UNIT - III

Network layer: Logical Addressing- IPV4- Classful and Classless Addressing, Subnetting, NAT, IPV6 Addressing. **Internetworking**- Tunneling, Address mapping- ARP, RARP, Bootp, DHCP, ICMP, IGMP.

Routing – Distance Vector Routing, Link State Routing, Shortest path routing.

Transport Layer: Process to Process Delivery, UDP and TCP protocols, TCP Connection Control. Congestion Control- Open loop Congestion Control, Closed Loop Congestion Control, Congestion control in TCP.

NETWORK LAYER

LOGICAL ADDRESSING

- Communication at the network layers is host-to-host (Computer-to-Computer) i.e.,A computer somewhere in the world needs to communicate with another computer through the Internet.
- The packet transmitted by the sending computer may pass through several LANs or WANs before reaching the destination computer.
- For this level of communication, we need a global addressing scheme; we called this logical addressing.
- we use the term IP address to mean a logical address in the network layer of the TCP/IP protocol suite.

There are 2 types of addressing mechanisms are present:

1. IPv4 (IP version4)-32 bits
2. IPv6 (IP version 6)-128 bits

IPv4 ADDRESSES

An **IPv4** address is a **32-bit** address that **uniquely** and **universally** defines the connection of a device to the Internet.

Unique: Two devices on the Internet can never have the same address at the same time.

Universal: The addressing system must be accepted by any host that wants to be connected to the Internet.

Address Space

- An address space is the total number of addresses used by the protocol.
- If a protocol uses **N** bits to define an address, the address space is $\boldsymbol{2^N}$ because each bit can have two different values (0 or 1) and N bits can have $\boldsymbol{2^N}$ values.
- IPv4 uses **32**-bit addresses, which means that the address space is $\mathbf{2^{32}}$ or **4,294,967,296** (more than **4 billion**).

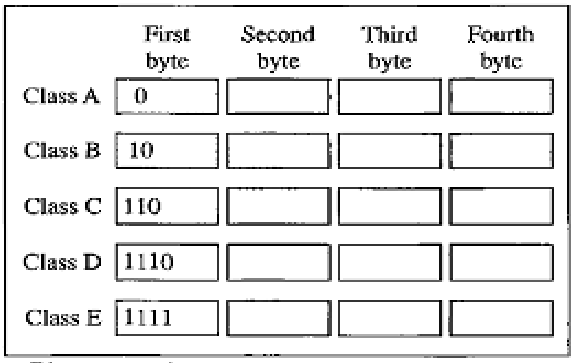
Notations

There are two notations to show an IPv4 address: Binary and Dotted-Decimal Notation.

Binary	**Dotted-Decimal**
• IPv4 address is displayed as 32 bits. Each octet is often referred to as a byte.	• Internet addresses are written in decimal form with a dot separating the bytes.
• It is a 4 byte address	• Each number in dotted-decimal notation is a value ranging from 0 to 255.
Ex: 10000000 00001011 00000011 00011111	Ex: 128.11.3.31

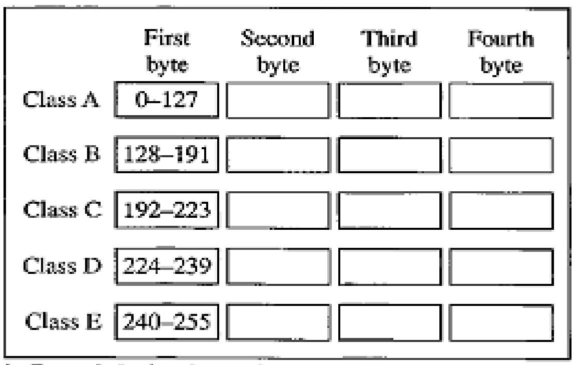
CLASSFUL ADDRESSING

- Initially IPv4 used the concept of Classful addressing.
- In classful addressing, the address space is divided into five classes: A, B, C, D, and E.
- If the address is given in binary notation, the first few bits can immediately tell us the class of the address.
- If the address is given in decimal-dotted notation, the first byte defines the class.



	First byte	Second byte	Third byte	Fourth byte
Class A	0			
Class B	10			
Class C	110			
Class D	1110			
Class E	1111			

a. Binary notation



	First byte	Second byte	Third byte	Fourth byte
Class A	0–127			
Class B	128–191			
Class C	192–223			
Class D	224–239			
Class E	240–255			

b. Dotted-decimal notation

Classes and Blocks

- Each class is divided into a fixed number of blocks.
- Size of the each block is also fixed.

Class	No of Blocks	Block Size	Application
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

Purpose of classes:

- **Class A** addresses were designed for large organizations with a large number of attached hosts or routers.
- **Class B** addresses were designed for midsize organizations with tens of thousands of attached hosts or routers.
- **Class C** addresses were designed for small organizations with a small number of attached hosts or routers.

- **Class D** addresses were designed for multicasting.
- **Class E** addresses were reserved for future use.

Problem with above classes and block sizes: **A lot of addresses are wasted.**

1. A block in **class A** address is too large for almost any organization. (i.e) most of the addresses in class A were wasted and were not used.
2. A block in **class B** is also very large, probably too large for many of the organizations that received a class B block.
3. A block in **class C** is probably too small for many organizations.
4. **Class D** addresses were designed for multicasting, each address in class D is used to define one group of hosts on the Internet. The Internet authorities wrongly predicted a need for 268,435,456 groups.
5. **Class E** addresses were reserved for future use; only a few addresses were used till now.

Netid and Hostid

- In classful addressing, an IP address in class A, B, or C is divided into Netid and Hostid.
- For Class D,E there were no Netid and Hostid.
- In class A, one byte defines the Netid and three bytes define the Hostid.
- In class B, two bytes define the Netid and two bytes define the Hostid.
- In class C, three bytes define the Netid and one byte defines the Hostid.

Mask or Default Mask

- A default mask is a 32-bit number made of contiguous 1's followed by contiguous 0's.
- The mask can help us to find the Netid and the Hostid.
- For example, the mask for a class A address has eight 1s, which means the first 8 bits of any address in class A define the Netid; the next 24 bits define the Hostid.

The masks for classes A, B, and C are:

Class	Binary	Dotted-Decimal	CIDR
A	**11111111** 00000000 00000000 00000000	**255**.0.0.0	**/8**
B	**11111111 11111111** 00000000 00000000	**255.255**.0.0	**/16**
C	**11111111 11111111 11111111** 00000000	**255.255.255**.0	**/24**

Subnetting

- Subnetting is a process of dividing a large block into smaller contiguous groups and assigns each group to smaller networks (subnets) or share a part of the addresses with neighbors.
- Subnetting increases the number of 1's in the mask.

Supernetting

- In supernetting, an organization can combine several blocks to create a larger range of addresses. Supernetting decreases the number of 1's in the mask.

Example: an organization that needs 1000 addresses can be granted four contiguous class C blocks. The organization can then use these addresses to create one supernetwork.

Address Depletion

- The number of available IPv4 addresses is decreasing as the number of internet users are increasing.
- We have run out of class A and B addresses, and a class C block is too small for most midsize organizations.
- One solution that has alleviated the problem is the idea of **Classless Addressing**.

<u>CLASSLESS ADDRESSING</u>

Purpose

- Classless addressing was designed and implemented to overcome address depletion and give more organizations access to the Internet.
- In this scheme, there are no classes, but the addresses are still granted in blocks.

Address Blocks

- In classless addressing, when a small or large entity, needs to be connected to the Internet, it is granted a block of addresses.
- The size of the block (the number of addresses) varies based on the nature and size of the entity.

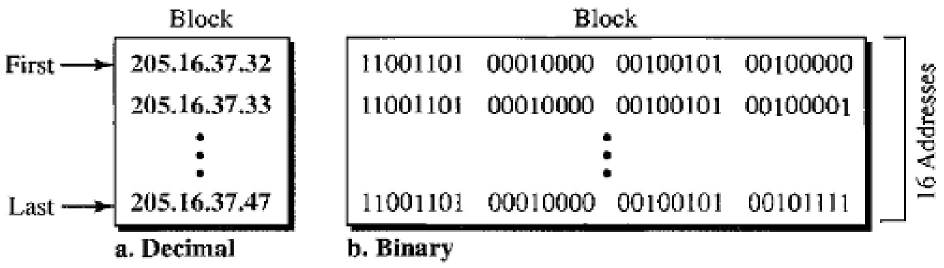
Example:

- For a large organization may be given thousands of addresses
- For a house two addresses are sufficient
- An Internet service provider may be given hundreds of thousands based on the number of customers it may serve.

Restrictions on classless address blocks

1. The addresses in a block must be contiguous, one after another.
2. The number of addresses in a block must be a power of 2 (1, 2, 4, 8, ...).
3. The first address must be evenly divisible by the number of addresses.

Consider the below figure for classless addressing that shows a block of addresses, in both binary and dotted-decimal notation, granted to a small business that needs 16 addresses.



It satisfies all 3 restrictions:

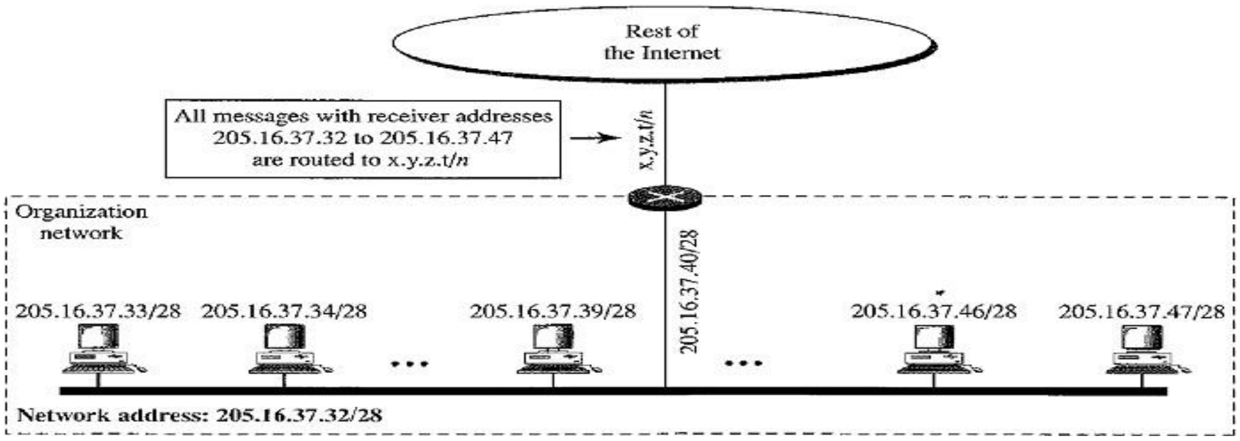
- The addresses are contiguous.
- The number of addresses is a power of 2 (16 = $\mathbf{2^4}$).
- The first address is divisible by 16. The first address, when converted to a decimal number, is 3,440,387,360, which when divided by 16 results in 215,024,210.

Mask

A mask is a 32-bit number in which the ***n*** leftmost bits are 1's and the **32 - *n*** rightmost bits are 0's, where ***n= 0 to 32.***

In 1Pv4 addressing, a block of addresses can be defined as **x.y.z.t/n** in which **x.y.z.t** defines one of the addresses and the ***/n*** defines the mask. ***/n*** is called as CIDR notation.

- **First Address** in the block can be found by setting the rightmost **32 - *n*** bits to 0's.
- **Last Address** in the block can be found by setting the rightmost **32 - *n*** bits to 1's.
- **Number of Addresses** in the block can be found by using the formula 2^{32-n}.
- **Network Address** is the first address in the block and defines the organization network. Usually the first address is used by routers to direct the message sent to the organization from the outside.
-



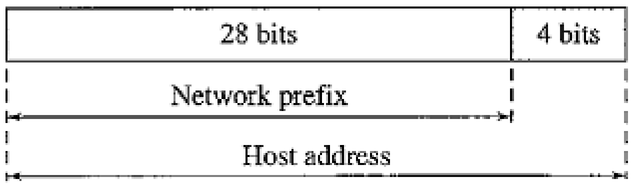
Example: **205.16.37.39/28** or **11001101 00010000 00100101 00100111**

- First address: 11001101 000100000100101 0010000 or 205.16.37.**32**
- Last address: 11001101 00010000 001001010010 1111 or 205.16.37.**47**
- Number of Addresses: $\mathbf{2^{32-28} = 2^4 = 16.}$
- Network Address (First Address) 11001101 000100000100101 0010000 or 205.16.37.32

Netid and Hostid

- The ***n*** leftmost bits of the address **x.y.z.t/n** define the **network address** or **prefix**.
- The **(32 – *n*)** rightmost bits define the particular **suffix** or **host address** (computer or router) connected to the network.

205.16.37.39/28 or **11001101 00010000 00100101 0010 0111**



Network Address Translation (NAT)

- As the number of home users and small business users are increasing day by day it is not possible to give each and every user to one IPv4 address due to shortage of IPv4 addresses.
- In order to overcome this problem the developers designed the concepts of private IP address and Network Address Translation (NAT).
- NAT enables a user to have a large set of addresses internally (private IP addresses) anda small set of addresses externally (public IP addresses).

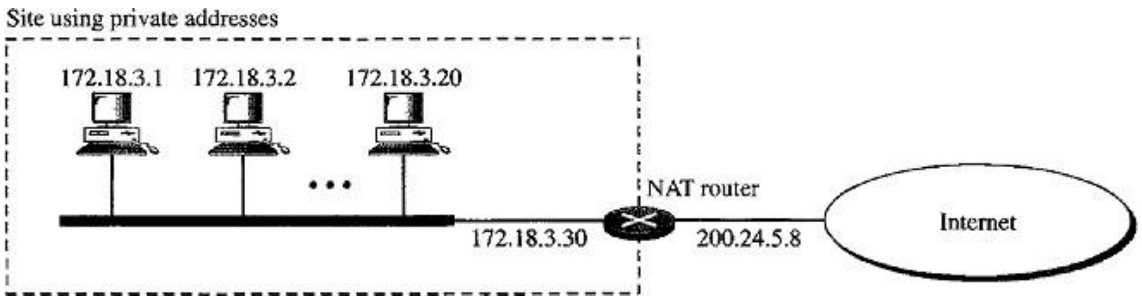
The Internet authorities have reserved three sets of addresses as private addresses:

Range	Total
10.0.0.0 to 10.255.255.255	2^{24}
172.16.0.0 to 172.31.255.255	2^{20}
192.168.0.0 to 192.168.255.255	2^{16}

- Any organization can use an address out of this set without permission from the Internet authorities.
- They are unique inside the organization, but they are not unique globally. No router will forward a packet that has one of these addresses as the destination address.

Example:

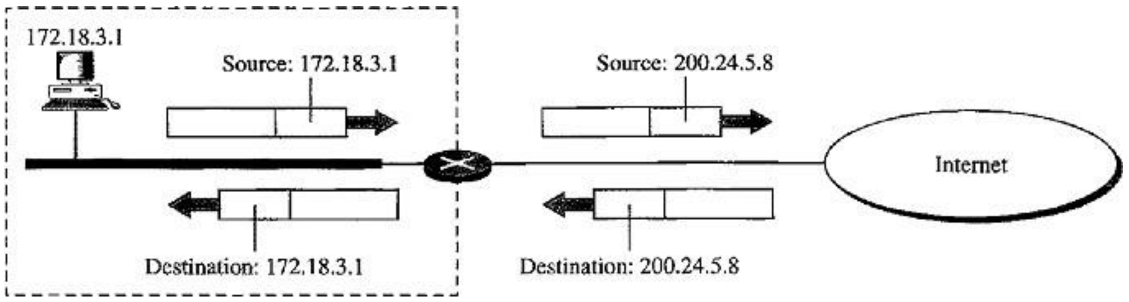
Consider the below figure describes the private network with private addresses:



- The NAT router has one public address **200.24.5.8,** it is a global address.
- The internal devices having addresses from **172.18.3.1** to **172.18.3.30** these are local addresses.

Address Translation

- All the outgoing packets go through the NAT router, which replaces the **source address** in the packet with the **global** NAT address.
- All incoming packets also pass through the NAT router, which replaces the **destination address** in the packet (the NAT router global address) with the appropriate **private address.**



<u>**Translation Table**</u>

There are two types of translation tables:

1. Two Column translation table (Using one IP address)
2. Five column translation table (Using IP addresses and Port Numbers)

Two Column Translation Table

- It contains two columns: Private Address and External Address.

- In this strategy, communication must always be initiated by the private network.
- When the router translates the source address of the outgoing packet, it also makes note of the destination address-where the packet is going.
- When the response comes back from the destination, the router uses the source address of the packet (as the external address) to find the private address of the packet.

Translation Table

Private	**External**
172.18.3.1	25.8.2.10
…	….
….	….

Five Column Translation Table

To allow a many-to-many relationship between private-network hosts and external server programs, we need more information (columns) in the translation table such as:

1. Private Address
2. Private Port
3. External Address
4. External Port
5. Transport Protocol

Example: Suppose two hosts with addresses 172.18.3.1 and 172.18.3.2 inside a private network need to access the HTTP server on external host 25.8.3.2.

Translation table has five columns instead of two, that include the source and destination port numbers of the transport layer protocol the ambiguity is eliminated.

When the response from HTTP comes back, the combination of source address (25.8.3.2) and destination port number (1400) defines the-private network host to which the response should be directed.

Note: The temporary port numbers must be unique.

Private Address	**Private Port**	**External Address**	**External Port**	**Transport Protocol**
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
…	…	…	…	…

IPv6 ADDRESSES (Internetworking Protocol version6)

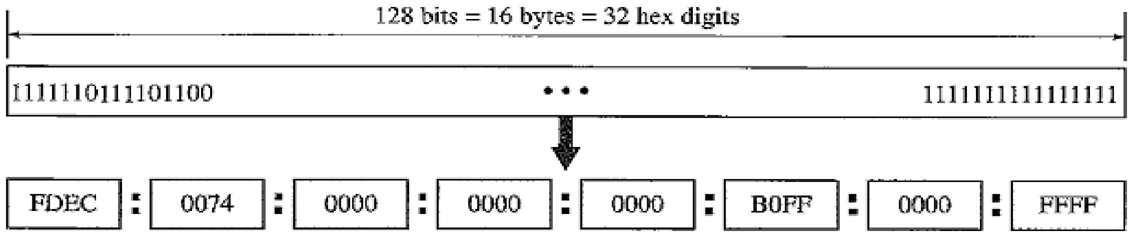
Why IPv6?

- In order to overcome the problems of address depletion.
- It eliminates the concept of NAT and Private Addresses.
- There is no need for classless addressing and DHCP.

Structure

IPv6 specifies hexadecimal colon notation (0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F).

An IPv6 address consists of 16 bytes (octets); it is 128 bits long. 128 bits is divided into eight



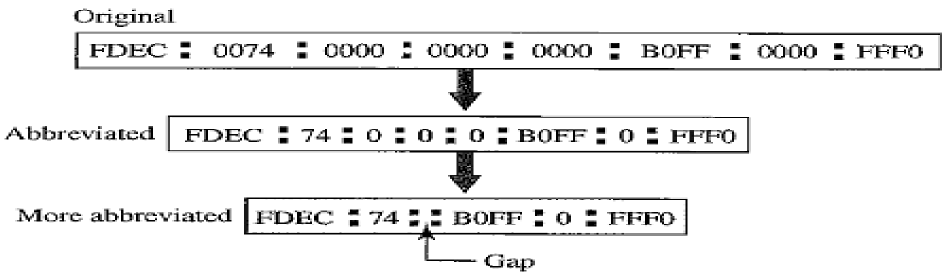
sections. Each of 4 hex digits separated by a colon.

Abbreviation

- Hexa decimal format of IPv6 is very long and many of the digits are zeros.
- We can abbreviate this address as the leading zeros of a section (four digits between two colons) can be omitted.

Note: Only leading zeros are omitted not trailing zeros.

Example:


Original
FDEC : 0074 : 0000 : 0000 : 0000 : B0FF : 0000 : FFF0
Abbreviated
FDEC : 74 : 0 : 0 : 0 : B0FF : 0 : FFF0
More abbreviated
FDEC : 74 : : B0FF : 0 : FFF0
Gap

- In the above example: 0074 written as 74, 0000 written as 0.
- If there are consecutive sections consisting of zeros only. We can remove the zeros altogether and replace them with a double semicolon.

Address Space

- IPv6 has 2^{128} addresses are available. It is a much larger address space than IPv4.
- The designers of IPv6 divided the address into several categories.
- A few leftmost bits called the **type prefix**, in each address define its category. Type Prefix is variable in length.

