UNIT -III

Network Layer

3.1 Virtual and Datagram Circuits

Computer networks that provide connection-oriented service are called Virtual Circuits while those providing connection-less services are called as Datagram networks.

Virtual Circuits:

1. It is connection-oriented simply meaning that there is a reservation of resources like buffers, CPU, bandwidth, etc. for the time in which the newly setup VC is going to be used by a data transfer session.
2. First packet goes and reserves resources for the subsequent packets which as a result follow the same path for the whole connection time.
3. Since all the packets are going to follow the same path, a global header is required only for the first packet of the connection and other packets generally don't require global headers.
4. Since data follows a particular dedicated path, packets reach inorder to the destination.
5. From above points, it can be concluded that Virtual Circuits are highly reliable means of transfer.
6. Since each time a new connection has to be setup with reservation of resources and extra information handling at routers, it's simply costly to implement Virtual Circuits.

Datagram Networks:

1. It is connectionless service. There is no need of reservation of resources as there is no dedicated path for a connection session.
2. All packets are free to go to any path on any intermediate router which is decided on the go by dynamically changing routing tables on routers.
3. Since every packet is free to choose any path, all packets must be associated with a header with proper information about source and the upper layer data.
4. The connectionless property makes data packets reach destination in any order, means they need not reach in the order in which they were sent.
5. Datagram networks are not reliable as Virtual Circuits.
6. But it is always easy and cost efficient to implement datagram networks as there is no extra headache of reserving resources and making a dedicated each time an application has to communicate.

3.2 Routing

Routing is the process of forwarding of a packet in a network so that it reaches its intended destination. The main goals of routing are:

1. **Correctness:** The routing should be done properly and correctly so that the packets may reach their proper destination.
2. **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.
3. **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.
4. **Stability:** The routing algorithms should be stable under all possible circumstances.
5. **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.
6. **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.

Classification of Routing Algorithms

The routing algorithms may be classified as follows:

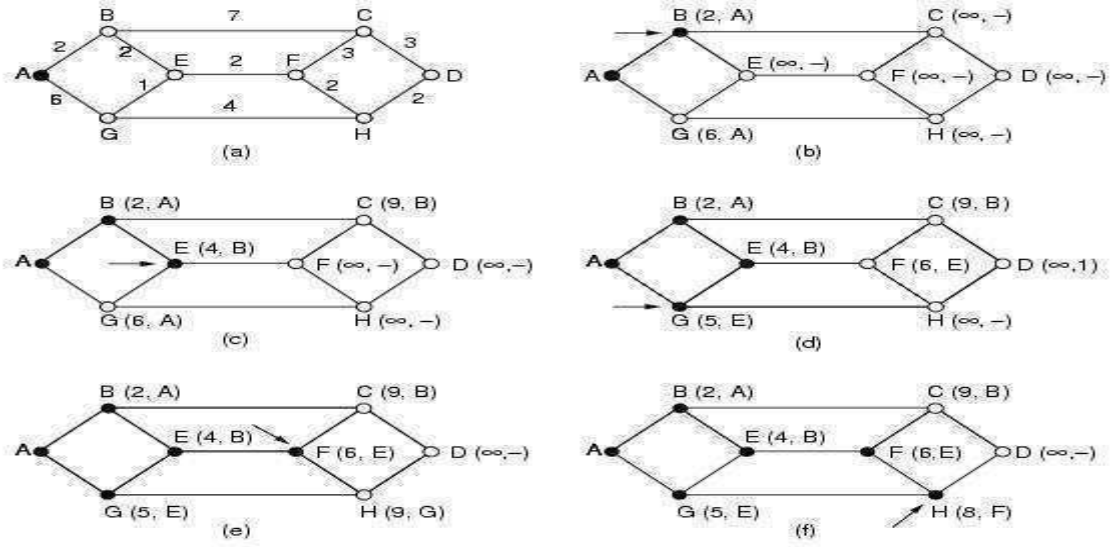
1. **Adaptive Routing Algorithm:** These algorithms change their routing decisions to reflect changes in the topology and in traffic as well. These get their routing information from adjacent routers or from all routers. The optimization parameters are the distance, number of hops and estimated transit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

