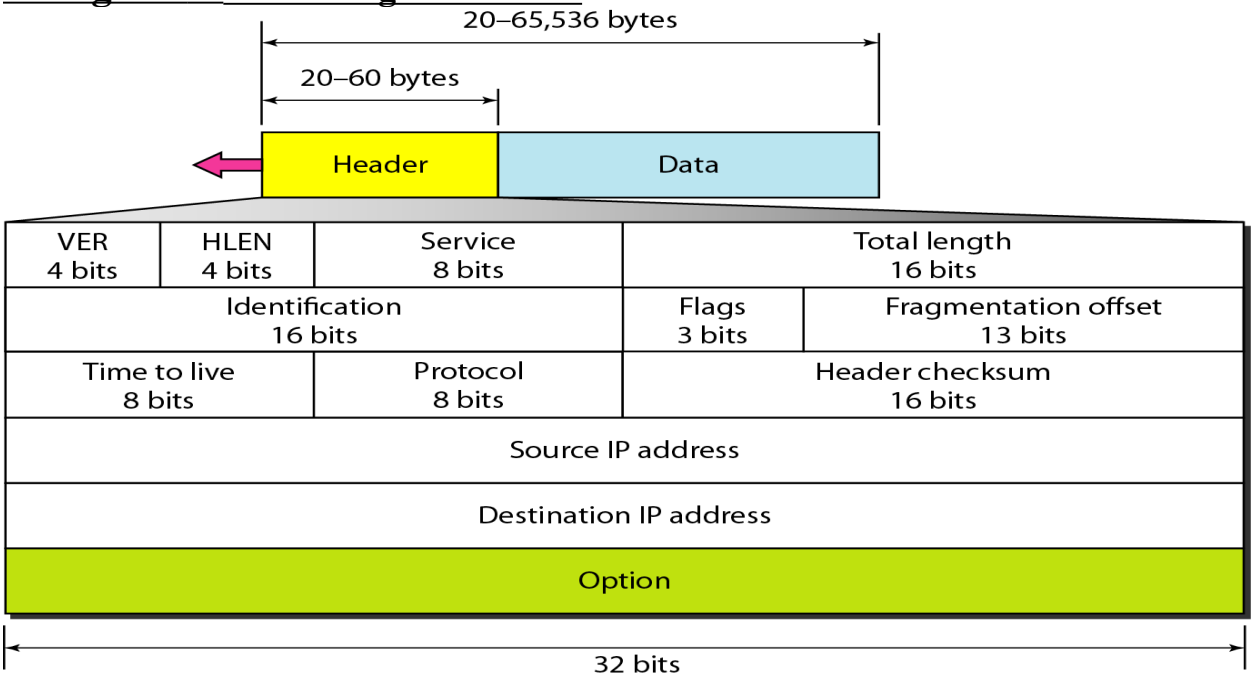


Fig: *Position of IPv4 in TCP/IP protocol suite*

Datagram - IPv4 datagram format



Packets in the IPv4 layer are called datagrams. A datagram is a variable-length packet consisting of two parts: header and data.The header is 20 to 60 bytes in length and contains information essential to routing and delivery.

IPv4 Datagram Format

- Version (VER) - This 4-bit field defines the version of the IPv4 protocol. Currentlythe version is 4.
- Header length (HLEN) - This 4-bit field defines the total length of the datagram header in 4-byte words.
- Services - service type or differentiated services (not used now).
- Total length - This is a In-bit field that defines the total length (header plus data) of the IPv4 datagram in bytes.
  Total length of data = total length – header length
- Identification - This field is used in fragmentation
- Flags - This field is used in fragmentation
- Fragmentation offset - This field is used in fragmentation

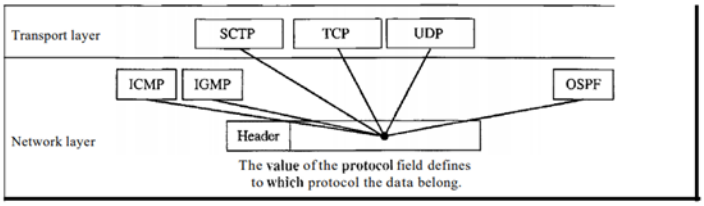
- Time to live(TTL) :

  This field is used to control the maximum number of hops (routers) visited by the datagram. When a source host sends the datagram, it stores a number in this field. This value is approximately 2 times the maximum number of routes between any two hosts. Each router that processes the datagram decrements this number by 1. If this value, after being decremented, is zero, the router discards the datagram. Another use of this field is to intentionally limit the journey of the packet. For example, if the source wants to confine the packet to the local network, it can store 1 in this field. When the packet arrives at the first router, this value is decremented to 0, and the datagram is discarded.

- Protocol :

  This 8-bit field defines the higher-level protocol that uses the services of the IPv4 layer. since the IPv4 protocol carries data from different other protocols, the value of this field helps the receiving network layer know to which protocol the data belong.

Figure 20.8 *Protocol field and encapsulated data*



The value of this field for each higher-level protocol is shown in Table 20.4.

- Source address :

  This 32-bit field defines the IPv4 address of the source. This field must remain unchanged during the time the IPv4 datagram travels from the source host to the destination host.

- Destination address :

This 32-bit field defines the IPv4 address of the destination. This field must remain unchanged during the time the IPv4 datagram travels from the source host to the destination host.

- Options :

The header of the IPv4 datagram is made of two parts: a fixed palt and a variable part.The fixed part is 20 bytes long, The variable part comprises the options

that can be a maximum of 40 bytes. Options, as the name implies, are not required for a datagram. They can be used for network testing and debugging.

No Operation - A no-operation option is a 1-byte option used as a filler between options.

End of Option - An end-of-option option is a 1-byte option used for padding at the end of the option field.

Record Route - A record route option is used to record the Internet routers that handle the datagram.

Strict Source Route - A strict source route option is used by the source to predetermine a route for the datagramas it travels through the Internet.

Loose Source Route - Each router in the list must be visited, but the datagram can visit other routers as well.

Timestamp - A timestamp option is used to record the time of datagram processing by a router.

The MTU is the maximum number of bytes that a data link protocol can encapsulate. MTUs vary from protocol to protocol.

Fragmentation is the division of a datagram into smaller units to accommodate the MTU of a data link protocol. To make the IPv4 protocol independent of the physical network, the designers decided to make the maximum length of the IPv4 datagram equal to 65,535 bytes. This makes transmission more efficient if we use a protocol with an MTU of this size. However, for other physical networks, we must divide the datagram to make it possible to pass through these networks. This is called **fragmentation**. The source usually does not fragment the IPv4 packet. The transport layer will instead segment the data into a size that can be accommodated by IPv4 and the data link layer in use.

IPv6 (Internetworking Protocol, version 6), also known as **IPng** (Internetworking Protocol, next generation), was proposed and is now a standard.