Type Prefix	**Type**	**Fraction**
00000000	Reserved	1/256
0000001	ISO network address	1/128
0000010	IPX Novell network address	1/128
010	Provider-based unicast addresses	1/8
100	Geographic-based unicast addresses	1/8
1111 111010	Link local addresses	1/1024
1111 1110 11	Site local addresses	1/1024
11111111	Multicast addresses	1/256

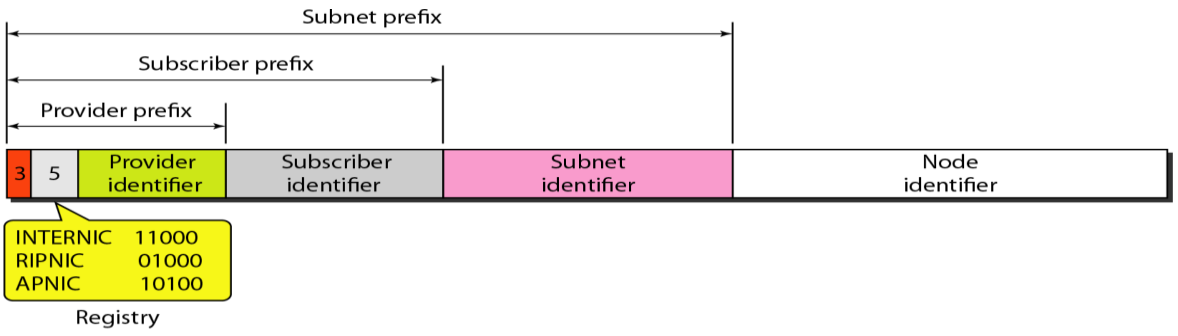
Unicast Addresses

A **unicast address** defines a single computer.

The packet sent to a unicast address must be delivered to that specific computer.

IPv6 defines two types of unicast addresses: geographically based and provider-based.

The provider-based address is generally used by a normal host as a unicast address.

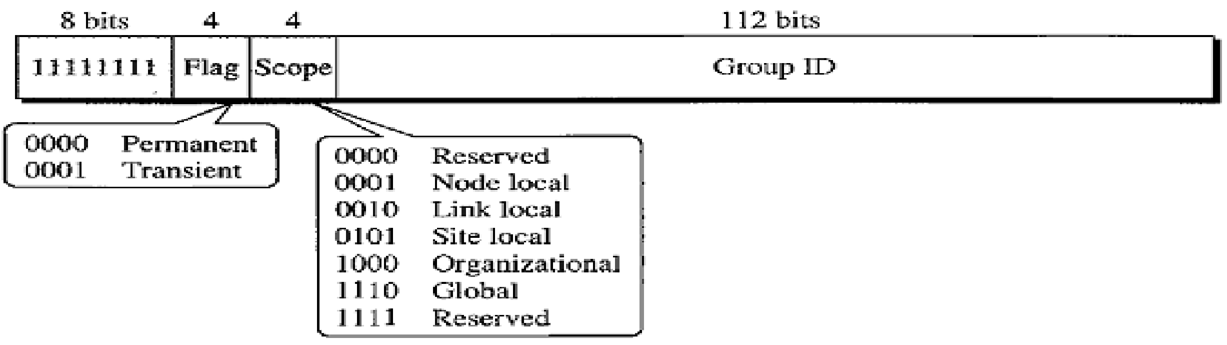


Fields for the provider-based address are as follows:

- **Type identifier (3 bits)** It defines the address as a provider-based address.
- **Registry identifier (5 bits)**
  This field indicates the agency that has registered the address. Currently three registry centers have been defined.
  - INTERNIC (code 11000) is the center for North America.
  - RIPNIC (code 01000) is the center for European registration.
  - APNIC (code 10100) is for Asian and Pacific countries.
- **Provider identifier (16 bits)**
  This variable-length field identifies the provider for Internet access (such as an ISP).
- **Subscriber identifier (24 bits)**
  When an organization subscribes to the Internet through a provider, it is assigned subscriber identification.
- **Subnet identifier (32-bits)**
  The subnet identifier defines a specific subnetwork under the territory of the subscriber.
- **Node identifier (48 bits)**
  It defines the identity of the node connected to a subnet.
  A length of 48 bits is recommended for this field to make it compatible with the 48-bit link (physical) address used by Ethernet.

Multicast Addresses

Multicast addresses are used to define a group of hosts instead of just one. A packet sent to a multicast address must be delivered to each member of the group.



- The second field is a flag that defines the group address as either permanent or transient.
- A permanent group address is defined by the Internet authorities and can be accessed at all times.
- A transient group address is used only temporarily. Systems engaged in a teleconference can use a transient group address.
- The third field defines the scope of the group address.

Reserved Addresses

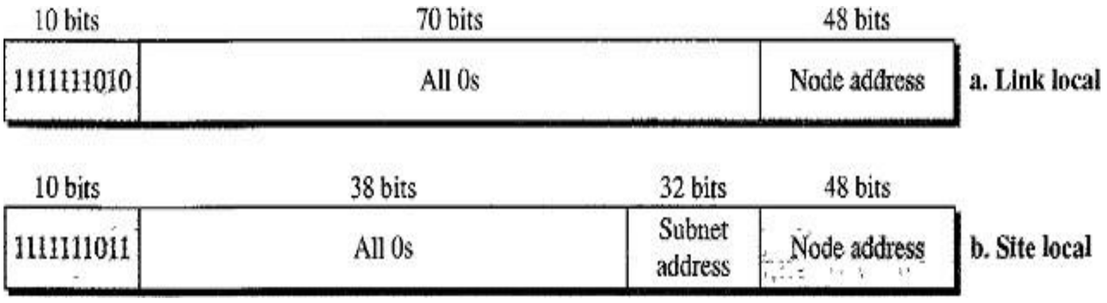
Reserved address are the another category of Address Space. These addresses start with eight 0's (type prefix is 00000000).

Local Addresses

- These addresses are used when an organization wants to use IPv6 protocol without being connected to the global Internet.
- That means they provide addressing for private networks. Nobody outside the organization can send a message to the nodes using these addresses.

There are two types of addresses are defined for this purpose: Link local and Site local.

- A link local address is used in an isolated subnet.
- A Site Local address is used in an isolated site with several subnets.



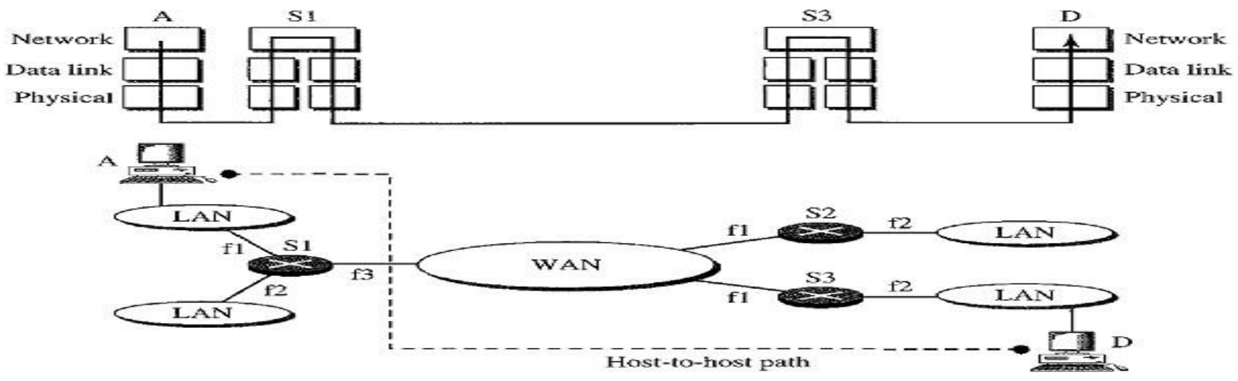
INTERNETWORKING

- This Internetwork is made of five networks: four LANs and one WAN in the following figure If host A needs to send a data packet to host D, the packet needs to go

  first from A to Rl (a switch or router) then

  from Rl to R3, and finally

  from R3 to host D.
- In each link, two physical and two data link layers are involved. However, there is a big problem here. When data arrive at interface Rl, how does RI know that Rf3 is the outgoing interface?
- There is no provision in the data link or physical layer to help Rl make the right decision.
- The frame does not carry any routing information either. The frame contains the MAC address of A as the source and the MAC address of Rl as the destination.
- To solve the problem of delivery through several links, the network layer (or the internetwork layer, as it is sometimes called) was designed. It is responsible for host-to-host delivery and for routing the packets through the routers or switches.



Network Layer at Source

- The network layer at the source is responsible for creating a packet from the data coming from another protocol such as a transport layer protocol or a routing protocol.
- The header of the packet contains the logical addresses of the source and destination.
- The network layer is responsible for checking its routing table to find the routing information such as the outgoing interface of the packet or the physical address of the next node.
- If the packet is too large then the packet is fragmented.

Network Layer at Router

- The network layer at the switch or router is responsible for routing the packet.
- When a packet arrives, the router or switch consults its routing table and finds the interface from which the packet must be sent.
- After some changes made in the packet Header, the routing information is passed to the Data-link layer again.

Network Layer at Destination

- The network layer at the destination is responsible for address verification; it makes sure that the destination address on the packet is the same as the address of the host.
- If the packet is a fragment, the network layer waits until all fragments have arrived, and then reassembles them and delivers the reassembled packet to the transport layer.

<u>Delivery of Packets</u>

The packet delivery will be done in two ways:

1. Connection Oriented Network
2. Connection less Network

<u>Packet delivery in Connection-Oriented Network</u>

- Delivery of a packet can be accomplished by using either a connection-oriented or a connectionless network service.
- In a connection-oriented service, the source first makes a connection with the destination before sending a packet.
- When the connection is established, a sequence of packets from the same source to the same destination can be sent one after another.
- In this case, there is a relationship between packets. They are sent on the same path in sequential order.
- When all packets of a message have been delivered, the connection is terminated.

- In a connection-oriented protocol, the decision about the route of a sequence of packets with the same source and destination addresses can be made at the time of connection establishment only.
- Switches do not recalculate the route for each individual packet.
- This type of service is used in a virtual-circuit approach to packet switching such as in Frame Relay and ATM.

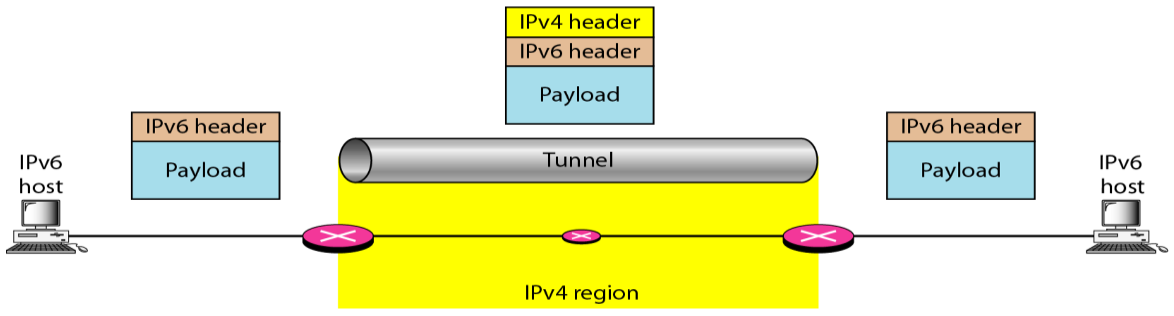
Internet as a Datagram Network (Connectionless Network)

- The Internet at the network layer is a packet-switched network uses datagram approach.
- In connection-less service, the network layer protocol treats each packet independently, with each packet having no relationship to any other packet.
- The packets in a message mayor may not travel the same path to their destination.
- The Internet uses datagram approach to switching the packets in the network layer. It uses the universal addresses defined in the network layer to route packets from the source to the destination.
- **Note**: The reason for using datagram network in internet is that "the Internet is made of so many heterogeneous networks that it is almost impossible to create a connection from the source to the destination without knowing the nature of the networks in advance".

TRANSITION FROM IPv4 TO IPv6

Tunneling

- Tunneling is a strategy used when two computers using IPv6 want to communicate with each other and the packet must pass through a region that uses IPv4.
- To pass through this region, the packet must have an IPv4 address.
- So the IPv6 packet is encapsulated in an IPv4 packet when it enters the region, and it leaves its capsule when it exits the region.
- It seems as if the IPv6 packet goes through a tunnel at one end and emerges at the other end. The IPv4 packet is carrying an IPv6 packet as data, the protocol value is set to 41.



ADDRESS MAPPING

- IP packets use Logical Addresses (Host-to-Host).
- Frame needs Physical Addresses (Node-to-Node).
- IP packets need to be encapsulated in a frame.

The physical address and the logical address are two different identifiers.

- A physical network such as Ethernet can have two different protocols at the network layer such as IP and IPX (Novell) at the same time.

- Delivery of a packet to a host or a router requires two levels of addressing: Logical and Physical.
- We need to be able to map a Logical Address to its corresponding Physical Address and Physical address to corresponding Logical Address.

Mapping can be done by using two ways:

- Mapping with Logical to Physical (uses ARP Protocol)
- Mapping with Physical to Logical(Uses RARP, BOOTP, and DHCP)

ARP (Mapping Logical to Physical address)

Need for Physical Address

- Anytime a host or a router has an IP datagram to send to another host or router, it has the logical (IP) address of the receiver.
- The logical (IP) address is obtained in two ways:
  i. If the sender is the host then logical address is obtained from DNS.
  ii. If the sender is router then logical address is obtained from a routing table.
- But the IP datagram must be encapsulated in a frame to be able to pass through the physical network.
- This means that the sender needs the physical address of the receiver.
- In order to know the physical address of the receiver the sender uses ARP protocol.

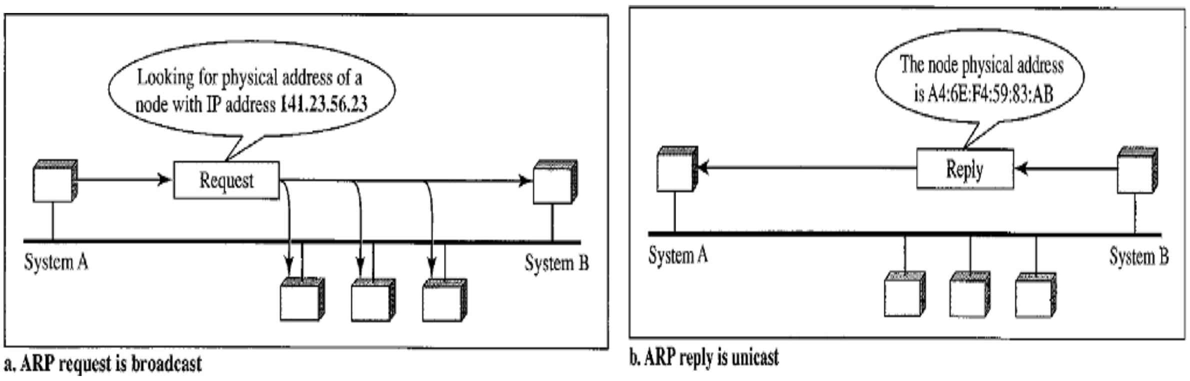
Process of ARP

- The sender knows the IP address of the target. The host or the router sends an ARP query packet.
- IP asks ARP to create an ARP request packet. The packet includes the **Physical address** and **IP addresses** of the **Sender** and the **IP address** of the **Receiver**.
- The target physical address field is filled with all 0's. Because the sender does not know the physical address of the receiver and the query is broadcast over the network.
- Every host or router on the network receives and processes the ARP query packet, but only the intended recipient recognizes its IP address and sends back an ARP reply packetwhereas the remaining devices discard the packet.
- The reply packet contains the recipient's IP and physical addresses.
- The packet is unicast directly to the sender by using the physical address received in the query packet.
- The sender receives the reply message. It now knows the physical address of the target machine.
- The IP datagram that carries data for the target machine is now encapsulated in a frame and datagram is unicasted to the destination.

Note

∗ A system is normally sends several packets to same destination.
∗ A system that receives an ARP reply stores the mapping in the cache memory and keeps it until the space in the cache is exhausted.

* Before sending an ARP request, the system first checks its cache to see if it can find the mapping.


Looking for physical address of a node with IP address 141.23.56.23
Request
System A
System B
The node physical address is A4:6E:F4:59:83:AB
Reply
System A
System B
a. ARP request is broadcast   b. ARP reply is unicast

The above figure describes:

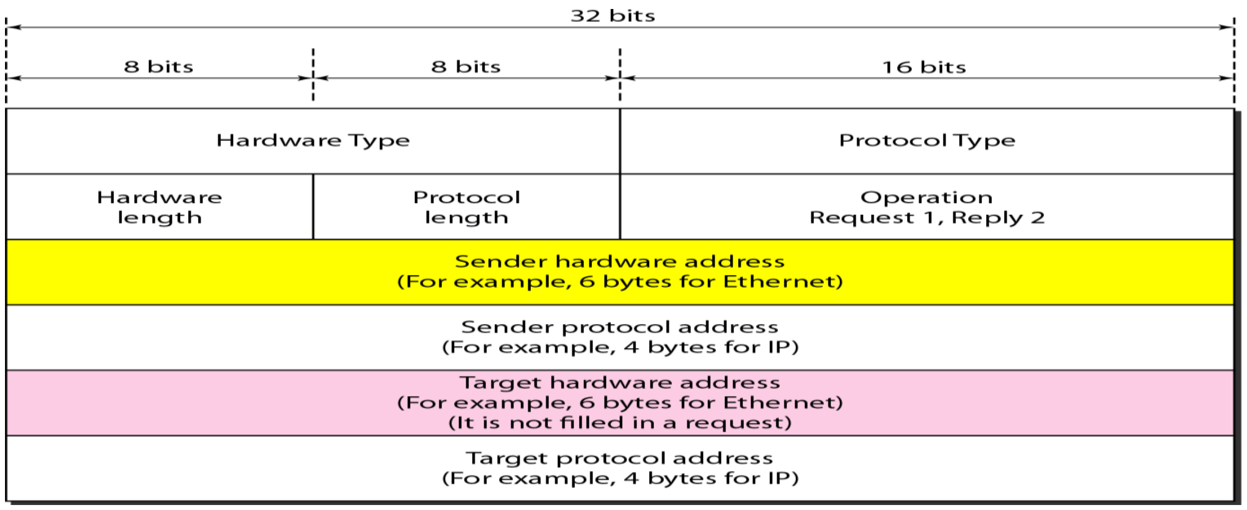
- System A has a packet that needs to be delivered to another system B with IP address 141.23.56.23.
- System A doesnot know the physical address of 141.23.56.23.
- System A broadcast ARP request packet to ask for physical address of 141.23.56.23
- This packet is received by every system on the physical network, but only system B will respond by sending ARP reply packet that includes its physical address (A4:6E:F4:59:83:AB).
- Now system A can send all the packets it has for the destination (System B) by using the physical address it received.

Packet Format

The fields are divided into fixed length fields(5) and Variable length fields(4).

Fixed length fields:

- **Hardware type (16 bit)**
  It defining the type of the network on which ARP is running.
  Each LAN has been assigned an integer based on its type. **Ex:** Ethernet is given type 1.
  ARP can be used on any physical network.
- **Protocol type (16-bit)**
  This field defines the protocol. For IPv4 Protocol the value is **0800_{16}.**
- **Hardware length (8-bit)**
  This field defines the length of the physical address in bytes. Ex: for Ethernet value = 6.
- **Protocol length (8-bit)**
  This defines the length of logical address in bytes. For IPv4 protocol the value is 4.
- **Operation (16-bit)**
  This field defines the type of packet. Two packet types are defined: ARP request (1) and ARP reply (2).

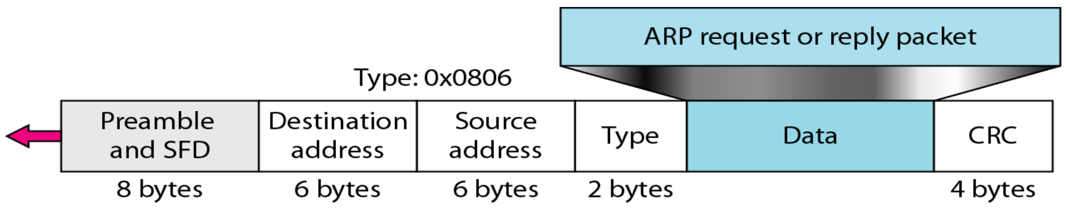

32 bits
8 bits
8 bits
16 bits
Hardware Type
Protocol Type
Hardware length
Protocol length
Operation Request 1, Reply 2
Sender hardware address (For example, 6 bytes for Ethernet)
Sender protocol address (For example, 4 bytes for IP)
Target hardware address (For example, 6 bytes for Ethernet) (It is not filled in a request)
Target protocol address (For example, 4 bytes for IP)

Variable length fields:

- **Sender hardware address**
  This is a variable length field defines the physical address of the sender. For example, for Ethernet this field is 6 bytes long.
- **Sender protocol address**
  It is a variable length field defines the the logical address (IP address) of the sender. For the IP protocol, this field is 4 bytes long.
- **Target hardware address**
  This is a variable-length field defining the physical address of the target (receiver).
  For example, for Ethernet this field is 6 bytes long.
  For an ARP request message, this field is all **0's** because the sender does not know the physical address of the target.
- **Target protocol address**
  This is a variable-length field defining the logical address of the target. For the IPv4 protocol, this field is 4 bytes long.

Encapsulation

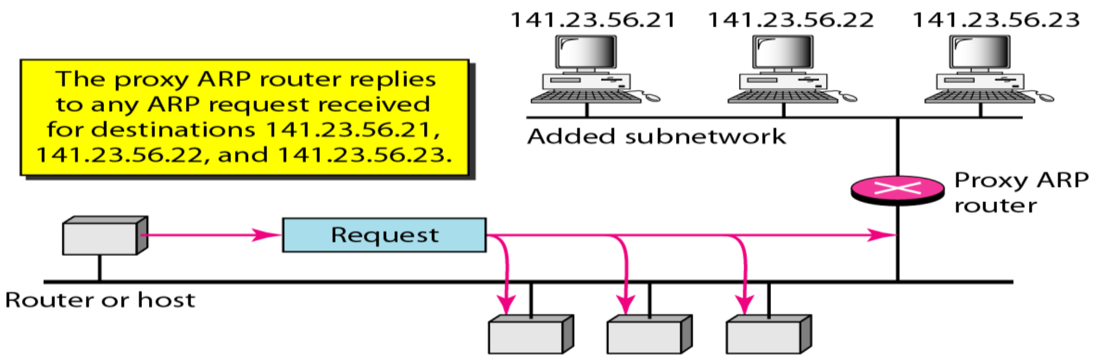
An ARP packet is encapsulated directly into a data link frame (i.e. Ethernet frame). Note that the type field indicates that the data carried by the frame are an ARP packet.



