UNIT – III

The Network Layer

Network layer

- The network layer or layer 3 of the OSI (Open Systems Interconnection) model is concerned delivery of data packets from the source to the destination across multiple hops or links.
- It is the lowest layer that is concerned with end − to − end transmission. The designers who are concerned with designing this layer needs to cater to certain issues.
- These issues encompass the services provided to the upper layers as well as internal design of the layer.
- Segment in Network layer is referred as Packet.
- Network layer is implemented by networking devices such as Routers.
- The functions of the Network layer are Routing and Logical addressing

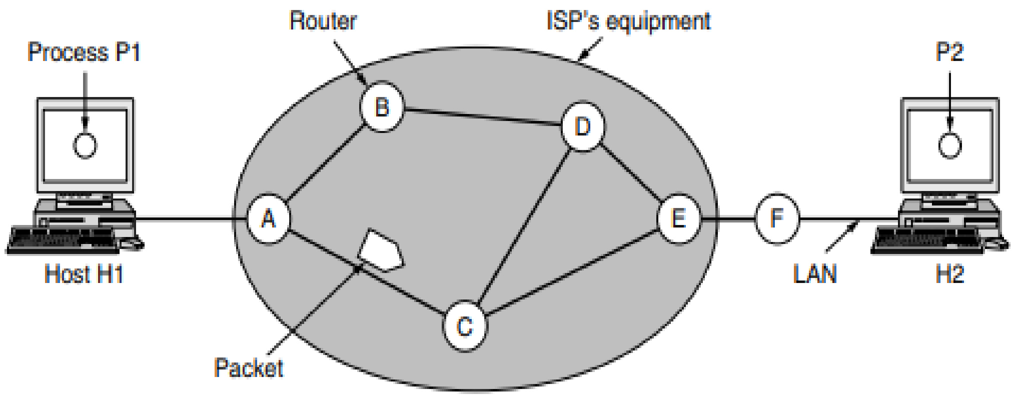
The design issues can be elaborated under four heads

1. Store − and − Forward Packet Switching
2. Services to Transport Layer
3. Providing Connection Oriented Service
4. Providing Connectionless Service

1. **Store – and – Forward Packet Switching**
   - The network layer operates in an environment that uses store and forward packet switching.
   - The node which has a packet to send, delivers it to the nearest router.
   - The packet is stored in the router until it has fully arrived and its checksum is verified for error detection.
   - Once, this is done, the packet is forwarded to the next router. Since, each router needs to store the entire packet before it can forward it to the next hop, the mechanism is called store − and − forward switching.
   - Not suited for streaming media and real time applications.

   Working Principle

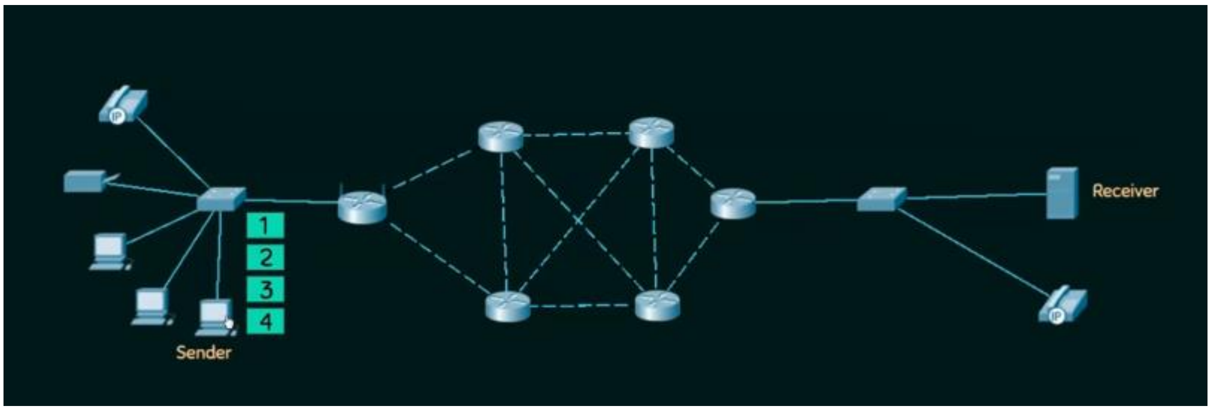
     - The node which has a packet to send, delivers it to the nearest node, i.e. router.
     - The packet is stored in the router until it has fully arrived and its checksum is verified for error detection.
     - Once, this is done, the packet is transmitted to the next router. The same process is continued in each router until the packet reaches its destination.
     - The following scenario exemplifies the mechanism.

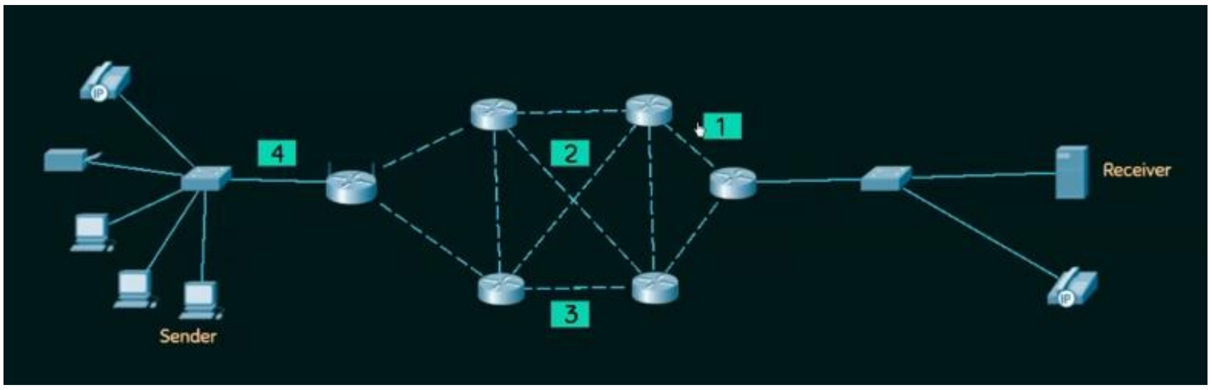


2. **Services to Transport Layer**
   - The network layer provides service its immediate upper layer, namely transport layer, through the network − transport layer interface. The two types of services provided are
   - Connection − Oriented Service − In this service, a path is setup between the source and the destination, and all the data packets belonging to a message are routed along this path.
   - Connectionless Service − In this service, each packet of the message is considered as an independent entity and is individually routed from the source to the destination.
   - The objectives of the network layer while providing these services are
     1. The services should not be dependent upon the router technology.
     2. The router configuration details should not be of a concern to the transport layer.
     3. A uniform addressing plan should be made available to the transport layer whether the network is a LAN, MAN or WAN.
3. **Implementation of Connectionless Service**
   - A Connectionless service is a data communication between two nodes where the sender sends data without ensuring whether the receiver is available to receive the data.
   - Here, each data packet has the destination address and is routed independently irrespective of the other packets.
   - Thus the data packets may follow different paths to reach the destination.
   - There's no need to setup connection before sending a message and relinquish it after the message has been sent. The data packets in a connectionless service are usually called datagrams.

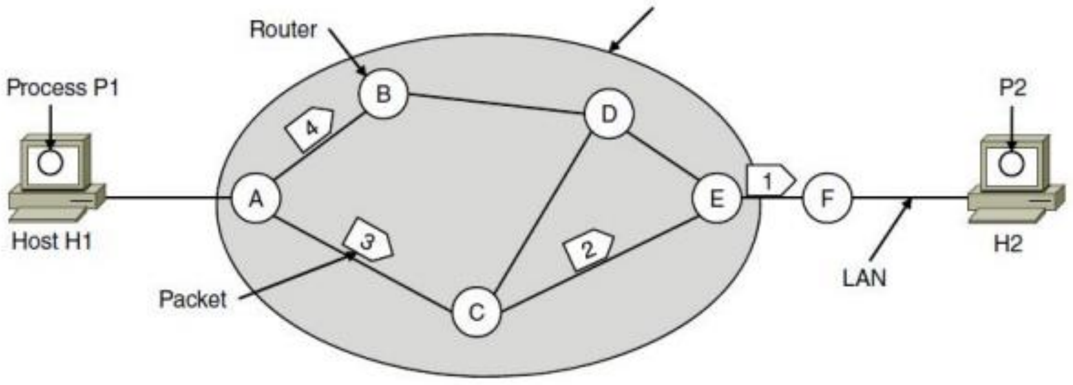
Datagram approach

- This is approach each independent entity is called as datagram.
- Datagrams contain destination information and the intermediary devices uses this information to forward datagrams to right destination
- In this approach the path is not fixed.
- Intermediate nodes take the routing decisions to forward the packets.





Example



A's table (initially)

A	⊠
B	B
C	C
D	B
E	C
F	C

Dest. Line

A's table (later)

A	⊠
B	B
C	C
D	B
E	D
F	D

C's Table

A	A
B	A
C	⊠
D	E
E	E
F	E

E's Table

A	C
B	D
C	C
D	D
E	⊠
F	F

- Let us assume for this example 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 send each of them in turn to router A.
- Every router has an internal table telling it where to send packets for each of the possible destinations. Each table entry is a pair(destination and the outgoing line). Only directly connected lines can be used. A’s initial routing table is shown in the figure under the label ‘‘initially.’’ At A, packets 1, 2, and 3 are stored briefly, having arrived on the incoming link.
- Then each packet is forwarded according to A’s table, onto the outgoing link to C within a new frame. Packet 1 is then forwarded to E and then to F. However, something different happens to packet 4. When it gets to A it is sent to router B, even though it is also destined for F.
- For some reason (traffic jam along ACE path), A decided to send packet 4 via a different route than that of the first three packets. Router A 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

Connectionless Protocols:

- These protocols simply allow data to be transferred without any link among processes. Some Of data packets may also be lost during transmission. Some of protocols for connectionless services are given below:
    1. **Internet Protocol (IP)**
       This protocol is connectionless. In this protocol, all packets in IP network are routed independently. They might not go through same route.
    2. **User Datagram Protocol (UDP)**
       This protocol does not establish any connection before transferring data. It just sends data that’s why UDP is known as connectionless.
    3. **Internet Control Message Protocol (ICMP)**
       ICMP is called connectionless simply because it does not need any hosts to handshake before establishing any connection.

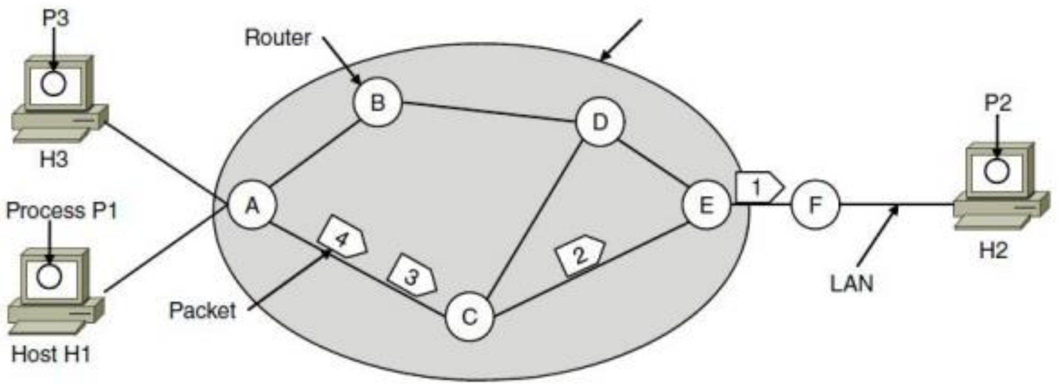
4. A connection-oriented service

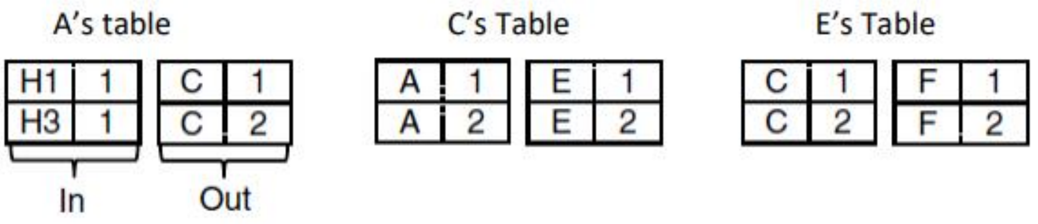
- If connection-oriented service is used, a path from the source router all the way to the destination router must be established before any data packets can be sent. This connection is called a **Virtual Circuit(VC)**, and the network is called a **Virtual- Circuit Network**
- When a connection is established, 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.