IPv6, the latest version of the Internet Protocol, has a 128-bit address space, a revised header format, new options, an allowance for extension, support for resource allocation, and increased security measures.An IPv6 datagram is composed of a base header and a payload. Extension headers add functionality to the IPv6 datagram.

Advantages

The next-generation IP, or IPv6, has some advantages over IPv4 that can be summarized as follows:

- **Larger address space**.

  An IPv6 address is 128 bits long, Compared with the 32-bit address of IPv4, this is a huge (2^{96}) increase inthe address space.

- **Better header format.**

  IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the upper-layer data. This simplifies and speeds up the routing process .

- **New options.**

IPv6 has new options to allow for additional functionalities.

- **Allowance for extension**.

  IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.

- **Support for resource allocation**.

  In IPv6, the type-of-service field has been removed, but a mechanism (called Flow *label)* has been added to enable the source to request special handling of the packet. This mechanism can be used to support traffic such as real-time audio and video.

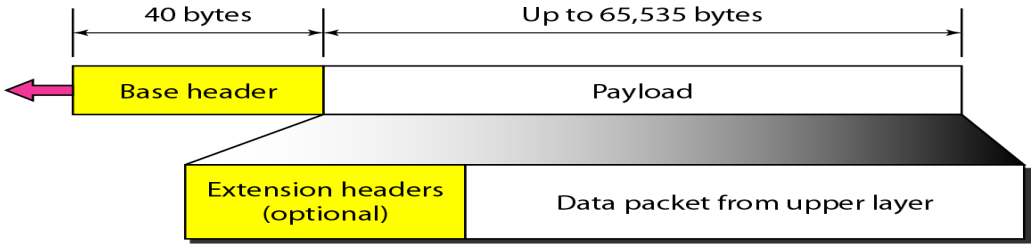
- **Support for more security.**

  The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet.

Packet Format

The IPv6 packet is composed of a mandatory base header followed by the payload. The payload consists of two parts: optional extension headers and data from an upper layer. The base header occupies 40 bytes, whereas the extension headers and data from the upper layer contain up to 65,535 bytes of information.

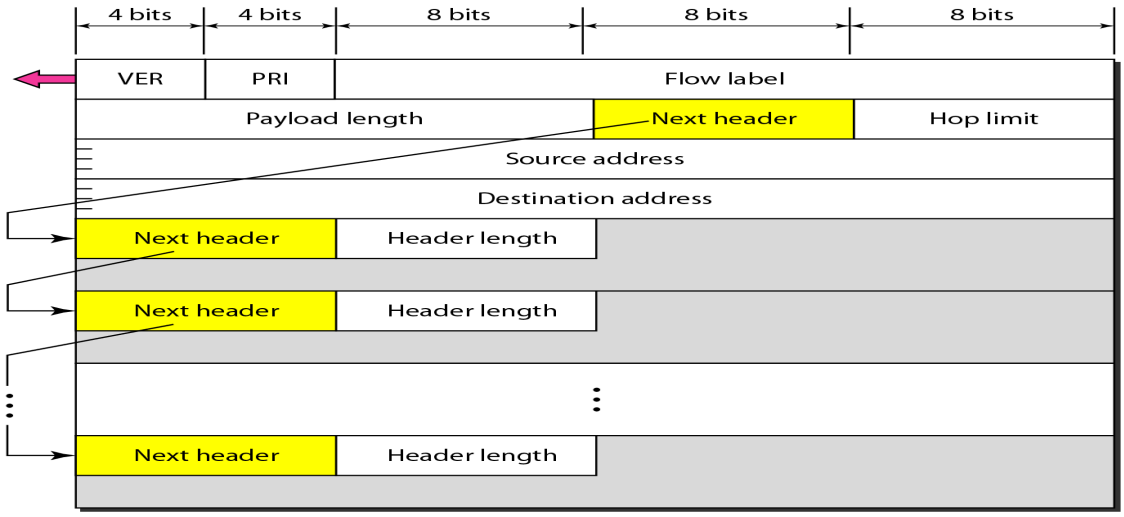
IPv6 datagram header and payload



Base Header

The below figure shows the base header with its eight fields.

Format of an IPv6 datagram



These fields are as follows:

- **Version.**

  This 4-bit field defines the version number of the IP. For IPv6, the value is 6.

- **Priority**.

  The 4-bit priority field defines the priority of the packet with respect to traffic congestion

- **Flow label**.

  The flow label is a 3-byte (24-bit) field that is designed to provide special handling for a particular flow of data.

- **Payload length**. Explain? (2marks)

  The 2-byte payload length field defines the length of the IP datagram excluding the base header.

- **Next header**.

  The next header is an 8-bit field defining the header that follows the base header in the datagram. The next header is either one of the optional extension headers used by IP or the header of an encapsulated packet such as UDP or TCP. Each extension header also contains this field.

- **Hop limit.**

  This 8-bit hop limit field serves the same purpose as the TTL field in IPv4.

- **Source address.**

  The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram.

- **Destination address.**

  The destination address field is a 16-byte (128-bit) Internet address that usually identifies the final destination of the datagram.

A sequence of packets, sent from a particular source to a particular destination, that needs special handling by routers is called a *flow* of packets. The combination of the source address and the value of the *flow label* uniquely defines a flow of packets.

A flow label can be used to speed up the processing of a packet by a router. When a router receives a packet, instead of consulting the routing table and going through a routing algorithm to define the address of the next hop, it can easily look in a flow label table for the next hop.

A flow label can be used to support the transmission of real-time audio and video. Real-time audio or video, particularly in digital form, requires resources such as high bandwidth, large buffers, long processing time, and so on. A process can make a reservation for these resources beforehand to guarantee that real-time data will not be delayed due to a lack of resources. The use of real-time data and the reservation of these resources require other protocols such as Real-Time Protocol (RTP) and Resource Reservation Protocol (RSVP) in addition to IPv6.

PROCESS-TO-PROCESS DELIVERY

The data link layer is responsible for delivery of frames between two neighboring nodes over a link. This is called *node-to-node delivery.* The network layer is responsible for delivery of datagrams between two hosts. This is called *host-to-host delivery.*

The transport layer is responsible for **process-to-process delivery**-the delivery of a packet, part of a message, from one process to another.

Connectionless Versus Connection-Oriented Service

A transport layer protocol can either be connectionless or connection-oriented.

Connectionless Service

In a connectionless service, the packets are sent from one party to another with no need for connection establishment or connection release. The packets are not numbered; they may be delayed or lost or may arrive out of sequence. There is no acknowledgment either.

Connection Oriented Service

In a connection-oriented service, a connection is first established between the sender and the receiver. Data are transferred. At the end, the connection is released.

In a connection-oriented service, the source first makes a connection with the destination before sending a packet. When the connection is established, a sequence of packets from the same source to the same destination can be sent one after another. In this case, there is a relationship between packets. They are sent on the same path in sequential order. A packet is logically connected to the packet traveling before it and to the packet traveling after it. When all packets of a message have been delivered, the connection is terminated.

