CS19541 COMPUTER NETWORKS

UNIT-I FUNDAMENTALS AND DATA LINK LAYER

Building a network – Requirements – Layering and protocols – Internet Architecture – Network software – Application Programming Interface (sockets) - Performance – Link layer Services - Framing – Error Detection and Correction - Reliable transmission

UNIT-II MEDIA ACCESS AND INTERNETWORKING

Media Access Protocols – ALOHA - CSMA/CA/CD –Ethernet – Wireless LANs - 802.11- Bluetooth - Switching and Forwarding - Bridges and LAN Switches – Basic Internetworking- IP Service Model – IP fragmentation - Global Addresses – ARP - DHCP – ICMP- Virtual Networks and Tunnels.

UNIT-III ROUTING

Routing – Network as Graph - Distance Vector – Link State – Global Internet – Subnetting - Classless Routing (CIDR) - BGP- IPv6 – Multicast routing - DVMRP-PIM.

UNIT-IV TRANSPORT LAYER

Overview of Transport layer – UDP – TCP - Segment Format – Connection Management – Adaptive Retransmission - TCP Congestion control - Congestion avoidance (DECbit, RED) – QoS – Application requirements.

UNIT-V APPLICATION LAYER

E-Mail (SMTP, MIME, POP3, IMAP), HTTP – DNS - FTP - Telnet – web services - SNMP – MIB – RMON.

Text Books

1. Larry L. Peterson, Bruce S. Davie, “Computer Networks: A Systems Approach”, Fifth Edition, Morgan Kaufmann Publishers Inc., 2011.
2. Behrouz A. Forouzan, “Data Communications and Networking”, Fifth Edition, McGrawHill, 2017

Reference Books

1. William Stallings, “SNMP, SNMPv2, SNMPv3 and RMON 1 and 2”, Third Edition, Pearson Edition, 2009.
2. James F. Kurose, Keith W. Ross,” Computer Networking - A Top-Down Approach Featuring the Internet”, Seventh Edition, Pearson Education, 2017.
3. Andrew S. Tanenbaum, David J. Wetherall, “Computer Networks”, 5th Edition, Prentice Hall publisher, 2010.
4. William Stallings, “Data and Computer Communications”, Eighth Edition, Pearson Education, 2011.

CS19541 - COMPUTER NETWORKS

UNIT-I FUNDAMENTALS AND DATA LINK LAYER

Building a network – Requirements – Layering and protocols – Internet Architecture – Network software –Application Programming Interface (sockets) - Performance – Link layer Services - Framing– Error Detection and Correction - Reliable transmission

Computer network

Computer network is defined as the interconnection of nodes (computers and other devices) connected by a communication channel (wired or wireless) that facilitates communication among users and allows them to share resources (Information, hardware and software resources.)

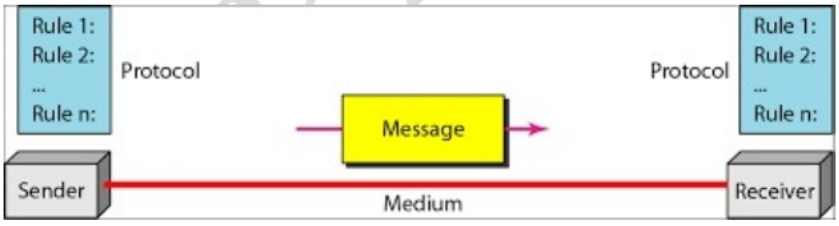
INTRODUCTION

Data Communications is the transfer of data or information between a sender and a receiver. The sender transmits the data and the receiver receives it.

The effectiveness of a data communication depends on three characteristics,

Delivery : The system must deliver data to correct destination.
Accuracy : The system must deliver data accurately.
Timeliness : The system must deliver data in a timely manner.

Components of data communication



- **Sender:** It is the transmitter of data. Some examples are Terminal, Computer, and Mainframe.
- **Medium:** The communication stream through which the data is being transmitted. Some examples are: Cabling, Microwave, Fiber optics, Radio Frequencies (RF), Infrared Wireless
- **Receiver:** The receiver of the data transmitted. Some examples are Printer, Terminal, Mainframe, and Computer.
- **Message:** It is the data that is being transmitted from the Source/Sender to the Destination/Receiver.

- **Protocol:** It is the set of rules and regulations (resides in the form of software and hardware) that are to be followed for communication. If protocol is not present it implies the nodes are connected but they can't communicate.

Figure 1.2: DTE and DCE

- **DCE:** The interface between the Sender and the Medium, and the Medium & the Receiver is called the DCE (Data Communication Equipment) and is a physical piece of equipment.
- **DTE:** Data Terminal Equipment is the Telecommunication name given to the Source and Receiver's equipment.

Direction of Data Flow

It defines how the data flows between two end points. Based on the direction and time of flow there are three kinds of data flow,

1. Simplex
2. Half-Duplex
3. Full-duplex.

Simplex:

In this type of data communication, the data flows in only one direction on the data communication line (medium).

Examples are Keyboard, Monitor, Radio and Television broadcasts

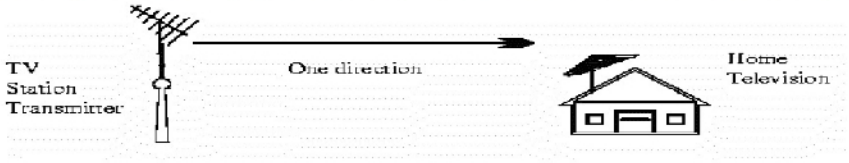


Figure 1.3: Simplex Data Flow

Half-Duplex:

In this type of data communication, the data flows in both directions but at a time in only one direction on the data communication line.

Example Conversation on walkie-talkies is a half-duplex data flow.

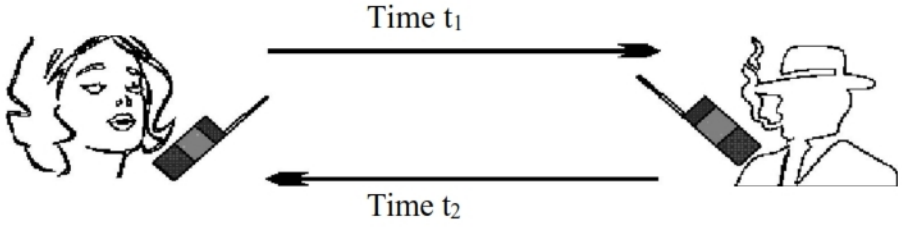


Figure: Half-Duplex Data Flow

Full-Duplex:

In this type of data communication, the data flows in both directions simultaneously. Example Telephones and Modems

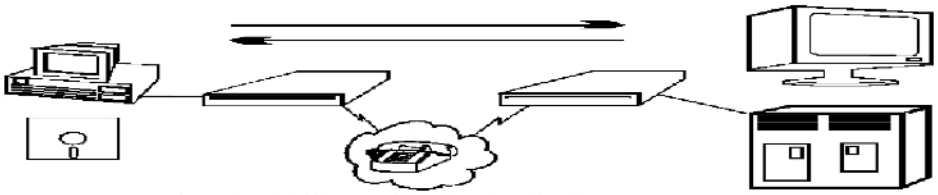


Figure: Full-Duplex Data Flow

Types of Connections / Line configuration / Direct links

There are two types of line configuration, they are

1. point to po4int
2. multipoint

Point to Point

- It provides a dedicated link between two devices.
- The entire capacity (bandwidth) of the link is reserved for transmission between these two devices.
  The two devices are connected by means of a pair of wires or using a microwave or satellite link.
- Eg : Computers connected by telephone line
  PPP connection between remote and TV

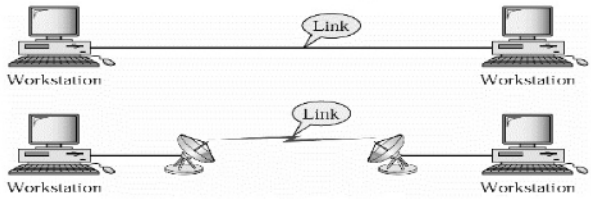


Figure: Point-to-point link

Multipoint (Multiple Access)

- It is a connection in which more than two devices share a single link.
- In this environment a single channel is shared, either spatially or temporally.
- If several devices can use the link at the same time it said to be spatially shared.
- If the devices take turn to use the link then it is referred to as timesharing.

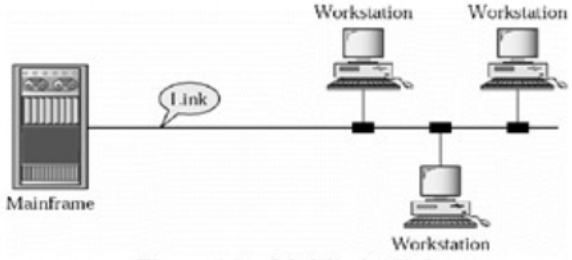


Figure : Multipoint link

Different Communication Modes

1) **Unicast - one to one**
   Unicast packets are sent from host to host. The communication is from a single host to another single host.
2) **Broadcast – one to all**
   Broadcast is when a single device is transmitting a packet to all other devices in a given address range.
3) **Multicast – one to many**
   Multicast enables a single device to communicate with a specific set of hosts.

Categories of networks

The three primary categories of network are Local Area Network (LAN), Metropolitan Area Network (MAN), and Wide Area Network (WAN). The category into which a network fall is determined by its size, ownership, the distance it covers and its physical architecture.

LAN

- A LAN is usually privately owned and links the devices in a single office, building or campus.
- A LAN can be as simple as two PCs or it can extend throughout a company. LAN size is limited to a few kilometers.
- The most widely used LAN technology is the Ethernet technology developed by the Xerox Corporation.

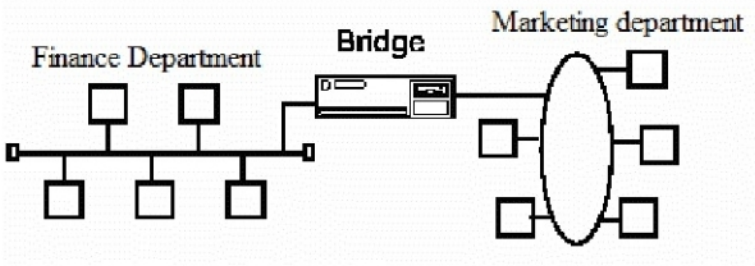


Figure 1.8: Local Area Network

MAN

- A MAN is designed to extend over an entire city.
- It could be a single network such as cable TV network or connect a number of LANs into a larger network.
- A MAN can be owned by a private company or it may be a service provided by a public company, such as local telephone company.
- Telephone companies provide a popular MAN service called (SMDS) Switched Multi-megabit Data Services.

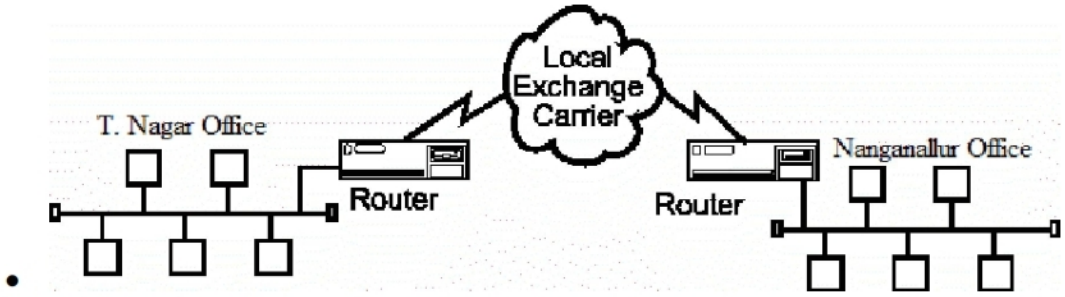


Figure: Metropolitan Area Network

WAN

- A WAN provides long distance transmission of data, voice, image and video information over large geographic areas.
- Transmission rates are typically 2 Mbps, 34 Mbps, 45 Mbps, 155 Mbps and 625 Mbps. WAN utilize public, leased, or private communication equipment usually in combinations and therefore span an unlimited number of miles.
- A WAN that is wholly owned and used by a single company is referred to as an Enterprise Network.

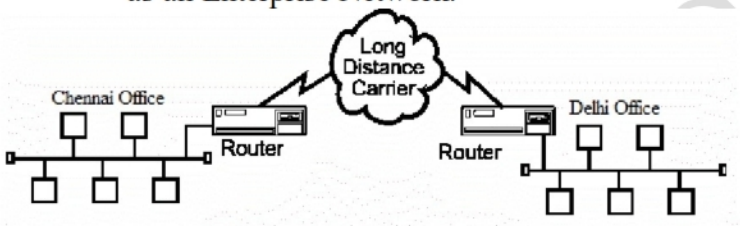


Figure 1.10: Wide Area Network

PAN

- A Personal Area Network (PAN) is the interconnection of devices within the range of an individual person, typically within a range of 10 meters.
- For example, a person traveling with a laptop, a personal digital assistant (PDA), and a portable printer could interconnect them without having to plug anything in, using some form of wireless technology.
- Typically, this kind of personal area network could also be interconnected without wires to the Internet or other networks.

Figure: Personal Area Network

Topology

Physical Topology refers to the fashion in which nodes in the network is laid out physically. The topology of a network is the geometric representation of the relationship of all the links and the linking devices tone another. It defines the physical layout of the network.

The basic topologies are:

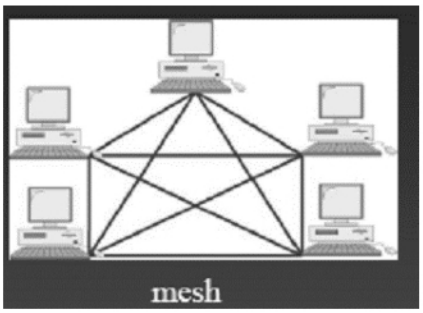
- Mesh
- Star
- Bus
- Ring
- Hybrid (combination of other types)

Mesh

- In a mesh topology each device has a dedicated point to point link to every other device.
- The term dedicated means that the link carries traffic only between the two devices it connects.
- A fully connected mesh network therefore has n (n-1)/2 physical channels to link n devices. To accommodate that many links every device on the network should have (n-1) I/O ports.

Merits

- Eliminates the traffic problems that occur when the links are shared by multiple devices.
- If one link becomes unusable, it does not incapacitate the entire system.
- Since every message travels along a dedicated line only the intended recipient will receive the message and hence the data is secure.

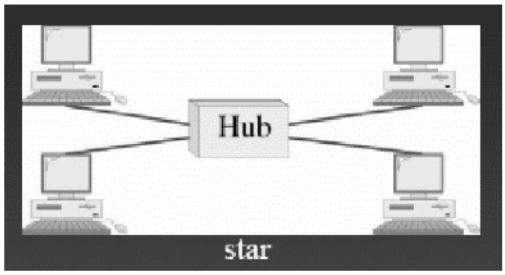


Demerits

- The amount of cabling and the I/O ports required increases with the number of devices connected in the network
- Installation and reconnection are difficult
- The sheer bulk of the wire accommodates more space than available.
- The hardware required to connect each link can be prohibitively expensive.

Star

- Each device has a dedicated point to point link only to a central controller usually called a hub.
- If one device has to send data to another it sends the data to the controller, which then relays the data to the other connected device.



Merits

- Less expensive than a mesh topology. Each device needs only one link and I/O port.
- Installation and reconfigure is easy.
- Robustness. If one link fails only that link is affected.
- Requires less cable than a mesh.

Demerits

- Require more cable compared to bus and ring topologies.
- Failure of the central controller incapacitates the entire network.

Bus

- One long cable acts as a backbone to link all the devices in a network.
- Nodes are connected to the bus cable by drop lines and taps.
- A drop line is a connection running between the device and the main cable.
- A tap is a connector that either splices into the main cable to create a contact with a metallic core.
- As the signal travels farther and farther, it becomes weaker. So there is limitation in the number of taps a bus can support and on the distance between those taps.

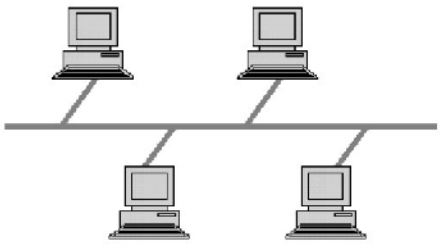


Figure 1.15: Bus Topology

Merits

- Ease of installation.
- Bus uses less cabling than mesh or star topologies.

Demerits

- Difficult reconnection and isolation.
- Signal reflection at the taps can cause degradation in quality.
- A fault or break in the bus cable stops all transmission.
- It also reflects signals back in the direction of origin creating noise in both directions.

Ring

- Each device has a dedicated point to point connection only with the two devices on either side of it.
- A signal is passed along the ring in one direction from device to device until it reaches the destination.
- Each device in the ring incorporates a repeater

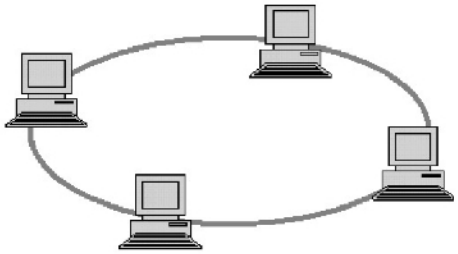


Figure 1.16: Ring Topology

Merits:

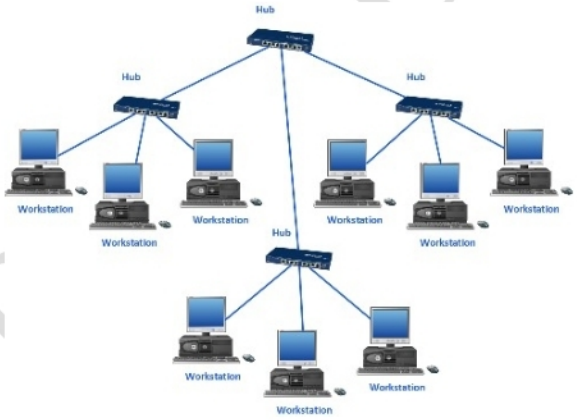
- Easy to install and reconfigure.
- To add or delete a device requires changing only two connections.

Demerits

- A break in the ring disables the entire network. It can be solved by using a dual ring or a switch capable of closing off the break.

Hybrid Topology

- A hybrid topology is a type of network topology that uses two or more other network topologies, including bus topology, mesh topology, ring topology, star topology, and tree topology.



Requirements

The following are the requirements to be followed to build any network,

- ✓ Identification of constraints and requirements
- ✓ Connectivity need to be decided
- ✓ Cost-effective resource sharing
- ✓ Support for common services

Identification of constraints and requirements of a network

Three groups of people might list their requirements for a network,

1. Application Programmer
   - List the services that his application needs: delay bounded delivery of data
2. Network Designer
   - Designs a cost-effective network with sharable resources
3. Network Provider
   - List the characteristics of a system that is easy to manage

Connectivity in a network:

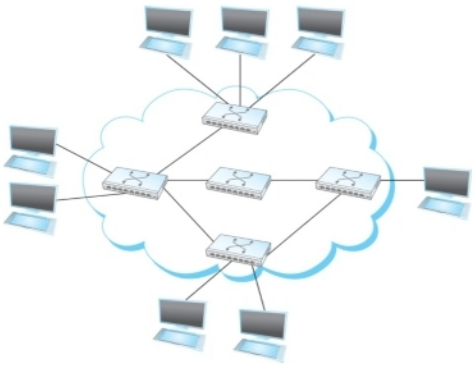
Links, Nodes, and Clouds

Network connectivity occurs at many different levels. At the lowest level, a network can consist of two or more computers directly connected by some physical medium, such as a coaxial cable or an optical fiber. We call such a physical medium a *link*, and we often refer to the computers it connects as *nodes*.

A set of computers can be indirectly connected. A set of independent networks (clouds) are interconnected to form an internetwork. A node that is connected to two or more networks is commonly called a router or gateway. A router/gateway forwards messages from one network to another.

Switched Network

Those nodes that are attached to at least two links run software that forwards data received on one link out on another. If organized in a systematic way, these forwarding nodes form a switched network. There are numerous types of switched networks, of which the two most common are circuit switched and packet switched.



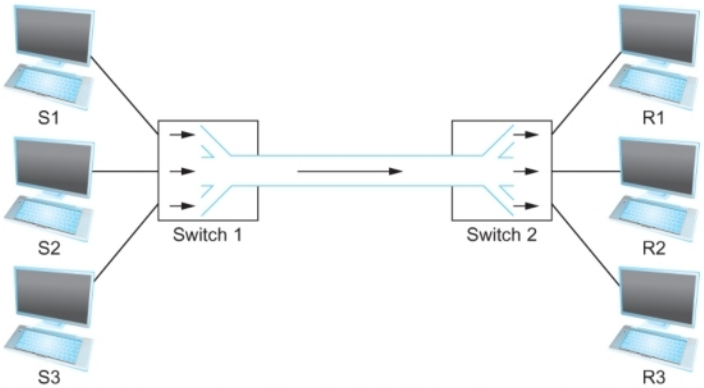
Types of switched networks

1. **Circuit Switched**
   First establishes a dedicated circuit across a sequence of links and then allows the source node to send a stream of bits across this circuit to a destination node.

2. **Packet Switched networks**
   It uses a strategy called store-and-forward. Each node in a store-and-forward network first receives a complete packet over some link, stores the packet in its internal memory, and then forwards the complete packet to the next node.

Cost-Effective Resource Sharing

- Resource: links and nodes
- How to share a link?
  - **Multiplexing**
    Analogy to a timesharing computer system.
  - **De-multiplexing**



Multiplexing multiple logical flows over a single physical link

Synchronous Time-division Multiplexing (STDM)

- Equal-sized quanta
- Round-robin fashion
- Time slots/data transmitted in predetermined slots

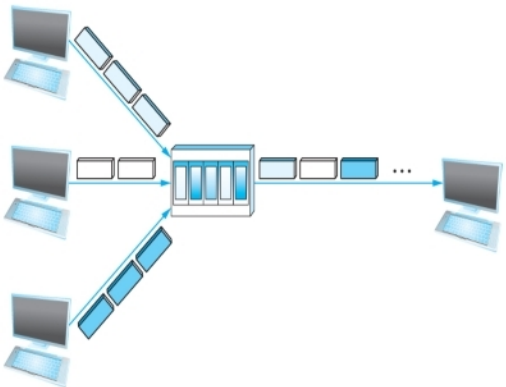
Frequency Division Multiplexing(FDM)

Eg:Diff. TV stations with diff. frequencies.

- Both STDM and FDM waste resources and hard to accommodate changes (fixed time slots and frequencies)

Statistical Multiplexing

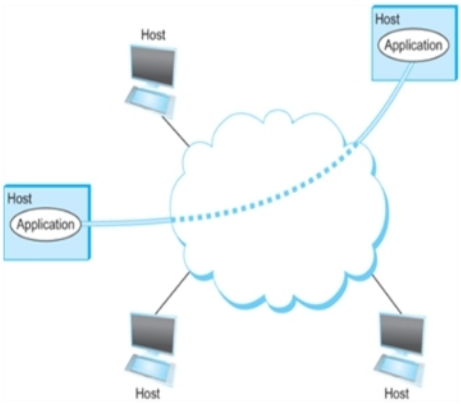
Like STDM: sharing over time but data is transmitted based on demand rather than during a predetermined time slot.



A switch multiplexing packets from multiple sources onto one shared link

Support for Common Services

- Logical Channels
  - Application-to-Application communication path or a pipe
- Client/Server
- Two types of communication channel
  - Request/Reply Channels
  - Message Stream Channels



Process communicating over an abstract channel

NETWORK ARCHITECTURES

Network designers have developed general blueprints—usually called network architectures—that guide the design and implementation of networks.

1. Layering & protocols
2. OSI layers
3. Internet Architecture

Layering and Protocols

Layering provides two nice features.

- It decomposes the problem of building a network into more manageable components.
- It provides a more modular design.

First, it decomposes the problem of building a network into more manageable components. Rather than implementing a monolithic piece of software that does everything you will ever want, you can implement several layers, each of which solves one part of the problem.

Second, it provides a more modular design. If you decide that you want to add some new service, you may only need to modify the functionality at one layer, reusing the functions provided at all the other layers.

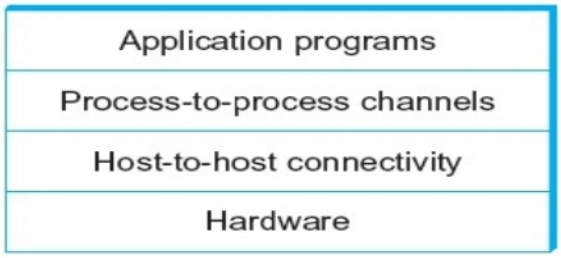
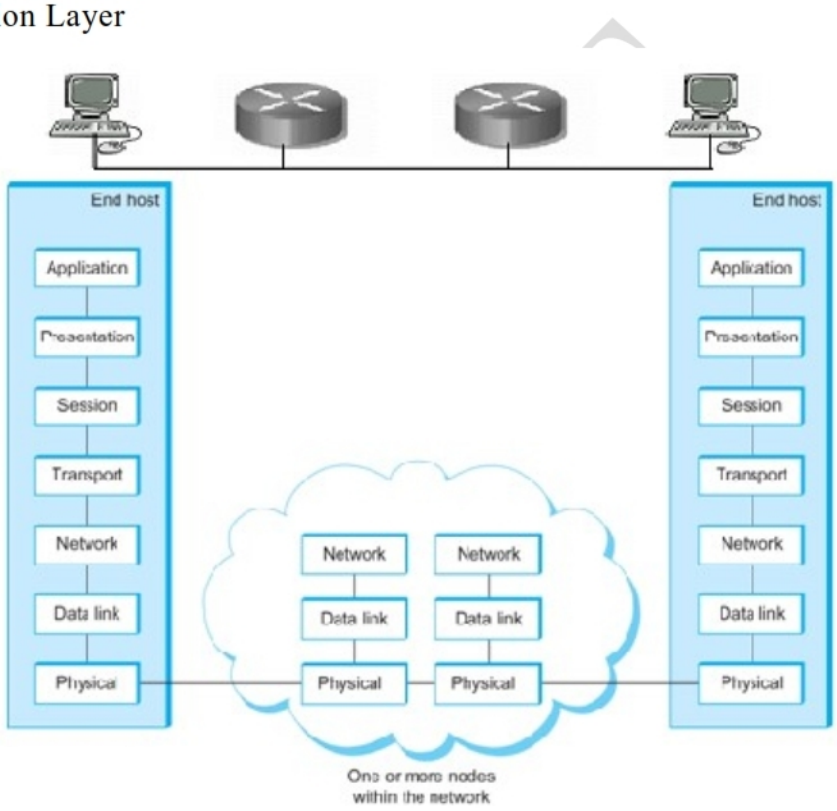


Fig: Example of a layered network system

ISO - OSI reference Model

- The International Standards Organization (ISO) - Open Systems Interconnect (OSI) is a standard which defines a set of rules to govern the data communication between two devices without worrying about the underlying architecture of the devices. It describes the functionalities for transfer of data between each layer. Each layer has a specific function.
  - *ISO is the organization; OSI is the model.*
  - There are 7 Layers in the OSI model

1. Physical Layer
2. Datalink Layer
3. Network Layer
4. Transport Layer
5. Session Layer
6. Presentation Layer
7. Application Layer



Two interfaces

- Service interface
- Peer interface

Service interface- defines the operations that local objects can perform on the protocol.

Peer interface- defines the form and meaning of messages exchanged between protocol peers to implement the communication service.

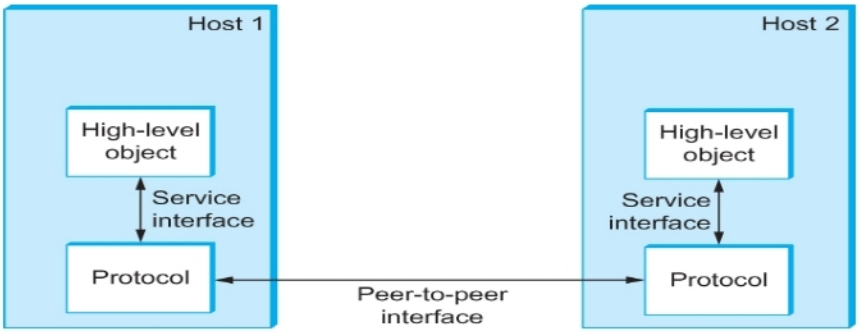
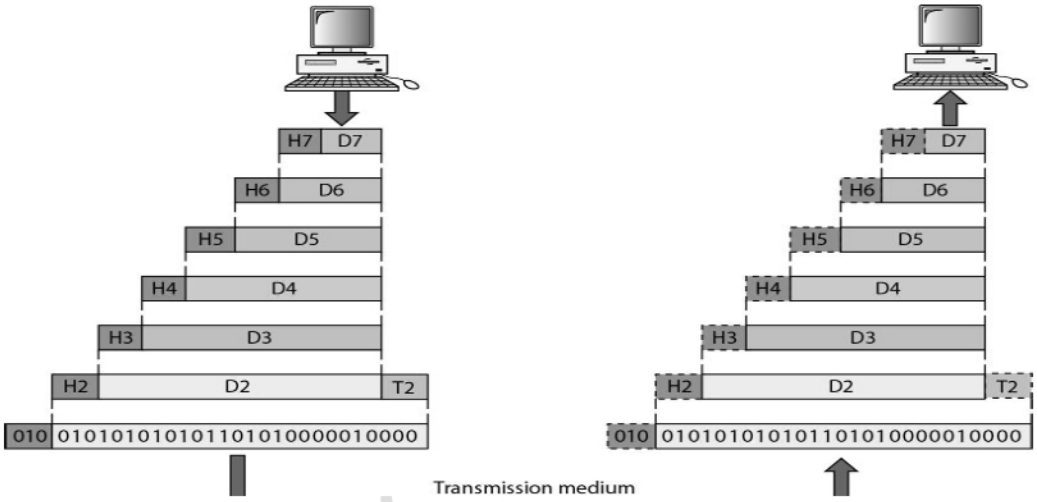


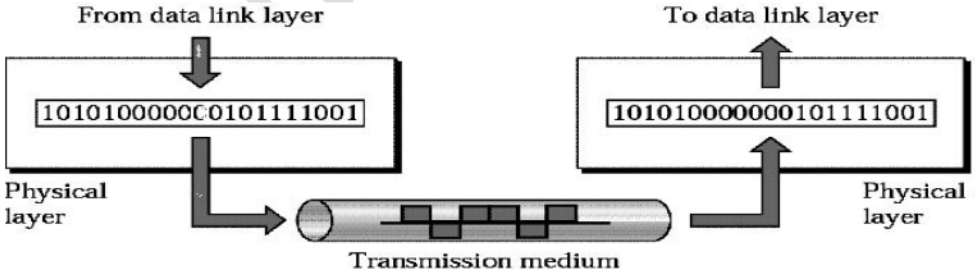
Fig : Service and Peer Interfaces

A Data exchange using the OSI model



1. PHYSICAL LAYER (bit level transmission)

The physical layer coordinates the functions required to transmit a bit stream over a physical medium. It is responsible for moving bits from one node to the next.



The physical layer is concerned with the following:

- ✓ **Physical characteristics of interfaces and media -** The physical layer defines the characteristics of the interface between the devices and the transmission medium.

- ✓ **Representation of bits -** To transmit the stream of bits, it must be encoded to signals. The physical layer defines the type of encoding.

- ✓ **Data Rate or Transmission rate -** The number of bits sent each second – is also defined by the physical layer.

- ✓ **Synchronization of bits -** The sender and receiver must be synchronized at the bit level. Their clocks must be synchronized.

- ✓ **Line Configuration -** In a point-to-point configuration, two devices are connected together through a dedicated link. In a multipoint configuration, a link is shared between several devices.

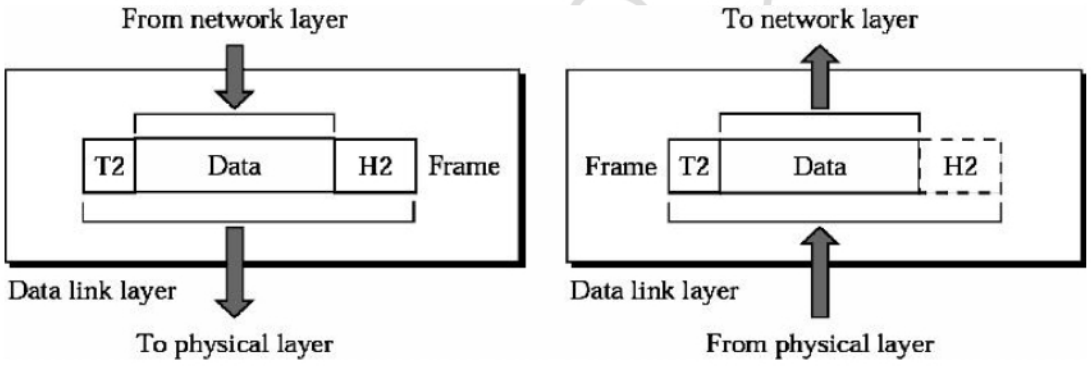
- ✓ **Physical Topology -** The physical topology defines how devices are connected to make a network. Devices can be connected using a mesh, bus, star or ring topology.

- ✓ **Transmission Mode -** The physical layer also defines the direction of transmission between two devices: simplex, half-duplex or full-duplex.

2. DATA LINK LAYER

(Node to node delivery or hop to hop delivery of frames)

It is responsible for transmitting frames from one node to next node.



The other responsibilities of this layer are

- ✓ **Framing -** Divides the stream of bits received into data units called frames.

- ✓ **Physical addressing** – If frames are to be distributed to different systems on the n/w, data link layer adds a header to the frame to define the sender and receiver.

- ✓ **Flow control**- If the rate at which the data are absorbed by the receiver is less than the rate produced in the sender, the Data link layer imposes a flow control mechanism.

- ✓ **Error control**- Used for detecting and retransmitting damaged or lost frames and to prevent duplication of frames. This is achieved through a trailer added at the end of the frame.

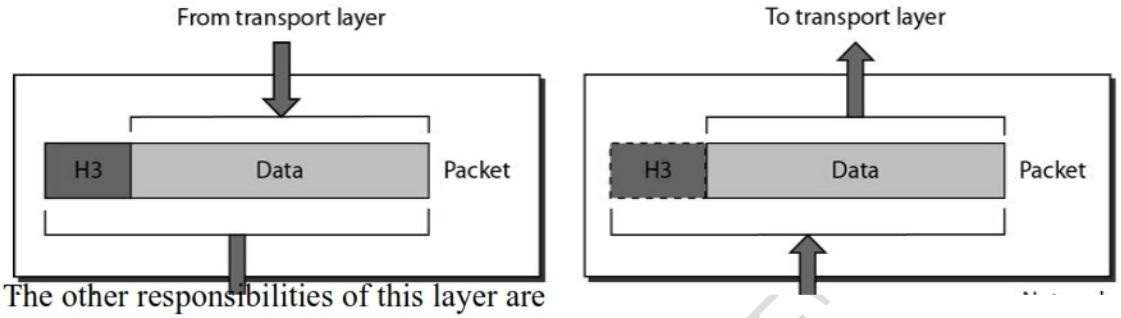
- ✓ **Access control** -Used to determine which device has control over the link at any given time.

3. NETWORK LAYER

(Source to destination delivery of individual packets)

This layer is responsible for the delivery of packets from source to destination. It is mainly required, when it is necessary to send information from one network to another.

It ensures that each packet gets from its point of origin to its final destination.



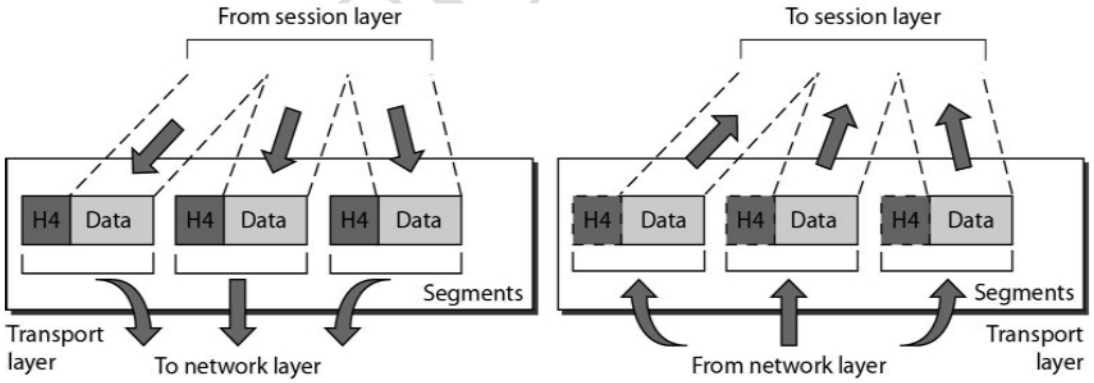
The other responsibilities of this layer are

- ✓ **Logical addressing** - If a packet passes the n/w boundary, we need another addressing system for source and destination called logical address.
- ✓ **Routing** – The devices which connects various networks called routers are responsible for delivering packets to final destination.

4. TRANSPORT LAYER

(Source to destination delivery of entire message)

- ✓ It is responsible for **Process to Process** delivery of entire message.
- ✓ It also ensures whether the message arrives in order or not.



The other responsibilities of this layer are

- ✓ **Port addressing** - The header in this must therefore include a address called port address. This layer gets the entire message to the correct process on that computer.

- ✓ **Segmentation and reassembly** - The message is divided into segments and each segment is assigned a sequence number. These numbers are arranged correctly on the arrival side by this layer.

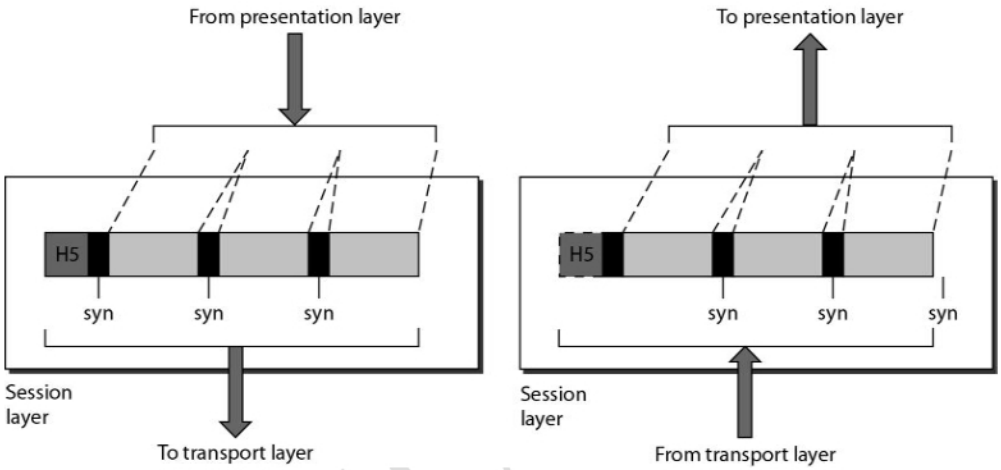
- ✓ **Connection control** - This can either be **connectionless or connection-oriented.** The connectionless treats each segment as a individual packet and delivers to the destination. The connection-oriented makes connection on the destination side before the delivery. After the delivery the termination will be terminated.

- ☐ **Flow and error control** - Similar to data link layer, but process to process take place. ie end to end error & flow control

5. SESSION LAYER

(Responsible for dialog control and synchronization)

This layer establishes, manages and terminates connections between applications.



The other responsibilities of this layer are

- ✓ **Dialog control** - This session allows two systems to enter into a dialog either in half duplex or full duplex.

- ✓ **Synchronization**-This allows to add checkpoints into a stream of data. Example: If a system is sending a file of 2000 pages, check points may be inserted after every 100 pages to ensure that each 100 page unit is advised and acknowledged independently. So if a crash happens during the transmission of page 523, retransmission begins at page 501, pages 1 to 500 need not be retransmitted.

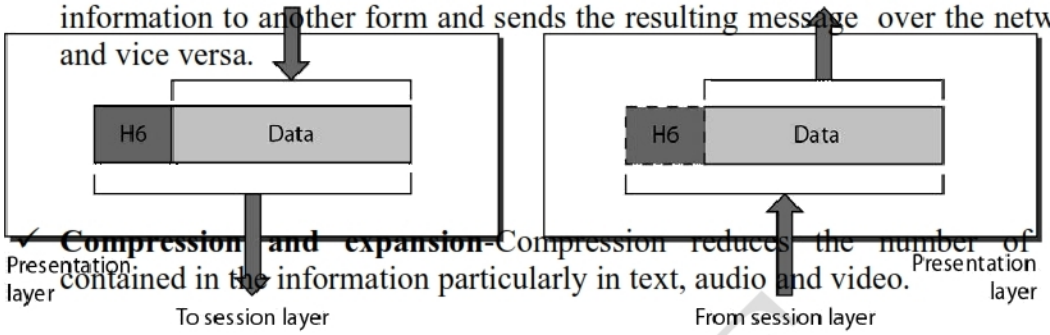
6. PRESENTATION LAYER

Presentation Layer is concerned with the syntax and semantics of the information exchanged between two systems.

The presentation layer is responsible for translation, compression, and encryption. It is concerned with the syntax and semantics of information exchanged between two systems.

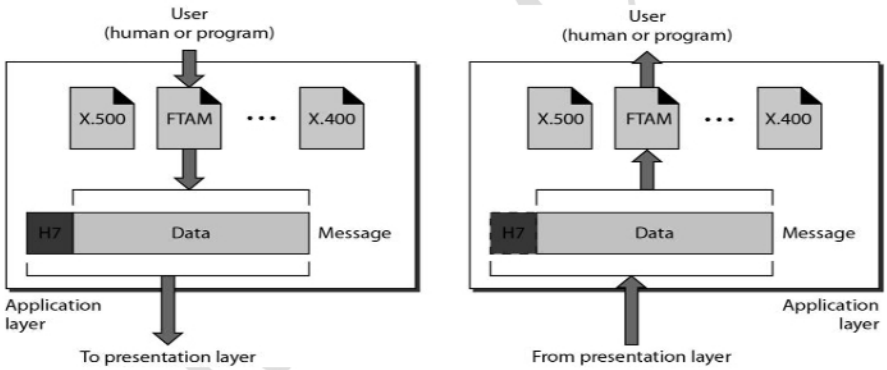
The other responsibilities of this layer are

- ✓ **Translation** – Different computers use different encoding system, this layer is responsible for interoperability between these different encoding methods. It will change the message into some common format.
- ✓ **Encryption and decryption**-It means that sender transforms the original information to another form and sends the resulting message over the network and vice versa.
- ✓ **Compression and expansion**-Compression reduces the number of bits contained in the information particularly in text, audio and video.



7. APPLICATION LAYER

This layer enables the **user to access the network resources**. This allows the user to log on to remote user.



The other responsibilities of this layer are

- ✓ **FTAM(file transfer,access,mgmt)** - Allows user to access files in a remote host.
- ✓ **Network Virtual terminal:** A network virtual terminal is a software version of a physical terminal and allows a user to log on to a remote host.
- ✓ **Mail services** - Provides email forwarding and storage.
- ✓ **Directory services** - Provides database sources to access information about various sources and objects.

Internet(TCP/IP) Architecture

The Internet architecture, which is sometimes called the TCP/IP architecture after its two main protocols. The Internet architecture evolved out of experiences with an earlier packet-switched network called the ARPANET. Both the Internet and the ARPANET were funded by the Advanced Research Projects Agency (ARPA), one of the research and development funding agencies of the U.S. Department of Defense. The Internet and ARPANET were around before the OSI model, and the experience gained from building them was a major influence on the OSI reference model.

Layers of TCP / IP model

Application layer
Transport layer
Internet layer
Network interface layer

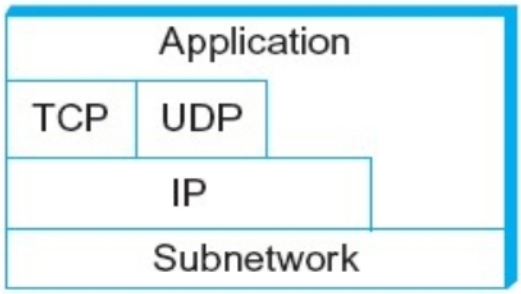
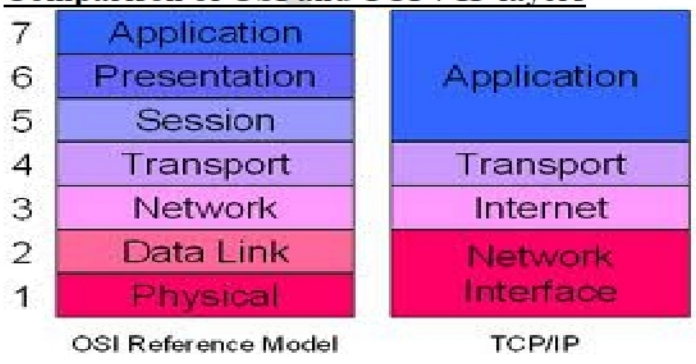


Fig: Alternative view of the Internet architecture. The "Network Interface" layer shown here is sometimes referred to as the "sub-network" or "link" layer.

While the 7-layer OSI model can, with some imagination, be applied to the Internet, a 4-layer model is often used instead.

The TCP/IP protocol suite was developed prior to the OSI model. Therefore, the layers in the TCP/IP protocol suite do not exactly match those in the OSI model.

Comparison of OSI and TCP / IP layers



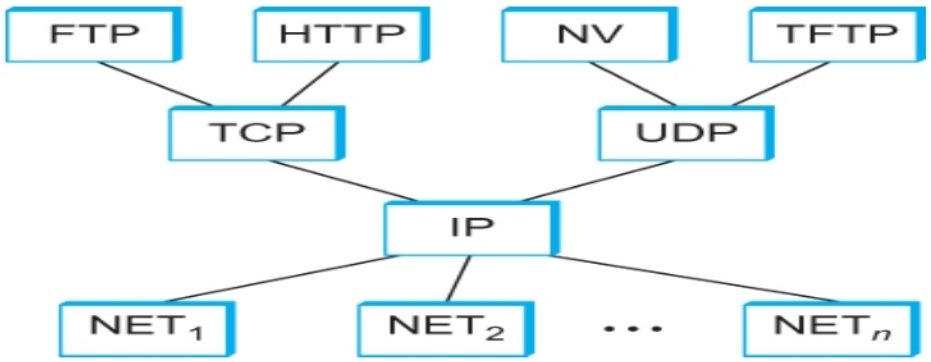


fig: Internet Protocol Graph

At the lowest level is a wide variety of network protocols, denoted NET1, NET2, and so on. In practice, these protocols are implemented by a combination of hardware (e.g., a network adaptor) and software (e.g., a network device driver). For example, you might find Ethernet or wireless protocols (such as the 802.11 Wi-Fi standards) at this layer. (These protocols in turn may actually involve several sublayers, but the Internet architecture does not presume anything about them.)

The second layer consists of a single protocol—the Internet Protocol (IP). This is the protocol that supports the interconnection of multiple networking technologies into a single, logical internetwork.

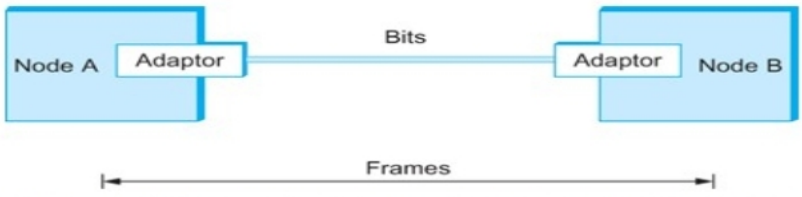
The third layer contains two main protocols—the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP).TCP and UDP provide alternative logical channels to application programs: TCP provides a reliable byte-stream channel, and UDP provides an unreliable datagram delivery channel (datagram may be thought of as a synonym for message). In the language of the Internet, TCP and UDP are sometimes called end-to-end protocols, although it is equally correct to refer to them as transport protocols.

Running above the transport layer is a range of application protocols, such as HTTP, FTP, Telnet (remote login), and the Simple Mail Transfer Protocol (SMTP), that enable the interoperation of popular applications.

Framing-Datalink Layer

Data link layer divides the stream of bits received from the upper layer (network layer) into manageable data units called frames. It adds a header to the frame to define the physical address (source address & destination address).

- Blocks of data (called frames at this level) are exchanged between nodes.
- It is the network adaptor that enables the nodes to exchange frames.
- When node A wishes to transmit a frame to node B, it tells its adaptor to transmit a frame from the node's memory.
- This results in a sequence of bits being sent over the link.
- The adaptor on node B then collects together the sequence of bits arriving on the link and deposits the corresponding frame in B's memory.
- Recognizing exactly what set of bits constitutes a frame—that is, determining where the frame begins and ends—is the central challenge faced by the adaptor.



Bits flow between adaptors, frames between hosts

Fixed-Size Framing: Frames can be of fixed or variable size. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter. An example of this type of framing is the ATM wide-area network, which uses frames of fixed size called cells.

Variable-Size Framing: In variable-size framing, we need a way to define the end of the frame and the beginning of the next. Three approaches were used for this purpose:

Framing Protocols:

(i) Byte-oriented protocols
(ii) Bit-oriented protocols
(iii) Clock-based protocols

Byte-oriented protocols:

- sentinel approach
    1. BISYNC – Binary Synchronous Communication
    2. PPP – Point-to-point Protocol
- Byte-counting approach
    3. DDCMP- Digital Data Communication Message Protocol

Bit-oriented protocols:

(i) HDLC – High Level Data Link Control

Clock-based protocols:

(i) SONET - Synchronous Optical Network

Byte oriented Protocols

Sentinel Approach

- It uses special characters known as sentinel characters to indicate where the frame starts and ends.

Binary Synchronous Communication (BISYNC)

- The beginning of a frame is denoted by sending a special SYN (synchronization) character.
- The data portion of the frame is then contained between special *sentinel characters:* STX (start of text) and ETX (end of text).
- The SOH (start of header) field serves much the same purpose as the STX field.

BISYNC Frame Format

- The problem with the sentinel approach, of course, is that the ETX character might appear in the data portion of the frame.
- BISYNC overcomes this problem by "escaping" the ETX character by preceding it with a DLE (data-link-escape) character whenever it appears in the body of a frame; the DLE character is also escaped (by preceding it with an extra DLE) in the frame body. This approach is often called **byte stuffing** or **character stuffing** because extra characters are inserted in the data portion of the frame.

Example

- Solution: Byte-stuffing (or "character stuffing")
  - Sender: insert a special escape character DLE before any occurence of ETX in Data portion of frame
    - Data Link Escape (DLE) = 00010000
    - For consistency, must also "escape" any occurrences of DLE in the data
  - Receiver: while looking for ETX, if DLE is encountered, throw it away and treat the following character as data

PPP - Point-to-Point Protocol (PPP):

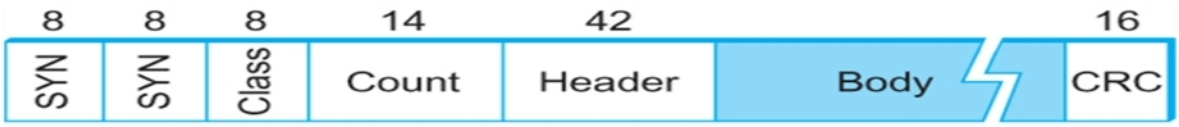
- It is similar to BISYNC in that it uses character stuffing. The format for a PPP frame is given below.
- The special start-of-text character, denoted as the Flag field is 01111110, which is byte stuffed if it occurs within the payload field.
- The Address and Control fields usually contain default values.
- The Address field which is always set to the binary value 11111111, indicates that all stations are to accept the frame. This value avoids the issue of using data link addresses.
- The default value of the Control field is 00000011. This value indicates an unnumbered frame. In other words, PPP does not provide reliable transmission using sequence numbers and acknowledgements.
- The Protocol field is used for demultiplexing. It identifies the high-level protocol such as IP or IPX (an IP-like protocol developed by Novell).
- The frame payload size can be negotiated, but it is 1500 bytes by default.
- The Checksum field is either 2 (by default) or 4 bytes long.
- The PPP frame format is unusual in that several of the field sizes are negotiated rather than fixed.
- This negotiation is conducted by a protocol called LCP (Link Control Protocol). PPP and LCP work in tandem: LCP is also involved in establishing a link between two peers when both sides detect the **carrier signal.**

PPP Frame Format

Byte-counting approach

DDCMP protocol – Digital Data Communication Message Protocol:

- The COUNT field specifies how many bytes are contained in the frame's body.
- One danger with this approach is that a transmission error could corrupt the COUNT field, in which case the end of the frame would not be correctly detected.This is sometimes called **a framing error.**
- The receiver will then wait until it sees the next SYN character to start collecting the bytes that make up the next frame.



DDCMP Frame Format

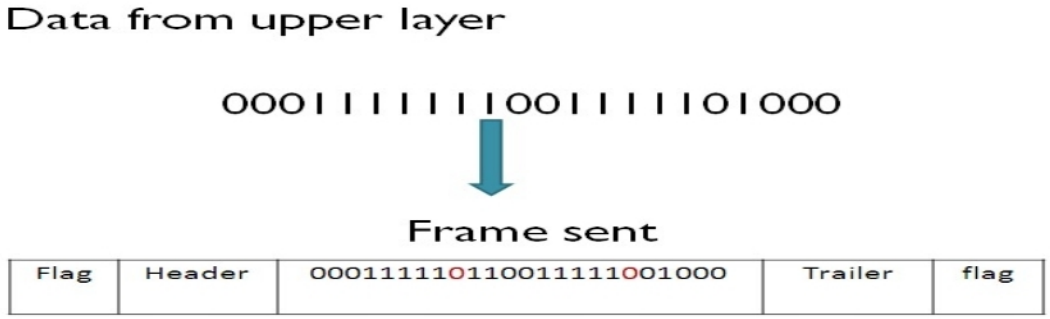
Bit-Oriented Protocols
(HDLC)

- A bit oriented protocol is not concerned with byte boundaries—it simply views the frame as a collection of bits.
- The Synchronous Data Link Control (SDLC) protocol developed by IBM is an example of a bit-oriented protocol; SDLC was later standardized by the ISO as the High-Level Data Link Control (HDLC) protocol.

- HDLC denotes both the beginning and the end of a frame with the distinguished bit sequence 01111110.
- This sequence is also transmitted during any times that the link is idle so that the sender and receiver can keep their clocks synchronized.
- Because this sequence might appear anywhere in the body of the frame—in fact, the bits 01111110 might cross byte boundaries—bit-oriented protocols use a technique known as ***bit stuffing***.

Bit stuffing in the HDLC protocol works as follows.

- On the sending side, whenever 0 followed by five consecutive 1's has been seen in the body of the message the sender inserts a 0.
- On the receiving side, when 0 followed by five consecutive l's arrive, the receiver makes its decision based on the next bit it sees (i.e., the bit following the five 1' s).
- If the next bit is 0, it must have been stuffed, and so the receiver removes it.
- If the next bit is a 1, then one of two things is true:
  Either this is the end-of frame marker or an error has been introduced into the bit stream.



After receiving 5 1s

- next bit
  - 0 >> *stuffed bit* >> removed

    Example
    - bits received 0111 1101 010; bits retained (data): 0111 1110 10
    - bits received 0111 1100 010; bits retained (data): 0111 1100 10
  - 1 >> bits received 0111 1110 or 0111 1111

    next bit
    - 0 END of Frame marker
    - 1 Error , frame is discarded; receiver waits for next 0111 1110 to start receiving next frame

Clock based Protocols

Clock-Based Framing (SONET):

- A third approach to framing is exemplified by the Synchronous Optical Network (SONET) standard.
- SONET addresses both the framing problem and the encoding problem.
- It also addresses a problem that is very important for phone companies—the multiplexing of several low-speed links onto one high-speed link.
- No bit stuffing is used, so that a frame's length does not depend on the data being sent.
- A STS-1 (lowest-speed SONET link runs at 51.84 Mbps.) frame is shown in Figure 2.13.
- It is arranged as nine rows of 90 bytes each.
- The first 3 bytes of each row are overhead, with the rest being available for data that is being transmitted over the link.
- The first 2 bytes of the frame contain a special bit pattern, and it is these bytes that enable the receiver to
  the receiver looks for the special bit pattern consistently, hoping to see it appearing once every 810 bytes, since each frame is 9 × 90 = 810 bytes long.
- When the special pattern turns up in the right place enough times, the receiver concludes that it is in sync and can then interpret the frame correctly.

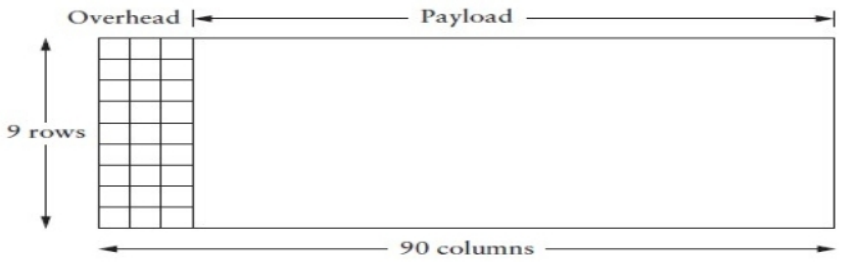


Fig : A SONET STS-1 Frame

For example, 64 Kbps of a SONET link's capacity is set aside for a voice channel that is used for maintenance.

The overhead bytes of a SONET frame are encoded using Non Return to Zero (NRZ), the simple encoding described in the previous section where 1s are high and 0s are low.

However, to ensure that there are plenty of transitions to allow the receiver to recover the sender's clock, the payload bytes are *scrambled*.

This is done by calculating the exclusive-OR (XOR) of the data to be transmitted and by the use of a well-known bit pattern. The bit pattern, which is 127 bits long, has plenty of transitions from 1 to 0, so that XORing it with the transmitted data is likely to yield a signal with enough transitions to enable clock recovery.

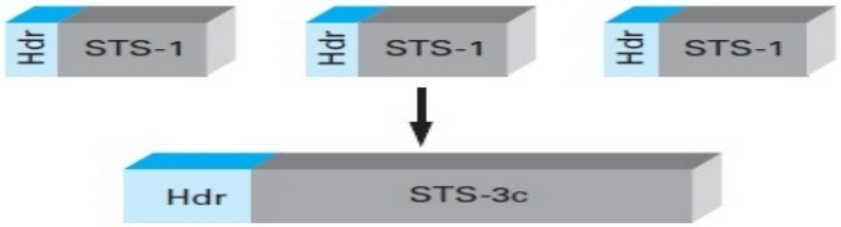


Fig : Three STS-1 Frames multiplexed onto one STS-3c frame.

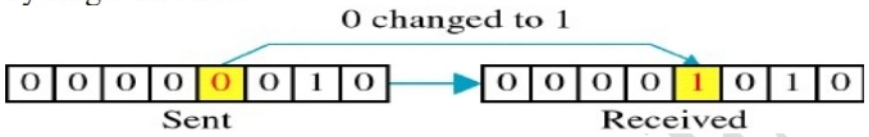
ERROR DETECTION AND CORRECTION

ERROR

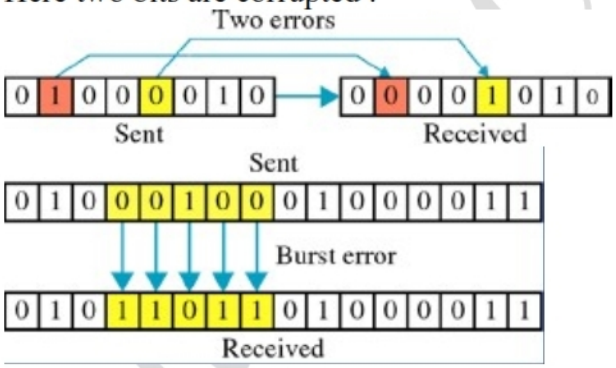
Data can be corrupted during transmission. Signals flows from one point to another. They are subjected to unpredictable interferences from heat, magnetism and other forms of electricity. For reliable communication, errors must be detected and corrected.

TYPES OF ERRORS

- **Single bit Error:**
  The term single bit error means that only one bit of a given data unit is changed from 1 to 0 or 0 to 1. 010101 is changed to 110101 here only one bit is changed by single bit error.



- **Burst Error:**
  A burst error means that 2 or more bits in the data unit have changed.
  Example:
  Here two bits are corrupted .



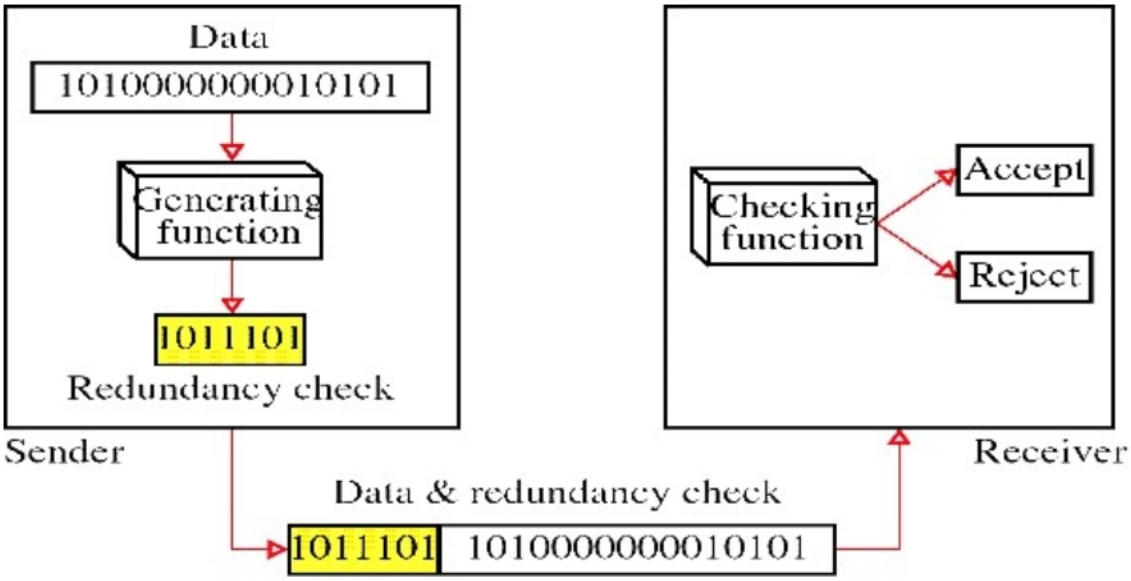
Errors can happen in the following ways,

- The bits in the frame can be inverted, anywhere within the frame including the data bits or the frame's control bits.
- Additional bits can be inserted into the frame, before the frame or after the frame and
- Bits can be deleted from the frame.

ERROR DETECTION

Redundancy

Error detection use the concept of redundancy, which means adding extra bits for detecting errors .i.e., instead of repeating the entire data stream, a shorter group of bits may be appended to the end of each unit.



Error Detection methods

1. **Parity check**
   - *Simple parity check (Vertical Redundancy Check)*
   - Two dimensional Parity Check (*Longitudinal Redundancy Check)*
2. **Cyclic redundancy check**
3. **Checksum**

Parity check

A redundant bit called parity bit, is added to every data unit so that the total number of 1's in the unit becomes even (or odd).

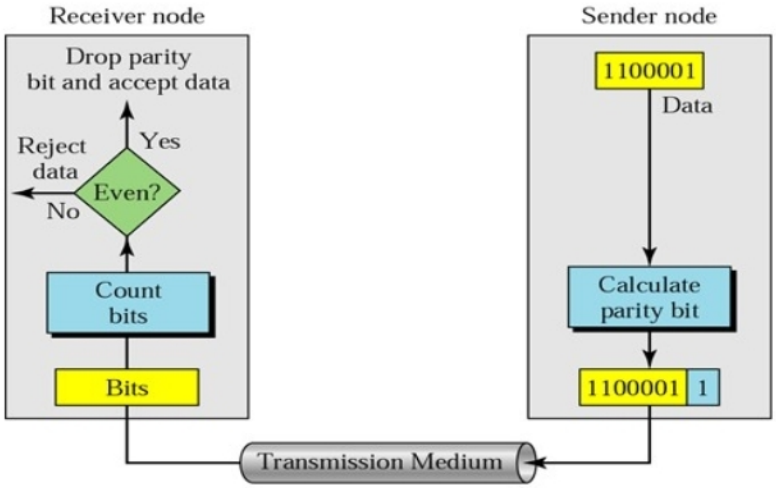
Simple parity check (Vertical Redundancy Check)

Sender Side:

In a simple parity check a redundant bit is added to a stream of data so that total number of 1's in the data become even or odd.

Receiver Side:

The total data bit+ Redundant bit is then passed through parity checking function. For even parity, it checks for even number of 1's and for odd parity it checks even number of 1's. If the check fails, it implies error is detected the data is rejected otherwise accepted.

