
UNIT – 1
DATA COMMUNICATIONS

1. The OSI Reference Model:

   This model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers (Day and Zimmermann, 1983). It was revised in 1995 (Day, 1995). The model is called the ISO-OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems.

   The OSI model has seven layers. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:
1. A layer should be created where a different abstraction is needed.
2. Each layer should perform a well-defined function.
3. The function of each layer should be chosen with an eye toward defining internationally standardized protocols.
4. The layer boundaries should be chosen to minimize the information flow across the interfaces.
5. The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity and small enough that the architecture does not become unwieldy.

The Physical Layer:

   The physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, not as a 0 bit.

The Data Link Layer:

   The main task of the data link layer is to transform a raw transmission facility into a line that appears free of undetected transmission errors to the network layer. It accomplishes this task by having the sender break up the input data into data frames (typically a few hundred or a few thousand bytes) and transmits the frames sequentially. If the service is reliable, the receiver confirms correct receipt of each frame by sending back an acknowledgement frame.
Another issue that arises in the data link layer (and most of the higher layers as well) is how to keep a fast transmitter from drowning a slow receiver in data. Some traffic regulation mechanism is often needed to let the transmitter know how much buffer space the receiver has at the moment. Frequently, this flow regulation and the error handling are integrated.

The Network Layer:

The network layer controls the operation of the subnet. A key design issue is determining how packets are routed from source to destination. Routes can be based on static tables that are "wired into" the network and rarely changed. They can also be determined at the start of each conversation, for example, a terminal session (e.g., a login to a remote machine). Finally, they can be highly dynamic, being determined anew for each packet, to reflect the current network load.

If too many packets are present in the subnet at the same time, they will get in one another's way, forming bottlenecks. The control of such congestion also belongs to the network layer. More generally,
the quality of service provided (delay, transit time, jitter, etc.) is also a network layer issue.

When a packet has to travel from one network to another to get to its destination, many problems can arise. The addressing used by the second network may be different from the first one. The second one may not accept the packet at all because it is too large. The protocols may differ, and so on. It is up to the network layer to overcome all these problems to allow heterogeneous networks to be interconnected. In broadcast networks, the routing problem is simple, so the network layer is often thin or even nonexistent.

**The Transport Layer:**

The basic function of the transport layer is to accept data from above, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end. Furthermore, all this must be done efficiently and in a way that isolates the upper layers from the inevitable changes in the hardware technology.

The transport layer also determines what type of service to provide to the session layer, and, ultimately, to the users of the network. The most popular type of transport connection is an error-free point-to-point channel that delivers messages or bytes in the order in which they were sent. However, other possible kinds of transport service are the transporting of isolated messages, with no guarantee about the order of delivery, and the broadcasting of messages to multiple destinations. The type of service is determined when the connection is established.

The transport layer is a true end-to-end layer, all the way from the source to the destination. In other words, a program on the source machine carries on a conversation with a similar program on the destination machine, using the message headers and control messages. In the lower layers, the protocols are between each machine and its immediate neighbours, and not between the ultimate source and destination machines, which may be separated by many routers.

**The Session Layer:**

The session layer allows users on different machines to establish sessions between them. Sessions offer various services, including dialog control (keeping track of whose turn it is to transmit), token management (preventing two parties from attempting the same critical operation at the same time), and synchronization (check pointing long transmissions to allow them to continue from where they were after a crash).

**The Presentation Layer:**

The presentation layer is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used "on the wire." The presentation layer manages these abstract data structures and allows higher-level data structures (e.g., banking records), to be defined and exchanged.

**The Application Layer:**

The application layer contains a variety of protocols that are commonly needed by users. One widely-used application protocol is HTTP (Hypertext Transfer Protocol), which is the basis for the World Wide Web. When a browser wants a Web page, it sends the name of the page it wants to the server using HTTP. The server then sends the page back. Other application protocols are used for file transfer, electronic mail, and network news.
2. The TCP/IP Reference Model:

The TCP/IP reference model was developed prior to OSI model. The major design goals of this model were,

1. To connect multiple networks together so that they appear as a single network.
2. To survive after partial subnet hardware failures.
3. To provide a flexible architecture.

Unlike OSI reference model, TCP/IP reference model has only 4 layers. They are,

1. Host-to-Network Layer
2. Internet Layer
3. Transport Layer
4. Application Layer

TCP/IP Reference model

Host-to-Network Layer:

The TCP/IP reference model does not really say much about what happens here, except to point out that the host has to connect to the network using some protocol so it can send IP packets to it. This protocol is not defined and varies from host to host and network to network.

Internet Layer:

This layer, called the internet layer, is the linchpin that holds the whole architecture together. Its job is to permit hosts to inject packets into any network and have they travel independently to the destination (potentially on a different network). They may even arrive in a different order than they were sent, in which case it is the job of higher layers to rearrange them, if in-order delivery is desired. Note that "internet" is used here in a generic sense, even though this layer is present in the Internet.

The internet layer defines an official packet format and protocol called IP (Internet Protocol). The job of the internet layer is to deliver IP packets where they are supposed to go. Packet routing is clearly the major issue here, as is avoiding congestion. For these reasons, it is reasonable to say that the TCP/IP internet layer is similar in functionality to the OSI network layer. Fig.6.1 shows this correspondence.
The Transport Layer:

The layer above the internet layer in the TCP/IP model is now usually called the transport layer. It is designed to allow peer entities on the source and destination hosts to carry on a conversation, just as in the OSI transport layer. Two end-to-end transport protocols have been defined here. The first one, TCP (Transmission Control Protocol), is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet. It fragments the incoming byte stream into discrete messages and passes each one on to the internet layer. At the destination, the receiving TCP process reassembles the received messages into the output stream. TCP also handles flow control to make sure a fast sender cannot swamp a slow receiver with more messages than it can handle.

The second protocol in this layer, UDP (User Datagram Protocol), is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own. It is also widely used for one-shot, client-server-type request-reply queries and applications in which prompt delivery is more important than accurate delivery, such as transmitting speech or video. The relation of IP, TCP, and UDP is shown in Fig. 6.2. Since the model was developed, IP has been implemented on many other networks.

Fig.6.1: The TCP/IP reference model.

Fig.6.2: Protocols and networks in the TCP/IP model initially.
The Application Layer:
The TCP/IP model does not have session or presentation layers.

On top of the transport layer is the application layer. It contains all the higher-level protocols. The early ones included virtual terminal (TELNET), file transfer (FTP), and electronic mail (SMTP), as shown in Fig.6.2. The virtual terminal protocol allows a user on one machine to log onto a distant machine and work there. The file transfer protocol provides a way to move data efficiently from one machine to another. Electronic mail was originally just a kind of file transfer, but later a specialized protocol (SMTP) was developed for it.

Many other protocols have been added to these over the years: the Domain Name System (DNS) for mapping host names onto their network addresses, NNTP, the protocol for moving USENET news articles around, and HTTP, the protocol for fetching pages on the World Wide Web, and many others.

3. Various types of network topology and the implications of having different topology

Network topologies:
Network topology defined as the logical connection of various computers in the network. The six basic network topologies are: bus, ring, star, tree, mesh and hybrid.

1. Bus Topology:
In bus topology all the computers are connected to a long cable called a bus. A node that wants to send data puts the data on the bus which carries it to the destination node. In this topology any computer can data over the bus at any time. Since, the bus is shared among all the computers. When two or more computers to send data at the same time, an arbitration mechanism is needed to prevent simultaneous access to the bus.

![Figure 22.1: Bus Topology](image)

A bus topology is easy to install but is not flexible i.e., it is difficult to add a new node to bus. In addition to this the bus stops functioning even if a portion of the bus breaks down. It is also very difficult to isolate fault.
2. **Ring Topology:**

In ring topology, the computers are connected in the form of a ring. Each node has exactly two adjacent neighbors. To send data to a distant node on a ring it passes through many intermediate nodes to reach to its ultimate destination.

![Figure 22.2: Ring Topology](image)

A ring topology is easy to install and reconfigure. In this topology, fault isolation is easy because a signal that circulates all the time in a ring helps in identifying a faulty node.

The data transmission takes place in only one direction. When a node fails in ring, it breaks down the whole ring. To overcome this drawback some ring topologies use dual rings. The topology is not useful to connect large number of computers.

3. **Star Topology:** In star topology all the nodes are connected to a central node called a hub. A node that wants to send some data to some other node on the network, send data to a hub which in turn sends it the destination node. A hub plays a major role in such networks.

![Figure: Star Topology](image)

Star topology is easy to install and reconfigure. If a link fails then it separates the node connected to link from the network and the network continues to function. However, if the hub goes down, the entire network collapses.
4. Tree Topology:

Tree topology is a hierarchy of various hubs. The entire nodes are connected to one hub or the other. There is a central hub to which only a few nodes are connected directly.

![Tree Network Topology](image)

**Figure: Tree topology**

The central hub, also called active hub, looks at the incoming bits and regenerates them so that they can traverse over longer distances. The secondary hubs in tree topology may be active hubs or passive hubs. The failure of a transmission line separates a node from the network.

5. Mesh Topology:

A mesh topology is also called complete topology. In this topology, each node is connected directly to every other node in the network. That is if there are n nodes then there would be \( n(n - 1)/2 \) physical links in the network.

![Mesh Topology](image)

**Figure: Mesh Topology**

As there are dedicated links, the topology does not have congestion problems. Further it does not need a special Media Access Control (MAC) protocol to prevent simultaneous access to the transmission media since links are dedicated, not shared. The topology also provides data security.

The network can continue to function even in the failure of one of the links. Fault identification is also easy.

The main disadvantage of mesh topology is the complexity of the network and the cost associated with the cable length. The mesh topology is not useful for medium to large networks.

6. Hybrid Topology:

Hybrid topology is formed by connecting two or more topologies together. For example, hybrid topology can be created by using the bus, star and ring topologies, as shown in figure 22.6.
Coaxial Cable:

Coaxial cable is a common transmission medium. It has better shielding than twisted pairs, so it can span longer distances at higher speeds. Two kinds of coaxial cable are widely used. One kind, 50-ohm cable, is commonly used when it is intended for digital transmission from the start. The other kind, 75-ohm cable, is commonly used for analog transmission and cable television but is becoming more important with the advent of Internet over cable. This distinction is based on historical, rather than technical, factors (e.g., early dipole antennas had an impedance of 300 ohms, and it was easy to use existing 4:1 impedance matching transformers).

A coaxial cable consists of a stiff copper wire as the core, surrounded by an insulating material. The insulator is encased by a cylindrical conductor, often as a closely-woven braided mesh. The outer conductor is covered in a protective plastic sheath. A cutaway view of a coaxial cable is shown in Fig.1.

The construction and shielding of the coaxial cable give it a good combination of high bandwidth and excellent noise immunity. The bandwidth possible depends on the cable quality, length, and signal-to-noise ratio of the data signal. Modern cables have a bandwidth of close to 1 GHz. Coaxial cables used to be widely used within the telephone system for long-distance lines but have now largely been replaced by fiber optics on long-haul routes. Coax is still widely used for cable television and metropolitan area networks.
Twisted Pair:

Twisted pair is the oldest and most common transmission media. A twisted pair consists of two insulated copper wires, typically about 1 mm thick. The wires are twisted together in a helical form, just like a DNA molecule. Twisting is done because two parallel wires constitute a fine antenna. When the wires are twisted, the waves from different twists cancel out, so the wire radiates less effectively. The most common application of the twisted pair is the telephone system. Nearly all telephones are connected to the telephone company (Telco) office by a twisted pair.

Twisted pairs can run several kilometres without amplification, but for longer distances, repeaters are needed. When many twisted pairs run in parallel for a substantial distance, such as all the wires coming from an apartment building to the telephone company office, they are bundled together and encased in a protective sheath.

The pairs in these bundles would interfere with one another if it were not for the twisting. In parts of the world where telephone lines run on poles above ground, it is common to see bundles several centimetres in diameter. Twisted pairs can be used for transmitting either analog or digital signals. The bandwidth depends on the thickness of the wire and the distance traveled, but several megabits/sec can be achieved for a few kilometers in many cases. Due to their adequate performance and low cost, twisted pairs are widely used and are likely to remain so for years to come.

Twisted pair cabling comes in several varieties, two of which are important for computer networks. Category 3 twisted pairs consist of two insulated wires gently twisted together. Four such pairs are typically grouped in a plastic sheath to protect the wires and keep them together. Prior to about 1988, most office buildings had one category 3 cable running from a central wiring closet on each floor into each office. This scheme allowed up to four regular telephones or two multiline telephones in each office to connect to the telephone company equipment in the wiring closet.

All of these wiring types are often referred to as UTP (Unshielded Twisted Pair), to contrast them with the bulky, expensive, shielded twisted pair cables IBM introduced in the early 1980s, but which have not proven popular outside of IBM installations. Twisted pair cabling is illustrated in Fig.2.

![Fig.2: (a) Category 3 UTP. (b) Category 5 UTP.](image)

4. Error correction and detection methods in the data link layer.

Error-Correcting Codes:

Network designers have developed two basic strategies for dealing with errors. One way is to include enough redundant information along with each block of data sent, to enable the receiver to deduce what the transmitted data must have been. The other way is to include only enough redundancy to allow the receiver to deduce that an error occurred, but not which error, and have it request a retransmission. The former strategy uses error-correcting codes and the latter uses error-detecting codes. The use of error-correcting codes is often referred to as forward error correction.

Each of these techniques occupies a different ecological niche. On channels that are highly reliable, such as fiber, it is cheaper to use an error detecting code and just retransmit the occasional
block found to be faulty. However, on channels such as wireless links that make many errors, it is better to add enough redundancy to each block for the receiver to be able to figure out what the original block was, rather than relying on a retransmission, which itself may be in error.

To understand how errors can be handled, it is necessary to look closely at what an error really is. Normally, a frame consists of m data (i.e., message) bits and r redundant, or check, bits. Let the total length be n (i.e., n = m + r). An n-bit unit containing data and check bits is often referred to as an n-bit codeword.

Given any two code words, say, 10001001 and 10110001, it is possible to determine how many corresponding bits differ. In this case, 3 bits differ. To determine how many bits differ, just exclusive OR the two code words and count the number of 1 bits in the result, for example:

The number of bit positions in which two code words differ is called the Hamming distance. Its significance is that if two codewords are a Hamming distance d apart, it will require d single-bit errors to convert one into the other.

Error-Detecting Codes:
Error-correcting codes are widely used on wireless links, which are notoriously noisy and error prone when compared to copper wire or optical fibers. Without error-correcting codes, it would be hard to get anything through. However, over copper wire or fiber, the error rate is much lower, so error detection and retransmission is usually more efficient there for dealing with the occasional error. As a simple example, consider a channel on which errors are isolated and the error rate is 10^-6 per bit. Let the block size be 1000 bits. To provide error correction for 1000-bit blocks, 10 check bits are needed; a megabit of data would require 10,000 check bits. To merely detect a block with a single 1-bit error, one parity bit per block will suffice. Once every 1000 blocks, an extra block (1001 bits) will have to be transmitted. The total overhead for the error detection + retransmission method is only 2001 bits per megabit of data, versus 10,000 bits for a Hamming code.

If a single parity bit is added to a block and the block is badly garbled by a long burst error, the probability that the error will be detected is only 0.5, which is hardly acceptable. The odds can be improved considerably if each block to be sent is regarded as rectangular matrix n bits wide and k bits high, as described above. A parity bit is computed separately for each column and affixed to the matrix as the last row. The matrix is then transmitted one row at a time. When the block arrives, the receiver checks all the parity bits. If any one of them is wrong, the receiver requests a retransmission of the block. Additional retransmissions are requested as needed until an entire block is received without any parity errors.

The algorithm for computing the checksum is as follows:

1. Let r be the degree of G(x). Append r zero bits to the low-order end of the frame so it now contains m + r bits and corresponds to the polynomial \( x^r \) M(x).

2. Divide the bit string corresponding to G(x) into the bit string corresponding to \( x^r \) M(x), using modulo 2 division.
3. Subtract the remainder (which is always \( r \) or fewer bits) from the bit string corresponding to \( x^r M(x) \) using modulo 2 subtractions. The result is the checksummed frame to be transmitted. Call its polynomial \( T(x) \).

4. Figure illustrates the calculation for a frame 1101011011 using the generator \( G(x) = x^4 + x + 1 \).

![Fig.5.1. Calculation of the polynomial code checksum](image)

6. Sliding window protocols

**Sliding Window Protocols:**

In the previous protocols, data frames were transmitted in one direction only. In most practical situations, there is a need for transmitting data in both directions. One way of achieving full-duplex data transmission is to have two separate communication channels and use each one for simplex data traffic (in different directions). If this is done, we have two separate physical circuits, each with a "forward" channel (for data) and a "reverse" channel (for acknowledgements). In both cases the bandwidth of the reverse channel is almost entirely wasted. In effect, the user is paying for two circuits but using only the capacity of one. A better idea is to use the same circuit for data in both directions. After all, in protocols 2 and 3 it was already being used to transmit frames both ways, and the reverse channel has the same capacity as the forward channel. In this model the data frames from A to B are intermixed with the acknowledgement frames from A to B. By looking at the kind field in the header of an incoming frame, the receiver can tell whether the frame is data or acknowledgement.

Although interleaving data and control frames on the same circuit is an improvement over having two separate physical circuits, yet another improvement is possible. When a data frame arrives, instead of immediately sending a separate control frame, the receiver restrains itself and waits until the network layer passes it the next packet. The acknowledgement is attached to the outgoing data frame (using the ack field in the frame header). In effect, the acknowledgement gets a free ride on the next outgoing data
The technique of temporarily delaying outgoing acknowledgments so that they can be hooked on to the next outgoing data frames is called piggybacking.

### A One-Bit Sliding Window Protocol:

Before tackling the general case, let us first examine a sliding window protocol with a maximum window size of 1. Such a protocol uses stop-and-wait since the sender transmits a frame and waits for its acknowledgement before sending the next one. Figure 9.1 depicts such a protocol. Like the others, it starts out by defining some variables. `next_frame_to_send` tells which frame the sender is trying to send. Similarly, `frame_expected` tells which frame the receiver is expecting. In both cases, 0 and 1 are the only possibilities.

When the first valid frame arrives at computer B, it will be accepted and `frame_expected` will be set to 1. All the subsequent frames will be rejected because B is now expecting frames with sequence number 1, not 0. Furthermore, since all the duplicates have `ack = 1` and B is still waiting for an acknowledgement of 0, B will not fetch a new packet from its network layer. After every rejected duplicate comes in, B sends A a frame containing `seq = 0` and `ack = 0`. Eventually, one of these arrives correctly at A, causing A to begin sending the next packet. No combination of lost frames or premature timeouts can cause the protocol to deliver duplicate packets to either network layer, to skip a packet, or to deadlock. However, a peculiar situation arises if both sides simultaneously send an initial packet. This synchronization difficulty is illustrated by the below Fig. In part (a), the normal operation of the protocol is shown. In (b) the peculiarity is illustrated. If B waits for A’s first frame before sending one of its own, the sequence is as shown in (a), and every frame is accepted. However, if A and B simultaneously initiate communication, their first frames cross, and the data link layers then get into situation (b). In (a) each frame arrival brings a new packet for the network layer; there are no duplicates. In (b) half of the frames contain duplicates, even though there are no transmission errors. Similar situations can occur as a result of premature timeouts, even when one side clearly starts first. In fact, if multiple premature timeouts occur, frames may be sent three or more times.

**Fig:** Two scenarios for protocol 4 (a) Normal case (b) Abnormal case. The notation is `(seq, ack, packet number)`. An asterisk indicates where a network layer accepts a packet.
A Protocol Using Go Back N:
Until now we have made the tacit assumption that the transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to come back is negligible. Sometimes this assumption is clearly false. In these situations the long round-trip time can have important implications for the efficiency of the bandwidth utilization. As an example, consider a 50-kbps satellite channel with a 500-msec round-trip propagation delay. Let us imagine trying to use protocol 4 to send 1000-bit frames via the satellite. At $t = 0$ the sender starts sending the first frame. At $t = 20$ msec the frame has been completely sent. Not until $t = 270$ msec has the frame fully arrived at the receiver, and not until $t = 520$ msec has the acknowledgement arrived back at the sender, under the best of circumstances (no waiting in the receiver and a short acknowledgement frame). This means that the sender was blocked during 500/520 or 96 percent of the time. In other words, only 4 percent of the available bandwidth was used. Clearly, the combination of a long transit time, high bandwidth, and short frame length is disastrous in terms of efficiency.

Pipelining frames over an unreliable communication channel raises some serious issues. First, what happens if a frame in the middle of a long stream is damaged or lost? Large numbers of succeeding frames will arrive at the receiver before the sender even finds out that anything is wrong. When a damaged frame arrives at the receiver, it obviously should be discarded, but what should the receiver do with all the correct frames following it? Remember that the receiving data link layer is obligated to hand packets to the network layer in sequence. In Fig.10.1 we see the effects of pipelining on error recovery.

![Fig: Pipelining and error recovery Effect of an error when (a) receiver's window size is 1 and (b) receiver's window size is large](image)

Two basic approaches are available for dealing with errors in the presence of pipelining. One way, called go back n, is for the receiver simply to discard all subsequent frames, sending no acknowledgements for the discarded frames. This strategy corresponds to a receive window of size 1. In other words, the data link layer refuses to accept any frame except the next one it must give to the network layer. If the sender's window fills up before the timer runs out, the pipeline will begin to empty. Eventually, the sender will time out and retransmit all unacknowledged frames in order, starting with
the damaged or lost one. This approach can waste a lot of bandwidth if the error rate is high.

In the above Fig (a) we see go back n for the case in which the receiver's window is large. Frames 0 and 1 are correctly received and acknowledged. Frame 2, however, is damaged or lost. The sender, unaware of this problem, continues to send frames until the timer for frame 2 expires. Then it backs up to frame 2 and starts all over with it, sending 2, 3, 4, etc. all over again.

The other general strategy for handling errors when frames are pipelined is called selective repeat. When it is used, a bad frame that is received is discarded, but good frames received after it are buffered. When the sender times out, only the oldest unacknowledged frame is retransmitted. If that frame arrives correctly, the receiver can deliver to the network layer, in sequence, all the frames it has buffered.

Selective repeat is often combined with having the receiver send a negative acknowledgement (NAK) when it detects an error, for example, when it receives a checksum error or a frame out of sequence. NAKs stimulate retransmission before the corresponding timer expires and thus improve performance. In Fig.10.1 (b), frames 0 and 1 are again correctly received and acknowledged and frame 2 is lost. When frame 3 arrives at the receiver, the data link layer there notices that it has missed a frame, so it sends back a NAK for 2 but buffers 3. When frames 4 and 5 arrive, they, too, are buffered by the data link layer instead of being passed to the network layer. Eventually, the NAK 2 gets back to the sender, which immediately resends frame 2. When that arrives, the data link layer now has 2, 3, 4, and 5 and can pass all of them to the network layer in the correct order. It can also acknowledge all frames up to and including 5, as shown in the figure. If the NAK should get lost, eventually the sender will time out for frame 2 and send it (and only it) of its own accord, but that may be a quite a while later. In effect, the NAK speeds up the retransmission of one specific frame.

A Protocol Using Selective Repeat:
This protocol works well if errors are rare, but if the line is poor, it wastes a lot of bandwidth on retransmitted frames. An alternative strategy for handling errors is to allow the receiver to accept and buffer the frames following a damaged or lost one. Such a protocol does not discard frames merely because an earlier frame was damaged or lost. In this protocol, both sender and receiver maintain a window of acceptable sequence numbers. The sender's window size starts out at 0 and grows to some predefined maximum, MAX_SEQ. The receiver's window, in contrast, is always fixed in size and equal to MAX_SEQ. The receiver has a buffer reserved for each sequence number within its fixed window. Associated with each buffer is a bit (arrived) telling whether the buffer is full or empty. Whenever a frame arrives, its sequence number is checked by the function between to see if it falls within the window. If so and if it has not already been received, it is accepted and stored. This action is taken without regard to whether or not it contains the next packet expected by the network layer. Of course, it must be kept within the data link layer and not passed to the network layer until all the lower-numbered frames have already been delivered to the network layer in the correct order.
UNIT – 2 : Data Link Layer

1. Pure ALOHA and Slotted ALOHA

ALOHA:

In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. Their work has been extended by many researchers since then (Abramson, 1985). Although Abramson’s work, called the ALOHA system, used ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single shared channel.

We will discuss two versions of ALOHA here: pure and slotted. They differ with respect to whether or not time is divided up into discrete slots into which all frames must fit. Pure ALOHA does not require global time synchronization; slotted ALOHA does.

Pure ALOHA

The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent. There will be collisions, of course, and the colliding frames will be damaged. However, due to the feedback property of broadcasting, a sender can always find out whether or not its frame was destroyed by listening to the channel, the same way other users do. With a LAN, the feedback is immediate; with a satellite, there is a delay of 270 msec before the sender knows if the transmission was successful. If listening while transmitting is not possible for some reason, acknowledgements are needed. If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems in which multiple users share a common channel in a way that can lead to conflicts are widely known as contention systems.

A sketch of frame generation in an ALOHA system is given in Fig. 4-1. We have made the frames all the same length because the throughput of ALOHA systems is maximized by having a uniform frame size rather than allowing variable length frames.

![Figure 4-1. In pure ALOHA, frames are transmitted at completely arbitrary times.](image)

Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled. If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed, and both will have to be retransmitted later. The checksum cannot (and should not) distinguish between a total loss and a near miss. Bad is bad.

A most interesting question is: What is the efficiency of an ALOHA channel? That is, what fractions of all transmitted frames escape collisions under these chaotic circumstances? Let us first consider an infinite collection of interactive users sitting at their computers (stations). A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. The station then transmits a frame containing the line and checks the channel to see if it was successful. If so, the user sees the reply and goes back to typing. If not, the user continues to wait and the frame is retransmitted over and over until it has been successfully sent.

Slotted ALOHA

In 1972, Roberts published a method for doubling the capacity of an ALOHA system (Roberts, 1972).
His proposal was to divide time up into discrete intervals, each interval corresponding to one frame. This approach requires the users to agree on slot boundaries. One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock.

In Roberts’ method, which has come to be known as slotted ALOHA, in contrast to Abramson’s pure ALOHA, a computer is not permitted to send whenever a carriage return is typed. Instead, it is required to wait for the beginning of the next slot. Thus the continuous pure ALOHA is turned into a discrete one. Since the vulnerable period is now halved, the probability of no other traffic during the same slot as our test frame is $e^{-G}$ which leads to

$$S = Ge^{-G} \quad (4-3)$$

As you can see from Fig. 4-3, slotted ALOHA peaks at $G = 1$, with a throughput of $S = 1/e$ or about 0.368, twice that of pure ALOHA. If the system is operating at $G = 1$, the probability of an empty slot is 0.368 (from Eq. 4-2). The best we can hope for using slotted ALOHA is 37 percent of the slots empty, 37 percent successes, and 26 percent collisions. Operating at higher values of $G$ reduces the number of empties but increases the number of collisions exponentially. To see how this rapid growth of collisions with $G$ comes about, consider the transmission of a test frame. The probability that it will avoid a collision is $e^{-G}$, the probability that all the other users are silent in that slot. The probability of a collision is then just $1 - e^{-G}$. The probability of a transmission requiring exactly $k$ attempts, (i.e., $k - 1$ collisions followed by one success) is

$$P_k = e^{-G} (1 - e^{-G})^{k-1}$$

The expected number of transmissions, $E$, per carriage return typed is then

$$E \bigcup kP_k \bigcup ke^{-G} (1 - 1 - 1 - e^{-G})^{k-1}$$

As a result of the exponential dependence of $E$ upon $G$, small increases in the channel load can drastically reduce its performance.

