Program : **B.Tech**

Subject Name: **Computer Networks**

Subject Code: **CS-602**

Semester: **6th**

Department of Computer Science and Engineering
CS602 Computer Networks
Subject Notes: UNIT-IV

Syllabus: **Network Layer: Need, Services Provided , Design issues, Routing algorithms: Least CostRouting algorithm, Dijkstra's algorithm, Bellman-ford algorithm, Hierarchical Routing,Broadcast Routing, Multicast Routing. IP Addresses, Header format, Packet forwarding,Fragmentation and reassembly, ICMP, Comparative study of IPv4 & IPv6.**

Network Layer: Need

The network layer is considered the backbone of the OSI Model. It selects and manages the best logical path for data transfer between nodes. This layer contains hardware devices such as routers, bridges, firewalls and switches, but it actually creates a logical image of the most efficient communication route and implements it with a physical medium.

Network layer protocols exist in every host or router. The router examines the header fields of all the IP packets that pass through it. Internet Protocol and Netware IPX/SPX are the most common protocols associated with the network layer.

In the OSI model, the network layer responds to requests from the layer above it (transport layer) and issues requests to the layer below it (data link layer).

Network Layer: Services

It translates logical network address into physical address.

1. Routers and gateways operate in the network layer. Mechanism is provided by Network Layer for routing the packets to final destination.
2. Connection services are provided including flow control, error control and packet sequence control.
3. Breaks larger packets into small packets.

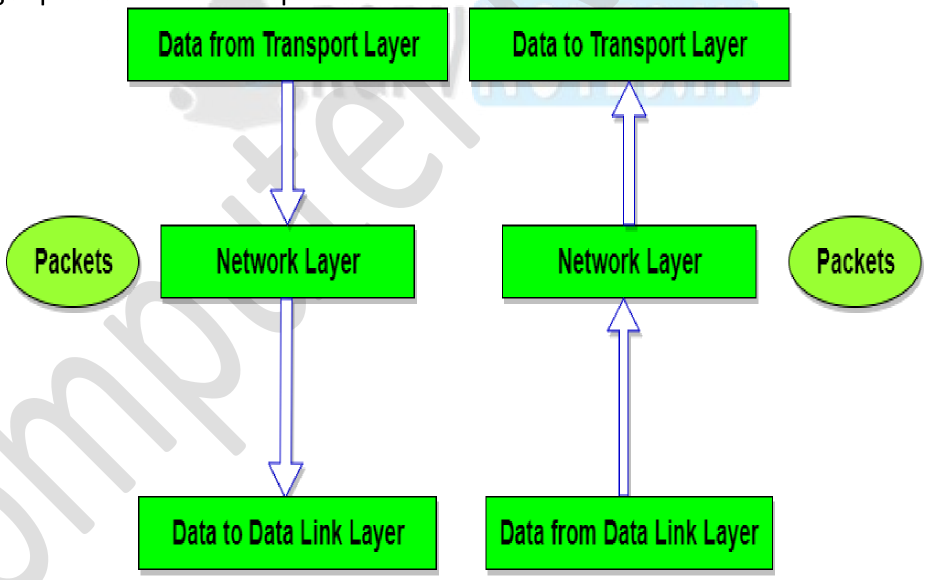


Fig 4.1 Network Layer

There are two types of service that can be provided by the network layer:

1. An unreliable connectionless service.
2. A connection-oriented, reliable or unreliable, service.

Network Layer: Design issues

a) Store-and-Forward Packet Switching
b) Services Provided to the Transport Layer
c) Implementation of Connectionless Service
d) Implementation of Connection-Oriented Service

a) Store-and-Forward Packet Switching

A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-pointlink to the carrier. The packet is stored there until it has fully arrived so the checksum can be verified. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered. This mechanism is store-and-forward packet switching.

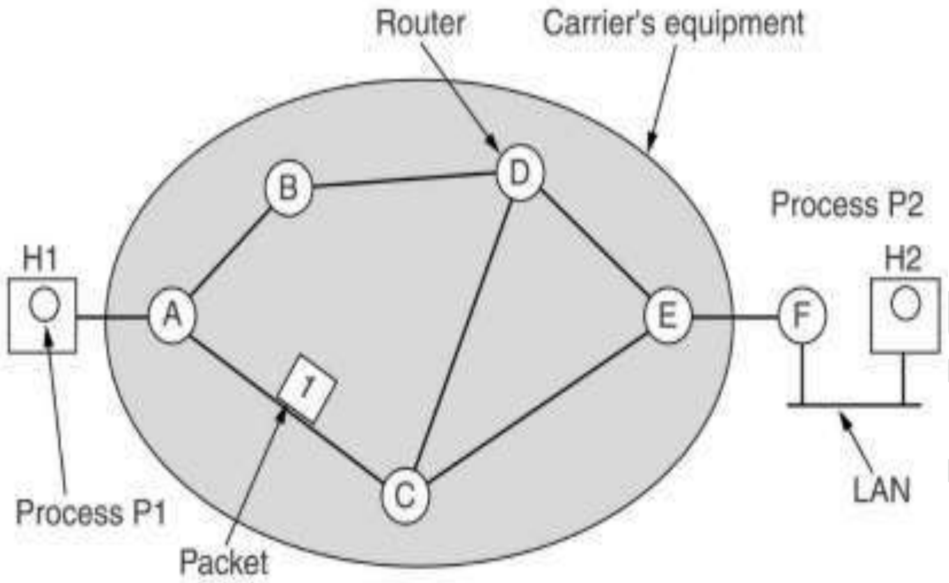


Fig. 4.2Store and Forward Packet Switching

b) Services Provided to the Transport Layer

The network layer services have been designed with the following goals:

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 should be made available to the transport with a uniform numbering plan, even across LANs and WANs.

c) Implementation of Connectionless Service

If connectionless service is offered, packets are injected into the subnet individually and routed independently of each other. No advance setup is needed. In this context, the packets are frequently called **datagrams** and the subnet is called a datagram subnet.

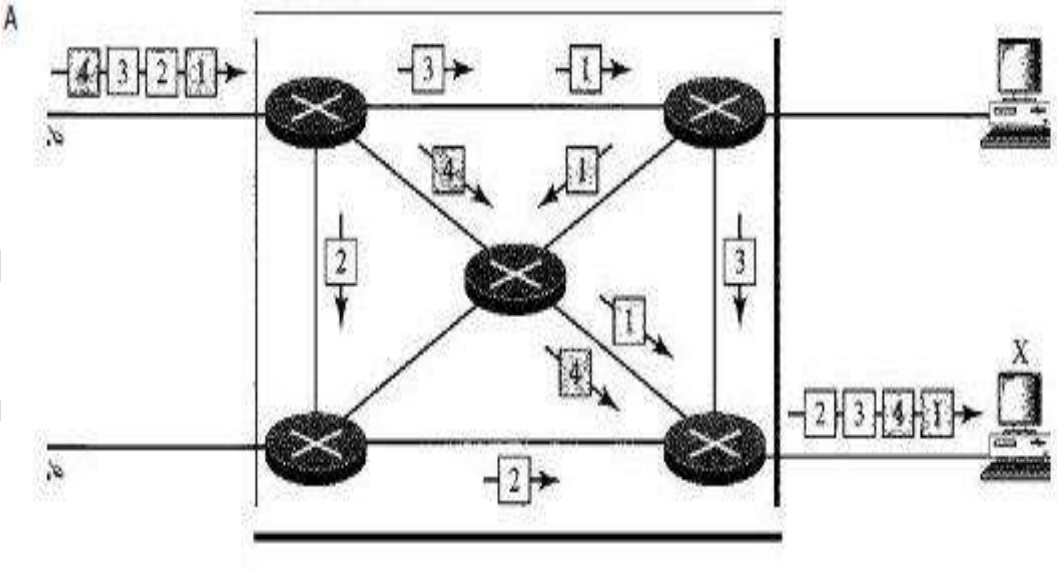


Fig. 4.3Connectionless Service

d) Implementation of Connection-Oriented Service

If connection-oriented service is used, a path from the source router to the destination router must be established before any data packets can be sent. This connection is called a **VC (virtual circuit)** and the subnet is called a virtual-circuit subnet.

The Process is completed in three phase

1. Establishment Phase.
2. Data transfer Phase.
3. Connection release Phase.

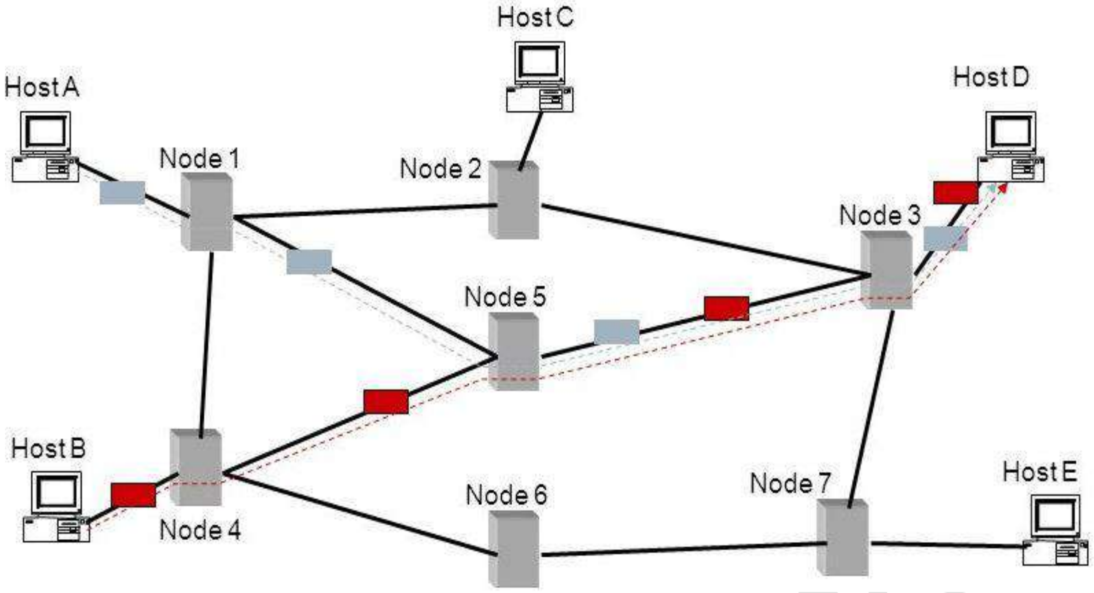


Fig. 4.4Connection-Oriented Service

Comparison of datagram and virtual-circuit subnets

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

Table 4.1 Comparison of datagram and virtual-circuit subnets

Routing algorithms:

A routing algorithm is a set of step-by-step operations used to direct Internet traffic efficiently. When a packet of data leaves its source, there are many different paths it can take to its destination. The routing algorithm is used to determine mathematically the best path to take.