  Virtual Circuit-Switched Connection

  Virtual Circuit-Switched Connection or Virtual Circuit Switching is also known as Connection-Oriented Switching. In this connection, a preplanned route or path is established before data or messages are transferred or sent.

- The message Is transferred over this network is such a way that it seems to user that there is a dedicated route or path from source or sender to destination or receiver.





A's table

In		Out	
H1	1	C	1
H3	1	C	2

C's Table

A	1	E	1
A	2	E	2

E's Table

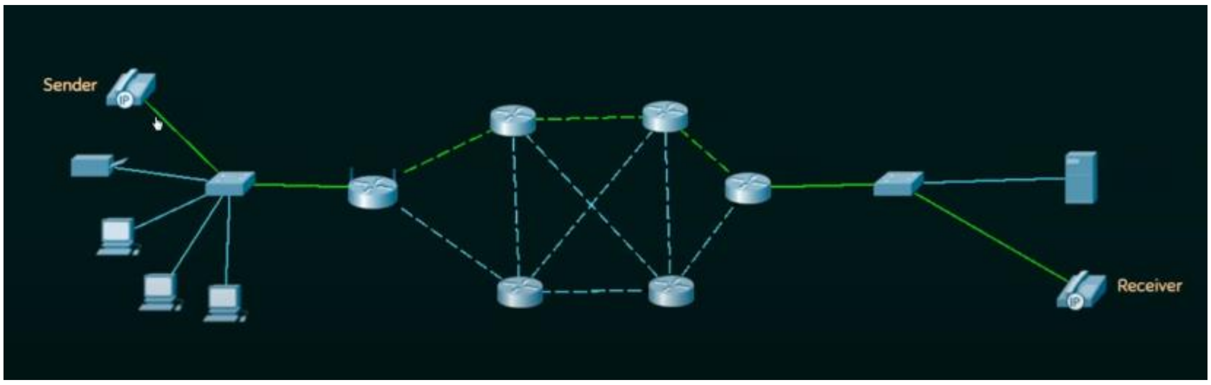
C	1	F	1
C	2	F	2

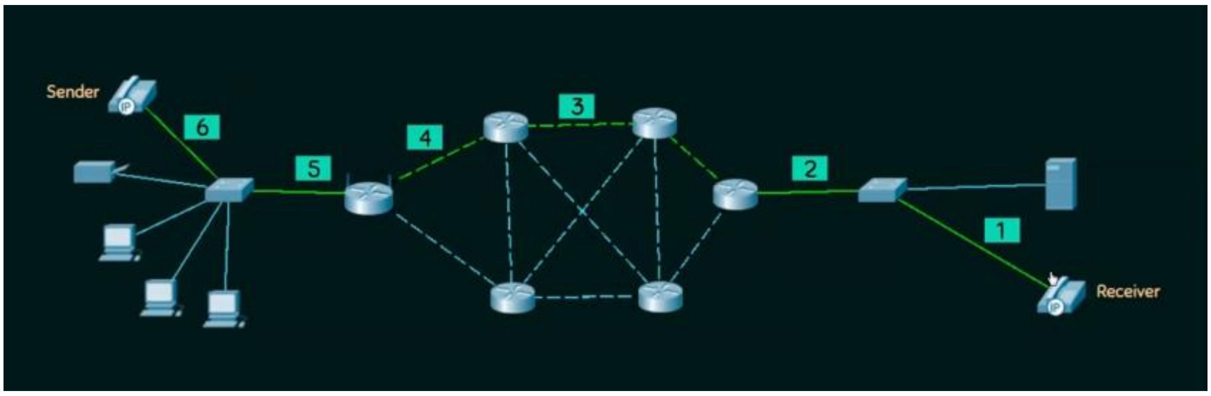
- As an example, consider the situation shown in Figure. Here, host H1 has established connection 1 with host H2.
- This connection is remembered as the first entry in each of the routing tables. The first line of A's table says that if a packet bearing connection identifier 1 comes in from H1, it is 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.
- Now let us consider what happens if H3 also wants to establish a connection to H2. It chooses connection identifier 1 (because it is initiating the connection and this is its only connection) and tells the network 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.

Operations:

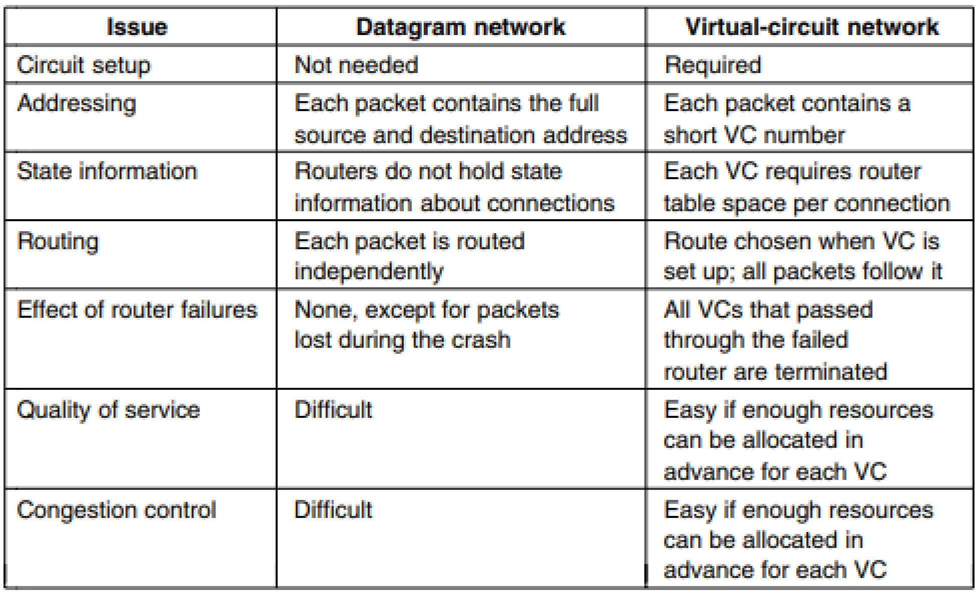
- There is a sequence of operations that are needed to b followed by users. These operations are given below :
  1. **Establishing Connection –**
     It generally requires a session connection to be established just before any data is transported or sent with a direct physical connection among sessions.

2. **Transferring Data or Message –**
   When this session connection is established, then we transfer or send message or data.
3. **Releasing the Connection –**
   After sending or transferring data, we release connection.





Comparison of Virtual Circuit and Datagram Networks



Issue	Datagram network	Virtual-circuit network
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 algorithms

- A routing algorithm is a procedure that lays down the route or path to transfer data packets from source to the destination.
- They help in directing Internet traffic efficiently. After a data packet leaves its source, it can choose among the many different paths to reach its destination.
- Routing algorithm mathematically computes the best path, i.e. "least – cost path" that the packet can be routed through.
- 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, optimality

Classification of Routing Algorithms:

The routing algorithms can be classified as follows:

1. Adaptive Algorithms

2. Non-adaptive algorithms

Adaptive Routing Algorithms

- Adaptive routing algorithms, also known as dynamic routing algorithms, makes routing decisions dynamically depending on the network conditions.
- It constructs the routing table depending upon the network traffic and topology. change their routing decisions to reflect changes in the topology, and usually the traffic as well.
- Adaptive algorithms differ in
  1) Where they get their information (e.g., locally, from adjacent routers, or from all routers),
  2) When they change the routes (e.g., every ΔT sec, when the load changes or when the topology changes), and
  3) What metric is used for optimization (e.g., distance, number of hops, or estimated transit time). This procedure is called dynamic routing
- Adaptive algorithms are

  1. Shortest path routing algorithm

  2. Flooding

Non – Adaptive Routing Algorithms

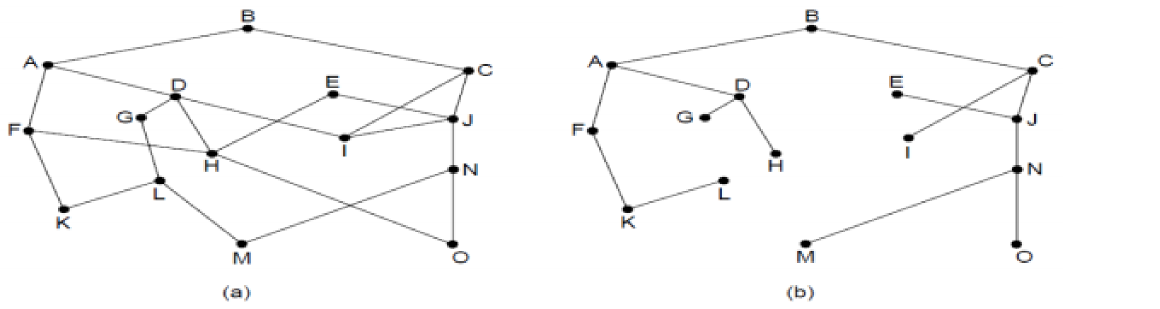
- Non-adaptive Routing algorithms, also known as static routing algorithms, construct a static routing table to determine the path through which packets are to be sent.
- The static routing table is constructed based upon the routing information stored in the routers when the network is booted up.
- Non-adaptive algorithms are

  1. Distance vector routing algorithm

  2. Link state routing

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
- As a direct consequence of the optimality principle, we 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. The goal of all routing algorithms is to discover and use the sink trees for all routers



(a) A network. (b) A sink tree for router *B.*

Shortest path routing algorithm

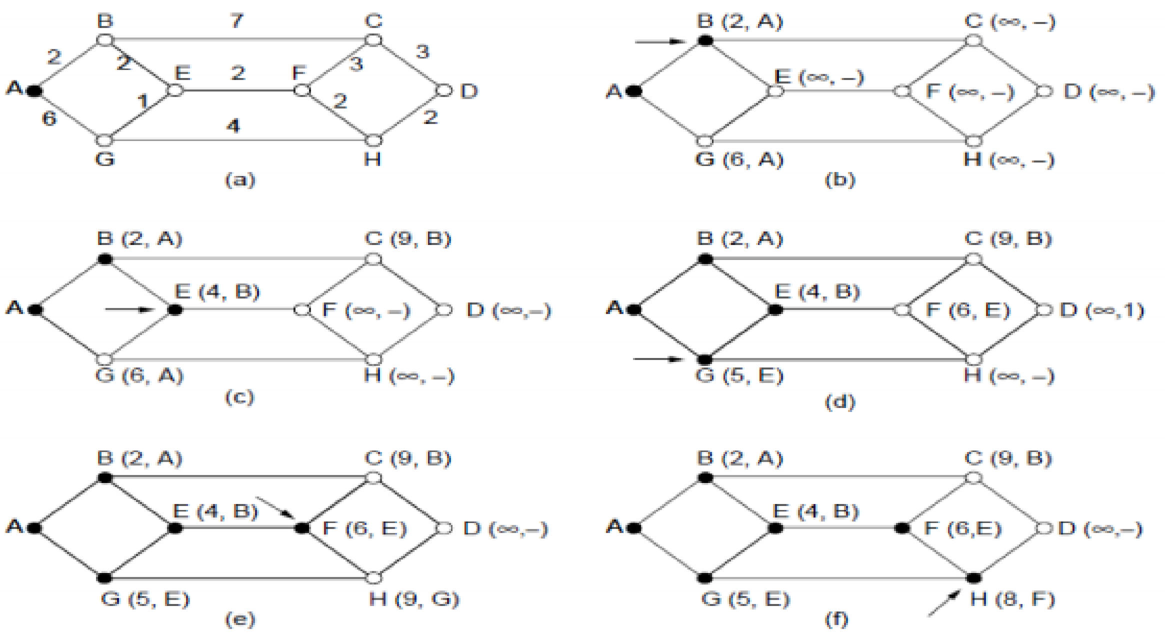
- In shortest path routing, The topology communications network is represented using a directed weighted graph.
- The nodes in the graph represent witching elements and the directed arcs in the graph represent communication links between switching elements.
- Each arc has a weight that represent the cost of sending a packet between two node in a particular direction
- This cost is generally a positive value that can inculcate such factors a delay, throughput, errorrate, monetary cost.
- A path between two nodes may go through several intermediate nodes and arc.
- The objective in shortest path routing is to find a path between two nodes that has the smallest total cost where the total cost of a path is the sum of the arc costs in that path

