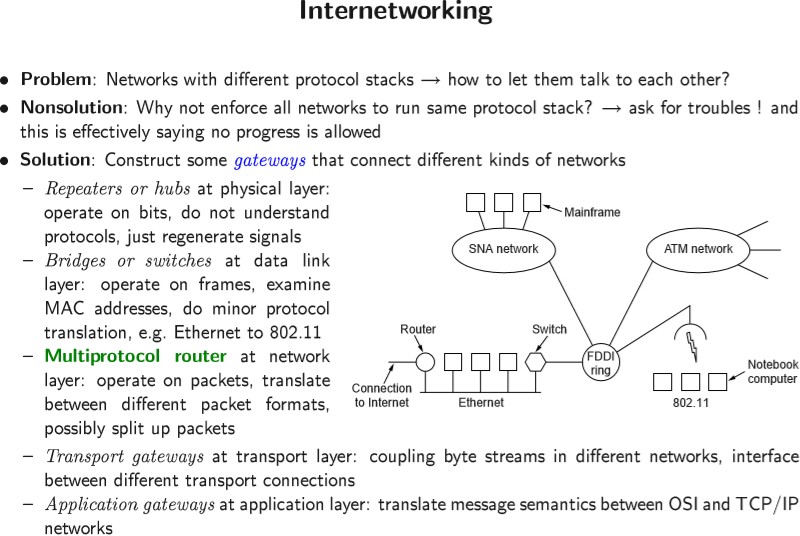
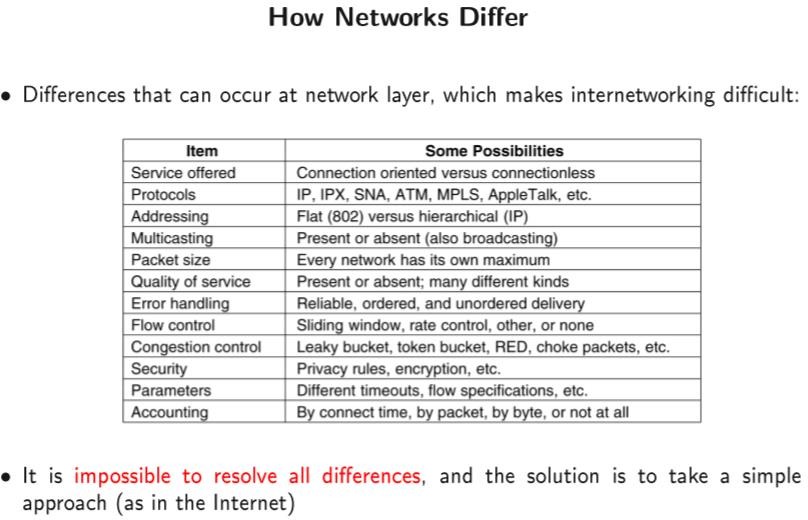
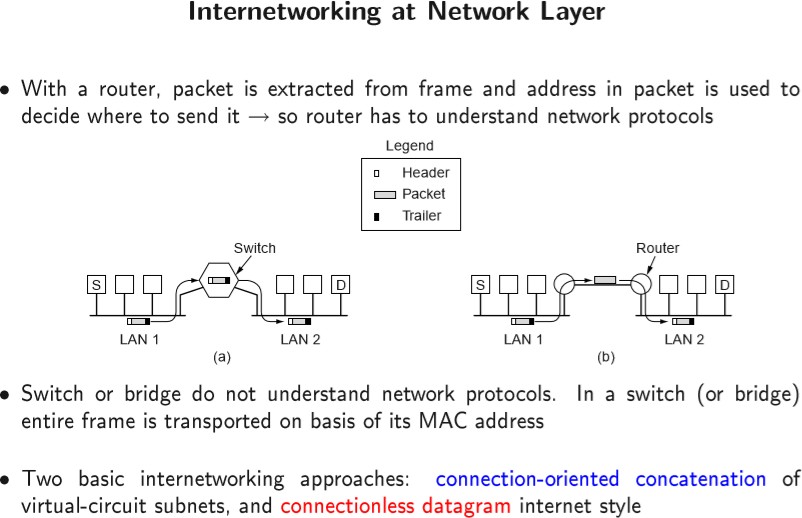
**UNIT II**

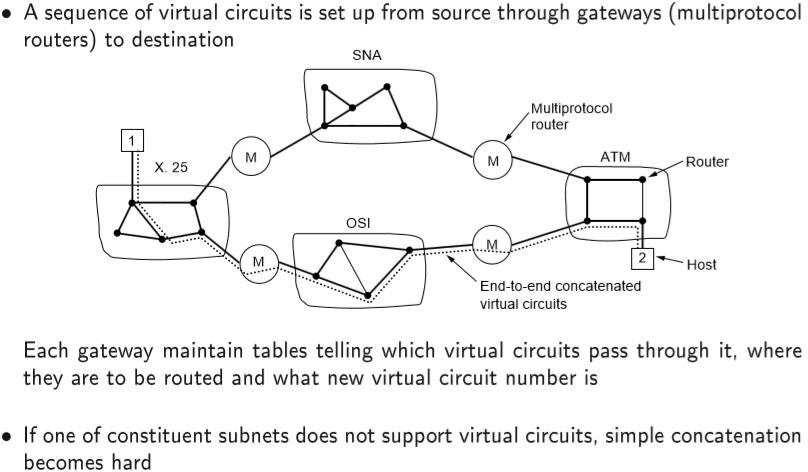
# Internetworking

Internetworking is "the concept of interconnecting different types of networks to build a large, global network" such that any pair of connected hosts can exchange packets