Reliable Versus Unreliable

The transport layer service can be reliable or unreliable. If the application layer program needs reliability, we use a reliable transport layer protocol by implementing flow and error control at the transport layer. This means a slower and more complex service. On the other hand, if the application program does not need reliability because it uses its own flow and error control mechanism or it needs fast service or the nature of the service does not demand flow and error control (real-time applications), then an unreliable protocol can be used.

Transmission Control Protocol (TCP)

TCP is called a *connection-oriented, reliable* transport protocol. It adds connection-oriented and reliability features to the services of IP. TCP, is a process-to-process (program-to-program) Protocol.

TCP Services

The services offered by TCP to the processes at the application layer are:

- ***Process-to-Process Communication***

  TCP provides process-to-process communication using port numbers.

- ***Stream Delivery Service***

  TCP, is a stream-oriented protocol. TCP, 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 creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their data across the Internet.

- ***Sending and Receiving Buffers***

  Because the sending and the receiving processes may not write or read data at the same speed, TCP needs buffers for storage. There are two buffers, the sending buffer and the receiving buffer, one for each direction.

- ***Segments***

  In IP layer TCP, needs to send data in packets, not as a stream of bytes. At the transport layer, TCP groups a number of bytes together into a packet called a ***segment***.TCP adds a header to each segment (for control purposes) and delivers the segment to the IP layer for transmission. The segments are encapsulated in IP datagrams and transmitted.This entire operation is transparent to the receiving process.

- ***Full-Duplex Communication***

  TCP offers full-duplex service, in which data can flow in both directions at the same time.

- ***Connection-Oriented Service***

  TCP, unlike UDP, is a connection-oriented protocol. When a process at site A wants to send and receive data from another process at site B, the following occurs:

  1. The two TCPs establish a connection between them.

  2. Data are exchanged in both directions.

  3. The connection is terminated.

- ***Reliable Service***

TCP is a reliable transport protocol. It uses an acknowledgment mechanism to check the safe and sound arrival of data.

TCP Features

- ***Numbering System***

  *T*here are two fields called the sequence number and the acknowledgment number.

Byte Number - TCP numbers all data bytes that are transmitted in a connection.

The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.

Sequence Number - After the bytes have been numbered, TCP assigns a sequence number to each segment that is being sent. The sequence number for each segment is the number of the first byte carried in that segment.

Acknowledgment Number - The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive. The acknowledgment number is cumulative.

- ***Flow Control***

  TCP, provides *flow control.* The receiver of the data controls the amount of data that are to be sent by the sender. This is done to prevent the receiver from being overwhelmed with data. The numbering system allows TCP to use a byte-oriented flow control.

- ***Error Control***

  To provide reliable service, TCP implements an error control mechanism.

  Although error control considers a segment as the unit of data for error detection

  (loss or corrupted segments), error control is byte-oriented.

- ***Congestion Control :*** TCP, takes into account congestion in the network. The amount of data sent by a sender is not only controlled by the receiver (flow control), but is also determined by the level of congestion in the network.

TCP Segment

The segment consists of a 20- to 60-byte header, followed by data from the application program. The header is 20 bytes if there are no options and up to 60 bytes if it contains options.

Format

The format of a segment is shown in Figure:

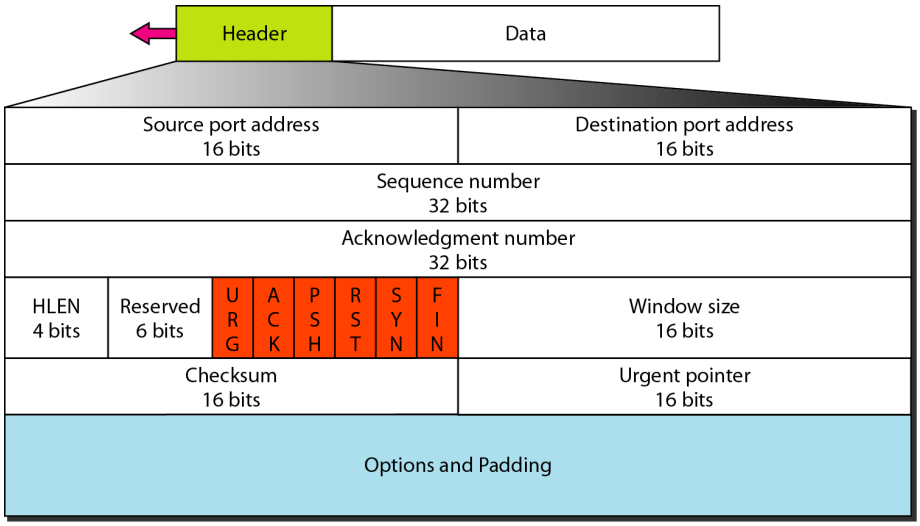


Fig:TCP segment format

- 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.

- 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

- Sequence number.

  This 32-bit field defines the number assigned to the first byte of data contained in this segment.

- 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 defines *x* + I as the acknowledgment number.

- Header length.

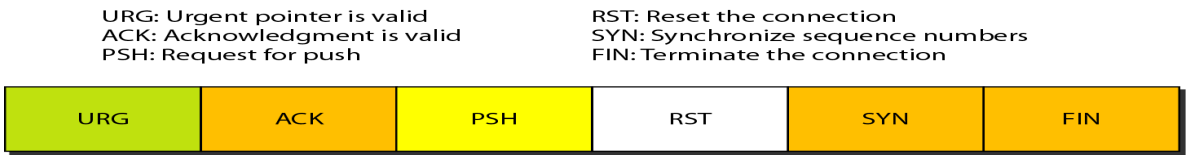
  This 4-bit field indicates the number of 4-byte words in the TCP header. The length of the header can be between 20 and 60 bytes

- Reserved.

  This is a 6-bit field reserved for future use.

- Control field.

  This field defines 6 different control bits or flags One or more of these bits can be set at a time.



- Window size.

  This field defines the size of the window, in bytes, that the other party must maintain.

- Checksum.

  This 16-bit field contains the checksum.

A TCP Connection

TCP is connection-oriented. A connection-oriented transport protocol establishes a virtual path between the source and destination. All the segments belonging to a message are then sent over this virtual path. Using a single virtual pathway for the entire

message facilitates the acknowledgment process as well as retransmission of damaged

or lost frames.

In TCP, connection-oriented transmission requires three phases:

- connection establishment
- data transfer
- connection termination.

Connection Establishment

TCP transmits data in full-duplex mode. When two TCPs in two machines are connected, they are able to send segments to each other simultaneously. This implies that each party must initialize communication and get approval from the other party before any data are transferred.

- **Three-Way Handshaking:**

The connection establishment in TCP is called three way handshaking. In our example, an application program, called the client, wants to make a connection with another application program, called the server, using TCP as the transport layer protocol.