Properties of routing algorithm:

Correctness: The routing should be done properly and correctly so that the packets may reach their proper destination.

Simplicity: The routing should be done in a simple manner so that the overhead is as low as possible. With increasing complexity of the routing algorithms the overhead also increases.

Robustness: Once a major network becomes operative, it may be expected to run continuously for years without any failures. The algorithms designed for routing should be robust enough to handle hardware and software failures and should be able to cope with changes in the topology and traffic without requiring all jobs in all hosts to be aborted and the network rebooted every time some router goes down.

Stability: The routing algorithms should be stable under all possible circumstances.

Fairness: Every node connected to the network should get a fair chance of transmitting their packets. This is generally done on a first come first serve basis.

Optimality: The routing algorithms should be optimal in terms of throughput and minimizing mean packet delays. Here there is a trade-off and one has to choose depending on his suitability.

Routing can be grouped into two categories

1. Adaptive Routing Algorithm: These algorithms change their routing decisions to reflect changes in the topology and in traffic as well. These get their routing information from adjacent routers or from all routers. The optimization parameters are the distance, number of hops and estimated transit time. This can be further classified as follows:

1. Centralized: In this type some central node in the network gets entire information about the network topology, about the traffic and about other nodes. This then transmits this information to the respective routers. The advantage of this is that only one node is required to keep the information. The disadvantage is that if the central node goes down the entire network is down, i.e. single point of failure.

2. Isolated: In this method the node decides the routing without seeking information from other nodes. The sending node does not know about the status of a particular link. The disadvantage is that the packet may be send through a congested route resulting in a delay. Some examples of this type of algorithm for routing are:

a. Hot Potato: When a packet comes to a node, it tries to get rid of it as fast as it can, by putting it on the shortest output queue without regard to where that link leads. A variation of this algorithm is to combine static routing with the hot potato algorithm. When a packet arrives, the routing algorithm takes into account both the static weights of the links and the queue lengths.

b. Backward Learning: In this method the routing tables at each node gets modified by information from the incoming packets. One way to implement backward learning is to include the identity of the source node in each packet, together with a hop counter that is incremented on each hop. When a node receives a packet in a particular line, it notes down the number of hops it has taken to reach it from the source node. If the previous value of hop count stored in the node is better than the current one then nothing is done but if the current value is better than the value is updated for future use. The problem with this is that when the best route goes down then it cannot recall the second best route to a particular node. Hence all the nodes have to forget the stored information periodically and start all over again.

3. Distributed: In this the node receives information from its neighbouring nodes and then takes the decision about which way to send the packet. The disadvantage is that if in between the interval it receives information and sends the packet something changes then the packet may be delayed.

2. Non-Adaptive Routing Algorithm: These algorithms do not base their routing decisions on measurements and estimates of the current traffic and topology. Instead the route to be taken in going from one node to the other is computed in advance, off-line, and downloaded to the routers when the network is booted. This is also known as static routing. This can be further classified as:

1. Flooding: Flooding adapts the technique in which every incoming packet is sent on every outgoing line except the one on which it arrived. One problem with this method is that packets may go in a loop. As a result of this a node may receive several copies of a particular packet which is undesirable. Some techniques adapted to overcome these problems are as follows:

a. Sequence Numbers: Every packet is given a sequence number. When a node receives the packet it sees its source address and sequence number. If the node finds that it has sent the same packet earlier then it will not transmit the packet and will just discard it.

b. Hop Count: Every packet has a hop count associated with it. This is decremented (or incremented) by one by each node which sees it. When the hop count becomes zero (or a maximum possible value) the packet is dropped.

c. Spanning Tree: The packet is sent only on those links that lead to the destination by constructing a spanning tree routed at the source. This avoids loops in transmission but is possible only when all the intermediate nodes have knowledge of the network topology.

Flooding is not practical for general kinds of applications. But in cases where high degree of robustness is desired such as in military applications, flooding is of great help.

2. Random Walk: In this method a packet is sent by the node to one of its neighbours randomly. This algorithm is highly robust. When the network is highly interconnected, this algorithm has the property of making excellent use of alternative routes. It is usually implemented by sending the packet onto the least queued link.

The Optimality Principle

The optimality principle states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route. As a consequence of that 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 tree is called a **sink tree.**

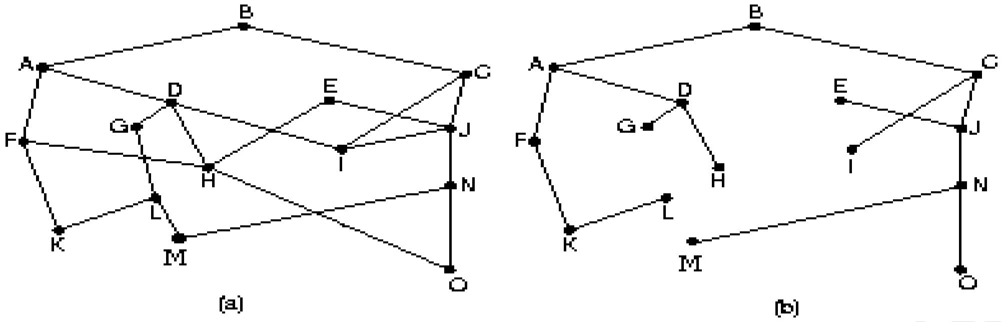


Fig. 4.5 (a) Subnet (b) Sink tree for router B

Shortest Path Algorithm (Least Cost Routing algorithm)

- In this the path length between each node is measured as a function of distance, Bandwidth, average traffic, communication cost, mean queue length, measured delay etc.
- By changing the weighing function, the algorithm then computes the shortest path measured according to any one of a number of criteria or a combination of criteria.
- For this a graph of subnet is drawn. With each node of graph representing a router and each arc of the graph representing a communication link. Each link has a cost associated with it.

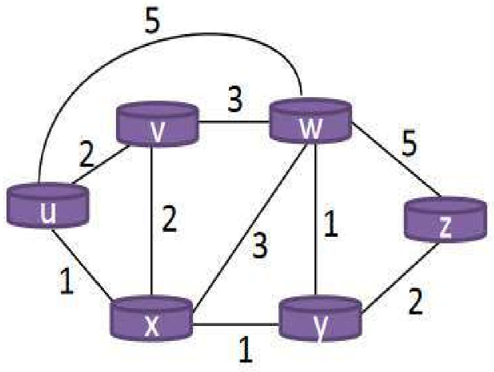
Two algorithms for computing the shortest path between two nodes of a graph are:-

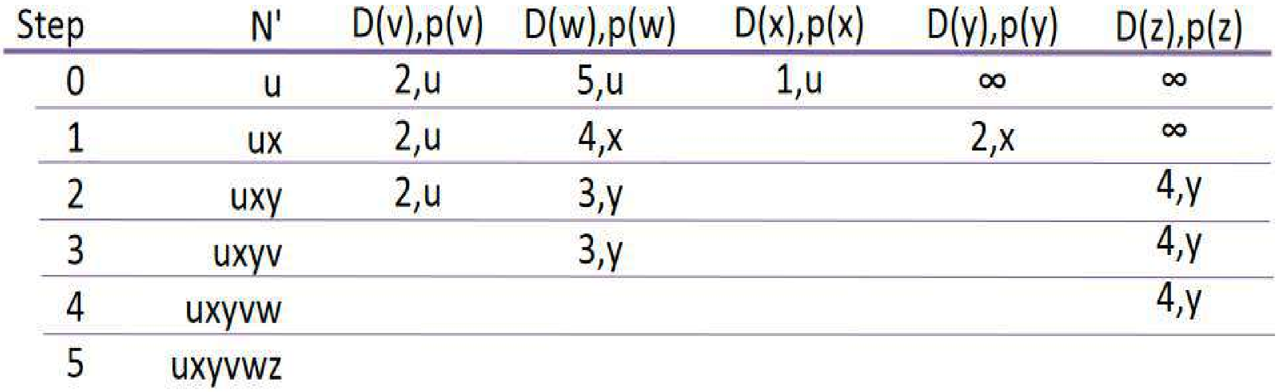
1. Dijkstra's Algorithm 2. Bellnam-Ford Algorithm

1. Dijkstra's algorithm:

1. Compute the least cost path from one node to all other nodes in the network.
2. Iterative algorithm - After the kth iteration, the least cost paths for k destination nodes are found.
3. D(v): cost of the least cost path from source node to destination v
4. p(v): previous node of v along the least-cost path from source.
5. N': set of nodes to which the least-cost path is found.

- Source is node u.





Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					

Bellman-ford algorithm:

Following are the detailed steps.

Input: Graph and a source vertex *src*

Output: Shortest distance to all vertices from *src*. If there is a negative weight cycle, then shortest distances are not calculated, negative weight cycle is reported.

1) This step initializes distances from source to all vertices as infinite and distance to source itself as 0. Create an array dist[] of size |V| with all values as infinite except dist[src] where src is source vertex.

2) This step calculates shortest distances. Do following |V|-1 times where |V| is the number of vertices in given graph.

a) Do following for each edge u-v

If dist[v] >dist[u] + weight of edge uv, then update dist[v]

dist[v] = dist[u] + weight of edge uv

3) This step reports if there is a negative weight cycle in graph. Do following for each edge u-v

If dist[v] >dist[u] + weight of edge uv, then “Graph contains negative weight cycle”

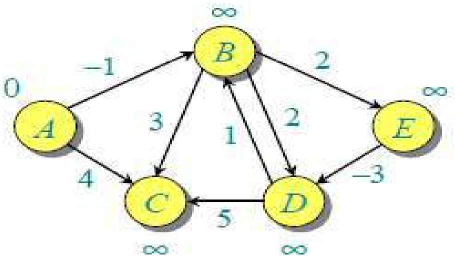
The idea of step 3 is, step 2 guarantees shortest distances if graph doesn’t contain negative weight cycle. If we iterate through all edges one more time and get a shorter path for any vertex, then there is a negative weight cycle

How does this work? Like other Dynamic Programming Problems, the algorithm calculates shortest paths in bottom-up manner. It first calculates the shortest distances which have at-most one edge in the path. Then, it calculates shortest paths with at-most 2 edges, and so on. After the i-th iteration of outer loop, the shortest paths with at most i edges are calculated. There can be maximum |V| – 1 edge in any simple path that is why the outer loop runs |v| – 1 times. The idea is, assuming that there is no negative weight cycle, if we have calculated shortest paths with at most i edges, then an iteration over all edges guarantees to give shortest path with at-most (i+1) edges

Example

let us understand the algorithm with following example graph. The images are taken from this source.

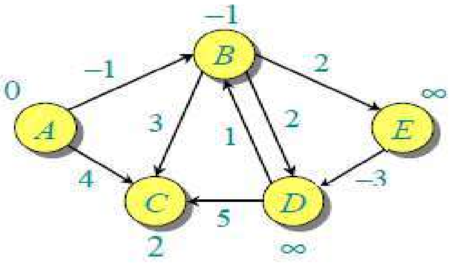
Let the given source vertex be 0. Initialize all distances as infinite, except the distance to source itself. Total number of vertices in the graph is 5, so *all edges must be processed 4 times.*



A	B	C	D	E
0	∞	∞	∞	∞

Fig 4.7Bellman-ford algorithm

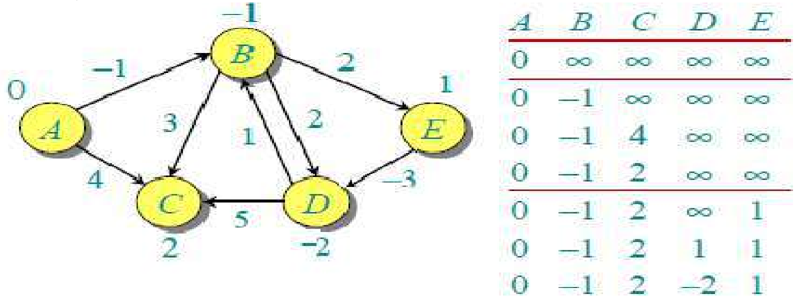
Let all edges are processed in following order: (B,E), (D,B), (B,D), (A,B), (A,C), (D,C), (B,C), (E,D). We get following distances when all edges are processed first time. The first row in shows initial distances. The second row shows distances when edges (B, E), (D,B), (B,D) and (A,B) are processed. The third row shows distances when (A,C) is processed. The fourth row shows when (D,C), (B,C) and (E,D) are processed.



A	B	C	D	E
0	∞	∞	∞	∞
0	−1	∞	∞	∞
0	−1	4	∞	∞
0	−1	2	∞	∞

Fig 4.8Bellman-ford algorithm (Example Step-1)

The first iteration guarantees to give all shortest paths which are at most 1 edge long. We get following distances when all edges are processed second time (The last row shows final values).



A	B	C	D	E
0	∞	∞	∞	∞
0	−1	∞	∞	∞
0	−1	4	∞	∞
0	−1	2	∞	∞
0	−1	2	∞	1
0	−1	2	1	1
0	−1	2	−2	1

Fig 4.9Bellman-ford algorithm (Example Step-2)

The second iteration guarantees to give all shortest paths which are at most 2 edges long. The algorithm processes all edges 2 more times. The distances are minimized after the second iteration, so third and fourth iterations don't update the distances.

Hierarchical Routing:

1. As the number of routers becomes large, the overhead involved in maintaining routing information becomes prohibitive.
2. Internet providers want to manage their network as they wish, while still being able to connect to other networks.
3. Organizing routers into autonomous systems (ASs) solve these problems.
4. Routers within the same AS all run the same routing algorithm (e.g., Dijkstrar DV).Intra-AS routing protocol
5. One or more routers in an AS are responsible to forward packets to destinations outside AS.
6. How to route packets outside an AS?
7. Inter-AS routing protocol: – Obtain reachability information from neighbouring ASs, and Propagate the reachability information to all routers in AS.
8. In the Internet, all ASs run the same inter-AS routing protocol: BGP (Border Gateway Protocol)–Uses a DV-like algorithm.

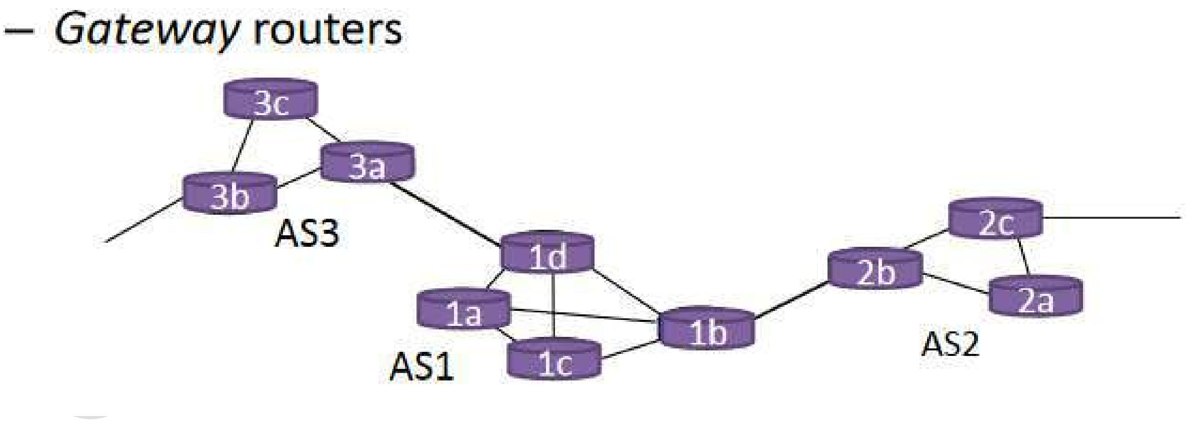


Fig 4.10Hierarchical Routing

Broadcast Routing:

Delivering a packet sent from a source node to all other nodes in the network. By default, the broadcast packets are not routed and forwarded by the routers on any network. Routers create broadcast domains. But it can be configured to forward broadcasts in some special cases. A broadcast message is destined to all network devices.

Broadcast routing can be done in two ways (algorithm):

- A router creates a data packet and then sends it to each host one by one. In this case, the router creates multiple copies of single data packet with different destination addresses. All packets are sent as unicast but because they are sent to all, it simulates as if router is broadcasting.

- This method consumes lots of bandwidth and router must destination address of each node.
- Secondly, when router receives a packet that is to be broadcasted, it simply floods those packets out of all interfaces. All routers are configured in the same way.

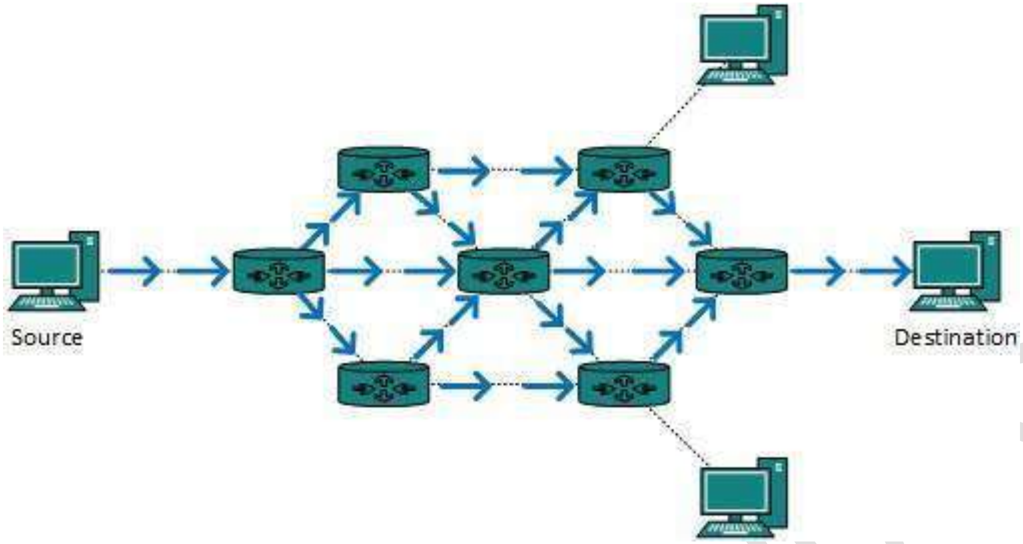


Fig 4.11Broadcast routing

This method is easy on router's CPU but may cause the problem of duplicate packets received from peer routers.