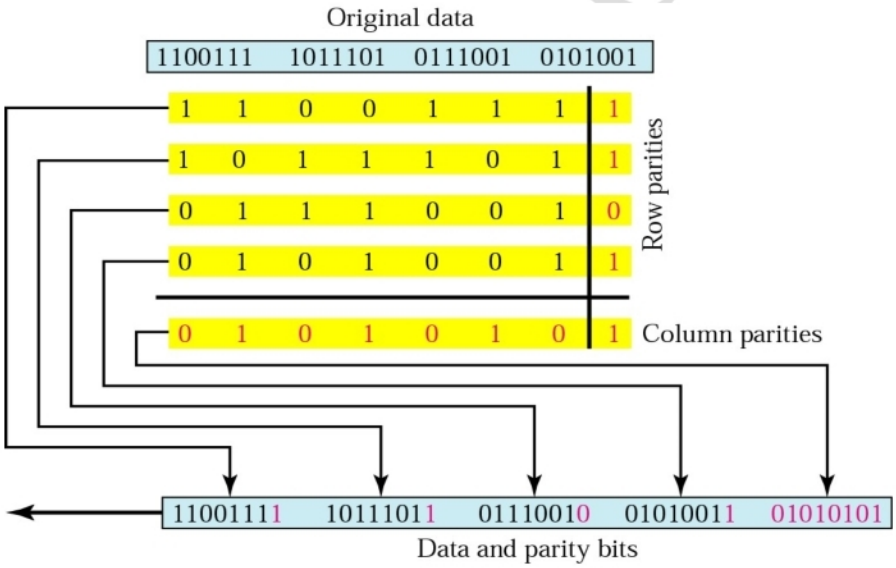


Note :

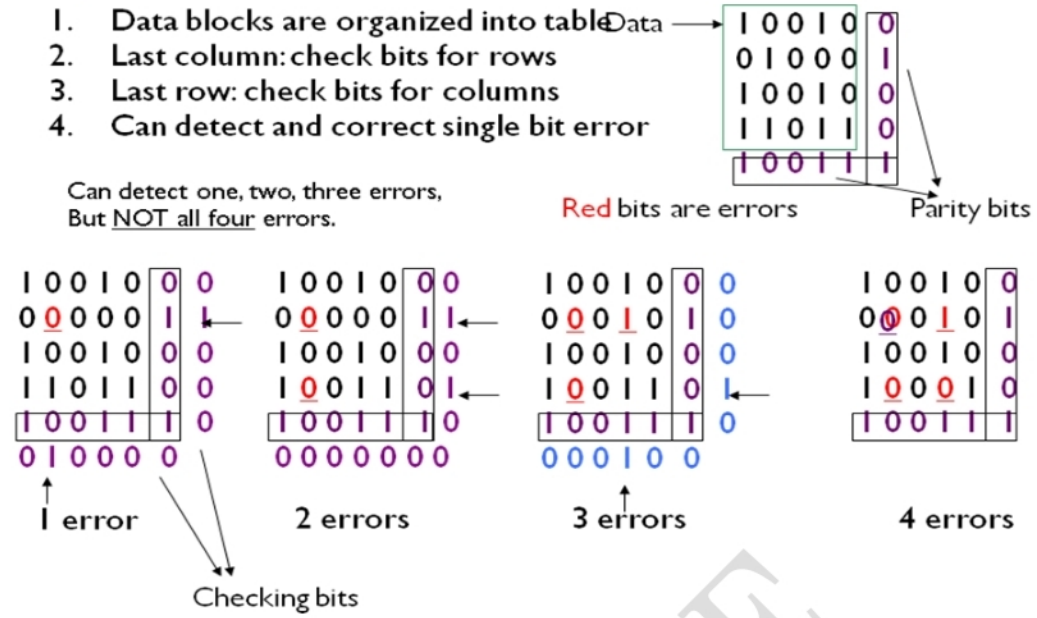
- *Simple parity check can detect all single-bit errors. It can detect burst errors only if the* total number *of bits changed in each data unit is odd.*

Two dimensional Parity Check (*Longitudinal Redundancy Check)*

In two-dimensional parity check, a block of bits is divided into rows to form a 2-dimensional array; add single parity check bits to each row and each column; transmit row-by-row.



Example 2



CYCLIC REDUNDANCY CHECK (CRC)

CRC is based on binary division. In CRC, a sequence of redundant bits, called the CRC or the CRC remainder is generated by performing binary division (using a predetermined divisor) and it is appended to the end of the data unit at the sender side.

At the receiver side, if the resulting unit (data + CRC) becomes exactly divisible by the same predetermined binary number (divisor) , the data is accepted. If a remainder results it indicates that the data unit has been damaged in transit and therefore must be rejected.

STEP BY STEP PROCEDURE

- First a starting of n 0's is appended to the data unit. The number n is one less than the number of bits in the predetermined divisor, which is n+ 1 bit.
- The newly elongated data unit is divided by the divisor, using a process called binary division. The remainder resulting from this division is the CRC.
- The CRC of n bits derived in step 2 replaces the appended 0s at the end of the data unit and the data is transmitted.(Note: CRC can also be all 0's)
- The data unit arrives at the receiver data first, followed by the CRC. The receiver treats the whole stream as unit and divides it by the same divisor that was used to find the CRC remainder.
- If the string arrives without error, the CRC checker yields a remainder of zero and the data unit passes. If the stream has been changed in transit, the division yields a non zero remainder and the data does not pass.

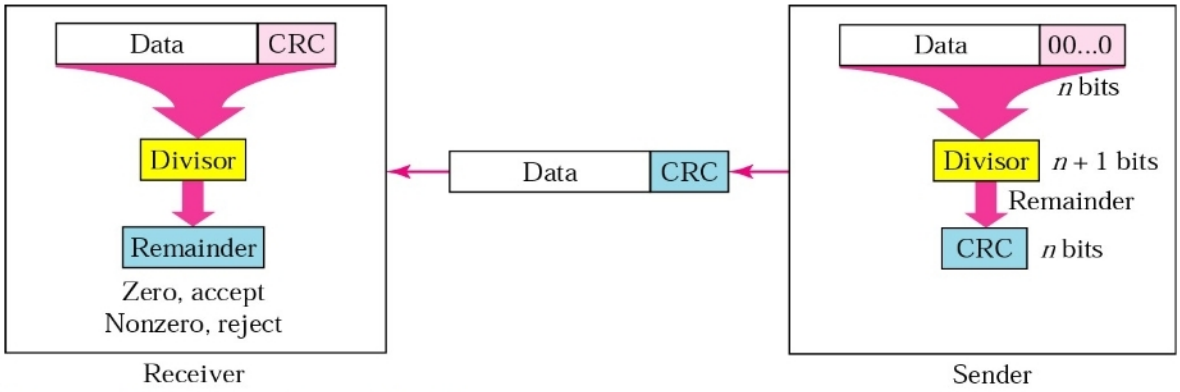


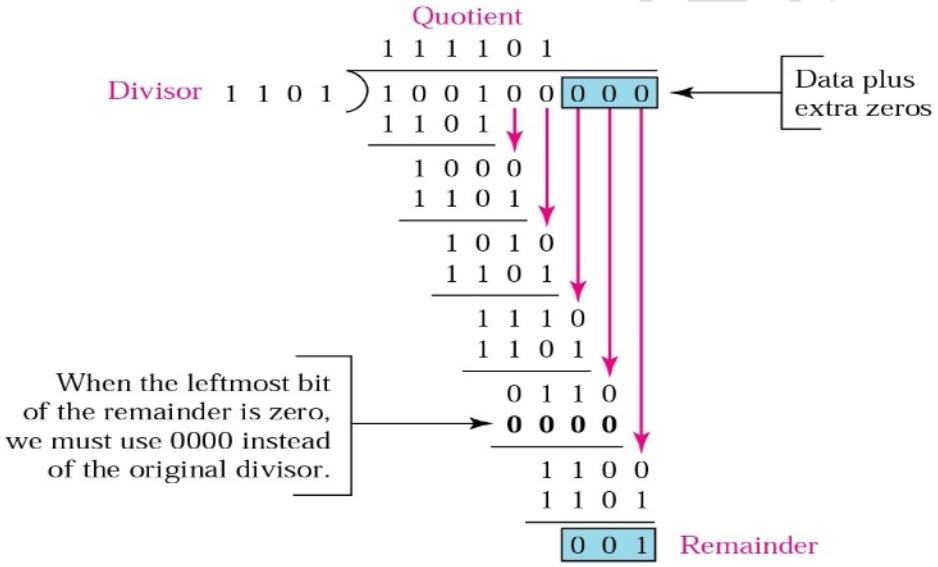
Figure : *CRC generator and checker*

CRC GENERATOR AND CHECKER

Example: Data is 100100 and Divisor is 1101

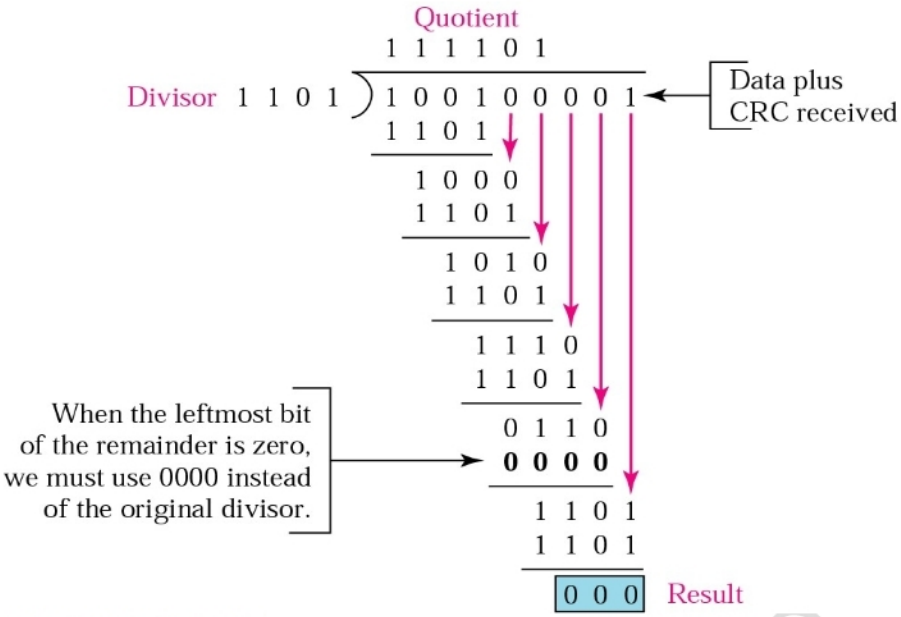
CRC GENERATOR

It uses modulo-2 division. The following figure shows this process.



CRC CHECKER

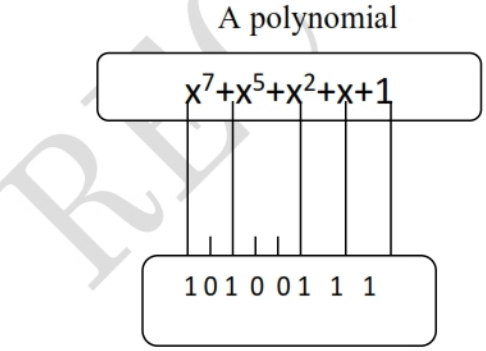
- A CRC checker function is exactly as the generator does. After receiving the data appended with the CRC, it does the same modulo-2 division.
- If the remainder is all 0s, the CRC is dropped and the data are accepted; otherwise, the received stream of bits is discarded and data are resent.
- The following figure shows the process of division in the receiver.



POLYNOMIALS

The divisor in the CRC most often represented not as a string of 1s and 0s, but as an algebraic polynomial. The polynomial format is useful to solve the concept mathematically.

$$x^7+x^5+x^2+x+1$$



The polynomial generator should have following properties:

1. It should have at least two terms.
2. The coefficient of the term x^0 should be 1.
3. It should not be divisible by x.
4. It should be divisible by x+ 1.

There are several different standard polynomials used by popular protocols for CRC generation.

Name	Polynomial	Application
CRC-8	$x^8 + x^2 + x + 1$	ATM header
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^2 + 1$	ATM AAL
CRC-16	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	LANs

CRC Performance

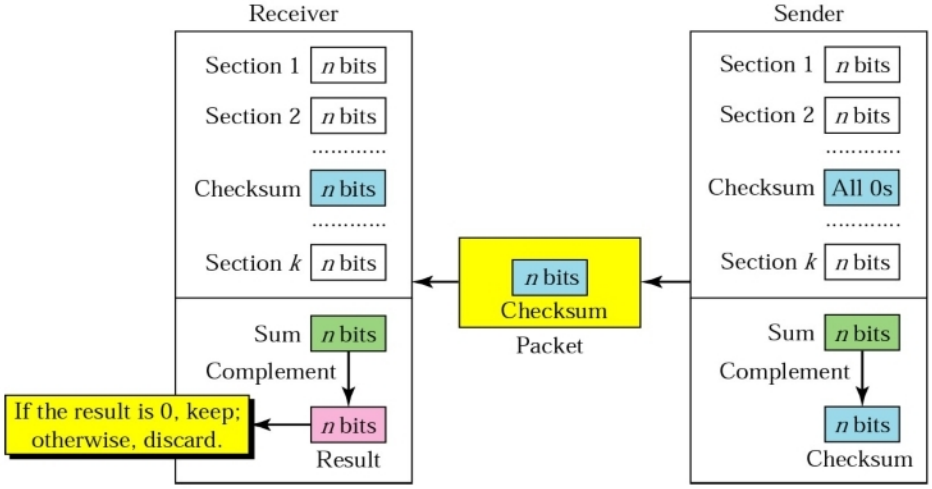
- *All* burst errors with length **less than** or **equal** to the degree of the polynomial (r)
- CRC can detect ***all*** burst errors that affect an **odd number** of bits
- CRC can detect ***all*** burst errors of length **equal** to the degree of the polynomial+1 **(r+1)** with probability $1 - (1/2)^{r-1}$.
- CRC can detect burst errors of length **greater** than the degree of the polynomial+1 **(r+1)** with probability $1 - (1/2)^r$
  (where r is the degree of the CRC polynomial)

CHECKSUM

The checksum is based on the redundancy.

CHECKSUM GENERATOR (sender side)

1. The message is divided into 16-bit words.
2. The value of the checksum word is set to 0.
3. All words including the checksum are added using one's complement addition.
4. The sum is complemented and becomes the checksum (redundancy bits).
5. The checksum is sent with the data across the network



CHECKSUM CHECKER (receiver side)

1. The message (including checksum) is divided into 16-bit words.
2. All words are added using one's complement addition.
3. The sum is complemented and becomes the new checksum.
4. If the value of checksum is 0, the message is accepted; otherwise, it is rejected.

EXAMPLE

Suppose the block of 16 bits is to be sent using a checksum of 8 bits.

10101001 00111001

the numbers are added using one's complement arithmetic

```
1 0 1 0 1 0 0 1     block 1
0 0 1 1 1 0 0 1     block 2
-----------------
1 1 1 0 0 0 1 0
0 0 0 0 0 0 0 0     initial checksum
-----------------
1 1 1 0 0 0 1 0     sum

0 0 0 1 1 1 0 1     Checksum ( 1's complement value of sum )
```

The data sent it is 10101001 00111001 00011101

Now the receiver receives the pattern with no error

10101001 00111001 00011101

the receiver adds these three sections, it will get all ones, which, after complementing, is all 0s and shows that there is no error.

```
1 0 1 0 1 0 0 1
0 0 1 1 1 0 0 1
0 0 0 1 1 1 0 1
---------------
1 1 1 1 1 1 1 1     sum

0 0 0 0 0 0 0 0     complement (DATA is correct )
```

suppose there is a burst error of length 5 that affects four bits.

10101***11***1 ***11***111001 00011101

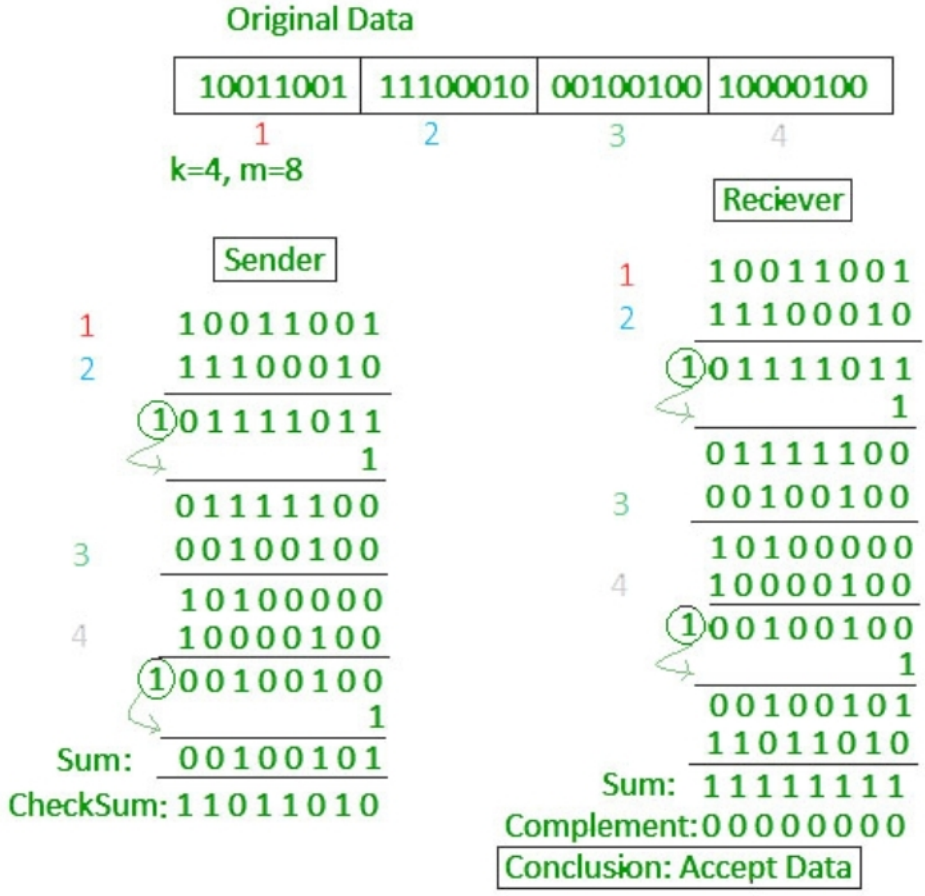
when the receiver adds these sections, it gets

```
1 0 1 0 1 1 1 1
  1 1 1 1 1 0 0 1
---------------
(1)1 0 1 0 1 0 0 0
             1 (carry 1)
---------------
1 0 1 0 1 0 0  1
  0 0 0 1 1 1 0 1
---------------
  1 1 0 0 0 1 1 1  (carry 1)
```

1 1 0 0 0 1 0 1 sum
0 0 1 1 1 0 0 0 complement

As the complement is non zero we assume that (the data is errored)

Example 2:



PERFORMANCE

- It detects all errors involving an odd number of bits as well as most errors involving an even number of bits.
- If one or more bits of a segment are damaged and the corresponding bit or bits of opposite value in the second segment are also damaged, the sum of those columns will not change and the receiver will not detect the problem.

Error correction

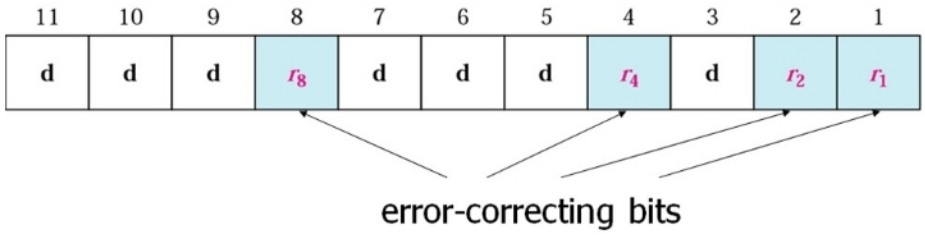
- Two methods
  1. Retransmission after detecting error
     -retransmit the data if error is detected
  2. Forward error correction (FEC)
     -hamming code is used to correct the error

HAMMING CODE

- ✓ A ***minimum number of redundancy bits*** needed to correct any single bit error in the data
- ✓ A minimum of 4 redundancy bits is needed if the number of data bits is 4.
- ✓ Redundancy bits in the Hamming code are placed in the codeword bit positions that are a power of 2
- ✓ Each redundancy bit is the parity bit for a different combination of data bits
- ✓ Each data bit may be included in more than one parity check.
- ✓ But the r bits are also transmitted along with data; hence
- ✓ $2^r \geq k+r+1$

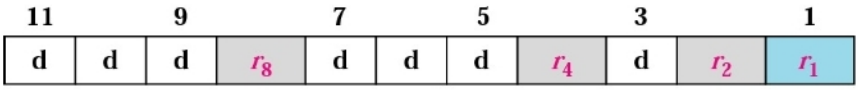
Number of Redundant Bits

Number of data bits *k*	Number of redundancy bits *r*	Total bits *k + r*
1	2	3
2	3	5
3	3	6
4	3	7
5	4	9
6	4	10
7	4	11

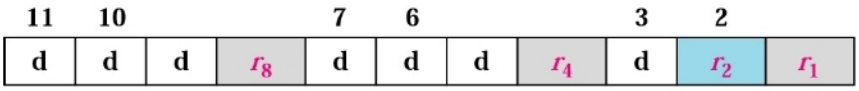


Redundant Bit Calculation

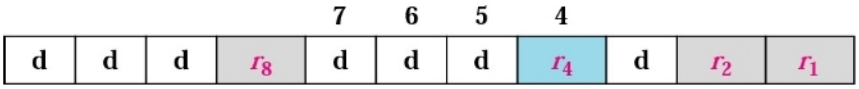
r_1 will take care of these bits.



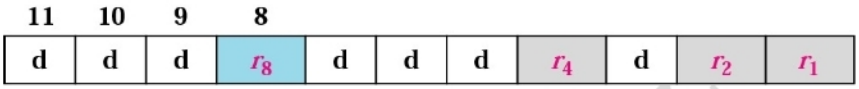
r_2 will take care of these bits.



r_4 will take care of these bits.



r_8 will take care of these bits.



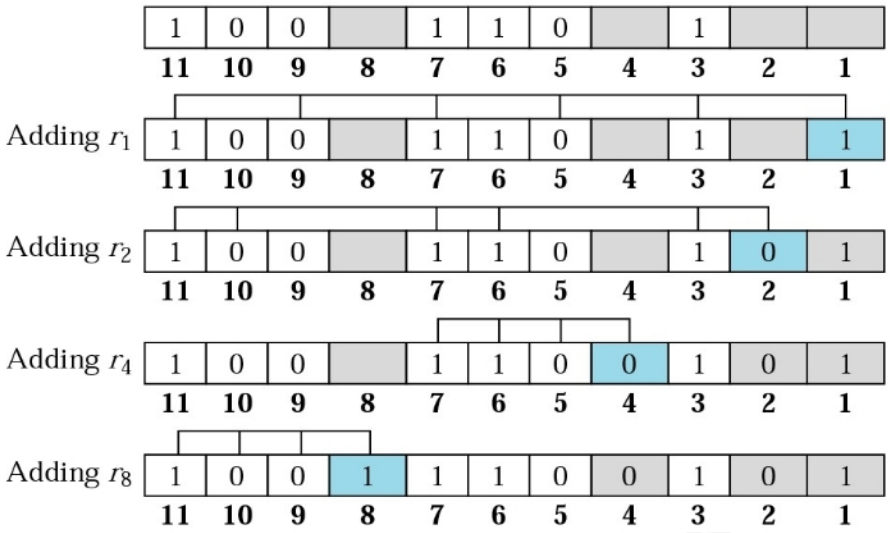
For r_1,
The bits considered are -1,3,5,7,9,11(to decide on these bits, follow this trick, since r_1 bit, start from 1st bit, take one and leave one and so on)

For r_2,
The bits considered are -2,3,6,7,10,11(to decide on these bits, follow this trick, since r_2 bit, start from 2nd bit, take two and leave two and so on)

For r_4,
The bits considered are -4,5,6,7(to decide on these bits, follow this trick, since r_4 bit, start from 4th bit, take four and leave four and so on)

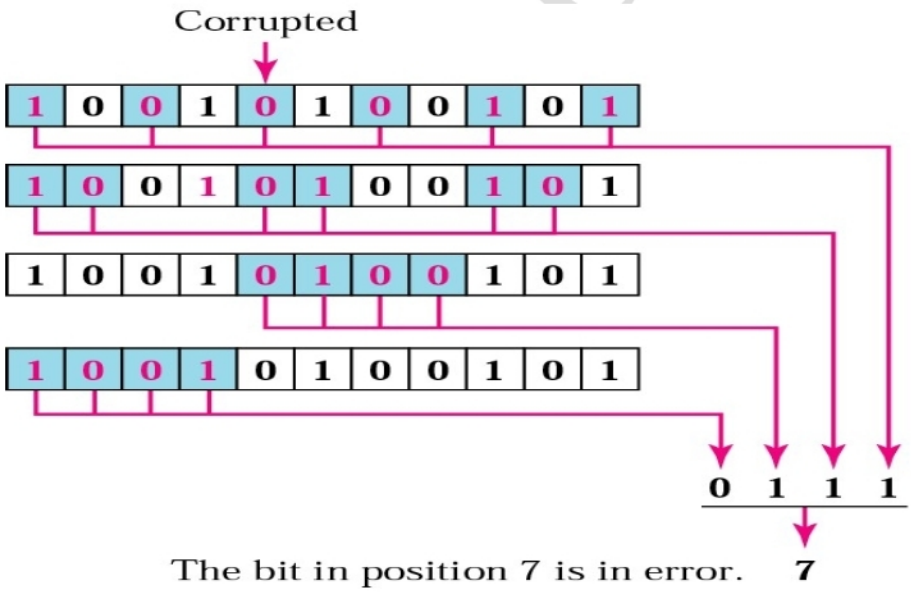
For r_8,
The bits considered are -8,9,10,11(to decide on these bits, follow this trick, since r_8 bit, start from 8th bit, take eight and leave eight and so on)

Example: ***Hamming Code***



Example: ***Correcting Error***

- **Receiver receives 10010100101**



FLOW CONTROL (RELIABLE TRANSMISSION)

Flow control is a technique used for the following reasons,

- If sender sends frames faster than recipient processes, then buffer overflow occurs at the receiver side.
- Receiver needs some time to process incoming frames.

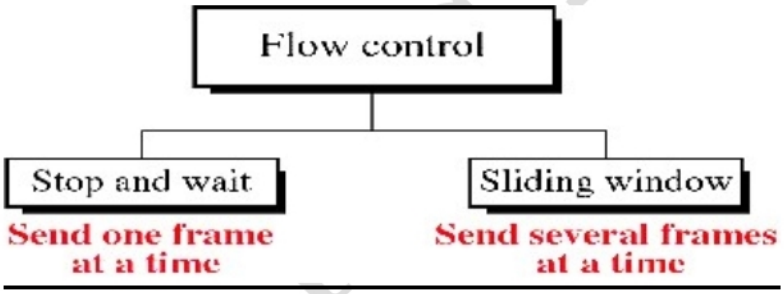
Flow control coordinates the amount of data that can be sent before receiving acknowledgement from the receiver. It is one of the most important duties of the data link layer.

- ***Flow control Refers to a set of procedures used to restrict the amount of data that the sender can send before waiting for acknowledgment from the receiver.***

ACK

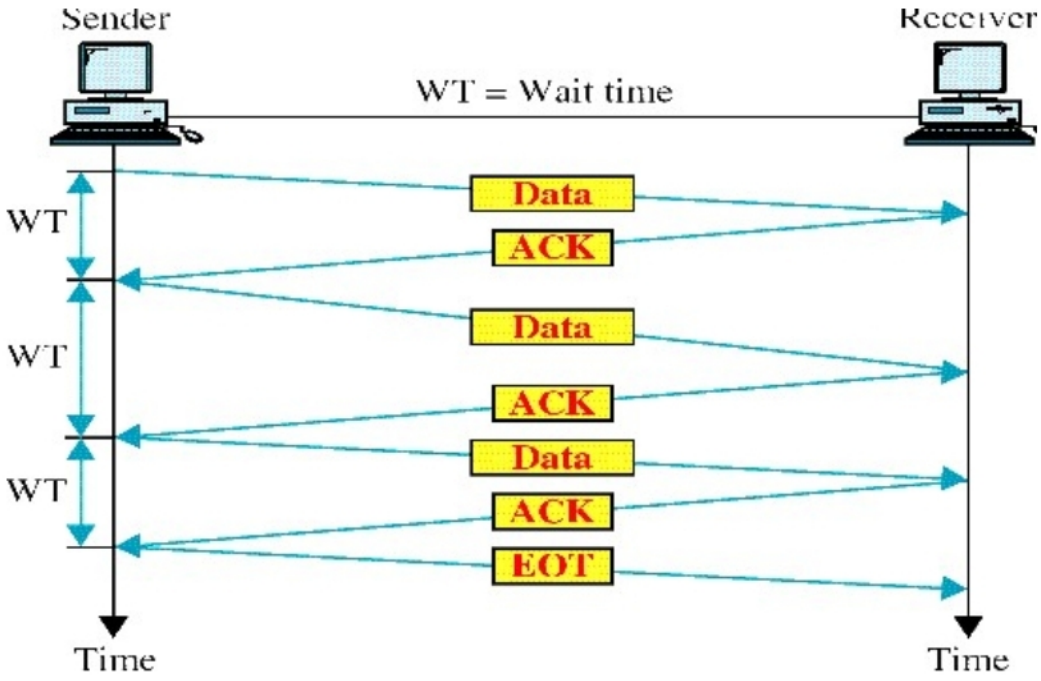
- An *acknowledgement* (ACK for short) is a small control frame that a protocol sends back to its peer saying that it has received the earlier frame.
  - A control frame is a frame with header only (no data).
  - The receipt of an *acknowledgement* indicates to the sender of the original frame that its frame was successfully delivered.

MECHANISMS



1. Stop and Wait Protocol

- Idea of stop-and-wait protocol is straightforward.
- After transmitting one frame, the sender waits for an acknowledgement before transmitting the next frame.
- If the acknowledgement does not arrive after a certain period of time, the sender times out and retransmits the original frame.



Timeline showing 4 different scenarios for the stop-and-wait algorithm

a) The ACK is received before the timer expires;
b) the original frame is lost;
c) the ACK is lost;
d) the timeout fires too soon

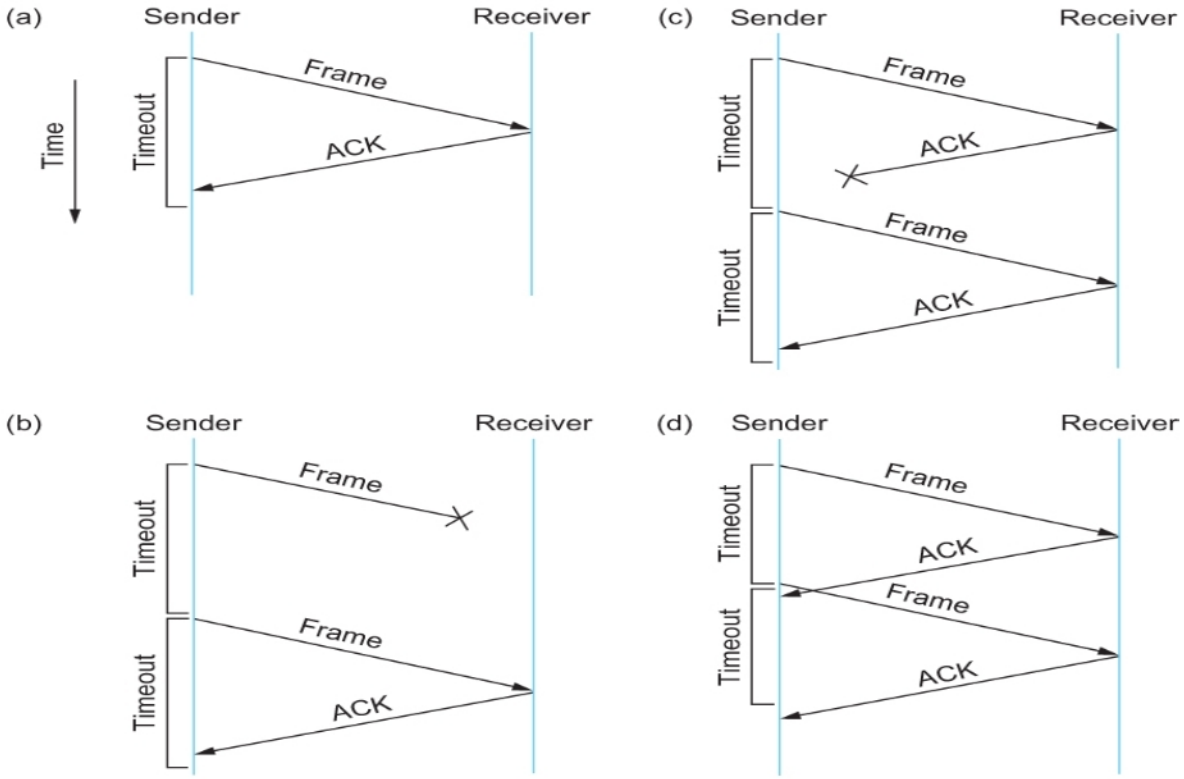
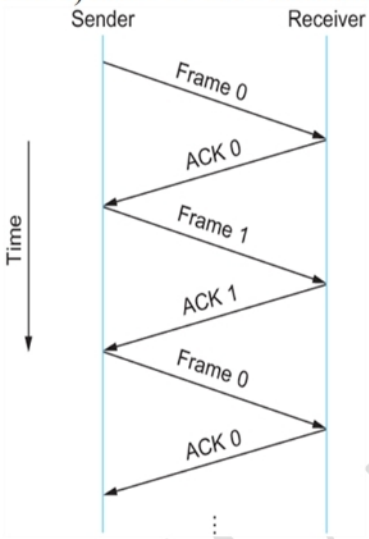


Fig: illustrates four different scenarios that result from the basic algorithm. The sending side is represented on the left, the receiving side is depicted on the right, and time flows from top to bottom.

- ✓ In Fig (a) ACK is received before the timer expires, (b) and (c) show the situation in which the original frame and the ACK, respectively, are lost, and (d) shows the situation in which the timeout fires too soon..

- ✓ Suppose the sender sends a frame and the receiver acknowledges it, but the acknowledgment is either lost or delayed in arriving. This situation is in (c) and (d). In both cases, the sender times out and retransmit the original frame, but the receiver will think that it is the next frame, since it correctly received and acknowledged the first frame.

- ✓ This makes the receiver to receive the duplicate copies. To avoid this two sequence numbers (0 and 1) must be used alternatively.



Timeline for stop-and-wait with 1-bit sequence number

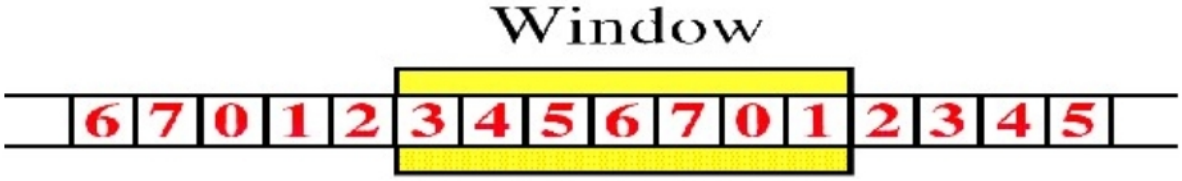
The main drawback of the stop-and-wait algorithm is that it allows the sender have only one outstanding frame on the link at a time.

2. Sliding Window

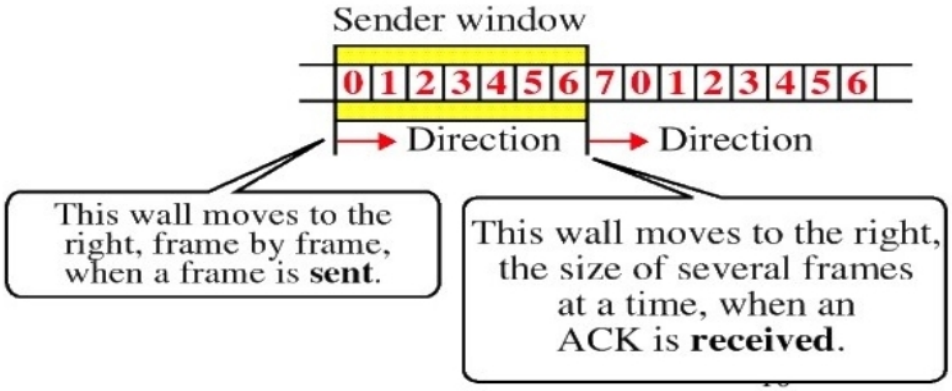
- ✓ The sender can transmit several frames before needing an acknowledgement.
- ✓ Frames can be sent one right after another meaning that the link can carry several frames at once and it s capacity can be used efficiently.
- ✓ The receiver acknowledges only some of the frames, using a single ACK to confirm the receipt of multiple data frames
- ✓ Sliding Window refers to imaginary boxes at both the sender and the receiver.
- ✓ Window can hold frames at either end and provides the upper limit on the number of frames that can be transmitted before requiring an acknowledgement.
- ✓ Frames are numbered modulo-n which means they are numbered from 0 to n-1
- ✓ For eg. If n=8 the frames are numbered 0,1,2,3,4,5,6,7. i.e the size of the window is n -1.
- ✓ When the receiver sends ACK it includes the number of the next frame it expects to receive.
  - ✓ When the sender sees an ACK with the number 5, it knows that all frames up

through number 4 have been received.

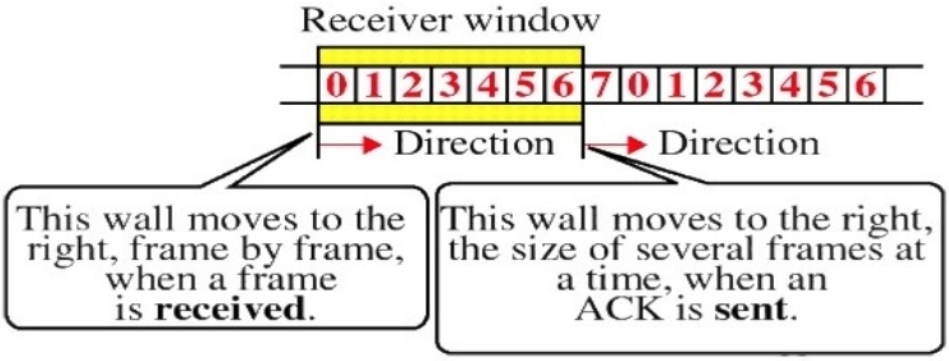
Sliding Window

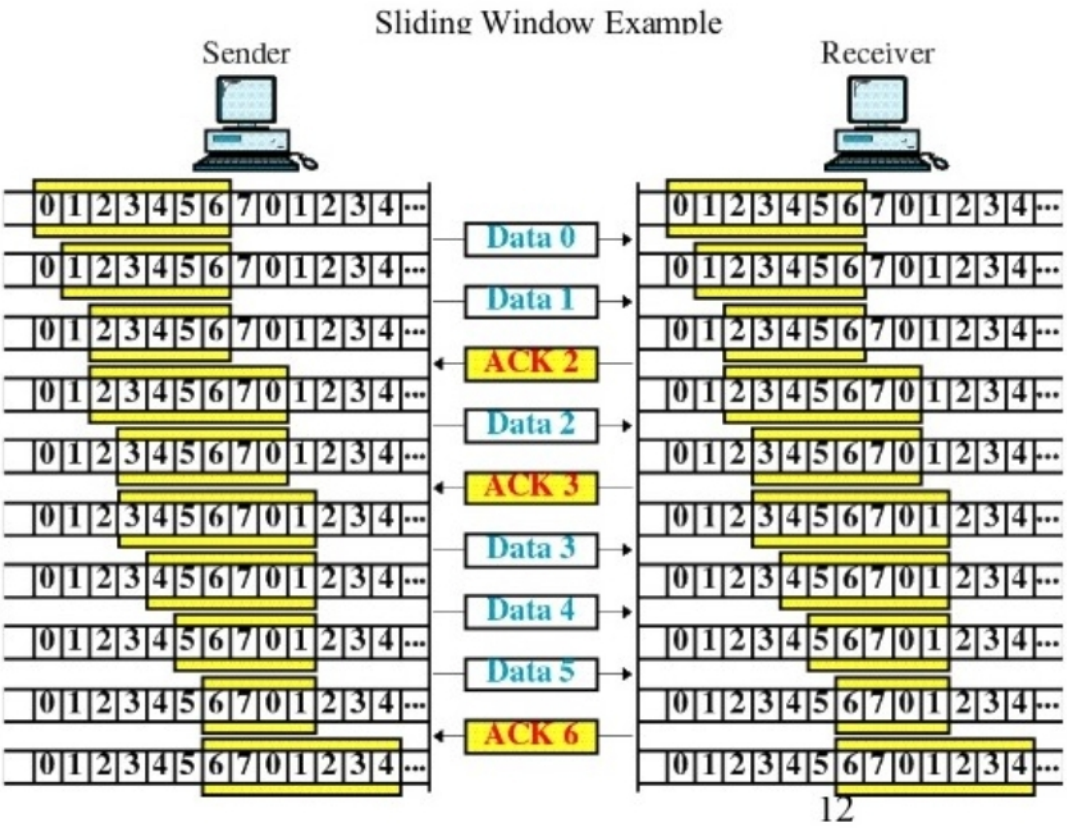


Sender Sliding Window



Receiver Sliding Window




Sliding Window Example
Sender
Receiver
Data 0
Data 1
ACK 2
Data 2
ACK 3
Data 3
Data 4
Data 5
ACK 6

12

UNIT-II MEDIA ACCESS AND INTERNETWORKING

Media Access Protocols – ALOHA - CSMA/CA/CD –Ethernet – Wireless LANs - 802.11- Bluetooth - Switching and Forwarding - Bridges and LAN Switches – Basic Internetworking- IP Service Model – IP fragmentation - Global Addresses – ARP - DHCP – ICMP- Virtual Networks and Tunnels.

Media access control

The data link layer is divided into two sub layers,

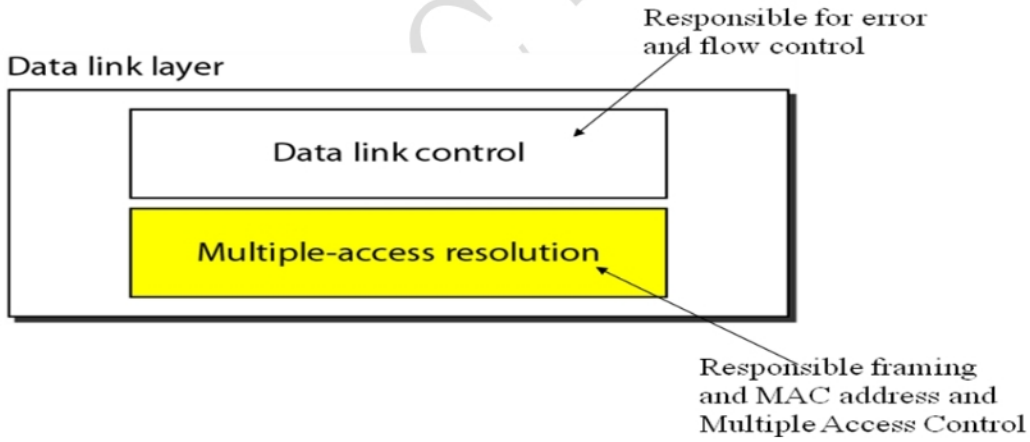
1. Logical link Control Layer
2. Medium Access Control Layer

Logical Link Control Layer:

The LCL layer is concerned with managing traffic (flow and error control) over the physical medium and may also assign sequence numbers to frames and track acknowledgements.

Medium Access Control Layer:

Media Access Control layer is one of two sublayers of the Data Link Control layer and is concerned with sharing the physical connection to the network among several computers.



Medium Access Control:

- **Problem**: When two or more nodes transmit at the same time, their frames will collide and the link bandwidth is **wasted** during collision
  How to coordinate the access of multiple sending/receiving nodes to the shared link**???**

- **Solution**: We need a **protocol** to coordinate the transmission of the active nodes

- These protocols are called **Medium or Multiple Access Control (MAC) Protocols** belong to a **sublayer** of the data link layer called **MAC** (Medium Access Control)

- What is expected from Medium Access Control Protocols:

    1. Main task is to **minimize collisions** in order to **utilize the bandwidth** by:
    2. Determining **when** a station can use the link (medium)
    3. **what** a station should do when the link is **busy**
    4. **what** the station should do when it is involved in **collision**
    5. To avoid the multiple access problem is used in which the station is forced to sense the medium before transmitting. This is called as Carrier Sense Multiple Access (CSMA).

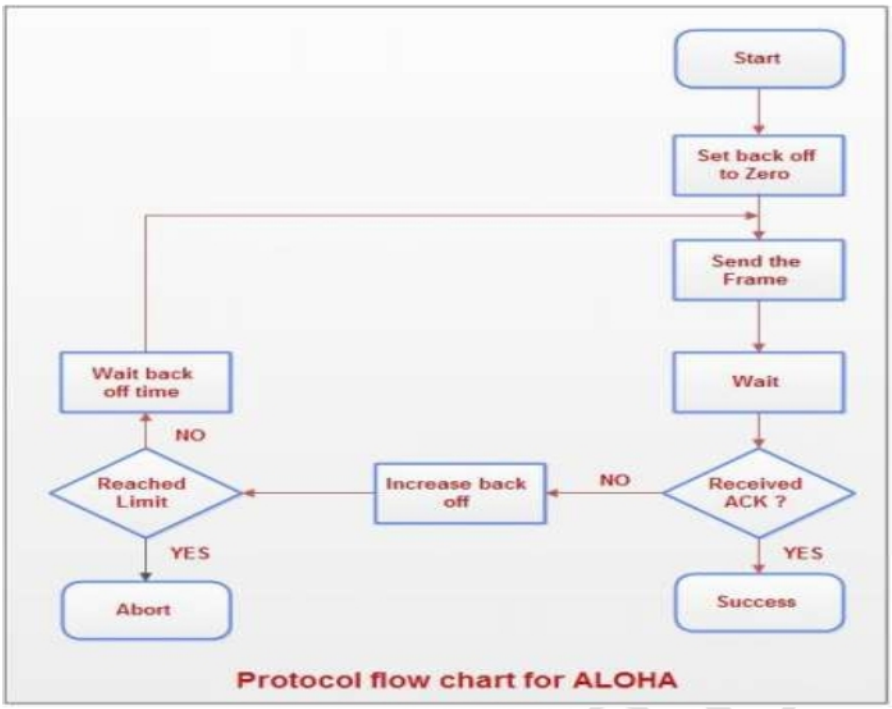
The two commonly used medium access protocols are

1. ALOHA
2. CSMA
3. CSMA/Collision Detection (CSMA/CD)
4. CSMA/Collision Avoidance(CSMA/CA)

ALOHA

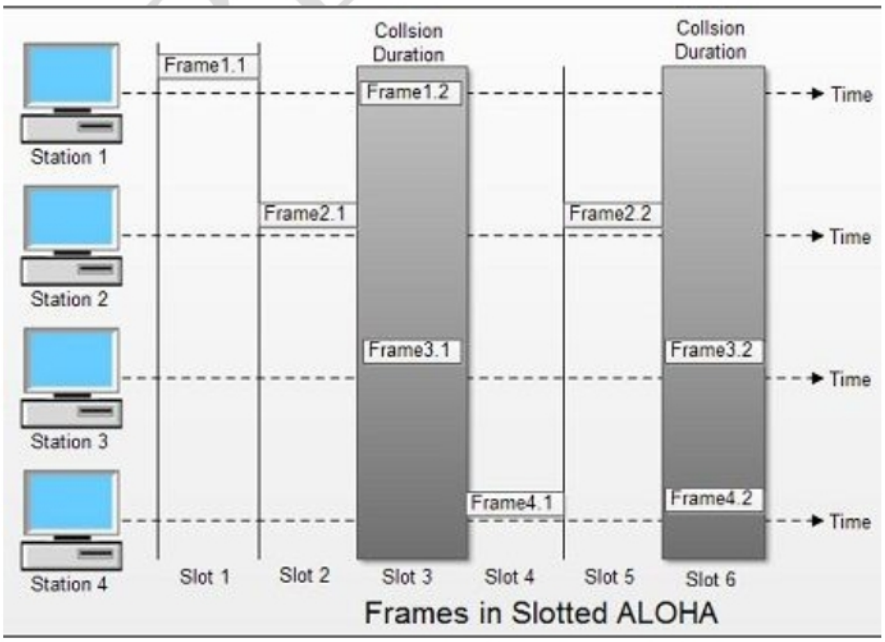
PURE ALOHA

- Each source (transmitter) in a network sends data whenever there is a frame to send.
- If the frame successfully reaches the destination (receiver), the next frame is sent.
- When two or more stations transmit simultaneously, there is collision and the frames are destroyed.
- If acknowledgement is not received within specified time, the station assumes that the frame (or acknowledgement) has been destroyed.
- If the frame is destroyed because of collision the station waits for a random amount of time and sends it again. This waiting time must be random otherwise same frames will collide again and again.
- Therefore, pure ALOHA dictates that when time-out period passes, each station must wait for a random amount of time before resending its frame. This randomness will help avoid more collisions.



SLOTTED ALOHA

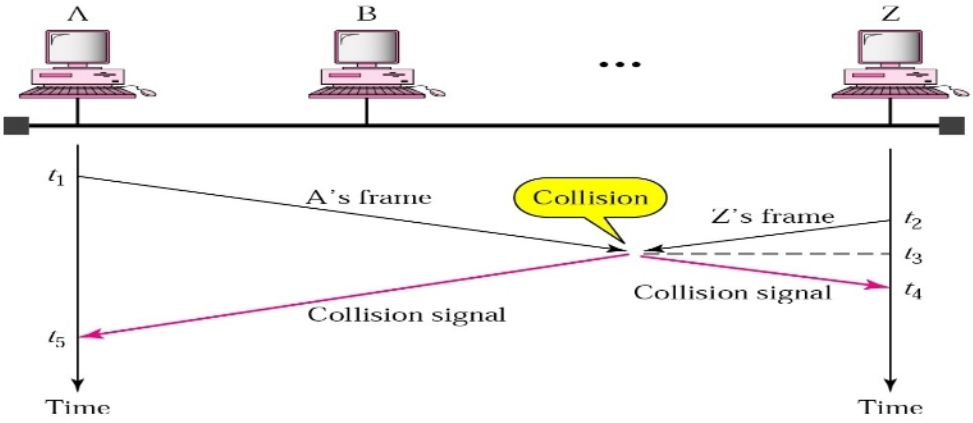
- The time of the shared channel is divided into discrete intervals called slots
- The stations can send a frame only at the beginning of the slot and only one frame is sent in each slot.
- In slotted ALOHA, if any station is not able to place the frame onto the channel at the beginning of the slot *i.e.* it misses the time slot then the station has to wait until the beginning of the next time slot.
- In slotted ALOHA, there is still a possibility of collision if two stations try to send at the beginning of the same time slot



CSMA (Carrier Sense Multiple Access)

CSMA is based on the principle "Sense Before Transmit or Listen Before Talk. Each station must listen before transmitting.

- ➔ If a frame was sent by a station, All stations knows immediately so they can **wait before start sending**
  - ➔ A station with frames to be sent, should **sense the medium** for the presence of another transmission (carrier) before it starts its own transmission
- CSMA can **reduce** the possibility of collision but it cannot eliminate it.
  - Collision can only happen when more than one station begin transmitting within a short time (the **propagation time** period)

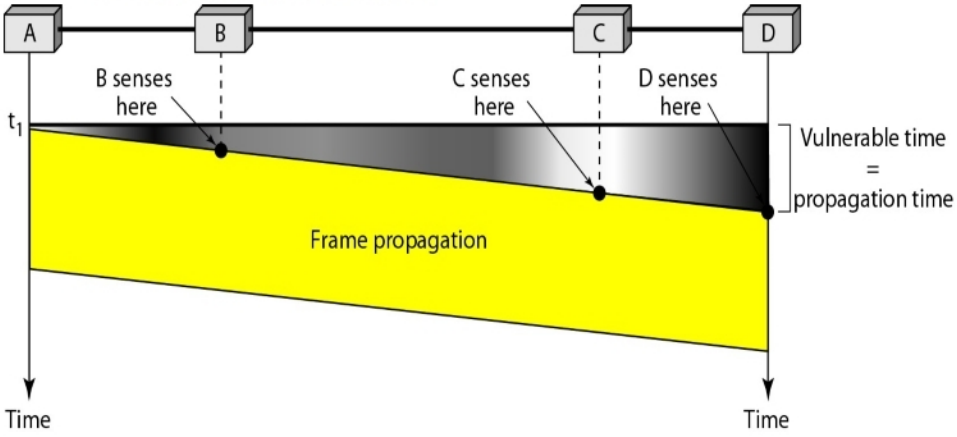


The possibility of collision still exists because of propagation delay; when a station sends a frame, it still takes time (although very short) for the first bit to reach every station and for every station to sense it. In other words, a station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.

Vulnerable time

Vulnerable time for CSMA is the **maximum propagation time**

- The longer the propagation delay, the worse the performance of the protocol because of the above case.



- When a station sends a frame, and any other station tries to send a frame during this time, a collision will result.
- But if the first bit of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending. Figure shows the worst case. The vulnerable time for CSMA is the propagation time tp .
- The leftmost station A sends a frame at time *t1* which reaches the rightmost station D at time *t1+ tp* . The gray area shows the vulnerable area in time and space.

Types of CSMA Protocols

Different CSMA protocols that determine:

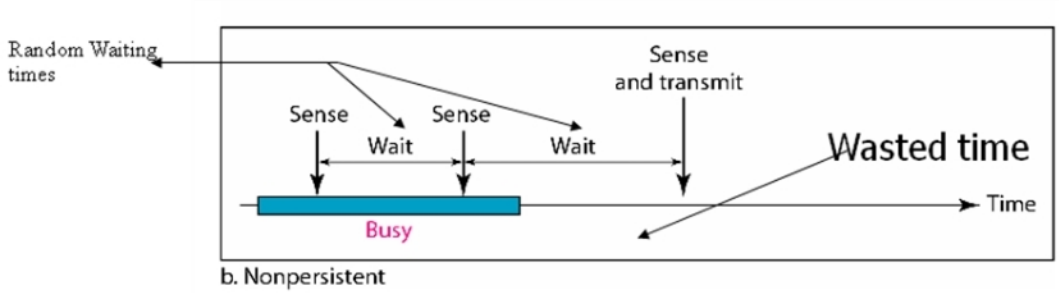
- What a station should do when the medium is **idle**?
- What a station should do when the medium is **busy**?

The channel can access the medium using any of three persistence methods,

1. Non-Persistent CSMA
2. 1-Persistent CSMA
3. p-Persistent CSMA

No persistent CSMA

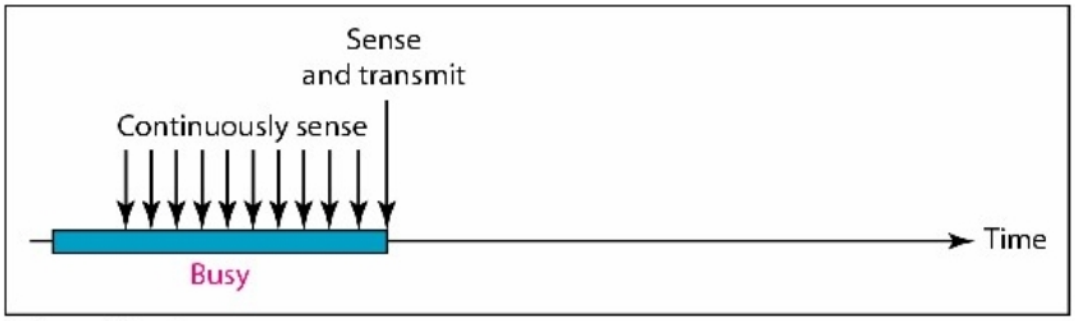
- A station with frames to be sent, should sense the medium
  1. If medium is idle, **transmit**; otherwise, go to 2
  2. If medium is busy, (**backoff**) wait a ***random*** **amount of time** and repeat **1**
- Non-persistent Stations are **deferential (respect others)**
- Performance:
  1. Random delays reduces probability of collisions because two stations with data to be transmitted will wait for different amount of times.
  2. Bandwidth is **wasted** if waiting time (backoff) is large because medium will remain idle following end of transmission even if one or more stations have frames to send



b. Nonpersistent

1-persistent CSMA

- To avoid idle channel time, 1-persistent protocol used
- Station wishing to transmit listens to the medium:
  1. If medium idle, **transmit** immediately;
  2. If medium busy, **continuously listen** until medium becomes idle; then transmit immediately with probability 1
- Performance
  - 1-persistent stations are **selfish**
  - If two or more stations becomes ready at the same time, **collision guaranteed**

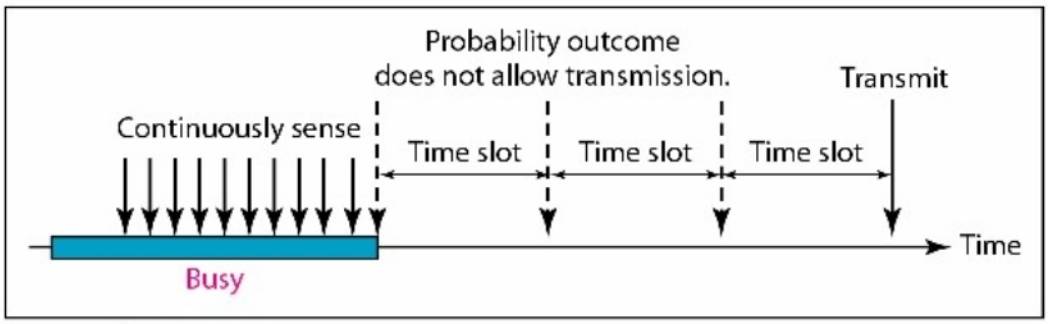


a. 1-persistent

P-persistent CSMA

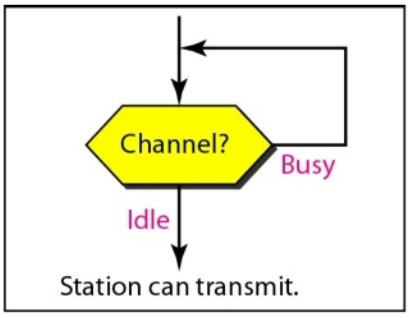
- Time is divided to slots where each Time unit (slot) typically equals **maximum propagation delay**
- Station wishing to transmit listens to the medium:

1. If medium idle,
   - transmit with probability (**p**), OR
   - wait **one time unit (slot)** with probability (**q= 1 – p**), then repeat 1.
2. If medium busy, **continuously listen until idle** and repeat step **1**
3. Performance
   - Reduces the possibility of collisions like **nonpersistent**
   - Reduces channel idle time like **1-persistent**

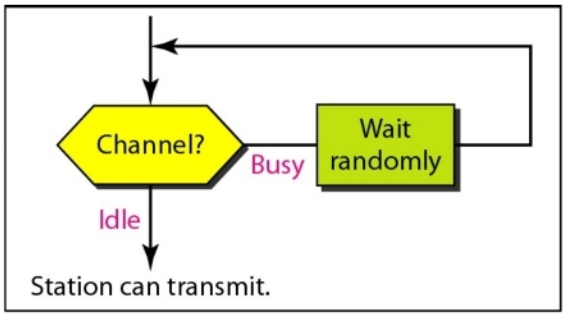


c. p-persistent

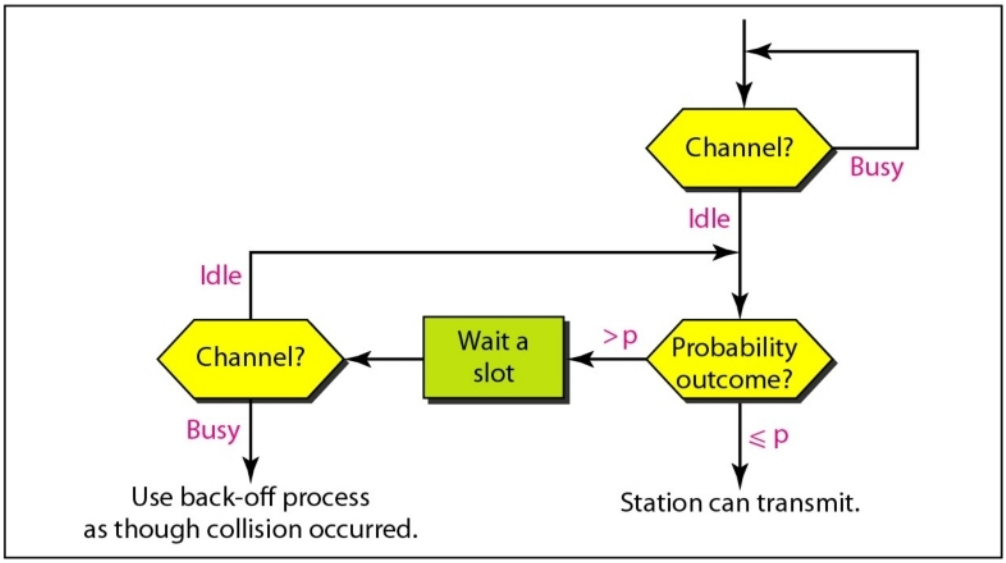
Flow diagram for three persistence methods



a. 1-persistent



b. Nonpersistent



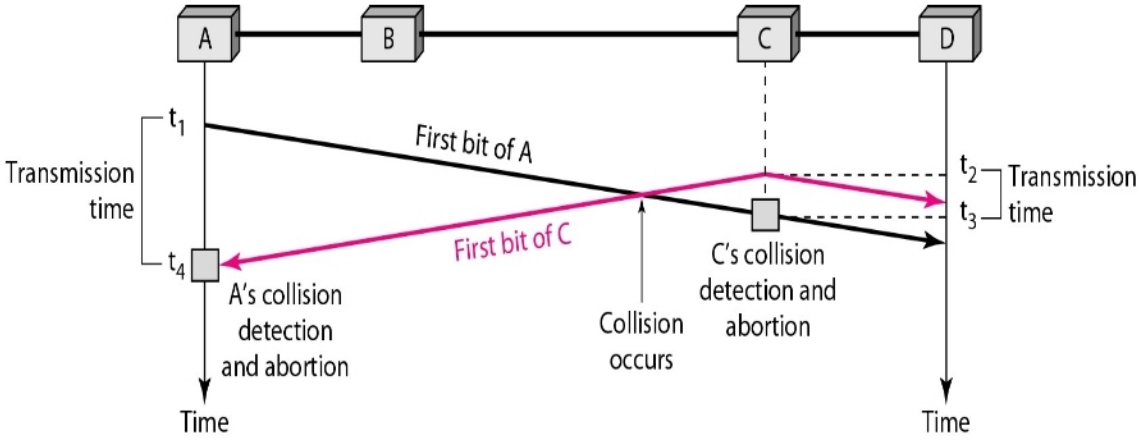
c. p-persistent

CSMA/CD (CSMA - Collision Detection)

- ***CSMA (all previous methods) has an inefficiency:***
  - If a collision has occurred, the channel is **unstable** until colliding packets have **been fully transmitted**
- ***CSMA/CD*** *(Carrier Sense Multiple Access with Collision Detection)* ***overcomes this as follows:***
  - While transmitting, the sender is **listening to medium** for collisions.
  - Sender **stops transmission** if collision has occurred **reducing channel wastage** .

CSMA/CD is Widely used for **bus topology LANs** (IEEE 802.3, **Ethernet).**

Collision of the first bit in CSMA/CD

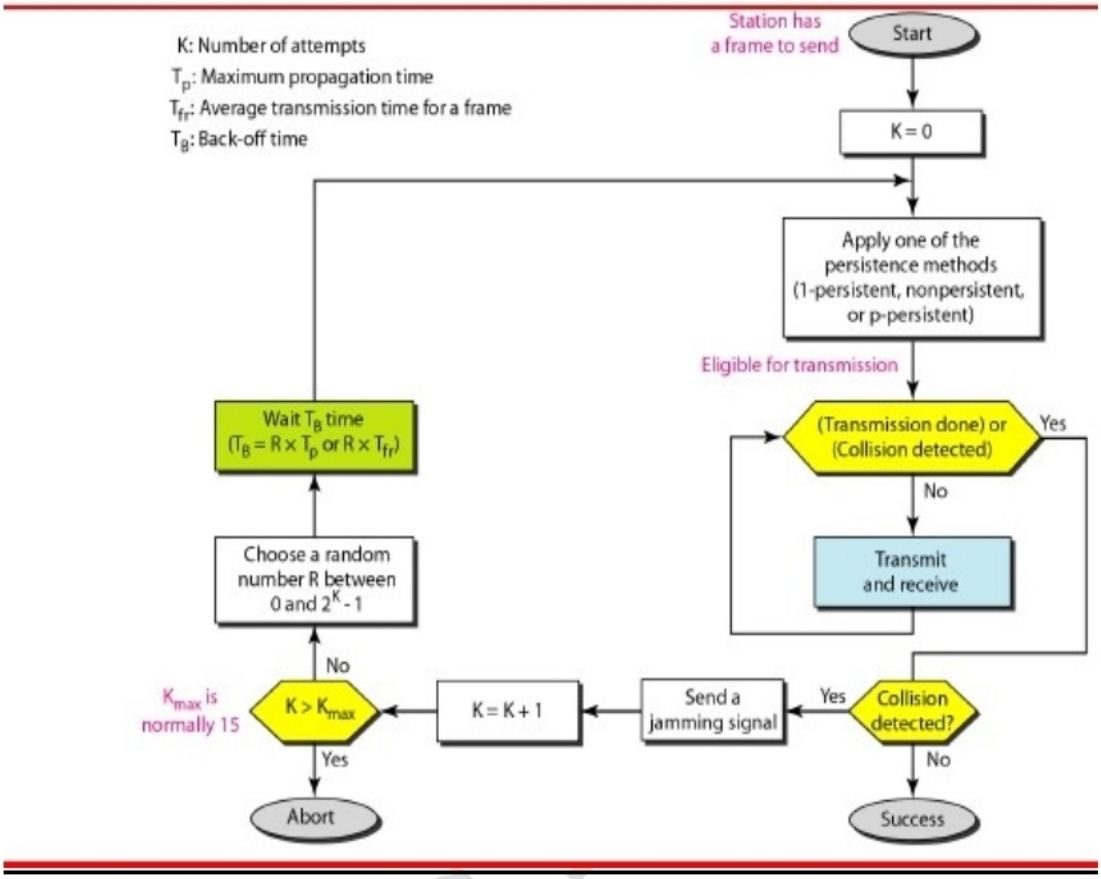


To better understand CSMA/CD, let us look at the first bits transmitted by the two stations involved in the collision. Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In Figure , stations A and C are involved in the collision.