The process starts with the server. The server program tells its TCP that it is ready

to accept a connection. This is called a request for a *passive open.* Although the server

TCP is ready to accept any connection from any machine in the world, it cannot make the connection itself.

The client program issues a request for an *active open.* A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server. TCP can now start the three-way handshaking process as shown in Figure:

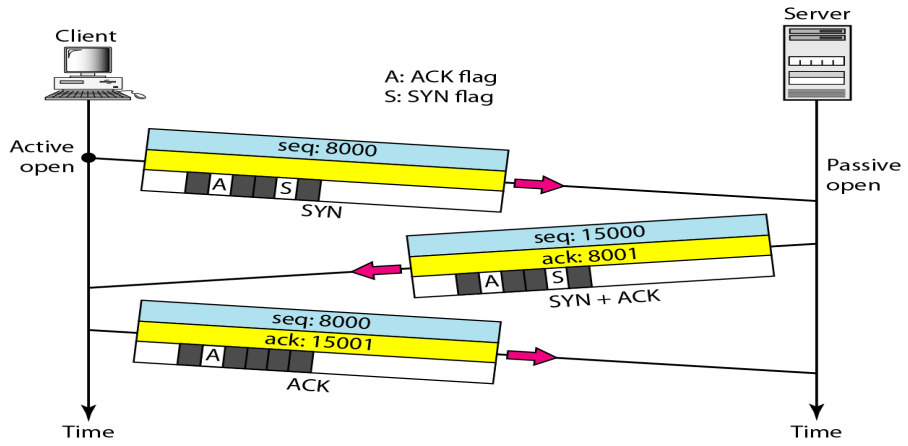


Fig: *Connection establishment using three-way handshaking*

The three steps in this phase are as follows:

1. The client sends the first segment, a SYN segment, in which only the SYN flag is set.This segment is for synchronization of sequence numbers. It consumes one sequence number. **A SYN segment cannot carry data, but it consumes one sequence number.**

2. The server sends the second segment, a SYN +ACK segment, with 2 flag bits set: SYN and ACK. **A SYN + ACK segment cannot carry data, but does consume one sequence number.**

3. The client sends the third segment. This is just an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. **An ACK segment, if carrying no data, consumes no sequence number.**

Data Transfer

After connection is established, bidirectional data transfer can take place. The client and server can both send data and acknowledgments. The acknowledgment is piggybacked with the data. Below Figure shows an example.

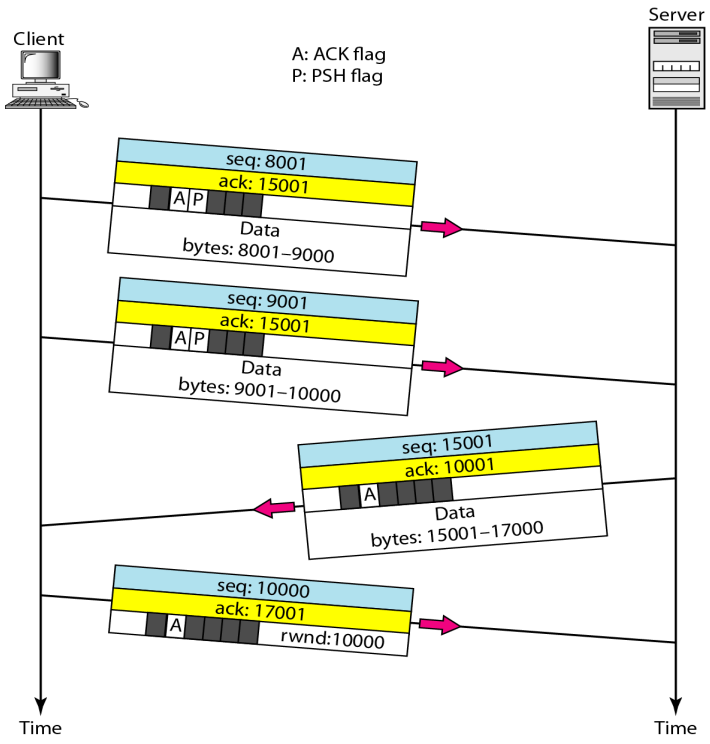


Fig:Data Transfer

In this example, after connection is established (not shown in the figure), the client sends 2000 bytes of data in two segments. The server then sends 2000 bytes in one segment.The client sends one more segment. The first three segments carry both data and acknowledgment, but the last segment carries only an acknowledgment because there are no more data to be sent.

Connection Termination

Any of the two parties involved in exchanging data (client or server) can close the connection,it is usually initiated by the client.

- **Three-Way Handshaking in TCP.EXPLAIN (5 MARKS)**

  Most implementations today allow *three-way handshaking with half close* for connection termination as shown in Figure.

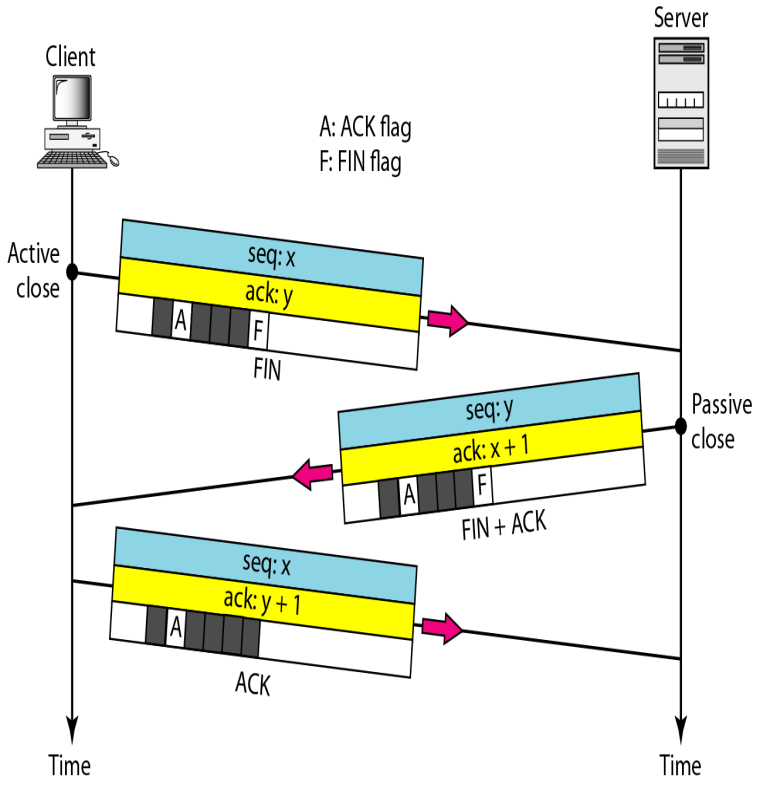


Fig:Connection termination using three – way handshaking

1. In a normal situation, the client TCP, after receiving a close command from the client process, sends the first segment, a FIN segment in which the FIN flag is set.