Reverse path forwarding is a technique, in which router knows in advance about its predecessor from where it should receive broadcast. This technique is used to detect and discard duplicates.

Multicast Routing:

Multicast routing is special case of broadcast routing with significance difference and challenges. In broadcast routing, packets are sent to all nodes even if they do not want it. But in Multicast routing, the data is sent to only nodes which wants to receive the packets.

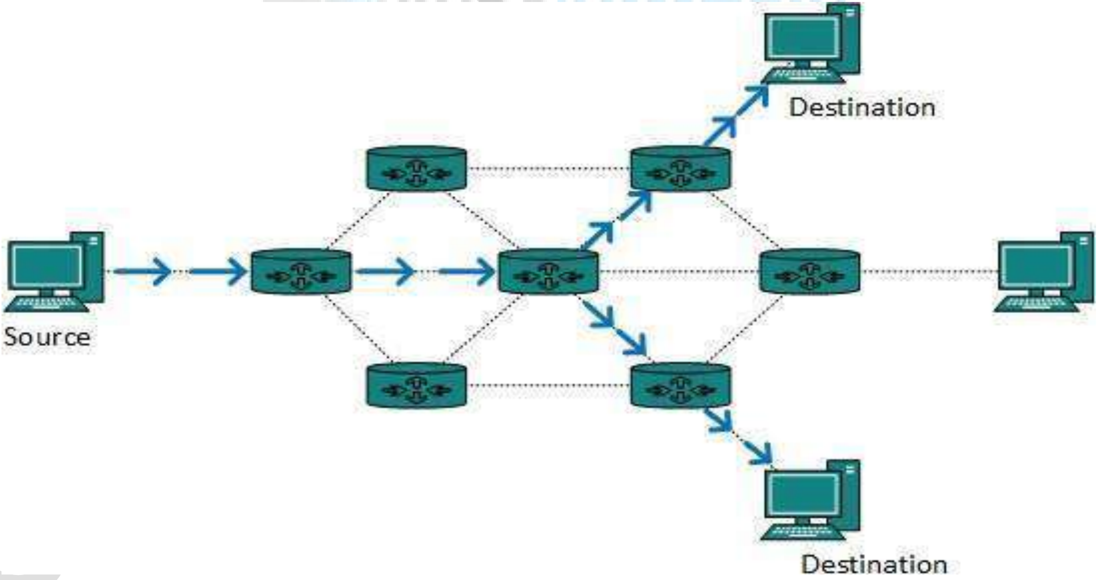


Fig 4.12Multicast routing

The router must know that there are nodes, which wish to receive multicast packets (or stream) then only it should forward. Multicast routing works spanning tree protocol to avoid looping. Multicast routing also uses reverse path Forwarding technique, to detect and discard duplicates and loops.

Unicast routing protocols use graphs while Multicast routing protocols use trees, i.e. spanning tree to avoid loops. The optimal tree is called shortest path spanning tree.

- **DVMRP**- Distance Vector Multicast Routing Protocol
- **MOSPF**- Multicast Open Shortest Path First
- **CBT**- Core Based Tree
- **PIM**- Protocol independent Multicast

Protocol Independent Multicast is commonly used now. It has two flavour's:

- **PIM Dense Mode**
  This mode uses source-based trees. It is used in dense environment such as LAN.
- **PIM Sparse Mode**
  This mode uses shared trees. It is used in sparse environment such as WAN.

Congestion Control Algorithms:

Congestion

A state occurring in network layer when the message traffic is so heavy that it slows down network response time.

Effects of Congestion

- As delay increases, performance decreases.
- If delay increases, retransmission occurs, making situation worse.

General Principles of Congestion Control Principles

1. Many problems in complex systems, such as computer networks, can be viewed from a control theory point of view. This approach leads to dividing all solutions into two groups: open loop and closed loop.
2. Open loop solutions attempt to solve the problem by good design, in essence, to make sure it does not occur in the first place. Once the system is up and running, midcourse corrections are not made.
3. Tools for doing open-loop control include deciding when to accept new traffic, deciding when to discard packets and which ones, and making scheduling decisions at various points in the network. All of these have in common the fact that they make decisions without regard to the current state of the network.
4. In contrast, closed loop solutions are based on the concept of a feedback loop. This approach has three parts when applied to congestion control:
   1. Monitor the system to detect when and where congestion occurs.
   2. Pass this information to places where action can be taken.
   3. Adjust system operation to correct the problem.

Congestion control algorithms

- **Leaky Bucket Algorithm**

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.

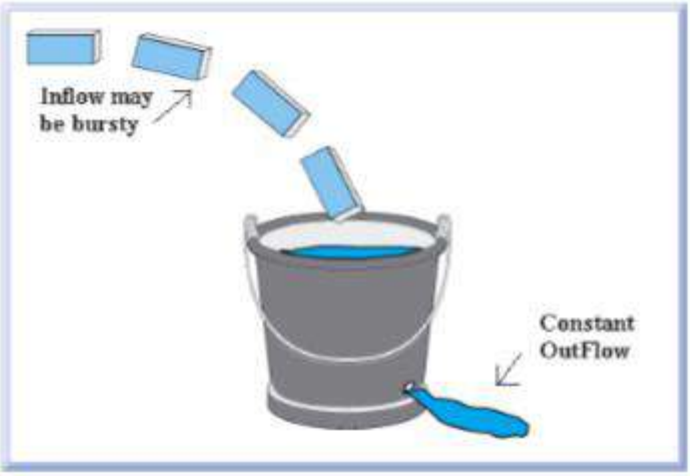


Fig 4.13Leaky bucket algorithm

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.

- **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. ƒ
2. The bucket has a maximum capacity. ƒ
3. If there is a ready packet, a token is removed from the bucket, and the packet is send.
4. If there is no token in the bucket, the packet cannot be send.

Let's understand with an example,

In figure (A) we see a bucket holding three tokens, with five packets waiting to be transmitted. For a packet to be transmitted, it must capture and destroy one token. In figure (B) We see that three of the five packets have gotten through, but the other two are stuck waiting for more tokens to be generated.

Let's understand with an example,

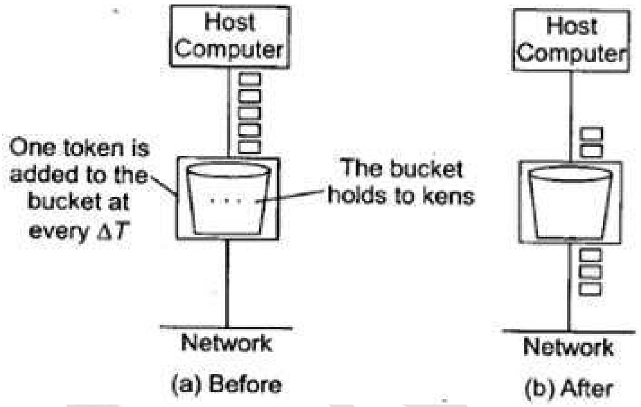
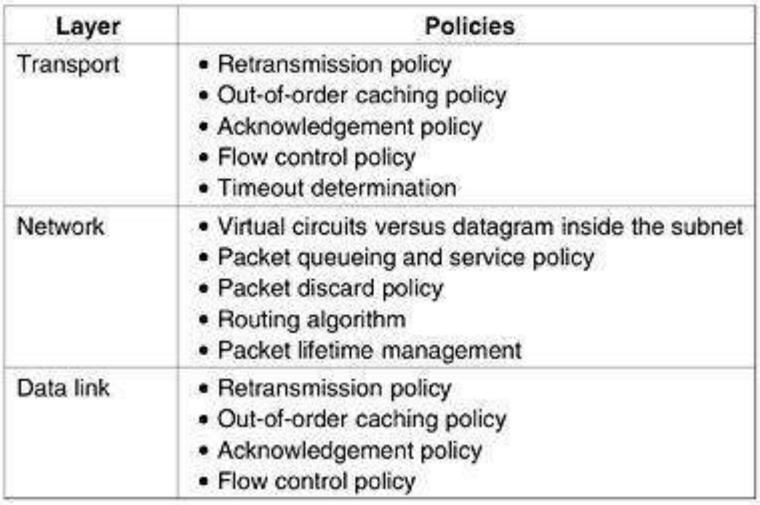


Fig 4.14Token bucket algorithm

Prevention Policies:

These systems are designed to minimize congestion in the first place, rather than letting it happen and reacting after the fact. They try to achieve their goal by using appropriate policies at various levels.

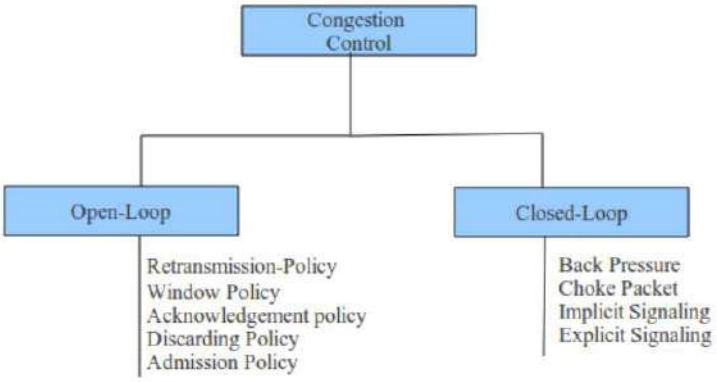


Layer	Policies
Transport	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy • Timeout determination
Network	• Virtual circuits versus datagram inside the subnet • Packet queueing and service policy • Packet discard policy • Routing algorithm • Packet lifetime management
Data link	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy

Table 4.2 Prevention Policies

- A jumpy sender that times out quickly and retransmits all outstanding packets using go back n will put a heavier load on the system than will a leisurely sender that uses selective repeat. Closely related to this is the buffering policy.
- If receivers routinely discard all out-of-order packets, these packets will have to be transmitted again later, creating extra load. With respect to congestion control, selective repeat is clearly better than go back n.
- In the transport layer, the same issues occur as in the data link layer, but in addition, determining the timeout interval is harder because the transit time across the network is less predictable than the transit time over a wire between two routers.

