<u>UNIT 3</u> - <u>NETWORK LAYER</u>

- The Network Layer is the third layer of the OSI model.
- It handles the service requests from the transport layer and further forwards the service request to the data link layer.
- The network layer translates the logical addresses into physical addresses
- It determines the route from the source to the destination and also manages the traffic problems such as switching, routing and controls the congestion of data packets.
- The main role of the network layer is to move the packets from sending host to the receiving host.

The main functions performed by the network layer are:

- **Routing:** When a packet reaches the router's input link, the router will move the packets to the router's output link. For example, a packet from S1 to R1 must be forwarded to the next router on the path to S2.
- **Logical Addressing:** The data link layer implements the physical addressing and network layer implements the logical addressing. Logical addressing is also used to distinguish between source and destination system. The network layer adds a header to the packet which includes the logical addresses of both the sender and the receiver.
- **Internetworking:** This is the main role of the network layer that it provides the logical connection between different types of networks.
- **Fragmentation:** The fragmentation is a process of breaking the packets into the smallest individual data units that travel through different networks.

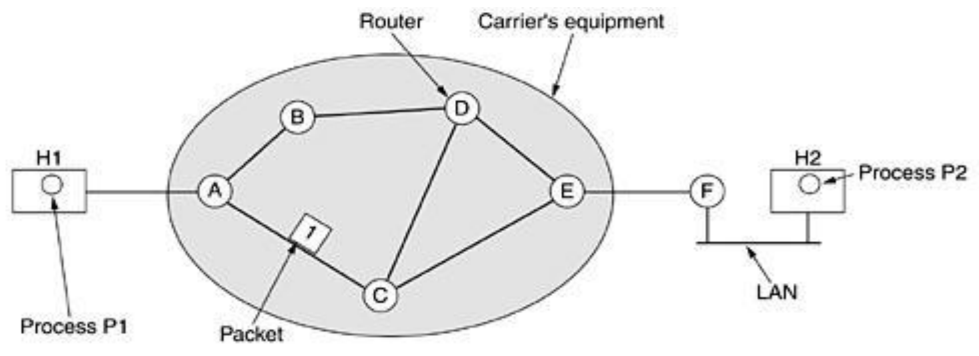
Network Layer Design Issues

- **1. Store-and-Forward Packet Switching**

  · The major components of the system are the carrier's equipment (routers connected by transmission lines), shown inside the shaded oval, and the customers' equipment, shown outside the oval.

  · Host H1 is directly connected to one of the carrier's routers, A, by a leased line. In contrast, H2 is on a LAN with a router, F, owned and operated by the customer. This router also has a leased line to the carrier's equipment.

· We have shown F as being outside the oval because it does not belong to the carrier, but in terms of construction, software, and protocols, it is probably no different from the carrier's routers.



○

· This equipment is used as follows. A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the carrier. The packet is stored there until it has fully arrived so the checksum can be verified.

· Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered. This mechanism is store-and-forward packet switching.

2. Services provided to the Transport Layer

· The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is what kind of services the network layer provides to the transport layer.

The network layer services have been designed with the following goals in mind.

1. The services should be independent of the router technology.

2. The transport layer should be shielded from the number, type, and topology of the routers present.

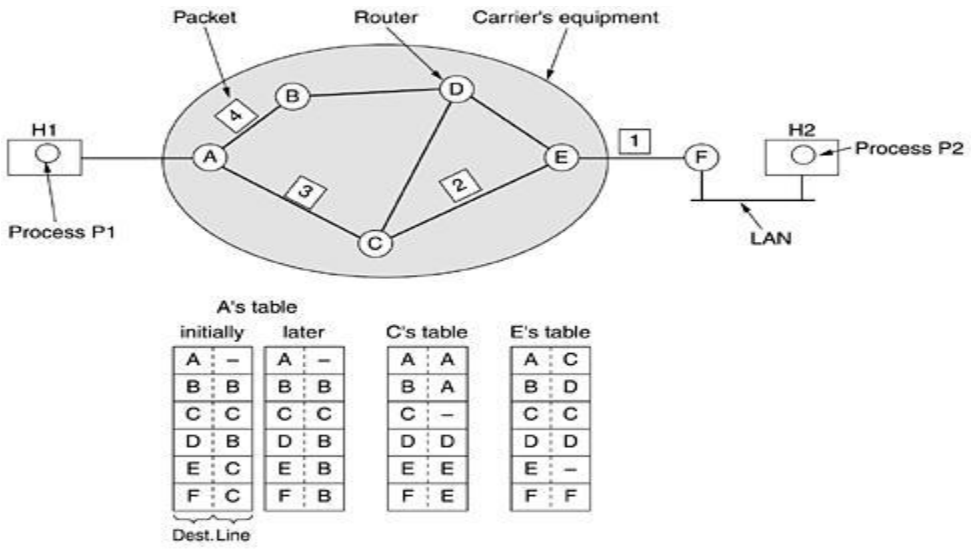
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

3. Implementation of Connectionless Service

If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed.

In this context, the packets are frequently called datagrams (in analogy with telegrams) and the subnet is called a datagram subnet. If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent.

This connection is called a VC (virtual circuit), in analogy with the physical circuits set up by the telephone system, and the subnet is called a virtual-circuit subnet. Let us now see how a datagram subnet works. Suppose that the process P1 in Fig. 3-2 has a long message for P2. It hands the message to the transport layer with instructions to deliver it to process P2 on host H2.
.



A's table initially		A's table later		C's table		E's table	
A	–	A	–	A	A	A	C
B	B	B	B	B	A	B	D
C	C	C	C	C	–	C	C
D	B	D	B	D	D	D	D
E	C	E	B	E	E	E	–
F	C	F	B	F	E	F	F

Let us assume that the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4 and sends each of them in turn to router A using some point-to-point protocol, for example, PPP.

At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination.

A has only two outgoing lines—to B and C—so every incoming packet must be sent to one of these routers, even if the ultimate destination is some other router. A's initial routing table is shown in the figure under the label "initially."

However, something different happened to packet 4. When it got to A it was sent to router B, even though it is also destined for F. For some reason, A decided to send packet 4 via a different route than that of the first three.

Perhaps it learned of a traffic jam somewhere along the ACE path and updated its routing table, as shown under the label "later." The algorithm that manages the tables and makes the routing decisions is called the routing algorithm.

