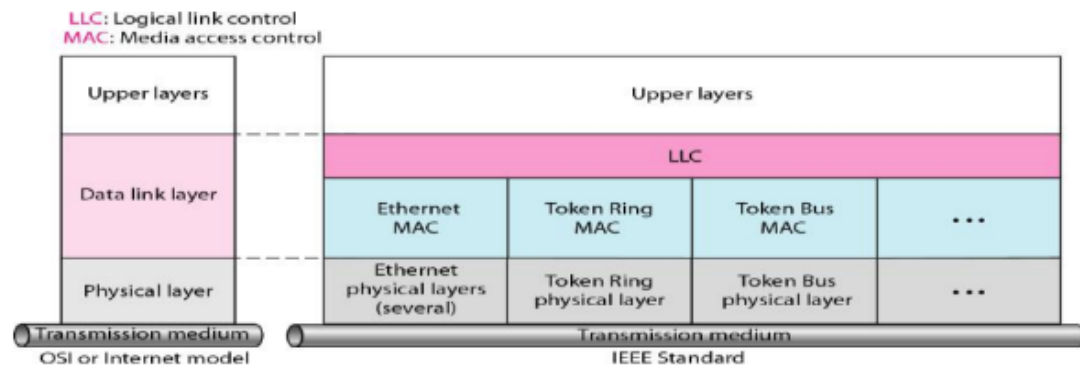


## Wired LANs: Ethernet

In 1985, the Computer Society of the IEEE started a project, called Project 802, to set standards to enable intercommunication among equipment from a variety of manufacturers. Project 802 is a way of specifying functions of the physical layer and the data link layer of major LAN protocols.

The relationship of the 802 Standard to the traditional OSI model is shown in below Figure. The IEEE has subdivided the data link layer into two sub layers: logical link control (LLC) and media access control (MAC).

IEEE has also created several physical layer standards for different LAN protocols



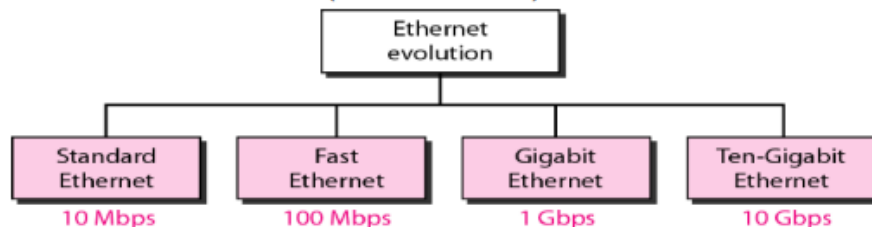
*IEEE standard for LANs*

## STANDARD ETHERNET

The original Ethernet was created in 1976 at Xerox's Palo Alto Research Center (PARC). Since then, it has gone through four generations.

Standard Ethernet (10 Mbps), Fast Ethernet (100 Mbps), Gigabit Ethernet (1 Gbps), and Ten-Gigabit Ethernet (10 Gbps),

We briefly discuss the Standard (or traditional) Ethernet in this section



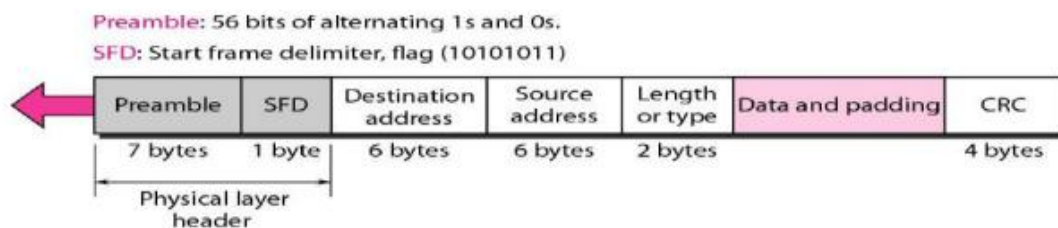
*Ethernet evolution through four generations*

## **MAC Sublayer**

In Standard Ethernet, the MAC sublayer governs the operation of the access method. It also frames data received from the upper layer and passes them to the physical layer.

### **Frame Format**

The Ethernet frame contains seven fields: preamble, SFD, DA, SA, length or type of protocol data unit (PDU), upper-layer data, and the CRC. Ethernet does not provide any mechanism for acknowledging received frames, making it what is known as an unreliable medium. Acknowledgments must be implemented at the higher layers. The format of the MAC frame is shown in below figure



**Preamble.** The first field of the 802.3 frame contains 7 bytes (56 bits) of alternating 0s and 1s that alerts the receiving system to the coming frame and enables it to synchronize its input timing. The pattern provides only an alert and a timing pulse. The 56-bit pattern allows the stations to miss some bits at the beginning of the frame. The preamble is actually added at the physical layer and is not (formally) part of the frame.

**Start frame delimiter (SFD).** The second field (1 byte: 10101011) signals the beginning of the frame. The SFD warns the station or stations that this is the last chance for synchronization. The last 2 bits is 11 and alerts the receiver that the next field is the destination address.

**Destination address (DA).** The DA field is 6 bytes and contains the physical address of the destination station or stations to receive the packet.

**Source address (SA).** The SA field is also 6 bytes and contains the physical address of the sender of the packet.

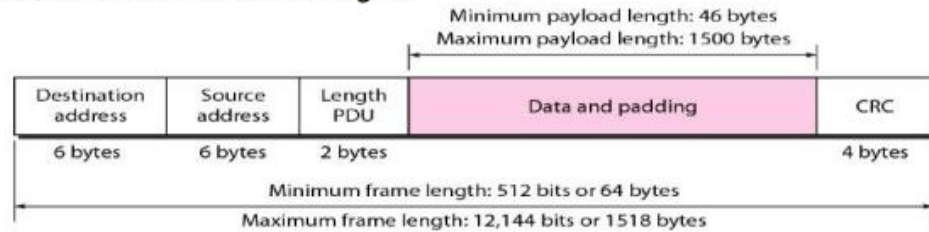
**Length or type.** This field is defined as a type field or length field. The original Ethernet used this field as the type field to define the upper-layer protocol using the MAC frame. The IEEE standard used it as the length field to define the number of bytes in the data field. Both uses are common today.

**Data.** This field carries data encapsulated from the upper-layer protocols. It is a minimum of 46 and a maximum of 1500 bytes.

**CRC.** The last field contains error detection information, in this case a CRC-32

### **Frame Length**

Ethernet has imposed restrictions on both the minimum and maximum lengths of a frame, as shown in below Figure



*Minimum and maximum lengths*

### **Addressing**

The Ethernet address is 6 bytes (48 bits), normally written in hexadecimal notation, with a colon between the bytes.

**Example of an Ethernet address in hexadecimal notation**

**06 : 01 : 02 : 01 : 2C : 4B**

6 bytes = 12 hex digits = 48 bits

If the least significant bit of the first byte in a destination address is 0, the address is unicast; otherwise, it is multicast.



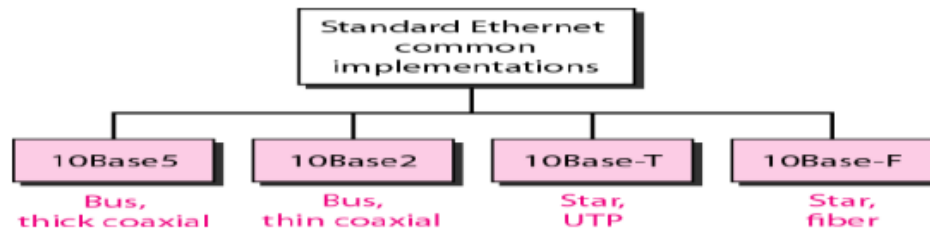
*Unicast and multicast addresses*

A unicast destination address defines only one recipient; the relationship between the sender and the receiver is one-to-one.

A multicast destination address defines a group of addresses; the relationship between the sender and the receivers is one-to-many.

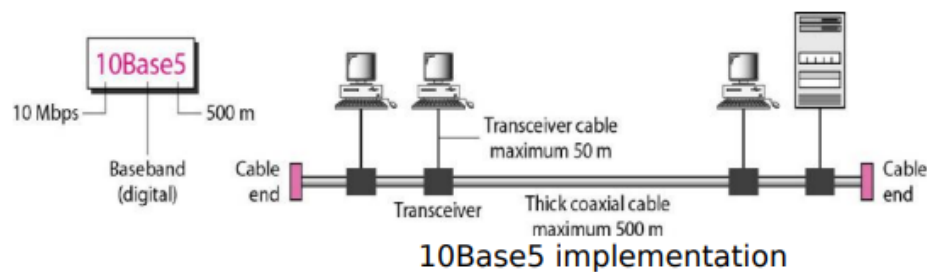
The broadcast address is a special case of the multicast address; the recipients are all the stations on the LAN. A broadcast destination address is forty-eight 1s.