At time *t* 1, station A has executed its persistence procedure and starts sending the bits of its frame. At time *t2,* station C has not yet sensed the first bit sent by A. Station C executes its persistence procedure and starts sending the bits in its frame, which propagate both to the left and to the right. The collision occurs sometime after time *t2'* Station C detects a collision at time *t3* when it receives the first bit ofA's frame. Station C immediately (or after a short time, but we assume immediately) aborts transmission. Station A detects collision at time *t4* when it receives the first bit ofC's frame; it also immediately aborts transmission.

Looking at the figure, we see that A transmits for the duration *t4 - tl;* C transmits for the duration *t3 - t2'* Later we show that, for the protocol to work, the length of any frame divided by the bit rate in this protocol must be more than either of these durations. At time *t4,* the transmission of*A:s* frame, though incomplete, is aborted; at time *t 3* , the transmission ofB's frame, though incomplete, is aborted.

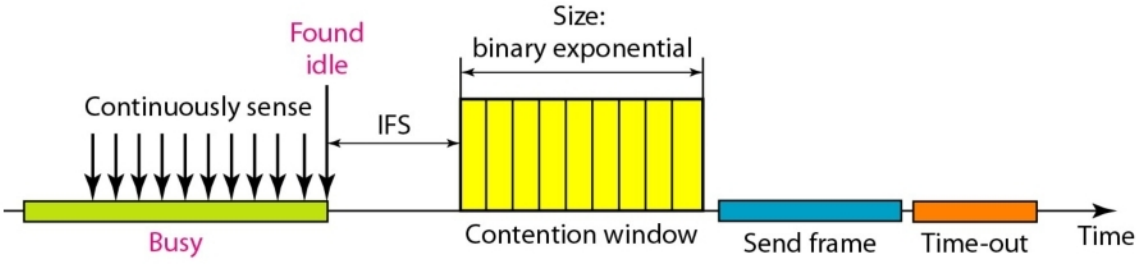
Figure 12.14 *Flow diagram for the CSMA/CD*



CSMA/CA (CSMA – Collision Avoidance)

- **C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **A**voidance
- Used in a network where collision cannot be detected
  - E.g., wireless LAN

Collisions are avoided in CSMA / CD using three strategies,
1. Inter Frame Space [IFS]
2. Contention window
3. Acknowledgements



Inter Frame Sequence [IFS]

- Collisions are avoided by transmitting if a channel is found idle
- If the channel is found idle, the station does not send immediately
- It waits for a period of time called interframe space (IFS)

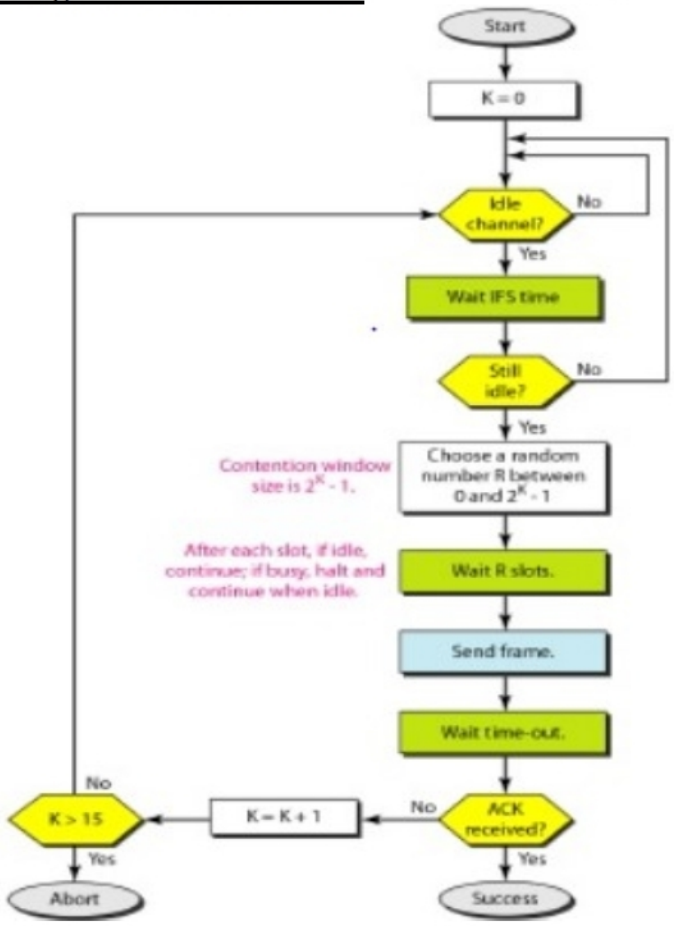
Contention window

- contention window is an amount of time divided into slots.
- The station that is ready to send chooses a random number of slots as its wait time.
- The number of slots in the window changes according to binary exponential back off ie) it is set to one slot the first time and then doubles each time the station cannot detect an idle channel.
- In CSMA/CA, if the station finds the channel busy, it does not restart the timer of the contention window; it stops the timer and restarts it when the channel becomes idle

Acknowledgements

- Collisions can be avoided by setting positive acknowledgements to assure that the receiver has received the frame or not.

Flow Diagram for CSMA / CA:



Ethernet (802.3)

Wired LAN

- Most successful local area networking technology of last 20 years.
- Ethernet is a multiple access network with a set of nodes that send and receive frames over a shared link.

Uses CSMA/CD technology

- Carrier Sense Multiple Access with Collision Detection.
- A set of nodes send and receive frames over a shared link.
- Carrier sense means that all nodes can distinguish between an idle and a busy link.
- Collision detection means that a node listens as it transmits and can therefore detect when a frame it is transmitting has collided with a frame transmitted by another node.

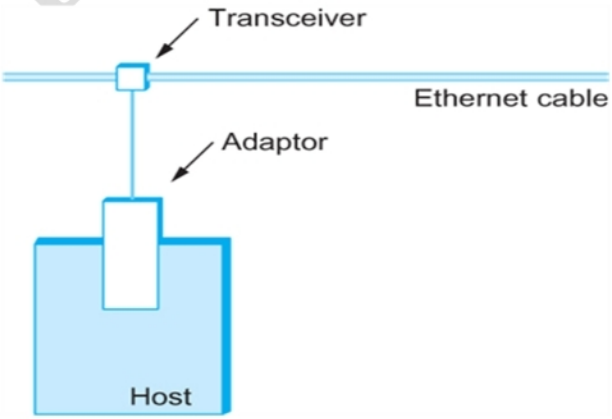
Ethernet can operate at A SPEED

- 100 Mbps (Fast Ethernet)
- 1000 Mbps (Gigabit Ethernet)

Ethernet also used in full duplex, point-to-point configuration

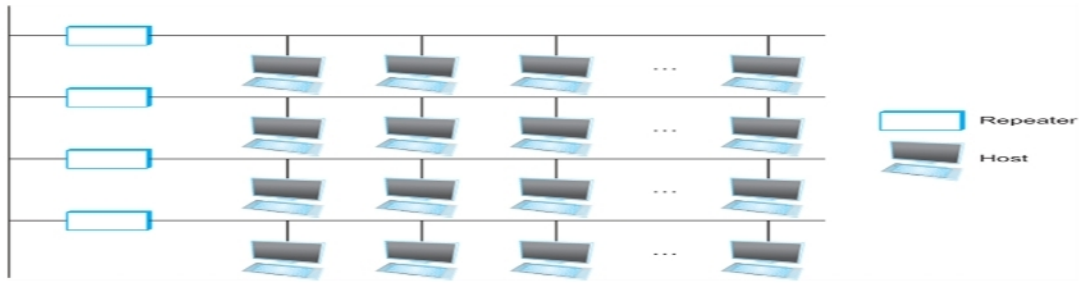
Physical Properties

- An Ethernet segment is implemented on a coaxial cable of up to 500 m.
- Hosts connect to an Ethernet segment by tapping into it.
- A **transceiver** (a small device directly attached to the tap) detects when the line is idle and drives signal when the host is transmitting.
- The transceiver also receives incoming signal.
- The transceiver is connected to an Ethernet adaptor which is plugged into the host.
- The protocol is implemented on the adaptor.



Ethernet transceiver and adaptor

- Multiple Ethernet segments Can be joined together by ***repeaters.***
- A *repeater* is a device that strengthens and forwards weakened digital signals.



Ethernet repeater

Any signal placed on the Ethernet by a host is broadcast over the entire network

- Signal is propagated in both directions.
- Repeaters forward the signal on all outgoing segments.
- Terminators attached to the end of each segment absorb the signal.

Table : Types of 10 Mbps Ethernets

Name	Cable	Max. segment	Nodes /segment	Signaling technique	Topology used	Cable diameter	Access Method	advantage
10Base5	Thick coax	500 m	100	Baseband (Manchester)	Bus	10	CSMA/CD	Good For back bones
10Base2	Thin coax	200 m	30	Baseband (Manchester)	Bus	5	CSMA/CD	Cheapest system
10Base-T	Twisted pair	100 m	1024	Based band (Manchester)	Star	0.4 to 0.6	CSMA/CD	Easy maintenance
10Base-F	Fiber optics	2000m	1024	Manchester/ on-off	Star	62.5/125 pm	CSMA/CD	Best between buildings

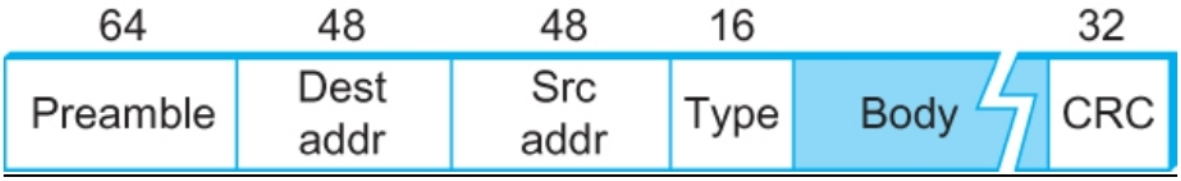
10Base5 cable. 10Base5 cable or **"Thick Ethernet"** or Thicknet is the cable which is the oldest in the category. it is called as thicknet because of the use of thick coaxial cable. The cable is marked after each 2.5 meters.

10Base2 Cable. 10Base2 cable also called **"Thin Ethernet" or Thinnet or cheapnet or cheapernet,** was designed after the thick Ethernet cable. This type of cable is usually thin, flexible and bends easily. It also make use of bus topology. It is also a coaxial cable that is having a smaller diameter than the 10Base5 cable.

10BaseT Cable. l0BaseT cable or **"Twisted Pair"** Cable is cheapest and easiest to maintain. This type of cabling is most popular among local area networks. It make use of unshielded twisted pair and provides maximum segment length of 100 m. It make use of start topology. In this type of network, every station is having a wired link to a central device, called "Hub".

l0BaseF Cable. 10BaseF cable or **"Fiber Optics Cable"** is the most efficient and fastest cable in the category of cables for 802 LANs. The fiber optic cable is very expensive as compared to above discussed cables but it offers a very high data transmission speed and noise immunity. This type of cabling is preferred for running networks between buildings or widely separated hubs. It has the highest length per cable segment i.e. 2000 meters and it can support 1024 nodes per cable segment.

Ethernet Frame Format



Preamble (64bit):
- allows the receiver to synchronize with the signal .
(sequence of alternating 0s and 1s).

Host and Destination Address physical addresses of the nodes(48bit each).

Packet type (16bit): - acts as demux key to identify the higher level protocol

Data (up to 1500 bytes)
- Minimally a frame must contain at least 46 bytes of data.
Frame must be long enough to detect collision.

CRC (32bit) - error detection

Ethernet Addresses

- Each host on an Ethernet (in fact, every Ethernet host in the world) has a unique Ethernet Address.
- The address belongs to the adaptor, not the host.
  - It is usually burnt into ROM.
- Ethernet addresses are typically printed in a human readable format
  - As a sequence of six numbers separated by colons.
  - Each number corresponds to 1 byte of the 6 byte address and is given by a pair of hexadecimal digits, one for each of the 4-bit nibbles in the byte
  - For example, 8:0:2b:e4:b1:2 is
    - 00001000 00000000 00101011 11100100 10110001 00000010
- To summarize, an Ethernet adaptor receives all frames and accepts
  - Frames addressed to its own address
  - Frames addressed to the broadcast address
  - Frames addressed to a multicast addressed if it has been instructed

Ethernet Transmitter Algorithm

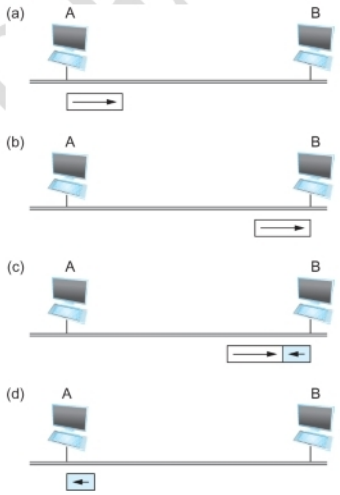
- Algorithm is defined as follows,"When the adaptor has a frame to send and theline is idle, it transmits the frame immediately"

- When the adaptor has a frame to send and the line is busy, it waits for the line to go idle and then transmits immediately.
- The Ethernet is said to be 1-persistent protocol because an adaptor with a frame to send transmits with probability 1 whenever a busy line goes idle.When this happens, the two (or more) frames are said to be *collide* on the network.
- Since Ethernet supports collision detection, each sender is able to determine that a collision is in progress.

- At the moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a 32-bit jamming sequence and then stops transmission.
    - Thus, a transmitter will minimally send 96 bits in the case of collision 64-bit preamble + 32-bit jamming sequence
- One way that an adaptor will send only 96 bit (called a *runt frame*) is if the two hosts are close to each other.
    - Had they been farther apart,They would have had to transmit longer, and thus send more bits, before detecting the collision.
- The worst case scenario happens when the two hosts are at opposite ends of the Ethernet.
- To know for sure that the frame its just sent did not collide with another frame, the transmitter may need to send as many as 512 bits.
    - Every Ethernet frame must be at least 512 bits (64 bytes) long.
        - 14 bytes of header + 46 bytes of data + 4 bytes of CRC

Ethernet Transmitter Algorithm

- A begins transmitting a frame at time *t*
- *d* denotes the one link latency
- The first bit of A's frame arrives at B at time *t* + *d*
- Suppose an instant before host A's frame arrives, host B begins to transmit its own frame
- B's frame will immediately collide with A's frame and this collision will be detected by host B
- Host B will send the 32-bit jamming sequence
- Host A will not know that the collision occurred until B's frame reaches it, which will happen at *t* + 2 * *d*
- Host A must continue to transmit until this time in order to detect the collision
- Host A must transmit for *2 * d* to be sure that it detects all possible collisions



Worst-case scenario:

(a) A sends a frame at time *t;*

(b) A's frame arrives at B at time *t + d;*

(c) B begins transmitting at time t + d and collides with A's frame;

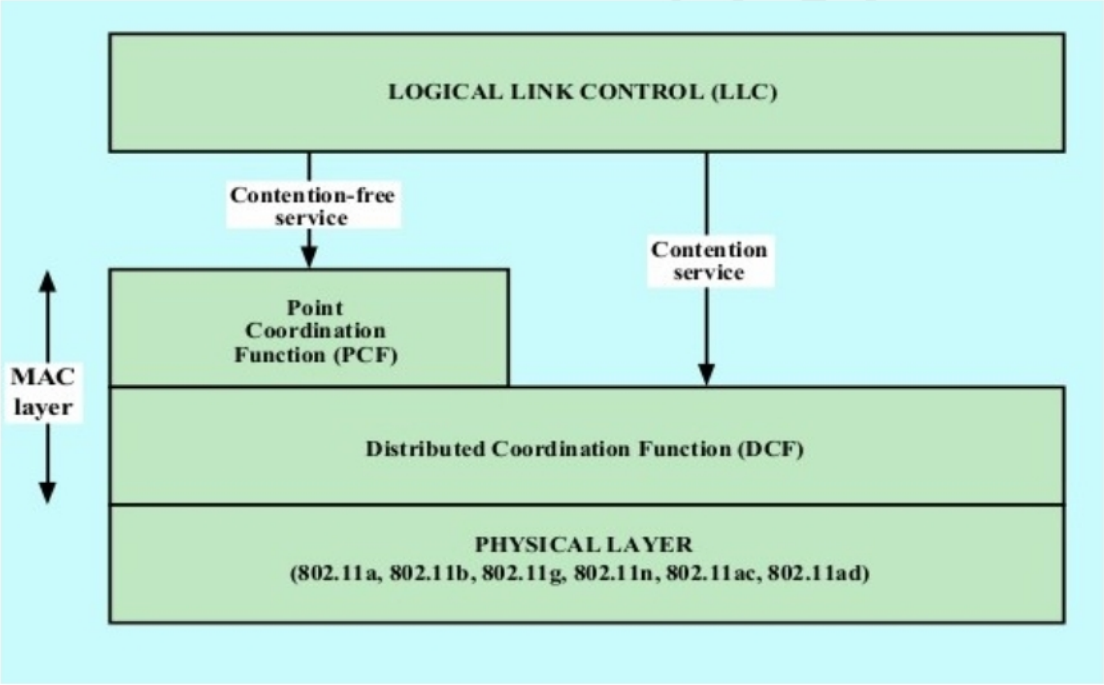
(d) B's runt (32-bit) frame arrives at A at time *t + 2d.*

- Once an adaptor has detected a collision, and stopped its transmission, it waits a certain amount of time and tries again.
- Each time the adaptor tries to transmit but fails, it doubles the amount of time it waits before trying again.
- This strategy of doubling the delay interval between each retransmission attempt is known as *Exponential Backoff.*

Wireless LAN (IEEE 802.11) / Wi-Fi

1. Wireless networking that is Standard IEEE 802.11 is a rapidly evolving technology for connecting computers. It is also known as Wi-Fi.
2. Like Ethernet and token ring siblings, 802.11 is designed for use in a limited geographical area (homes, office buildings, campuses)

Protocol Architecture



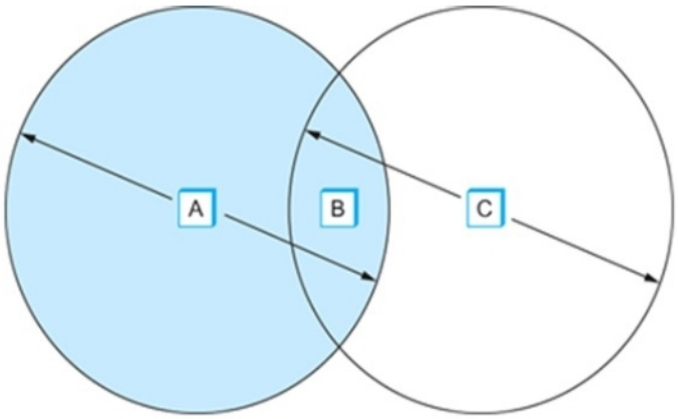
Physical Layer Properties:

- 802.11 runs over six different physical layer protocols (so far). Five are based on spread spectrum radio, and one on diffused infrared. The fastest runs at a maximum of 54 Mbps.

- Original 802.11 standard defined two radio-based physical layer standard
  - One using the frequency hopping
    - Over 79 1-MHz-wide frequency bandwidths
  - Second using direct sequence
    - Using 11-bit chipping sequence
  - Both standards run in the 2.4-GHz and provide up to 2 Mbps

- Then physical layer standard 802.11b was added
  - Using a variant of direct sequence 802.11b provides up to 11 Mbps
  - Uses license-exempt 2.4-GHz band
- Then came 802.11a which delivers up to 54 Mbps using OFDM
  - 802.11a runs on license-exempt 5-GHz band
- Most recent standard is 802.11g which is backward compatible with 802.11b
  - Uses 2.4 GHz band, OFDM and delivers up to 54 Mbps

Example of a wireless network:

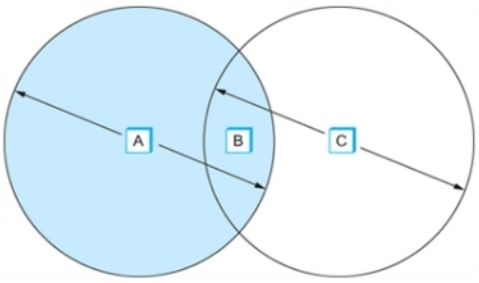


IEEE 802.11 Multiple Access Collision Avoidance (MACA)

- Multiple Access with Collision Avoidance (MACA) is a protocol for slotted media access control used in wireless LAN data transmission.
- MACA is used to avoid data collisions caused by hidden station problems as well as simplifying known station problems.

Hidden nodes problem

- Suppose both A and C want to communicate with B and so they each send it a frame.
  - A and C are unaware of each other since their signals do not carry that far
  - These two frames collide with each other at B
    - But unlike an Ethernet, neither A nor C is aware of this collision
  - A and C are said to ***hidden nodes*** with respect to each other

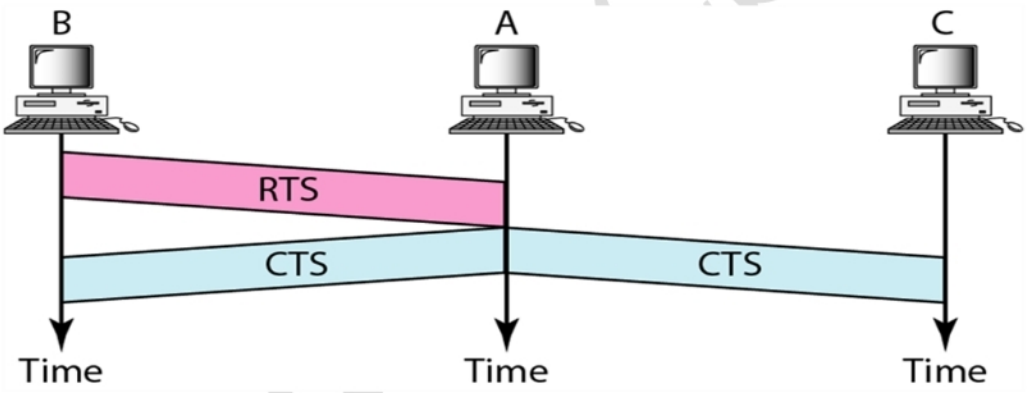


The "Hidden Node" Problem. Although A and C are hidden from each other, their signals can collide at B. (B's reach is not shown.)

- **Key Idea-Solution**
  - Sender and receiver exchange control frames with each other before the sender actually transmits any data.
  - This exchange informs all nearby nodes that a transmission is about to begin
  - Sender transmits a *Request to Send* (RTS) frame to the receiver.
    - The RTS frame includes a field that indicates how long the sender wants to hold the medium

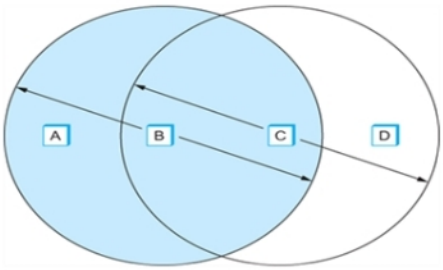
  -Length of the data frame to be transmitted
  - Receiver replies with a *Clear to Send* (CTS) frame
    - This frame echoes this length field back to the sender
- Any node that sees the CTS frame knows that
  - it is close to the receiver, therefore
  - cannot transmit for the period of time it takes to send a frame of the specified length
- Any node that sees the RTS frame but not the CTS frame
  - is not close enough to the receiver to interfere with it, and
  - so is free to transmit

Use of handshaking to prevent hidden station problem



Exposed node problem

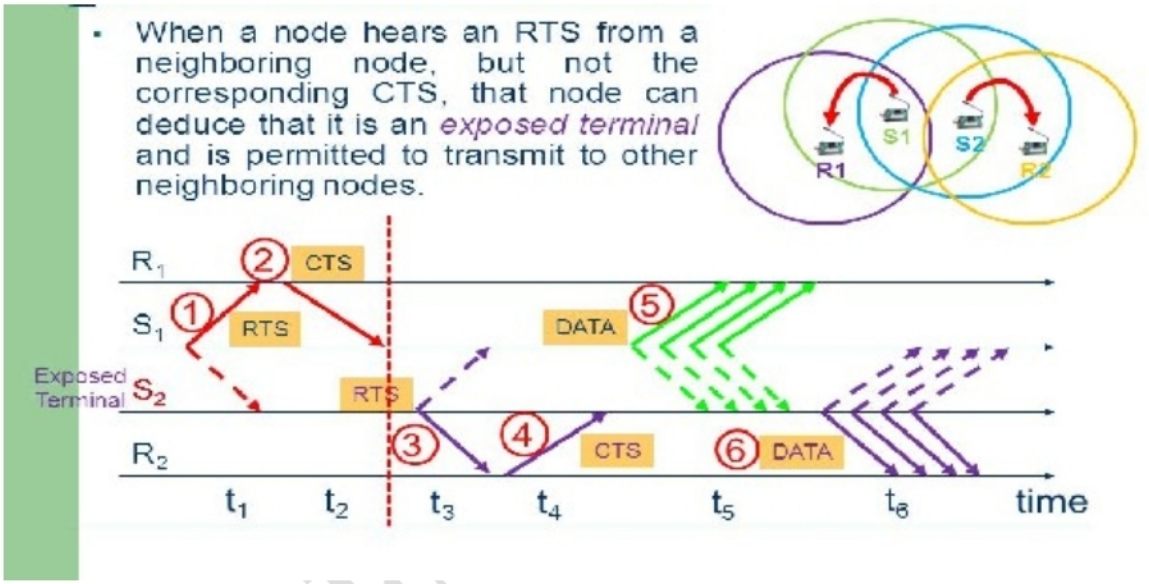
- Another problem called ***exposed node*** problem occurs
  - Suppose B is sending to A. Node C is aware of this communication because it hears B's transmission.
  - It would be a mistake for C to conclude that it cannot transmit to anyone just because it can hear B's transmission.
  - Suppose C wants to transmit to node D.
  - This is not a problem since C's transmission to D will not interfere with A's ability to receive from B.



C->D progress
If B->A not possible
Since C is exposed to B

Although B and C are exposed to each other's signals, there is no interference if B transmits to A while C transmits to D. (A and D's reaches are not shown.)

Solution for exposed node problem:



MACA: Medium Access Collision Avoidance:

MACA is for Wireless LANs

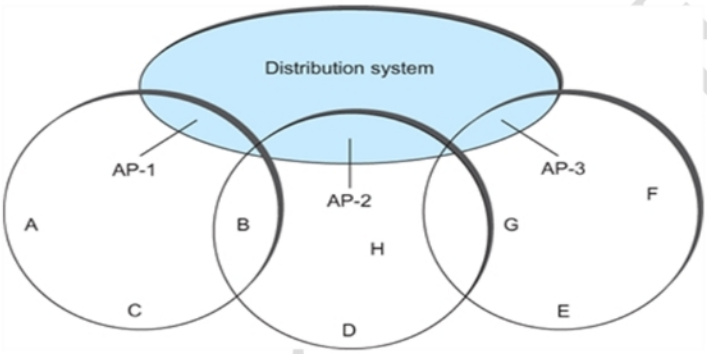
- Receiver sends an ACK to the sender after successfully receiving a frame
- All nodes must wait for this ACK before trying to transmit
- If two or more nodes detect an idle link and try to transmit an RTS frame at the same time
  - Their RTS frame will collide with each other
- 802.11 does not support collision detection
  - So the senders realize the collision has happened when they do not receive the CTS frame after a period of time
  - In this case, they each wait a random amount of time before trying again.
  - The amount of time a given node delays is defined by the same ***exponential backoff*** algorithm used on the Ethernet.
- 802.11 is suitable for an ad-hoc configuration of nodes that may or may not be able to communicate with all other nodes.
- Nodes are free to move around

- The set of directly reachable nodes may change over time
- To deal with this mobility and partial connectivity,
  - 802.11 defines additional structures on a set of nodes
  - Instead of all nodes being created equal,
    - some nodes are allowed to roam
    - some are connected to a wired network infrastructure

- they are called *Access Points* (AP) and they are connected to each other by a so-called *distribution system*

IEEE 802.11 – Distribution System

- Following figure illustrates a distribution system that connects three access points, each of which services the nodes in the same region
- Each of these regions is analogous to a cell in a cellular phone system with the APIs playing the same role as a base station
- The distribution network runs at layer 2 of the ISO architecture

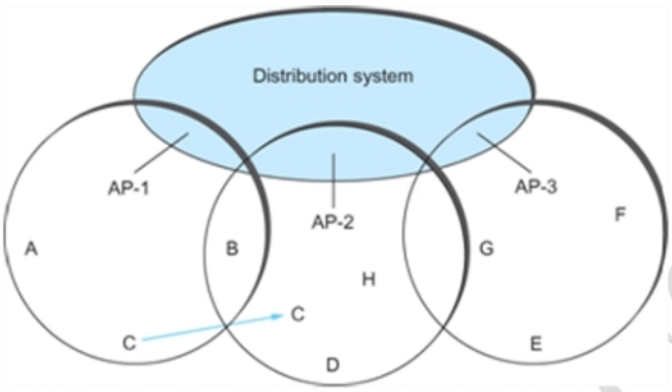


Access points connected to a distribution network

- Although two nodes can communicate directly with each other if they are within reach of each other, the idea behind this configuration is
  - Each nodes associates itself with one access point
  - For node A to communicate with node E, A first sends a frame to its AP-1 which forwards the frame across the distribution system to AP-3, which finally transmits the frame to E
- The technique for selecting an AP is called *scanning*
  - The node sends a *Probe* frame
  - All APs within reach reply with a *Probe Response* frame
  - The node selects one of the access points and sends that AP an *Association Request* frame
  - The AP replies with an *Association Response* frame
- A node engages this protocol whenever
  - it joins the network, as well as
  - when it becomes unhappy with its current AP
    - This might happen, for example, because the signal from its current AP has weakened due to the node moving away from it
    - Whenever a node acquires a new AP, the new AP notifies the old AP of the change via the distribution system

Node Mobility

- Consider the situation shown in the following figure when node C moves from the cell serviced by AP-1 to the cell serviced by AP-2.
- As it moves, it sends *Probe* frames, which eventually result in *Probe Responses* from AP-2.
- At some point, C prefers AP-2 over AP-1 , and so it associates itself with that access point.
  - This is called ***active scanning*** since the node is actively searching for an access point



Node Mobility

- APs also periodically send a *Beacon* frame that advertises the capabilities of the access point; these include the transmission rate supported by the AP
  - This is called ***passive scanning***
  - A node can change to this AP based on the *Beacon* frame simply by sending it an *Association Request* frame back to the access point.

IEEE 802.11 – Frame Format

- Source and Destinations addresses: each 48 bits
- Data: up to 2312 bytes
- CRC: 32 bit
- Control field: 16 bits
  - Contains three subfields (of interest)
    - 6 bit **Type** field: indicates whether the frame is an RTS or CTS frame or being used by the scanning algorithm
    - A pair of 1 bit fields : called **ToDS** and **FromDS**
- Addr1 – target node
- Addr2 – immediate sender
- Addr3 – intermediate destination node
- Addr4 – original source

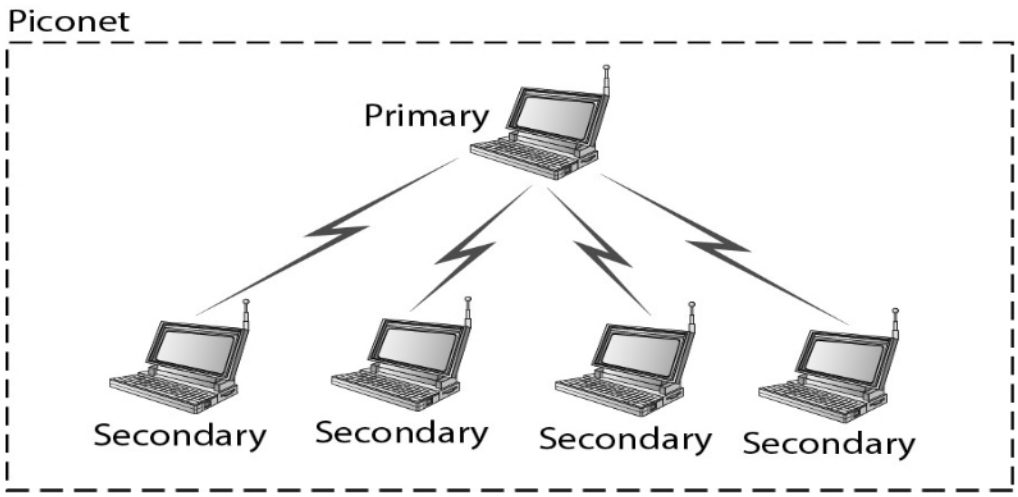
Bluetooth (802.15.1)

Bluetooth is a wireless LAN technology designed to connect devices of different functions such as telephones, notebooks, computers, cameras, printers, coffee makers, and so on. A Bluetooth LAN is an ad hoc network, which means that the network is formed spontaneously. Used for very short range communication between mobile phones, PDAs, notebook computers and other personal or peripheral devices.

Bluetooth is an adhoc network which means the network is formed spontaneously ie the devices find each other and make a network called piconet.

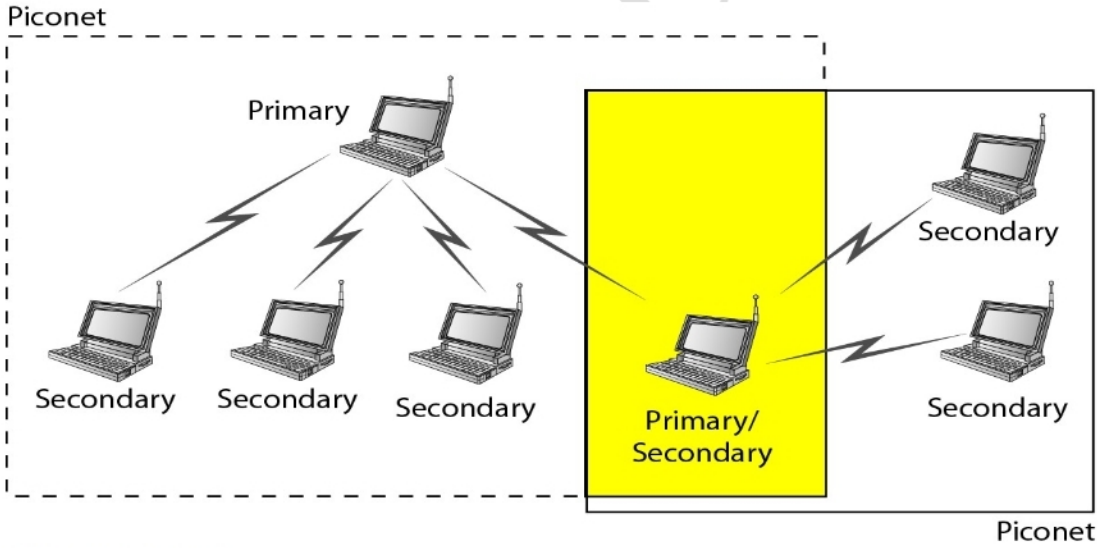
- Has a range of only 10 m
- Communication devices typically belong to one individual or group
  - Sometimes categorized as Personal Area Network (PAN)
- Version 2.0 provides speeds up to 2.1 Mbps
- Power consumption is low
- Bluetooth is specified by an industry consortium called the Bluetooth Special Interest Group
- It specifies an entire suite of protocols, going beyond the link layer to define application protocols, which it calls *profiles*, for a range of applications
  - There is a profile for synchronizing a PDA with personal computer
  - Another profile gives a mobile computer access to a wired LAN
- The basic Bluetooth network configuration is called a *piconet*
  - Consists of a master device and up to seven slave devices
  - Any communication is between the master and a slave
  - The slaves do not communicate directly with each other
  - A slave can be *parked*: set to an inactive, low-power state

A Bluetooth Piconet

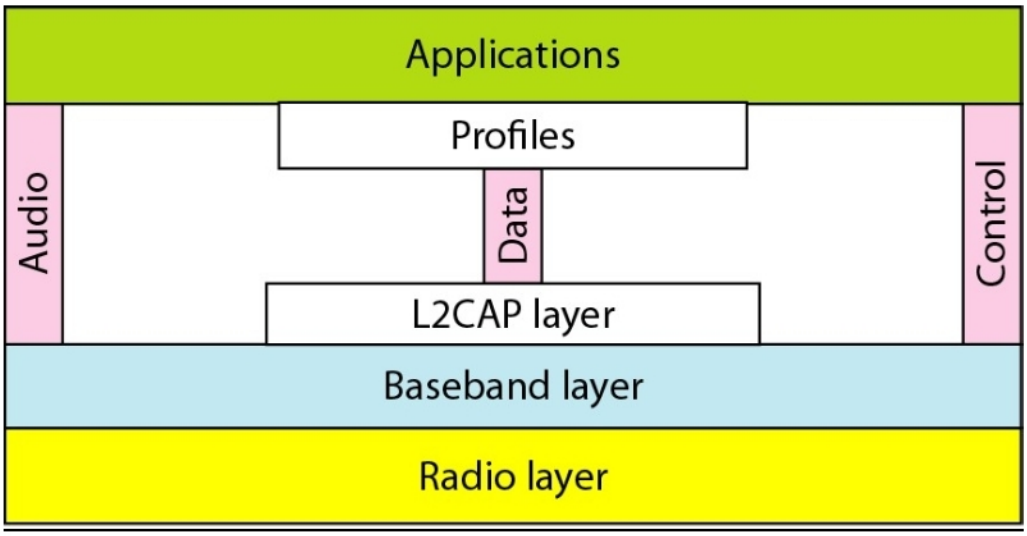


Scatternet

A scatternet is a number of interconnected piconets that supports communication between more than 8 devices. Scatternets can be formed when a member of one piconet (either the master or one of the slaves) elects to participate as a slave in a second, separate piconet. The device participating in both piconets can relay data between members of both ad hoc networks.



Bluetooth layers



L2CAP Layer

- Logical Link Control and Adaptation Layer
- Used for data exchange, It can do multiplexing

Baseband Layers

- Equivalent to MAC sublayer in LAN
- The primary and secondary stations communicate with each other using time slots (TDMA-Time Division Multiple Access)

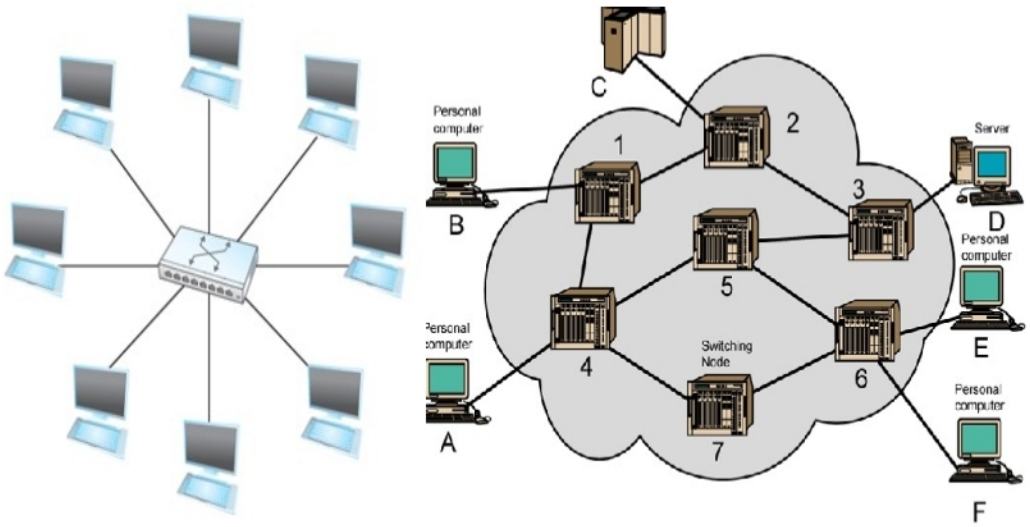
Radio layer

- Equivalent to physical layer of the internet model. Bluetooth devices are low-power and have a range of 10 m

Switch

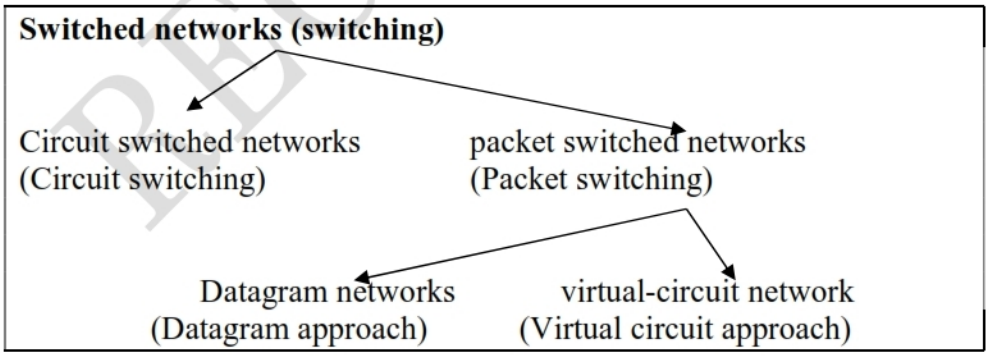
1. A mechanism that allows us to interconnect links(LANs) to form a large network is called **switching.**
2. Switches are devices capable of creating temporary connections between two or more devices linked to the switch.
3. A switch's primary job is to receive incoming packets on one of its links and to transmit them on some other link. This function is referred as switching and forwarding.
4. A multi-input, multi-output device which transfers packets from an input to one or more outputs.

EXAMPLE:



Advantages of switches:

1. Large networks can be built by interconnecting a number of switches.
2. We can connect switches to each other and to hosts using point-to-point links, which typically means that we can build networks of large geographic scope.
3. Adding a new host to the network by connecting it to a switch does not necessarily mean that the hosts already connected will get worse performance from the network.
4. A switch is connected to a set of links and for each of these links, runs the appropriate data link protocol to communicate with each node attached on link.



Circuit switched networks

In circuit switched network, it consists of set of switches connected by physical links. A connection between two stations is a dedicated path made of one or more links. Physical layer is responsible for circuit switching.

The communication in circuit-switched network has three phases,

1) Connection setup
2) Data transfer
3) Teardown (connection termination)

In these types of networks, data are not packetized and there is a continuous flow by the source station to destination station.

Packet switched networks

If the message is passing through a packet switched network, it needs to be divided into packets of fixed or variable size. This type of switching is done by network layer.

The two approaches commonly used are,

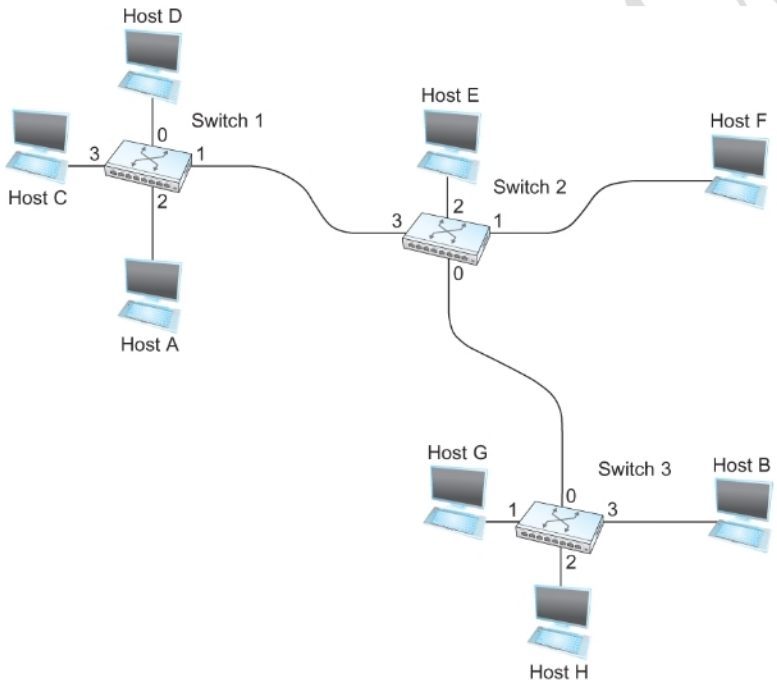
- ***Datagram or Connectionless approach***
- ***Virtual circuit or Connection-oriented approach***

Datagram approach

Every packet contains enough information i.e. destination address that enable any switch to decide where the packet has to go.

Consider An example network with many hosts for datagram forwarding,

- To decide how to forward a packet, a switch consults a *forwarding table* (sometimes called a *routing table*)



Destination	Port
A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0

Forwarding Table for Switch 2

Fig :
Datagram forwarding : an example network

Characteristics of Connectionless (Datagram) Network

- A host can send a packet anywhere at any time.
- When a host sends a packet, it has no way of knowing if the network is capable of delivering it or if the destination host is even up and running.
- Each packet is forwarded independently of previous packets that might have been sent to the same destination.

- Thus two successive packets from host A to host B may follow completely different paths
- A switch or link failure might not have any serious effect on communication if it is possible to find an alternate route around the failure and update the forwarding table accordingly.

Virtual Circuit Switching

- Widely used technique for packet switching
- Uses the concept of ***virtual circuit*** **(VC)**
- Also called a connection-oriented model
- First set up a virtual connection from the source host to the destination host and then send the data

- Host A wants to send packets to host B

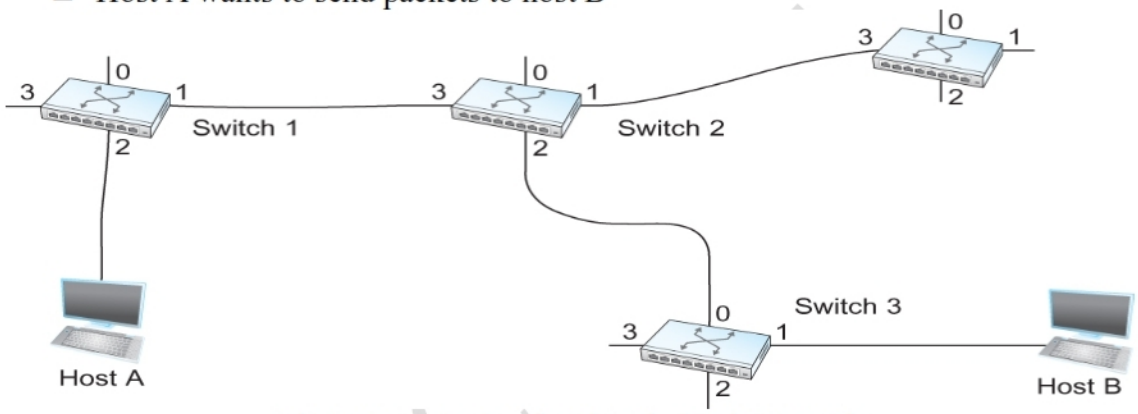


Fig : An example of a virtual circuit network

Two-stage process

- Connection setup
- Data Transfer

Connection setup

- Establish "connection state" in each of the switches between the source and destination hosts
- The connection state for a single connection consists of an entry in the "VC table" in each switch through which the connection passes

One entry in the VC table on a single switch contains

- A virtual circuit identifier (VCI) that uniquely identifies the connection at this switch and that will be carried inside the header of the packets that belong to this connection
- An incoming interface on which packets for this VC arrive at the switch
- An outgoing interface in which packets for this VC leave the switch

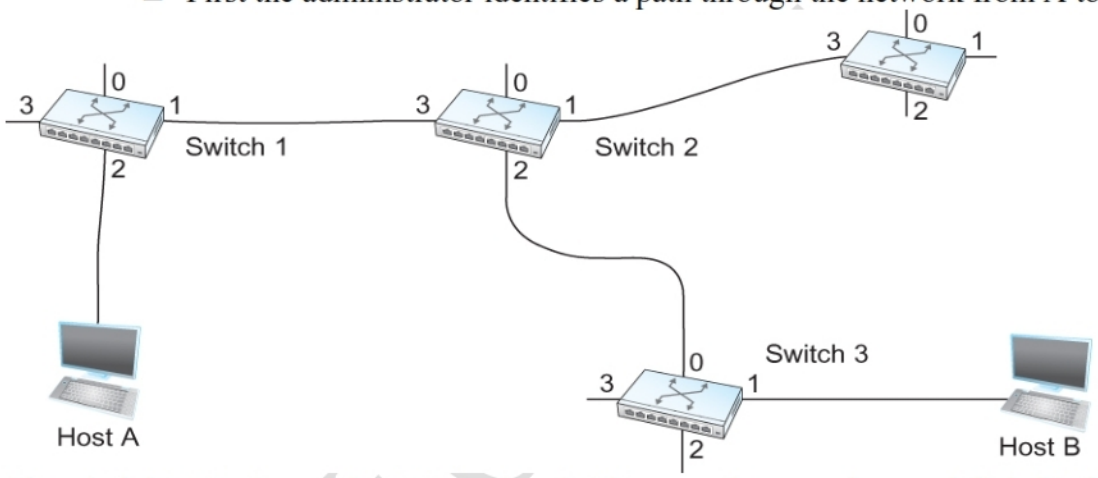
A potentially different VCI that will be used for outgoing packets.

Two broad classes of approach to establishing connection state

- Network Administrator will configure the state
  - The virtual circuit is permanent (PVC)
  - The network administrator can delete this
  - Can be thought of as a long-lived or administratively configured VC
- A host can send messages into the network to cause the state to be established
  - This is referred as signalling and the resulting virtual circuit is said to be switched (SVC)
  - A host may set up and delete such a VC dynamically without the involvement of a network administrator

Let's assume that a network administrator wants to manually create a new virtual connection from host A to host B

- First the administrator identifies a path through the network from A to B



The administrator then picks a VCI value that is currently unused on each link for the connection

- For our example,
  - Suppose the VCI value 5 is chosen for the link from host A to switch 1
  - 11 is chosen for the link from switch 1 to switch 2
  - So the switch 1 will have an entry in the VC table

For switch 1

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
2	5	1	11

Similarly, suppose

- VCI of 7 is chosen to identify this connection on the link from switch 2 to switch 3
- VCI of 4 is chosen for the link from switch 3 to host B
- Switches 2 and 3 are configured with the following VC table

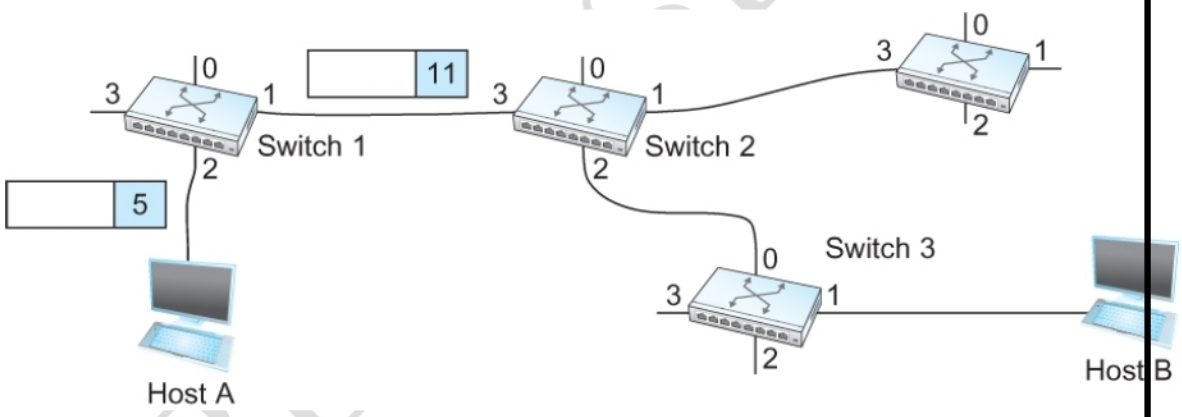
Switch 2

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
3	11	2	7

Switch 3

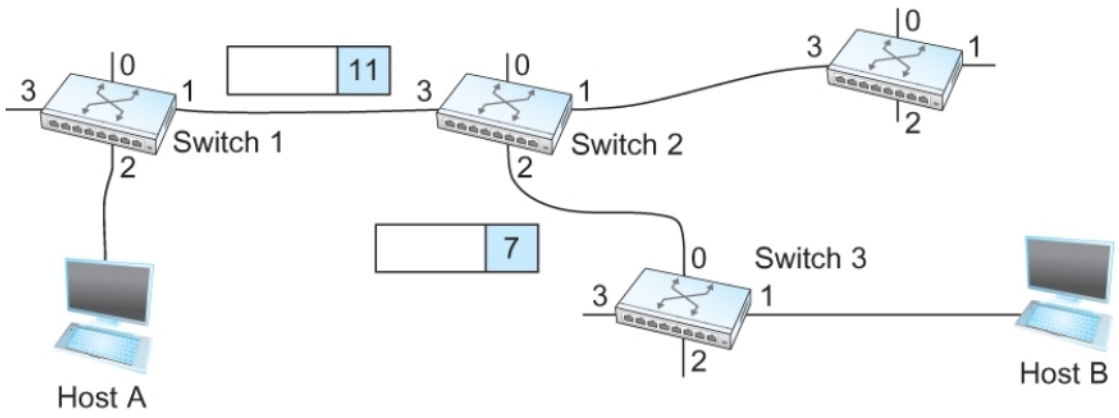
Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
0	7	1	4

- For any packet that A wants to send to B, A puts the VCI value 5 in the header of the packet and sends it to switch 1
- Switch 1 receives any such packet on interface 2, and it uses the combination of the interface and the VCI in the packet header to find the appropriate VC table entry.
- The table entry on switch 1 tells the switch to forward the packet out of interface 1 and to put the VCI value 11 in the header



Packet will arrive at switch 2 on interface 3 bearing VCI 11

- Switch 2 looks up interface 3 and VCI 11 in its VC table and sends the packet on to switch 3 after updating the VCI value appropriately
- This process continues until it arrives at host B with the VCI value of 4 in the packet
- To host B, this identifies the packet as having come from host A



Data transfer

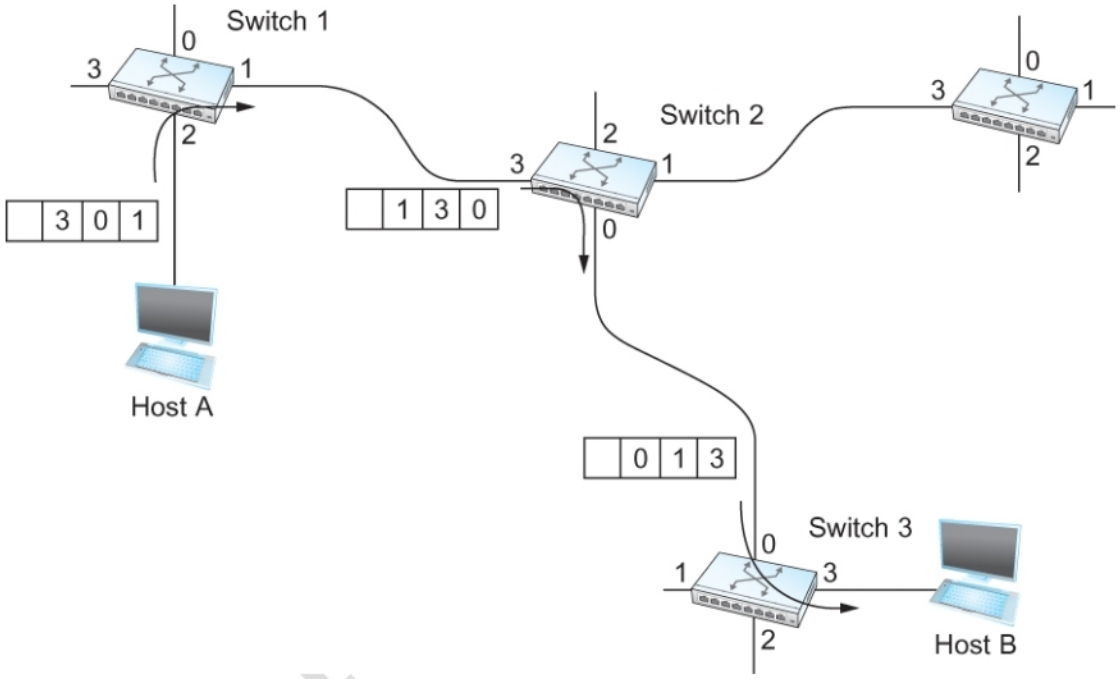
The data transfer phase is active until the source sends all its frame to destination when the connection state is established.

Teardown

In this phase, source A, after sending all frames to B, sends a special frame called teardown request. Destination B responds with a teardown confirmation frame and all switches delete the corresponding entry from their tables.

Source Routing

- Its a type if less commonly used approach in packet switching
- All the information about network topology that is required to switch a packet across the network is provided by the source host



Circuit switching	Packet switching
The connection between two station is a dedicated path using physical links	Virtual connection is established between two station
Physical layer is responsible	Network layer is responsible
Messages are not packetized and there is a continuous flow of messages	Messages are divided into packets of fixed or variable size
Used in telephone companies	Used in switched WANs such as frame relay and ATM networks
-	Types are • Datagram approach • Virtual circuit approach • Source routing

Bridges

- Bridge is a connecting device that is used to connect two or more LANs. It operates in both the physical and the data link layer.
- As a physical layer device, it regenerates the signal it receives.
- As a data link layer device, the bridge can check the physical (MAC) addresses (source and destination) contained in the frame.
- Switch that is used to forward packets between shared-media LANs such as Ethernets.
- It is also called as LAN switches.

Transparent bridges

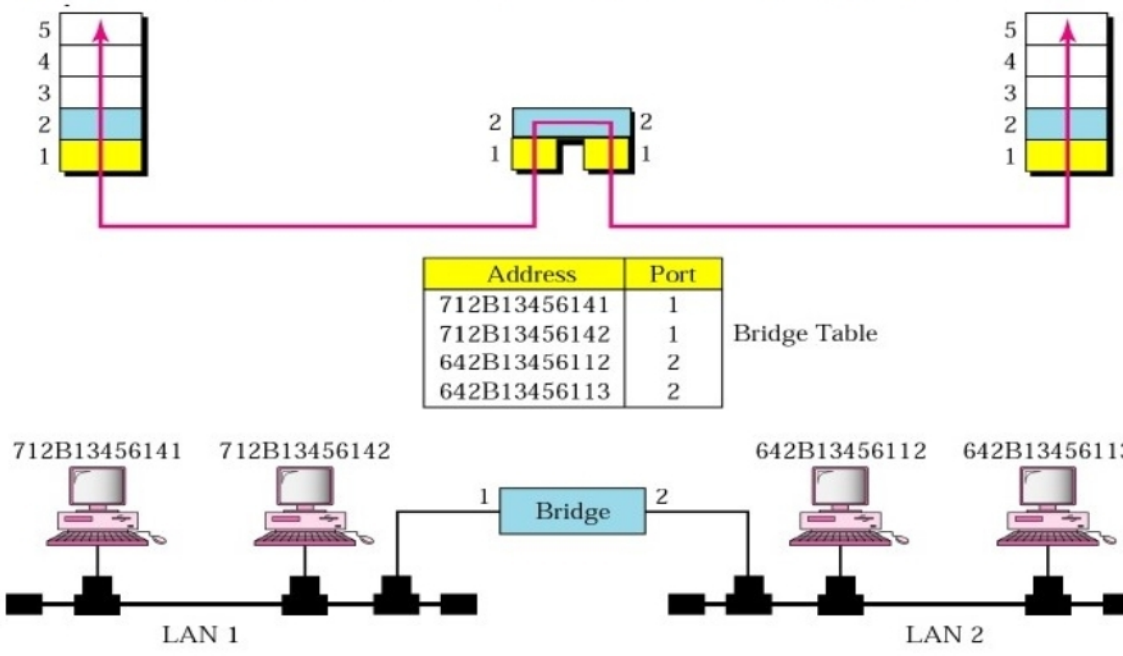
It is a bridge in which the stations are completely unaware of bridge's existence. If a bridge is added or deleted from the system, reconfiguration is not needed.
According to the IEEE 802.1 d specification, a system equipped with transparent bridges must meet three criteria, as follows

Functions of bridges (three criteria)

1. Frame filtering or forwarding
2. Learning
3. Avoidance of loops in system

Frame filtering or forwarding

Bridges has a table which is used for filtering or forwarding decisions. It can check the destination address of frame and decide if the frame has to be forwarded or dropped. If the frame is to be forwarded, the decision must specify the port.



Address	Port
712B13456141	1
712B13456142	1
642B13456112	2
642B13456113	2

Bridge Table

Let us give an example. In Figure two LANs are connected by a bridge. If a frame destined for station 712B13456142 arrives at port 1, the bridge consults its table to find the departing port. According to its table, frames for 7l2B 13456142 leave through port 1; therefore, there is no need for forwarding, and the frame is dropped. On the other hand, if a frame for 712B13456141 arrives at port 2, the departing port is port 1 and the frame is forwarded.

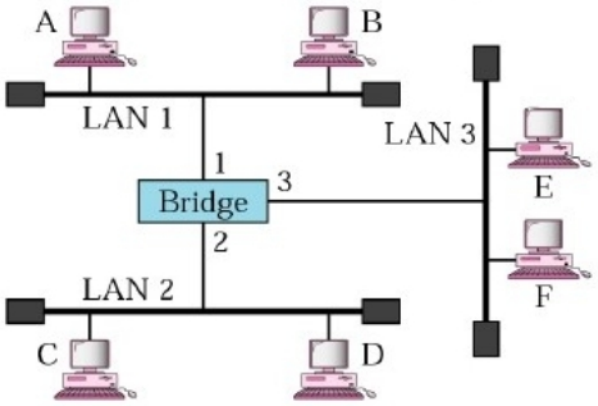
Learning

The earliest bridges had forwarding tables that were static. The systems administrator would manually enter each table entry during bridge setup. Although the process was simple, it was not practical.

If a station was added or deleted, the table had to be modified manually. The same was true if a station's MAC address changed, which is not a rare event.

A better solution to the static table is a dynamic table that maps addresses to ports automatically. To make a table dynamic, we need a bridge that gradually learns from the frame movements.

To do this, the bridge inspects both the destination and the source addresses. The destination address is used for the forwarding decision (table lookup); the source address is used for adding entries to the table and for updating purposes.



Address	Port

a. Original

Address	Port
A	1

b. After A sends a frame to D

Address	Port
A	1
E	3

c. After E sends a frame to A

Address	Port
A	1
E	3
B	1

d. After B sends a frame to C

1. When station A sends a frame to station D, the bridge does not have an entry for either D or A. The frame goes out from all three ports; the frame floods the network. However, by looking at the source address, the bridge learns that station A must be located on the LAN connected to port 1. This means that frames destined for A, in the future, must be sent out through port 1. The bridge adds this entry to its table. The table has its first entry now.

2. When station E sends a frame to station A, the bridge has an entry for A, so it forwards the frame only to port 1. There is no flooding. In addition, it uses the source address of the frame, E, to add a second entry to the table.
3. When station B sends a frame to C, the bridge has no entry for C, so once again it floods the network and adds one more entry to the table.
4. The process of learning continues as the bridge forwards frames.

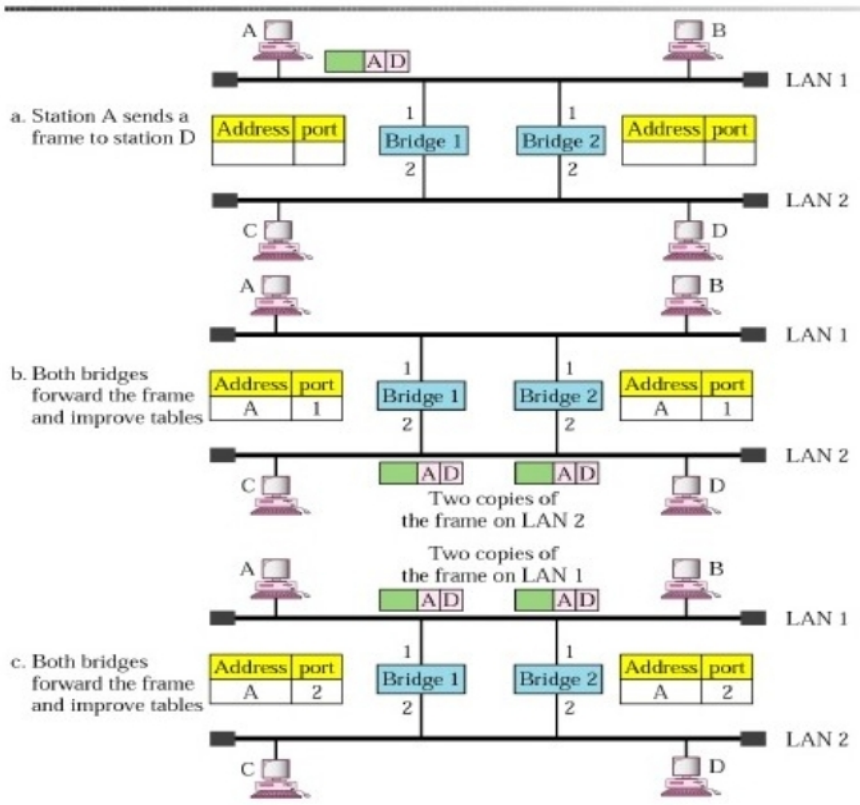
Loop Problem

Loop Problem Transparent bridges work fine as long as there are no redundant bridges in the system. Systems administrators, however, like to have redundant bridges (more than one bridge between a pair of LANs) to make the system more reliable.