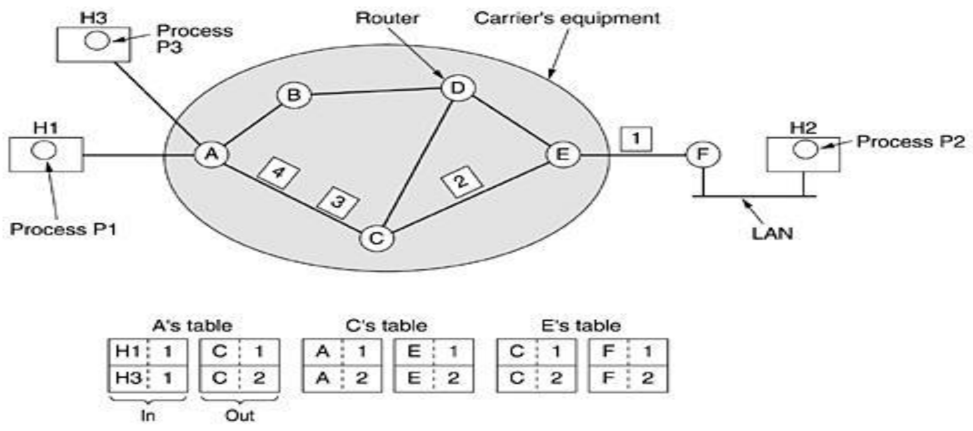
4. Implementation of Connection-Oriented Service

For connection-oriented service, we need a virtual-circuit subnet. The idea behind virtual circuits is to avoid having to choose a new route for every packet sent, as in Fig. 3-2.

Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers. That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.

When the connection is released, the virtual circuit is also terminated. With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to. As an example, consider the situation of Fig. 3-3. Here, host H1 has established connection 1 with host H2.

It is remembered as the first entry in each of the routing tables. The first line of A's table says that if a packet bearing connection identifier 1 comes in from H1, it is to be sent to router C and given connection identifier 1. Similarly, the first entry at C routes the packet to E, also with connection identifier 1.



A's table In		A's table Out	
H1	1	C	1
H3	1	C	2

C's table In		C's table Out	
A	1	E	1
A	2	E	2

E's table In		E's table Out	
C	1	F	1
C	2	F	2

○

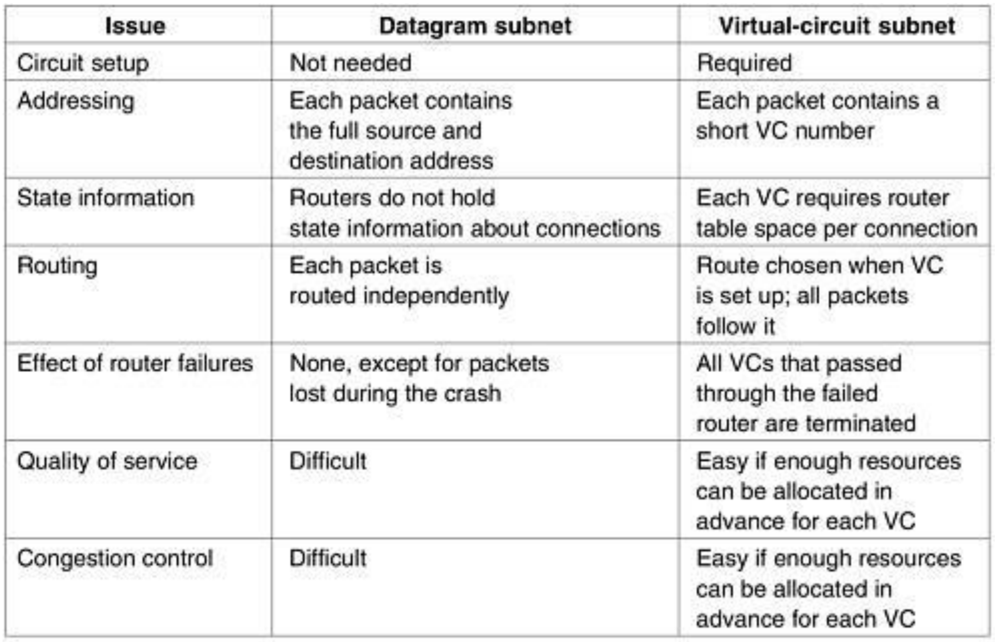
Now let us consider what happens if H3 also wants to establish a connection to H2. It chooses connection identifier 1 and tells the subnet to establish the virtual circuit. This leads to the second row in the tables.

Note that we have a conflict here because although A can easily distinguish connection 1 packets from H1 from connection 1 packets from H3, C cannot do this. For this reason, A assigns a different connection identifier to the outgoing traffic for the second connection.

Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets. In some contexts, this is called label switching.

○

5. Comparison of Virtual-Circuit and Datagram Subnets
Both virtual circuits and datagrams have their supporters and their detractors. We will now attempt to summarize the arguments both ways. The major issues are listed in Fig. 3-4, although purists could probably find a counterexample for everything in the figure.



Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

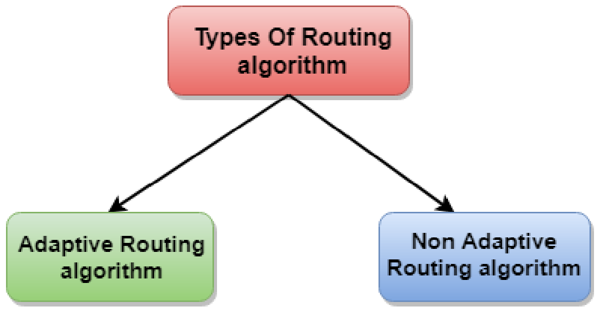
Routing algorithm

- In order to transfer the packets from source to the destination, the network layer must determine the best route through which packets can be transmitted.
- Whether the network layer provides datagram service or virtual circuit service, the main job of the network layer is to provide the best route. The routing protocol provides this job.
- The routing protocol is a routing algorithm that provides the best path from the source to the destination. The best path is the path that has the "least-cost path" from source to the destination.
- Routing is the process of forwarding the packets from source to the destination but the best route to send the packets is determined by the routing algorithm.

Classification of a Routing algorithm

The Routing algorithm is divided into two categories:

- Adaptive Routing algorithm
- Non-adaptive Routing algorithm


Types Of Routing algorithm
Adaptive Routing algorithm
Non Adaptive Routing algorithm

Adaptive Routing algorithm

- An adaptive routing algorithm is also known as dynamic routing algorithm.
- This algorithm makes the routing decisions based on the topology and network traffic.
- The main parameters related to this algorithm are hop count, distance and estimated transit time.

