Computer Network

- A computer network is a system in which multiple computers are connected to each other to share information and resources.
- The physical connection between networked computing devices is established using either cable media or wireless media.
- The best-known computer network is the Internet.

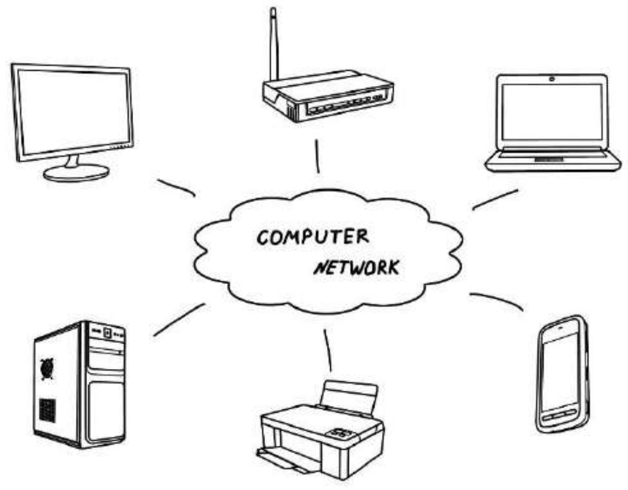


Figure 1: Computer Network

Advantages of Computer Networks

- **File sharing**
  The major advantage of a computer network is that allows file sharing and remote file access. A person sitting at one workstation that is connected to a network can easily see files present on another workstation, provided he is authorized to do so.
- **Resource sharing**
  All computers in the network can share resources such as printers, fax machines, modems, and scanners.
- **Better connectivity and communications**
  It allows users to connect and communicate with each other easily. Various communication applications included e-mail and groupware are used. Through e-mail, members of a network can send a message and ensure safe delivery of data to other members, even in their absence.
- **Internet access**
  Computer networks provide internet service over the entire network. Every single computer attached to the network can experience the high-speed internet.
- **Entertainment**
  Many games and other means of entertainment are easily available on the internet. Furthermore, Local Area Networks (LANs) offers and facilitates other ways of enjoyments, such as many players are connected through LAN and play a particular game with each other from a remote location.

- **Inexpensive system**
  Shared resources mean reduction in hardware costs. Shared files mean reduction in memory requirement, which indirectly means a reduction in file storage expenses. A particular software can be installed only once on the server and made available across all connected computers at once. This saves the expense of buying and installing the same software as many times for as many users.
- **Flexible access**
  A user can log on to a computer anywhere on the network and access his files. This offers flexibility to the user as to where he should be during the course of his routine.
- **Instant and multiple access**
  Computer networks are multiple processes. Many users can access the same information at the same time. Immediate commands such as printing commands can be made with the help of computer networks.

Disadvantages of Computer Networks

- **Lack of data security and privacy**
  Because there would be a huge number of people who would be using a computer network to get and share some of their files and resources, a certain user's security would be always at risk. There might even be illegal activities that would occur, which you need to be careful about and aware of.
- **Presence of computer viruses and malware**
  If even one computer on a network gets affected by a virus, there is a possible threat for the other systems getting affected too. Viruses can spread on a network easily, because of the inter-connectivity of workstations. Moreover, multiple systems with common resources are the perfect breeding ground for viruses that multiply.
- **Lack of Independence**
  Since most networks have a centralized server and dependent clients, the client users lack any freedom whatsoever. Centralized decision making can sometimes hinder how a client user wants to use his own computer.
- **Lack of Robustness**
  As previously stated, if a computer network's main server breaks down, the entire system would become useless. Also, if it has a bridging device or a central linking server that fails, the entire network would also come to a standstill.
- **Need an efficient handler**
  For a computer network to work efficiently and optimally, it requires high technical skills and know-how of its operations and administration. A person just having basic skills cannot do this job. Take note that the responsibility to handle such a system is high, as allotting permissions and passwords can be daunting. Similarly, network configuration and connection is very tedious and cannot be done by an average technician who does not have advanced knowledge.

Use (Applications) of Computer Networks

- **Financial services**
  Nowadays, almost all the financial services depend on the computer network. You can access the financial services across the world. For example, a user can transfer money from one place to another by using the electronic fund transfer feature. You can use networking in various financial areas such as ATM, foreign exchange and credit history search.
- **Business**

Nowadays, most of the works of businesses are done over the computers. To exchange the data and ideas, you need effective data and resources sharing features. To do this, you need to connect the computer with each other through a network. For example, a person of one department of an organization can share or access the electronic data of other departments through a network.

- **Email services**
  A computer network provides you the facility to send or receive emails across the globe in few seconds.
- **Mobile applications**
  By using mobile applications, such as cellular or wireless phones, you can communicate (exchange your views and ideas) with one other.
- **Directory services**
  It provides you the facility to store files on a centralized location to increase the speed of search operation worldwide.
- **Teleconferencing**
  It contains voice conferencing and video conferencing which are based on networking. In teleconferencing, the participants need not be presented at the same location.

Types of Computer Networks

LAN (Local Area Network)

- It is privately-owned networks within a single building or campus of up to a few kilometers in size.
- They are widely used to connect personal computers and workstations in company offices and factories to share resources (e.g., printers) and exchange information.
- LANs are easy to design and troubleshoot
- In LAN, all the machines are connected to a single cable.
- Different types of topologies such as Bus, Ring, Star, and Tree are used.
- The data transfer rates for LAN is up to 10 Gbits/s.
- They transfer data at high speeds. The high transmission rate is possible in LAN because of the short distance between various computer networks.
- They exist in a limited geographical area.
- **Advantages**
  - LAN transfers data at high speed.
  - LAN technology is generally less expensive.

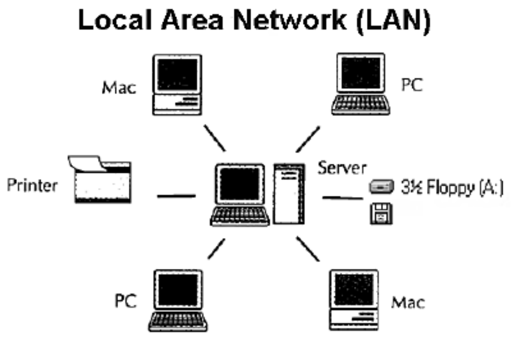


Figure 2: Local Area Network

MAN (Metropolitan Area Network)

- MAN is a larger version of LAN which covers an area that is larger than the covered by LAN but smaller than the area covered by WAN.
- A metropolitan area network or MAN covers a city. The best-known example of a MAN is the cable television network available in many cities.
- MAN connects two or more LANs.
- At first, the companies began jumping into the business, getting contracts from city governments to wire up an entire city.
- The next step was television programming and even entire channels designed for cable only.

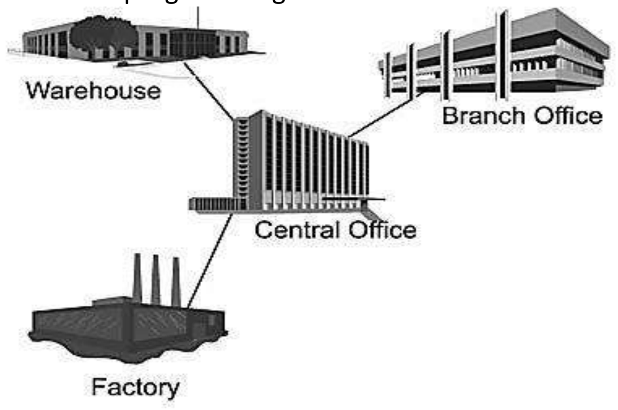


Figure 3: Metropolitan Area Network

WAN (Wide Area Network)

- WAN spans a large geographical area, often a country or region.
- WAN links different metropolitan's countries and national boundaries thereby enabling easy communication.
- It may be located entirely within a state or a country or it may be interconnected around the world.
- It contains a collection of machines intended for running user (i.e., application) programs. We will follow traditional usage and call these machines hosts.
- The communication between different users of WAN is established using leased telephone lines or satellite links and similar channels.

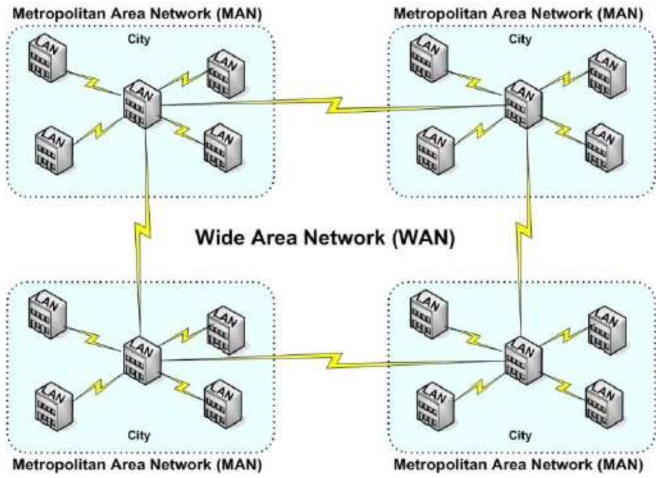

Metropolitan Area Network (MAN)
City
LAN
Metropolitan Area Network (MAN)
City
Wide Area Network (WAN)
City
Metropolitan Area Network (MAN)
City
Metropolitan Area Network (MAN)

Figure 4: Wide Area Network

Difference between LAN, MAN, and WAN.

Parameter	**LAN**	**MAN**	**WAN**
Area covered	Covers a small area. i.e. within building	Covers larger than LAN & smaller than WAN	Covers large area
Error rates	Lowest	Moderate	Highest
Transmission speed	High speed	Moderate speed	Low speed
Equipment cost	Inexpensive	Moderate-expensive	Most expensive
Design & maintenance	Easy	Moderate	Difficult

Internet

- The internet is a type of world-wide computer network.
- The internet is the collection of infinite numbers of connected computers that are spread across the world.
- We can also say that the Internet is a computer network that interconnects hundreds of millions of computing devices throughout the world.
- It is established as the largest network and sometimes called a network of a network that consists of numerous academic, business and government networks, which together carry various information.
- The Internet is a global computer network providing a variety of information and communication facilities, consisting of interconnected networks using standardized communication protocols.
- When two computers are connected over the Internet, they can send and receive all kinds of information such as text, graphics, voice, video, and computer programs.

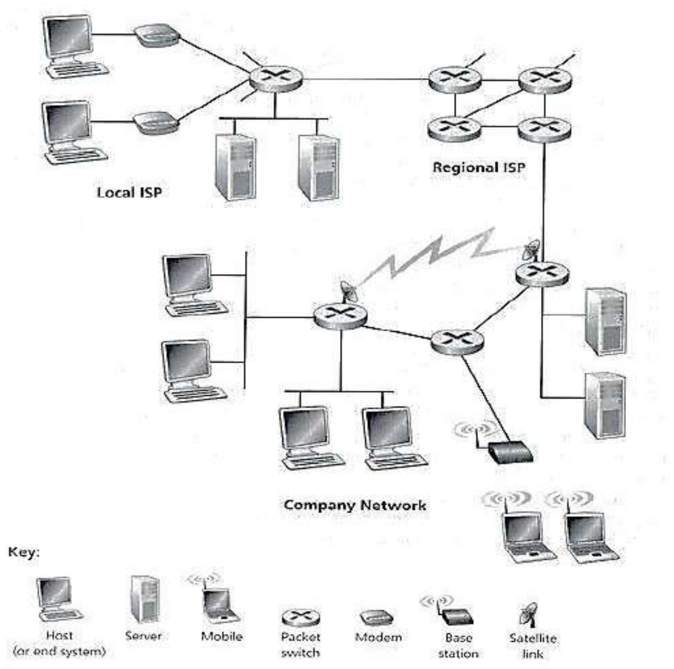


Figure 5: Some pieces of the Internet

Protocol

- A protocol is a set of rules that govern (manages) data communications.
- Protocols define methods of communication, how to communicate when to communicate etc.
- A protocol is an agreement between the communicating parties on how communication is to proceed.
- Important elements of protocols are
  1. Syntax 2. Semantics 3. Timing
- **Syntax**:- Syntax means format of data or the structure how it is presented e.g. first eight bits are for sender address, next eight bits are for receiver address and rest of the bits for message data.
- **Semantics**:- Semantics is the meaning of each section of bits e.g. the address bit means the route of transmission or final destination of a message.
- **Timing**:- Timing means, at what time data can be sent and how fast data can be sent.
- Some protocols also support message acknowledgment and data compression designed for reliable and/or high-performance network communication.
- Example: HTTP, IP, FTP etc…

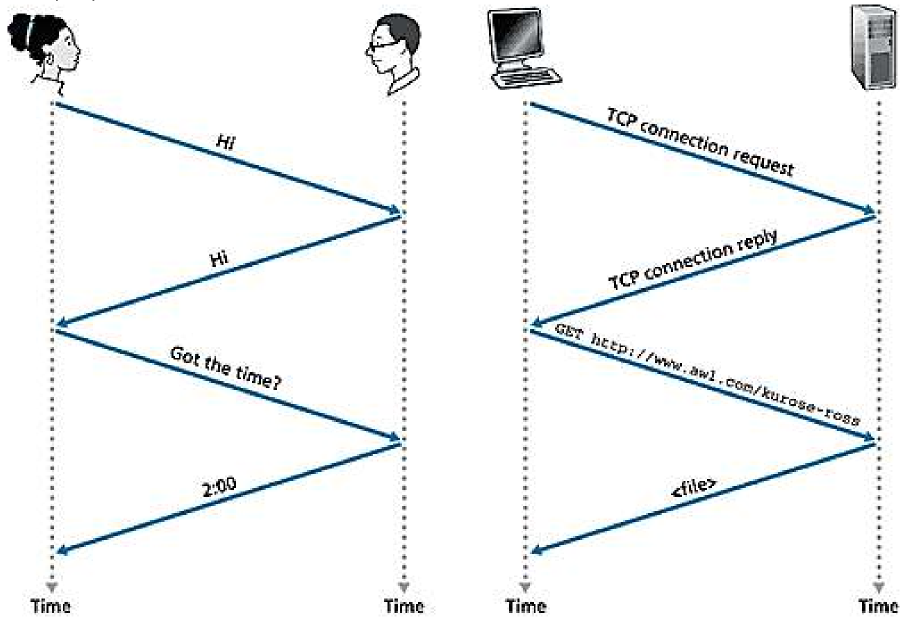


Figure 6: A human protocol and a computer network protocol

The Network Edge

- It defines those computers of the network used at the edge (end) of the network. These computers are known as hosts or end system.
- A host can be classified into the following two types:
  - **Clients**: Refer to the computer systems that request servers for the completion of a task. The clients are generally called desktop PCs or workstations.
  - **Servers**: Refer to the computer systems that receive requests from the clients and process them. After the processing is complete, the servers send a reply to the clients who sent the request.

- The concept of clients and servers is essential in the network design. The various networks design models are as follows:
    1. Peer to Peer network
    2. Client-Server network

Peer to Peer network

- In this network group of computers is connected together so that users can share resources and information.
- There is no central location (server) for authenticating users, storing files, or accessing resources and each of them works as both client and server.
- This means that users must remember which computers in the workgroup have the shared resource or information that they want to access.
- **Advantage**:
    - ➢ It is easy to set up.
    - ➢ There is no need for any committed server as each peer acts as both server and client.
    - ➢ The network implementation is quite cheap.
    - ➢ The resources of a peer can be shared with other peers very easily in the network.
- **Disadvantage:**
    - ➢ The speed of the network decreases due to heavy usage.
    - ➢ It is not easy to keep track of information on each computer.
    - ➢ There is no central backup of files and folders.
    - ➢ Network and data security are weak.

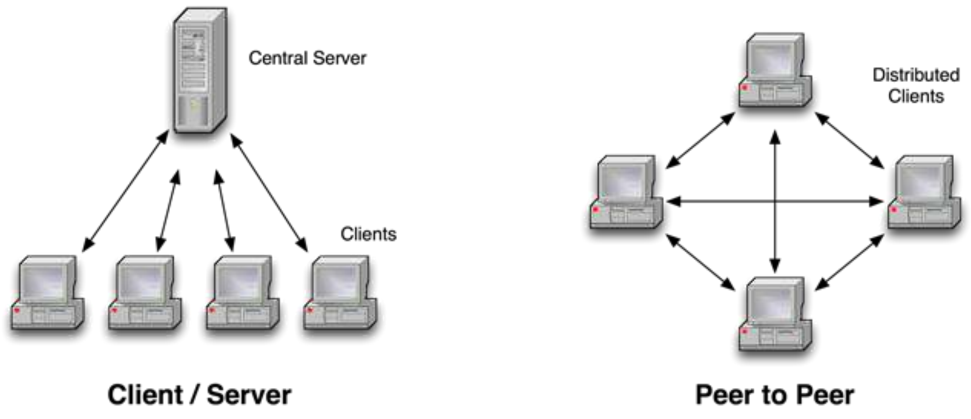


Figure 7: Network Edge - Client/Server Network and Peer to Peer

Client/Server network

- A client/server network is a system where one or more computers called clients to connect to a central computer named as a server to share or use resources.
    - The client requests a service from a server, which may include running an application, querying a database, printing a document, performing a backup or recovery procedure. The request made by the client is handled by a server.
    - A client/server network is that in which the files and resources are centralized. This means that the server can hold them and other computers (Client) can access them.

    - **Advantage**:
        - ➢ The server system holds the shared files.
        - ➢ The server system can be scheduled to take the file backups automatically.

- ➢ Network access is provided only to authorized users through user security at the server.
- ➢ The server system is a kind of central repository for sharing a printer with clients.
- ➢ Internet access, e-mail routing, and such other networking tasks are quite easily managed by the server.
- ➢ The software applications shared by the server are accessible to the clients.

- **Disadvantage:**
  - ➢ The implementation of the network is quite expensive.
  - ➢ An NOS (Network Operating System) is essential.
  - ➢ If a server fails, the entire network crashes.
  - ➢ There may be congestion if more than one client requests for a service at the same time.

Techniques used in data communications to transfer data

1. Connection-oriented method
2. Connectionless method

Connection-oriented method

- Connection-oriented communication includes the steps of setting up a call from one computer to another, transmitting/receiving data, and then releasing the call, just like a voice phone call.
- However, the network connecting the computers is a packet switched network, unlike the phone system's circuit switched network.
- Connection-oriented communication is done in one of two ways over a packet switched network:
  1. Without virtual circuits
  2. With virtual circuits.

Without virtual circuits:

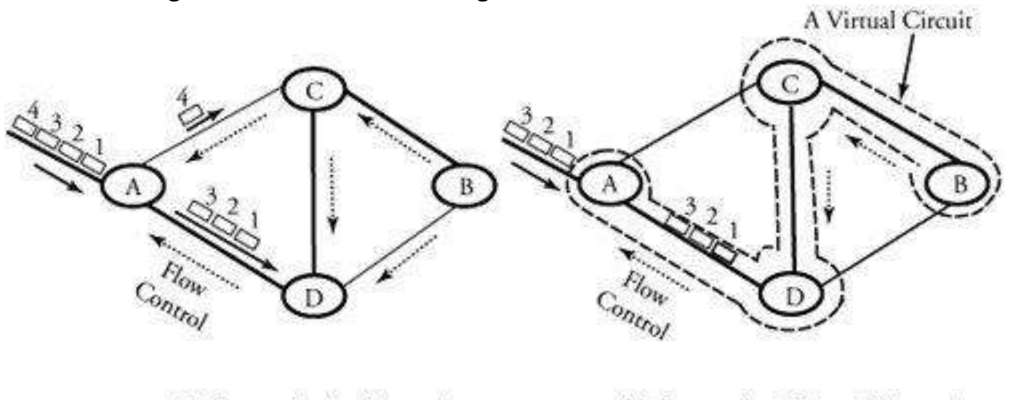
- This is what TCP does on the Internet.
- The only two machines on the Internet are aware of the connection which is established between the two computers at the endpoints.
- The Internet itself, its routers and links have no information about the presence of a connection between the two computers.
- This means that all of the packets flowing between the two computers can follow different routes.
- One benefit of establishing the connection is that the flow of packets from the source to the destination can be slowed down if the Internet is congested and speeded up when congestion disappears.
- Another benefit is that the endpoints can anticipate traffic between them, and agree to cooperate to ensure the integrity and continuity of the data transfers. This allows the network to be treated as a "stream" of data.

With virtual circuit:

- This is not used on the Internet, but is used in other types of networks (eg. the "X.25" protocol, still popular in Europe).
- The routers within the network route all packets in one connection over the same route. The advantage is that video and voice traffic is easier to carry because routers can reserve memory space to buffer the transmission.

Connectionless method

- Connectionless communication is just packet switching where no call establishment and release occur.
- A message is broken into packets, and each packet is transferred separately. Moreover, the packets can travel a different route to the destination since there is no connection.
- Connectionless service is typically provided by the UDP (User Datagram Protocol). The packets transferred using UDP are also called datagrams.



(a) Connectionless Network (b) Connection-Oriented Network

Feature	Connectionless	Connection-oriented
How is data sent?	One packet at a time	Continuous stream of packets
Do packets follow the same route?	No	Virtual circuit: yes Without virtual circuit: no
Are resources reserved in the network?	No	Virtual circuit: yes Without virtual circuit: no
Are resources reserved in communicating hosts?	No	Yes
Is connection establishment done?	No	Yes
Is state information stored at network nodes?	No	Virtual circuit: yes Without virtual circuit: no
What is the impact of node/switch crash?	Only packets at a node are lost	All virtual circuits through node fail
What addressing information is needed on each packet?	Full source and destination address	Virtual circuit: virtual circuit number Without virtual circuit: full source and destination address

Transmission Media

- A transmission media can be defined as anything that can carry information from a source to a destination.
- On the basis of transmission of data, the transmission media can be classified into two categories:
    1. Guided (Physical) transmission media
    2. Unguided (Wireless) transmission media

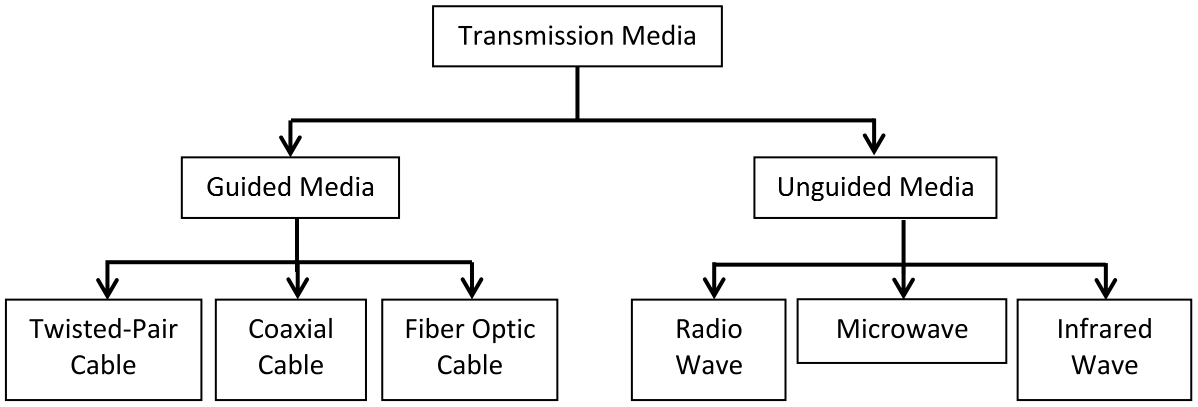


Figure 8: Classification Transmission Media

Guided Transmission Media

- Guided media are those that provide a channel from one device to another.
- The three Guided (Physical) media commonly used for data transmission are:
    1. Twisted-Pair 2. Coaxial 3. Fiber Optics

1. Twisted Pair

- A twisted pair consists of two insulated copper wires, typically about 1 mm thick.
- The wires are twisted together in a helical form, just like a DNA molecule.
- Twisting is done because two parallel wires constitute a fine antenna.
- When the wires are twisted, the waves from different twists cancel out, so the wire radiates less effectively.

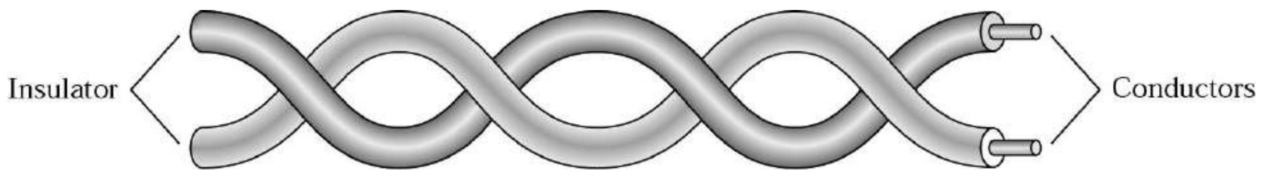


Figure 9: Twisted Pair Cable

Why cable is twisted?

- If the two wires are parallel, the effect of these unwanted signals is not the same in both wires because they are at different locations relatives to the noise or crosstalk sources.
- This results in a difference at the receiver.
- By twisting the pair, a balance is maintained.

Types of Twisted-Pair Cable

1) **Unshielded twisted-pair (UTP)**

- Twisted pair cabling comes in several varieties, two of which are important for computer networks.
- **Category 3** twisted pairs consist of two insulated wires gently twisted together.
- Most office buildings had one category 3 cable running from a central wiring closet on each floor into each office.
- **Category 5** is the more advanced twisted pairs were introduced.
- They are similar to category 3 pairs, but with more twists per centimeter, which results in less crosstalk and a better-quality signal over longer distances, making them more suitable for high-speed computer communication.
- Up-and-coming categories are 6 and 7, which are capable of handling signals with bandwidths of 250 MHz and 600 MHz, respectively (versus a mere 16 MHz and 100 MHz for categories 3 and 5 respectively).