If a bridge fails, another bridge takes over until the failed one is repaired or replaced. Redundancy can create loops in the system, which is very undesirable.

Figure shows a very simple example of a loop created in a system with two LANs connected by two bridges.

1. Station A sends a frame to station D. The tables of both bridges are empty. Both forward the frame and update their tables based on the source address A.
2. Now there are two copies of the frame on LAN 2. The copy sent out by bridge 1 is received by bridge 2, which does not have any information about the destination address D; it floods the bridge. The copy sent out by bridge 2 is received by bridge 1 and is sent out for lack of information about D. Note that each frame is handled separately because bridges, as two nodes on a network sharing the medium, use an access method such as CSMA/CD. The tables of both bridges are updated, but still there is no information for destination D.
3. Now there are two copies of the frame on LAN 1. Step 2 is repeated, and both copies flood the network.
4. The process continues on and on. Note that bridges are also repeaters and regenerate frames. So in each iteration, there are newly generated fresh copies of the frames. To solve the looping problem, the IEEE specification requires that bridges use the spanning tree algorithm to create a loopless topology.

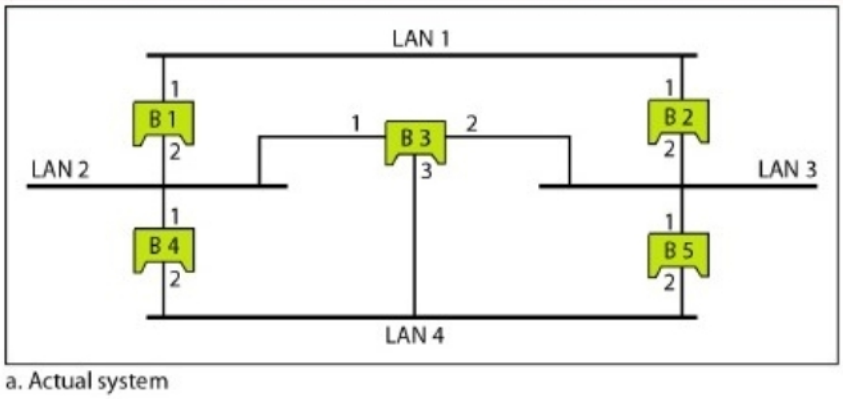


Spanning Tree

In graph theory, a **spanning tree** is a graph in which there is no loop. In a bridged LAN, this means creating a topology in which each LAN can be reached from any other LAN through one path only (no loop).

We cannot change the physical topology of the system because of physical connections between cables and bridges, but we can create a logical topology that overlays the physical one.

The following figure shows a system with four LANs and five bridges.

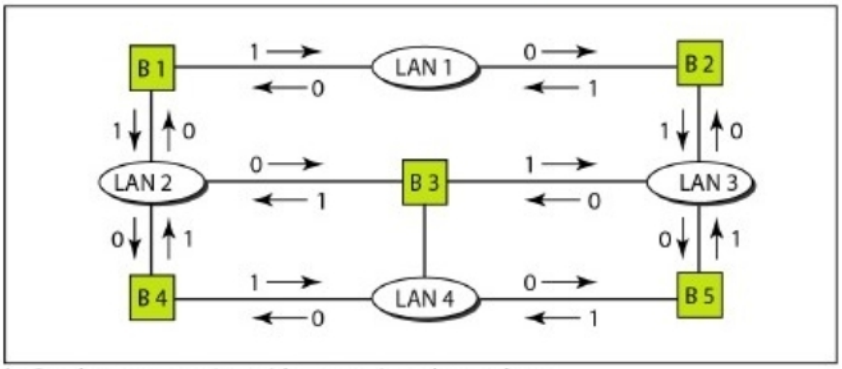


a. Actual system

The graph representation with cost assigned to each arc of above system is as follows.

The connecting arcs show the connection of a LAN to a bridge and vice versa. To find the spanning tree, we need to assign a cost (metric) to each arc.

[Note : assign 1 from Bridge to LAN, assign 0 from LAN to Bridge]

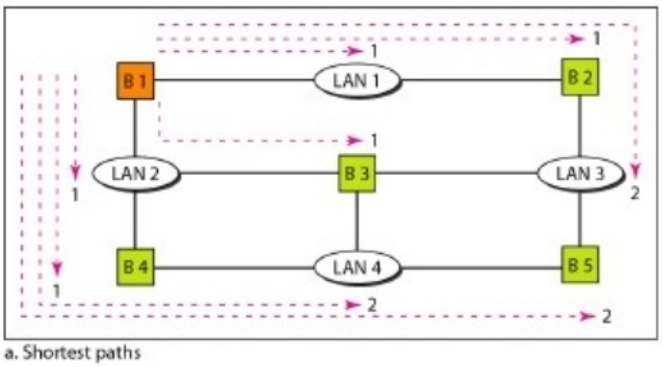


b. Graph representation with cost assigned to each arc

In spanning tree approach, the cost to each arc is assigned by system administrator.

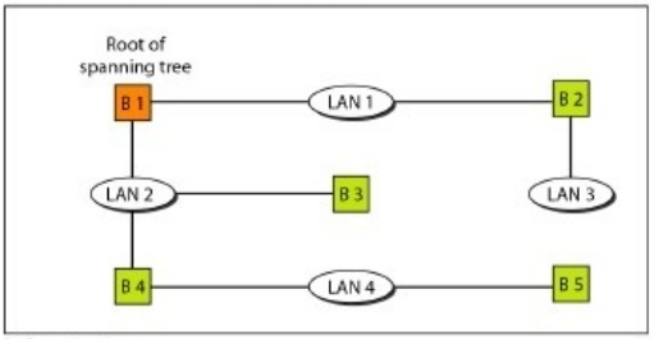
The process to find the spanning tree involves three steps:

1. Every bridge has a built-in ID (normally the serial number, which is unique). Each bridge broadcasts this ID so that all bridges know which one has the smallest ID. The bridge with the smallest ID is selected as the *root* bridge (root of the tree). We assume that bridge B1 has the smallest ID. It is, therefore, selected as the root bridge.
2. The algorithm tries to find the shortest path (a path with the shortest cost) from the root bridge to every other bridge or LAN as follows,



a. Shortest paths

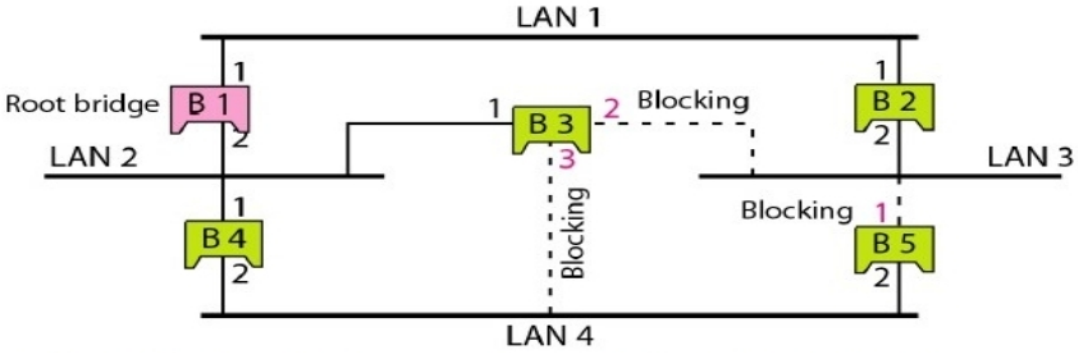
3. The combination of the shortest paths creates the shortest tree, which is also shown in the above figure.



b. Spanning tree

4. Based on the spanning tree approach some ports are forwarding ports, which forward a frame that the bridge receives. Some ports are blocking ports which block the frames received by bridge.

Forwarding and blocking ports after using spanning tree algorithm



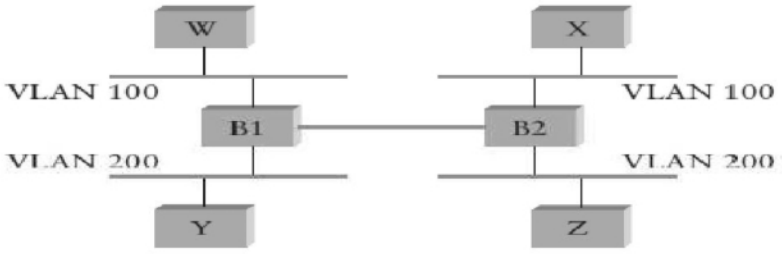
Ports 2 and 3 of bridge B3 are blocking ports. Port1 of bridge B5 is also blocking port.

- Spanning tree algorithm is a **dynamic algorithm** .
- Each bridge is equipped with a software process that carries the process dynamically. Each bridge send special messages to one another called bridge protocol data units (BPDUs) to update spanning tree.

Limitations of Bridges

- On the issue of scale, it is not realistic to connect more than a few LANs by means of bridges, where in practice "few" typically means "tens of.
- One reason for this is that the spanning tree algorithm scales linearly; that is, there is no provision for imposing a hierarchy on the extended LAN.
- A second reason is that bridges forward all broadcast frames.
- One approach to increasing the scalability of extended LANs is the *virtual LAN*(VLAN). VLANs allow a single extended LAN to be partitioned into several seemingly separate LANs.
- Each virtual LAN is assigned an identifier (sometimes called a *color),*and packets can only travel from one segment to another if both segments have the same identifier.

Two virtual LANs share a common backbone



- shows four hosts on four different LAN segments. In the absence of VLANs, any broadcast packet from any host will reach all the other hosts.

Internetworking

The term —internetwork or sometimes just internet refers to an arbitrary collection of networks interconnected to provide some sort of host to- host packet delivery service.

- *internetwork is often referred to as a "net work of networks" because it is made up of lots of smaller networks.al* network built out of a collection of physical networks.
- The nodes that interconnect the networks are called routers. They are also sometimes called gateways.
- The Internet Protocol is the key tool used today to build scalable, heterogeneous internetworks. It was originally known as the Kahn-Cerf protocol after its inventors.

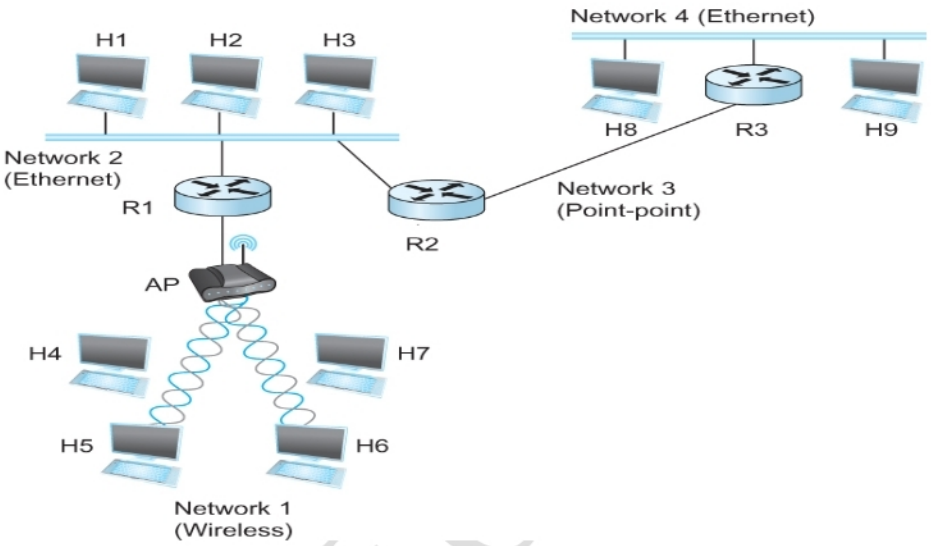


Fig: A simple internetwork. H*n* = host; R*n* = router.

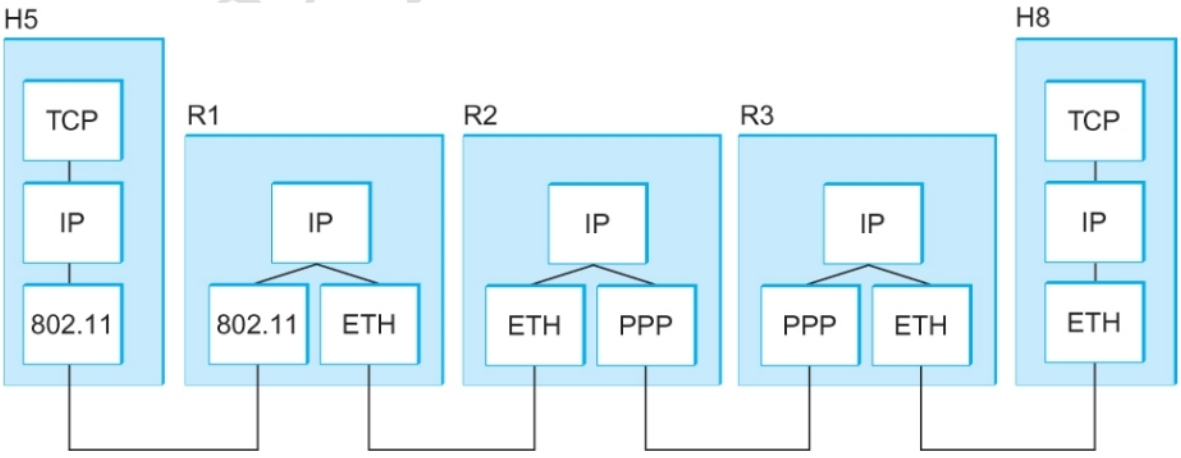


Fig : A simple internetwork, showing the protocol layers used to connect H5 to H8

IP (Internet Protocol)

- IP stands for Internet Protocol
- Key tool used today to build scalable, heterogeneous internetworks
- It runs on all the nodes in a collection of networks and defines the infrastructure that allows these nodes and networks to function as a single logical internetwork

Service Model

Service model is, the host-to-host services you want to provide.

- It has two parts
  1. Datagram Delivery Model
     - Connectionless model for data delivery
  2. Global Addressing Scheme
     - Provides a way to identify all hosts in the network

Datagram Delivery

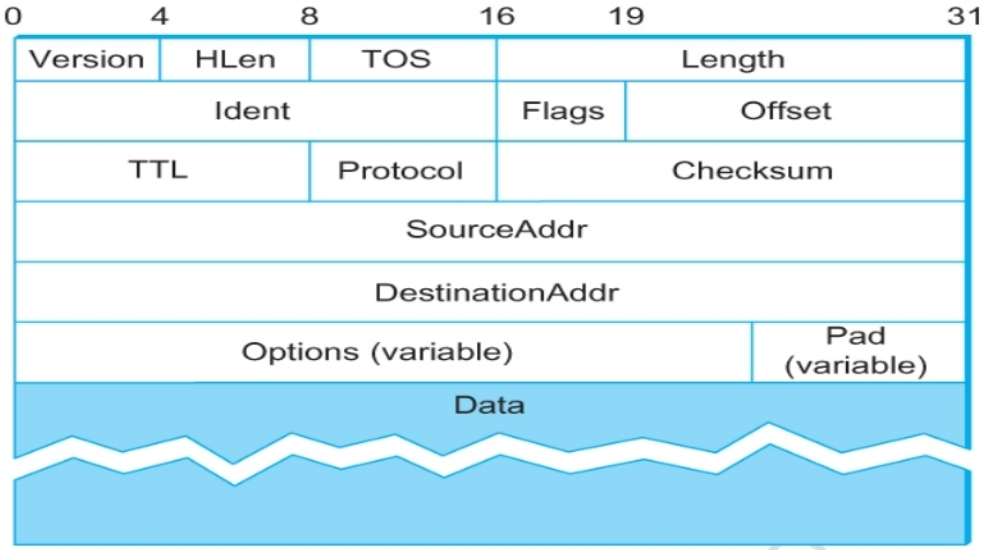
5. Datagram is a type of packet that happens to be sent in a connectionless manner over a network. Every datagram carries enough information to let the network forward the packet to its correct destination; there is no need for any advance setup mechanism to tell the network what to do when the packet arrives.

- Best-effort delivery (unreliable service)
  - » packets are lost or
  - » packets are delivered out of order or
  - » duplicate copies of a packet are delivered or
  - » packets can be delayed for a long time

Advantages of Datagram Delivery Model

- ✓ Best-effort, connectionless service
- ✓ simplest service model
- ✓ asking for a reliable packet delivery service may need to include a lot of extra functionality into the router.
- ✓ It enables IP to "run over anything" ie today IP can run over many network technologies
- ✓ Higher level protocols such as TCP that run over IP is aware of all failure modes.

IPv4 Packet Format



- Version (4): currently 4
- Hlen (4): number of 32-bit words in header
- TOS (8): type of service (not widely used)
- Length (16): number of bytes in this datagram. Max size of IP datagram is 65535 Bytes which is not supported for physical networks. So IP supports fragmentation and reassembly.
- Ident (16): used by fragmentation
- Flags/Offset (16): used by fragmentation
- TTL (8): specifies no of sec that a packet would be allowed to live
- Protocol (8): demux key (TCP=6, UDP=17)
- Checksum (16): for error detection
- DestAddr & SrcAddr (32)

IP Fragmentation and Reassembly

- Each network has some MTU (Maximum Transmission Unit)
  - Ethernet (1500 bytes), FDDI (4500 bytes)
- Strategy
  - Fragmentation occurs in a router when it receives a datagram that it wants to forward over a network which has (MTU < datagram)
  - Reassembly is done at the receiving host
  - All the fragments carry the same identifier in the *Ident* field
  - Fragments are self-contained datagrams
  - IP does not recover from missing fragments

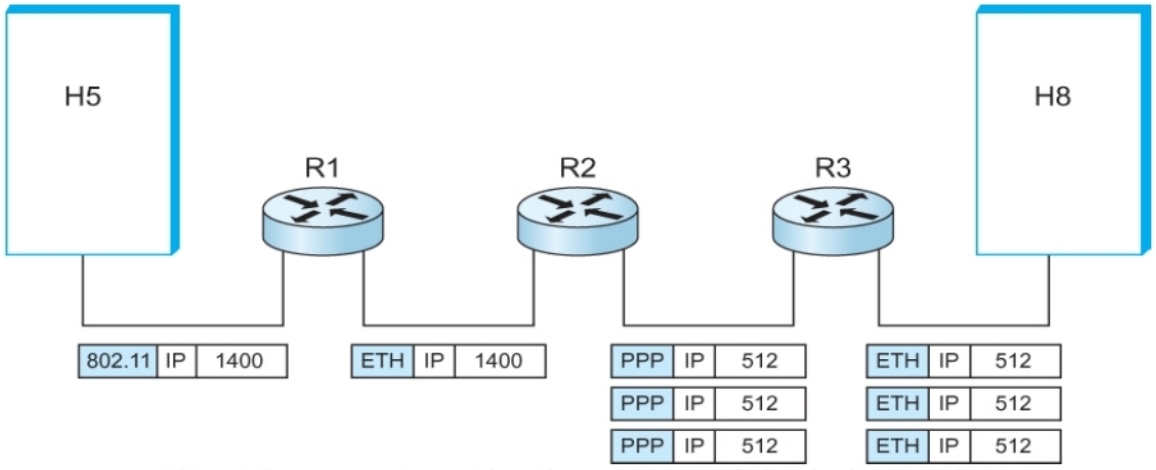


Fig: IP datagrams traversing the sequence of physical networks

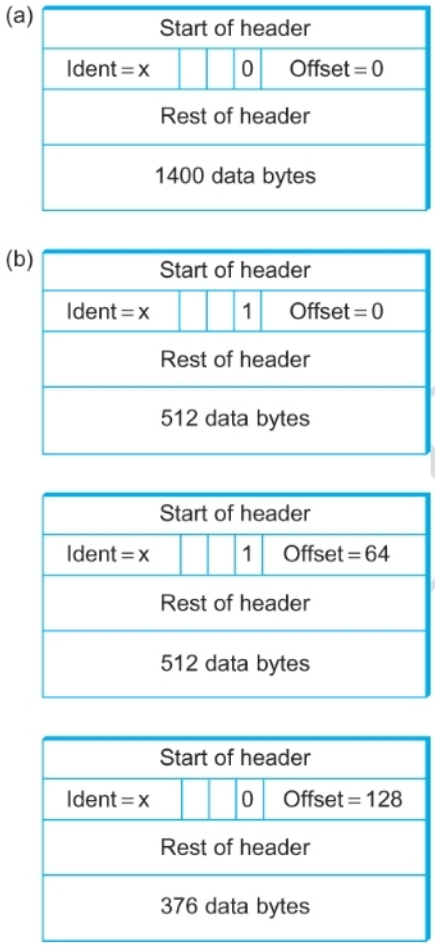
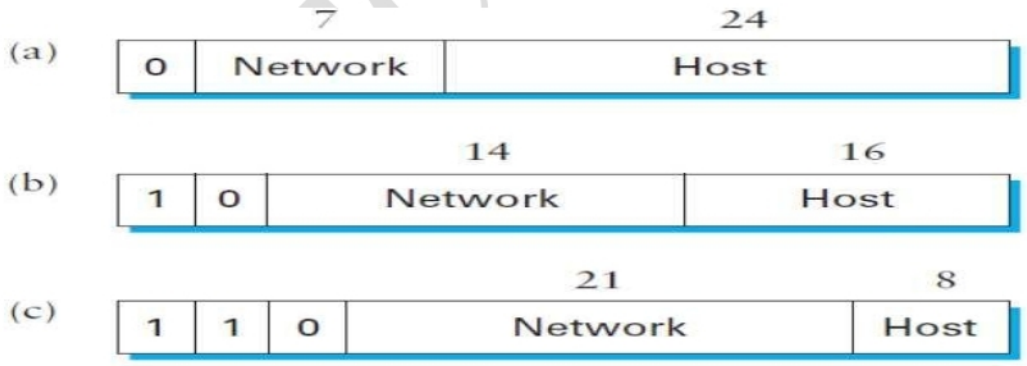


Fig: Header fields used in IP fragmentation. (a) Unfragmented packet; (b) fragmented packets

Global Addresses

- Global uniqueness is the first property that should be provided in an addressing scheme.
- Ethernet addresses are globally unique, but that alone does not suffice for an addressing scheme in a large internetwork. Ethernet addresses are also *flat*, which means that they have no structure and provide very few clues to routing protocols.
- In contrast, IP addresses are *hierarchical*, by which we mean that they are made up of several parts that correspond to some sort of hierarchy in the internetwork.
- **Specifically, IP addresses consist of two parts, a network part and a host part.** This is a fairly logical structure for an internetwork, which is made up of many interconnected networks.
- The network part of an IP address identifies the network to which the host is attached; all hosts attached to the same network have the same network part in their IP address.
- The host part then identifies each host uniquely on that particular network.
    - IP addresses are divided into three different classes.
- (There are also class D addresses that specify a multicast group, and class E addresses that are currently unused.)



IP addresses : (a)class A;(b)class B;class C.

The class of an IP address is identified in the most significant few bits. If the first bit is 0, it is a class A address. If the first bit is 1 and the second is 0, it is a class B address.

If the first two bits are 1 and the third is 0, it is a class C address. Thus, of the approximately 4 billion possible IP addresses, half are class A, one quarter are class B, and one-eighth are class C.

Each class allocates a certain number of bits for the network part of the address and the rest for the host part.

Class A networks have 7 bits for the network part and 24 bits for the host part, meaning that there can be only 126 class A networks (the values 0 and 127 are reserved), but each of them can accommodate up to 224 − 2 (about 1 6 million) hosts (again, there are two reserved values).

Class B addresses allocate 14 bits for the network and 16 bits for the host, meaning that each class B network has room for 65,534 hosts. Finally, class C addresses have only 8 bits for the host and 21 for the network part.

Therefore, a class C network can have only 256 unique host identifiers, which means only 254 attached hosts (one host identifier, 255, is reserved for broadcast, and 0 is not a valid host number). However, the addressing scheme supports 221 class C networks.

By convention, IP addresses are written as four *decimal* integers separated by dots. Each integer represents the decimal value contained in 1 byte of the address, starting at the most significant. For example, the address of the computer on which this sentence was typed is 171.69.210.245.

Datagram Forwarding in IP

- Strategy
  - every datagram contains destination's address
  - if directly connected to destination network, then forward to host
  - if not directly connected to destination network, then forward to some router
  - forwarding table maps network number into next hop
  - each host has a default router
  - each router maintains a forwarding table

Algorithm

```
if (NetworkNum of destination = NetworkNum of one of my interfaces) then
        deliver packet to destination over that interface
else
        if (NetworkNum of destination is in my forwarding table) then
                deliver packet to NextHop router
        else
                deliver packet to default router
```

For a host with only one interface and only a default router in its forwarding table, this simplifies to

```
if (NetworkNum of destination = my NetworkNum)then
        deliver packet to destination directly
else
        deliver packet to default router
```

SUBNETTING

The original intent of IP addresses was that the network part would uniquely identify exactly one physical network.

It has two drawbacks.

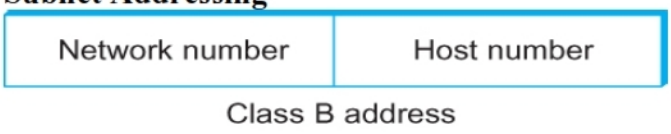
- 1. Address assignment inefficiency.
- 2. Assigning network number to every physical network uses
- the IP address space potentially much faster.

Assigning many network numbers has another drawback that becomes apparent when you think about routing.

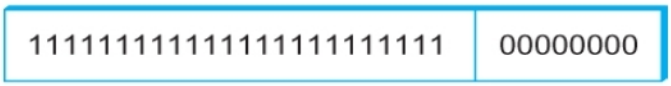
- ***Subnetting provides an elegantly simple way to reduce the total number of network* numbers that are assigned.**
- The idea is to take a single IP network number and allocate the IP addresses with that network number to several physical networks, which are now referred to as *subnets*.

- First, the subnets should be close to each other.
- This is because at a distant point in the Internet, they will all look like a single network, having only one network number between them.
- This means that a router will only be able to select one route to reach any of the subnets, so they had better all be in the same general direction.
- A perfect situation in which to use subnetting is a large campus or corporation that has many physical networks.
- From outside the campus, all you need to know to reach any subnet inside the campus is where the campus connects to the rest of the Internet.
- The mechanism by which a single network number can be shared among multiple networks involves configuring all the nodes on each subnet with a *subnet mask*.
- With simple IP addresses, all hosts on the same network must have the same network number.
- The subnet mask enables us to introduce a *subnet number; all hosts on the same* physical network will have the same subnet number, which means that hosts may be on different physical networks but share a single network number.
- For example, suppose that we want to share a single class B address among several physical networks. We could use a subnet mask of 255.255.255.0.
- (Subnet masks are written down just like IP addresses; this mask is therefore all 1s in the upper 24 bits and 0s in the lower 8 bits.)
- In effect, this means that the top 24 bits(where the mask has 1s) are now defined to be the network number, and the lower 8 bits (where the mask has 0s) are the host number.
- Since the top 16 bits identify the network in a class B address, we may now think of the address as having not two parts but three: a network part, a subnet part, and a host part.

Subnet Addressing



Class B address



Subnet mask (255.255.255.0)

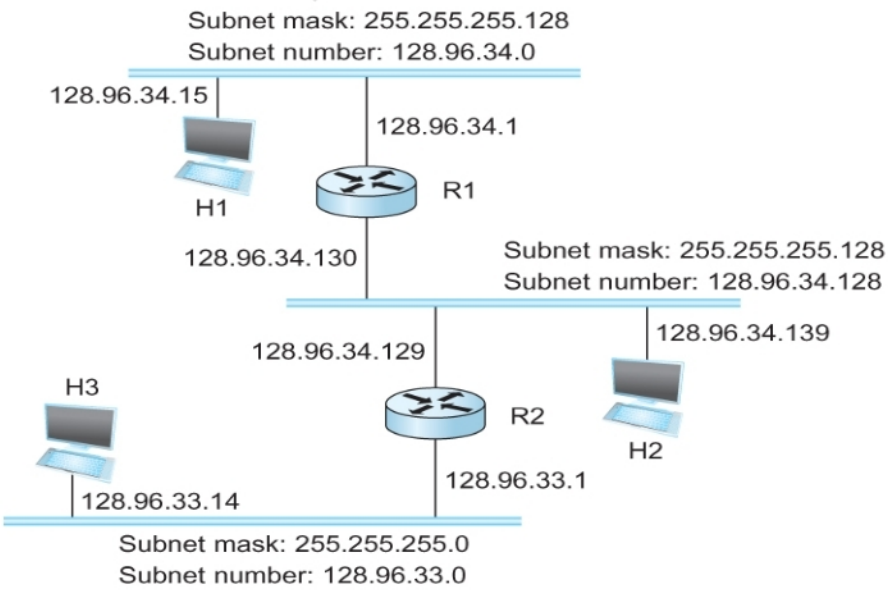
Subnetted address

- What subnetting means to a host is that it is now configured with both an IP address and a subnet mask for the subnet to which it is attached.
- For example, host H1 in Figure 4.26 is configured with an address of 128.96.34.15 and a subnet mask of 255.255.255.128.
- The bitwise AND of these two numbers defines the subnet number of the host and of all other hosts on the same subnet.
- In this case, 128.96.34.15 AND 255.255.255.128 equals 128.96.34.0, so this is the subnet number for the topmost subnet.

Forwarding Packet through Subnet

When the host wants to send a packet to a certain IP address, the first thing it does is to perform a bitwise AND between its own subnet mask and the destination IP address. If the results are not equal, the packet needs to be sent to a router to be forwarded to another subnet.

For example, if H1 is sending to H2, then H1 ANDs its subnet mask (255.255.255.128) with the address for H2 (128.96.34.139) to obtain 128.96.34.128. This does not match the subnet number for H1 (128.96.34.0) so H1 knows that H2 is on a different subnet. Since H1 cannot deliver the packet toH2 directly over the subnet, it sends the packet to its default router R1.



To support subnetting, the routing table must now hold entries of the form (SubnetNumber, SubnetMask, NextHop). To find the right entry in the table,the router ANDs the packet's destination address with the SubnetMask for each entry in turn; if the result matches the SubnetNumber of the entry, then this is the right entry to use, and it forwards the packet to the next hop router indicated.

Forwarding Algorithm

```
D = destination IP address
for each entry < SubnetNum, SubnetMask, NextHop>
D1 = SubnetMask & D
if D1 = SubnetNum
if NextHop is an interface
  deliver datagram directly to destination
else
  deliver datagram to NextHop (a router)
```

Example Subnetting Table

SubnetNumber	SubnetMask	NextHop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2

Classless Routing (CIDR)

Classless Inter Domain Routing (CIDR, pronounced —cider) is a technique that addresses two scaling concerns in the Internet.

- Do not use classes to determine network ID
- Assign any range of addresses to network
  - Use common part of address as network number
  - E.g., addresses 192.4.16 - 192.4.31 have the first 20 bits in common. Thus, we use these 20 bits as the network number
  - netmask is /20, /xx is valid for almost any xx
- Enables more efficient usage of address space (and router tables)

The growth of backbone routing tables as more and more network numbers need to be stored in them, and the potential for the 32-bit IP address space to be exhausted well before the four-billionth host is attached to the Internet.

This address space exhaustion is called address assignment inefficiency.

The inefficiency arises because the IP address structure, with class A, B, and C addresses, forces us to hand out network address space in fixed-sized chunks of three very different sizes.

- A network with two hosts needs a class C address
  - Address assignment efficiency = 2/255 = 0.78
- A network with 256 hosts needs a class B address
  - Address assignment efficiency = 256/65535 = 0.39

Even though subnetting can help us to assign addresses carefully, it does not get around the fact that any autonomous system with more than 255 hosts, or an expectation of eventually having that many, wants a class B address.
As it turns out, exhaustion of the IP address space centers on exhaustion of the class B network numbers.

One way to deal with that would seem to be saying no to any AS (Autonomous Systems) that requests a class B address unless they can show a need for something close to 64K addresses, and instead giving them an appropriate number of class C addresses to cover the expected number of hosts. Since we would now be handing out address space in chunks of 256 addresses at a time, we could more accurately match the amount of address space consumed to the size of the AS.

For any AS with at least 256 hosts (which means the majority of ASs), we can guarantee an address utilization of at least 50%, and typically much more.

This solution, however, raises a problem that is at least as serious: excessive storage requirements at the routers.

- If a single AS has, say 16 class C network numbers assigned to it,
  - Every Internet backbone router needs 16 entries in its routing tables for that AS
  - This is true, even if the path to every one of these networks is the same
- If we had assigned a class B address to the AS
  - The same routing information can be stored in one entry
  - Efficiency = 16 × 255 / 65, 536 = 6.2%

CIDR, therefore, tries to balance the desire to minimize the number of routes that a router needs to know against the need to hand out addresses efficiently.

To do this, CIDR helps us to *aggregate* routes.

- Uses a single entry in the forwarding table to tell the router how to reach a lot of different networks
- Breaks the rigid boundaries between address classes

To understand how this works, consider our hypothetical AS with 16 class C network numbers. Instead of handing out 16 addresses at random, we can hand out a block of *contiguous* class C addresses.

- Suppose we assign the class C network numbers from 192.4.16 through 192.4.31
- Observe that top 20 bits of all the addresses in this range are the same (11000000 00000100 0001)
  - We have created a 20-bit network number (which is in between class

B network number and class C number in terms of the number of hosts that it can support)

In other words, we get both the high address efficiency of handing out addresses in chunks smaller than a class B network and a single network prefix that can be used in forwarding tables.

Observe that for this scheme to work, we need to hand out blocks of class C addresses that share a common prefix, which means that each block must contain a number of class C networks that is a power of two.

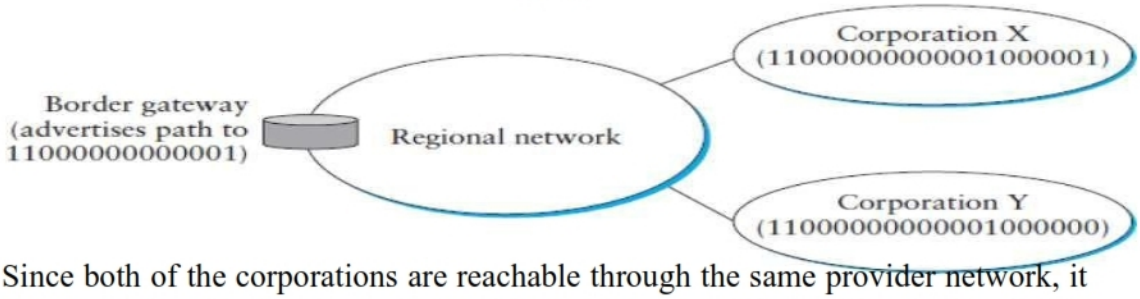
All we need now to make CIDR solve our problems is a routing protocol that can deal with these —classless addresses, which means that it must understand that a network number may be of any length.

Modern routing protocols do exactly that. The network numbers that are carried in such a routing protocol are represented simply by _length, value_ pairs, where the length gives the number of bits in the network prefix—20 in the above example.

- Requires to hand out blocks of class C addresses that share a common prefix
- The convention is to place a /X after the prefix where X is the prefix length in bits
- For example, the 20-bit prefix for all the networks 192.4.16 through 192.4.31 is represented as 192.4.16/20
- By contrast, if we wanted to represent a single class C network number, which is 24 bits long, we would write it 192.4.16/24
- How do the routing protocols handle this classless addresses
  - It must understand that the network number may be of any length
- Represent network number with a single pair
  <length, value>
- All routers must understand CIDR addressing

Consider the example Figure The two corporations served by the provider network have been assigned adjacent 20- bit network prefixes.

Route Aggregate with CIDR

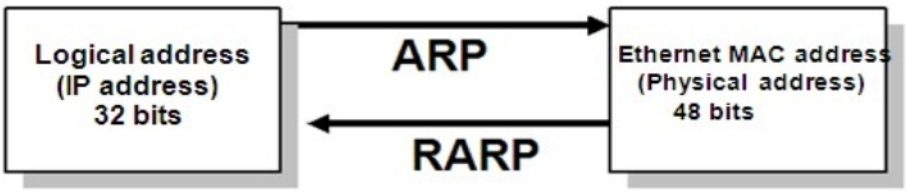


Since both of the corporations are reachable through the same provider network, it can advertise a single route to both of them by just advertising the common 19-bit prefix they share.
In general, it is possible to aggregate routes repeatedly if addresses are assigned carefully.

This means that we need to pay attention to which provider a corporation is attached to before assigning it an address if this scheme is to work. One way to accomplish that is to assign a portion of address space to the provider and then to let the network provider assign addresses from that space to its customers.

ARP Address Resolution Protocol

- The Address Resolution Protocol (ARP) is used to map logical address (IP address) into physical address.
- On a typical physical network, such as a LAN, each device on a link is identified by a physical or station address, usually imprinted on the network interface card (NIC).
- ARP is used to find the physical address of the node when its Internet address is known.



ARP operation

Two operations are

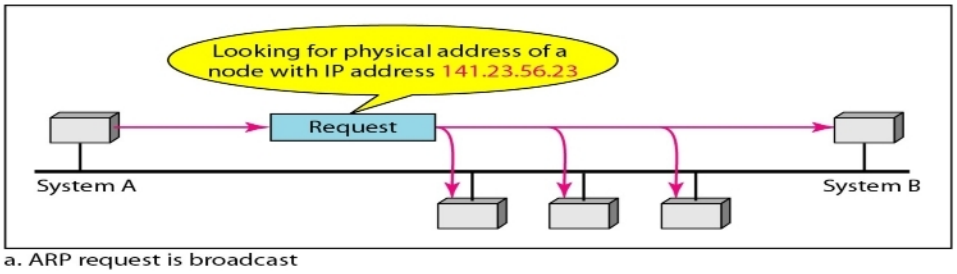
1. ARP request
2. ARP reply

ARP request

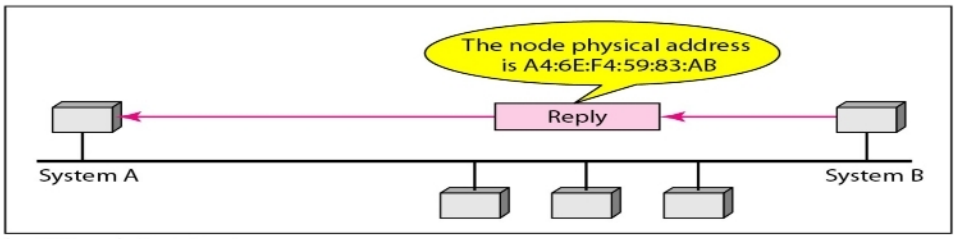
To determine the physical address for a particular IP, the node broadcasts an ARP query or ARP request containing that IP address in the network.

ARP reply

The node with that IP address responds to the sender of the ARP request with an ARP response or ARP reply containing its link-layer address or physical address,



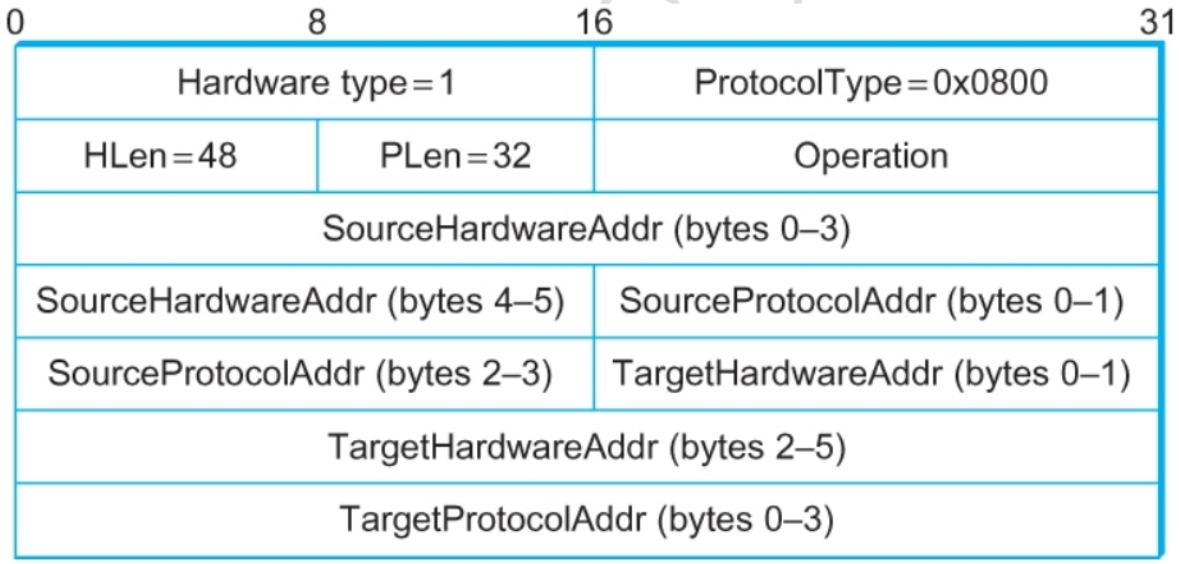
a. ARP request is broadcast



b. ARP reply is unicast

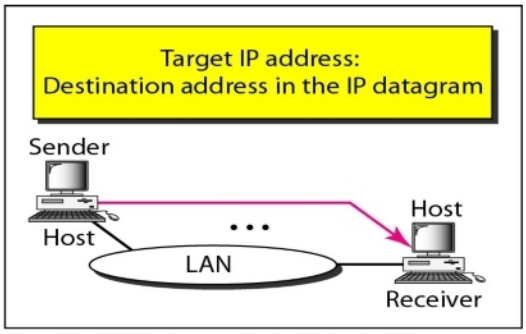
For providing mapping between logical address and physical address, each nodes maintain an ARP cache or ARP table. This table maintains an address pairs that map logical address to physical address.

ARP Packet Format

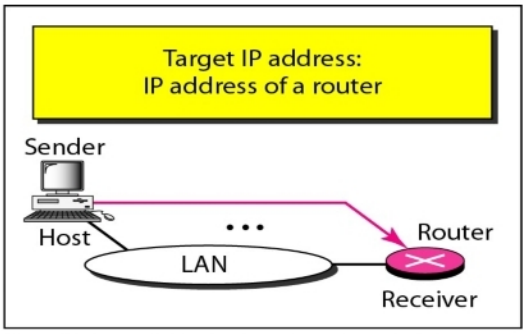


- HardwareType: type of physical network (e.g., Ethernet)
- ProtocolType: type of higher layer protocol (e.g., IP)
- HLEN : length of physical addresses in bytes eg : ethernet – 6 bytes
- PLEN: length of logical addresses in bytes. (eg: IPv4 – 4 bytes)
- Operation: request or response
- Source/Target Physical/Protocol addresses

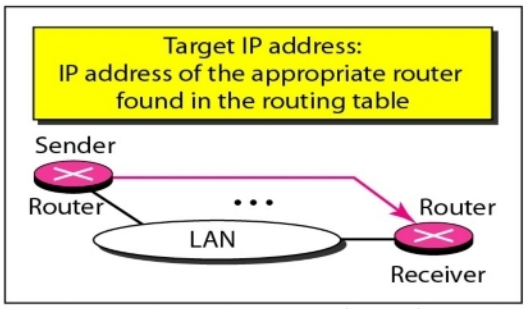
Four cases using ARP



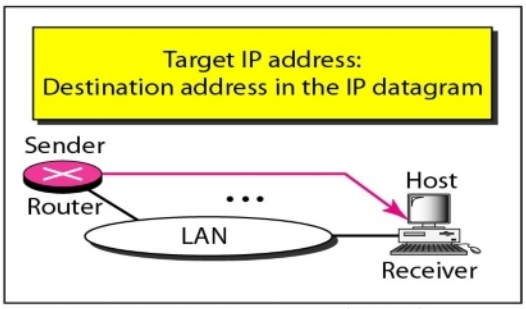
Case 1. A host has a packet to send to another host on the same network.



Case 2. A host wants to send a packet to another host on another network. It must first be delivered to a router.



Case 3. A router receives a packet to be sent to a host on another network. It must first be delivered to the appropriate router.



Case 4. A router receives a packet to be sent to a host on the same network.

RARP

Reverse Address Resolution Protocol (RARP)
Finds the logical address for a machine that knows only its physical address.

DHCP - Dynamic Host Configuration Protocol

- For Mapping Physical to Logical Address can use RARP/BOOTP/DHCP

- DHCP server is responsible for providing configuration information to hosts
- DHCP is a protocol that dynamically assigns IP addresses to host.

Need for dynamic host configuration

1. Ethernet addresses are configured into network by manufacturer and they are unique
2. IP addresses must be unique on a given internetwork but also must reflect the structure of the internetwork
3. Most host Operating Systems provide a way to manually configure the IP information for the host
4. Drawbacks of manual configuration
    i. A lot of work to configure all the hosts in a large network
    ii. Configuration process is error-prune
5. For this reason, automated Configuration Process is required
6. DHCP server is responsible for providing configuration information to hosts. There is at least one DHCP server for an administrative domain
7. DHCP server maintains a pool of available addresses.
8. DHCP is a protocol that dynamically assigns IP addresses to host.
9. DHCP relies on the existence of a DHCP server which is responsible for providing configuration information to host.
10. The use of DHCP avoids the network administrator from assigning addresses to individual host.

The Purpose of DHCP

- Provide dynamic allocation of IP client configuration for a lease period
- Eliminate the work necessary to administer a large IP Network

The first problem faced in DHCP is server discovery. To avoid this, a host undergo the following activities,

1. Newly booted or attached host sends DHCPDISCOVER message to a special IP address (255.255.255.255) which is an IP broadcast address that is received by all hosts and routers on that network.
2. DHCP uses the concept of relay agent, there is at least one relay agent on each network.
3. When a relay agent receives a DHCPDISCOVER message, it unicasts it to DHCP server and awaits for the response, which is then sent back to the requesting client.

The process of relaying a message from a host to remote DHCP server is as follows,

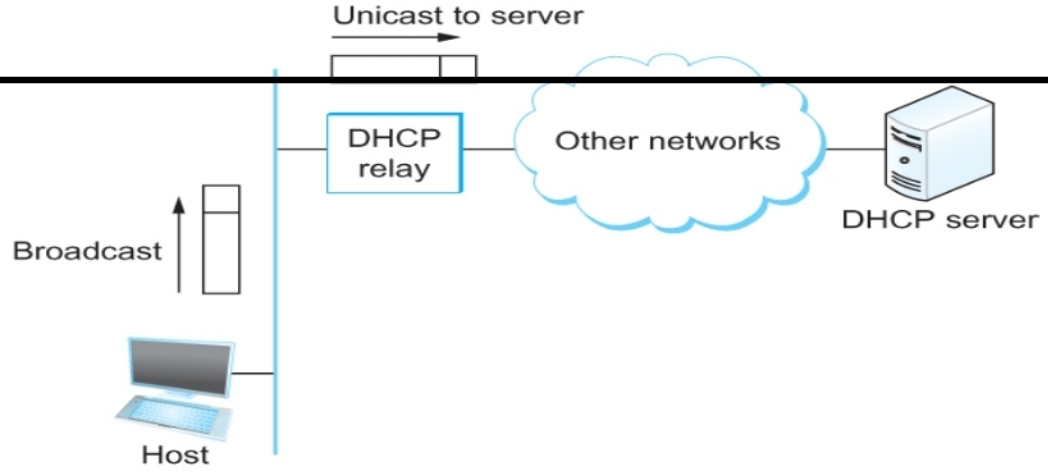
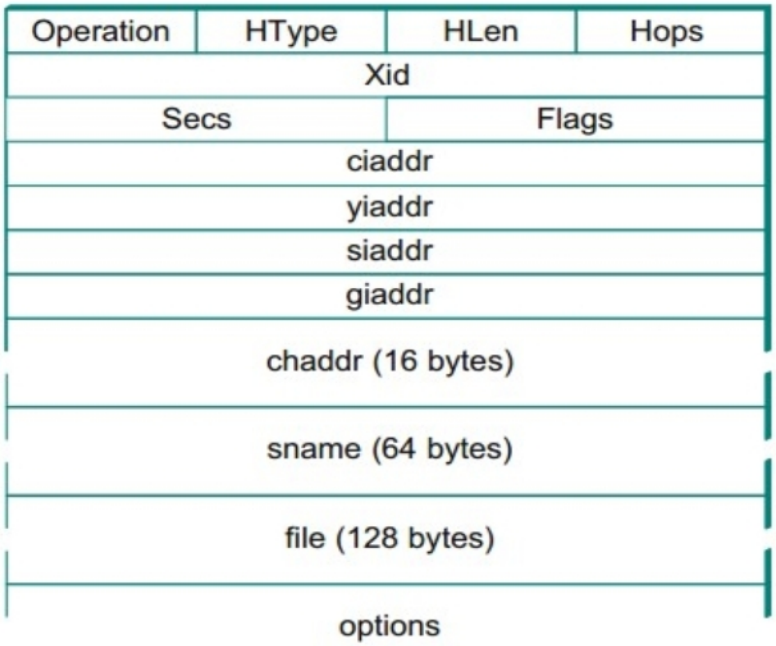


FIGURE : A DHCP relay agent receives a broadcast DHCPDISCOVERmessage from a host and sends a unicast DHCPDISCOVER to the DHCP server.

DHCP packet format

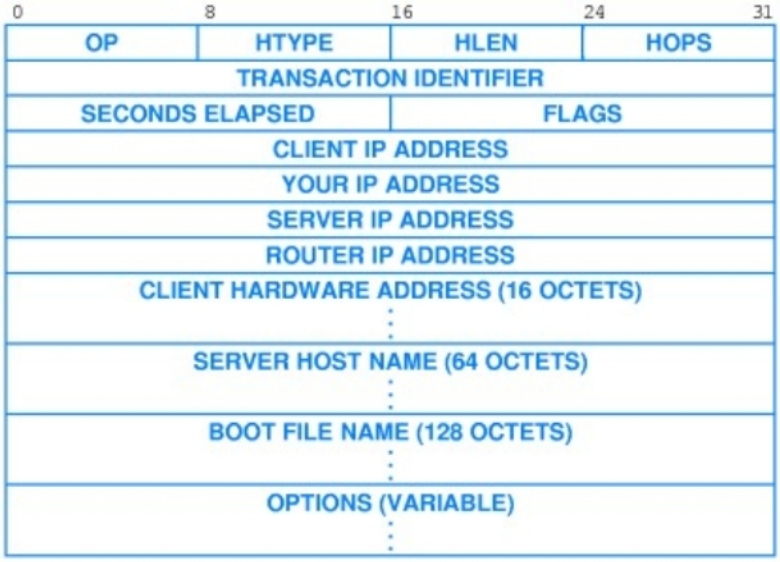
Figure shows the format of a DHCP message. The message is actually sent using a protocol called the User Datagram Protocol (UDP) that runs over IP. UDP does in this context is to provide a demultiplexing key that says, "This is a DHCP packet."

DHCP is derived from an earlier protocol called BOOTP, and some of the packet fields are thus not strictly relevant to host configuration.



- Operation : *1 (Request), 2(Reply)*
- Hardware Type: *1 (for Ethernet)*
- Hardware address length: *6 (for Ethernet)*
- Hop count: *set to 0 by client*
- Transaction ID: *Integer (used to match reply to response)*
- Seconds: *number of seconds since the client started to boot*
- ciaddr : client IP address
- yiaddr : your IP address
- siaddr : server IP address
- giaddr : gateway IP address

- chaddr : client's hardware address(Ethernet address)
- sname : server host name
- file : boot file name
- *client fills in the information that it has, leaves rest blank*

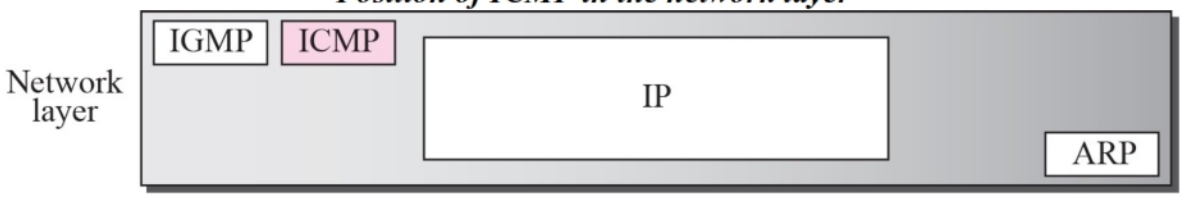


ICMP (Internet Control Message Protocol)

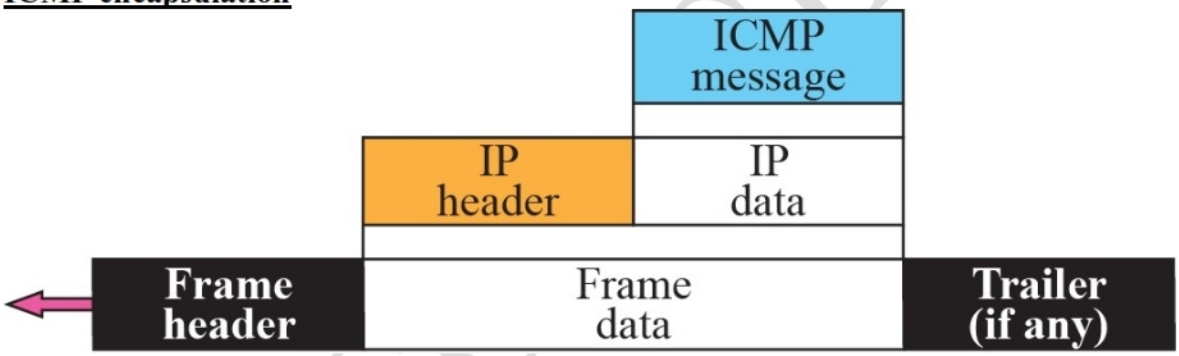
ICMP is used for error and control information

- Defines a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully
- ***ICMP always reports error messages to the original source.***

Position of ICMP in the network layer



ICMP encapsulation



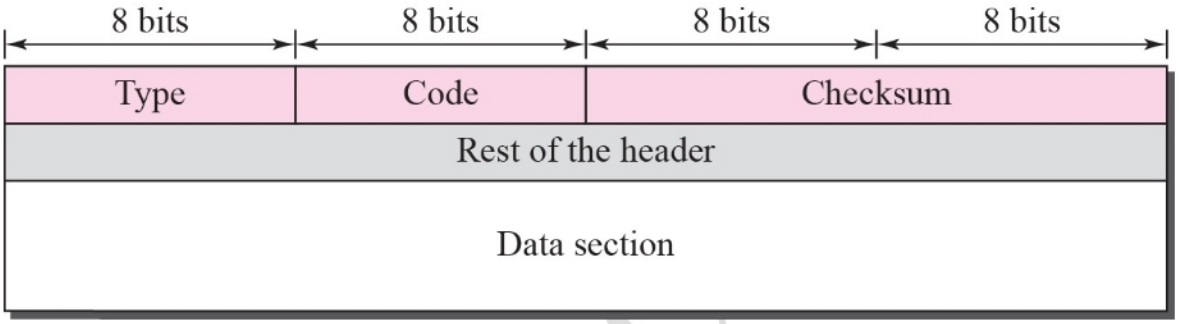
ICMP messages are divided into two broad categories:

1. error-reporting messages
   - The error-reporting messages report problems that a router or a host (destination) may encounter when it processes an IP packet.
2. query messages.
   - The query messages, which occur in pairs, help a host or a network manager get specific information from a router or another host.

ICMP messages

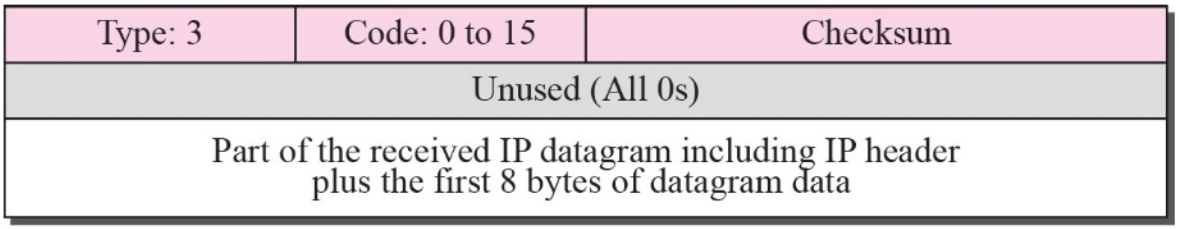
Category	*Type*	*Message*
Error-reporting messages	3	Destination unreachable
	4	Source quench
	11	Time exceeded
	12	Parameter problem
	5	Redirection
Query messages	8 or 0	Echo request or reply
	13 or 14	Timestamp request or reply

General format of ICMP messages



Destination-unreachable

When a router cannot route a datagram or a host cannot deliver a datagram, the datagram is discarded and 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

A source-quench message informs the source that a datagram has been discarded due to congestion in a router or the destination host. The source must slow

down the sending of datagram's until the congestion is relieved.

Type: 4	Code: 0	Checksum
Unused (All 0s)		
Part of the received IP datagram including IP header plus the first 8 bytes of datagram data		

Time Exceeded

When the final destination does not receive all of the fragments in a set time, it discards the received fragments and sends a time-exceeded message to the original source.

Type: 11	Code: 0 or 1	Checksum
Unused (All 0s)		
Part of the received IP datagram including IP header plus the first 8 bytes of datagram data		

Parameter-problem

Any ambiguity in the header part of a datagram can Create serious problems as the datagram travels through the Internet. 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.

Type: 12	Code: 0 or 1	Checksum
Pointer	Unused (All 0s)	
Part of the received IP datagram including IP header plus the first 8 bytes of datagram data		

Redirection

ICMP Redirect is sent by a router (R1) to the sender of an IP datagram (host) when the datagram should have been sent to a different router (R2)

Type: 5	Code: 0 to 3	Checksum
IP address of the target router		
Part of the received IP datagram including IP header plus the first 8 bytes of datagram data		

Echo-request and echo-reply

- Echo-request and echo-reply messages can be used by network managers to check the operation of the IP protocol.
- Echo-request and echo-reply messages can test the reachability of a host. This

is usually done by invoking the ping command.

Type 8: Echo request
Type 0: Echo reply

Type: 8 or 0	Code: 0	Checksum
Identifier		Sequence number
Optional data Sent by the request message; repeated by the reply message		

Timestamp-request and timestamp-reply message

Timestamp-request and timestamp-reply messages can be used to calculate the round-trip time between a source and a destination machine even if their clocks are not synchronized.

Type 13: request
Type 14: reply

Type: 13 or 14	Code: 0	Checksum
Identifier		Sequence number
Original timestamp		
Receive timestamp		
Transmit timestamp		

Virtual Networks and Tunnels

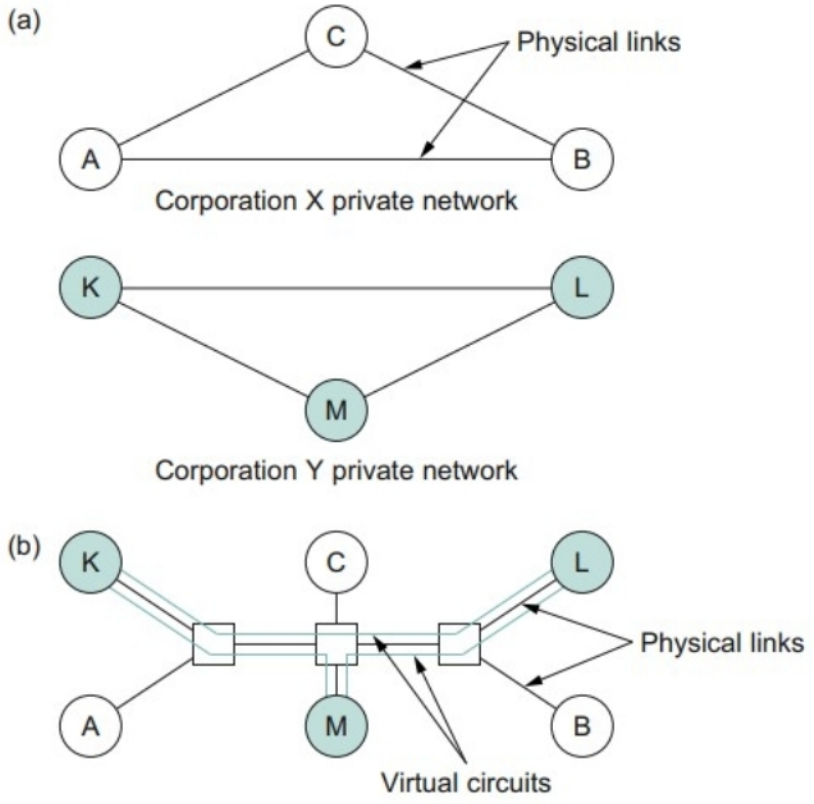
There are many situations where more controlled connectivity is required. An important example of such a situation is the virtual private network (VPN).

The term VPN is heavily overused and definitions vary, but intuitively we can define a VPN by considering first the idea of a private network. Corporations with many sites often build private networks by leasing transmission lines from the phone companies and using those lines to interconnect sites.

In such a network, communication is restricted to take place only among the sites of that corporation, which is often desirable for security reasons. To make a private network virtual, the leased transmission lines—which are not shared with any other corporations—would be replaced by some sort of shared network. A virtual circuit (VC) is a very reasonable replacement for a leased line because it still provides a logical point-to-point connection between the corporation's sites.

For example, if corporation X has a VC from site A to site B, then clearly it can send packets between sites A and B. But there is no way that corporation Y can get its packets delivered to site B without first establishing its own virtual circuit to site B, and the establishment of such a VC can be administratively prevented, thus preventing unwanted connectivity between corporation X and corporation Y.

Figure 3.26(a) shows two private networks for two separate corporations. In Figure 3.26(b) they are both migrated to a virtual circuit network.



■ FIGURE 3.26 An example of virtual private networks: (a) two separate private networks; (b) two virtual private networks sharing common switches.

The limited connectivity of a real private network is maintained, but since the private networks now share the same transmission facilities and switches we say that two virtual private networks have been created. In Figure 3.26, a virtual circuit network (using Frame Relay or ATM, for example) is used to provide the controlled connectivity among sites. It is also possible to provide a similar function using an IP network—an internetwork—to provide the connectivity.

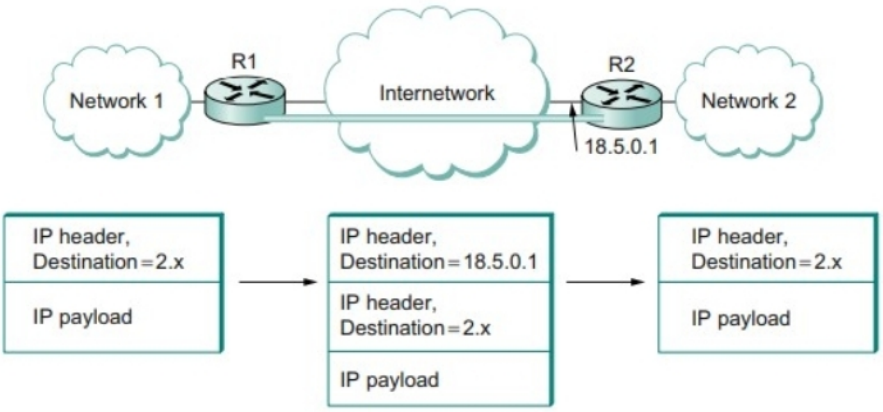
However, we cannot just connect the various corporations' sites to a single internetwork because that would provide connectivity between corporation X and

corporation Y, which we wish to avoid. To solve this problem, we need to introduce a new concept, the IP tunnel.

We can think of an IP tunnel as a virtual point-to-point link between a pair of nodes that are actually separated by an arbitrary number of networks. The virtual link is created within the router at the entrance to the tunnel by providing it with the IP address of the router at the far end of the tunnel.

Whenever the router at the entrance of the tunnel wants to send a packet over this virtual link, it encapsulates the packet inside an IP datagram. The destination address in the IP header is the address of the router at the far end of the tunnel, while the source address is that of the encapsulating router.

In the forwarding table of the router at the entrance to the tunnel, this virtual link looks much like a normal link. Consider, for example, the network in Figure 3.27. A tunnel has been configured from R1 to R2 and assigned a virtual interface number of 0. The forwarding table in R1 might therefore look like Table 3.8



■ **FIGURE 3.27** A tunnel through an internetwork. 18.5.0.1 is the address of R2 that can be reached from R1 across the internetwork.