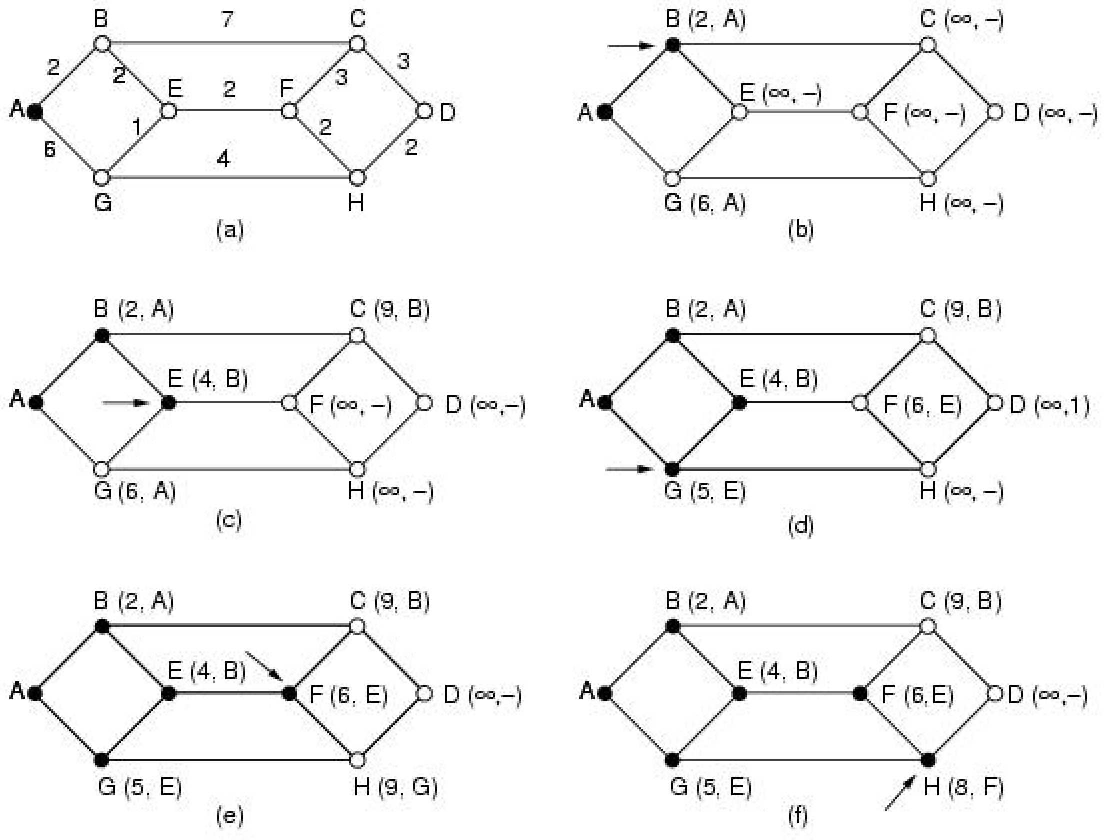
Non-Adaptive Routing algorithm

- Non Adaptive routing algorithm is also known as a static routing algorithm.
- When booting up the network, the routing information stores to the routers.
- Non Adaptive routing algorithms do not take the routing decision based on the network topology or network traffic.

Shortest Path Routing

In computer networks, the shortest path algorithms aim to find the optimal paths between the network nodes so that routing cost is minimized. They are direct applications of the shortest path algorithms proposed in graph theory.

Consider that a network comprises of N vertices (nodes or network devices) that are connected by M edges (transmission lines). Each edge is associated with a weight, representing the physical distance or the transmission delay of the transmission line. The target of shortest path algorithms is to find a route between any pair of vertices along the edges, so the sum of weights of edges is minimum. If the edges are of equal weights, the shortest path algorithm aims to find a route having minimum number of hops.



- Initially, no path is known. So all the nodes are labeled as at an infinite distance from source node.

- As the algorithm proceeds, the labels of the nodes changes accordingly reflecting a better path from the given source to the given sink.

- Start from a node, and examine all adjacent node(s) to it. If the sum of labels of nodes and distance from working node to the node being examined is less than the label on that node, then we have a shortest path, and the node is re-labeled. In a similar fashion, all the adjacent nodes to the working node are inspected and the tentative labels are changed.

- If possible the entire graph is searched for tentatively labeled nodes with the smallest value, the node is made the permanent node. With the progress of the algorithm, all permanent nodes are encircled, so the shortest path could be reconstructed.

Flooding in Computer Networks

In computer networks, flooding is an easy and straightforward routing technique in which the source or node sends packets over each of the outgoing links. Flooding is a very simple routing algorithm that sends all the packets arriving via each outgoing link. Flooding is used in computer networking routing algorithms where each incoming packet is transmitted through every outgoing link, except for the one on which it arrived. Flooding algorithms are guaranteed to find and exploit the shortest paths to the sent packets, as floods use each route in a network naturally.

The Concept of Flooding in Computer Networks

Data packets do not contain network routing information at first. To monitor network topology, or traverse network routes, a hop count algorithm is used. A packet attempts to access all possible network pathways before arriving at its destination; however, packet replication is always a possibility. To avoid communication delay and duplication, a hop count and various selective flooding methods are employed.

Types of Network Flooding

Controlled flooding, uncontrolled flooding, and selective flooding are the three popular types of network flooding.

- **Controlled Flooding:** They employ a number of techniques to manage packet transport to neighbouring nodes. Two algorithms are employed in controlled flooding to ensure that the flooding is confined, and they are Sequence Number Controlled Flooding and Reverse Path Forwarding.
- **Uncontrolled Flooding:** Each router transmits all incoming data packets to all of its neighbours indiscriminately.
- **Selective Flooding:** Instead of transmitting incoming packets down all possible paths, the routers only transmit them along those paths that are headed roughly in the appropriate direction.

<u>Distance Vector Routing Algorithm</u>

- **The Distance vector algorithm is iterative, asynchronous and distributed.**
    - **Distributed:** It is distributed in that each node receives information from one or more of its directly attached neighbors, performs calculation and then distributes the result back to its neighbors.
    - **Iterative:** It is iterative in that its process continues until no more information is available to be exchanged between neighbors.
    - **Asynchronous:** It does not require that all of its nodes operate in the lock step with each other.
- The Distance vector algorithm is a dynamic algorithm.
- It is mainly used in ARPANET, and RIP.
- Each router maintains a distance table known as **Vector**.

The **Distance Vector routing algorithm**(DVR) shares the information of the routing table with the other routers in the network and keeps the information up-to-date to select an optimal path from source to destination.

How DVR Protocol Works ?

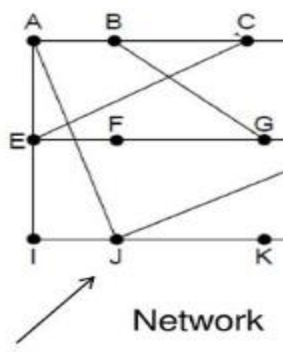
The distance vector routing algorithm works by having each router maintain a routing table, giving the best-known distance from source to destination and which route is used to get there.

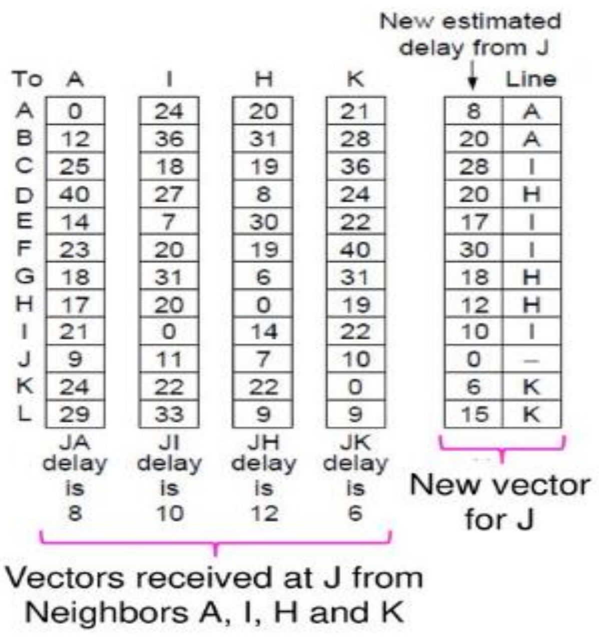
These tables are updated by exchanging the information with the neighbor having a direct link. Tables contain one entry for each route, this entry contains two-part, the preferred outgoing line use to reach the destination or an estimate of the time or distance to that destination.