Shortest Path Routing (Dijkstra's) 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 or link.
- To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph
  1. Start with the local node (router) as the root of the tree. Assign a cost of 0 to this node and make it the first permanent node.
  2. Examine each neighbour of the node that was the last permanent node.
  3. Assign a cumulative cost to each node and make it tentative

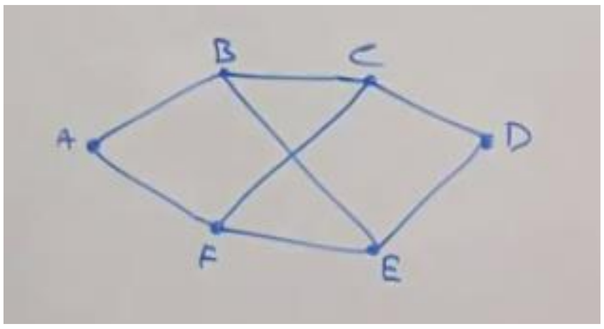
4. Among the list of tentative nodes a. Find the node with the smallest cost and make it Permanent b. If a node can be reached from more than one route then select the route with the shortest cumulative cost.

5. Repeat steps 2 to 4 until every node becomes permanent



Flooding

- Flooding is a non-adaptive routing technique following this simple method: when a data packet arrives at a router, it is sent to all the outgoing links except the one it has arrived on.
- Requires no network information like topology, load condition ,cost of different paths
- One major problem of this algorithm is that it generates a large number of duplicate packets on the network.



Several measures are takes to stop the duplication of packets. These are:

1. One solution is to include a **hop counter** in the header of each packet. This counter is decremented at each hop along the path. When this counter reaches zero the packet is discarded. Ideally, the hop counter should become zero at the destination hop, indicating that there are no more intermediate hops and destination is reached. This requires the knowledge of exact number of hops from a source to destination.

2. Another technique is to keep the track of the packed that have been flooded, to avoid sending

them a second time. For this, the source router put a **sequence number** in each packet it receives from its hosts. Each router then needs a list per source router telling which sequence numbers originating at that source have already been seen. If an incoming packet is on the list, it is not flooded.

3. Another solution is to use **selective flooding**. In selective flooding the routers do not send every incoming packet out on every output line. Instead packet is sent only on those lines which are approximately going in the right direction.

Distance vector routing algorithm

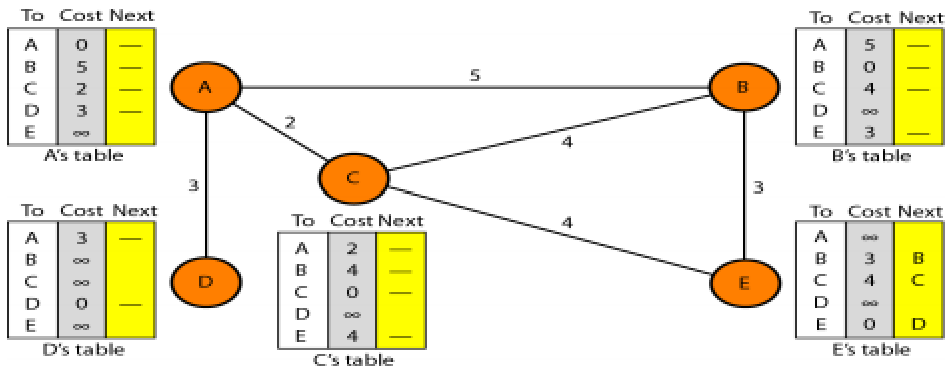
- A non-adaptive routing algorithm example is distance vector routing algorithm which dynamically changes a routing path along with decisions between nodes which is generally apply on mobile node of a network
- The Distance vector algorithm is iterative, asynchronous and distributed.
    - **Distributed:** It is distributed in that each node receives information from one or more of its directly attached neighbours, performs calculation and then distributes the result back to its neighbours.
    - **Iterative:** It is iterative in that its process continues until no more information is available to be exchanged between neighbours.
    - **Asynchronous:** It does not require that all of its nodes operate in the lock step with each other.
- Distance vector routing can find out an optimal path between node of a network where each and every node maintains a vector (table) of minimum distances to every node
- If the distance changed between the nodes that will reflects an optimal path and network topology. For this reason the vectors or tables dynamically updated depending on network nodes mobility.
- The distance vector routing algorithm is introduced by **Bell-man Ford in the year of 1962** with major revision.
- The algorithm is practically applied on the network known as **ARPANET**(Advance research for project agency network)

Three Keys to understand the working of Distance Vector Routing Algorithm:

1. **Knowledge about the whole network:** Each router shares its knowledge through the entire network. The Router sends its collected knowledge about the network to its neighbours.
2. **Routing only to neighbours:** The router sends its knowledge about the network to only those routers which have direct links. The router sends whatever it has about the network through the ports. The information is received by the router and uses the information to update its own routing table.
3. **Information sharing at regular intervals:** Within 30 seconds, the router sends the information to the neighbouring routers.

Mainly 3 things in this Initialization Sharing Updating

1. **Initialization:** Each node can know only the distance between itself and its immediate neighbours, those directly connected to it. So for the moment, we assume that each node can send a message to the immediate neighbours and find the distance between itself and these neighbours. Below fig shows the initial tables for each node. The distance for any entry that is not a neighbour is marked as infinite (unreachable).



To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	∞	

A's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	∞	
E	3	—

B's table

To	Cost	Next
A	3	—
B	∞	
C	∞	
D	0	—
E	∞	

D's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	∞	
E	4	—

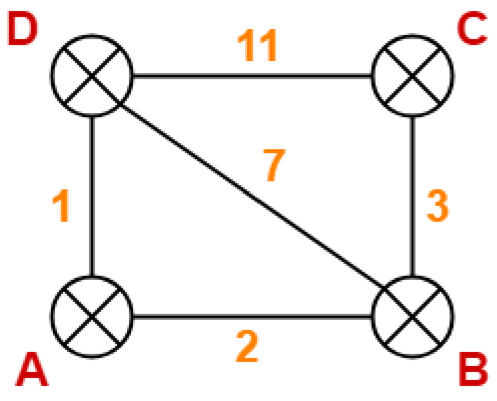
C's table

To	Cost	Next
A	∞	
B	3	B
C	4	C
D	∞	
E	0	D

E's table

2. **Sharing:**The whole idea of distance vector routing is the sharing of information between neighbours. Although node A does not know about node E, node C does. So if node C shares its routing table with A, node A can also know how to reach node E. On the other hand, node C does not know how to reach node D, but node A does. If node A shares its routing table with node C, node C also knows how to reach node D. In other words, nodes A and C, as immediate neighbours, can improve their routing tables if they help each other. NOTE: In distance vector routing, each node shares its routing table with its immediate neighbours periodically and when there is a change
3. **Updating:**When a node receives a two-column table from a neighbour, it needs to update its routing table.

Example



Step-01:

- Each router prepares its routing table using its local knowledge.
- Routing table prepared by each router is shown below-

At Router A

Destination	Distance	Next hop
A	0	A
B	2	B
C	∞	-
D	1	D

At Router B

Destination	Distance	Next hop
A	2	A
B	0	B
C	3	C
D	7	D

At Router C

Destination	Distance	Next hop
A	∞	-
B	3	B
C	0	C
D	11	D

At Router D

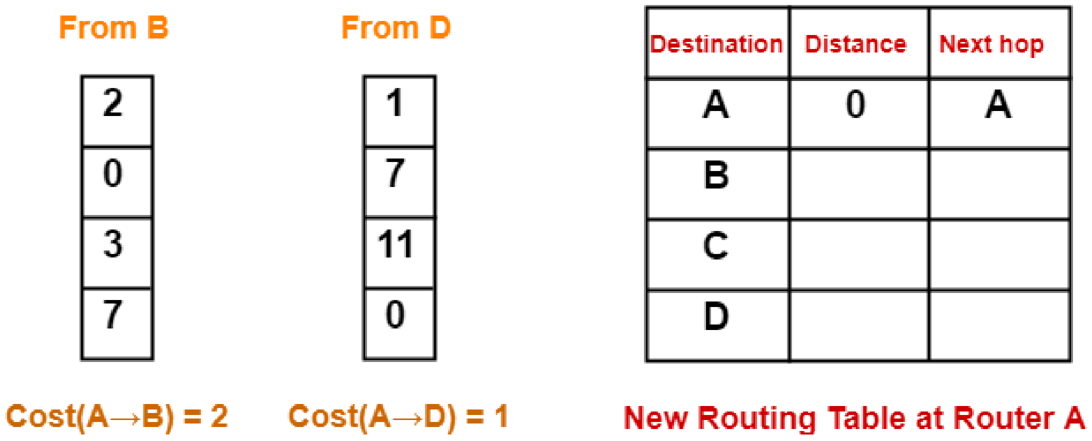
Destination	Distance	Next hop
A	1	A
B	7	B
C	11	C
D	0	D

Step-02:

- Each router exchanges its distance vector obtained in Step-01 with its neighbours.
- After exchanging the distance vectors, each router prepares a new routing table.

At Router A-

- Router A receives distance vectors from its neighbours B and D.
- Router A prepares a new routing table as-



From B

2
0
3
7

Cost(A→B) = 2

From D

1
7
11
0

Cost(A→D) = 1

Destination	Distance	Next hop
A	0	A
B		
C		
D		

New Routing Table at Router A

- Cost of reaching destination B from router A = min { 2+0 , 1+7 } = 2 via B.
- Cost of reaching destination C from router A = min { 2+3 , 1+11 } = 5 via B.
- Cost of reaching destination D from router A = min { 2+7 , 1+0 } = 1 via D.

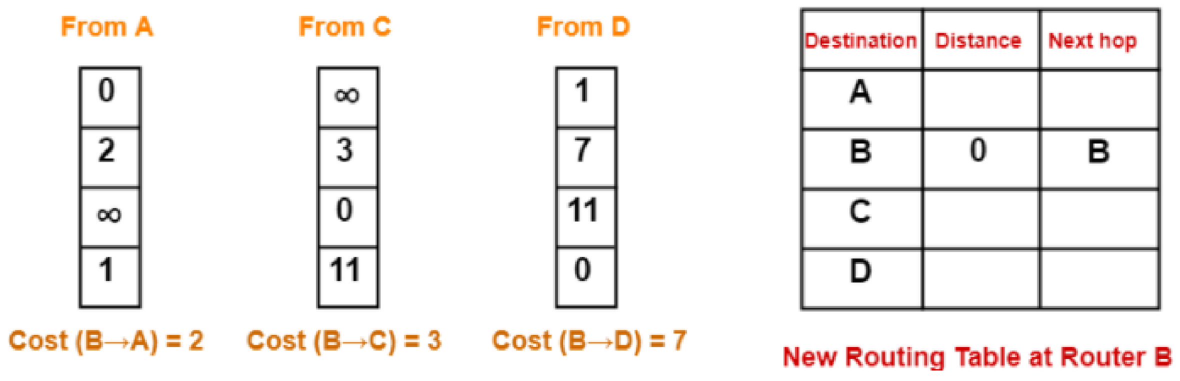
Thus, the new routing table at router A is-

Destination	Distance	Next Hop
A	0	A
B	2	B
C	5	B
D	1	D

At Router B-

Router B receives distance vectors from its neighbours A, C and D.