Table 3.8 Forwarding Table for Router R1 in Figure 3.27

NetworkNum	NextHop
1	Interface 0
2	Virtual interface 0
Default	Interface 1

R1 has two physical interfaces. Interface 0 connects to network 1; interface 1 connects to a large internetwork and is thus the default for all traffic that does not match something more specific in the forwarding table. In addition, R1 has a virtual interface, which is the interface to the tunnel. Suppose R1 receives a packet from network 1 that contains an address in network 2. The forwarding table says this packet should be sent out virtual interface 0. In order to send a packet out this interface, the router takes the packet, adds an IP header addressed to R2, and then proceeds to forward the packet as if it had just been received. R2's address is 18.5.0.1; since the network number of this address is 18, not 1 or 2, a packet destined for R2 will be forwarded out the default interface into the internetwork.

Once the packet leaves R1, it looks to the rest of the world like a normal IP packet destined to R2, and it is forwarded accordingly. All the routers in the internetwork forward it using normal means, until it arrives at R2. When R2 receives the packet, it finds that it carries its own address, so it removes the IP header and looks at the payload of the packet. What it finds is an inner IP packet whose destination address is in network 2. R2 now processes this packet like any other IP packet it receives. Since R2 is directly connected to network 2, it forwards the packet on to that network. Figure 3.27 shows the change in encapsulation of the packet as it moves across the network.

Tunneling does have its downsides.

One is that it increases the length of packets; this might represent a significant waste of bandwidth for short packets. Longer packets might be subject to fragmentation, which has its own set of drawbacks. There may also be performance implications for the routers at either end of the tunnel, since they need to do more work than normal forwarding as they add and remove the tunnel header

Finally, there is a management cost for the administrative entity that is responsible for setting up the tunnels and making sure they are correctly handled by the routing protocols.

UNIT III ROUTING

Routing – Network as Graph - Distance Vector – Link State – Global Internet –Subnetting - Classless Routing (CIDR) - BGP- IPv6 – Multicast routing - DVMRP- PIM.

ROUTING:

- In a network there are multiple routes available between a source and a destination **The process of finding the shortest route from the source to a destination is defined as routing.**
- The routing table in a router is the table which contains the shortest route to reach a destination.
- It is built up by the routing algorithms. It generally contains mappings from network numbers to next hops.

Forwarding versus Routing

Forwarding:

Used to o select an output port based on destination address and routing table

Routing:

Process by which routing table is built

Forwarding table VS Routing table

Forwarding table

- Used when a packet is being forwarded and so must contain enough information to accomplish the forwarding function
- A row in the forwarding table contains the mapping from a network number to an outgoing interface and some MAC information, such as Ethernet Address of the next hop. (Shown in fig. b)

Routing table

- Built by the routing algorithm
- Generally, contains mapping from network numbers to next hops.(shown in fig. a)

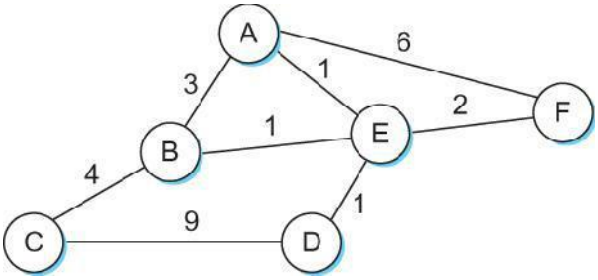
(a)

Prefix/Length	Next Hop
18/8	171.69.245.10

(b)

Prefix/Length	Interface	MAC Address
18/8	if0	8:0:2b:e4:b:1:2

Network as a Graph



- The basic problem of routing is to find the lowest-cost path between any two nodes
- Where the cost of a path equals the sum of the costs of all the edges that make up the path
- For a simple network, we can calculate all shortest paths and load them into some nonvolatile storage on each node.
- Such a static approach has several shortcomings
- It does not deal with node or link failures
- It does not consider the addition of new nodes or links
- It implies that edge costs cannot change

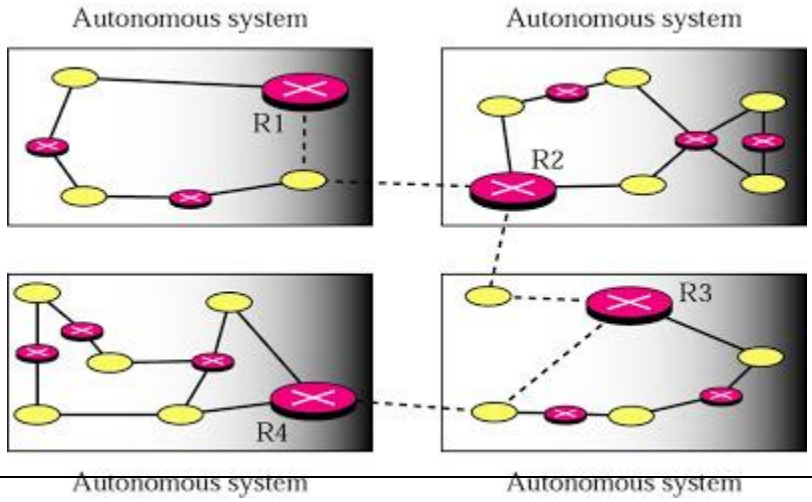
Hence we need a solution in the form of dynamic protocols (implemented with routing algorithms) to find shortest route between nodes.

Before starting with it, we need to know that Internet is a Collection of Millions of Networks and we can't represent the entire Internet as a single entity (single network/graph) to find the shortest routes between nodes. The Internet is divided into blocks termed as Autonomous Systems.

Grouping of Collection of Networks in Internetworks: Autonomous system

An autonomous system (AS) is a network or a collection of networks that are all managed and supervised by a single entity or organization.It is a group of networks and routers under authority of a single administrator. An AS is a heterogeneous network typically governed by a large enterprise.

The number of unique autonomous networks in the routing system of the Internet exceeded 5,000 in 1999, 30,000 in late 2008, 35,000 in mid-2010, 42,000 in late 2012, 54,000 in mid-2016 and 60,000 in early 2018.



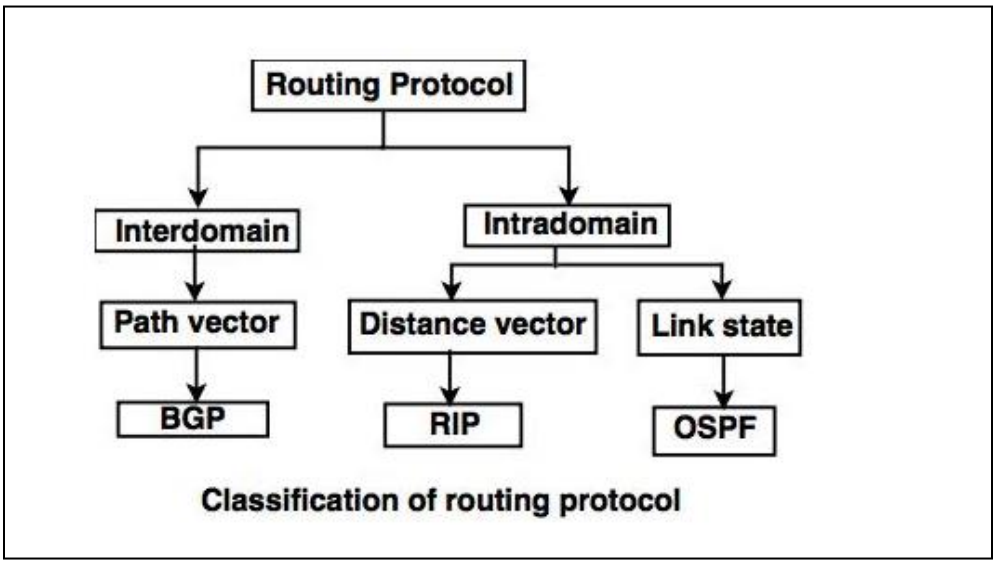
Since the Internet is divided into Autonomous Systems(AS), it is necessary to perform the routing process within an AS and between ASs.

Intra-Domain Routing Protocols (Interior Gateway Protocols):

- Used to construct routing table within an Autonomous System.
- RIP (Routing Information Protocol) is based on Distance Vector Routing
- OSPF (Open Shortest Path First Protocol is based on Link State Routing

Inter-Domain Routing Protocols (Exterior Gateway Protocols):

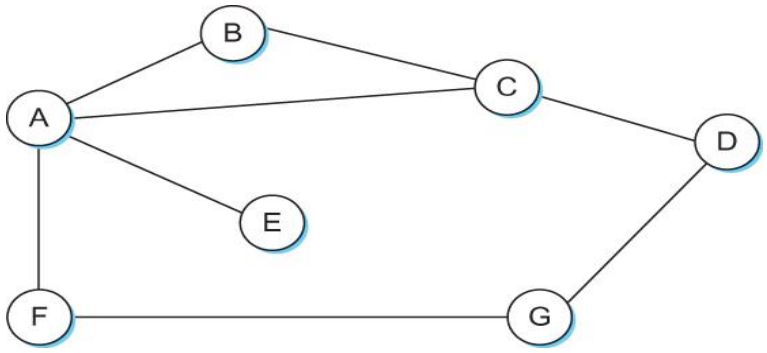
- Used to construct routing table between Autonomous Systems
- BGP(Border Gateway Protocol) is based on Path Vector Routing.



Classification of routing protocol

DISTANCE VECTOR ROUTING

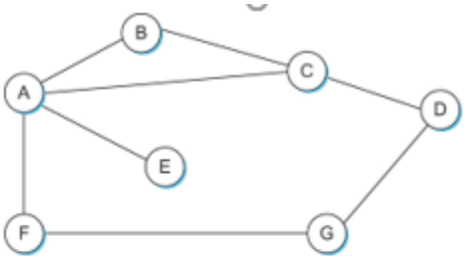
- Working Principle: Each node(router) shares the entire routing information (entire AS) to its neighbors periodically.
- That is, each node constructs a one dimensional array (a vector) containing the "distances" (costs) to all other nodes and distributes that vector to its immediate neighbors
- Starting assumption is that each node knows the cost of the link to each of its directly connected neighbors.
- Consider the below example,



The below table shows the global view of all vectors at each node, first row is the initial vector at router A and so on for each router.

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

The Initial Routing Table of Router A is Shown below,



Destination	Cost	NextHop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—

Initial routing table at node A

Whenever a node receives the routing information from another node, it applies an updating algorithm to update its routing table,

Before applying the algorithm, it does two steps,

- Let 'D' represent each entry in the received Routing Information.
- **Increment the cost of each entry by 1.**
- Let 'c' be the node from which the routing information came.
- **Add a column with a next hop value to be 'c' from which the routing information came.**
- Let 'A' represent the Router.

The Router updates its own table according to the following three rules:

1. New destination: D is a previously unknown destination. Router A adds ⟨D,cost,c⟩ to its routing table.
2. Lower cost: D is a known destination with entry ⟨D,cost(old),c⟩, but the new total cost c is less than the old. A switches to the cheaper route, updating its entry for D to ⟨D,cost(new),c⟩. otherwise it retains the old entry.
3. Next_hop increase: A has an existing entry ⟨D,cost(old),c⟩and the new total cost c is greater than the old cost. Because this is a cost increase from the neighbor c that A is currently using to reach D, A must incorporate the increase in its table. A updates its

Eg:(follow class notes)

After receiving the distance vector of its neighbors the routing table at node A is as follows,

Destination	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Final routing table at node A

After a few exchanges of information between neighbors, all nodes have consistent routing table with correct distance information. The process of getting consistent routing information to all the nodes is calledconvergence.

The following figure shows the final distances stored at each node.

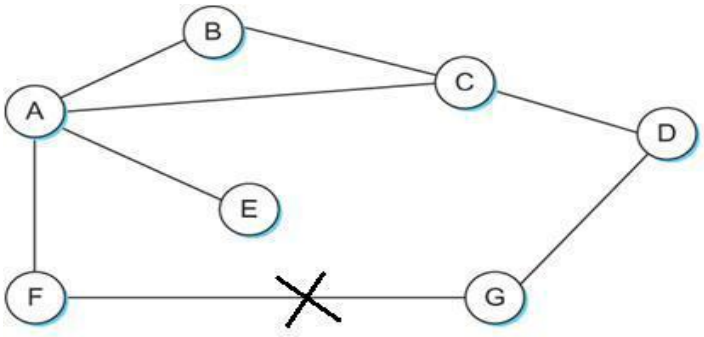
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Final distances stored at each node (global view)

There are two different conditions under which a node decides to send a routing table to neighbors

1. **Periodic update**
   Each node automatically sends an update message every seconds orminutes even though there is no change
2. **Triggered update**
   - whenever a node's routing table changes, it sends its updated distance information to its neighbors.
   - Triggered updates are sent generally, whenever a link fails and causes changes in the routing table.

Link Failure example

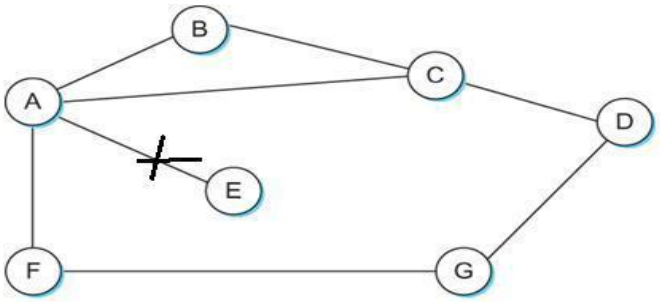


Consider, in this F detects its link to G is failed.

When a node detects a link failure (G)

1. F detects that link to G has failed
2. F sets distance to G to infinity and sends update to A
3. A sets distance to G to infinity since it uses F to reach G
4. A receives periodic update from C with 2-hop path to G
5. A sets distance to G to 3 and sends update to F
6. F decides it can reach G in 4 hops via A

Link Failure with Count-to-infinity problem



In this example, consider the link A-E fails.

1. Slightly different circumstances can prevent the network from stabilizing
   a. Suppose the link from A to E goes down
   b. In the next round of updates, A advertises a distance of infinity to E, but B and C advertise a distance of 2 to E
2. Depending on the exact timing of events, the following might happen
   a. Node B, upon hearing that E can be reached in 2 hops from C, concludes that it can reach E in 3 hops and advertises this to A
   b. Node A concludes that it can reach E in 4 hops and advertises this to C
   c. Node C concludes that it can reach E in 5 hops; and so on.
   d. This cycle stops only when the distances reach some number that is large enough to be considered infinite

Partial solutions to Count to infinity problems:

1. Use some relatively small number as an **approximation of infinity.**
   - For example, the maximum number of hops to get across a certain network is never going to be **more than 16.**

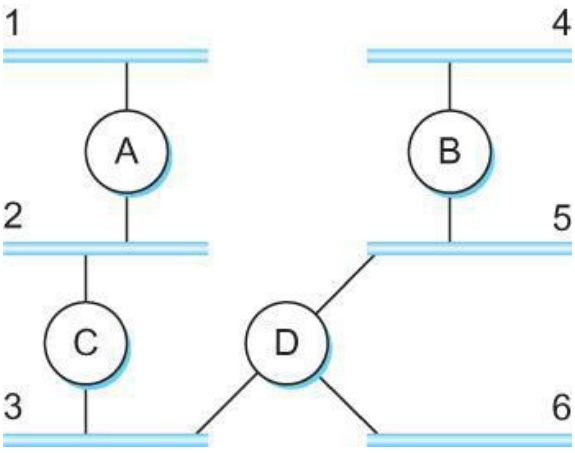
2. One technique to improve the time to stabilize routing is called ***splithorizon*** *(means*Never sent same information back to the interface itcame from)
   - When a node sends a routing update to its neighbors, it does not send those routes it learned from each neighbor back to that neighbor
   - For example, if B has the route (E, 2, A) in its table, then it knows it must have learned this route from A, and so whenever B sends a routing update to A, it does not include the route (E, 2) in that update

3. In a stronger version of split horizon, called ***split horizon withpoison reverse.***
   - B actually sends that back route to A, but it puts negative information in the route to ensure that A will not eventually use B to get to E
   - For example, B sends the route (E, ∞) to A

Routing Information Protocol (RIP)

- Widely used routing protocol in IP networks
- It is a routing protocol built on distance vector algorithm
- Routers running RIP actually advertise distances to networks. They send periodic updates every 30 seconds and triggered updates when the routing table changes.
- RIP uses link costs equal to 1 and uses 16 to represent infinity. So a network running RIP must have a maximum hops of 16.

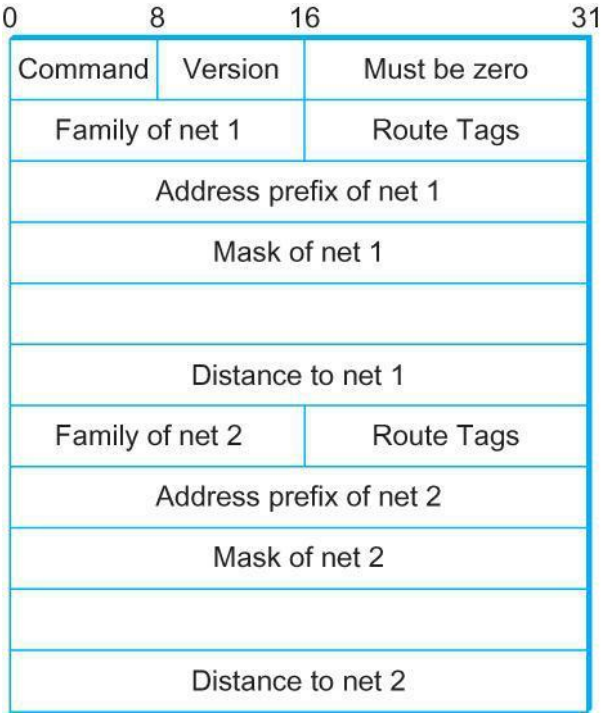
The following figure shows an example network running on RIP.



In this example each network, Router C advertises to router A that it can reach,
Nw 2 and 3 at cost 0
Nw 5 and 6 at cost 1
Nw 4 at cost 2

RIP Packet Format

RIP version 2 supports CIDR.
RIP messages are encapsulated in a UDP datagram
RIP uses the services of UDP on well-known port 520



Command - request or response
version - 2
must be zero - unused
Address - IP address
Family - network address designed to carry information to different Protocols
Distance - metric value that determines how many hops to reach its destination
(1 to 15 are valid routes, 16 is unreachable)
mask - subnet masking
next hop - indicates the IP address of the next hop

Route tag- Distinguishes between internal & external routes (internal routes are learned by diff protocols like RIP, OSPF etc)
(external routes are learn using only one protocol. Eg: BGP)

LINK STATE ROUTING

- Link-state routing is the second major class of intra-domain routing protocol.
- The starting assumptions for link-state routing are rather similar to those for distance vector routing.
- The basic idea behind link-state protocols is very simple: Everynode nowshow to reach its directly connected neighbors, and if we make sure that the totality of this knowledge is disseminated to every node, then every node will have enough knowledge of the network to build a complete map of the network.
- **Working Principle: Each node shares the Routing Information about the neighbors to all the other nodes periodically.**

LINK STATE PACKET:

- Each router creates a link state packet (LSP) which contains names (e.g. network addresses) and cost to each of its neighbours.
- Information in LSP:
    - id of the node that created the LSP
    - cost of link to each directly connected neighbor
    - sequence number (SEQNO)
    - time-to-live (TTL) for this packet
- The LSP is transmitted to all other routers, who each update their own records
- When a router receives LSPs from all routers, it can use (collectively) that information to construct the routing table.

How LSP is shared to all routers? Reliable Flooding

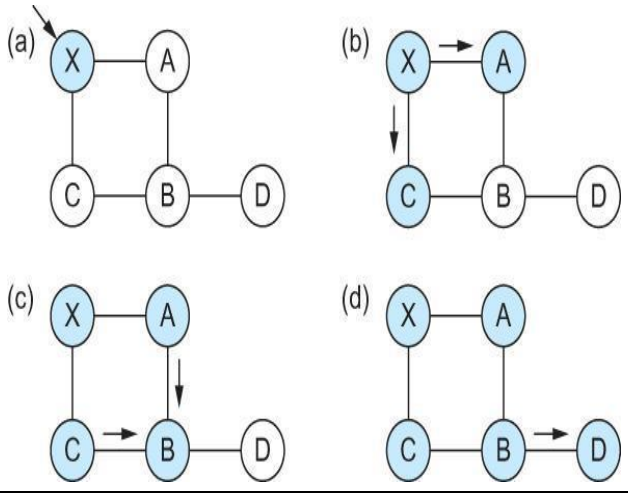
- *Reliable flooding* is the process of making sure that all the nodesparticipating in the routing protocol get a copy of the link-state information from all the other nodes.
    - As the term ―flooding suggests, the basic idea is for a node to send its link-state information out on all of its directly connected links, with each node that receives this information forwarding it out on all of *its* links.
- Initially each node knows only the state of the link to each of its neighbor.
- The information of each node is put into update packet called as Link State Packet (LSP) and its flooded to all other packets. Ie This process continues until the information has reached all the nodes in the network.

In short the process is can be summarized as,

- store most recent LSP from each node
- forward LSP to all nodes but one that sent it
- generate new LSP periodically; increment SEQNO
- start SEQNO at 0 when reboot
- decrement TTL of each stored LSP; discard when TTL=0

In flooding, each node forwards the LSP to all neighbors except the one from which the LSP was received.

EXAMPLE: LSP FLOODED IN A SMALL NETWORK



Flooding of link-state packets.

(a) LSP arrives at node X;
(b) X floods LSP to A and C;
(c) A and C flood LSP to B (but not X);
(d) flooding is complete

Each node becomes shaded as it stores the new LSP. In Figure (a) the LSP arrives at node X, which sends it to neighbors A and in Figure (b) A and C do not send it back to X, but send it on to B. Since B receives two identical copies of the LSP, it will accept whichever arrived first and ignore the second as a duplicate. It then passes the LSP on to D, who has no neighbors to flood it to, and the process is complete.

Just as in RIP, each node generates LSPs under two circumstances. Either the expiry of a periodic timer or a change in topology can cause a node to generate a new LSP.

LSP contains sequence numbers that make it possible to distinguish new LSP from old one.

Route Calculation-SHORTEST PATH ALGORITHM

In practice, each router computes its routing table directly from the LSPs it has collected.

Once a node has a copy of LSP from every other node, it can compute a complete map of network topology and finds the shortest route (routing table) using Dijkstra's algorithm called the *forward search* algorithm.

- Specifically, each router maintains two lists, known as
  1. **Tentative**
  2. **Confirmed**
- Each of these lists contains a set of entries of the form

  (Destination, Cost, NextHop)

The algorithm

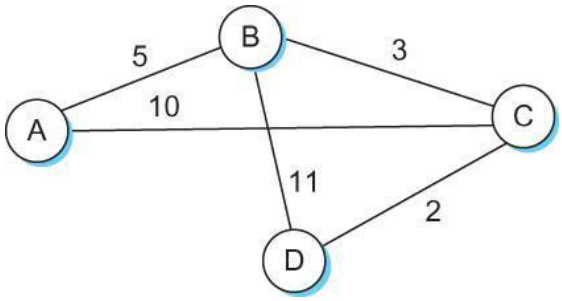
1. Initialize the **Confirmed** list with an entry for myself; this entry has a cost of 0
2. For the node just added to the **Confirmed** list in the previous step, call it node **Next**, select its LSP
3. For each neighbor (Neighbor) of **Next**, calculate the cost (Cost) to reach this Neighbor as the sum of the cost from myself to Next and from Next to Neighbor

a) If Neighbor is currently on neither the **Confirmed** nor the **Tentative** list, then add (Neighbor, Cost, Nexthop) to the **Tentative** list, where Nexthop is the direction I go to reach Next.

b) If Neighbor is currently on the **Tentative** list, and the Cost is less than the currently listed cost for the Neighbor, then replace the current entry with **(Neighbor, Cost, Nexthop)** where Nexthop is the direction I go to reach Next.

4. If the **Tentative** list is empty, stop. Otherwise, pick the entry from the **Tentative** list with the lowest cost, move it to the **Confirmed** list, and return to Step 2.

Consider an example network for link-state routing



The following table list the steps for building routing table for **node D.**

Step	Confirmed	Tentative	Comments
1	(D,0,–)		Since D is the only new member of the confirmed list, look at its LSP.
2	(D,0,–)	(B,11,B) (C,2,C)	D's LSP says we can reach B through B at cost 11, which is better than anything else on either list, so put it on Tentative list; same for C.
3	(D,0,–) (C,2,C)	(B,11,B)	Put lowest-cost member of Tentative (C) onto Confirmed list. Next, examine LSP of newly confirmed member (C).
4	(D,0,–) (C,2,C)	(B,5,C) (A,12,C)	Cost to reach B through C is 5, so replace (B,11,B). C's LSP tells us that we can reach A at cost 12.
5	(D,0,–) (C,2,C) (B,5,C)	(A,12,C)	Move lowest-cost member of Tentative (B) to Confirmed, then look at its LSP.
6	(D,0,–) (C,2,C) (B,5,C)	(A,10,C)	Since we can reach A at cost 5 through B, replace the Tentative entry.
7	(D,0,–) (C,2,C) (B,5,C) (A,10,C)		Move lowest-cost member of Tentative (A) to Confirmed, and we are all done.

The difference between distance vector and link state routing algorithm is– in distance vector, each node talks only to its directly connected neighbors about the distance to all nodes. In link state, each node talks to all other nodes only about the state of its directly connected links.

OSPF:OPEN SHORTEST PATH FIRST PROTOCOL

- One of the most widely used link-state routing protocols is OSPF.
- The first word, —Open, refers to the fact that it is an open, nonproprietary standard, created under the auspices of the IETF. The —SPF part comes from an alternative name for link state routing

OSPF adds quite a number of features to the basic link-state algorithm described above, including the following:

- Authentication of routing messages
- Additional hierarchy
- Load balancing

Authentication of routing messages

Protection against misconfigured routers by providing 8 byte password for authentication

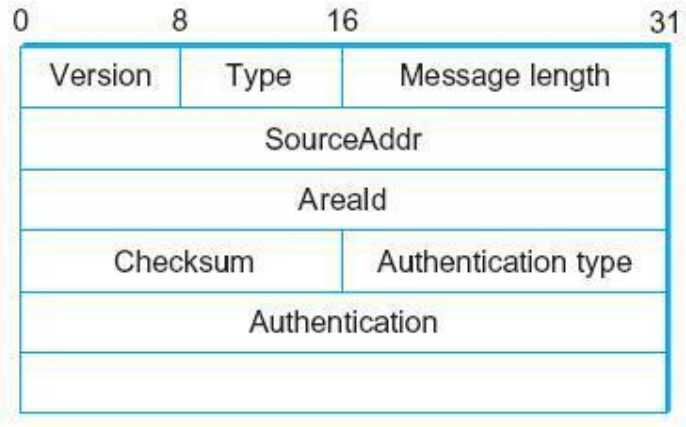
Additional hierarchy

- Introduces additional hierarchy by allowing arouting domain to be portioned into areas.
- Reduces the amount of information that must be transmitted to and stored in each node.
- Router doesn't need to know how to reach each network in its domain; it may know how to get to right area.

Load balancing

OSPF allows distributing traffic among multiple routes of same cost.

OSPF Header Format



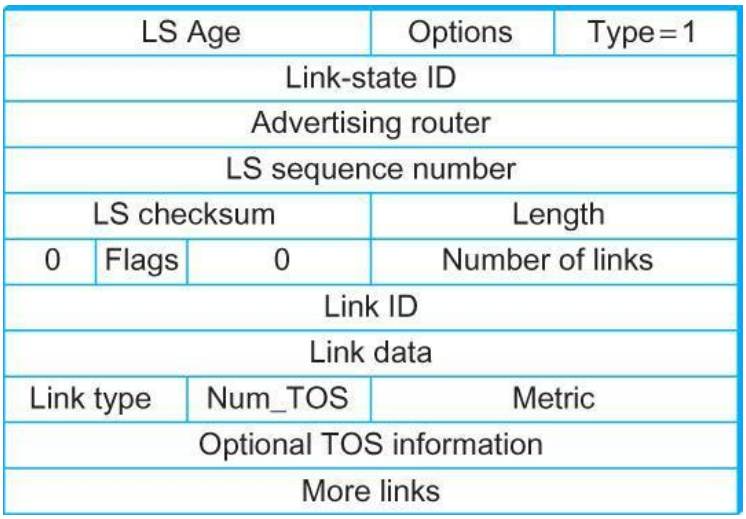
- Version - set to 2
- Type – may take the values 1 through 5
- SourceAddr - identifies the sender of the message
- Authentication type
  - 0 if authentication is used
  - 1 if password is used
  - 2 if cryptographic authentication checksum is used
- Area id – 32 bit identifier of the area in which node is located
- Authentication - password or cryptographic checksum Checksum - the entire packet, except the authentication data, is protected by a 16-bit checksum using the same algorithm as the IP header.

Type - OSPF Message Types

Type 1 -> "hello" msg (notficationmsg to nofity that it is alive)
Type 2 -> request
Type 3 -> response
Type 4 –> send
Type 5 -> acknowledge the receipt of link state msg

The basic building blocks of link state messages is known as link state advertisement (LSA). One message may contain one or many LSAs.

The packet format for type1 link-state advertisement is as follows:-

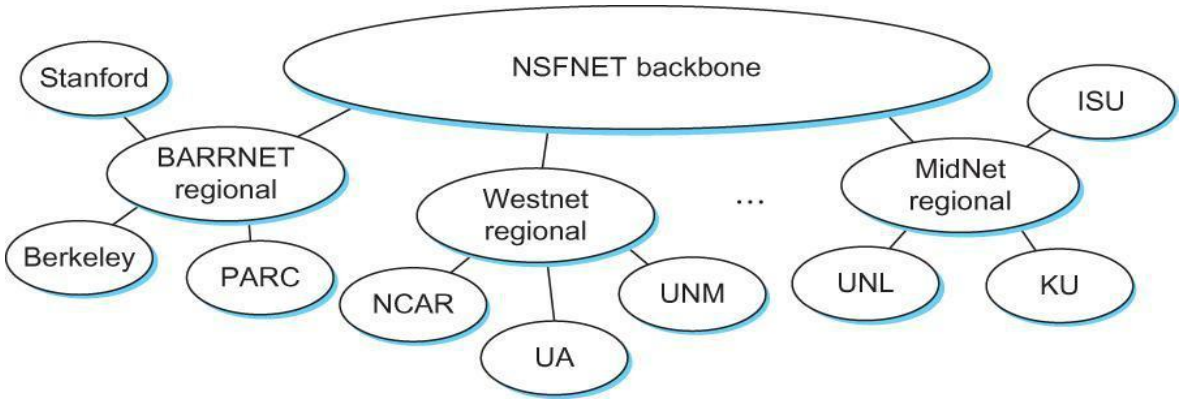


OSPF Link State Advertisement

- LSAge – Equivalent to time to live(TTL), LSA expires when the age reaches the maximum value (difference is TTL counts down, LSAge counts Up)
- Type = 1 LSAs advertise the cost of a link between routers.
- [Type= 2 are used to advertise the networks to which the advertising routers are connected
- Other Types are used to support additional hierarchy]
- Link state ID & Advertising router
  -identical in Type 1 LSA, it should be unique in routing domain
  - 32 bit identifier for a router that created this LSA
- LS sequence number – to detect old or duplicate LSAs
- LS Checksum – similar to the other checksum ,
  - used to verify data
  - covers all the fields except LSAge
  - no need to compute checksum everytimeLSAge is incremented
- Length – length in bytes of complete LSA
- TOS - Type of Service
- Link Type
  - Type 1 point to point
  - Type 2 Transient (in between nodes)
  - Type 3Stub
  - Type 4 Virtual
- Metric - cost of link

GLOBAL INTERNET

- Internetworking is a heterogeneous of networks with tens of thousands of networks connected to it.
- The Global Internet



The tree structure of the Internet in 1990

- As the internet grows it poses,

- large no of network numbers are used, routers have to maintain a large routing table

Global internet is not just a random interconnection of Ethernets, but instead it takes on a shape that reflects the fact that it interconnects many different organizations.

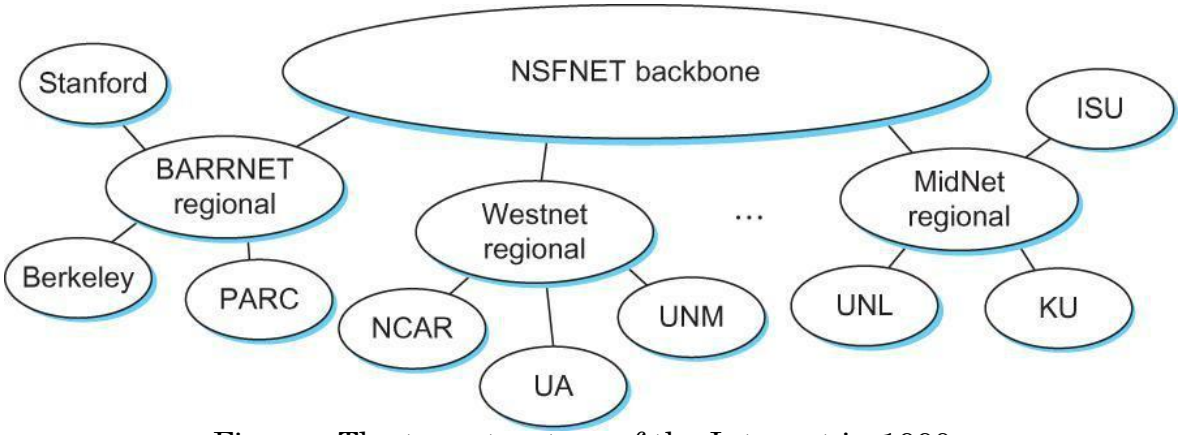


Figure : The tree structure of the Internet in 1990

Salient features of this topology is,

- It has end user sites that connect to service provider etwork
- Eg: user site – Stanford university
  Service provider network - e.g., BARRNET was a provider network that served sites in that area

Many providers served a limited geographic region and known as RegionalNetworks.

These Regional networks were in turn connected by Nation Wide Backbone –was funded by **NSF (National Science Foundation). Therefore, its called as NSFNET backbone.**

This has some significant consequence in routing.

Eg: AS decide the best routing protocol used in their network.

As the internet grows it poses

- large no of network numbers are used
- routers have to maintain a large routing table
- large amount of IP address space are wasted

To tackle the 2 issues of scalability we use,

- Supernetting or classless routing
- Subnetting

ROUTING AREAS:

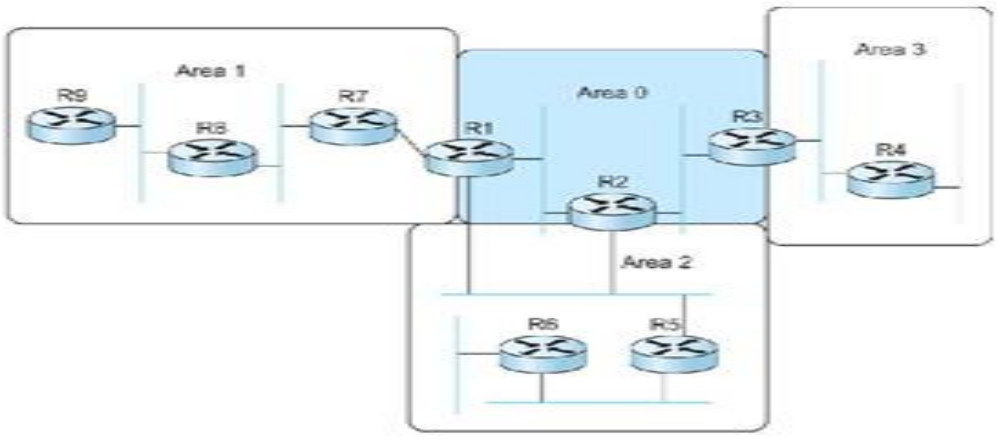
As a first example of using hierarchy to scale up the routing system, we'll examine how link-state routing protocols (such as OSPF and IS-IS) can be used to partition a routing domain into **subdomains called *areas*.**

By adding this extra level of hierarchy, we enable single domains to grow larger without overburdening the routing protocols or resorting to the more complex interdomain routing protocols

Area is a set of routers that are administratively configured to exchange link-state information with each other.

There is one special area—the backbone area, also known as area 0.

An example of a routing domain divided into areas is shown in Figure



- ✓ Routers R1, R2, and R3 are members of the backbone area. They are also members of at least one nonbackbone area.
- ✓ R1 is actually a member of both area 1 and area 2.
- ✓ A router that is a member of both the backbone area and a nonbackbone area is an area border router (ABR). The routers that are at the edge of an AS, which are referred to as AS border routers

Routing within a single area

- ✓ All the routers in the area send link-state advertisements to each other and thus develop a complete, consistent map of the area
- ✓ The link-state advertisements of routers that are not area border routers do not leave the area in which they originated.
- ✓ This has the effect of making the flooding androute calculation processes considerably more scalable.
- ✓ For example, router R4 in area 3 will never see a link-state advertisement from router R8 in area 1. As a consequence, it will know nothing about the detailed topology of areas other than its own.

How does a router in one area determine the right next hop for a packet destined to a network in another area?

The answer to this becomes clear if we imagine the path of a packet that has to travel from one nonbackbone area to another as being split into three parts.

1. First, it travels from its source network to the backbone area
2. then it crosses the backbone,
3. then it travels from the backbone to the destination network

To make this work, the area border routers summarize routing information that they have learned from one area and make it available in their advertisements to other areas.

For example, R1 receives link-state advertisements from all the routers in area 1 and can thus determine the cost of reaching any network in area 1.

When R1 sends link-state advertisements into area 0, it advertises the costs of reaching the networks in area 1 much as if all those networks were directly connected to R1.

This enables all the area 0 routers to learn the cost to reach all networks in area 1. The area border routers (ABR) then summarize this information and advertise it into the non-backbone areas. Thus, all routers learn how to reach all networks in the domain.

In the case of area 2, there are two ABRs and that routers in area 2 will thus have to make a choice as to which one they use to reach the backbone.

Scalability & optimality

When dividing a domain into areas, the network administrator makes a tradeoff between scalability and optimality of routing.

The use of areas forces all packets traveling from one area to another to go via the back- bone area, even if a shorter path might have been available.

For example, even if R4 and R5 were directly connected, packets would not flow between them because they are in different non-backbone areas.

It turns out that the need for scalability is often more important than the need to use the absolute shortest path.

Finally, we note that there is a trick by which network administrators can more flexibly decide which routers go in area
0. This trick uses the idea of a *virtual link* between routers.Such a virtual link is obtained by configuring a router that is not directly connected to area 0 to exchange backbone routing information with a router that is.

For example, a virtual link could be configured from R8 to R1, thus making R8 part of the backbone. R8 would now participate in link-state advertisement flooding with the other routers in area 0.

The cost of the virtual link from R8 to R1 is determined by the exchange of routing information that takes place in area 1. This technique can help to improve the optimality of routing

INTER-DOMAIN ROUTING

- Inter domain routing is used for complex network.
- Internet is organized as autonomous systems (AS) each of which is under the control of a single administrative entity.
- Autonomous System (AS)
  - corresponds to an administrative domain
  - examples: University, company, backbone network

A corporation's internal network might be a single AS, as may the network of a single Internet service provider.

Challenges in Inter-Domain Routing

Perhaps the most important challenge of inter-domain routing today is the need for each AS to determine its own routing*policies*. A key design goal of inter-domain routing is that policies and much more complex ones, should be supported by the inter-domain routing system. We are concerned with reachability than optimality.

- Finding path anywhere close to optimal is considered to be a great achievement.

- Scalability: An Internet backbone router must be able to forward any packet destined anywhere in the Internet

  - ✓ Having a routing table that will provide a match for any valid IP address

- Autonomous nature of the domains
  - ✓ It is impossible to calculate meaningful path costs for a path that crosses multiple ASs
  - ✓ A cost of 1000 across one provider might imply a great path but it might mean an unacceptable bad one from another provid

- Issues of trust
  - ✓ Provider A might be unwilling to believe certain advertisements from provider B

There are two Inter-domain Routing Protocols

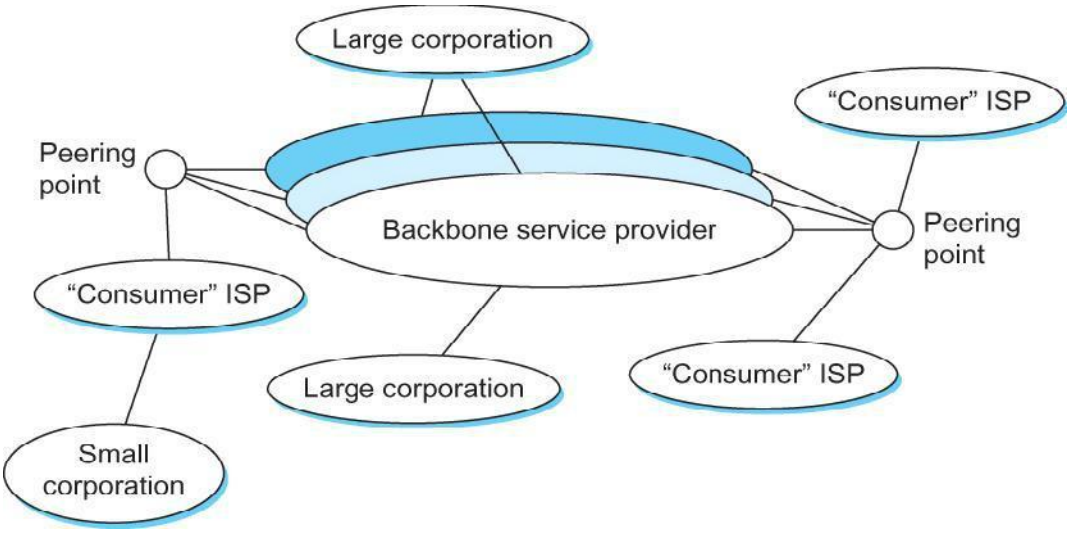
1. Exterior Gateway Protocol (EGP)
2. Border Gateway Protocol (BGP)

- **Exterior Gateway Protocol (EGP)**
  - ✓ Forced a tree-like topology onto the Internet
  - ✓ Did not allow for the topology to become general
    - ➢ Tree like structure: there is a single backbone and autonomous systems are connected only as parents and children and not as peers
- **Border Gateway Protocol (BGP)**
  - ✓ BGP version 4 is often regarded as one of the more complex parts of the internet.
  - ✓ BGP makes virtually no assumptions about how autonomous systems are interconnected—they form an arbitrary graph.

BORDER GATEWAY PROTOCOL:

- ➢ BGP is used for routing interconnected set of Autonomous Systems(ASs)
- ➢ It Assumes that the Internet is an arbitrarily interconnected set of ASs.

Figure shows today's multibackbone Internet. today's Internet consists of a richly interconnected set of networks, mostly operated by private companies (ISPs) rather than governments. Many Internet Service Providers (ISPs) exist mainly to provide service to "consumers" (i.e., individuals with computers in their homes), while others offer something more like the old backbone service, interconnecting other providers and sometimes larger corporations. Often, many providers arrange to interconnect with each other at a single *peering point*.



It defines **<u>two types of traffic</u>**

1) local traffic

- as traffic that originates at or terminates on nodes within an AS

2) *transit traffic*

- as traffic that passes through an AS.

<u>Three types of AS are,</u>

- ✓ *<u>Stub AS</u>*: an AS that has only a single connection to one other AS;such an AS will only carry local traffic (*small corporation in the figure of the previous page*).
- ✓ *<u>Multihomed AS</u>*: an AS that has connections to more than one otherAS, but refuses to carry transit traffic (*large corporation at the top inthe figure of the previous page*).
- ✓ *<u>Transit AS</u>*: an AS that has connections to more than one other AS, andis designed to carry both transit and local traffic (*backbone providers inthe figure of the previous page*).

The *<u>goal</u>* of Inter-domain routing is

- ➢ To find any path to the intended destination that is loop free.
- ➢ Paths must be compliant with policies of various ASs along the paths.

<u>Basics of BGP</u>

1. Each AS has one or more border routers through which packets enter and leave the AS.
2. A Border Router is simple an IP router that is charged with the task of forwarding packet between autonomous systems.

Each AS has

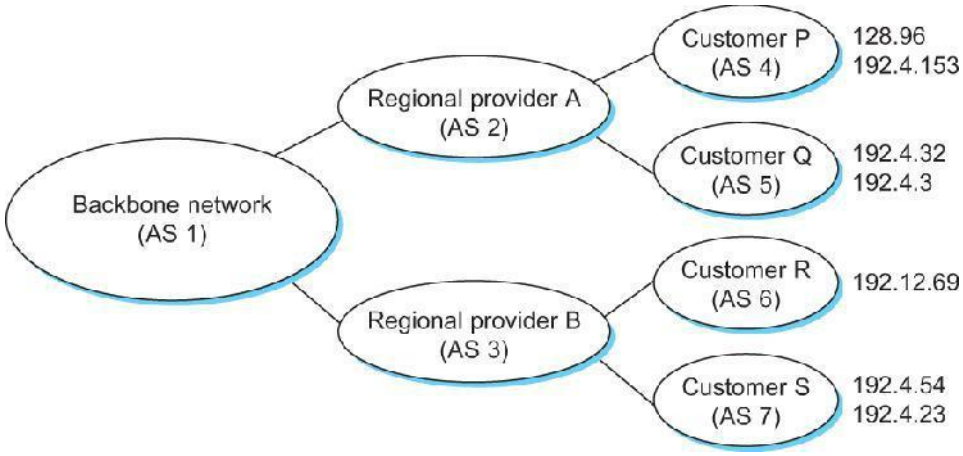
- One BGP *speaker* that advertises:
  - ✓ local networks
  - ✓ other reachable networks (transit AS only)
  - ✓ gives *path* information

- In addition to the BGP speakers, the AS has one or more border "gateways" which need not be the same as the speakers
- The border gateways are the routers through which packets enter and leave the AS
- BGP does not belong to either of the two main classes of routing protocols (distance vectors and link-state protocols).

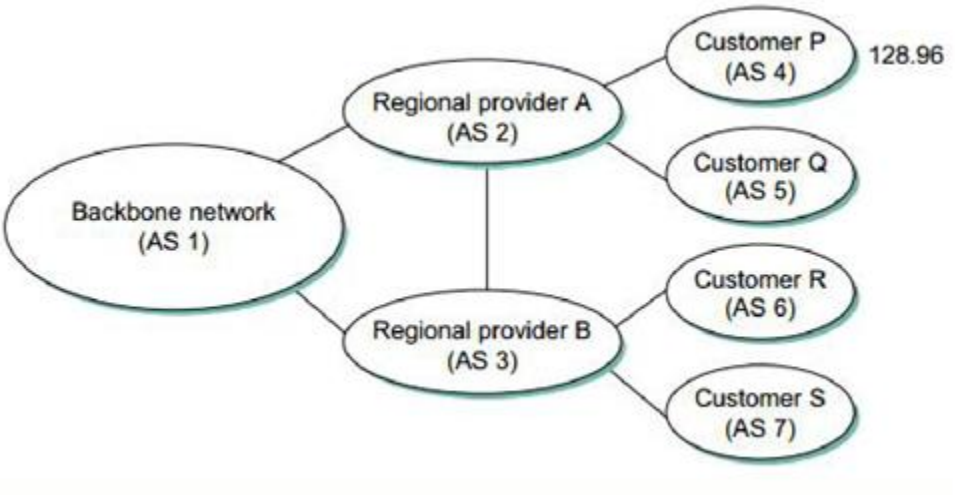
- BGP advertises *complete paths* as an enumerated lists of ASs to reach a particular network. It is sometimes called a *path-vector* protocol for this reason.

  consider the very simple example network in Figure. Assume that the providers are transit networks, while the customer networks are stubs.

An Example of a network running BGP



- A BGP speaker for the AS of provider A (AS 2) advertises reachability to P and Q
  - ✓ Network 128.96, 192.4.153, 192.4.32, and 192.4.3, can be reached directly from AS 2.
- Speaker for backbone network then advertises
  - ✓ Networks 128.96, 192.4.153, 192.4.32, and 192.4.3 can be reached along the path <AS 1, AS 2>.
- Speaker can also cancel previously advertised paths
- An important job of BGP is to prevent the establishment of looping paths.



In figure only in the addition of an extra link between AS 2 and AS 3, but the effect now is that the graph of autonomous systems has a loop in it.

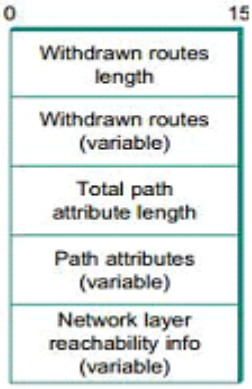
Suppose AS 1 learns that it can reach network 128.96 through AS 2, so it advertises this fact to AS 3, who in turn advertises it back to AS 2. In the absence of any loop prevention mechanism, AS 2 could now decide that AS 3 was the preferred route for packets destined for 128.96.

If AS 2 starts sending packets addressed to 128.96 to AS 3, AS 3 would send them to AS 1; AS 1 would send them back to AS 2; and they would loop forever. This is prevented by carrying the complete AS path in the routing messages.

In this case, the advertisement for a path to 128.96 received by AS 2 from AS 3 would contain an AS path of <AS 3, AS 1, AS 2, AS 4>. AS 2 sees itself in this path, and thus concludes that this is not a useful path for it to use.

In order for this loop prevention technique to work, the AS numbers carried in BGP clearly need to be unique. For example, AS 2 can only recognize itself in the AS path in the above example if no other AS identifies itself in the same way. AS numbers have until recently been 16-bit numbers, and they are assigned by a central authority to assure uniqueness.

BGP-4 Update Packet Format



Given that links fail and policies change, BGP speakers need to be able to cancel previously advertised paths.

This is done with a form of negative advertisement known as a *withdrawn route*. Both positive and negative reachability information are carried in a BGP update message, the format of which is shown in Figure.

Common AS Relationships and Policies

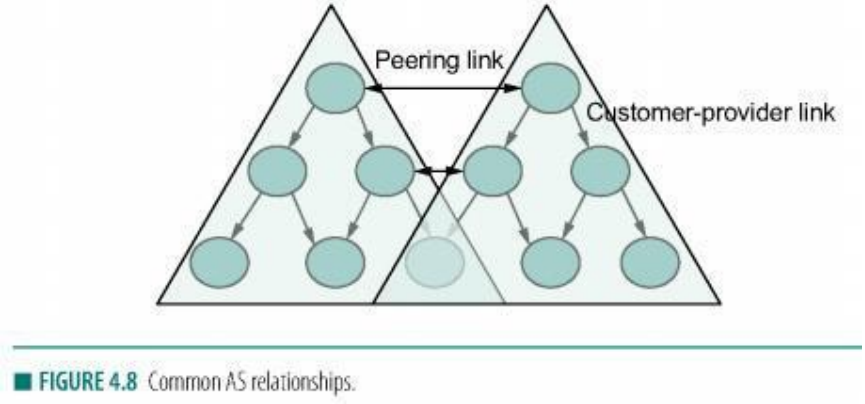


FIGURE 4.8 Common AS relationships.

Three common relationships and the policies are,

1. Provider-Customer

Providers are in business of connecting their customers to rest of internet. A customer might be s smaller ISP. so the common policy is to advertise all the routes I know to my customer, and advertise routes I learn from my customer to everyone.

2. Customer-Provider

Advertise my own prefixes and routes learned from my provider to my customers to my provider, advertises routes learned from my providers to my customers , but don't advertise routes learned from one provider to another provider.

3. Peer

Third option is a symmetrical peering between autonomous systems. policy here is to advertise routes learned from my customers to my peer, advertise routes learned from my peer to my customers, but don't advertise routes from my peer to any provider or vice versa.

Integrating Inter-domain and Intra-domain Routing

Consider, for example, the border router of a provider AS that connects to a customer AS. That router could learn that the network prefix 192.4.54/24 is located inside the customer AS, either through BGP or because the information is con- figured into the border router. It could inject a route to that prefix into the routing protocol running inside the provider AS. This would be an advertisement of the sort, "I have a link to 192.4.54/24 of cost X."

This would cause other routers in the provider AS to learn that this border router is the place to send packets destined for that prefix.

The routers in a backbone network use a variant of BGP called ***interior BGP*(iBGP)**to effectively redistribute the informationthat is learned by the BGP speakers at the edges of the AS to all the other routers in the AS. (The other variant of BGP, discussed above, runs between autonomous systems and is called ***exterior BGP*, or eBGP**).

iBGP enables any router in the AS to learn the best border router to use when sending a packet to any address. At the same time, each router in the AS keeps track of how to get to each border router using a conventional intradomain protocol with no injected information. By combining these two sets of information, each router in the AS is able to determine the appropriate next hop for all prefixes.

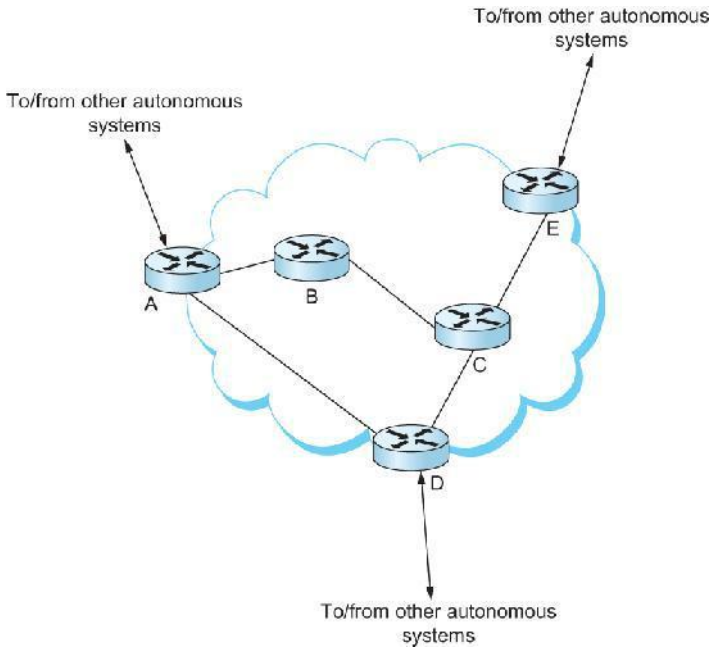


FIGURE 4.9 Example of interdomain and intradomain routing. All routers run iBGP and an intradomain routing protocol. Border routers A, D, and E also run eBGP to other autonomous systems.

Consider the simple example network, rep- resenting a single AS, in Figure 4.9. The three border routers, A, D, and E, speak eBGP to other autonomous systems and learn how to reach variousprefixes. These three border routers communicate with other and with the interior routers B and C by building a mesh of iBGP sessions among all the routers in the AS. Let's now focus in on how router B builds up its complete view of how to forward packets to any prefix.

Look at the table at the top left of Figure 4.10 which shows the information that router B learns from its iBGP sessions. It learns that some prefixes are best reached via router A, some via D, and some via E. At the same time, all the routers in the AS are also running some intra-domain routing protocol such as Routing Information Protocol (RIP) or Open Shortest Path First (OSPF).

(A generic term for intra-domain protocols is an interior gateway protocol, or IGP.) From this completely separate protocol, B learns how to reach other nodes *inside* the domain, as shown in the top right table.

For example, to reach router E, B needs to send packets toward router C. Finally, in the bottom table, B puts the whole picture together, combining the information about external prefixes learned from iBGP with the infor- mation about interior routes to the border routers learned from the IGP.

Thus, if a prefix like 18.0/16 is reachable via border router E, and the best interior path to E is via C, then it follows that any packet destined for 18.0/16 should be forwarded toward C. In this way, any router in the AS can build up a complete routing table for any prefix that is reachable via some border router of the AS.

All routers run iBGP and an intradomain routing protocol. Border routers (A, D, E) also run eBGP to other ASs

BGP routing table, IGP routing table, and combined table at router B is,

Prefix	BGP Next Hop
18.0/16	E
12.5.5/24	A
128.34/16	D
128.69./16	A

BGP table for the AS

Router	IGP Path
A	A
C	C
D	C
E	C

IGP table for router B

Prefix	IGP Path
18.0/16	C
12.5.5/24	A
128.34/16	C
128.69./16	A

Combined table for router B

FIGURE 4.10 BGP routing table, IGP routing table, and combined table at router B.

IPV6

The motivation for new version of IP is

- To deal with scaling problem
- To achieve 100 % address utilization efficiency

Historical perspective

- IEFT (Internet Engg Task Force), which is responsible for specification of standards and protocols looked into the problem in 1991
- Since the IP address is carried in the header of every IP packet, increasing the size of the address dictates a change in the packet header.
- The effort to define a new version of IP was known as IP NextGeneration, or IPng.
- An official IP version number was assigned, so IPng is now known as IPv6

In addition to deal with scalable routing and addressing, IPv6 should also

- support for real-time services
- supports multicast
- security support
- auto configuration (i.e., the ability of hosts to automatically configure themselves with such information as their own IP address and domain name)
- End-to-end fragmentation
- Enhanced routing functionality, including support for mobile hosts.

Addresses and Routing

- **IPv6 provides a 128-bit address space.**
- IPv6 can address 3.4×10^{38} nodes,
- IPv6 address space is predicted to provide over 1500 addresses per square foot of the earth's surface

Address Space Allocation

- **IPv6 also follows CIDR like IPv4, so IPv6 addresses are classless**
- The address prefix assignments for IPv6 is as follows,

At the time of writing, IPv6 unicast addresses are being allocated from the block that begins 001, with the remaining address space—about 87%—being reserved for future use.

Prefix	Use
00...0 (128 bits)	Unspecified
00...1 (128 bits)	Loopback
1111 1111	Multicast addresses
1111 1110 10	Link local unicast
1111 1110 11	Site local unicast
Everything else	Global unicast

Fig : Address Prefix Assignments for IPv6

Multicast address

- Similar to class D address in IPV4
- The multicast address space is for multicast,
- Start with a byte of all 1s.

Link local unicast

- Enable a host to construct an address that will work on the network to which it is connected, without being concerned about the global uniqueness of the address.
- This may be useful for autoconfiguration

Site local unicast

- Intended to allow valid address to be constructed on site.
- (e.g., a private corporate network) that is not connected to the larger Internet;

global unicast address

- Some important special types of addresses.
- Two special address types have uses in the IPv4-to-IPv6 transition.
  1. IPv4 compatible IPv6 address
  2. IPv4 mapped IPv6 address

IPv4 compatible IPv6 address

- One type, the *IPv4-compatible IPv6 address*, is used for devices that are compatible with both IPv4 and IPv6;
- Begins with 96 bits zeros then followed by 32 bits IPv4 address

IPv4 mapped IPv6 address

- A node that is only capable of understanding IPv4 can be assignedan IPv4-mapped IPv6 address by prefixing the 32-bit IPv4 address with 2 bytes of all 1s and then zero-extending the result to 128 bits.
- Begins with 80 bits zeros, followed by 16 bits of ones and 32 bits IPv4 address

Address Notation

The standard representation is

x:x:x:x:x:x:x:x

where each –x is a hexadecimal representation of a 16-bit piece of the

Example IPV6 Address

47CD:1234:4422:ACO2:0022:1234:A456:0124

If we have an address with a large number of contiguous 0s can be written more compactly by omitting all the 0 fields.

Example1

47CD:0000:0000:0000:0000:0000:A456:0124

could be written as 47CD::A456:0124

Example 2

3FFE:085B:1F1F:0000:0000:0000:00A9:1234

could be written as 3FFE:85B:1F1F::A9:1234

- 8 groups of 16-bit hexadecimal numbers separated by ":" & Leading zeros can be removed. **::** = all zeros in one or more group of 16-bit hexadecimal numbers
- Clearly, this formof shorthand can only be used for one set of contiguous 0s in an address to avoid ambiguity.

The two types of IPv6 addresses that contain an embedded IPv4 address have their own special notation that makes extraction of the IPv4 address easier.

For example, the IPv4 -mapped IPv6 address of a host whose IPv4 address was 128.96.33.81 could be written as:: FFFF:128.96.33.81

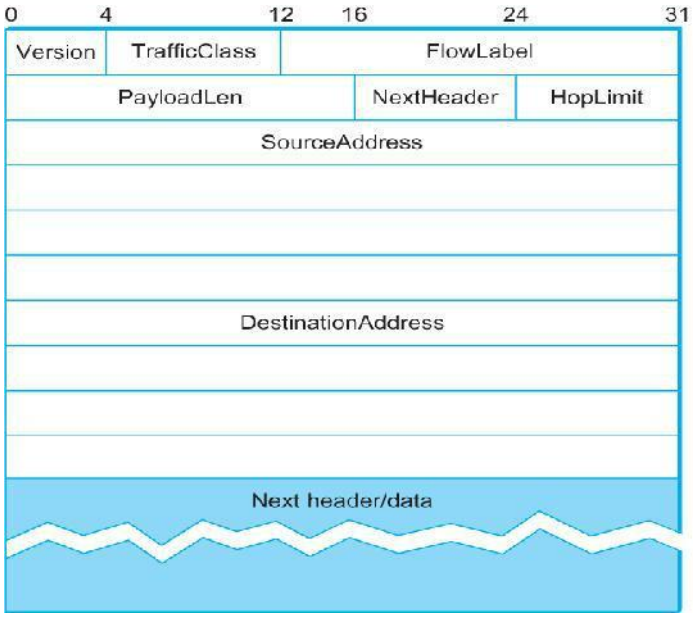
That is, the last 32 bits are written in IPv4 notation, rather than as a pair of hexadecimal numbers separated by a colon.Note that the double colon at the front indicates the leading 0s.

IPv6 provider-based unicast address is as follows:

The Registry ID might be an identifier assigned to a European address registry, with different IDs assigned to other continents or countries**.**

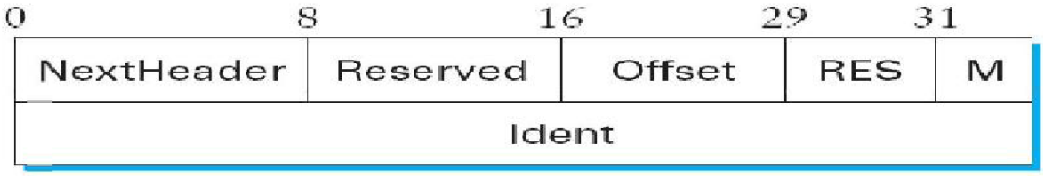
Packet Format

The IPV6 packet header format is as follows



- ✓ Version - set to 6 for IPv6.
- ✓ TrafficClass and FlowLabel– deals with quality of service issues.
- ✓ PayloadLen -Length of the packet excluding the IPv6 header,measured in bytes.
- ✓ NextHeader – if special headers follow IP header, it indicates option.
- ✓ If there are no special headers, it indicates the highlevel protocols running over IP. Eg: TCP and UDP
- ✓ HopLimit - field is simply the TTL of IPv4.

Each Option has its own type of extension header. The IPv6 fragmentationextension header is as follows,



Assuming it is the only extension header present, then the NextHeader field of the IPv6 header would contain the value 44, which is the value assigned to indicate the fragmentation header

- The NextHeader field of the fragmentation header itself contains a value describing the header that follows it. Again, assuming no other extension headers are present, then the next header might be the TCP header, which results in NextHeader containingthe value 6, just as the Protocol field would in IPv4.
- If the fragmentation header were followed by, say, an authentication header, then the fragmentation header's NextHeaderfield would contain the value 51.

MULTICASTING

- IP multicast is a method of sending Internet Protocol (IP) datagrams to a group of interested receivers in a single transmission.
- It is often employed for streaming media applications on the Internet and private networks. The method is the IP-specific version of the general concept of multicast networking.