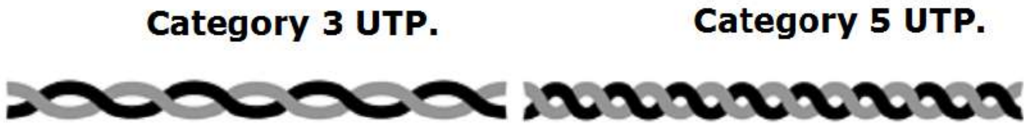


Figure 10: Unshielded twisted-pair

2) **Shielded twisted-pair (STP).**

- STP cable has a metal foil or braided mesh covering that encases each pair of insulated conductors.
- Metal casing improves the quality of cable by preventing the penetration of noise or crosstalk.
- It is bulkier and more expensive.
- **Applications:**
  - ➢ Used in telephone lines to provide voice and data channels.
  - ➢ The DSL lines use by telephone companies use the high-bandwidth capability of UTP cables.
  - ➢ LANs, such as 10Base-T, 100Base-T also uses twisted-pair cables.

2. **Coaxial Cable**

- It has better shielding than twisted pairs, so it can span longer distances at higher speeds.
- Two kinds of the coaxial cable are widely used. One kind is a 50-ohm cable which is commonly used when it is intended for digital transmission from the start.
- The other kind is a 75-ohm cable which is commonly used for analog transmission and cable television but is becoming more important with the advent of the Internet over cable.
- A coaxial cable consists of stiff copper wire as the core surrounded by an insulating material.
- The insulator is encased by a cylindrical conductor, often as a closely-woven braided mesh.
- The outer conductor is covered in a protective plastic sheath.
- The construction and shielding of the coaxial cable give it a good combination of high bandwidth and excellent noise immunity.
- The bandwidth possible depends on the cable quality, length, and signal-to-noise ratio of the data signal. Modern cables have a bandwidth of close to 1 GHz.

- Coaxial cables used is widely used within the telephone system for long-distance lines but have now largely been replaced by fiber optics on long-haul routes.

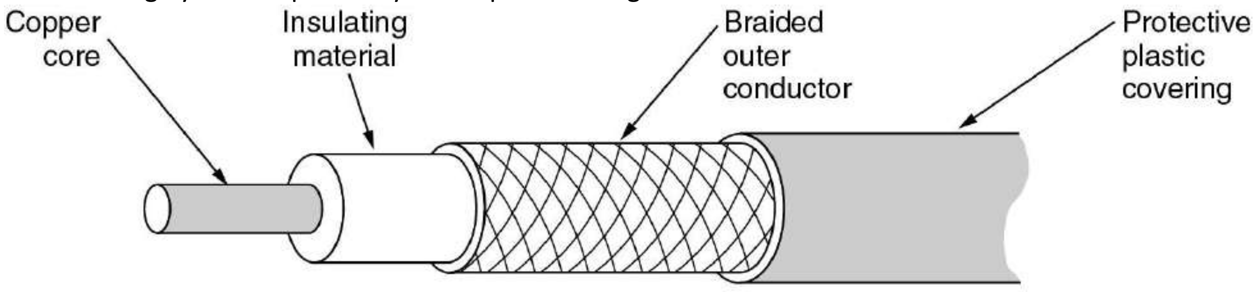


Figure 11: Coaxial Cable

3. Fiber Optics

- A fiber-optic cable is made of glass or plastic and transmits signals in the form of light.
- Optical fibers use reflection to guide light through a channel.
- A glass or plastic core is surrounded by a cladding of less dense glass or plastic.
- The difference in density of the two materials must be such that a beam of light moving through a core is reflected off the cladding instead of being refracted into it.

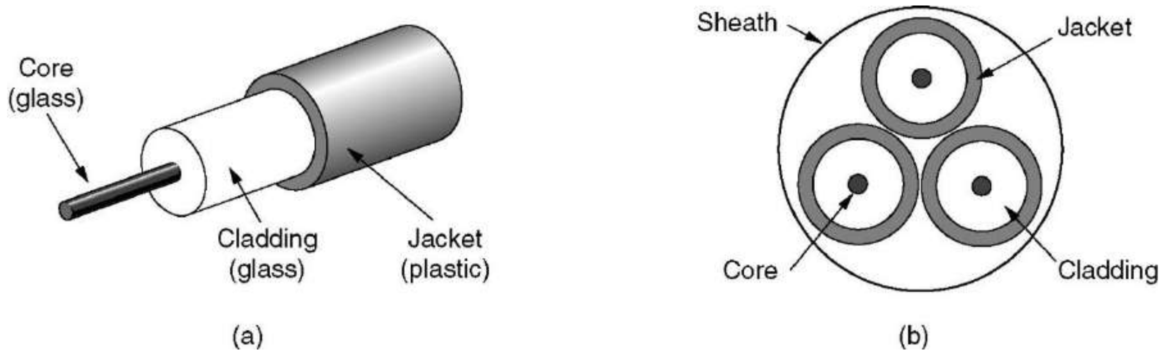


Figure 12: Fiber Optic Cable

- Fiber optic cables are similar to coax, except without the braid.
- The figure shows a single fiber viewed from the side. At the center is the glass core through which the light propagates.
- The core is surrounded by a glass cladding with a lower index of refraction than the core, to keep all the light in the core.
- Next comes a thin plastic jacket to protect the cladding. Fibers are typically grouped in bundles, protected by an outer sheath. The figure shows a sheath with three fibers.

Unguided (Wireless) transmission media

- Unguided media transport electromagnetic waves without using a physical conductor. This type of communication is often referred to as wireless communication.
  1. Radio Transmission
  2. Microwave Transmission
  3. Infrared
  4. Lightwave Transmission

1. **Radio Transmission**
    - Radio waves are easy to generate, can travel long distances, and can penetrate buildings easily, so they are widely used for communication, both indoors and outdoors.
    - Radio waves also are omnidirectional, meaning that they travel in all directions from the source, so the transmitter and receiver do not have to be carefully aligned physically.
    - The properties of radio waves are frequency dependent.
    - At low frequencies, radio waves pass through obstacles well, but the power falls off sharply with distance from the source, roughly as $1/r^2$ in the air.
    - At high frequencies, radio waves tend to travel in straight lines and bounce off obstacles. They are also absorbed by rain.
    - At all frequencies, radio waves are subject to interference from motors and other electrical equipment.

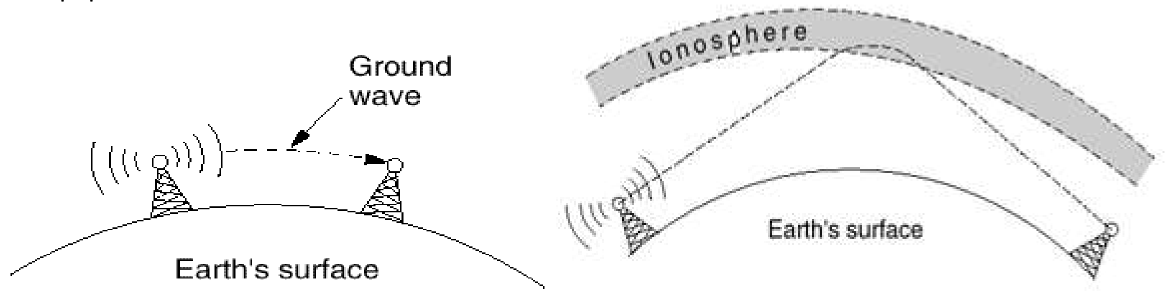


Figure 13: Ground wave

- In the VLF, LF, and MF bands, radio waves follow the curvature of the earth.
- In the HF they bounce off the ionosphere.

2. **Microwave Transmission**
    - Since the microwaves travel in a straight line, if the towers are too far apart, the earth will get in the way. Consequently, repeaters are needed periodically.
    - Unlike radio waves at lower frequencies, microwaves do not pass through buildings well. In addition, even though the beam may be well focused at the transmitter, there is still some divergence in space.
    - Above 100 MHz, the waves **travel in straight lines** and can, therefore, be narrowly focused. Concentrating all the energy into a small beam using a **parabolic antenna** gives a much higher signal to noise ratio.
    - **Advantages:**
        - No right way is needed (compared to wired media).
        - Relatively inexpensive.
        - Simple to install.
    - **Disadvantages:**
        - Do not pass through buildings well.
        - Multipath fading problem (the delayed waves cancel the signal).
        - Absorption by rain above 8 GHz.
        - A severe shortage of spectrum.
3. **Infrared**
    - Unguided infrared and millimetre waves are widely used for short-range communication.
    - The remote controls used on televisions, VCRs, and stereos all use infrared communication.

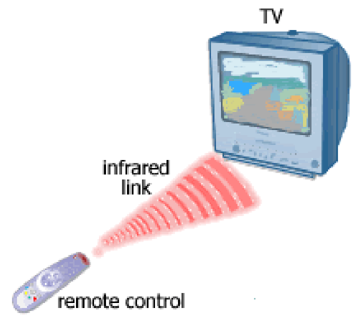


Figure 14: Infrared wave connection

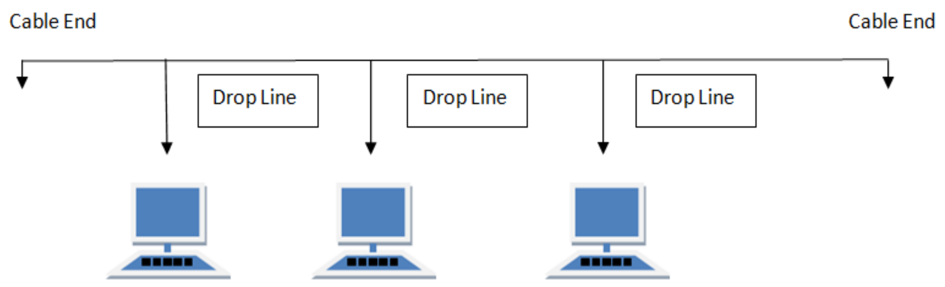
- They are relatively directional, cheap, and easy to build but have a major drawback: they do not pass through solid objects (try standing between your remote control and your television and see if it still works).
- In general, as we go from long-wave radio toward visible light, the waves behave more and more like light and less and less like a radio.
- On the other hand, the fact that infrared waves do not pass through solid walls well is also a plus.
- It means that an infrared system in one room of a building will not interfere with a similar system in adjacent rooms or buildings.
- Furthermore, security of infrared systems against eavesdropping is better than that of radio systems precisely for this reason.
- Therefore, no government license is needed to operate an infrared system, in contrast to radio systems, which must be licensed outside the ISM bands.

Topologies (Network Topologies)

- Network Topology is the schematic description of a network arrangement, connecting various nodes (sender and receiver) through lines of connection.
- A Network Topology is the arrangement with which computer systems or network devices are connected to each other.
- Types of network topologies :
    1. Bus
    2. Ring
    3. Star
    4. Mesh
    5. Tree
    6. Hybrid

Bus Topology

- Bus topology is a network type in which every computer and network device is connected to a single cable.



Features:

- It transmits data only in one direction.
- Every device is connected to a single cable.

Advantages:

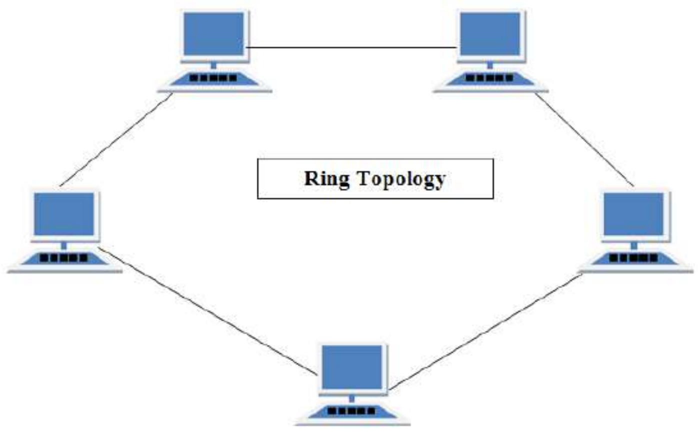
- It is cost effective (cheaper).
- Cable required is least compared to other network topology.
- Used in small networks.
- It is easy to understand.
- Easy to expand joining two cables together.

Disadvantages:

- Cables fail then the whole network fails.
- If network traffic is heavy or nodes are more the performance of the network decreases.
- Cable has a limited length.

Ring Topology

- It is called ring topology because it forms a ring as each computer is connected to another computer, with the last one connected to the first. Exactly two neighbors for each device.



Features:

- A number of repeaters are used and the transmission is unidirectional.
- A date is transferred in a sequential manner that is bitten by bit.

Advantages:

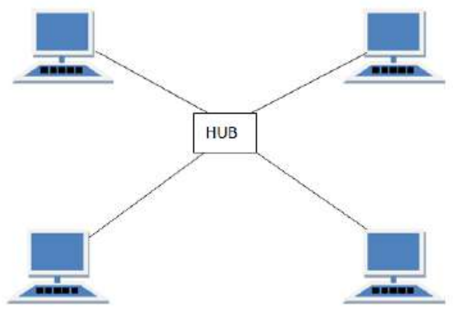
- Transmitting network is not affected by high traffic or by adding more nodes, as only the nodes having tokens can transmit data.
- Cheap to install and expand.

Disadvantages:

- Troubleshooting is difficult in a ring topology.
- Adding or deleting the computers disturbs the network activity.
- Failure of one computer disturbs the whole network.

Star Topology

- In this type of topology, all the computers are connected to a single hub through a cable. This hub is the central node and all others nodes are connected to the central node.



Features:

- Every node has its own dedicated connection to the hub.
- Acts as a repeater for data flow.
- Can be used with twisted pair, Optical Fibre or coaxial cable.

Advantages:

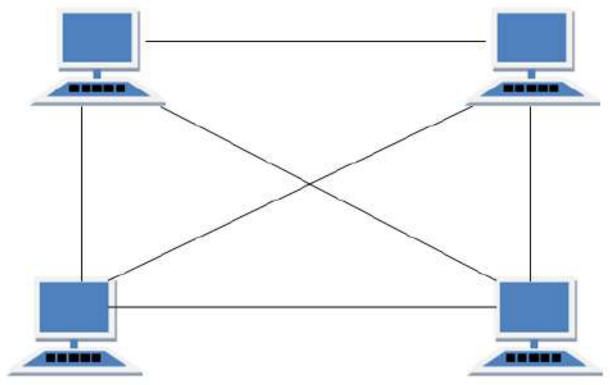
- Fast performance with few nodes and low network traffic.
- Hub can be upgraded easily.
- Easy to troubleshoot.
- Easy to set up and modify.
- Only that node is affected which has failed rest of the nodes can work smoothly.

Disadvantages:

- Cost of installation is high.
- Expensive to use.
- If the hub is affected then the whole network is stopped because all the nodes depend on the hub.
- Performance is based on the.

Mesh Topology

- It is a point-to-point connection to other nodes or devices.
- Traffic is carried only between two devices or nodes to which it is connected.



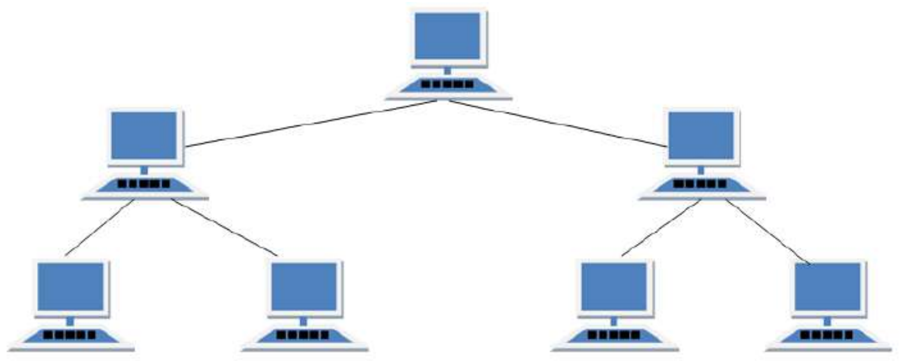
Features:

- Fully connected.
- Robust.
- Not flexible.

Advantages:
- Each connection can carry its own data load.
- It is robust.
- A fault is diagnosed easily.
- Provides security and privacy.

Disadvantages:
- Installation and configuration are difficult.
- Cabling cost is more.
- Bulk wiring is required.

Tree Topology
- It has a root node and all other nodes are connected to it forming a hierarchy.
- It is also called hierarchical topology.
- It should at least have three levels to the hierarchy.



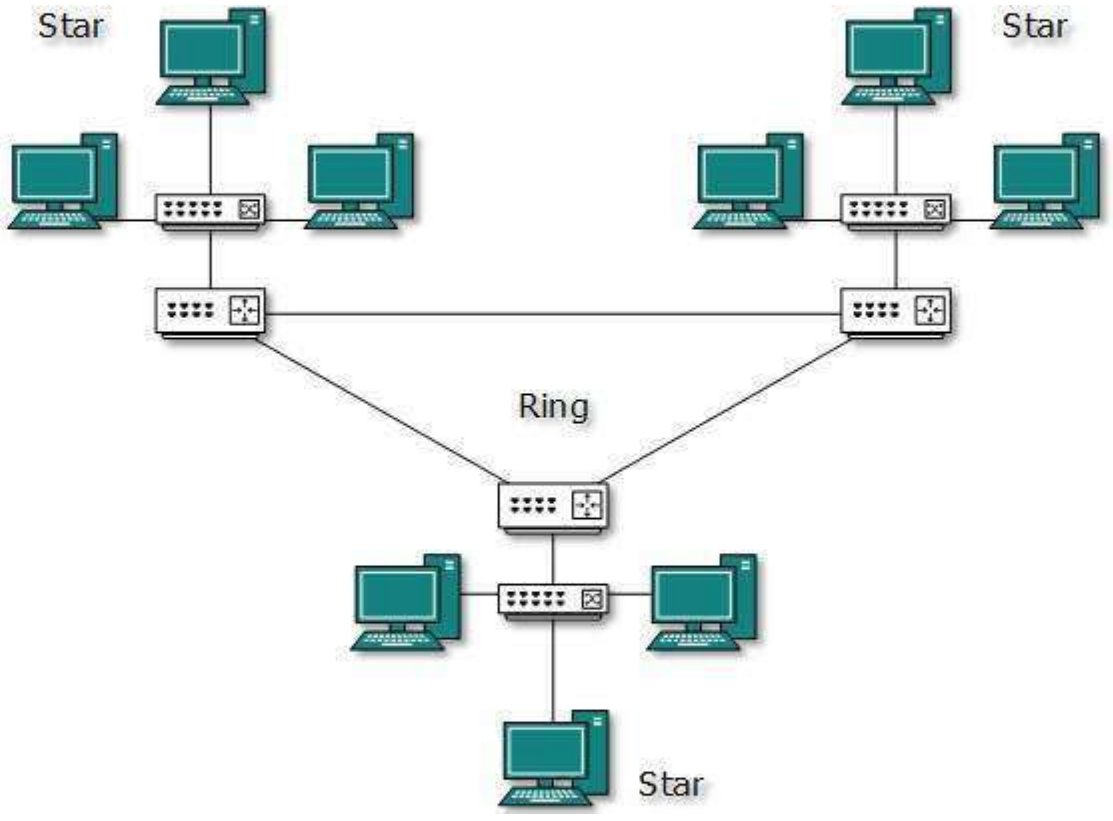
Features:
- Ideal if workstations are located in groups.
- Used in Wide Area Network.

Advantages:
- Extension of bus and star topologies.
- Expansion of nodes is possible and easy.
- Easily managed and maintained.
- Error detection is easily done.

Disadvantages:
- Heavily cabled.
- Costly.
- If more nodes are added maintenance is difficult.
- Central hub fails then network fails.

Hybrid Topology
- A network structure whose design contains more than one topology is said to be hybrid topology.

- For example, if in an office in one department ring topology is used and in another star, topology is used, connecting these topologies will result in Hybrid Topology (ring topology and star topology).



Features:

- It is a combination of two or more topologies
- Inherits the advantages and disadvantages of the topologies included

Advantages:

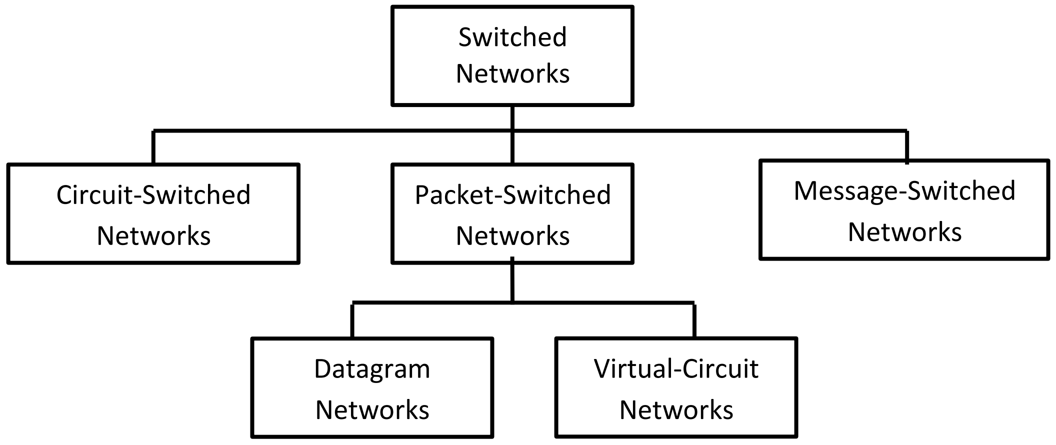
- Reliable as error detecting and troubleshooting is easy.
- Scalable as size can be increased easily.
- Flexible.

Disadvantages:

- Complex in design.
- Costly.

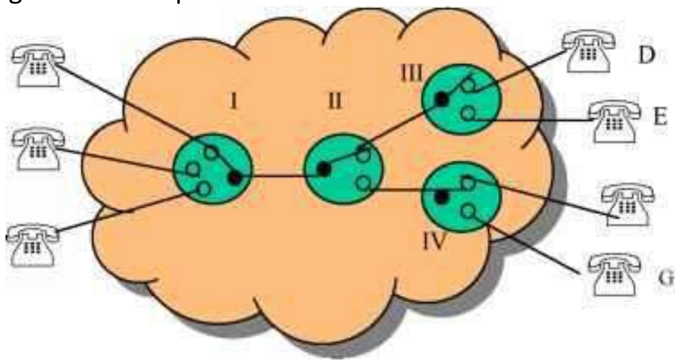
The Network Core

- Network core defines the connection of different network segments together and the process to transmit the data packets across the network.
- The network core is implemented through the use of switching techniques.
- The classification of a switching network is shown below:

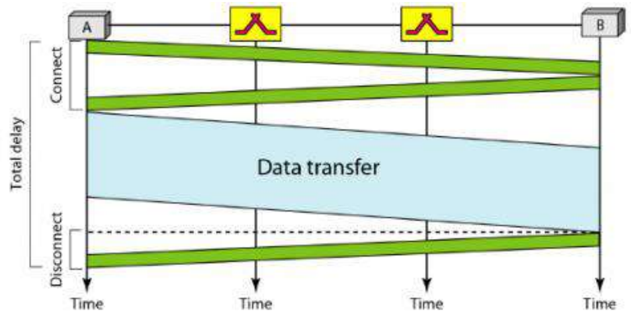


Circuit Switching

- Circuit switching is used in public telephone networks and is the basis for private networks built on leased-lines.
- Circuit switching was developed to handle voice traffic but also digital data (although inefficient)
- With circuit switching a dedicated path is established between two stations for communication.



- Switching and transmission resources within the network are reserved for the exclusive use of the circuit for the duration of the connection.
- The connection is transparent: once it is established, it appears to attach devices as if there were a direct connection.
- Communication via circuit switching involves three phases:
    1. Circuit Establishment
    2. Data Transfer
    3. Circuit Disconnect



- Connection path must be established before data transmission begins. Nodes must have switching capacity and channel capacity to establish a connection.
- Circuit switching is inefficient
    1. Channel capacity dedicated for a duration of a connection
    2. If no data, capacity wasted
- Set up (connection) takes time
- Once connected, a transfer is transparent to the users
    1. Data is transmitted at a fixed data rate with no delay (except for the propagation delay)
- Developed for voice traffic (phone)
    1. May also be used for data traffic via modem
- Interconnection of telephones within a building or office.
- In circuit switching, a direct physical connection between two devices is created by space-division switches, time-division switches, or both OR Circuit switching use any of below two technologies:

Space Division Switching

- In a space-division switch, the path from one device to another is spatially separate from other paths.
- Developed for the analogue environment.
- A crossbar is the most common space-division switch. It connects n inputs to m outputs via n × m cross points.
- Crossbar switch.

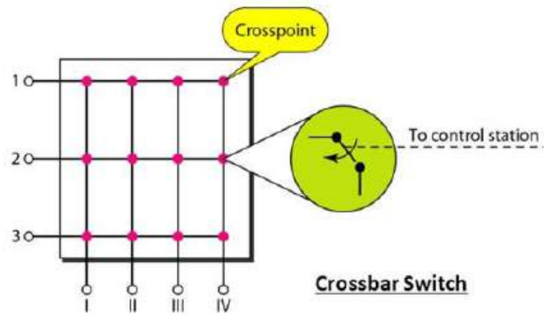


Figure 15: Space Division Switching

Time Division Switching

- In a time-division switch, the inputs are divided in time, using TDM. A control unit sends the input to the correct output device.
- Use digital time division techniques to set up and maintain virtual circuits.