   The FIN segment consumes one sequence number ifit does not carry data

2. The server TCP, after receiving the FIN segment, informs its process of the situation and sends the second segment, a FIN +ACK segment, to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the

connection in the other direction. This segment can also contain the last chunk of data from the server. **The FIN +ACK segment consumes one sequence number ifit does not carry data.**

4. The client TCP sends the last segment, an ACK segment, to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is 1 plus the sequence number received in the FIN segment from the server. This segment cannot carry data and consumes no sequence numbers.

SYN Flooding Attack

The connection establishment procedure in TCP is susceptible to a serious security problem called the SYN flooding attack. This happens when a malicious attacker sends a large number of SYN segments to a server, pretending that each of them is corning from a different client by faking the source IP addresses in the datagrams. The server, assuming that the clients are issuing an active open, allocates the necessary resources, such as creating communication tables and setting timers. The TCP server then sends the SYN +ACK segments to the fake clients, which are lost. During this time, however, a lot ofresources are occupied without being used. If, during this short time, the number of SYN segments is large, the server eventually runs out of resources and may crash. This SYN flooding attack belongs to a type of security attack known as a denial-of-service attack, in which an attacker monopolizes a system with so many service requests that the system collapses and denies service to every request. Some implementations of TCP have strategies to alleviate the effects of a SYN attack. Some have imposed a limit on connection requests during a specified period of time. Others filter out datagrams coming from unwanted source addresses. One recent strategy is to postpone resource allocation until the entire connection is set up, using what is called a cookie.

Pushing Data: consider an application program that communicates interactively with another application program on the other end. The application program on one site wants to send a keystroke to the application at the other site and receive an immediate response. Delayed

transmission and delayed delivery of data may not be acceptable by the application program. TCP can handle such a situation.

The application program at the sending site can request a push operation. This means that the sending TCP must not wait for the window to be filled. It must create a segment and send it immediately. The sending TCP must also set the push bit (PSH) to let the receiving TCP know that the segment includes data that must be delivered to the receiving application program as soon as possible and not to wait for more data to come.

URGENT FLAG

Suppose an application program is sending data to be processed by the receiving application program. When the result of processing comes back, the sending application program finds that everything is wrong. It wants to abort the process, but it has already sent a huge amount of data. Ifit issues an abort command (control +C), these two characters will be stored at the end ofthe receiving TCP buffer. It will be delivered to the receiving application program after all the data have been processed.

The solution is to send a segment with the URG bit set. The sending application program tells the sending TCP that the piece of data is urgent. The sending TCP creates a segment and inserts the urgent data at the beginning of the segment. The rest of the segment can contain normal data from the buffer. The urgent pointer field in the header defines the end of the urgent data and the start of normal data. When the receiving TCP receives a segment with the URG bit set, it extracts the urgent data from the segment, using the value of the urgent pointer, and delivers them, out of order, to the receiving application program

- **Half-Close**

In TCP, one end can stop sending data while still receiving data. This is called a half-close. Although either end can issue a half-close, it is normally initiated by the client. It can occur when the server needs all the data before processing can begin.

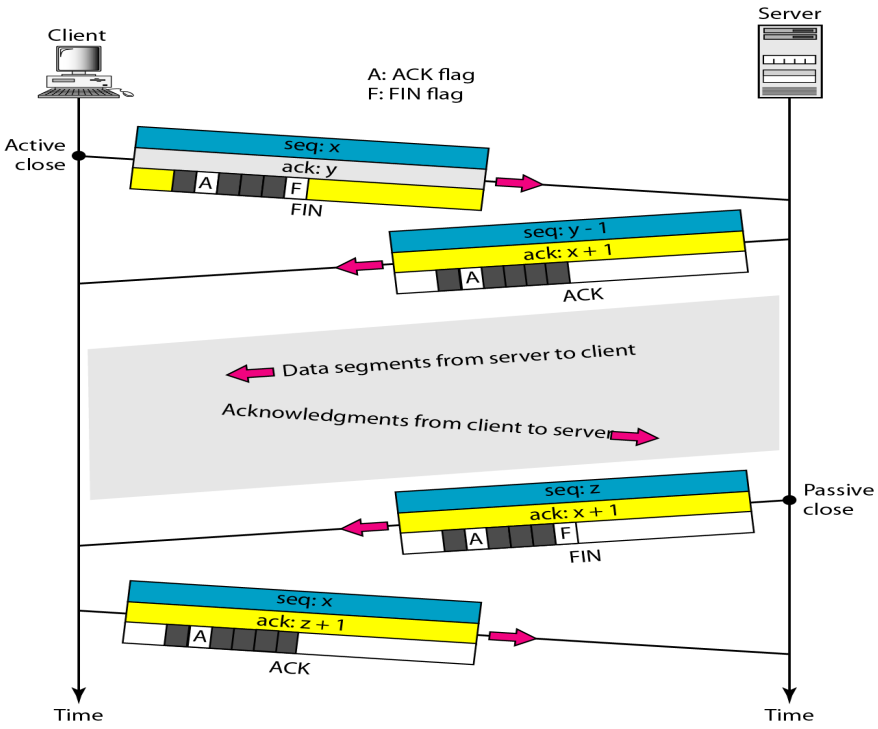


Fig: Half Close

Figure above shows an example of a half-close. The client half-closes the connection by sending a FIN segment. The server accepts the half-close by sending the ACK segment. The data transfer from the client to the server stops. The server, however, can still send data. When the server has sent all the processed data, it sends a FIN segment, which is acknowledged by an ACK for the client.

EXPLANATION:

A good example is sorting. When the client sends data to the server to be sorted, the server needs to receive all the data before sorting can start. This means the client, after sending all the data, can close the connection in the outbound direction. However, the inbound direction must remain open to receive the sorted data. The server, after receiving the data, still needs time for sorting; its outbound direction must remain open. Figure 23.21 shows an example of a half-close. The client half-closes the connection by sending a FIN segment. The server accepts the half-close by sending the ACK segment. The data transfer from the client to the

server stops. The server, however, can still send data. When the server has sent all the processed data, it sends a FIN segment, which is acknowledged by an ACK from the client. After half-closing of the connection, data can travel from the server to the client and acknowledgments can travel from the client to the server. The client cannot send any more data to the server. Note the sequence numbers we have used.

The second segment (ACK) consumes no sequence number. Although the client has received sequence number y - 1 and is expecting y, the server sequence number is still y - 1. When the connection finally closes, the sequence number of the last ACK segment is still x, because no sequence numbers are consumed during data transfer in that direction.

FLOW CONTROL

A sliding window is used to make transmission more efficient as weD as to control the flow of data so that the destination does not become overwhelmed with data. TCP sliding windows are byte-oriented. , the TCP's sliding window is of variable size; The imaginary window has two walls: one left and one right.