The metric used can be the number of hops required to reach from source to destination. Time delay in milliseconds, the router can measure it with a special echo signal which the receiver can timestamp and send as soon as possible.

The router exchanges the network topology information periodically with its neighboring node and updates the information in the routing table. The cost of reaching the destination is estimated based on the metric, and an optimal path is obtained to send data packets from source to destination.

This updating process is illustrated in Fig. 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, respectively.





To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	–
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New vector for J	

Vectors received at J from Neighbors A, I, H and K

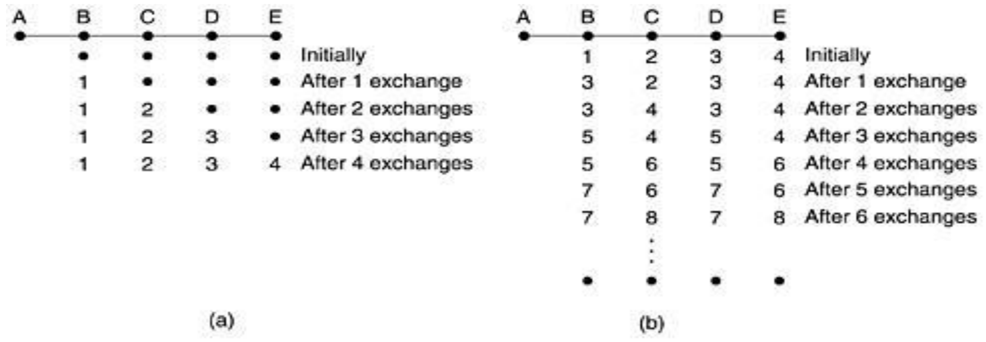
Consider how J computes its new route to router G. It knows that it can get to A in 8 msec, and A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec to G if it forwards packets bound for G to A.

· Similarly, it computes the delay to G via I, H, and K as 41 (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 Count-to-Infinity Problem

- Distance vector routing works in theory but has a serious drawback in practice: although it converges to the correct answer, it may do so slowly. In particular, it reacts rapidly to good news, but leisurely to bad news. Consider a router whose best route to destination X is large.

- To see how fast good news propagates, consider the five-node (linear) subnet of Fig, where the delay metric is the number of hops. Suppose A is down initially and all the other routers know this. In other words, they have all recorded the delay to A as infinity.



A B C D E
• • • • Initially
1 • • • After 1 exchange
1 2 • • After 2 exchanges
1 2 3 • After 3 exchanges
1 2 3 4 After 4 exchanges
(a)
A B C D E
1 2 3 4 Initially
3 2 3 4 After 1 exchange
3 4 3 4 After 2 exchanges
5 4 5 4 After 3 exchanges
5 6 5 6 After 4 exchanges
7 6 7 6 After 5 exchanges
7 8 7 8 After 6 exchanges
• • • •
(b)

- When A comes up, the other routers learn about it via the vector exchanges. For simplicity we will assume that there is a gigantic gong somewhere that is struck periodically to initiate a vector exchange at all routers simultaneously.

- At the time of the first exchange, B learns that its left neighbor has zero delay to A. B now makes an entry in its routing table that A is one hop away to the left. All the other routers still think that A is down.

- At this point, the routing table entries for A are as shown in the second row of Fig.(a). On the next exchange, C learns that B has a path of length 1 to A, so it updates its routing table to indicate a path of length 2, but D and E do not hear the good news until later.

- Clearly, the good news is spreading at the rate of one hop per exchange. In a subnet whose longest path is of length N hops, within N exchanges everyone will know about newly-revived lines and routers.

- On the second exchange, C notices that each of its neighbors claims to have a path to A of length 3. It picks one of the them at random and makes its new distance to A 4, as shown in the third row of Fig.

- Subsequent exchanges produce the history shown in the rest of Fig.From this figure, it should be clear why bad news travels slowly: no router ever has a value more than one higher than the minimum of all its neighbors.

- Gradually, all routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinity. For this reason, it is wise to set infinity to the longest path plus 1.

Hierarchical Routing and Broadcast Routing

Hierarchical Routing

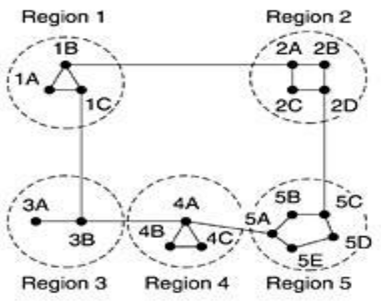
- As networks grow in size, the router routing tables grow proportionally. Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them.

- At a certain point the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network.

- When hierarchical routing is used, the routers are divided into what we will call regions, with each router knowing all the details about how to route packets to destinations within its own region, but knowing nothing about the internal structure of other regions.

- When different networks are interconnected, it is natural to regard each one as a separate region in order to free the routers in one network from having to know the topological structure of the other ones.

- Figure 3-15 gives a quantitative example of routing in a two-level hierarchy with five regions. The full routing table for router 1A has 17 entries, as shown in Fig. 3-15(b). When routing is done hierarchically, as in Fig. C there are entries for all the local routers as before.



-

(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

But all other regions have been condensed into a single router, so all traffic for region 2 goes via the 1B -2A line, but the rest of the remote traffic goes via the 1C -3B line. Hierarchical routing has reduced the table from 17 to 7 entries.
For example, the best route from 1A to 5C is via region 2, but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.

Broadcast Routing

- In some applications, hosts need to send messages to many or all other hosts. For example, a service distributing weather reports, stock market updates, or live radio programs might work best by broadcasting to all machines and letting those that are interested read the data.

- Sending a packet to all destinations simultaneously is called broadcasting; various methods have been proposed for doing it. One broadcasting method that requires no special features from the subnet is for the source to simply send a distinct packet to each destination.

- Not only is the method wasteful of bandwidth, but it also requires the source to have a complete list of all destinations. In practice this may be the only possibility, but it is the least desirable of the methods.

- Flooding is another obvious candidate. Although flooding is ill-suited for ordinary point-to-point communication, for broadcasting it might rate serious consideration, especially if none of the methods described below are applicable.

- The problem with flooding as a broadcast technique is the same problem it has as a point-to-point routing algorithm: it generates too many packets and consumes too much bandwidth. A third algorithm is multi -destination routing. If this method is used, each packet contains either a list of destinations or a bit map indicating the desired destinations.

- When a packet arrives at a router, the router checks all the destinations to determine the set of output lines that will be needed. (An output line is needed if it is the best route to at least one of the destinations.) The router generates a new copy of the packet for each output line to be used and includes in each packet only those destinations that are to use the line.