Overview

- One-to-many
  - ✓ Radio station broadcast
  - ✓ Transmitting news, stock-price
  - ✓ Software updates to multiple hosts
- Many-to-many
  - ✓ Multimedia teleconferencing
  - ✓ Online multi-player games
  - ✓ Distributed simulations

Without support for multicast

- ✓ A source needs to send a separate packet with the identical data to each member of the group
  - ➢ This redundancy consumes more bandwidth
  - ➢ Redundant traffic is not evenly distributed, concentrated near the sending host
- ✓ Source needs to keep track of the IP address of each member in the group
  - ➢ Group may be dynamic

- **To support many-to-many and one-to-many IP provides an IP-level multicast**

- Basic IP multicast model is many-to-many based on multicast groups
  - ✓ Each group has its own IP multicast address
  - ✓ Hosts that are members of a group receive copies of any packets sent to that group's multicast address
  - ✓ A host can be in multiple groups
  - ✓ A host can join and leave groups
- Using IP multicast to send the identical packet to each member of the group
  - ✓ A host sends a single copy of the packet addressed to the group's multicast address
  - ✓ The sending host does not need to know the individual unicast IP address of each member
  - ✓ Sending host does not send multiple copies of the packet
- IP's original many-to-many multicast has been supplemented with support for a form of one-to-many multicast

One-to-many multicast

Source specific multicast (SSM)

- ✓ Single sender, multiple receivers
- ✓ A receiving host specifies both a multicast group and a specific sending host
  E.g. radio stations, TV stations

Many-to-many model

Any source multicast (ASM)

- ✓ Some or all nodes can become sender
  E.g. teleconferencing, online video games

- A host signals its desire to join or leave a multicast group by communicating with its local router using a special protocol
  - ✓ In IPv4, the protocol is Internet Group Management Protocol **(IGMP)**
  - ✓ In IPv6, the protocol is Multicast Listener Discovery**(MLD**)

- The router has the responsibility for making multicast behave correctly with regard to the host

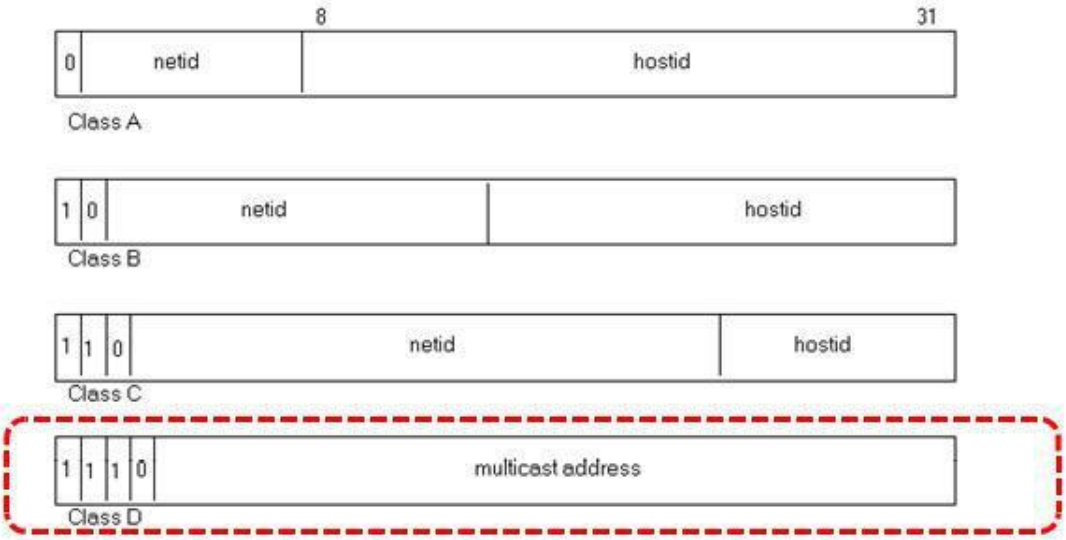
Multicast address

- A **multicast address** is a logical identifier for a group of hosts in a computer network, that are available to process datagrams or frames intended to be multicast for a designatednetwork service.
- IPv6 has a portion of address space 11111111 reserved for multicast group addresses.
- When a host on the Ethernet joins an IP multicast group, it configures its Ethernet interface to receive packets with corresponding Ethernet multicast address.
- This causes the receiving host to receive not only the multicast traffic but also traffic sent to any of the other multicast groups. Thus, IP header of any receiving host examine IP header of any multicast packet to determine whether the packet belongs to desire group.

Multicasting with IPv4

- Multicast addressing can be used in the Link Layer (Layer 2 in the OSI model), such as Ethernet multicast, and at the InternetLayer (Layer 3 for OSI) for Internet Protocol Version 4 (IPv4) or Version 6 (IPv6) multicast.
- IPv4 multicast addresses are defined by the leading address bits of *1110*, originating from the classful network design of the early Internet when this group of addresses was designated as *Class D*.

- The Classless Inter-Domain Routing (CIDR) prefix of this group is 224.0.0.0/4
- The group includes the addresses from 224.0.0.0 to 239.255.255.255.



Example list of notable well-known IPv4 addresses that are reserved for IP multicasting and that are registered with the Internet Assigned Numbers Authority(IANA).

224.0.0.1	The *All Hosts* multicast group addresses all hosts on the same network segment.
224.0.0.2	The *All Routers* multicast group addresses all routers on the same network segment.
224.0.0.4	This address is used in the Distance Vector Multicast RoutingProtocol (DVMRP) to address multicast routers.
224.0.0.	The Open Shortest Path First (OSPF) *All OSPF Routers* address is used to send Hello packets to all OSPF routers on a network segment.

Multicasting with IPv6

- Multicast addresses in IPv6 have the prefix ff00::/8. IPv6 multicast addresses are generally formed from four bit groups, illustrated as follows:

General multicast address format

Bits	8	4	4	112
Field	*prefix*	*flags*	*scope*	*group ID*

- The *prefix* holds the binary value 11111111 for any multicast address. Currently, 3 of the 4 flag bits in the *flags* field are defined; the most-significant flag bit is reserved for future use. The other three flags are known as *R*, *P* and *T*.

Multicast address flags

Bit	Flag	0	1
0 (MSB)	(Reserved)	(Reserved)	(Reserved)
1	R (Rendezvous)	Rendezvous point not embedded	Rendezvous point embedded
2	P (Prefix)	Without prefix information	Address based on network prefix
3 (LSB)	T (Transient)	Well-known multicast address	Dynamically assigned multicast address

Some Well-known IPv6 multicast addresses

Address	Description
ff02::1	All nodes on the local network segment
ff02::2	All routers on the local network segment
ff02::5	OSPFv3 All SPF routers
ff02::6	OSPFv3 All DR routers
ff02::9	RIP routers
ff02::d	PIM routers
ff0x::114	Used for experiments

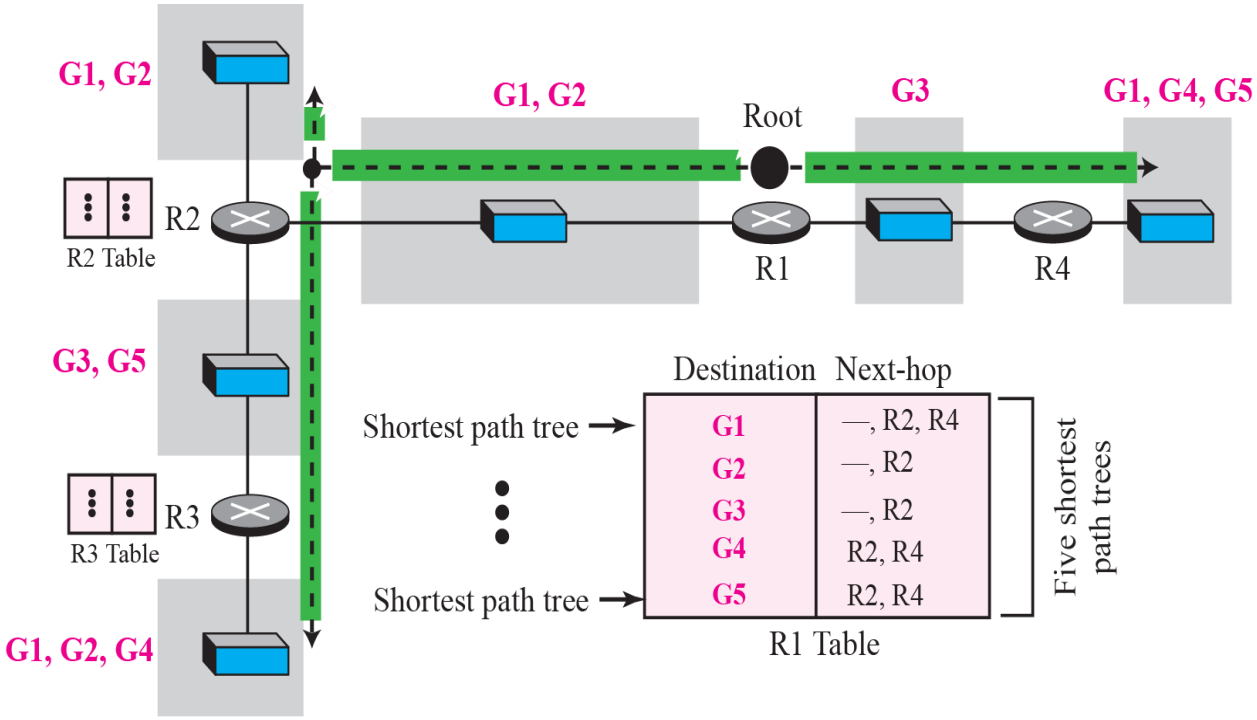
MULTICAST ROUTING

- A router's unicast forwarding tables indicate for any IP address, which link to use to forward the unicast packet
- To support multicast, a router must additionally have multicastforwarding tables that indicate, based on multicast address, which links to use to forward the multicast packet
- Unicast forwarding tables collectively specify a set of paths
- Multicast forwarding tables collectively specify a set of trees

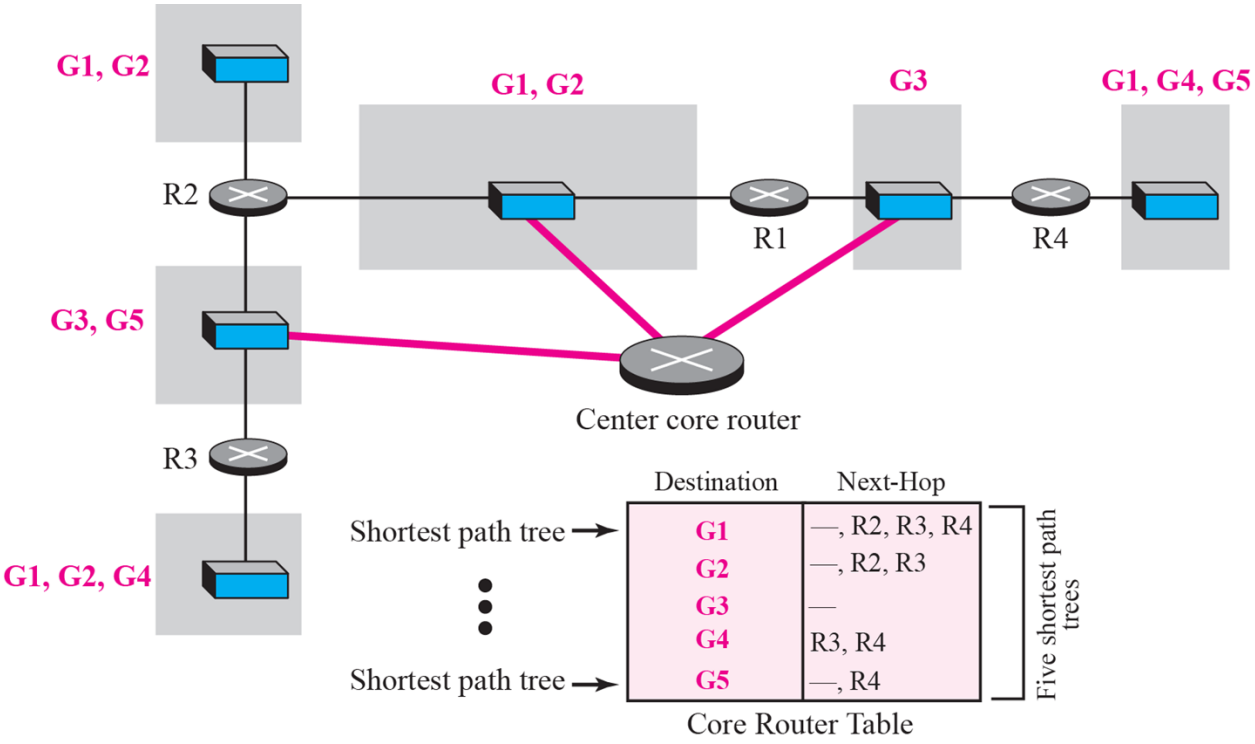
✓ **Multicast distribution trees**

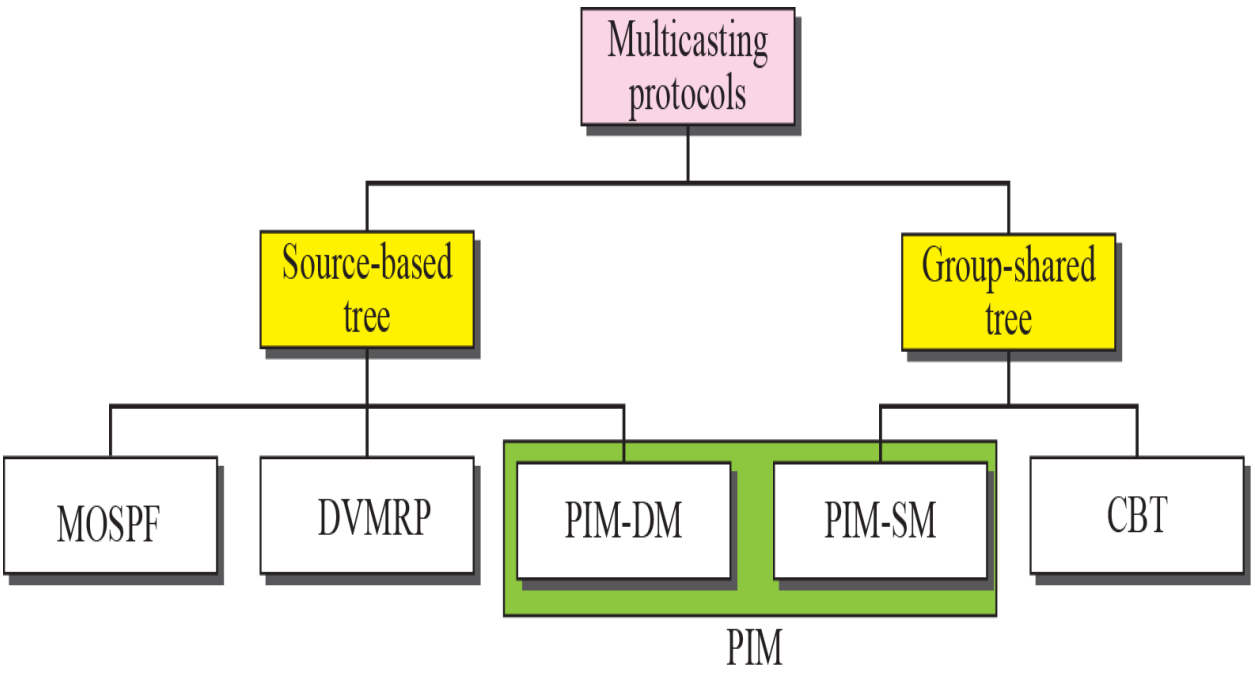
- *In multicast routing, each involved router needs to construct a shortest path tree for each group.*
- In the source-based tree approach, each router needs to have one shortest path tree for each group and source.
- In the group-shared tree approach, only the core router, which has a shortest path tree for each group, is involved in multicasting.

SOURCE BASED APPROACH



GROUP SHARED TREE APPROACH:





DISTANCE VECTOR MRP:

Using DVMRP, multicasting is a two stage process, they are

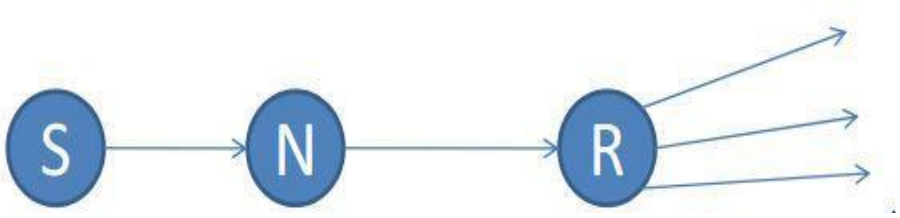
1. Flood: design a broadcast that can forward packets to all nodes on the Internet
2. Prune: refine broadcast to prune networks that have no nodes in multicast group

Decision-making strategies

- Flooding
- Reverse Path Forwarding (RPF)
- Reverse Path Broadcasting (RPB)

Reverse Path Forwarding (RPF):

- Each router already knows that shortest path to source S goes through router N.
- When receive multicast packet from S, forward on all outgoing links (except the one on which the packet arrived), if packet arrived from N. flood to all links except to the link connected to S.



Two shortcomings

1. It truly floods the network.
   - ✓ Cannot avoid flooding to LANS with no multicast group participants
2. A packet is forwarded to a LAN by ALL routers connected to it.

Two shortcomings

3. It truly floods the network.
   - ✓ Cannot avoid flooding to LANS with no multicast group participants
4. A packet is forwarded to a LAN by ALL routers connected to it.

Solution to shortcoming1: REVERSE PATH BROADCAST (RPB)

Reverse Path?We are considering shortest path towards rootwhen making forwarding decisions– (unicast look at shortest path to destination).

Goal: **Prune networks that have no hosts in group G**
– Done in 2 steps.

Step 1: Determine if LAN is a *leaf* with no members in G
– **leaf** if parent is only router on the LAN
– determine if any hosts are members of G using IGMP (nodes in G periodically announce membership in network)

Parent router decide to forward multicast packet to LAN based on membership announcements

Step 2: Propagate "no members of G here" information

– LAN send Parent router a **set of groups for which thisnetwork is interested in receiving multicast packets**.

– This information is propagated from router to router

– Now each router know for each of its links, for what groups it should forward multicast packets.

Solution to shortcoming2

- **Eliminate duplicate broadcast packets**
- Allow only the designated router (parent router) forward packets to LAN
- Parent Router selection.
  - Router with shortest path to S
  - Use smallest address to break ties

- A router can learn if it is the parent for the LAN for a given source.

PROTOCOL INDEPENDENT MULTICAST:

Protocol Independent Multicast (PIM) is developed

i. To solve scaling issues.
ii. To limit the traffic received by each group.

PIM is divided into two groups,

1. PIM – DM (Dense Mode)
2. PIM – SM (Sparse Mode)

PIM – DM

- PIM – Dense mode uses flood and prune algorithm (used in DVMRP).
- Suffers from scaling issues

PIM - SM

- Routers explicitly join and leave the group
  - Uses "Join" and "Leave" messages.
- PIM assigns a representative node called the "RendezvousPoint" (or RP)to each multicast group.
- RP's IP address is known to all the routers in a domain.
- PIM-SM defines a set of procedures by which all routers in a domain can agree to use RP for a given group.
- Multicast forwarding tree is built as a result of routers sending join messages to RP.

- PIM-SM use join message to build <u>two kinds of trees</u>
  1. Shared Tree: Used by all senders
  2. Source-Specific Tree: Used by only a specific sending host.

Under **normal course of operations**:

– A shared tree is built first.

– 1 or more source specific trees are constructed if there is enough traffic towarrant this.

PIM-SM first creates shared tree first followed by one or more source specific trees.

When a router sends a join message towards RP for a group G, it is sent as normal IP unicast transmission.

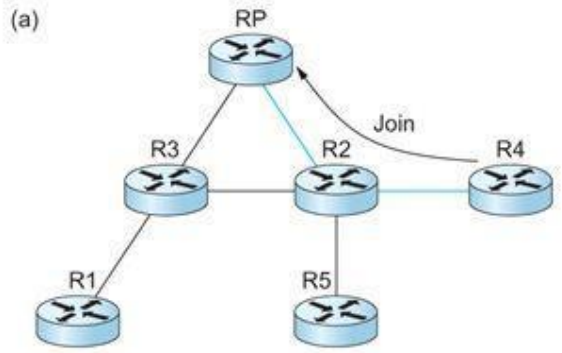


Fig : (a) R4 sends Join to RP and joins shared tree

Join message has to pass through a sequence of routers before reaching RP. Each router along the path look at join and create a forwarding table entry for shared tree. In the figure a) the Join message from R4 passes through R2 to reach RP.

As more routers send join messages towards RP, they form new branches to be added to tree as follows,

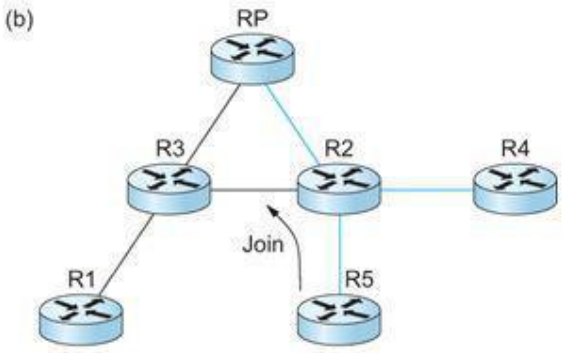


Fig : (b) R5 joins shared tree

If R1 wants to build a source-specific tree, RP sends a join message to R1.

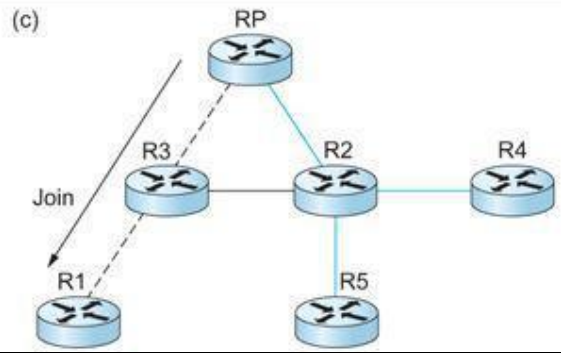


Fig: (c) RP builds source-specific tree to R1 by sending Join to R1

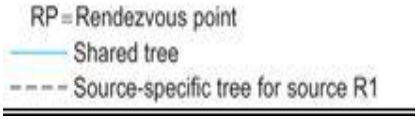
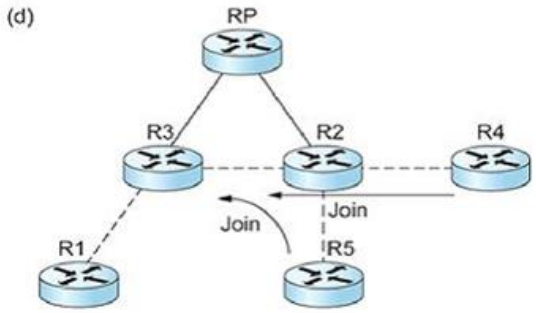


Fig : (d) R4 and R5 can build a source – specific tree to R1 by sending joins to R1.



Traditional multicast

– A group address is a single IP address taken from a reserved range (224.0.0.0/4 for IPv4, FF00::/8 for IPv6)

- **Source-Specific Multicast (SSM) -**RFC4607

– An SSM group, called a *channel*, is identified as (S,G)

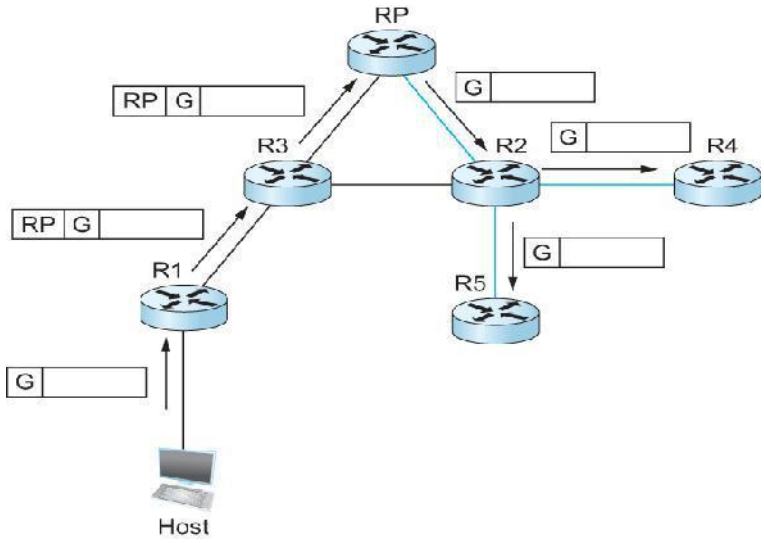
- S: source address , G: group address
- ASM multicast route written as (*,G)

– reserved IPv4 address range 232.0.0.0/8 and the IPv6 range FF3x::/32

Advantages of SSM

1. Because an SSM channel is defined by **both a sourceand a group address**,group addresses can be re-usedby multiple sources while keeping**channels unique**.
   - E.g. SSM channel (192.168.45.7, **232.7.8.9**) is different than (192.168.3.104, **232.7.8.9**)
   - hosts subscribed to one will not receive traffic from the other
   - hosts will only receive traffic from explicitly requested sources
2. SSM does not rely on the designation of a rendezvous point (RP)to establish a multicast tree
   - **Why?**
   - because the **source of an SSM channel is alwaysknown in advance**, multicast trees are efficientlybuilt from channel hosts toward the source (based on the **unicast routing topology**) without the need for an RP.

The delivery of packet along a shared tree is as follows,



- **R1 tunnels the packet to the RP** (as R1 has no state in multicastgroup with RP)
  - -R1 encapsulate multicast packet inside a PIM registermessage and send to RP
- **RP sends to R2** which forwards it along the shared tree to **R4 and R5**.

UNIT IV - TRANSPORT LAYER

Overview of Transport layer – UDP – Reliable byte stream (TCP) – Connection management – Flow control – Retransmission – TCP Congestion control – Congestion avoidance (DECbit, RED) – QoS – Application requirements

Overview of Transport layer

Role of Transport Layer

- Communication between processes (e.g., socket)
- Relies on network layer and serves the application layer
- E.g., TCP and UDP

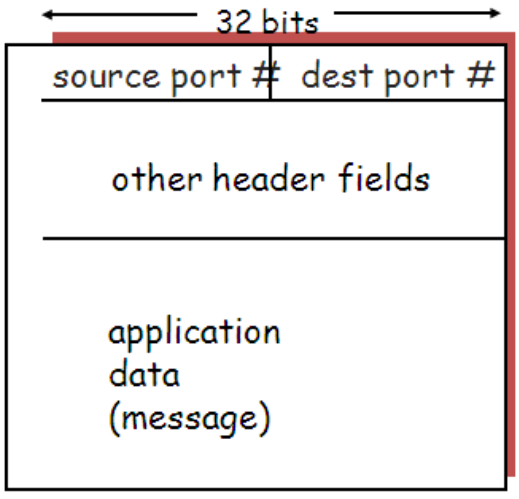
- Provide *logical communication* between application processes running on different hosts
- Run on end hosts
  - Sender: breaks application messages into segments, and passes to network layer
  - Receiver: reassembles segments into messages, passes to application layer
- Multiple transport protocol available to applications
  - Internet: TCP and UDP

Internet Transport Protocols

- Datagram messaging service (UDP)
  - "best-effort"
- Reliable, in-order delivery (TCP)
  - Connection set-up
  - Discarding of corrupted packets
  - Retransmission of lost packets
  - Flow control
  - Congestion control (next lecture)
- Other services not available
  - Delay guarantees
  - Bandwidth guarantees

Multiplexing and Demultiplexing

- Host receives IP datagrams
  - Each datagram has source and destination IP address,
  - Each datagram carries one transport-layer segment
  - Each segment has source and destination port number
- Host uses IP addresses and port numbers to direct the segment to appropriate socket



TCP/UDP segment format

Simple Demultiplexer (UDP)

The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication**.**

UDP is a simple transport protocol that extends the host-to-host delivery of packets of the underlying network into a process-to-process communication. Since there are many processes running on a given host (e.g. multiple Internet browsers), UDP needs to add a level of demultiplexing, allowing multiple application processes on each host to share the network.

It is host to host(process to process) communication transport protocol.

A more common approach, and the one used by UDP, is for processes to *indirectly identify each other using an abstract locator, often called a port* or *mailbox.*

The basic idea is for a source process to send a message to a port and for the destination process to receive the message from a port.

Notice that the UDP port field is only 16 bits long. This means that there are up to 64K possible ports, clearly not enough to identify all the processes on all the hosts in the Internet.

That is, a process is really identified by a port on some particular host—a(port,host) pair.

A client process initiates message exchange with a server .

Once a client has contacted a server, the server knows the client's port (it was contained in the message header) and can reply to it. The real problem, therefore, is how the client learns the server's port in the first place.

A common approach is for the server to accept messages at a *well-known port*. That is, each server receives its messages at some fixed port that is widely published. In the Internet, for example, the Domain Name Server (DNS) receives messages at well-known port 53 on each host.

The mail service listens for messages at port 25, and the Unix talk program accepts messages at well-known port 517, and so on.

An alternative strategy is to generalize this idea, so that there is only a single well-known port— the one at which the —"Port Mapper" service accepts messages.

A client would send a message to the Port Mapper's well-known port asking for the port it should use to talk to the —whatever service, and the Port Mapper returns the appropriate port.

It is host to host(process to process) communication transport protocol.

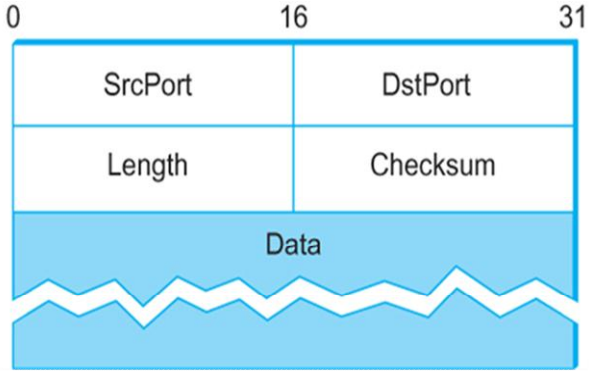
The only interesting issue in such a protocol is the form of the address used to identify the target process.

This strategy makes it easy to change the port associated with different services over time, and for each host to use a different port for the same service. This strategy makes it easy to change the port associated with different services over time, and for each host to use a different port for the same service.

Unreliable Message Delivery Service

- Lightweight communication between processes
  - Avoid overhead and delays of ordered, reliable delivery
  - Send messages to and receive them from a socket

Format of UDP Header



Format for UDP header

- source port: 16 bit port number of the source
- destination port: 16 bit port number of the destination
- length: 16 bit number representing the length in bytes of the UDP datagram (including the header)
- checksum: 16 bit checksum used for error detection (later)
- data: the message

Source port number -This field identifies the sender's port when meaningful and should be assumed to be the port to reply to if needed.
- If not used, then it should be zero.
- If the source host is the client, the port number is likely to be an ephemeral port number.
- If the source host is the server, the port number is likely to be a well-known port number.

Destination port number -This field identifies the receiver's port and is required.
- Similar to source port number, if the client is the destination host then the port number will likely be an ephemeral port number. if the destination host is the server then the port number will likely be a well-known port number.

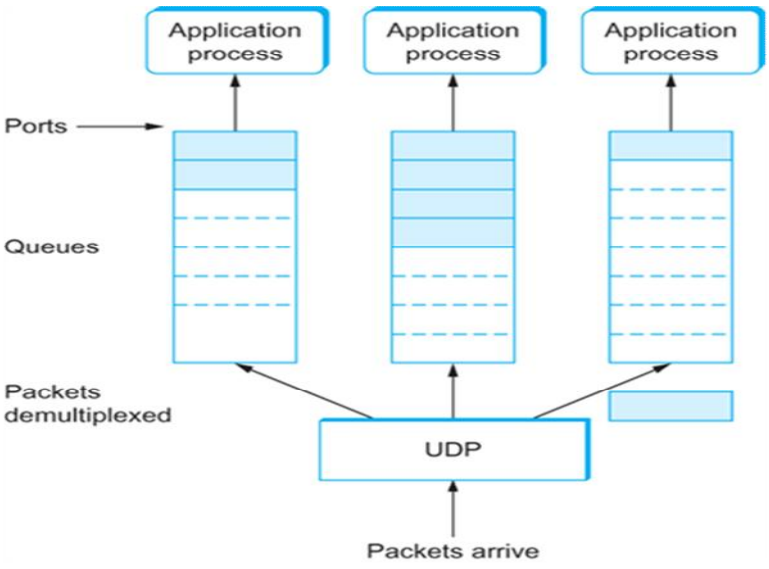
Length -A field that specifies the length in bytes of the entire datagram: header and data.
- The minimum length is 8 bytes since that's the length of the header.
- The field size sets a theoretical limit of 65,535 bytes (8 byte header 65,527 bytes of data) for a UDP datagram.

Checksum-The checksum field is used for error-checking of the header *and* data. If no checksum is generated by the transmitter, the field uses the value all-zeros.

UDP Message Queue

- Typically, a port is implemented by a message queue.
- When a message arrives, the protocol (e.g., UDP) appends the message to the end of the queue.
- Should the queue be full, the message is discarded. There is no flow-control mechanism that tells the sender to slow down.
- When an application process wants to receive a message, one is removed from the front of the queue.
- If the queue is empty, the process blocks until a message becomes available.

Finally, although UDP does not implement flow control or reliable/ordered delivery, it does a little more work than to simply demultiplex messages to some application process—it also ensures the correctness of the message by the use of a checksum.



UDP Message Queue

UDP computes its checksum over the UDP header, the contents of the message body, and something called the *pseudo header*.

The pseudo header consists of three fields from the IP header—protocol number, source IP address, and destination IP address— plus the UDP length field.

UDP uses the same checksum algorithm as IP.

The motivation behind having the pseudo header is to verify that this message has been delivered between the correct two endpoints.

For example, if the destination IP address was modified while the packet was in transit, causing the packet to be misdelivered, this fact would be detected by the UDP checksum.

Why Would Anyone Use UDP?

- Finer control over what data is sent and when
  - As soon as an application process writes into the socket
  - … UDP will package the data and send the packet
- No delay for connection establishment
  - UDP just blasts away without any formal preliminaries
  - … which avoids introducing any unnecessary delays
- No connection state
  - No allocation of buffers, parameters, sequence #s, etc.
  - … making it easier to handle many active clients at once
- Small packet header overhead
  - UDP header is only eight-bytes long

Popular Applications That Use UDP

- Multimedia streaming
  - Retransmitting lost/corrupted packets is not worthwhile
  - By the time the packet is retransmitted, it's too late
  - E.g., telephone calls, video conferencing, gaming
- Simple query protocols like Domain Name System
  - Overhead of connection establishment is overkill
  - Easier to have application retransmit if needed

Reliable Byte Stream (TCP)

- In contrast to UDP, Transmission Control Protocol (TCP) offers the following services
  - Reliable
  - Connection oriented
  - Byte-stream service

- It is a more sophisticated transport protocol is one that offers a reliable, connection-oriented, byte-stream service.

- Transmission Control Protocol (TCP) is probably the most widely used protocol.

- It also includes a flow-control.

- like UDP, TCP supports a demultiplexing mechanism that allows multiple application programs on any given host to simultaneously carry on a conversation with their peers.

- TCP also implements a highly tuned congestion-control mechanism.

End to End Issues

At the heart of TCP is the sliding window algorithm.

TCP supports logical connections between processes that are running on any two computers in the Internet.

This means that TCP needs an explicit connection establishment phase during which the two sides of the connection agree to exchange data with each other.

This difference is analogous to having to dial up the other party, rather than having a dedicated phone line.

TCP also has an explicit connection teardown phase.

One of the things that happen during connection establishment is that the two parties establish some shared state to enable the sliding window algorithm to begin.

Connection teardown is needed so each host knows it is OK to free this state.

Second, whereas a single physical link that always connects the same two computers has a fixed RTT, TCP connections are likely to have widely different round-trip times.

Variations in the RTT are even possible during a single TCP connection that lasts only a few minutes.

What this means to the sliding window algorithm is that the timeout mechanism that triggers retransmissions must be adaptive.

A third difference is that packets may be reordered as they cross the Internet, but this is not possible on a point-to-point link where the first packet put into one end of the link must be the first to appear at the other end.

Packets that are slightly out of order do not cause a problem since the sliding window algorithm can reorder packets correctly using the sequence number.

The real issue is how far out-of-order packets can get, or said another way, how late a packet can arrive at the destination

In the worst case, a packet can be delayed in the Internet until IP's time to live (TTL) field expires, at which time the packet is discarded.

Knowing that IP throws packets away after their TTL expires, TCP assumes that each packet has a maximum lifetime. The exact lifetime, known as the *maximum segment lifetime (MSL).*

The current recommended setting is 120 seconds.

For example, if a link's delay × bandwidth product is computed to be 8 KB—meaning that a window size is selected to allow up to 8 KB of data to be unacknowledged at a given time—then

it is likely that the computers at either end of the link have the ability to buffer up to 8 KB of data.

Fifth, because the transmitting side of a directly connected link cannot send any faster than the bandwidth of the link allows, and only one host is pumping data into the link, it is not possible to unknowingly congest the link.

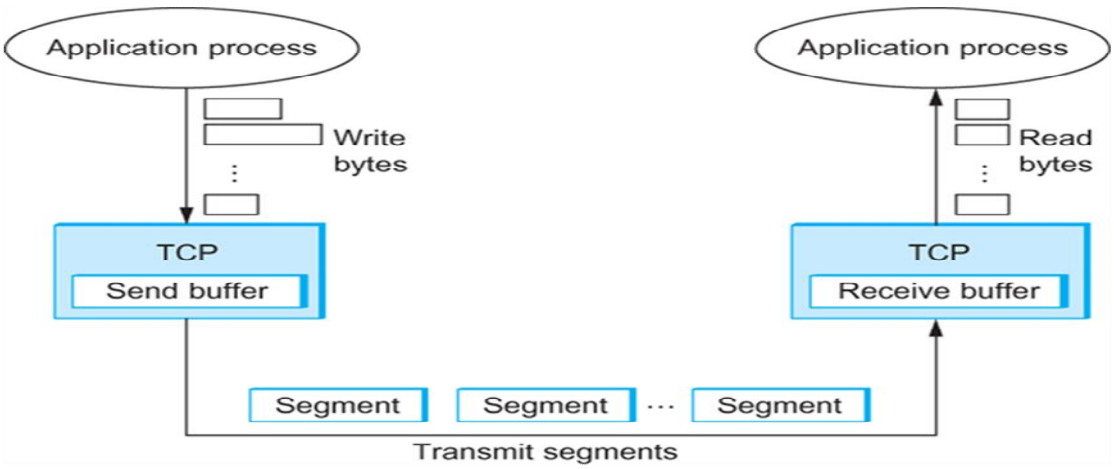
The load on the link is visible in the form of a queue of packets at the sender.

Segment Format

TCP is a byte-oriented protocol, which means that the sender writes bytes into a TCP connection and the receiver reads bytes out of the TCP connection.

Instead, TCP on the source host buffers enough bytes from the sending process to fill a reasonably sized packet and then sends this packet to its peer on the destination host.

TCP on the destination host then empties the contents of the packet into a receive buffer, and the receiving process reads from this buffer at its leisure.



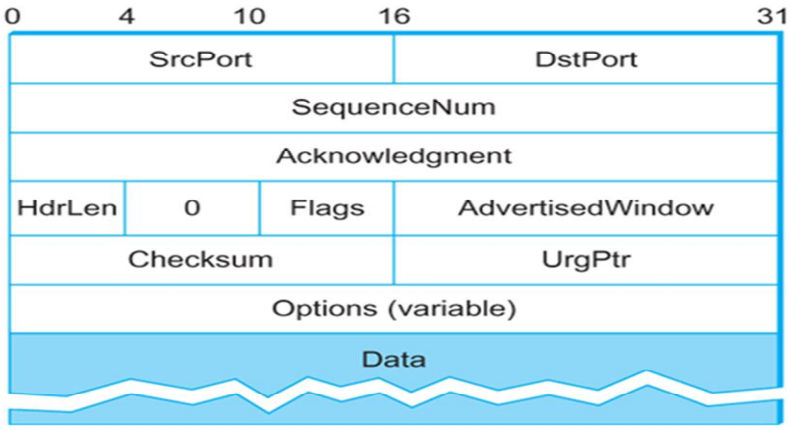
How TCP manages a byte stream.

In general, a single TCP connection supports byte streams flowing in both directions.

Segment Format

The packets exchanged between TCP peers in are called *segments.*

since each one carries a segment of the byte stream. Each TCP segment contains the header.



TCP Header Format

source port: 16 bit port number of the source
destination port: 16 bit port number of the destination
sequence number: 32 bit *SN*
request number: 32 bit *RN*
HdrLen: length of header in 32 bit words, needed because of options, also known as offset field
flags: 6 bits for SYN, FIN, RESET, PUSH, URG, and ACK

- **SYN and FIN** are used to establish and terminate a TCP connection (see next section)
- **RESET** is set by the receiver to abort the connection, e.g. when the receiver is confused upon the receipt of an unexpected segment
- **ACK** is set by the receiver whenever an acknowledgment must be read
- **PUSH** is set by the sender to instruct TCP to flush the sending buffer, also used by the receiver to break the TCP byte stream into records (not supported by socket API)
- **URG** is set by the sender to signify that the segment contains urgent data

advertised window: 16 bit number used for flow control (later)
checksum: 16 bit checksum computed over the TCP header, the TCP data, and the pseudoheader (same algorithm as for UDP)
urgent pointer: when URG flag is set, urgent pointer indicates where the urgent data ends (it starts at the first byte of data)
option: variable
data: the message

[in detail

- SrcPort and DstPort fields identify the source and destination ports, respectively.
- Acknowledgment, SequenceNum, and AdvertisedWindow fields are all involved in TCP's sliding window algorithm.
- Because TCP is a byte-oriented protocol, each byte of data has a sequence number; the SequenceNum field contains the sequence number for the first byte of data carried in that segment.
- The Acknowledgment and AdvertisedWindow fields carry information about the flow of data going in the other direction.
- 6-bit Flags field - used to relay control information between TCP peers.

The possible flags include SYN, FIN, RESET, PUSH, URG, and ACK.

- SYN and FIN flags are used when establishing and terminating a TCP connection, respectively.
- ACK flag is set any time the Acknowledgment field is valid, implying that the receiver should pay attention to it.
- URG flag signifies that this segment contains urgent data. When this flag is set, the UrgPtr field indicates where the nonurgent data contained in this segment begins.
- The urgent data is contained at the front of the segment body, up to and including a value of UrgPtr bytes into the segment.
- PUSH flag signifies that the sender invoked the push operation, which indicates to the receiving side of TCP that it should notify the receiving process of this fact.
- Finally, the RESET flag signifies that the receiver has become confused, it received a segment it did not expect to receive—and so wants to abort the connection.
- Finally, the Checksum field is used in exactly the same way as for UDP—it is computed over the TCP header, the TCP data, and the pseudoheader, which is made up of the source address, destination address, and length fields from the IP header.]

The SrcPort and DstPort fields identify the source and destination ports, respectively, just as in UDP.

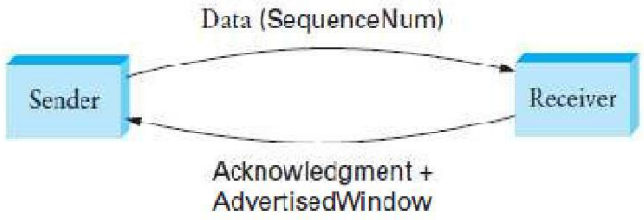
These two fields, plus the source and destination IP addresses, combine to uniquely identify each TCP connection. That is, TCP's demuxkey is given by the 4-tuple

(SrcPort, SrcIPAddr, DstPort, DstIPAddr)

The Acknowledgment, SequenceNum, and AdvertisedWindow fields are all involved in TCP's sliding window algorithm.

Because TCP is a byte-oriented protocol, each byte of data has a sequence number; the SequenceNum field contains the sequence number for the first byte of data carried in that segment.

Fig : The Acknowledgment and AdvertisedWindow fields carry information about the flow of data going in the other direction.



Comparison

	TCP	UDP
Acronym for	Transmission Control Protocol	User Datagram Protocol or Universal Datagram Protocol
Connection	TCP is a connection-oriented protocol.	UDP is a connectionless protocol.
Function	Reliable Since it follows Error control, Flow control, Acknowledgement mechanism	Unreliable Since there is no connection, error control, flow control and acknowledgement
	As a message makes its way across the internet from one computer to another. This is connection based.	UDP is also a protocol used in message transport or transfer. This is not connection based which means that one program can send a load of packets to another and that would be the end of the relationship.
Usage	TCP is suited for applications that require high reliability, and transmission time is relatively less critical.	UDP is suitable for applications that need fast, efficient transmission, such as games. UDP's stateless nature is also useful for servers that answer small queries from huge numbers of clients.
Use by other protocols	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, TFTP, SNMP, RIP, VOIP.
Ordering of data packets	TCP rearranges data packets in the order specified.	UDP has no inherent order as all packets are independent of each other. If ordering is required, it has to be managed by the application layer.
Speed of transfer	The speed for TCP is slower than UDP.	UDP is faster because there is no error-checking for packets.
Reliability	There is absolute guarantee that the data transferred remains intact and arrives in	There is no guarantee that the messages or packets sent would reach at all.

	TCP	UDP
	the same order in which it was sent.	
Header Size	TCP header size is 20 bytes	UDP Header size is 8 bytes.
Common Header Fields	Source port, Destination port, Check Sum	Source port, Destination port, Check Sum
Streaming of data	Data is read as a byte stream, no distinguishing indications are transmitted to signal message (segment) boundaries.	Packets are sent individually and are checked for integrity only if they arrive. Packets have definite boundaries which are honored upon receipt, meaning a read operation at the receiver socket will yield an entire message as it was originally sent.
Weight	TCP is heavy-weight. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP is lightweight. There is no ordering of messages, no tracking connections, etc. It is a small transport layer designed on top of IP.
Data Flow Control	TCP does Flow Control. TCP requires three packets to set up a socket connection, before any user data can be sent. TCP handles reliability and congestion control.	UDP does not have an option for flow control
Error Checking	TCP does error checking	UDP does error checking, but no recovery options.
Fields	1. Sequence Number, 2. AcK number, 3. Data offset, 4. Reserved, 5. Control bit, 6. Window, 7. Urgent Pointer 8. Options, 9. Padding, 10. Check Sum, 11. Source port, 12. Destination port	1. Length, 2. Source port, 3. Destination port, 4. Check Sum
Acknowledgement	Acknowledgement segments	No Acknowledgment
Handshake	SYN, SYN-ACK, ACK	No handshake (connectionless protocol)
Checksum	Checksum	to detect errors

Connection Establishment and Termination

A TCP connection begins with a client (caller) doing an active open to a server (callee).Assuming that the server had earlier done a passive open, the two sides engage in an exchange of messages to establish the connection.

Only after this connection establishment phase is over do the two sides begin sending data. Likewise, as soon as a participant is done sending data, it closes one direction of the connection, which causes TCP to initiate a round of connection termination messages.

Notice that while connection setup is an asymmetric activity (one side does a passive open and the other side does an active open),

Connection teardown is symmetric (each side has to close the connection independently).

Therefore, it is possible for one side to have done a close, meaning that it can no longer send data, but for the other side to keep the other half of the bidirectional connection open and to continue sending data.

Three-Way Handshake

The idea is that two parties want to agree on a set of parameters, which, in the case of opening a TCP connection, are the starting sequence numbers the two sides plan to use for their respective byte streams.

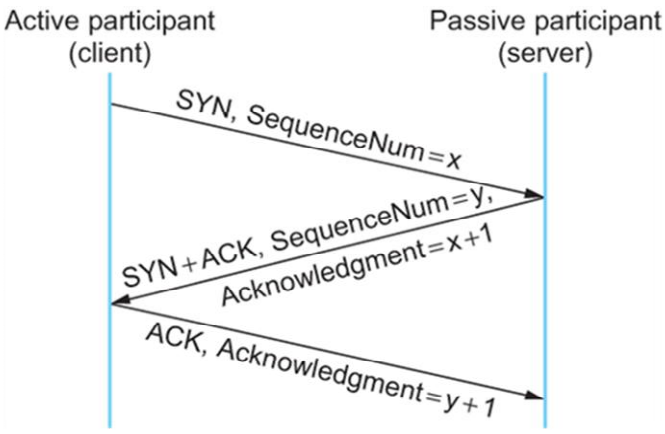
First, the client (the active participant) sends a segment to the server (the passive participant) stating the initial sequence number it plans to use (Flags = SYN, SequenceNum = *x).*

The server then responds with a single segment that both acknowledges the client's sequence number (Flags =ACK, Ack = *x + 1) and states its own beginning sequence number (Flags = SYN,* SequenceNum = *y). That is, both the SYN and ACK bits are set in the Flags field of this* second message.

Finally, the client responds with a third segment that acknowledges

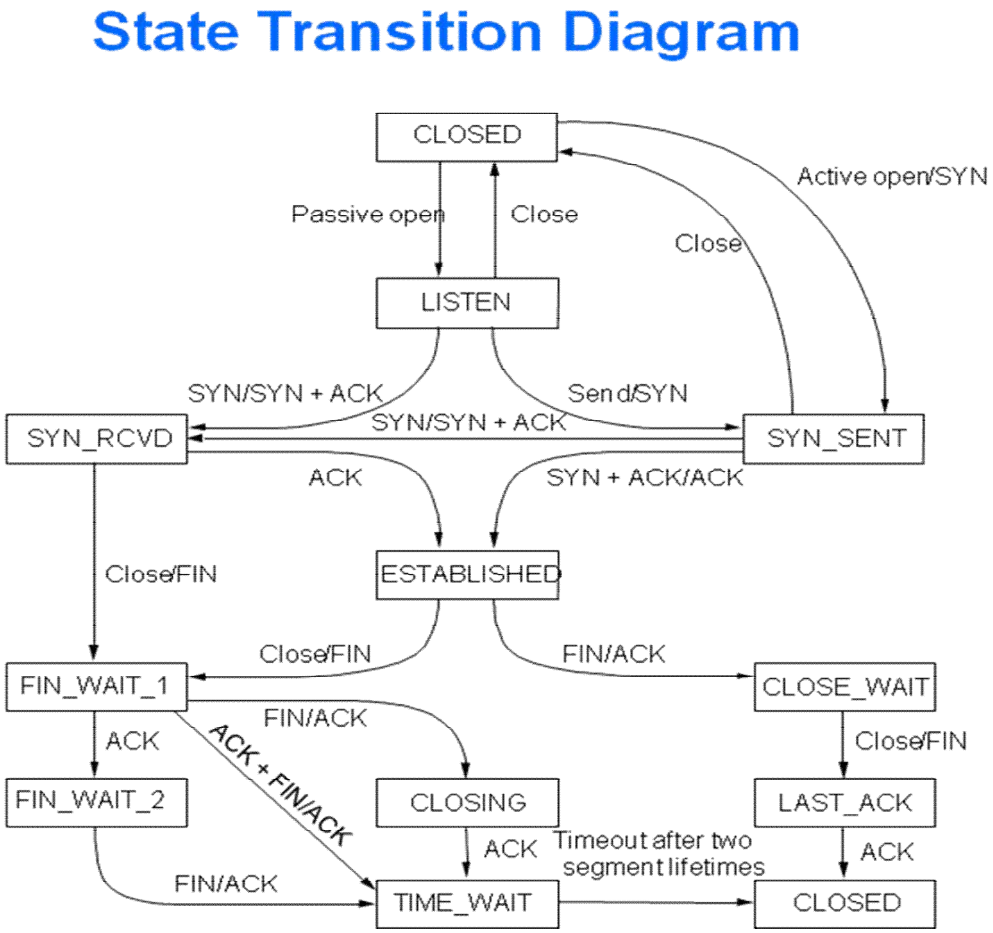
The server's sequence number (Flags = ACK, Ack = *y + 1).*

The reason that each side acknowledges a sequence number that is one larger than the one sent is that the Acknowledgment field actually identifies the —next sequence number expected, thereby implicitly acknowledging all earlier sequence numbers.



Timeline for three-way handshake algorithm

TCP state transition diagram



TCP is complex enough that its specification includes a state transition diagram. TCP's state transition diagram is fairly easy to understand. Each circle denotes a state that one end of a TCP connection can find itself in.

All connections start in the CLOSED state. As the connection progresses, the connection moves from state to state according to that arcs. Each arc is labelled with a t.ag of the form *event/action*.

Notice that two kinds of events trigger a state transition:

(1) a segment arrives from the peer (e.g., the event on the arc from LISTEN to SYN RCVD),
(2) the local application process invokes an operation on TCP (e.g., the *active open event on the arc* from CLOSE to SYN SENT).

In other words, TCP's state transition diagram effectively defines the *semantics of both its peer-to-peer interface and its service interface*

TCP makes different transitions from state to state. When opening a connection, the server first invokes a passive open operation on TCP, which causes TCP to move to the LISTEN state.

At some later time, the client does an active open, which causes its end of the connection to send a SYN segment to the server and to move to the SYN SENT state.

When the SYN segment arrives at the server, it moves to the SYN RCVD state and responds with a SYN+ACK segment. The arrival of this segment causes the client to move to the ESTABLISHED state and to send an ACK back to the server.

When this ACK arrives, the server finally moves to the ESTABLISHED state. In other words, we have just traced the three-way handshake.

There are three things to notice about the connection establishment half of the state transition diagram. First, if the client's ACK to the server is lost, corresponding to the third leg of the three-way handshake, then the connection still functions correctly.

This is because the client side is already in the ESTABLISHED state, so the local application process can start sending data to the other end.

The second thing to notice about the state transition diagram is that there is a funny transition out of the LISTEN state whenever the local process invokes a *send* operation on TCP. That is, it is possible for a passive participant to identify both ends of the connection and then to change its mind about waiting for the other side and instead actively establish the connectionection .

The final thing to notice about the diagram is the arcs that are not shown. Specifically, most of the states that involve sending a segment to the other side also schedule a timeout that eventually causes the segment to be resent if the expected response does not happen.

Turning our attention now to the process of terminating a connection, the important thing to keep in mind is that the application process on both sides of the connection must independently close its half of the connection. If only

one side closes the connection, then this means it has no more data to send, but it is still available to receive data from the other side.

This complicates the state transition diagram because it must account for the possibility that the two sides invoke the *close operator* at the same time, as well as the possibility that first one

side invoke. Thus, on any one side there are three combinations of transitions that get a connection from the ESTABLISHED state to the CLOSED state:

This side closes first:

ESTABLISHED → FIN WAIT 1 → FIN WAIT 2 → TIME WAIT →CLOSED.
The other side closes first:

ESTABLISHED → CLOSE WAIT → LAST ACK → CLOSED.
Both sides close at the same time:

ESTABLISHED → FIN WAIT 1 → CLOSING → TIME WAIT →CLOSED.

TCP Flow Control

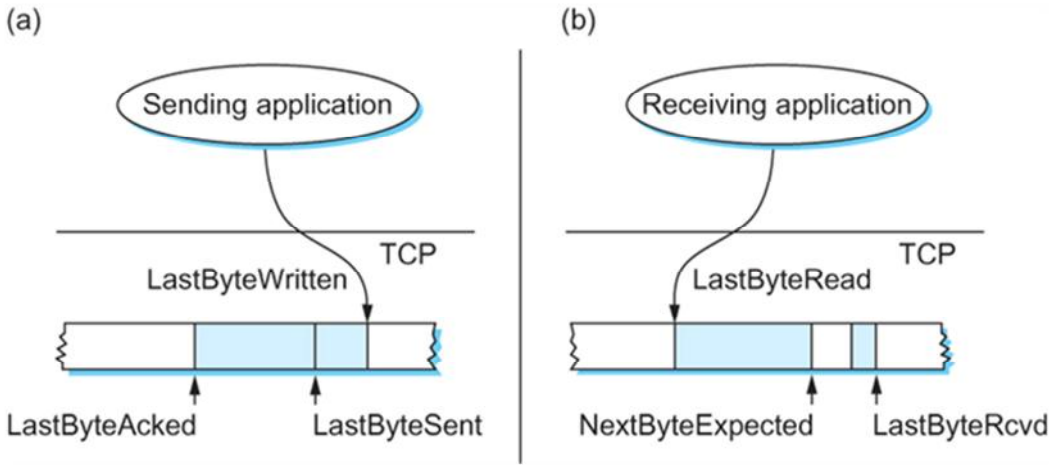
Sliding Window Revisited

- TCP uses the sliding window algorithm
- TCP's variant of the sliding window algorithm, which serves several purposes:

(1) it guarantees the reliable delivery of data,
(2) it ensures that data is delivered in order, and
(3) it enforces flow control between the sender and the receiver.

Reliable and Ordered Delivery

- TCP on the sending side maintains a send buffer. This buffer is used to store data that has been sent but not yet acknowledged, as well as data that has been written by the sending application, but not transmitted
- On the receiving side, TCP maintains a receive buffer.
- This buffer holds data that arrives out of order, as well as data that is in the correct order (i.e., there are no missing bytes earlier in the stream)



Relationship between TCP send buffer (a) and receive buffer (b).

At the sending side three pointers are maintained into the send buffer

1. LastByteAcked,
2. LastByteSent
3. LastByteWritten.

A set of pointers (sequence numbers) are maintained on the receiving side

1. LastByteRead
2. NextByteExpected
3. LastByteRcvd

- Sending Side
  - **LastByteAcked ≤ LastByteSent**

since the receiver cannot have acknowledged a byte that has not yet been sent,

  - **LastByteSent ≤ LastByteWritten**

since TCP cannot send a byte that the application process has not yet written.

- Receiving Side
  - **LastByteRead < NextByteExpected**
  - is true because a byte cannot be read by the application until it is received *and* all preceding bytes have also been received. NextByteExpected points to the byte immediately after the latest byte to meet this criterion.
  - **NextByteExpected ≤ LastByteRcvd + 1**
- since, if data has arrived in order, NextByteExpected points to the byte after LastByteRcvd

Flow control

- In a sliding window protocol, the size of the window sets the amount of data that can be sent without waiting for acknowledgment from the receiver.
- Thus, the receiver throttles the sender by advertising a window that is no larger than the amount of data that it can buffer.
- TCP on the receive side must keep

$$\textbf{LastByteRcvd} - \textbf{LastByteRead} \leq \textbf{MaxRcvBuffer}$$

It therefore advertises a window size of

AdvertisedWindow
= MaxRcvBuffer − ((NextByteExpecte d − 1) − LastByteRead)

which represents the amount of free space remaining in its buffer.

TCP on the sender side

LastByteSent − LastByteAcked= AdvertisedWindow

The sender computes an *effective window that limits how much data* it can send.

EffectiveWindow
= AdvertisedWindow−(LastByteSent−LastByteAcked)

- The sender side must also make sure that the local application process does not overflow the send buffer.

LastByteWritten − LastByteAcked≤MaxSendBuffer

If the sending process tries to write *y bytes to TCP, but*

(LastByteWritten−LastByteAcked)+*y* >*MaxSendBuffer*

Then TCP blocks the sending process and does not allow it to generate more data.

Adaptive Retransmission

Because TCP guarantees the reliable delivery of data, it retransmits each segment if an ACK is not received in a certain period of time. TCP sets this timeout as a function of the RTT it expects between the two ends of the connection.

Unfortunately, given the range of possible RTTs between any pair of hosts in the Internet, as well as the variation in RTT between the same two hosts over time, choosing an appropriate timeout value is not that easy.

To address this problem, TCP uses an adaptive retransmission mechanism.

Original Algorithm

The idea is to keep a running average of the RTT and then to compute the timeout as a function of this RTT. Specifically, every time TCP sends a data segment, it records the time.

When an ACK for that segment arrives, TCP reads the time again and then takes the difference between these two times as a SampleRTT.

TCP then computes an EstimatedRTT as a weighted average between the previous estimate and this new sample.

That is,
$$\text{EstimatedRTT} = \alpha \times \text{EstimatedRTT} + (1 - \alpha) \times \text{SampleRTT}$$

The parameter α is selected to *smooth* the EstimatedRTT.

A small α tracks changes in the RTT but is perhaps too heavily influenced by temporary fluctuations. On the other hand, a large α is more stable but perhaps not quick enough to adapt to real changes. The original TCP specification recommended a setting of α between 0.8 and 0.9.

TCP then uses EstimatedRTT to compute the timeout in a rather conservative way:

$$\text{TimeOut} = 2 \times \text{EstimatedRTT}$$

Karn/Partridge Algorithm

Whenever a segment is retransmitted and then an ACK arrives at the sender, it is impossible to determine if this ACK should be associated with the first or the second transmission of the segment for the purpose of measuring the sample RTT. It is necessary to know which transmission to associate it with so as to compute an accurate SampleRTT.

In the figure a: if you assume that the ACK is for the original transmission but it was really for the second, then the SampleRTT is too large.

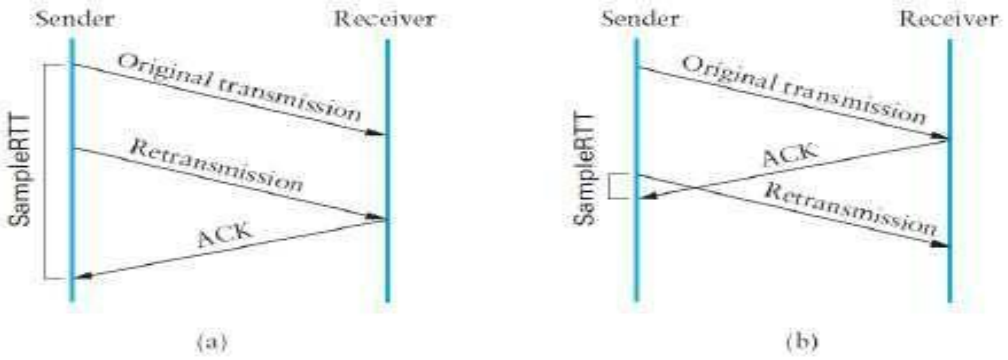


Fig : Associating the ACK with (a) original transmission versus (b) retransmission.

In the figure b: if you assume that the ACK is for the second transmission but it was actually for the first then the SampleRTT is too small.

The **solution is** surprisingly simple. Whenever TCP retransmits a segment, it stops taking samples of the RTT; it only measures SampleRTT for segments that have been sent only once.

This solution is known as the Karn/Partridge algorithm, after its inventors. Their proposed fix also includes a second small change to TCP's timeout mechanism. Each time TCP retransmits, it sets the next timeout to be twice the last timeout, rather than basing it on the last EstimatedRTT. That is, Karn and Partridge proposed that TCP use exponential backoff, similar to what the Ethernet does.

The motivation for using exponential backoff is simple: Congestion is the most likely cause of lost segments, meaning that the TCP source should not react too aggressively to a timeout.

In fact, the more times the connection times out, the more cautious the source should become.

Jacobson/Karels Algorithm

The Karn/Partridge algorithm was introduced at a time when the Internet was suffering from high levels of network congestion. Their approach was designed to fix some of the causes of that congestion, and although it was an improvement, the congestion was not eliminated.

A couple of years later, two other researchers—Jacobson and Karels—

proposed a more drastic change to TCP to battle congestion.

As an aside, it should be clear how the timeout mechanism is related to congestion—if you time out too soon, you may unnecessarily retransmit a segment, which only adds to the load on the network.

The other reason for needing an accurate timeout value is that a timeout is taken to imply congestion, which triggers a congestion-control mechanism. Finally, note that there is nothing about the Jacobson/Karels timeout computation that is specific to TCP. It could be used by any end-to-end protocol.

The main problem with the original computation is that it does not take the variance of the sample RTTs into account. Intuitively, if the variation among samples is small, then the EstimatedRTT can be better trusted and there is no reason for multiplying this estimate by 2 to compute the timeout. On the other hand, a large variance in the samples suggests that the timeout value should not be too tightly coupled to the EstimatedRTT.

In the new approach, the sender measures a new SampleRTT as before. It then folds this new sample into the timeout calculation as follows:

$$Difference = SampleRTT - EstimatedRTT$$
$$EstimatedRTT = EstimatedRTT + (\delta \times Difference)$$
$$Deviation = Deviation + \delta(|Difference| - Deviation)$$

where δ is a fraction between 0 and 1.

That is, we calculate both the mean RTT and the variation in that mean.TCP then computes the timeout value as a function of both EstimatedRTT and Deviation as follows:

$$TimeOut = \mu \times EstimatedRTT + \varphi \times Deviation$$

where based on experience, μ is typically set to 1 and φ is set to 4. Thus, when the variance is small, TimeOut is close to EstimatedRTT; a large variance causes the Deviation term to dominate the calculation.

Implementation

There are two items of note regarding the implementation of timeouts in TCP. The first is that it is possible to implement the calculation for EstimatedRTT and Deviation without using floating-point arithmetic.

Instead, the whole calculation is scaled by 2n, with δ selected to be 1/2n. This allows us to do integer arithmetic, implementing multiplication and division using shifts, thereby achieving higher performance.

The resulting calculation is given by the following code fragment, where n = 3 (i.e., δ = 1/8). Note that EstimatedRTT and Deviation are stored in their scaled-up forms, while the value of SampleRTT at the start of the code and of TimeOut at the end are real, unscaled values.

If you find the code hard to follow, you might want to try plugging some real numbers into it and verifying that it gives the same results as the equations above.

```
{
SampleRTT -= (EstimatedRTT >> 3);
 EstimatedRTT += SampleRTT;

if (SampleRTT < 0)
        SampleRTT = - SampleRTT;

SampleRTT -= (Deviation >> 3);
Deviation += SampleRTT;

TimeOut = (EstimatedRTT >> 3) + (Deviation >> 1);
}
```

The second point of note is that the Jacobson/Karels algorithm is only as good as the clock used to read the current time. On a typical Unix implementation, the clock granularity is as large as 500 ms, which is significantly larger than the average cross-country RTT of somewhere between 100 and 200 ms. To make matters worse, the Unix implementation of TCP onlychecks to see if a timeout should happen everytime this 500-ms clock ticks, and it only takes a sample of the round-trip time once per RTT.

TCP Congestion Control

- The idea of TCP congestion control is for each source to determine how much capacity is available in the network, so that it knows how many packets it can safely have in transit.
  - Once a given source has this many packets in transit, it uses the arrival of an ACK as a signal that one of its packets has left the network, and that it is therefore safe to insert a new packet into the network without adding to the level of congestion.
  - By using ACKs to pace the transmission of packets, TCP is said to be *self-clocking.*

Three mechanisms used for congestion control are,

1. Additive Increase / Multiplicative Decrease
2. Slow start
3. Fast transmit and fast recovery

1. Additive Increase Multiplicative Decrease

- TCP maintains a new state variable for each connection, called *CongestionWindow*, which is used by the source to limit how much data it is allowed to have in transit at a given time.
- The congestion window is congestion control's counterpart to flow control's advertised window.
- TCP is modified such that the maximum number of bytes of unacknowledged data allowed is now the minimum of the congestion window and the advertised window

- TCP's effective window is revised as follows:

MaxWindow = MIN(CongestionWindow, AdvertisedWindow)
EffectiveWindow = MaxWindow − (LastByteSent − LastByteAcked).