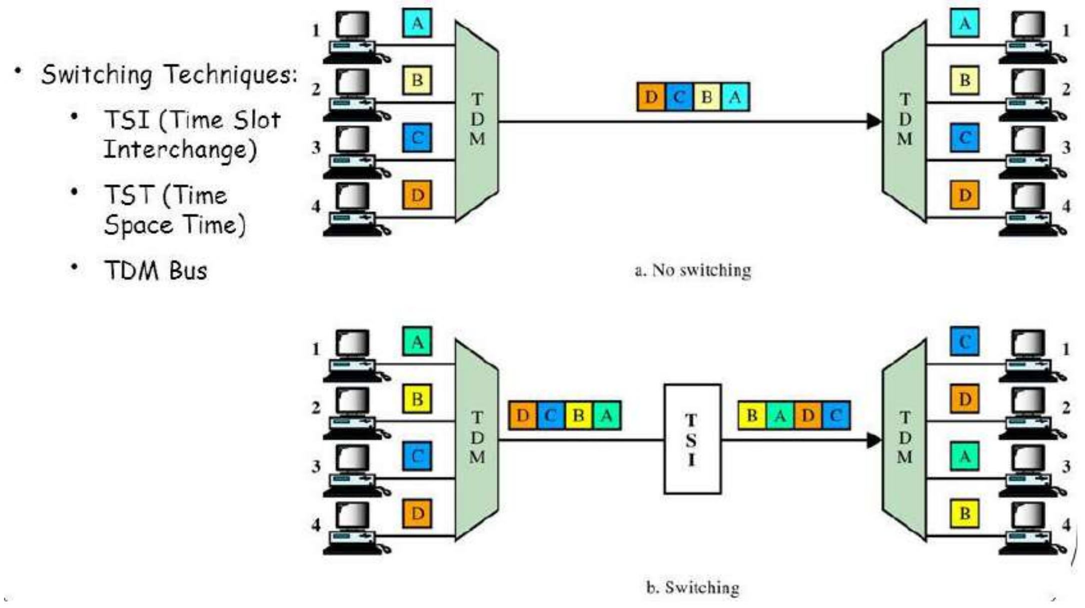


Figure 16: Time Division Switching

Packet Switching

- Packet switching was designed to provide a more efficient facility than circuit-switching for burst data traffic.
- With packet switching, a station transmits data in small blocks, called packets.
- At each node, packets are received, stored briefly (buffered) and passed on to the next node.
    1. Store and forward mechanism
- Each packet contains some portion of the user data plus control info needed for proper functioning of the network.
- A key element of packet-switching networks is whether the internal operation is datagram or virtual circuit (VC).
    1. With internal VCs, a route is defined between two endpoints and all packets for that VC follow the same route.
    2. With internal diagrams, each packet is treated independently, and packets intended for the same destination may follow different routes.
- Examples of packet switching networks are X.25, Frame Relay, ATM and IP.
- Station breaks a long message into packets. Packets sent one at a time to the network.
- Packets handled in two ways:
    1. **Datagram**
        - Each packet treated independently
        - Packets can take any practical route
        - Packets may arrive out of order
        - Packets may go missing
        - Up to receiver to re-order packets and recover from missing packets
    2. **Virtual Circuit**
        - Pre-planned route established before any packets sent.

  - Once the route is established, all the packets between the two communicating parties follow the same route through the network
  - Call request and call accept packets to establish a connection (handshake)
  - Each packet contains a Virtual Circuit Identifier (VCI) instead of a destination address
  - No routing decisions required for each packet
  - Clear request to drop circuit
  - Not a dedicated path

Message Switching

- This technique was somewhere in the middle of circuit switching and packet switching.
- In message switching, the whole message is treated as a data unit and is transferred in its entirety.
- A switch working on message switching first receives the whole message and buffers it until there are resources available to transfer it to the next hop.
- If the next hop is not having enough resource to accommodate large size message, the message is stored and switch waits.

Protocols layers and their service model

OSI Layer Architecture

- OSI model is based on a proposal developed by the International Standards Organization (ISO) as the first step toward international standardization of the protocols used in the various layers.
- It was revised in 1995.
- The model is called the OSI (Open Systems Interconnection) Reference Model because it deals with connecting open systems—that is, systems that are open for communication with other systems.
- The OSI model has seven layers.
  1. Physical Layer
  2. Data Link Layer
  3. Network Layer
  4. Transport Layer
  5. Session Layer
  6. Presentation Layer
  7. Application Layer

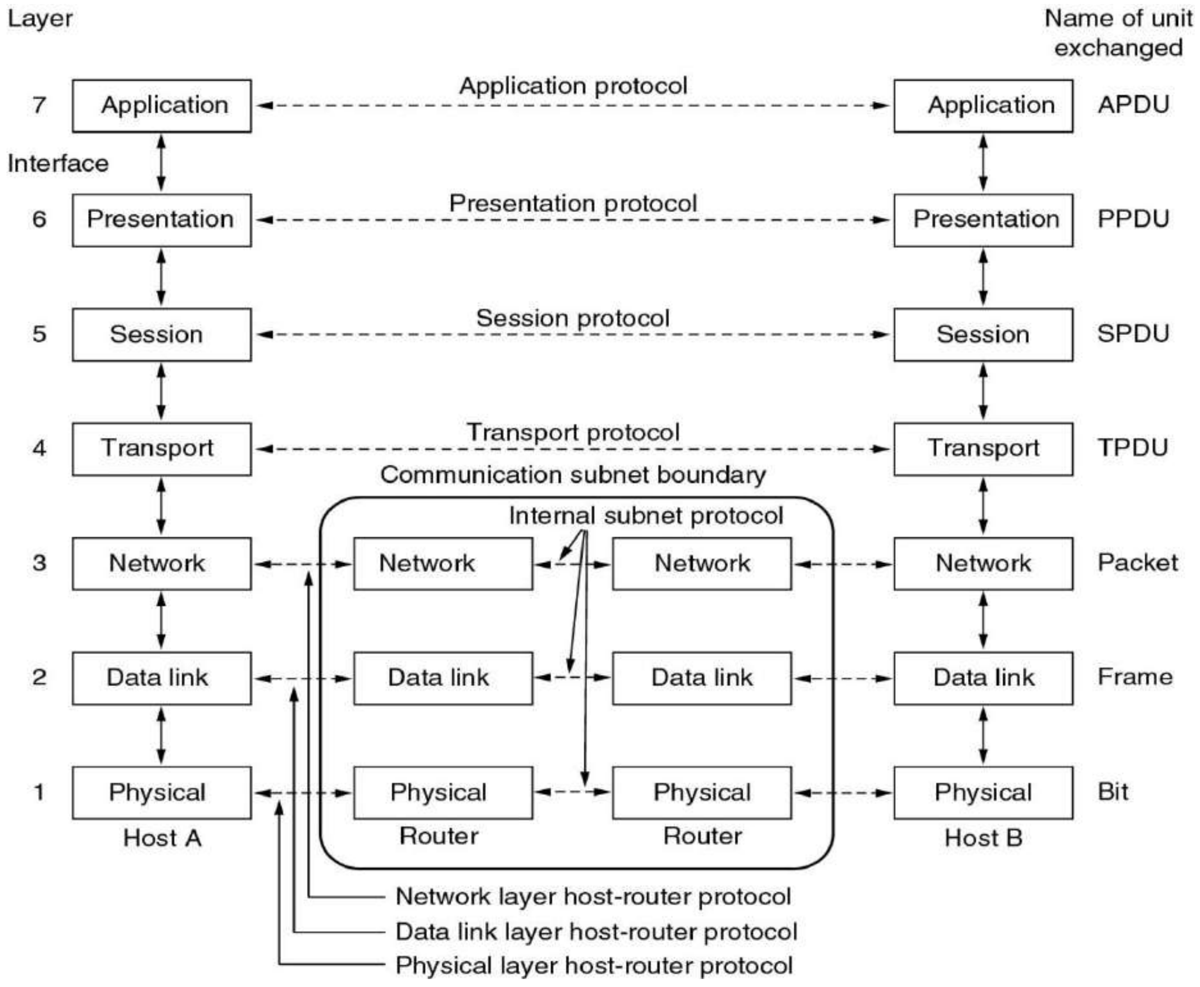


Figure 17: OSI Reference Model

Physical Layer

- The physical layer, the lowest layer of the OSI model, is concerned with the transmission and reception of the unstructured raw bit stream over a physical medium.
- It describes the electrical/optical, mechanical, and functional interfaces to the physical medium, and carries the signals for all of the higher layers. It provides:
- **Data encoding**: modifies the simple digital signal pattern (1s and 0s) used by the PC to better accommodate the characteristics of the physical medium, and to aid in a bit and frame synchronization.
- **Transmission technique**: determines whether the encoded bits will be transmitted by baseband (digital) or broadband (analog) signalling.
- **Physical medium transmission**: transmits bits as electrical or optical signals appropriate for the physical medium.

Data link Layer

- The data link layer provides error-free transfer of data frames from one node to another over the physical layer, allowing layers above it to assume virtually error-free transmission over the link.
- To do this, the data link layer provides:
- **Link establishment and termination**: establishes and terminates the logical link between two nodes.
- **Frame traffic control**: tells the transmitting node to "back-off" (stop) when no frame buffers are available.
- **Frame sequencing**: transmits/receives frames sequentially.
- **Frame acknowledgment**: provides/expects frame acknowledgments. Detects and recovers from errors that occur in the physical layer by retransmitting non-acknowledged frames and handling duplicate frame receipt.
- **Frame delimiting**: creates and recognizes frame boundaries.
- **Frame error checking**: checks received frames for integrity.
- **Media access management**: determines when the node "has the right" to use the physical medium.

Network Layer

- The network layer controls the operation of the subnet, deciding which physical path the data should take based on network conditions, a priority of service, and other factors.
- To do this, the data link layer provides:
- **Routing**: routes frames among networks.
- **Subnet traffic control**: routers (network layer intermediate systems) can instruct a sending station to "throttle back" its frame transmission when the router's buffer fills up.
- **Frame fragmentation**: if it determines that a downstream router's maximum transmission unit (MTU) size is less than the frame size, a router can fragment a frame for transmission and re-assembly at the destination station.
- **Logical-physical address mapping** translates logical addresses or names, into physical addresses.
- **Subnet usage accounting**: has accounting functions to keep track of frames forwarded by subnet intermediate systems, to produce billing information.

Transport Layer

- The transport layer ensures that messages are delivered error-free, in sequence, and with no losses or duplications. It relieves (release) the higher layer protocols from any concern with the transfer of data between them and their peers.
- The size and complexity of a transport protocol depend on the type of service it can get from the network layer. For a reliable network layer with virtual circuit capability, a minimal transport layer is required. If the network layer is unreliable and/or only supports datagrams, the transport protocol should include extensive error detection and recovery.
- The transport layer provides:
- **Message segmentation**: accepts a message from the (session) layer above it, splits the message into smaller units (if not already small enough), and passes the smaller units down to the network layer. The transport layer at the destination station reassembles the message.
- **Message acknowledgment**: provides reliable end-to-end message delivery with acknowledgments.

- **Message traffic control**: tells the transmitting station to "back-off" when no message buffers are available.
- Typically, the transport layer can accept relatively large messages, but there are strict message size limits imposed by the network (or lower) layer. Consequently, the transport layer must break up the messages into smaller units, or frames, prepending a header to each frame.
- The transport layer header information must then include control information, such as message start and message end flags, to enable the transport layer on the other end to recognize message boundaries.
- In addition, if the lower layers do not maintain sequence, the transport header must contain sequence information to enable the transport layer on the receiving end to get the pieces back together in the right order before handing the received message up to the layer above.

Session Layer

- The session layer allows session establishment between processes running on different stations. It provides:
- **Session establishment, maintenance, and termination**: allows two application processes on different machines to establish, use and terminate a connection, called a session.
- **Session support**: performs the functions that allow these processes to communicate over the network, performing security, name recognition, logging, and so on.

Presentation Layer

- The presentation layer formats the data to be presented to the application layer. It can be viewed as the translator for the network. This layer may translate data from a format used by the application layer into a common format at the sending station, then translate the common format to a format known to the application layer at the receiving station.
- The presentation layer provides:
- **Character code translation**: for example, ASCII to EBCDIC.
- **Data conversion**: bit order, CR-CR/LF, integer-floating point, and so on.
- **Data compression** reduces the number of bits that need to be transmitted on the network.
- **Data encryption**: encrypt data for security purposes. For example, password encryption.

Application Layer

- The application layer serves as the window for users and application processes to access network services.
- This layer contains a variety of commonly needed functions:
  1. Resource sharing and device redirection
  2. Remote file access
  3. Remote printer access
  4. Inter-process communication
  5. Network management
  6. Directory services
  7. Electronic messaging (such as mail)
  8. Network virtual terminals

TCP/IP Reference Model (Internet Protocol Stack layers)

- Transmission Control Protocol/Internet Protocol (TCP/IP) protocol suite is the engine for the Internet and networks worldwide.
- TCP/IP either combines several OSI layers into a single layer or does not use certain layers at all.

- TCP/IP is a set of protocols developed to allow cooperating computers to share resources across the network.
- The TCP/IP model has five layers.
    1. Application Layer
    2. Transport Layer
    3. Internet Layer
    4. Data Link Layer
    5. Physical Network

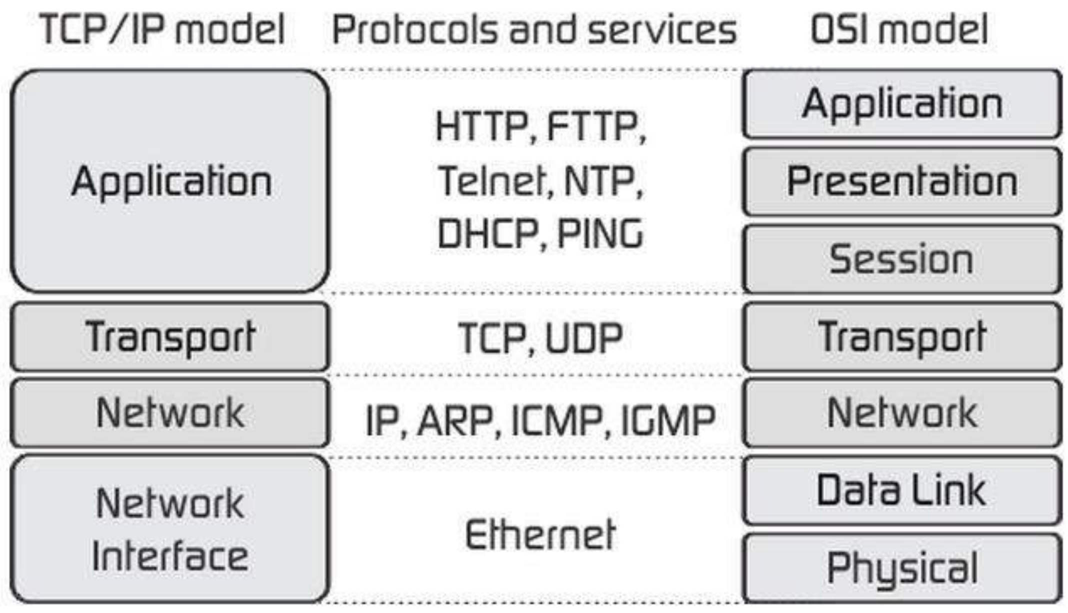


Figure 18: TCP/IP Reference Model

- As we can see from the above figure, the presentation and session layers are not there in the TCP/IP model. Also, note that the Network Access Layer in the TCP/IP model combines the functions of Data link Layer and Physical Layer.

Application Layer

- The application layer is the topmost layer of the four-layer TCP/IP model.
- The application layer is present on the top of the Transport layer.
- Application layer defines TCP/IP application protocols and how host programs interface with Transport layer services to use the network.
- Application layer includes all the higher-level protocols like DNS (Domain Naming System), HTTP (Hypertext Transfer Protocol), Telnet, SSH, FTP (File Transfer Protocol), TFTP (Trivial File Transfer Protocol), SNMP (Simple Network Management Protocol), SMTP (Simple Mail Transfer Protocol), DHCP (Dynamic Host Configuration Protocol), X Windows, RDP (Remote Desktop Protocol) etc.

Transport Layer

- The purpose of the Transport layer is to permit devices on the source and destination hosts to carry on a conversation.

- Transport layer defines the level of service and status of the connection used when transporting data.
- The transport layer provides the end-to-end data transfer by delivering data from an application to its remote peer.
- The most-used transport layer protocol is the Transmission Control Protocol (TCP), which provides:
  - Reliable delivery data
  - Duplicate data suppression
  - Congestion control
  - Flow control
- Another transport layer protocol is the User Datagram Protocol (UDP), which provides:
  - Connectionless
  - Unreliable
  - Best-effort service
- UDP is used by applications that need a fast transport mechanism and can tolerate the loss of some data.

Network Layer (Internet Layer)

- The internet layer also called the network layer.
- Internet layer pack data into data packets known as IP datagrams, which contain source and destination address (logical address or IP address) information that is used to forward the datagrams between hosts and across networks.
- The Internet layer is also responsible for the routing of IP datagrams.
- Internet Protocol (IP) is the most important protocol in this layer.
- It is a connectionless protocol that does not assume reliability from lower layers. IP does not provide reliability, flow control or error recovery.
- IP provides a routing function that attempts to deliver transmitted messages to their destination.
- These message units in an IP network are called an IP datagram.
- Example: IP, ICMP, IGMP, ARP, and RARP.

Network Interface Layer (Network Access Layer)

- Network Access Layer defines details of how data is physically sent through the network, including how bits are electrically or optically signalled by hardware devices that interface directly with a network medium, such as coaxial cable, optical fiber, or twisted pair copper wire.
- The protocols included in Network Access Layer are Ethernet, Token Ring, FDDI, X.25, Frame Relay etc.

OSI (Open System Interconnection)	TCP/IP (Transmission Control Protocol / Internet Protocol)
• It has 7 layers	• It has 4 layers
• OSI provides layer functioning and also defines functions of all the layers.	• TCP/IP model is more based on protocols and protocols are not flexible with other layers.
• In the OSI model, the transport layer guarantees the delivery of packets	• In the TCP/IP model, the transport layer does not guarantee delivery of packets.
• Follows horizontal approach	• Follows a vertical approach.
• OSI model has a separate presentation layer	• TCP/IP doesn't have a separate presentation layer

• OSI is a general model.	• TCP/IP model cannot be used in any other application.
• The network layer of the OSI model provides both connection-oriented and connectionless service.	• The Network layer in the TCP/IP model provides connectionless service.
• OSI model has a problem of fitting the protocols in the model	• TCP/IP model does not fit any protocol
• Protocols are hidden in the OSI model and are easily replaced as the technology changes.	• In TCP/IP replacing protocol is not easy.
• OSI model defines services, interfaces, and protocols very clearly and makes a clear distinction between them.	• In TCP/IP, it is not clearly separated its services, interfaces, and protocols.

Understanding of Delay, Loss, and Throughput in the Packet Switching Network

Basics

- Recall that a packet starts in a host (the source), passes through a series of routers, and ends its journey in another host (the destination).
- As a packet travels from one node (host or router) to the subsequent node (host or router) along this path, the packet suffers from several types of delays at each node along the path.
- The most important of these delays are the
  - Nodal processing delay
  - Queuing delay
  - Transmission delay
  - Propagation delay
- Together, these delays accumulate to give a total nodal delay.
- The performance of many Internet applications—such as search, Web browsing, email, maps, instant messaging, and voice-over-IP—are greatly affected by network delays.

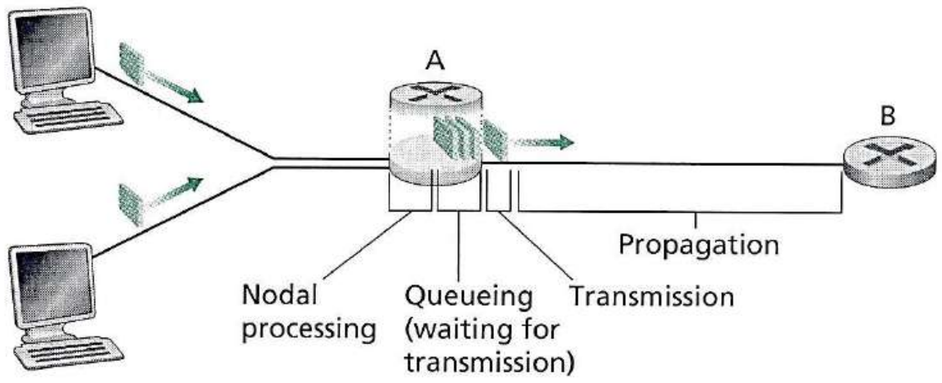


Figure 19: Delay in Packet Switched Network

Processing Delay

- The time required to examine the packet's header and determine where to direct the packet is part of the processing delay.

- The processing delay can also include other factors, such as the time needed to check for bit-level errors in the packet that occurred in transmitting the packet's bits from the upstream node to the router.
- It is typically on the order of microseconds or less.

Queuing Delay

- At the queue, the packet experiences a queuing delay as it waits to be transmitted onto the link.
- The length of the queuing delay of a specific packet will depend on the number of earlier-arriving packets that are queued and waiting for transmission onto the link.
- If the queue is empty and no other packet is currently being transmitted, then our packet's queuing delay will be zero.
- On the other hand, if the traffic is heavy and many other packets are also waiting to be transmitted, the queuing delay will be long.
- Queuing delays can be on the order of microseconds to milliseconds in practice.

Transmission Delay

- Assuming that packets are transmitted in a first-come-first-served manner like packet-switched networks.
- Now packet can be transmitted only after all the packets that have arrived before it have been transmitted.
- Denote the length of the packet by L bits, and denote the transmission rate of the link from a router to a router by R bits/sec.
- The transmission delay is L/R.
- Transmission delays are typically on the order of microseconds to milliseconds in practice.

Propagation Delay

- Once a bit is pushed into the link, it needs to propagate to router B. The time required to propagate from the beginning of the link to router B is the propagation delay.
- The bit propagates at the propagation speed of the link.
- The propagation speed depends on the physical medium of the link.
- Propagation delays are on the order of milliseconds.
- Propagations delay=d (Length of Physical Link) /s (Propagation speed in medium).

Packet Loss

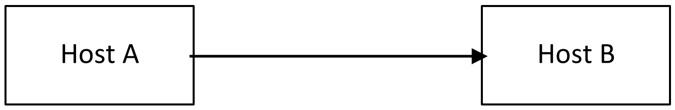
- Packet loss is the failure of one or more transmitted packets to arrive at their destination.
- This event can cause noticeable effects on all types of digital communications.
- The loss of data packets depends on the switch queue. The loss of data packets increases with the increases in the traffic intensity.
- It affects the performance of the network.

Throughput

- Throughput or Network Throughput is the rate of successful message delivery over a communication channel.
- The data these messages belong to may be delivered over a physical or logical link or it can pass through a certain network node.
- Throughput is usually measured in bits per second (bit/s or bps), and sometimes in data packets per second (p/s or pps) or data packets per time slot.

Problem:

In this problem, we consider sending real-time voice from Host A to Host B over a packet-switched network (VoIP). Host A converts analog voice to a digital 64 kbps bit stream on the fly. Host A then groups the bits into 56-byte packets. There is one link between Hosts A and B; its transmission rate is 2 Mbps and its propagation delay is 10 msec. As soon as Host A gathers a packet, it sends it to Host B. As soon as Host B receives an entire packet, it converts the packet's bits to an analog signal. How much time elapses from the time a bit is created (from the original analog signal at Host A) until the bit is decoded (as part of the analog signal at Host B)?



Given:

- Analog to Digital conversion rate = 64 kbps
- Packet size = 56 bytes (Convert into bits so 56 bytes = 56 * 8 = 448 bits).
- Transmission rate = 2 Mbps
- Propagation delay = 10 msec

- Since this is a packet switched network, the data will be transmitted packet by packet.
- A packet is 56 byte and the analog to digital conversation rate is 64 kbps, thus the preparing time
  PT for a packet is 448/(64*1000)= 0.007 sec = 7 msec
- The transition delay TD for a packet is (Size or Length of packet) / (Speed or Transmission rate)
  So, TD = (56*8) / (2*1000*1000) =0.000224 s = 0.224msec.
- Propagation delay PD = 10msec (Given in sum)
- Finally, the total time elapses from the time a bit is create until the bit is decoded is PT+TD+PD= 7+0.224+10 = 17.224 msec.