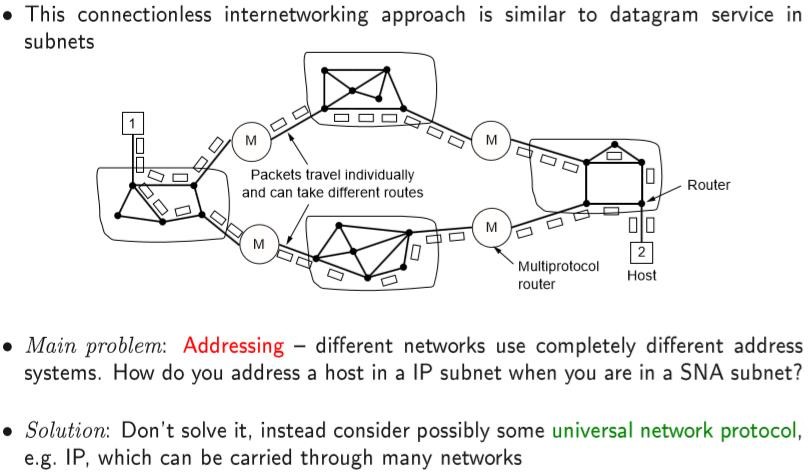




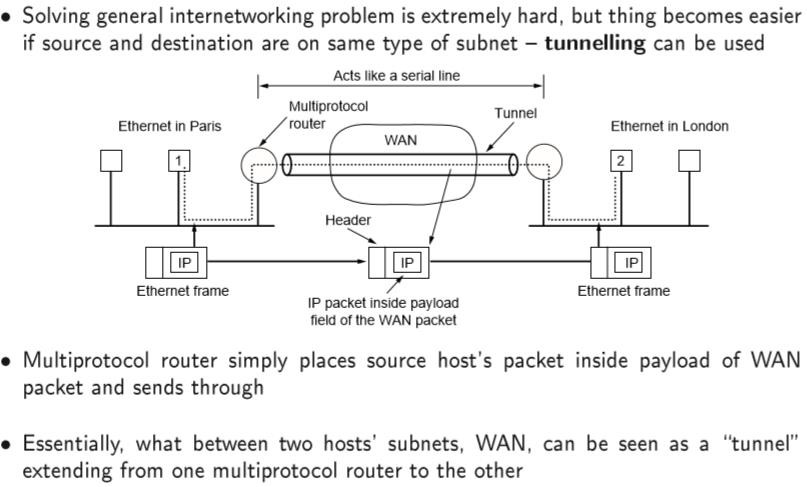
* 1. **Concatenated Virtual Circuits**



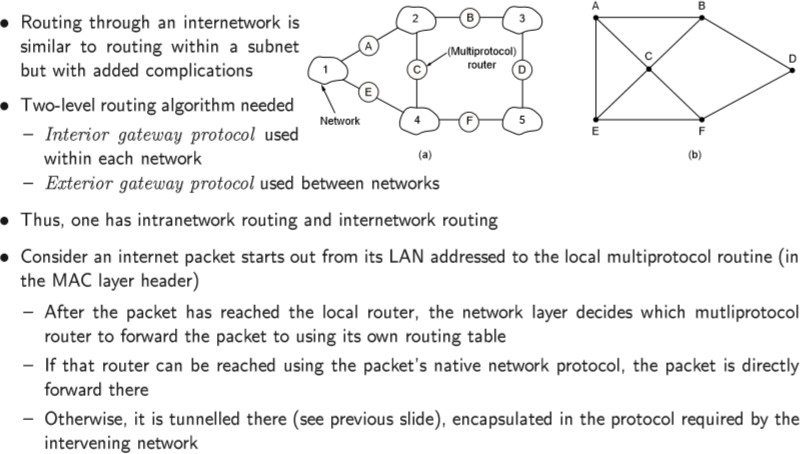
* 1. **Connectionless Internetworking**



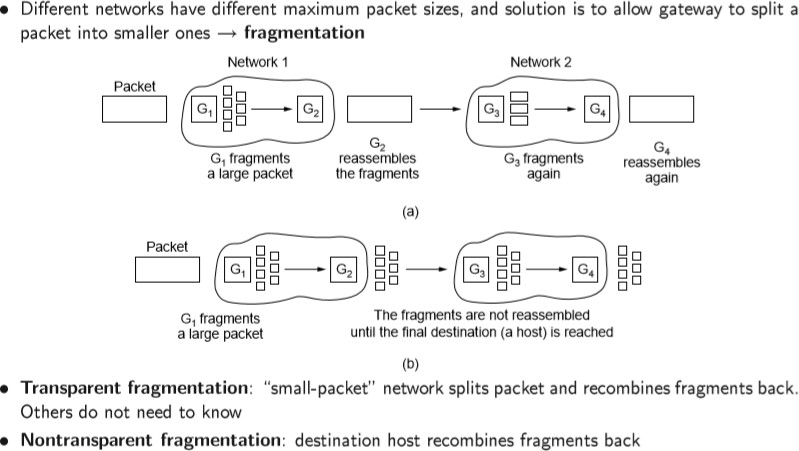
* 1. **Tunneling**

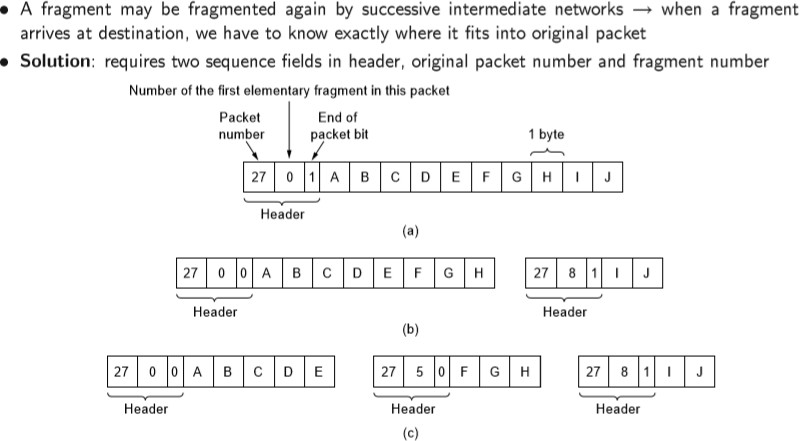


* 1. **Internetwork Routing**



* 1. **Fragmentation**





* 1. **Network layer in the Internet**

Network Layer Design Principles:

* + 1. **Make sure it works.** Do not finalize the design or standard until multiple prototypes have successfully communicated with each other.
    2. **Keep it simple.** When in doubt, use the simplest solution.
    3. **Make clear choices.** If there are several ways of doing the same thing, choose one.
    4. **Exploit modularity.** This principle leads directly to the idea of having protocol stacks, each of whose layers is independent of all the other ones.
    5. **Expect heterogeneity.** Different types of hardware, transmission facilities, and applications will occur on any large network.
    6. **Avoid static options and parameters.** If parameters are unavoidable
    7. **Look for a good design; it need not be perfect.** The designers should go with the good design and put the burden of working around it on the people with the strange requirements.
    8. **Be strict when sending and tolerant when receiving.** only send packets that rigorously comply with the standards.
    9. **Think about scalability.** Load must be spread as evenly as possible over available resources.
    10. **Consider performance and cost.** If a network has poor performance or outrageous costs, nobody will use it.

At the network layer, the Internet can be viewed as a collection of subnetworks or **Autonomous Systems** that are interconnected. There is no real structure, but several major backbones exist. These are constructed from high-bandwidth lines and fast routers. Attached to the backbones are regional (midlevel) networks and attached to these regional networks are the LANs at many universities, companies, and Internet service providers. The glue that holds the whole Internet together is the network layer protocol, **IP** (**Internet Protocol**).

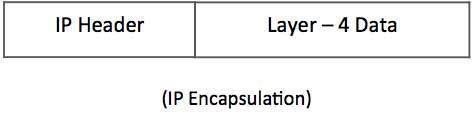
Communication in the Internet works as follows.

* The transport layer takes data streams and breaks them up into datagrams.
* Each datagram is transmitted through the Internet, possibly being fragmented into smaller units as it goes.
* When all the pieces finally get to the destination machine, they are reassembled by the network layer into the original datagram.
* This datagram is then handed to the transport layer, which inserts it into the receiving process' input stream.