- Router B prepares a new routing table as-



Destination	Distance	Next hop
A		
B	0	B
C		
D		

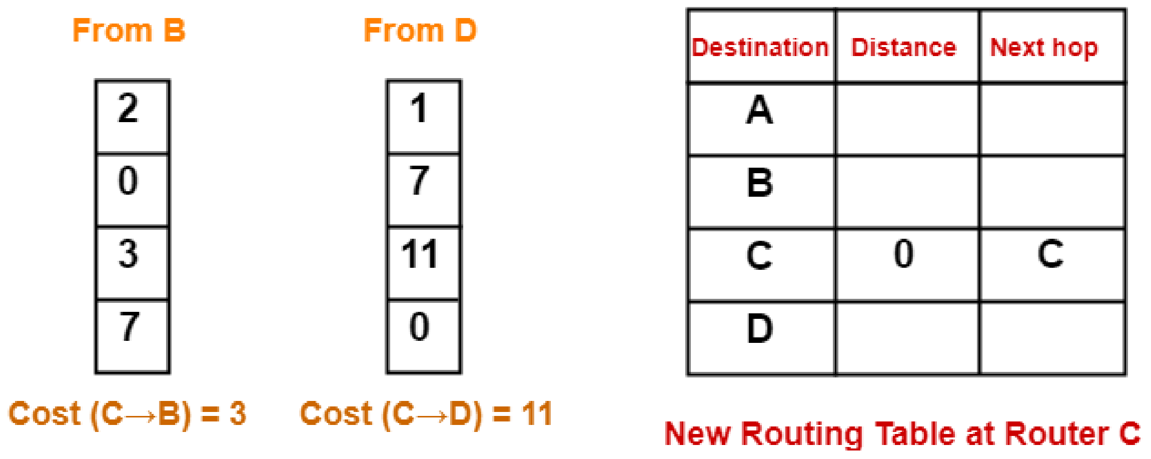
- Cost of reaching destination A from router B = min { 2+0 , 3+∞ , 7+1 } = 2 via A.
- Cost of reaching destination C from router B = min { 2+∞ , 3+0 , 7+11 } = 3 via C.
- Cost of reaching destination D from router B = min { 2+1 , 3+11 , 7+0 } = 3 via A.

Thus, the new routing table at router B is-

Destination	Distance	Next Hop
A	2	A
B	0	B
C	3	C
D	3	A

At Router C-

- Router C receives distance vectors from its neighbors B and D.
- Router C prepares a new routing table as-



From B	From D
2	1
0	7
3	11
7	0
Cost (C→B) = 3	Cost (C→D) = 11

Destination	Distance	Next hop
A		
B		
C	0	C
D		

New Routing Table at Router C

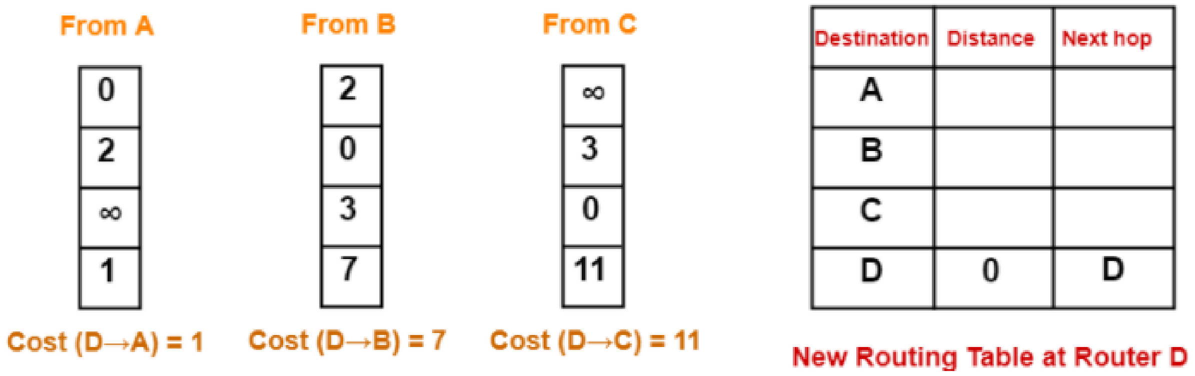
- Cost of reaching destination A from router C = min { 3+2 , 11+1 } = 5 via B.
- Cost of reaching destination B from router C = min { 3+0 , 11+7 } = 3 via B.
- Cost of reaching destination D from router C = min { 3+7 , 11+0 } = 10 via B.

Thus, the new routing table at router C is-

Destination	Distance	Next Hop
A	5	B
B	3	B
C	0	C
D	10	B

At Router D-

- Router D receives distance vectors from its neighbors A, B and C.
- Router D prepares a new routing table as-



From A	From B	From C
0	2	∞
2	0	3
∞	3	0
1	7	11
Cost (D→A) = 1	Cost (D→B) = 7	Cost (D→C) = 11

Destination	Distance	Next hop
A		
B		
C		
D	0	D

New Routing Table at Router D

- Cost of reaching destination A from router D = min { 1+0 , 7+2 , 11+∞ } = 1 via A.
- Cost of reaching destination B from router D = min { 1+2 , 7+0 , 11+3 } = 3 via A.
- Cost of reaching destination C from router D = min { 1+∞ , 7+3 , 11+0 } = 10 via B.

Thus, the new routing table at router D is-

Destination	Distance	Next Hop
A	1	A
B	3	A
C	10	B
D	0	D

Step-03:

- Each router exchanges its distance vector obtained in Step-02 with its neighboring routers.
- After exchanging the distance vectors, each router prepares a new routing table.

At Router A-

- Router A receives distance vectors from its neighbours B and D.
- Router A prepares a new routing table as-
  - Cost of reaching destination B from router A = min { 2+0 , 1+3 } = 2 via B.
  - Cost of reaching destination C from router A = min { 2+3 , 1+10 } = 5 via B.
  - Cost of reaching destination D from router A = min { 2+3 , 1+0 } = 1 via D.

Thus, the new routing table at router A is-

Destination	Distance	Next Hop
A	0	A
B	2	B
C	5	B
D	1	D

At Router B-

- Router B receives distance vectors from its neighbours A, C and D.
- Router B prepares a new routing table as-
  - Cost of reaching destination A from router B = min { 2+0 , 3+5 , 3+1 } = 2 via A.
  - Cost of reaching destination C from router B = min { 2+5 , 3+0 , 3+10 } = 3 via C.
  - Cost of reaching destination D from router B = min { 2+1 , 3+10 , 3+0 } = 3 via A.
  - Thus, the new routing table at router B is-

Destination	Distance	Next Hop
A	2	A
B	0	B
C	3	C
D	3	A

At Router C-

- Router C receives distance vectors from its neighbors B and D.
- Router C prepares a new routing table as-
    - Cost of reaching destination A from router C = min { 3+2 , 10+1 } = 5 via B.
    - Cost of reaching destination B from router C = min { 3+0 , 10+3 } = 3 via B.
    - Cost of reaching destination D from router C = min { 3+3 , 10+0 } = 6 via B.

Thus, the new routing table at router C is-

Destination	Distance	Next Hop
A	5	B
B	3	B
C	0	C
D	6	B

At Router D-

- Router D receives distance vectors from its neighbours A, B and C.
- Router D prepares a new routing table as-
    - Cost of reaching destination A from router D = min { 1+0 , 3+2 , 10+5 } = 1 via A.
    - Cost of reaching destination B from router D = min { 1+2 , 3+0 , 10+3 } = 3 via A.
    - Cost of reaching destination C from router D = min { 1+5 , 3+3 , 10+0 } = 6 via A.

Thus, the new routing table at router D is-

Destination	Distance	Next Hop
A	1	A
B	3	A
C	6	A
D	0	D

These will be the final routing tables at each router.

Link State Routing

- Link state routing is a technique in which each router shares the knowledge of its neighborhood with every other router in the internetwork.
- Link State Routing Link state routing is based on the assumption that, although the global knowledge about the topology is not clear, each node has partial knowledge: it knows the state (type, condition, and cost) of its links. In other words, the whole topology can be compiled from the partial knowledge of each node

The three keys to understand the Link State Routing algorithm:

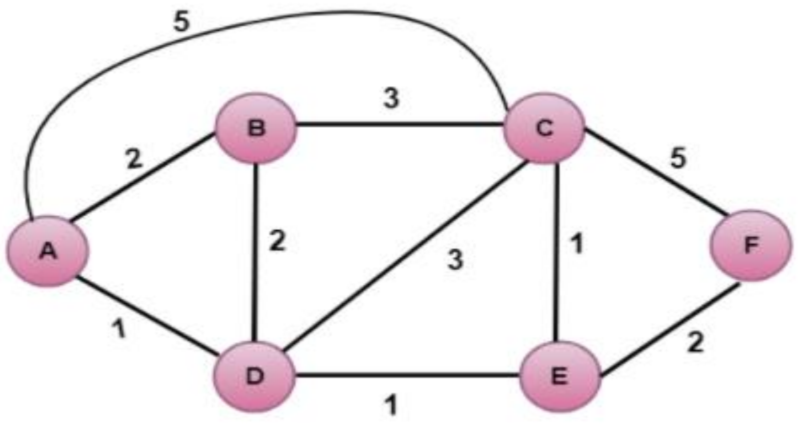
- Knowledge about the neighborhood: Instead of sending its routing table, a router sends the information about its neighborhood only. A router broadcast its identities and cost of the directly attached links to other routers.
- Flooding: Each router sends the information to every other router on the internetwork except its neighbors. This process is known as Flooding. Every router that receives the packet sends the copies to all its neighbors. Finally, each and every router receives a copy of the same information.
- Information sharing: A router sends the information to every other router only when the change occurs in the information.

Route Calculation

- Each node uses Dijkstra's algorithm on the graph to calculate the optimal routes to all nodes.
- The Link state routing algorithm is also known as Dijkstra's algorithm which is used to find the shortest path from one node to every other node in the network.
- The Dijkstra's algorithm is an iterative, and it has the property that after k^{th} iteration of the algorithm, the least cost paths are well known for k destination nodes.
- Let's describe some notations:
- c(i , j): Link cost from node i to node j. If i and j nodes are not directly linked, then c(i , j) = ∞.
- D(v): It defines the cost of the path from source code to destination v that has the least cost currently.
- P(v): It defines the previous node (neighbor of v) along with current least cost path from source to v.
- N: It is the total number of nodes available in the network.

  D(v) = min(D(v) , D(w) + c(w,v))

Example



- In the above figure, source vertex is A.

Step 1:

The first step is an initialization step. The currently known least cost path from A to its directly attached neighbours, B, C, D are 2,5,1 respectively. The cost from A to B is set to 2, from A to D is set to 1 and from A to C is set to 5. The cost from A to E and F are set to infinity as they are not directly linked to A.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞

Step 2:

In the above table, we observe that vertex D contains the least cost path in step 1. Therefore, it is added in N. Now, we need to determine a least-cost path through D vertex.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞

Step 3:

In the above table, we observe that both E and B have the least cost path in step 2. Let's consider the E vertex. Now, we determine the least cost path of remaining vertices through E.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E

Step 4:

In the above table, we observe that B vertex has the least cost path in step 3. Therefore, it is added in N. Now, we determine the least cost path of remaining vertices through B.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E

Step 5:

In the above table, we observe that C vertex has the least cost path in step 4. Therefore, it is added in N. Now, we determine the least cost path of remaining vertices through C.