3.3 Dijkstra's Routing Algorithm

At the end each node will be labelled (see Figure) with its distance from source node along the best known path. Initially, no paths are known, so all nodes are labelled with infinity. As the algorithm proceeds and paths are found, the labels may change reflecting better paths. Initially, all labels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

Look at the weighted undirected graph of Figure(a), where the weights represent, for example, distance. We want to find shortest path from A to D. We start by making node A as permanent, indicated by a filled in circle. Then we examine each of the nodes adjacent to A (the working node), relabeling each one with the distance to A. Whenever a node is relabelled, we also label it with the node from which the probe was made so that we can construct the final path later. Having examined each of the nodes adjacent to A, we examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent, as shown in Figure (b). This one becomes new working node.

We now start at B, and examine all nodes adjacent to it. If the sum of the label on B and the distance from B to the node being considered is less than the label on the node, we have a shorter path, so the node is relabelled. After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively labelled node with the smallest value. This node is made permanent and becomes the working node for the next round. The Figure shows the first five steps of the algorithm.



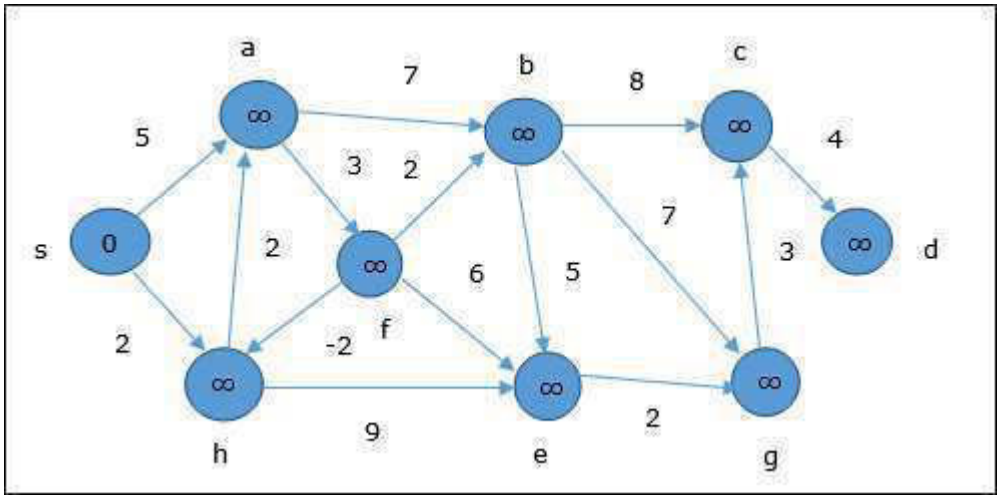
3.4 Bellman Ford Algorithm

This algorithm solves the single source shortest path problem of a directed graph ***G = (V, E)*** in which the edge weights may be negative. Moreover, this algorithm can be applied to find the shortest path, if there does not exist any negative weighted cycle.

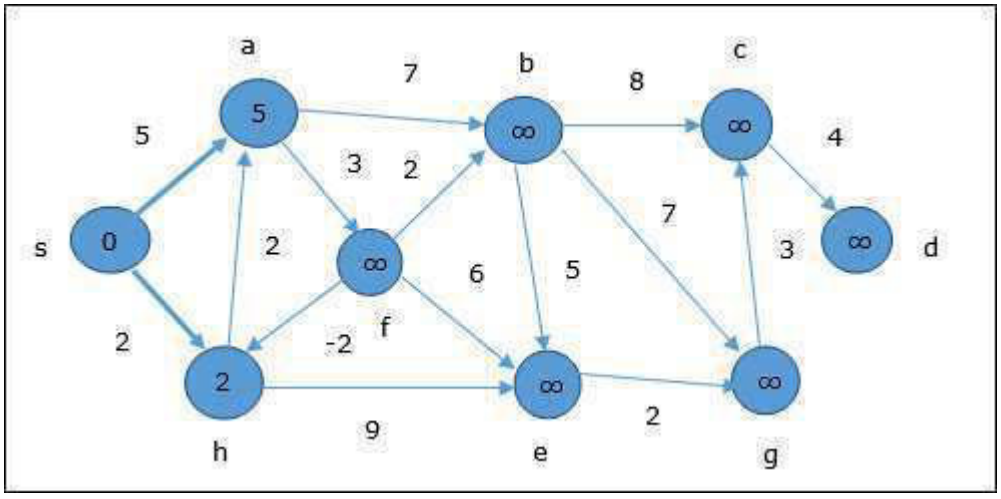
Example

The following example shows how Bellman-Ford algorithm works step by step. This graph has a negative edge but does not have any negative cycle, hence the problem can be solved using this technique.