Slotted Aloha is important for a reason that may not be initially obvious. It was devised in the 1970s, used in a few early experimental systems, then almost forgotten. When Internet access over the cable was invented, all of a sudden there was a problem of how to allocate a shared channel among multiple competing users, and slotted Aloha was pulled out of the garbage can to save the day. It has often happened that protocols that are perfectly valid fall into disuse for political reasons (e.g., some big company wants everyone to do things its way), but years later some clever person realizes that a long-discarded protocol solves his current problem. For this reason, in this chapter we will study a number of elegant protocols that are not currently in widespread use, but might easily be used in future applications, provided that enough network designers are aware of them. Of course, we will study various protocols that are in current use as well.

2. Persistent Methods

Carrier Sense Multiple Access Protocols:

Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called carrier sense protocols. A number of them have been proposed. Kleinrock and Tobagi (1975) have analyzed several such protocols in detail. Below we will mention several versions of the carrier sense protocols.

Persistent and Non persistent CSMA

The first carrier sense protocol that we will study here is called 1-persistent CSMA (Carrier Sense Multiple Access). When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1-persistent because the station transmits with a probability of 1 whenever it finds the channel idle.
The propagation delay has an important effect on the performance of the protocol. There is a small chance that just after a station begins sending, another station will become ready to send and sense the channel. If the first station’s signal has not yet reached the second one, the latter will sense an idle channel and will also begin sending, resulting in a collision. The longer the propagation delay, the more important this effect becomes, and the worse the performance of the protocol.

Even if the propagation delay is zero, there will still be collisions. If two stations become ready in the middle of a third station’s transmission, both will wait politely until the transmission ends and then both will begin transmitting exactly simultaneously, resulting in a collision. If they were not so impatient, there would be fewer collisions. Even so, this protocol is far better than pure ALOHA, because both stations have the decency to desist from interfering with the third station’s frame. Intuitively, this will lead to a higher performance than pure ALOHA. Exactly the same holds for slotted ALOHA.

A second carrier sense protocol is nonpersistent CSMA. In this protocol, a conscious attempt is made to be less greedy than in the previous one. Before sending, a station senses the channel. If no one else is sending, the station begins doing so itself. However, if the channel is already in use, the station does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and then repeats the algorithm. Intuitively this algorithm should lead to better channel utilization and longer delays than 1-persistent CSMA.

The last protocol is p-persistent CSMA. It applies to slotted channels and works as follows. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability p. With a probability \( q = 1 - p \) it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities p and q. This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, it acts as if there had been a collision (i.e., it waits a random time and starts again). If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm. Figure 4-4 shows the computed throughput versus offered traffic for all three protocols, as well as for pure and slotted ALOHA.

3. Bit-Map Collision free protocols

Collision-Free Protocols

Although collisions do not occur with CSMA/CD once a station has unambiguously seized the channel, they can still occur during the contention period. These collisions adversely affect the system performance, especially when the cable is long (i.e., large \( \tau \)) and the frames are short. As very long, high-bandwidth fiber optic networks come into use, the combination of large \( \tau \) and short frames will become an increasingly serious problem. In this section, we will examine some protocols that resolve the contention for the channel without any collisions at all, not even during the contention period.

In the protocols to be described, we make the assumption that there are exactly \( N \) stations, each with a unique address from 0 to \( N - 1 \) “wired” into it. It does not matter that some stations may be inactive part of the time. We also assume that propagation delay is negligible. The basic question remains: Which station gets the channel after a successful transmission? We continue using the model of Fig. 4-5 with its discrete contention slots.

A Bit-Map Protocol

In our first collision-free protocol, the basic bit-map method, each contention period consists of exactly \( N \) slots. If station 0 has a frame to send, it transmits a 1 bit during the zeroth slot. No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 during slot 1, but only if it has a frame queued. In general, station \( j \) may announce the fact that it has a frame to send by inserting a 1 bit into slot \( j \). After all \( N \) slots have passed by, each station has complete knowledge of which stations wish to transmit. At that point, they begin transmitting in numerical order (see Fig. 4-6).

<table>
<thead>
<tr>
<th>8 Contention slots</th>
<th>Frames</th>
<th>8 Contention slots</th>
<th>1</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 1 2 3 4 5 6</td>
<td>0 1 2 3 4 5 6</td>
<td>0 1 2 3 4 5 6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another $N$ bit contention period is begun. If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again. Protocols like this in which the desire to transmit is broadcast before the actual transmission are called reservation protocols.

Let us briefly analyze the performance of this protocol. For convenience, we will measure time in units of the contention bit slot, with data frames consisting of $d$ time units. Under conditions of low load, the bit map will simply be repeated over and over, for lack of data frames.

Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, on average, the station will have to wait $N/2$ slots for the current scan to finish and another full $N$ slots for the following scan to run to completion before it may begin transmitting.

The prospects for high-numbered stations are brighter. Generally, these will only have to wait half a scan ($N/2$ bit slots) before starting to transmit. High-numbered stations rarely have to wait for the next scan. Since low-numbered stations must wait on the average $1.5N$ slots and high-numbered stations must wait on the average $0.5N$ slots, the mean for all stations is $N$ slots. The channel efficiency at low load is easy to compute. The overhead per frame is $N$ bits, and the amount of data is $d$ bits, for an efficiency of $d/(N + d)$. At high load, when all the stations have something to send all the time, the $N$ bit contention period is prorated over $N$ frames, yielding an overhead of only 1 bit per frame, or an efficiency of $d/(d + 1)$. The mean delay for a frame is equal to the sum of the time it queues inside its station, plus an additional $N(d + 1)/2$ once it gets to the head of its internal queue.

**Binary Countdown**

A problem with the basic bit-map protocol is that the overhead is 1 bit per station so it does not scale well to networks with thousands of stations. We can do better than that by using binary station addresses. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. The bits in each address position from different stations are BOOLEAN ORed together. We will call this protocol binary countdown. It was used in Datakit (Fraser, 1987). It implicitly assumes that the transmission delays are negligible so that all stations see asserted bits essentially instantaneously.

To avoid conflicts, an arbitration rule must be applied: as soon as a station sees that a high-order bit position that is 0 in its address has been overwritten with a 1, it gives up. For example, if stations 0010, 0100, 1001, and 1010 are all trying to get the channel, in the first bit time the stations transmit 0, 0, 1, and 1, respectively. These are ORed together to form a 1. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue.

The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010, because it has the highest address. After winning the bidding, it may now transmit a frame, after which another bidding cycle starts. The protocol is illustrated in Fig. 4-7. It has the property that higher-numbered stations have a higher priority than lower-numbered stations, which may be either good or bad, depending on the context.

**Figure 4-6.** The basic bit-map protocol.
The channel efficiency of this method is \( d / (d \log_2 N) \). If, however, the frame format has been cleverly chosen so that the sender’s address is the first field in the frame, even these \( \log_2 N \) bits are not wasted, and the efficiency is 100 per-cent.

Mok and Ward (1979) have described a variation of binary countdown using a parallel rather than a serial interface. They also suggest using virtual station numbers, with the virtual station numbers from 0 up to and including the successful station being circularly permuted after each transmission, in order to give higher priority to stations that have been silent unusually long. For example, if stations C, H, D, A, G, B, E, F have priorities 7, 6, 5, 4, 3, 2, 1, and 0, respectively, then a successful transmission by D puts it at the end of the list, giving a priority order of C, H, A, G, B, E, F, D. Thus C remains virtual station 7, but A moves up from 4 to 5 and D drops from 5 to 0. Station D will now only be able to acquire the channel if no other station wants it.

Binary countdown is an example of a simple, elegant, and efficient protocol that is waiting to be rediscovered. Hopefully, it will find a new home some day.

4. Adaptive Tree Protocol in collision free protocol

The Adaptive Tree Walk Protocol

One particularly simple way of performing the necessary assignment is to use the algorithm devised by the U.S. Army for testing soldiers for syphilis during World War II (Dorfman, 1943). In short, the Army took a blood sample from \( N \) soldiers. A portion of each sample was poured into a single test tube. This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through \( N/2 \) and one from the rest. The process was repeated recursively until the infected soldiers were determined.

For the computer version of this algorithm (Capetanakis, 1979) it is convenient to think of the stations as the leaves of a binary tree, as illustrated in Fig. 4-9. In the first contention slot following a successful frame transmission, slot 0, all stations are permitted to try to acquire the channel. If one of them does so, fine. If there is a collision, then during slot 1 only those stations falling under node 2 in the tree may compete. If one of them acquires the channel, the slot following the frame is reserved for those stations under node 3. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4’s turn during slot 2.
In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node’s left and right children. If a bit slot is idle or if there is only one station that transmits in it, the searching of its node can stop, because all ready stations have been located. (Were there more than one, there would have been a collision.)

When the load on the system is heavy, it is hardly worth the effort to dedicate slot 0 to node 1, because that makes sense only in the unlikely event that precisely one station has a frame to send. Similarly, one could argue that nodes 2 and 3 should be skipped as well for the same reason. Put in more general terms, at what level in the tree should the search begin? Clearly, the heavier the load, the farther down the tree the search should begin. We will assume that each station has a good estimate of the number of ready stations, \( q \), for example, from monitoring recent traffic.

To proceed, let us number the levels of the tree from the top, with node 1 in Fig. 4-9 at level 0, nodes 2 and 3 at level 1, etc. Notice that each node at level \( i \) has a fraction \( 2^{-i} \) of the stations below it. If the \( q \) ready stations are uniformly distributed, the expected number of them below a specific node at level \( i \) is just \( 2^{-i} q \). Intuitively, we would expect the optimal level to begin searching the tree as the one at which the mean number of contending stations per slot is 1, that is, the level at which \( 2^{-i} q \) \( \geq 1 \). Solving this equation we find that \( i \geq \log_2 q \).

Numerous improvements to the basic algorithm have been discovered and are discussed in some detail by Bertsekas and Gallager (1992). For example, consider the case of stations \( G \) and \( H \) being the only ones wanting to transmit. At node 1 a collision will occur, so 2 will be tried and discovered idle. It is pointless to probe node 3 since it is guaranteed to have a collision (we know that two or more stations under 1 are ready and none of them are under 2 so they must all be under 3). The probe of 3 can be skipped and 6 tried next. When this probe also turns up nothing, 7 can be skipped and node \( G \) tried next.
5. Cable topologies of Ethernet

Ethernet Cabling

Since the name ‘`Ethernet’` refers to the cable (the ether), let us start our discussion there. Four types of cabling are commonly used, as shown in Fig. 4-13. Historically, **10Base5** cabling, popularly called thick Ethernet, came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go. (The 802.3 standard does not actually require the cable to be yellow, but it does suggest it.) Connections to it are generally made using **vampire taps**, in which a pin is very carefully forced halfway into the coaxial cables core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500 meters. In other words, the first number is the speed in Mbps, finally an indication of the transmission medium. If the medium is coax, its length is given in units of 100 m.

Detecting cable breaks, bad taps, or loose connectors can be a major problem with both media. For this reason, techniques have been developed to track them down. Basically, a pulse of known shape is injected into the cable. If the pulse hits an obstacle or the end of the cable, an echo will be generated and sent back. By carefully timing the interval between sending the pulse and receiving the echo, it is possible to localize the origin of the echo. This technique is called **time domain reflectometry**.

The problems associated with finding cable drove systems toward a different kind of wiring pattern, in which all stations have a cable running to a central **hub** in which they are all connected electrically. Usually, these wires are telephone company twisted pairs, since most office buildings are already wired this way, and there are normally plenty of spare pairs available. This scheme is called **10Base-T**. We will discuss an improved version of this idea—switches—later in this chapter.

These three wiring schemes are illustrated in Fig. 4-14. For 10Base5, a **transceiver** is clamped securely around the cable so that its tap makes contact with the inner core. The transceiver contains the electronics that handle carrier detection and collision detection. When a collision is detected, the transceiver also puts a special invalid signal on the cable to ensure that all other transceivers also realize that a collision has occurred.

With 10Base5, a **transceiver cable** connects the transceiver to an interface board in the computer. The transceiver cable may be up to 50 meters long and contains five individually shielded twisted pairs. Two of the pairs are for data in and data out, respectively. Two more are for control signals in and out. The fifth pair, which is not always used, allows the computer to power the transceiver electronics. Some transceivers allow up to eight nearby computers to be attached to them, to reduce the number of transceivers needed.

The transceiver cable terminates on an interface board inside the computer.

With 10Base2, the connection to the cable is just a passive BNC T-junction connector. The transceiver electronics are on the controller board, and each station always has its own transceiver.

With 10Base-T, there is no cable at all, just the hub (a box full of electronics). Adding or removing a station is simpler in this configuration, and cable breaks can be detected easily. The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100 meters, maybe 200 meters if very high-quality category 5 twisted pairs are used. Nevertheless, 10Base-T quickly became dominant due to its use of existing wiring and the ease of maintenance that it offers. A faster version of 10Base-T (100Base-T) will be discussed later in this chapter.

A fourth cabling option for Ethernet is **10Base-F**, which uses fiber optics. This alternative is expensive due to the cost of the connectors and terminators, but it has excellent noise immunity and is the
method of choice when running between buildings or widely separated hubs. Runs of up to km are allowed. It also offers good security since wiretapping fiber is much more difficult than wire-tapping copper wire.

different ways of wiring up a building. In Fig. (a), a single cable is snaked from room to room, with each station tapping onto it at the nearest point. In Fig. (b), a vertical spine runs from the basement to the roof, with horizontal cables on each floor connected to it by special amplifiers (repeaters). In some buildings the horizontal cables are thin, and the backbone is thick. The most general topology is the tree, as in Fig. 4-15(c), because a network with two paths between some pairs of stations would suffer from interference between the two signals.

![Diagram of cable topologies](image)

**Figure 4-15.** Cable topologies. (a) Linear. (b) Spine. (c) Tree. (d) Segmented.

Each version of Ethernet has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by **repeaters**, as shown in Fig. 4-15(d). A repeater is a physical layer device. It receives, amplifies, and re-transmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different than a single cable (except for some delay introduced by the repeaters). A system may contain multiple cable segments and multiple repeaters, but no two transceivers may be more than 2.5 km apart and no path between any two transceivers may traverse more than four repeaters.

**6. Frame format of 802.11?**

**The 802.11 Frame Structure**

The 802.11 standard defines three classes of frames: data, control, and management. Each of these has a header with a variety of fields used within the MAC sublayer. In addition, there are some headers used by the physical layer but these mostly deal with the modulation so we will not discuss them further here.

The format of the data frame is shown in Fig. 4-30. First comes the **Frame Control** field. It itself has 11 subfields. The first of these is the **Protocol version**, which allows two versions of the protocol to operate at the same time in the same cell. Then come the **Type** (data, control, or management) and **Subtype** fields (e.g., RTS or CTS). The **To DS** and **From DS** bits indicate the frame is going to or coming from the intercell distribution system (e.g., Ethernet). The **MF** bit means that more fragments will follow. The **Retry** bit marks a retransmission of a frame sent earlier. The **Power management** bit is used by the base station to put the receiver in sleep state or take it out of sleep state. The **More** bit indicates that the sender has additional frames for the receiver. The **W** bit specifies that the frame body has been encrypted using the **WEP (Wired Equivalent Privacy)** algorithm. Finally, the **O** bit tells the receiver that a sequence of frames with this bit on must be processed strictly in order.

The second field of the data frame, the **Duration** field, tells how long the frame and its
acknowledgement will occupy the channel. This field is also present in the control frames, and is how other stations manage the NAV mechanism. The frame header contains four addresses, all in standard IEEE 802 format. The source and destination are obviously needed, but what are the other two for? Remember that frames may enter or leave a cell via a base station. The other two addresses are used for the source and destination base stations for intercell traffic.

The Sequence field allows fragments to be numbered. Of the 16 bits available, 12 identify the frame and 4 identify the fragment. The Data field contains the payload, up to 2312 bytes, followed by the usual Checksum.

Figure 4-30. The 802.11 Data frame.

Management frames have a similar format to data frames, except without one of the base station addresses, because management frames are restricted to a single cell. Control frames are shorter still, having only one or two addresses, no Data field, and no Sequence field. The key information here is in the Subtype field, usually RTS, CTS, or ACK.
UNIT – 3: NETWORK LAYER

1. Uni-cast routing protocol.
A routing table can be either static or dynamic. A static table is one with manual entries. A dynamic table, on the other hand, is one that is updated automatically when there is a change somewhere in the internet. Today, an internet needs dynamic routing tables. The tables need to be updated as soon as there is a change in the internet. For instance, they need to be updated when a router is down, and they need to be updated whenever a better route has been found. Routing protocols have been created in response to the demand for dynamic routing tables. A routing protocol is a combination of rules and procedures that let routers in the internet inform each other of changes. It allows routers to share whatever they know about the internet or their neighborhood. The sharing of information allows a router in San Francisco to know about the failure of a network in Texas. The routing protocols also include procedures for combining information received from other routers.

Optimization
A router receives a packet from a network and passes it to another network. A router is usually attached to several networks. When it receives a packet, to which network should it pass the packet? The decision is based on optimization: Which of the available pathways is the optimum pathway? What is the definition of the term optimum? One approach is to assign a cost for passing through a network. We call this cost a metric. However, the metric assigned to each network depends on the type of protocol. Some simple protocols, such as the Routing Information Protocol (RIP), treat all networks as equals. The cost of passing through a network is the same; it is one hop count. So if a packet passes through 10 networks to reach the destination, the total cost is 10 hop counts.

Distance Vector Routing
In distance vector routing, the least-cost route between any two nodes is the route with minimum distance. In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node. The table at each node also guides the packets to the desired node by showing the next stop in the route (next-hop routing). We can think of nodes as the cities in an area and the lines as the roads connecting them. A table can show a tourist the minimum distance between cities.

The table for node A shows how we can reach any node from this node. For example, our least cost to reach node E is 6. The route passes through C.

Initialization
Each node knows how to reach any other node and the cost. At the beginning, however, this is not the case. Each node can know only the distance between itself and its immediate neighbors, those directly connected to it. So for the moment, we assume that each node can send a message to the immediate neighbors and find the distance between itself and these neighbors. Figure 22.15 shows the initial tables for each node. The distance for any entry that is not a neighbor is marked as infinite (unreachable).

2. Routing algorithm and the classifications of it.

The main function of the network layer is routing packets from the source machine to the destination machine. In most subnets, packets will require multiple hops to make the journey. The only notable exception is for broadcast networks, but even here routing is an issue if the source and destination are not on the same network. The algorithms that choose the routes and the data structures that they use are a major area of network layer design.

The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagrams internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. If the subnet uses virtual circuits internally, routing decisions are made only when a new virtual circuit is being set up. Thereafter, data packets just follow the previously-established route. The latter case is sometimes called session routing because a route remains in force for an entire user session (e.g., a login session at a terminal or a file transfer).

It is sometimes useful to make a distinction between routing, which is making the decision which routes to use, and forwarding, which is what happens when a packet arrives. One can think of a router as having two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing tables. This process is forwarding. The other process is responsible for filling in and updating the routing tables.

Regardless of whether routes are chosen independently for each packet or only when new connections are established, certain properties are desirable in a routing algorithm: correctness, simplicity, robustness, stability, fairness, and optimality. Correctness and simplicity hardly require comment, but the need for robustness may be less obvious at first. Once a major network comes on the air, it may be expected to run continuously for years without system wide failures. During that period there will be hardware and software failures of all kinds. Hosts, routers, and lines will fail repeatedly, and the topology will change many times. The routing algorithm should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network to be rebooted every time some router crashes.

Stability is also an important goal for the routing algorithm. There exist routing algorithms that never converge to equilibrium, no matter how long they run. A stable algorithm reaches equilibrium and
stays there. Fairness and optimality may sound obvious.

Minimizing mean packet delay is an obvious candidate, but so is maximizing total network throughput. Furthermore, these two goals are also in conflict, since operating any queuing system near capacity implies a long queuing delay. As a compromise, many networks attempt to minimize the number of hops a packet must make, because reducing the number of hops tends to improve the delay and also reduce the amount of bandwidth consumed, which tends to improve the throughput as well.

Routing algorithms can be grouped into two major classes: non-adaptive and adaptive. Non-adaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J (for all I and J) is computed in advance, off-line, and downloaded to the routers when the network is booted. This procedure is sometimes called static routing.

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get their information (e.g., locally, from adjacent routers, or from all routers), when they change the routes (e.g., every DT sec, when the load changes or when the topology changes), and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time).


One can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the optimality principle. It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route. To see this, call the part of the route from I to Jr1 and the rest of the route r2. If a route better than r2 existed from J to K, it could be concatenated with r1 to improve the route from I to K, contradicting our statement that r1r2 is optimal.

As a direct consequence of the optimality principle, one can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree and is illustrated in Fig.6, where the distance metric is the number of hops. Note that a sink tree is not necessarily unique; other trees with the same path lengths may exist. The goal of all routing algorithms is to discover and use the sink trees for all routers.

![Fig.6 (a) A subnet. (b) A sink tree for router B.](image)

Since a sink tree is indeed a tree, it does not contain any loops, so each packet will be delivered within a finite and bounded number of hops. Links and routers can go down and come back up during operation, so different routers may have different ideas about the current topology. The optimality principle and the sink tree provide a benchmark against which other routing algorithms can be measured.

4. Shortest path routing algorithm

The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (often called a link). To choose a route between
a given pair of routers, the algorithm just finds the shortest path between them on the graph.

One way of measuring path length is the number of hops. Using this metric, the paths ABC and ABE in Fig.7 are equally long. Another metric is the geographic distance in kilometers, in which case ABC is clearly much longer than ABE (assuming the figure is drawn to scale).

Several algorithms for computing the shortest path between two nodes of a graph are known. This one is due to Dijkstra (1959). Each node is labeled (in parentheses) with its distance from the source node along the best known path. Initially, no paths are known, so all nodes are labeled with infinity. As the algorithm proceeds and paths are found, the labels may change, reflecting better paths. A label may be either tentative or permanent. Initially, all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

To illustrate how the labeling algorithm works, look at the weighted, undirected graph of Fig.7(a), where the weights represent, for example, distance. We want to find the shortest path from A to D. Mark node A as permanent, indicated by a filled-in circle. Then examine, in turn, each of the nodes adjacent to A (the working node), relabeling each one with the distance to A. Whenever a node is relabeled, label it with the node from which the probe was made so that one can reconstruct the final path later. Having examined each of the nodes adjacent to A, examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent, as shown in Fig.7(b). This becomes the new working node.

Now start at B and examine all nodes adjacent to it. If the sum of the label on B and the distance from B to the node being considered is less than the label on that node, it is the shorter path, so the node is relabeled.

After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively-labeled node with the smallest value. This node is made permanent and becomes the working node for the next round. Fig.7 shows the first five steps of the algorithm.

To see why the algorithm works, consider Fig.7(c). At that point E is made permanent. Suppose that there were a shorter path than ABE, say AXYZE. There are two possibilities: either node Z has already been made permanent, or it has not been. If it has, then E has already been probed (on the round following the one when Z was made permanent), so the AXYZE path has not escaped our attention and thus cannot be a shorter path.
Now consider the case where Z is still tentatively labeled. Either the label at Z is greater than or equal to that at E, in which case AXYZE cannot be a shorter path than ABE, or it is less than that of E, in which case Z and not E will become permanent first, allowing E to be probed from Z.

5. Distance vector routing algorithm.

Distance vector routing algorithms operate by having each router maintain a table (i.e. a vector) giving the best known distance to each destination and which line to use to get there. These tables are updated by exchanging information with the neighbors.

The distance vector routing algorithm is sometimes called by other names, most commonly the distributed Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm; it was the original ARPANET routing algorithm and was also used in the Internet under the name RIP.

The router is assumed to know the "distance" to each of its neighbors. If the metric is hops, the distance is just one hop. If the metric is queue length, the router simply examines each queue. If the metric is delay, the router can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can.

As an example, assume that delay is used as a metric and that the router knows the delay to each of its neighbors. Once every T msec each router sends to each neighbor a list of its estimated delays to each destination. It also receives a similar list from each neighbor. Imagine that one of these tables has just come in from neighbor X, with Xi being X's estimate of how long it takes to get to router i. If the router knows that the delay to X is m msec, it also knows that it can reach router i via X in Xi + m msec. By performing this calculation for each neighbor, a router can find out which estimate seems the best and use that estimate and the corresponding line in its new routing table. Note that the old routing table is not used in the calculation.
This updating process is illustrated in Fig. 9. Part (a) shows a subnet. The first four columns of part (b) show the delay vectors received from the neighbors of router J. A claims to have a 12-msec delay to B, a 25-msec delay to C, a 40-msec delay to D, etc. Suppose that J has measured or estimated its delay to its neighbors, A, I, H, and K as 8, 10, 12, and 6 msec.

![Fig. 9 (a) A subnet. (b) Input from A, I, H, K, and the new routing table for J.](image)

Consider how J computes its new route to router G. It knows that it can get to A in 8 msec, an A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec to G if it forwards packets bound for G to A. Similarly, it computes the delay to G via I, H, and K as 4 (31 + 10), 18 (6 + 12), and 37 (31 + 6) msec, respectively. The best of these values is 18, so it makes an entry in its routing table that the delay to G is 18 msec and that the route to use is via H. The same calculation is performed for all the other destinations, with the new routing table shown in the last column of the figure.
UNIT – 4: TRANSPORT LAYER

1. Connectionless Internetworking.

A routing decision is made separately for each packet, possibly depending on the traffic at the moment the packet is sent. This strategy can use multiple routes and thus achieve a higher bandwidth than the concatenated virtual-circuit model. On the other hand, there is no guarantee that the packets arrive at the destination in order, assuming that they arrive at all.

The model of Fig. 9 is not quite as simple as it looks. For one thing, if each network has its own network layer protocol, it is not possible for a packet from one network to transit another one. One could imagine the multiprotocol routers actually trying to translate from one format to another, but unless the two formats are close relatives with the same information fields, such conversions will always be incomplete and often doomed to failure. For this reason, conversion is rarely attempted.

A second, and more serious, problem is addressing. Imagine a simple case: a host on the Internet is trying to send an IP packet to a host on an adjoining SNA network. The IP and SNA addresses are different. One would need a mapping between IP and SNA addresses in both directions. Furthermore, the concept of what is addressable is different. In IP, hosts (actually, interface cards) have addresses. In SNA, entities other than hosts (e.g., hardware devices) can also have addresses. At best, someone would have to maintain a database mapping everything to everything to the extent possible, but it would constantly be a source of trouble.