The Standard Ethernet defines several physical layer implementations; four of the most common, are shown in Figure



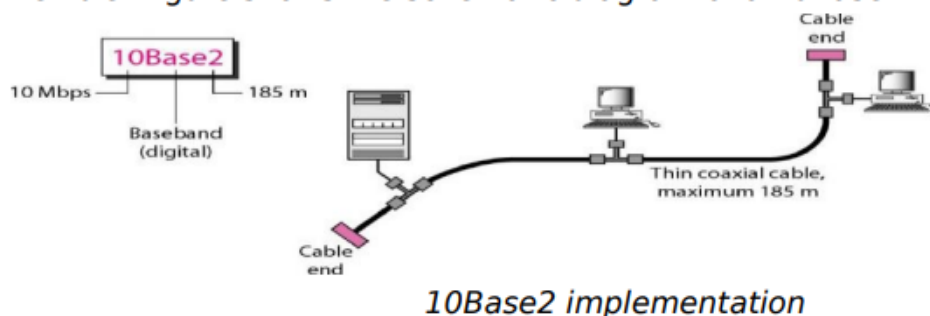
### 10Base5: Thick Ethernet

The first implementation is called **10Base5, thick Ethernet, or Thicknet**. 10Base5 was the first Ethernet specification to use a bus topology with an external **transceiver** (transmitter/receiver) connected via a tap to a thick coaxial cable. Figure shows a schematic diagram of a 10Base5 implementation



### 10Base2: Thin Ethernet

The second implementation is called 10 Base2, **thin** Ethernet, or Cheapernet. 10Base2 also uses a bus topology, but the cable is much thinner and more flexible. Figure shows the schematic diagram of a 10Base2 implementation.



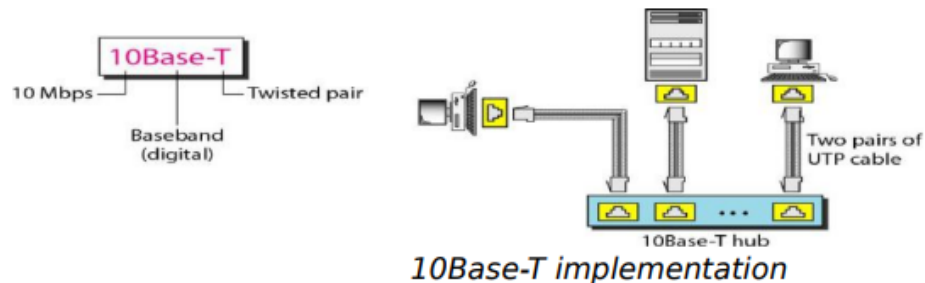


thin coaxial cable is less expensive than thick coaxial. Installation is simpler because the thin coaxial cable is very flexible. However, the length of each segment cannot exceed 185 m (close to 200 m) due to the high level of attenuation in thin coaxial cable.

### 10Base-T: Twisted-Pair Ethernet

The third implementation is called 10Base-T or twisted-pair Ethernet. It uses a physical star topology. The stations are connected to a hub via two pairs of twisted cable, as shown in Figure

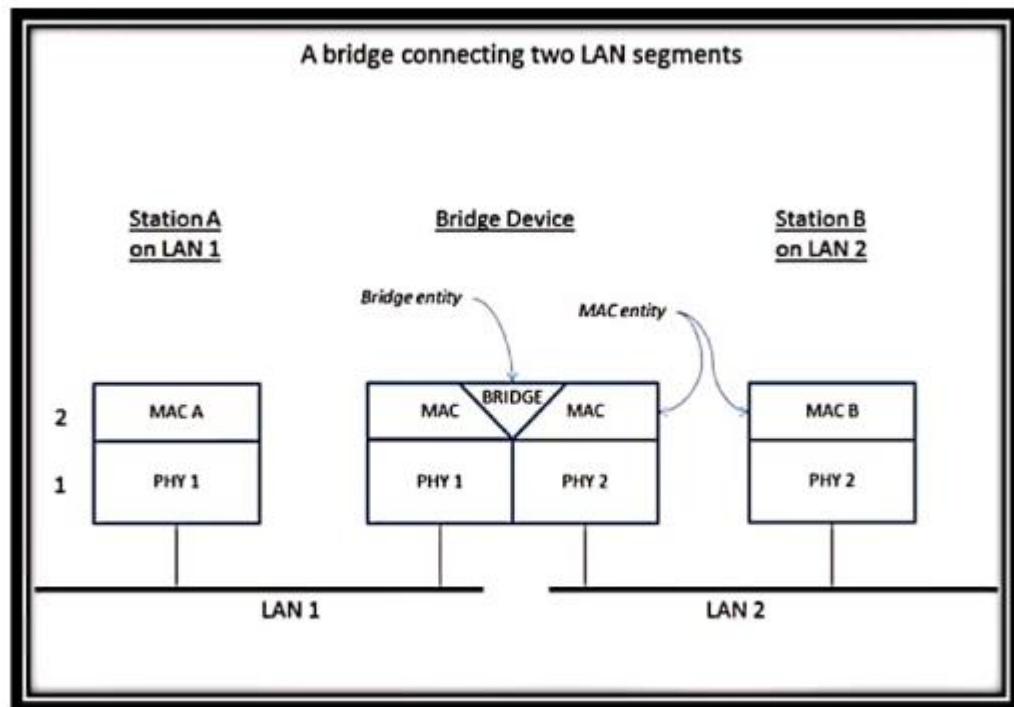
The maximum length of the twisted cable here is defined as 100 m, to minimize the effect of attenuation in the twisted cable



### **BRIDGE**

Bridges can be used to connect two or more LAN segments of the same type (e.g. Ethernet to Ethernet, or Token-Ring to Token-Ring). Like repeaters, bridges can extend the length of a network, but unlike repeaters they can also extend the capacity of a network, since each port on a bridge has its own MAC address. When bridges are powered on in an Ethernet network, they start to learn the network's topology by analysing the source addresses of incoming frames from all attached network segments (a process called backward learning ).

Over a period of time, they build up a routing table . Unless the source and the destination are on different network segments, there is no need for the bridge to transfer an incoming frame to another network segment. If the source and the destination are on different segments, the bridge needs to be able to determine which segment the destination device belongs to.



**Fig: A high-level overview of network bridging, using the ISO/OSI layers and terminology**

### Types

There are four types of network bridging technologies: simple bridging, multiport bridging, learning or transparent bridging, and source route bridging.

### Transparent bridging

A transparent bridge uses a forwarding database to send frames across network segments. The forwarding database is initially empty and entries in the database are built as the bridge receives frames. If an address entry is not found in the forwarding database, the frame is flooded to all other ports of the bridge, flooding the frame to all segments except the one from which it was received. By means of these flooded frames, the destination network will respond and a forwarding database entry will be created.

In the context of a two-port bridge, the forwarding database can be thought of as a filtering database. A bridge reads a frame's destination address and decides to either forward or filter. If the bridge determines that the destination node is on another segment on the network, it forwards (retransmits) the frame to that segment. If the destination address belongs to the same segment as the source address, the bridge filters (discards) the frame. As nodes transmit data through the bridge, the bridge establishes a filtering database of known MAC addresses and their locations on the network. The bridge uses its filtering database to determine whether a packet should be forwarded or filtered.

### **Simple bridging**

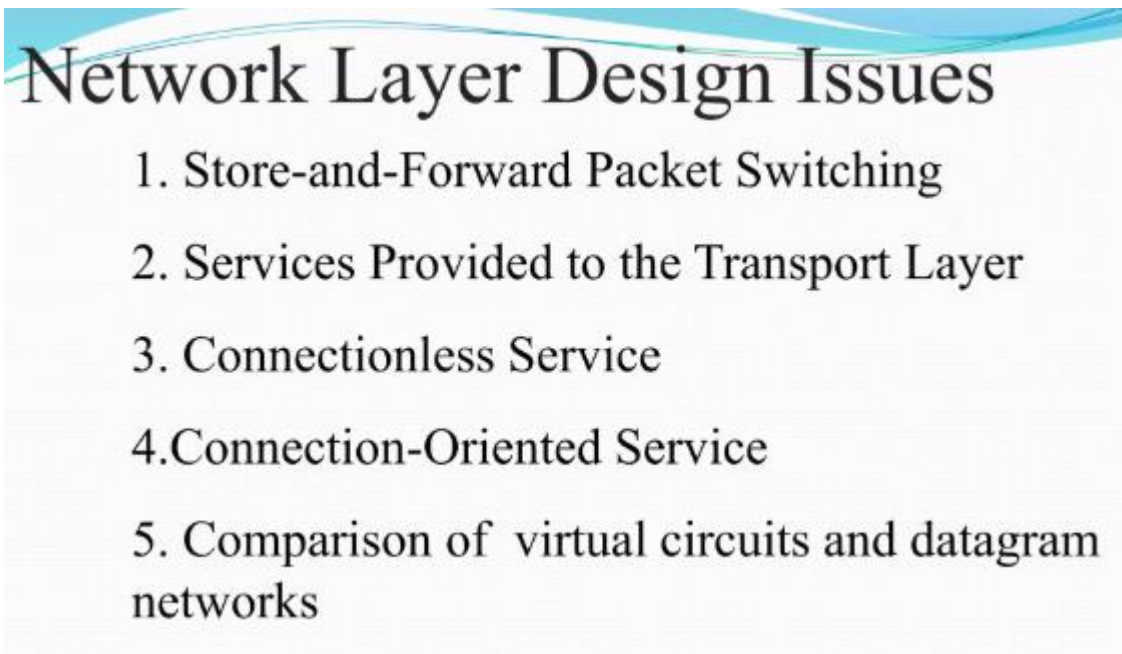
The simple bridge connects two network segments. It typically operates transparently and decides on a packet-by-packet basis whether or not to forward from one network to the other. A store and forward technique is typically used such that, during forwarding, packet integrity is verified on the source network and CSMA/CD delays are accommodated on the destination network. In contrast to simple repeaters that were used to just extend the maximum reach of a segment, a bridge also reduces collisions by splitting the collision domain. Additionally, it lowers overall traffic by only forwarding those packets that are required to cross the bridge.