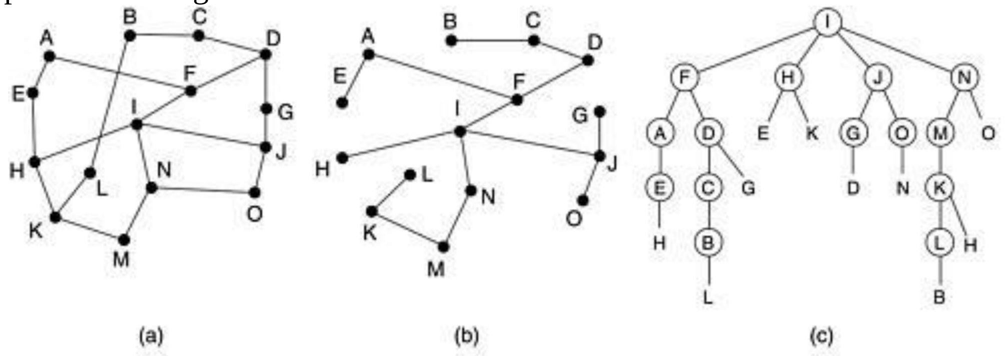
- In effect, the destination set is partitioned among the output lines. After a sufficient number of hops, each packet will carry only one destination and can be treated as a normal packet. Multidestination routing is like separately addressed packets, except that when several packets must follow the same route, one of them pays full fare and the rest ride free.

- A fourth broadcast algorithm makes explicit use of the sink tree for the router initiating the broadcast—or any other convenient spanning tree for that matter. A spanning tree is a subset of the subnet that includes all the routers but contains no loops.

- If each router knows which of its lines belong to the spanning tree, it can copy an incoming broadcast packet onto all the spanning tree lines except the one it arrived on. This method makes excellent use of bandwidth, generating the absolute minimum number of packets necessary to do the job.

- The only problem is that each router must have knowledge of some spanning tree for the method to be applicable. Sometimes this information is available (e.g., with link state routing) but sometimes it is not (e.g., with distance vector routing).

Figure 3-16. Reverse path forwarding. (a) A subnet. (b) A sink tree. (c) The tree built by reverse path forwarding.



- The principal advantage of reverse path forwarding is that it is both reasonably efficient and easy to implement. It does not require routers to know about spanning trees, nor does it have the overhead of a destination list or bit map in each broadcast packet as does multi destination addressing.

Congestion Control and Mechanisms

Congestion is a situation in Communication Networks in which too many packets are present in a part of the subnet, performance degrades. Congestion in a network may occur when the load on the network (i.e. the number of packets sent to the network) is greater than the capacity of the network (i.e. the number of packets a network can handle.). Network congestion occurs in case of traffic overloading.

In other words when too much traffic is offered, congestion sets in and performance degrades sharply.

The various causes of congestion in a subnet are:

• The input traffic rate exceeds the capacity of the output lines.

The routers are too slow to perform bookkeeping tasks (queuing buffers, updating tables, etc.).
• The routers' buffer is too limited.
• Congestion in a subnet can occur if the processors are slow. Slow speed CPU at routers will perform the routine tasks such as queuing buffers,

Congestion Control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. Congestion control mechanisms are divided into two categories, one category prevents the congestion from happening and the other category removes congestion after it has taken place.

• These two categories are:

1) **Open Loop Congestion Control**

i) In this method, policies are used to prevent the congestion before it happens.

ii) Congestion control is handled either by the source or by the destination. The various methods used for open loop congestion control are:

1) Retransmission Policy

a). The sender retransmits a packet, if it feels that the packet it has sent is lost or corrupted.
b). However retransmission in general may increase the congestion in the network. But we need to implement good retransmission policy to prevent congestion.
c). The retransmission policy and the retransmission timers need to be designed to optimize efficiency and at the same time prevent the congestion.

2) Window Policy

a). To implement window policy, selective reject window method is used for congestion control.
b). Selective Reject method is preferred over Go-back-n window as in Go-back-n method, when timer for a packet times out, several packets are resent, although some may have arrived safely at the receiver. Thus, this duplication may make congestion worse.
c). Selective reject method sends only the specific lost or damaged packets.

3) Acknowledgement Policy

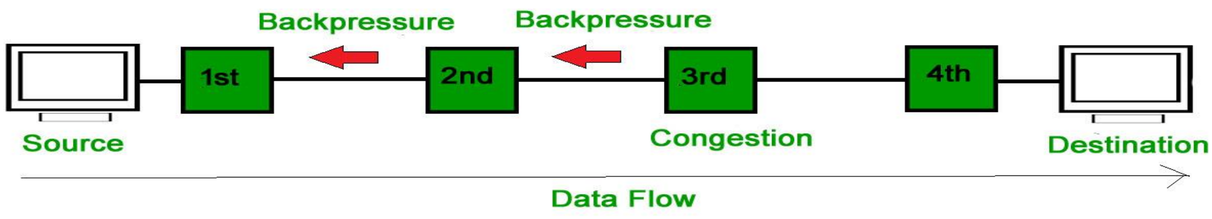
a). The acknowledgement policy imposed by the receiver may also affect congestion.
b). If the receiver does not acknowledge every packet it receives it may slow down the sender and help prevent congestion.
c). Acknowledgments also add to the traffic load on the network. Thus, by sending fewer acknowledgements we can reduce load on the network.
d). To implement it, several approaches can be used:
A receiver may send an acknowledgement only if it has a packet to be sent.
A receiver may send an acknowledgement when a timer expires.
A receiver may also decide to acknowledge only N packets at a time.

2) **Closed Loop Congestion Control**

Closed loop congestion control mechanisms try to remove the congestion after it happens.The various methods used for closed loop congestion control are:

i. Backpressure
a). Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow.

b). The backpressure technique can be applied only to virtual circuit networks. In such virtual circuit each node knows the upstream node from which a data flow is coming.
c). In this method of congestion control, the congested node stops receiving data from the immediate upstream node or nodes.
d). This may cause the upstream node on nodes to become congested, and they, in turn, reject data from their upstream node or nodes.



e. As shown in fig node 3 is congested and it stops receiving packets and informs its upstream node 2 to slow down. Node 2 in turns may be congested and informs node 1 to slow down. Now node 1 may create congestion and informs the source node to slow down. In this way the congestion is alleviated. Thus, the pressure on node 3 is moved backward to the source to remove the congestion.

ii. Choke Packet
a). In this method of congestion control, congested router or node sends a special type of packet called choke packet to the source to inform it about the congestion.
b). Here, congested node does not inform its upstream node about the congestion as in backpressure method.

c). In choke packet method, congested node sends a warning directly to the source station i.e. the intermediate nodes through which the packet has traveled are not warned.