Another idea is to design a universal "internet" packet and have all routers recognize it. This approach is, in fact, what IP is—a packet designed to be carried through many networks. Of course, it may turn out that IPv4 (the current Internet protocol) drives all other formats out of the market, IPv6 (the future Internet protocol) does not catch on, and nothing new is ever invented, but history suggests otherwise. Getting everybody to agree to a single format is difficult when companies perceive it to their commercial advantage to have a proprietary format that they control.

Tunneling is a method of transmitting data that is intended for use only within a private network through a public network in such a way that the routing nodes in the public network are unaware that the transmission is a part of private network.

Handling the general case of making two different networks interwork is exceedingly difficult. However, there is a common special case that is manageable. This case is where the source and destination hosts are on the same type of network, but there is a different network in between. As an example, think of an international bank with a TCP/IP-based Ethernet in Paris, a TCP/IP-based Ethernet in London, and a non-IP wide area network (e.g., ATM) in between, as shown in Fig.
The solution to this problem is a technique called tunneling. To send an IP packet to host 2, host 1 constructs the packet containing the IP address of host 2, inserts it into an Ethernet frame addressed to the Paris multiprotocol router, and puts it on the Ethernet. When the multiprotocol router gets the frame, it removes the IP packet, inserts it in the payload field of the WAN network layer packet, and addresses the latter to the WAN address of the London multiprotocol router. When it gets there, the London router removes the IP packet and sends it to host 2 inside an Ethernet frame.

2. IP protocol and IP header format.

IP protocol is a protocol of network layer whose main objective is to support internetworking. An IP datagram consists of a header part and a text part. The header has a 20-byte fixed part and a variable length optional part. The header format is shown in Fig. 13. It is transmitted in big-endian order: from left to right, with the high-order bit of the Version field going first.

The Version field keeps track of which version of the protocol the datagram belongs to. By including the version in each datagram, it becomes possible to have the transition between versions take years, with some machines running the old version and others running the new one.

Since the header length is not constant, a field in the header, IHL, is provided to tell how long the header is, in 32-bit words. The minimum value is 5, which applies when no options are present. The maximum value of this 4-bit field is 15, which limits the header to 60 bytes, and thus the Options field to 40 bytes. For some options, such as one that records the route a packet has taken, 40 bytes is far too small, making that option useless.

The Type of service field is one of the few fields that has changed its meaning (slightly) over the years. It was and is still intended to distinguish between different classes of service. Various combinations of reliability and speed are possible. For digitized voice, fast delivery beats accurate delivery. For file transfer, error-free transmission is more important than fast transmission.
Originally, the 6-bit field contained (from left to right), a three-bit Precedence field and three flags, D, T, and R. The Precedence field was a priority, from 0 (normal) to 7 (network control packet). The three flag bits allowed the host to specify what it cared most about from the set {Delay, Throughput, Reliability}. In theory, these fields allow routers to make choices between, for example, a satellite link with high throughput and high delay or a leased line with low throughput and low delay. In practice, current routers often ignore the Type of service field altogether.

The Total length includes everything in the datagram—both header and data. The maximum length is 65,535 bytes. At present, this upper limit is tolerable, but with future gigabit networks, larger datagrams may be needed.

The Identification field is needed to allow the destination host to determine which datagram a newly arrived fragment belongs to. All the fragments of a datagram contain the same Identification value.

Next comes an unused bit and then two 1-bit fields. DF stands for Don't Fragment. It is an order to the routers not to fragment the datagram because the destination is incapable of putting the pieces back together again.

MF stands for More Fragments. All fragments except the last one have this bit set. It is needed to know when all fragments of a datagram have arrived.

The Fragment offset tells where in the current datagram this fragment belongs. All fragments except the last one in a datagram must be a multiple of 8 bytes, the elementary fragment unit.

Since 13 bits are provided, there is a maximum of 8192 fragments per datagram, giving a maximum datagram length of 65,536 bytes, one more than the Total length field.

The Time to live field is a counter used to limit packet lifetimes. It is supposed to count time in seconds, allowing a maximum lifetime of 255 sec. It must be decremented on each hop and is supposed to be decremented multiple times when queued for a long time in a router. In practice, it just counts hops. When it hits zero, the packet is discarded and a warning packet is sent back to the source host. This feature prevents datagrams from wandering around forever, something that otherwise might happen if the routing tables ever become corrupted.

When the network layer has assembled a complete datagram, it needs to know what to do with it. The Protocol field tells it which transport process to give it to. TCP is one possibility, but so are UDP and
The Header checksum verifies the header only. Such a checksum is useful for detecting errors generated by bad memory words inside a router. The algorithm is to add up all the 16-bit halfwords as they arrive, using one's complement arithmetic and then take the one's complement of the result. For purposes of this algorithm, the Header checksum is assumed to be zero upon arrival. This algorithm is more robust than using a normal add.

The Source address and Destination address indicate the network number and host number. The Options field was designed to provide an escape to allow subsequent versions of the protocol to include information not present in the original design, to permit experimenters to try out new ideas, and to avoid allocating header bits to information that is rarely needed.

The options are variable length. Each begins with a 1-byte code identifying the option. Some options are followed by a 1-byte option length field, and then one or more data bytes. The Options field is padded out to a multiple of four bytes.

3. Fragmentation and different types of Fragmentation.

Each network imposes some maximum size on its packets. These limits have various causes, among them:

1. Hardware (e.g., the size of an Ethernet frame).
2. Operating system (e.g., all buffers are 512 bytes).
3. Protocols (e.g., the number of bits in the packet length field).
4. Compliance with some (inter)national standard.
5. Desire to reduce error-induced retransmissions to some level.
6. Desire to prevent one packet from occupying the channel too long.

The result of all these factors is that the network designers are not free to choose any maximum packet size they wish. Maximum payloads range from 48 bytes (ATM cells) to 65,515 bytes (IP packets), although the payload size in higher layers is often larger.

An obvious problem appears when a large packet wants to travel through a network whose maximum packet size is too small. One solution is to make sure the problem does not occur in the first place. In other words, the internet should use a routing algorithm that avoids sending packets through networks that cannot handle them. However, this solution is no solution at all. Basically, the only solution to the problem is to allow gateways to break up packets into fragments, sending each fragment as a separate internet packet.

Two opposing strategies exist for recombining the fragments back into the original packet. The first strategy is to make fragmentation caused by a "small-packet" network transparent to any subsequent networks through which the packet must pass on its way to the ultimate destination. This option is shown in Fig. 16(a). In this approach, the small-packet network has gateways (most likely, specialized routers) that interface to other networks. When an oversized packet arrives at a gateway, the gateway breaks it up into fragments. Each fragment is addressed to the same exit gateway, where the pieces are recombined. In this way passage through the small-packet network has been made transparent.
Subsequent networks are not even aware that fragmentation has occurred. ATM networks, for example, have special hardware to provide transparent fragmentation of packets into cells and then reassembly of cells into packets. In the ATM world, fragmentation is called segmentation; the concept is the same, but some of the details are different.

Fig. (a) Transparent fragmentation. (b) Nontransparent fragmentation

Transparent fragmentation is straightforward but has some problems. For one thing, the exit gateway must know when it has received all the pieces, so either a count field or an "end of packet" bit must be provided. For another thing, all packets must exit via the same gateway. By not allowing some fragments to follow one route to the ultimate destination and other fragments a disjoint route, some performance may be lost. ATM requires transparent fragmentation.

Nontransparent fragmentation also has some problems. For example, it requires every host to be able to do reassembly. Yet another problem is that when a large packet is fragmented, the total overhead increases because each fragment must have a header. Whereas in the first method this overhead disappears as soon as the small-packet network is exited, in this method the overhead remains for the rest of the journey. An advantage of nontransparent fragmentation, however, is that multiple exit gateways can now be used and higher performance can be achieved. Of course, if the concatenated virtual-circuit model is being used, this advantage is of no use.

When a packet is fragmented, the fragments must be numbered in such a way that the original data stream can be reconstructed. One way of numbering the fragments is to use a tree. If packet 0 must be split up, the pieces are called 0.0, 0.1, 0.2, etc. If these fragments themselves must be fragmented later on, the pieces are numbered 0.0.0, 0.0.1, 0.0.2, . . . , 0.1.0, 0.1.1, 0.1.2, etc. If enough fields have been reserved in the header for the worst case and no duplicates are generated anywhere, this scheme is sufficient to ensure that all the pieces can be correctly reassembled at the destination, no matter what order they arrive in.

However, if even one network loses or discards packets, end-to-end retransmissions are needed, with unfortunate effects for the numbering system. Suppose that a 1024-bit packet is initially fragmented into four equal-sized fragments, 0.0, 0.1, 0.2, and 0.3. Fragment
0.1 is lost, but the other parts arrive at the destination. Eventually, the source times out and retransmits the original packet again. Only this time Murphy's law strikes and the route taken passes through a network with a 512-bit limit, so two fragments are generated. When the new fragment 0.1 arrives at the destination, the receiver will think that all four pieces are now accounted for and reconstruct the packet incorrectly.

A completely different (and better) numbering system is for the internetwork protocol to define an elementary fragment size small enough that the elementary fragment can pass through every network. When a packet is fragmented, all the pieces are equal to the elementary fragment size except the last one, which may be shorter. An internet packet may contain several fragments, for efficiency reasons. The internet header must provide the original packet number and the number of the (first) elementary fragment contained in the packet. As usual, there must also be a bit indicating that the last elementary fragment contained within the internet packet is the last one of the original packet.

This approach requires two sequence fields in the internet header: the original packet number and the fragment number. There is clearly a trade-off between the size of the elementary fragment and the number of bits in the fragment number.

4. Address Resolution Protocol (ARP):

Address Resolution Protocol (ARP) is a protocol used by Internet Protocol (IP), specifically IPv4, to map IP network address to the hardware addresses used by a datalink protocol. The protocol operates below the network layer as a part of the interface between OSI network and OSI link layer.

The term address resolution refers to the process of finding an address of a computer in a network. The address is resolved using a protocol in which a piece of information is sent by a client process executing on the local computer to a server process executing on a remote computer. The information received by the server allows the server to uniquely identify the network system for which the address was required and therefore it provides the required address. The address resolution procedure is completed when the client receives a response from the server containing the required address. Thus, learning of 48-bit Ethernet address from 32-bit IP address is done with the help of Address Resolution Protocol (ARP).

Consider the following class C networks which are interconnected by using FDDI (Fibre Distributed Data Interface).
When host 1 on CS-Ethernet wants to send data to host 2 of CS-Ethernet, it should know the IP address of host 2, so it broadcasts a message asking for the name of the owner of this IP address. When host 2 receives the broadcast message it replies to host 1 by specifying its ethernet address as E. Thus, the Ethernet address is appended to the transmitting frame of host 1 by the data link layer. The frame is transmitted and is received by host 2.

Thus, with the help of ARP the host can learn the physical address of the destination host when IP address is known.

### Working of ARP:

When an incoming packet destined for a host machine on a particular Local Area Network (LAN) which arrives at gateway, the gateway asks the ARP program to find a physical host or MAC address that matches the IP address. The ARP program looks in the ARP cache and if it finds the IP address, provides it and sends it to the machine. If IP address is not found, ARP broadcasts a request packet in a special format to all machines on the LAN to see if any machine knows this IP address. If a machine recognizes the IP address as its own, it sends a reply. ARP updates the ARP cache for future reference and then sends the packet to the MAC address that replied.

### 5. DHCP: Dynamic Host Configuration Protocol

Is a network protocol that is used to configure devices which are connected to a network so that they can communicate on an IP network? It involves clients and a server operating in a client-server model. In a typical personal home local area network (LAN), a router is the server while clients are personal computers or printers. The router receives this information through a modem from an internet service provider which also operate DHCP servers where the modems are clients. The clients request configuration settings using the DHCP protocol such as an IP address, a default route and one or more DNS server addresses. Once the client implements these settings, the host is able to communicate on that internet.

DHCP was first defined as a standards track protocol in RFC 1531 in October 1993. DHCP is often used together with network address translation (NAT). Network address translation separates public (external) and private (internal) IP addresses. In home networks, the ISP server may assign a globally unique external IP address to a home router or modem and this IP address is used in internet communications. The router will then assign internal IP addresses to the clients connected to it, allowing the clients to broadcast only the external IP address. This improves security by limiting access to devices and also helps to conserve IPv4 addresses.

### Technical Overview

Dynamic Host Configuration Protocol automates network-parameter assignment to network devices from one or more DHCP servers. Even in small networks, DHCP is useful because it makes it easy to add new machines to the network. When a DHCP-configured client (a computer or any other network-aware device) connects to a network, the DHCP client sends a broadcast query requesting necessary information to a DHCP server.
The DHCP server manages a pool of IP addresses and information about client configuration parameters such as default gateway, domain name, the name servers, other servers such as time servers, and so forth. On receiving a valid request, the server assigns the computer an IP address, a lease (length of time the allocation is valid), and other IP configuration parameters, such as the subnet mask and the default gateway. The query is typically initiated immediately after booting, and must complete before the client can initiate IP-based communication with other hosts. Upon disconnecting, the IP address is returned to the pool for use by another computer. This way, many other computers can use the same IP address within minutes of each other. Depending on implementation, the DHCP server may have three methods of allocating IP addresses:

1. **Dynamic allocation**: A network administrator assigns a range of IP addresses to DHCP, and each client computer on the LAN is configured to request an IP address from the DHCP server during network initialization. The request-and-grant process uses a lease concept with a controllable time period, allowing the DHCP server to reclaim (and then reallocate) IP addresses that are not renewed.

2. **Automatic allocation**: The DHCP server permanently assigns a free IP address to a requesting client from the range defined by the administrator. This is like dynamic allocation, but the DHCP server keeps a table of past IP address assignments, so that it can preferentially assign to a client the same IP address that the client previously had.

3. **Static allocation**: The DHCP server allocates an IP address based on a table with MAC address/IP address pairs, which are manually filled in (perhaps by a network administrator). Only clients with a MAC address listed in this table will be allocated an IP address.

DHCP uses two ports: destination UDP port 67 for sending data to the server, and UDP port 68 for data to the client. DHCP communications are connectionless in nature. DHCP operations fall into four basic phases: **IP discovery**, **IP lease offer**, **IP request**, and **IP lease acknowledgment**. These points are often abbreviated as DORA (Discovery, Offer, Request, and Acknowledgment).

DHCP clients and servers on the same subnet communicate via UDP broadcasts, initially. If the client and server are on different subnets, a DHCP Helper or DHCP Relay Agent may be used. Clients requesting renewal of an existing lease may communicate directly via UDP unicast, since the client already has an established IP address at that point.

1. **DHCP discovery**: The client broadcasts messages on the physical subnet to discover available DHCP servers by creating a User Datagram Protocol (UDP) packet with the broadcast destination of 255.255.255.255 or the specific subnet broadcast address. A DHCP client can also request its last-known IP address.

2. **DHCP offer**: When a DHCP server receives an IP lease request from a client, it reserves an IP address for the client and extends an IP lease offer by sending a DHCPOFFER message to the client. This message contains the client's MAC address, the IP address that the server is offering, the subnet mask, the lease duration, and the IP address of the DHCP server making the offer. The server determines the configuration based on the client's hardware address as specified in the CHADDR (Client Hardware Address) field. Here the server, 192.168.1.1, specifies the client's IP address in the YIADDR (Your IP Address) field.

3. **DHCP request**: In response to the DHCP offer, the client replies with a DHCP request, unicast to the server, requesting the offered address. Based on the Transaction ID field in the request, the server is informed which client has accepted.

4. **DHCP acknowledgment**: When the DHCP server receives the DHCPREQUEST message from the client, the configuration process enters its final phase. The acknowledgment phase involves sending a DHCPACK packet to the client. This packet includes the lease duration and any other configuration information that the client might have requested. At this point, the IP configuration process is completed. The protocol expects the DHCP client to configure its network interface with the negotiated parameters.
UNIT – 5: APPLICATION LAYER

1. **UDP – connection less protocol.**

The Internet protocol suite supports a connectionless transport protocol, UDP (User Datagram Protocol). UDP provides a way for applications to send encapsulated IP datagrams and send them without having to establish a connection. UDP is described in RFC 768. UDP transmits segments consisting of an 8-byte header followed by the payload. The header is shown in Fig. 7. The two ports serve to identify the end points within the source and destination machines. When a UDP packet arrives, its payload is handed to the process attached to the destination port. This attachment occurs when BIND primitive or something similar is used, for TCP (the binding process is the same for UDP). In fact, the main value of having UDP over just using raw IP is the addition of the source and destination ports. Without the port fields, the transport layer would not know what to do with the packet. With them, it delivers segments correctly.

![Fig. the UDP header](image)

The source port is primarily needed when a reply must be sent back to the source. By copying the source port field from the incoming segment into the destination port field of the outgoing segment, the process sending the reply can specify which process on the sending machine is to get it.

The UDP length field includes the 8-byte header and the data. The UDP checksum is optional and stored as 0 if not computed (a true computed 0 is stored as all 1s). Turning it off is foolish unless the quality of the data does not matter (e.g., digitized speech). It is probably worth mentioning explicitly some of the things that UDP does not do. It does not do flow control, error control, or retransmission upon receipt of a bad segment. All of that is up to the user processes. What it does do is provide an interface to the IP protocol with the added feature of demultiplexing multiple processes using the ports. That is all it does. For applications that need to have precise control over the packet flow, error control, or timing, UDP provides just what the doctor ordered.

One area where UDP is especially useful is in client-server situations. Often, the client sends a short request to the server and expects a short reply back. If either the request or reply is lost, the client can just time out and try again. Not only is the code simple, but fewer messages are required (one in each direction) than with a protocol requiring an initial setup.

2. **TCP protocol.**

TCP (Transmission Control Protocol) was specifically designed to provide a reliable end-to-end byte stream over an unreliable internetwork. An internetwork differs from a single network because different parts may have wildly different topologies, bandwidths, delays, packet sizes, and other parameters.

TCP was designed to dynamically adapt to properties of the internetwork and to be robust in the face of many kinds of failures. TCP was formally defined in RFC 793. As time went on, various errors and inconsistencies were detected, and the requirements were changed in some areas. These clarifications and some bug fixes are detailed in RFC 1122. Extensions are given in RFC 1323.
Each machine supporting TCP has a TCP transport entity, either a library procedure, a user process, or part of the kernel. In all cases, it manages TCP streams and interfaces to the IP layer. A TCP entity accepts user data streams from local processes, breaks them up into pieces not exceeding 64 KB (in practice, often 1460 data bytes in order to fit in a single Ethernet frame with the IP and TCP headers), and sends each piece as a separate IP datagram. When datagrams containing TCP data arrive at a machine, they are given to the TCP entity, which reconstructs the original byte streams. For simplicity, we will sometimes use just "TCP" to mean the TCP transport entity (a piece of software) or the TCP protocol (a set of rules). From the context it will be clear which is meant. For example, in "The user gives TCP the data," the TCP transport entity is clearly intended.

The IP layer gives no guarantee that datagrams will be delivered properly, so it is up to TCP to time out and retransmit them as need be. Datagrams that do arrive may well do so in the wrong order; it is also up to TCP to reassemble them into messages in the proper sequence. In short, TCP must furnish the reliability that most users want and that IP does not provide.

The TCP Segment Header

Figure shows the layout of a TCP segment. Every segment begins with a fixed-format, 20-byte header. The fixed header may be followed by header options. After the options, if any, up to 65,535 - 20 = 65,495 data bytes may follow, where the first 20 refer to the IP header and the second to the TCP header. Segments without any data are legal and are commonly used for acknowledgements and control messages.

![TCP header diagram]

The Source port and Destination port fields identify the local end points of the connection. The source and destination end points together identify the connection. The sequence number and Acknowledgement number fields perform their usual functions. Note that the latter specifies the next byte expected, not the last byte correctly received. Both are 32 bits long because every byte of data is numbered in a TCP stream.

The TCP header length tells how many 32-bit words are contained in the TCP header. This information is needed because the Options field is of variable length, so the header is, too.
Technically, this field really indicates the start of the data within the segment, measured in 32-bit words, but that number is just the header length in words, so the effect is the same.

Next comes a 6-bit field that is not used. The fact that this field has survived intact for over a quarter of a century is testimony to how well think out TCP is. Lesser protocols would have needed it to fix bugs in the original design.

Now comes six 1-bit flags. URG is set to 1 if the Urgent pointer is in use. The Urgent pointer is used to indicate a byte offset from the current sequence number at which urgent data are to be found. This facility is in lieu of interrupt messages. As we mentioned above, this facility is a bare-bones way of allowing the sender to signal the receiver without getting TCP itself involved in the reason for the interrupt.

The ACK bit is set to 1 to indicate that the Acknowledgement number is valid. If ACK is 0, the segment does not contain an acknowledgement so the Acknowledgement number field is ignored.

The PSH bit indicates PUSH ed data. The receiver is hereby kindly requested to deliver the data to the application upon arrival and not buffer it until a full buffer has been received (which it might otherwise do for efficiency).

The RST bit is used to reset a connection that has become confused due to a host crash or some other reason. It is also used to reject an invalid segment or refuse an attempt to open a connection. In general, if you get a segment with the RST bit on, you have a problem on your hands.

The SYN bit is used to establish connections. The connection request has SYN = 1 and ACK = 0 to indicate that the piggyback acknowledgement field is not in use. The connection reply does bear an acknowledgement, so it has SYN = 1 and ACK = 1. In essence the SYN bit is used to denote CONNECTION REQUEST and CONNECTION ACCEPTED, with the ACK bit used to distinguish between those two possibilities.

The FIN bit is used to release a connection. It specifies that the sender has no more data to transmit. Both SYN and FIN segments have sequence numbers and are thus guaranteed to be processed in the correct order.

Flow control in TCP is handled using a variable-sized sliding window. The Window size field tells how many bytes may be sent starting at the byte acknowledged. A Window size field of 0 is legal and says that the bytes up to and including Acknowledgement number - 1 have been received, but that the receiver is currently badly in need of a rest and would like no more data for the moment. The receiver can later grant permission to send by transmitting a segment with the same Acknowledgement number and a nonzero Window size field.

3. TCP Congestion Control.
When the load offered to any network is more than it can handle, congestion builds up. The Internet is no exception. Although the network layer also tries to manage congestion, most of the heavy lifting is done by TCP because the real solution to congestion is to slow down the data rate. In theory, congestion can be dealt with by employing a principle borrowed from physics: the law of conservation of packets. The idea is to refrain from injecting a new packet into the network until an old one leaves (i.e., is delivered). TCP attempts to achieve this goal by dynamically manipulating the window size. The first step in managing congestion is detecting it. In the old days, detecting congestion was difficult. A timeout caused by a lost packet could have been caused by either (1) noise on a transmission line or (2) packet discard at a congested router. Telling the difference was difficult. Nowadays, packet loss due to transmission errors...
is relatively rare because most long-haul trunks are fiber (although wireless networks are a different story). Consequently, most transmission timeouts on the Internet are due to congestion. All the Internet TCP algorithms assume that timeouts are caused by congestion and monitor timeouts for signs of trouble the way miners watch their canaries. When a connection is established, a suitable window size has to be chosen. The receiver can specify a window based on its buffer size. If the sender sticks to this window size, problems will not occur due to buffer overflow at the receiving end, but they may still occur due to internal congestion within the network.

In Fig, we see this problem illustrated hydraulically. In Fig. 11 (a), we see a thick pipe leading to a small-capacity receiver. As long as the sender does not send more water than the bucket can contain, no water will be lost. In Fig.11 (b), the limiting factor is not the bucket capacity, but the internal carrying capacity of the network. If too much water comes in too fast, it will back up and some will be lost (in this case by overflowing the funnel). The Internet solution is to realize that two potential problems exist—network capacity and receiver capacity—and to deal with each of them separately. To do so, each sender maintains two windows: the window the receiver has granted and a second window, the congestion window. Each reflects the number of bytes the sender may transmit. The number of bytes that may be sent is the minimum of the two windows. Thus, the effective window is the minimum of what the sender thinks is all right and what the receiver thinks is all right. If the receiver says "Send 8 KB" but the sender knows that bursts of more than 4 KB clog the network, it sends 4 KB. On the other hand, if the receiver says "Send 8 KB" and the sender knows that bursts of up to 32 KB get through effortlessly, it sends the full 8 KB requested.

When a connection is established, the sender initializes the congestion window to the size of the maximum segment in use on the connection. It then sends one maximum segment. If this segment is acknowledged before the timer goes off, it adds one segment's worth of bytes to the congestion window to make it two maximum size segments and sends two segments. As each of these segments is acknowledged, the congestion window is increased by one maximum segment size. When the congestion window is n segments, if all n are acknowledged on time, the congestion window is increased by the byte count corresponding to n segments. In effect, each burst acknowledged doubles the congestion window.

Fig (a) A fast network feeding a low-capacity receiver (b) A slow network feeding a high-capacity receiver
The congestion window keeps growing exponentially until either a timeout occurs or the receiver's window is reached. The idea is that if bursts of size, say, 1024, 2048, and 4096 bytes work fine but a burst of 8192 bytes gives a timeout, the congestion window should be set to 4096 to avoid congestion. As long as the congestion window remains at 4096, no bursts longer than that will be sent, no matter how much window space the receiver grants. This algorithm is called slow start, but it is not slow at all (Jacobson, 1988).

4. Electronic Mail.
Electronic mail, or e-mail, as it is known to its many fans, has been around for over two decades. Before 1990, it was mostly used in academia. During the 1990s, it became known to the public at large and grew exponentially to the point where the number of e-mails sent per day now is vastly more than the number of snail mail (i.e., paper) letters.

E-mail, like most other forms of communication, has its own conventions and styles. In particular, it is very informal and has a low threshold of use. People who would never dream of calling up or even writing a letter to a Very Important Person do not hesitate for a second to send a sloppily-written e-mail.

E-mail is full of jargon such as BTW (By The Way), ROTFL (Rolling On The Floor Laughing), anIMHO (In My Humble Opinion). Many people also use little ASCII symbols called smiley’s or emoticons in their e-mail.

The first e-mail systems simply consisted of file transfer protocols, with the convention that the first line of each message (i.e., file) contained the recipient's address. As time went on, the limitations of this approach became more obvious.

Some of the complaints were as follows:

1. Sending a message to a group of people was inconvenient. Managers often need this facility to send memos to all their subordinates.

2. Messages had no internal structure, making computer processing difficult. For example, if a forwarded message was included in the body of another message, extracting the forwarded part from the received message was difficult.

3. The originator (sender) never knew if a message arrived or not.

4. If someone was planning to be away on business for several weeks and wanted all incoming e-mail to be handled by his secretary, this was not easy to arrange.

5. The user interface was poorly integrated with the transmission system requiring users first to edit a file, then leave the editor and invoke the file transfer program.

6. It was not possible to create and send messages containing a mixture of text, drawings, facsimile, and voice.

As experience was gained, more elaborate e-mail systems were proposed. In 1982, the ARPANET e-mail proposals were published as RFC 821 (transmission protocol) and RFC 822 (message format). Minor revisions, RFC 2821 and RFC 2822, have become Internet standards, but everyone still refers to Internet e-mail as RFC 822.

In 1984, CCITT drafted its X.400 recommendation. After two decades of competition, e-mail
systems based on RFC 822 are widely used, whereas those based on X.400 have disappeared. How a system hacked together by a handful of computer science graduate students beat an official international standard strongly backed by all the PTTs in the world, many governments, and a substantial part of the computer industry brings to mind the Biblical story of David and Goliath.