### **Multiport bridging**

A multiport bridge connects multiple networks and operates transparently to decide on a packet-by-packet basis whether and where to forward. Like the simple bridge, a multiport bridge typically uses store and forward operation. The multiport bridge function is the basis for a network switch.

### **Source-Route Bridging (Token Ring)**

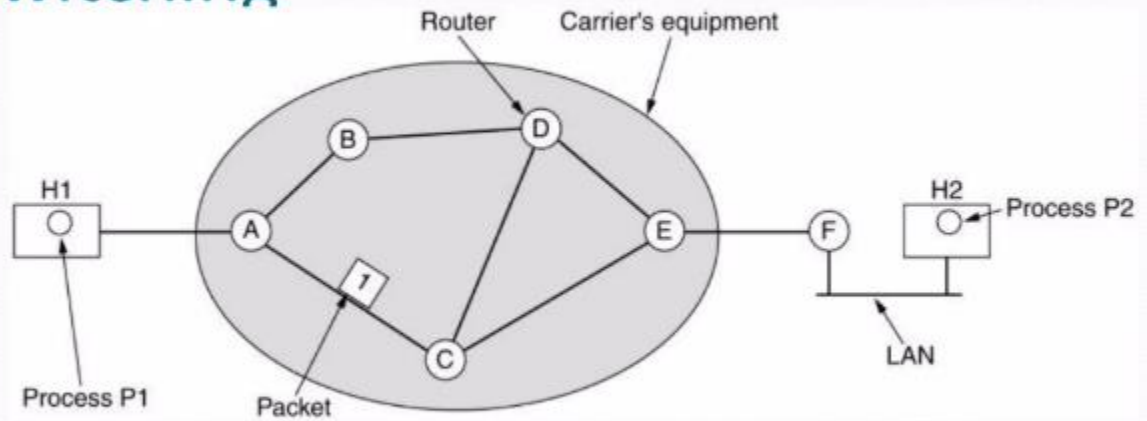
The source-route bridging (SRB) algorithm was developed by IBM for Token Ring networks, and gets its name from the fact that routing information is placed in all inter-segment frames by the sending device. Bridges forward frames according to the routing information carried within the frame. A simple source-route bridging network is illustrated below.



## Network Layer Design Issues

1. Store-and-Forward Packet Switching
2. Services Provided to the Transport Layer
3. Connectionless Service
4. Connection-Oriented Service
5. Comparison of virtual circuits and datagram networks

# 1.Store and Forward Packet switching



The environment of the network layer protocols.

## 1.Store and Forward Packet switching

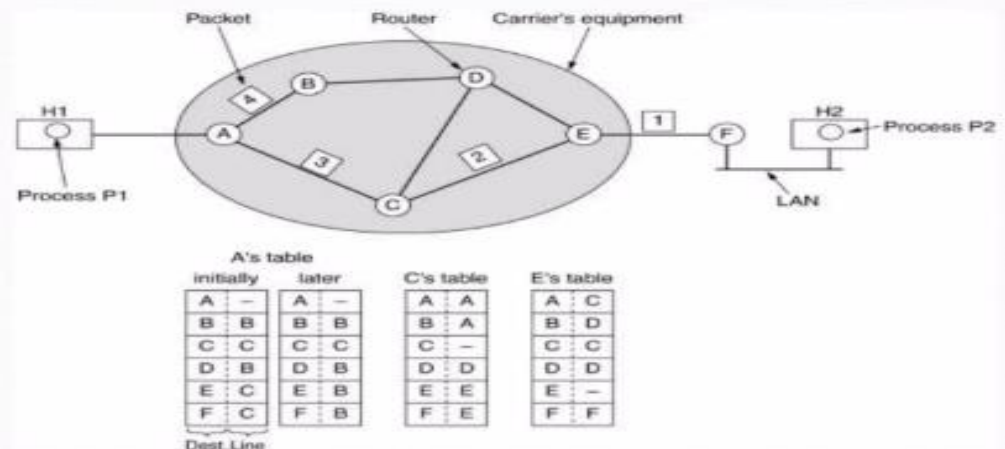
- A host with a packet to send transmits it to the nearest router by using its own LAN or Point to point link to the ISP . The packet is stored until it fully arrive and link has finished its processing by verifying the checksum. Then it is forwarded to the next route along the path until it reaches the destination host when it is to be delivered this mechanism is called store and forward packet switching



## 2. Services provided to the transport Layer

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. Connectionless Service

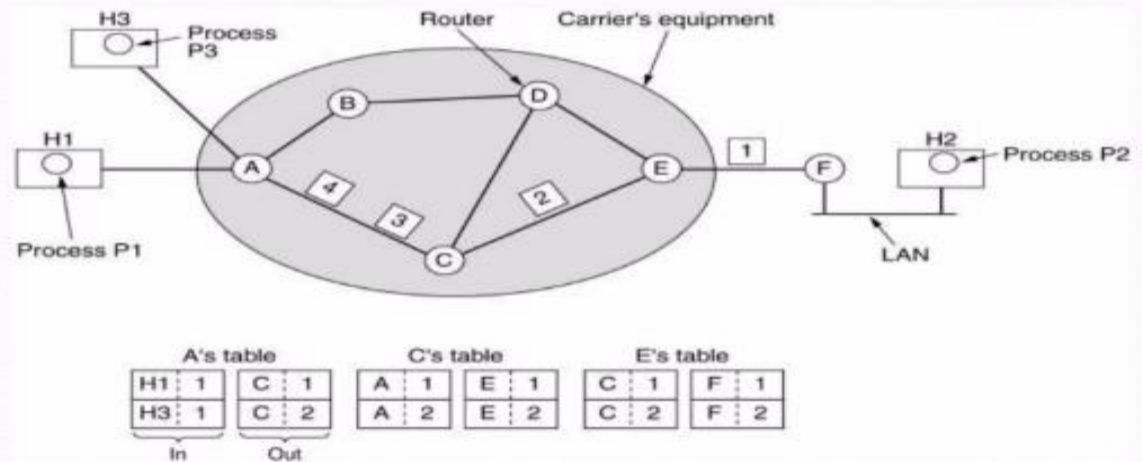


Routing within a diagram subnet.

### 3. Connectionless Service

- > Packets are injected into the network individually and routed independently of each other . Packets are called as Datagrams
- > Destination and outgoing line table is maintained at every router  
Internal router table will guide where to send the packets from the different available possibilities for reaching the destination
- > Incoming packets checksum verified before forwarding to the next link

## 4. Connection-Oriented Service



Routing within a virtual-circuit subnet.

## 4. Connection-Oriented Service

- Connection is established a route from the source machine to the destination machine is chosen as part of the connection set up and stored in tables inside the routers that route is used for all traffic flowing over the connection when the connection is released virtual circuit will be terminated

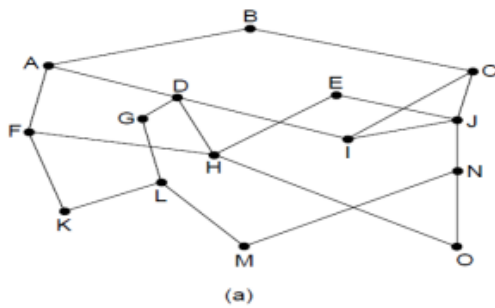
Eg: Telephone Lines

### The Optimality Principle

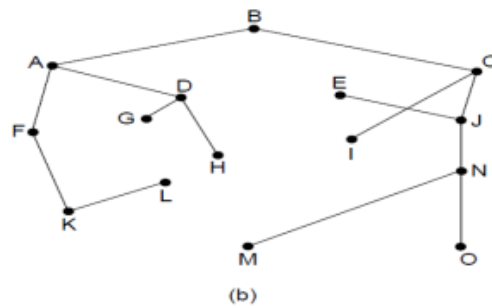
One can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the optimality principle.

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.

## Routing Algorithms

The main function of NL (Network Layer) is routing packets from the source machine to the destination machine.

There are two processes inside router:

- a) One of them handles each packet as it arrives, looking up the outgoing line to use for it in the routing table. This process is forwarding.
- b) The other process is responsible for filling in and updating the routing tables. That is where the routing algorithm comes into play. This process is routing.

Routing algorithms can be grouped into two major classes:

- 1) nonadaptive (Static Routing)
- 2) adaptive. (Dynamic Routing)

Nonadaptive algorithm do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from I to J is computed in advance, off line, and downloaded to the routers when the network is booted. This procedure is sometimes called static routing.