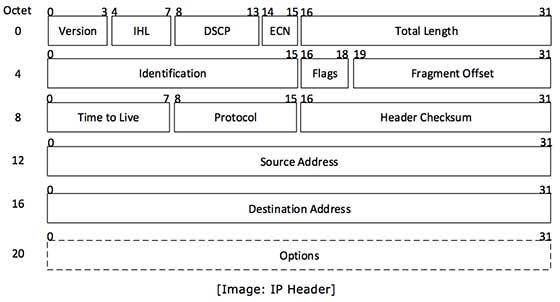
# IP Protocol

Internet Protocol: It's the set of rules that govern how packets are transmitted over a network.

Internet Protocol being a layer-3 protocol (OSI) takes data Segments from layer-4 (Transport) and divides it into packets. IP packet encapsulates data unit received from above layer and add to its own header information.



The encapsulated data is referred to as IP Payload. IP header contains all the necessary information to deliver the packet at the other end.



IP header includes many relevant information including Version Number, which, in this context, is 4. Other details are as follows:

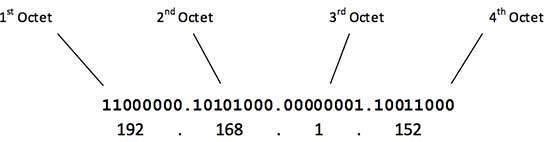
* **Version:** Version no. of Internet Protocol used (e.g. IPv4).
* **IHL:** Internet Header Length; Length of entire IP header.
* **DSCP:** Differentiated Services Code Point; this is Type of Service.
* **ECN:** Explicit Congestion Notification; It carries information about the congestion seen in the route.
* **Total Length:** Length of entire IP Packet (including IP header and IP Payload).
* **Identification:** If IP packet is fragmented during the transmission, all the fragments contain same identification number. to identify original IP packet they belong to.
* **Flags:** As required by the network resources, if IP Packet is too large to handle, these ‘flags’ tells if they can be fragmented or not. In this 3-bit flag, the MSB is always set to ‘0’.
* **Fragment Offset:** This offset tells the exact position of the fragment in the original IP Packet.
* **Time to Live:** To avoid looping in the network, every packet is sent with some TTL value set, which tells the network how many routers (hops) this packet can cross. At each hop, its value is decremented by one and when the value reaches zero, the packet is discarded.
* **Protocol:** Tells the Network layer at the destination host, to which Protocol this packet belongs to, i.e. the next level Protocol. For example protocol number of ICMP is 1, TCP is 6 and UDP is 17.
* **Header Checksum:** This field is used to keep checksum value of entire header which is then used to check if the packet is received error-free.
* **Source Address:** 32-bit address of the Sender (or source) of the packet.
* **Destination Address:** 32-bit address of the Receiver (or destination) of the packet.
* **Options:** This is optional field, which is used if the value of IHL is greater than 5. These options may contain values for options such as Security, Record Route, Time Stamp, etc.

# IP Addresses

Internet Protocol hierarchy contains several classes of IP Addresses to be used efficiently in various situations as per the requirement of hosts per network. Broadly, the IPv4 Addressing system is divided into five classes of IP Addresses. All the five classes are identified by the first octet of IP Address.

Internet Corporation for Assigned Names and Numbers is responsible for assigning IP addresses.

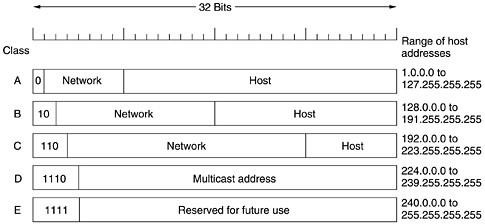
The first octet referred here is the left most of all. The octets numbered as follows depicting dotted decimal notation of IP Address:



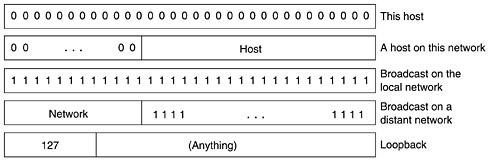
The number of networks and the number of hosts per class can be derived by this formula:



When calculating hosts' IP addresses, 2 IP addresses are decreased because they cannot be assigned to hosts, i.e. the first IP of a network is network number and the last IP is reserved for Broadcast IP.



IP addresses were divided into the five categories. This allocation is called classful addressing. The class A, B, C, and D formats allow for up to 128 networks with 16 million hosts each, 16,384 networks with up to 64K hosts, and 2 million networks (e.g., LANs) with up to 256 hosts each (although a few of these are special). Also supported is multicast, in which a datagram is directed to multiple hosts. Addresses beginning with 1111 are reserved for future use.



### Figure: Special IP addresses.

1. **Subnets**

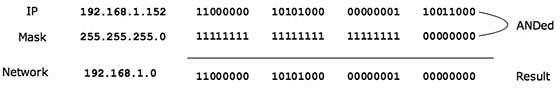
All the hosts in a network must have the same network number. This property of IP addressing can cause problems as networks grow. The solution is to allow a network to be split into several parts for internal use but still act like a single network to the outside world. The parts of the network are called **subnets**.

When a packet comes into the main router, how does it know which subnet (Ethernet) to give it to? To implement subnetting, the main router needs a **subnet mask** that indicates the split between network

+ subnet number and host.

**Subnet Mask:**

The 32-bit IP address contains information about the host and its network. It is very necessary to distinguish both. For this, routers use Subnet Mask, which is as long as the size of the network address in the IP address. Subnet Mask is also 32 bits long. If the IP address in binary is ANDed with its Subnet Mask, the result yields the Network address. For example, say the IP Address is 192.168.1.152 and the Subnet Mask is 255.255.255.0 then:



This way the Subnet Mask helps extract the Network ID and the Host from an IP Address. It can be identified now that 192.168.1.0 is the Network number and 192.168.1.152 is the host on that network.

## CIDR—Classless InterDomain Routing

CIDR stands for Classless Inter-Domain Routing and is used for IP addressing and routing. It allocates IP addresses in a more flexible manner as compared to the original system of Internet Protocol (IP) address classes. In this way, it increases the number of available IP addresses with extensive use of NAT (Network Address Translation).

CIDR IP addresses can be described as consisting of two groups of bits. The most significant group of bits denotes the prefix i.e., a network address that is used for the identification of a network or sub- network. The least significant group of bits is known as host identifier that determines the total number of bits in the address. It is used to signify the device on the work that will receive incoming information packets.

For example, consider the following CIDR Notation 182.0.1.2/28

Here, the prefix is – 182.0.1.2, and

The total number of bits in this address is 28.

**CIDR Block**

The prefix, first group of bits in the notation allows you to group the multiple blocks of network addresses into a single routing network. CIDR blocks share the first group of bits (the binary representation of the network addresses). The blocks are also identified using same decimal dot notation system as IPv4 addresses.

For example, a CIDR block is shown below 10.0.1.0/24

Here /24 signifies the total number of 1’s bits in the routing mask (network mask). This IP address can be shown as below in the binary format: 11111111.11111111.11111111.00000000

Here the first 24 bits are marked as 1.