The reason for RFC 822's success is not that it is so good, but that X.400 was so poorly designed and so complex that nobody could implement it well. Given a choice between a simple-minded, but working, RFC 822-based e-mail system and a supposedly truly wonderful, but nonworking, X.400 e-mail system, most organizations chose the former.

5. DNS and usage of resource records.

Domain Name System:

The Domain Name Service (DNS) is a hierarchical distributed method of organizing the name space of the Internet. The DNS administratively groups hosts into hierarchy of authority that allows addressing and other information to be widely distributed and maintained. A key advantage to the DNS is that it eliminates dependence on a centrally maintained file that maps host names addresses. DNS is supported via asset of network-resident servers, also called domain name servers.

The IP address is a numeric address that serves role analogous to a telephone number. In representation, addresses always consist of four numbers; four decimal values separated by periods. Figure 17.1 illustrates the addresses. The computer named mugwump.cl.msu.edu for instance, is assigned a number of 35.8:1.212. The reason a computer would have two names is that IP addresses numeric; they can be easily understood and manipulated by the hardware and software that must move information over the Internet. So IP addresses are better-suited to computers, and domain addresses are better-suited to humans. DNS allows a translation between the domain name and the IP address. Domain names do not necessarily have four parts. They might have only two parts—a top-level domain such as “edu” or “corn,” preceded by a sub domain or three, four, or many. The limitations are,

(i) A domain- Mime cannot exceed 255 characters and
(ii) Each part of the name cannot exceed 63 characters.

The DNS translates the plain english address, www.metahouse.com, for example, into numbers that Internet computers can understand, such as 123.23.43. 121. In order to do this efficiently, the Internet has been organized into a number of major domains. Major domains refer to the letters at the end of a plain english address, such as .com. A number of common domains are used in the United States: .com (commercial); .edu (education); .gov government); .mil (military), .net (Internet service providers and networks-companies and groups concerned with has been growing exponentially, the domain name system is being expanded and may also include at least seven additional domains, such as .web for Web. Only two letters are used outside the United States to identify the domains; for example, .au for Australia; .ca for Canada; .uk for United Kingdom; and for France.

IP address read from general to specific 35.8.1.212

<table>
<thead>
<tr>
<th>Network address</th>
<th>Host address</th>
</tr>
</thead>
<tbody>
<tr>
<td>spacialink.msfc.nasa.gov</td>
<td></td>
</tr>
</tbody>
</table>

Domains are organized in a hierarchical manner, so that beneath major domains are many minor domains. As an example of how the DNS and domains work, looks at NASA’s SPACE link Internet address: spacelink.msfc.nasa.gov.
The .top domain is .gov, which stands for government. The domain just below that is .nasa, which is the NASA domain. Then below that, .msfc (Marshall Space Flight Center) is one of NASA’s many computer networks. SPACE link identifies the NASA computer that runs the SPACE link program. SPACE link’s numeric IP address has changed through the years, but its Internet address has stayed the same.

![Figure 17.2: A Portion of the Internet Domain Name Spice](image)

The top-level domains come in two flavors: generic and countries. The generic domains are corn (commercial), edu (educational institutions), goy (the U.S. federal government), mil (certain international organizations) mil (the U.S. armed forces), net (network providers), and org (non-profit organizations). The country domains include one entry for every country, as defined in ISO 3166.

Each domain is named by the path upward from it to the (unnamed) root. The components are separated by periods (pronounced “dot”). Thus, Sun Microsystems engineering department might be eng.sun.com rather than a UNIX-style name such as /com/sun/eng.

**Absolute and Relative Domain Names:**

Domain names can be either absolute or relative. An absolute domain name ends with a period (e.g., eng.sun.com) whereas a relative one does not. Relative names have to be interpreted in some context to uniquely determine their true meaning. In both cases, a named domain refers to a specific node in the tree and all the nodes under it.

Domain names are case insensitive, so edu and EDU mean the same thing. Component names can be up to 63 characters long, and full path names must not exceed 255 characters.

In principle, domains can be inserted into the tree in two different ways. For example, cs.yale.edu could equally well be listed under the country domain ascs.yale.ct.us.

**Resource Records:**

The ‘name servers’ that together implement the DNS distributed database, store resource records (RR) for the hostname to IP address mapping. Each DNS reply message carries one or more resource records.

For a single host, the most common resource record is just its IP address, but many other kinds of resource records also exist. When a resolver gives a domain name to DNS, it gets back the resource records associated with that name. Thus the real function of DNS is to map domain names onto resource records.

A resource record is a five-tuple. Its format is as follows.

1. Domain_name
2. Time_to_live
3. Type
(iv) Class

(v) Value

(i) The Domain_name tells the domain to which this record applies. Normally, many records exist for each domain and each copy of the database holds information about multiple domains. The field is thus the primary search key used to satisfy queries.

(ii) The Time_to_live field gives an indication of the stability of record. Information that is highly stable is assigned a large value. Such as 86400 (the number of seconds in 1 day). Information that is highly volatile is assigned a small value, such as 60 (1 minute).

(iii) The type field tells the record type, as listed in the table below.

(iv) The fourth field of every resource record is the Class. For Internet information, it is always in. For non-Internet information, other codes can be used.

(v) Value field, can be a number, a domain name, or an ASCII string. The semantics depend on the records type.

<table>
<thead>
<tr>
<th>Type</th>
<th>Meaning</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOA</td>
<td>Start of Authority</td>
<td>Parameters for this zone</td>
</tr>
<tr>
<td>A</td>
<td>IP address of a host</td>
<td>32-Bit integer</td>
</tr>
<tr>
<td>MX</td>
<td>Mail exchange</td>
<td>Priority, domain willing to accept e-mail</td>
</tr>
<tr>
<td>NS</td>
<td>Name Server</td>
<td>Name of a server for this domain</td>
</tr>
<tr>
<td>CNAME</td>
<td>Canonical name</td>
<td>Domain name</td>
</tr>
<tr>
<td>PTR</td>
<td>Pointer</td>
<td>Alias for an IP address</td>
</tr>
<tr>
<td>HINFO</td>
<td>Host description</td>
<td>CPU and OS in ASCII</td>
</tr>
<tr>
<td>TXT</td>
<td>Text</td>
<td>Uninterpreted ASCII text</td>
</tr>
</tbody>
</table>
16. Unit-wise Question bank

**Unit 1**

**Data Communication**

1. Define Network?
   A computer network is a set of computers connected together for the purpose of sharing resources. The most common resource shared today is connection to the Internet. Other shared resources can include a printer or a file server. The Internet itself can be considered a computer network.

   A computer network is a set of connected computers. Computers on a network are called nodes. The connection between computers can be done via cabling, most commonly the Ethernet cable, or wirelessly through radio waves. Connected computers can share resources, like access to the Internet, printers, file servers, and others. A network is a multipurpose connection, which allows a single computer to do more.

2. What is mean by data communication?
   Data communication is the exchange of data (in the form of 1s and 0s) between two devices via some form of transmission medium (such as a wire cable).

3. What are the three criteria necessary for an effective and efficient network?
   The most important criteria are performance, reliability and security. Performance of the network depends on number of users, type of transmission medium, and the capabilities of the connected h/w and the efficiency of the s/w.

   Reliability is measured by frequency of failure, the time it takes a link to recover from the failure and the network’s robustness in a catastrophe.

   Security issues include protecting data from unauthorized access and viruses.

4. What are the three fundamental characteristics determine the effectiveness of the data communication system?
   The effectiveness of the data communication system depends on three fundamental characteristics:

   Delivery: The system must deliver data to the correct destination.
   Accuracy: The system must deliver data accurately.
   Timeliness: The system must deliver data in a timely manner.

5. What are the advantages of distributed processing?
   Advantages of distributed processing include security/encapsulation, distributed databases, faster problem solving, security through redundancy and collaborative processing.

3 mark questions and answers:

1. Explain different types of networks?
   A computer network is a set of connected computers. Computers on a network are called nodes. The connection between computers can be done via cabling, most commonly the Ethernet cable, or wirelessly through radio waves. Connected computers can share resources, like access to the Internet, printers, file servers, and others. A network is a multipurpose connection, which allows a single computer to do more.

   Types of Network Connections
   Computer networks can be broken down historically into topologies, which is a technique of connecting computers. The most common topology today is a collapsed ring. This is due to the success of a network protocol called the Ethernet. This protocol, or network language, supports the Internet, Local Area Networks, and Wide Area Networks.
Star Topology

A **star topology** is a design of a network where a central node extends a cable to each computer on the network. On a star network, computers are connected independently to the center of the network. If a cable is broken, the other computers can operate without problems. A star topology requires a lot of cabling.

Bus Topology

A **bus topology** is another type of design where a single cable connects all computers and the information intended for the last node on the network must run through each connected computer. If a cable is broken, all computers connected down the line cannot reach the network. The benefit of a bus topology is a minimal use of cabling.

Ring Topology

A similar topology is called a **ring**. In this design, computers are connected via a single cable, but the end nodes also are connected to each other. In this design, the signal circulates through the network until it finds the intended recipient. If a network node is not configured properly, or it is down temporarily for another reason, the signal will make a number of attempts to find its destination.

A **collapsed ring** is a topology where the central node is a network device called a hub, a router, or a switch. This device runs a ring topology internally and features plugins for cables. Next, each computer has an independent cable, which plugs into the device. Most modern offices have a **cabling closet**, or a space containing a switch device that connects the network. All computers in the office connect to the cabling closet and the switch. Even if a network plug is near a desk, the plug is connected via a cable to the cabling closet.

2. **State the goals of networks?**

**NETWORK GOALS:**

- The main goal of networking is "Resource sharing", and it is to make all programs, data and equipment available to anyone on the network without the regard to the physical location of the resource and the user.

- A second goal is to provide high reliability by having alternative sources of supply. For example, all files could be replicated on two or three machines, so if one of them is unavailable, the other copies could be available.

- Another goal is saving money. Small computers have a much better price/performance ratio than larger ones. Mainframes are roughly a factor of ten times faster than the fastest single chip microprocessors, but they cost thousand times more. This imbalance has caused many system designers to build systems consisting of powerful personal computers, one per user, with data kept on one or more shared file server machines. This goal leads to networks with many computers located in the same building. Such a network is called a LAN (local area network).

- Another closely related goal is to increase the systems performance as the work load increases by just adding more processors. With central mainframes, when the system is full, it must be replaced by a larger one, usually at great expense and with even greater disruption to the users.

- Computer networks provide a powerful communication medium. A file that was updated or modified on a network can be seen by the other users on the network immediately.

**NETWORK APPLICATIONS:**

- Access to remote programs.
- Access to remote databases.
- Value-added communication facilities.
3. **List two advantages of layering principle in computer networks?**

The general principle of using layered architecture is to reduce the design complexity, most of the networks are organized as a series of layers or levels, each one build upon one below it.

The basic idea of a layered architecture is to divide the design into small pieces. Each layer adds to the services provided by the lower layers in such a manner that the highest layer is provided a full set of services to manage communications and run the applications.

The benefits of the layered models are modularity and clear interfaces, i.e. open architecture and comparability between the different providers' components.

A basic principle is to ensure independence of layers by defining services provided by each layer to the next higher layer without defining how the services are to be performed. This permits changes in a layer without affecting other layers.

Prior to the use of layered protocol architectures, simple changes such as adding one terminal type to the list of those supported by an architecture often required changes to essentially all communications software at a site.

The number of layers, functions and contents of each layer differ from network to network. However in all networks, the purpose of each layer is to offer certain services to higher layers, shielding those layers from the details of how the services are actually implemented.

The basic elements of a layered model are services, protocols and interfaces. A service is a set of actions that a layer offers to another (higher) layer.

Protocol is a set of rules that a layer uses to exchange information with a peer entity. These rules concern both the contents and the order of the messages used. Between the layers service interfaces are defined. The messages from one layer to another are sent through those interfaces.

In n-layer architecture, layer n on one machine carries on conversation with the layer n on other machine. The rules and conventions used in this conversation are collectively known as the layer-n protocol. Basically, a protocol is an agreement between the communicating parties on how communication is to proceed.

The entities comprising the corresponding layers on different machines are called peers. In other words, it is the peers that communicate using protocols. In reality, no data is transferred from layer n on one machine to layer n of another machine. Instead, each layer passes data and control information to the layer immediately below it, until the lowest layer is reached.

Between each pair of adjacent layers there is an interface. The interface defines which primitives operations and services the lower layer offers to the upper layer adjacent to it.

When network designer decides how many layers to include in the network and what each layer should do, one of the main considerations is defining clean interfaces between adjacent layers. Doing so, in turns requires that each layer should perform well-defined functions. In addition to minimize the amount of information passed between layers, clean-cut interface also makes it simpler to replace the implementation of one layer with a completely different implementation, because all what is required of new implementation is that it offers same set of services to its upstairs neighbor as the old implementation.
4. **Group the OSI layers by function.**
   The seven layers of the OSI model belonging to three subgroups. Physical, data link and network layers are the network support layers; they deal with the physical aspects of moving data from one device to another. Session, presentation and application layers are the user support layers; they allow interoperability among unrelated software systems. The transport layer ensures end-to-end reliable data transmission.

5. **What are header and trailers and how do they get added and removed?**
   Each layer in the sending machine adds its own information to the message it receives from the layer just above it and passes the whole package to the layer just below it. This information is added in the form of headers or trailers. Headers are added to the message at the layers 6,5,4,3, and 2. A trailer is added at layer 2. At the receiving machine, the headers or trailers attached to the data unit at the corresponding sending layers are removed, and actions appropriate to that layer are taken.

**5 mark questions and answers:**

1. **The OSI Reference Model:**

   This model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers (Day and Zimmermann, 1983). It was revised in 1995 (Day, 1995). The model is called the ISO-OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems.

   The OSI model has seven layers. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:

   6. A layer should be created where a different abstraction is needed.
   7. Each layer should perform a well-defined function.
   8. The function of each layer should be chosen with an eye toward defining internationally standardized protocols.
   9. The layer boundaries should be chosen to minimize the information flow across the interfaces.
   10. The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity and small enough that the architecture does not become unwieldy.

   **The Physical Layer:**
   The physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, not as a 0 bit.

   **The Data Link Layer:**
   The main task of the data link layer is to transform a raw transmission facility into a line that appears free of undetected transmission errors to the network layer. It accomplishes this task by having the sender break up the input data into data frames (typically a few hundred or a few thousand bytes) and transmits the frames sequentially. If the service is reliable, the receiver confirms correct receipt of each frame by sending back an acknowledgement frame.
Another issue that arises in the data link layer (and most of the higher layers as well) is how to keep a fast transmitter from drowning a slow receiver in data. Some traffic regulation mechanism is often needed to let the transmitter know how much buffer space the receiver has at the moment. Frequently, this flow regulation and the error handling are integrated.

The Network Layer:

The network layer controls the operation of the subnet. A key design issue is determining how packets are routed from source to destination. Routes can be based on static tables that are "wired into" the network and rarely changed. They can also be determined at the start of each conversation, for example, a terminal session (e.g., a login to a remote machine). Finally, they can be highly dynamic, being determined anew for each packet, to reflect the current network load.

If too many packets are present in the subnet at the same time, they will get in one another's way, forming bottlenecks. The control of such congestion also belongs to the network layer. More generally, the quality of service provided (delay, transit time, jitter, etc.) is also a network layer issue.

When a packet has to travel from one network to another to get to its destination, many problems can arise. The addressing used by the second network may be different from the first one. The second one may not accept the packet at all because it is too large. The protocols may differ, and so on. It is up to the network layer to overcome all these problems to allow heterogeneous networks to be interconnected. In broadcast networks, the routing problem is simple, so the network layer is often thin or even nonexistent.

The Transport Layer:

The basic function of the transport layer is to accept data from above, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end. Furthermore, all this must be done efficiently and in a way that isolates the upper layers from the inevitable changes in the hardware technology.
The transport layer also determines what type of service to provide to the session layer, and, ultimately, to the users of the network. The most popular type of transport connection is an error-free point-to-point channel that delivers messages or bytes in the order in which they were sent. However, other possible kinds of transport service are the transporting of isolated messages, with no guarantee about the order of delivery, and the broadcasting of messages to multiple destinations. The type of service is determined when the connection is established.

The transport layer is a true end-to-end layer, all the way from the source to the destination. In other words, a program on the source machine carries on a conversation with a similar program on the destination machine, using the message headers and control messages. In the lower layers, the protocols are between each machine and its immediate neighbours, and not between the ultimate source and destination machines, which may be separated by many routers.

**The Session Layer:**

The session layer allows users on different machines to establish sessions between them. Sessions offer various services, including dialog control (keeping track of whose turn it is to transmit), token management (preventing two parties from attempting the same critical operation at the same time), and synchronization (check pointing long transmissions to allow them to continue from where they were after a crash).

**The Presentation Layer:**

The presentation layer is concerned with the syntax and semantics of the information transmitted. In order to make it possible for computers with different data representations to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used "on the wire." The presentation layer manages these abstract data structures and allows higher-level data structures (e.g., banking records), to be defined and exchanged.

**The Application Layer:**

The application layer contains a variety of protocols that are commonly needed by users. One widely-used application protocol is HTTP (Hypertext Transfer Protocol), which is the basis for the World Wide Web. When a browser wants a Web page, it sends the name of the page it wants to the server using HTTP. The server then sends the page back. Other application protocols are used for file transfer, electronic mail, and network news.

2. **The TCP/IP Reference Model:**

The TCP/IP reference model was developed prior to OSI model. The major design goals of this model were,

1. To connect multiple networks together so that they appear as a single network.
2. To survive after partial subnet hardware failures.
3. To provide a flexible architecture. Unlike OSI reference model, TCP/IP reference model has only 4 layers. They are,
   5. Host-to-Network Layer
   6. Internet Layer
   7. Transport Layer
8. Application Layer

TCP/IP Reference model

Host-to-Network Layer:

The TCP/IP reference model does not really say much about what happens here, except to point out that the host has to connect to the network using some protocol so it can send IP packets to it. This protocol is not defined and varies from host to host and network to network.

Internet Layer:

This layer, called the internet layer, is the linchpin that holds the whole architecture together. Its job is to permit hosts to inject packets into any network and have they travel independently to the destination (potentially on a different network). They may even arrive in a different order than they were sent, in which case it is the job of higher layers to rearrange them, if in-order delivery is desired. Note that "internet" is used here in a generic sense, even though this layer is present in the Internet.

The internet layer defines an official packet format and protocol called IP (Internet Protocol). The job of the internet layer is to deliver IP packets where they are supposed to go. Packet routing is clearly the major issue here, as is avoiding congestion. For these reasons, it is reasonable to say that the TCP/IP internet layer is similar in functionality to the OSI network layer. Fig.6.1 shows this correspondence.

The Transport Layer:

The layer above the internet layer in the TCP/IP model is now usually called the transport layer. It is designed to allow peer entities on the source and destination hosts to carry on a conversation, just as in the OSI transport layer. Two end-to-end transport protocols have been defined here. The first one, TCP (Transmission Control Protocol), is a reliable connection-oriented protocol that allows a byte stream originating on one machine to be delivered without error on any other machine in the internet. It fragments the incoming byte stream into discrete messages and passes each one on to the internet layer. At the destination, the receiving TCP process reassembles the received messages into the output stream. TCP also handles flow control to make sure a fast sender cannot swamp a slow receiver with more messages than it can handle.
The second protocol in this layer, UDP (User Datagram Protocol), is an unreliable, connectionless protocol for applications that do not want TCP's sequencing or flow control and wish to provide their own. It is also widely used for one-shot, client-server-type request-reply queries and applications in which prompt delivery is more important than accurate delivery, such as transmitting speech or video. The relation of IP, TCP, and UDP is shown in Fig. 6.2. Since the model was developed, IP has been implemented on many other networks.

The Application Layer:
The TCP/IP model does not have session or presentation layers.

On top of the transport layer is the application layer. It contains all the higher-level protocols. The early ones included virtual terminal (TELNET), file transfer (FTP), and electronic mail (SMTP), as shown in Fig.6.2. The virtual terminal protocol allows a user on one machine to log onto a distant machine and work there. The file transfer protocol provides a way to move data efficiently from one machine to another. Electronic mail was originally just a kind of file transfer, but later a specialized protocol (SMTP) was developed for it.

Many other protocols have been added to these over the years: the Domain Name System (DNS) for mapping host names onto their network addresses, NNTP, the protocol for moving USENET news
articles around, and HTTP, the protocol for fetching pages on the World Wide Web, and many others.

3. Various types of network topology and the implications of having different topology

Network topologies:

Network topology defined as the logical connection of various computers in the network.

The six basic network topologies are: bus, ring, star, tree, mesh and hybrid.

Bus Topology:

In bus topology all the computers are connected to a long cable called a bus. A node that wants to send data puts the data on the bus which carries it to the destination node. In this topology any computer can data over the bus at any time. Since, the bus is shared among all the computers. When two or more computers to send data at the same time, an arbitration mechanism is needed to prevent simultaneous access to the bus.

![Bus Topology](image-1)

**Figure 22.1: Bus Topology**

A bus topology is easy to install but is not flexible i.e., it is difficult to add a new node to bus. In addition to this the bus stops functioning even if a portion of the bus breaks down. It is also very difficult to isolate fault.

Ring Topology:

In ring topology, the computers are connected in the form of a ring. Each node has exactly two adjacent neighbors. To send data to a distant node on a ring it passes through many intermediate nodes to reach to its ultimate destination.

![Ring Topology](image-2)

**Figure 22.2: Ring Topology**

A ring topology is as to install and reconfigure. In this topology, fault isolation is easy because a signal that circulates all the time in a ring helps in identifying a faulty node.

The data transmission takes place in only one direction. When a node fails in ring, it breaks down the whole ring. To overcome this drawback some ring topologies use dual rings. The topology is not useful to connect large number of computers.
**Star Topology:** In star topology all the nodes are connected to a central node called a hub. A node that wants to send some data to some other node on the network, send data to a hub which in turn sends it the destination node. A hub plays a major role in such networks.

![Star Topology Diagram](image)

Star topology is easy to install and reconfigure. If a link fails then it separates the node connected to link from the network and the network continues to function. However, if the hub goes down, the entire network collapses.

**Tree Topology:**

Tree topology is a hierarchy of various hubs. The entire nodes are connected to one hub or the other. There is a central hub to which only a few nodes are connected directly.

![Tree Topology Diagram](image)

The central hub, also called active hub, looks at the incoming bits and regenerates them so that they can traverse over longer distances. The secondary hubs in tree topology may be active hubs or passive hubs. The failure of a transmission line separates a node from the network.
Mesh Topology:

A mesh topology is also called complete topology. In this topology, each node is connected directly to every other node in the network. That is if there are n nodes then there would be \( \frac{n(n - 1)}{2} \) physical links in the network.

![Figure: Mesh Topology](image)

As there are dedicated links, the topology does not have congestion problems. Further it does not need a special Media Access Control (MAC) protocol to prevent simultaneous access to the transmission media since links are dedicated, not shared. The topology also provides data security.

The network can continue to function even in the failure of one of the links. Fault identification is also easy.

The main disadvantage of mesh topology is the complexity of the network and the cost associated with the cable length. The mesh topology is not useful for medium to large networks.

Hybrid Topology:

Hybrid topology is formed by connecting two or more topologies together. For example, hybrid topology can be created by using the bus, star and ring topologies, as shown in figure 22.6.

![Figure: Hybrid Topology](image)

4. Explain about Coaxial Cable and Twisted Pair.

Coaxial Cable:

Coaxial cable is a common transmission medium. It has better shielding than twisted pairs, so it can span longer distances at higher speeds. Two kinds of coaxial cable are widely used. One kind, 50-ohm cable, is commonly used when it is intended for digital transmission from the start. The other kind, 75-ohm cable, is commonly used for analog transmission and cable television but is becoming more...
important with the advent of Internet over cable. This distinction is based on historical, rather than technical, factors (e.g., early dipole antennas had an impedance of 300 ohms, and it was easy to use existing 4:1 impedance matching transformers).

A coaxial cable consists of a stiff copper wire as the core, surrounded by an insulating material. The insulator is encased by a cylindrical conductor, often as a closely-woven braided mesh. The outer conductor is covered in a protective plastic sheath. A cutaway view of a coaxial cable is shown in Fig.1.

**Fig.1: A coaxial cable.**

The construction and shielding of the coaxial cable give it a good combination of high bandwidth and excellent noise immunity. The bandwidth possible depends on the cable quality, length, and signal-to-noise ratio of the data signal. Modern cables have a bandwidth of close to 1 GHz. Coaxial cables used to be widely used within the telephone system for long-distance lines but have now largely been replaced by fiber optics on long-haul routes. Coax is still widely used for cable television and metropolitan area networks.

**Twisted Pair:**

Twisted pair is the oldest and most common transmission media. A twisted pair consists of two insulated copper wires, typically about 1 mm thick. The wires are twisted together in a helical form, just like a DNA molecule. Twisting is done because two parallel wires constitute a fine antenna. When the wires are twisted, the waves from different twists cancel out, so the wire radiates less effectively. The most common application of the twisted pair is the telephone system. Nearly all telephones are connected to the telephone company (Telco) office by a twisted pair.

Twisted pairs can run several kilometres without amplification, but for longer distances, repeaters are needed. When many twisted pairs run in parallel for a substantial distance, such as all the wires coming from an apartment building to the telephone company office, they are bundled together and encased in a protective sheath.

The pairs in these bundles would interfere with one another if it were not for the twisting. In parts of the world where telephone lines run on poles above ground, it is common to see bundles several centimetres in diameter. Twisted pairs can be used for transmitting either analog or digital signals. The bandwidth depends on the thickness of the wire and the distance traveled, but several megabits/sec can be achieved for a few kilometers in many cases. Due to their adequate performance and low cost, twisted pairs are widely used and are likely to remain so for years to come.

Twisted pair cabling comes in several varieties, two of which are important for computer networks. Category 3 twisted pairs consist of two insulated wires gently twisted together. Four such pairs are typically grouped in a plastic sheath to protect the wires and keep them together. Prior to about 1988, most office buildings had one category 3 cable running from a central wiring closet on each floor into each office. This scheme allowed up to four regular telephones or two multiline telephones in each office to connect to the telephone company equipment in the wiring closet.

All of these wiring types are often referred to as UTP (Unshielded Twisted Pair), to contrast them
with the bulky, expensive, shielded twisted pair cables IBM introduced in the early 1980s, but which have not proven popular outside of IBM installations. Twisted pair cabling is illustrated in Fig.2.

![Twisted pair cabling](image)

**Fig.2: (a) Category 3 UTP. (b) Category 5 UTP.**

### 5. Explain error correction and detection methods in the data link layer.

#### Error-Correcting Codes:

Network designers have developed two basic strategies for dealing with errors. One way is to include enough redundant information along with each block of data sent, to enable the receiver to deduce what the transmitted data must have been. The other way is to include only enough redundancy to allow the receiver to deduce that an error occurred, but not which error, and have it request a retransmission. The former strategy uses error-correcting codes and the latter uses error-detecting codes. The use of error-correcting codes is often referred to as forward error correction.