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E
5	ADEBC					4,E

Final table

Step	N	D(B),P(B)	D(C),P(C)	D(D),P(D)	D(E),P(E)	D(F),P(F)
1	A	2,A	5,A	1,A	∞	∞
2	AD	2,A	4,D		2,D	∞
3	ADE	2,A	3,E			4,E
4	ADEB		3,E			4,E
5	ADEBC					4,E
6	ADEBCF					

Congestion, in the context of networks, refers to a network state where a node or link carries so much data that it may deteriorate network service quality, resulting in queuing delay, frame or data packet loss and the blocking of new connections. In a congested network, response time slows with reduced network throughput. Congestion occurs when bandwidth is insufficient and network data traffic exceeds capacity.

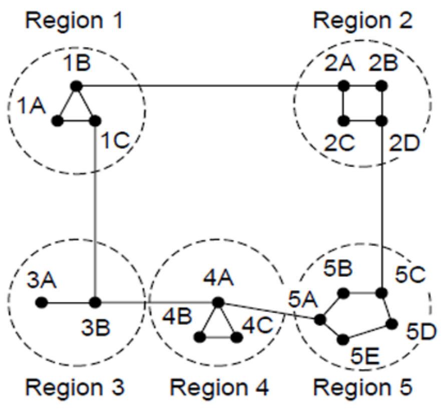
Hierarchical Routing:

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.

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.

For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run out of names for aggregations.

Below Fig. (a) 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. (b). When routing is done hierarchically, as in Fig. (c), there are entries for all the local routers as before, 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. As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase.



(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)

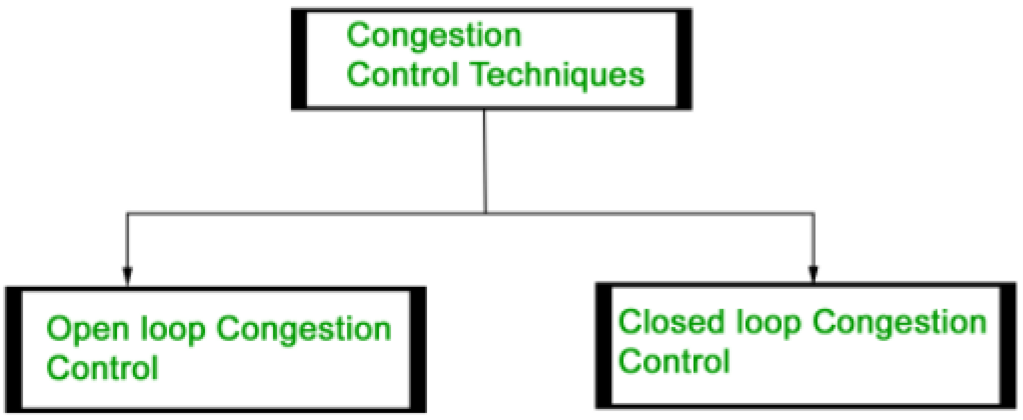
Hierarchical table for 1A

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

(c)

Congestion Control techniques in Computer Networks

Congestion control refers to the techniques used to control or prevent congestion. Congestion control techniques can be broadly classified into two categories:


Congestion Control Techniques
Open loop Congestion Control
Closed loop Congestion Control

Open Loop Congestion Control

- Open loop congestion control policies are applied to prevent congestion before it happens. The congestion control is handled either by the source or the destination.

Policies adopted by open loop congestion control –

1. **Retransmission Policy :**
   It is the policy in which retransmission of the packets are taken care. If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. This transmission may increase the congestion in the network.
   To prevent congestion, retransmission timers must be designed to prevent congestion and also able to optimize efficiency.
2. **Window Policy :**
   The type of window at the sender side may also affect the congestion. Several packets in the Go-back-n window are resent, although some packets may be received successfully at the receiver side. This duplication may increase the congestion in the network and making it worse.
   Therefore, Selective repeat window should be adopted as it sends the specific packet that may have been lost.
3. **Discarding Policy :**
   A good discarding policy adopted by the routers is that the routers may prevent congestion and at the same time partially discards the corrupted or less sensitive package and also able to maintain the quality of a message.
   In case of audio file transmission, routers can discard less sensitive packets to prevent congestion and also maintain the quality of the audio file.
4. **Acknowledgment Policy :**
   Since acknowledgement are also the part of the load in network, the acknowledgment policy imposed by the receiver may also affect congestion. Several approaches can be used to prevent congestion related to acknowledgment.

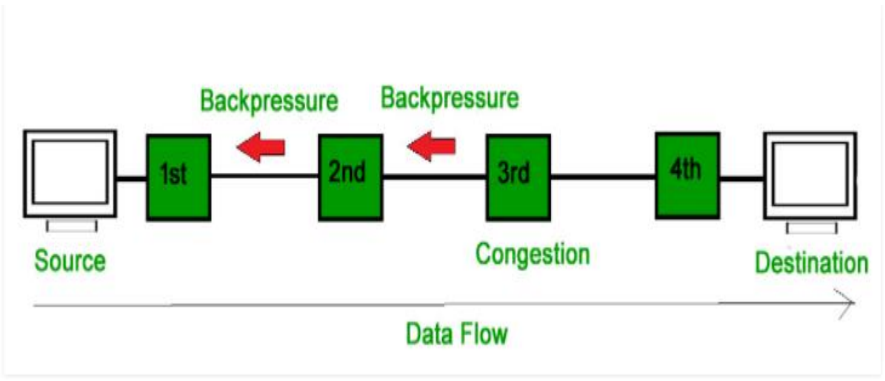
The receiver should send acknowledgement for N packets rather than sending acknowledgement for a single packet. The receiver should send a acknowledgment only if it has to sent a packet or a timer expires.

5. **Admission Policy :**
   In admission policy a mechanism should be used to prevent congestion. Switches in a flow should first check the resource requirement of a network flow before transmitting it further. If there is a chance of a congestion or there is a congestion in the network, router should deny establishing a virtual network connection to prevent further congestion.

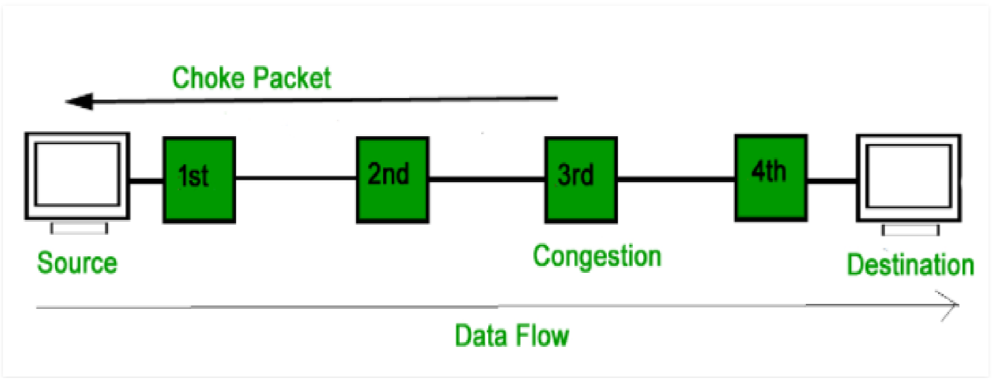
Closed Loop Congestion Control

- Closed loop congestion control technique is used to treat or alleviate congestion after it happens. Several techniques are used by different protocols; some of them are:

  1. **Backpressure:**
     Backpressure is a technique in which a congested node stops receiving packet from upstream node. This may cause the upstream node or nodes to become congested and rejects receiving data from above nodes. Backpressure is a node-to-node congestion control technique that propagates in the opposite direction of data flow. The backpressure technique can be applied only to virtual circuit where each node has information of its above upstream node.



  2. **Choke Packet Technique :**
     Choke packet technique is applicable to both virtual networks as well as datagram subnets. A choke packet is a packet sent by a node to the source to inform it of congestion. Each router monitor its resources and the utilization at each of its output lines. whenever the resource utilization exceeds the threshold value which is set by the administrator, the router directly sends a choke packet to the source giving it a feedback to reduce the traffic. The intermediate nodes through which the packets has traveled are not warned about congestion.



3. **Implicit Signaling:**
   In implicit signaling, there is no communication between the congested nodes and the source. The source guesses that there is congestion in a network. For example when sender sends several packets and there is no acknowledgment for a while, one assumption is that there is congestion.

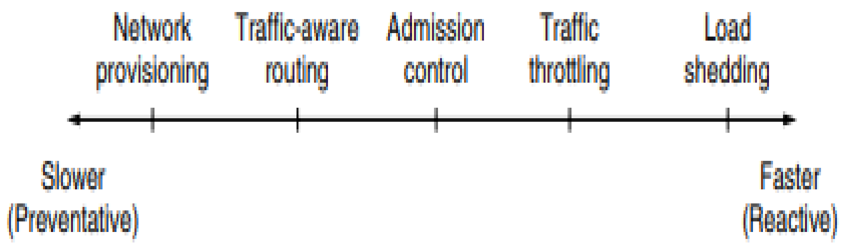
4. **Explicit Signaling :**
   In explicit signaling, if a node experiences congestion it can explicitly sends a packet to the source or destination to inform about congestion. The difference between choke packet and explicit signaling is that the signal is included in the packets that carry data rather than creating different packet as in case of choke packet technique.
   Explicit signaling can occur in either forward or backward direction.

   a. **Forward Signaling :** In forward signaling signal is sent in the direction of the congestion. The destination is warned about congestion. The receiver in this case adopt policies to prevent further congestion.

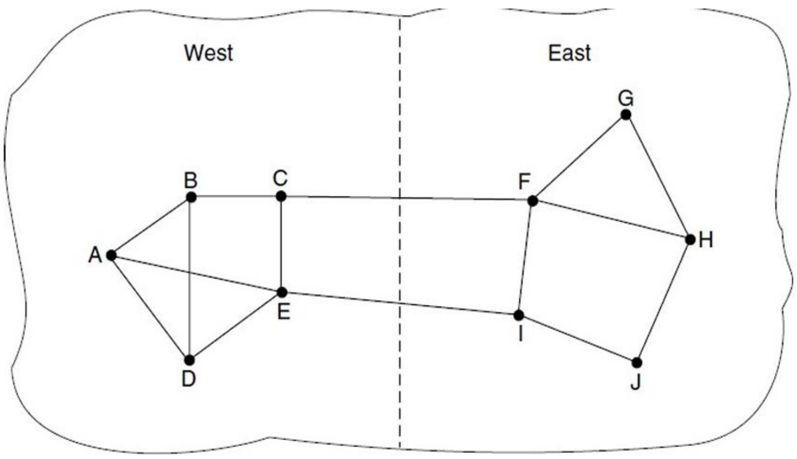
   b. **Backward Signaling :** In backward signaling signal is sent in the opposite direction of the congestion. The source is warned about congestion and it needs to slow down.

Approaches to congestion control

- The presence of congestion means that the load is (temporarily) greater than the resources (in a part of the network) can handle. Two solutions come to mind: increase the resources or decrease the load. As shown in Fig. 5-22, these solutions are usually applied on different time scales to either prevent congestion or react to it once it has occurred.



1. **Network provisioning**
    - The most basic way to avoid congestion is to build a network that is well matched to the traffic that it carries.
    - If there is a low-bandwidth link on the path along which most traffic is directed, congestion is likely.
    - Sometimes resources can be added dynamically when there is serious congestion, for example, turning on spare routers or enabling lines that are normally used only as backups (to make the system fault tolerant) or purchasing bandwidth on the open market.
    - More often, links and routers that are regularly heavily utilized are upgraded at the earliest opportunity. This is called provisioning and happens on a time scale of months, driven by long-term traffic trends.
2. **Traffic -aware routing**
    - TRAFFIC AWARE ROUTING These schemes adapted to changes in topology, but not to changes in load. The goal in taking load into account when computing routes is to shift traffic away from hotspots that will be the first places in the network to experience congestion.
    - The most direct way to do this is to set the link weight to be a function of the (fixed) link bandwidth and propagation delay plus the (variable) measured load or average queuing delay. Least-weight paths will then favor paths that are more lightly loaded, all else being equal.
    - Consider the network of Fig. 5-23, which is divided into two parts, East and West, connected by two links, CF and EI. Suppose that most of the traffic between East and West is using link CF, and, as a result, this link is heavily loaded with long delays. Including queueing delay in the weight used for the shortest path calculation will make EI more attractive. After the new routing tables have been installed, most of the East-West traffic will now go over EI, loading this link. Consequently, in the next update, CF will appear to be the shortest path. As a result, the routing tables may oscillate wildly, leading to erratic routing and many potential problems.