ProxyARP

- A technique called **Proxy ARP** is used to create a subnetting effect.
- A proxy ARP is an ARP that acts on behalf of a set of hosts.
- Whenever a router running a proxy ARP receives an ARP request looking for the IP address of one of these hosts, the router sends an ARP reply announcing its own hardware(physical) address.

- After the router receives the actual IP packet, it sends the packet to the appropriate host or router.


141.23.56.21
141.23.56.22
141.23.56.23
The proxy ARP router replies to any ARP request received for destinations 141.23.56.21, 141.23.56.22, and 141.23.56.23.
Added subnetwork
Proxy ARP router
Request
Router or host

In the above figure: The ARP installd on the right-hand host will answer only to anARP request with a target IP address of 141.23.56.23.

- The administrator may need to create a subnet without changing the whole system to recognize subnetted addresses. One solution is to add a router running a proxy ARP.
- In this case, the router acts on behalf of all the hosts installed on the subnet.
- When it receives an ARP request with a target IP address that matches the address of one of its protégés (141.23.56.21, 141.23.56.22, or 141.23.56.23), it sends an ARP reply and announces its hardware address as the target hardware address.
- When the router receives the IP packet, it sends the packet to the appropriate host.

Mapping Physical to Logical Address: RARP, BOOTP, and DHCP

There are occasions in which a host knows its physical address, but needs to know its logical address. This may happen in two cases:

1. An organization does not have enough IP addresses to assign to each station; it needs to assign IP addresses on demand. The station can send its physical address and ask for a short time lease.
2. A diskless station is just booted. The station can find its physical address by checking its interface, but it does not know its IP address.

Note:

A **Diskless machine** is usually booted from ROM, which has minimum booting information. The ROM is installed by the manufacturer. It cannot include the IP address because the IP addresses on a network are assigned by the network administrator.

REVERSE ADDRESS RESOLUTION PROTOCOL (RARP)

- RARP finds the logical address for a machine that knows only its physical address.
- Each host or router is assigned one or more logical (IP) addresses, which are unique and independent of the physical (hardware) address of the machine.
- To create an IP datagram, a host or a router needs to know its own IP address.
- IP address of a machine is usually read from its configuration file stored on a disk file.
- The machine can get its physical address by reading its NIC, which is unique locally.
- It can then use the physical address to get the logical address by using the RARP protocol.
- A RARP request is created and broadcast on the local network.
- Another machine on the local network that knows all the IP addresses will respond with a

RARP reply.

Note: The requesting machine must be running a RARP client program and the responding machine must be running a RARP server program.

Problems with RARP

- Broadcasting is done at the data link layer.
- The physical broadcast address, all 1's in the case of Ethernet. It does not pass the boundaries of a network. (i.e.) if an administrator has several networks or several subnets, it needs to assign a RARP server for each network or subnet.

Note: BOOTP and DHCP are implemented to solve the problems with RARP

Bootstrap Protocol (BOOTP)

- BOOTP is a client/server protocol designed to provide physical address to logical address mapping.
- BOOTP is an application layer protocol. The administrator may put the client and the server on the same network or on different networks.
- BOOTP messages are encapsulated in a UDP packet, and the UDP packet itself is encapsulated in an IP packet.

Advantage of BOOTP over RARP

The client and server are application-layer processes. As in other application-layer processes, a client can be in one network and the server in another, separated by several other networks.

Note:

How a client can send an IP datagram when it knows neither its own IP address (the source address) nor the server's IP address (the destination address).

The client simply uses all 0's as the source address and all 1's as the destination address.

Problem

The BOOTP request is broadcast because the client does not know the IP address of the server. A broadcast IP datagram cannot pass through any router.

Solution

- There is a need for intermediary host or router.
- One of the hosts or a router that can be configured to operate at the application layer can be used as a relay. The host is called a relay agent.
- The relay agent knows the unicast address of a BOOTP server.
- When it receives this type of packet, it encapsulates the message in a unicast datagram and sends the request to the BOOTP server.
- The packet carrying a unicast destination address is routed by any router and reaches the BOOTP server.
- The BOOTP server knows the message comes from a relay agent because one of the fields in the request message defines the IP address of the relay agent.
- The relay agent after receiving the reply, sends it to the BOOTP client.

DYNAMIC HOST CONFIGURATION PROTOCOL (DHCP)

DHCP is a static and dynamic configuration protocol whereas BOOTP is a static configuration protocol only.

Why DHCP?

- When a client requests its IP address, the BOOTP server consults a table that matches the physical address of the client with its IP address.
- This implies that the binding between the physical address and the IP address of the client already exists. The binding is predetermined.
- The binding or mapping between the physical address and IP addresses is static and fixed in a table until changed by the administrator. BOOTP is a static configuration protocol.

There are situations where BOOTP fails to handle:

i. What if a host moves from one physical network to another.
ii. What if a host wants a temporary IP address.

DHCP has been devised to provide **Static** and **Dynamic Address Allocation** that can be manual or automatic.

Static Address Allocation

- In this capacity DHCP acts as BOOTP.
- It is backward compatible with BOOTP, which means a host running the BOOTP client can request a static address from a DHCP server.
- A DHCP server has a database that statically binds physical addresses to IP addresses.

Dynamic Address Allocation

- DHCP has a second database with a pool of available IP addresses. This second database makes DHCP dynamic.
- When a DHCP client requests a temporary IP address, the DHCP server goes to the pool of available (unused) IP addresses and assigns an IP address for a negotiable period of time.
- When a DHCP client sends a request to a DHCP server, the server first checks its static database.
- If an entry with the requested physical address exists in the static database, the permanent IP address of the client is returned.
- If the entry does not exist in the static database, the server selects an IP address from the available pool, assigns the address to the client, and adds the entry to the dynamic database.
- The dynamic aspect of DHCP is needed when a host moves from network to network or is connected and disconnected from a network then DHCP provides temporary IP addresses for a limited time.

Temporary Addresses

- The addresses assigned from the pool are temporary addresses.
- The DHCP server issues a lease for a specific time. When the lease expires, the client must either stop using the IP address or renew the lease.
- The server has the option to agree or disagree with the renewal. If the server disagrees, the client stops using the address.

Advantages of DHCP over BOOTP

- One major problem with the BOOTP protocol is that the table mapping. The IP addresses to physical addresses needs to be manually configured.
- This means the administrator needs to manually enter the changes every time there is a change in a physical address or IP address.

- DHCP allows both manual and automatic configurations.
- Static addresses are created manually whereas dynamic addresses are created automatically.

INTERNET CONTROL MESSAGE PROTOCOL (ICMP)

The IP provides unreliable and connectionless datagram delivery. (i.e.) IP does not provide any Error control mechanisms and it does not provide any host management queries.
ICMP has been designed to compensate for the above two deficiencies. It is a companion to the IP protocol.

Types of Messages
ICMP messages are divided into two broad categories:

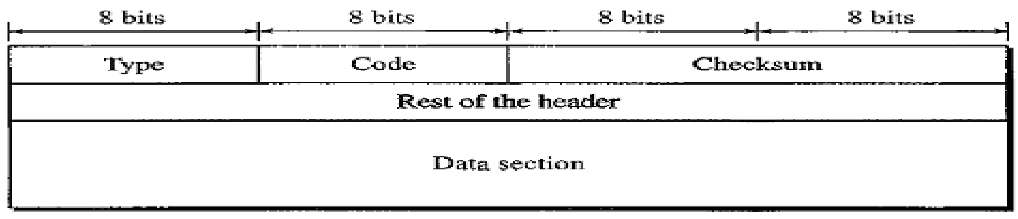
1. Error-Reporting Messages
2. Query Messages

- **Error-reporting messages** report problems that a router or a host (destination) may encounter when it processes an IP packet.
- **Query messages** occur in pairs, help a host or a network manager get specific information from a router or another host.

Example: nodes can discover their neighbors, and also hosts can discover and learn about routers on their network, and routers can help a node redirect its messages.

Message Format
An ICMP message has an 8-byte header and a variable-size data section.

The general format of the header is different for each message type, the first 3 fields Type, Code, Checksum are common to all. These common fields consisting of 4 bytes.

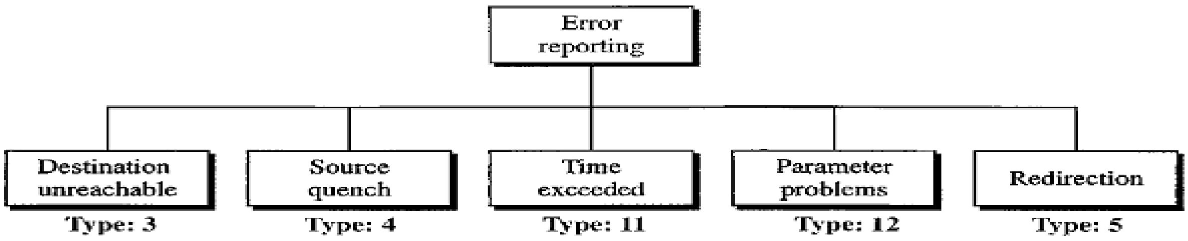


- **ICMP Type** defines the type of the message.
- **Code** field specifies the reason for the particular message type.
- **Rest of the header** is specific for each message type.
- **Checksum** is calculated over the entire message (header and data).
- **Data section** in Error messages carries information for finding the original packet that had the error. **Data Section** in Query messages the data section carries extra information based on the type of the query.

Error Reporting

- The main responsibilities of ICMP are to report errors. ICMP does not correct errors
- Error messages are always sent to the original source because the only information available in the datagram about the route is the source and destination IP addresses.
- ICMP uses the source IP address to send the error message to the original source of the datagram.

ICMP handles 5 types of errors:



Destination Unreachable (Type 3)

- When a router cannot route a datagram or a host cannot deliver a datagram then the datagram is discarded.
- The router or the host sends a destination-unreachable message back to the source host that initiated the datagram.
- Note that destination-unreachable messages can be created by either a router or the destination host.

Source Quench (Type 4)

- IP does not have a flow control mechanism embedded in the protocol.
- The lack of flow control can create major problems such as Congestion in routers or the destination host. When there is a congestion the router or host may discard the packets.
- When a router or host discards a datagram due to congestion, it sends a source-quench message to the sender of the datagram.

Source Quench message has two purposes.

i. It informs the source that the datagram has been discarded.
ii. It warns the source that there is congestion somewhere in the path and that the source should slow down (quench) the sending process.

Redirection (Type 5)

- Routing is dynamic. Routing table will be updated by routers. Host does not involve in the process of updation of routing tables.
- The hosts usually use static routing. Routing table has a limited number of entries.
- Host usually knows the IP address of the default router only. For this reason, the host may send a datagram, which is destined for another network to the wrong router.
- In this case the router that receives the datagram will forward the datagram to the correct router.
- To update the routing table of the host, it sends a redirection message to the host.

Time Exceeded (Type 11)

- Each datagram contains a field called Time To Live (TTL).
- When a datagram visits a router, the value of TTL field is decremented by 1.
- When the TTL value reaches 0 the router discards the datagram.
- When the datagram is discarded, a time-exceeded message must be sent by the router to the original source.

Note: A time-exceeded message is also generated when not all fragments that make up a message arrive at the destination host within a certain time limit.

Parameter Problem (Type 12)

- If a router or the destination host discovers an ambiguous or missing value in any field of the datagram, it discards the datagram and sends a parameter-problem message back to the source.

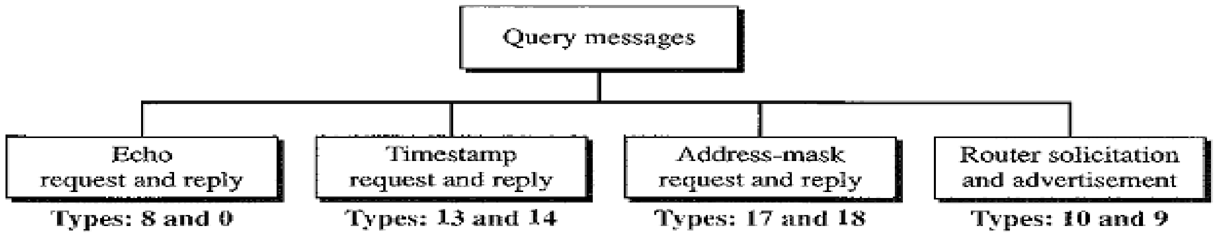
Note: There are some cases where ICMP error messages will not be generated.

i. No ICMP error message will be generated in response to a datagram carrying an ICMP error message.
ii. No ICMP error message will be generated for a fragmented datagram that is not the first fragment.
iii. No ICMP error message will be generated for a datagram having a **Multicast Address**.
iv. No ICMP error message will be generated for a datagram having a special address such as 127.0.0.0 or 0.0.0.0.

Query Messages

- In this type of ICMP message, a node sends a message that is answered in a specific format by the destination node.
- A query message is encapsulated in an IP packet, which in turn is encapsulated in a data link layer frame.

Here no bytes of the original IP are included in the message.



Echo Request and Echo Reply

- The echo-request and reply messages can be used to determine if there is communication at the IP level.
- These are used for diagnostic purpose, network managers and users utilize this pair of messages to identify network problems.
- ICMP messages are encapsulated in IP datagrams.
- The receipt of an echo-reply message by the machine that sent the echo request is proof that the IP protocols in the sender and receiver are communicating with each other using the IP datagram.

Example: **ping** command.

Timestamp Request and Reply

- Two machines (hosts or routers) can use the timestamp request and timestamp reply messages to determine the round-trip time needed for an IP datagram to travel between them. It can also be used to synchronize the clocks in two machines.

Address-Mask Request and Reply

- A host may know its IP address, but it may not know the corresponding mask.
- To obtain its mask, a host sends an address-mask-request message to a router on the LAN.
- **Ex:** A host IP address is 159.31.17.24, but it doesn't know its corresponding mask **/24**.
- If the host knows the address of the router, it sends the request directly to the router.
- If it does not know, it broadcasts the message.

- The router receiving the address-mask-request message responds with an address-mask-reply message, providing the necessary mask for the host.

Router Solicitation and Advertisement

- The router-solicitation and router-advertisement messages can help whether the router is functioning or not.
- A host can broadcast (or) multicast a router-solicitation message.
- Routers that receive the solicitation message broadcast their routing information using the router-advertisement message.
- In router advertisement message it announces its own presence and all routers on the network which it is aware of.

Note: A router can also periodically send router-advertisement messages even if no host has solicited.

Debugging Tools

There are several tools that can be used in the Internet for debugging.
There are two tools that are used for ICMP debugging: **ping** and **traceroute.**

Ping

- Ping program to find if a host is alive and responding.
- The source host sends ICMP echo-request messages (type: 8, code: 0); the destination, if alive, responds with ICMP echo-reply messages.
- The *ping* program sets the identifier field in the echo-request and echo-reply message and starts the sequence number from 0; this number is incremented by 1 each time a new message is sent.
- *Ping* can calculate the round-trip time. It inserts the sending time in the data section of the message. When the packet arrives, it subtracts the arrival time from the departure time to get the round-trip time (RTT).

Example: ping program to test the server fhda.edu. The result is shown below:

```
$ ping thda.edu
PING fhda.edu (153.18.8.1) 56 (84) bytes of data.
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=0      ttl=62      time=1.91 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=1      ttl=62      time=2.04 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=2      ttl=62      time=1.90 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=3      ttl=62      time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=4      ttl=62      time=1.93 ms
--- fhda.edu ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time xxxxx ms
rtt min/avg/max = 1.911/1.955/2.04 ms.
```

- **ttl=62**: means that the packet cannot travel more than 62 hops.
- *ping* defines the number of data bytes as 56 and the total number of bytes as 84. It is obvious that if we add 8 bytes of ICMP header and 20 bytes of IP header to 56, the result is 84.
- **Note**: In each probe *ping* defines the number of bytes as 64. This is the total number of bytes in the ICMP packet (56 + 8).

Traceroute

- The traceroute program in UNIX or *tracert* in Windows can be used to trace the route of a packet from the source to the destination.
- The program elegantly uses two ICMP messages, time exceeded and destination unreachable, to find the route of a packet.
- This is a program at the application level that uses the services of UDP

INTERNET GROUP MANAGEMENT PROTOCOL (IGMP)

IGMP is a companion to IP protocol. IGMP is not a multicasting routing protocol. It is a protocol that manages Group Membership.

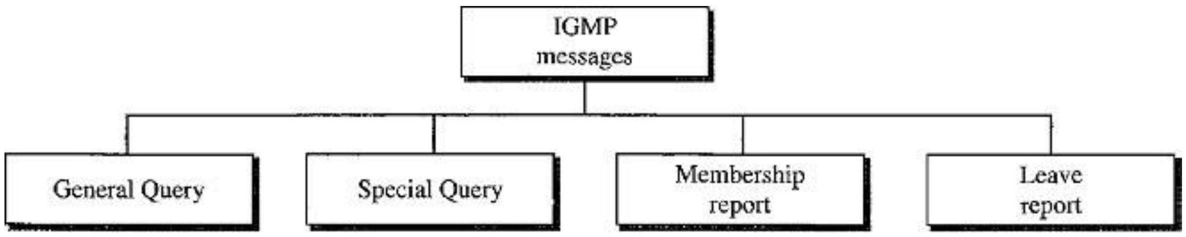
In any network, there are one or more multicast routers that distribute multicast packets to hosts or other routers. The IGMP protocol gives the multicast routers information about the membership status of hosts (routers) connected to the network.

Need for IGMP

- A multicast router may receive thousands of multicast packets every day for different groups.
- If a router has no knowledge about the membership status of the hosts, it must broadcast all these packets. This creates a lot of traffic and consumes bandwidth.
- A better solution is to keep a list of groups in the network for which there is at least one loyal member.
- IGMP helps the multicast router create and update this list.

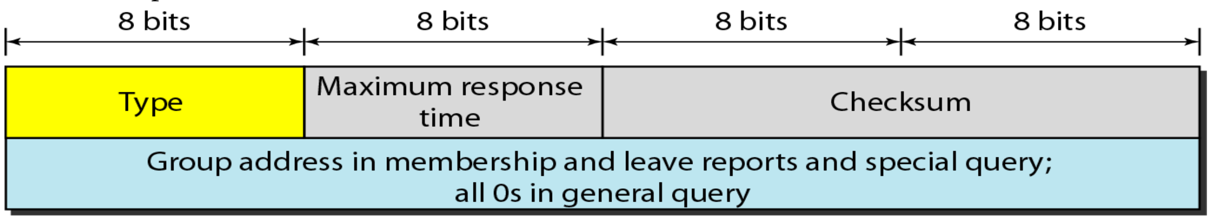
IGMP Messages and Message Format

IGMP has 3 types of messages: Query, Membership report, Leave report.
Query messages can be divided into two types: General and Special



The message format of IGMP contains four fields:

1. Type
2. Maximum Response Time
3. Checksum
4. Group Address


8 bits
8 bits
8 bits
8 bits
Type
Maximum response time
Checksum
Group address in membership and leave reports and special query; all 0s in general query

- **Type**
  This 8-bit field defines the type of message.

Type	Value

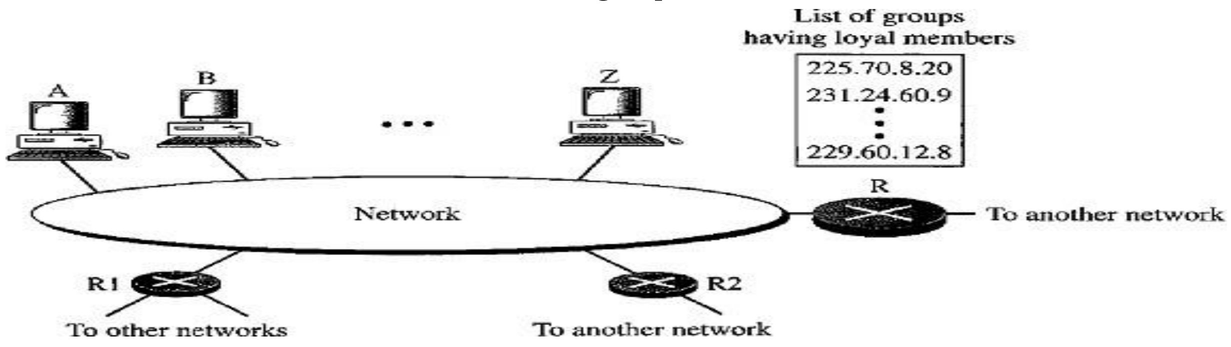
General or Special Query	0x11 or 00010001
Membership Report	0x16 or 00010110
Leave Report	0x17 or 00010111

- **Maximum Response Time**
  This 8-bit field defines the amount of time in which a query must be answered. The value is in tenths of a second.
  Example: Value=100 it means 10 sec.
  The value is nonzero in the query message.
  For Membership and Leave report the value is 0.
- **Checksum**
  This is a 16-bit field carrying the checksum. The checksum is calculated over the 8-byte message.
- **Group address**
  The value of this field is 0 for a general query message.
  The value defines the groupid (multicast address of the group) in the special query, the membership report, and the leave report messages.