- That is, MaxWindow replaces AdvertisedWindow in the calculation of EffectiveWindow.
- Thus, a TCP source is allowed to send no faster than the slowest component—the network or the destination host—can accommodate.

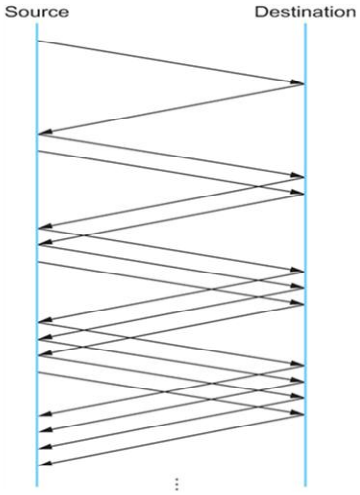
- TCP source sets the CongestionWindow based on the level of congestion it perceives to exist in the network.
- *This involves decreasing the congestion window when the level of congestion goes up and increasing the congestion window when the level of congestion goes down.*

- Taken together, the mechanism is commonly called ***additive increase/multiplicative decrease (AIMD)***
- then, is how does the source determine that the network is congested and that it should decrease the congestion window?
- The answer is based on the observation that the main reason packets are not delivered, and a timeout results, is that a packet was dropped due to congestion. It is rare that a packet is dropped because of an error during transmission.
- Therefore, TCP interprets timeouts as a sign of congestion and reduces the rate at which it is transmitting.

- Specifically, each time a timeout occurs, the source sets CongestionWindow to half of its previous value. This halving of the CongestionWindow for each timeout corresponds to the "multiplicative decrease" part of AIMD.
- For example, suppose the CongestionWindow is currently set to 16 packets. If a loss is detected, CongestionWindow is set to 8.
- Additional losses cause CongestionWindow to be reduced to 4, then 2, and finally to 1 packet.
- CongestionWindow is not allowed to fall below the size of a single packet, or in TCP terminology, the *maximum segment size (MSS)*.

A congestion-control strategy that only decreases the window size is obviously too conservative.

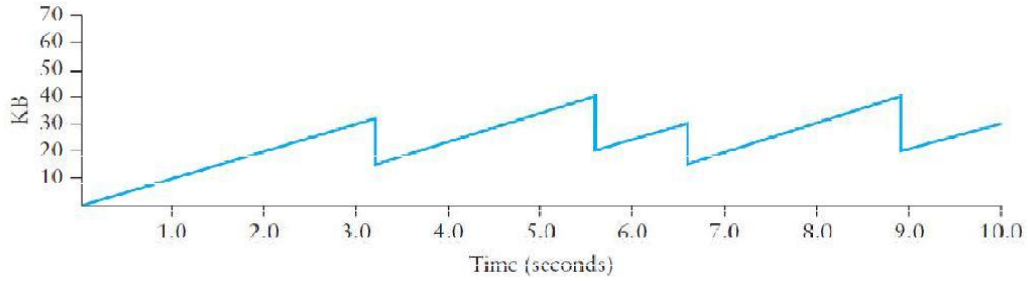
- ✓ We also need to be able to increase the congestion window to take advantage of newly available capacity in the network.
- ✓ This is the "additive increase" part of AIMD, and it works as follows.
- Every time the source successfully sends a CongestionWindow's worth of packets—that is, each packet sent out during the last RTT has been ACKed—it adds the equivalent of 1 packet to CongestionWindow.



Packets in transit during additive increase, with one packet being added each RTT.

- Note that in practice, TCP does not wait for an entire window's worth of ACKs to add 1 packet's worth to the congestion window, but instead increments CongestionWindow by a little for each ACK that arrives.
- Specifically, the congestion window is incremented as follows each time an ACK arrives:
- Increment = MSS × (MSS/CongestionWindow)
- CongestionWindow+= Increment
- That is, rather than incrementing CongestionWindow by an entire MSS bytes each RTT, we increment it by a fraction of MSS every time an ACK is received.
- Assuming that each ACK acknowledges the receipt of MSS bytes, then that fraction is MSS/CongestionWindow.

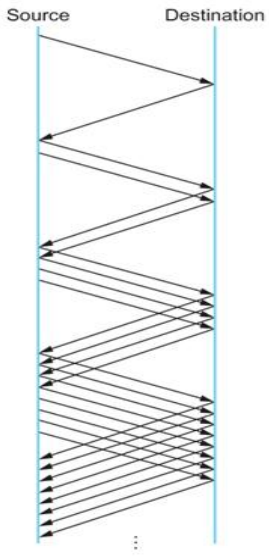
Typical TCP Sawtooth



This figure helps to visualize what happens. If you think of a TCP stream as a conveyer belt with —full containers (data segments) going in one direction and empty containers (ACKs) going in the reverse direction, then MSS-sized segments correspond to large containers and 1-byte segments correspond to very small containers.

2. Slow Start

- The additive increase mechanism just described is the right approach to use when the source is operating close to the available capacity of the network,
- TCP therefore provides a second mechanism, ironically called *slow start,* that is used to increase the congestion window rapidly from a cold start.
- Slow start effectively increases the congestion window **exponentially, rather than linearly.**
- Specifically, the source starts out by setting CongestionWindow to one packet.
- When the ACK for this packet arrives, TCP adds 1 to CongestionWindow and then sends two packets.
- Upon receiving the corresponding two ACKs, TCP increments CongestionWindow by 2—one for each ACK—and next sends four packets.
- The end result is that TCP effectively doubles the number of packets it has in transit every RTT.



Packets in transit during slow start.

- There are actually two different situations in which slow start runs.
  The first is at the very beginning of a connection, at which time the source has no idea how many packets it is going to be able to have in transit at a given time.
  - In this situation, slow start continues to double CongestionWindow each RTT until there is a loss, at which time a timeout causes multiplicative decrease to divide CongestionWindow by 2.

There are actually two different situations in which slow start runs.

- The second situation in which slow start is used is a bit more subtle; it occurs when the connection goes dead while waiting for a timeout to occur.
- Eventually, a timeout happens, but by this time there are no packets in transit, meaning that the source will receive no ACKs to "clock" the transmission of new packets.

- Although the source is using slow start again, it now knows more information than it did at the beginning of a connection.
- Specifically, the source has a current (and useful) value of CongestionWindow; this is the value of CongestionWindow that existed prior to the last packet loss, divided by 2 as a result of the loss.
- We can think of this as the "target" congestion window.
  - **Slow start is used to rapidly increase the sending rate up to this value, and then additive increase is used beyond this point.**
- Notice that we have a small bookkeeping problem to take care of, in that we want to remember the "target" congestion window resulting from

multiplicative decrease as well as the "actual" congestion window being used by slow start.

- To address this problem, TCP introduces a temporary variable to store the target window, typically called CongestionThreshold, that is set equal to the CongestionWindow value that results from multiplicative decrease.
- The variable CongestionWindow is then reset to one packet, and it is incremented by one packet for every ACK that is received until it reaches.
- CongestionThreshold, at which point it is incremented by one packet per RTT.

In other words, TCP increases the congestion window as defined by the following code fragment:

```
{
    u_int cw = state->CongestionWindow;
    u_int incr = state->maxseg;
    if (cw > state->CongestionThreshold)
        incr = incr * incr / cw;
    state->CongestionWindow = MIN(cw + incr, TCP_MAXWIN);
}
```

- where state represents the state of a particular TCP connection and TCP MAXWIN defines an upper bound on how large the congestion window is allowed to grow.

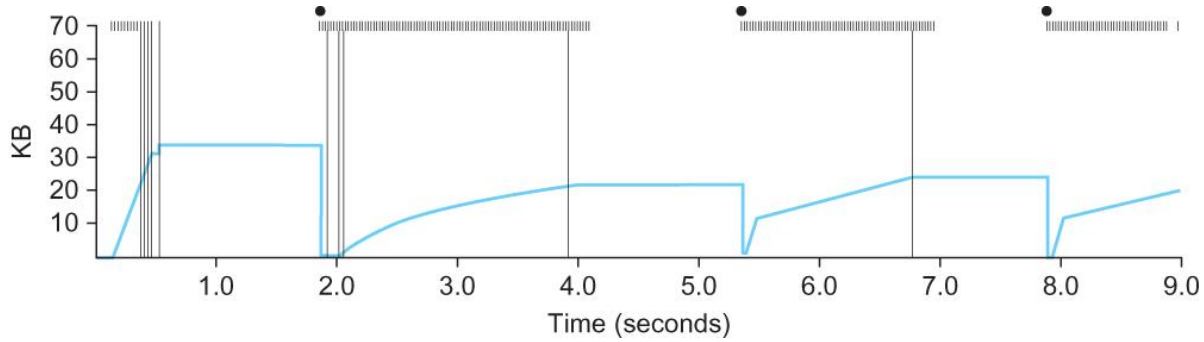


Figure : Behavior of TCP congestion control. Colored line = value of CongestionWindow over time; solid bullets at top of graph = timeouts; hash marks at top of graph = time when each packet is transmitted; vertical bars = time when a packet that was eventually retransmitted was first transmitted.

This Figure traces how TCP's CongestionWindow increases and decreases over time and serves to illustrate the interplay of slow start and additive increase/ multiplicative decrease.

This trace was taken from an actual TCP connection and shows the current value of CongestionWindow—the colored line—over time.

There are several things to notice about this trace. The first is the rapid increase in the congestion window at the beginning of the connection

This corresponds to the initial slow start phase. The slow start phase continues until several packets are lost atabout 0.4 seconds into the connection, at which timeCongestionWindow flattens out at about 34 KB.

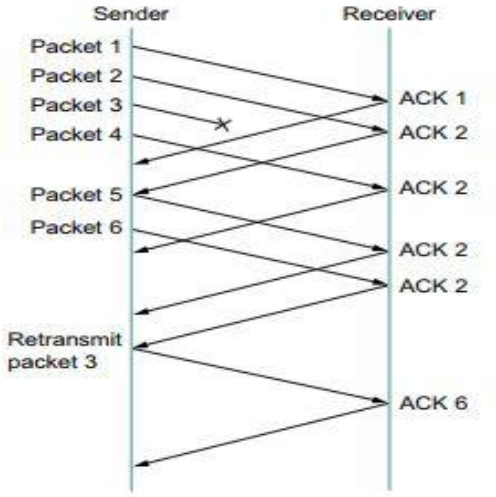
Slow Start works like the following 4 steps

1. A timeout happens, causing the congestion window to be divided by 2, and CongestionThreshold is set to this amount.
2. CongestionWindow is reset to one packet, as the sender enters slow start.
3. Slow start causes CongestionWindow to grow exponentially until it reaches CongestionThreshold.
4. CongestionWindow then grows linearly

3. Fast Retransmit and Fast Recovery

- Fast retransmit is a heuristic that sometimes triggers the retransmission of a dropped packet sooner than the regular timeout mechanism.
- The idea of fast retransmit is straightforward.
- Every time a data packet arrives at the receiving side, the receiver responds with an acknowledgment, even if this sequence number has already been acknowledged.
- Thus, when a packet arrives out of order— that is, TCP cannot yet acknowledge the data the packet contains because earlier data has not yet arrived—TCP resends the same acknowledgment it sent the last time.

- This second transmission of the same acknowledgment is called a *duplicate ACK.*
- When the sending side sees a duplicate ACK, it knows that the other side must have received a packet out of order, which suggests that an earlier packet might have been lost.



- Since it is also possible that the earlier packet has only been delayed rather than lost, the sender waits until it sees some number of duplicate ACKs and then retransmits the missing packet. In practice, TCP waits until it has seen three duplicate ACKs before retransmitting the packet.

  - Destination receives packets 1 and 2, but packet 3 lost in the network.
  - Destination will send a duplicate ACK for packet 2 when packet 4 arrives, again when packet 5 arrives and so on.
  - When sender sees 3rd duplicate ACK for packet 2-the one sent because the receiver had gotten packet 6-it retransmits packet3

  - Note that when the transmitted copy of packet 3 arrives at destination, the receiver sends a cumulative ACK for everything up to and including packet back to the source.
  - When the fast retransmit mechanism signals congestion, it is possible to use the ACKs that are still in the pipe to clock the sending of packets.

This mechanism, which is called **fast recovery**, effectively removes the slow start phase that happens between when fast retransmit detects a lost packet and additive increase begins.

For example, fast recovery avoids the slow and instead simply cuts the congestionwindow in half and resumes additive increase.

- In other words slow start is only used at the beginning of a connection.
- At all the times, congestion window is following a pure additive increase / multiplicative decrease pattern.

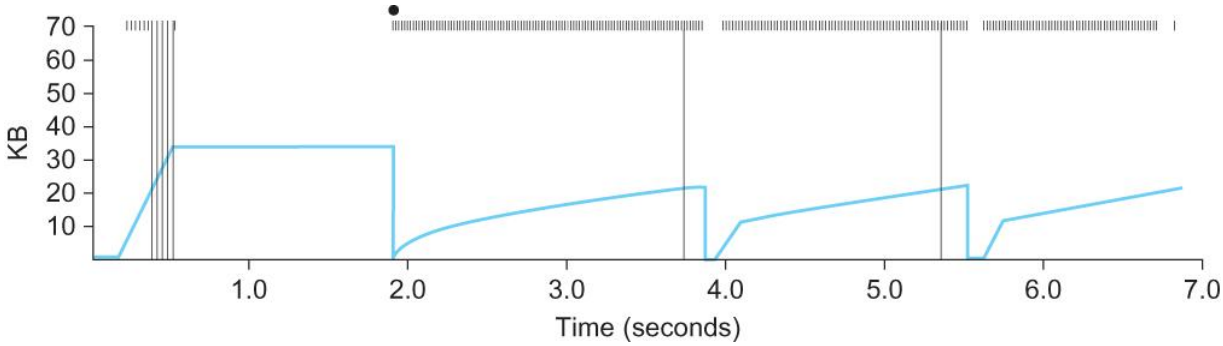


Figure : Trace of TCP with fast retransmit. Colored line = CongestionWindow; solid bullet = timeout; hash marks = time when each packet is transmitted; vertical bars = time when a packet that was eventually retransmitted was first transmitted.

Congestion Avoidance Mechanism

- ✓ It is important to understand that TCP's strategy is to control congestion once it happens, as opposed to trying to avoid congestion in the first place.
- ✓ In fact, TCP repeatedly increases the load it imposes on the network in an effort to find the point at which congestion occurs, and then it backs off from this point.
- ✓ An appealing alternative, but one that has not yet been widely adopted, is to predict when congestion is about to happen and then to reduce the rate at which hosts send data just before packets start being discarded.
- ✓ We call such a strategy *congestion avoidance, to* distinguish it from *congestion control.*

There are methods used for congestion avoidance as follows,

1. DECbit
2. RED (Random Early Detection)
3. Source-based congestion avoidance

DEC Bit

- ✓ The first mechanism was developed for use on the Digital Network Architecture (DNA), a connectionless network with a connection-oriented transport protocol.
- ✓ This mechanism could, therefore, also be applied to TCP and IP.
- ✓ As noted above, the idea here is to more evenly split the responsibility for congestion control between the routers and the end nodes.

- ✓ Each router monitors the load it is experiencing and explicitly notifies the end nodes when congestion is about to occur.
- ✓ This notification is implemented by setting a binary congestion bit in the packets that flow through the router; hence the name DECbit.
- ✓ The destination host then copies this congestion bit into the ACK it sends back to the source.
- ✓ Finally, the source adjusts its sending rate so as to avoid congestion

- ✓ A single congestion bit is added to the packet header. A router sets this bit in a packet if its average queue length is greater than or equal to 1 at the time the packet arrives.
- ✓ This average queue length is measured over a time interval that spans the last busy+idle cycle, plus the current busy cycle.

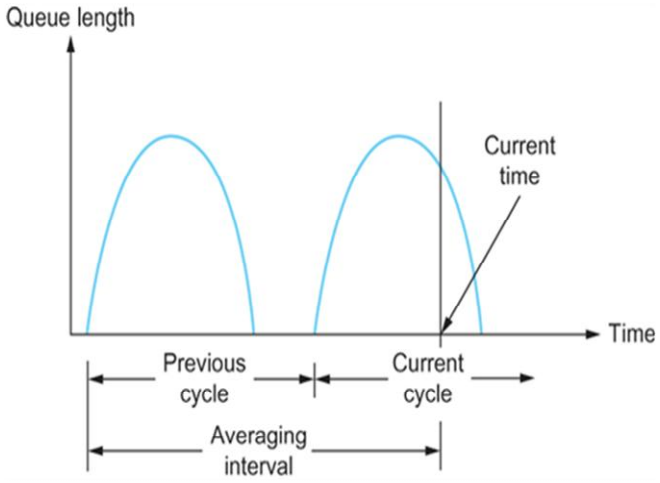
- ✓ Essentially, the router calculates the area under the curve and divides this value by the time interval to compute the average queue length.

- ✓ Using a queue length of 1 as the trigger for setting the congestion bit is a trade-off between significant queuing (and hence higher throughput) and increased idle time (and hence lower delay).
- ✓ In other words, a queue length of 1 seems to optimize the power function.

- ✓ The source records how many of its packets resulted in some router setting the congestion bit.
- ✓ In particular, the source maintains a congestion window, just as in TCP, and watches to see what fraction of the last window's worth of packets resulted in the bit being set.

- ✓ If less than 50% of the packets had the bit set, then the source increases its congestion window by one packet.
- ✓ If 50% or more of the last window's worth of packets had the congestion bit set, then the source decreases its congestion window to 0.875 times the previous value.

- ✓ The value 50% was chosen as the threshold based on analysis that showed it to correspond to the peak of the power curve. The "increase by 1, decrease by 0.875" rule was selected because additive increase/multiplicative decrease makes the mechanism stable.



Computing average queue length at a router

This figure shows the queue length at a router as a function of time. Essentially, the router calculates the area under the curve and divides this value by the time interval to compute the average queue length.

Random Early Detection (RED)

- A second mechanism, called *random early detection (RED),* is similar to the DECbit scheme in that each **router** is programmed to monitor its own queue length, and when it detects that congestion is imminent, to notify the source to adjust its congestion window.

- RED differs from the DECbit scheme in two major ways:

✓ The first is that rather than explicitly sending a congestion notification message to the source, RED is most commonly implemented such that it *implicitly notifies* the source of congestion by dropping one of its packets.

✓ The source is, therefore, effectively notified by the subsequent timeout or duplicate ACK.

✓ RED is designed to be used in conjunction with TCP, which currently detects congestion by means of timeouts (or some other means of detecting packet loss such as duplicate ACKs).

✓ As the "early" part of the RED acronym suggests, the gateway drops the packet earlier than it would have to, so as to notify the source that it should decrease its congestion window sooner than it would normally have.

✓ In other words, the router drops a few packets before it has exhausted its buffer space completely, so as to cause the source to slow down, with the hope that this will mean it does not have to drop lots of packets later on.

✓ The second difference between RED and DECbit is in the details of how RED decides when to drop a packet and what packet it decides to drop.

✓ To understand the basic idea, consider a simple FIFO queue. Rather than wait for the queue to become completely full and then be forced to drop each arriving packet, we could decide to drop each arriving packet with some *drop probability* whenever the queue length exceeds some *drop level.*

✓ This idea is *called early random drop. The RED algorithm defines the details of how to monitor the queue length and* when to drop a packet.

RED Algorithm

✓ First, RED computes an average queue length using a weighted running average similar to the one used in the original TCP timeout computation. That is, AvgLen is computed as

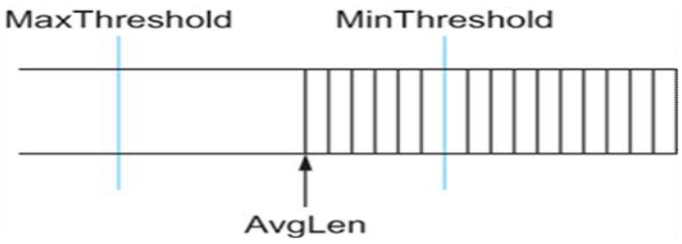
- AvgLen = (1 − Weight) × AvgLen + Weight × SampleLen
- where 0 < Weight < 1 and SampleLen is the length of the queue when a sample measurement is made.

- ✓ In most software implementations, the queue length is measured every time a new packet arrives at the gateway.
- ✓ In hardware, it might be calculated at some fixed sampling interval.

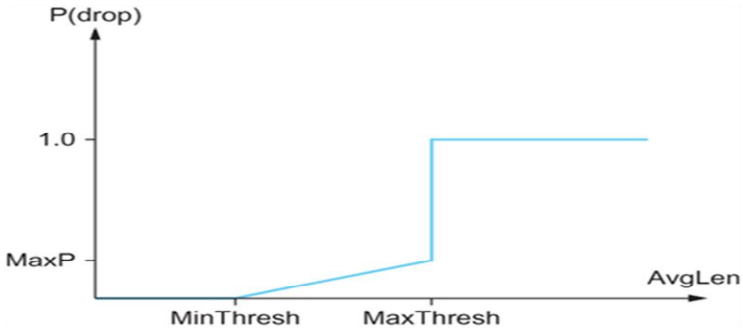
- ✓ Second, RED has two queue length thresholds that trigger certain activity: MinThreshold and MaxThreshold.
- ✓ When a packet arrives at the gateway, RED compares the current AvgLen with these two thresholds, according to the following rules:
  - ➢ if AvgLen ≤ MinThreshold
    - queue the packet
  - ➢ if MinThreshold < AvgLen < MaxThreshold
    - calculate probability P

    drop the arriving packet with probability P
  - ➢ if MaxThreshold ≤ AvgLen
    - drop the arriving packet

- ✓ P is a function of both AvgLen and how long it has been since the last packet was dropped.
- ✓ Specifically, it is computed as follows:
  - ■ TempP = MaxP × (AvgLen − MinThreshold)/(MaxThreshold − MinThreshold)
  - ■ P = TempP/(1 − count × TempP)



RED thresholds on a FIFO queue



Drop probability function for RED

Quality of Service

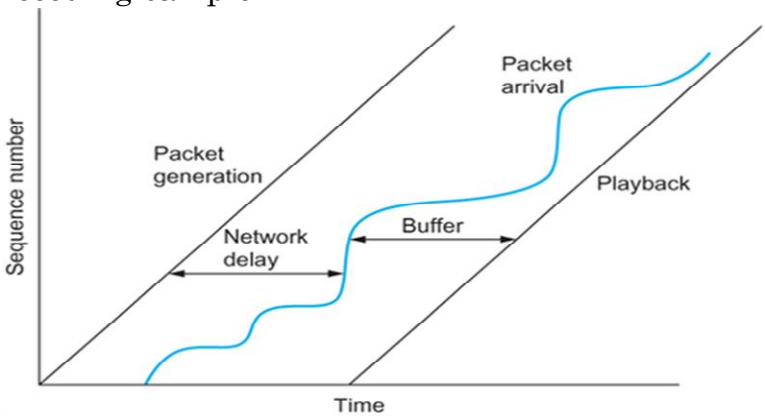
- ✓ For many years, packet-switched networks have offered the promise of supporting multimedia applications, that is, those that combine audio, video, and data.
- ✓ After all, once digitized, audio and video information become like any other form of data—a stream of bits to be transmitted. One obstacle to the fulfillment of this promise has been the need for higher-bandwidth links.
- ✓ Recently, however, improvements in coding have reduced the bandwidth needs of audio and video applications, while at the same time link speeds have increased.
- ✓ There is more to transmitting audio and video over a network than just providing sufficient bandwidth, however.
- ✓ Participants in a telephone conversation, for example, expect to be able to converse in such a way that one person can respond to something said by the other and be heard almost immediately.
- ✓ Thus, the timeliness of delivery can be very important. We refer to applications that are sensitive to the timeliness of data as *real-time applications.*
- ✓ Voice and video applications tend to be the canonical examples, but there are others such as industrial control—you would like a command sent to a robot arm to reach it before the arm crashes into something.
- ✓ Even file transfer applications can have timeliness constraints, such as a requirement that a database update complete overnight before the business that needs the data resumes on the next day.
- ✓ The distinguishing characteristic of real-time applications is that they need some sort of assurance *from the network that data is likely to arrive on time (for some definition* of "on time").
- ✓ Whereas a non-real-time application can use an end-to-end retransmission strategy to make sure that data arrives *correctly, such a strategy cannot provide* timeliness.
- ✓ This implies that the network will treat some packets differently from others—something that is not done in the best-effort model.
- ✓ A network that can provide these different levels of service is often said to support quality of service (QoS).

Application Requirement

- **Real-Time Applications**

- ✓ Data is generated by collecting samples from a microphone and digitizing them using an A →D converter
- ✓ The digital samples are placed in packets which are transmitted across the network and received at the other end
- ✓ At the receiving host the data must be played back at some appropriate rate

- ✓ For example, if voice samples were collected at a rate of one per 125 μs, they should be played back at the same rate
- ✓ We can think of each sample as having a particular playback time
- ✓ The point in time at which it is needed at the receiving host
- ✓ In this example, each sample has a playback time that is 125 μs later than the preceding sample



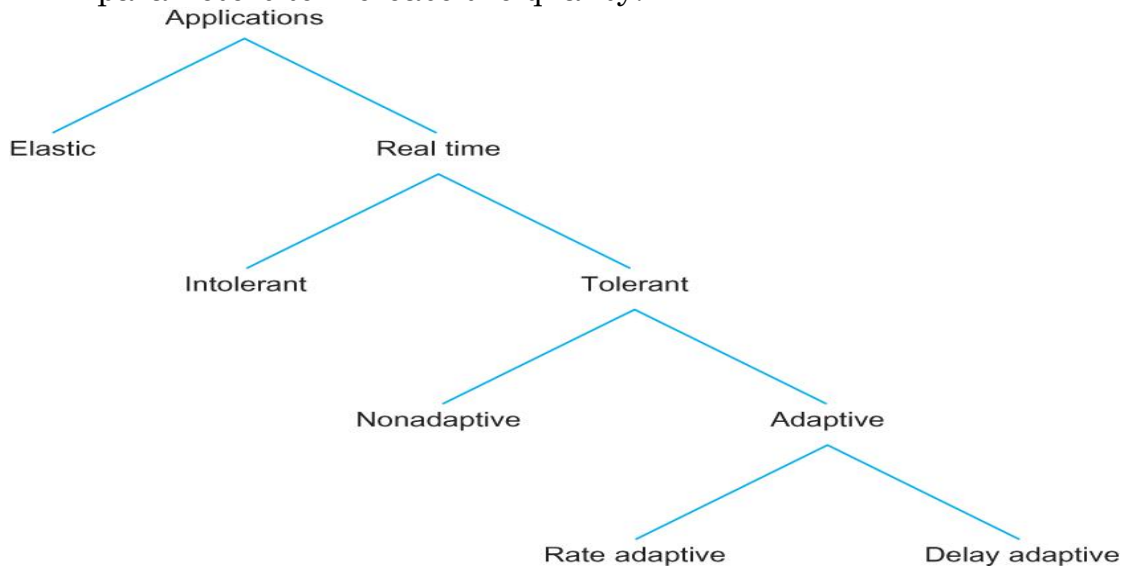
A playback buffer

- ✓ If data arrives after its appropriate playback time, it is useless
- ✓ For some audio applications, there are limits to how far we can delay playing back data
- ✓ It is hard to carry on a conversation if the time between when you speak and when your listener hears you is more than 300 ms
- ✓ We want from the network a guarantee that all our data will arrive within 300 ms
- ✓ If data arrives early, we buffer it until playback time

Taxonomy of Real-Time Applications

- ✓ The first characteristic by which we can categorize applications is their tolerance of loss of data, where "loss" might occur because a packet arrived too late to be played back as well as arising from the usual causes in the network.
- ✓ On the one hand, one lost audio sample can be interpolated from the surrounding samples with relatively little effect on the perceived audio quality. It is only as more and more samples are lost that quality declines to the point that the speech becomes incomprehensible.
- ✓ On the other hand, a robot control program is likely to be an example of a real-time application that cannot tolerate loss—losing the packet that contains the command instructing the robot arm to stop is unacceptable.
- ✓ Thus, we can categorize real-time applications as *tolerant or intolerant* depending on whether they can tolerate occasional loss
- ✓ A second way to characterize real-time applications is by their adaptability.

- ✓ For example, an audio application might be able to adapt to the amount of delay that packets experience as they traverse the network.
  - If we notice that packets are almost always arriving within 300 ms of being sent, then we can set our playback point accordingly, buffering any packets that arrive in less than 300 ms.
  - Suppose that we subsequently observe that all packets are arriving within 100 ms of being sent.
  - If we moved up our playback point to 100 ms, then the users of the application would probably perceive an improvement. The process of shifting the playback point would actually require us to play out samples at an increased rate for some period of time.
- ✓ We call applications that can adjust their playback point *delay-adaptive applications.*
- ✓ Another class of adaptive applications are *rate adaptive. For example, many* video coding algorithms can trade off bit rate versus quality. Thus, if we find that the network can support a certain bandwidth, we can set our coding parameters accordingly.
- ✓ If more bandwidth becomes available later, we can change parameters to increase the quality.



■ Approaches to QoS Support

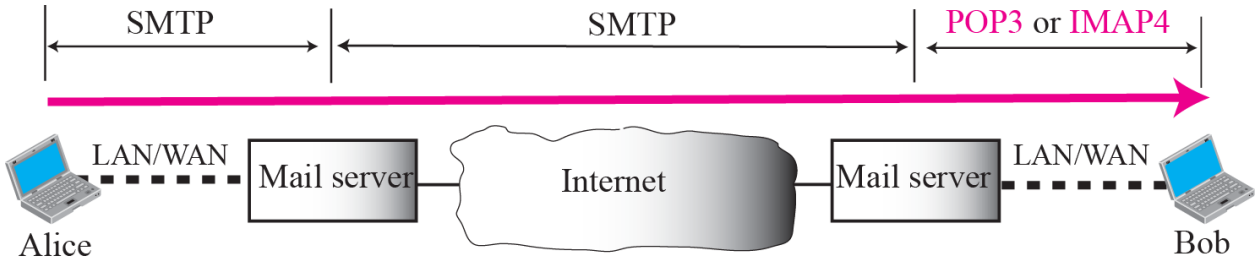
- ➢ *fine-grained approaches, which provide QoS to individual applications or flows*
- ➢ *coarse-grained approaches, which provide QoS to large classes of data or aggregated* traffic
- ➢ In the first category we find "Integrated Services," a QoS architecture developed in the IETF and often associated with RSVP (Resource Reservation Protocol).
- ➢ In the second category lies "Differentiated Services," which is probably the most widely deployed QoS mechanism.

UNIT V APPLICATION LAYER

Electronic Mail (SMTP, POP3, IMAP, MIME) – HTTP – DNS – FTP – Telnet – Web Services - SNMP – MIB - RMON

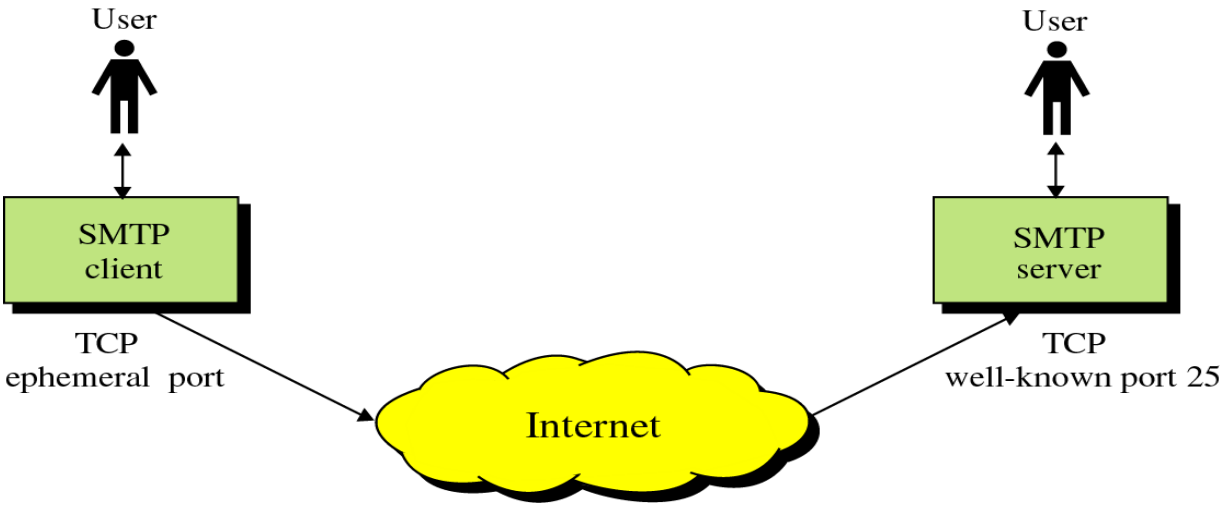
SMTP - Simple Mail Transfer Protocol

- The TCP/IP protocol supports electronic mail on the Internet is called Simple Mail Transfer (SMTP). It is a system for sending messages to other computer users based on e-mail addresses
- SMTP provides mail exchange between users on the same or different computers.
-

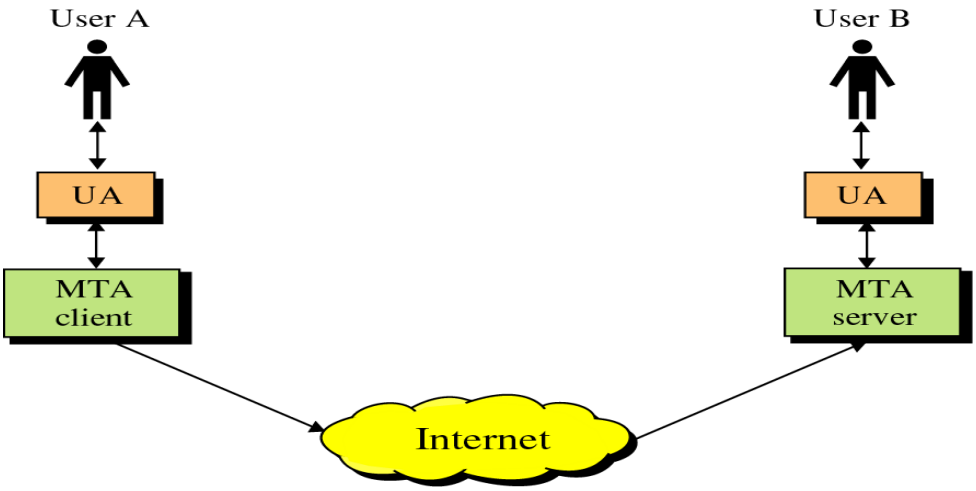


Protocol originated in 1982 (RFC821, Jon Postel)

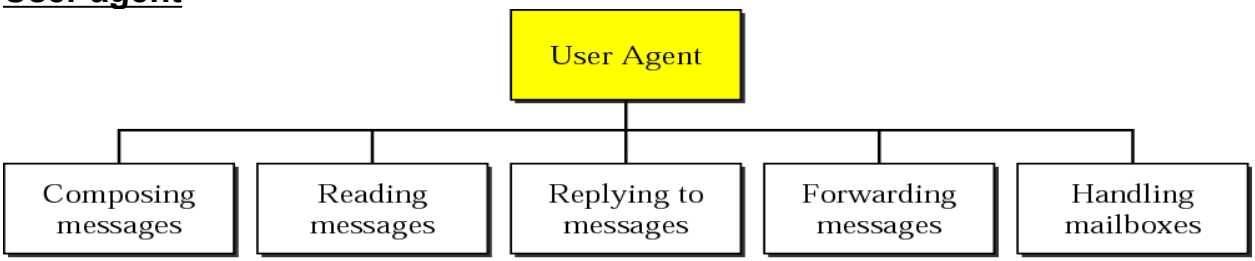
- Standard message format (RFC822,2822, D. Crocker)
- Goal: To transfer mail reliably and efficiently



- SMTP clients and servers have two main components
  - User Agents – Prepares the message, encloses it in an envelope. (ex. Thunderbird, Eudora)
  - Mail Transfer Agent – Transfers the mail across the internet (ex. Sendmail, Exim)
  - Analogous to the postal system in many ways



User agent

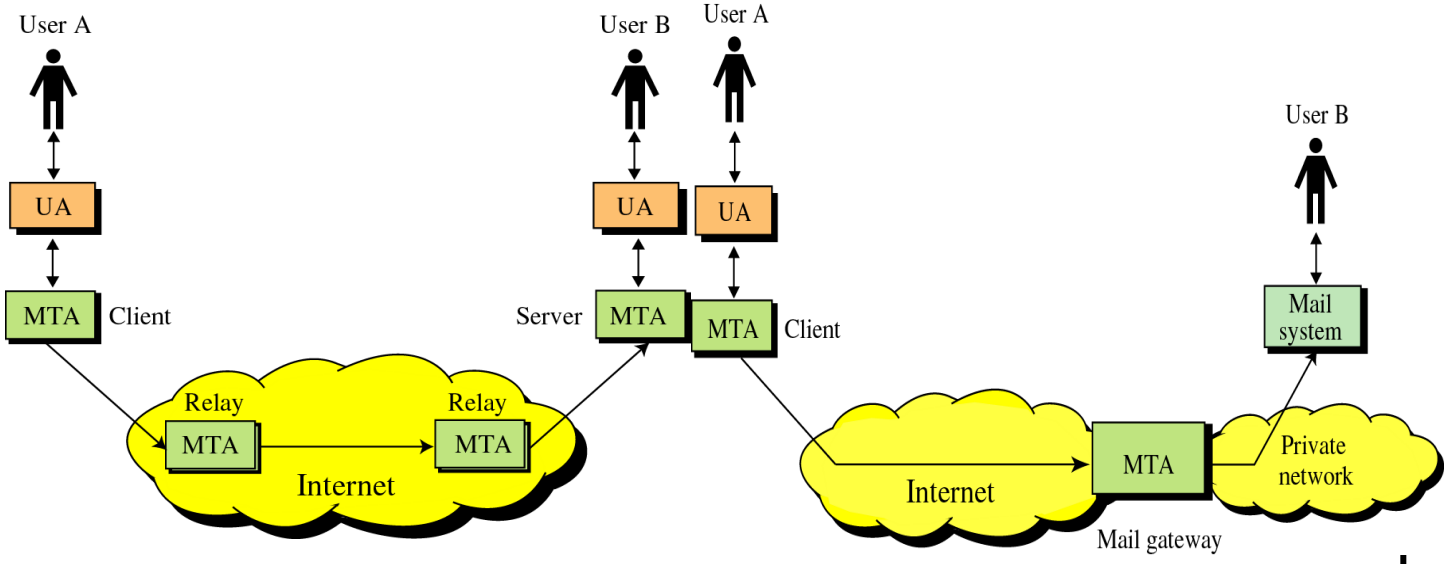


Some examples of command-driven user agents are mail, pine, and elm
Some examples of GUI-based user agents are Eudora, Outlook, and Netscape.

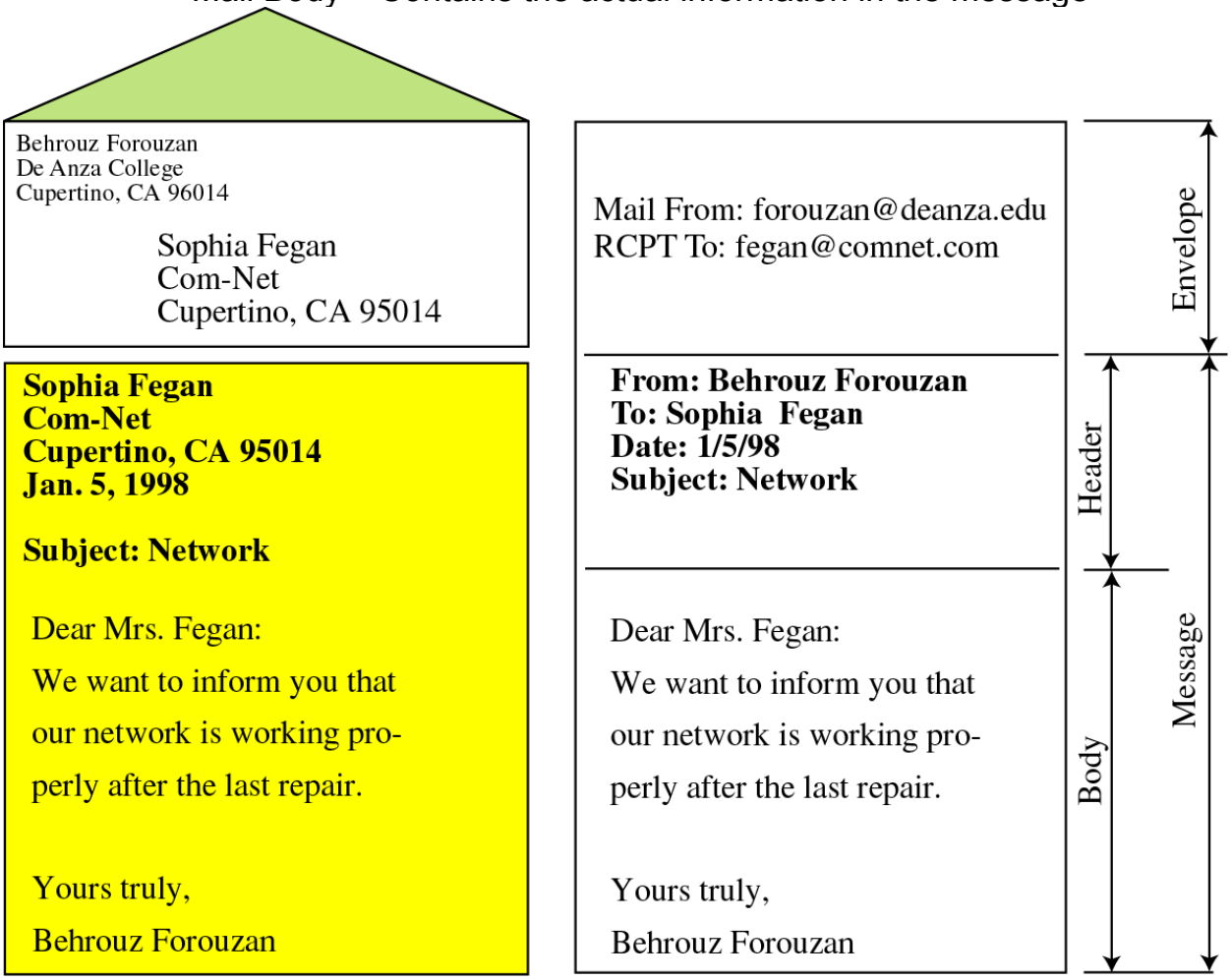
- SMTP also allows the use of Relays allowing other MTAs to relay the mail
- Mail Gateways are used to relay mail prepared by a protocol other than SMTP and convert it to SMTP

Format of an email

- Mail is a text file
- Envelope –
  - sender address
  - receiver address
  - other information

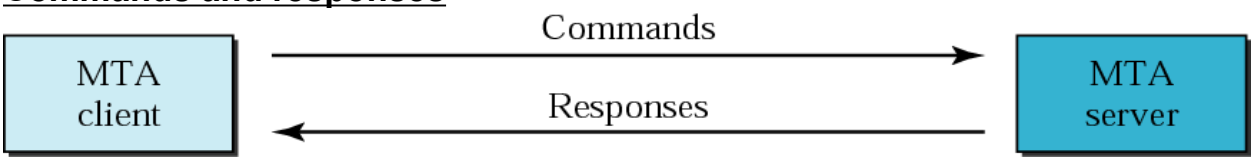


- Message –
  - Mail Header – defines the sender, the receiver, the subject of the message, and other information
  - Mail Body – Contains the actual information in the message



E-mail address

Commands and responses



Commands

Keyword	*Argument(s)*
HELO	Sender's host name
MAIL FROM	Sender of the message
RCPT TO	Intended recipient of the message
DATA	Body of the mail
QUIT	
RSET	
VRFY	Name of recipient to be verified
NOOP	
TURN	
EXPN	Mailing list to be expanded
HELP	Command name
SEND FROM	Intended recipient of the message
SMOL FROM	Intended recipient of the message
SMAL FROM	Intended recipient of the message

Responses

Code	*Description*
	Positive Completion Reply
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
	Positive Intermediate Reply
354	Start mail input
	Transient Negative Completion Reply
421	Service not available
450	Mailbox not available
451	Command aborted: local error
452	Command aborted; insufficient storage

Code	*Description*
Positive Completion Reply	
211	System status or help reply
214	Help message
220	Service ready
221	Service closing transmission channel
250	Request command completed
251	User not local; the message will be forwarded
Positive Intermediate Reply	
354	Start mail input
Transient Negative Completion Reply	
421	Service not available
450	Mailbox not available
451	Command aborted: local error
452	Command aborted; insufficient storage
Permanent Negative Completion Reply	
500	Syntax error; unrecognized command
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command temporarily not implemented
550	Command is not executed; mailbox unavailable
551	User not local
552	Requested action aborted; exceeded storage location
553	Requested action not taken; mailbox name not allowed
554	Transaction failed

How SMTP works (A-PDU's)

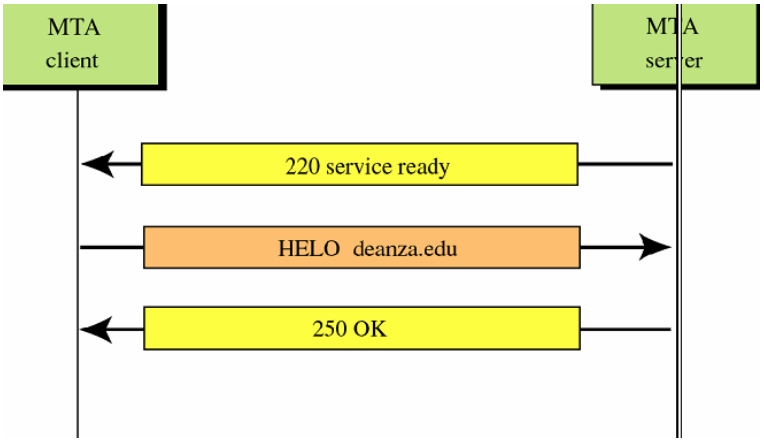
Keyword	Arguments
HELO	Sender's Host Domain Name
MAIL FROM:	Email Address of sender
RCPT TO:	Email of Intended recipient
DATA	Body of the message
QUIT	
RSET	
VRFY	Name to be verified
NOOP	
TURN	
EXPN	Mailing list to expand
HELP	Command Name

Status Codes

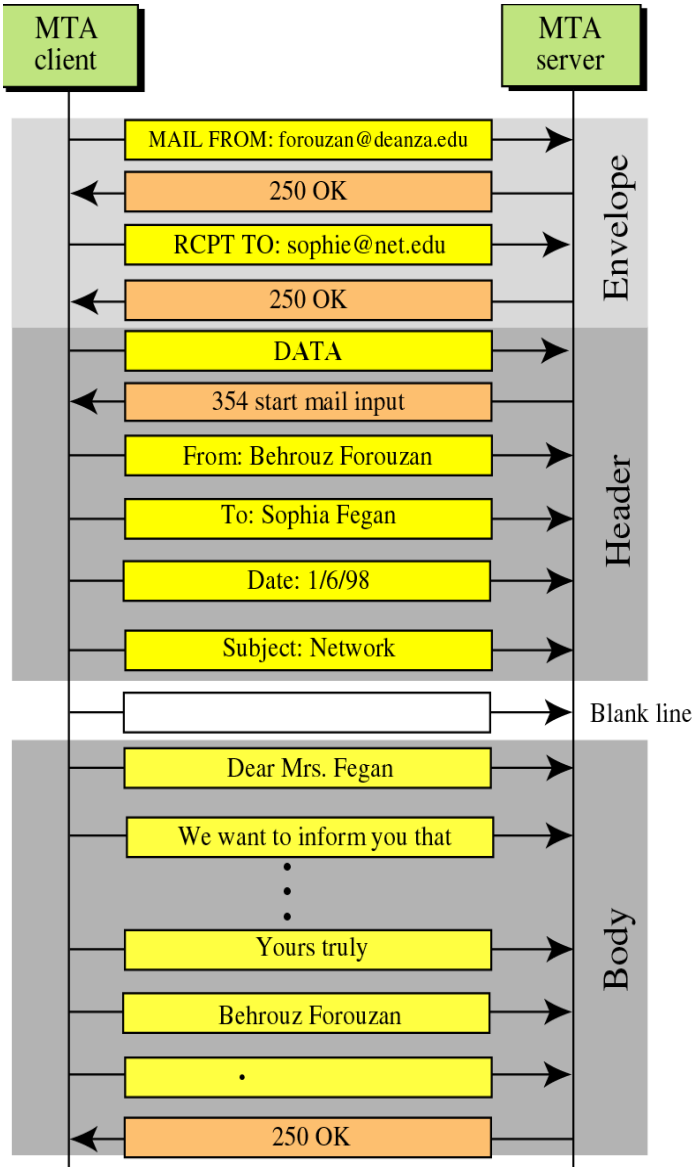
The Server responds with a 3 digit code that may be followed by text info

- 2## - Success
- 3## - Command can be accepted with more information
- 4## - Command was rejected, but error condition is temporary
- 5## - Command rejected, Bad User!

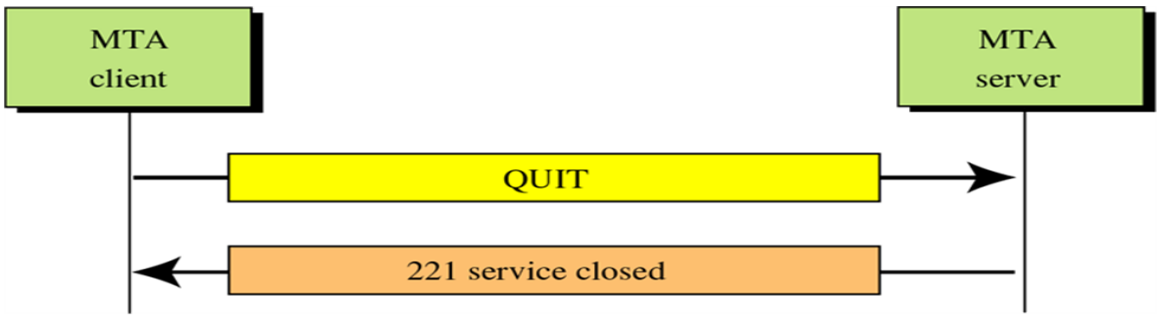
Connection Establishment



Message Progress



Connection Termination



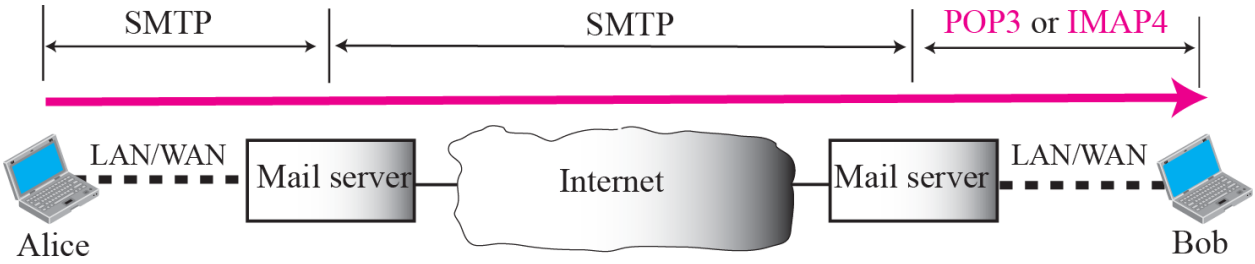
TCP Connection Termination

Limitations in SMTP

- Only uses NVT 7 bit ASCII format
  - How to represent other data types?
- No authentication mechanisms
- Messages are sent un-encrypted
- Susceptible to misuse (Spamming,faking sender address)

POP 3 (*Post Office Protocol)*

- Short for ***P**ost **O**ffice **P**rotocol,* a protocol used to retrieve e-mail from a mail server. Most e-mail applications (sometimes called an *e-mail client*) use the POP protocol, although some can use the newer IMAP (Internet Message Access Protocol).
- There are two versions of POP.
- The first, called *POP2,* became a standard in the mid-80's and requires SMTP to send messages. The newer version, POP3, can be used with or without SMTP. POP3 uses TCP/IP port 110.



- Workstations interact with the SMTP host, which receives the mail on behalf of every host in the organization, to retrieve messages by using a client-server protocol such as Post Office Protocol, version 3(POP3). Although POP3 is used to download messages from the server, the SMTP client still needed on the desktop to forward messages from the workstation user to its SMTP mail server.

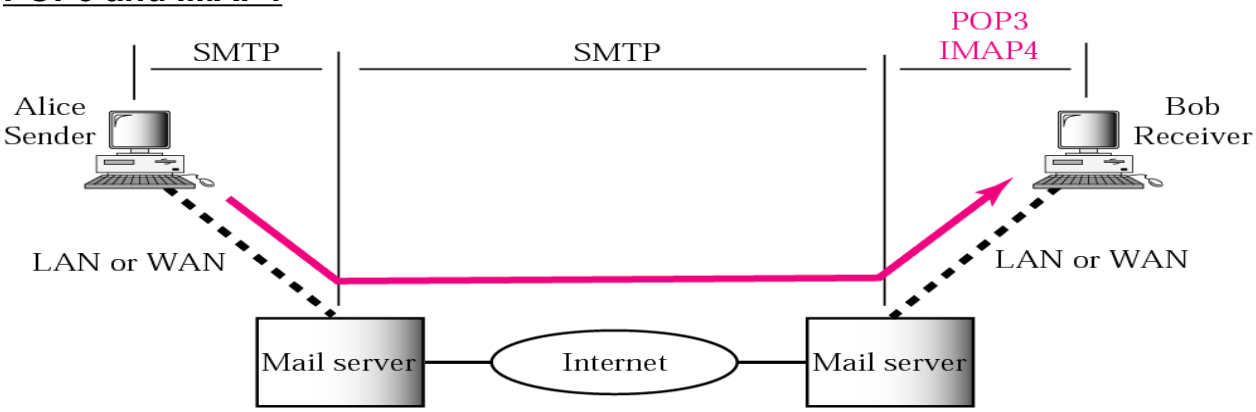
Post Office Protocol v3

- Simple
- Allows the user to obtain a list of their Emails
- Users can retrieve their emails
- Users can either delete or keep the email on their system
- Minimizes server resources
- Post Office Protocol, version 3 (POP3) is simple and limited in functionality.
- The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server.

- Mail access starts with the client when the user needs to download e-mail from the mailbox on the mail server.

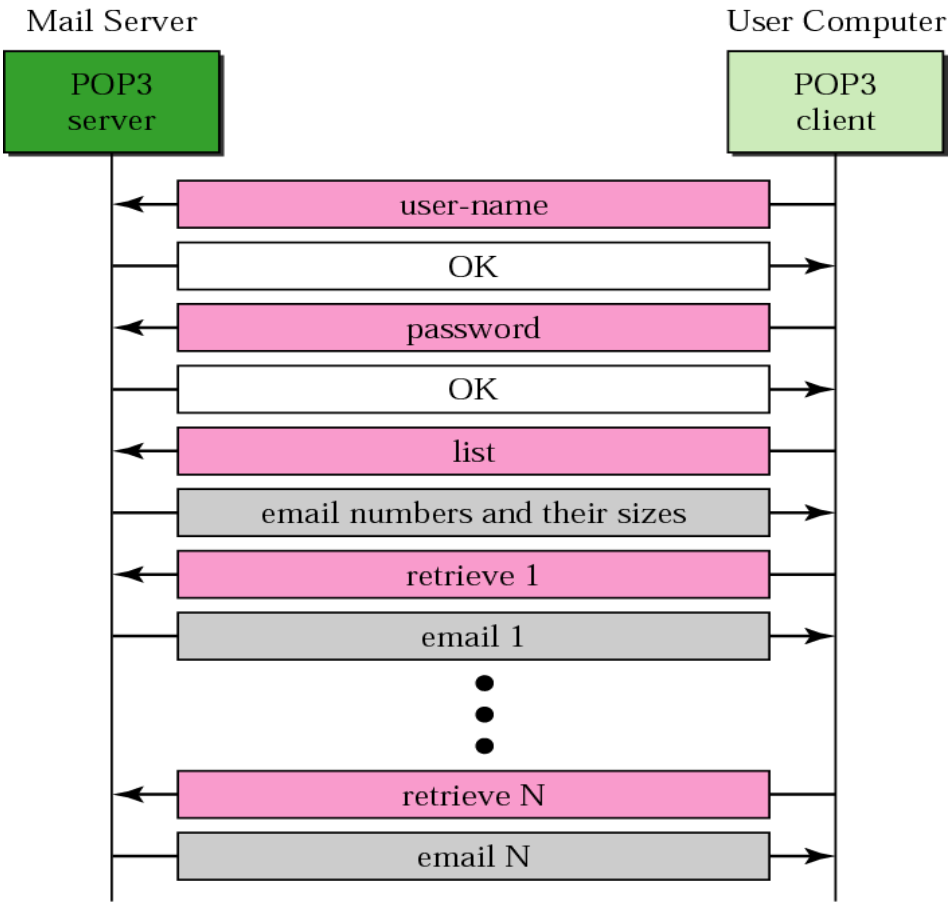
- The client opens a connection to the server on TCP port 110. It then sends its user name and password to access the mailbox.
- The user can then list and retrieve the mail messages, one by one.
- Figure shows an example of downloading using POP3. POP3 has two modes: the delete mode and the keep mode.
- In the delete mode, the mail is deleted from the mailbox after each retrieval. In the keep mode, the mail remains in the mailbox after retrieval.
- The delete mode is normally used when the user is working at her permanent computer and can save and organize the received mail after reading or replying.
- The keep mode is normally used when the user accesses her mail away from her primary computer (e.g., a laptop). The mail is read but kept in the system for later retrieval and organizing

POP3 and IMAP4

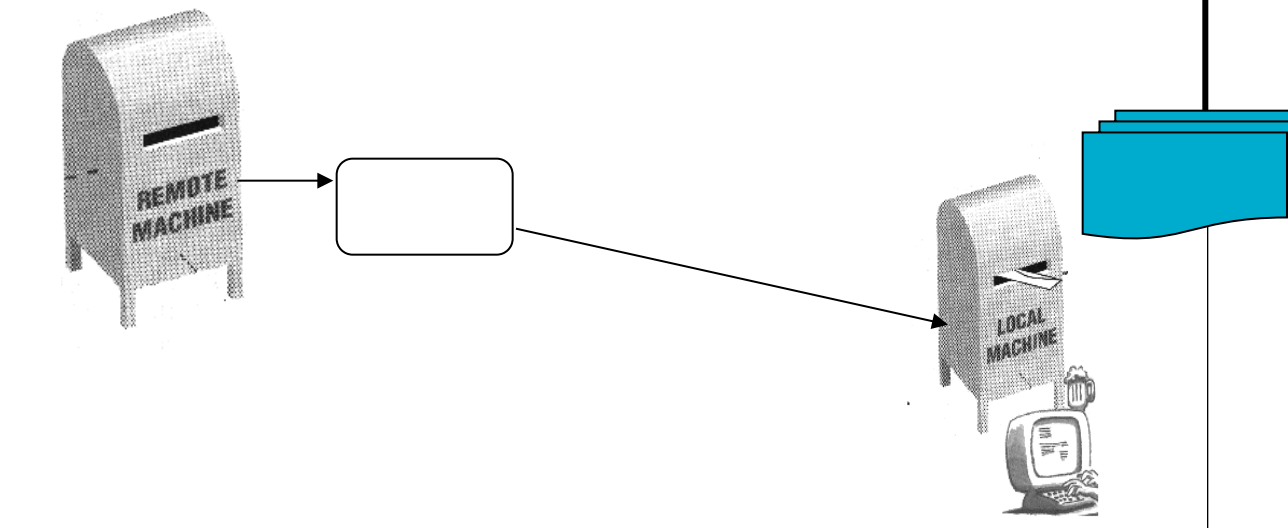


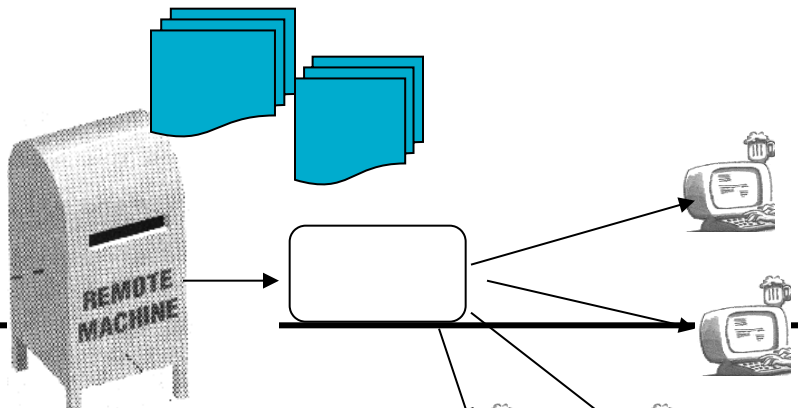
POP3 Commands

- **USER** name: User name for authentication
- **PASS** password: Password used for authentication
- **STAT:** Get number and total size of message
- **LIST:** [msg] get size of message
- **RETR:** msg Send message to client
- **DELE:** msg Delete message from mailbox
- **RSET:** Cancel previous delete requests.
- **QUIT:** Updates mailbox (deletes messages) and quits.


Mail Server
User Computer
POP3 server
POP3 client
user-name
OK
password
OK
list
email numbers and their sizes
retrieve 1
email 1
retrieve N
email N

POP vs. IMAP


REMOTE MACHINE
LOCAL MACHINE


REMOTE MACHINE

POP3 is deficient in several ways.

- It does not allow the user to organize her mail on the server; the user cannot have different folders on the server. (Of course, the user can create folders on her own computer.)
- POP3 does not allow the user to partially check the contents of the mail before downloading.

Pop vs IMAP

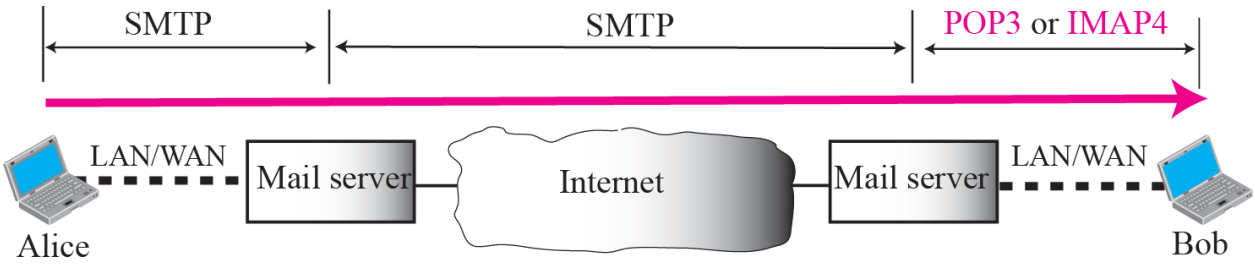
- Similarities
  - Mail delivered to a shared, constanly connected server
  - New mail accessible anywhere in network on a variety of platforms
  - For access only, Need SMTP to send mail
- Differences
  - POP simpler and more established (more clients and servers that support it)
  - IMAP is stateful protocol with more features

IMAP (Internet Message Access Protocol)

- ✓ IMAP is an **Internet Message Access Protocol**. It is a method of accessing electronic mail messages that are kept on a possibly shared mail server.
- ✓ In other words, it permits a "client" email program to access remote message stores as if they were local.
- ✓ For example, email stored on an IMAP server can be manipulated from a desktop computer at home, a workstation at the office, and a notebook computer while travelling, without the need to transfer messages or files back and forth between these computers. IMAP uses TCP/IP port 143.

Internet Mail Access Protocol v4

- Has more features than POP3
- User can check the email header before downloading
- Emails can be accessed from any location
- Can search the email for a specific string of characters before downloading
- User can download parts of an email
- User can create, delete, or rename mailboxes on a server
- IMAP4 is similar to POP3, but it has more features; IMAP4 is more powerful and more complex.

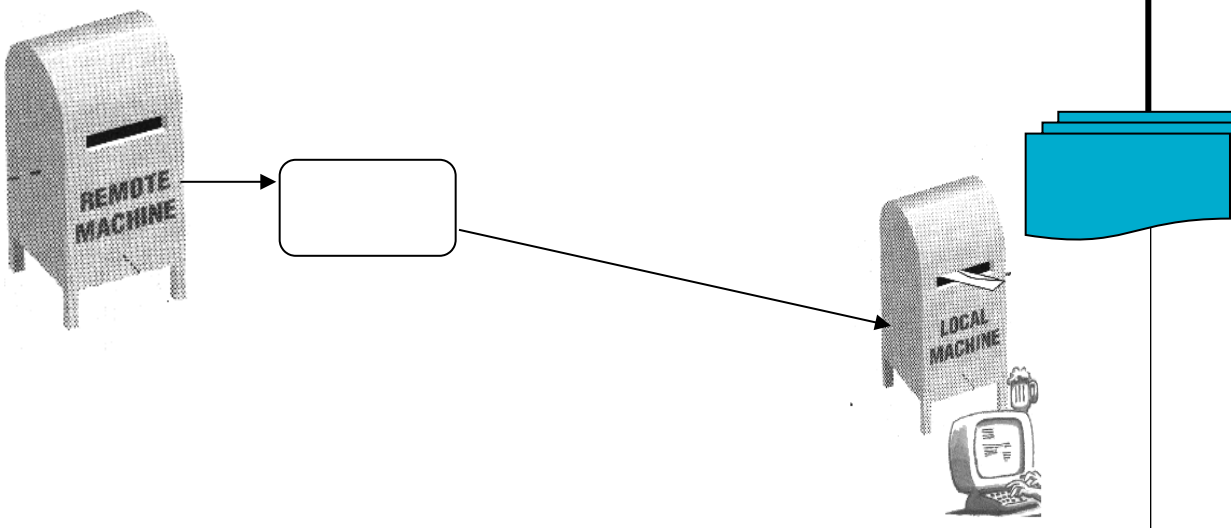


- It defines an abstraction known as a MAILBOX. Mailboxes are located on the same computer as a server.
- IMAP4 is a method for accessing electronic mail messages that are kept on a mail server. It permits a client e-mail program to view and manipulate those messages.
- Electronic mail stored on an IMAP server can be viewed or manipulated from a desktop computer at home,a notebook computer, or at a workstation. We can also say that mail messages can be accessed from multiple locations.