Each of these techniques occupies a different ecological niche. On channels that are highly reliable, such as fiber, it is cheaper to use an error detecting code and just retransmit the occasional block found to be faulty. However, on channels such as wireless links that make many errors, it is better to add enough redundancy to each block for the receiver to be able to figure out what the original block was, rather than relying on a retransmission, which itself may be in error.

To understand how errors can be handled, it is necessary to look closely at what an error really is. Normally, a frame consists of m data (i.e., message) bits and r redundant, or check, bits. Let the total length be n (i.e., n = m + r). An n-bit unit containing data and check bits is often referred to as an n-bit codeword.

Given any two code words, say, 10001001 and 10110001, it is possible to determine how many corresponding bits differ. In this case, 3 bits differ. To determine how many bits differ, just exclusive OR the two code words and count the number of 1 bits in the result, for example:

The number of bit positions in which two code words differ is called the Hamming distance. Its significance is that if two codewords are a Hamming distance d apart, it will require d single-bit errors to convert one into the other.

#### Error-Detecting Codes:

Error-correcting codes are widely used on wireless links, which are notoriously noisy and error prone when compared to copper wire or optical fibers. Without error-correcting codes, it would be hard to get anything through. However, over copper wire or fiber, the error rate is much lower, so error detection and retransmission is usually more efficient there for dealing with the occasional error. As a simple example, consider a channel on which errors are isolated and the error rate is 10-6 per bit. Let the block...
size be 1000 bits. To provide error correction for 1000-bit blocks, 10 check bits are needed; a megabit of data would require 10,000 check bits. To merely detect a block with a single 1-bit error, one parity bit per block will suffice. Once every 1000 blocks, an extra block (1001 bits) will have to be transmitted. The total overhead for the error detection + retransmission method is only 2001 bits per megabit of data, versus 10,000 bits for a Hamming code.

If a single parity bit is added to a block and the block is badly garbled by a long burst error, the probability that the error will be detected is only 0.5, which is hardly acceptable. The odds can be improved considerably if each block to be sent is regarded as rectangular matrix n bits wide and k bits high, as described above. A parity bit is computed separately for each column and affixed to the matrix as the last row. The matrix is then transmitted one row at a time. When the block arrives, the receiver checks all the parity bits. If any one of them is wrong, the receiver requests a retransmission of the block. Additional retransmissions are requested as needed until an entire block is received without any parity errors.

The algorithm for computing the checksum is as follows:
Let r be the degree of G(x). Append r zero bits to the low-order end of the frame so it now contains m + r bits and corresponds to the polynomial \( x^r \) M(x).

Divide the bit string corresponding to G(x) into the bit string corresponding to \( x^r \) M(x), using modulo 2 division.

Subtract the remainder (which is always r or fewer bits) from the bit string corresponding to \( x^r \) M(x) using modulo 2 subtractions. The result is the checksummed frame to be transmitted. Call its polynomial T(x).

Figure illustrates the calculation for a frame 1101011011 using the generator \( G(x) = x^4 + x + 1 \).

![Fig.5.1. Calculation of the polynomial code checksum](image-url)
OBJECTIVE QUESTIONS:

1. Computer Network is
   1. Collection of hardware components and computers
   2. Interconnected by communication channels
   A. Sharing of resources and information
   B. All of the Above

2. What is a Firewall in Computer Network?
   1. The physical boundary of Network
   2. An operating System of Computer Network
   3. A system designed to prevent unauthorized access
   5. A web browsing Software

3. How many layers does OSI Reference Model has?
   1. 4
   2. 5
   3. 6
   4. 7

4. DHCP is the abbreviation of
   1. Dynamic Host Control Protocol
   2. Dynamic Host Configuration Protocol
   3. Dynamic Hyper Control Protocol
   4. Dynamic Hyper Configuration Protocol

5. IPV4 Address is
   1. 8 bit
   2. 16 bit
   3. 32 bit
   4. 64 bit

6. DNS is the abbreviation of
   1. Dynamic Name System
   2. Dynamic Network System
   3. Domain Name System
   4. Domain Network Service

7. What is the meaning of Bandwidth in Network?
   a. Transmission capacity of a communication channels
   b. Connected Computers in the Network
   c. Class of IP used in Network
   d. None of Above

8. ADSL is the abbreviation of
   A. Asymmetric Dual Subscriber Line
   B. Asymmetric Digital System Line
   C. Asymmetric Dual System Line
   D. Asymmetric Digital Subscriber Line

9. What is the use of Bridge in Network?
   1. to connect LANs
   2. to separate LANs
   3. to control Network Speed
   4. All of the above

10. Router operates in which layer of OSI Reference Model?
A. Layer 1 (Physical Layer)
B. Layer 3 (Network Layer)
C. Layer 4 (Transport Layer)
D. Layer 7 (Application Layer)

Click Here for Answers

<table>
<thead>
<tr>
<th>Q. NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>D</td>
</tr>
<tr>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>D</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
</tr>
<tr>
<td>5</td>
<td>C</td>
</tr>
<tr>
<td>6</td>
<td>C</td>
</tr>
<tr>
<td>7</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>D</td>
</tr>
<tr>
<td>9</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>B</td>
</tr>
</tbody>
</table>

Fill in the blanks
1. PPP stands for --------------.
2. A conversation on walkie-talkie is a -------------- data flow.
3. In data communication, the interface between the source and the medium is called ----.
4. You can host Internet and Intranet websites on a Linux server by --------------.
5. The connection between the modem and terminal/computer is a --------------.
6. Routers are part of the -------------- layer.
7. A computer network that spans a relatively large geographical area is called ------.
8. WAN stands for --------------.
9. ISP denotes --------------.
10. Ethernet uses an access method called -----------.

KEY

<table>
<thead>
<tr>
<th>Q. NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Point-to-point protocol</td>
</tr>
<tr>
<td>2</td>
<td>Half-Duplex</td>
</tr>
<tr>
<td>3</td>
<td>DCE</td>
</tr>
<tr>
<td>4</td>
<td>Apache server</td>
</tr>
<tr>
<td>5</td>
<td>Digital</td>
</tr>
<tr>
<td>6</td>
<td>Data link</td>
</tr>
<tr>
<td>7</td>
<td>WAN</td>
</tr>
<tr>
<td>8</td>
<td>Wide Area Network</td>
</tr>
<tr>
<td>9</td>
<td>Internet service provider</td>
</tr>
<tr>
<td>10</td>
<td>CSMA/CD</td>
</tr>
</tbody>
</table>
UNIT – 2

DATA COMMUNICATION AND COMPUTER NETWORKS

DATA LINK LAYER

2 mark questions with answers:

1. **Mention the types of errors.**
   There are 2 types of errors
   a) Single-bit error.
   b) Burst-bit error.

2. **Define the following terms.**
   Single bit error: The term single bit error means that only one bit of a given data unit (such as byte character/data unit or packet) is changed from 1 to 0 or from 0 to 1.
   Burst error: Means that 2 or more bits in the data unit have changed from 1 to 0 from 0 to 1

3. **What is redundancy?**
   It is the error detecting mechanism, which means a shorter group of bits or extra bits may be appended at the destination of each unit.

4. **Write short notes on LRC.**
   In longitudinal redundancy check (LRC), a block of bits is divided into rows and a redundant row of bits is added to the whole block.

5. **Write short notes on CRC.**
   The third and most powerful of the redundancy checking techniques is the cyclic redundancy checks (CRC). CRC is based on binary division. Here a sequence of redundant bits, called the CRC remainder is appended to the end of data unit.

3 mark questions and answers:

1. **What are the responsibilities of data link layer?**
   Specific responsibilities of data link layer include the following.
   1. Framing
   2. Physical addressing
   3. Flow control
   4. Error control
   5. Access control
   6.

2. **List out the available detection methods.**
   There are 4 types of redundancy checks are used in data communication.
   i. Vertical redundancy checks (VRC).
   ii. Longitudinal redundancy checks (LRC).
   iii. Cyclic redundancy checks (CRC).
   iv. Checksum.

3. **List out the steps followed is checksum checker side.**
   The receiver must follow these steps
   a) The unit is divided into k section each of n bits.
   b) All sections are added together using 1’s complement to get the sum .The sum is complemented.If the result is zero.
4. Mention the categories of flow control.
   There are 2 methods have been developed to control flow of data across
   communication links.
   Stop and wait- send one from at a time.
   Sliding window- send several frames at a time.

5. Write short notes on VRC.
   The most common and least expensive mechanism for error detection is the vertical redundancy
   check (VRC) often called a parity check. In this technique a redundant bit called a parity bit, is
   appended to every data unit so that the total number of 0’s in the unit (including the parity bit)
   becomes even.

5 mark questions and answers:

Q1. What is pure ALOHA and Slotted ALOHA? Explain in detail?

ALOHA:
   In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and
elegant method to solve the channel allocation problem. Their work has been extended by many
researchers since then (Abramson, 1985). Although Abramson’s work, called the ALOHA system, used
ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users
are competing for the use of a single shared channel.
   We will discuss two versions of ALOHA here: pure and slotted. They differ with respect to whether or
not time is divided up into discrete slots into which all frames must fit. Pure ALOHA does not require
global time synchronization; slotted ALOHA does.

Pure ALOHA
   The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent.
   There will be collisions, of course, and the colliding frames will be damaged. However, due to the
feedback property of broadcasting, a sender can always find out whether or not its frame was destroyed by
listening to the channel, the same way other users do. With a LAN, the feedback is immediate; with a
satellite, there is a delay of 270 msec before the sender knows if the transmission was successful. If
listening while transmitting is not possible for some reason, acknowledgements are needed. If the frame
was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must
be random or the same frames will collide over and over, in lockstep. Systems in which multiple users
share a common channel in a way that can lead to conflicts are widely known as contention systems.

A sketch of frame generation in an ALOHA system is given in Fig. 4-1. We have made the frames all the
same length because the throughput of ALOHA systems is maximized by having a uniform frame size
rather than allowing variable length frames.
In pure ALOHA, frames are transmitted at completely arbitrary times.

Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled. If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed, and both will have to be retransmitted later. The checksum cannot (and should not) distinguish between a total loss and a near miss. Bad is bad.

A most interesting question is: What is the efficiency of an ALOHA channel? That is, what fractions of all transmitted frames escape collisions under these chaotic circumstances? Let us first consider an infinite collection of interactive users sitting at their computers (stations). A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. The station then transmits a frame containing the line and checks the channel to see if it was successful. If so, the user sees the reply and goes back to typing. If not, the user continues to wait and the frame is retransmitted over and over until it has been successfully sent.

**Slotted ALOHA**

In 1972, Roberts published a method for doubling the capacity of an ALOHA system (Roberts, 1972). His proposal was to divide time up into discrete intervals, each interval corresponding to one frame. This approach requires the users to agree on slot boundaries. One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock.

Slotted ALOHA, in contrast to Abramson’s pure ALOHA, a computer is not permitted to send whenever a carriage return is typed. Instead, it is required to wait for the beginning of the next slot. Thus the continuous pure ALOHA is turned into a discrete one. Since the vulnerable period is now halved, the probability of no other traffic during the same slot as our test frame is $e^{-G}$ which leads to

$$S \geq Ge^{-G} \quad (4-3)$$

As you can see from Fig. 4-3, slotted ALOHA peaks at $G = 1$, with a throughput of $S = 1/e$ or about 0.368, twice that of pure ALOHA. If the system is operating at $G \geq 1$, the probability of an empty slot is 0.368 (from Eq. 4-2). The best we can hope for using slotted ALOHA is 37 percent of the slots empty, 37 percent successes, and 26 percent collisions. Operating at higher values of $G$ reduces the number of empties but increases the number of collisions exponentially. To see how this rapid growth of collisions with $G$ comes about, consider the transmission of a test frame. The probability that it will avoid a collision is $e^{-G}$, the probability that all the other users are silent in that slot. The probability of a collision is then $1 - e^{-G}$. The probability of a transmission requiring exactly $k$ attempts, (i.e., $k - 1$ collisions followed by one success) is

$$P_k = e^{-G} (1 - e^{-G})^{k-1}$$

The expected number of transmissions, $E$, per carriage return typed is then

$$E \geq \sum_{k=1}^{\infty} kP_k \geq \sum_{k=1}^{\infty} ke^{-G} (1 - 1) \geq 1 \geq e^G$$

As a result of the exponential dependence of $E$ upon $G$, small increases in the channel load can drastically reduce its performance.

Slotted Aloha is important for a reason that may not be initially obvious. It was devised in the 1970s, used in a few early experimental systems, then almost forgotten. When Internet access over the cable was invented, all of a sudden there was a problem of how to allocate a shared channel among multiple
competing users, and slotted Aloha was pulled out of the garbage can to save the day. It has often happened that protocols that are perfectly valid fall into disuse for political reasons (e.g., some big company wants everyone to do things its way), but years later some clever person realizes that a long-discarded protocol solves his current problem. For this reason, in this chapter we will study a number of elegant proto-cols that are not currently in widespread use, but might easily be used in future applications, provided that enough network designers are aware of them. Of course, we will study various protocols that are in current use as well.

Q2. Explain the Persistent Methods in detail?

Carrier Sense Multiple Access Protocols:

Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called carrier sense protocols. A number of them have been proposed. Kleinrock and Tobagi (1975) have analyzed several such protocols in detail. Below we will mention several versions of the carrier sense protocols.

Persistent and Non persistent CSMA

The first carrier sense protocol that we will study here is called 1-persistent CSMA (Carrier Sense Multiple Access). When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1-persistent because the station transmits with a probability of 1 whenever it finds the channel idle.

The propagation delay has an important effect on the performance of the protocol. There is a small chance that just after a station begins sending, another station will become ready to send and sense the channel. If the data frame has not yet reached the second one, the latter will sense an idle channel and will also begin sending, resulting in a collision. The longer the propagation delay, the more important this effect becomes, and the worse the performance of the protocol.

Even if the propagation delay is zero, there will still be collisions. If two stations become ready in the middle of a third station’s transmission, both will wait politely until the transmission ends and then both will begin transmitting exactly simultaneously, resulting in a collision. If they were not so impatient, there would be fewer collisions. Even so, this protocol is far better than pure ALOHA, because both stations have the decency to desist from interfering with the third station’s frame. Intuitively, this will lead to a higher performance than pure ALOHA. Exactly the same holds for slotted ALOHA.

A second carrier sense protocol is nonpersistent CSMA. In this protocol, a conscious attempt is made to be less greedy than in the previous one. Before sending, a station senses the channel. If no one else is sending, the station begins doing so itself. However, if the channel is already in use, the station does not con-tinually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and then repeats the algorithm. Intuitively this algorithm should lead to better channel utilization and longer delays than 1-persistent CSMA.

The last protocol is p-persistent CSMA. It applies to slotted channels and works as follows. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability p. With a probability q = 1 − p it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities p and q. This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, it acts as if there had been a collision (i.e., it waits a random time and starts again). If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm. Figure 4-4 shows the computed throughput versus offered traffic for all three protocols, as well as for pure and slotted ALOHA.
Q3. Explain about Bit – Map Collision free protocols?

Collision-Free Protocols

Although collisions do not occur with CSMA/CD once a station has unambiguously seized the channel, they can still occur during the contention period. These collisions adversely affect the system performance, especially when the cable is long (i.e., large \( \tau \)) and the frames are short. As very long, high-bandwidth fiber optic networks come into use, the combination of large \( \tau \) and short frames will become an increasingly serious problem. In this section, we will examine some protocols that resolve the contention for the channel without any collisions at all, not even during the contention period.

In the protocols to be described, we make the assumption that there are exactly \( N \) stations, each with a unique address from 0 to \( N - 1 \). It does not matter that some stations may be inactive part of the time. We also assume that propagation delay is negligible. The basic question remains: Which station gets the channel after a successful transmission? We continue using the model of Fig. 4-5 with its discrete contention slots.

A Bit-Map Protocol

In our first collision-free protocol, the basic bit-map method, each contention period consists of exactly \( N \) slots. If station 0 has a frame to send, it transmits a 1 bit during the zeroth slot. No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 during slot 1, but only if it has a frame queued. In general, station \( j \) may announce the fact that it has a frame to send by inserting a 1 bit into slot \( j \). After all \( N \) slots have passed by, each station has complete knowledge of which stations wish to transmit. At that point, they begin transmitting in numerical order (see Fig. 4-6).

```
<table>
<thead>
<tr>
<th>8 Contention slots</th>
<th>8 Contention slots</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5 6</td>
<td>0 1 2 3 4 5 6</td>
<td>0 1 2 3 4 5 6</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>
```

Figure 4-6. The basic bit-map protocol.

Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another \( N \) bit contention period is begun. If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again. Protocols like this in which the desire to transmit is broadcast before the actual transmission are called reservation protocols.

Let us briefly analyze the performance of this protocol. For convenience, we will measure time in units of the contention bit slot, with data frames consisting of \( d \) time units. Under conditions of low load, the bit map will simply be repeated over and over, for lack of data frames.

Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, when it becomes ready to send, the “current” slot will be somewhere in the middle of the bit map. On the average, the station will have to wait \( N/2 \) slots for the current scan to finish and another full \( N \) slots for
the following scan to run to completion before it may begin transmitting.

The prospects for high-numbered stations are brighter. Generally, these will only have to wait half a scan \((N/2)\) bit slots) before starting to transmit. High-numbered stations rarely have to wait for the next scan. Since low-numbered stations must wait on the average \(1.5N\) slots and high-numbered stations must wait on the average \(0.5N\) slots, the mean for all stations is \(N\) slots. The channel efficiency at low load is easy to compute. The overhead per frame is \(N\) bits, and the amount of data is \(d\) bits, for an efficiency of \(d / (N \cdot d)\).

At high load, when all the stations have something to send all the time, the \(N\) bit contention period is prorated over \(N\) frames, yielding an overhead of only 1 bit per frame, or an efficiency of \(d / (d + 1)\). The mean delay for a frame is equal to the sum of the time it queues inside its station, plus an additional \(N / (d + 1)\) once it gets to the head of its internal queue.

**Binary Countdown**

A problem with the basic bit-map protocol is that the overhead is 1 bit per station so it does not scale well to networks with thousands of stations. We can do better than that by using binary station addresses. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. The bits in each address position from different stations are BOOLEAN ORed together. We will call this protocol **binary countdown**. It was used in Datakit (Fraser, 1987). It implicitly assumes that the transmission delays are negligible so that all stations see asserted bits essentially instantaneously.

To avoid conflicts, an arbitration rule must be applied: as soon as a station sees that a high-order bit position that is 0 in its address has been overwritten with a 1, it gives up. For example, if stations 0010, 0100, 1001, and 1010 are all trying to get the channel, in the first bit time the stations transmit 0, 0, 1, and 1, respectively. These are ORed together to form a 1. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue.

The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010, because it has the highest address. After winning the bidding, it may now transmit a frame, after which another bidding cycle starts. The protocol is illustrated in Fig. 4-7. It has the property that higher-numbered stations have a higher priority than lower-numbered stations, which may be either good or bad, depending on the context.

<table>
<thead>
<tr>
<th>Bit time</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Result</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Stations</td>
<td>0010</td>
<td>1001</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure 4-7. The binary countdown protocol. A dash indicates silence.

The channel efficiency of this method is \( d / (d \log_2 N) \). If, however, the frame format has been cleverly chosen so that the sender’s address is the first field in the frame, even these \( \log_2 N \) bits are not wasted, and the efficiency is 100 per-cent.

Mok and Ward (1979) have described a variation of binary countdown using a parallel rather than a serial interface. They also suggest using virtual station numbers, with the virtual station numbers from 0 up to and including the successful station being circularly permuted after each transmission, in order to give higher priority to stations that have been silent unusually long. For example, if stations C, H, D, A, G, B, E, F have priorities 7, 6, 5, 4, 3, 2, 1, and 0, respectively, then a successful transmission by D puts it at the end of the list, giving a priority order of C, H, A, G, B, E, F, D. Thus C remains virtual station 7, but A moves up from 4 to 5 and D drops from 5 to 0. Station D will now only be able to acquire the channel if no other station wants it.

Binary countdown is an example of a simple, elegant, and efficient protocol that is waiting to be rediscovered. Hopefully, it will find a new home some day.

Q4. What is Adaptive Tree Protocol? Why it is considered as collision free protocol?

The Adaptive Tree Walk Protocol

One particularly simple way of performing the necessary assignment is to use the algorithm devised by the U.S. Army for testing soldiers for syphilis during World War II (Dorfman, 1943). In short, the Army took a blood sample from \( N \) soldiers. A portion of each sample was poured into a single test tube. This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through \( N/2 \) and one from the rest. The process was repeated recursively until the infected soldiers were determined.

For the computer version of this algorithm (Capetanakis, 1979) it is convenient to think of the stations as the leaves of a binary tree, as illustrated in Fig. 4-9. In the first contention slot following a successful frame transmission, slot 0, all stations are permitted to try to acquire the channel. If one of them does so, fine. If there is a collision, then during slot 1 only those stations falling under node 2 in the tree may compete. If one of them acquires the channel, the slot following the frame is reserved for those stations under node 3. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4’s turn during slot 2.
In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node’s left and right children. If a bit slot is idle or if there is only one station that transmits in it, the searching of its node can stop, because all ready stations have been located. (Were there more than one, there would have been a collision.)

When the load on the system is heavy, it is hardly worth the effort to dedicate slot 0 to node 1, because that makes sense only in the unlikely event that precisely one station has a frame to send. Similarly, one could argue that nodes 2 and 3 should be skipped as well for the same reason. Put in more general terms, at what level in the tree should the search begin? Clearly, the heavier the load, the farther down the tree the search should begin. We will assume that each station has a good estimate of the number of ready stations, \( q \), for example, from monitoring recent traffic.

To proceed, let us number the levels of the tree from the top, with node 1 in Fig. 4-9 at level 0, nodes 2 and 3 at level 1, etc. Notice that each node at level \( i \) has a fraction \( 2^{-i} \) of the stations below it. If the \( q \) ready stations are uniformly distributed, the expected number of them below a specific node at level \( i \) is just \( 2^{-i} q \). Intuitively, we would expect the optimal level to begin searching the tree as the one at which the mean number of contending stations per slot is 1, that is, the level at which \( 2^{-i} q = 1 \). Solving this equation we find that \( i = \log_2 q \).

Numerous improvements to the basic algorithm have been discovered and are discussed in some detail by Bertsekas and Gallager (1992). For example, consider the case of stations \( G \) and \( H \) being the only ones wanting to transmit. At node 1 a collision will occur, so 2 will be tried and discovered idle. It is pointless to probe node 3 since it is guaranteed to have a collision (we know that two or more stations under 1 are ready and none of them are under 2 so they must all be under 3). The probe of 3 can be skipped and 6 tried next. When this probe also turns up nothing, 7 can be skipped and node \( G \) tried next.

Q5. Write about cable topologies of Ethernet?

Ethernet Cabling

Since the name “Ethernet” refers to the cable (the ether), let us start our discussion there. Four types of cabling are commonly used, as shown in Fig. 4-13. Historically, **10Base5** cabling, popularly called **thick Ethernet**, came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go. (The 802.3 standard does not actually require the cable to be yellow, but it does suggest it.) Connections to it are generally made using **vampire taps**, in which a pin is very carefully forced halfway into the coaxial cables core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500 meters. In other words, the first number is finally an indication of the transmission medium. If the medium is coax, its length is given in units of 100
m.

Detecting cable breaks, bad taps, or loose connectors can be a major problem with both media. For this reason, techniques have been developed to track them down. Basically, a pulse of known shape is injected into the cable. If the pulse hits an obstacle or the end of the cable, an echo will be generated and sent back. By carefully timing the interval between sending the pulse and receiving the echo, it is possible to localize the origin of the echo. This technique is called **time domain reflectometry**.

The problems associated with finding cable drove systems toward a different kind of wiring pattern, in which all stations have a cable running to a central **hub** in which they are all connected electrically. Usually, these wires are telephone company twisted pairs, since most office buildings are already wired this way, and there are normally plenty of spare pairs available. This scheme is called **10Base-T**. We will discuss an improved version of this idea—switches—later in this chapter.

These three wiring schemes are illustrated in Fig. 4-14. For 10Base5, a **transceiver** is clamped securely around the cable so that its tap makes contact with the inner core. The transceiver contains the electronics that handle carrier detection and collision detection. When a collision is detected, the transceiver also puts a special invalid signal on the cable to ensure that all other transceivers also realize that a collision has occurred.

With 10Base5, a **transceiver cable** connects the transceiver to an interface board in the computer. The transceiver cable may be up to 50 meters long and contains five individually shielded twisted pairs. Two of the pairs are for data in and data out, respectively. Two more are for control signals in and out. The fifth pair, which is not always used, allows the computer to power the transceiver electronics. Some transceivers allow up to eight nearby computers to be attached to them, to reduce the number of transceivers needed.

The transceiver cable terminates on an interface board inside the computer.

With 10Base2, the connection to the cable is just a passive BNC T-junction connector. The transceiver electronics are on the controller board, and each station always has its own transceiver.

With 10Base-T, there is no cable at all, just the hub (a box full of electronics). Adding or removing a station is simpler in this configuration, and cable breaks can be detected easily. The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100 meters, maybe 200 meters if very high-quality category 5 twisted pairs are used. Nevertheless, 10Base-T quickly became dominant due to its use of existing wiring and the ease of maintenance that it offers. A faster version of 10Base-T (100Base-T) will be discussed later in this chapter.

A fourth cabling option for Ethernet is **10Base-F**, which uses fiber optics. This alternative is expensive due to the cost of the connectors and terminators, but it has excellent noise immunity and is the method of choice when running between buildings or widely separated hubs. Runs of up to km are allowed. It also offers good security since wiretapping fiber is much more difficult than wire-tapping copper wire.

different ways of wiring up a building. In Fig. (a), a single cable is snaked from room to room, with each station tapping onto it at the nearest point. In Fig. (b), a vertical spine runs from the basement to the roof, with horizontal cables on each floor connected to it by special amplifiers (repeaters). In some buildings the horizontal cables are thin, and the backbone is thick. The most general topology is the tree, as in Fig. 4-15(c), because a network with two paths between some pairs of stations would suffer from interference between the two signals.
Each version of Ethernet has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by **repeaters**, as shown in Fig. 4-15(d). A repeater is a physical layer device. It receives, amplifies, and re-transmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different than a single cable (except for some delay introduced by the repeaters). A system may contain multiple cable segments and multiple repeaters, but no two transceivers may be more than 2.5 km apart and no path between any two transceivers may traverse more than four repeaters.