Congestion control algorithms

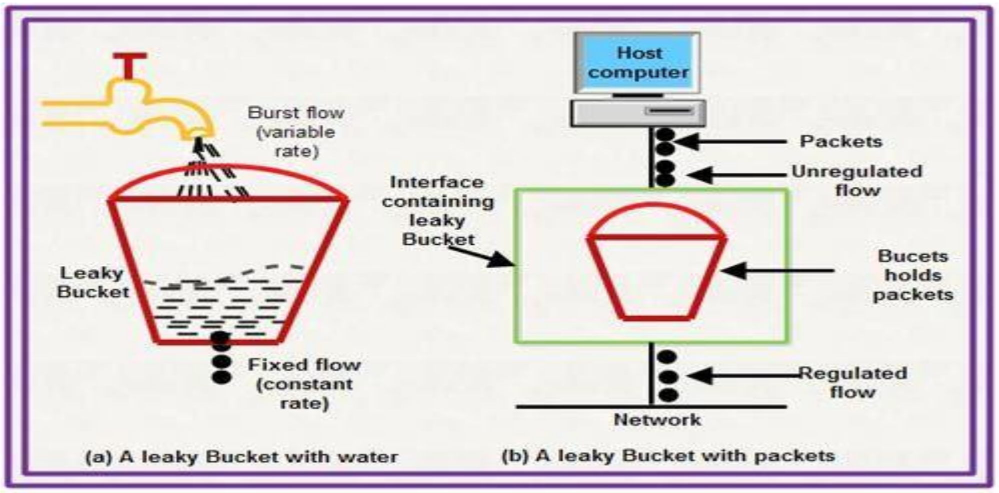
1.Leaky Bucket Algorithm

• It is a traffic shaping mechanism that controls the amount and the rate of the traffic sent to the network.

• A leaky bucket algorithm shapes bursty traffic into fixed rate traffic by averaging the data rate.

• Imagine a bucket with a small hole at the bottom.

• The rate at which the water is poured into the bucket is not fixed and can vary but it leaks from the bucket at a constant rate. Thus (as long as water is present in bucket), the rate at which the water leaks does not depend on the rate at which the water is input to the bucket. Also, when the bucket is full, any additional water that enters into the bucket spills over the sides and is lost.


Burst flow (variable rate)
Leaky Bucket
Fixed flow (constant rate)
(a) A leaky Bucket with water
Host computer
Packets
Unregulated flow
Interface containing leaky Bucket
Bucets holds packets
Regulated flow
Network
(b) A leaky Bucket with packets

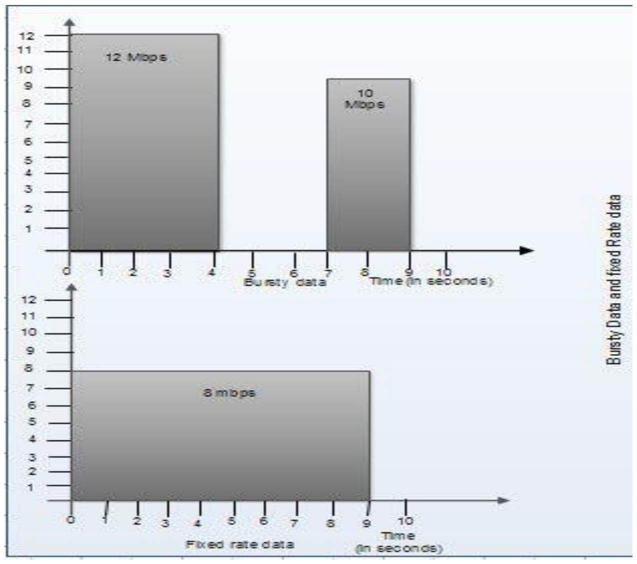
•

• The same concept can be applied to packets in the network. Each network interface contains a leaky bucket and the following **steps** are involved in leaky bucket algorithm:

1. When host wants to send packet, packet is thrown into the bucket.
2. The bucket leaks at a constant rate, meaning the network interface transmits packets at a constant rate.
3. Bursty traffic is converted to a uniform traffic by the leaky bucket.
4. In practice the bucket is a finite queue that outputs at a finite rate.

Consider that data is coming from the source at variable speeds. Suppose that a source sends data at 12 Mbps for 4 seconds. Then there is no data for 3 seconds. The source again transmits data at a rate of 10 Mbps for 2 seconds. Thus, in a time span of 9 seconds, 68 Mb data has been transmitted.

If a leaky bucket algorithm is used, the data flow will be 8 Mbps for 9 seconds. Thus constant flow is maintained.

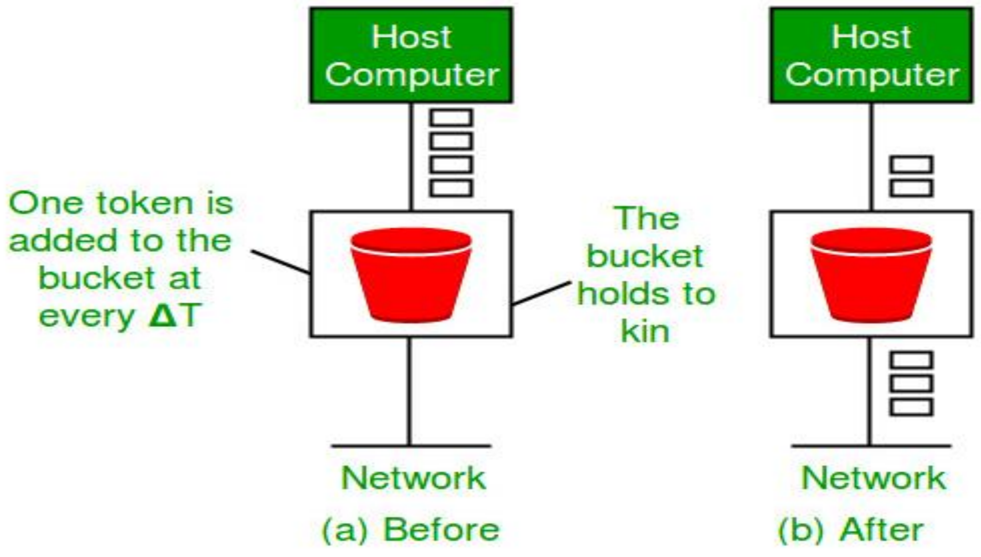


2. Token bucket Algorithm

- The leaky bucket algorithm has a rigid output design at an average rate independent of the bursty traffic.
- In some applications, when large bursts arrive, the output is allowed to speed up. This calls for a more flexible algorithm, preferably one that never loses information. Therefore, a token bucket algorithm finds its uses in network traffic shaping or rate-limiting.
- It is a control algorithm that indicates when traffic should be sent. This order comes based on the display of tokens in the bucket.
- The bucket contains tokens. Each of the tokens defines a packet of predetermined size. Tokens in the bucket are deleted for the ability to share a packet.
- When tokens are shown, a flow to transmit traffic appears in the display of tokens.
- No token means no flow sends its packets. Hence, a flow transfers traffic up to its peak burst rate in good tokens in the bucket.

Need of token bucket Algorithm:-

- The leaky bucket algorithm enforces output pattern at the average rate, no matter how bursty the traffic is. So in order to deal with the bursty traffic we need a flexible algorithm so that the data is not lost. One such algorithm is token bucket algorithm.



Steps of this algorithm can be described as follows:
1. In regular intervals tokens are thrown into the bucket. ƒ
2. The bucket has a maximum capacity. ƒ
3. If there is a ready packet, a token is removed from the bucket, and the packet is sent.
4. If there is no token in the bucket, the packet cannot be sent.