If load is ignored and only bandwidth and propagation delay are considered, this problem does not occur. Attempts to include load but change weights within a narrow range only slow down routing oscillations. Two techniques can contribute to a successful solution. The first is multipath routing, in which there can be multiple paths from a source to a destination. In our example this means that the traffic can be spread across both of the East to West links. The second one is for the routing scheme to shift traffic across routes slowly enough that it is able to converge.

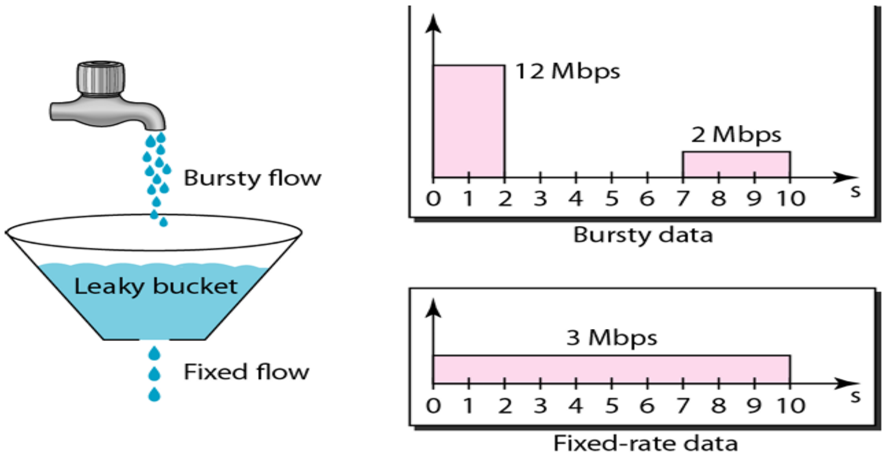
3. **Admission control.**
   - It is a mechanism should be used to prevent congestion switches in a flow should first check the resource requirement of a network flow before transmitting it further
   - If there is a chance of congestion or there is a congestion in network router should deny establishing a virtual network connection to prevent further congestion.
   - However, sometimes it is not possible to increase capacity. The only way then to beat back the congestion is to decrease the load. In a virtual-circuit network, new connections can be refused if they would cause the network to become congested. This is called admission control.
4. **Traffic throttling**
   - In the Internet and many other computer networks, senders adjust their transmissions to send as much traffic as the network can readily deliver.
   - In this setting, the network aims to operate just before the onset of congestion. When congestion is imminent, it must tell the senders to throttle back their transmissions and slow down.
   - This feedback is business as usual rather than an exceptional situation. The term congestion avoidance is sometimes used to contrast this operating point with the one in which the network has become (overly) congested
5. **Load Shedding**
   - When buffers become full, routers simply discard packets.
   - Which packet is chosen to be the victim depends on the application and on the error strategy used in the data link layer.
   - For a file transfer, for, e.g. cannot discard older packets since this will cause a gap in the received data.
   - For real-time voice or video it is probably better to throw away old data and keep new packets.
   - Get the application to mark packets with discard priority.

Traffic Control Algorithm

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

1.Leaky Bucket

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



In the figure, we assume that the network has committed a bandwidth of 3 Mbps for a host. The use of the leaky bucket shapes the input traffic to make it conform to this commitment. In Figure 24.19 the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data. The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data. In all, the host has sent 30 Mbits of data in l0s. The leaky bucket smooth's the traffic by sending out data at a rate of 3 Mbps during the same 10 s. Without the leaky bucket, the beginning burst may have hurt the network by consuming more bandwidth than is set aside for this host. We can also see that the leaky bucket may prevent congestion.

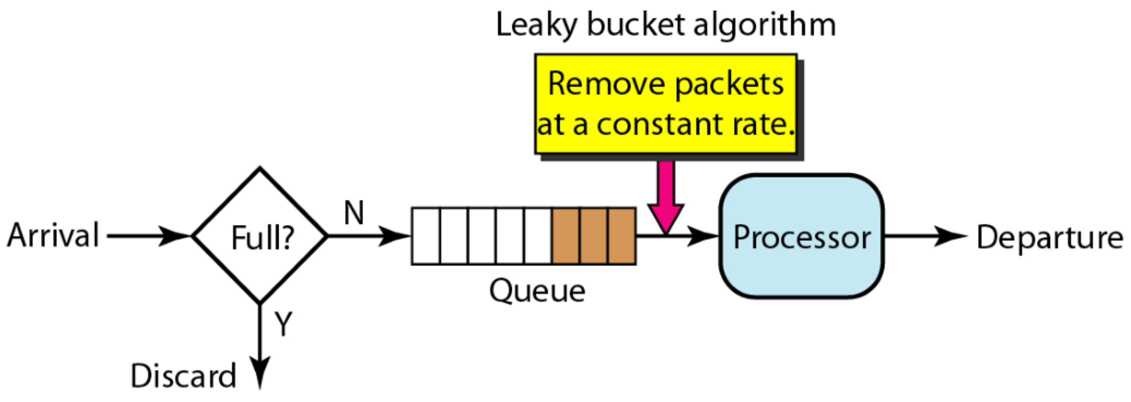
A simple leaky bucket implementation is shown in below Figure. A FIFO queue holds the packets. If the traffic consists of fixed-size packets, the process removes a fixed number of packets from the queue at each tick of the clock. If the traffic consists of variable-length packets, the fixed output rate must be based on the number of bytes or bits.

The following is an algorithm for variable-length packets:

1. Initialize a counter to n at the tick of the clock.

2. If n is greater than the size of the packet, send the packet and decrement the counter by the packet size. Repeat this step until n is smaller than the packet size.

3. Reset the counter and go to step 1.

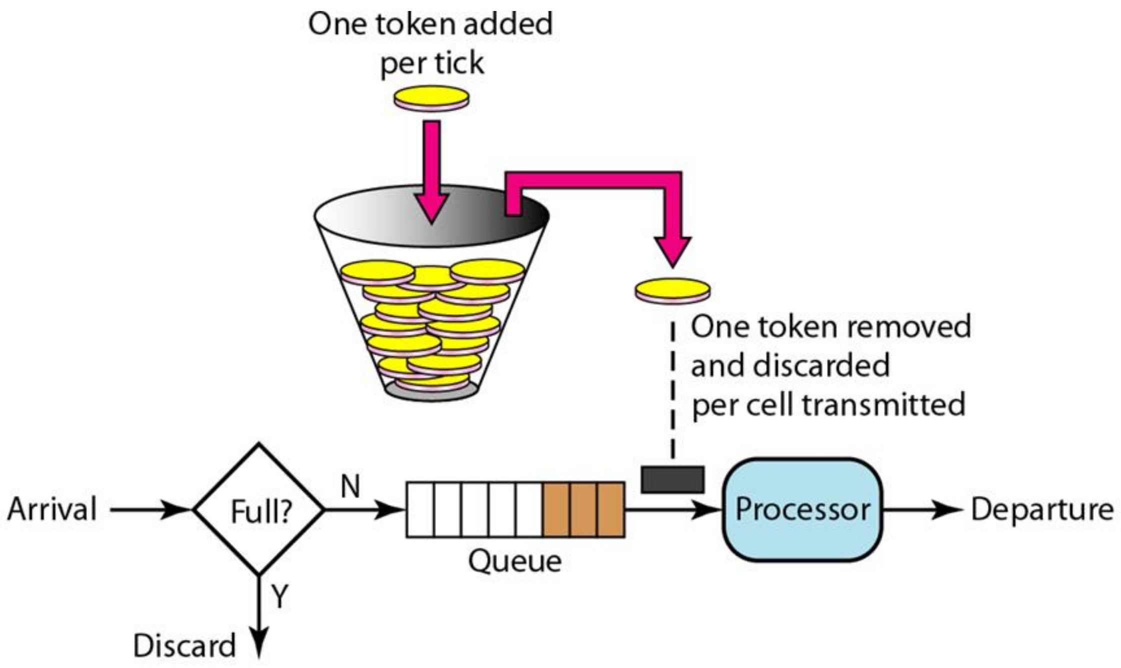


-

2.Token Bucket

The leaky bucket is very restrictive. It does not credit an idle host. For example, if a host is not sending for a while, its bucket becomes empty. Now if the host has bursty data, the leaky bucket allows only an average rate. The time when the host was idle is not taken into account. On the other hand, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens. For each tick of the clock, the system sends n tokens to the bucket. The system removes one token for every cell (or byte) of data sent. For example, if n is 100 and the host is idle for 100 ticks, the bucket collects 10,000 tokens. Now the host can consume all these tokens in one tick with 10,000 cells, or the host takes 1000 ticks with 10 cells per tick. In other words, the host can send bursty data as long as the bucket is not empty.

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



Tunnelling

- 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.

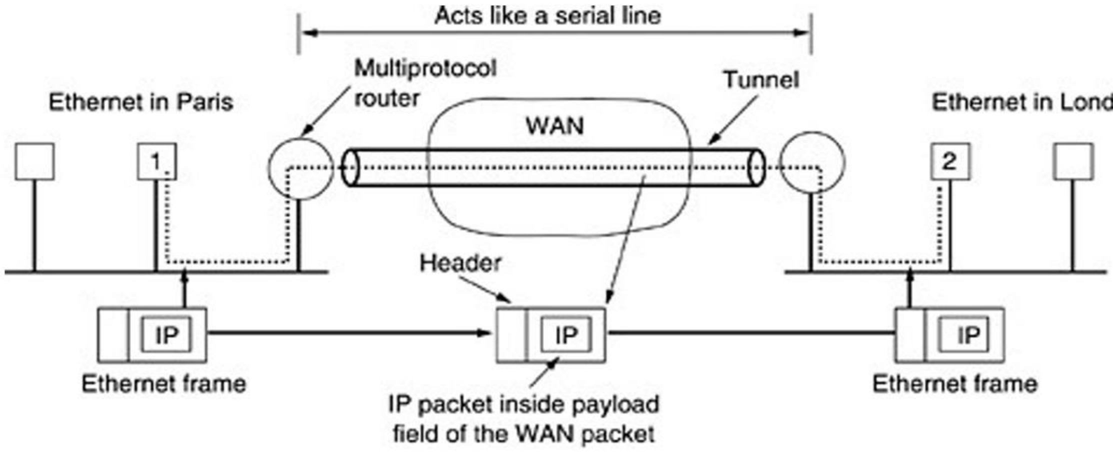


Fig:Tunneling

- 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. 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.

The Network Layer in The Internet

Top 10 principles

1. **Make sure it works**. Do not finalize the design or standard until multiple prototypes have successfully communicated with each other. All too often designers first write a 1000-page standard, get it approved, then discover it is deeply flawed and does not work. Then they write version 1.1 of the standard. This is not the way to go.
2. **Keep it simple**. When in doubt, use the simplest solution. William of Occam stated this principle (Occam's razor) in the 14th century. Put in modern terms: fight features. If a feature is not absolutely essential, leave it out, especially if the same effect can be achieved by combining other features.
3. **Make clear choices.** If there are several ways of doing the same thing, choose one. Having two or more ways to do the same thing is looking for trouble. Standards often have multiple options or modes or parameters because several powerful parties insist that their way is best. Designers should strongly resist this tendency. Just say no.
4. **Exploit modularity**. This principle leads directly to the idea of having protocol stacks, each of whose layers is independent of all the other ones. In this way, if circumstances that require one module or layer to be changed, the other ones will not be affected.
5. **Expect heterogeneity.** Different types of hardware, transmission facilities, and applications will occur on any large network. To handle them, the network design must be simple, general, and flexible.