- If the timeout interval is too short, extra packets will be sent unnecessarily. If it is too long, congestion will be reduced but the response time will suffer whenever a packet is lost.



Open Loop Congestion Control:

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

Retransmission Policy

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

Window Policy

The type of window at the sender may also affect congestion. The Selective Repeat window is better than the Go-Back-N window for congestion control. In the Go-Back-N window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver. This duplication may make the congestion worse. The Selective Repeat window, on the other hand, tries to send the specific packets that have been lost or corrupted.

Acknowledgment Policy:

The acknowledgment policy imposed by the receiver may also affect congestion. If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion. Several approaches are used in this case. A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires. A receiver may decide to acknowledge only N packets at a time. We need to know that the acknowledgments are also part of the load in a network. Sending fewer acknowledgments means imposing fewer loads on the network.

Discarding Policy:

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

Admission Policy:

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

Closed-Loop Congestion Control

Closed-loop congestion control mechanisms try to alleviate congestion after it happens. Several mechanisms have been used by different protocols.

Back-pressure:

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

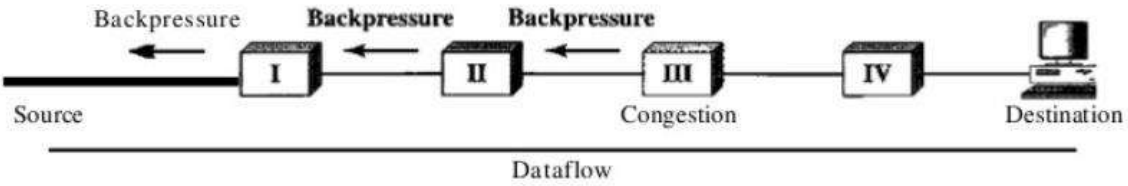


Fig 4.15Back pressure

Node III in the figure has more input data than it can handle. It drops some packets in its input buffer and informs node II to slow down. Node II, in turn, may be congested because it is slowing down the output flow of data. If node II is congested, it informs node I to slow down, which in turn may create congestion. If so, node I inform the source of data to slow down. This, in time, alleviates the congestion. Note that the pressure on node III is moved backward to the source to remove the congestion. None of the virtual-circuit networks we studied in this book use backpressure. It was, however, implemented in the first virtual-circuit network, X.25. The technique cannot be implemented in a datagram network because in this type of network, a node (router) does not have the slightest knowledge of the upstream router.

Choke Packet

A choke packet is a packet sent by a node to the source to inform it of congestion. Note the difference between the backpressure and choke packet methods. In backpressure, the warning is from one node to its upstream node, although the warning may eventually reach the source station. In the choke packet method, the warning is from the router, which has encountered congestion, to the source station directly. The intermediate nodes through which the packet has travelled are not warned. We have seen an example of this type of control in ICMP. When a router in the Internet is overwhelmed datagrams, it may discard some of them; but it informs the source host, using a source quench ICMP message. The warning message goes directly to the source station; the intermediate routers, and does not take any action. Figure shows the idea of a choke packet.

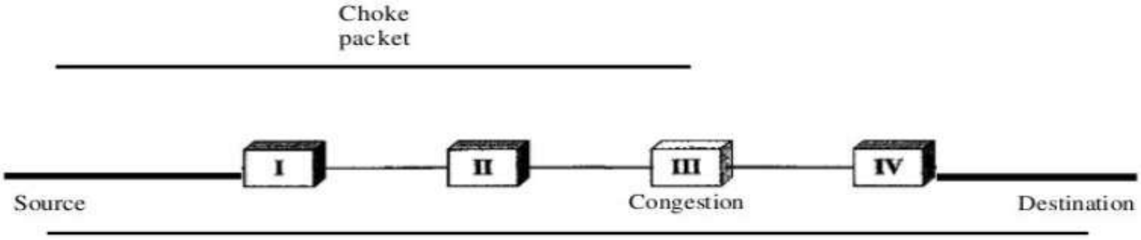


Fig 4.16Choke packet

Implicit Signalling

In implicit signalling, there is no communication between the congested node or nodes and the source. The source guesses that there is congestion somewhere in the network from other symptoms. For example, when a source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested. The delay in receiving an acknowledgment is interpreted as congestion in the network; the source should slow down. We will see this type of signalling when we discuss TCP congestion control later in the chapter.

Explicit Signalling

The node that experiences congestion can explicitly send a signal to the source or destination. The explicit signalling method, however, is different from the choke packet method. In the choke packet method, a separate packet is used for this purpose; in the explicit signalling method, the signal is included in the packets that carry data. Explicit signalling, as we will see in Frame Relay congestion control, can occur in either the forward or the backward direction.

Backward Signalling

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

Forward Signalling

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

Congestion Control in Virtual-Circuit Subnets

1. Admission control: In this approach, once the congestion is signalled, no new connections are set up until the problem is solved. This type of approach is often used in normal telephone networks. When the exchange is overloaded, then no new calls are established.
2. Allow new virtual connections other than the congested area.
3. Negotiate an agreement between the host and the network when the connection is setup. This agreement specifies the volume and shape of traffic, quality of service, maximum delay and other parameters. The network will reserve resources (Buffer space, Bandwidth and CPU cycle) along the path when the connection is set up. Now congestion is unlikely to occur on the new connections because all the necessary resources are guaranteed to be available. The disadvantage of this approach is that it may leads to wasted bandwidth because of some idle connection.

Congestion Control in Datagram subnets

Congestion control in Datagram Subnets is achieved by sending warning to sender in advance. Each router can easily monitor the utilization of its output lines. If utilization is greater than threshold value then output line may be congested in future so mark it as warning state. Each newly arriving packet is checked to see if its output line is in warning state. If it is, some action is taken. The actions are:

1. The warning bit
2. Choke packets
3. Hop-by-hop choke packet

1. The warning bit

When a new packet is to be transmitted on the output line marked as warning state, a special bit is added in header to signal this state. At the destination, this information is sent back with ACK to the sender so that it could cut the traffic. When warning bit is absent, sender increases its transmitting rate.

Note: It uses a whole trip (source -> destination -> source) to tell the source to slow down.

2. Choke packets

In this approach, the router sends a choke packet back to the source host. The original packet is marked so that it would not generate any more choke packets further along the path and is then forwarded in the usual way. When the source gets the choke packet, it is required to reduce the traffic by X packets.

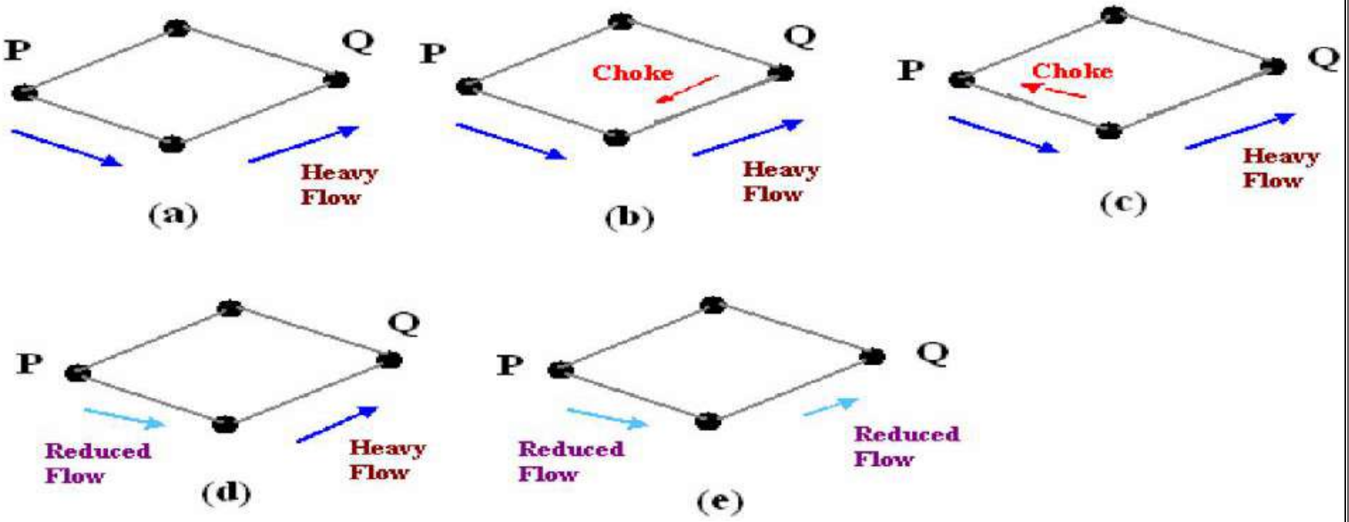


Fig 4.17Functioning of choke packets, (a) Heavy traffic between nodes P and Q, (b) Node Q sends the Choke packet to P, (c) Choke packet reaches P, (d) P reduces the flow and sends a reduced flow out, (e) Reduced flow reaches node Q.

Problem: It does not work well if the choke packet travels a long distance to reach the source because reduction of flow starts from source node rather than intermediate node. This problem can be solved by hop-by-hop approach.