IGMP Operation

- IGMP operates locally.
- A multicast router connected to a network has a list of multicast addresses of the groups with at least one loyal member in that network.
- For each group, there is one router that has the duty of distributing the multicast packets destined for that group.
- If there are three multicast routers connected to a network, their lists of groupid's are mutually exclusive.

Example: Consider the below figure the routers R, R1, R2 are connected to the network but only router R distributes packets with the multicast address of 225.70.8.20.
R1 and R2 are the multicast routers of other groups.



Group Membership or Group Interest

- A host or multicast router can have membership in a group.
- When a host has membership, one of the application program processes of host receives multicast packets from some group.
- When a router has membership, a network connected to one of the other interfaces of the router receives these multicast packets.
- In both of the cases, the host and the router keep a list of Group-id's and relay their interest to the distributing router.

- Membership in a group is also called as an **Interest in the Group.**

Query Router
Query messages may create a lot of responses. To prevent unnecessary traffic, IGMP designates one router as the query router for each network. Only this designated router sends the query message and the other routers are passive and they receive responses and update their lists.
There are 4 operations performed in IGMP:

1. Joining a group
2. Leaving a group
3. Monitoring Membership
4. Delayed Response

Joining a group

- A host or a router can join a group.
- A host maintains a list of processes that have membership in a group.
- When a process wants to join a new group, it sends its request to the host.
- The host adds the name of the process and the name of the requested group to its list.
- If this is the first entry for this particular group, the host sends a membership report message.
- If this is not the first entry, there is no need to send the membership report since the host is already a member of the group and it already receives multicast packets for this group.

Note: The protocol requires that the membership report be sent twice, one after the other within a few moments because if the first one is lost or damaged, the second one replaces it.

Leaving a Group

- When a host sees that no process is interested in a specific group, host sends a leave report.
- When a router sees that none of the networks connected to its interfaces is interested in a specific group, it sends a leave report about that group.
- When a multicast router receives a leave report, it cannot immediately purge that group from its list because the report comes from just one host or router, there may be other hosts or routers that are still interested in that group.
- The router sends a special query message and inserts the groupid or multicast address related to the group.
- The router waits for a specified time for any host or router to respond.
- During this time, if no interest (membership report) is received then the router assumes that there are no loyal members in the network and purges the group from its list.

Monitoring Membership

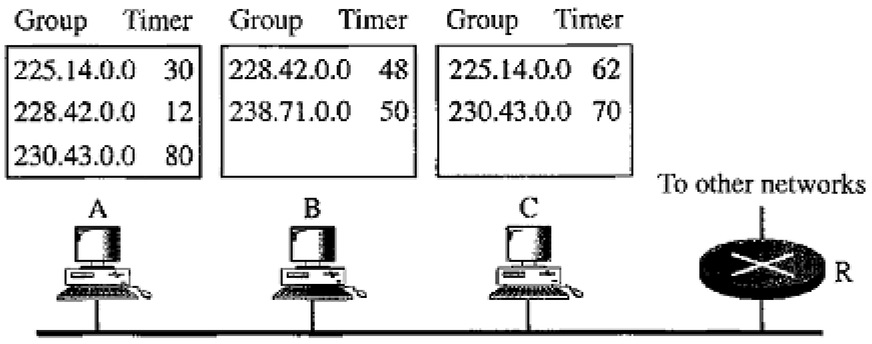
- The multicast router is responsible for monitoring all the hosts or routers in a LAN to see if they want to continue their membership in a group.
- The router (Query Router) periodically sends a general query message.
- The group address field of the message is set to 0.0.0.0. (i.e.) the query for membership continuation is sent to all groups in which a host is involved.
- The router expects an answer for each group in its group list.
- The query message has a maximum response time of 10 s.
- When a host or router receives the general query message, it responds with a membership

report if it is interested in a group.

Delayed Response

- IGMP uses Delayed response strategy to prevent the traffic.
- If there is a common interest (i.e) two hosts are interested in the same group only one response is sent for that group to prevent unnecessary traffic. This is called a delayed response.
- When a host or router receives a query message, it does not respond immediately, it delays the response.
- Each host or router uses a random number to create a timer, which expires between 1 and l0 sec. The expiration time can be in steps of 1sec or less.
- A timer is set for each group in the list.
- Each host or router waits until its timer has expired before sending a membership report message. During this waiting time, if the timer of another host or router for the same group expires earlier, that host or router sends a membership report.
- Because the report is broadcast, the waiting host or router receives the report and knows that there is no need to send a duplicate report for this group; thus, the waiting station cancels its corresponding timer.

Example: A query message was received at time 0; the random delay time (in tenths of seconds) for each group is shown next to the group address.

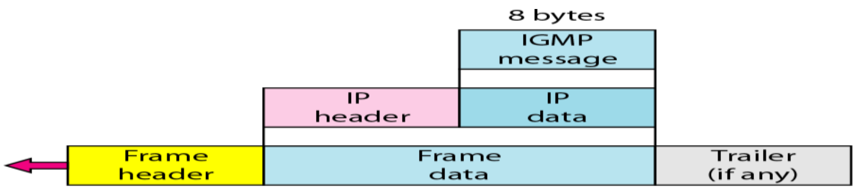


The events occur in this sequence:

- **Time 12:** The timer for 228.42.0.0 in host A expires, and a membership report is sent, which is received by the router and every host including host B which cancels its timer for 228.42.0.0.
- **Time 30:** The timer for 225.14.0.0 in host A expires, and a membership report is sent, which is received by the router and every host including host C which cancels its timer for 225.14.0.0.
- **Time 50:** The timer for 238.71.0.0 in host B expires, and a membership report is sent, which is received by the router and every host.
- **Time 70:** The timer for 230.43.0.0 in host C expires, and a membership report is sent, which is received by the router and every host including host A which cancels its timer for 230.43.0.0.

Encapsulation

The IGMP message is encapsulated in an IP datagram, which is itself encapsulated in a frame. When the message is encapsulated in the IP datagram, the value of TTL must be 1.



Netstat Utility

The netstat utility can be used to find the multicast addresses supported by an interface.
We use **netstat** with three options: **-n, -r**, and **-a**.
-n : gives the numeric versions of IP addresses
-r: gives the routing table
-a: gives all addresses (unicast and multicast).

```
$ netstat –nra
Kernel IP routing table
Destination       Gateway          Mask            Flags      Iface
153.18.16.0       0.0.0.0          255.255.240.0   U          eth0
169.254.0.0       0.0.0.0          255.255.0.0     U          eth0
127.0.0.0         0.0.0.0          255.0.0.0       U          10
224.0.0.0         0.0.0.0          224.0.0.0       U          eth0
0.0.0.0           153.18.31.254    0.0.0.0         UG         eth0
```

Gateway defines the router. Iface defines the interface.

CHAPTER 2

Network Layer: Routing

ROUTING TABLE

A host or a router has a routing table with an entry for each destination or a combination of destinations to route IP packets. A routing table can be of two types:

1. Static Routing
2. Dynamic Routing

Static Routing Table

- A static routing table contains information will be entered manually by the administrator.
- The administrator enters the route for each destination into the table.
- When a table is created, it cannot update automatically when there is a change in the Internet. The table must be manually altered by the administrator.

Note: A static routing table can be used in a small internet that does not change very often.

Dynamic Routing Table

- A dynamic routing table is updated periodically by using one of the dynamic routing protocols such as RIP, OSPF, or BGP.
- Whenever there is a change in the Internet, such as a shutdown of a router or breaking of a link then the dynamic routing protocols update all the tables in the routers automatically.
- The routers in a big internet need to be updated dynamically for efficient delivery of the IP packets.

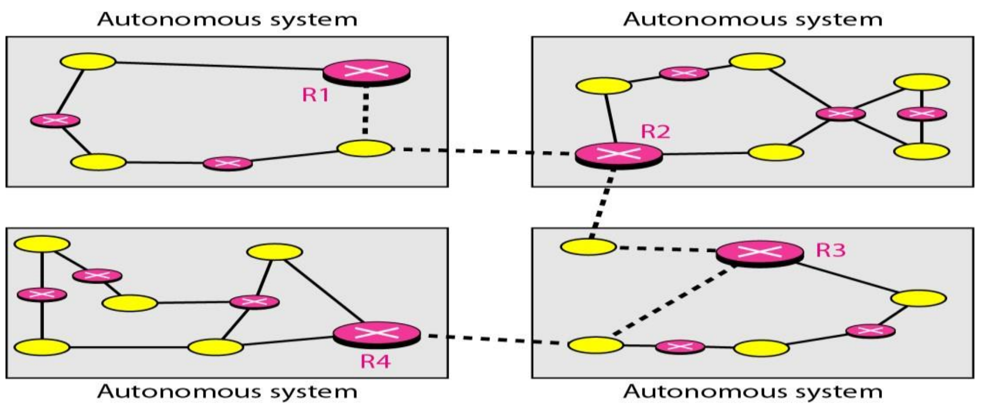
ROUTING PROTOCOLS

- Routing protocols have been created in response to the demand for dynamic routing tables.
- A Routing protocol is a combination of rules and procedures that lets routers in the internet inform each other of changes
- Routing Protocols allows routers to share whatever they know about the internet or their neighborhood.
- The sharing of information allows a router in one location (Hyderabad) to know about the failure of a network in other location (Bangalore).
- The routing protocols also include procedures for combining information received from other routers.

Intra and Inter domain Routing

An internet is divided into **Autonomous Systems** because internet is so large that one routing protocol cannot handle the task of updating the routing tables of all routers.

- **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.**
- Routing between autonomous systems is referred to as **Inter-Domain Routing**.

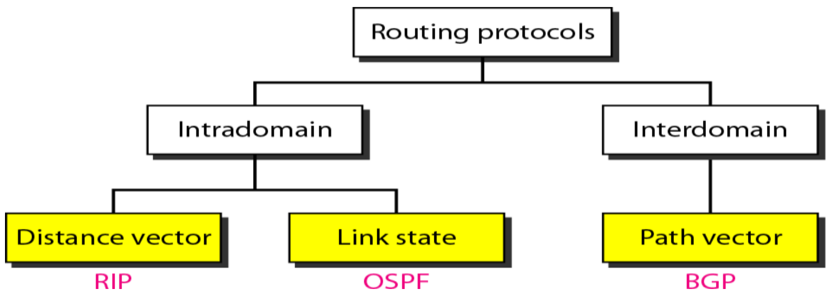


There are two Intra-domain routing protocols:

- Distance Vector Routing (RIP is an implementation of DVR)
- Link State Routing (OSPF is an implementation of LSR)

There is one Inter-Domain routing protocol:

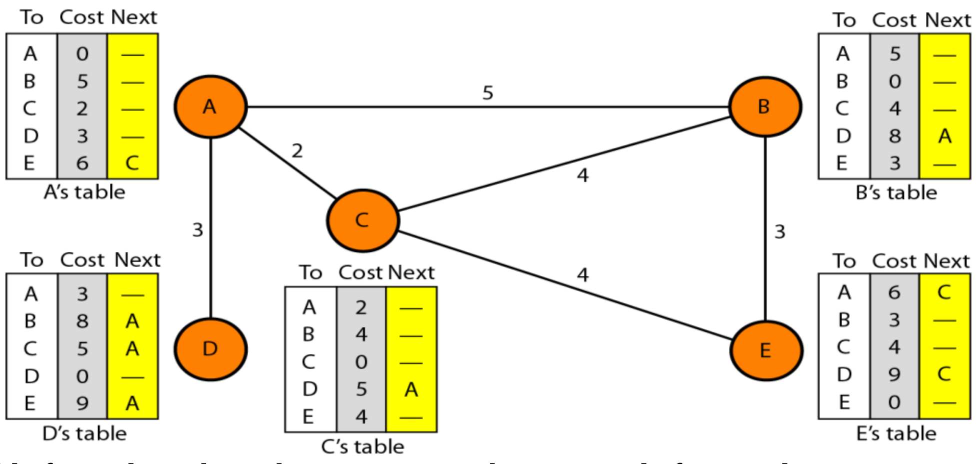
- Path Vector Routing (BGP is an implementation of PVR)



DISTANCE VECTOR ROUTING

In distance vector routing, the least-cost route between any two nodes is the route with minimum distance. The term **Vector** means a **Table**.

- In this protocol each node maintains a table of minimum distances to every node.
- The table at each node also guides the packets to the desired node by showing the next hop in the route.



To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

A's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	8	A
E	3	—

B's table

To	Cost	Next
A	3	—
B	8	A
C	5	A
D	0	—
E	9	A

D's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	5	A
E	4	—

C's table

To	Cost	Next
A	6	C
B	3	—
C	4	—
D	9	C
E	0	—

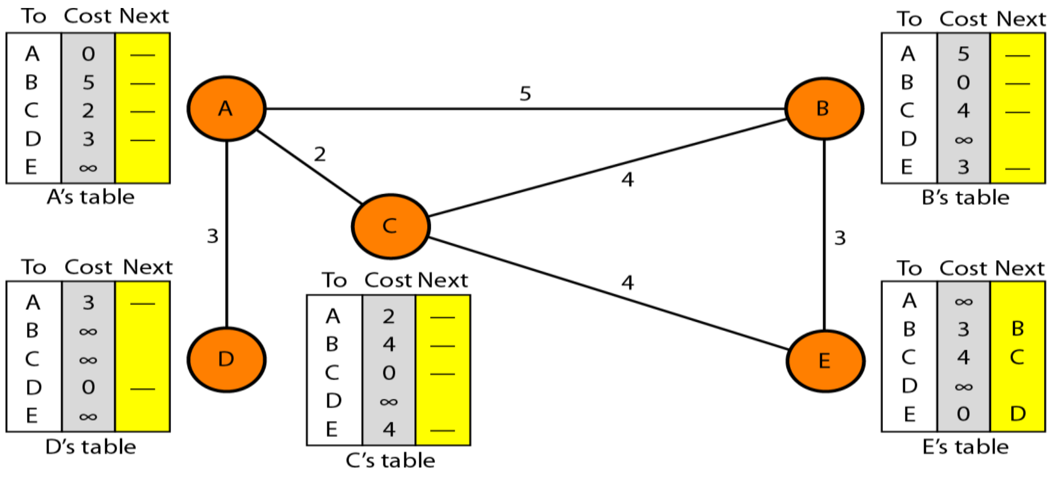
E's table

The table for node A shows how we can reach to any node from node A.
Ex: From node A the least cost to reach node E is 6. The route passes through C.

Initialization

- The above table is the final step of Distance vector routing each node knows about each and every node in the network.
- But at the initial stage each node can only know the distance between itself and its immediate neighbors. Neighbors are the nodes which are directly connected to the node.

The below figre shows the initialization of the table:



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

A's table

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

B's table

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

D's table

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

C's table

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

E's table

- Each node will take the immediate neighbor distance into the table.
- The distance for any entry that is not a neighbor is marked as infinite.
- Infinite means Unreachable.

Sharing

The idea behind Distance Vector Routing is the **sharing** of information between **Neighbors**.
By observing the above figure:

1. Node A does not know about node E but node C knows how to reach node E.
   If Node C shares its routing table with node A then node A can also know how to reach E.

2. Node C does not know how to reach Node D but Node A knows how to reach Node D. If Node A shares its routing table with node C then node C can also know how to reach node A.
   (i.e.) Immediate Neighbor's nodes A and C can improve their routing tables if they share each other's routing tables.

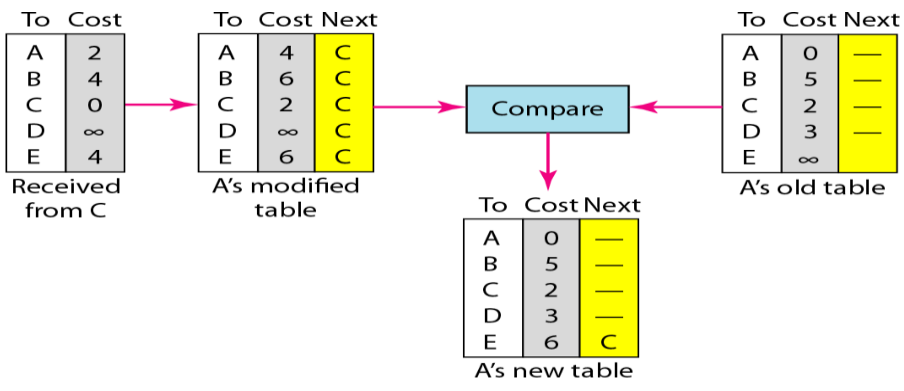
Problem: How many columns of the table must be shared with each neighbor?

- A node is not aware of a neighbor's table.
- A node can send only the first two columns of its table to any neighbor.
- Because the third column of a table next hop is not useful for the neighbor.
- When the neighbor receives a table, third column needs to be replaced with the sender's name.

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.
   Example:
   - If node C claims that its distance to a destination is **x** km
   - The distance between A and C is **y** km then,
   - The distance between A and that destination via C is (**x + y)** km.
2. The receiving node needs to add the name of the sending node to each row as the third column 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. For example, suppose node C has previously advertised a route to node X with distance 3.Suppose that now there is no path between C and X; node C now advertises this route with a distance of infinity. Node A must not ignore this value even though its old entry is smaller. The old route does not exist anymore. The new route has a distanceof infinity.



Note:

1. The modified table shows how to reach A from A via C. If A needs to reach itself via C, it needs to go to C and come back, so the distance will be 4 (A→C=2 and C→A=2).

2. The only benefit from this updating of node A is that A now knows how to reach E with cost=6 via C.

Each node can update its table by using the tables received from other nodes. If there is no change in the network itself, such as a failure in a link, each node reaches a stable condition in which the contents of its table remain the same.

When to Share

When does a node send its partial routing table (only two columns) to all its immediate neighbors?

1. **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.
2. **Triggered Update** A node sends its two-column routing table to its neighbors anytime there is a change in its routing table.

The change can result from the following:

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

Count to Infinity Problems

Two-Node Loop Instability

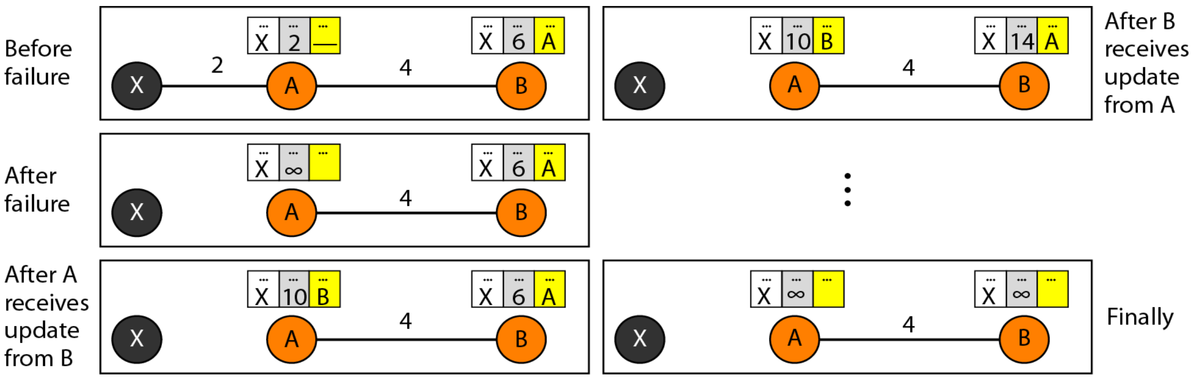
A problem with distance vector routing is instability, which means that a network using this protocol can become unstable.

Consider the above figure:

It shows a system with 3 nodes: X, A and B.

- At the beginning, both nodes A and B know how to reach node X.
- But suddenly, the link between A and X fails.
- Node A changes its table.

If A can send its table to B immediately there will be no problem because B can identify by looking at the table value ∞.



Problem is: The system becomes unstable if B sends its routing table to A before receiving A's routing table.

- Node A receives the update and, assuming that B has found a way to reach X and immediately updates its routing table.

- Based on the triggered update strategy, A sends its new update to B.
- Now B thinks that something has been changed around A and updates its routing table.
- The cost of reaching X increases gradually until it reaches infinity.
- At this moment, both A and B know that X cannot be reached.

During this counting to infinity the system is not stable:

- Node A thinks that the route to X is via B.
- Node B thinks that the route to X is via A.
- If A receives a packet destined for X, it goes to B and then comes back to A.
- If B receives a packet destined for X, it goes to A and comes back to B.
- Packets bounce between A and B, creating a two-node loop problem.

Solution for Two-node loop instability or Two Node loop count to Infinity problem:

i. Defining Infinity
ii. Split Horizon
iii. Split Horizon and Poison Reverse.

Defining Infinity

- Here we define Infinity to smaller number.
- Most implementations of the distance vector protocol define the distance between each node to be 1 and define 16 as infinity. This means that the size of the network in each direction cannot exceed 15 hops.
- If we give 16 as infinity, Distance vector routing cannot be used for large systems.