Quality of Service (QOS)

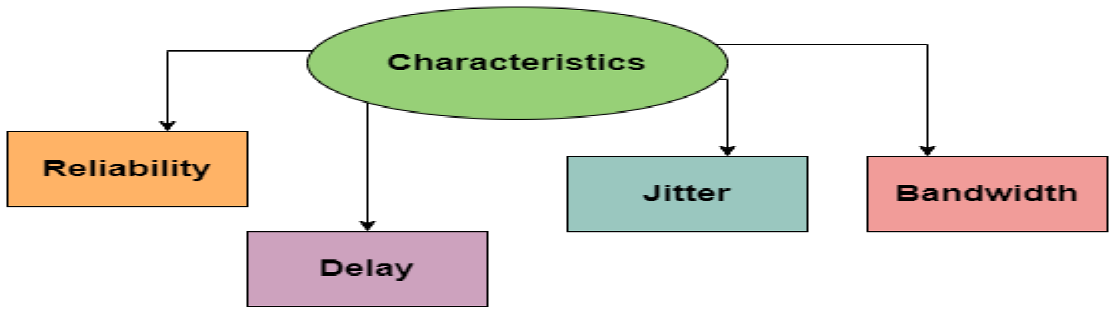
Quality of Service (QoS) is a group of technologies that operate on a network to ensure that high-priority traffic and applications may be reliably carried out even when the network's capacity is constrained.

Additionally, the QoS specifies that supporting priority for one or more flows will not fail other flows. A flow can consist of a packet from a particular application or an incoming interface as well as source and destination addresses, source and destination socket numbers, session identifiers, and packets.

Flow Characteristics

Given below are four types of characteristics that are mainly attributed to the flow and these are as follows:

- Reliability
- Delay
- Jitter
- Bandwidth



Reliability

It is one of the main characteristics that the flow needs. If there is a lack of reliability then it simply means losing any packet or losing an acknowledgement due to which retransmission is needed.

Reliability becomes more important for electronic mail, file transfer, and for internet access.

Delay

Another characteristic of the flow is the delay in transmission between the source and destination. During audio conferencing, telephony, video conferencing, and remote conferencing there should be a minimum delay.

Jitter

It is basically the variation in the delay for packets that belongs to the same flow. Thus Jitter is basically the variation in the packet delay. Higher the value of jitter means there is a large delay and the low jitter means the variation is small.

Bandwidth

The different applications need different bandwidth.

There are 2 types of Quality of Service Solutions:

1. **<u>Stateless solution:</u>** Here, the server is not required to keep or store the server information or session details to itself. The **routers** maintain no fine-grained state about traffic, one positive factor of this is, that it's **scalable and robust**. But also, it has weak services as there is **no guarantee about the kind of performance delay** in a particular application which we encounter. In the stateless solution, the server and client are **loosely coupled** and can act.

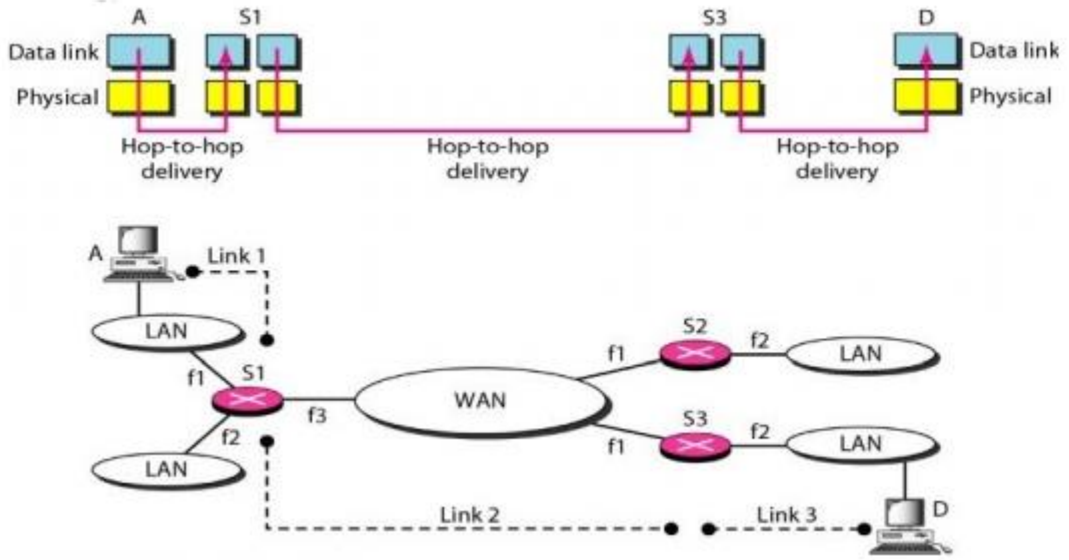
2. **Stateful solution:** Here, the server is required to maintain the **current state and session information**, the **routers maintain per-flow state** as the flow is very important in providing the Quality-of-Service which is providing powerful services such as guaranteed services and high resource utilization, provides protection, and is much **less scalable and robust**. Here, the server and client are **tightly bounded**.

INTERNETWORKING

The physical layer and the data link layer of the particular network locally. These two layers are used for the delivery of the data on the network from one node to the other node.

Need for Network Layer-

- To solve the problem of the delivery through different links the network or the internetwork layer was designed. The network layer is responsible for host to host delivery and for the routing of the packets through switches.



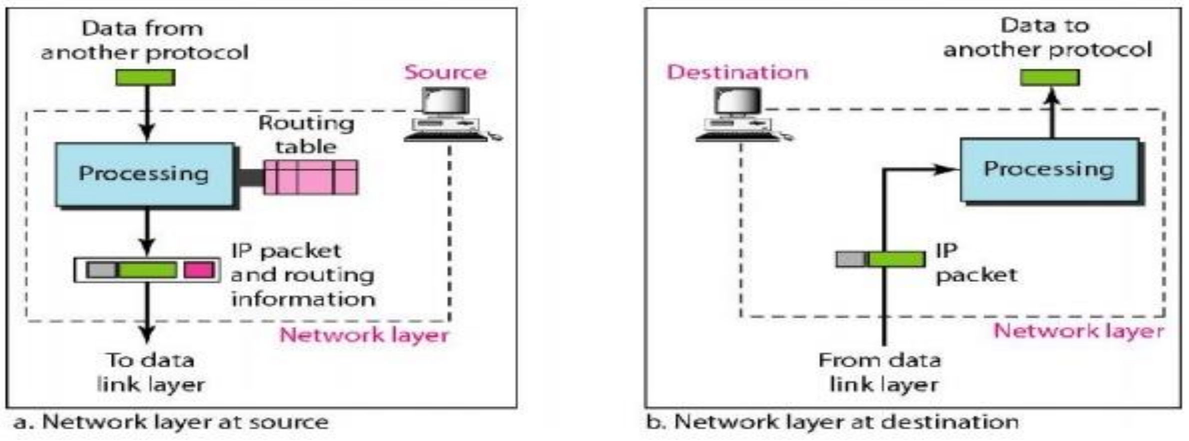
-
- When the packet arrives the switch consults its routing table and then finds the interface from which the packet must be sent. Then the packet after some changes in the header with the routing informations is passed to the data link layer. The network layer at the destination is responsible for some address verification.