It would be equivalent to a network mask of 255.255.255.0

Note that the network addresses that have the identical prefix and the same number of bits, always belong the same block. Also, the large and small blocks can be distinguished by the length of the prefix.

## NAT—Network Address Translation

The basic idea behind NAT is to assign each company a single IP address (or at most, a small number of them) for Internet traffic. *Within* the company, every computer gets a unique IP address, which is used for routing intramural traffic. However, when a packet exits the company and goes to the ISP, an address translation takes place. To make this scheme possible, three ranges of IP addresses have been declared as private. Companies may use them internally as they wish. The only rule is that no packets containing these addresses may appear on the Internet itself. The three reserved ranges are:

10.0.0.0 – 10.255.255.255/8 (16,777,216 hosts)

172.16.0.0 – 172.31.255.255/12 (1,048,576 hosts)

192.168.0.0 – 192.168.255.255/16 (65,536 hosts)

Objection over NAT are given below:

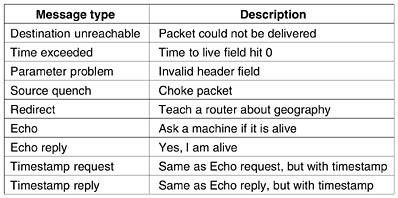
* + First, NAT violates the architectural model of IP
  + Second, NAT changes the Internet from a connectionless network to a kind of connection- oriented network.
  + Third, NAT violates the most fundamental rule of protocol layering: layer *k* may not make any assumptions about what layer *k* + 1 has put into the payload field.
  + Fourth, processes on the Internet are not required to use TCP or UDP.
  + Fifth, some applications insert IP addresses in the body of the text. The receiver then extracts these addresses and uses them. Since NAT knows nothing about these addresses, it cannot replace them, so any attempt to use them on the remote side will fail.

# Internet Control Protocols

the Internet has several control protocols used in the network layer, including ICMP, ARP, RARP, BOOTP, and DHCP

## The Internet Control Message Protocol

The operation of the Internet is monitored closely by the routers. When something unexpected occurs, the event is reported by the **ICMP** (**Internet Control Message Protocol**), which is also used to test the Internet. About a dozen types of ICMP messages are defined. The most important ones are listed below. Each ICMP message type is encapsulated in an IP packet.



*Figure: The principal ICMP message types.*

## ARP—The Address Resolution Protocol

Address Resolution Protocol is used to acquire the MAC address of a host whose IP address is known. ARP is a Broadcast packet which is received by all the host in the network segment. But only the host whose IP is mentioned in ARP responds to it providing its MAC address.

## RARP, BOOTP, and DHCP

ARP solves the problem of finding out which Ethernet address corresponds to a given IP address. Sometimes the reverse problem has to be solved: Given an Ethernet address, what is the corresponding IP address?

The first solution devised was to use **RARP** (**Reverse Address Resolution Protocol**)

This protocol allows a newly-booted workstation to broadcast its Ethernet address. Does anyone out there know my IP address? The RARP server sees this request, looks up the Ethernet address in its configuration files, and sends back the corresponding IP address.

A disadvantage of RARP is that it uses a destination address of all 1s (limited broadcasting) to reach the RARP server. However, such broadcasts are not forwarded by routers, so a RARP server is needed on each network. To get around this problem, an alternative bootstrap protocol called **BOOTP** was invented. Unlike RARP, BOOTP uses UDP messages, which are forwarded over routers. It also provides a diskless workstation with additional information, including the IP address of the file server holding the memory image, the IP address of the default router, and the subnet mask to use.

A serious problem with BOOTP is that it requires manual configuration of tables mapping IP address to Ethernet address. When a new host is added to a LAN, it cannot use BOOTP until an administrator has assigned it an IP address and entered its (Ethernet address, IP address) into the BOOTP configuration tables by hand. To eliminate this error-prone step, BOOTP was extended and given a new name: **DHCP** (**Dynamic Host Configuration Protocol**). DHCP allows both manual IP address assignment and automatic assignment. Like RARP and BOOTP, DHCP is based on the idea of a special server that assigns IP addresses to hosts asking for one. This server need not be on the same LAN as the requesting host. Since the DHCP server may not be reachable by broadcasting, a **DHCP relay agent** is needed on each LAN.

To find its IP address, a newly-booted machine broadcasts a DHCP DISCOVER packet. The DHCP relay agent on its LAN intercepts all DHCP broadcasts. When it finds a DHCP DISCOVER packet, it sends the packet as a unicast packet to the DHCP server, possibly on a distant network. The only piece of information the relay agent needs is the IP address of the DHCP server.

# OSPF

Routers connect networks using the Internet Protocol (IP), and OSPF (Open Shortest Path First) is a router protocol used to find the best path for packets as they pass through a set of connected networks. OSPF is designated by the Internet Engineering Task Force (IETF) as one of several Interior Gateway Protocols (IGPs) -- that is, protocols aimed at traffic moving around within a larger autonomous system network like a single enterprise's network, which may in turn be made up of many separate local area networks linked through routers.

OSPF supports three kinds of connections and networks:

* + 1. Point-to-point lines between exactly two routers.
    2. Multiaccess networks with broadcasting (e.g., most LANs).
    3. Multiaccess networks without broadcasting (e.g., most packet-switched WANs).

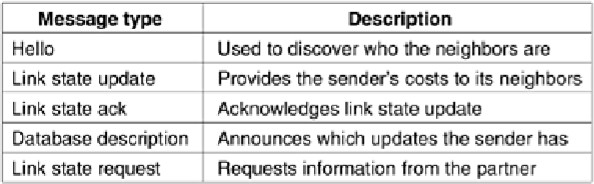
A multiaccess network is one that can have multiple routers on it, each of which can directly communicate with all the others. OSPF operates by abstracting the collection of actual networks, routers, and lines into a directed graph in which each arc is assigned a cost (distance, delay, etc.). It then computes the shortest path based on the weights on the arcs. A serial connection between two routers is represented by a pair of arcs, one in each direction. Their weights may be different. A multiaccess network is represented by a node for the network itself plus a node for each router. The arcs from the network node to the routers have weight 0 and are omitted from the graph.

OSPF distinguishes four classes of routers:

1. Internal routers are wholly within one area.
2. Area border routers connect two or more areas.
3. Backbone routers are on the backbone.
4. AS boundary routers talk to routers in other ASes.

OSPF works by exchanging information between adjacent routers, which is not the same as between neighboring routers. In particular, it is inefficient to have every router on a LAN talk to every other router on the LAN. To avoid this situation, one router is elected as the designated router. It is said to be adjacent to all the other routers on its LAN, and exchanges information with them. Neighboring routers that are not adjacent do not exchange information with each other. A backup designated router is always kept up to date to ease the transition should the primary designated router crash and need to replaced immediately.

The five types of OSPF messages:



Using flooding, each router informs all the other routers in its area of its neighbors and costs. This information allows each router to construct the graph for its area(s) and compute the shortest path.