Adaptive algorithm, in contrast, 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  $\Delta T$  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

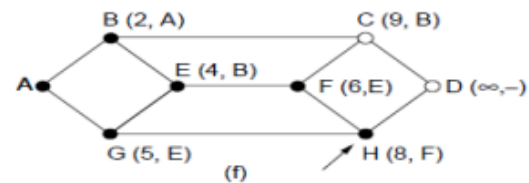
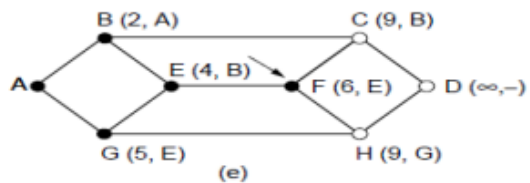
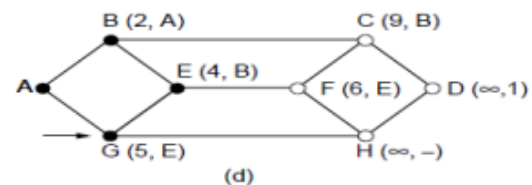
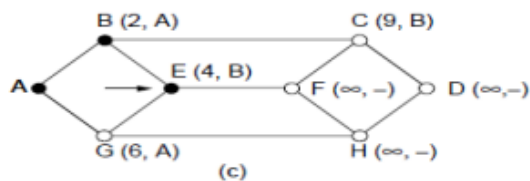
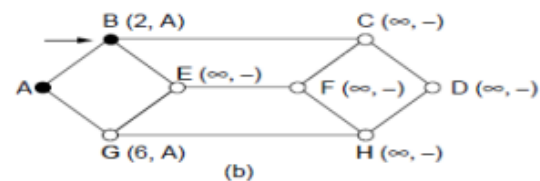
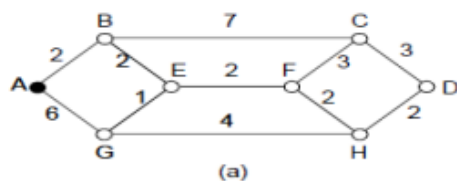


### Shortest Path Routing (Dijkstra's)

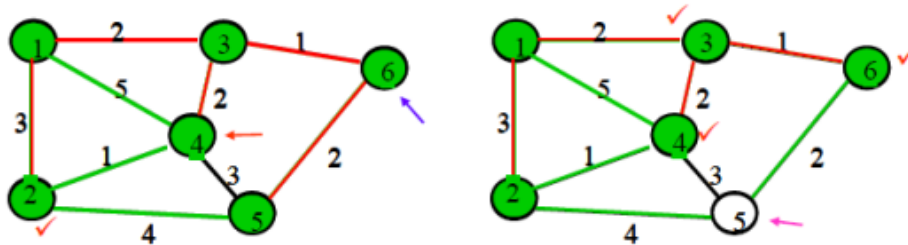
The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line or link.

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 neighbor 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



## Execution of Dijkstra's algorithm



Iteration	Permanent	tentative	D <sub>2</sub>	D <sub>3</sub>	D <sub>4</sub>	D <sub>5</sub>	D <sub>6</sub>
Initial	{1}	{2,3,4}	3	2 ✓	5	∞	∞
1	{1,3}	{2,4,6}	3 ✓	2	4	∞	3
2	{1,2,3}	{4,6,5}	3	2	4	7	3 ✓
3	{1,2,3,6}	{4,5}	3	2	4 ✓	5	3
4	{1,2,3,4,6}	{5}	3	2	4	5 ✓	3
5	{1,2,3,4,5,6}	{}	3	2	4	5	3

### Flooding

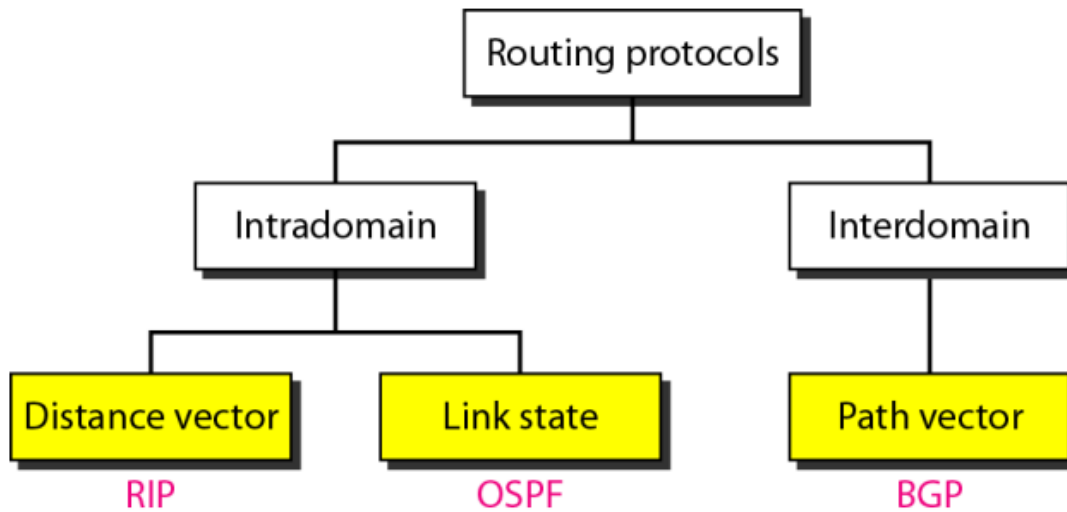
- Another static algorithm is flooding, in which every incoming packet is sent out on every outgoing line except the one it arrived on.
- Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process.
- One such measure is to have a hop counter contained in the header of each packet, which is decremented at each hop, with the packet being discarded when the counter reaches zero. Ideally, the hop counter should be initialized to the length of the path from source to destination.
- A variation of flooding that is slightly more practical is selective flooding. In this algorithm the routers do not send every incoming packet out on every line, only on those lines that are going approximately in the right direction.
- Flooding is not practical in most applications.

### Intra- and Inter domain Routing

An autonomous system (AS) is a group of networks and routers under the authority of a single administration.

Routing inside an autonomous system is referred to as intra domain routing. (DISTANCE VECTOR, LINK STATE)

Routing between autonomous systems is referred to as inter domain routing. (PATH VECTOR)  
Each autonomous system can choose one or more intra domain routing protocols to handle routing inside the autonomous system. However, only one inter domain routing protocol handles routing between autonomous systems.



### **Distance Vector Routing**

In distance vector routing, the least-cost route between any two nodes is the route with minimum distance. In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node.

Mainly 3 things in this

***Initialization***

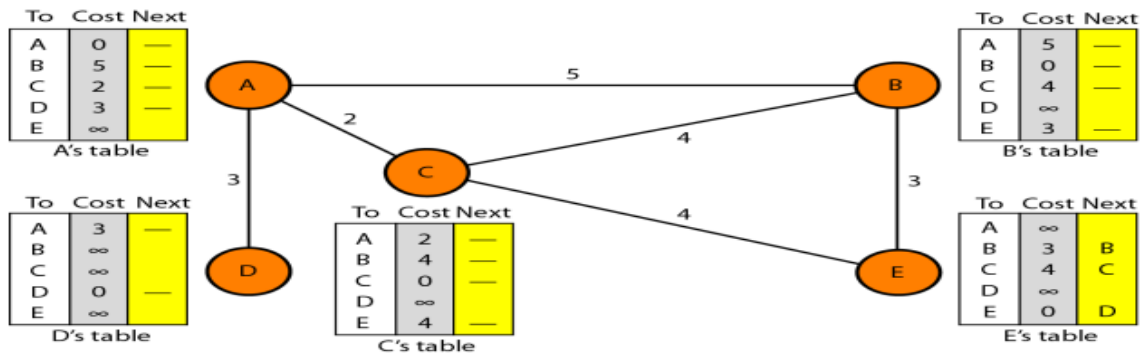
***Sharing***

***Updating***

### **Initialization**

Each node can know only the distance between itself and its immediate neighbors, those directly connected to it. The distance for any entry that is not a neighbor is marked as infinite (unreachable).

### Initialization of tables in distance vector routing



### Sharing

The whole idea of distance vector routing is the sharing of information between neighbors. 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.