Network Applications OR Principles of Network Applications

- A Network application is an application running on one host and provides communication to another application running on a different host.
- At the core of network application development is writing programs that run on different end systems and communicate with each other over the network.
- In the Web application, there are two distinct programs that communicate with each other: the browser program running in the user's host; and the Web server program running in the Web server host.
- Examples of network applications are
    - e-mail
    - web
    - text messaging
    - remote login
    - P2P file sharing
    - multi-user network games
    - streaming stored video (YouTube)
    - voice over IP (e.g., Skype)
    - real-time video conferencing
    - social networking

Network Application Architectures

Network Application Architectures

- There are two possible structure of applications
    1. Client-Server architecture
    2. P2P (Peer to Peer) architecture

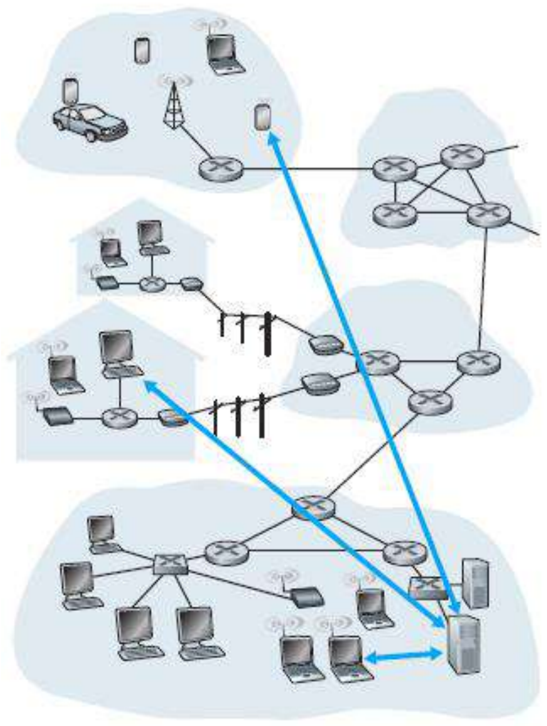
Client-Server architecture

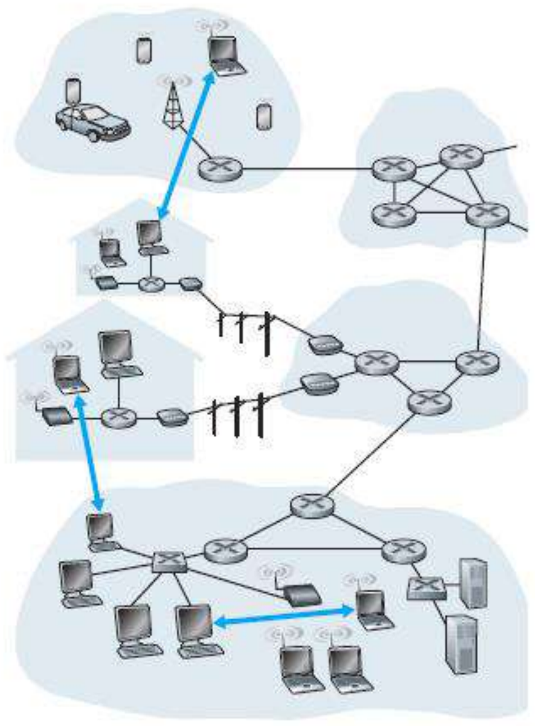
- In a client-server architecture, there is an always-on host, called the server, which provides services requested from many other hosts, called clients.
- A classic example is the Web application for which an always-on Web server services requests from browsers running on client hosts. When a Web server receives a request for an object from a client host, it responds by sending the requested object to the client host.
- Note that with the client-server architecture, clients do not directly communicate with each other; for example, in the Web application, two browsers do not directly communicate with each other.
- Another characteristic of the client-server architecture is that the server has a fixed, well-known address, called an IP address. Because the server is always on, a client can always contact the server by sending a packet to the server’s IP address.
- Some of the better-known applications with client-server architecture include the Web, FTP, Telnet, and e-mail.

P2P architecture

- In P2P architecture, there is no dedicated server.
- Pairs of hosts, called peers communicate directly with each other.

- Because the peers communicate without passing through a dedicated server, the architecture is called peer-to-peer.
- Many of today's most popular and traffic-intensive applications are based on P2P architectures.


Client-Server architecture


P2P (Peer to Peer) architecture

Processes Communicating

Process

- A process is an instance of a program running on a computer or we can say that process is a program under execution.
- When processes are running on the same end system, they can communicate with each other with interprocess communication, using rules that are governed by the end system's operating system.
- Processes on two different end systems communicate with each other by exchanging messages across the computer network.
- A sending process creates and sends messages into the network; a receiving process receives these messages and possibly responds by sending messages back.
- In the context of a communication session between a pair of processes, the process that initiates the communication is called the client. The process that waits to be contacted to begin the session is the server.
- The client is a process (program) that sends a message to a server process (program), requesting that the server perform a task (service).
- A server is a process (program) that fulfills the client request by performing the task requested by the client. Server programs generally receive requests from client programs, execute it and dispatch responses to client requests.

An interface between the Process and the Computer Network OR Socket

- Any message sent from one process to another must go through the underlying network.
- A process sends messages into and receives messages from, the network through a software interface called a socket.
- A process is similar to a house and its socket is similar to its door. When a process wants to send a message to another process on another host, it shoves (passes) the message out its door (socket). This sending process assumes that there is a transportation infrastructure on the other side of its door that will transport the message to the door of the destination process. Once the message arrives at the destination host, the message passes through the receiving process's door (socket), and the receiving process then acts on the message.

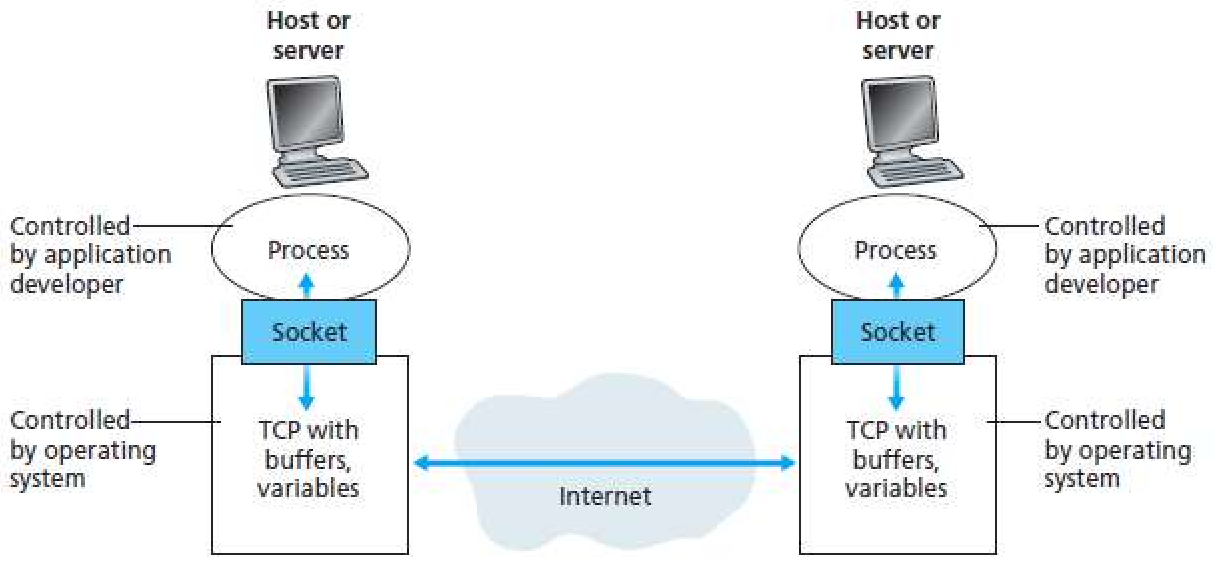

Host or server
Host or server
Controlled by application developer
Process
Socket
Controlled by operating system
TCP with buffers, variables
Internet
Process
Socket
TCP with buffers, variables
Controlled by application developer
Controlled by operating system

Fig. 2 Application processes, sockets, and underlying transport protocol

Addressing Processes

- To identify the receiving process, two pieces of information need to be specified:
    1. the address of the host and
    2. an identifier that specifies the receiving process in the destination host.
- On the Internet, the host is identified by its IP address.
- An IP address is a 32-bit that uniquely identify the host.
- In addition to knowing the address of the host to which a message is destined, the sending process must also identify the receiving process (more specifically, the receiving socket) running in the host.
- This information is needed because in general, a host could be running many network applications.

Transport Services available to Applications

Reliable Data Transfer

- Reliable data transfer is to guarantee that the data sent by one end of the application is delivered correctly and completely to the other end of the application.

- If a protocol provides such a guaranteed data delivery service, it is said to provide reliable data transfer.
- One important service that a transport-layer protocol can potentially provide to an application is process-to-process reliable data transfer.
- When a transport protocol provides this service, the sending process can just pass its data into the socket and know with complete confidence that the data will arrive without errors at the receiving process.

Throughput

- Throughput is the rate at which the sending process can deliver bits to the receiving process.
- The transport protocol ensures that the available throughput is always at least r bits/sec.

Timing

- A transport-layer protocol can also provide timing guarantees.
- An example guarantee might be that every bit that the sender pumps into the socket arrives at the receiver's socket no more than 100 msec later. Such a service would be appealing to interactive real-time applications, such as Internet telephony, virtual environments, teleconferencing, and multiplayer games, all of which require tight timing constraints on data delivery in order to be effective.

Security

- Finally, a transport protocol can provide an application with one or more security services.
- For example, in the sending host, a transport protocol can encrypt all data transmitted by the sending process, and in the receiving host, the transport-layer protocol can decrypt the data before delivering the data to the receiving process.

Difference between TCP (Service) and UDP (Service)

Description	*TCP*	*UDP*
Full Name	Transmission Control Protocol	User Datagram Protocol
Connection	TCP is a connection-oriented protocol.	UDP is a connectionless protocol.
Function	A point to point connection is established between client and server before sending a message.	A point to point connection is not established before sending messages.
Usage	TCP is suited for applications that require high reliability, and transmission time is relatively less critical.	UDP is suitable for applications that need a fast, efficient transmission, such as games.
Reliability	There is an absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent.	There is no guarantee that the messages or packets sent would reach at all.
Use by other protocols	HTTP, HTTPs, FTP, SMTP, Telnet	DNS, DHCP, SNMP, RIP, VOIP
The 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.

Description	*TCP*	*UDP*
Speed of transfer	The speed for TCP is slower than UDP.	UDP is faster because there is no error-checking for packets.
Header Size	TCP header size is 20 bytes	UDP Header size is 8 bytes.
Data Flow Control	TCP does Flow Control.	UDP does not have an option for flow control.
Error Checking	TCP does error checking	UDP does error checking, but no recovery options.
Acknowledgment	Acknowledgment segments	No Acknowledgment
Handshake	SYN, SYN-ACK, ACK	No handshake

The Web and HTTP

Web

- A Web page consists of objects.
- An object is simply a file - such as an HTML file, a JPEG image, a Java applet, or a video clip, that is addressable by a single URL.
- Most Web pages consist of a base HTML file and several referenced objects.
- For example, if a Web page contains HTML text and five JPEG images, then the Web page has six objects: the base HTML file plus the five images.
- The base HTML file references the other objects in the page with the objects' URLs.
- Each URL has two components: the hostname of the server that houses the object and the object's path name.
- For example, the URL *"http://www.someSchool.edu/someDepartment/picture.gif"* has "www.someSchool.edu" is hostname and "/someDepartment/picture.gif" is a pathname.

HTTP (HyperText Transfer Protocol)

- HTTP is Web's application layer protocol which defines how Web clients request Web pages from Web servers and how servers transfer Web pages to clients.

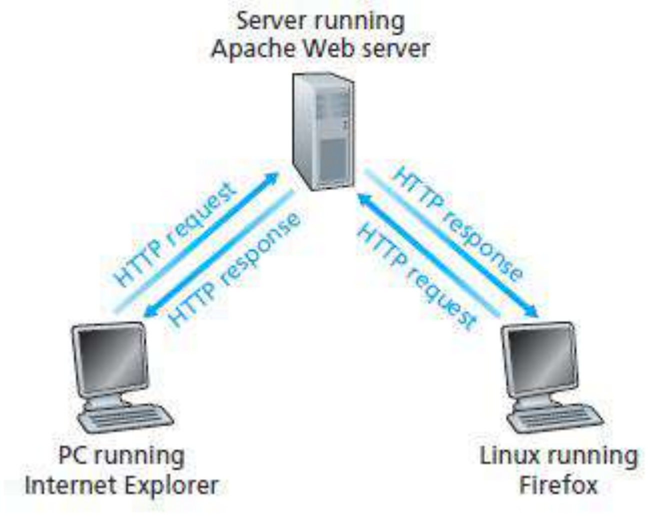
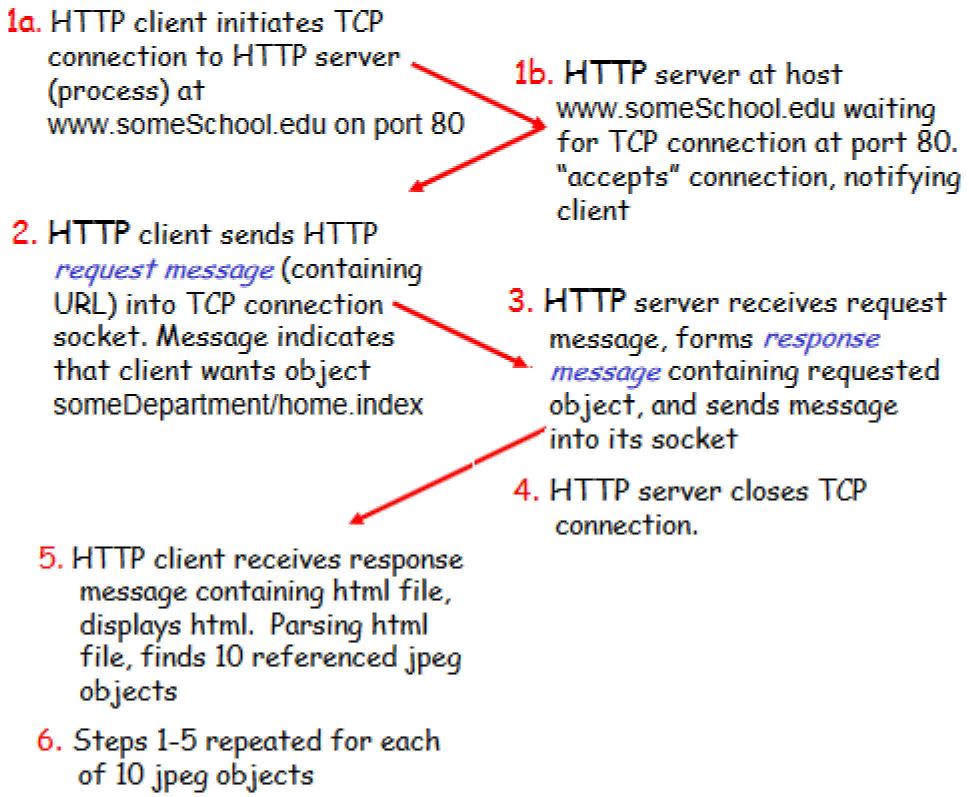


Fig. 3 HTTP request-response behavior

- When a user requests a Web page (for example, clicks on a hyperlink), the browser sends HTTP request messages for the objects in the page to the server.
- The server receives the requests and responds with HTTP response messages that contain the objects.
- HTTP uses TCP as its underlying transport protocol.
- The HTTP client first initiates a TCP connection with the server.
- Once the connection is established, the browser and the server processes access TCP through their socket interfaces.
- HTTP follows client/server model
  - **client**: a browser that requests, receives, (using HTTP protocol) and "displays" Web objects
  - **server**: Web server sends (using HTTP protocol) objects in response to requests
- HTTP connection types
  1. Non-persistent HTTP
  2. Persistent HTTP

Non-persistent HTTP

- A non-persistent connection is the one that is closed after the server sends the requested object to the client. In other words, the connection is used exactly for one request and one response.
- For downloading multiple objects it required multiple connections.
- Non-persistent connections are the default mode for HTTP/1.0.
- suppose a user enters URL: "www.someSchool.edu/someDepartment/home.index"
- Above link contains text and references to 10 jpeg images.


1a. HTTP client initiates TCP connection to HTTP server (process) at www.someSchool.edu on port 80
1b. HTTP server at host www.someSchool.edu waiting for TCP connection at port 80. "accepts" connection, notifying client
2. HTTP client sends HTTP request message (containing URL) into TCP connection socket. Message indicates that client wants object someDepartment/home.index
3. HTTP server receives request message, forms response message containing requested object, and sends message into its socket
4. HTTP server closes TCP connection.
5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects
6. Steps 1-5 repeated for each of 10 jpeg objects

1. The HTTP client process initiates a TCP connection to the server www.someSchool.edu on port number 80, which is the default port number for HTTP. Associated with the TCP connection, there will be a socket at the client and a socket at the server.
2. The HTTP client sends an HTTP request message to the server via its socket. The request message includes the path name /someDepartment/home.index.
3. The HTTP server process receives the request message via its socket, retrieves the object /someDepartment/home.index from its storage (RAM or disk), encapsulates the object in an HTTP response message, and sends the response message to the client via its socket.
4. The HTTP server process tells TCP to close the TCP connection. (But TCP doesn't actually terminate the connection until it knows for sure that the client has received the response message intact.).
5. The HTTP client receives the response message. The TCP connection terminates. The message indicates that the encapsulated object is an HTML file. The client extracts the file from the response message, examines the HTML file, and finds references to the 10 JPEG objects.
6. The first four steps are then repeated for each of the referenced JPEG objects.

- **RTT** (**Round Trip Time) -** which is the time it takes for a small packet to travel from client to server and then back to the client.

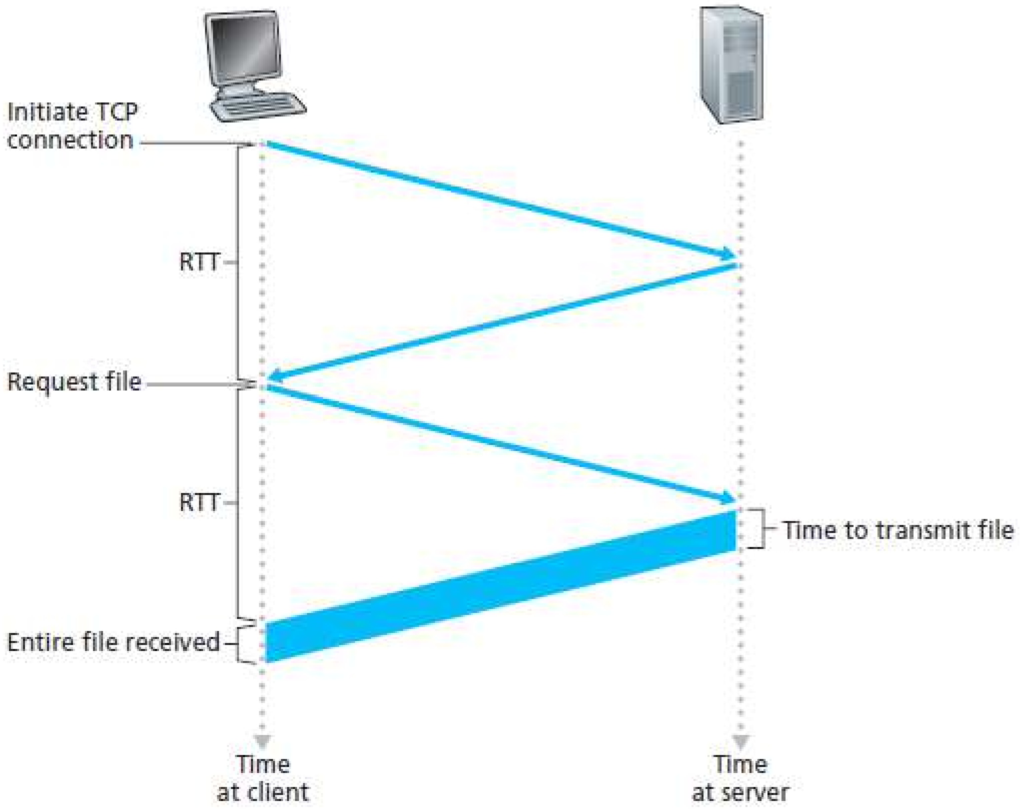


Fig. 4 Calculation for the time needed to request and receive an HTML file

- HTTP response time:
    1. one RTT to initiate TCP connection
    2. one RTT for HTTP request and first few bytes of the HTTP response to return
    3. file transmission time
- Non-persistent HTTP response time = 2RTT+ file transmission time
- It is overhead for each TCP connection.

Persistent HTTP

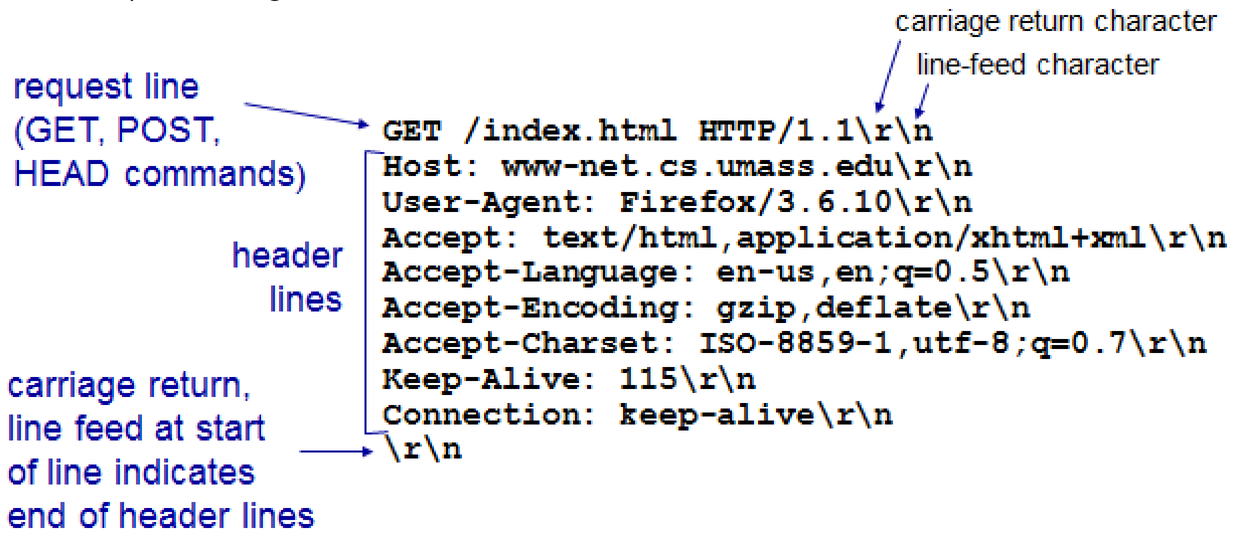
- With persistent connections, the server leaves the TCP connection open after sending responses and hence the subsequent requests and responses between the same client and server can be sent.
- The server closes the connection only when it is not used for a certain configurable amount of time.
- It requires as little as one RTT for all the referenced objects
- With persistent connections, the performance is improved by 20%.
- Persistent connections are the default mode for HTTP/1.1.

HTTP message format

- There are two types of HTTP messages:
    1. Request
    2. Response

HTTP request message

- An HTTP request message is in ASCII format which means in human-readable format.


carriage return character
line-feed character
request line (GET, POST, HEAD commands)
header lines
carriage return, line feed at start of line indicates end of header lines

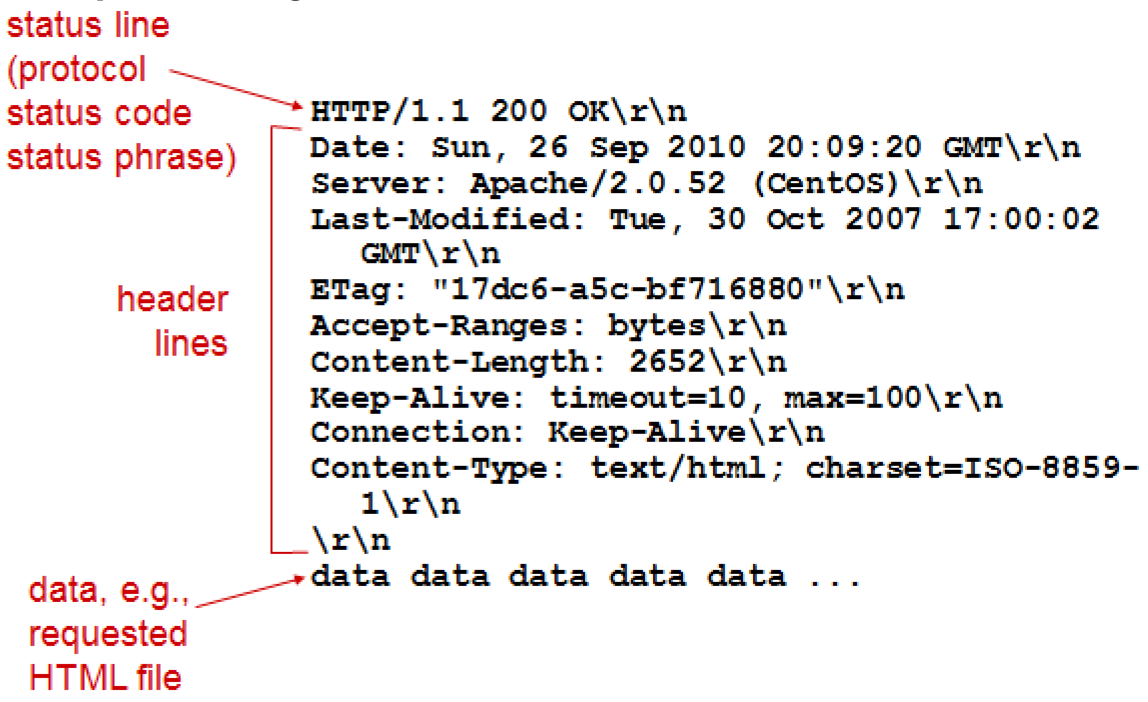
```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

Fig. 5 HTTP request message

- HTTP request message consist three part
    1. Request line
    2. Header line
    3. Carriage return
- The message is written in ordinary ASCII text so that your ordinary computer-literate human being can read it.
- Each line is followed by a carriage return and a line feed.
- The last line is followed by an additional carriage return and line feed.
- The first line of an HTTP request message is called the request line; the subsequent lines are called the header lines.

- The request line has three fields: the method field, the URL field, and the HTTP version field.
- The method field can take on several different values, including getting, POST, HEAD, PUT, and DELETE.
- In this example, the browser is requesting the object /somedir/page.html. The version is self-explanatory; in this example, the browser implements version HTTP/1.1.
- The header line Host: www.someschool.edu specifies the host on which the object resides.

HTTP response message