**OBJECTIVE TYPE QUESTIONS**

1 **Each IP packet must contain**
   A. Only Source address  
   B. Only Destination address  
   C. Source and Destination address  
   D. Source or Destination address

2 **Bridge works in which layer of the OSI model?**
   A. Application layer  
   B. Transport layer  
   C. Network layer  
   D. Datalink layer

3 _______ provides a connection-oriented reliable service for sending messages
   A. TCP  
   B. IP  
   C. UDP  
   D. All of the above

4 **Which layers of the OSI model are host-to-host layers?**
   A. Transport, Session, Presentation, Application  
   B. Network, Transport, Session, Presentation  
   C. Datalink, Network, Transport, Session  
   D. Physical, Datalink, Network, Transport

5 **Which of the following IP address class is Multicast**
   A. Class A  
   B. Class B  
   C. Class C  
   D. Class D
6 Which of the following is correct regarding Class B Address of IP address
A. Network bit – 14, Host bit – 16
B. Network bit – 16, Host bit – 14
C. Network bit – 18, Host bit – 16
D. Network bit – 12, Host bit – 14

7 The last address of IP address represents
A. Unicast address
B. Network address
C. Broadcast address
D. None of above

8 How many bits are there in the Ethernet address?
A. 64 bits
B. 48 bits
C. 32 bits
D. 16 bits

9 How many layers are in the TCP/IP model?
A. 4 layers
B. 5 layers
C. 6 layers
D. 7 layers

10 Which of the following layer of OSI model also called end-to-end layer?
A. Presentation layer
B. Network layer
C. Session layer
D. Transport layer

Answers

<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>C</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
</tr>
<tr>
<td>5</td>
<td>D</td>
</tr>
<tr>
<td>6</td>
<td>A</td>
</tr>
<tr>
<td>7</td>
<td>C</td>
</tr>
<tr>
<td>8</td>
<td>B</td>
</tr>
<tr>
<td>9</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>D</td>
</tr>
</tbody>
</table>

Fill in the blanks
1. A modem is a ---------- device.
2. ---------- is a protocol which allows users to download E Mail messages from mail server to a local computer.
3. ---------- is used to access and operate a remote computer on a network.
4. ---------- is the service in Windows 2000 that allows computers to connect to the server using Dial Up networking facility.
5. Gateways used in VPN are called ---------- gateways.
6. ---------- is a way of sending several channels over a single line.
7. ---------- provides ISP in India.
8. The language used to develop web pages is called ----------.
9. A network of networks is known as ----------.
10. In a network a machine is identified by unique address called ----------.
<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bidirectional</td>
</tr>
<tr>
<td>2</td>
<td>IMAP</td>
</tr>
<tr>
<td>3</td>
<td>Telnet</td>
</tr>
<tr>
<td>4</td>
<td>RAS</td>
</tr>
<tr>
<td>5</td>
<td>Tunneling</td>
</tr>
<tr>
<td>6</td>
<td>Multiplexing</td>
</tr>
<tr>
<td>7</td>
<td>VSNL</td>
</tr>
<tr>
<td>8</td>
<td>HTML</td>
</tr>
<tr>
<td>9</td>
<td>Internet</td>
</tr>
<tr>
<td>10</td>
<td>IP address</td>
</tr>
</tbody>
</table>
UNIT – 3
Network Layer

2 MARK question and answers.

1. **What are the network support layers and the user support layers?**

   **Network support layers:**
   The network support layers are Physical layer, Data link layer and Network layer. These deals with electrical specifications, physical connection, transport timing and reliability.

   **User support layers:**
   The user support layers are: Session layer, Presentation layer, Application layer. These allow interoperability among unrelated software systems.

2. **With a neat diagram explain the relationship of IEEE Project to the OSI model?**

   The IEEE has subdivided the data link layer into two sub layers:
   1. Logical link control (LLC)
   2. Medium access control (MAC)

   LLC is non-architecture specific. The MAC sub layer contains a number of distinct modules, each carries proprietary information specific to the LAN product being used.

3. **What are the functions of LLC?**

   The IEEE project 802 models takes the structure of an HDLC frame and divides it into 2 sets of functions. One set contains the end user portion of the HDLC frame – the Other layers Network Data link Physical Other layers Network Logical Link Control

   Media Access Control Physical logical address, control information, and data. These functions are handled by the IEEE 802.2 logical link control (LLC) protocol.

4. **What are the functions of MAC?**

   MAC sub layer resolves the contention for the shared media. It contains synchronization, flag, flow and error control specifications necessary to move information from one place to another, as well as the physical address of the next station to receive and route a packet.

5. **What is protocol data unit?**

   The data unit in the LLC level is called Protocol Data Unit (PDU). It contains four fields. Destination Service Point Address (DSAP)

   Source Service Access Point

   Control field
3 marks questions and answers.

1. **What is meant by hop count?**
The pathway requiring the smallest number of relays, it is called hop-count routing, in which every link is considered to be of equal length and given the value one.

2. **How can the routing be classified? The routing can be classified as, Adaptive routing**
Non-adaptive routing. What is time-to-live or packet lifetime?
As the time-to-live field is generated, each packet is marked with a lifetime, usually the number of hops that are allowed before a packet is considered lost and accordingly, destroyed. The time- to-live determines the lifetime of a packet.

3. **What is meant by brouter?**
A brouter is a single protocol or multiprotocol router that sometimes acts as a router and sometimes act as a bridge.

4. **Write the keys for understanding the distance vector routing.**
The three keys for understanding the algorithm are Knowledge about the whole networks Routing only to neighbors Information sharing at regular intervals

5. **Write the keys for understanding the link state routing.**
The three keys for understanding the algorithm are Knowledge about the neighborhood.Routing to all neighbors. Information sharing when there is a range.

5 mark questions and answers:

**Q1. Write about unicast routing protocol?**
A routing table can be either static or dynamic. A static table is one with manual entries. A dynamic table, on the other hand, is one that is updated automatically when there is a change somewhere in the internet. Today, an internet needs dynamic routing tables. The tables need to be updated as soon as there is a change in the internet. For instance, they need to be updated when a router is down, and they need to be updated whenever a better route has been found. Routing protocols have been created in response to the demand for dynamic routing tables. A routing protocol is a combination of rules and procedures that let routers in the internet inform each other of changes. It allows routers to share whatever they know about the internet or their neighborhood. The sharing of information allows a router in San Francisco to know about the failure of a network in Texas. The routing protocols also include procedures for combining information received from other routers.

**Optimization**
A router receives a packet from a network and passes it to another network. A router is usually attached to several networks. When it receives a packet, to which network should it pass the packet? The decision is based on optimization: Which of the available pathways is the optimum pathway? What is the definition of the term optimum? One approach is to assign a cost for passing through a network. We call this cost a metric. However, the metric assigned to each network depends on the type of protocol. Some simple protocols, such as the Routing Information Protocol (RIP), treat all networks as equals. The cost of passing through a network is the
same; it is one hop count. So if a packet passes through 10 networks to reach the destination, the total cost is 10 hop counts.

Distance Vector Routing
In distance vector routing, the least-cost route between any two nodes is the route with minimum distance. In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node. The table at each node also guides the packets to the desired node by showing the next stop in the route (next-hop routing). We can think of nodes as the cities in an area and the lines as the roads connecting them. A table can show a tourist the minimum distance between cities.

![Table for nodes A, B, C, D, E]

The table for node A shows how we can reach any node from this node. For example, our least cost to reach node E is 6. The route passes through C.

Initialization
Each node knows how to reach any other node and the cost. At the beginning, however, this is not the case. Each node can know only the distance between itself and its immediate neighbors, those directly connected to it. So for the moment, we assume that each node can send a message to the immediate neighbors and find the distance between itself and these neighbors. Figure 22.15 shows the initial tables for each node. The distance for any entry that is not a neighbor is marked as infinite (unreachable).

Q2. What is routing algorithm? What are the classifications of it?

The main function of the network layer is routing packets from the source machine to the destination machine. In most subnets, packets will require multiple hops to make the journey. The only notable exception is for broadcast networks, but even here routing is an issue if the source and destination are not on the same network. The algorithms that choose the routes and the data structures that they use are a major area of network layer design.

The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagrams internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. If the subnet uses virtual circuits internally, routing decisions are made only when a new virtual circuit is being set up. Thereafter, data packets just follow the previously-established route. The latter case is sometimes called session...
Routing because a route remains in force for an entire user session (e.g., a login session at a terminal or a file transfer).

It is sometimes useful to make a distinction between routing, which is making the decision which routes to use, and forwarding, which is what happens when a packet arrives. One can think of a router as having two processes inside it. One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing tables. This process is forwarding. The other process is responsible for filling in and updating the routing tables.

Regardless of whether routes are chosen independently for each packet or only when new connections are established, certain properties are desirable in a routing algorithm: correctness, simplicity, robustness, stability, fairness, and optimality. Correctness and simplicity hardly require comment, but the need for robustness may be less obvious at first. Once a major network comes on the air, it may be expected to run continuously for years without system wide failures. During that period there will be hardware and software failures of all kinds. Hosts, routers, and lines will fail repeatedly, and the topology will change many times. The routing algorithm should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network to be rebooted every time some router crashes.

Stability is also an important goal for the routing algorithm. There exist routing algorithms that never converge to equilibrium, no matter how long they run. A stable algorithm reaches equilibrium and stays there. Fairness and optimality may sound obvious.

Minimizing mean packet delay is an obvious candidate, but so is maximizing total network throughput. Furthermore, these two goals are also in conflict, since operating any queuing system near capacity implies a long queuing delay. As a compromise, many networks attempt to minimize the number of hops a packet must make, because reducing the number of hops tends to improve the delay and also reduce the amount of bandwidth consumed, which tends to improve the throughput as well.

Routing algorithms can be grouped into two major classes: non-adaptive and adaptive. Non-adaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J (for all I and J) is computed in advance, off-line, and downloaded to the routers when the network is booted. This procedure is sometimes called static routing. Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get their information (e.g., locally, from adjacent
routers, or from all routers), when they change the routes (e.g., every DT sec, when the load changes or when the topology changes), and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time).

Q3. Explain the Optimality Principle?

One can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the optimality principle. It states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route. To see this, call the part of the route from I to Jr1 and the rest of the route r2. If a route better than r2 existed from J to K, it could be concatenated with r1 to improve the route from I to K, contradicting our statement that r1r2 is optimal.

As a direct consequence of the optimality principle, one can see that the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a sink tree and is illustrated in Fig.6, where the distance metric is the number of hops. Note that a sink tree is not necessarily unique; other trees with the same path lengths may exist. The goal of all routing algorithms is to discover and use the sink trees for all routers.

![Fig.6 (a) A subnet. (b) A sink tree for router B.](image)

Since a sink tree is indeed a tree, it does not contain any loops, so each packet will be delivered within a finite and bounded number of hops. Links and routers can go down and come back up during operation, so different routers may have different ideas about the current topology. The optimality principle and the sink tree provide a benchmark against which other routing algorithms can be measured.

Q4. Explain shortest path routing algorithm?

The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line (often called a link). To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.

One way of measuring path length is the number of hops. Using this metric, the paths ABC and ABE in Fig.7 are equally long. Another metric is the geographic distance in kilometers, in which case ABC is clearly much longer than ABE (assuming the figure is drawn to scale).

Several algorithms for computing the shortest path between two nodes of a graph are known. This one is due to Dijkstra (1959). Each node is labeled (in parentheses) with its distance from the source node along the best known path. Initially, no paths are known, so all nodes are labeled with infinity. As the algorithm proceeds...
and paths are found, the labels may change, reflecting better paths. A label may be either tentative or permanent. Initially, all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

To illustrate how the labeling algorithm works, look at the weighted, undirected graph of Fig. 7(a), where the weights represent, for example, distance. We want to find the shortest path from A to D. Mark node A as permanent, indicated by a filled-in circle. Then examine, in turn, each of the nodes adjacent to A (the working node), relabeling each one with the distance to A. Whenever a node is relabeled, label it with the node from which the probe was made so that one can reconstruct the final path later. Having examined each of the nodes adjacent to A, examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent, as shown in Fig. 7(b). This becomes the new working node.

Now start at B and examine all nodes adjacent to it. If the sum of the label on B and the distance from B to the node being considered is less than the label on that node, is is the shorter path, so the node is relabeled.

After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively-labeled node with the smallest value. This node is made permanent and becomes the working node for the next round. Fig. 7 shows the first five steps of the algorithm.

To see why the algorithm works, consider Fig. 7(c). At that point E is made permanent. Suppose that there were a shorter path than ABE, say AXYZE. There are two possibilities: either node Z has already been made permanent, or it has not been. If it has, then E has already been probed (on the round following the one when Z was made permanent), so the AXYZE path has not escaped our attention and thus cannot be a shorter path.

Now consider the case where Z is still tentatively labeled. Either the label at Z is greater than or equal to that at E, in which case AXYZE cannot be a shorter path than ABE, or it is less than that of E, in which case Z
Q5. Explain distance vector routing algorithm?

Distance vector routing algorithms operate by having each router maintain a table (i.e. a vector) giving the best known distance to each destination and which line to use to get there. These tables are updated by exchanging information with the neighbors.

The distance vector routing algorithm is sometimes called by other names, most commonly the distributed Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm; it was the original ARPANET routing algorithm and was also used in the Internet under the name RIP.

The router is assumed to know the "distance" to each of its neighbors. If the metric is hops, the distance is just one hop. If the metric is queue length, the router simply examines each queue. If the metric is delay, the router can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can.

As an example, assume that delay is used as a metric and that the router knows the delay to each of its neighbors. Once every T msec each router sends to each neighbor a list of its estimated delays to each destination. It also receives a similar list from each neighbor. Imagine that one of these tables has just come in from neighbor X, with Xi being X's estimate of how long it takes to get to router i. If the router knows that the delay to X is m msec, it also knows that it can reach router i via X in Xi + m msec. By performing this calculation for each neighbor, a router can find out which estimate seems the best and use that estimate and the corresponding line in its new routing table. Note that the old routing table is not used in the calculation.

This updating process is illustrated in Fig.9 Part (a) shows a subnet. The first four columns of part (b) show the delay vectors received from the neighbors of router J. A claims to have a 12-msec delay to B, a 25-msec delay to C, a 40-msec delay to D, etc. Suppose that J has measured or estimated its delay to its neighbors, A, I, H, and K as 8, 10, 12, and 6 msec.

Consider how J computes its new route to router G. It knows that it can get to A in 8 msec, an A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec to G it forwards packets bound for
G to A. Similarly, it computes the delay to G via I, H, and K as 4 (31 + 10), 18 (6 + 12), and 37 (31 + 6) msec, respectively. The best of these values is 18, so it makes an entry in its routing table that the delay to G is 18 msec and that the route to use is via H. The same calculation is performed for all the other destinations, with the new routing table shown in the last column of the figure.

OBJECTIVE TYPE QUESTIONS

1. Why IP Protocol is considered as unreliable?
   A. A packet may be lost
   B. Packets may arrive out of order
   C. Duplicate packets may be generated
   D. All of the above

2. What is the minimum header size of an IP packet?
   A. 16 bytes
   B. 10 bytes
   C. 20 bytes
   D. 32 bytes

3. Which of following provides reliable communication?
   A. TCP
   B. IP
   C. UDP
   D. All of the above

4. What is the address size of IPv6?
   A. 32 bit
   B. 64 bit
   C. 128 bit
   D. 256 bit

5. What is the size of Network bits & Host bits of Class A of IP address?
   A. Network bits 7, Host bits 24
   B. Network bits 8, Host bits 24
   C. Network bits 7, Host bits 23
   D. Network bits 8, Host bits 23

6. What does Router do in a network?
   A. Forwards a packet to all outgoing links
   B. Forwards a packet to the next free outgoing link
   C. Determines on which outing link a packet is to be forwarded
   D. Forwards a packet to all outgoing links except the originated link

7. The Internet is an example of
   A. Cell switched network
   B. circuit switched network
   C. Packet switched network
   D. All of above

8. What does protocol defines?
   A. Protocol defines what data is communicated.
   B. Protocol defines how data is communicated.
   C. Protocol defines when data is communicated.
   D. All of above
9. **What is the uses of subnetting?**
   A. It divides one large network into several smaller ones
   B. It divides network into network classes
   C. It speeds up the speed of network
   D. None of above

10. **Repeater operates in which layer of the OSI model?**
    A. Physical layer
    B. Data link layer
    C. Network layer
    D. Transport layer

**Click Here for Answers**

<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>D</td>
</tr>
<tr>
<td>2</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>C</td>
</tr>
<tr>
<td>7</td>
<td>C</td>
</tr>
<tr>
<td>8</td>
<td>D</td>
</tr>
<tr>
<td>9</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
</tr>
</tbody>
</table>

**Fill in the blanks**
1. IP stands for ___________.
2. HTML stands for ___________.
3. ___________ is a web browser.
4. The site which stores web pages is called ___________.
5. The unique address of web page on the web is called ___________.
6. TCP/IP stands for ___________.
7. ___________ was the predecessor to the internet.
8. ASCII stands for ___________.
9. The ___________ is the method used to make hyper text document readable on the WWW.
10. The collection of information for communication is known as ___________.

**ANS KEY**

<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>2</td>
<td>Hyper Text Markup language</td>
</tr>
<tr>
<td>3</td>
<td>Internet explorer</td>
</tr>
<tr>
<td>4</td>
<td>Web sites</td>
</tr>
<tr>
<td>5</td>
<td>URL</td>
</tr>
<tr>
<td>6</td>
<td>Transmission control protocol/Internet protocol</td>
</tr>
<tr>
<td>7</td>
<td>Arpanet</td>
</tr>
<tr>
<td>8</td>
<td>American standard code for information interchange</td>
</tr>
<tr>
<td>9</td>
<td>HTTP</td>
</tr>
</tbody>
</table>
2 mark questions and answers

1. What is function of transport layer?
The protocol in the transport layer takes care in the delivery of data from one application program on one
device to an application program on another device. They act as a link between the upper layer protocols and
the services provided by the lower layer.

2. What are the duties of the transport layer?
The services provided by the transport layer End-to-end delivery
Addressing Reliable delivery
Flow control Multiplexing

3. What is the difference between network layer delivery and the transport layer delivery?
Network layer delivery
* The network layer is responsible for the source-to-destination delivery of packet
* The transport layer is responsible for source-to-destination delivery of the entire message. Transport
layer delivery across multiple network links.
The transport layer is responsible for source-to-destination delivery of the entire message.

4. What are the four aspects related to the reliable delivery of data?
The four aspects are, Error control, Sequence control, Loss control, Duplication control.

5. What is meant by segment?
At the sending and receiving end of the transmission, TCP divides long transmissions into smaller data units
and packages each into a frame called a segment.

3 marks questions and answers:

1. The transport layer creates the connection between source and destination. What are the three events
involved in the connection?
For security, the transport layer may create a connection between the two end ports. A connection is a single
logical path between the source and destination that is associated with all packets in a message. Creating a
connection involves three steps:
Connection establishment
Data transfer & Connection release.
2. **What is meant by congestion?**
Congestion in a network occurs if user sends data into the network at a rate greater than that allowed by network resources.

3. **Why the congestion occurs in network?**
Congestion occurs because the switches in a network have a limited buffer size to store arrived packets.

4. **What is meant by quality of service?**
The quality of service defines a set of attributes related to the performance of the connection. For each connection, the user can request a particular attribute each service class is associated with a set of attributes.

5. **What are the two categories of QoS attributes?**
The two main categories are
User Oriented Network Oriented

5 marks questions and answers

Q1. **Discuss briefly about Connectionless Internetworking?**

A routing decision is made separately for each packet, possibly depending on the traffic at the moment the packet is sent. This strategy can use multiple routes and thus achieve a higher bandwidth than the concatenated virtual-circuit model. On the other hand, there is no guarantee that the packets arrive at the destination in order, assuming that they arrive at all.

The model of Fig. 9 is not quite as simple as it looks. For one thing, if each network has its own network layer protocol, it is not possible for a packet from one network to transit another one. One could imagine the multiprotocol routers actually trying to translate from one format to another, but unless the two formats are close relatives with the same information fields, such conversions will always be incomplete and often doomed to failure. For this reason, conversion is rarely attempted.

A second, and more serious, problem is addressing. Imagine a simple case: a host on the Internet is trying to send an IP packet to a host on an adjoining SNA network. The IP and SNA addresses are different.
One would need a mapping between IP and SNA addresses in both directions. Furthermore, the concept of what is addressable is different. In IP, hosts (actually, interface cards) have addresses. In SNA, entities other than hosts (e.g., hardware devices) can also have addresses. At best, someone would have to maintain a database mapping everything to everything to the extent possible, but it would constantly be a source of trouble.

Another idea is to design a universal "internet" packet and have all routers recognize it. This approach is, in fact, what IP is—a packet designed to be carried through many networks. Of course, it may turn out that IPv4 (the current Internet protocol) drives all other formats out of the market, IPv6 (the future Internet protocol) does not catch on, and nothing new is ever invented, but history suggests otherwise. Getting everybody to agree to a single format is difficult when companies perceive it to their commercial advantage to have a proprietary format that they control.

Tunneling is a method of transmitting data that is intended for use only within a private network through a public network in such a way that the routing nodes in the public network are unaware that the transmission is a part of private network.

Handling the general case of making two different networks interwork is exceedingly difficult. However, there is a common special case that is manageable. This case is where the source and destination hosts are on the same type of network, but there is a different network in between. As an example, think of an international bank with a TCP/IP-based Ethernet in Paris, a TCP/IP-based Ethernet in London, and a non-IP wide area network (e.g., ATM) in between, as shown in Fig.

![Tunneling a packet from Paris to London.](image)

The solution to this problem is a technique called tunneling. To send an IP packet to host 2, host 1 constructs the packet containing the IP address of host 2, inserts it into an Ethernet frame addressed to the Paris multiprotocol router, and puts it on the Ethernet. When the multiprotocol router gets the frame, it removes the IP packet, inserts it in the payload field of the WAN network layer packet, and addresses the latter to the WAN address of the London multiprotocol router. When it gets there, the London router removes the IP packet and sends it to host 2 inside an Ethernet frame.
The WAN can be seen as a big tunnel extending from one multiprotocol router to the other. The IP packet just travels from one end of the tunnel to the other, snug in its nice box. It does not have to worry about dealing with the WAN at all. Neither do the hosts on either Ethernet. Only the multiprotocol router has to understand IP and WAN packets. In effect, the entire distance from the middle of one multiprotocol router to the middle of the other acts like a serial line. An analogy may make tunneling clearer. Consider a person driving her car from Paris to London. Within France, the car moves under its own power, but when it hits the English Channel, it is loaded into a high-speed train and transported to England through the Chunnel.

![Diagram of tunneling a car from France to England.](image)

**Fig: Tunneling a car from France to England.**

**Q2. Explain IP protocol and IP header format?**

IP protocol is a protocol of network layer whose main objective is to support internetworking. An IP datagram consists of a header part and a text part. The header has a 20-byte fixed part and a variable length optional part. The header format is shown in Fig. 13. It is transmitted in big-endian order: from left to right, with the high-order bit of the Version field going first.

The Version field keeps track of which version of the protocol the datagram belongs to. By including the version in each datagram, it becomes possible to have the transition between versions take years, with some machines running the old version and others running the new one.

Since the header length is not constant, a field in the header, IHL, is provided to tell how long the header is, in 32-bit words. The minimum value is 5, which applies when no options are present. The maximum value of this 4-bit field is 15, which limits the header to 60 bytes, and thus the Options field to 40 bytes. For some options, such as one that records the route a packet has taken, 40 bytes is far too small, making that option useless.

The Type of service field is one of the few fields that has changed its meaning (slightly) over the years. It was and is still intended to distinguish between different classes of service. Various combinations of reliability and speed are possible. For digitized voice, fast delivery beats accurate delivery. For file transfer, error-free transmission is more important than fast transmission.
Originally, the 6-bit field contained (from left to right), a three-bit Precedence field and three flags, D, T, and R. The Precedence field was a priority, from 0 (normal) to 7 (network control packet). The three flag bits allowed the host to specify what it cared most about from the set {Delay, Throughput, Reliability}. In theory, these fields allow routers to make choices between, for example, a satellite link with high throughput and high delay or a leased line with low throughput and low delay. In practice, current routers often ignore the Type of service field altogether.

The Total length includes everything in the datagram—both header and data. The maximum length is 65,535 bytes. At present, this upper limit is tolerable, but with future gigabit networks, larger datagrams may be needed.

The Identification field is needed to allow the destination host to determine which datagram a newly arrived fragment belongs to. All the fragments of a datagram contain the same Identification value.

Next comes an unused bit and then two 1-bit fields. DF stands for Don’t Fragment. It is an order to the routers not to fragment the datagram because the destination is incapable of putting the pieces back together again.

MF stands for More Fragments. All fragments except the last one have this bit set. It is needed to know when all fragments of a datagram have arrived.

The Fragment offset tells where in the current datagram this fragment belongs. All fragments except the last one in a datagram must be a multiple of 8 bytes, the elementary fragment unit.

Since 13 bits are provided, there is a maximum of 8192 fragments per datagram, giving a maximum datagram length of 65,536 bytes, one more than the Total length field.

The Time to live field is a counter used to limit packet lifetimes. It is supposed to count time in seconds, allowing a maximum lifetime of 255 sec. It must be decremented on each hop and is supposed to be decremented multiple times when queued for a long time in a router. In practice, it just counts hops. When it hits zero, the packet is discarded and a warning packet is sent back to the source host. This feature prevents
datagrams from wandering around forever, something that otherwise might happen if the routing tables ever become corrupted.

When the network layer has assembled a complete datagram, it needs to know what to do with it. The Protocol field tells it which transport process to give it to. TCP is one possibility, but so are UDP and some others.

The Header checksum verifies the header only. Such a checksum is useful for detecting errors generated by bad memory words inside a router. The algorithm is to add up all the 16-bit halfwords as they arrive, using one's complement arithmetic and then take the one's complement of the result. For purposes of this algorithm, the Header checksum is assumed to be zero upon arrival. This algorithm is more robust than using a normal add.

The Source address and Destination address indicate the network number and host number. The Options field was designed to provide an escape to allow subsequent versions of the protocol to include information not present in the original design, to permit experimenters to try out new ideas, and to avoid allocating header bits to information that is rarely needed.

The options are variable length. Each begins with a 1-byte code identifying the option. Some options are followed by a 1-byte option length field, and then one or more data bytes. The Options field is padded out to a multiple of four bytes.

Q3. What is Fragmentation? Explain different types of Fragmentation.

Each network imposes some maximum size on its packets. These limits have various causes, among them:

7. Hardware (e.g., the size of an Ethernet frame).
8. Operating system (e.g., all buffers are 512 bytes).
9. Protocols (e.g., the number of bits in the packet length field).
10. Compliance with some (inter)national standard.
11. Desire to reduce error-induced retransmissions to some level.
12. Desire to prevent one packet from occupying the channel too long.

The result of all these factors is that the network designers are not free to choose any maximum packet size they wish. Maximum payloads range from 48 bytes (ATM cells) to 65,515 bytes (IP packets), although the payload size in higher layers is often larger.

An obvious problem appears when a large packet wants to travel through a network whose maximum packet size is too small. One solution is to make sure the problem does not occur in the first place. In other words, the internet should use a routing algorithm that avoids sending packets through networks that cannot handle them. However, this solution is no solution at all.
Basically, the only solution to the problem is to allow gateways to break up packets into fragments, sending each fragment as a separate internet packet.

Two opposing strategies exist for recombining the fragments back into the original packet. The first strategy is to make fragmentation caused by a "small-packet" network transparent to any subsequent networks through which the packet must pass on its way to the ultimate destination. This option is shown in Fig. 16(a). In this approach, the small-packet network has gateways (most likely, specialized routers) that interface to other networks. When an oversized packet arrives at a gateway, the gateway breaks it up into fragments. Each fragment is addressed to the same exit gateway, where the pieces are recombined. In this way passage through the small-packet network has been made transparent. Subsequent networks are not even aware that fragmentation has occurred. ATM networks, for example, have special hardware to provide transparent fragmentation of packets into cells and then reassembly of cells into packets. In the ATM world, fragmentation is called segmentation; the concept is the same, but some of the details are different.