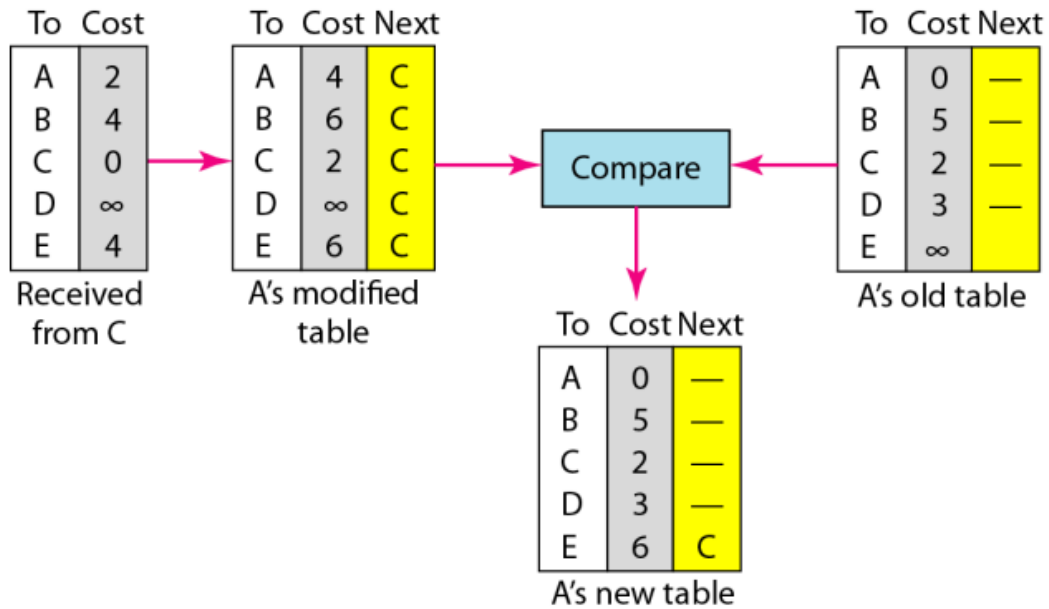
### Updating

When a node receives a two-column table from a neighbor, it needs to update its routing table. Updating takes three steps:

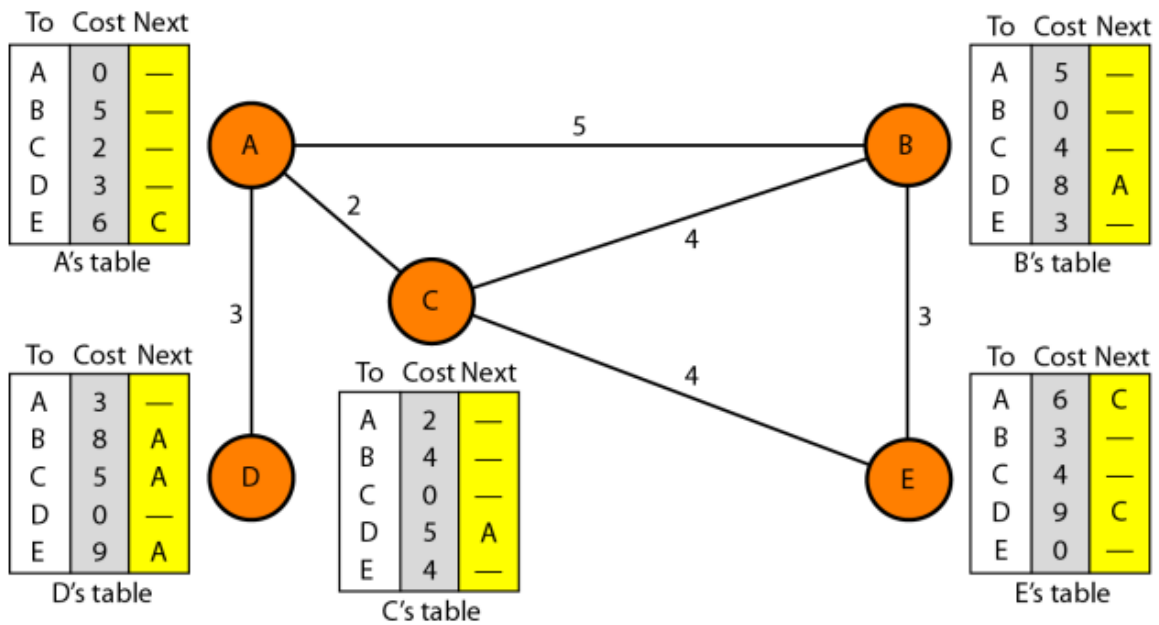
1. The receiving node needs to add the cost between itself and the sending node to each value in the second column. ( $x+y$ )
2. If the receiving node uses information from any row. The sending node is the next node in the route.
3. The receiving node needs to compare each row of its old table with the corresponding row of the modified version of the received table.
  - a. If the next-node entry is different, the receiving node chooses the row with the smaller cost. If there is a tie, the old one is kept.
  - b. If the next-node entry is the same, the receiving node chooses the new row.



### Updating in distance vector routing



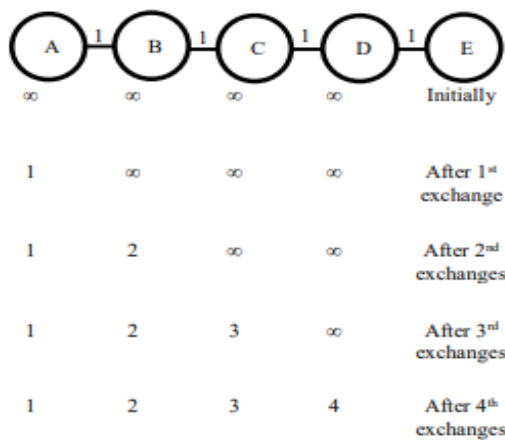
### Final Diagram



**Periodic Update** A node sends its routing table, normally every 30 s, in a periodic update. The period depends on the protocol that is using distance vector routing.

**Triggered Update** A node sends its two-column routing table to its neighbors anytime there is a change in its routing table. This is called a triggered update. The change can result from the following.

1. A node receives a table from a neighbor, resulting in changes in its own table after updating.
2. A node detects some failure in the neighboring links which results in a distance change to infinity.



(a). Before Count to Infinity Problem.

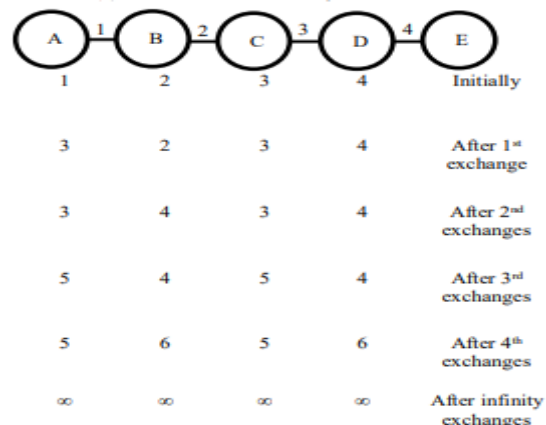


Fig. 1. (b). After Count to Infinity Problem.

## SOLUTIONS FOR INSTABILITY

### Defining Infinity:

most implementations of the distance vector protocol define the distance between each node to be 1 and define 16 as infinity. However, this means that the distance vector routing cannot be used in large systems. The size of the network, in each direction, cannot exceed 15 hops.

### Split Horizon:

In this strategy, instead of flooding the table through each interface, each node sends only part of its table through each interface. If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows)

### Split Horizon and Poison Reverse

Split Horizon and Poison Reverse Using the split horizon strategy has one drawback. Normally, the distance vector protocol uses a timer, and if there is no news about a route, the node deletes the route from its table

The split horizon strategy can be combined with the poison reverse strategy. Node B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."

### **Link State Routing Algorithm**

- Link state routing is the second family of routing protocols.
- While distance vector routers use a distributed algorithm to compute their routing tables, link-state routing uses link-state routers to exchange messages that allow each router to learn the entire network topology.
- Based on this learned topology, each router is then able to compute its routing table by using a shortest path computation.

### **Features –**

- **Link state packet** – A small packet that contains routing information.
- **Link state database** – A collection information gathered from link state packet.
- **Shortest path first algorithm (Dijkstra algorithm)** – A calculation performed on the database results into shortest path
- **Routing table** – A list of known paths and interfaces.

### **Calculation of shortest path –**

To find shortest path, each node need to run the famous Dijkstra algorithm. This famous algorithm uses the following steps:

**Step-1:** The node is taken and chosen as a root node of the tree, this creates the tree with a single node, and now set the total cost of each node to some value based on the information in Link State Database

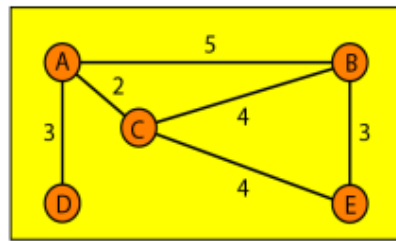
**Step-2:** Now the node selects one node, among all the nodes not in the tree like structure, which is nearest to the root, and adds this to the tree. The shape of the tree gets changed .

**Step-3:** After this node is added to the tree, the cost of all the nodes not in the tree needs to be updated because the paths may have been changed.