```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.0.52 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02
   GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-
   1\r\n
\r\n
data data data data data ...
```

Fig. 6 HTTP request message

- HTTP response message consist three part
  1. Status line 2. Header line 3. Data (Entity body)
- The status line has three fields: the protocol version field, a status code, and a corresponding status message.
- In this example, the status line indicates that the server is using HTTP/1.1 and that everything is OK.
- The Date: header line indicates the time and date when the HTTP response was created and sent by the server.
- The Server: header line indicates that the message was generated by an Apache Web server.
- The Last-Modified: header line indicates the time and date when the object was created or last modified.
- The Content-Length: header line indicates the number of bytes in the object being sent.
- The Content-Type: header line indicates that the object in the entity body is HTML text.

User-Server Interaction OR Cookies

- A small text file created by a Web site that is stored in the user's computer either temporarily for that session only or permanently on the hard disk (persistent cookie).
- Cookies provide a way for the Web site to recognize you and keep track of your preferences.

- Cookie technology has four components
    1. a cookie header line in the HTTP response message
    2. a cookie header line in the HTTP request message
    3. a cookie file kept on the user's end system and managed by the user's browser
    4. a back-end database at the Web site

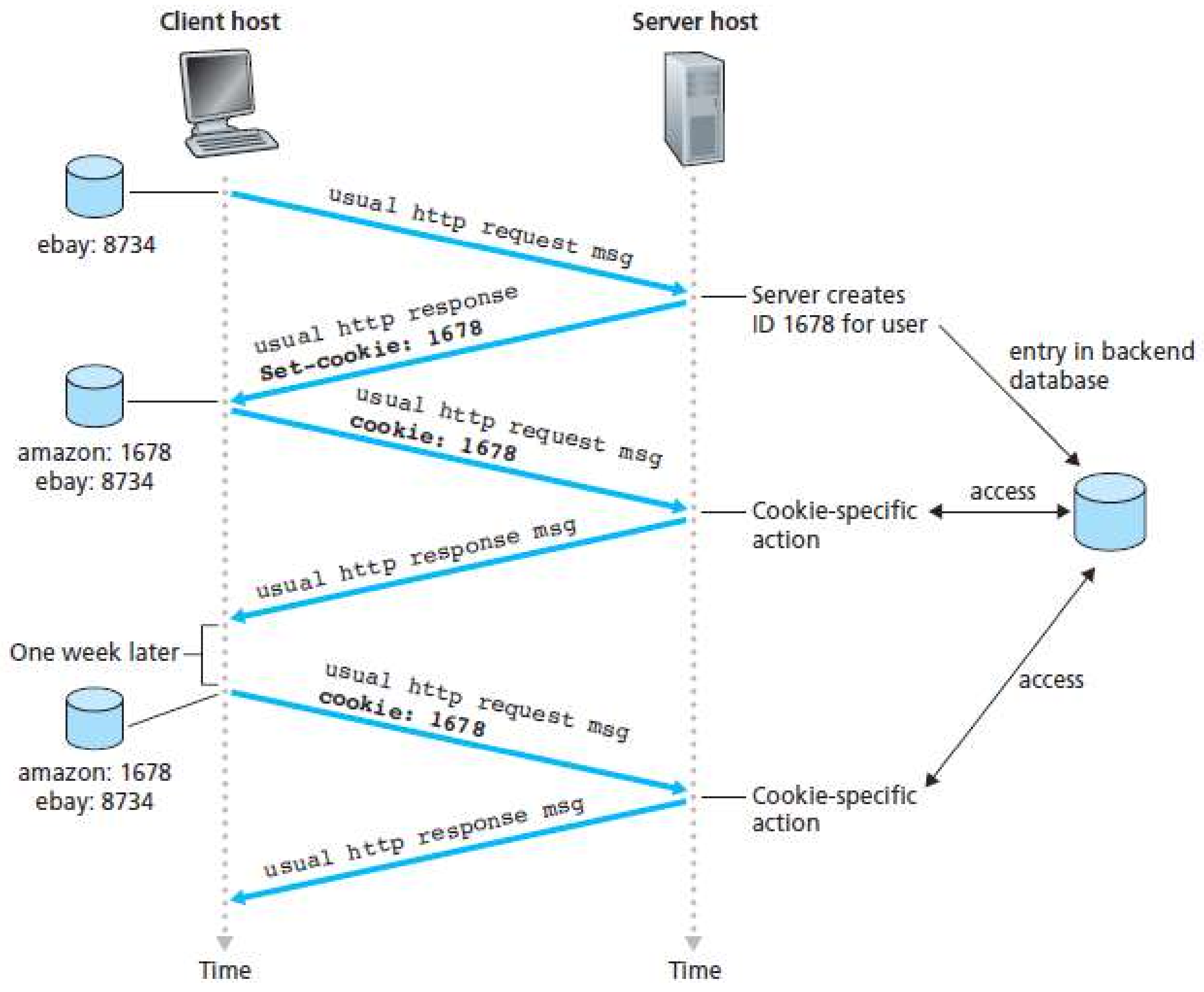


Fig. 7 keeping user state with cookies

- Suppose Susan, access Amazon.com for the first time.
- Let us suppose that in the past she has already visited the eBay site.
- When the request comes into the Amazon Web server, the server creates a unique identification number and creates an entry in its back-end database by the identification number.
- The Amazon Web server then responds to Susan's browser, including in the HTTP response a set-cookie: header, which contains the identification number.
- For example, the header line might be Set-cookie: 1678.
- When Susan's browser receives the HTTP response message, it sees the Set-cookie: header.
- The browser then appends a line to the special cookie file that it manages.

- This line includes the hostname of the server and the identification number in the Set-cookie: header.
- The cookie file already has an entry for eBay, since Susan has visited that site in the past.
- As Susan continues to browse the Amazon site, each time she requests a Web page, her browser consults her cookie file, extracts her identification number for this site, and puts a cookie header line that includes the identification number in the HTTP request.
- Specifically, each of her HTTP requests to the Amazon server includes the header line: Cookie: 1678.
- In this manner, the Amazon server is able to track Susan's activity at the Amazon site.

Use of cookies

- authorization
- shopping carts
- recommendations
- user session state (Web e-mail)

Web Caching OR Proxy Server

- A Web cache OR a proxy server is a network entity that satisfies HTTP requests on the behalf of an origin Web server.
- The Web cache has its own disk storage and keeps copies of recently requested objects in this storage.

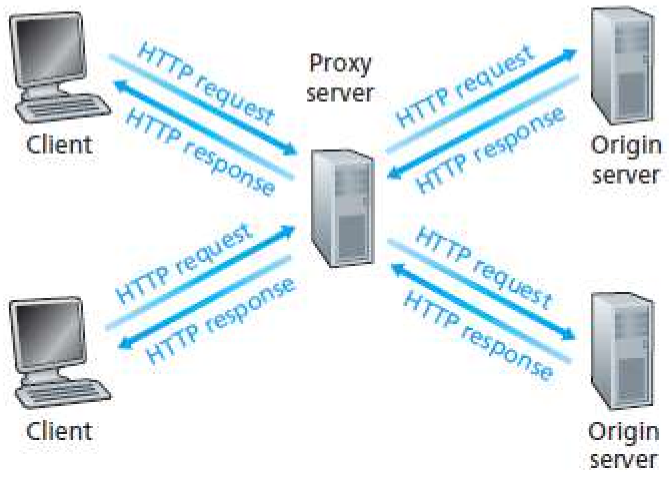


Fig. 8 Clients requesting objects through a Web cache

- A Web cache OR a proxy server is a network entity that satisfies HTTP requests on the behalf of an origin Web server.
- The Web cache has its own disk storage and keeps copies of recently requested objects in this storage.
- A user's browser can be configured so that all of the user's HTTP requests are first directed to the Web cache.
- As an example, suppose a browser is requesting the object http://www.someschool.edu/campus.gif.
- Here is what happens
  1. The browser establishes a TCP connection to the Web cache and sends an HTTP request for the object to the Web cache.

2. The Web cache checks to see if it has a copy of the object stored locally. If it does, the Web cache returns the object within an HTTP response message to the client browser.
3. If the Web cache does not have the object, the Web cache opens a TCP connection to the origin server, that is, to www.someschool.edu. The Web cache then sends an HTTP request for the object into the cache-to-server TCP connection. After receiving this request, the origin server sends the object within an HTTP response to the Web cache.
4. When the Web cache receives the object, it stores a copy in its local storage and sends a copy, within an HTTP response message, to the client browser.

- Note that a cache is both a server and a client at the same time.
- When it receives requests from and sends responses to a browser, it is a server.
- When it sends requests to and receives responses from an origin server, it is a client.

Why Web caching is needed (Required)? OR Advantages of Caching

- To reduce response time for a client request
- To reduce traffic on an institution’s access link
- To enable "poor" content providers to effectively deliver content

FTP (File Transfer Protocol)

- File Transfer Protocol (FTP) is the commonly used protocol for exchanging files over the Network or Internet.
- FTP uses the Internet's TCP/IP protocols to enable data transfer.
- FTP uses client-server architecture.
- FTP promotes sharing of files via remote computers with reliable and efficient data transfer.

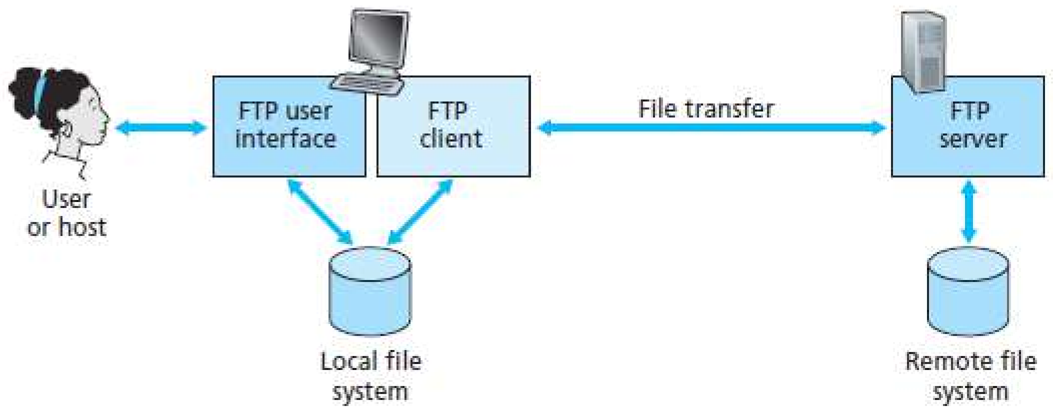


Fig. 9 FTP moves files between local and remote file systems

- In the above figure, a user interacts with FTP through an FTP user agent.
- The user first provides the hostname of the remote host, causing the FTP client process in the local host to establish a TCP connection with the FTP server process in the remote host.
- The user then provides the user identification and password, which are sent over the TCP connection as part of FTP commands.
- Once the server has authorized the user, the user copies one or more files stored in the local file system into the remote file system (or vice versa).
- FTP uses two parallel TCP connections to transfer a file,
    1. control connection
    2. data connection

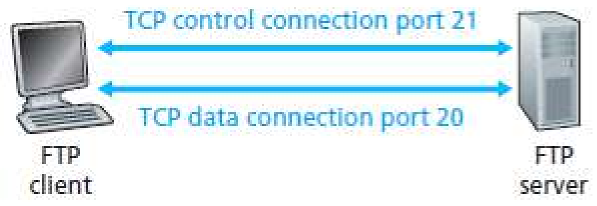


Fig. 10 Control and data connections

- The control connection is used for sending control information between the two hosts such as user identification, password, commands to change remote directory and commands to put and get files.
- The data connection is used to actually send a file.
- Because FTP uses a separate control connection, FTP is said to send its control information out-of-band.
- When a user starts an FTP session with a remote host, the client side of FTP (user) first initiates a control TCP connection with the server side (remote host) on server port number 21.
- The client side of FTP sends the user identification and password over this control connection.
- The client side of FTP also sends, over the control connection, commands to change the remote directory.
- When the server side receives a command for a file transfer over the control connection (either to or from, the remote host), the server side initiates a TCP data connection to the client side.
- FTP sends exactly one file over the data connection and then closes the data connection.
- If during the same session, the user wants to transfer another file, FTP opens another data connection.
- Thus, with FTP, the control connection remains open throughout the duration of the user session, but a new data connection is created for each file transferred within a session (that is, the data connections are non-persistent).

Electronic mail on the Internet (Email)

- As with ordinary postal mail, e-mail is an asynchronous communication medium in which people send and read messages when it is convenient for them, without having to coordinate with other people's schedules.
- In contrast with postal mail, electronic mail is fast, easy to distribute, and inexpensive.
- Modern e-mail has many powerful features, including messages with attachments, hyperlinks, HTML-formatted text, and embedded photos.
- Email has three major components
    1. user agents
    2. mail servers
    3. Simple Mail Transfer Protocol (SMTP)
- User agents allow users to read, reply to, forward, save, and compose messages.
- Microsoft Outlook and Apple Mail are examples of user agents for e-mail.

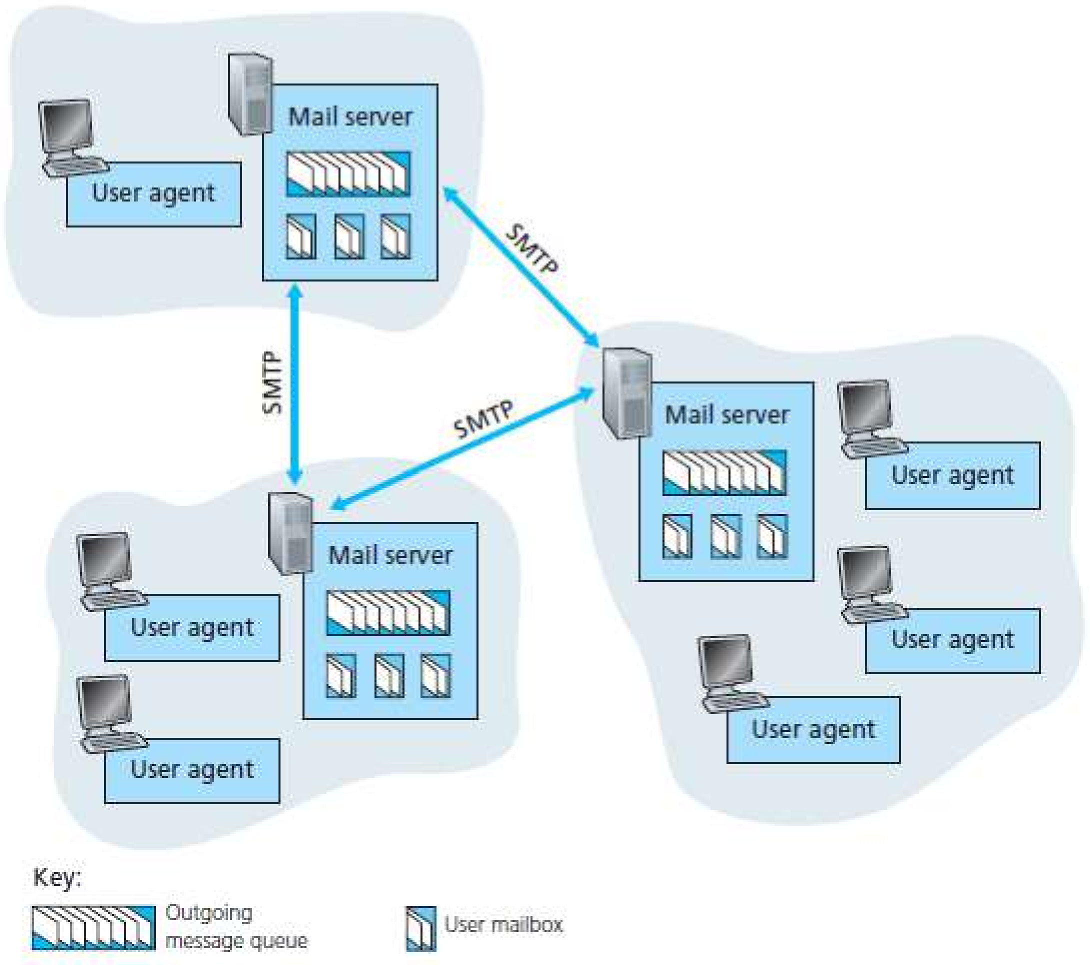


Fig. 11 A high-level view of the Internet e-mail system

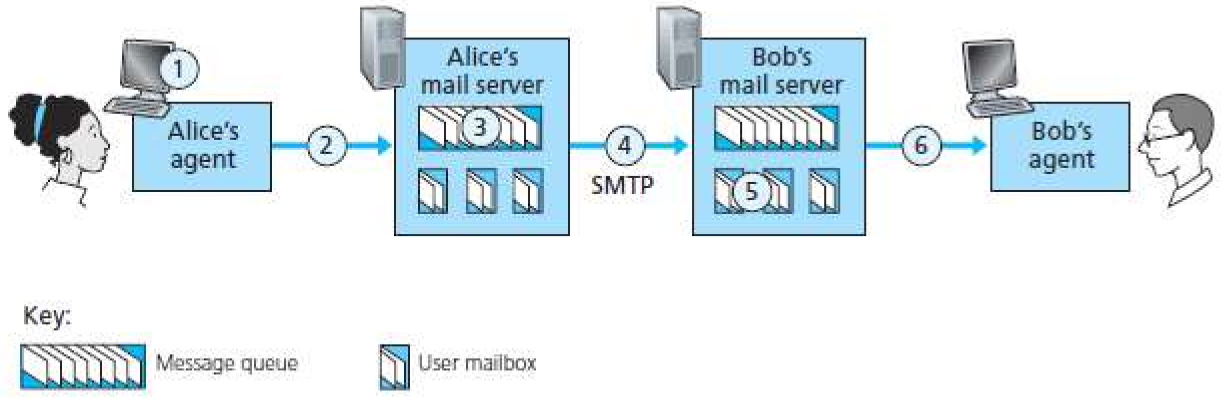


Fig. 12 Alice sends a message to Bob

- In the above figure when Alice is finished composing her message, her user agent sends the message to her mail server, where the message is placed in the mail server's outgoing message queue.

- When Bob wants to read a message, his user agent retrieves the message from his mailbox in his mail server.
- Mail servers form the core of the e-mail infrastructure.
- Each recipient, such as Bob, has a mailbox located in one of the mail servers.
- Bob's mailbox manages and maintains the messages that have been sent to him.
- A typical message starts its journey in the sender's user agent, travels to the sender's mail server and travels to the recipient's mail server, where it is deposited in the recipient's mailbox.
- When Bob wants to access the messages in his mailbox, the mail server containing his mailbox authenticates Bob (with usernames and passwords).
- Alice's mail server must also deal with failures in Bob's mail server.
- If Alice's server cannot deliver mail to Bob's server, Alice's server holds the message in a message queue and attempts to transfer the message later.
- Reattempts are often done every 30 minutes or so; if there is no success after several days, the server removes the message and notifies the sender (Alice) with an e-mail message.
- SMTP is the principal application-layer protocol for Internet electronic mail. It uses the reliable data transfer service of TCP to transfer mail from the sender's mail server to the recipient's mail server.
- SMTP has two sides: a client side, which executes on the sender's mail server, and a server side, which executes on the recipient's mail server.
- Both the client and server sides of SMTP run on every mail server.
- When a mail server sends mail to other mail servers, it acts as an SMTP client. When a mail server receives mail from other mail servers, it acts as an SMTP server.

SMTP (Simple Mail Transfer Protocol)

- SMTP transfers messages from senders' mail servers to the recipients' mail servers.
- It restricts the body (not just the headers) of all mail messages to simple 7-bit ASCII.
- To illustrate the basic operation of SMTP, let's take a common scenario. Suppose Alice wants to send Bob a simple ASCII message.

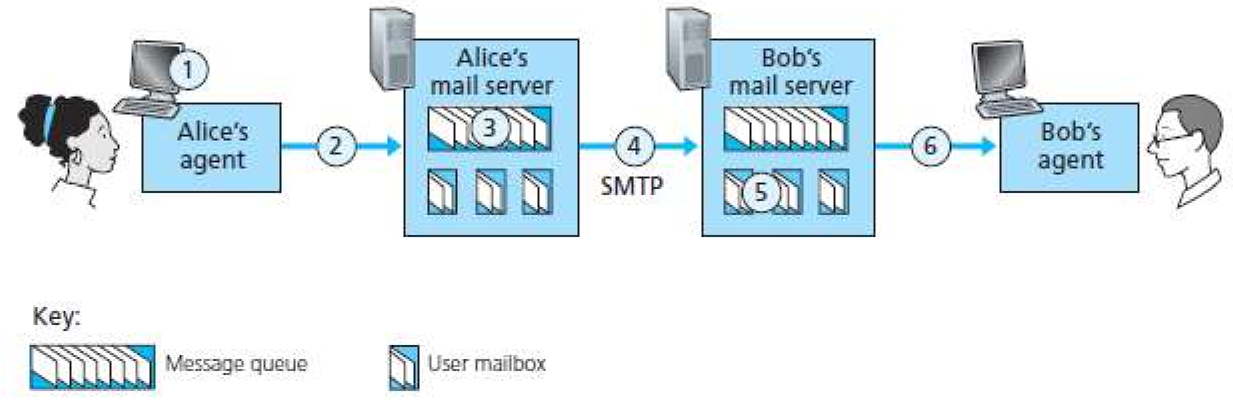


Fig. 13 Alice sends a message to Bob

1. Alice invokes her user agent for e-mail, provides Bob's e-mail address (for example, bob@someschool.edu), composes a message and instructs the user agent to send the message.

2. Alice's user agent sends the message to her mail server, where it is placed in a message queue.
3. The client side of SMTP, running on Alice's mail server, sees the message in the message queue. It opens a TCP connection to an SMTP server, running on Bob's mail server.
4. After some initial SMTP handshaking, the SMTP client sends Alice's message into the TCP connection.
5. At Bob's mail server, the server side of SMTP receives the message. Bob's mail server then places the message in Bob's mailbox.
6. Bob invokes his user agent to read the message at his convenience.

- SMTP does not normally use intermediate mail servers for sending mail, even when the two mail servers are located at opposite ends of the world.
- If Bob's mail server is down, the message remains in Alice's mail server and waits for a new attempt and the message does not get placed in some intermediate mail server.

How SMTP transfers a message from a sending mail server to a receiving mail server

- First, the client SMTP (running on the sending mail server host) has TCP establish a connection to port 25 at the server SMTP (running on the receiving mail server host).
- If the server is down, the client tries again later.
- Once this connection is established, the server and client perform some application-layer handshaking, just as humans often introduce themselves before transferring information from one to another.
- During this SMTP handshaking phase, the SMTP client indicates the e-mail address of the sender (the person who generated the message) and the e-mail address of the recipient.
- Once the SMTP client and server have introduced themselves to each other, the client sends the message.
- SMTP can count on the reliable data transfer service of TCP to get the message to the server without errors.
- The client then repeats this process over the same TCP connection if it has other messages to send to the server; otherwise, it instructs TCP to close the connection.

Comparison of SMTP with HTTP

- HTTP is mainly a pull protocol-someone loads information on a web server and users use HTTP to pull the information from the server. On the other hand, SMTP is primarily a push protocol-the sending mail server pushes the file to receive mail server.
- SMTP requires each message, including the body of each message, to be in seven-bit ASCII format. If the message contains binary data, then the message has to be encoded into seven-bit ASCII format. HTTP does not have this restriction.