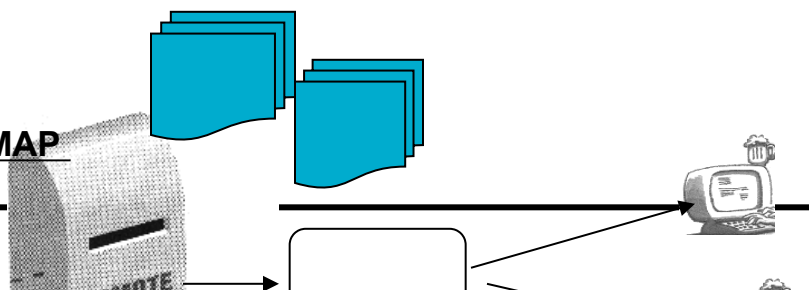
Functions of IMAP4

- IMAP provides extended functionality for message retrieval and processing.
- Users can obtain information about a message or examine header fields without retrieving the entire message.
- Users can search for a specified string and retrieve portions of a message. This is useful for slow-speed dialup connections since they wont need to download useless information.

POP vs. IMAP



Pop vs IMAP



Similarities

- ✓ Mail delivered to a shared, constanly connected server
- ✓ New mail accessible anywhere in network on a variety of platforms
- ✓ For access only, Need SMTP to send mail

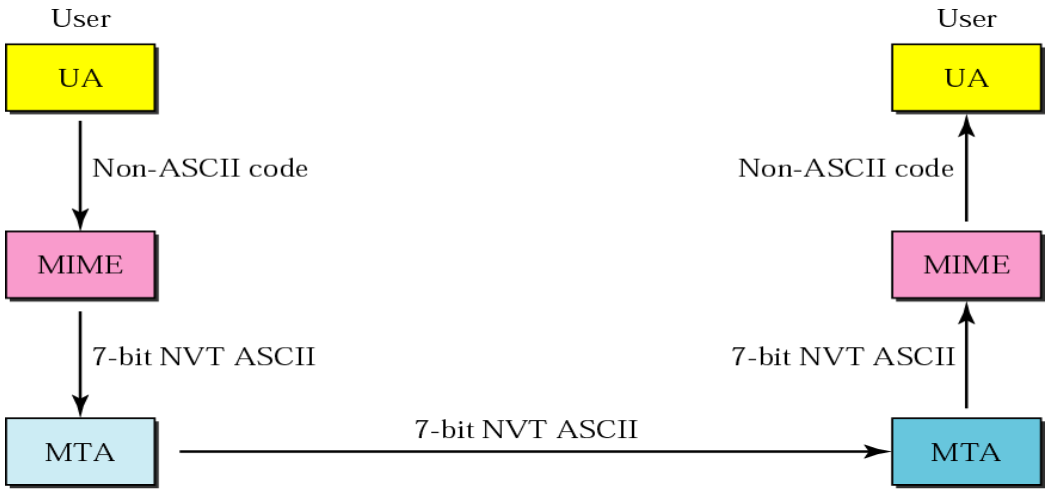
Differences

- ✓ POP simpler and more established (more clients and servers that support it)
- ✓ IMAP is stateful protocol with more features

- With IMAP, all your mail stays on the server in multiple folders, some of which you have created. This enables you to connect to any computer and see all your mail and mail folders. In general, IMAP is great if you have a dedicated connection to the Internet or you like to check your mail from various locations.
- With POP3 you only have one folder, the Inbox folder. When you open your mailbox, new mail is moved from the host server and saved on your computer. If you want to be able to see your old mail messages, you have to go back to the computer where you last opened your mail.
- With POP3 "leave mail on server" only your email messages are on the server, but with IMAP your email folders are also on the server.

Multipurpose Internet Mail Extensions (MIME)

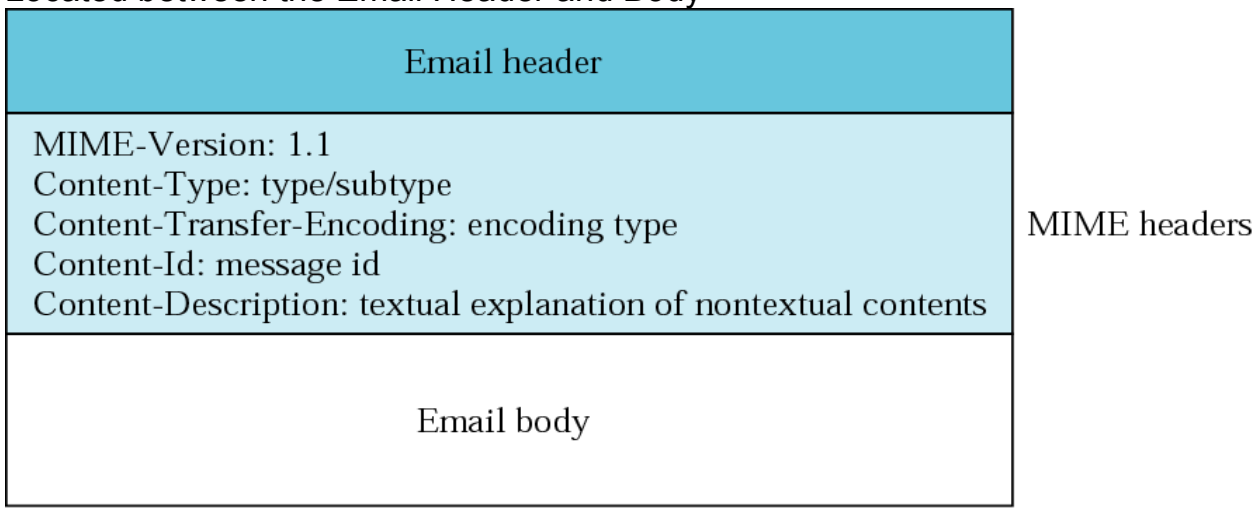
- MIME is defined to allow transmission of non-ASCII or arbitrary data through a standard e-mail message.
- In other words, MIME has a mechanism for sending multimedia data over e-mail.
- Transforms non-ASCII data to NVT (Network Virtual Terminal) ASCII data
  - ✓ Text
  - ✓ Application
  - ✓ Image
  - ✓ Audio
  - ✓ Video



- Electronic mail has a simple structure. Its simplicity, however, comes at a price. It can send messages only in NVT 7-bit ASCII format.
  It has some limitations. For example, it cannot be used for languages that are not supported by 7-bit ASCII characters (such as French, German, Hebrew, Russian, Chinese, and Japanese). Also, it cannot be used to send binary files or video or audio data.
- Multipurpose Internet Mail Extensions (MIME) is a supplementary protocol that allows non-ASCII data to be sent through e-mail.
- MIME transforms non-ASCII data at the sender site to NVT ASCII data and delivers them to the client MTA to be sent through the Internet.
- The message at the receiving side is transformed back to the original data. We can think of MIME as a set of software functions that transforms non-ASCII data (stream of bits) to ASCII data and vice versa, as shown

MIME Headers

- Located between the Email Header and Body



- Content-Type – Type of data used in the Body
  - Text: plain, unformatted text; HTML

- Multipart: Body contains different data types
- Message: Body contains a whole, part, or pointer to a message
- Image: Message contains a static image (JPEG, GIF)
- Video: Message contains an animated image (MPEG)
- Audio: Message contains a basic sound sample (8kHz)
- Application: Message is of data type not previously defined

- Content-Transfer-Encoding – How to encode the message
  - 7 bit – no encoding needed
  - 8 bit – Non-ASCII, short lines
  - Binary – Non-ASCII, unlimited length lines
  - Base64 – 6 bit blocks encoded into 8-bit ASCII
  - Quoted-printable – send non-ASCII characters as 3 ASCII characters, =##, ## is the hex representation of the byte

Data types and subtypes in MIME

Type	*Subtype*	*Description*
Text	Plain	Unformatted
	HTML	HTML format (see Chapter 22)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the same message

Type	*Subtype*	*Description*
Message	RFC822	Body is an encapsulated message
	Partial	Body is a fragment of a bigger message
	External-Body	Body is a reference to another message
Image	JPEG	Image is in JPEG format
	GIF	Image is in GIF format
Video	MPEG	Video is in MPEG format
Audio	Basic	Single channel encoding of voice at 8 KHz
Application	PostScript	Adobe PostScript
	Octet-stream	General binary data (eight-bit bytes)

Content-transfer-encoding

Type	*Description*
7bit	NVT ASCII characters and short lines
8bit	Non-ASCII characters and short lines
Binary	Non-ASCII characters with unlimited-length lines
Base64	6-bit blocks of data are encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code

MIME Messages

- MIME information is contained in the mail header using standard RFC 2822 format.
- MIME header specifies version, data type, encoding used to convert the data to ASCII.
- Example:
- From: bill@college.edu
- To: john@example.com
- MIME-Version: 1.0
- Content-Type: image/jpeg
- Content-Transfer-Encoding: base64
- ..data for the image..

MIME Multipart Messages

MIME multipart messages within the Content-Type adds considerable flexibility.

- There are four subtypes for a multipart message. The four subtypes are:
  - mixed: allows a single message to contain multiple, independent sub-messages each having its independent type and encoding.
  - alternative: allows a single message include multiple representations of the same data.
  - parallel: allows a single message to include subparts that should be viewed together. (such as video and audio subparts)
  - digest: allows a single message to contain a set of other messages.
- To summarize, a multipart message can contain both a short text explaining the purpose of the message and some non-textual information

Example:

From: carrieerpdgns@yahoo.com

To: john@example.com
MIME-Version: 1.0
Content-Type: Multipart/Mixed; Boundary=StartOfNextPart
--StartOfNextPart
Content-Type: text/plain
Content-Transfer-Encoding: 7bit
John,
Here is the photo of the carrier pigeons I saw last week.
Sincerely,
Carrie Erpigeons
--StartOfNextPart
Content-Type: image/gif
Content-Transfer-Encoding: base64
..data for the image..
The keyword “Boundary=“ is used to separte parts of the message. StartOfNextPart is used to serve as the boundary.

HTTP - Hypertext Transfer Protocol

- *HTTP* - the Hypertext Transfer Protocol - provides a standard for Web browsers and servers to communicate. The definition of HTTP is a technical specification of a network protocol that software must implement.
- HTTP is an application layer network protocol built on top of TCP. HTTP clients (such as Web browsers) and servers communicate via HTTP messages.

- HTTP is the protocol that supports communication between web browsers and web servers.
- A “Web Server” is a HTTP server
- Most clients/servers today speak version 1.1, but 1.0 is also in use.
  - RFC 1945 (HTTP 1.0)
  - RFC 2616 (HTTP 1.1)

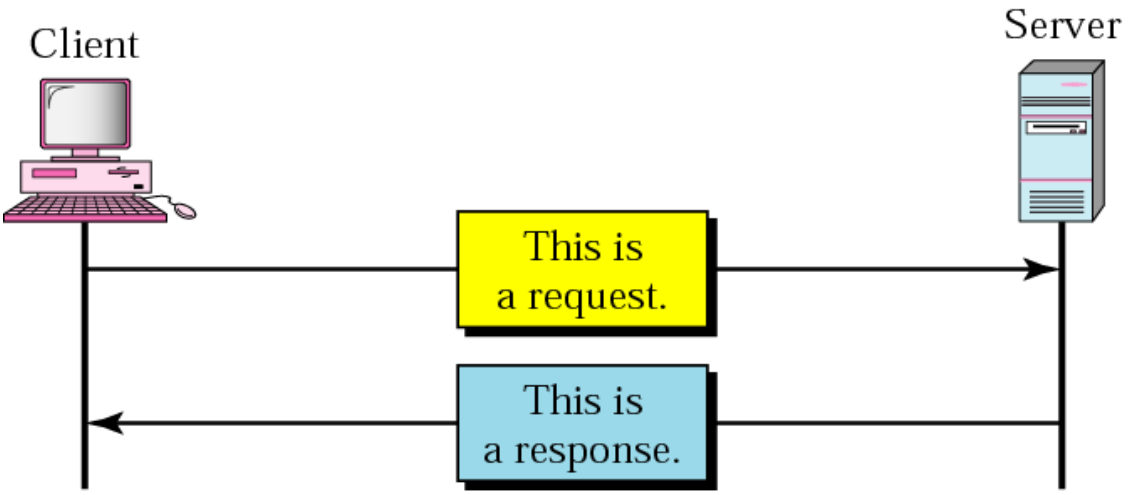
From the RFC
- “HTTP is an application-level protocol with the lightness and speed necessary for distributed, hypermedia information systems.”
- Transport Independence
  - The HTTP protocol generally takes place over a TCP connection,
  - but the protocol itself is not dependent on a specific transport layer.
- Protocol for transfer of various data formats between server and client
  - Plaintext
  - Hypertext
  - Images

  - Video
  - Sound
- Meta-information also transferred
- The rules governing the conversation between a Web client and a Web server
- ***HTTP uses the services of TCP on well-known port 80.***

HTTP transaction

Two types of messages are

1. Request message
2. Response message



- HTTP has a simple structure:
  - client sends a request
  - server returns a reply.
  - HTTP can support multiple request-reply exchanges over a single TCP connection.
- The “well known” TCP port for HTTP servers is port 80.
  - Other ports can be used as well...

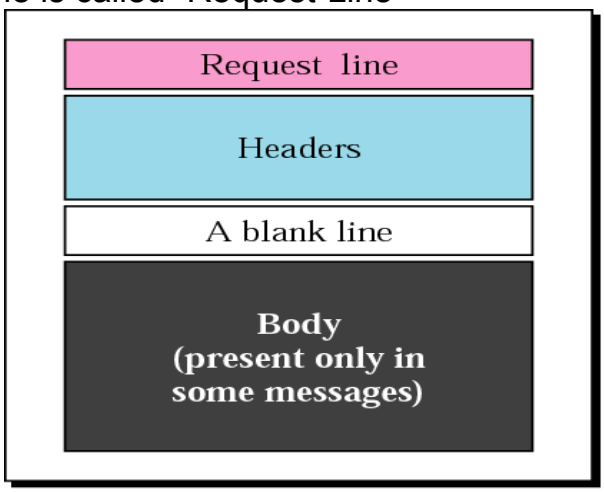
Request Messages

The first line of an HTTP request message specifies three things:
  - the operation to be performed,
  - the Web page the operation should be performed on
  - and the version of HTTP being used.

Although HTTP defines a wide assortment of possible request operations—including “write” operations that allow a Web page to be posted on a

server—the two most common operations are GET (fetch the specified Web page) and HEAD (fetch status information about the specified Web page).

- Lines of text (ASCII).
- Lines end with CRLF **"\r\n"**
- First line is called "Request-Line"



Request message

Request line

URL

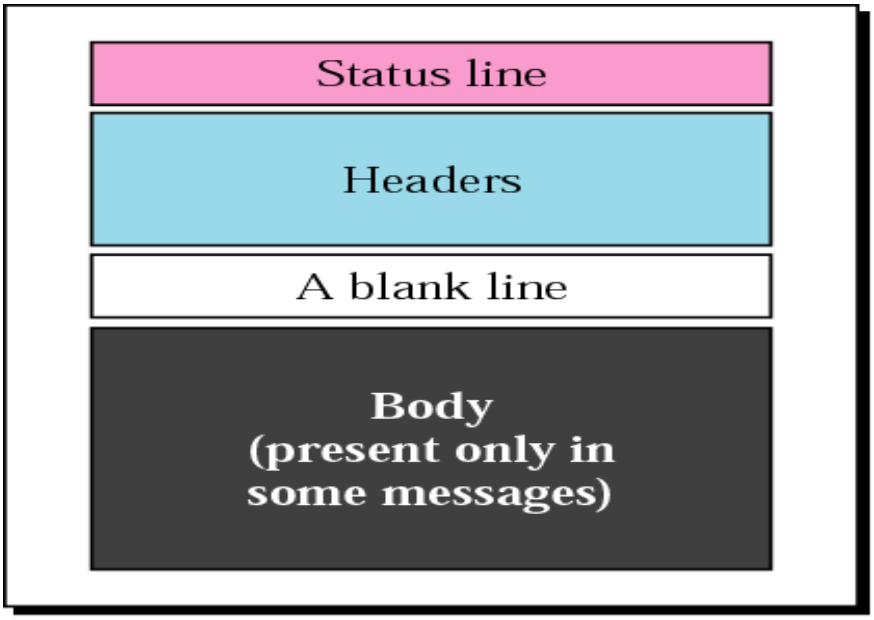
HTTP request operations

Operation	Description
OPTIONS	Request information about available options
GET	Retrieve document identified in URL
HEAD	Retrieve metainformation about document identified in URL
POST	Give information (e.g., annotation) to server
PUT	Store document under specified URL
DELETE	Delete specified URL
TRACE	Loopback request message
CONNECT	For use by proxies

Response Messages

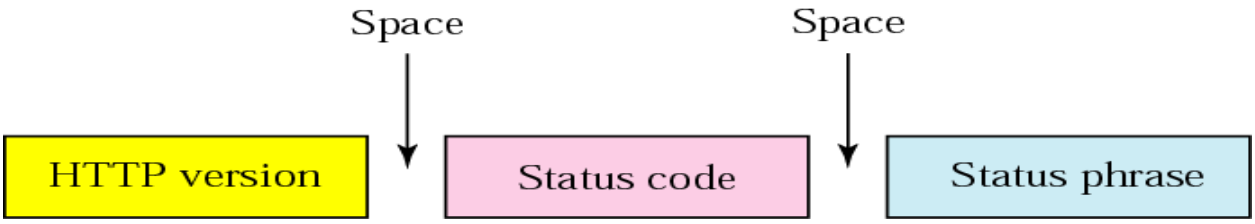
Like request messages, response messages begin with a single START LINE.

- In this case, the line specifies the version of HTTP being used, a three-digit code indicating whether or not the request was successful, and a text string giving the reason for the response.
- ASCII Status Line
- Headers Section
- Content can be anything (not just text)
    - typically an HTML document or some kind of image.



Response message

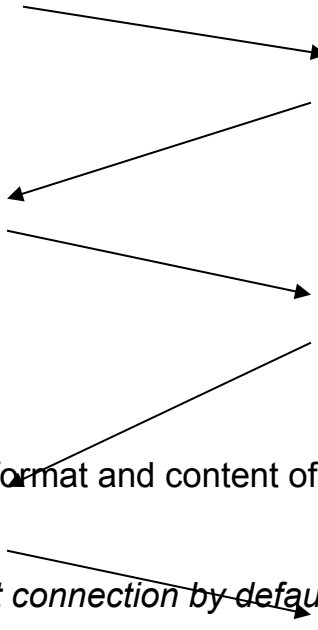
Status line



Five types of HTTP result codes

Code	Type	Example Reasons
1xx	Informational	request received, continuing process
2xx	Success	action successfully received, understood, and accepted
3xx	Redirection	further action must be taken to complete the request
4xx	Client Error	request contains bad syntax or cannot be fulfilled
5xx	Server Error	server failed to fulfill an apparently valid request

An HTTP conversation



HTTP is the set of rules governing the format and content of the conversation between a Web client and server

HTTP version 1.1 specifies a persistent connection by default.

Uniform Resources

- URL
  - Uniform Resource Locator
  - Refers to an existing protocol
    - http:, wais:, ftp:, mailto:, gopher:, news:
  - Points to a document on a specific server
- URN
  - Uniform Resource Name
  - Globally unique, persistent identifier
    - Independent of location
- URI
  - Uniform Resource Identifier
  - Collection of URL's and URN's

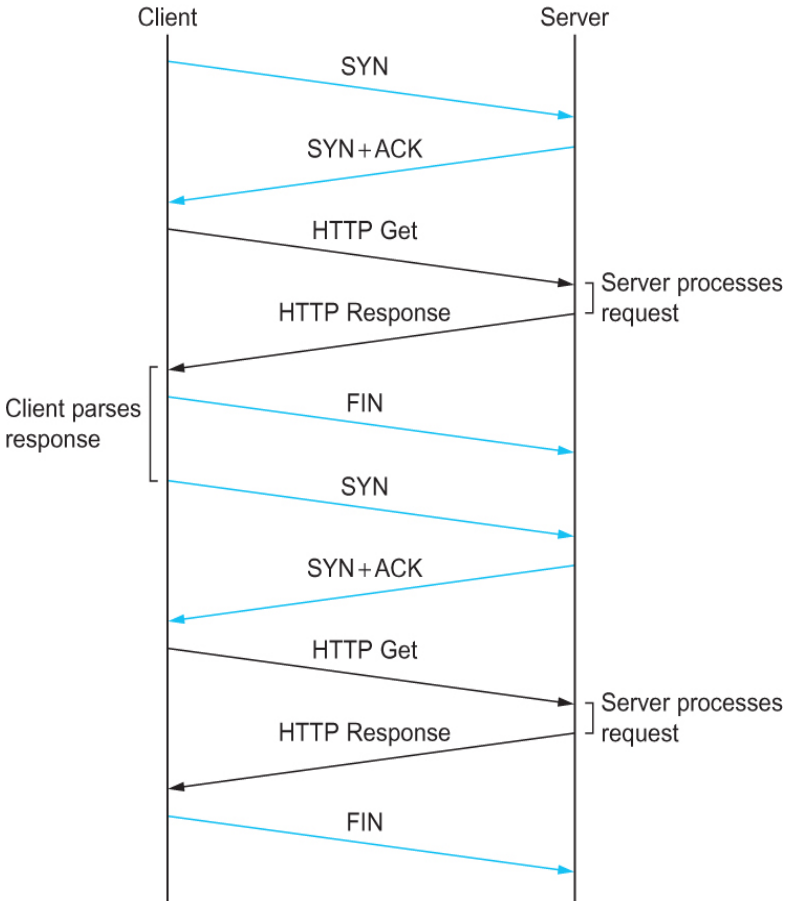
TCP Connections

- The original version of HTTP (1.0) established a separate TCP connection for each data item retrieved from the server.
- It's not too hard to see how this was a very inefficient mechanism: connection setup and teardown messages had to be exchanged between the client and server even if all the client wanted to do was verify that it had the most recent copy of a page.
- Thus, retrieving a page that included some text and a dozen icons or other small graphics would result in 13 separate TCP connections being established and closed.
- To overcome this situation, HTTP version 1.1 introduced *persistent connections*— the client and server can exchange multiple request/response messages over the same TCP connection.

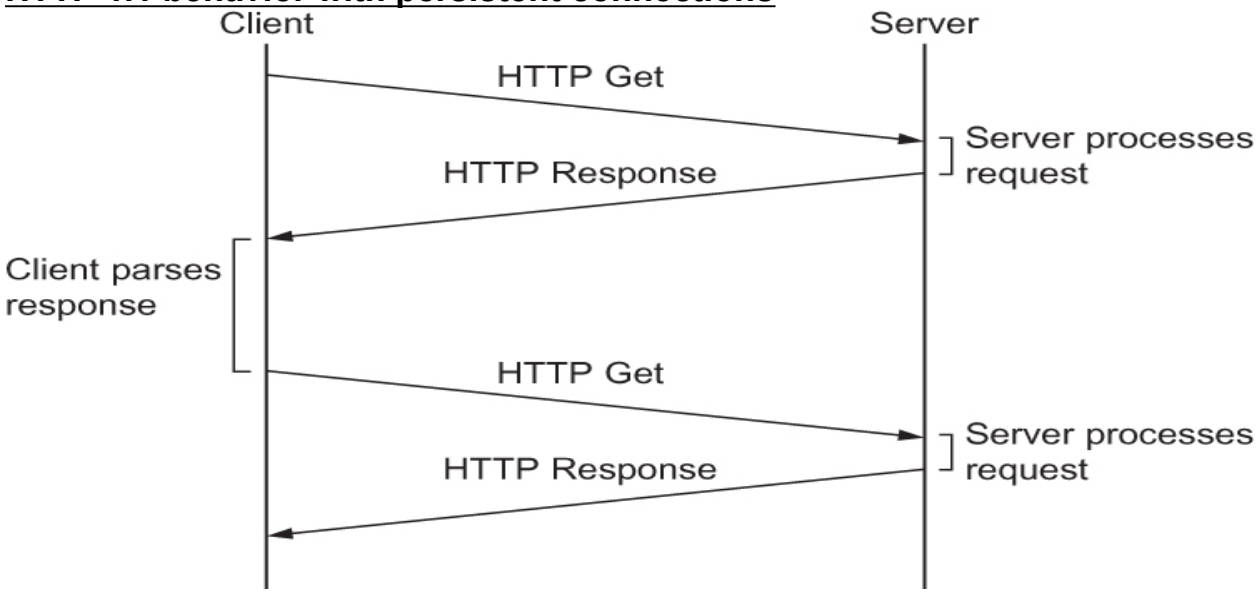
Persistent connections have many advantages.

- First, they obviously eliminate the connection setup overhead, thereby reducing the load on the server, the load on the network caused by the additional TCP packets, and the delay perceived by the user.
- Second, because a client can send multiple request messages down a single TCP connection, TCP's congestion window mechanism is able to operate more efficiently.
- This is because it's not necessary to go through the slow start phase for each page.

HTTP 1.0 behavior



HTTP 1.1 behavior with persistent connections



- Much of the motivation for enabling direct application-to-application communication comes from the business world.
- Historically, interactions between enterprises—businesses or other organizations—have involved some manual steps such as filling out an order form or making a phone call to determine whether some product is in stock.
- Even within a single enterprise it is common to have manual steps between software systems that cannot interact directly because they were developed independently.
- Increasingly such manual interactions are being replaced with direct application-to application interaction.
- An ordering application at enterprise A would send a message to an order fulfillment application at enterprise B, which would respond immediately indicating whether the order can be filled.
- Perhaps, if the order cannot be filled by B, the application at A would immediately order from another supplier, or solicit bids from a collection of suppliers.
- Two architectures have been advocated as solutions to this problem.
- Both architectures are called *Web Services,* taking their name from the term for the individualapplications that offer a remotely-accessible service to client applications to form network applications.
- The terms used as informal shorthand to distinguish the two Web Services architectures are *SOAP and REST (as in, "the SOAP vs. REST debate").*
- The SOAP architecture's approach to the problem is to make it feasible, at least in theory, to generate protocols that are customized to each network application.
- The key elements of the approach are a framework for protocol specification, software toolkits for automatically generating protocol implementations from the specifications, and modular partial specifications that can be reused across protocols.

Custom Application Protocols (WSDL, SOAP)

- The architecture informally referred to as SOAP is based on *Web Services Description Language (WSDL) and SOAP.4*
- Both of these standards are issued by the World Wide Web Consortium (W3C).
- This is the architecture that people usually mean when they use the term Web Services without any preceding qualifier.
- Just like the traditional applications described earlier in this chapter, multimedia applications such as telephony and videoconferencing need their own protocols.
- We have already seen a number of protocols that multimedia applications use.
- The Real-Time Transport Protocol (RTP) provides many of the functions that are common to multimedia applications such as conveying timing

information and identifying the coding schemes and media types of an application.

Defining Application Protocols

WSDL has chosen a procedural operation model of application protocols. An abstract Web Service interface consists of a set of named operations, each representing a simple interaction between a client and theWeb Service An operation is analogous to a remotely callable procedure in an RPC system. An example from W3C's WSDL Primer is a hotel reservation Web Service with two operations, CheckAvailability and MakeReservation.

- Each operation specifies a Message Exchange Pattern (MEP) that gives the sequence in which the messages are to be transmitted, including the fault messages to be sent when an error disrupts the message flow.
- MEPs are templates that have placeholders instead of specific message types or formats, so part of the definition of an operation involves specifying which message formats to map into the placeholders in the pattern.
- WSDL nicely separates the parts of a protocol that can be specified abstractly—operations, MEPs, abstract message formats—from the parts that must be concrete. WSDL's concrete part specifies an underlying protocol, how MEPs are mapped onto it, and what bit-level representation is used for messages on the wire.

Defining Transport Protocols

Although SOAP is sometimes called a protocol, it is better thought of as a framework for defining protocols. As the SOAP 1.2 specification explains, "SOAP provides a simple messaging framework whose core functionality is concerned with providing extensibility." SOAP uses many of the same strategies as WSDL, including message formats defined using XML Schema, bindings to underlying protocols, Message Exchange Patterns, and reusable specification elements identified using XML namespaces.

- SOAP is used to define transport protocols with exactly the features needed to support a particular application protocol. SOAP aims to make it feasible to define many such protocols by using reusable components. Each component captures the header information and logic that go into implementing a particular feature. To define a protocol with a certain set of features, just compose the corresponding components. Let's look more closely at this aspect of SOAP.
- The second and more flexible way to implement features involves header blocks. A SOAP message consists of an Envelope, which contains a Header that contains header blocks, and a Body, which contains the payload destined for the ultimate receiver. This message structure is illustrated in Figure

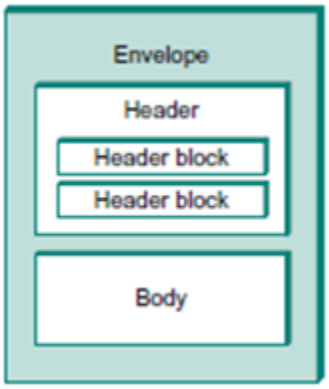


Figure : SOAP message structure

Standardizing Web Services Protocols:

- As we've said, WSDL and SOAP aren't protocols; they are standards for specifying protocols. For different enterprises to implementWeb Services that interoperate with each other, it is not enough to agree to use WSDL and SOAP to define their protocols; they must agree on—standardize—specific protocols.
- The broadest and most widely adopted profile is known as the WS-I Basic Profile. It was proposed by the Web Services Interoperability Organization (WS-I), an industry consortium, while WSDL and SOAP are specified by the World Wide Web Consortium (W3C). The Basic Profile resolves some of the most basic choices faced in defining a Web Service Most notably it requires that WSDL be bound exclusively to SOAP and SOAP be bound exclusively to HTTP and use the HTTP POST method. It also specifies which versions of WSDL and SOAP must be used.

- The payload is a representation of the abstract state of a resource. For example, a GET could return a representation of the current state of the resource, and a POST could send a representation of a desired state of the resource.

A Generic Application Protocol (REST):

- The WSDL/SOAP Web Services architecture is based on the assumption that the best way to integrate applications across networks is via protocols that are customized to each application.
- This model, articulated byWeb architect Roy Fielding, is known as Representational State Transfer (REST). There is no need for a new REST architecture for Web Services—the existing Web architecture is sufficient, although a few extensions are probably necessary.
- An area where WSDL/SOAP may have an advantage is in adapting or wrapping previously written, "legacy" applications to conform to Web Services. This is an important point since most Web Services will be based on legacy applications for the near future at least. These

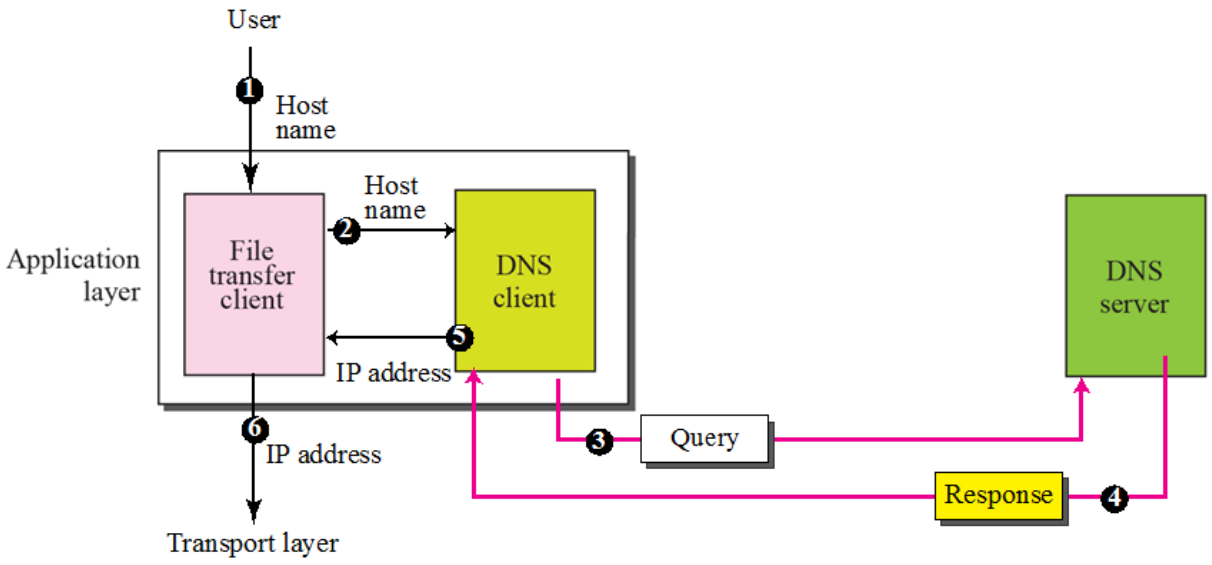
applications usually have a procedural interface that maps more easily into WSDL's operations than REST states.

– The online retailer Amazon.com, as it happens, was an earlyadopter (2002) of Web Services. Interestingly, Amazon made its systems publicly accessible via *both* of the Web Services chitectures, and according to some reports a substantial majority of developers use the REST interface. Of course, this is just one data point and may well reflect factors specific to Amazon.

DNS – Domain Name System

- Domain Name System can map a name to an address and conversely an address to name.
- DNS is a host name to IP address translation service
- DNS is
  - a distributed database implemented in a hierarchy of name servers
  - an application level protocol for message exchange between clients and servers
- To identify an entity, TCP/IP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of numeric addresses. Therefore, we need a system that can map a name to an address or an address to a name.

Purpose of DNS



How does it work?

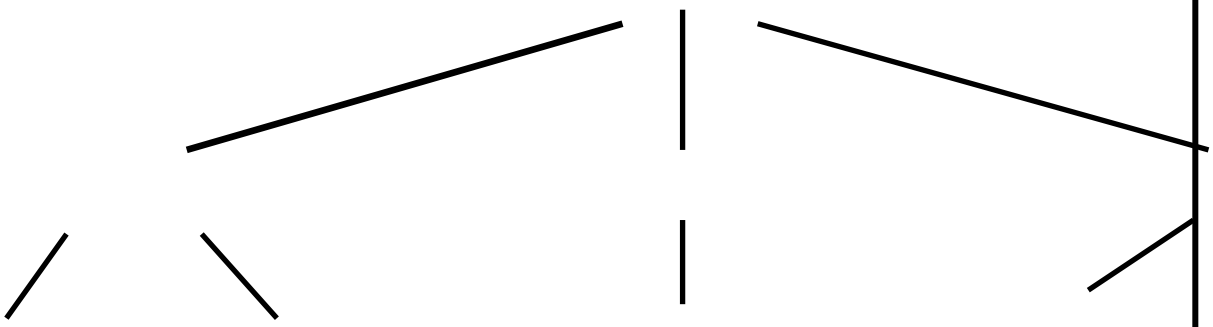
- DNS works by exchanging messages between client and server machines.
- A client application will pass the destination host name to the DNS process (in Unix referred to as the gethostbyname() routine) to get the IP address.
- The application then sits and waits for the response to return.

Why not centralize DNS?

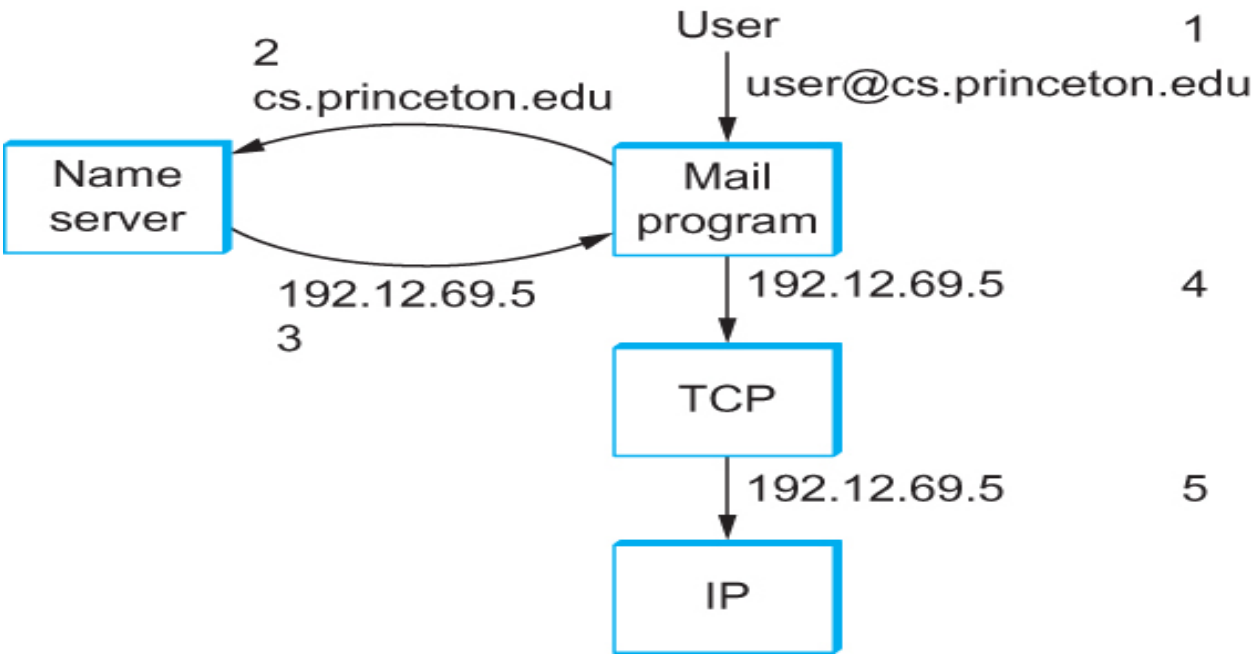
- single point of failure
- traffic volume
- distant centralized database
- maintenance

doesn't *scale!*
So use Distributed, Hierarchical Database

Distributed, Hierarchical Database



Infrastructure Services
Name Service (DNS)



Names translated into addresses, where the numbers 1–5 show the sequence of steps in the process

DNS Components

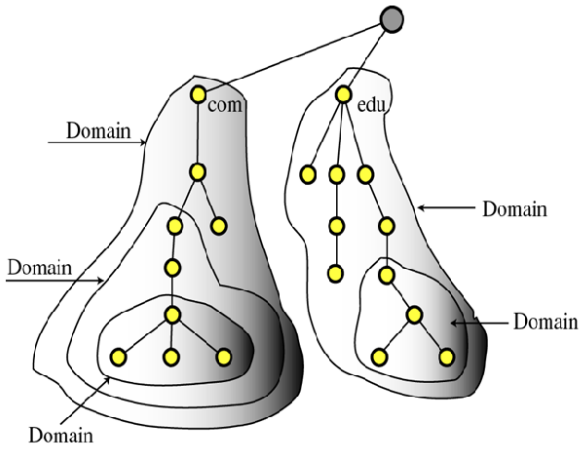
There are 3 components:

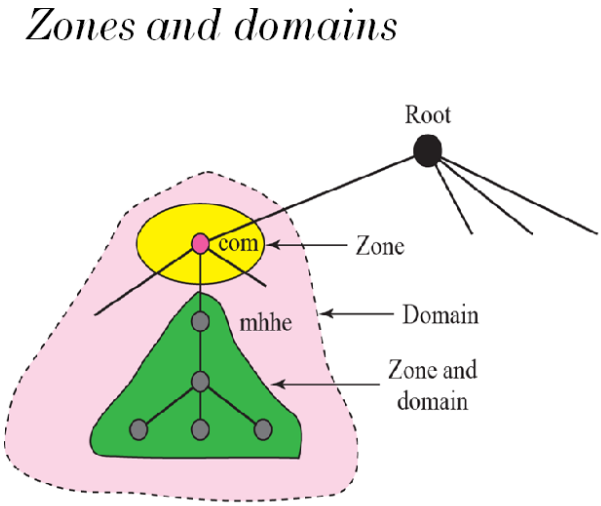
- Name Space:
  Specifications for a structured name space and data associated with the names
- Resolvers:
  Client programs that extract information from Name Servers.
- Name Servers:

Server programs which hold information about the structure and the names.

name space

- To be unambiguous, the names assigned to machines must be carefully selected from a name space with complete control over the binding between the names and IP addresses. In other words, the names must be unique because the addresses are unique.
- A name space that maps each address to a unique name can be organized in two ways:
  flat or hierarchical.
  - A name space can be either
  - flat (names are not divisible into components)
  - hierarchical (Unix file names are an obvious example).



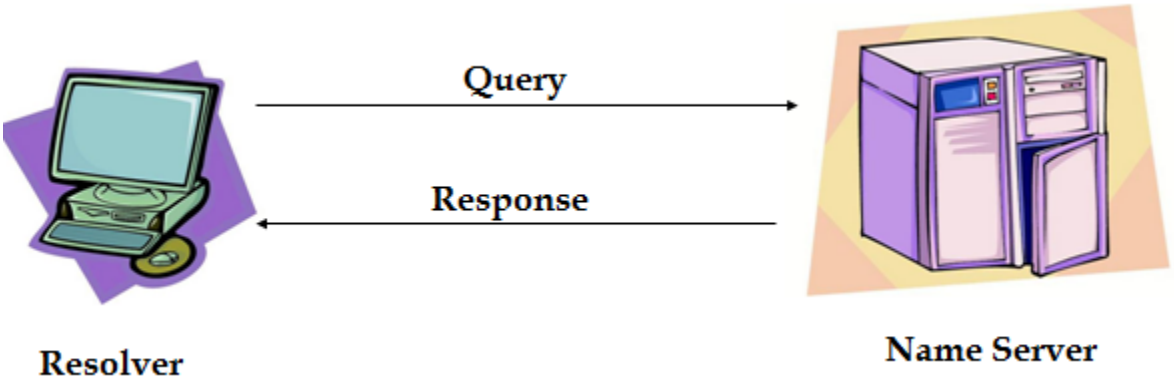


- *The naming system* maintains a collection of *bindings of names to values. The value can be anything we* want the naming system to return when presented with a name; in many cases it is an address.
- a *resolution mechanism is a procedure that, when invoked with a* name, returns the corresponding value. A *name server is a specific implementation of a* resolution mechanism that is available on a network and that can be queried by sending it a message.

Resolvers

A Resolver maps a name to an address and vice versa.

Resolvers ask the questions to the DNS system on behalf of the application.



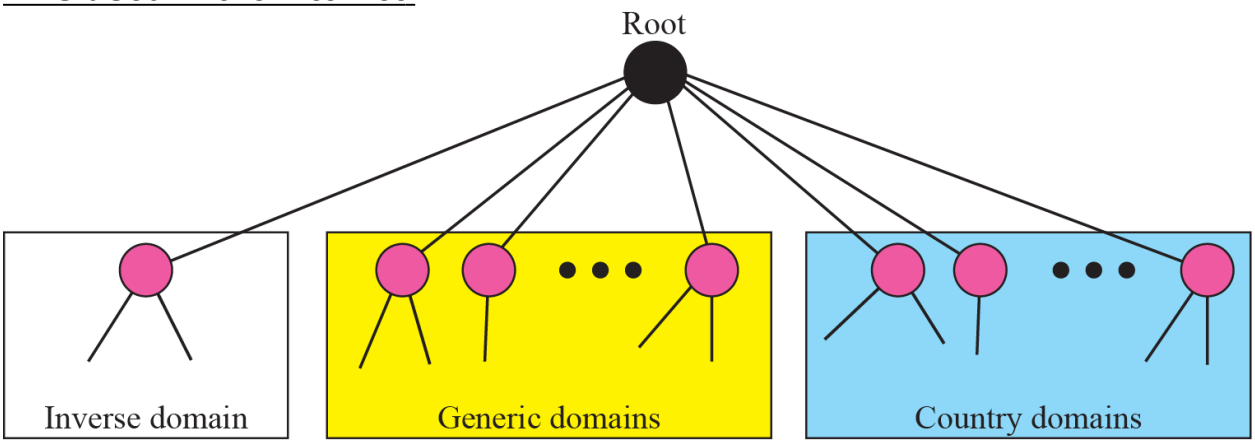
DNS In The Internet

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and the inverse domain

Three domains are,

1. Generic Domains
2. Country Domains
3. Inverse Domain

DNS used in the Internet



Generic domains

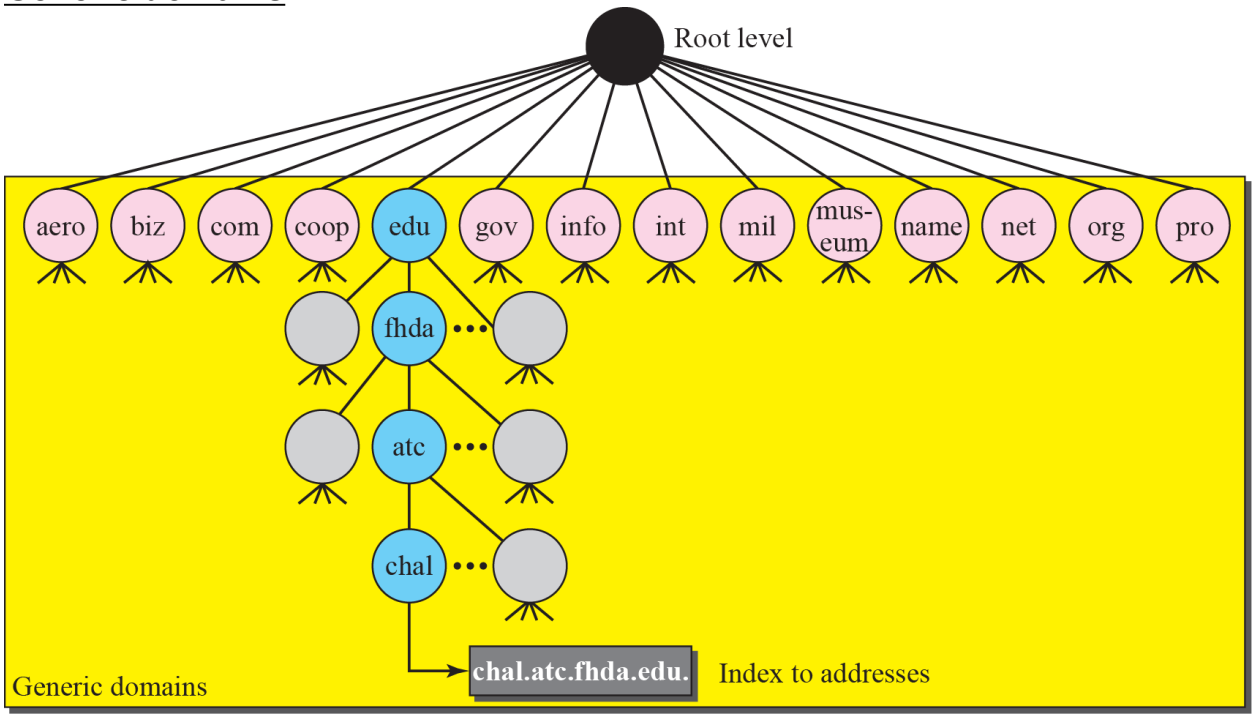
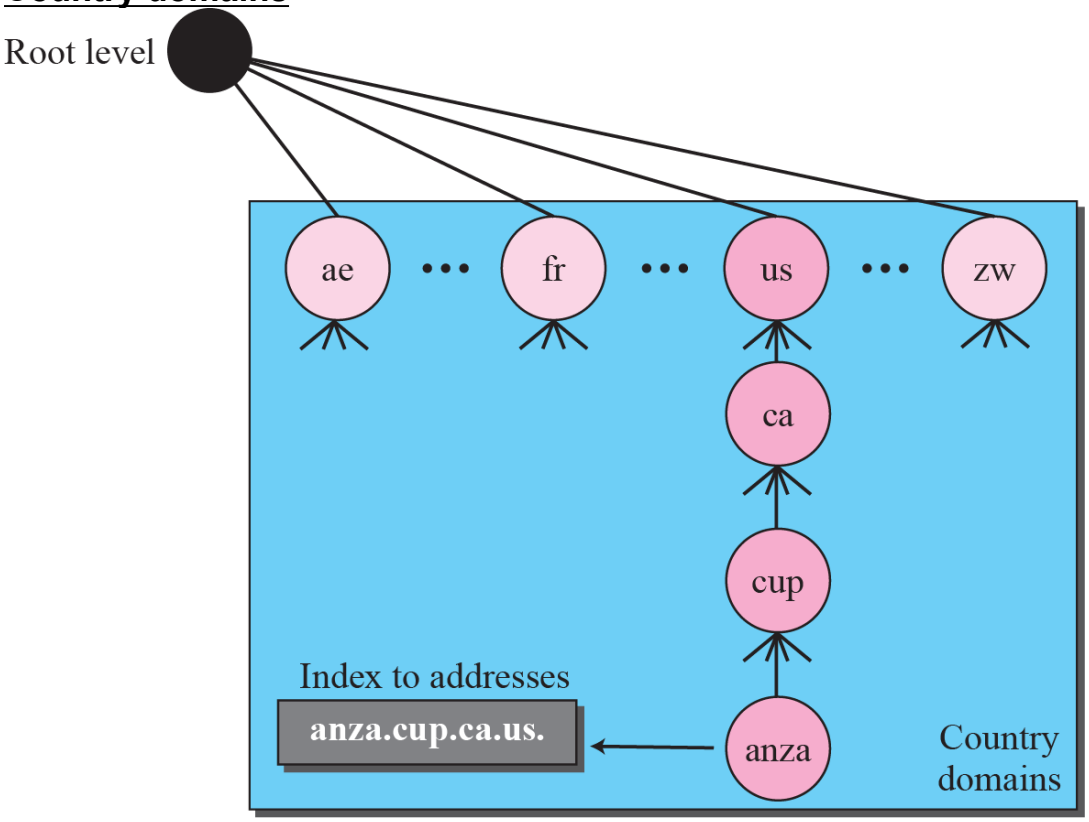


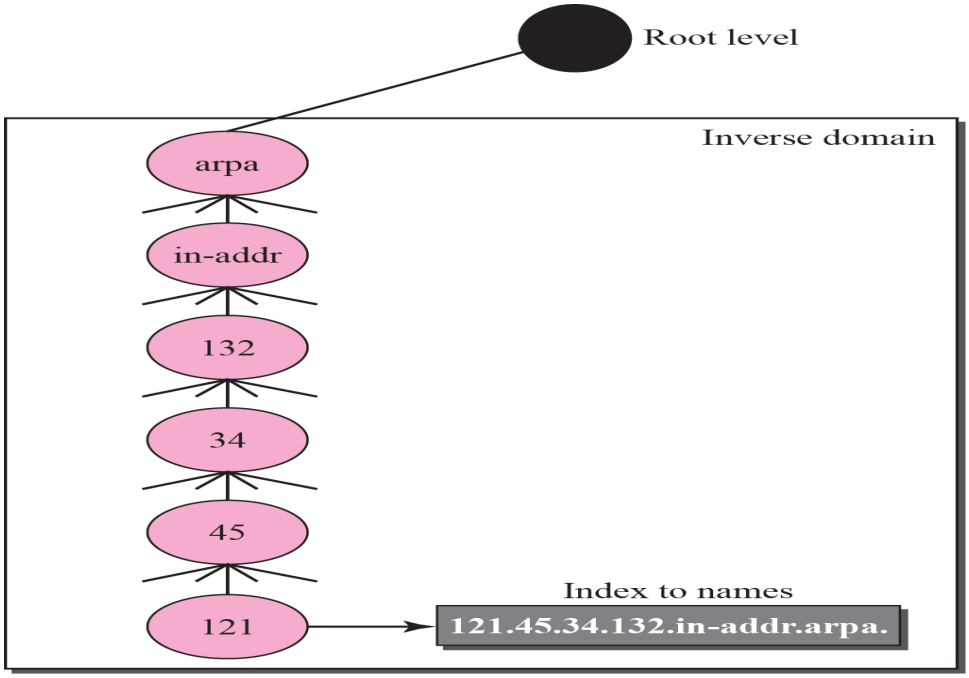
Table 19.1 *Generic domain labels*

Label	*Description*
aero	Airlines and aerospace companies
biz	Businesses or firms (similar to "com")
com	Commercial organizations
coop	Cooperative business organizations
edu	Educational institutions
gov	Government institutions
info	Information service providers
int	International organizations
mil	Military groups
museum	Museums and other non-profit organizations
name	Personal names (individuals)
net	Network support centers
org	Nonprofit organizations
pro	Professional individual organizations

Country domains



Inverse domain



RESOLUTION

Mapping a name to an address or an address to a name is called name-address resolution.

DNS MESSAGES

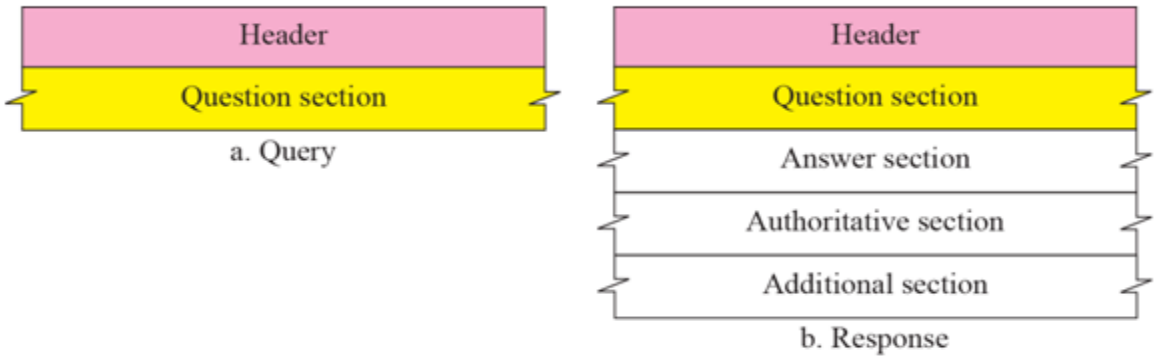
DNS has two types of messages:

1. Query messages
2. Response messages

Query messages can be of two formats

a. Recursive
b. Iterative

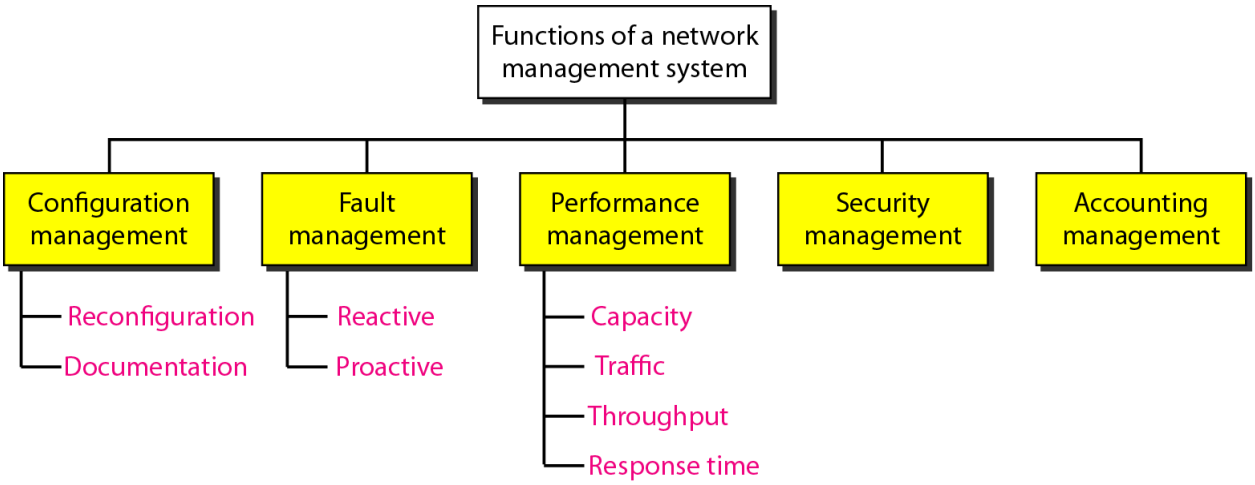
Query and response messages



SNMP (SIMPLE NETWORK MANAGEMANT PROTOCOL)

- *The Simple Network Management Protocol (SNMP) is a framework for managing devices in an internet using the TCP/IP protocol suite. It provides a set of fundamental operations for monitoring and maintaining an internet.*

- A framework for managing devices in an internet using TCP/IP.
- Provides a set of fundamental operations for monitoring and maintaining an internet.
- An application level allows it to monitor devices from various manufactures.



Agent

- A router or a host that runs the SNMP server program
- Keeps performance information in a database
- Can send a trap to the manager if something usual occurs.

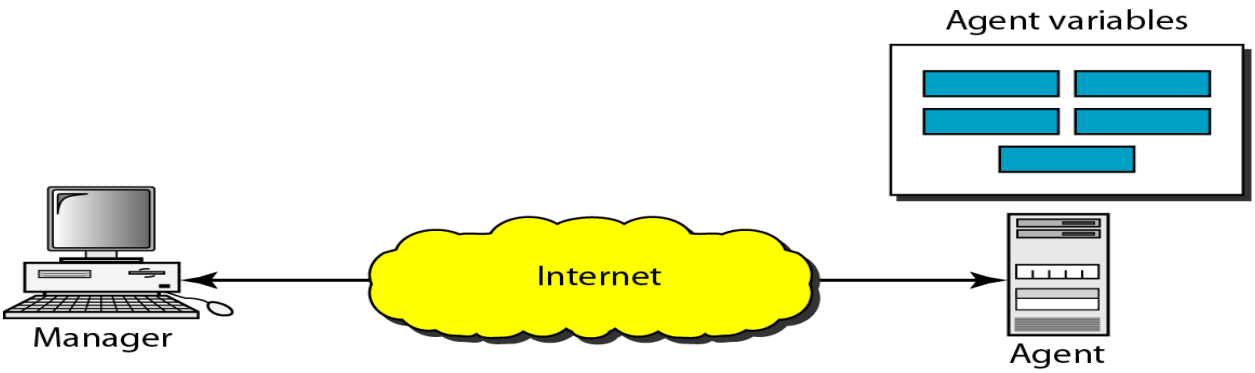
Manager

- A host that runs the SNMP client program
- Has access to the values in the database

Management Components

- SNMP uses two other protocols
  - SMI - Structure of Management Information
  - MIB - Management Information Base

SNMP concept



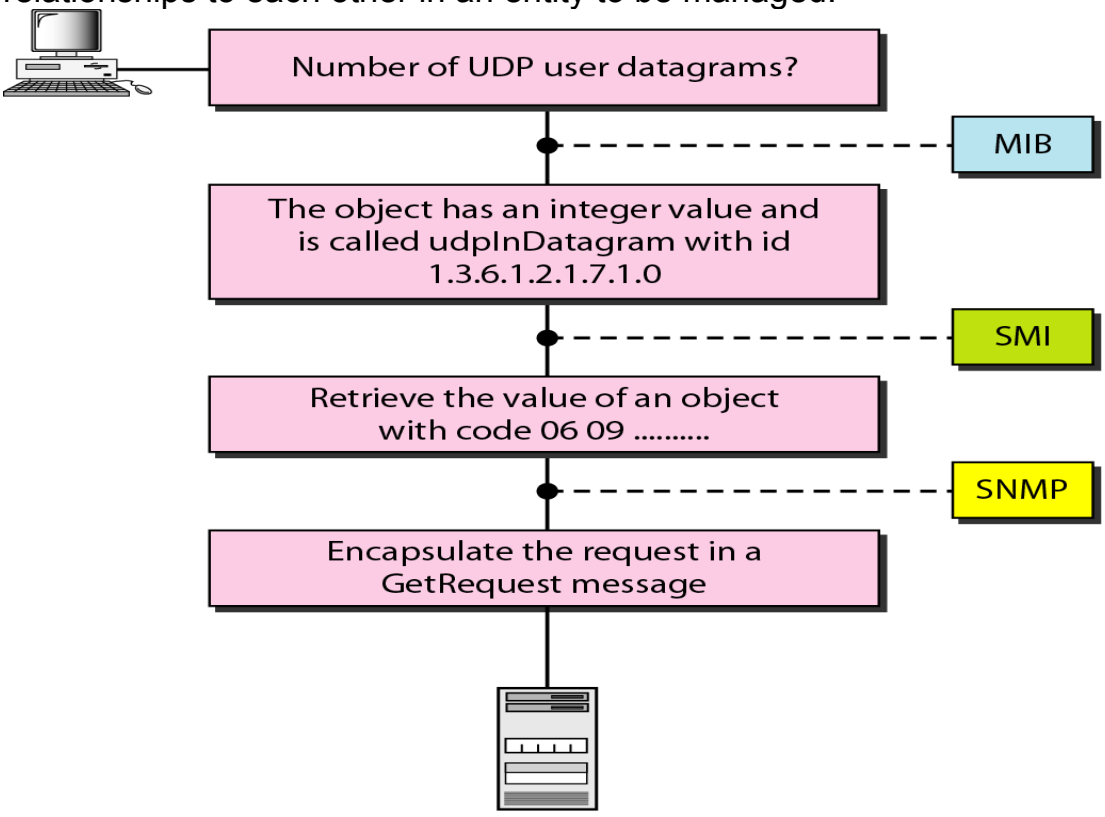
Components of network management on the Internet

1. SMI
2. MIB

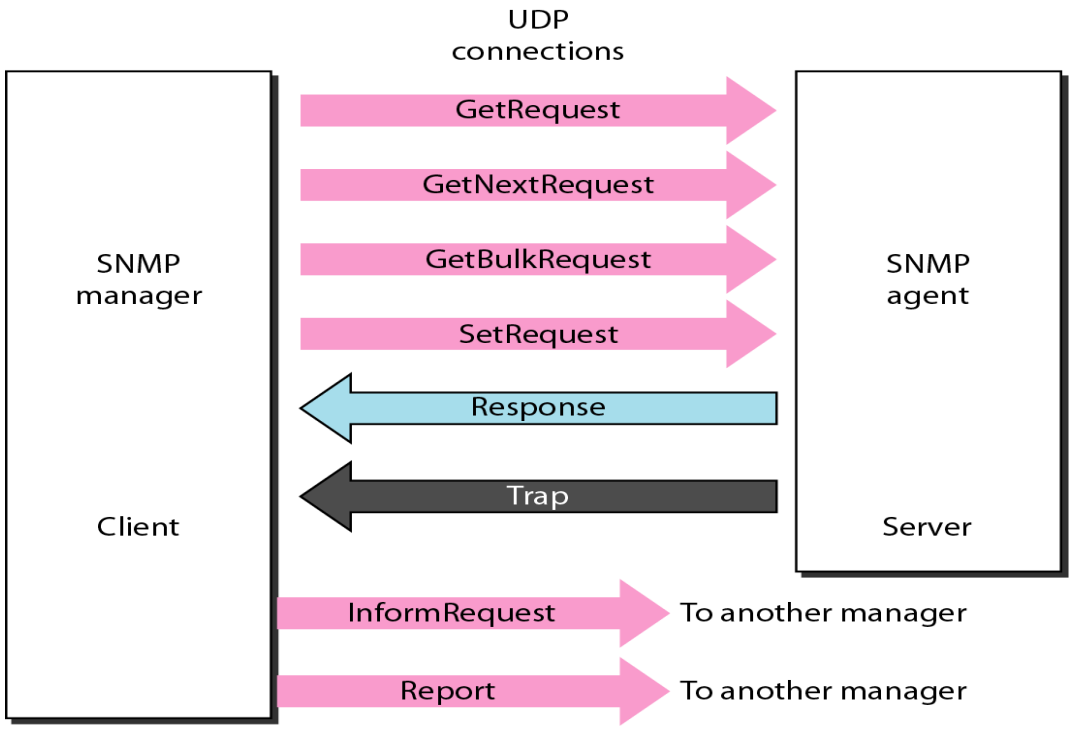
SNMP defines the format of packets exchanged between a manager and an agent. It reads and changes the status (values) of objects (variables) in SNMP packets.

SMI defines the general rules for naming objects, defining object types (including range and length), and showing how to encode objects and values. SMI does not define the number of objects an entity should manage or name the objects to be managed or define the association between the objects and their values.

MIB creates a collection of named objects, their types, and their relationships to each other in an entity to be managed.



SNMP PDUs


UDP
connections
SNMP
manager
SNMP
agent
GetRequest
GetNextRequest
GetBulkRequest
SetRequest
Response
Trap
Client
Server
InformRequest
To another manager
Report
To another manager