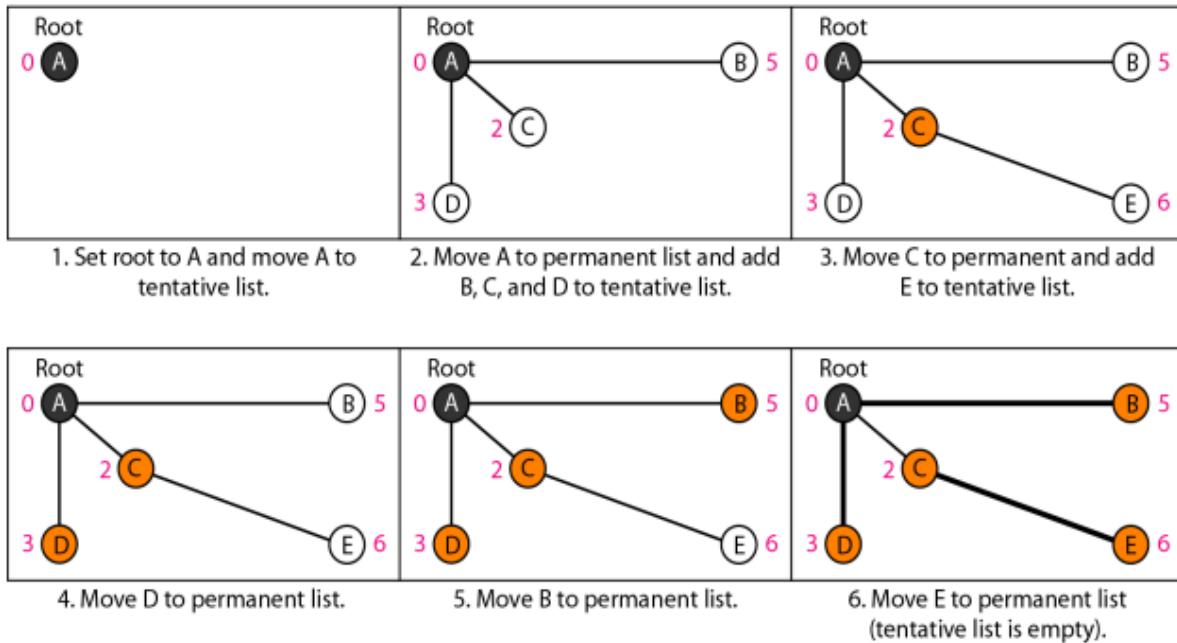
**Step-4:** The node repeats the Step 2. and Step 3. until all the nodes are added in the tree

Link State protocols in comparison to Distance Vector protocols have:

1. It requires large amount of memory.
2. Shortest path computations require many CPU cycles.
3. If network use the little bandwidth ; it quickly reacts to topology changes
4. All items in the database must be sent to neighbors to form link state packets.
5. All neighbors must be trusted in the topology.
6. Authentication mechanisms can be used to avoid undesired adjacency and problems.
7. No split horizon techniques are possible in the link state routing.



Topology



#### IV. Calculation of a routing table

routing table for node A

<i>Node</i>	<i>Cost</i>	<i>Next Router</i>
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C



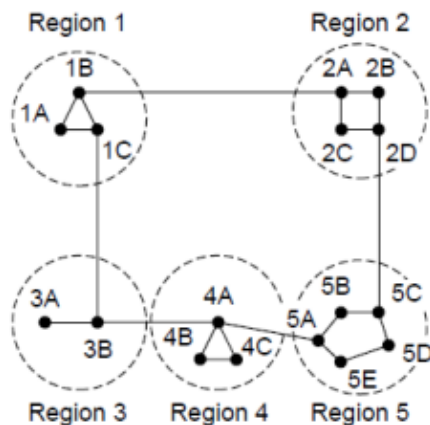
## Hierarchical Routing

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. Each router knows all the details about how to route packets to destinations within its own region but knows 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



(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 ALGORITHMS

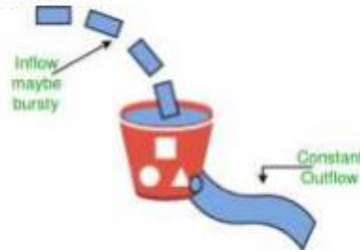
Too many packets present in (a part of) the network causes packet delay and loss that degrades performance. This situation is called congestion. The network and transport layers share the responsibility for handling congestion.



- **Leaky Bucket Algorithm**

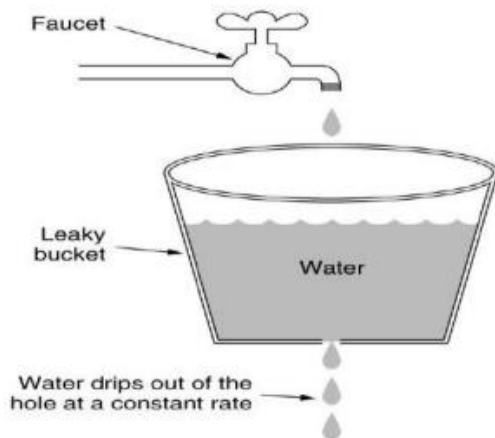
Let us consider an example to understand

Imagine a bucket with a small hole in the bottom. No matter at what rate water enters the bucket, the outflow is at constant rate. When the bucket is full with water additional water entering spills over the sides and is lost.



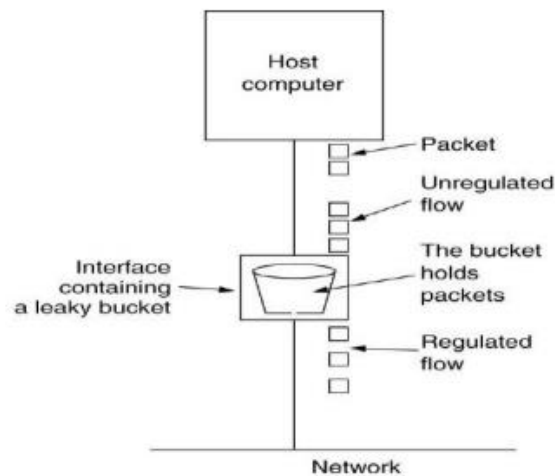
Similarly, each network interface contains a leaky bucket and the following **steps** are involved in leaky bucket algorithm:

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.



(a)

(a) A leaky bucket with water.



(b)

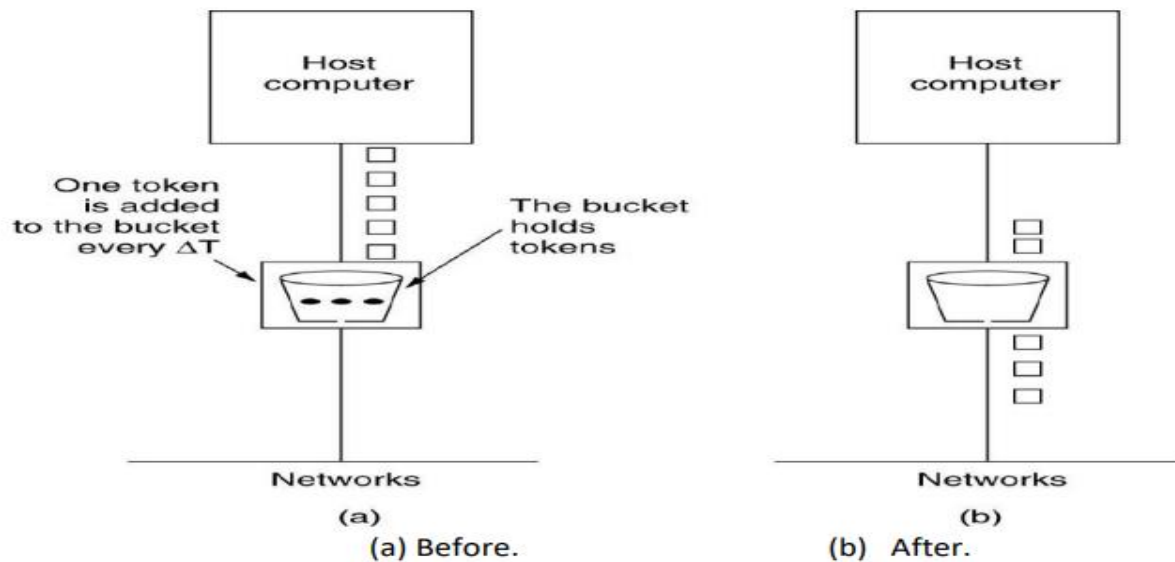
(b) a leaky bucket with packets.

- **Token bucket Algorithm**

- 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.  $f$
2. The bucket has a maximum capacity.  $f$
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.





### Leaky Bucket vs. Token Bucket

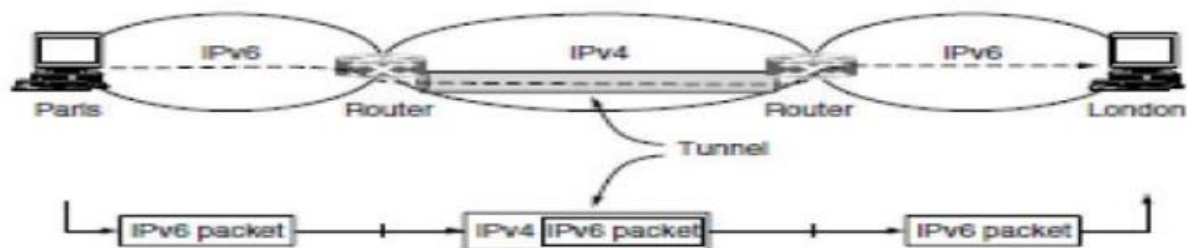
1. LB discards packets; TB does not. TB discards tokens.
2. With TB, a packet can only be transmitted if there are enough tokens to cover its length in bytes.
3. LB sends packets at an average rate. TB allows for large bursts to be sent faster by speeding up the output.
4. TB allows saving up tokens (permissions) to send large bursts. LB does not allow saving.

## INTERNETWORKING

Many different networks exist, including PANs, LANs, MANs, and WANs. We have described Ethernet, Internet over cable, the fixed and mobile telephone networks, 802.11, 802.16, and more. Numerous protocols are in widespread use across these networks in every layer. In the following sections, we will take a careful look at the issues that arise when two or more networks are connected to form an internetwork, or more simply an internet.