Transparent fragmentation is straightforward but has some problems. For one thing, the exit gateway must know when it has received all the pieces, so either a count field or an "end of packet" bit must be provided. For another thing, all packets must exit via the same gateway. By not allowing some fragments to follow one route to the ultimate destination and other fragments a disjoint route, some performance may be lost. ATM requires transparent fragmentation.

Nontransparent fragmentation also has some problems. For example, it requires every host to be able to do reassembly. Yet another problem is that when a large packet is fragmented, the total overhead increases because each fragment must have a header. Whereas in the first method this overhead disappears as soon as the small-packet network is exited, in this method the overhead remains for the rest of the journey. An advantage of nontransparent fragmentation, however, is that multiple exit gateways can now be used and higher performance can be achieved. Of course, if the concatenated virtual-circuit model is being used, this advantage is of no use.

When a packet is fragmented, the fragments must be numbered in such a way that the original data
stream can be reconstructed. One way of numbering the fragments is to use a tree. If packet 0 must be split up, the pieces are called 0.0, 0.1, 0.2, etc. If these fragments themselves must be fragmented later on, the pieces are numbered 0.0.0, 0.0.1, 0.0.2, . . . , 0.1.0, 0.1.1, 0.1.2, etc. If enough fields have been reserved in the header for the worst case and no duplicates are generated anywhere, this scheme is sufficient to ensure that all the pieces can be correctly reassembled at the destination, no matter what order they arrive in.

However, if even one network loses or discards packets, end-to-end retransmissions are needed, with unfortunate effects for the numbering system. Suppose that a 1024-bit packet is initially fragmented into four equal-sized fragments, 0.0, 0.1, 0.2, and 0.3. Fragment 0.1 is lost, but the other parts arrive at the destination. Eventually, the source times out and retransmits the original packet again. Only this time Murphy's law strikes and the route taken passes through a network with a 512-bit limit, so two fragments are generated. When the new fragment 0.1 arrives at the destination, the receiver will think that all four pieces are now accounted for and reconstruct the packet incorrectly.

A completely different (and better) numbering system is for the internetwork protocol to define an elementary fragment size small enough that the elementary fragment can pass through every network. When a packet is fragmented, all the pieces are equal to the elementary fragment size except the last one, which may be shorter. An internet packet may contain several fragments, for efficiency reasons. The internet header must provide the original packet number and the number of the (first) elementary fragment contained in the packet. As usual, there must also be a bit indicating that the last elementary fragment contained within the internet packet is the last one of the original packet.

This approach requires two sequence fields in the internet header: the original packet number and the fragment number. There is clearly a trade-off between the size of the elementary fragment and the number of bits in the fragment number.
Q4. Explain address resolution protocol?

Address Resolution Protocol (ARP):

Address Resolution Protocol (ARP) is a protocol used by Internet Protocol (IP), specifically IPV4, to map IP network address to the hardware addresses used by a datalink protocol. The protocol operates below the network layer as a part of the interface between OSI network and OSI link layer.

The term address resolution refers to the process of finding an address of a computer in a network. The address is resolved using a protocol in which a piece of information is sent by a client process executing on the local computer to a server process executing on a remote computer. The information received by the server allows the server to uniquely identify the network system for which the address was required and therefore it provides the required address. The address resolution procedure is completed when the client receives a response from the server containing the required address. Thus, learning of 48-bit Ethernet address from 32-bit IP address is done with the help of Address Resolution Protocol (ARP).

Consider the following class C networks which are interconnected by using FDDI (Fibre Distributed Data Interface).

When host 1 on CS-Ethernet wants to send data to host 2 of CS-Ethernet, it should know the IP address of host 2, so it broadcasts a message asking for the name of the owner of this IP address. When host 2 receives the broadcast message it replies to host 1 by specifying it’s Ethernet address as E. Thus, the Ethernet address is appended to the transmitting frame of host 1 by the data link layer. The frame is transmitted and is received by host 2.

Thus, with the help of ARP the host can learn the physical address of the destination host when IP address is known.

Working of ARP:

When an incoming packet destined for a host machine on a particular Local Area Network (LAN) which arrives at gateway, the gateway asks the ARP program to find a physical host or MAC address that matches the
IP address. The ARP program looks in the ARP cache and if it finds the IP address, provides it and sends it to the machine. If IP address is not found, ARP broadcasts a request packet in a special format to all machines on the LAN to see if any machine knows this IP address. If a machine recognizes the IP address as its own, it sends a reply. ARP updates the ARP cache for future reference and then sends the packet to the MAC address that replied.

Q5. Define DHCP? Explain in detail?

DHCP: Dynamic Host Configuration Protocol

DHCP is a network protocol that is used to configure devices which are connected to a network so that they can communicate on an IP network. It involves clients and a server operating in a client-server model. In a typical personal home local area network (LAN), a router is the server while clients are personal computers or printers. The router receives this information through a modem from an internet service provider which also operates DHCP servers where the modems are clients. The clients request configuration settings using the DHCP protocol such as an IP address, a default route and one or more DNS server addresses. Once the client implements these settings, the host is able to communicate on that internet.

DHCP was first defined as a standards track protocol in RFC 1531 in October 1993. DHCP is often used together with network address translation (NAT). Network address translation separates public (external) and private (internal) IP addresses. In home networks, the ISP server may assign a globally unique external IP address to a home router or modem and this IP address is used in internet communications. The router will then assign internal IP addresses to the clients connected to it, allowing the clients to broadcast only the external IP address. This improves security by limiting access to devices and also helps to conserve IPv4 addresses.

Technical Overview

Dynamic Host Configuration Protocol automates network-parameter assignment to network devices from one or more DHCP servers. Even in small networks, DHCP is useful because it makes it easy to add new machines to the network. When a DHCP-configured client (a computer or any other network-aware device) connects to a network, the DHCP client sends a broadcast query requesting necessary information to a DHCP server.

The DHCP server manages a pool of IP addresses and information about client configuration parameters such as default gateway, domain name, the name servers, other servers such as time servers, and so forth. On receiving a valid request, the server assigns the computer an IP address, a lease (length of time the allocation is valid), and other IP configuration parameters, such as the subnet mask and the default gateway. The query is typically initiated immediately after booting, and must complete before the client can initiate IP-based communication with other hosts. Upon disconnecting, the IP address is returned to the pool for use by another computer. This way, many other computers can use the same IP address within minutes of each other. Depending on implementation, the DHCP server may have three methods of allocating IP addresses:

1. Dynamic allocation: A network administrator assigns a range of IP addresses to DHCP, and each client computer on the LAN is configured to request an IP address from the DHCP server during network initialization. The request-and-grant process uses a lease concept with a controllable time period, allowing the DHCP server to reclaim (and then reallocate) IP addresses that are not renewed.
2. Automatic allocation: The DHCP server permanently assigns a free IP address to a requesting client from the range defined by the administrator. This is like dynamic allocation, but the DHCP server keeps a table of past IP address assignments, so that it can preferentially assign to a client the same IP address that the client previously had.
3. **Static allocation**: The DHCP server allocates an IP address based on a table with MAC address/IP address pairs, which are manually filled in (perhaps by a network administrator). Only clients with a MAC address listed in this table will be allocated an IP address.

DHCP uses two ports: destination UDP port 67 for sending data to the server, and UDP port 68 for data to the client. DHCP communications are connectionless in nature. DHCP operations fall into four basic phases: **IP discovery**, **IP lease offer**, **IP request**, and **IP lease acknowledgment**. These points are often abbreviated as DORA (Discovery, Offer, Request, and Acknowledgment).

DHCP clients and servers on the same subnet communicate via UDP broadcasts, initially. If the client and server are on different subnets, a DHCP Helper or DHCP Relay Agent may be used. Clients requesting renewal of an existing lease may communicate directly via UDP unicast, since the client already has an established IP address at that point.

1. **DHCP discovery** - The client broadcasts messages on the physical subnet to discover available DHCP servers by creating a User Datagram Protocol (UDP) packet with the broadcast destination of 255.255.255.255 or the specific subnet broadcast address. A DHCP client can also request its last-known IP address.

2. **DHCP offer** - When a DHCP server receives an IP lease request from a client, it reserves an IP address for the client and extends an IP lease offer by sending a DHCPOFFER message to the client. This message contains the client's MAC address, the IP address that the server is offering, the subnet mask, the lease duration, and the IP address of the DHCP server making the offer. The server determines the configuration based on the client's hardware address as specified in the CHADDR (Client Hardware Address) field. Here the server, 192.168.1.1, specifies the client's IP address in the YIADDR (Your IP Address) field.

3. **DHCP request** - In response to the DHCP offer, the client replies with a DHCP request, unicast to the server, requesting the offered address. Based on the Transaction ID field in the request, the server is informed which client has accepted.

4. **DHCP acknowledgment** - When the DHCP server receives the DHCPREQUEST message from the client, the configuration process enters its final phase. The acknowledgment phase involves sending a DHCPACK packet to the client. This packet includes the lease duration and any other configuration information that the client might have requested. At this point, the IP configuration process is completed. The protocol expects the DHCP client to configure its network interface with the negotiated parameters.

**OBJECTIVE TYPE QUESTIONS**

1. **What is the benefit of the Networking?**
   A. File Sharing  
   B. Easier access to Resources  
   C. Easier Backups  
   D. All of the Above

2. **Which of the following is not the Networking Devices?**
   A. Gateways  
   B. Linux  
   C. Routers  
   D. Firewalls

3. **What is the size of MAC Address?**
   A. 16-bits  
   B. 32-bits  
   C. 48-bits  
   D. 64-bits

4. **Which of the following can be Software?**
   A. Routers
B. Firewalls
C. Gateway
D. Modems

5. What is the use of Ping command?
A. To test a device on the network is reachable
B. To test a hard disk fault
C. To test a bug in an Application
D. To test a Printer Quality

6. MAC Address is the example of
A. Transport Layer
B. Data Link Layer
C. Application Layer
D. Physical Layer

7. Routing tables of a router keeps track of
A. MAC Address Assignments
B. Port Assignments to network devices
C. Distribute IP address to network devices
D. Routes to use for forwarding data to its destination

8. Layer-2 Switch is also called
A. Multiport Hub
B. Multiport Switch
C. Multiport Bridge
D. Multiport NIC

9. Difference between T568A and T568B is
A. Difference in wire color
B. Difference in number of wires
C. Just different length of wires
D. Just different manufacturer standards

10. The meaning of Straight-through Cable is
A. Four wire pairs connect to the same pin on each end
B. The cable Which Directly connects Computer to Computer
C. Four wire pairs not twisted with each other
D. The cable which is not twisted

Click Here for Answers

<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>D</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
</tr>
<tr>
<td>5</td>
<td>A</td>
</tr>
<tr>
<td>6</td>
<td>B</td>
</tr>
<tr>
<td>7</td>
<td>D</td>
</tr>
<tr>
<td>8</td>
<td>C</td>
</tr>
<tr>
<td>9</td>
<td>D</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
</tr>
</tbody>
</table>
Fill in the blanks
1. A ___________ is the whole data displayed on the screen at a time.
2. The software which helps to view the websites is called ___________.
3. ___________ is a logical concept.
4. The port number of E mail is ___________.
5. HTTP stands for ___________.
6. FTP stands for ___________.
7. The ___________ program tells you whether or not a person is on particular computer.
8. ___________ is a high level protocol that manages the data.
TCP
9. The computer is identified by ___________.
10. Every computer on the internet has an ___________.

ANS KEY

<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Webpage</td>
</tr>
<tr>
<td>2</td>
<td>Web browser</td>
</tr>
<tr>
<td>3</td>
<td>Port</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
</tr>
<tr>
<td>5</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>6</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>7</td>
<td>Finger</td>
</tr>
<tr>
<td>8</td>
<td>TCP</td>
</tr>
<tr>
<td>9</td>
<td>32 Bit IP address</td>
</tr>
<tr>
<td>10</td>
<td>Address</td>
</tr>
</tbody>
</table>
UNIT - 5
Application Layer

2 mark questions and answers:
1. What is the purpose of Domain Name System?
Domain Name System can map a name to an address and conversely an address to name.

2. Discuss the three main division of the domain name space.
Domain name space is divided into three different sections: generic domains, country domains & inverse domain. Generic domain: Define registered hosts according to their generic behavior, uses generic suffixes.
    Country domain: Uses two characters to identify a country as the last suffix. Inverse domain: Finds the domain name given the IP address.
3. Discuss the TCP connections needed in FTP.
FTP establishes two connections between the hosts. One connection is used for data transfer, the other for control information. The control connection uses very simple rules of communication. The data connection needs more complex rules due to the variety of data types transferred.
4. Discuss the basic model of FTP.
The client has three components: the user interface, the 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.
5. What is the function of SMTP?
The TCP/IP protocol supports electronic mail on the Internet is called Simple Mail Transfer (SMTP). It is a system for sending messages to other computer users based on e-mail addresses. SMTP provides mail exchange between users on the same or different computers.

3 mark questions and answers:
1. What is the purpose of HTML?
HTML is a computer language for specifying the contents and format of a web document. It allows additional text to include codes that define fonts, layouts, embedded graphics and hypertext links.

2. Define CGI.
CGI is a standard for communication between HTTP servers and executable programs. It is used in creating dynamic documents.

Name four factors needed for a secure network. Privacy: The sender and the receiver expect confidentiality.
Authentication: The receiver is sure of the sender’s identity and that an imposter has not sent the message.
Integrity: The data must arrive at the receiver exactly as it was sent. Non-Reputation: The receiver must able to prove that a received message came from a specific sender.
3. **How is a secret key different from public key?**
In secret key, the same key is used by both parties. The sender uses this key and an encryption algorithm to encrypt data; the receiver uses the same key and the corresponding decryption algorithm to decrypt the data. The public key is announced to the public.

4. **What is a digital signature?**
Digital signature is a method to authenticate the sender of a message. It is similar to that of signing transactions documents when you do business with a bank. In network transactions, you can create an equivalent of an electronic or digital signature by the way you send data.

5. **What are the advantages & disadvantages of public key encryption?**
**Advantages:**
1. Remove the restriction of a shared secret key between two entities. Here each entity can create a pair of keys, keep the private one, and publicly distribute the other one.
2. The no. of keys needed is reduced tremendously. For one million users to communicate, only two million keys are needed.

**Disadvantages:**
If you use large numbers the method to be effective. Calculating the cipher text using the long keys takes a lot of time. So it is not recommended for large amounts of text.

16. **What are the advantages & disadvantages of secret key encryption?**
**Advantages:**
Secret Key algorithms are efficient: it takes less time to encrypt a message. The reason is that the key is usually smaller. So it is used to encrypt or decrypt long messages.

**Disadvantages:**
1. Each pair of users must have a secret key. If N people in world want to use this method, there needs to be N (N-1)/2 secret keys. For one million people to communicate, a half-billion secret keys are needed.
2. The distribution of the keys between two parties can be difficult.

17. Define permutation.
Permutation is transposition in bit level.
Straight permutation: The no. of bits in the input and output are preserved. Compressed permutation: The no. of bits is reduced (some of the bits are dropped). Expanded permutation: The no. of bits is increased (some bits are repeated).

5 mark questions and answers

**Q1. Explain the UDP.**
**UDP:**
The Internet protocol suite supports a connectionless transport protocol, UDP (User Datagram Protocol). UDP provides a way for applications to send encapsulated IP data grams and send them without having to establish a connection. UDP is described in RFC 768. UDP transmits segments consisting of an 8-byte header followed by the payload. The header is shown in Fig.7. The two ports serve to identify the end points within the source and destination machines. When a UDP packet arrives, its payload is handed to the process attached to the destination port. This attachment occurs when BIND primitive or something similar is used, for TCP (the
binding process is the same for UDP). In fact, the main value of having UDP over just using raw IP is the addition of the source and destination ports. Without the port fields, the transport layer would not know what to do with the packet. With them, it delivers segments correctly.

![Fig. the UDP header](image)

The source port is primarily needed when a reply must be sent back to the source. By copying the source port field from the incoming segment into the destination port field of the outgoing segment, the process sending the reply can specify which process on the sending machine is to get it.

The UDP length field includes the 8-byte header and the data. The UDP checksum is optional and stored as 0 if not computed (a true computed 0 is stored as all 1s). Turning it off is foolish unless the quality of the data does not matter (e.g., digitized speech). It is probably worth mentioning explicitly some of the things that UDP does not do. It does not do flow control, error control, or retransmission upon receipt of a bad segment. All of that is upto the user processes. What it does do is provide an interface to the IP protocol with the added feature of demultiplexing multiple processes using the ports. That is all it does. For applications that need to have precise control over the packet flow, error control, or timing, UDP provides just what the doctor ordered.

One area where UDP is especially useful is in client-server situations. Often, the client sends a short request to the server and expects a short reply back. If either the request or reply is lost, the client can just time out and try again. Not only is the code simple, but fewer messages are required (one in each direction) than with a protocol requiring an initial setup.
Q2. Describe about the TCP protocol?

TCP (Transmission Control Protocol) was specifically designed to provide a reliable end-to-end byte stream over an unreliable internetwork. An internetwork differs from a single network because different parts may have wildly different topologies, bandwidths, delays, packet sizes, and other parameters.

TCP was designed to dynamically adapt to properties of the internetwork and to be robust in the face of many kinds of failures. TCP was formally defined in RFC 793. As time went on, various errors and inconsistencies were detected, and the requirements were changed in some areas. These clarifications and some bug fixes are detailed in RFC 1122. Extensions are given in RFC 1323.

Each machine supporting TCP has a TCP transport entity, either a library procedure, a user process, or part of the kernel. In all cases, it manages TCP streams and interfaces to the IP layer. A TCP entity accepts user data streams from local processes, breaks them up into pieces not exceeding 64 KB (in practice, often 1460 data bytes in order to fit in a single Ethernet frame with the IP and TCP headers), and sends each piece as a separate IP datagram. When datagrams containing TCP data arrive at a machine, they are given to the TCP entity, which reconstructs the original byte streams. For simplicity, we will sometimes use just "TCP" to mean the TCP transport entity (a piece of software) or the TCP protocol (a set of rules). From the context it will be clear which is meant. For example, in "The user gives TCP the data," the TCP transport entity is clearly intended.

The IP layer gives no guarantee that datagrams will be delivered properly, so it is up to TCP to time out and retransmit them as need be. Datagrams that do arrive may well do so in the wrong order; it is also up to TCP to reassemble them into messages in the proper sequence. In short, TCP must furnish the reliability that most users want and that IP does not provide.

The TCP Segment Header:

Figure shows the layout of a TCP segment. Every segment begins with a fixed-format, 20-byte header. The fixed header may be followed by header options. After the options, if any, up to 65,535 - 20 - 20 = 65,495 data bytes may follow, where the first 20 refer to the IP header and the second to the TCP header. Segments without any data are legal and are commonly used for acknowledgements and control messages.

The Source port and Destination port fields identify the local end points of the connection. The source and destination end points together identify the connection. The sequence number and Acknowledgement
number fields perform their usual functions. Note that the latter specifies the next byte expected, not the last byte correctly received. Both are 32 bits long because every byte of data is numbered in a TCP stream.

The TCP header length tells how many 32-bit words are contained in the TCP header. This information is needed because the Options field is of variable length, so the header is, too.

Technically, this field really indicates the start of the data within the segment, measured in 32-bit words, but that number is just the header length in words, so the effect is the same.

Next comes a 6-bit field that is not used. The fact that this field has survived intact for over a quarter of a century is testimony to how well think out TCP is. Lesser protocols would have needed it to fix bugs in the original design.

Now comes six 1-bit flags. URG is set to 1 if the Urgent pointer is in use. The Urgent pointer is used to indicate a byte offset from the current sequence number at which urgent data are to be found. This facility is in lieu of interrupt messages. As we mentioned above, this facility is a bare-bones way of allowing the sender to signal the receiver without getting TCP itself involved in the reason for the interrupt.

The ACK bit is set to 1 to indicate that the Acknowledgement number is valid. If ACK is 0, the segment does not contain an acknowledgement so the Acknowledgement number field is ignored.

The PSH bit indicates PUSH ed data. The receiver is hereby kindly requested to deliver the data to the application upon arrival and not buffer it until a full buffer has been received (which it might otherwise do for efficiency).

The RST bit is used to reset a connection that has become confused due to a host crash or some other reason. It is also used to reject an invalid segment or refuse an attempt to open a connection. In general, if you get a segment with the RST bit on, you have a problem on your hands.

The SYN bit is used to establish connections. The connection request has SYN = 1 and ACK = 0 to indicate that the piggyback acknowledgement field is not in use. The connection reply does bear an acknowledgement, so it has SYN = 1 and ACK = 1. In essence the SYN bit is used to denote CONNECTION REQUEST and CONNECTION ACCEPTED, with the ACK bit used to distinguish between those two possibilities.

The FIN bit is used to release a connection. It specifies that the sender has no more data to transmit. Both SYN and FIN segments have sequence numbers and are thus guaranteed to be processed in the correct order.

Flow control in TCP is handled using a variable-sized sliding window. The Window size field tells how many bytes may be sent starting at the byte acknowledged. A Window size field of 0 is legal and says that the bytes up to and including Acknowledgement number - 1 have been received, but that the receiver is currently badly in need of a rest and would like no more data for the moment. The receiver can later grant permission to send by transmitting a segment with the same Acknowledgement number and a nonzero Window size field.
Q3. Explain how TCP controls congestion?

TCP Congestion Control:
When the load offered to any network is more than it can handle, congestion builds up. The Internet is no exception. Although the network layer also tries to manage congestion, most of the heavy lifting is done by TCP because the real solution to congestion is to slow down the data rate. In theory, congestion can be dealt with by employing a principle borrowed from physics: the law of conservation of packets. The idea is to refrain from injecting a new packet into the network until an old one leaves (i.e., is delivered). TCP attempts to achieve this goal by dynamically manipulating the window size. The first step in managing congestion is detecting it. In the old days, detecting congestion was difficult. A timeout caused by a lost packet could have been caused by either (1) noise on a transmission line or (2) packet discard at a congested router. Telling the difference was difficult. Nowadays, packet loss due to transmission errors is relatively rare because most long-haul trunks are fiber (although wireless networks are a different story). Consequently, most transmission timeouts on the Internet are due to congestion. All the Internet TCP algorithms assume that timeouts are caused by congestion and monitor timeouts for signs of trouble the way miners watch their canaries. When a connection is established, a suitable window size has to be chosen. The receiver can specify a window based on its buffer size. If the sender sticks to this window size, problems will not occur due to buffer overflow at the receiving end, but they may still occur due to internal congestion within the network.

In Fig. we see this problem illustrated hydraulically. In Fig. 11 (a), we see a thick pipe leading to a small-capacity receiver. As long as the sender does not send more water than the bucket can contain, no water will be lost. In Fig. 11 (b), the limiting factor is not the bucket capacity, but the internal carrying capacity of the network. If too much water comes in too fast, it will back up and some will be lost (in this case by overflowing the funnel). The Internet solution is to realize that two potential problems exist—network capacity and receiver capacity—and to deal with each of them separately. To do so, each sender maintains two windows: the window the receiver has granted and a second window, the congestion window. Each reflects the number of bytes the sender may transmit. The number of bytes that may be sent is the minimum of the two windows. Thus, the effective window is the minimum of what the sender thinks is all right and what the receiver thinks is all right. If the receiver says "Send 8 KB" but the sender knows that bursts of more than 4 KB clog the network, it sends 4 KB. On the other hand, if the receiver says "Send 8 KB" and the sender knows that bursts of up to 32 KB get through effortlessly, it sends the full 8 KB requested.

![Fig (a) A fast network feeding a low-capacity receiver (b) A slow network feeding a high-capacity receiver](image-url)
When a connection is established, the sender initializes the congestion window to the size of the maximum segment in use on the connection. It then sends one maximum segment. If this segment is acknowledged before the timer goes off, it adds one segment's worth of bytes to the congestion window to make it two maximum size segments and sends two segments. As each of these segments is acknowledged, the congestion window is increased by one maximum segment size. When the congestion window is n segments, if all n are acknowledged on time, the congestion window is increased by the byte count corresponding to n segments. In effect, each burst acknowledged doubles the congestion window.

The congestion window keeps growing exponentially until either a timeout occurs or the receiver’s window is reached. The idea is that if bursts of size, say, 1024, 2048, and 4096 bytes work fine but a burst of 8192 bytes gives a timeout, the congestion window should be set to 4096 to avoid congestion. As long as the congestion window remains at 4096, no bursts longer than that will be sent, no matter how much window space the receiver grants. This algorithm is called slow start, but it is not slow at all (Jacobson, 1988). It is exponential. All TCP implementations are required to support it.

Q4. Explain in detail about Electronic Mail?

Electronic Mail:

Electronic mail, or e-mail, as it is known to its many fans, has been around for over two decades. Before 1990, it was mostly used in academia. During the 1990s, it became known to the public at large and grew exponentially to the point where the number of e-mails sent per day now is vastly more than the number of snail mail (i.e., paper) letters.

E-mail, like most other forms of communication, has its own conventions and styles. In particular, it is very informal and has a low threshold of use. People who would never dream of calling up or even writing a letter to a Very Important Person do not hesitate for a second to send a sloppily-written e-mail.

E-mail is full of jargon such as BTW (By The Way), ROTFL (Rolling On The Floor Laughing), anIMHO (In My Humble Opinion). Many people also use little ASCII symbols called smiley’s or emoticons in their e-mail.

The first e-mail systems simply consisted of file transfer protocols, with the convention that the first line of each message (i.e., file) contained the recipient’s address. As time went on, the limitations of this approach became more obvious.

Some of the complaints were as follows:

Sending a message to a group of people was inconvenient. Managers often need this facility to send memos to all their subordinates.

Messages had no internal structure, making computer processing difficult. For example, if a forwarded message was included in the body of another message, extracting the forwarded part from the received message was difficult.

The originator (sender) never knew if a message arrived or not.
If someone was planning to be away on business for several weeks and wanted all incoming e-mail to be handled by his secretary, this was not easy to arrange.

The user interface was poorly integrated with the transmission system requiring users first to edit a file, then leave the editor and invoke the file transfer program.

It was not possible to create and send messages containing a mixture of text, drawings, facsimile, and voice. As experience was gained, more elaborate e-mail systems were proposed. In 1982, the ARPANET e-mail proposals were published as RFC 821 (transmission protocol) and RFC 822 (message format). Minor revisions, RFC 2821 and RFC 2822, have become Internet standards, but everyone still refers to Internet e-mail as RFC 822.

In 1984, CCITT drafted its X.400 recommendation. After two decades of competition, e-mail systems based on RFC 822 are widely used, whereas those based on X.400 have disappeared. How a system hacked together by a handful of computer science graduate students beat an official international standard strongly backed by all the PTTs in the world, many governments, and a substantial part of the computer industry brings to mind the Biblical story of David and Goliath.