3. Hop-by-hop choke packet

In this approach, unlike choke packet, reduction of flow starts from intermediate node rather than source node. To understand this, let us refer the figure 2. When the choke packet reaches the nearest router (say R) from router Q, it reduces the flow. However, router R now requires devoting more buffers to the flow since the source is still sending at full blast but it gives router Q immediate relief. In the next step, the choke packet reaches P and flow genuinely slow down. The net effect of hop-by-hop scheme is to provide quick relief at the point of congestion at the price of using up more buffers upstream.

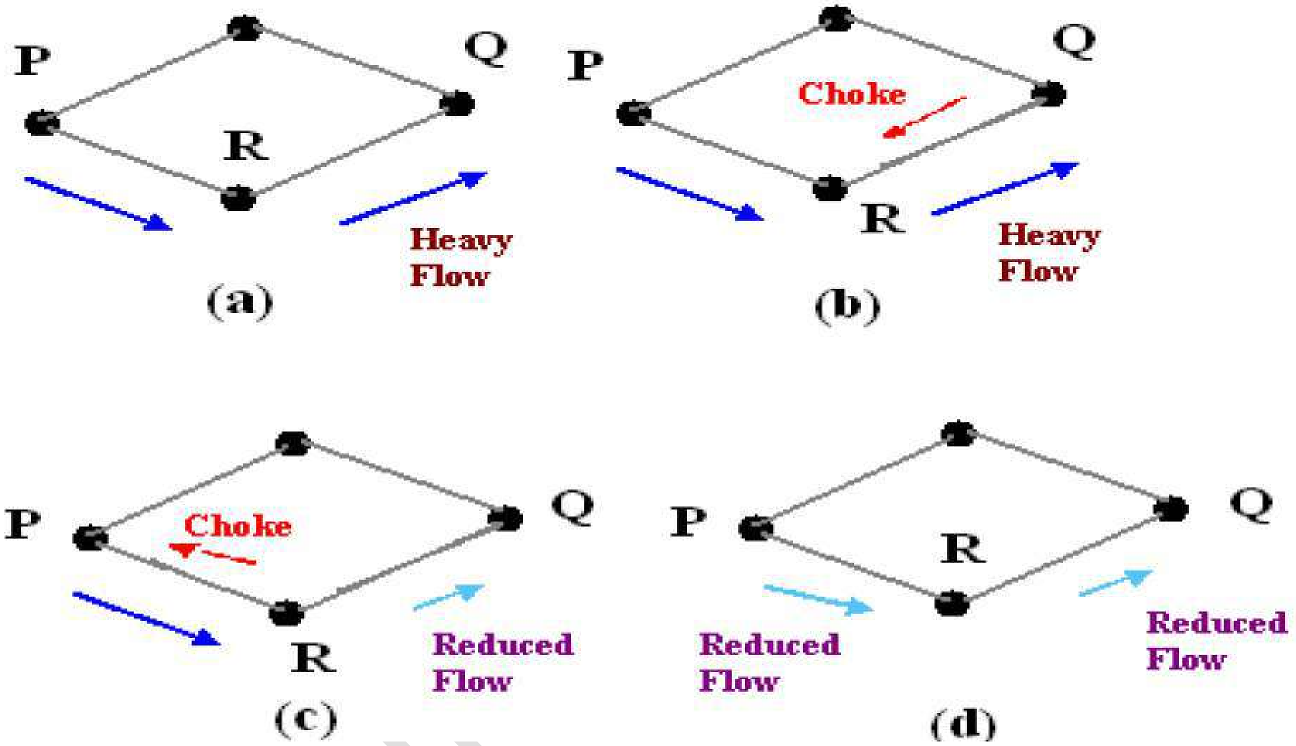


Fig 4.18Functioning of Hop-by-Hop choke packets, (a) Heavy traffic between nodes P and Q, (b) Node Q sends the Choke packet to P, (c) Choke packet reaches R, and the flow between R and Q is decreased, (d) Choke packer reaches P, and P reduces the flow out.

IP protocol:

Internet Protocol (IP) is the principal set (or communications protocol) of digital message formats and rules for exchanging messages between computers across a single network or a series of interconnected networks, using the Internet Protocol Suite (often referred to as TCP/IP). Messages are exchanged as datagrams, also known as data packets or just packets.

IP is the primary protocol in the Internet Layer of the Internet Protocol Suite, which is a set of communications protocols consisting of four abstraction layers: link layer (lowest), Internet layer, transport layer and application layer (highest).

The main purpose and task of IP is the delivery of datagrams from the source host (source computer) to the destination host (receiving computer) based on their addresses. To achieve this, IP includes methods and structures for putting tags (address information, which is part of metadata) within datagrams. The process of putting these tags on datagrams is called encapsulation.

Think of an analogy with the postal system. IP is similar to the U.S. Postal System in that it allows a package (a datagram) to be addressed (encapsulation) and put into the system (the Internet) by the sender (source host). However, there is no direct link between sender and receiver.

The package (datagram) is almost always divided into pieces, but each piece contains the address of the receiver (destination host). Eventually, each piece arrives at the receiver, often by different routes and at different times. These routes and times are also determined by the Postal System, which is the IP. However, the Postal System (in the transport and application layers) puts all the pieces back together before delivery to the receiver (destination host).

Note: IP is actually a connectionless protocol, meaning that the circuit to the receiver (destination host) does not need be set up before transmission (by the source host). Continuing the analogy, there does not need to be a direct connection between the physical return address on the letter/package and the recipient address before the letter/package is sent.

When format and rules were applied to allow connections, the connection-oriented Transmission Control Protocol was created. The two together forms the Internet Protocol Suite, often referred to as TCP/IP.

Internet Protocol version 4 (IPv4) was the first major version of IP. This is the dominant protocol of the Internet. However, iPv6 is active and in use, and its deployment is increasing all over the world.

Addressing and routing are the most complex aspects of IP. However, intelligence in the network is located at nodes (network interconnection points) in the form of routers which forward datagrams to the next known gateway on the route to the final destination. The routers use interior gateway protocols (IGPs) or external gateway protocols (EGPs) to help with making forwarding route decisions. Routes are determined by the routing prefix within the datagrams. The routing process can therefore become complex. But at the speed of light (or nearly so) the routing intelligence determines the best route, and the datagram pieces and datagram all eventually arrive at their destination.

IP Address

An Internet Protocol address (IP address) is a logical numeric address that is assigned to every single computer, printer, switch, router or any other device that is part of a TCP/IP-based network.

The IP address is the core component on which the networking architecture is built; no network exists without it. An IP address is a logical address that is used to uniquely identify every node in the network. Because IP addresses are logical, they can change. They are similar to addresses in a town or city because the IP address gives the network node an address so that it can communicate with other nodes or networks, just like mail is sent to friends and relatives.

The numerals in an IP address are divided into 2 parts:

- The network part specifies which networks this address belongs to and
- The host part further pinpoints the exact location.

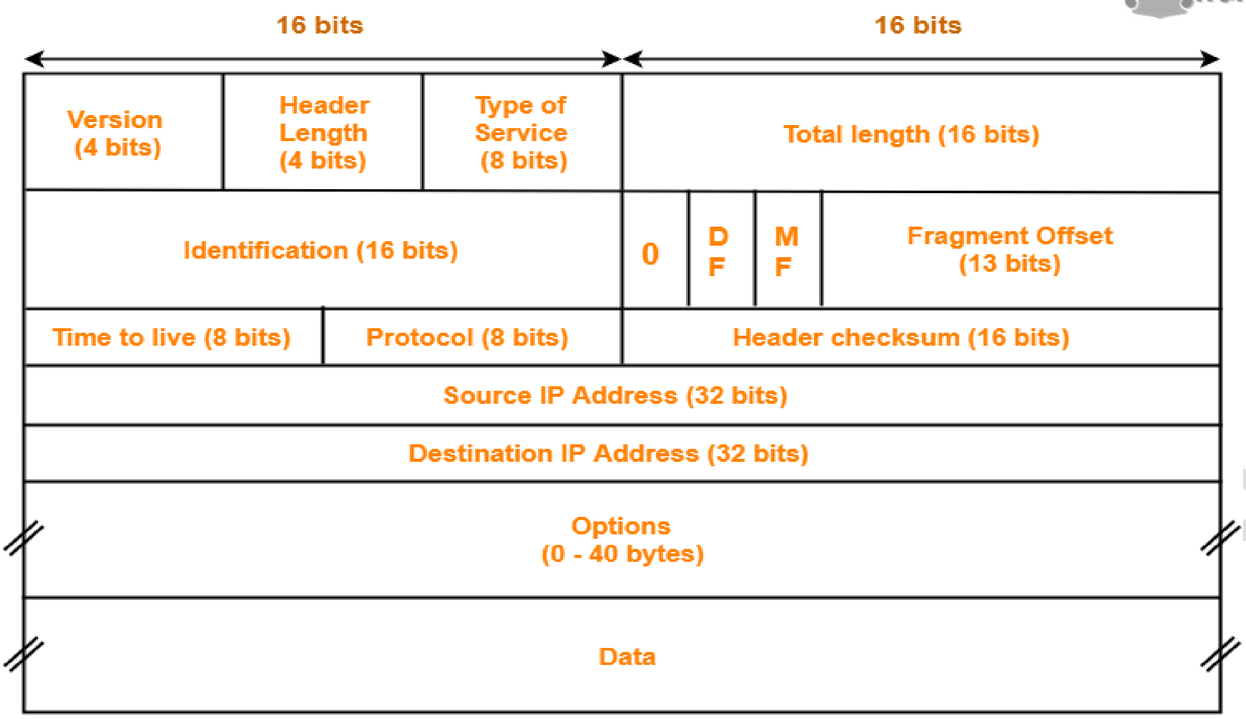
An IP address is the most significant and important component in the networking phenomena that binds the World Wide Web together. The IP address is a numeric address assigned to every unique instance that is connected to any computer communication network using the TCP/IP communication protocols.