# BGP

BGP (Border Gateway Protocol) is protocol that manages how packets are routed across the internet through the exchange of routing and reachability information between edge routers.

Exterior gateway protocols in general, and BGP, have been designed to allow many kinds of routing policies to be enforced in the interAS traffic.

Typical policies involve political, security, or economic considerations. A few examples of routing constraints are:

* + 1. No transit traffic through certain ASes.
    2. Never put Iraq on a route starting at the Pentagon.
    3. Do not use the United States to get from British Columbia to Ontario.
    4. Only transit Albania if there is no alternative to the destination.
    5. Traffic starting or ending at IBM should not transit Microsoft.

Policies are typically manually configured into each BGP router (or included using some kind of script). They are not part of the protocol itself.

Pairs of BGP routers communicate with each other by establishing TCP connections. Operating this way provides reliable communication and hides all the details of the network being passed through.

BGP is fundamentally a distance vector protocol, but quite different from most others such as RIP. Instead of maintaining just the cost to each destination, each BGP router keeps track of the path used.

Similarly, instead of periodically giving each neighbor its estimated cost to each possible destination, each BGP router tells its neighbors the exact path it is using.

BGP sends updated router table information only when something changes -- and even then, it sends only the affected information. BGP has no automatic discovery mechanism, which means connections between peers have to be set up manually, with peer addresses programmed in at both ends.

# Internet Multicasting

Normal IP communication is between one sender and one receiver. However, for some applications it is useful for a process to be able to send to a large number of receivers simultaneously. IP supports multicasting, using class D addresses. Each class D address identifies a group of hosts. Twenty-eight bits are available for identifying groups, so over 250 million groups can exist at the same time. When a process sends a packet to a class D address, a best-efforts attempt is made to deliver it to all the members of the group addressed, but no guarantees are given. Some members may not get the packet.

Two kinds of group addresses are supported: permanent addresses and temporary ones. A permanent group is always there and does not have to be set up. Each permanent group has a permanent group address. Some examples of permanent group addresses are:

* + - 1. All systems on a LAN
      2. All routers on a LAN
      3. All OSPF routers on a LAN
      4. All designated OSPF routers on a LAN

Temporary groups must be created before they can be used. A process can ask its host to join a specific group. It can also ask its host to leave the group. When the last process on a host leaves a group, that group is no longer present on the host. Each host keeps track of which groups its processes currently belong to. Multicasting is implemented by special multicast routers, which may or may not be co allocated with the standard routers. Multicast routing is done using spanning trees.

# Mobile IP,

When people began demanding the ability to connect their notebook computers to the Internet wherever they were, IETF set up a Working Group to find a solution. The Working Group quickly formulated a number of goals considered desirable in any solution. The major ones were:

* + 1. Each mobile host must be able to use its home IP address anywhere.
    2. Software changes to the fixed hosts were not permitted.
    3. Changes to the router software and tables were not permitted.
    4. Most packets for mobile hosts should not make detours on the way.
    5. No overhead should be incurred when a mobile host is at home.

The solution chosen was that every site that wants to allow its users to roam has to create a home agent. Every site that wants to allow visitors has to create a foreign agent. When a mobile host shows up at a foreign site, it contacts the foreign host there and registers. The foreign host then contacts the user's home agent and gives it a **care-of address**, normally the foreign agent's own IP address.

At the time the mobile host moves, the router probably has its (soon-to-be-invalid) Ethernet address cached. Replacing that Ethernet address with the home agent's is done by a trick called **gratuitous ARP**. This is a special, unsolicited message to the router that causes it to replace a specific cache entry,

in this case, that of the mobile host about to leave. When the mobile host returns later, the same trick is used to update the router's cache again.

# IPv6.

IPV6 major goals were:

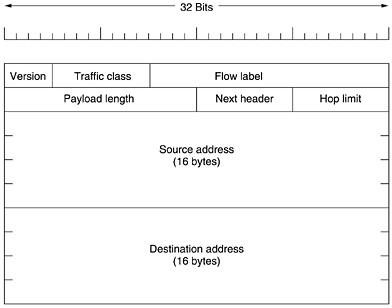
* + 1. Support billions of hosts, even with inefficient address space allocation.
    2. Reduce the size of the routing tables.
    3. Simplify the protocol, to allow routers to process packets faster.
    4. Provide better security (authentication and privacy) than current IP.
    5. Pay more attention to type of service, particularly for real-time data.
    6. Aid multicasting by allowing scopes to be specified.
    7. Make it possible for a host to roam without changing its address.
    8. Allow the protocol to evolve in the future.
    9. Permit the old and new protocols to coexist for years.

IPv6 meets the goals fairly well. It maintains the good features of IP, discards or deemphasizes the bad ones, and adds new ones where needed.

* First and foremost, IPv6 has longer addresses than IPv4.
* The second major improvement of IPv6 is the simplification of the header.
* The third major improvement was better support for options.
* A fourth area in which IPv6 represents a big advance is in security.
* Finally, more attention has been paid to quality of service.

## The Main IPv6 Header

The IPv6 header is shown below. The *Version* field is always 6 for IPv6 (and 4 for IPv4).



*Figure: The IPv6 fixed header (required).*

The *Traffic class* field is used to distinguish between packets with different real-time delivery requirements. The *Flow label* field is also still experimental but will be used to allow a source and destination to set up a pseudo connection with properties and requirements. Each flow is designated by the source address, destination address, and flow number, so many flows may be active at the same time between a given pair of IP addresses. The *Payload length* field tells how many bytes follow the

40-byte header. The *Next header* field tells which of the (currently) six extension headers, if any, follow this one. If this header is the last IP header, the *Next header* field tells which transport protocol handler (e.g., TCP, UDP) to pass the packet to. The *Hop limit* field is used to keep packets from living forever. It is, in practice, the same as the *Time to live* field in IPv4, namely, a field that is decremented on each hop. Next come the *Source address* and *Destination address* fields.

A new notation has been devised for writing 16-byte addresses. They are written as eight groups of four hexadecimal digits with colons between the groups, like this:

8000:0000:0000:0000:0123:4567:89AB:CDEF

Since many addresses will have many zeros inside them, three optimizations have been authorized. First, leading zeros within a group can be omitted, so 0123 can be written as 123. Second, one or more groups of 16 zero bits can be replaced by a pair of colons. Thus, the above address now becomes

8000::123:4567:89AB:CDEF

Finally, IPv4 addresses can be written as a pair of colons and an old dotted decimal number, for example

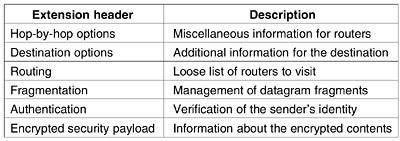
::192.31.20.46

Perhaps it is unnecessary to be so explicit about it, but there are a lot of 16-byte addresses. Specifically, there are 2128 of them, which is approximately 3 x 1038.