Split Horizon

- In this strategy, instead of flooding the table through each interface, each node sends only part of its table through each interface.
- According to its table, node B thinks that the optimum route to reach X is via A. B does not need to advertise this piece of information to A.
- The information has come from A (A already knows). Taking information from node A, modifying it, and sending it back to node A creates the confusion.
- In this case node B eliminates the last line of its routing table before it sends it to A.
- In this case, node A keeps the value of infinity as the distance to X.
- Later when node A sends its routing table to B, node B also corrects its routing table.
- The system becomes stable after the first update: both node A and B know that X is not reachable.

Drawback of split horizon:

- The distance vector protocol uses a timer, and if there is no news about a route, the node deletes the route from its table.
- When node B in the previous case eliminates the route to X from its advertisement to A.
- Node A cannot guess that this is due to the split horizon strategy (the source of information was A) or because B has not received any news about X recently.

Split Horizon and Poison Reverse

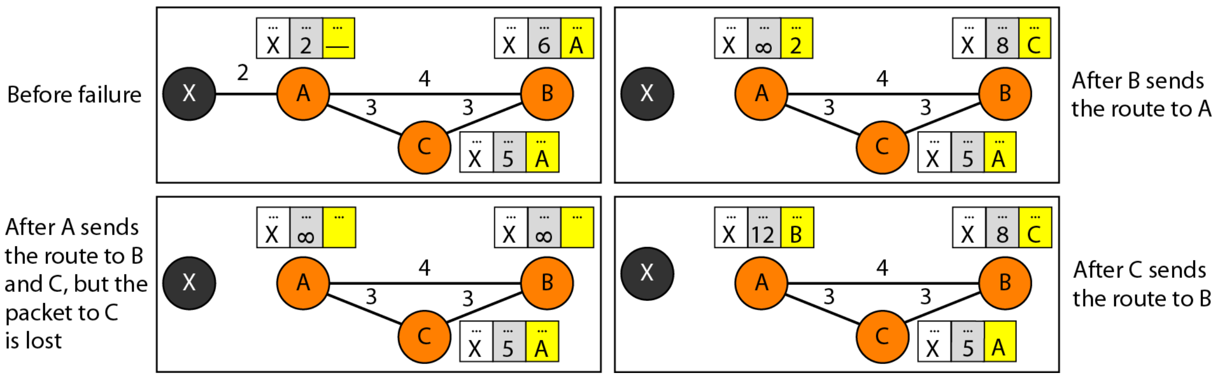
- To overcome the drawback of Split Horizon Strategy we need to combine the Split horizon with Poison Reverse strategy.

- 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."

Three-Node Instability or Three node count to infinity problem

Split horizon strategy combined with poison reverse is sufficient for avoiding Two node instability problem but if the instability is between three nodes, stability cannot be guaranteed.

Consider the below figure:


Before failure
After B sends the route to A
After A sends the route to B and C, but the packet to C is lost
After C sends the route to B
X
A
B
C
2
4
3
3
X 2 —
X 6 A
X 5 A
X ∞ 2
X 8 C
X ∞
X 12 B

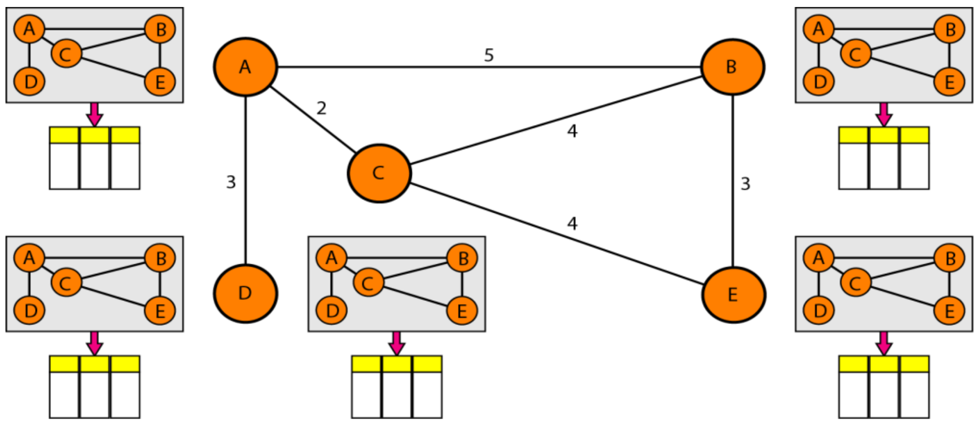
- After finding that X is not reachable, node A sends a packet to B and C to inform them of the situation.
- Node B immediately updates its table, but the packet to C is lost in the network and never reaches C.
- Node C remains in the dark and still thinks that there is a route to X via A with a distance of 5.
- After a while, node C sends its routing table to B, which includes the route to X.
- Node B is fooled here. It receives information on the route to X from C and according to the algorithm it updates its table showing the route to X via C with a cost of 8.
- This information has come from C, not from A, so after a while node B may advertise this route to A.
- Now A is fooled and updates its table to show that A can reach X via B with a cost of 12 and the loop continues.
- Now A advertises the route to X to C with increased cost but not to B.
- Node C then advertises the route to B with an increased cost. Node B does the same to A and so on.
- The loop stops when the cost in each node reaches infinity.

<u>Link State Routing (LSR)</u>

In Link State Routing, Each node in the domain has the entire topology of the domain –

- List of nodes and links
- How they are connected including the type
- Cost (metric)
- Condition of the links (up or down)

The node can use Dijkstra's algorithm to build a routing table.



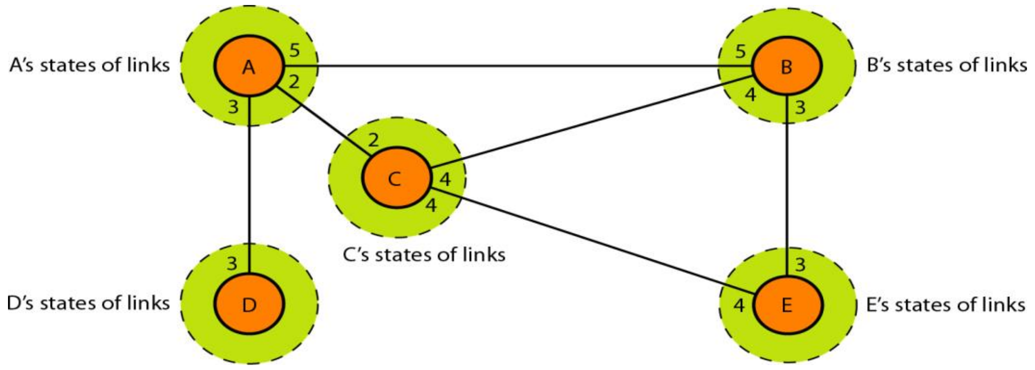
The above figure shows a Simple domain with Five Nodes.

- Each node uses the same topology to create a routing table, but the routing table for each node is unique because the calculations are based on different interpretations of the topology.
- The topology must be dynamic, representing the latest state of each node and each link.
- If there are changes in any point in the network the topology must be updated for each node.

For Example: a link is down then each and every node in the domain should update this change.

How can a common topology be dynamic and stored in each node?

In LSR each node in the domain has the partial knowledge about the state of its links. The state means its type, condition, and cost.



Consider the above figure that shows List of nodes and their partial knowledge.

Node A knows that it is connected

- To Node B with metric 5
- To Node C with metric 2
- To Node D with metric 3

Node C knows that it is connected

- To Node A with metric 2
- To Node B with metric 4
- To Node E with metric 4

Although there is an overlap in the knowledge, the overlap guarantees the creation of a common topology-a picture of the whole domain for each node.

Building Routing Tables

In link state routing, four sets of actions are required to ensure that each node has the routing table showing the least-cost node to every other node.

1. Creation of the states of the links by each node, called the Link State Packet (LSP).
2. Dissemination (Distribution) of LSPs to every other router called **Flooding**. The flooding can be done in an efficient and reliable way.
3. Formation of a shortest path tree for each node.
4. Calculation of a routing table based on the shortest path tree.

Creation of Link State Packet (LSP)

A link state packet can carry a large amount of information such as the node identity, the list of links, a sequence number, and age etc.

- Node identity and the List of links are needed to make the topology.
- Sequence number facilitates flooding and distinguishes new LSPs from old ones.
- Age prevents old LSP's from remaining LSP's in the domain for a long time.

LSPs are generated on two occasions:

1. **When there is a change in the topology of the domain:**
   Triggering of LSP dissemination is the main way of quickly informing any node in the domain to update its topology.
2. **On a periodic basis:**
   - It is done to ensure that old information is removed from the domain.
   - The timer set for periodic dissemination is normally in the range of 60 min or 2 hours based on the implementation.
   - A longer period ensures that flooding does not create too much traffic on the network.

Note: As a matter of fact, there is no actual need for this type of LSP dissemination.

Flooding of LSP's

After a node has prepared an LSP, it must be disseminated to all other nodes, not only to its neighbors. The process is called flooding.

Flooding will be done based on the following:

1. The creating node sends a copy of the LSP out of each interface.
2. A node that receives an LSP compares it with the copy it may already have.
   Each and every LSP will be given a Sequence number at the time of their creation.
   Comparison of sequence numbers determines which LSP is older and which LSP is latest.
   If the newly arrived LSP is older than the one it already has, then the node discards the LSP.

   If LSP arrived is newer, the node does the following:
   - It discards the old LSP and keeps the new one.
   - It sends a copy of it out of each interface except the one from which the packet arrived. This guarantees that flooding stops somewhere in the domain where a node has only one interface.

Formation of Shortest Path Tree: Dijkstra Algorithm

- After receiving all LSPs, each node will have a copy of the whole topology.
- The topology is not sufficient to find the shortest path to every other node; a shortest path

tree is needed.

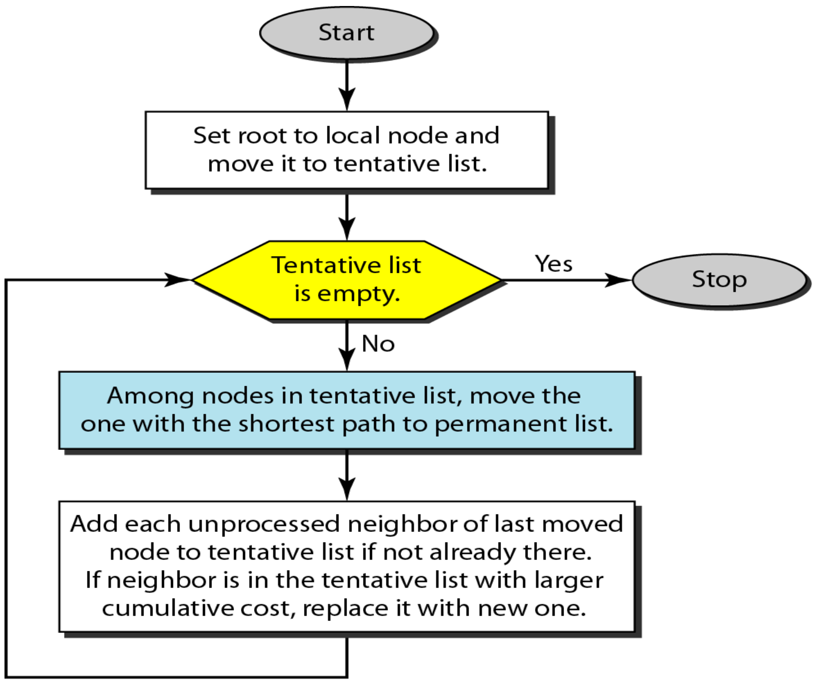
- A tree is a graph of nodes and links, where one node is called Root.
- A shortest path tree is a tree in which the path between the root and every other node is the shortest.
- The Dijkstra's algorithm creates a shortest path tree from a graph.

The algorithm divides the nodes into two sets:

1. Tentative nodes
2. Permanent nodes

Dijkstra's algorithm finds the neighbors of a current node, makes them tentative, examines them, and if they pass the criteria, makes them permanent.

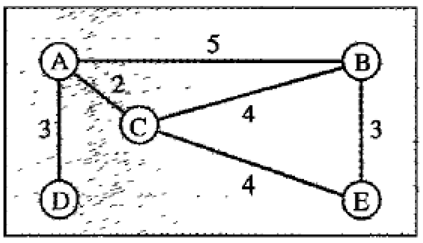
Flow Chart of Dijksta's Algorithm


Start
Set root to local node and move it to tentative list.
Tentative list is empty.
Yes
Stop
No
Among nodes in tentative list, move the one with the shortest path to permanent list.
Add each unprocessed neighbor of last moved node to tentative list if not already there. If neighbor is in the tentative list with larger cumulative cost, replace it with new one.

Example:

Consider the below graph with five nodes: A,B,C,D,E.

- Apply the Dijkstra's algorithm to node A.
- To find the shortest path in each step, we need the cumulative cost from the root to each node, which is shown next to the node.

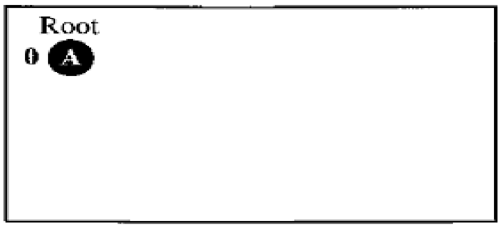


Topology

At the end of each step, we show the permanent (filled circles) and the tentative (open circles) nodes and lists with the cumulative costs.

Step 1: We make node A the root of the tree and move it to the tentative list. Our two lists are

Permanent list: **Empty** Tentative list: **A(0)**

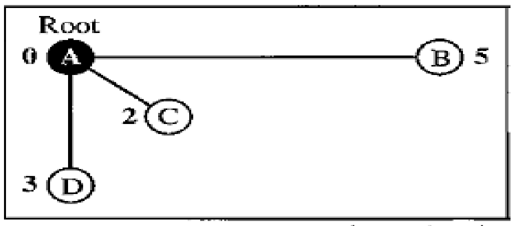


1. Set root to A and move A to tentative list.

Step 2: Node A has the shortest cumulative cost from all nodes in the tentative list.

We move A to the permanent list and add all neighbors of A to the tentative list. Our new lists are

Permanent list: A(0) Tentative list: B(5), C(2), D(3)

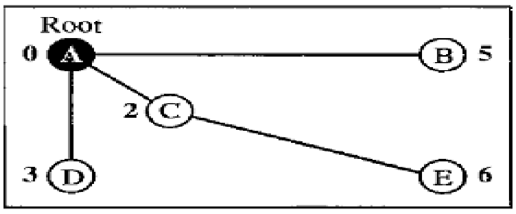


2. Move A to permanent list and add B, C, and D to tentative list.

Step 3: Node C has the shortest cumulative cost from all nodes in the tentative list.

- We move C to the permanent list.
- Node C has three neighbors, but node A is already processed, which makes the unprocessed neighbors just B and E.
- However, B is already in the tentative list with a cumulative cost of 5.
- Node A could also reach node B through C with a cumulative cost of 6.
- Since 5 is less than 6, we keep node B with a cumulative cost of 5 in the tentative list and do not replace it.

Our new lists are : Permanent list: A(0), C(2) Tentative list: B(5), D(3), E(6).

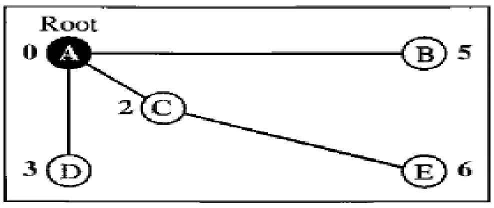


3. Move C to permanent and add E to tentative list.

Step 4: Node D has the shortest cumulative cost of all the nodes in the tentative list.

- We move D to the permanent list. Node D has no unprocessed neighbor to be added to the tentative list.

Our new lists are: Permanent List: A(0), C(2), D(3) Tentative List: B(5), E(6).

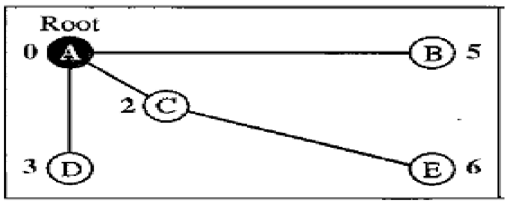


4. Move D to permanent list.

Step 5: Node B has the shortest cumulative cost of all the nodes in the tentative list.

- We move B to the permanent list. We need to add all unprocessed neighbors of B to the tentative list (i.e. just node E).
- E(6) is already in the list with a smaller cumulative cost.
- The cumulative cost to node E, as the neighbor of B, is 8. We keep node E(6) in the tentative list.

Our new lists are: Permanent list: A(0), B(5), C(2), D(3) Tentative list: E(6)

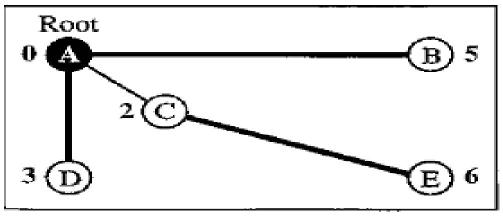


5. Move B to permanent list.

Step 6: Node E has the shortest cumulative cost from all nodes in the tentative list.

- Move E to the permanent list. Node E has no neighbor. Now the tentative list is empty.
- We stop the process here. The shortest path tree is ready for graph ABCDE.

The final lists are: Permanent list: A(0), B(5), C(2), D(3), E(6) Tentative list: Empty



6. Move E to permanent list (tentative list is empty).

Calculation of Routing Table from Shortest Path Tree

- Each node uses the shortest path tree protocol to construct its routing table.
- The routing table shows the cost of reaching each node from the root.

The below table shows routing table for Node A.

Node	Cost	Next Router
A	0	-
B	5	-
C	2	-
D	3	-
E	6	C

PATH VECTOR ROUTING (PVR)

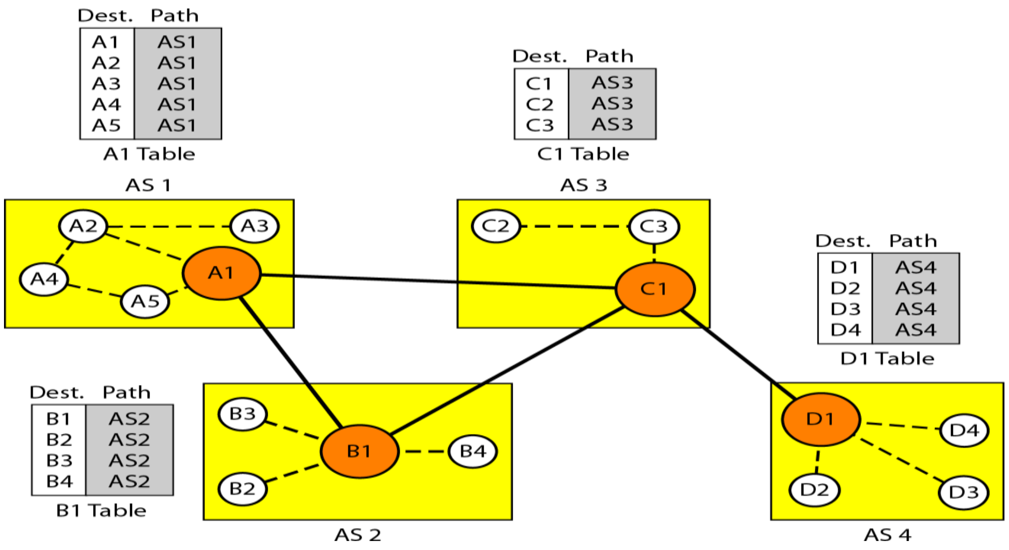
Path vector routing proved to be useful for interdomain routing.

- In path vector routing one node in each autonomous system that acts on behalf of the entire autonomous system. The node is called **Speaker Node**.
- The speaker node in an Autonomous System creates a **Routing Table** and advertises it to speaker nodes in the neighboring Autonomous Systems.
- In PVR only speaker nodes in each Autonomous System can communicate with each other.
- A speaker node advertises the path in its autonomous system or other autonomous systems. (paths such as AS1, AS1-AS2, AS1-AS2-AS4 etc).

Initialization

At the beginning, each speaker node can know only the reachability of nodes inside its autonomous system.

Consider the below figure that shows the initial tables for each speaker node in a system made of four Autonomous Systems.



In the above figure :

- AS1, AS2, AS3, AS4 are the four autonomous systems.
- Node Al, B1, C1, D1 are the Speaker Nodes of Autonomous Systems AS1, AS2, AS3, AS4 respectively.
- The tables of the autonomous systems are created by Speaker Nodes A1, B1, C1 and D1.
- Node Al creates an initial table that shows Al to A5 are located in AS1 and can be reached through it.
- Node Bl advertises that Bl to B4 are located in AS2 and can be reached through Bl and so on.

Sharing

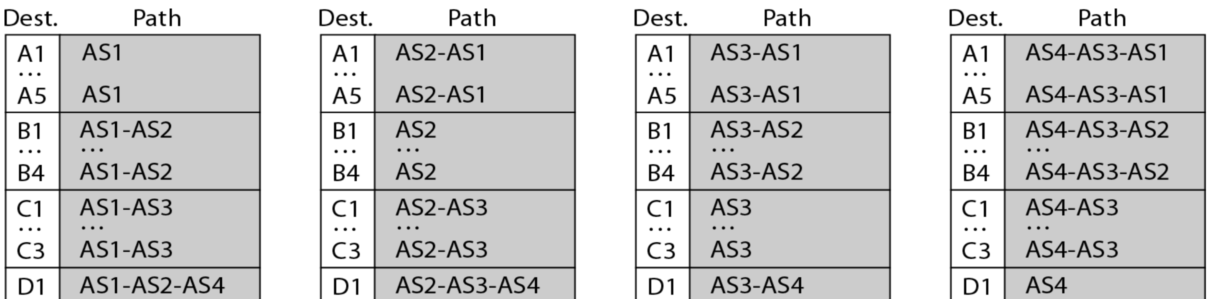
In path vector routing a speaker node in an autonomous system shares its table with immediate neighbors.