Network nodes are assigned IP addresses by the Dynamic Host Configuration Protocol server as soon as the nodes connect to a network. DHCP assigns IP addresses using a pool of available addresses which are part of the whole addressing scheme. Though DHCP only provides addresses that are not static, many machines reserve static IP addresses that are assigned to that entity forever and cannot be used again.

IP addresses falls into two types:

- Classfull IP addressing is a legacy scheme which divides the whole IP address pools into 5 distinct classes—A, B, C, D and E.
- Classless IP addressing has an arbitrary length of the prefixes.

Header format of IPv4:



IPv4 Header

Fig:4.19 IPV4 Header

VERSION: Version of the IP protocol (4 bits), which is 4 for IPv4

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: Low Delay, High Throughput, Reliability (8 bits)

Total Length: Length of header + Data (16 bits), which has a minimum value 20 bytes and the maximum is 65,535 bytes.

Identification: Unique Packet Id for identifying the group of fragments of a single IP datagram (16 bits)

Flags: 3 flags of 1 bit each : reserved bit (must be zero), do not fragment flag, more fragments flag (same order)

Fragment Offset: Represents the number of Data Bytes ahead of the particular fragment in the particular Datagram. Specified in terms of number of 8 bytes, which has the maximum value of 65,528 bytes.

Time to live: Datagram's lifetime (8 bits), It prevents the datagram to loop through the network by restricting the number of Hops taken by a Packet before delivering to the Destination.

Protocol: Name of the protocol to which the data is to be passed (8 bits)

Header Checksum: 16 bits header checksum for checking errors in the datagram header.

Source IP address: 32 bits IP address of the sender

Destination IP address: 32 bits IP address of the receiver

Option: Optional information such as source route, record route. Used by the Network administrator to check whether a path is working or not.

Due to the presence of options, the size of the datagram header can be of variable length (20 bytes to 60 bytes).

Header format of IPV6:

Version (4-bits) : Indicates version of Internet Protocol which contains bit sequence 0110.

Traffic Class (8-bits) : The Traffic Class field indicates class or priority of IPv6 packet which is similar to *Service Field* in IPv4 packet. It helps routers to handle the traffic based on priority of the packet. If congestion occurs on router then packets with least priority will be discarded.

As of now only 4-bits are being used (and remaining bits are under research), in which 0 to 7 are assigned to Congestion controlled traffic and 8 to 15 are assigned to Uncontrolled traffic.

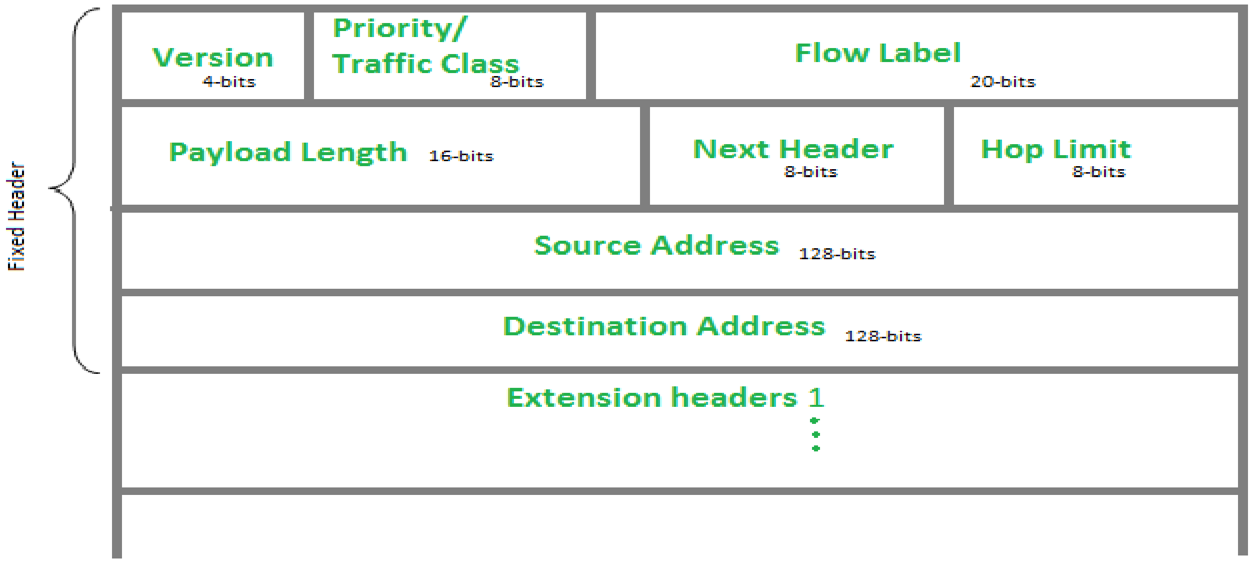


Fig:4.20 IPV6 Header

Flow Label (20-bits) : Flow Label field is used by source to label the packets belonging to the same flow in order to request special handling by intermediate IPv6 routers, such as non-default quality of service or real time service. In order to distinguish the flow, intermediate router can use source address, destination address and flow label of the packets. Between a source and destination multiple flows may exist because many processes might be running at the same time. Routers or Host that do not support the functionality of flow label field and for default router handling, flow label field is set to 0. While setting up the flow label, source is also supposed to specify the lifetime of flow.

Payload Length (16-bits) : It is a 16-bit (unsigned integer) field, indicates total size of the payload which tells routers about amount of information a particular packet contains in its payload. Payload Length field includes extension headers(if any) and upper layer packet. In case length of payload is greater than 65,535 bytes (payload up to 65,535 bytes can be indicated with 16-bits), then the payload length field will be set to 0 and jumbo payload option is used in the Hop-by-Hop options extension header.

Next Header (8-bits) : Next Header indicates type of extension header(if present) immediately following the IPv6 header. Whereas In some cases it indicates the protocols contained within upper-layer packet, such as TCP, UDP.

Hop Limit (8-bits) : Hop Limit field is same as TTL in IPv4 packets. It indicates the maximum number of intermediate nodes IPv6 packet is allowed to travel. Its value gets decremented by one, by each node that forwards the packet and packet is discarded if value decrements to 0. This is used to discard the packets that are stuck in infinite loop because of some routing error.

Source Address (128-bits) : Source Address is 128-bit IPv6 address of the original source of the packet.

Destination Address (128-bits) : Destination Address field indicates the IPv6 address of the final destination(in most cases). All the intermediate nodes can use this information in order to correctly route the packet.

Extension Headers : In order to rectify the limitations of *IPv4 Option Field*, Extension Headers are introduced in IPversion 6. The extension header mechanism is very important part of the IPv6 architecture. Next Header field of IPv6 fixed header points to the first Extension Header and this first extension header points to the second extension header and so on.

Comparative study of IPv4 & IPv6:

Sl. No.	IPv4	IPv6
1	Addresses are 32 bits (4 bytes) in length.	Addresses are 128 bits (16 bytes) in length.
2	Address (A) resource records in DNS to map host names to IPv4 addresses.	Address (AAAA) resource records in DNS to map host names to IPv6 addresses.
3	Pointer (PTR) resource records in the IN-ADDR.ARPA DNS domain to map IPv4 addresses to host names.	Pointer (PTR) resource records in the IP6.ARPA DNS domain to map IPv6 addresses to host names.
4	IPSec is optional and should be supported externally	IPSec support is not optional
5	Header does not identify packet flow for QoS handling by routers	Header contains Flow Label field, which Identifies packet flow for QoS handling by router.
6	Both routers and the sending host fragment packets.	Routers do not support packet fragmentation. Sending host fragments packets
7	Header includes a checksum.	Header does not include a checksum.
8	ARP uses broadcast ARP request to resolve IP to MAC/Hardware address.	Multicast Neighbour Solicitation messages resolve IP addresses to MAC addresses.
9	Internet Group Management Protocol (IGMP) manages membership in local subnet groups	Multicast Listener Discovery (MLD) messages manage membership in local subnet groups.
10	Broadcast addresses are used to send traffic to all nodes on a subnet.	IPv6 uses a link-local scope all-nodes multicast address.
11	Configured either manually or through DHCP.	Does not require manual configuration or DHCP.
12	Must support a 576-byte packet size (possibly fragmented).	Must support a 1280-byte packet size (without fragmentation).

Table 4.3 Comparison between IPv4 and IPv6

Packet Forwarding:

Packet forwarding is done when uIP receives a packet that has a destination IP address that does not match any of the IP addresses of the node. A node typically has multiple addresses: one or more unicast addresses and at least one broadcast or multicast address. Packets that do not match the addresses should be forwarded to a neighboring node, either because the address matches that of the neighbor or because the neighbor has a route to the destination address.

Packet forwarding occurs only when uIP has been configured to be a router. The packet forwarding mechanism is then invoked as part of the output processing.

The packet forwarding mechanism is modular and does not specify any particular routing mechanism to be used. Rather, a routing mechanism will register itself with the forwarding module upon startup. For every packet, the forwarding mechanism asks the routing module to look up the destination IP address and return the address to the next-hop neighbor. The routing module may implement this any way it wants by using a table of destination addresses, a table of network prefixes, a hash table of addresses, a cache of the recently used routes, or any other way it finds suitable. The routing protocol may perform a route discovery for each address not found in its cache.