All the fields relating to fragmentation were removed because IPv6 takes a different approach to fragmentation. Finally, the *Checksum* field is gone because calculating it greatly reduces performance.

## Extension Headers

Some of the missing IPv4 fields are occasionally still needed, so IPv6 has introduced the concept of an (optional) **extension header**. These headers can be supplied to provide extra information, but encoded in an efficient way.

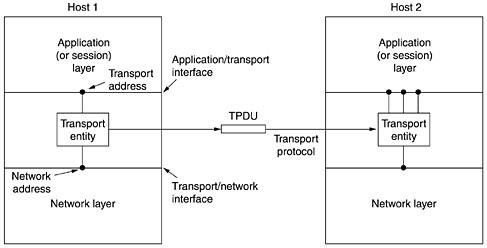


*Figure: IPv6 extension headers.*

# The Transport Service

## Services Provided to the Upper Layers

The ultimate goal of the transport layer is to provide efficient, reliable, and cost-effective service to its users, normally processes in the application layer. To achieve this goal, the transport layer makes use of the services provided by the network layer. The hardware and/or software within the transport layer that does the work is called the **transport entity**.



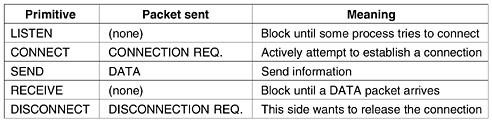
*Figure: The network, transport, and application layers.*

* + There are two types of transport service: The connection-oriented transport service and the connectionless transport service. In both cases, connections have three phases: establishment, data transfer, and release.
  + The transport code runs entirely on the users' machines.
  + The transport layer makes it possible for the transport service to be more reliable than the underlying network service. Lost packets and mangled data can be detected and compensated for by the transport layer.
  + Furthermore, the transport service primitives can be implemented as calls to library procedures in order to make them independent of the network service primitives.
  + Application programmers can write code according to a standard set of primitives and have these programs work on a wide variety of networks, without having to worry about dealing with different subnet interfaces and unreliable transmission.

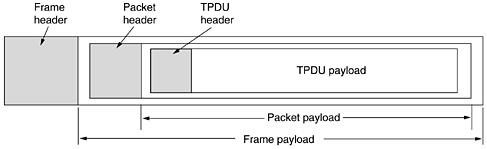
## Transport Service Primitives

To allow users to access the transport service, the transport layer must provide some operations to application programs, that is, a transport service interface. Each transport service has its own interface.

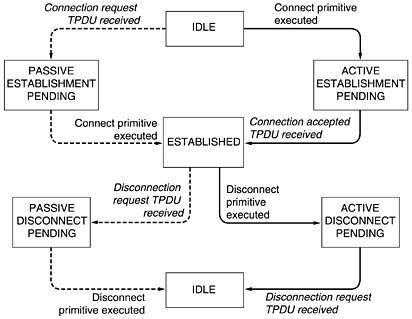
The Transport layer provides (connection-oriented) reliable and unreliable (datagram) service.



*Figure: The primitives for a simple transport service.*



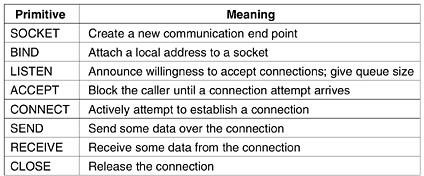
*Figure: Nesting of TPDUs, packets, and frames.*



*Figure: A state diagram for a simple connection management scheme*

## 6.1.3 Berkeley Sockets

The socket primitives used in Berkeley UNIX for TCP. These primitives are widely used for Internet programming.



### Figure: The socket primitives for TCP.

* 1. **Elements of Transport protocols**

The transport service is implemented by a **transport protocol** used between the two transport entities. The elements of transport layer are:

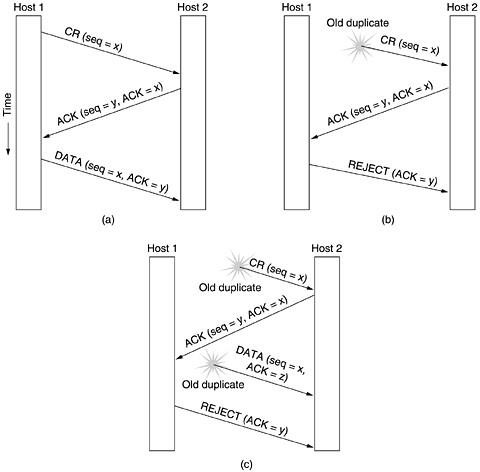
* Addressing
* Connection Establishment
* Connection Release
* Flow Control and Buffering
* Multiplexing
* Crash Recovery

## Addressing

When an application process wishes to set up a connection to a remote application process, it must specify which one to connect to. The method normally used is to define transport addresses to which processes can listen for connection requests. These end points are called **TSAP**, (**Transport Service Access Point**).

## Connection Establishment

Tomlinson (1975) introduced the **three-way handshake**. This establishment protocol does not require both sides to begin sending with the same sequence number, so it can be used with synchronization methods other than the global clock method. The normal setup procedure when host 1 initiates is shown in below figure. Host 1 chooses a sequence number, *x*, and sends a CONNECTION REQUEST TPDU containing it to host 2. Host 2 replies with an ACK TPDU acknowledging *x* and announcing its own initial sequence number, *y*. Finally, host 1 acknowledges host 2's choice of an initial sequence number in the first data TPDU that it sends.



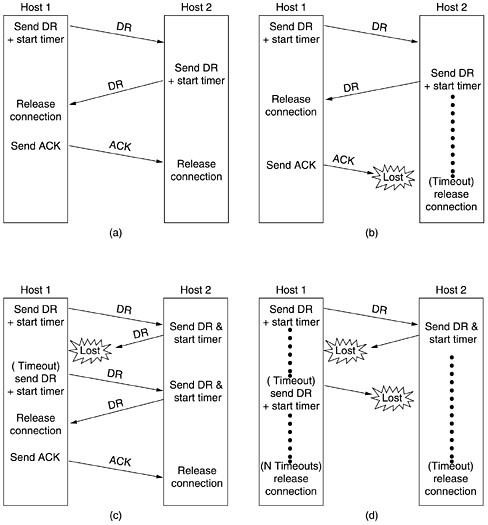
*Figure: Three protocol scenarios for establishing a connection using a three-way handshake. CR denotes CONNECTION REQUEST. (a) Normal operation. (b) Old duplicate CONNECTION REQUEST appearing out of nowhere. (c) Duplicate CONNECTION REQUEST and duplicate ACK.*

## Connection Release

Releasing a connection is easier than establishing one. Nevertheless, there are more pitfalls than one might expect. As we mentioned earlier, there are two styles of terminating a connection: asymmetric release and symmetric release. Asymmetric release is the way the telephone system works: when one party hangs up, the connection is broken. Symmetric release treats the connection as two separate unidirectional connections and requires each one to be released separately.

Asymmetric release is abrupt and may result in data loss.