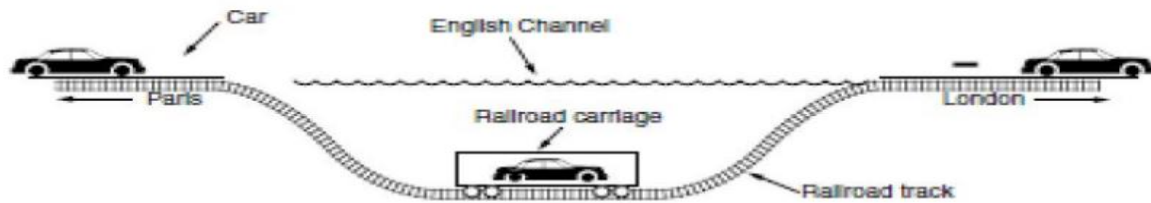
### Tunneling

Handling the general case of making two different networks interwork is exceedingly difficult. However, there is a common special case that is manageable even for different network protocols. 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 an IPv6 network in Paris, an IPv6 network in London and connectivity between the offices via the IPv4 Internet. This situation is shown in Fig



The solution to this problem is a technique called tunneling. To send an IP packet to a host in the London office, a host in the Paris office constructs the packet containing an IPv6 address in London, and sends it to the multiprotocol router that connects the Paris IPv6 network to the IPv4 Internet. When this router gets the IPv6 packet, it encapsulates the packet with an IPv4 header addressed to the IPv4 side of the multiprotocol router that connects to the London IPv6 network. When this wrapped packet arrives, the London router removes the original IPv6 packet and sends it onward to the destination host. The path through the IPv4 Internet can be seen as a big tunnel extending from one multiprotocol router to the other. The IPv6 packet just travels from one end of the tunnel to the other, snug in its nice box.

Tunneling a car from France to England.

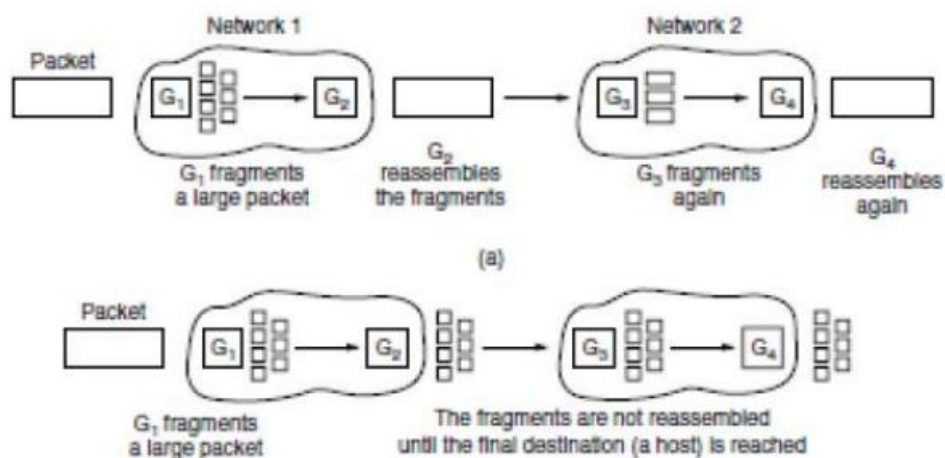


## Packet Fragmentation

An obvious internetworking problem appears when a large packet wants to travel through a network whose maximum packet size is too small. . A source does not usually know the path a packet will take through the network to a destination, so it certainly does not know how small packets must be to get there. This packet size is called the Path MTU (Path Maximum Transmission Unit). Even if the source did know the path MTU, packets are routed independently in a connectionless network such as the Internet. This routing means that paths may suddenly change, which can unexpectedly change the path MTU. The alternative solution to the problem is to allow routers to break up packets into **fragments**, sending each fragment as a separate network layer packet.

Packet-switching networks, too, have trouble putting the fragments back together again. Two opposing strategies exist for recombining the fragments back into the original packet. The first strategy is to make fragmentation caused by a “small packet” network transparent to any subsequent networks through which the packet must pass on its way to the ultimate destination. This option is shown in Fig (a). In this approach, when an oversized packet arrives at G<sub>1</sub>, the router breaks it up into fragments. Each fragment is addressed to the same exit router, G<sub>2</sub>, where the pieces are recombined. In this way, passage through the small-packet network is made transparent. Subsequent networks are not even aware that fragmentation has occurred.

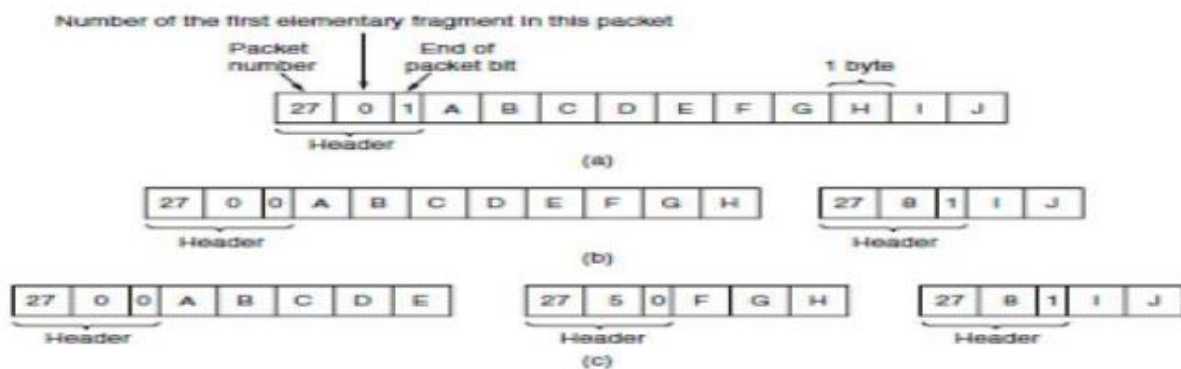
Transparent fragmentation is straightforward but has some problems. For one thing, the exit router must know when it has received all the pieces, so either a count field or an “end of packet” bit must be provided



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

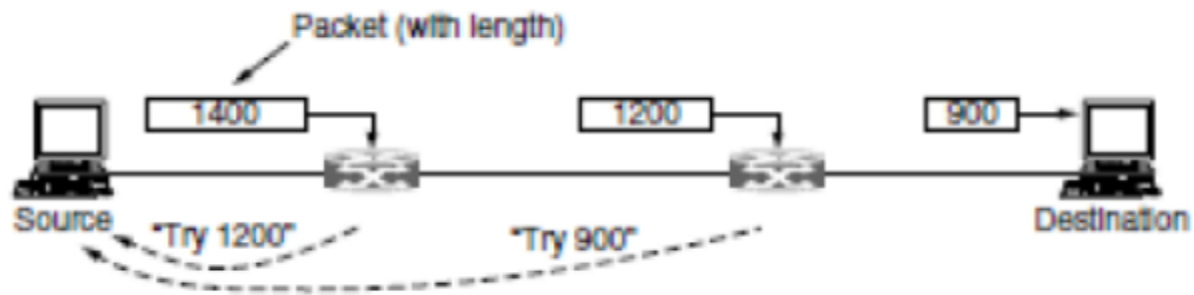
The routers pass the fragments, as shown in Fig. (b), and reassembly is performed only at the destination host. The main advantage of nontransparent fragmentation is that it requires routers to do less work. IP works this way. A complete design requires that the fragments be numbered in such a way that the original data stream can be reconstructed.

Fragments can also be fragmented if they pass over a network with a yet smaller MTU. This is shown in Fig. (c). Retransmissions of the packet (if all fragments were not received) can be fragmented into different pieces. Finally, fragments can be of arbitrary size, down to a single byte plus the packet header. In all cases, the destination simply uses the packet number and fragment offset to place the data in the right position, and the end-of-packet flag to determine when it has the complete packet. Unfortunately, this design still has problems. The overhead can be higher than with transparent fragmentation because fragment headers are now carried over some links where they may not be needed.



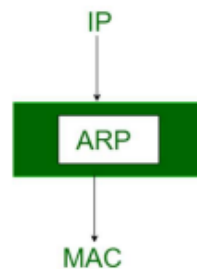
But the real problem is the existence of fragments in the first place. Kent and Mogul (1987) argued that fragmentation is detrimental to performance because, as well as the header overheads, a whole packet is lost if any of its fragments are lost and because fragmentation is more of a burden for hosts than was originally realized.

This leads us back to the original solution of getting rid of fragmentation in the network, the strategy used in the modern Internet. The process is called **path MTU discovery** (Mogul and Deering, 1990). It works as follows. Each IP packet is sent with its header bits set to indicate that no fragmentation is allowed to be performed. If a router receives a packet that is too large, it generates an error packet, returns it to the source, and drops the packet.



### ARP (Address Resolution Protocol)

- Most of the computer programs/applications use logical address (IP address) to send/receive messages, however the actual communication happens over the physical address (MAC address) i.e from layer 2 of OSI model.
- So, our mission is to get the destination MAC address which helps in communicating with other devices.
- This is where ARP comes into the picture, its functionality is to translate IP address to physical address.