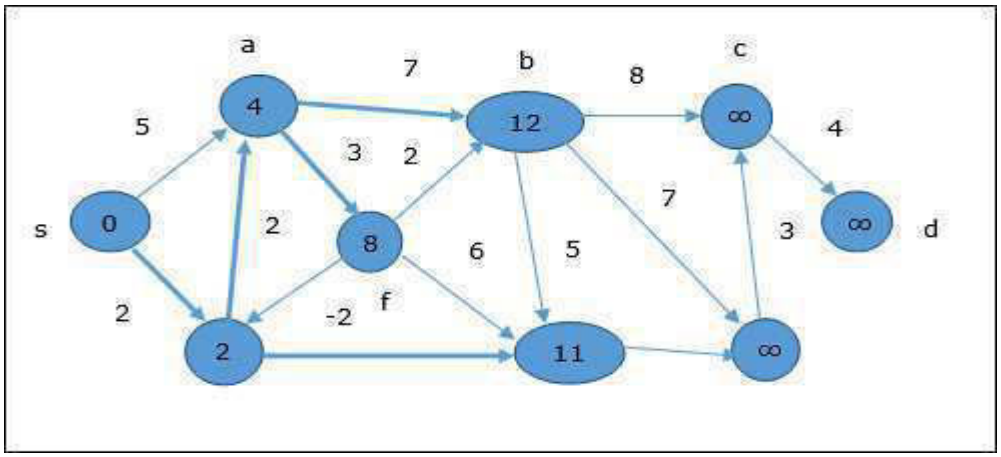
At the time of initialization, all the vertices except the source are marked by ∞ and the source is marked by **0**.



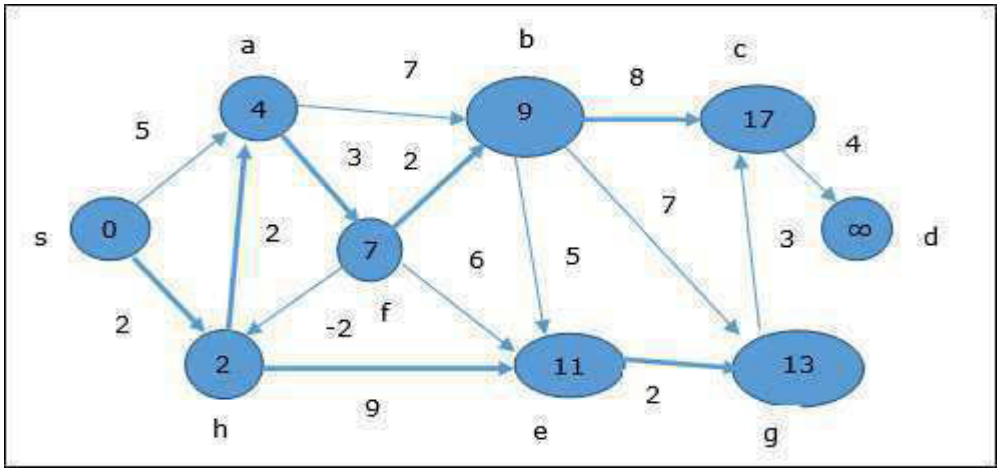
In the first step, all the vertices which are reachable from the source are updated by minimum cost. Hence, vertices ***a*** and ***h*** are updated.