Internet as a Datagram Network

The internet at the network layer is the packet switched network. The internet has been chosen as the datagram approach to the switching in the network layer. It uses the universal addresses defined in the network layer to route packets from the source to the destination. The switching at the network layer in the internet uses the datagram approach to the packet switching. Packets in the IPv4 layer are called as datagrams.

Internet as a Connectionless Network-

The delivery of the packet can be accomplished by connection oriented or connection less network. In the connection oriented the source first makes the connection with the destination before sending the packet. after the connection is established a sequence of the packets from same source to the destination are sent one after the other in the sequential manner.when all the packet data are sent the connection is terminated.



Fragmentation

A datagram can flow through different networks.Each router decodes the Ipv4 datagram from the frame it receives then processes it further encaptulates it in the other frame.The size of the frame received depends on the protocol used by the physical layer through which the frame travelled.

Checksum

The implementation of the checksum in the IPv4 packet follows the same principles.

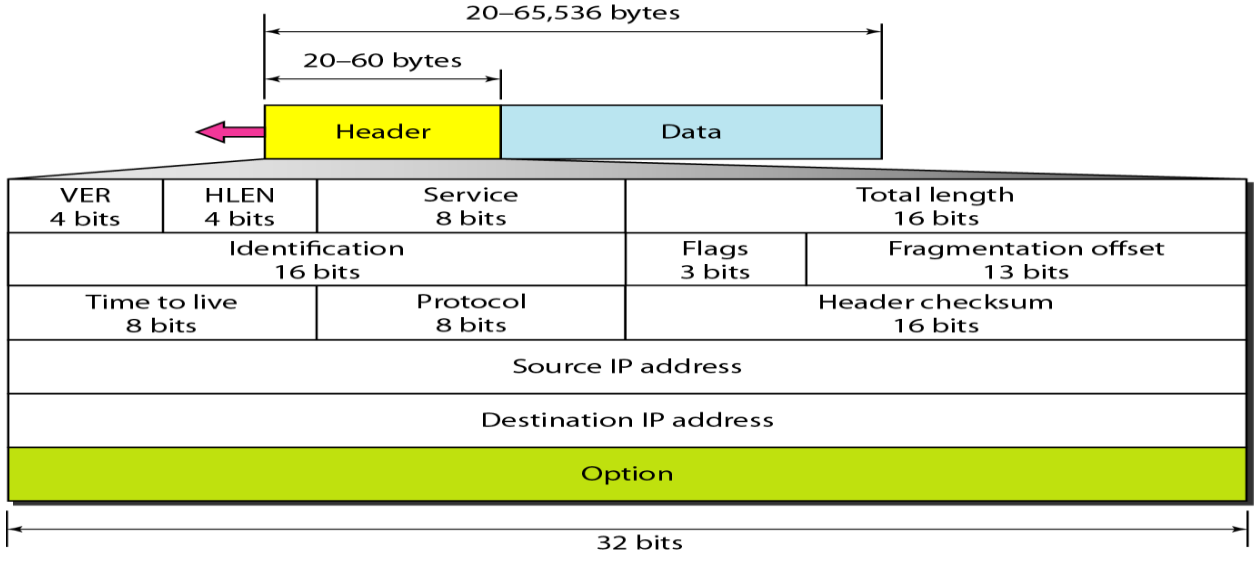
- The value of the checksum field is set to 0.
- The header is divided into 16 bit sections and then added together.The result is complemented then inserted into the chechsum field.
- The checksum in the IPv4 covers only the packets not data.

Internet Protocol (IP)

The Internet Protocol (IP) is the network layer communications protocol in the Internet protocol suite that delivers datagrams (basic transfer units associated with a packet-switched network) across network boundaries. Its routing function enables internetworking, and essentially establishes the Internet.

In simpler words, the Internet Protocol (IP) is a protocol, or set of rules that govern the routing of data packets such that it travels across networks and reaches its desired destinations. Data passing through the Internet is divided into smaller pieces known as packets. All devices and/or domains connected to the Internet come with an IP address, and each data packet has an IP information attached to it, which in turn helps routers send these packets to the correct place. After the packets reach their respective destinations, they are handled as per the transport protocol being used in combination with IP. There are two generations of IP packets, called IPv4 (IP version 4) and IPv6 (IP version 6).

IPV4 Datagram Format



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

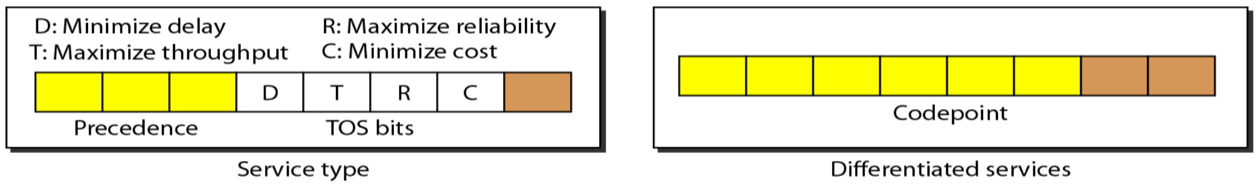
Version (VER): This 4-bit field defines the version of the IPv4 protocol. Currently the version is 4. In IPv4, the value of its four bits is set to 0100. However, version 6 (or IPv6) may totally replace version 4 in the future.

Internet Header Length: Internet header length, shortly known as IHL, is 4 bits in size. It is also called HELEN (Header Length).

Services: This 8 hit field was previously called services type but is now called differentiated services.

The various bits in service type are:

• A 3-bit precedence field that defines the priority of datagram in issues such as congestion. This 3-bit subfield ranges from 0 (000 in binary) to 7 (111 in binary).



• After 3-bit precedence there are four flag bits. These bits can be either 0 or 1 and only one of the bits can have value of 1 in each datagram.

The various flag bits are:

D : Minimize delay

T : Maximize throughout

R : Maximize reliability

C : Minimize Cost

Total length. This is a In-bit field that defines the total length (header plus data) of theIPv4 datagram in bytes.

Identification: Identification is a packet that is used to identify fragments of an IP datagram uniquely.

IP Flags: Flag is a three-bit field that helps you to control and identify fragments.

Bit 0: is reserved and has to be set to zero

Bit 1: means do not fragment

Bit 2: means more fragments.

Fragment Offset − This offset tells the exact position of the fragment in the original IP Packet.

Time to live: It is an 8-bit field that indicates the maximum time the Datagram will be live in the internet system. The time duration is measured in seconds, and when the value of TTL is zero, the Datagram will be erased.

Protocol: This 8-bit field defines the higher-level protocol that uses the services of the IPv4 layer. An IPv4 datagram can encapsulate data from several higher-level protocols such as TCP, UDP, ICMP, and IGMP.

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

Destination address: This 32-bit field defines the IPv4 address of the destination