- HTTP encapsulate each object of a message in its own response message while SMTP places all of the message's objects into one message.

Mail Access Protocols (POP3 and IMAP)

POP3 (Post Office Protocol)

- POP3 is an extremely simple mail access protocol.
- POP3 begins when the user agent of the client opens a TCP connection to the mail server of the server on port 110.

- With the TCP connection established, POP3 progresses through three phases: authorization, transaction, and update.
- During the first phase, authorization, the user agent sends a username and a password to authenticate the user.
- During the second phase, transaction, the user agent retrieves messages; also during this phase, the user agent can mark messages for deletion, remove deletion marks and obtain mail statistics.
- The third phase, update, occurs after the client has issued the quit command, ending the POP3 session; at this time, the mail server deletes the messages that were marked for deletion.
- POP3 is designed to delete mail on the server as soon as the user has downloaded it.
- However, some implementations allow users or an administrator to specify that mail is saved for some period of time. POP can be thought of as a "store-and-forward" service.

IMAP (Internet Message Access Protocol)

- With POP3 access, once a receiver has downloaded his messages to the local machine, he can create mail folders and move the downloaded messages into the folders.
- A receiver can then delete messages, move messages across folders, and search for messages (by sender name or subject).
- But this paradigm—namely, folders and messages in the local machine—poses a problem for the roaming user, who would prefer to maintain a folder hierarchy on a remote server that can be accessed from any computer.
- This is not possible with POP3—the POP3 protocol does not provide any means for a user to create remote folders and assign messages to folders.
- An IMAP server will associate each message with a folder; when a message first arrives at the server, it is associated with the recipient's INBOX folder.
- The recipient can then move the message into a new, user-created folder, read the message, delete the message, and so on.
- The IMAP protocol provides commands to allow users to create folders and move messages from one folder to another.
- IMAP also provides commands that allow users to search remote folders for messages matching specific criteria.
- Another important feature of IMAP is that it has commands that permit a user agent to obtain components of messages. For example, a user agent can obtain just the message header of a message or just one part of a multipart MIME message.
- This feature is useful when there is a low-bandwidth connection (for example, a slow-speed modem link) between the user agent and its mail server.
- With a low bandwidth connection, the user may not want to download all of the messages in its mailbox, particularly avoiding long messages that might contain, for example, an audio or video clip.

DNS (Domain Name System)

- DNS (Domain Name System) is an internet service that translates domain names into IP addresses.
- Because domain names are alphabetic, they're easier to remember for a human being but The Internet is really based on IP addresses.

- Every time you use a domain name, therefore, a DNS service must translate the domain name into the corresponding IP address. For example, the domain name www.google.com might translate to 198.105.232.4.
- The DNS system is, in fact, its own network. If one DNS server doesn't know how to translate a particular domain name, it asks another one, and so on, until the correct IP address is returned.

Distributed database design is more preferred over a centralized design to implement DNS in the Internet

1. **A single point of failure:** If the DNS server crashes then the entire Internet will not stop.
2. **Traffic volume:** Today the Internet is so huge, with millions of device and users accessing its services from all over the globe at the same time. A Single DNS Server cannot handle the huge global DNS traffic but with a distributed system, this traffic is distributed and reduce overload on the server.
3. **Distant centralized database:** A single DNS server cannot be "close to" all the querying clients. If we put the single DNS server in New York City, then all queries from Australia must travel to the other side of the globe, perhaps over slow and congested links. This can lead to significant delays.
4. **Maintenance:** The single DNS server would have to keep records for all Internet hosts. Not only would this centralized database be huge, but it would have to be updated frequently to account for every new host.

A Distributed, Hierarchical Database

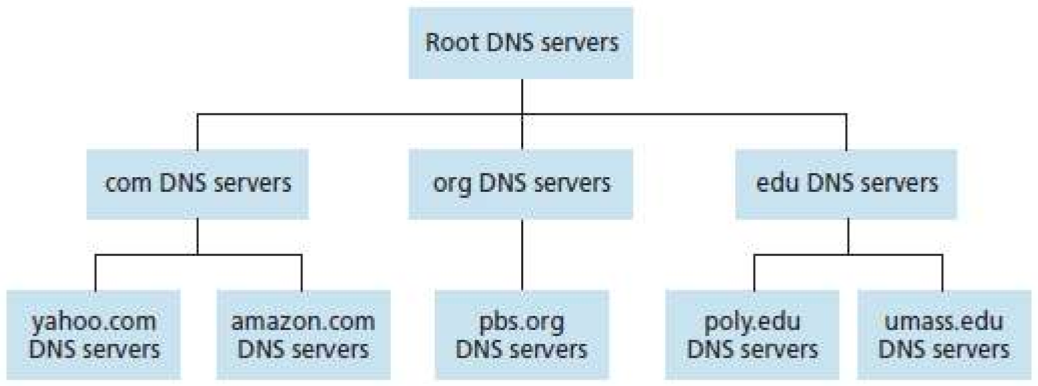


Fig. 14 Portion of the hierarchy of DNS servers

- suppose a DNS client wants to determine the IP address for the hostname www.amazon.com:
    1. The client first contacts one of the root servers, which returns IP addresses for TLD servers for the top-level domain com.
    2. The client then contacts one of these TLD servers, which returns the IP address of an authoritative server for amazon.com.
    3. Finally, the client contacts one of the authoritative servers for amazon.com, which returns the IP address for the hostname www.amazon.com.

Recursive queries and iterative queries

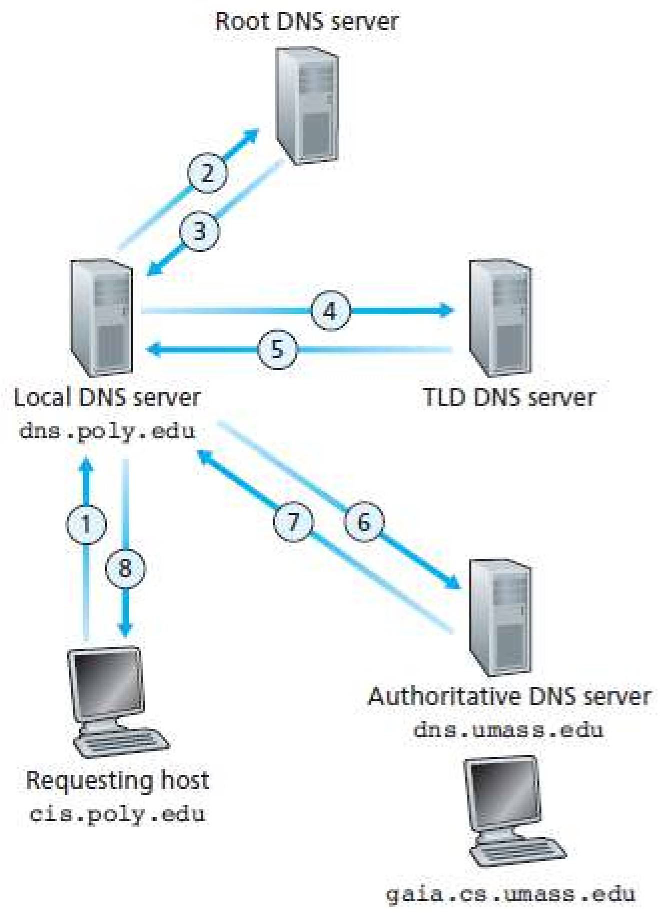


Fig. 15 Interaction of the various DNS servers

- The example shown in the above figure makes use of both recursive queries and iterative queries.
- The query 1 sent from cis.poly.edu to dns.poly.edu is a recursive query because the query asks dns.poly.edu to obtain the mapping on its behalf.
- But the subsequent three queries 2, 4 and 6 are iterative since all of the replies are directly returned to dns.poly.edu.

Transport-Layer services

- A transport-layer protocol provides logical communication between application processes running on different hosts.
- By logical communication, application processes communicate with each other by using the logical communication provided by the transport layer to send messages to each other, free from the worry of the details of the physical infrastructure used to carry these messages.
- Transport-layer protocols are implemented in the end systems but not in network routers.
- On the sending side, the transport layer converts the application-layer messages it receives from a sending application process into transport-layer packets, known as transport-layer segments.
- On the receiving side, the transport layer reassembles segments into messages, passes to application layer.
- A network-layer protocol provides logical communication between hosts.

Relationship between Transport and Network Layers

- The transport layer lies just above the network layer in the protocol stack.
- Whereas a transport-layer protocol provides logical communication between processes running on different hosts, a network-layer protocol provides logical communication between hosts.
- Let's examine this distinction with the aid of a household analogy.
- Consider two houses, one on the East Coast and the other on the West Coast, with each house being home to a dozen kids.
- The kids in the East Coast household are cousins of the kids in the West Coast household.
- The kids in the two households love to write to each other-each kid writes each cousin every week, with each letter delivered by the traditional postal service in a separate envelope.
- Thus, each household sends 144 letters to the other household every week.
- In each of the households there is one kid Ann in the West Coast house and Bill in the East Coast house responsible for mail collection and mail distribution.
- Each week Ann visits all her brothers and sisters, collects the mail, and gives the mail to a postal-service mail person who makes daily visits to the house.
- When letters arrive at the West Coast house, Ann also has the job of distributing the mail to her brothers and sisters. Bill has a similar job on the East Coast.
- In this example, the postal service provides logical communication between the two houses-the postal service moves mail from house to house, not from person to person.
- On the other hand, Ann and Bill provide logical communication among the cousins-Ann and Bill pick up mail from and deliver mail to their brothers and sisters.
- Note that from the cousins' perspective, Ann and Bill are the mail service, even though Ann and Bill are only a part (the end system part) of the end-to-end delivery process.

- This household example serves as a nice analogy for explaining how the transport layer relates to the network layer:

hosts (also called end systems)	houses
processes	cousins
application messages	letters in envelopes
network-layer protocol	postal service (including mail persons)
transport-layer protocol	Ann and Bill

- Continuing with this analogy, observe that Ann and Bill do all their work within their respective homes; they are not involved, for example, in sorting mail in any intermediate mail centre or in moving mail from one mail centre to another.
- Similarly, transport-layer protocols live in the end systems. Within an end system, a transport protocol moves messages from application processes to the network edge (that is, the network layer) and vice versa; but it doesn't have any say about how the messages are moved within the network core.
- In fact, intermediate routers neither act on, nor recognize, any information that the transport layer may have appended to the application messages.

Multiplexing and Demultiplexing

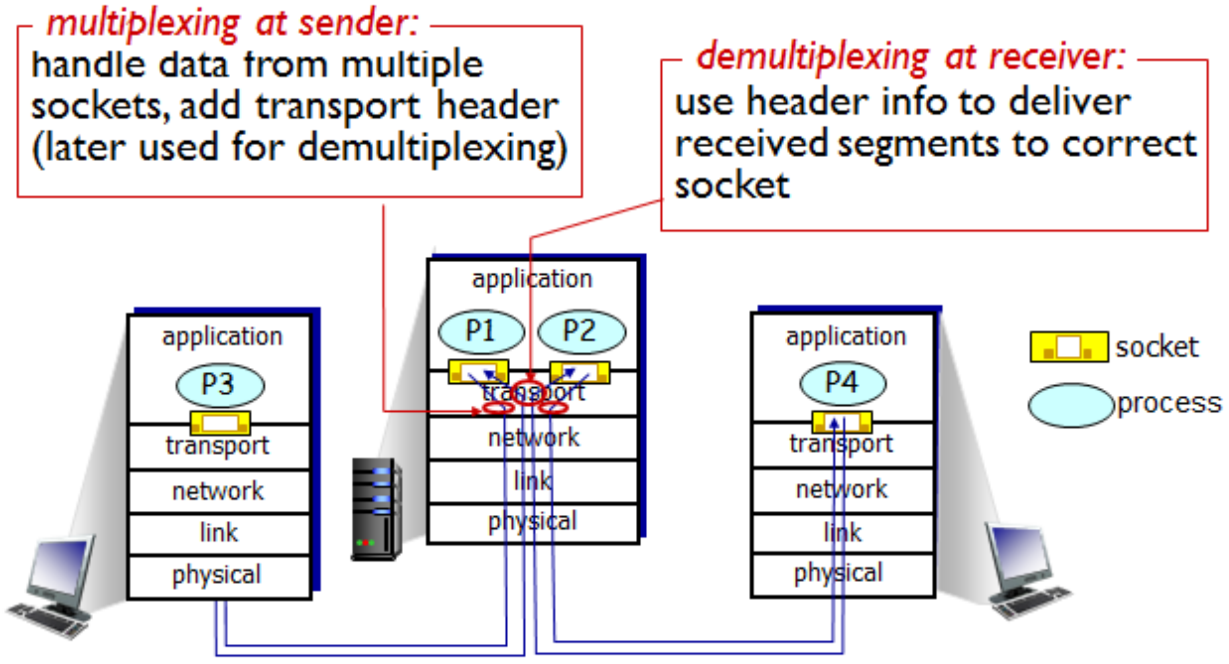

multiplexing at sender:
handle data from multiple sockets, add transport header (later used for demultiplexing)
demultiplexing at receiver:
use header info to deliver received segments to correct socket
application
P1
P2
P3
P4
transport
network
link
physical
socket
process

Fig. 1 Transport-layer multiplexing and demultiplexing

- The job of gathering data chunks at the source host from different sockets, encapsulating each data chunk with header information (that will later be used in demultiplexing) to create segments, and passing the segments to the network layer is called multiplexing.
- At the receiving end, the transport layer examines these fields to identify the receiving socket and then directs the segment to that socket. This job of delivering the data in a transport-layer segment to the correct socket is called demultiplexing.

- Transport layer in the middle host in Figure 1 must demultiplex segments arriving from the network layer below to either process P1 or P2 above; this is done by directing the arriving segment's data to the corresponding process's socket.
- The transport layer in the middle host must also gather outgoing data from these sockets, form transport-layer segments, and pass these segments down to the network layer.

Endpoint Identification

- Sockets must have unique identifiers.
- Each segment must include header fields identifying the socket, these header fields are the source port number field and the destination port number field.
- Each port number is a 16-bit number: 0 to 65535.

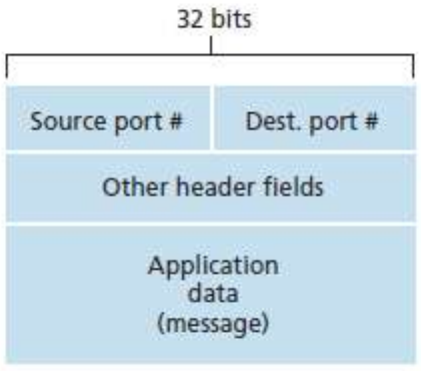


Fig. 2 Source and destination port-number fields in a transport-layer segment

Connectionless Multiplexing and Demultiplexing

- Suppose a process on Host A, with port number 19157, wants to send data to a process with UDP port 46428 on Host B.

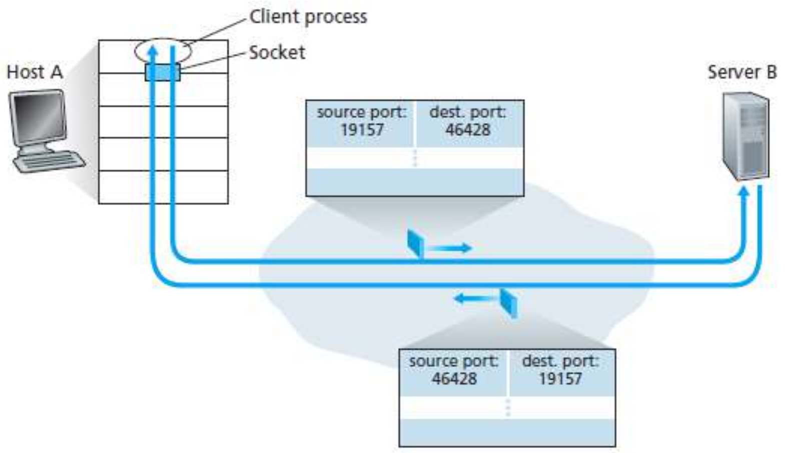


Fig. 3 The inversion of source and destination port numbers

- Transport layer in Host A creates a segment containing source port, destination port, and data and then passes it to the network layer in Host A.
- Transport layer in Host B examines destination port number and delivers segment to socket identified by port 46428.
- Note: a UDP socket is fully identified by a two-tuple consisting of
    1. a destination IP address

2. a destination port number

- Source port number from Host A is used at Host B as "return address".

Connection-Oriented Multiplexing and Demultiplexing

- Each TCP connection has exactly two end-points.
- This means that two arriving TCP segments with different source IP addresses or source port numbers will be directed to two different sockets, even if they have the same destination port number.
- So a TCP socket is identified by four-tuple.

source IP address
2. destination IP address
1. source port #
3. destination port #

- Whereas UDP is identified by only two-tuples
  1. destination IP address
  2. destination port #

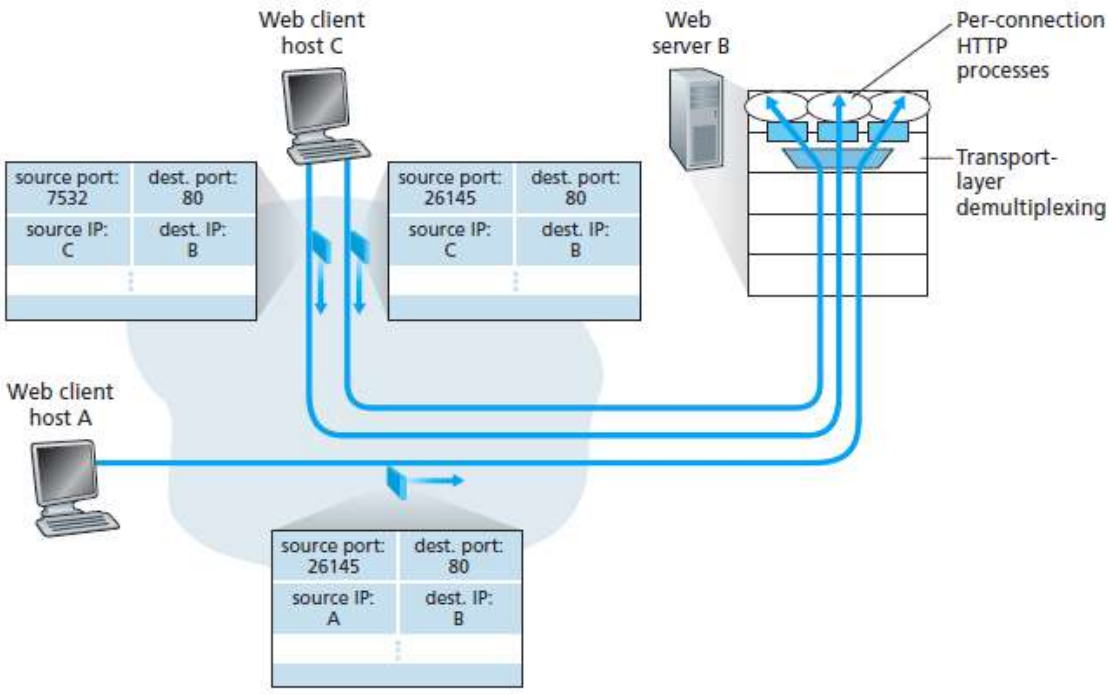


Fig. 4 Two clients, using the same destination port number (80) to communicate with the same Web server application

UDP Segment Structure

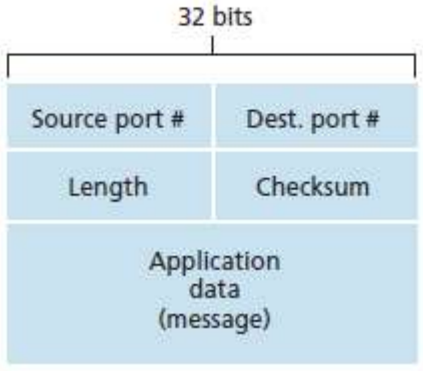
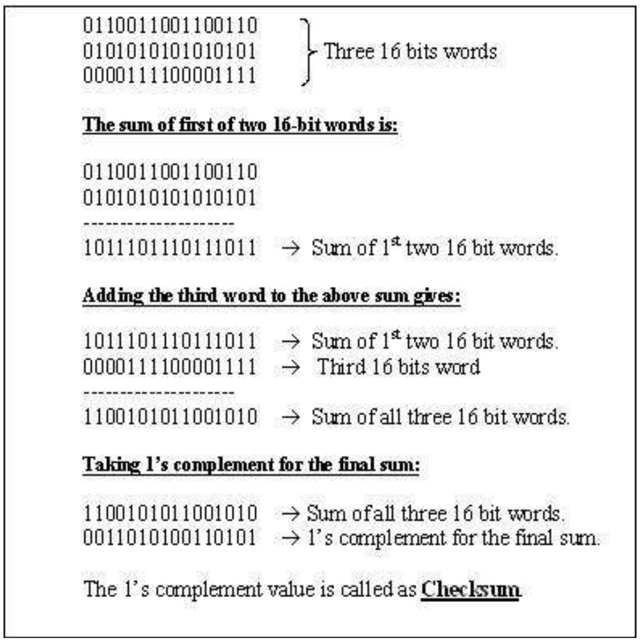


Fig. 5 UDP segment structure

- The port numbers allow the destination host to pass the application data to the correct process running on the destination end system (that is used to perform the demultiplexing function).
- The length field specifies the number of bytes in the UDP segment (header plus data).
- The checksum is used by the receiving host to check whether errors have been introduced into the segment.

How to calculate (find) checksum:

- The UDP checksum is calculated on the sending side by summing all the 16-bit words in the segment, with any overflow being wrapped around and then the 1's complement is performed and the result is added to the checksum field inside the segment.
- At the receiver side, all words inside the packet are added and the checksum is added upon them if the result is 1111 1111 1111 1111 then the segment is valid else the segment has an error.