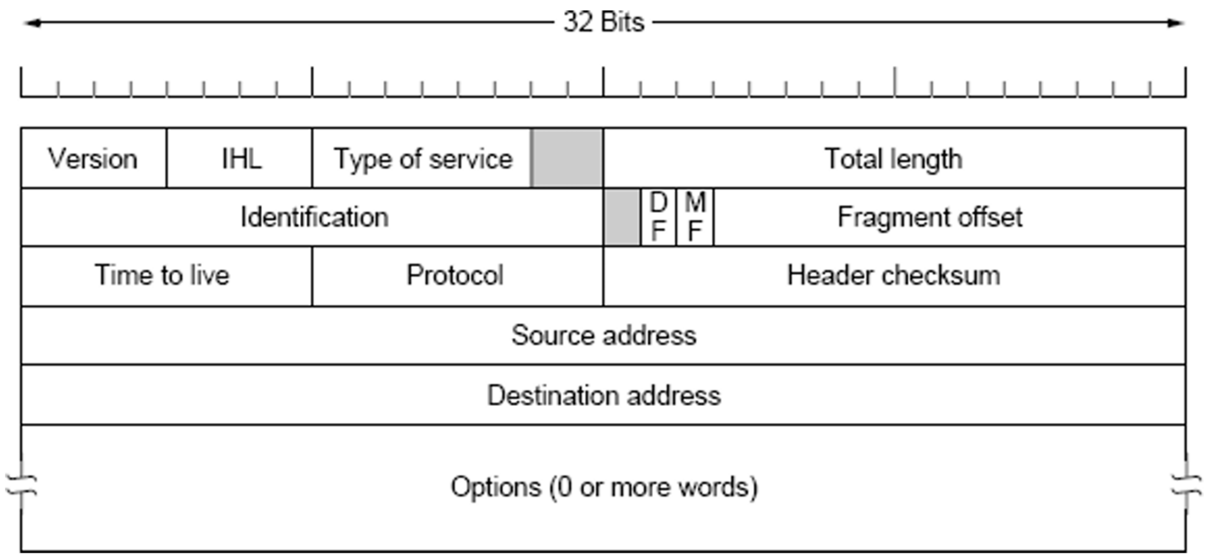
6. **Avoid static options and parameters**. If parameters are unavoidable (e.g., maximum packet size), it is best to have the sender and receiver negotiate a value than defining fixed choices.
7. **Look for a good design; it need not be perfect**. Often the designers have a good design but it cannot handle some weird special case. Rather than messing up the design, the designers should go with the good design and put the burden of working around it on the people with the strange requirements.
8. **Be strict when sending and tolerant when receiving**. In other words, only send packets that rigorously comply with the standards, but expect incoming packets that may not be fully conformant and try to deal with them.
9. **Think about scalability**. If the system is to handle millions of hosts and billions of users effectively, no centralized databases of any kind are tolerable and load must be spread as evenly as possible over the available resources.
10. **Consider performance and cost**. If a network has poor performance or outrageous costs, nobody will use it.

The IP Protocol

IPV4 stands for Internet Protocol version v4 ,is the most widely used system for identifying devices on a network. An IPv4 address consists of 32 bit (binary digit), grouped into four section of known as octets or bytes. Each octet has 8 bits and this bits can be represented only in 0 or 1 form, and when they grouped together, they form a binary number. Since each octet has 8 bits, it can represent 256 numbers ranging from o to 255. These four octets are represented as decimal numbers, separated by periods known as dotted decimal notation. For example IPv4 address 185.107.80.231 consists of four octets.

IPv4 Datagram Header

- 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.
- These fields together ensure the proper delivery, routing, fragmentation, and reassembly of IP packets across the network.

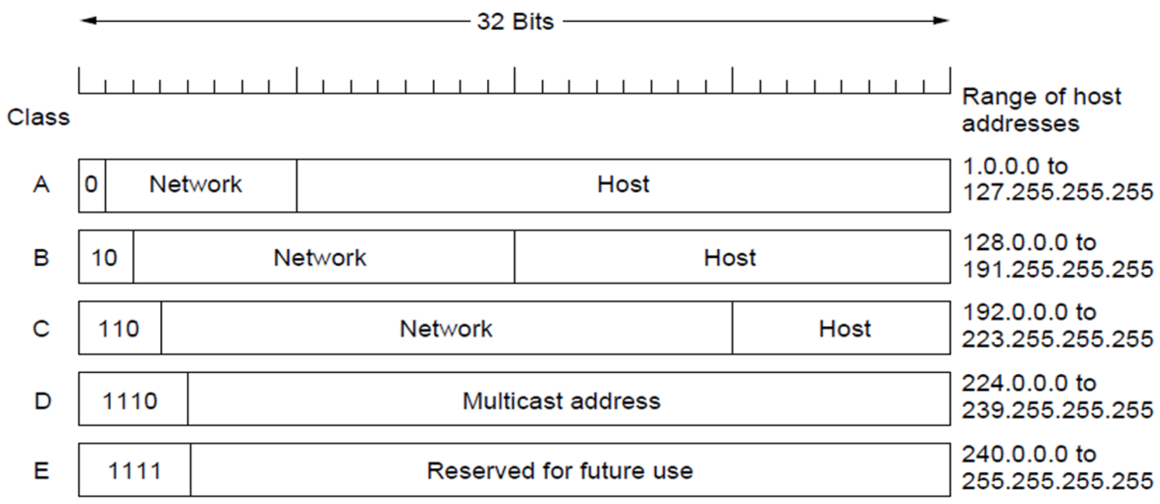


- **Version**- The Version field keeps track of which version of the protocol the datagram belongs to.

- **HLEN:** IP header length (4 bits), which is the number of 32 bit words in the header. The minimum value for this field is 5 and the maximum is 15.
- **Type of service** -The Type of service field is one of the few fields that have changed its meaning (slightly) over the years. It was and is still intended to distinguish between different classes of service-Low Delay, High Throughput, Reliability (8 bits)
- **Total length**-It includes everything in the datagram—both header and data. The maximum length is 65,535 bytes.
- **Identification-**this 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**.
- **Time to live field** -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.
- **Protocol-**this field tells it which transport process to give it to. TCP is one possibility, but so are UDP and some others. The numbering of protocols is global across the entire Internet.
- The **Header checksum** verifies the header only.
- 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.

IP Address

- IP addresses were divided into the five categories listed in Fig. This allocation has come to be called classful addressing. It is no longer used, but references to it in the literature are still common.


32 Bits
Class
Range of host addresses
A
0
Network
Host
1.0.0.0 to 127.255.255.255
B
10
Network
Host
128.0.0.0 to 191.255.255.255
C
110
Network
Host
192.0.0.0 to 223.255.255.255
D
1110
Multicast address
224.0.0.0 to 239.255.255.255
E
1111
Reserved for future use
240.0.0.0 to 255.255.255.255

- The class A, B, C, and D formats allow for up to 128 networks with 16 million hosts each, 16,384 networks with up to 64K hosts, and 2 million networks (e.g., LANs) with up to 256 hosts each (although a few of these are special). Also supported is multicast, in which a datagram is directed to multiple hosts. Addresses beginning with 1111 are reserved for future use.
- Network addresses, which are 32-bit numbers, are usually written in dotted decimal notation. In this format, each of the 4 bytes is written in decimal, from 0 to 255. For example, the 32-bit hexadecimal address C0290614 is written as 192.41.6.20. The lowest IP address is 0.0.0.0 and the highest is 255.255.255.255.

CIDR—Classless Inter Domain Routing:

- The basic idea behind CIDR, which is described in RFC 1519, is to allocate the remaining IP addresses in variable-sized blocks, without regard to the classes. If a site needs, say, 2000 addresses, it is given a block of 2048 addresses on a 2048-byte boundary.
- Using CIDR, each IP address has a network prefix that identifies either one or several network gateways. The length of the network prefix in IPv4 CIDR is also specified as part of the IP address and varies depending on the number of bits needed, rather than any arbitrary class assignment structure. A destination IP address or route that describes many possible destinations has a shorter prefix and is said to be less specific. A longer prefix describes a destination gateway more specifically.

CIDR notation: $\boxed{\mathbf{a.b.c.d/n}}$

Where n = number of network IDs

Host ID = 32-n

IP = $2^{(32-n)}$

Example: **20.10.20.100/20**

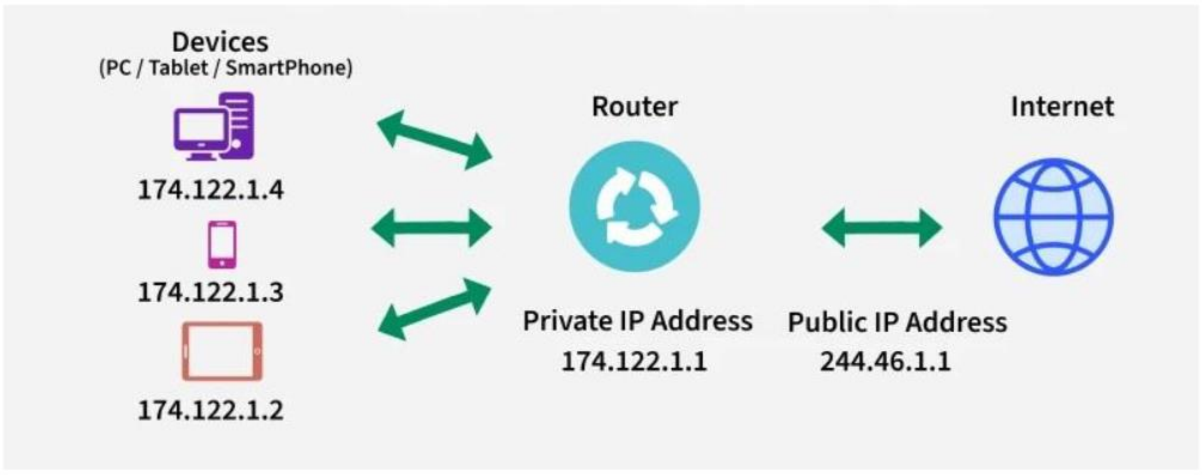
Network ID = 20.

Host ID = 32-20 = 12.

IP = $2^{(32-20)}$ = 4096.

NAT—Network Address Translation:

- IP addresses are scarce. An ISP might have a /16 (formerly class B) address, giving it 65,534 host numbers. If it has more customers than that, it has a problem. This quick fix came in the form of NAT (Network Address Translation), which is described in RFC 3022 and which we will summarize below.
- **Network Address Translation (NAT)** is a process in which a router or firewall **converts private IP addresses** used inside a local network into a **public IP address** (and back) so devices can communicate with the Internet.



Network Address Translation

Why NAT is needed

- IPv4 addresses are limited: There aren't enough public IPv4 addresses for every device.
- Companies and homes usually use private IP ranges (for example 192.168.x.x, 10.x.x.x), which are not routable on the Internet.
- NAT allows many devices inside a private network to share a single public IP address to access the Internet.

Types of NAT

Static NAT (One-to-One NAT)

- **Each private IP** is mapped to a **specific public IP**.
- The mapping **does not change**.
- Used when a device (like a web server) needs to be accessible from outside.

Dynamic NAT (Many-to-Many)

- Private IPs are mapped to **any available public IP** from a **pool**.
- Mapping changes dynamically (not fixed).

PAT (Port Address Translation)

- Also called NAT Overloading or Many-to-One NAT.
- Multiple private IPs share one public IP.
- NAT differentiates connections using port numbers.