- Node Al shares its table with nodes Bl and Cl.
- Node Bl shares its table with A1 and C1.
- Node Cl shares its table with nodes A1, B1, D1.
- Node Dl shares its table with Cl.

Updating

- When a speaker node receives a two-column table from a neighbor, it updates its own table by adding the nodes that are not in its routing table and adding its own autonomous system and the autonomous system that sent the table.
- After a while each speaker has a table and knows how to reach each node in other ASs.

The below figure shows the tables for each speaker node after the system is stabilized.



Dest.	Path
A1	AS1
⋯	
A5	AS1
B1	AS1-AS2
⋯	⋯
B4	AS1-AS2
C1	AS1-AS3
⋯	⋯
C3	AS1-AS3
D1	AS1-AS2-AS4

Dest.	Path
A1	AS2-AS1
⋯	
A5	AS2-AS1
B1	AS2
⋯	⋯
B4	AS2
C1	AS2-AS3
⋯	⋯
C3	AS2-AS3
D1	AS2-AS3-AS4

Dest.	Path
A1	AS3-AS1
⋯	
A5	AS3-AS1
B1	AS3-AS2
⋯	⋯
B4	AS3-AS2
C1	AS3
⋯	⋯
C3	AS3
D1	AS3-AS4

Dest.	Path
A1	AS4-AS3-AS1
⋯	
A5	AS4-AS3-AS1
B1	AS4-AS3-AS2
⋯	⋯
B4	AS4-AS3-AS2
C1	AS4-AS3
⋯	⋯
C3	AS4-AS3
D1	AS4

By observing the above figure following points to be noted:

- If router Al receives a packet for nodes A3, it knows that the path is in AS1 (i.e.) the packet is at home.
- If Router A1 receives a packet for Dl, it knows that the packet should go from ASl, to AS2 and then AS2 to AS3.
- The routing table shows the path completely.

Advantages of Path Vector Routing

1. Loop Prevention
2. Policy Routing
3. Optimum path

Loop prevention

- The instability of distance vector routing and the creation of loops can be avoided in path vector routing.
- When a router receives a message, it checks to see if its autonomous system is in the path list to the destination.
- If it is, looping is involved and the message is ignored.

Policy routing

- When a router receives a message, it can check the path. If one of the autonomous systems listed in the path is against router policy, it can ignore that path and that destination.
- Router does not update its routing table with this path, and router does not send this message to its neighbors.

Optimum path

- The optimum path in path vector routing is the path to a destination that is the best for the organization that runs the autonomous system.
- Metrics cannot include in this route because each autonomous system that is included in the path may use a different criterion for the metric.
- One system may use RIP that defines hop count as the metric, another system may use OSPF with minimum delay defined as the metric.
- The optimum path is the path that fits the organization.

Example: Consider the above figure after updating the table:

- Each autonomous system may have more than one path to a destination.
- A path from AS4 to AS1 can be AS4-AS3-AS2-AS1, or it can be AS4-AS3-AS1.
- For the table entry we have selected the path that had the smaller number of autonomous systems (i.e. AS4-AS3-AS1).
- But we may take other path when the criteria are related to security, safety and reliability can also be applied.

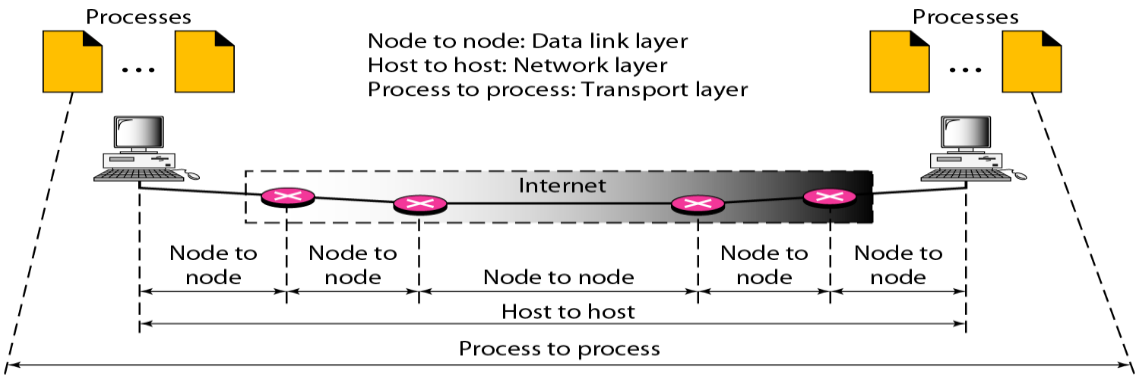
TRANSPORT LAYER

PROCESS TO PROCESS DELIVERY

A Process is an Application program that runs on the host. At any moment several processes may be running on the source host and destination host.

Transport Layer is responsible for delivery of a packet from one process on source host to another process on destination host.

Two processes communicate in a client/server relationship.



Client/Server Paradigm

A Client is a process on a local host, whereas Server is a process on remote host. A Client needs services from Server. Both Client and Server processes have the same name.

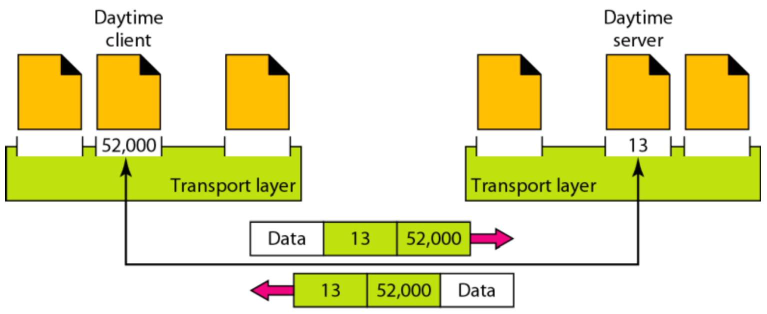
Example: To get the day and time from a remote machine, we need a Daytime client process running on the local host and a Daytime server process running on a remote machine.

For communication to be done between two processes we must have to define following:

1. Local host
2. Local process
3. Remote host
4. Remote process

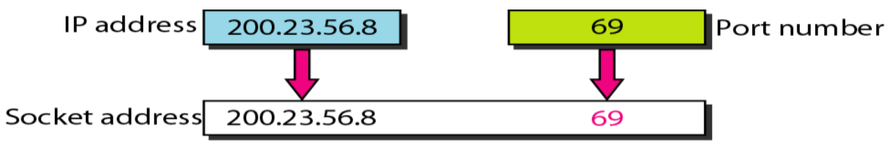
(a)Addressing

- Transport layer needs an address called a **Port Number** or **Port Address** to choose among multiple processes running on the destination host.
- The port numbers are 16-bit integers between 0 and 65,535.
- Destination Port number is needed for Delivery. Source Port number is needed for the Reply.
- At the client side the port numbers are defined randomly whereas at the server side it uses Well-Known Port numbers.



(b)Socket address

- The combination of an IP address and a port number is called a **Socket address**. UDP or TCP header contains the Port numbers.
- A transport layer protocol needs a pair of socket addresses: the client socket address and the server socket address.



(c)Internet Assigned Number Authority (IANA) Ranges

Port Name	Range	Description
Well-known	0-1023	These are assigned and controlled by IANA
Registered	1024-49151	Not Assigned or controlled by IANA but these are registered with IANA to prevent duplication.
Dynamic or Private	49152-65535	These are neither controlled nor registered. They can be used by any process

Multiplexing and Demultiplexing

The addressing mechanism allows multiplexing and de-multiplexing by the transport layer.

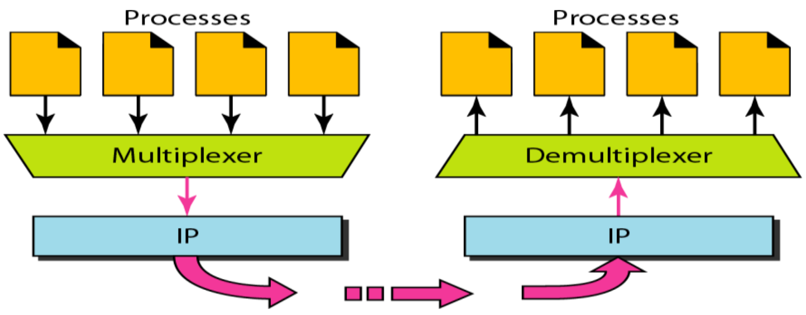
Multiplexing

- At the sender site, there may be several processes that need to send packets. But there is only one transport layer protocol at any time.
- The relationship in multiplexing is a many-to-one.
- The protocol accepts messages from different processes, differentiated by their assigned port numbers.
- After adding the header, the transport layer passes the packet to the network layer.

Demultiplexing

- At the receiver site, the relationship is one-to-many and requires demultiplexing.
- The transport layer receives datagrams from the network layer.

- After error checking and dropping of the header, the transport layer delivers each message to the appropriate process based on the port number.



Connectionless Service versus Connection-Oriented Service

Connection-Oriented Service (CO)	**Connectionless Service (CL)**
• In a CO service, to transfer the data 3 steps are followed by sender & receiver: 1. A connection is established 2. Data are transferred.	• In a CL service to transfer the data there is no need for connection establishment or connection release.
3. At the end, the connection is released. • The packets are numbered. • The packets arrived sequentially at the destination. Example: TCP, SCTP.	• The packets are not numbered. • The packets may arrive out of sequence at the destination. Example: UDP

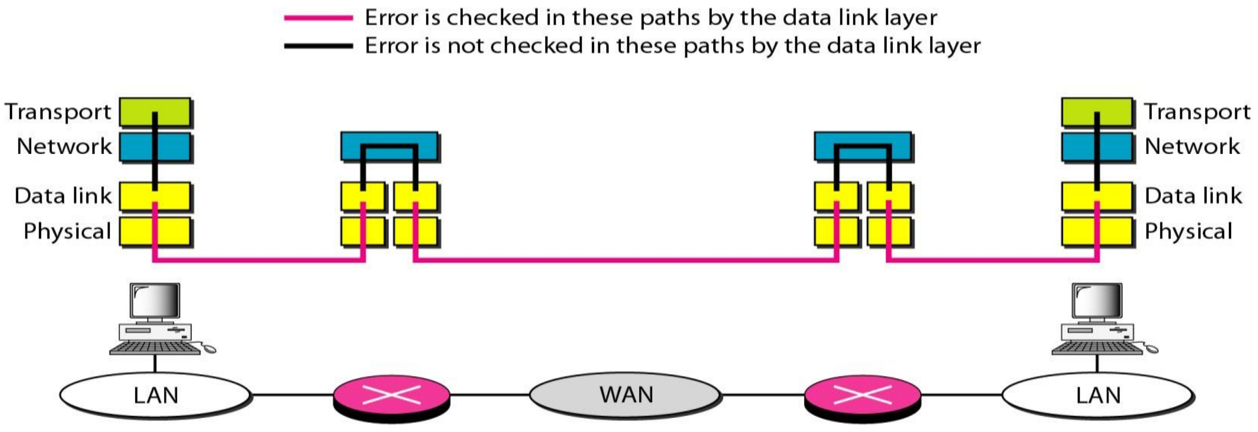
Reliable Versus Unreliable service

Reliable service	**Unreliable service**
• Reliable service implements flow and error control at the transport layer. • It is a slower and more complex service. • Example: TCP	• Unreliable service does not implements flow and error control at the transport layer. • It is a faster service. • Example: UDP

Note: Reliability **(**Flow and Error control) mechanisms need to be implemented in both Data-link layer and transport layer.

- Reliability at the data link layer is between two nodes. It is node-to-node communication.
- The network layer in the Internet is unreliable (best-effort delivery). Hence we need to implement the reliability between two ends at the transport layer.

- Error control at the data link layer does not guarantee error control at the transport layer.


Error is checked in these paths by the data link layer
Error is not checked in these paths by the data link layer
Transport
Network
Data link
Physical
Transport
Network
Data link
Physical
LAN
WAN
LAN

TRANSPORT LAYER PROTOCOL

There are 3 transport layer protocols are implemented:

1. UDP
2. TCP
3. SCTP

USER DATAGRAM PROTOCOL (UDP)

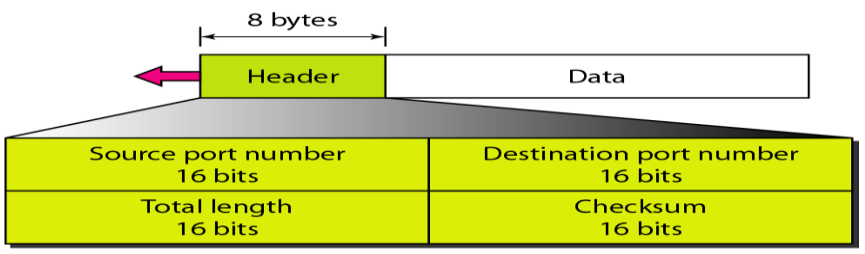
- UDP is called a connectionless, unreliable transport protocol.
- UDP is a very simple protocol using a minimum of overhead. UDP performs very limited error checking.
- UDP is used when a process wants to send a small message and does not need reliability.
- Sending a small message by using UDP takes much less interaction between the sender and receiver than using TCP or SCTP.

Well-Known Ports for UDP

Port	**Protocol**	**Description**
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
53	Nameserver	Domain Name Service
67	BOOTPs	Server port to download bootstrap information
68	BOOTPc	Client port to download bootstrap information
161	SNMP	Simple Network Management Protocol

User Datagram

UDP packets are called User Datagrams. Datagrams have fixed size header of length 8 bytes.



Fields of user datagram are:

Source port number (16 bits)

- This is the port number used by the process running on the source host.

- If the source host is the client (a client sending a request) the port number is an ephemeral port number requested by the process and chosen by the UDP software running on the source host.
- If the source host is the server (a server sending a response) the port number is a well-known port number.

Destination port number

- This is the port number used by the process running on the destination host.
- If the destination host is the server (a client sending a request) the port number is a well-known port number.
- If the destination host is the client (a server sending a response) the port number is an ephemeral port number, the server copies the ephemeral port number it has received in the request packet.

Note: Source and Destination port number can range from 0 to 65,535.

Total Length (16 bits)

The total length includes UDP header plus Data, total length ranges from 0-65535.

Checksum (16 bits)

This field is used to detect errors over the entire user datagram (header plus data).

UDP Operation

UDP uses the following four concepts:

1. Connectionless Service
2. No Flow and Error control
3. Encapsulation and Decapsulation
4. Queuing

Connectionless Services

- UDP provides a connectionless service. There is no connection establishment and no connection termination. Each user datagram sent by UDP is an independent datagram.
- The user datagrams are not numbered. Each user datagram can travel on a different path.
- There is no relationship between the different user datagrams even if they are coming from the same source process and going to the same destination program.
- UDP processes are capable of sending short messages only.

No Flow and Error Control

- There is no error control mechanism in UDP except for the checksum.
- There is no flow control in UDP; the receiver may overflow with incoming messages. The sender does not know if a message has been lost or duplicated.
- When the receiver detects an error through the checksum, the user datagram is silently discarded.

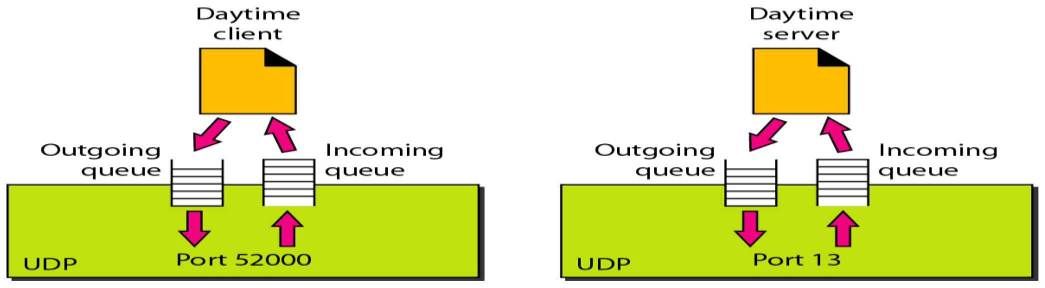
Encapsulation and Decapsulation

To send a message from one process to another, the UDP protocol encapsulates and decapsulates messages in an IP datagram.

Queuing

- Queues are associated with ports in UDP. Queues are associated with each process.
- Queues are created when a process is started. There are two types of queues are created: Incoming queue and Outgoing queue.

- A process wants to communicate with multiple processes; it obtains only one port number and one outgoing queue and one incoming queue.
- The queues function as long as the process is running. When the process terminates, the queues are destroyed.



Queues at Client site

- The queues opened by the client are identified by ephemeral port numbers.
- The client process can send messages to the outgoing queue by using the source port number specified in the request.
- After adding the UDP header, UDP removes the messages one by one and delivers them to IP.
- If there is an overflow in an outgoing queue then the operating system can ask the client process to wait before sending any more messages.
- When a message arrives for a client, UDP checks to see if an incoming queue has been created for the port number specified in the destination port number field of the user datagram.
- If there is such a queue, UDP sends the received user datagram to the end of the queue.
- If there is no such queue, UDP discards the user datagram and asks the ICMP protocol to send a **Port unreachable** message to the server.
- All the incoming messages for one particular client program, whether coming from the same or a different server, are sent to the same queue.
- If there is an overflow in an incoming queue then UDP drops the user datagram and asks for a port unreachable message to be sent to the server.

Queues at Server site

- At the server site a server asks for incoming and outgoing queues using its well-known port, when it starts running. The queues remain open as long as the server is running.
- When a message arrives for a server, UDP checks to see if an incoming queue has been created for the port number specified in the destination port number field of the user datagram.
- If there is such a queue, UDP sends the received user datagram to the end of the queue.
- If there is no such queue, UDP discards the user datagram and asks the ICMP protocol to send a port unreachable message to the client.
- All the incoming messages for one particular server, whether coming from the same or a different client, are sent to the same queue.
- If there is an overflow in an incoming queue then UDP drops the user datagram and asks for a port unreachable message to be sent to the client.
- When a server wants to respond to a client, it sends messages to the outgoing queue,

using the source port number specified in the request.

- After adding the UDP header, it removes the messages one by one and delivers them to IP.
- If there is an overflow in an outgoing queue then the operating system asks the server to wait before sending any more messages.

Uses of UDP

The following lists some uses of the UDP protocol:

- UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control.
- UDP is suitable for a process with internal flow and error control mechanisms.
- UDP is a suitable transport protocol for multicasting.
- UDP is used for management processes such as SNMP.
- UDP is used for Route Updating Protocols such as Routing Information Protocol (RIP).

TRANSMISSION CONTROL PROTOCOL (TCP)

TCP is called a connection-oriented, reliable, process-to-process transport protocol. It adds connection-oriented and reliability features to the services of IP.

TCP Services

The following five services offered by TCP to the processes at the application layer

1. Process-to-Process Communication
2. Full-Duplex Communication
3. Connection-Oriented Service
4. Reliable Service
5. Stream Delivery Service

Process-to-Process Communication

TCP provides process-to-process communication using Well-known port numbers.

Port	Protocol	Description
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
20	FTP, Data	File Transfer Protocol (data connection)
21	FTP, connection	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Service
67	BOOTP	Bootstrap protocol
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

Full-Duplex Communication

TCP offers full-duplex service, in which data can flow in both directions at the same time. Each TCP then has a sending and receiving buffer and segments move in both directions.

Connection-Oriented Service

- When a process at site A wants to send and receive data from another process at site B, the following steps will occur:
  1. The two TCPs establish a virtual connection between them.
  2. Data are exchanged in both directions.
  3. The connection is terminated.
- The TCP segment is encapsulated in an IP datagram and can be sent out of order or lost or corrupted must be resent.
- TCP creates a stream-oriented environment in which it accepts the responsibility of delivering the bytes in order to the other site.

Reliable Service

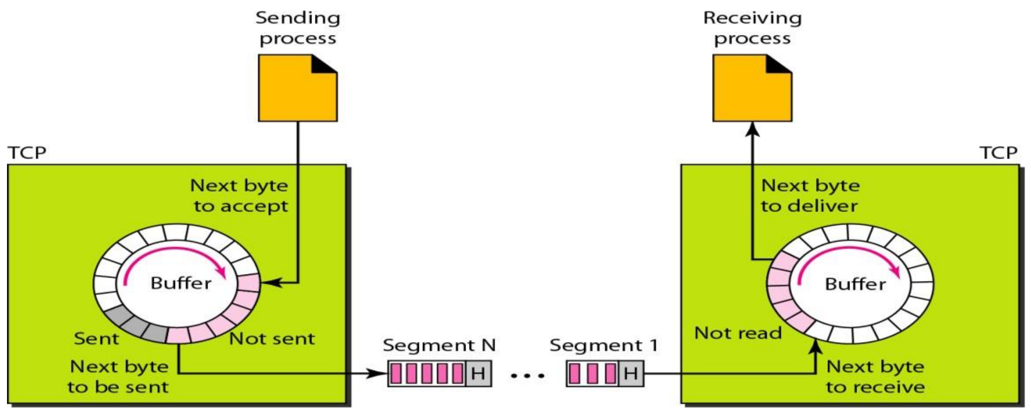
TCP is a reliable transport protocol. It uses an acknowledgment mechanism to check the safe and sound arrival of data.