```
0110011001100110  }
0101010101010101  } Three 16 bits words
0000111100001111  }
```

The sum of first of two 16-bit words is:

```
0110011001100110
0101010101010101
--------------------
1011101110111011    → Sum of 1st two 16 bit words.
```

Adding the third word to the above sum gives:

```
1011101110111011    → Sum of 1st two 16 bit words.
0000111100001111    →  Third 16 bits word
--------------------
1100101011001010    → Sum of all three 16 bit words.
```

Taking 1's complement for the final sum:

```
1100101011001010    → Sum of all three 16 bit words.
0011010100110101    → 1's complement for the final sum.
```

The 1's complement value is called as **Checksum**.

Principles of Reliable Data Transfer

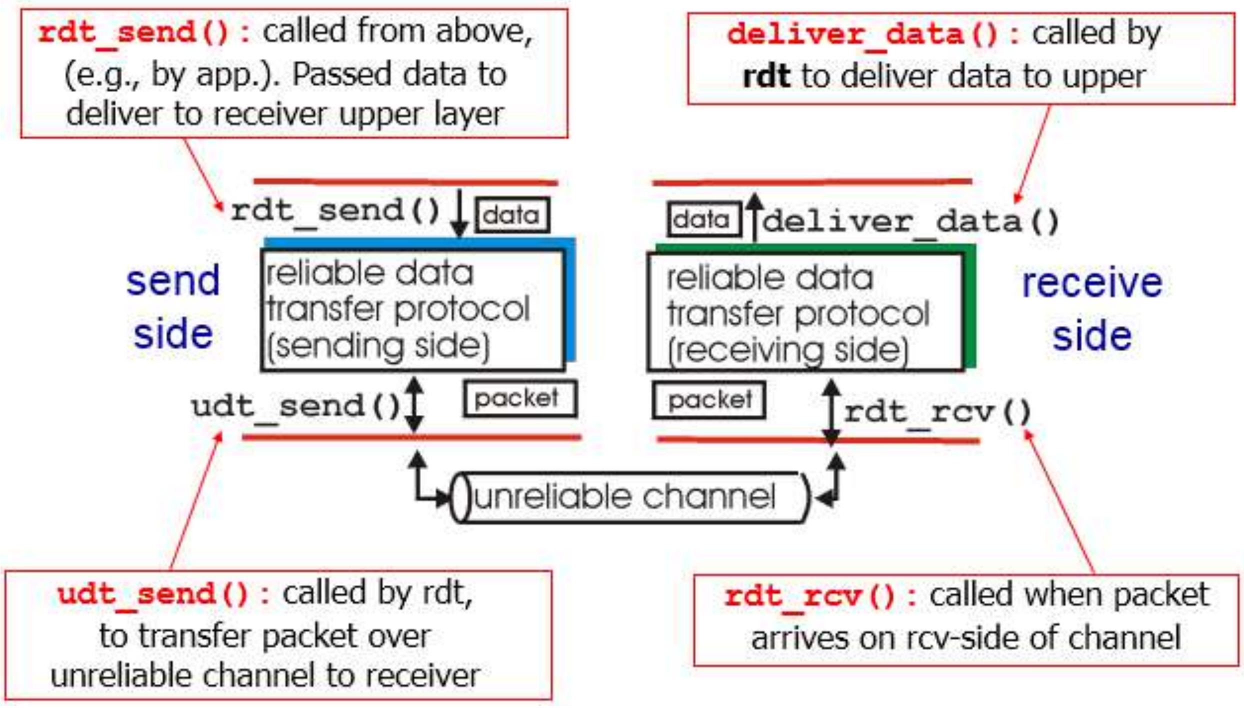

rdt_send() : called from above, (e.g., by app.). Passed data to deliver to receiver upper layer
deliver_data() : called by rdt to deliver data to upper
rdt_send()
data
deliver_data()
send side
reliable data transfer protocol (sending side)
reliable data transfer protocol (receiving side)
receive side
udt_send()
packet
packet
rdt_rcv()
unreliable channel
udt_send() : called by rdt, to transfer packet over unreliable channel to receiver
rdt_rcv() : called when packet arrives on rcv-side of channel

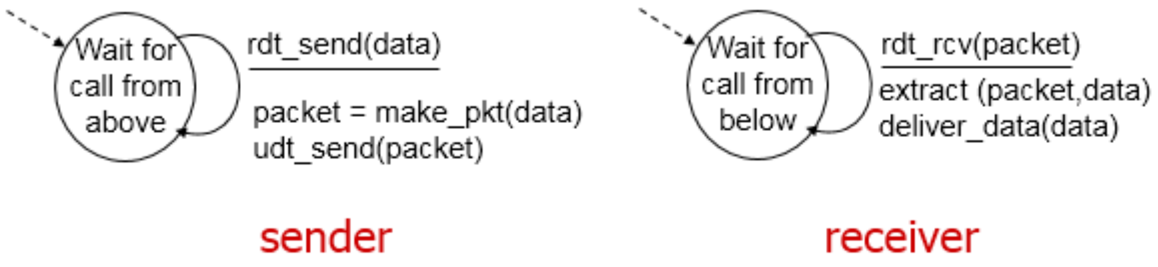
Fig. 7 Reliable data transfer commands

- The sending side of the data transfer protocol will be invoked from above by a call to rdt_send().
- On the receiving side, rdt_rcv() will be called when a packet arrives from the receiving side of the channel.
- When the rdt protocol wants to deliver data to the upper layer, it will do so by calling deliver_data().
- Both the send and receive sides of rdt send packets to the other side by a call to udt_send().

Building a Reliable Data Transfer Protocol

Reliable Data Transfer over a Perfectly Reliable Channel: rdt1.0

- We first consider the simplest case in which the underlying channel is completely reliable.
- The protocol itself, which we will call rdt1.0, is trivial.



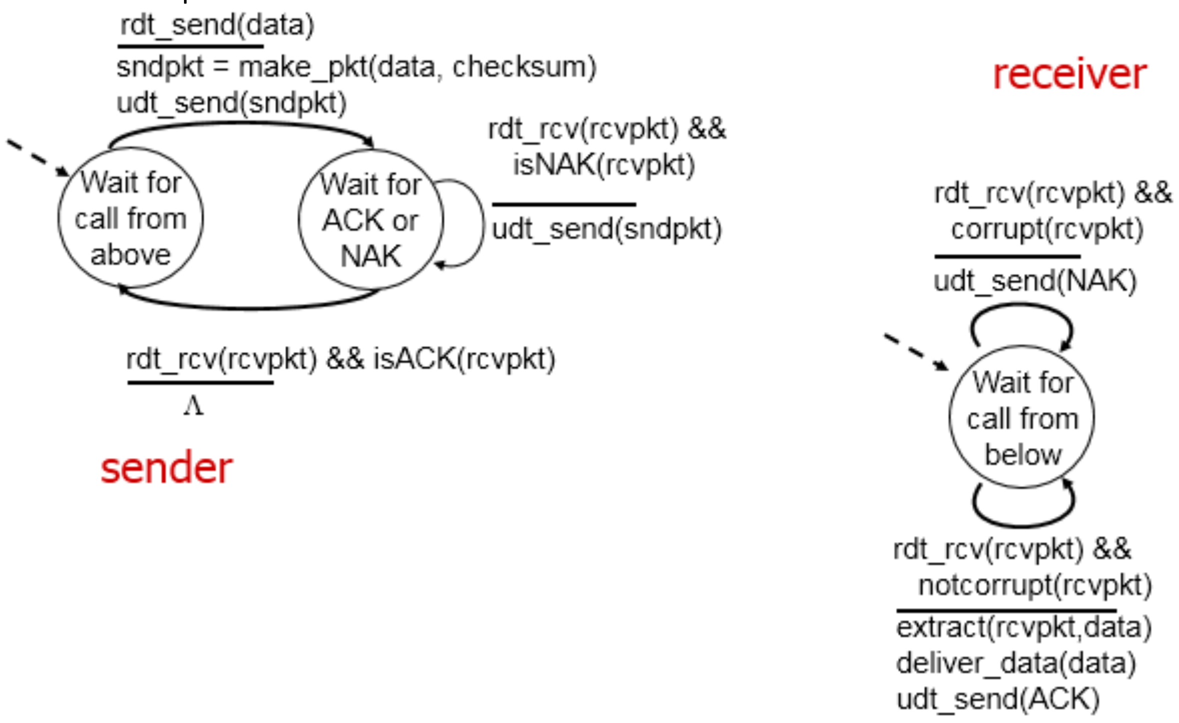
- The sender and receiver FSMs (Finite State Machines) have only one state.

- The arrows in the FSM description indicate the transition of the protocol from one state to another. (Since each FSM has just one state, a transition is necessarily from the one state back to itself).
- The event causing the transition is shown above the horizontal line labelling the transition, and the action(s) taken when the event occurs are shown below the horizontal line.
- The sending side of rdt simply accepts data from the upper-layer via the rdt_send(data) event, puts the data into a packet (via the action make_pkt(packet,data)) and sends the packet into the channel.
- On the receiving side, rdt receives a packet from the underlying channel via the rdt_rcv(packet) event, removes the data from the packet using the action extract(packet,data) and passes the data up to the upper-layer.
- Also, all packet flow is from the sender to receiver - with a perfectly reliable channel there is no need for the receiver side to provide any feedback to the sender since nothing can go wrong.

Reliable Data Transfer over a Channel with Bit Errors: rdt2.0

- A more realistic model of the underlying channel is one in which bits in a packet may be corrupted.
- Such bit errors typically occur in the physical components of a network as a packet is transmitted, propagates, or is buffered.
- We'll continue to assume for the moment that all transmitted packets are received (although their bits may be corrupted) in the order in which they were sent.
- Before developing a protocol for reliably communicating over such a channel, first consider how people might deal with such a situation.
- Consider how you yourself might dictate a long message over the phone. In a typical scenario, the message taker might say ``OK'' after each sentence has been heard, understood, and recorded. If the message taker hears a garbled sentence, you're asked to repeat the garbled sentence. This message dictation protocol uses both positive acknowledgements (``OK'') and negative acknowledgements (``Please repeat that''). These control messages allow the receiver to let the sender know what has been received correctly, and what has been received in error and thus requires repeating. In a computer network setting, reliable data transfer protocols based on such retransmission are known ARQ (Automatic Repeat reQuest) protocols.
- Fundamentally, two additional protocol capabilities are required in ARQ protocols to handle the presence of bit errors:
- **Error detection:** First, a mechanism is needed to allow the receiver to detect when bit errors have occurred. UDP transport protocol uses the Internet checksum field for exactly this purpose. Error detection and correction techniques allow the receiver to detect, and possibly correct packet bit errors.
- **Receiver feedback:** Since the sender and receiver are typically executing on different end systems, possibly separated by thousands of miles, the only way for the sender to learn of the receiver's view of the world (in this case, whether or not a packet was received correctly) is for the receiver to provide explicit feedback to the sender. The positive (ACK) and negative acknowledgement (NAK) replies in the message dictation

scenario are an example of such feedback. Our rdt2.0 protocol will similarly send ACK and NAK packets back from the receiver to the sender.



- The send side of rdt2.0 has two states.
- In one state, the send-side protocol is waiting for data to be passed down from the upper layer.
- In the other state, the sender protocol is waiting for an ACK or a NAK packet from the receiver.
- If an ACK packet is received (the notation rdt_rcv(rcvpkt) && isACK(rcvpkt), the sender knows the most recently transmitted packet has been received correctly and thus the protocol returns to the state of waiting for data from the upper layer.
- If a NAK is received, the protocol retransmits the last packet and waits for an ACK or NAK to be returned by the receiver in response to the retransmitted data packet.
- It is important to note that when the receiver is in the wait-for-ACK-or-NAK state, it cannot get more data from the upper layer; that will only happen after the sender receives an ACK and leaves this state.
- Thus, the sender will not send a new piece of data until it is sure that the receiver has correctly received the current packet.
- Because of this behaviour, protocols such as rdt2.0 are known as stop-and-wait protocols.
- The receiver-side FSM for rdt2.0 still has a single state.
- On packet arrival, the receiver replies with either an ACK or a NAK, depending on whether or not the received packet is corrupted.
- In above figure the notation rdt_rcv(rcvpkt) && corrupt(rcvpkt) corresponds to the event where a packet is received and is found to be in error.

rdt2.1: sender, handles garbled ACK/NAKs

- Protocol rdt2.0 may look as if it works but unfortunately has a fatal flaw.

- In particular, we haven't accounted for the possibility that the ACK or NAK packet could be corrupted.
- Minimally, we will need to add checksum bits to ACK/NAK packets in order to detect such errors.
- The more difficult question is how the protocol should recover from errors in ACK or NAK packets.
- The difficulty here is that if an ACK or NAK is corrupted, the sender has no way of knowing whether or not the receiver has correctly received the last piece of transmitted data.
- An approach is for the sender to simply resend the current data packet when it receives a garbled ACK or NAK packet.
- This, however, introduces duplicate packets into the sender-to-receiver channel.
- The fundamental difficulty with duplicate packets is that the receiver doesn't know whether the ACK or NAK it last sent was received correctly at the sender.
- Thus, it cannot know a priori whether an arriving packet contains new data or is a duplicate.
- A simple solution to this new problem is to add a new field to the data packet and have the sender number its data packets by putting a sequence number into this field.
- The receiver then need only check this sequence number to determine whether or not the received packet is a retransmission.
- For this simple case of a stop-and-wait protocol, a 1-bit sequence number will suffice, since it will allow the receiver to know whether the sender is resending the previously transmitted packet (the sequence number of the received packet has the same sequence number as the most recently received packet) or a new packet (the sequence number changes).
- Since we are currently assuming a channel that does not lose packets, ACK and NAK packets do not themselves need to indicate the sequence number of the packet they are ACKing or NAKing, since the sender knows that a received ACK or NAK packet (whether garbled or not) was generated in response to its most recently transmitted data packet.

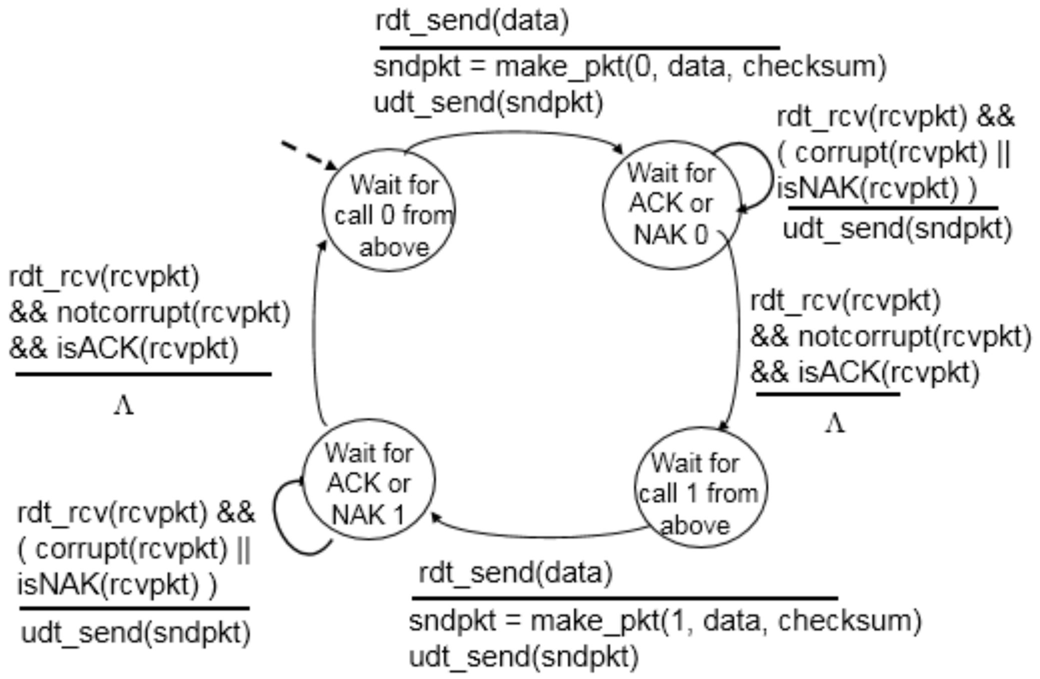

rdt_send(data)
sndpkt = make_pkt(0, data, checksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isNAK(rcvpkt))
udt_send(sndpkt)
Wait for call 0 from above
Wait for ACK or NAK 0
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt)
Λ
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt)
Λ
Wait for ACK or NAK 1
Wait for call 1 from above
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isNAK(rcvpkt))
udt_send(sndpkt)
rdt_send(data)
sndpkt = make_pkt(1, data, checksum)
udt_send(sndpkt)

Fig 10 rdt 2.1 Sender

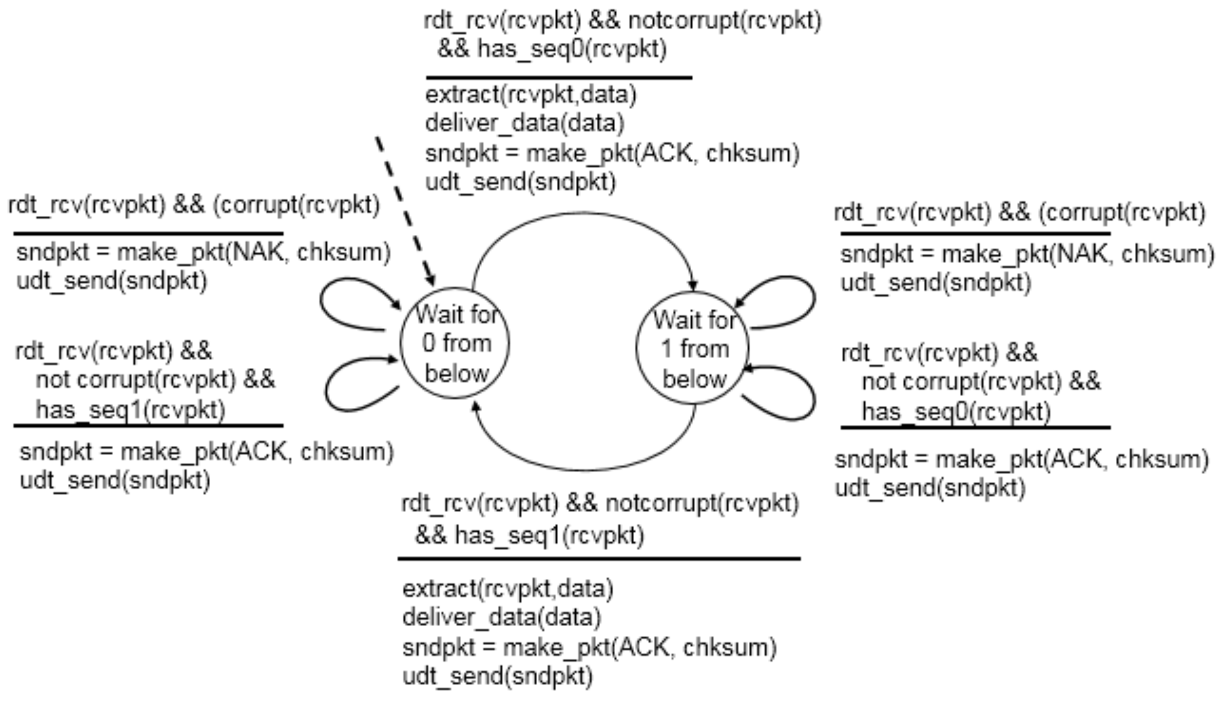

rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && has_seq0(rcvpkt)
extract(rcvpkt,data)
deliver_data(data)
sndpkt = make_pkt(ACK, chksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && (corrupt(rcvpkt)
sndpkt = make_pkt(NAK, chksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && not corrupt(rcvpkt) && has_seq1(rcvpkt)
sndpkt = make_pkt(ACK, chksum)
udt_send(sndpkt)
Wait for 0 from below
Wait for 1 from below
rdt_rcv(rcvpkt) && (corrupt(rcvpkt)
sndpkt = make_pkt(NAK, chksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && not corrupt(rcvpkt) && has_seq0(rcvpkt)
sndpkt = make_pkt(ACK, chksum)
udt_send(sndpkt)
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && has_seq1(rcvpkt)
extract(rcvpkt,data)
deliver_data(data)
sndpkt = make_pkt(ACK, chksum)
udt_send(sndpkt)

Fig 11 rdt 2.1 Receiver

- The rdt2.1 sender and receiver FSM's each now have twice as many states as rdt2.0.
- This is because the protocol state must now reflect whether the packet currently being sent (by the sender) or expected (at the receiver) should have a sequence number of 0 or 1.
- Note that the actions in those states where a 0-numbered packet is being sent or expected are mirror images of those where a 1-numbered packet is being sent or

expected; the only differences have to do with the handling of the sequence number.

- Protocol rdt2.1 uses both positive and negative acknowledgements from the receiver to the sender.
- A negative acknowledgement is sent whenever a corrupted packet, or an out of order packet, is received.
- We can accomplish the same effect as a NAK if instead of sending a NAK, we instead send an ACK for the last correctly received packet.
- A sender that receives two ACKs for the same packet (i.e., receives duplicate ACKs) knows that the receiver did not correctly receive the packet following the packet that is being ACKed twice. Our NAK-free reliable data transfer protocol for a channel with bit errors is rdt2.2.

Reliable Data Transfer over a Lossy Channel with Bit Errors: rdt3.0

- Suppose now that in addition to corrupting bits, the underlying channel can lose packets as well.
- Two additional concerns must now be addressed by the protocol: how to detect packet loss and what to do when this occurs.
- The use of checksumming, sequence numbers, ACK packets, and retransmissions - the techniques already developed in rdt 2.2 - will allow us to answer the latter concern. Handling the first concern will require adding a new protocol mechanism.
- There are many possible approaches towards dealing with packet loss.
- Suppose that the sender transmits a data packet and either that packet, or the receiver's ACK of that packet, gets lost.
- In either case, no reply is forthcoming at the sender from the receiver.
- If the sender is willing to wait long enough so that it is certain that a packet has been lost, it can simply retransmit the data packet.
- But how long must the sender wait to be certain that something has been lost? It must clearly wait at least as long as a round trip delay between the sender and receiver (which may include buffering at intermediate routers or gateways) plus whatever amount of time is needed to process a packet at the receiver.
- If an ACK is not received within this time, the packet is retransmitted.
- Note that if a packet experiences a particularly large delay, the sender may retransmit the packet even though neither the data packet nor its ACK have been lost.
- This introduces the possibility of duplicate data packets in the sender-to-receiver channel.
- Happily, protocol rdt2.2 already has enough functionality (i.e., sequence numbers) to handle the case of duplicate packets.
- From the sender's viewpoint, retransmission is a solution. The sender does not know whether a data packet was lost, an ACK was lost, or if the packet or ACK was simply overly delayed.
- In all cases, the action is the same: retransmit. In order to implement a time-based retransmission mechanism, a countdown timer will be needed that can interrupt the sender after a given amount of timer has expired.

- The sender will thus need to be able to (i) start the timer each time a packet (either a first time packet, or a retransmission) is sent, (ii) respond to a timer interrupt (taking appropriate actions), and (iii) stop the timer.
- The existence of sender-generated duplicate packets and packet (data, ACK) loss also complicates the sender's processing of any ACK packet it receives.
- If an ACK is received, how is the sender to know if it was sent by the receiver in response to its (sender's) own most recently transmitted packet, or is a delayed ACK sent in response to an earlier transmission of a different data packet? The solution to this dilemma is to augment the ACK packet with an acknowledgement field. When the receiver generates an ACK, it will copy the sequence number of the data packet being ACK'ed into this acknowledgement field. By examining the contents of the acknowledgment field, the sender can determine the sequence number of the packet being positively acknowledged.

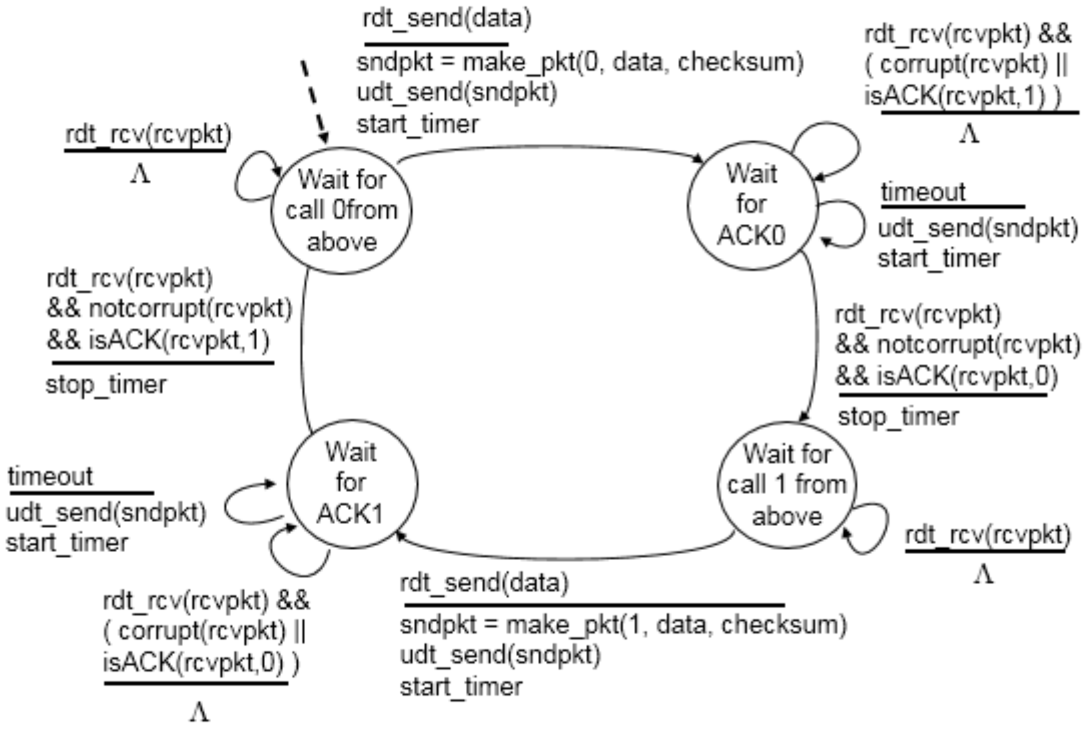

rdt_send(data)
sndpkt = make_pkt(0, data, checksum)
udt_send(sndpkt)
start_timer
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isACK(rcvpkt,1))
Λ
rdt_rcv(rcvpkt)
Λ
Wait for call 0from above
Wait for ACK0
timeout
udt_send(sndpkt)
start_timer
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,1)
stop_timer
rdt_rcv(rcvpkt) && notcorrupt(rcvpkt) && isACK(rcvpkt,0)
stop_timer
timeout
udt_send(sndpkt)
start_timer
Wait for ACK1
Wait for call 1 from above
rdt_rcv(rcvpkt)
Λ
rdt_rcv(rcvpkt) && (corrupt(rcvpkt) || isACK(rcvpkt,0))
Λ
rdt_send(data)
sndpkt = make_pkt(1, data, checksum)
udt_send(sndpkt)
start_timer

Fig 12 rdt 3.0 Sender

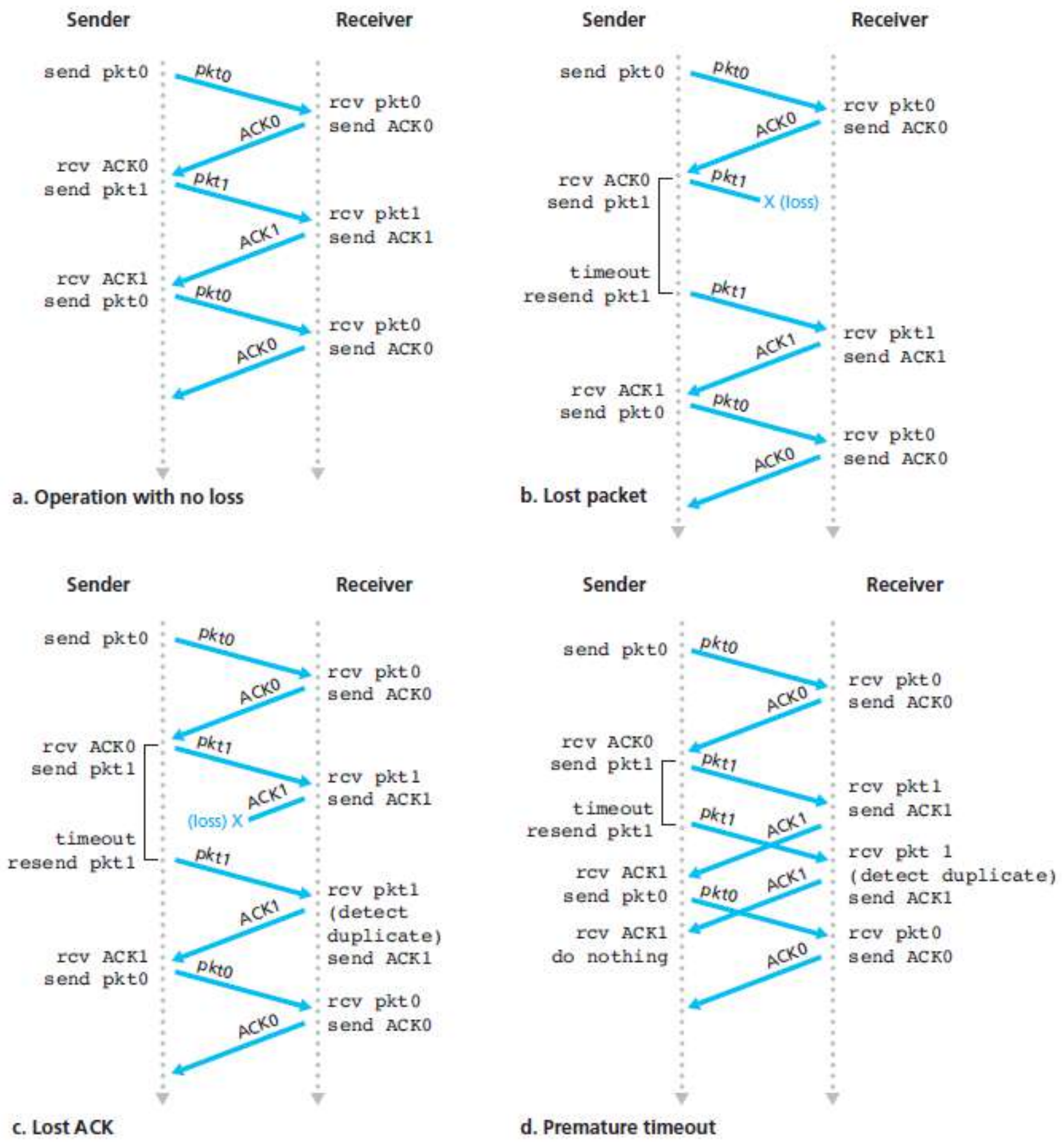
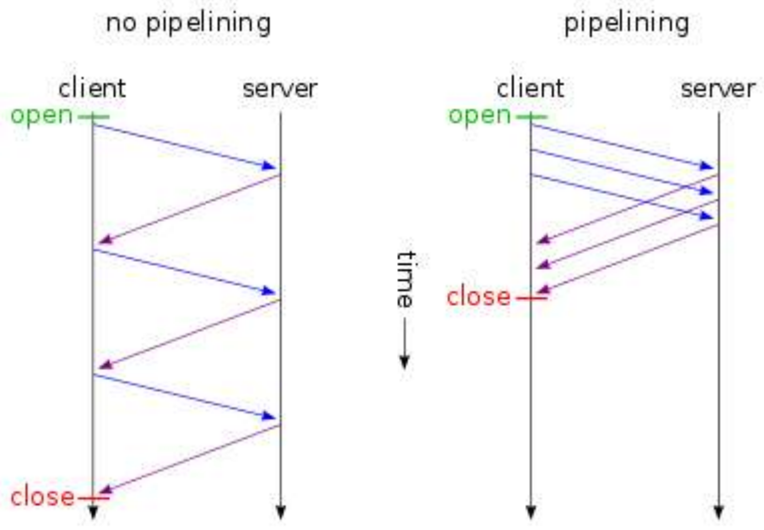


Fig. 13 Operation of rdt3.0, the alternating-bit protocol