IP Version 6

- Internet Protocol version 6 is a new addressing protocol designed to incorporate all the possible requirements of future Internet. This protocol as its predecessor IPv4, works on the Network Layer (Layer-3). Along with its offering of an enormous amount of logical address space, this protocol has ample features to address the shortcoming of IPv4. The major goals of IPV6 are:
    1. Support billions of hosts, even with inefficient address allocation.
    2. Reduce the size of the routing tables.
    3. Simplify the protocol, to allow routers to process packets faster.
    4. Provide better security (authentication and privacy).
    5. Pay more attention to the type of service, particularly for real-time data.
    6. Aid multicasting by allowing scopes to be specified.
    7. Make it possible for a host to roam without changing its address.
    8. Allow the protocol to evolve in the future.
    9. Permit the old and new protocols to coexist for years.
- First and foremost, IPv6 has longer addresses than IPv4. They are 128 bits long, which solves the problem that IPv6 set out to solve: providing an effectively unlimited supply of Internet addresses.
- The second major improvement of IPv6 is the simplification of the header. It contains only seven fields (versus 13 in IPv4). This change allows routers to process packets faster and thus improves throughput and delay.
- The third major improvement is better support for options. This change was essential with the new header because fields that previously were required are now optional (because they are not used so often). In addition, the way options are represented is different, making

it simple for routers to skip over options not intended for them. This feature speeds up packet processing time.

- A fourth area in which IPv6 represents a big advance is in security. Authentication and privacy are key features of the new IP.

The Main IPv6 Header

The IPv6 header is shown in Fig. 5-56. The Version field is always 6 for IPv6 (and 4 for IPv4). During the transition period from IPv4, which has already taken more than a decade, routers will be able to examine this field to tell what kind of packet they have.

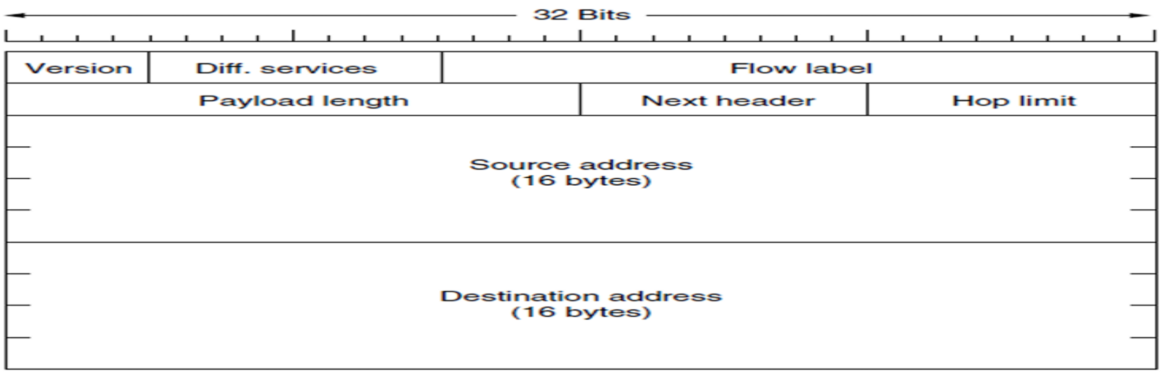


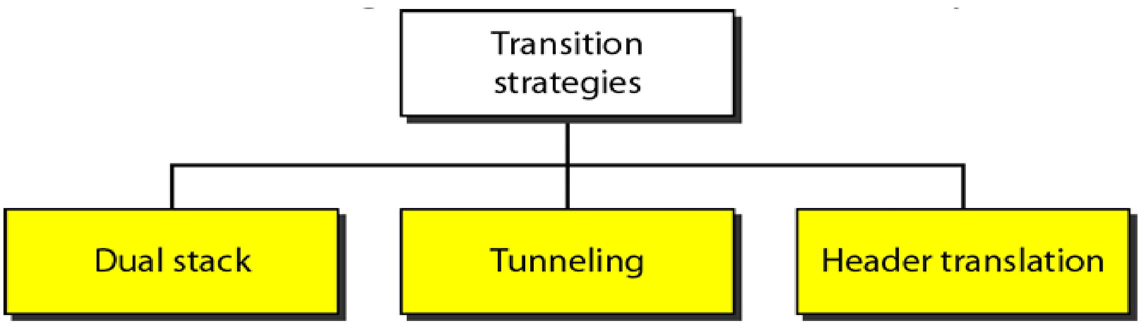
Fig :The IPV6 fixed header

- The **Differentiated services** field (originally called Traffic class) is used to distinguish the class of service for packets with different real-time delivery requirements. It is used with the differentiated service architecture for quality of service in the same manner as the field of the same name in the IPv4 packet.
- The **Flow label** field provides a way for a source and destination to mark groups of packets that have the same requirements and should be treated in the same way by the network, forming a pseudo connection. The flow can be set up in advance and given an identifier.
- The **Payload length** field tells how many bytes follow the 40-byte header of Fig. 5-56. The name was changed from the IPv4 Total length field because the meaning was changed slightly: the 40 header bytes are no longer counted as part of the length (as they used to be). This change means the payload can now be 65,535 bytes instead of a mere 65,515 bytes.
- The **Next header** field lets the cat out of the bag. The reason the header could be simplified is that there can be additional (optional) extension headers. This field tells which of the (currently) six extension headers, if any, follow this one.
- The **Hop limit** field is used to keep packets from living forever. It is, in practice, the same as the Time to live field in IPv4, namely, a field that is decremented on each hop.
- Next the **Source address and Destination address** fields which contains 1228 bit address. A new notation has been devised for writing 16-byte addresses. They are written as eight groups of four hexadecimal digits with colons between the groups, like this: 8000:0000:0000:0000:0123:4567:89AB:CDEF
- Since many addresses will have many zeros inside them, three optimizations have been authorized. First, leading zeros within a group can be omitted, so 0123 can be written as 123. Second, one or more groups of 16 zero bits can be replaced by a pair of colons. Thus, the above address now becomes

8000::123:4567:89AB:CDEF

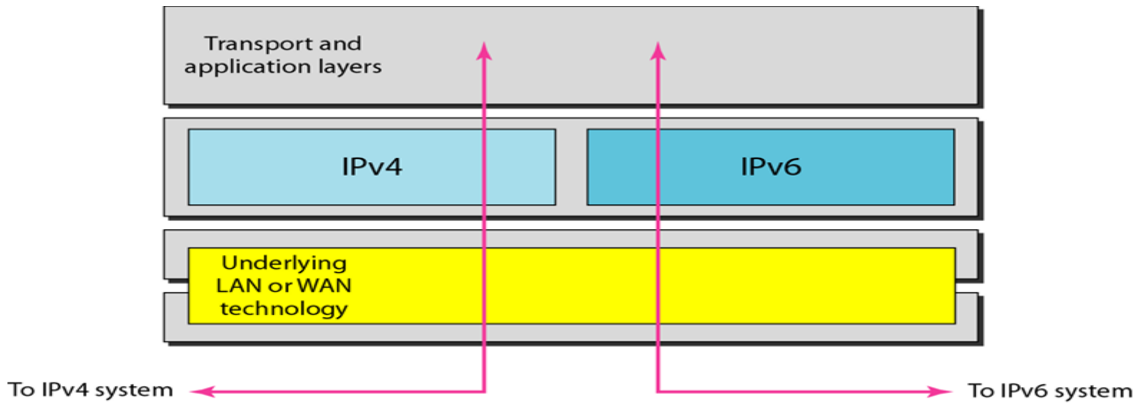
Transition from IPV4 to IPV6:

Because of the huge number of systems on the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. It takes a considerable amount of time before every system in the Internet can move from IPv4 to IPv6. The transition must be smooth to prevent any problems between IPv4 and IPv6 systems. Three strategies have been devised by the IETF to help the transition.



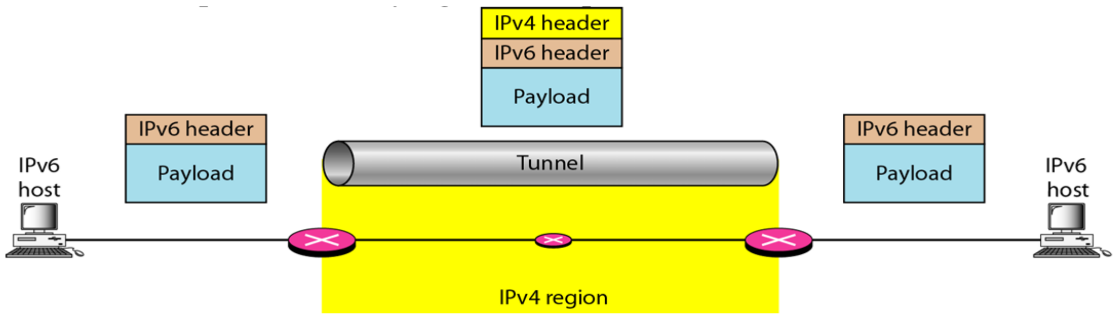
1. **Dual Stack**

- It is recommended that all hosts, before migrating completely to version 6, have a dual stack of protocols. In other words, a station must run IPv4 and IPv6 simultaneously until all the Internet uses IPv6. See below Figure for the layout of a dual-stack configuration. To determine which version to use when sending a packet to a destination, the source host queries the DNS. If the DNS returns an IPv4 address, the source host sends an IPv4 packet. If the DNS returns an IPv6 address, the source host sends an IPv6 packet.


Transport and application layers
IPv4
IPv6
Underlying LAN or WAN technology
To IPv4 system
To IPv6 system

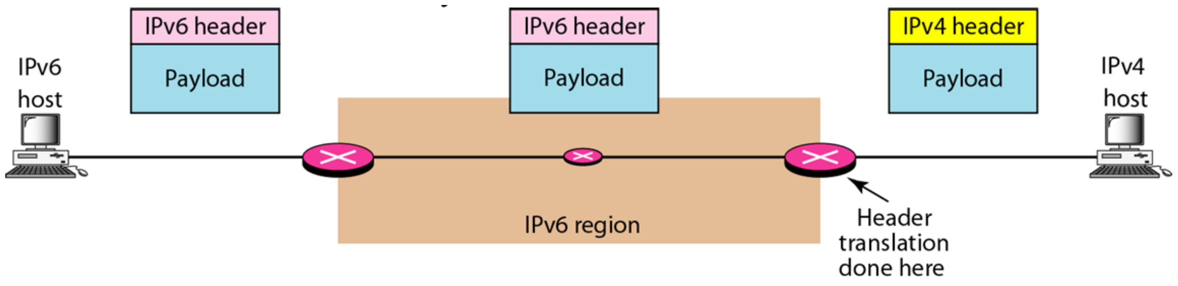
2. **Tunneling**

   Tunneling is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4. To pass through this region, the packet must have an IPv4 address. So the IPv6 packet is encapsulated in an IPv4 packet when it enters the region, and it leaves its capsule when it exits the region. It seems as if the IPv6 packet goes through a tunnel at one end and emerges at the other end. To make it clear that the IPv4 packet is carrying an IPv6 packet as data.



3. **Header Translation**

   Header translation is necessary when the majority of the Internet has moved to IPv6 but some systems still use IPv4. The sender wants to use IPv6, but the receiver does not understand IPv6. Tunneling does not work in this situation because the packet must be in the IPv4 format to be understood by the receiver.



Comparison of IPV4 & IPV6

IPv4 Address	IPv6 Address
Address Length – 32 bits	128 bits
Address Representation - decimal	hexadecimal
Internet address classes	Not applicable in IPv6
Multicast addresses (224.0.0.0/4)	IPv6 multicast addresses (FF00::/8)
Broadcast addresses	Not applicable in IPv6
Unspecified address is 0.0.0.0	Unspecified address is ::
Loopback address is 127.0.0.1	Loopback address is ::1
Public IP addresses	Global unicast addresses
Private IP addresses (10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16)	Site-local addresses (FEC0::/10)
Autoconfigured addresses (169.254.0.0/16)	Link-local addresses (FE80::/64)