Opening a window means moving the right wall to the right. This allows more new bytes in the buffer that are eligible for sending. Closing the window means moving the left wall to the right. This means that some bytes have been acknowledged and the sender rneed not worry about them anymore. Shrinking the window means moving the right wall to the left.

.**Error Control**

TCP is a reliable transport layer protocol.TCP provides reliability using error control.Error control includes mechanisms for detecting corrupted segments,lost frames,out-of-order segments and duplicated segments.Error control also includes a mechanism for correcting errors after they are detected.

Error detection and correction in TCP is achieved through the use of three simple tools : *checksum,acknowledgement,and time out.*

- **Checksum**

Each segment includes a checksum field which is used to check for a corrupted segment.If the segment is corrupted,it is discarded by the destination TCP and is considered as lost.TCP uses a 16–bit checksum that is mandatory in every segment.

- **Acknowledgement**

  TCP uses acknowledgements to confirm the receipt of data segments.ACK segments do not consume sequence numbers and are not acknowledged.

- **Retransmission**

  A retransmission occurs if the retransmission timer expires or three duplicate ACK segments have arrived.No retransmission timer is set for an ACK segment.

- **Out-of-Order Segments**

Data may arrive out of order and be temporarily stored but the TCP,but TCP guarantees that no out-of-order segment is delivered to the process.

USER DATAGRAM PROTOCOL (UDP).EXPLAIN? (5 MARKS)

- The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.
- It performs very limited error checking.
- UDP is a very simple protocol using a minimum of overhead. If a process wants to send a small message and does not care much about reliability, it can use UDP.
- Sending a small message by using UDP takes much less interaction between the sender and receiver than using TCP or SCTP.

User Datagram

UDP packets, called user datagrams, have a fixed-size header of 8 bytes.

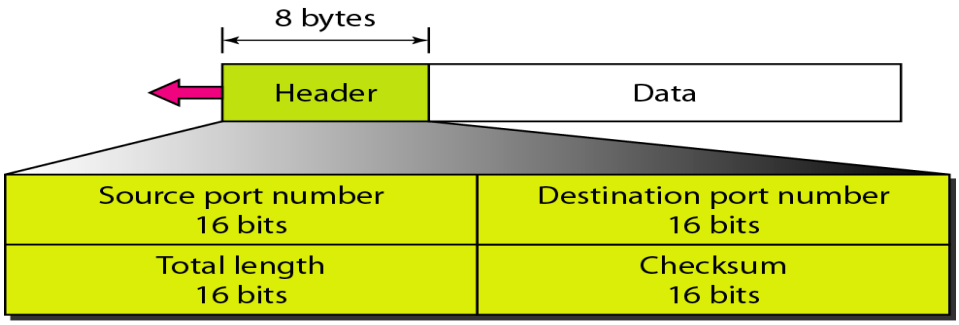


Fig:User Datagram Format

The fields are:

- **Source port number.**

This is the port number used by the process running on the source host. It is 16 bits long, which means that the port number can range from 0 to 65,535. If the source host is the client (a client sending a request), the port number, is an ephemeral port number requested by the process .If the source host is the server (a server sending a response), the port number, in most cases, is a well-known port number.

- **Destination port number**

This is the port number used by the process running on the destination host. It is also 16 bits long. If the destination host is the server (a client sending a request), the port number, in most cases, is a well-known port number. If the destination host is the client (a server sending a response), the port number, in most cases, is an ephemeral port number.

- **Length.**

This is a 16-bit field that defines the total length of the user datagram, header plus data.

UDP length = IP length - IP header's length

- **Checksum.**

This field is used to detect errors over the entire user datagram (header plus data).

UDP Operation

1. **Connectionless Services**

   - UDP provides a connectionless service. This means that each user datagram sent by UDP is an independent datagram. There is no relationship between the different user datagrams even if they are coming from the same source process and going to the same destination program.
   - The user datagrams are not numbered.
   - There is no connection establishment and no connection termination, as is the case for TCP. This means that each user datagram can travel on a different path.

2. **Flow and Error Control**

   - UDP is a very simple, unreliable transport protocol. There is no flow control and hence no window mechanism.
   - The receiver may overflow with incoming messages.
   - There is no error control mechanism in UDP except for the checksum. This means that the sender does not know if a message has been lost or duplicated. When the receiver detects an error through the checksum, the user datagram is silently discarded.
   - The lack of flow control and error control means that the process using UDP should provide these mechanisms.

3. **Encapsulation and Decapsulation**

   - To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages in an IP datagram.

Use of UDP

The following lists some uses of the UDP protocol:

- 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
- UDP is suitable for a process with internal flow and error control mechanisms.
- UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software but not in the TCP software.
- UDP is used for management processes such as SNMP .
- UDP is used for some route updating protocols such as Routing Information Protocol (RIP).

Congestion Control and Quality ofService

CONGESTION .EXPLAIN (15 MARKS)

An important issue in a packet-switched network is **congestion.** Congestion in a network may occur if the **load** on the network-the number of packets sent to the network-is greater than the *capacity* of the network-the number of packets a network can handle.

Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.

Congestion in a network or internetwork occurs because routers and switches have queues-buffers that hold the packets before and after processing. When a packet arrives at the incoming interface, it undergoes three steps before departing:

1. The packet is put at the end of the input queue while waiting to be checked.
2. The processing module of the router removes the packet from the input queue once it reaches the front of the queue and uses its routing table and the destination address to find the route.

3. The packet is put in the appropriate output queue and waits its turn to be sent.

The two issues are:

- If the rate of packet arrival is higher than the packet processing rate, the input queues become longer and longer.
- If the packet departure rate is less than the packet processing rate, the output queues become longer and longer.

Network Performance

Congestion control involves two factors that measure the performance of a network: ***delay*** and *throughput.*

- ✓ *Delay Versus Load*

  When the load is much less than the capacity of the network, the delay is at a minimum. The delay becomes infinite when the load is greater than the capacity.

- ✓ *Throughput Versus Load*

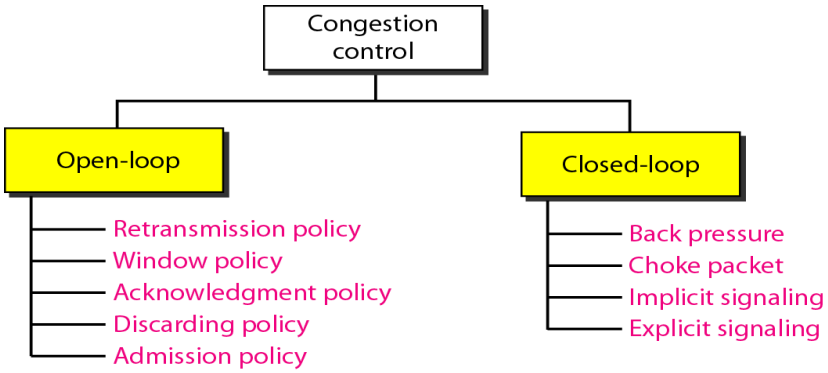
  Throughput in a network is defined as the number of packets passing through the network in a unit of time. When the load is below the capacity of the network, the throughput increases proportionally with the *load.*

CONGESTION CONTROL

Congestion control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened.