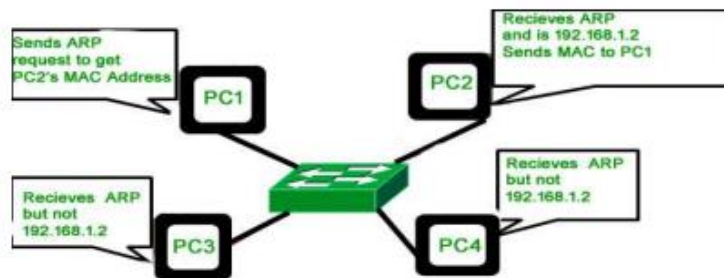
- The acronym ARP stands for Address Resolution Protocol which is one of the most important protocols of the Network layer in the OSI model.

**Note:** ARP finds the hardware address, also known as Media Access Control (MAC) address, of a host from its known IP address.

### ARP working

Before sending the IP packet, the MAC address of destination must be known. If not so, then sender broadcasts the ARP-discovery packet requesting the MAC address of intended destination. Since ARP-discovery is broadcast, every host inside that network will get this message but the packet will be discarded by everyone except that intended receiver host whose IP is associated. Now, this receiver will send a unicast packet with its MAC address (ARP-reply) to the sender of ARP-discovery packet. After the original sender receives the ARP-reply, it updates ARP-cache and start sending unicast message to the destination.

The above process continues till the second last network device in the path to reach the destination where it gets validated and ARP, in turn, responds with the destination MAC address.



The important terms associated with ARP are :

- **ARP Cache:** After resolving MAC address, the ARP sends it to the source where it stores in a table for future reference. The subsequent communications can use the MAC address from the table
- **ARP Cache Timeout:** It indicates the time for which the MAC address in the ARP cache can reside
- **ARP request:** This is nothing but broadcasting a packet over the network to validate whether we came across destination MAC address or not.

ARP request packet contains:

- The physical address of the sender.
- The IP address of the sender.
- The physical address of the receiver is 0s.
- The IP address of the receiver

Note, that the ARP packet is encapsulated directly into data link frame.

- **ARP response/reply:** It is the MAC address response that the source receives from the destination which aids in further communication of the data.



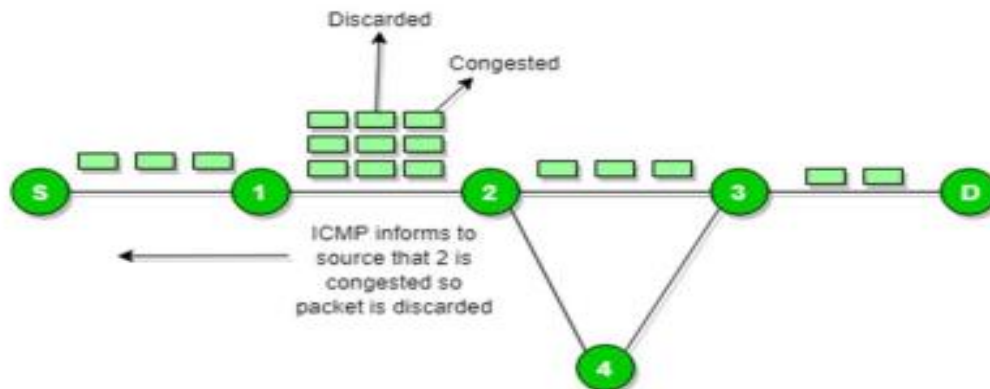
### Internet Control Message Protocol (ICMP)

- Since IP does not have a inbuilt mechanism for sending error and control messages. It depends on Internet Control Message Protocol(ICMP) to provide an error control.
- It is used for reporting errors and management queries.
- It is a supporting protocol and used by networks devices like routers for sending the error messages and operations information.
- e.g. the requested service is not available or that a host or router could not be reached.

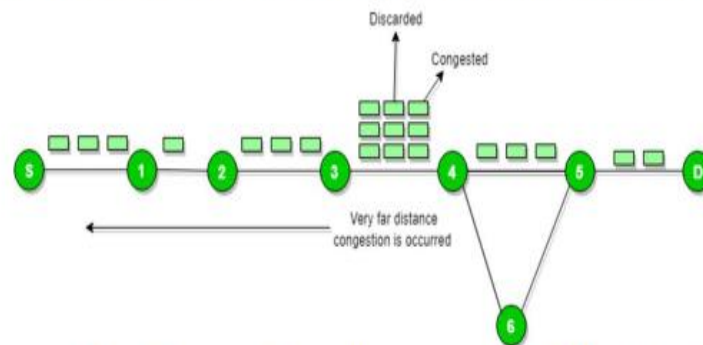
#### Source quench message:

Source quench message is a request to decrease traffic rate for messages sending to the host(destination). Or we can say, when receiving host detects that rate of sending packets (traffic rate) to it is too fast it sends the source quench message to the source to slow the pace down so that no packet can be lost.

ICMP will take source IP from the discarded packet and informs to source by sending source quench message.



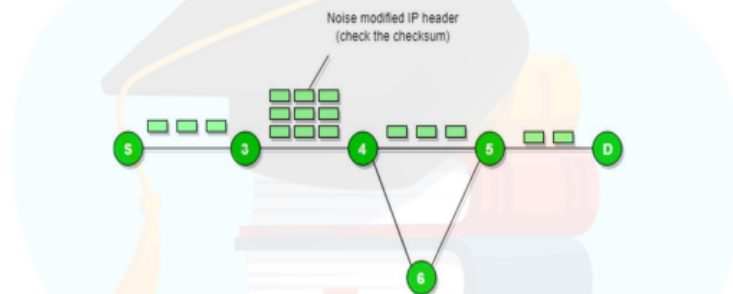
Then source will reduce the speed of transmission so that router will free for congestion.



When the congestion router is far away from the source the ICMP will send hop by hop source quench message so that every router will reduce the speed of transmission.

### **Parameter problem:**

Whenever packets come to the router then calculated header checksum should be equal to received header checksum then only packet is accepted by the router.

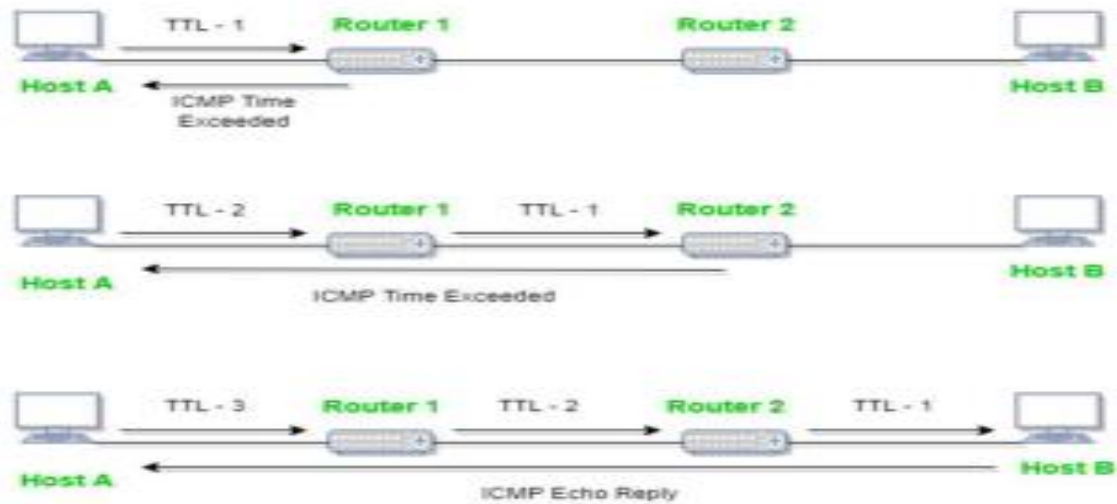


If there is mismatch packet will be dropped by the router.

ICMP will take the source IP from the discarded packet and informs to source by sending parameter problem message.

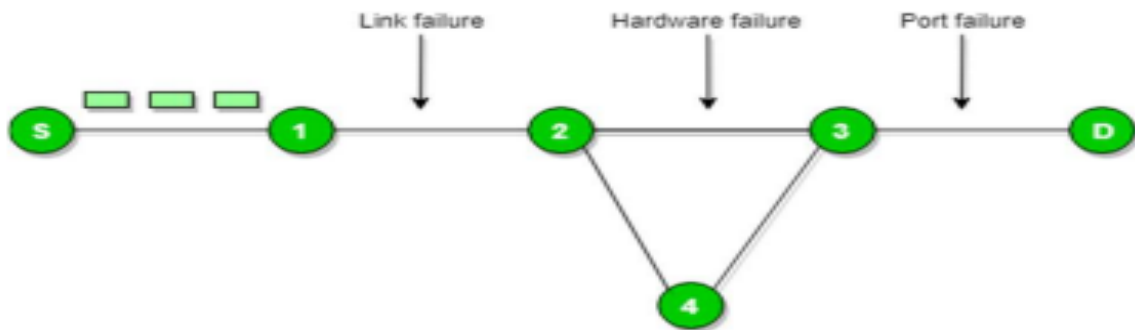
### **Time exceeded message:**

When some fragments are lost in a network then the holding fragment by the router will be dropped then ICMP will take source IP from discarded packet and informs to the source, of



### Destination un-reachable:

Destination unreachable is generated by the host or its inbound gateway to inform the client that the destination is unreachable for some reason.



### Redirection message:

Redirect requests data packets be sent on an alternate route. The message informs to a host to update its routing information (to send packets on an alternate route).