Stream Delivery Service

- TCP allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes.
- The sending process produces the stream of bytes, and the receiving process consumes them.

Sending and Receiving Buffers

- TCP needs buffers for storage, because the sending and the receiving processes may not write or read data at the same speed.
- The buffers are implemented by using Circular Array where each location carries 1-byte.
- There are two types of buffers are implemented: **Sending buffer** and **Receiving buffer**.



- At the sending site TCP keeps bytes in the buffer that have been sent but not yet acknowledged until it receives an acknowledgment.
- After the bytes in the buffer locations are acknowledged, the locations are recycled and they are available for use by the sending process.
- At the receiver site the circular buffer is divided into two areas:
- One area contains empty chambers to be filled by bytes received from the network.
- Other are contain received bytes that can be read by the receiving process.
- When a byte is read by the receiving process, the chamber is recycled and added to the pool of empty chambers.

Segments

- TCP groups a number of bytes together into a packet called a segment.
- TCP adds a header to each segment and delivers them to the IP layer for transmission.
- The segments are encapsulated in IP datagrams and transmitted.
- The segments are not necessarily the same size.

<u>TCP Features</u>

TCP has following features:

1. Numbering system
2. Flow control
3. Error control
4. Congestion control

Numbering System

TCP uses 3-types of numbering: Byte, Sequence and Acknowledgement numbering.

Byte Number

- All data bytes that are transmitted in a connection will be given a number called Byte number.
- When TCP receives bytes of data from a process, it stores them in the sending buffer and numbers them. Byte numbering is independent in each direction.
- The numbering does not necessarily start from 0.
- The first byte number will be any a random number between **0** and **2^{32} -1.**

Example: If the random number happens to be 1057 and the total data to be sent are 6000 bytes, the bytes are numbered from 1057 to 7056.

Sequence Number

- After the bytes have been numbered, TCP assigns a sequence number to each **Segment** that is being sent.
- The Sequence number for each segment is the **First Byte Number** of carried in that segment.

Example: Suppose a TCP connection is transferring a file of 5000 bytes. Each segment carries 1000 bytes. The first byte number in the first segment is numbered as 10,001. The list of sequence numbers for each segment is:

Segment 1	→	Sequence Number: 10,001 (range: 10,001 to 11,000)
Segment 2	→	Sequence Number: 11,001 (range: 11,001 to 12,000)
Segment 3	→	Sequence Number: 12,001 (range: 12,001 to 13,000)
Segment 4	→	Sequence Number: 13,001 (range: 13,001 to 14,000)
Segment 5	→	Sequence Number: 14,001 (range: 14,001 to 15,000)

Acknowledgment Number

- The acknowledgment number defines the number of the next byte that the party expects to receive.
- An acknowledgment number used to confirm the bytes that have received.
- The acknowledgment number is cumulative, that the party takes the number of the last byte that it has received is safe and sound and adds 1 to the number and announces this sum as the acknowledgment number.

Example: If a party uses 5643 as an acknowledgment number, it has received all bytes from the beginning number up to 5642.

Flow Control

- TCP provides *flow control*. The receiver of the data controls the amount of data that are to be sent by the sender.
- This is done to prevent the receiver from being overwhelmed with data.
- The numbering system allows TCP to use a byte-oriented flow control.

Error Control

- TCP implements an error control mechanism to provide reliable service.
- Error control is byte-oriented. Error control considers a segment as the unit of data for error detection.

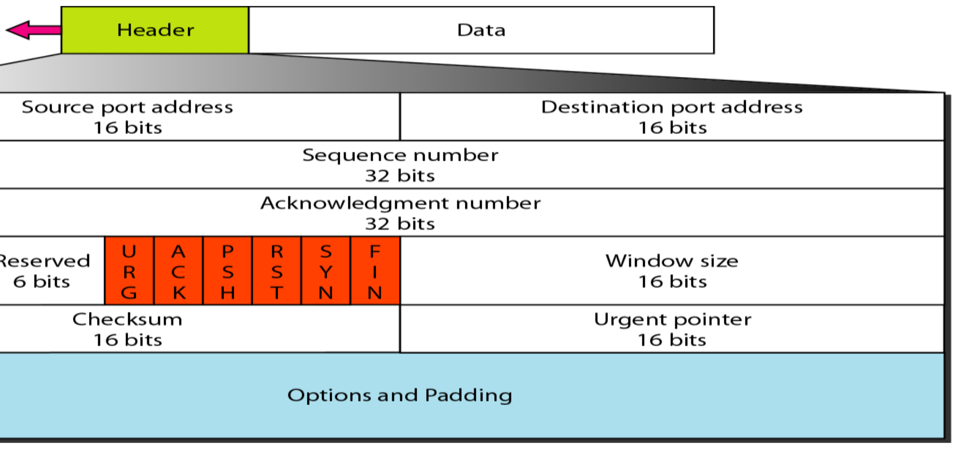
Congestion Control

- TCP takes into account congestion in the network.
- The amount of data sent by a sender is not only controlled by the receiver (flow control), but is also determined by the level of congestion in the network.

<u>TCP Segment</u>

- A packet in TCP is called a segment.
- A segment consists of Segment Header and Data.
- The segment Header consists of 20-60 Byte. Header is 20 bytes if there are no options. If there are any options the length of the header varied upto 60 bytes.

The segment format can be given as:



- **Source port address (16 bit)**
  It defines the port number of the application program in the host that is sending the segment.
- **Destination port address (16-bit)**
  It defines the port number of the application program in the host that is receiving the segment.
- **Sequence number (32 bit)**
  This field defines the number assigned to the first byte of data contained in this segment. The sequence number tells the destination which byte in this sequence comprises the first byte in the segment.
- **Acknowledgment number (32 bit)**
  This field defines the byte number that the receiver of the segment is expecting to receive from the other party. Acknowledgment and data can be piggybacked together.
- **Header length (4 bit)**

This field indicates the number of 4-byte words in the TCP header.The length of the header can be between 20 and 60 bytes.
The value of this field can be between 5 (5 x 4 =20) and 15 (15 x 4 =60).

- **Reserved (6 bit)**
  This field is reserved for future use.
- **Window size (16 bit)**
  This field defines the size of the window in bytes that the other party must maintain.The maximum size of the window is 65,535 bytes.
  This value is normally referred to as the receiving window (rwnd) and is determined bythe receiver.
- **Checksum (16 bit)**
  The checksum field in TCP is mandatory. For the TCP pseudoheader, the value for theprotocol field is 6.
- **Urgent pointer (l6-bit)**
  This field is valid only if the urgent flag is set. It is used when the segment containsurgent data.
  It defines the number that must be added to the sequence number to obtain the number ofthe last urgent byte in the data section of the segment.
- **Options (up to 40 bytes)** It defines the optional information in the TCP header.
- **Control flags (6 bit)**
  This field defines 6 different control bits or flags. One or more of these bits can be set at a time. These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.

Flag	**Description**
URG	The value of the urgent pointer field is valid
ACK	The value of the acknowledgement field is valid.
PSH	Push the data
RST	Reset the connection
SYN	Synchronize sequence numbers during connection
FIN	Terminate the connection

<u>TCP Connection</u>

- TCP is connection-oriented transport protocol establishes a virtual path between the source and destination.
- All the segments belonging to a message are then sent over this virtual path.
- TCP operates at a higher level. TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself.
- If a segment is lost or corrupted then the segment is retransmitted.
- If a segment arrives out of order then TCP holds it until the missing segments arrives. IP is unaware of this reordering.

Connection-oriented TCP transmission requires three phases:

1. Connection establishment
2. Data transfer
3. Connection termination

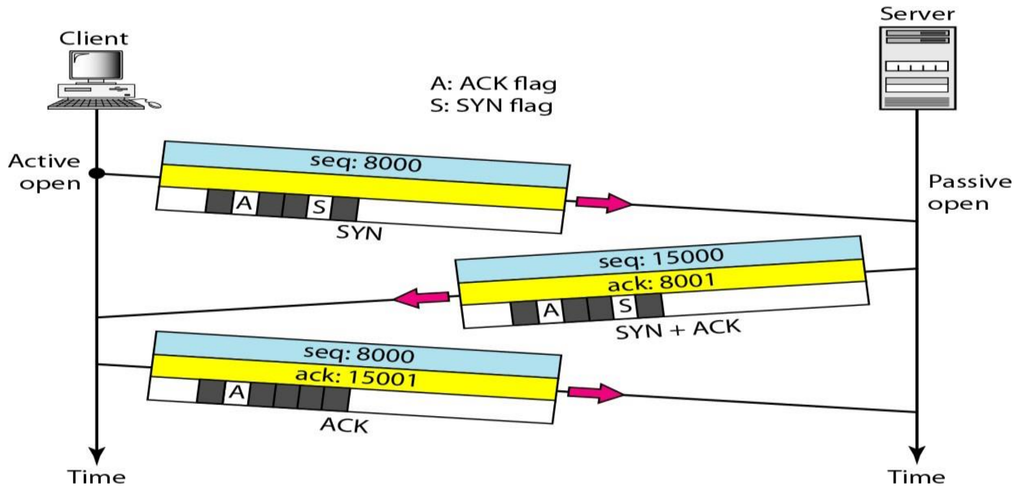
Connection Establishment Phase

TCP uses Three way handshaking to establish a connection. The three way handshaking uses the sequence number, the acknowledgment number, the control flags and the window size.

- The process starts with the server. The server program tells its TCP that it is ready to accept a connection. This is called a request for a **Passive open**.
- A client that wishes to connect to an open server tells its TCP that it needs to be connected to that particular server. The client program issues a request for an **Active open.**
- Now TCP starts the Three way handshaking protocol process.

The three steps in this phase are as follows:

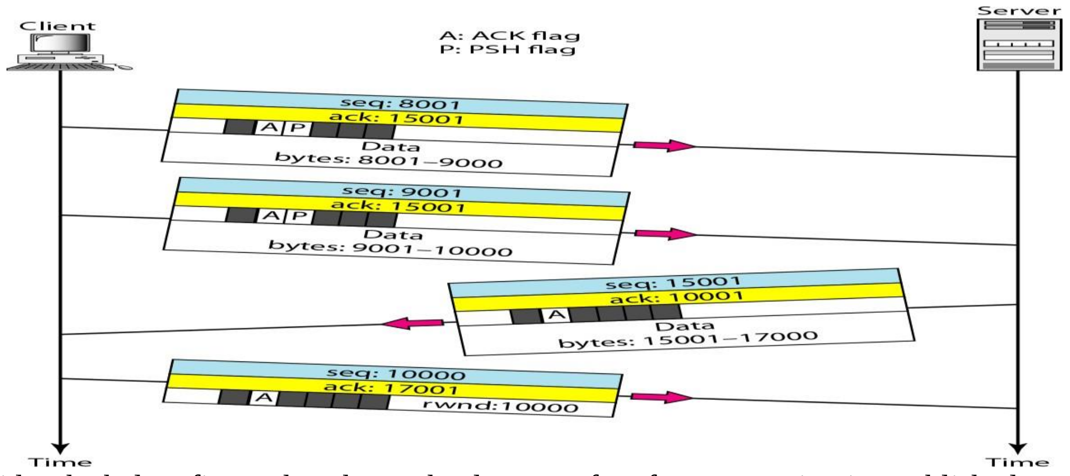
1. The client sends the first segment a SYN segment. The SYN flag in control field is set. This segment is for synchronization of sequence numbers. The SYN segment does not carry real data. SYN consumes one sequence number. When the data transfer starts the sequence number is incremented by 1.
2. The server sends the second segment a SYN + ACK segment with 2 flag bits set: SYN and ACK. This segment has a dual purpose. It is a SYN segment for communication in the other direction and serves as the acknowledgment for the SYN segment. It consumes one sequence number.
3. The client sends the third segment an ACK segment. It acknowledges the receipt of the second segment with the ACK flag and acknowledgment number field. The sequence number in this segment is the same as the one in the SYN segment.



Data Transfer Phase

- After connection is established, bidirectional data transfer can take place.

- The client and server can both send data and acknowledgments. The acknowledgment is piggybacked with the data.



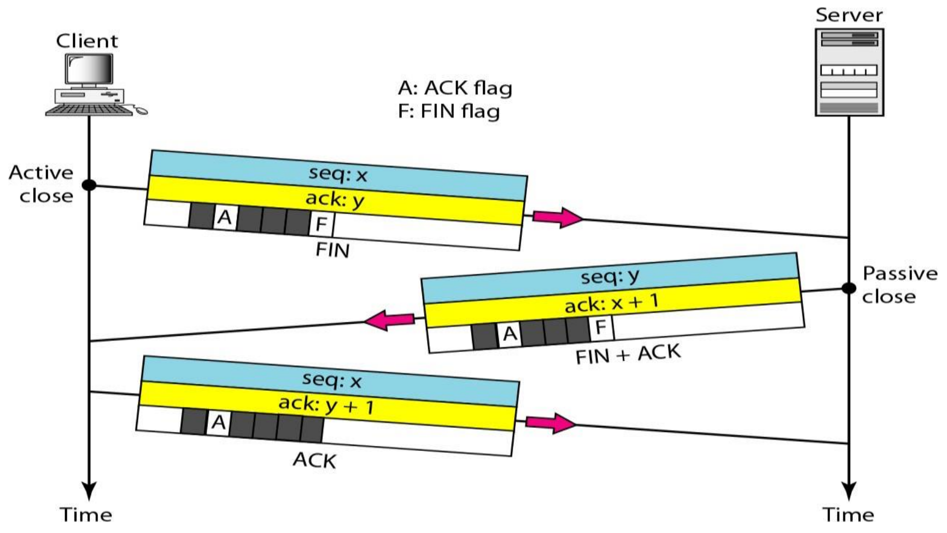
Consider the below figure that shows the data transfer after connection is established.

- The client sends 2000 bytes of data in two segments.
- The server then sends 2000 bytes in one segment.
- The client sends one more segment.
- The first three segments carry both data and acknowledgment, but the last segment carries only an acknowledgment because there are no more data to be sent.
- The data segments sent by the client have the PSH (push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received.
- The segment from the server does not set the push flag.

Connection Termination

Client and Server involved in exchanging data can close the connection is called Connection Termination. It is usually initiated by the client.

Three way Handshaking is also used to terminate the connection by following steps:



1. The client TCP after receiving a close command from the client process sends the first segment a **FIN** segment in which the FIN flag is set.

   A FIN segment can include the last chunk of data sent by the client or it can be just a control segment. If it is only a control segment, it consumes only one sequence number.
2. The server TCP after receiving the FIN segment informs server process of the situation and

sends the second segment a **FIN + ACK** segment to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection in the other direction.

This segment can also contain the last chunk of data from the server. If it does not carry data, it consumes only one sequence number.

3. The TCP client sends the last segment an ACK segment to confirm the receipt of the FIN segment from the TCP server. This segment contains the acknowledgment number, which is **(sequence number + 1)** received in the FIN segment from the server. This segment cannot carry data and consumes no sequence numbers.

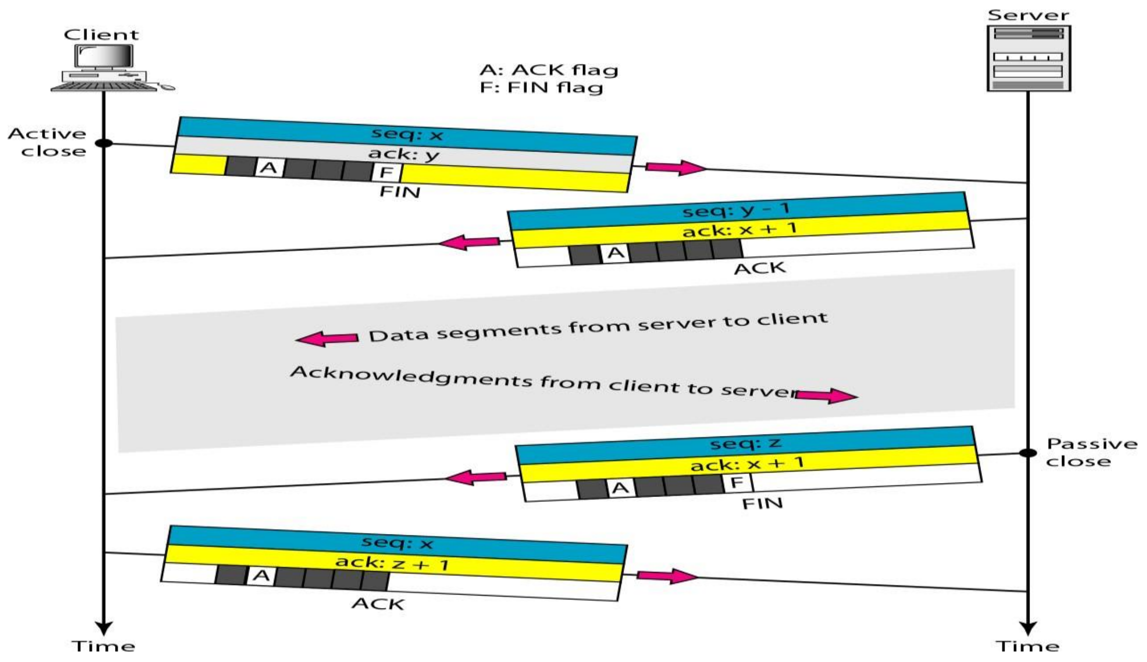
Note: In some rare situations Connection Termination can be done in Four way handshaking by using Half-close option.

Four way Handshaking using Half-Close

- In TCP one end can stop sending data while still receiving data is called a **Half-Close**. It is normally initiated by the client.
- It can occur when the server needs all the data before processing can begin.

Example: Sorting.

- When the client sends data to the server to be sorted, the server needs to receive all the data before sorting can start.
- This means the client after sending all the data can close the connection in the outbound direction.
- The inbound direction must remain open to receive the sorted data.
- The server after receiving the data still needs time for sorting. Its outbound direction must remain open.



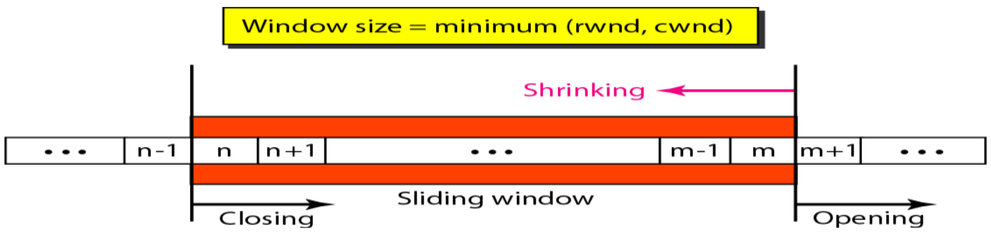
- The client half-closes the connection by sending a FIN segment. The server accepts the half-close by sending the ACK segment.
- The data transfer from the client to the server stops. The server can still send data.
- When the server has sent all the processed data, it sends a FIN segment which is acknowledged by an ACK from the client.
- After half-closing of the connection, the data can travel from the server to the client and acknowledgments can travel from the client to the server but the client cannot send any more data segments to the server.
- The second segment (ACK) consumes no sequence number. Although client has received sequence number y - 1 and is expecting y, the server sequence number is still y - 1.
- When the connection finally closes, the sequence number of the last ACK segment is still x, because no sequence numbers are consumed during data transfer in that direction.

TCP Flow control Mechanism

- TCP uses a sliding window to control the flow of data so that the destination does not become overwhelmed with data.
- The sliding window protocol used by TCP is between the **Go-Back-N** and **Selective Repeat** sliding window.
- The sliding window protocol in TCP looks like the Go-Back-N protocol because it does not use NAKs.
- It looks like Selective Repeat because the receiver holds the out-of-order segments until the missing ones arrive.

Difference between TCP sliding window and Data link layer Sliding window

1. The sliding window of TCP is byte-oriented whereas sliding window of data link layer is frame-oriented.
2. TCP's sliding window is of variable size whereas data link layer sliding window is of fixed size.



- The TCP sliding window spans a portion of the buffer containing bytes received from the process.
- Bytes inside the window are the bytes that can be sent without worrying about acknowledgment.
- The imaginary window has two walls: Left wall and Right wall.

The sliding window can be **Opened**, **Closed**, or **Shrunk.** The opening, closing, shrinking of window is in control of receiver only. The sender must obey the commands of the receiver.

- Opening a window allows more new bytes in the buffer that are eligible for sending.
- Closing the window means that some bytes have been acknowledged and the sender need not worry about them anymore.
- Shrinking the window means revoking the eligibility of some bytes for sending. This is a problem if the sender has already sent these bytes. Shrinking window means moving the right wall to the left.

Size of the sliding window

- The size of the window at sender is determined by the lesser of two values: receiver window (rwnd) or congestion window (cwnd).
- The receiver window is the value advertised by the opposite end in a segment containing acknowledgment. It is the number of bytes the other end can accept before its buffer overflows and data are discarded.
- The congestion window is a value determined by the network to avoid congestion.

<u>TCP Error Control Mechanism</u>

- TCP provides reliability using Error control that includes mechanisms for detecting corrupted segments, lost segments, out-of-order segments, and duplicated segments.
- Error control also includes a mechanism for correcting errors after they are detected.