### Four protocol scenarios for releasing a connection. (a) Normal case of three-way handshake. (b) Final ACK lost. (c) Response lost. (d) Response lost and subsequent DRs lost.



1. **Flow Control and Buffering**

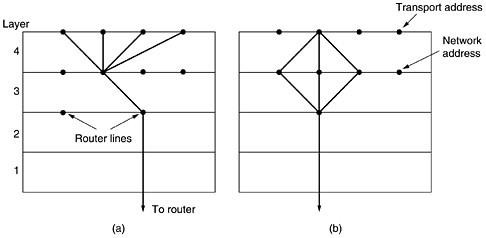
A sliding window or other scheme is needed on each connection to keep a fast transmitter from overrunning a slow receiver. If the network service is unreliable, the sender must buffer all TPDUs sent. if the sender knows that the receiver always has buffer space, it need not retain copies of the TPDUs it sends. Even if the receiver has agreed to do the buffering, there still remains the question of the buffer size. The optimum trade-off between source buffering and destination buffering depends on the type of traffic carried by the connection. To prevent deadlock, each host should periodically send control TPDUs giving the acknowledgement and buffer status on each connection. That way, the deadlock will be broken, sooner or later.

When buffer space no longer limits the maximum flow, another bottleneck will appear: the carrying capacity of the subnet. What is needed is a mechanism based on the subnet's carrying capacity rather than on the receiver's buffering capacity. Clearly, the flow control mechanism must be applied at the sender to prevent it from having too many unacknowledged

## Multiplexing

Multiplexing several conversations onto connections, virtual circuits, and physical links plays a role in several layers of the network architecture. In the transport layer the need for multiplexing can arise

in a number of ways. For example, if only one network address is available on a host, all transport connections on that machine have to use it. When a TPDU comes in, some way is needed to tell which process to give it to. This situation, called **upward multiplexing**.



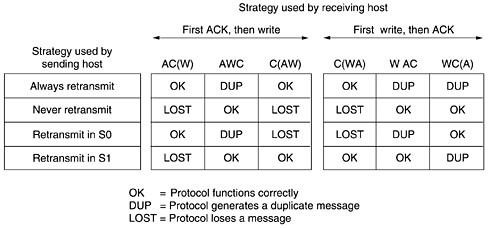
### Figure: (a) Upward multiplexing. (b) Downward multiplexing.

If a user needs more bandwidth than one virtual circuit can provide, a way out is to open multiple network connections and distribute the traffic among them on a round-robin basis, This mode of operation is called **downward multiplexing**.

## Crash Recovery

If hosts and routers are subject to crashes, recovery from these crashes becomes an issue. If the transport entity is entirely within the hosts, recovery from network and router crashes is straightforward. If the network layer provides datagram service, the transport entities expect lost TPDUs all the time and know how to cope with them. If the network layer provides connection- oriented service, then loss of a virtual circuit is handled by establishing a new one and then probing the remote transport entity to ask it which TPDUs it has received and which ones it has not received. The latter ones can be retransmitted.

A more troublesome problem is how to recover from host crashes. In an attempt to recover its previous status, the server might send a broadcast TPDU to all other hosts, announcing that it had just crashed and requesting that its clients inform it of the status of all open connections. Each client can be in one of two states: one TPDU outstanding, *S1*, or no TPDUs outstanding, *S0*. Based on only this state information, the client must decide whether to retransmit the most recent TPDU.



### Figure: Different combinations of client and server strategy.

**2.18 The Internet Transport Protocols: UDP, TCP**

There are two types of Internet Protocol (IP) traffic. They are TCP or Transmission Control Protocol

and UDP or User Datagram Protocol. TCP is connection oriented – once a connection is established, data can be sent bidirectional. UDP is a simpler, connectionless Internet protocol. Multiple messages are sent as packets in chunks using UDP.

## Transmission Control Protocol

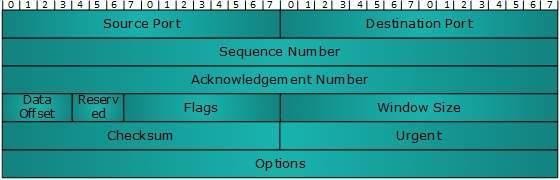
The transmission Control Protocol (TCP) is one of the most important protocols of Internet Protocols suite. It is most widely used protocol for data transmission in communication network such as internet.

## Features

* + TCP is reliable protocol. That is, the receiver always sends either positive or negative acknowledgement about the data packet to the sender, so that the sender always has bright clue about whether the data packet is reached the destination or it needs to resend it.
  + TCP ensures that the data reaches intended destination in the same order it was sent.
  + TCP is connection oriented. TCP requires that connection between two remote points be established before sending actual data.
  + TCP provides error-checking and recovery mechanism.
  + TCP provides end-to-end communication.
  + TCP provides flow control and quality of service.
  + TCP operates in Client/Server point-to-point mode.
  + TCP provides full duplex server, i.e. it can perform roles of both receiver and sender.

## Header

The length of TCP header is minimum 20 bytes long and maximum 60 bytes.



* + **Source Port (16-bits)** - It identifies source port of the application process on the sending device.
  + **Destination Port (16-bits)** - It identifies destination port of the application process on the receiving device.
  + **Sequence Number (32-bits)** - Sequence number of data bytes of a segment in a session.
  + **Acknowledgement Number (32-bits)** - When ACK flag is set, this number contains the next sequence number of the data byte expected and works as acknowledgement of the previous data received.
  + **Data Offset (4-bits)** - This field implies both, the size of TCP header (32-bit words) and the offset of data in current packet in the whole TCP segment.
  + **Reserved (3-bits)** - Reserved for future use and all are set zero by default.

## Flags (1-bit each)

* + - **NS** - Nonce Sum bit is used by Explicit Congestion Notification signaling process.
    - **CWR** - When a host receives packet with ECE bit set, it sets Congestion Windows Reduced to acknowledge that ECE received.
    - **ECE** -It has two meanings:
      * If SYN bit is clear to 0, then ECE means that the IP packet has its CE (congestion experience) bit set.
      * If SYN bit is set to 1, ECE means that the device is ECT capable.
    - **URG** - It indicates that Urgent Pointer field has significant data and should be processed.
    - **ACK** - It indicates that Acknowledgement field has significance. If ACK is cleared to 0, it indicates that packet does not contain any acknowledgement.
    - **PSH** - When set, it is a request to the receiving station to PUSH data (as soon as it comes) to the receiving application without buffering it.
    - **RST** - Reset flag has the following features:
      * It is used to refuse an incoming connection.
      * It is used to reject a segment.
      * It is used to restart a connection.
    - **SYN** - This flag is used to set up a connection between hosts.
    - **FIN** - This flag is used to release a connection and no more data is exchanged thereafter. Because packets with SYN and FIN flags have sequence numbers, they are processed in correct order.
  + **Windows Size** - This field is used for flow control between two stations and indicates the amount of buffer (in bytes) the receiver has allocated for a segment, i.e. how much data is the receiver expecting.
  + **Checksum** - This field contains the checksum of Header, Data and Pseudo Headers.
  + **Urgent Pointer** - It points to the urgent data byte if URG flag is set to 1.
  + **Options** - It facilitates additional options which are not covered by the regular header. Option field is always described in 32-bit words. If this field contains data less than 32-bit, padding is used to cover the remaining bits to reach 32-bit boundary.