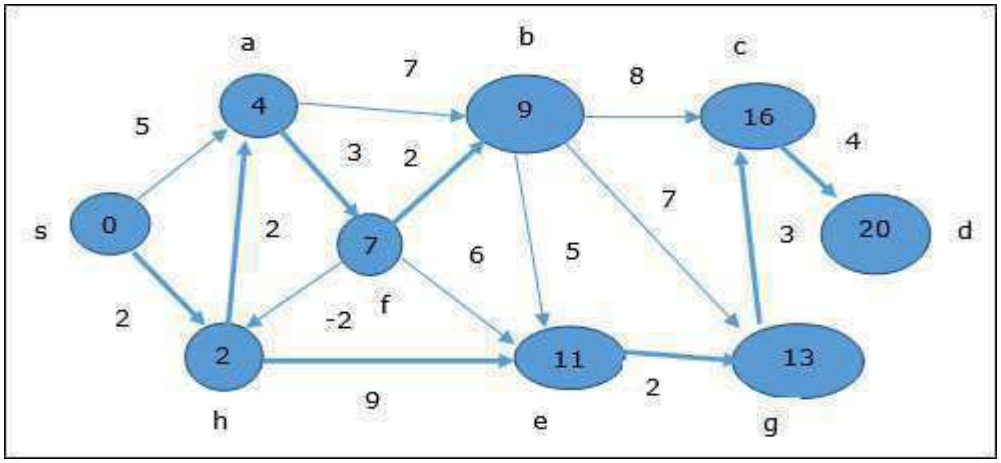
the next step, vertices ***a, b, f*** and ***e*** are updated.



Following the same logic, in this step vertices ***b, f, c*** and ***g*** are updated.



Here, vertices ***c*** and ***d*** are updated.



Hence, the minimum distance between vertex **s** and vertex **d** is **20**.

Based on the predecessor information, the path is s→ h→ e→ g→ c→ d

3.5 Congestion Control

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

Effects of Congestion

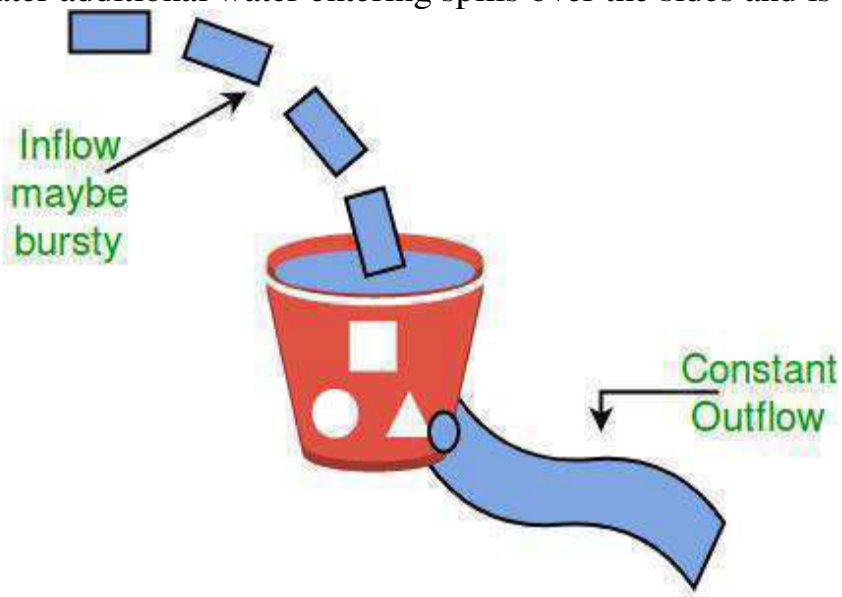
- As delay increases, performance decreases.
- If delay increases, retransmission occurs, making situation worse.

Congestion control algorithms

(a)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.