The congestion control mechanisms into two broad categories:

- open-loop congestion control (prevention)
- closed-loop congestion control (removal).



Open loop Congestion Control:-

In open loop congestion control, policies are applied to prevent congestion before it happens. Congestion control is handled by either the source or the destination.

- **Retransmission Policy**

If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. Retransmission in general may increase congestion in the network. A good retransmission policy can prevent congestion. The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.

- **Window Policy**

The Selective Repeat window is better than the Go-Back-N window for congestion control, because it tries to send the specific packets that have been lost or corrupted.

In the Go-Back-N window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver. This duplication may make the congestion worse.

- **Acknowledgment Policy .Explain(5 marks)**

If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion. A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires. A receiver may decide to acknowledge only N packets at a time. We need to know that the acknowledgments are

also part of the load in a network. Sending fewer acknowledgments means imposing less load on the network.

- **Discarding Policy**

A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission. For example, in audio transmission, if the policy is to discard less sensitive packets when congestion is likely to happen, the quality of sound is still preserved and congestion is prevented or alleviated.

- **Admission Policy**

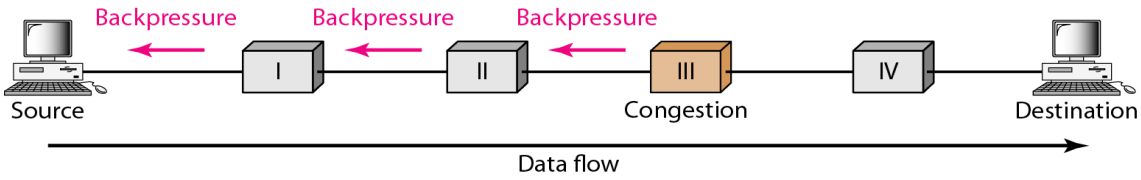
An admission policy, which is a quality-of-service mechanism, can also prevent congestion in virtual-circuit networks. Switches in a flow first check the resource requirement of a flow before admitting it to the network. A router can deny establishing a virtual circuit connection if there is congestion in the network or if there is a possibility of future congestion.

Closed-Loop Congestion control

Closed-loop congestion control mechanisms try to remove congestion after it happen.

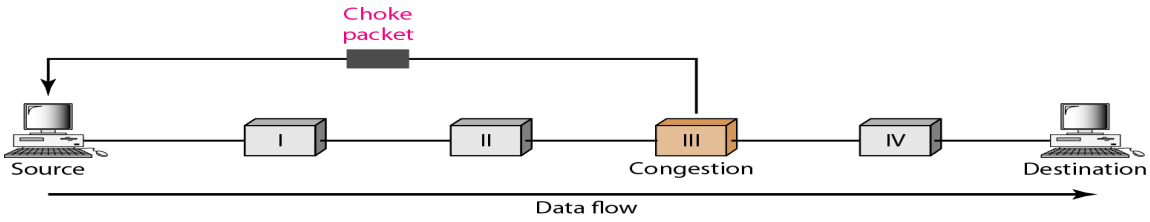
- **Backpressure .EXPLAIN(5 MARKS)**

The technique of backpressure refers to a congestion control mechanism in which a congested node stops receiving data from the immediate upstream node or nodes. This may cause the upstream node or nodes to become congested, and they, in turn, reject data from their upstream nodes or nodes. Back pressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source. The backpressure technique can be applied only to virtual circuit networks, in which each node knows the upstream node from which a flow of data is coming.



➢ Choke Packet

A choke packet is a packet sent by a node to the source to inform it of congestion. In the choke packet method, the warning is from the router, which has encountered congestion, to the source station directly. The intermediate nodes through which the packet has traveled are not warned.



➢ Implicit Signaling

In implicit signaling, there is no communication between the congested node or nodes and the source. The source guesses that there is a congestion somewhere in the network from other symptoms. For example, when a source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested. The delay in receiving an acknowledgment is interpreted as congestion in the network; the source should slow down.

➢ Explicit Signaling

The node that experiences congestion can explicitly send a signal to the source or destination. In the explicit signaling method, the signal is included in the packets that carry data. Explicit signaling, can occur in either the forward or the backward direction.

- Backward Signaling A bit can be set in a packet moving in the direction opposite to the congestion. This bit can warn the source that there is congestion and that it needs to slow down to avoid the discarding of packets.

- Forward Signaling

  A bit can be set in a packet moving in the direction of the congestion. This bit can warn the destination that there is congestion. The receiver in this case can use policies, such as slowing down the acknowledgments, to alleviate the congestion

QUALITY OF SERVICE (QoS).EXPLAIN(15 MARKS)

Quality of service (QoS) is an internetworking issue. We can informally define quality of service as something a flow seeks to attain.

Flow Characteristics

Four types of characteristics are attributed to a flow: *reliability, delay, jitter, and bandwidth.*

- Reliability

  Reliability is a characteristic that a flow needs. Lack of reliability means losing a packet or acknowledgment, which entails retransmission.

- Delay

  Source-to-destination delay is another flow characteristic. Again applications can tolerate delay in different degrees. In this case, telephony, audio conferencing, video conferencing, and remote log-in need minimum delay, while delay in file

transfer or e-mail is less important.

- Jitter

Jitter is defined as the variation in the packet delay. High jitter means the difference between delays is large; low jitter means the variation is small.

Jitter is the variation in delay for packets belonging to the same flow. For example, if four packets depart at times 0, 1, 2, 3 and arrive at 20, 21, 22, 23, all have the same delay, 20 units of time. On the other hand, if the above four packets arrive at 21, 23, 21, and 28, they will have different delays: 21,22, 19, and 24.

- Bandwidth

Different applications need different bandwidths

TECHNIQUES TO IMPROVE QoS

Four common methods can be used to improve the quality of service:

- ✓ scheduling,
- ✓ traffic shaping
- ✓ admission control
- ✓ resource reservation.

Scheduling

Packets from different flows arrive at a switch or router for processing. Several scheduling techniques are designed to improve the quality of service. Ther are:
FIFO queuing, priority queuing, and weighted fair queuing.

- *FIFO Queuing*

In first-in, first-out (FIFO) queuing, packets wait in a buffer (queue) until the node is ready to process them. If the average arrival rate is higher than the average processing rate, the queue will fill up and new packets will be discarded.