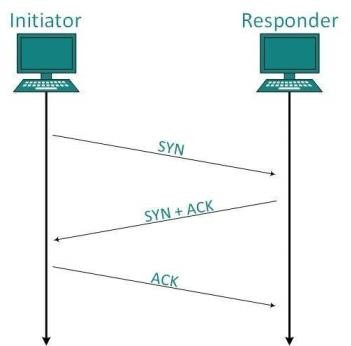
## Addressing

TCP communication between two remote hosts is done by means of port numbers (TSAPs). Ports numbers can range from 0 – 65535 which are divided as:

* + System Ports (0 – 1023)
  + User Ports ( 1024 – 49151)
  + Private/Dynamic Ports (49152 – 65535)

## Connection Management

TCP communication works in Server/Client model. The client initiates the connection and the server either accepts or rejects it. Three-way handshaking is used for connection management.



ESTABLISHMENT

Client initiates the connection and sends the segment with a Sequence number. Server acknowledges it back with its own Sequence number and ACK of client’s segment which is one more than client’s Sequence number. Client after receiving ACK of its segment sends an acknowledgement of Server’s response.

RELEASE

Either of server and client can send TCP segment with FIN flag set to 1. When the receiving end responds it back by ACKnowledging FIN, that direction of TCP communication is closed and connection is released.

## Error Control &and Flow Control

TCP uses port numbers to know what application process it needs to handover the data segment. Along with that, it uses sequence numbers to synchronize itself with the remote host. All data segments are sent and received with sequence numbers. The Sender knows which last data segment was received by the Receiver when it gets ACK. The Receiver knows about the last segment sent by the Sender by referring to the sequence number of recently received packet.

If the sequence number of a segment recently received does not match with the sequence number the receiver was expecting, then it is discarded and NACK is sent back. If two segments arrive with the same sequence number, the TCP timestamp value is compared to make a decision.

## Multiplexing

The technique to combine two or more data streams in one session is called Multiplexing. When a TCP client initializes a connection with Server, it always refers to a well-defined port number which indicates the application process. The client itself uses a randomly generated port number from private port number pools.

Using TCP Multiplexing, a client can communicate with a number of different application process in a single session. For example, a client requests a web page which in turn contains different types of data (HTTP, SMTP, FTP etc.) the TCP session timeout is increased and the session is kept open for longer time so that the three-way handshake overhead can be avoided.

This enables the client system to receive multiple connection over single virtual connection. These virtual connections are not good for Servers if the timeout is too long.

## Congestion Control

When large amount of data is fed to system which is not capable of handling it, congestion occurs. TCP controls congestion by means of Window mechanism. TCP sets a window size telling the other end how much data segment to send. TCP may use three algorithms for congestion control:

* + Additive increase, Multiplicative Decrease
  + Slow Start
  + Timeout React

## Timer Management

TCP uses different types of timer to control and management various tasks:

KEEP-ALIVE TIMER:

* + This timer is used to check the integrity and validity of a connection.
  + When keep-alive time expires, the host sends a probe to check if the connection still exists.

RETRANSMISSION TIMER:

* + This timer maintains stateful session of data sent.
  + If the acknowledgement of sent data does not receive within the Retransmission time, the data segment is sent again.

PERSIST TIMER:

* + TCP session can be paused by either host by sending Window Size 0.
  + To resume the session a host needs to send Window Size with some larger value.
  + If this segment never reaches the other end, both ends may wait for each other for infinite time.
  + When the Persist timer expires, the host re-sends its window size to let the other end know.
  + Persist Timer helps avoid deadlocks in communication.

TIMED-WAIT:

* + After releasing a connection, either of the hosts waits for a Timed-Wait time to terminate the connection completely.
  + This is in order to make sure that the other end has received the acknowledgement of its connection termination request.
  + Timed-out can be a maximum of 240 seconds (4 minutes).

## Crash Recovery

TCP is very reliable protocol. It provides sequence number to each of byte sent in segment. It provides the feedback mechanism i.e. when a host receives a packet, it is bound to ACK that packet having the next sequence number expected (if it is not the last segment).

When a TCP Server crashes mid-way communication and re-starts its process it sends TPDU broadcast to all its hosts. The hosts can then send the last data segment which was never unacknowledged and carry onwards.

## User Datagram Protocol

The User Datagram Protocol (UDP) is simplest Transport Layer communication protocol available of the TCP/IP protocol suite. It involves minimum amount of communication mechanism. UDP is said to be an unreliable transport protocol but it uses IP services which provides best effort delivery mechanism.

In UDP, the receiver does not generate an acknowledgement of packet received and in turn, the sender does not wait for any acknowledgement of packet sent. This shortcoming makes this protocol unreliable as well as easier on processing.

**Requirement of UDP**

A question may arise, why do we need an unreliable protocol to transport the data? We deploy UDP where the acknowledgement packets share significant amount of bandwidth along with the actual data. For example, in case of video streaming, thousands of packets are forwarded towards its users.

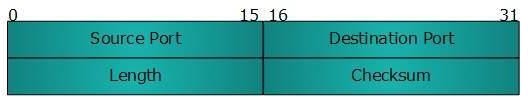
Acknowledging all the packets is troublesome and may contain huge amount of bandwidth wastage. The best delivery mechanism of underlying IP protocol ensures best efforts to deliver its packets, but even if some packets in video streaming get lost, the impact is not calamitous and can be ignored easily. Loss of few packets in video and voice traffic sometimes goes unnoticed.

**Features**

* + UDP is used when acknowledgement of data does not hold any significance.
  + UDP is good protocol for data flowing in one direction.
  + UDP is simple and suitable for query based communications.
  + UDP is not connection oriented.
  + UDP does not provide congestion control mechanism.
  + UDP does not guarantee ordered delivery of data.
  + UDP is stateless.
  + UDP is suitable protocol for streaming applications such as VoIP, multimedia streaming.

**UDP Header**

UDP header is as simple as its function.



UDP header contains four main parameters:

* + **Source Port** - This 16 bits information is used to identify the source port of the packet.
  + **Destination Port** - This 16 bits information, is used identify application level service on destination machine.
  + **Length** - Length field specifies the entire length of UDP packet (including header). It is 16- bits field and minimum value is 8-byte, i.e. the size of UDP header itself.
  + **Checksum** - This field stores the checksum value generated by the sender before sending. IPv4 has this field as optional so when checksum field does not contain any value it is made 0 and all its bits are set to zero.

**UDP application**

Here are few applications where UDP is used to transmit data:

* + Domain Name Services
  + Simple Network Management Protocol
  + Trivial File Transfer Protocol
  + Routing Information Protocol
  + Kerberos