(b) Token bucket Algorithm

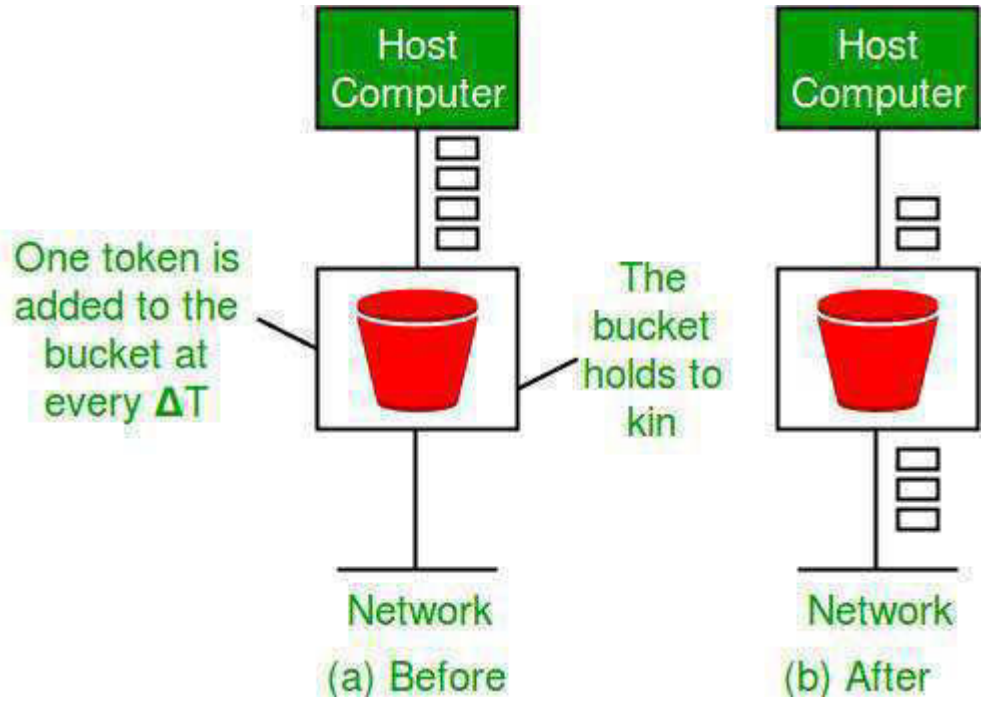
Need of token bucket Algorithm:-

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 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.



3.6 ATM Traffic Management

ATM technology is intended to support a wide variety of services and applications. The control of ATM network traffic is fundamentally related to the ability of the network to provide appropriately differentiated Quality of Service (QoS) for network applications.

A primary role of traffic management is to protect the network and the end-system from congestion in order to achieve network performance objectives. An additional role is to promote the efficient use of network resources. Proper traffic management helps ensure efficient and fair operation of networks in spite of constantly varying demand and ensure that users get their desired quality of service. One of the challenges in designing ATM traffic management was to maintain the QoS for various classes while attempting to make maximal use of network resources. This is what distinguishes traffic management from "congestion control" problem of the past. Congestion control deals only with the problem of reducing load during overload. Traffic management deals not only with load reduction under overload or load increase during underload but more importantly it tries to ensure that the QoS guarantees are met in spite of varying load conditions. Thus, traffic management is required even if the network is underloaded. The problem is especially difficult during periods of heavy load particularly if the traffic demands cannot be predicted in advance. This is why congestion control, although only a part of the traffic management issues, is the most essential aspect of traffic management.

A number of traffic control mechanisms are defined, which the network may utilize to meet the QoS objectives. To meet the QoS objectives, the following functions form a framework for managing and controlling traffic and congestion in ATM networks and may be used in appropriate combinations depending on the service category.

• **Connection Admission Control (CAC)** is defined as the set of actions taken by the network during the call set-up phase in order to determine whether a connection request can be accepted or should be rejected.

• **Feedback controls** are defined as the set of actions taken by the network and by end-systems to regulate the traffic submitted on ATM connections according to the state of network elements.

• **Usage Parameter Control (UPC)** is defined as the set of actions taken by the network to monitor traffic and enforce the traffic contract at the User Network Interface. Network Parameter Control (NPC) is a similarly defined set of actions at the Network Node Interface. The main purpose of UPC and NPC is to protect network resources from malicious as well as unintentional misbehavior, which

can affect the QoS of other already established connections, by detecting violations of negotiated parameters and taking appropriate actions. Such actions may include cell discard and cell tagging.

• **Cell Loss Priority control** may generate traffic flows of cells with Cell Loss Priority (CLP) marking. The network may follow models which treat this marking as transparent or as significant. If treated as significant, the network may selectively discard cells marked with a low priority to protect, as far as possible, the QoS objectives of cells with high priority.