Error detection and correction in TCP is achieved through the use of three simple tools:

1. Checksum
2. Acknowledgment
3. Time-out.

Checksum

Each segment includes a 16-bit checksum field used to check for a corrupted segment. If the segment is corrupted, it is discarded by the destination TCP and is considered as lost.

Acknowledgment

TCP uses acknowledgments to confirm the receipt of data segments. Control segments that carry no data but consume a sequence number are also acknowledged. ACK segments are never acknowledged.

Retransmission

A segment is retransmitted when it is corrupted, lost, or delayed. A segment is retransmitted on two occasions:

1. When a retransmission timer expires
2. When the sender receives three duplicate ACKs

Retransmission After Retransmission TimeOut

- TCP maintains one retransmission time-out (RTO) timer for all outstanding segments. Outstanding segments are segments which are sent but not acknowledged.
- When the timer expires, the earliest outstanding segment is retransmitted.
- No time-out timer is set for a segment that carries only an acknowledgment (i.e.) no such segment is resent.
- The value of RTO is dynamic in TCP and is updated based on the round-trip time (RTT) of segments.
- An RTT is the time needed for a segment to reach a destination and for an acknowledgment to be received. It uses a back-off strategy.

Retransmission After Three Duplicate ACK Segments

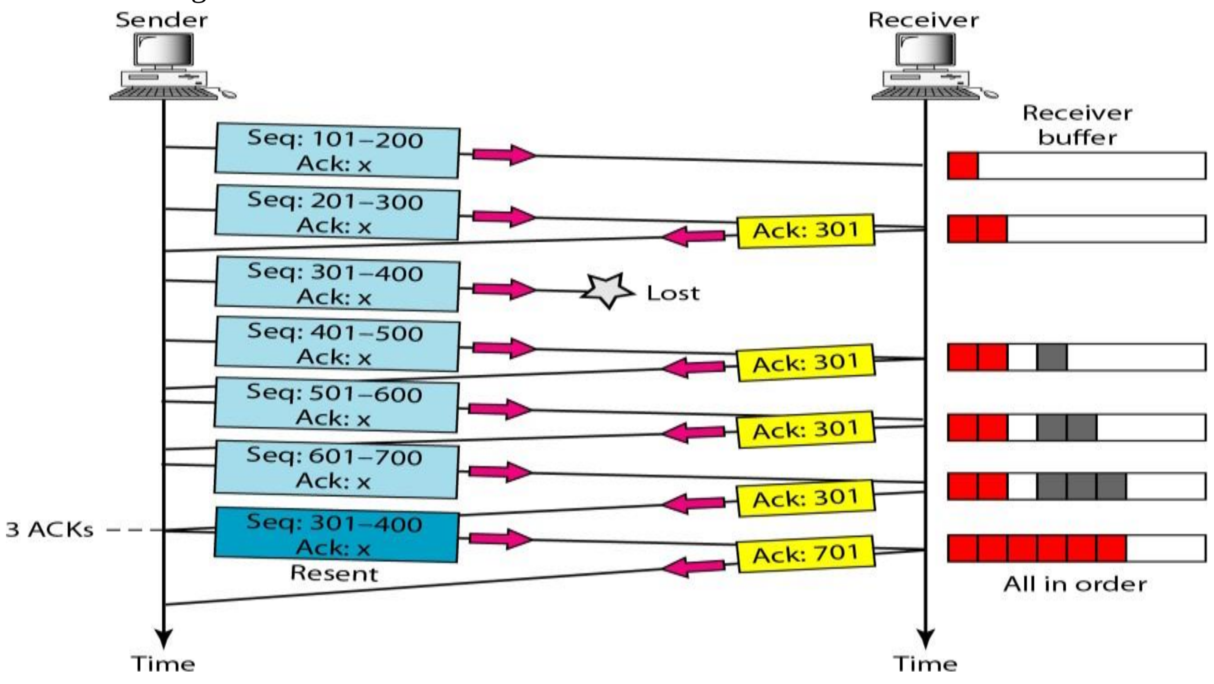
If one segment is lost and the receiver receives many out-of-order segments that they cannot be saved because of the limited buffer size

To avoid this problem we use three-duplicate-ACKs rule and retransmit the missing segment immediately. This feature is called as Fast Retransmission.

Fast Retransmission

Consider the below figure that implements the fast retransmission: the first two segments are received by the receiver buffer but the third segment is lost.

- When the receiver receives the fourth, fifth, and sixth segments, it triggers an acknowledgment.



- The sender receives four acknowledgments with the same value (three duplicates).
- The timer for **segment 3** has not expired yet, the fast transmission requires segment3 so that the segment3 will be resent immediately because all these acknowledgements expecting it.
- Only the third segment is retransmitted even though four segments are not acknowledged.
- When the sender receives the retransmitted ACK (i.e. ACK 701), it knows that the four segments are safe and sound because acknowledgment is cumulative.

CONGESTION

Congestion in a network may occur if the **load** on the networkis greater than the *capacity* of the network.
Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.

Note:
1. **Load** refers to the number of packets sent to the network.
2. **Capacity** refers to the number of packets a network can handle.

Congestion control involves two factors that measure the performance of a network: Delay and Throughput.

Delay versus Load

- The load is much less than the capacity of the network the delay is at a minimum. This minimum delay is composed of propagation delay and processing delay.
- When the load reaches the network capacity, the delay increases sharply due to additional waiting time in the queues will be added to the total delay.
- The delay becomes infinite when the load is greater than the capacity.
- When a packet is delayed, the source that is not receiving the acknowledgment will retransmit the packet. This makes the delay and the congestion even worse.

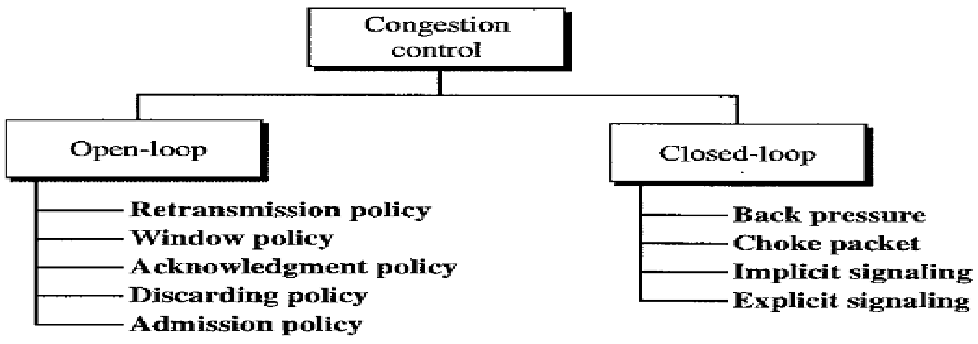
Throughput versus Load

- Throughput in a network is defined as the number of packets passing through the network in a unit of time.
- The load is below the capacity of the network, the throughput increases proportionally with the load.
- When the load exceeds the capacity, the queues become full and the routers have to discard some packets.
- Discarding packets does not reduce the number of packets in the network because the sources retransmit the packets using time-out mechanisms.

CONGESTION CONTROL

Congestion control refers to techniques and mechanisms that can either prevent congestion before it happens or remove congestion after it has happened.

Congestion control mechanisms can be divided into two categories:
1. Open-loop congestion control (prevention)
2. Closed-loop congestion control (removal)



Open-Loop Congestion Control

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

Retransmission Policy

- The packet needs to be retransmitted by sender, when a packet is lost or corrupted.
- Retransmission is sometimes unavoidable. It may increase congestion in the network.
- The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.

Example: Retransmission policy used by TCP is designed to prevent or alleviate congestion.

Window Policy

- The Selective Repeat window is better than the Go-Back-N window for congestioncontrol.
- In the Go-Back-N window, when the timer for a packet is expired several packets will be resent, although some may have arrived safe and sound at the receiver. This duplication may make the congestion worse.
- The Selective Repeat window tries to send the specific packets that have been lost or corrupted.

Acknowledgment Policy

- The acknowledgments are also part of the load in a network. Sending fewer acknowledgments means imposing less load on the network.
- If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion.
- A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires.

Discarding Policy

- A good discarding policy by routers may prevent congestion.
- Example: In audio transmission if the policy is to discard less sensitive packets when congestion happens, the quality of sound is still preserved and congestion is prevented.

Admission Policy

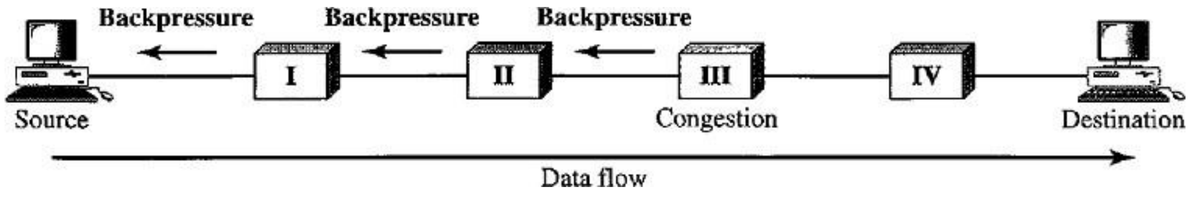
- An admission policy can prevent congestion in virtual-circuit networks.
- Switches first check the resource requirement of a data flow before admitting it to the network.
- A router can deny establishing a virtual-circuit connection if there is congestion in the network or if there is a possibility 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 are: Back pressure, Choke packet, Implicit signaling, Explicit signaling.

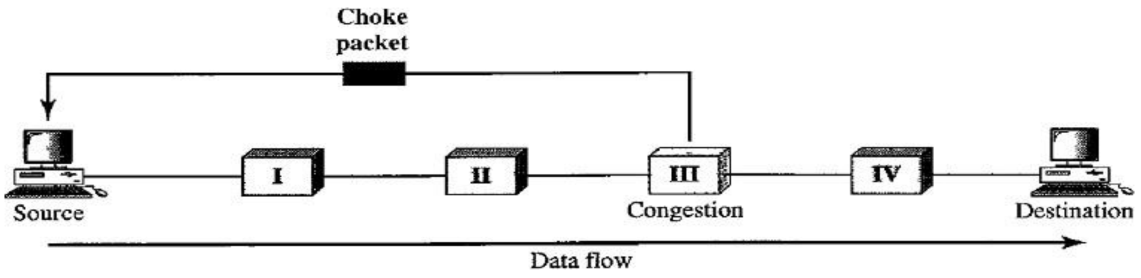
Backpressure

- In Backpressure mechanism, a congested node stops receiving data from the immediate upstream node.
- This may cause the upstream nodes to become congested and they reject data from their upstream nodes.
- 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 data flow is coming.



Choke Packet

- A choke packet is a packet sent by a node to the source to inform that congestion has occurred.
- In the choke packet method, the warning is sent from the router, which has encountered congestion to the source station directly. The intermediate nodes through which thepacket has traveled are not warned.



Implicit Signaling

- In implicit signaling, there is no communication between the congested nodes and the source.
- Source guesses that there is congestion somewhere in the network from other symptoms.

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 and the source should slow down sending speed.

Explicit Signaling

- The node that experiences congestion can explicitly send a signal to the source or destination.
- In explicit signaling method, the signal is included in the packets that carry data.
- Explicit signaling can occur in either the forward or the backward direction.

- **Backward Signaling** 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 Signaling** 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 get rid of the congestion.

CONGESTION CONTROL in TCP

TCP uses congestion control to avoid congestion or alleviate congestion in the network.

Congestion Window

- In TCP the sender window size is determined by the available buffer space in the receiver (rwnd). That means receiver dictates the sender about the size of the sender's window.
- If the network cannot deliver the data as fast as they are created by the sender then the network must tell the sender to slow down. So the network can also determine the size of the sender's window.
- Hence the Sender window size is determined by the receiver and by congestion in the network.
- The actual size of the window is the minimum of (receiver window, congestion window).

Actual Window Size = minimum (rwnd,cwnd)

Congestion Policy

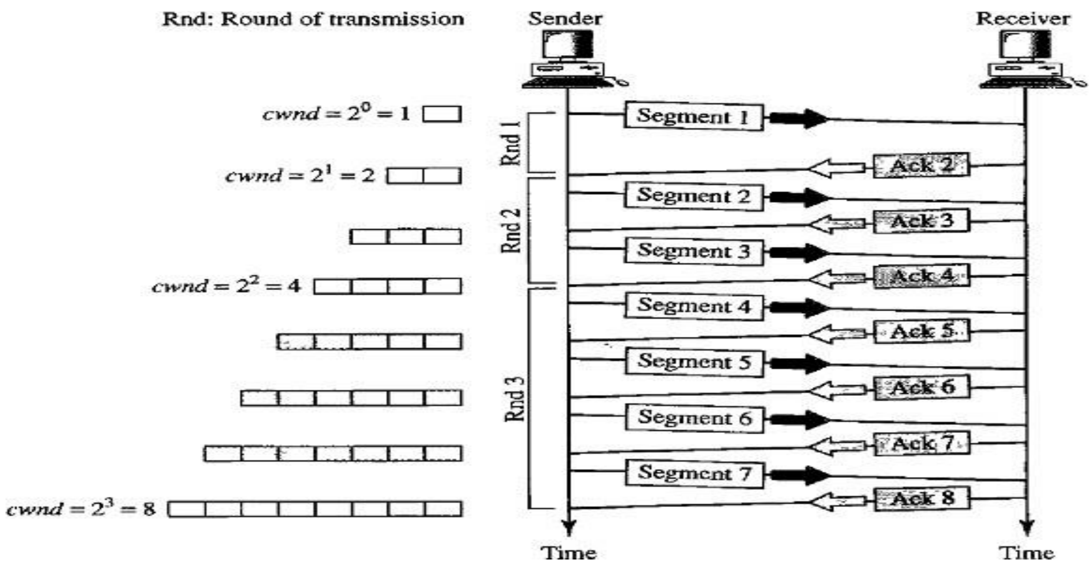
There are three phases in TCP's Congestion policy: Slow start, Congestion Avoidance and Congestion Detection.

Slow Start: Exponential Increase

- In this algorithm is based on the size of the congestion window (cwnd) starts with one maximum segment size (i.e. cwnd=1 MSS).
- The MSS is determined during connection establishment by using an option of the same name.
- The size of the window increases one MSS each time an acknowledgment is received.
- As the name implies, the window starts slowly but grows exponentially.

Example: Consider the below figure that shows the communication between client and server by using Slow start mechanism.

- If Receiver window (rwnd) is much higher than Congestion window (cwnd) then the sender window size always equals Congestion window size (cwnd). Each segment is acknowledged individually.
- The sender starts with **cwnd=1**MSS (i.e.) the sender can send only one segment.
- After receipt of the acknowledgment for segment 1, the size of the congestion window is increased by 1 (i.e.) **cwnd = 2.**
- Now two more segments can be sent. When each acknowledgment is received, the size of the window is increased by 1 MSS.
- When all seven segments are acknowledged **cwnd = 8.**



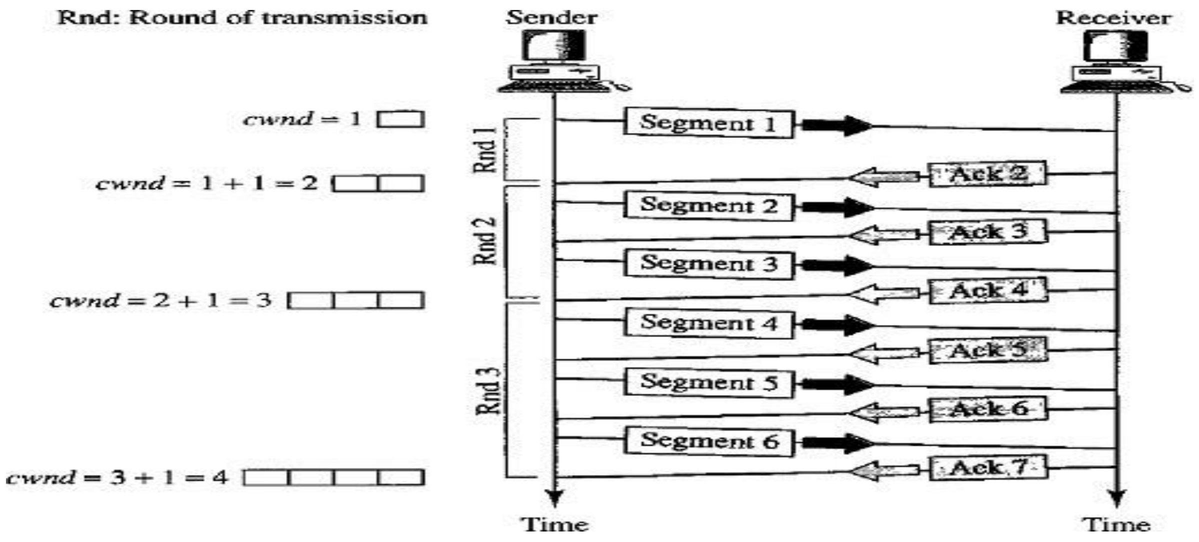
Start	→	**cwnd=1**
After round 1	→	**cwnd=2^1=2**
After round 2	→	**cwnd=2^2=4**
After round 3	→	**cwnd=2^3=8**

- If there is delayed ACKs, the increase in the size of the window is less than power of 2.
- Slow start cannot continue indefinitely. There must be a threshold to stop this phase.
- The sender keeps track of a variable called **ssthresh** (slow-start threshold). In general the value of **ssthresh** is 65,535 bytes.
- When the size of window in bytes reaches threshold value then the slow start stops and the next phase starts.

Congestion Avoidance: Additive Increase

- When the size of the congestion window reaches the slow-start threshold, the slow-start phase stops and the additive phase begins.
- In this algorithm, each time the whole window of segments is acknowledged (one round), the size of the congestion window is increased by 1.

Consider the below figure that implements Additive Increase algorithm:



In this case, after the sender has received acknowledgments for a complete window size of segments, the size of the window is increased by one segment.

Start	→	cwnd= l
After round 1	→	cwnd= 1+ 1 =2
After round 2	→	cwnd= 2+ 1 =3
After round 3	→	cwnd= 3+ 1 =4

Congestion Detection: Multiplicative Decrease

If congestion occurs, the congestion window size must be decreased.

- When there is a need to retransmit segment, then the sender guess that at some place the congestion has occurred.
- Retransmission can occur in one of two cases: when a timer times out or when three ACKs are received.
- In both cases, the size of the threshold is dropped to one-half (i.e.) a multiplicative decrease.

TCP implementations have two reactions:

1. If a time-out occurs, there is a stronger possibility of congestion. A segment has probably been dropped in the network and there is no news about the sent segments.

   In this case TCP reacts strongly:

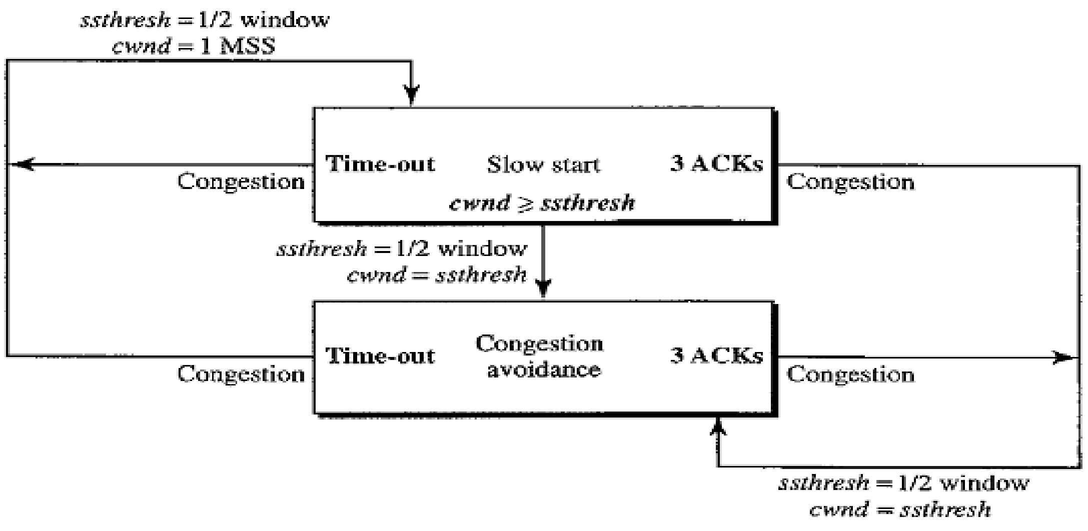
   a. It sets the value of the threshold to one-half of the current window size.
   b. It sets *cwnd* to the size of one segment (i.e.) cwnd=1.
   c. It starts the slow-start phase again.

2. If three ACKs are received, there is a weaker possibility of congestion. A segment may have been dropped but some segments after that may have arrived safely since three ACKs are received. This is called fast transmission and fast recovery.

   In this case, TCP has a weaker reaction:

   a. It sets the value of the threshold to one-half of the current window size.
   b. It sets cwnd to the value of the threshold (i.e.) cwnd= ssthresh.
   c. It starts the congestion avoidance phase.

<u>Summary of Congestion Policies:</u>



ssthresh = 1/2 window
cwnd = 1 MSS
Congestion
Time-out
Slow start
3 ACKs
Congestion
cwnd ≥ ssthresh
ssthresh = 1/2 window
cwnd = ssthresh
Congestion
Time-out
Congestion avoidance
3 ACKs
Congestion
ssthresh = 1/2 window
cwnd = ssthresh