By separating packet forwarding and packet routing, uIP can adapt a wide range of requirements such as routing performance and memory requirements, as well as take advantage of future development in routing protocols. A system with strict memory requirements and low routing performance requirements may use a cache configuration that prompts frequent network route discoveries, whereas a system with strict requirements on routing performance but lax memory requirements may choose a larger cache setting.

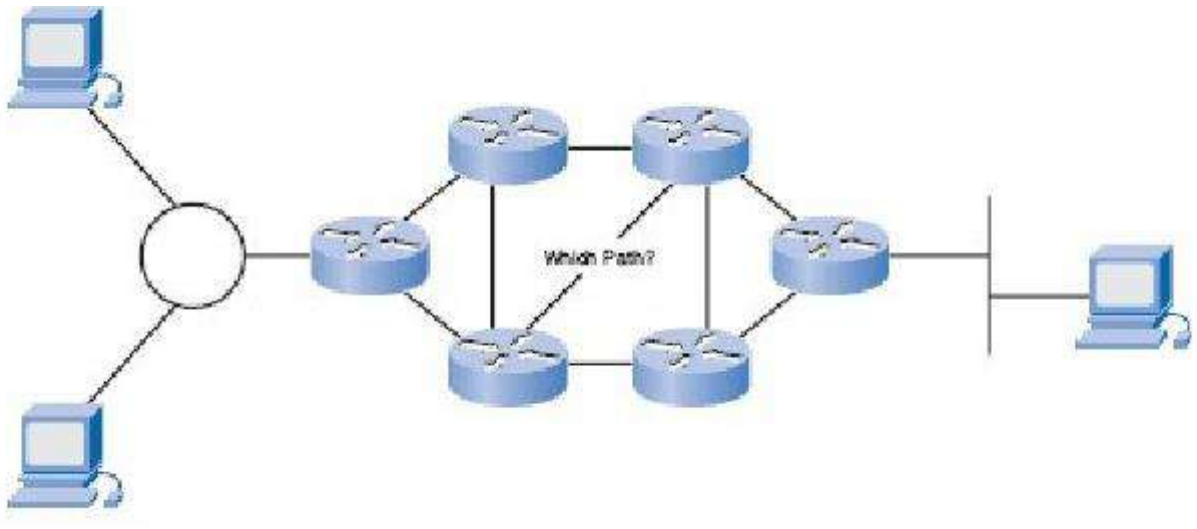


Fig. 4.21 Packet forwarding

Fragmentation:

Fragmentation is the process of breaking a packet into smaller pieces so that they will fit into the frames of the underlying network. The receiving system reassembles the pieces into the original packets. The term MTU (maximum transmission unit) refers to the maximum amount of data that can travel in a frame. Different networks have different MTU sizes, so packets may need to be fragmented in order to fit within the frames of the network that they transit.

Internetworking protocols such as IP use fragmentation because each of the networks that a packet may travel over could have a different frame size. Fragmentation occurs at routers that connect two networks with different MTUs. While it is possible to design an internal network with the same MTU size, this is not an option on the Internet, which includes thousands of independently managed interconnected networks. Fragmentation is always undesirable because it reduces performance. In fact, fragmentation is not allowed in IPv6. Large packets are always preferable.

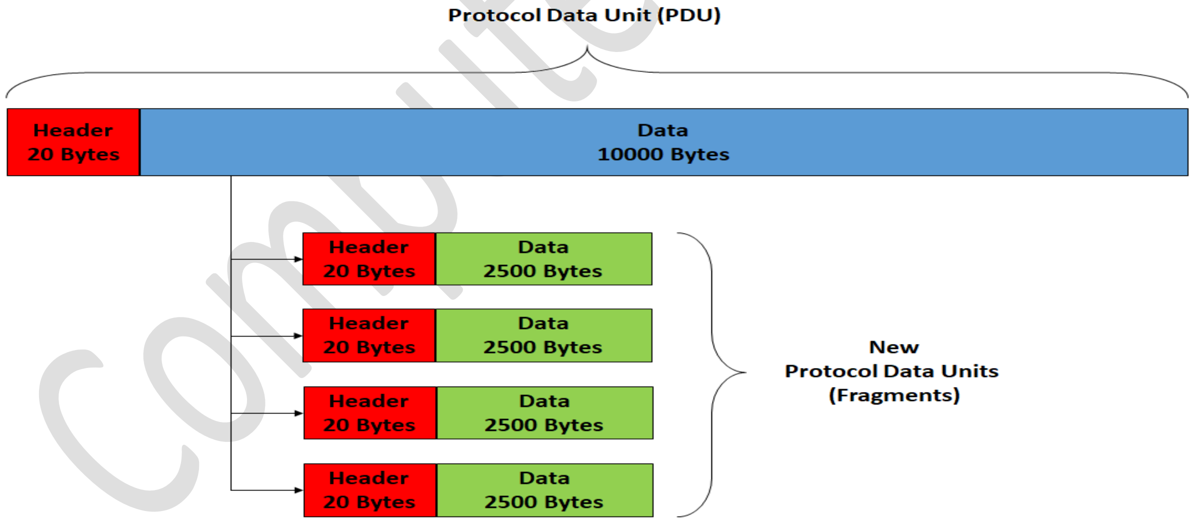


Fig:4. 22 Fragmentations

Reassembly is the reverse of segmentation. Protocol Data Units are put back together in the correct order to reassemble a stream of data in its original form.

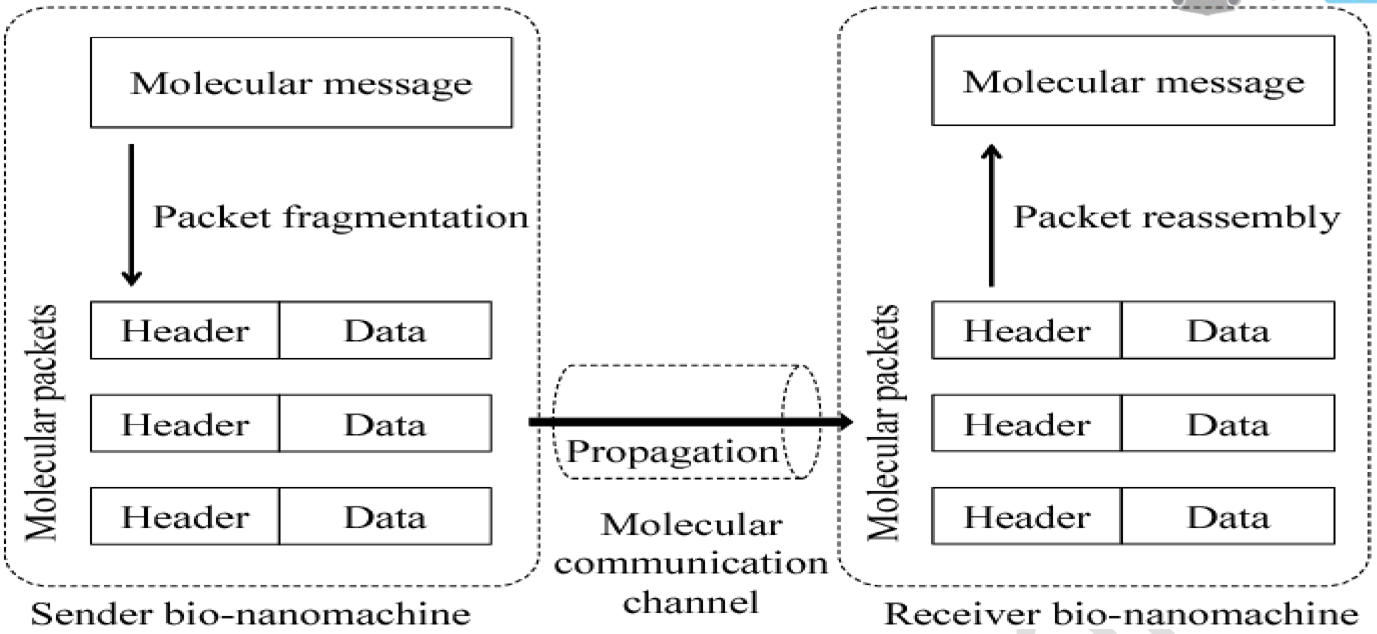


Fig:4.23 Packet fragmentation and Packet reassembly

ICMP:

ICMP (Internet Control Message Protocol) is an error-reporting protocol network devices like routers use to generate error messages to the source IP address when network problems prevent delivery of IP packets. ICMP creates and sends messages to the source IP address indicating that a gateway to the Internet that a router, service or host cannot be reached for packet delivery. Any IP network device has the capability to send, receive or process ICMP messages.

ICMP is *not* a transport protocol that sends data between systems.

While ICMP is not used regularly in end-user applications, it is used by network administrators to troubleshoot Internet connections in diagnostic utilities including ping and tracer route.

One of the main protocols of the IP suite, ICMP is used by routers, intermediary devices or hosts to communicate error information or updates to other routers, intermediary devices or hosts. The widely used IPv4 (Internet Protocol version 4) and the newerIPv6 use similar versions of the ICMP protocol (ICMPv4 and ICMPv6, respectively).

ICMP messages are transmitted as datagram’s and consist of an IP header that encapsulates the ICMP data. ICMP packets are IP packets with ICMP in the IP data portion. ICMP messages also contain the entire IP header from the original message, so the end system knows which packet failed

The ICMP header appears after the IPv4 or IPv6 packet header and is identified as IP protocol number 1. The complex protocol contains three fields:

- The major type that identifies the ICMP message;
- The minor code that contains more information about the type field; and
- The checksum that helps detect errors introduced during transmission.

Following the three fields is the ICMP data and the original IP header to identify which packets actually failed.

ICMP has been used to execute denial-of-services attacks (also called the ping of death) by sending an IP packet larger than the number of bytes allowed by the IP protocol.