• **Traffic Shaping** mechanisms may be used to achieve a desired modification to the traffic characteristics of a connection. The objectives of this function are to achieve a better network efficiency whilst meeting the QoS objectives and/or to ensure connection traffic conformance at a subsequent interface.

• **Network Resource Management** architecture allows logical separation of connections according to service characteristics. Although cell scheduling and resource provisioning are implementation and network specific, they can be utilized to provide appropriate isolation and access to resources. Virtual Paths are a useful tool for resource management.

• **Frame Discard** , a congested network that needs to discard cells may discard at the frame level rather than at the cell level.

3.7 Internet Layer Protocols

(1)Address Resolution Protocol (ARP)

If a machine talks to another machine in the same network, it requires its physical or MAC address. But ,since the application has given the destination's IP address it requires some mechanism to bind the IP address with its MAC address.This is done through Address Resolution protocol (ARP).IP address of the destination node is broadcast and the destination node informs the source of its MAC address.

1. Assume broadcast nature of LAN
2. Broadcast IP address of the destination
3. Destination replies it with its MAC address.
4. Source maintains a cache of IP and MAC address bindings

But this means that every time machine A wants to send packets to machine B, A has to send an ARP packet to resolve the MAC address of B and hence this will increase the traffic load too much, so to reduce the communication cost

computers that use ARP maintains a cache of recently acquired IP_to_MAC address bindings, i.e. they dont have to use ARP repeatedly. ARP Refinements Several refinements of ARP are possible: When machine A wants to send packets to machine B, it is possible that machine B is going to send packets to machine A in the near future .So to avoid ARP for machine B, A should put its IP_to_MAC address binding in the special packet while requesting for the MAC address of B. Since A broadcasts its initial request for the MAC address of B, every machine on the network should extract and store in its cache the IP_to_MAC address binding of A When a new machine appears on the network (e.g. when an operating system reboots) it can broadcast its IP_to_MAC address binding so that all other machines can store it in their caches..

(2) Reverse Address Resolution Protocol (RARP)

RARP is a protocol by which a physical machine in a local area network can request to learn its IP address from a gateway server's Address Resolution Protocol table or cache. This is needed since the machine may not have permanently attached disk where it can store its IP address permanently. A network administrator creates a table in a local area network's gateway router that maps the physical machine (or Medium Access Control - MAC) addresses to corresponding Internet Protocol addresses. When a new machine is set up, its RARP client program requests from the RARP server on the router to be sent its IP address. Assuming that an entry has been set up in the router table, the RARP server will return the IP address to the machine which can store it for future use. Both the machine that issues the request and the server that responds use physical network addresses during their brief communication. Usually, the requester does not know the physical address. So, the request is broadcasted to all the machines on the network. Now, the requester must identify istelf uniquely to the server. For this either CPU serial number or the machine's physical network address can be used. But using the physical address as a unique id has two advantages.

- These addresses are always available and do not have to be bound into bootstrap code.
- Because the identifying information depends on the network and not on the CPU vendor, all machines on a given network will supply unique identifiers.

(3) Internet Controlled Message Protocol (ICMP)

This protocol discusses a mechanism that gateways and hosts use to communicate control or error information. The Internet protocol provides unreliable, connectionless datagram service,and that a datagram travels from gateway to gateway until it reaches one that can deliver it directly to its final destination. If a gateway cannot route or deliver a datagram, or if the gateway detects an unusual condition, like network congestion, that affects its ability to forward the datagram, it needs to instruct the original source to take action to avoid or correct the problem. The Internet Control Message Protocol allows gateways to send error or control messages to other gateways or hosts;ICMP provides communication between the Internet Protocol software on one machine and the Internet Protocol software on another. This is a special purpose message mechanism added by the designers to the TCP/IP protocols. This is to allow gateways in an internet to report errors or provide information about unexpecter circumstances. The IP protocol itself contains nothing to help the sender test connectivity or learn about failures.

ICMPMessageDelivery

ICMP messages travel across the internet in the data portion of an IP datagram, which itself travels across the internet in the data portion of an IP datagram, which itself travels across each physical network in the data portion of a frame. Datagrams carrying ICMP messages are routed exactly like datagrams carrying information for users;there is no additional reliability or priority.