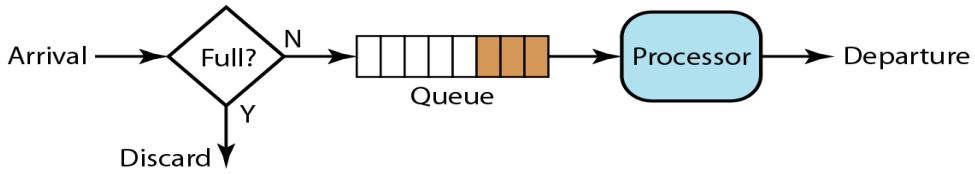
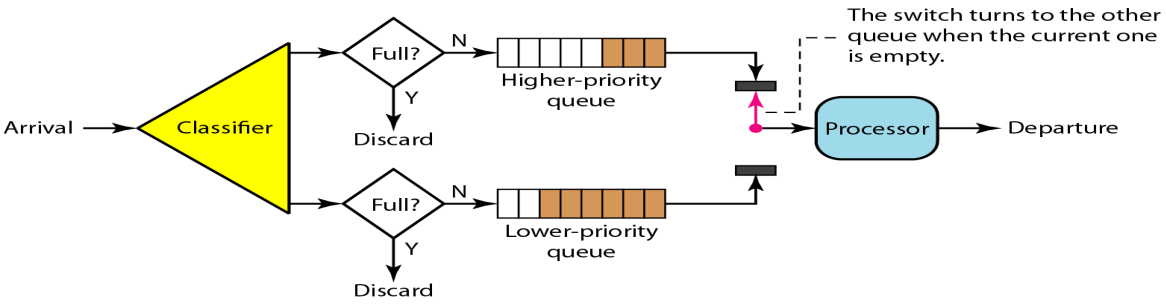


Fig: *FIFO queue*

➢ *Priority Queuing*

In priority queuing, packets are first assigned to a priority class. Each priority class has its own queue. The packets in the highest-priority queue are processed first. Packets in the lowest-priority queue are processed last.



➢ *Weighted Fair Queuing*

In this technique, the packets are still assigned to different classes and admitted to different queues. The queues, however, are weighted based on the priority of the queues; higher priority means a higher weight. The system processes packets in each queue in a round-robin fashion with the number of packets selected from each queue based on the corresponding weight.

Traffic Shaping.EXPLAIN(5 MARKS)

Traffic shaping is a mechanism to control the amount and the rate of the traffic sent to the network. Two techniques can shape traffic: *leaky bucket and token bucket.*

➢ *Leaky Bucket*

A leaky bucket algorithm shapes bursty traffic into fixed-rate traffic by averaging the data rate. It may drop the packets if the bucket is full.

If a bucket has a small hole at the bottom, the water leaks from the bucket at a constant rate as long as there is water in the bucket. The rate at which the water leaks does not depend on the rate at which the water is input to the bucket unless the bucket is empty. The input rate can vary, but the output rate remains constant. Similarly, in networking, a technique called *leaky bucket* can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate.

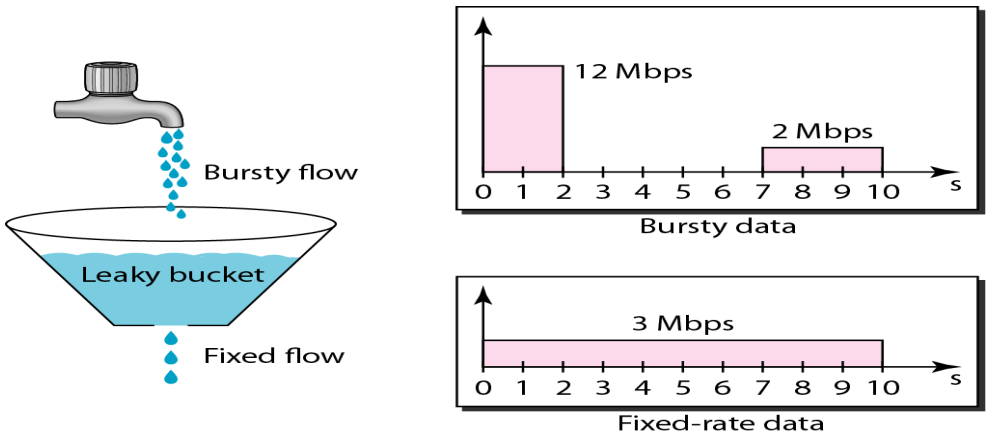


Fig:Leaky bucket

In the above figure, we assume that the network has committed a bandwidth of 3 Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment. In Figure , the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data. The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data. In all, the host has sent 30 Mbits of data in 10s. The leaky bucket smooths the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

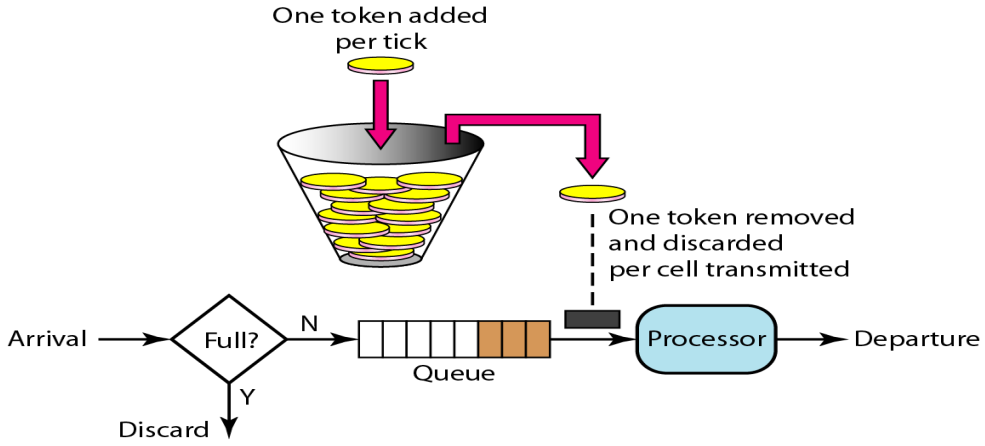
The following is an algorithm for variable-length packets:

1. Initialize a counter to *n* at the tick of the clock.
2. If *n* is greater than the size of the packet, send the packet and decrement the counter by the packet size. Repeat this step until *n* is smaller than the packet size.
3. Reset the counter and go to step 1.

- *Token Bucket*

The token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens. For each tick of the clock, the system sends *n* tokens to the bucket. The system removes one token for every cell (or byte) of data sent.

The token bucket can easily be implemented with a counter. The token is initialized to zero. Each time a token is added, the counter is incremented by 1. Each time a unit of data is sent, the counter is decremented by 1. When the counter is zero, the host cannot send data.



Resource Reservation

A flow of data needs resources such as a buffer, bandwidth, CPU time, and so on. The quality of service is improved if these resources are reserved before hand.

Admission Control

Admission control refers to the mechanism used by a router, or a switch, to accept or reject a flow based on predefined parameters called flow specifications. Before a router accepts a flow for processing, it checks the flow specifications to see if its capacity (in terms of bandwidth, buffer size, CPU speed, etc.) and its previous commitments to other flows can handle the new flow.