The reason for RFC 822's success is not that it is so good, but that X.400 was so poorly designed and so complex that nobody could implement it well. Given a choice between a simple-minded, but working, RFC 822-based e-mail system and a supposedly truly wonderful, but nonworking, X.400 e-mail system, most organizations chose the former.

Q5. What is DNS? Explain usage of resource records?

Domain Name System:

The Domain Name Service (DNS) is a hierarchical distributed method of organizing the name space of the Internet. The DNS administratively groups hosts into hierarchy of authority that allows addressing and other information to be widely distributed and maintained. A key advantage to the DNS is that it eliminates dependence on a centrally maintained file that maps host names to addresses. DNS is supported via an asset of network-resident servers, also called domain name servers.

The IP address is a numeric address that serves role analogous to a telephone number. In representation, addresses always consist of four numbers; four decimal values separated by periods. Figure 17.1 illustrates the addresses. The computer named mugwump.cl.msu.edu for instance, is assigned a number of 35.8.1.212. The reason a computer would have two names is that IP addresses numeric; they can be easily understood and manipulated by the hardware and software that must move information over the Internet. So IP addresses are better-suited to computers, and domain addresses are better-suited to humans. DNS allows a translation between the domain name and the IP address. Domain names do not necessarily have four parts. They might have only two parts—a top-level domain such as “edu” or “com,” preceded by a sub domain or three, four, or many. The limitations are,

(iii) A domain-Mime cannot exceed 255 characters and

(iv) Each part of the name cannot exceed 63 characters.

The DNS translates the plain English address, www.metahouse.com, for example, into numbers that Internet computers can understand, such as 123.23.43.121. In order to do this efficiently, the Internet has been organized into a number of major domains. Major domains refer to the letters at the end of a plain English
address, such as .com. A number of common domains are used in the United States: .com (commercial); .edu (education); .gov (government); .mil (military), .net (Internet service providers and networks-companies and groups concerned with has been growing exponentially, the domain name system is being expanded and may also include at least seven additional domains, such as .web for Web. Only two letters are used outside the United States to identify the domains; for example, .au for Australia; .ca for Canada; .uk for United Kingdom; and for France.

IP address read from general to specific 35.8.1.212
Network address Host address

Domains are organized in a hierarchical manner, so that beneath major domains are many minor domains. As an example of how the DNS and domains work, looks at NASA’s SPACE link Internet address: spacelink.msfc.nasa.gov.

The .top domain is .gov, which stands for government. The domain just below that is .nasa, which is the NASA domain. Then below that, .msfc (Marshall Space Flight Center) is one of NASA’s many computer networks. SPACE link identifies the NASA computer that runs the SPACE link program. SPACE link’s numeric IP address has changed through the years, but its Internet address has stayed the same.

Figure 17.2: A Portion of the Internet Domain Name Spice

The top-level domains come in two flavors: generic and countries. The generic domains are com (commercial), edu (educational institutions), gov (the U.S. federal government), mil (certain international organizations) mil (the U.S. armed forces), net (network providers), and org (non-profit organizations). The country domains include one entry for every country, as defined in ISO 3166.

Each domain is named by the path upward from it to the (unnamed) root. The components are separated by periods (pronounced “dot”). Thus, Sun Microsystems engineering department might be eng.sun.com rather than a UNIX-style name such as /com/sun/eng.

Absolute and Relative Domain Names:
Domain names can be either absolute or relative. An absolute domain name ends with \ a period (e.g., eng.sun.com) whereas a relative one does not. Relative names have to be interpreted in some context to uniquely determine their true meaning. In both cases, a named domain refers to a specific node in the tree and all the nodes under it.
Domain names are case insensitive, so edu and EDU mean the same thing. Component names can be up to 63 characters long, and full path names must not exceed 255 characters.
In principle, domains can be inserted into the tree in two different ways. For example, cs.yale.edu could equally well be listed under the country domain cs.yale.ct.us.

**Resource Records:**

The ‘name servers’ that together implement the DNS distributed database, store resource records (RR) for the hostname to IP address mapping. Each DNS reply message carries one or more resource records.

For a single host, the most common resource record is just its IP address, but many other kinds of resource records also exist. When a resolver gives a domain name to DNS, it gets back the resource records associated with that name. Thus the real function of DNS is to map domain names onto resource records.

A resource record is a five-tuple. Its format is as follows.

<table>
<thead>
<tr>
<th>Domain name</th>
<th>Time_to_live</th>
<th>Type</th>
<th>Class</th>
<th>Value</th>
</tr>
</thead>
</table>

(vi) The Domain_name tells the domain to which this record applies. Normally, many records exist for each domain and each copy of the database holds information about multiple domains. The field is thus the primary search key used to satisfy queries.

(vii) The Time_to_live field gives an indication of the stability of record. Information that is highly stable is assigned a large value. Such as 86400 (the number of seconds in a day). Information that is highly volatile is assigned a small value, such as 60 (1 minute).

(viii) The type field tells the record type, as listed in the table below.

(ix) The fourth field of every resource record is the Class. For Internet information, it is always in. For non Internet information, other codes can be used.

(x) Value field, can be a number, a domain name, or an ASCII string. The semantics depend on the records type.

**OBJECTIVE TYPE QUESTIONS:**

1 Which of the following is not the External Security Threats?
A. Front-door Threats
B. Back-door Threats
C. Underground Threats
D. Denial of Service (DoS)

2 What is the Demilitarized Zone?
A. The area between firewall & connection to an external network
B. The area between ISP to Military area
C. The area surrounded by secured servers
D. The area surrounded by the Military

3 What is the full form of RAID?
A. Redundant Array of Independent Disks
B. Redundant Array of Important Disks
C. Random Access of Independent Disks
D. Random Access of Important Disks

4 What is the maximum header size of an IP packet?
A. 32 bytes
5 What is the size of Host bits in Class B of IP address?
A. 04
B. 08
C. 16
D. 32

6 What is the usable size of Network bits in Class B of IP address?
A. 04
B. 08
C. 14
D. 16

7 In which type of RAID, data is mirrored between two disks.
A. RAID 0
B. RAID 1
C. RAID 2
D. RAID 3

8 What do you mean by broadcasting in Networking?
A. It means addressing a packet to all machine
B. It means addressing a packet to some machine
C. It means addressing a packet to a particular machine
D. It means addressing a packet to except a particular machine

9 Which of the following is/are Protocols of Application?
A. FTP
B. DNS
C. Telnet
D. All of above

10 Which of the following protocol is/are defined in Transport layer?
A. FTP
B. TCP
C. UDP
D. B & C
Fill in the blanks

1. URL stands for _____________.
2. A ____________ is a computer that performs actions for another computer.
3. ____________ is the most common technology for searching and browsing files developed in past five years.
4. A ____________ is the computer that asks for the action.
5. VPN stands for ____________.
6. ____________ is a very popular LAN.
7. SMTP stands for ____________.
8. All computers connected to the internet and wanting to use it for sending/receiving data must follow a common set of rules for communication called ____________.
9. E mail denotes ____________.
10. CSMA/CD stands for ____________.

ANS KEY

<table>
<thead>
<tr>
<th>Q.NO</th>
<th>ANS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>2</td>
<td>Server</td>
</tr>
<tr>
<td>3</td>
<td>Gopher</td>
</tr>
<tr>
<td>4</td>
<td>Client</td>
</tr>
<tr>
<td>5</td>
<td>Virtual private network</td>
</tr>
<tr>
<td>6</td>
<td>Ethernet</td>
</tr>
<tr>
<td>7</td>
<td>Simple mail transfer protocol</td>
</tr>
<tr>
<td>8</td>
<td>Protocol</td>
</tr>
<tr>
<td>9</td>
<td>Electronic mail</td>
</tr>
<tr>
<td>10</td>
<td>Carrier Sense Multiple Access with Collision Detection</td>
</tr>
</tbody>
</table>
17. Beyond syllabus topics with material

1. Application Layer

In the Open Systems Interconnection (OSI) communications model, the application layer provides services for an application program to ensure that effective communication with another application program in a network is possible.

- The application layer is not the application itself that is doing the communication. It is a service layer that provides these services:
- The application layer is the seventh layer of the OSI model and the only one that directly interacts with the end user.

The application layer provides many services, including:

- Simple Mail Transfer Protocol
- File transfer
- Web surfing
- Web chat
- Email clients
- Network data sharing
- Virtual terminals
- Various file and data operations

- When an application layer protocol wants to communicate with its peer application layer protocol on remote host, it hands over the data or information to the Transport layer.
- The transport layer does the rest with the help of all the layers below it.
- The application layer provides full end-user access to a variety of shared network services for efficient OSI model data flow. This layer has many responsibilities, including error handling and recovery, data flow over a network and full network flow. It is also used to develop network-based applications.
- More than 15 protocols are used in the application layer, including File Transfer Protocol, Telnet, Trivial File Transfer Protocol and Simple Network Management Protocol.

**Application layer paradigms**

A system can act as Server and Client simultaneously. That is, one process is acting as Server and another is acting as a client. This may also happen that both client and server processes reside on the same machine.

**Communication**

Two processes in client-server model can interact in various ways:
- Sockets
- Remote Procedure Calls (RPC)
Sockets
In this paradigm, the process acting as Server opens a socket using a well-known (or known by client) port and waits until some client request comes. The second process acting as a Client also opens a socket but instead of waiting for an incoming request, the client processes ‘requests first’.

When the request is reached to server, it is served. It can either be an information sharing or resource request.

CLIENT-SERVER PARADIGM
- The application layer programs are based on the concept of clients and servers. The purpose of a network, and in particular, the global Internet, is to provide a service to a user.
- A user at a local site wants to receive a service from a computer at a remote site.
- For example, perhaps a user wants to retrieve a file from a remote computer. How can this be done? Both computers must run programs.
- The local computer runs a program that requests a service from another program on the remote computer.
- This means that both computers must run a program, one to request a service and one to provide a service.
- At first glance, communication between two application programs—one running at the local site, the other running at the remote site—seems like a simple task.
- But many questions arise upon the actual implementation of the approach. Some questions that we may ask are:
  - Should the application program be able to request as well as provide services or should the application program just do one or the other?
  - One solution is to have an application program, called the client, running on the local machine, request a service from another application program, called the server, running on the remote machine.
  - In other words, the tasks of requesting a service and providing a service are separate from each other.
  - An application program is either a requester (a client) or a provider (a server). They come in pairs, client and server, both having the same name
  - Should an application program provide service for only one specific program installed elsewhere or should it provide service for any application program that requests this service?
  - The most common solution is a server providing a service for any client, not just one particular client. In other words, the client-server relationship is many-to-one.
  - Many clients can use the services of one server.
- When should an application program be running? All of the time or just when there is a need for the service? Generally, a client program, which requests a service, runs only when it is needed.
The server program, which provides a service, runs all of the time because it does not know when its service will be needed.

Should there be just one universal application program that can provide any type of service a user wants? Or should there be one application program for each type of service? Generally, a service needed frequently and by many users has a specific client-server application program.

For example, one client-server application Fig Client-server model Figure Client-server relationship Client Server Internet Client Server Client for97020_ch02.fm Page 33 Wednesday, July 10, 2002 11:45 AM 34 CHAPTER TWO program allows users to access files; a different client-server application program allows users to send email, and so on.

For services that are more customized, we have one generic application program that allows users to access the services available on a remote computer.

CLIENT A client is a program running on the local machine requesting service from a server.

A client program is started by the user when the service is needed and terminates when the service is complete. A client opens the communication channel, sends its request, receives the response, and closes the channel.

Multiple requests and responses are allowed in one session. SERVER A server is a program running on the remote machine providing service to the clients. When it starts, it opens the door for incoming requests from clients, but it never initiates a service until it is requested to do so.

A server program normally runs infinitely unless a problem arises. It waits for incoming requests from clients.

P2P is extremely interesting from a technical point of view. Its completely decentralized model enables the development of applications with

- high-availability
- fault-tolerance
- scalability characteristics previously unseen in Internet
- It exploits what has been defined the “dark matter” of Internet Levels of P2P-ness
- 12 P2P as a mindset
- Slashdot P2P as a model
- Gnutella P2P as an implementation choice
- Application-layer multicast P2P as an inherent property
- Ad-hoc networks P2P Services 13 Areas of applicability of P2P
- sharing of content
- file sharing, content delivery network
- Gnutella, Kazaa, Akamai
- Care Science Care Data Exchange – exchange of information between healthcare organizations:
  - clinical results
  - patient demographics
  - medical records – Aimed at giving medical personnel easy access to crucial health-related data – In the United States, 50,000 deaths per year are caused by the lack of information
- sharing of storage
- distributed file system, distributed search engine
- Ocean Store.

Simple Network Management Protocol

**Simple Network Management Protocol (SNMP)** is an "Internet-standard protocol for managing devices on IP networks. Devices that typically support SNMP include routers, switches, servers, workstations, printers,
modem racks, and more. It is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention.

- The Simple Network Management Protocol (SNMP) is a framework for managing devices in an Internet using the TCP/IP protocol suite.
- It provides a set of fundamental operations for monitoring and maintaining an Internet.
- Concept SNMP uses the concept of manager and agent. That is, a manager, usually a host, controls and monitors a set of agents, usually routers.

SNMP is an application-level protocol in which a few manager stations control a set of agents.

- The protocol is designed at the application level so that it can monitor devices made by different manufacturers and installed on different physical networks.
- Managers and Agents A management station, called a manager, is a host that runs the SNMP client program. .
- Management is achieved through simple interaction between a manager and an agent. The agent keeps performance information in a database. The manager has access to the values in the database.
- For example, a router can store in appropriate variables the number of packets received and forwarded. The manager can fetch and compare the values of these two variables to see if the router is congested or not. An SNMP-managed network consists of three key components:
  - Managed device
  - Agent — software which runs on managed devices

Network management system (NMS) —

- A managed device is a network node that implements an SNMP interface that allows unidirectional (read-only) or bidirectional access to node-specific information.
- Managed devices exchange node-specific information with the NMSs. Sometimes called network elements, the managed devices can be any type of device, including, but not limited to, routers, access servers, switches, bridges, hubs, IP telephones, IP video cameras, computer hosts, and printers.
- An agent is a network-management software module that resides on a managed device. An agent has local knowledge of management information and translates that information to or from an SNMP specific form.
- A network management system (NMS) executes applications that monitor and control managed devices. NMSs provide the bulk of the processing and memory resources required for network management. One or more NMSs may exist on any managed network.
- Management with SNMP is based on three basic ideas: 1. A manager checks an agent by requesting information that reflects the behavior of the agent. 2.

SNMP operates in the Application Layer of the Internet Protocol Suite (Layer 7 of the OSI model). The SNMP agent receives requests on UDP port 161. The manager may send requests from any available source port to port 161 in the agent. The agent response will be sent back to the source port on the manager. The manager receives notifications (Traps and Inform Requests) on port 162. The agent may generate notifications from any available port. To do management tasks,

SNMP uses two other protocols:

- Structure of Management Information (SMI)
- Management Information Base (MIB). Role of SNMP SNMP has some very specific roles in network management.
- It defines the format of the packet to be sent from a manager to an agent and vice versa. It also interprets the result and creates statistics (often with the help of other management software).
The packets exchanged contain the object (variable) names and their status (values). SNMP is responsible for reading and changing these values.

2. Ethernet - Ethernet cabling, Manchester Encoding

**ETHERNET**
- The most important of the survivors are 802.3 (Ethernet) and 802.11 (wireless LAN). With 802.15 (Bluetooth) and 802.16 (wireless MAN), it is too early to tell.
- Both 802.3 and 802.11 have different physical layers and different MAC sub layers but converge on the same logical link control sub layer (defined in 802.2), so they have the same interface to the network layer.
- Ethernet and IEEE 802.3 are identical except for two minor differences that we will discuss shortly, many people use the terms "Ethernet" and "IEEE 802.3" interchangeably, and we will do so, too.

**Ethernet Cabling**

<table>
<thead>
<tr>
<th>Name</th>
<th>Cable</th>
<th>Max. seg.</th>
<th>Nodes/seg.</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>10Base5</td>
<td>Thick coax</td>
<td>500 m</td>
<td>100</td>
<td>Original cable; now obsolete</td>
</tr>
<tr>
<td>10Base2</td>
<td>Thin ccax</td>
<td>185 m</td>
<td>30</td>
<td>No hub needed</td>
</tr>
<tr>
<td>10Base-T</td>
<td>Twisted pair</td>
<td>100 m</td>
<td>1024</td>
<td>Cheapest system</td>
</tr>
<tr>
<td>10Base-T</td>
<td>Fiber optics</td>
<td>2000 m</td>
<td>1024</td>
<td>Best between buildings</td>
</tr>
</tbody>
</table>

- The 10Base5 cabling, popularly called thick Ethernet, came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go.
- Timing the interval between sending the pulse and receiving the echo, it is possible to localize the origin of the echo. This technique is called time domain reflectometry.
- Usually, these wires are telephone company twisted pairs, since most office buildings are already wired this way, and normally plenty of spare pairs are available. This scheme is called 10Base-T.
- With 10Base5, a transceiver cable or drop cable connects the transceiver to an interface board in the computer.
- 10Base2, the connection to the cable is just a passive BNC T-junction connector.
- 10Base-T, there is no shared cable at all, just the hub (a box full of electronics) to which each station is connected by a dedicated (i.e., not shared) cable.
- The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100 meters; maybe 200 meters if very high quality category 5 twisted pairs are used.
- 10Base-T quickly became dominant due to its use of existing wiring and the ease of maintenance that it offers.
- 10Base-F, which uses fiber optics, is expensive due to the cost of the connectors and terminators.

**Figure: Three kinds of Ethernet cabling. (a) 10Base5. (b) 10Base2. (c) 10Base-T.**

With 10Base5, a transceiver cable or drop cable connects the transceiver to an interface board in the computer.

10Base2, the connection to the cable is just a passive BNC T-junction connector.

10Base-T, there is no shared cable at all, just the hub (a box full of electronics) to which each station is connected by a dedicated (i.e., not shared) cable.

The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100 meters; maybe 200 meters if very high quality category 5 twisted pairs are used.

10Base-T quickly became dominant due to its use of existing wiring and the ease of maintenance that it offers.

10Base-F, which uses fiber optics, is expensive due to the cost of the connectors and terminators.

**Figure: Cable topologies. (a) Linear. (b) Spine. (c) Tree. (d) Segmented.**
Manchester Encoding

- None of the versions of Ethernet uses straight binary encoding with 0 volts for a 0 bit and 5 volts for a 1 bit because it leads to ambiguities.
- If one station sends the bit string 0001000, others might falsely interpret it as 1000000 or 0100000 because they cannot tell the difference between an idle sender (0 volts) and a 0 bit (0 volts).
- Two such approaches are called **Manchester encoding** and **differential Manchester encoding**.

**Figure:** (a) Binary encoding. (b) Manchester encoding. (c) Differential Manchester encoding.

Differential Manchester encoding, shown in above Fig (c), is a variation of basic Manchester encoding.

The Ethernet MAC Sub layer Protocol

- The Manchester encoding of this pattern produces a 10-MHz square wave for 6.4 μsec to allow the receiver's clock to synchronize with the sender's.

**Figure:** Frame formats. (a) DIX Ethernet. (b) IEEE 802.3.

- Sending to a group of stations is called **multicast**. The address consisting of all 1 bits is reserved for **broadcast**.

**Figure:** Collision detection can take as long as 2τ.

The Binary Exponential Back off Algorithm

- After the second collision, each one picks 0, 1, 2, or 3 at random and waits that number of slot times. If a third collision occurs, then the next time the number of slots to wait is chosen at random from the interval 0 to 2^3 - 1.
- This algorithm, called **binary exponential back off**, was chosen to dynamically adapt to the number of stations trying to send.
Ethernet Performance

- It is probably worth mentioning that there has been a large amount of theoretical performance analysis of Ethernet (and other networks). Virtually all of this work has assumed that traffic is Poisson.

Switched Ethernet

- One way out is to go to a higher speed, say, from 10 Mbps to 100 Mbps. But with the growth of multimedia, even a 100-Mbps or 1-Gbps Ethernet can become saturated.

**Figure: A simple example of switched Ethernet.**

![Switched Ethernet Diagram]

- Collisions on this on-card LAN will be detected and handled the same as any other collisions on a CSMA/CD network—with retransmissions using the binary exponential back off algorithm.

Fast Ethernet

- To pump up the speed, various industry groups proposed two new ring-based optical LANs.
- One was called **FDDI (Fiber Distributed Data Interface)** and the other was called **Fibre Channel**.

Gigabit Ethernet

- All configurations of gigabit Ethernet are point-to-point rather than multidrug as in the original 10 Mbps standard, now honored as **classic Ethernet**.
- In the simplest gigabit Ethernet configuration, illustrated in Fig (a), two computers are directly connected to each other.

**Figure: (a) A two-station Ethernet. (b) A multistation Ethernet.**

![Gigabit Ethernet Diagrams]

- Gigabit Ethernet supports two different modes of operation: full-duplex mode and half-duplex mode.
- The second feature, called **frame bursting**, allows a sender to transmit a concatenated sequence of multiple frames in a single transmission.

**Figure: Gigabit Ethernet cabling.**
IEEE 802.2: Logical Link Control

- IEEE has defined one that can run on top of Ethernet and the other 802 protocols. In addition, this protocol, called **LLC (Logical Link Control)**, hides the differences between the various kinds of 802 networks by providing a single format and interface to the network layer.

- LLC forms the upper half of the data link layer, with the MAC sublayer below it, as shown in the below Fig.

![Figure: (a) Position of LLC. (b) Protocol formats.](image)

The 802.11 Protocol Stack

- A partial view of the 802.11 protocol stack is given in the below figure. The physical layer corresponds to the OSI physical layer fairly well, but the data link layer in all the 802 protocols is split into two or more sub layers.

- In 802.11, the MAC (Medium Access Control) sub layer determines how the channel is allocated, that is, who gets to transmit next.

![Figure: Part of the 802.11 protocol stack.](image)

- Cordless telephones and microwave ovens also use this band. All of these techniques operate at 1 or 2 Mbps and at low enough power that they do not conflict too much.

The 802.11 Physical Layer

<table>
<thead>
<tr>
<th>Name</th>
<th>Cable</th>
<th>Max. segment</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000Base-SX</td>
<td>Fiber optics</td>
<td>550 m</td>
<td>Multimode fiber (50, 62.5 microns)</td>
</tr>
<tr>
<td>1000Base-LX</td>
<td>Fiber optics</td>
<td>5000 m</td>
<td>Single (10 μ) or multimode (50, 62.5 μ)</td>
</tr>
<tr>
<td>1000Base-CX</td>
<td>2 Pairs of STP</td>
<td>25 m</td>
<td>Shielded twisted pair</td>
</tr>
<tr>
<td>1000Base-T</td>
<td>4 Pairs of UTP</td>
<td>100 m</td>
<td>Standard category 5 UTP</td>
</tr>
</tbody>
</table>
At 1 Mbps, an encoding scheme is used in which a group of 4 bits is encoded as a 16-bit codeword containing fifteen 0s and a single 1, using what is called Gray code. At 2 Mbps, the encoding takes 2 bits and produces a 4-bit codeword, also with only a single 1, that is one of 0001, 0010, 0100, or 1000.

The 802.11 MAC Sub layer Protocol

- The 802.11 MAC sub layer protocol is quite different from that of Ethernet due to the inherent complexity of the wireless environment compared to that of a wired system.
- To start with, there is the hidden station problem mentioned earlier and illustrated again in Fig (a).

![Figure: (a) The hidden station problem. (b) The exposed station problem.](image)

- To deal with this problem, 802.11 support two modes of operation. The first, called DCF (Distributed Coordination Function), does not use any kind of central control (in that respect, similar to Ethernet).
- The other, called PCF (Point Coordination Function), uses the base station to control all activity in its cell. All implementations must support DCF but PCF is optional.
- When DCF is employed, 802.11 uses a protocol called CSMA/CA (CSMA with Collision Avoidance).

![Figure: A fragment burst.](image)

- All of the above discussion applies to the 802.11 DCF modes. In this mode, there is no central control, and stations compete for air time, just as they do with Ethernet.
- The basic mechanism is for the base station to broadcast a beacon frame periodically (10 to 100 times per second).

The 802.11 Frame Structure

- The 802.11 standard defines three different classes of frames on the wire: data, control, and management. The format of the data frame is shown in the following figure.
- First comes the Frame Control field. It itself has 11 subfields. The first of these is the Protocol version, which allows two versions of the protocol to operate at the same time in the same cell.
- Then come the Type (data, control, or management) and Subtype fields (e.g., RTS or CTS). The two DS and From DS bits indicate the frame is going to or coming from the intercell distribution system (e.g., Ethernet).
- The MF bit means that more fragments will follow. The Retry bit marks a retransmission of a frame sent earlier.
- The Power management bit is used by the base station to put the receiver into sleep state or take it out.
of sleep state.
- The More bit indicates that the sender has additional frames for the receiver. The W bit specifies that the frame body has been encrypted using the **WEP (Wired Equivalent Privacy)** algorithm.

![Figure: The 802.11 data frame.](image)

3. Logical Addressing:

**ARP and RARP**

**ARP: The Address Resolution Protocol**
- Most hosts at companies and universities are attached to a LAN by an interface board that only understands LAN addresses.
- For example, every Ethernet board ever manufactured comes equipped with a 48-bit Ethernet address.
- The boards send and receive frames based on 48-bit Ethernet addresses. They know nothing at all about 32-bit IP addresses.

![Figure: Three interconnected /24 networks: two Ethers and an FDDI ring.](image)

- The protocol used for asking this question and getting the reply is called **ARP (Address Resolution Protocol)**. Almost every machine on the Internet runs it. ARP is defined in RFC 826.
- The advantage of using ARP over configuration files is the simplicity. The system manager does not have to do much except assign each machine an IP address and decide about subnet masks. ARP does the rest.
- This ARP broadcast can be avoided by having host 1 includes its IP-to-Ethernet mapping in the ARP packet.
When the ARP broadcast arrives at host 2, the pair (192.31.65.7, E1) is entered into host 2’s ARP cache for future use.

If a response does (unexpectedly) arrive, two machines have been assigned the same IP address. The new one should inform the system manager and not boot.

Using ARP will fail because host 4 will not see the broadcast (routers do not forward Ethernet-level broadcasts).

There are two solutions. First, the CS router could be configured to respond to ARP requests for network 192.31.63.0 (and possibly other local networks).

In this case, host 1 will make an ARP cache entry of (192.31.63.8, E3) and happily send all traffic for host 4 to the local router. This solution is called **proxy ARP**.

RARP: **Reverse Address Resolution Protocol**

The first solution devised was to use **RARP** (Reverse Address Resolution Protocol) (defined in RFC 903).

This protocol allows a newly-booted workstation to broadcast its Ethernet address and say:

The RARP server sees this request, looks up the Ethernet address in its configuration files, and sends back the corresponding IP address.

Using RARP is better than embedding an IP address in the memory image because it allows the same image to be used on all machines.

To get around this problem, an alternative bootstrap protocol called **BOOTP** was invented. Unlike RARP, BOOTP uses UDP messages, which are forwarded over routers.

It also provides a diskless workstation with additional information, including the IP address of the file server holding the memory image, the IP address of the default router, and the subnet mask to use. BOOTP is described in RFCs 951, 1048, and 1084.