Protocol pipelining

- Protocol pipelining is a technique in which multiple requests are written out to a single socket without waiting for the corresponding responses (acknowledged).
- Pipelining can be used in various application layer network protocols, like HTTP/1.1, SMTP and FTP.
- Range of sequence numbers must be increased.
- Data or Packet should be buffered at sender and/or receiver.



- Two generic forms of pipelined protocols are
    1. Go-Back-N
    2. Selective repeat

Go-Back-N

- Go-Back-N ARQ is a specific instance of the automatic repeat request (ARQ) protocol, in which the sending process continues to send a number of frames specified by a window size even without receiving an acknowledgement (ACK) packet from the receiver.
- The receiver process keeps track of the sequence number of the next frame it expects to receive, and sends that number with every ACK it sends.
- The receiver will discard any frame that does not have the exact sequence number it expects (either a duplicate frame it already acknowledged or an out-of-order frame it expects to receive later) and will resend an ACK for the last correct in-order frame.
- Once the sender has sent all of the frames in its window, it will detect that all of the frames since the first lost frame are outstanding, and will go back to the sequence number of the last ACK it received from the receiver process and fill its window starting with that frame and continue the process over again.
- Go-Back-N ARQ is a more efficient use of a connection than Stop-and-wait ARQ, since unlike waiting for an acknowledgement for each packet; the connection is still being utilized as packets are being sent.
- However, this method also results in sending frames multiple times – if any frame was lost or damaged or the ACK acknowledging them was lost or damaged then that frame and all following frames in the window (even if they were received without error) will be re-sent. To avoid this, Selective Repeat ARQ can be used.

How does Go-Back-N ARQ protocol works:

- Consider a scenario given in fig 14 there are four frames 0,1,2,3 respectively. Now as sending window size is 3 it will send frame 0, 1 and 2 at a time.

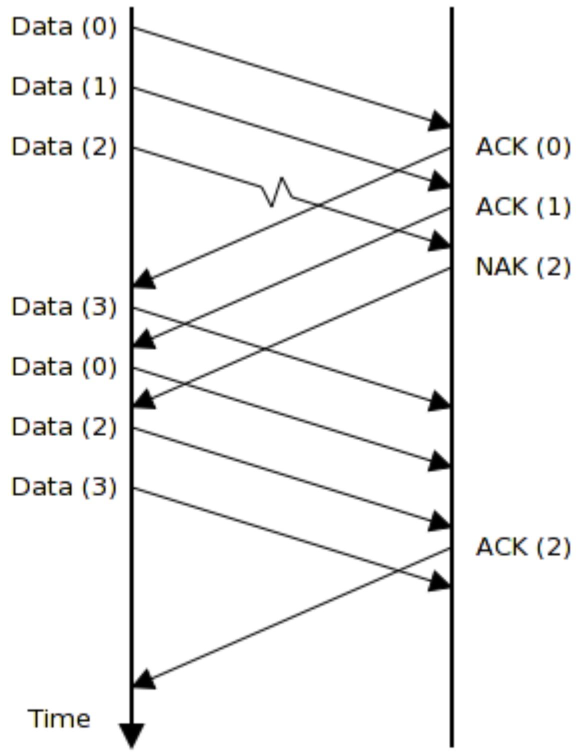


Fig. 14 Go-Back-N Protocol

- At receiver side frame 0 arrives it sends ACK for that.
- Now at sender side window shift at frame 3 and it sends frame no 3.
- At receiver side frame 1 arrives it sends ACK for that.
- Now at sender side window shift at frame 4 and it sends frame no 4.
- Now receiver detects frame 2 is missing or lost. It will send NAK for that.
- Sender receives NAK at this point sending window is pointing towards frame 2, 3 and 4 so send will resend frame 2, 3 and 4.

Selective repeat

- Selective Repeat attempts to retransmit only those packets that are actually lost due to errors.
- Receiver must be able to accept packets out of order
- Since receiver must release packets to higher layer in order, the receiver must be able to buffer some packets
- The receiver acknowledges every good packet, packets that are not ACKed before a time-out are assumed lost or in error.
- Notice that this approach must be used to be sure that every packet is eventually received.
- An explicit NAK (selective reject) can request retransmission of just one packet.
- This approach can speed up the retransmission but is not strictly needed.

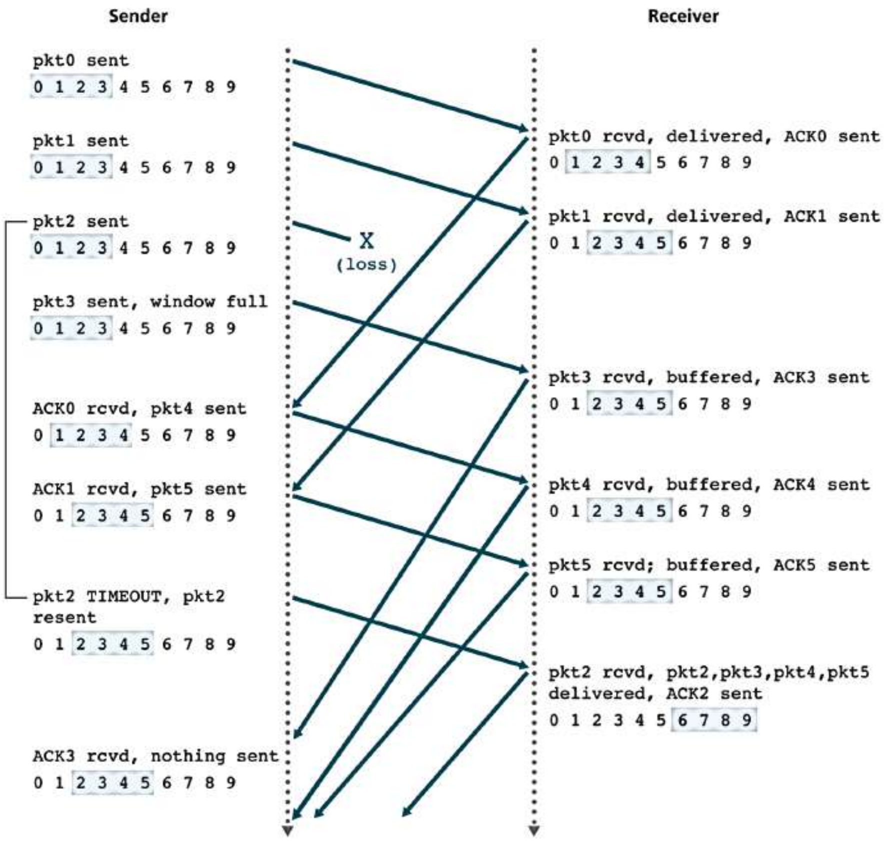

Sender
Receiver
pkt0 sent
pkt1 sent
pkt2 sent
X (loss)
pkt3 sent, window full
ACK0 rcvd, pkt4 sent
ACK1 rcvd, pkt5 sent
pkt2 TIMEOUT, pkt2 resent
ACK3 rcvd, nothing sent
pkt0 rcvd, delivered, ACK0 sent
pkt1 rcvd, delivered, ACK1 sent
pkt3 rcvd, buffered, ACK3 sent
pkt4 rcvd, buffered, ACK4 sent
pkt5 rcvd; buffered, ACK5 sent
pkt2 rcvd, pkt2,pkt3,pkt4,pkt5 delivered, ACK2 sent

Fig. 15 Selective Repeat

TCP segment structure

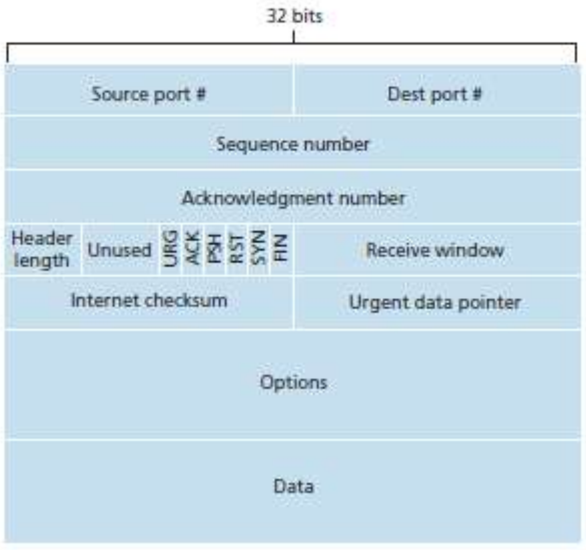


Fig. 16 TCP segment structure

- The unit of transmission in TCP is called segments.
- The header includes source and destination port numbers, which are used for multiplexing/demultiplexing data from/to upper-layer applications.

- The 32-bit sequence number field and the 32-bit acknowledgment number field are used by the TCP sender and receiver in implementing a reliable data transfer service.
- The sequence number for a segment is the byte-stream number of the first byte in the segment.
- The acknowledgment number is the sequence number of the next byte a Host is expecting from another Host.
- The 4-bit header length field specifies the length of the TCP header in 32-bit words. The TCP header can be of variable length due to the TCP options field.
- The 16-bit receive window field is used for flow control. It is used to indicate the number of bytes that a receiver is willing to accept.
- The 16-bit checksum field is used for error checking of the header and data.
- Unused 6 bits are reserved for future use and should be sent to zero.
- Urgent Pointer is used in combining with the URG control bit for priority data transfer. This field contains the sequence number of the last byte of urgent data.
- **Data**: The bytes of data being sent in the segment.
- **URG (1 bit)**: indicates that the Urgent pointer field is significant.
- **ACK (1 bit)**: indicates that the Acknowledgment field is significant.
- **PSH (1 bit)**: Push function. Asks to push the buffered data to the receiving application.
- **RST (1 bit)**: Reset the connection.
- **SYN (1 bit)**: Synchronize sequence numbers. Only the first packet sent from each end should have this flag set. Some other flags and fields change meaning based on this flag, and some are only valid for when it is set, and others when it is clear.
- **FIN (1 bit)**: No more data from sender.

Flow Control

- In data communications, flow control is the process of managing the rate of data transmission between two nodes to prevent a fast sender from overwhelming a slow receiver.
- It prevent receiver from becoming overloaded.
- Receiver advertises a window rwnd with each acknowledgement
- Window
  - closed (by sender) when data is sent and ack'd
  - opened (by receiver) when data is read
- The size of this window can be the performance limit (e.g. on a LAN).

Congestion Control

- When a connection is established, a suitable window size has to be chosen.
- The receiver can specify a window based on its buffer size.
- If the sender sticks to this window size, problems will not occur due to buffer overflow at the receiving end, but they may still occur due to internal congestion within the network.
- In Figure 17 (a), we see a thick pipe leading to a small-capacity receiver.
- As long as the sender does not send more water than the bucket can contain, no water will be lost.

- In Figure 17 (b), the limiting factor is not the bucket capacity, but the internal carrying capacity of the network.

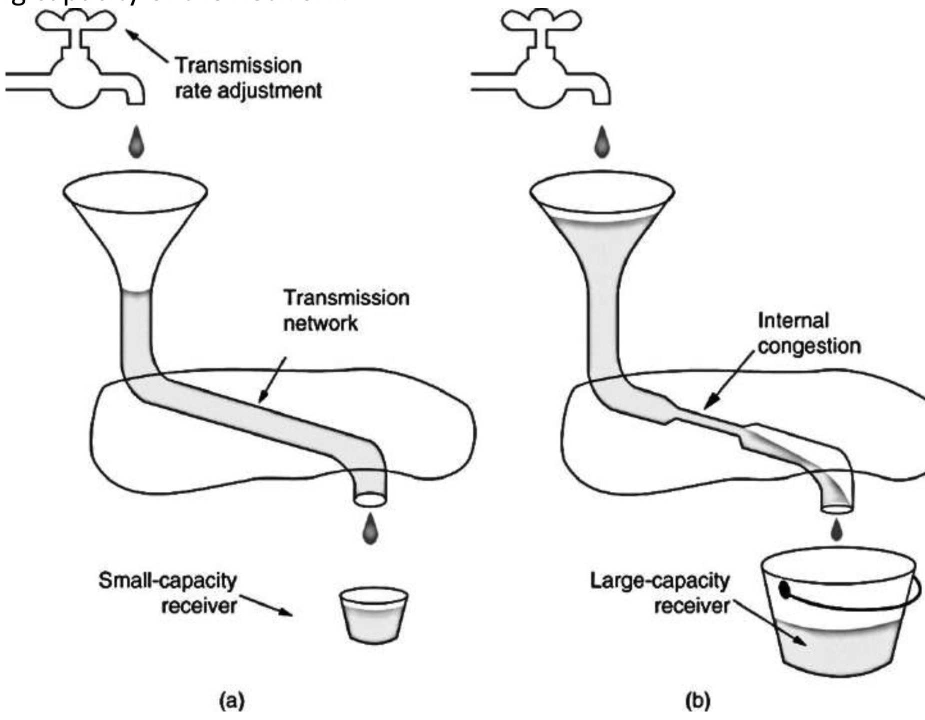


Fig. 17 TCP segment structure

- Figure 17: (a) a fast network feeding a low capacity receiver. Figure 17: (b) A slow network feeding a high-capacity receiver.
- If too much water comes in too fast, it will back up and some will be lost (in this case by overflowing the funnel).
- Each sender maintains two windows: the window the receiver has granted and a second window, the congestion window.
- Each reflects the number of bytes the sender may transmit.
- The number of bytes that may be sent is the minimum of the two windows.
- Thus, the effective window is the minimum of what the sender thinks is all right and what the receiver thinks is all right.
- When a connection is established, the sender initializes the congestion window to the size of the maximum segment in use on the connection.
- It then sends one maximum segment.
- If this segment is acknowledged before the timer goes off, it adds one segment's worth of bytes to the congestion window to make it two maximum size segments and sends two segments.
- As each of these segments is acknowledged, the congestion window is increased by one maximum segment size.
- When the congestion window is n segments, if all n are acknowledged on time, the congestion window is increased by the byte count corresponding to n segments.

TCP Slow Start

- Slow-start is part of the congestion control strategy used by TCP, the data transmission protocol used by many Internet applications.

- Slow-start is used in conjunction with other algorithms to avoid sending more data than the network is capable of transmitting to avoid causing network congestion.

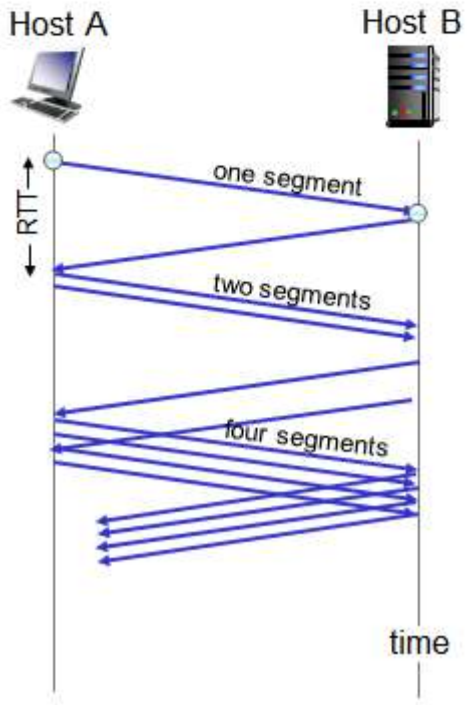


Fig. 18 TCP Slow start

- Slow-start begins initially with a congestion window Size (cwnd) of 1, 2 or 10.
- The value of the Congestion Window will be increased with each acknowledgement (ACK) received, effectively doubling the window size each round trip time ("although it is not exactly exponential because the receiver may delay its ACKs, typically sending one ACK for every two segments that it receives).
- The transmission rate will be increased with slow-start algorithm until either a loss is detected, or the receiver's advertised window (rwnd) is the limiting factor, or the slow start threshold (ssthresh) is reached.
- If a loss event occurs, TCP assumes that it is due to network congestion and takes steps to reduce the offered load on the network.
- Once ssthresh is reached, TCP changes from slow-start algorithm to the linear growth (congestion avoidance) algorithm. At this point, the window is increased by 1 segment for each RTT.

Routing and Forwarding

- Figure 1 shows a simple network with two hosts, H1 and H2, and several routers on the path between H1 and H2.
- Suppose that H1 is sending information to H2, and consider the role of the network layer in these hosts and in the intervening routers.
- The network layer in H1 takes segments from the transport layer in H1, encapsulates each segment into a datagram (that is, a network-layer packet), and then sends the datagrams to its nearby router, R1.
- At the receiving host, H2, the network layer receives the datagrams from its nearby router R2, extracts the transport-layer segments, and delivers the segments up to the transport layer at H2.
- The primary role of the routers is to forward datagrams from input links to output links.

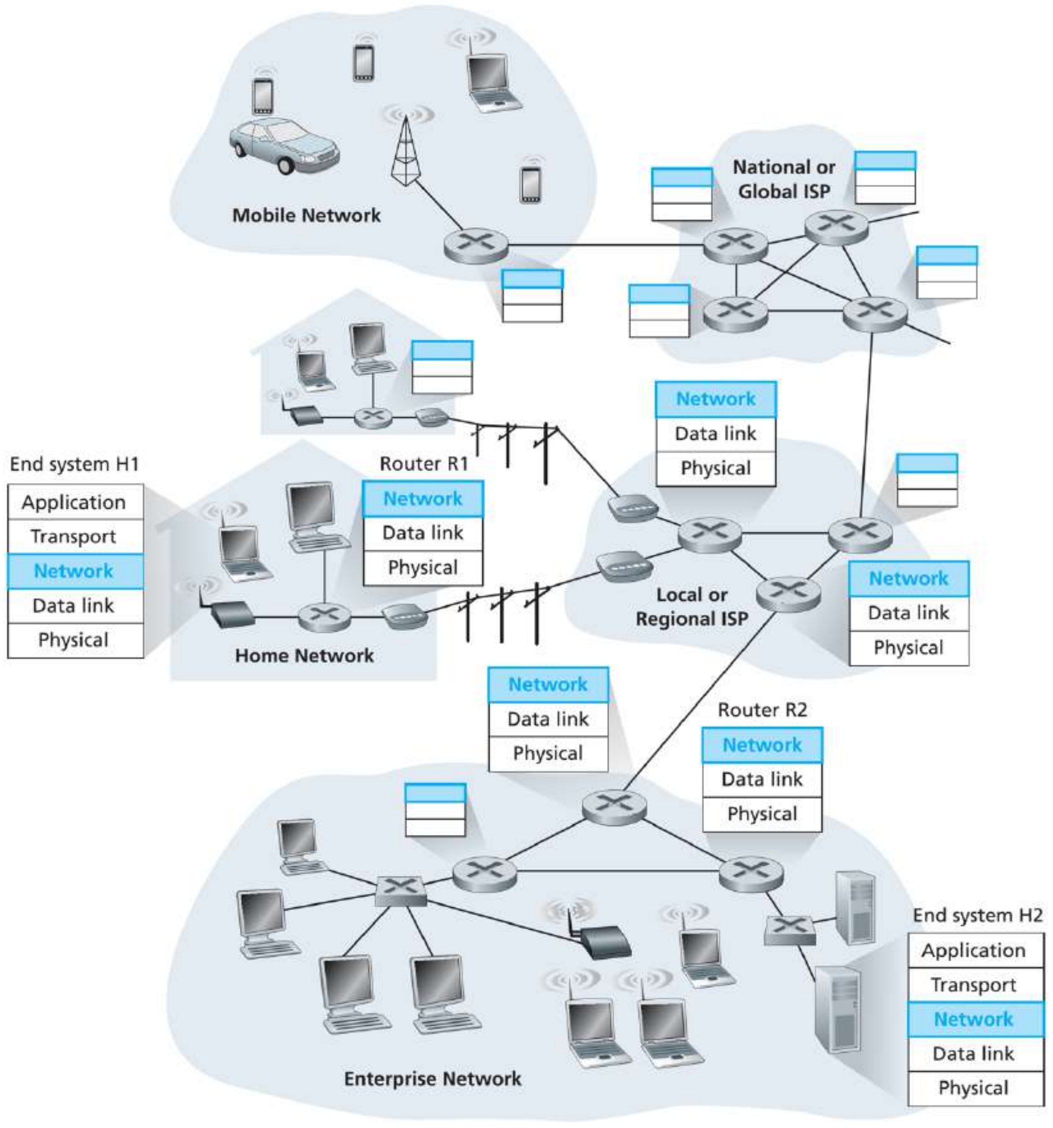


Fig. 1 Network Layer

- The role of the network layer is thus deceptively simple—to move packets from a sending host to a receiving host. To do so, two important network-layer functions can be identified:
  - **Forwarding**: When a packet arrives at a router's input link, the router must move the packet to the appropriate output link.
  - For example, a packet arriving from Host H1 to Router R1 must be forwarded to the next router on a path to H2.
  - **Routing:** Routing is the process of selecting best paths in a network.
  - The network layer must determine the route or path taken by packets as they flow from a sender to a receiver.
  - The algorithms that calculate these paths are referred to as routing algorithms. A routing algorithm would determine, for example, the path along which packets flow from H1 to H2.
- Every router has a **forwarding table**. A router forwards a packet by examining the value of a field in the arriving packet's header, and then using this header value to index into the router's forwarding table.
- The value stored in the forwarding table entry for that header indicates the router's outgoing link interface to which that packet is to be forwarded.
- Depending on the network-layer protocol, the header value could be the destination address of the packet or an indication of the connection to which the packet belongs.

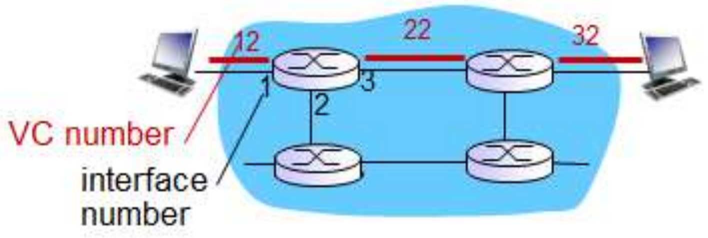
Network service model

- Services provided by network layer for individual datagrams
  1. **Guaranteed delivery:** This service guarantees that the packet will eventually arrive at its destination.
  2. **Guaranteed delivery with bounded delay:** This service not only guarantees delivery of the packet, but delivery within a specified host-to-host delay bound (for example, within 100 msec).
- Services provided by network layer for a flow of datagrams
  3. **In-order packet delivery:** This service guarantees that packets arrive at the destination in the order that they were sent.
  4. **Guaranteed minimal bandwidth:** This network-layer service emulates the behaviour of a transmission link of a specified bit rate (for example, 1 Mbps) between sending and receiving hosts. As long as the sending host transmits bits at a rate below the specified bit rate, then no packet is lost.
  5. **Guaranteed maximum jitter:** This service guarantees that the amount of time between the transmission of two successive packets at the sender is equal to the amount of time between their receipt at the.
  6. **Security services:** Using a secret session key known only by a source and destination host, the network layer in the source host could encrypt the payloads of all datagrams being sent to the destination host. The network layer in the destination host would then be responsible for decrypting the payloads. With such a service, confidentiality would be provided to all transport-layer segments (TCP and UDP) between the source and destination hosts.

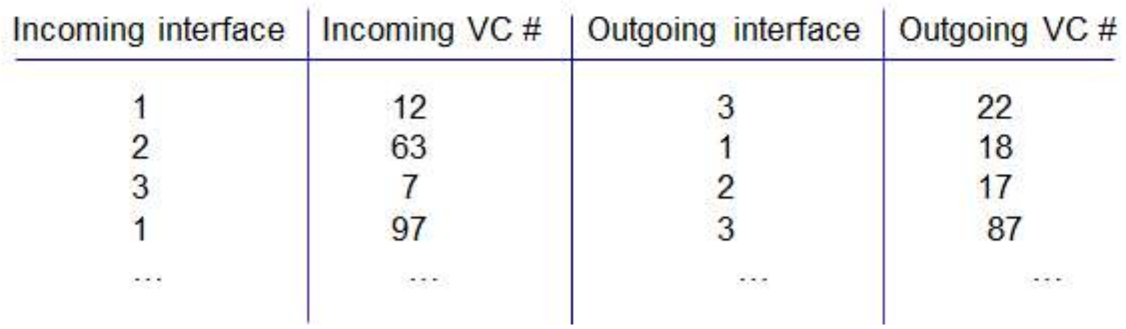
Virtual and Datagram networks

Virtual Circuit Switching (Connection Oriented Service)

- A VC consists of
  1. a path (that is, a series of links and routers) between the source and destination hosts
  2. VC numbers, one number for each link along the path
  3. Entries in the forwarding table in each router along the path.
- A packet belonging to a virtual circuit will carry a VC number in its header. Because a virtual circuit may have a different VC number on each link, each intervening router must replace the VC number of each traversing packet with a new VC number.
- The new VC number is obtained from the forwarding table.



forwarding table in northwest router:



Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
...	...	...	...

Fig. 2 A simple virtual circuit network

- The numbers next to the links of R1 in above figure are the link interface numbers.
- Suppose now that Host A requests that the network establishes a VC between itself and Host B.
- Suppose also that the network chooses the path A-R1-R2-B and assigns VC numbers 12, 22 and 32 to the three links in this path for this virtual circuit.
- In this case, when a packet in this VC leaves Host A, the value in the VC number field in the packet header is 12; when it leaves R1, the value is 22; and when it leaves R2, the value is 32.
- How does the router determine the replacement VC number for a packet traversing the router? For a VC network, each router's forwarding table includes VC number translation; for example, the forwarding table in R1 might look something like above fig. 2
- Whenever a new VC is established across a router, an entry is added to the forwarding table.
- Similarly, whenever a VC terminates, the appropriate entries in each table along its path are removed.
- A path from the source router to the destination router must be established before any data packets can be sent.
- This connection is called a VC (virtual circuit), and the subnet is called a virtual-circuit subnet.

- When a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers.
- That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.

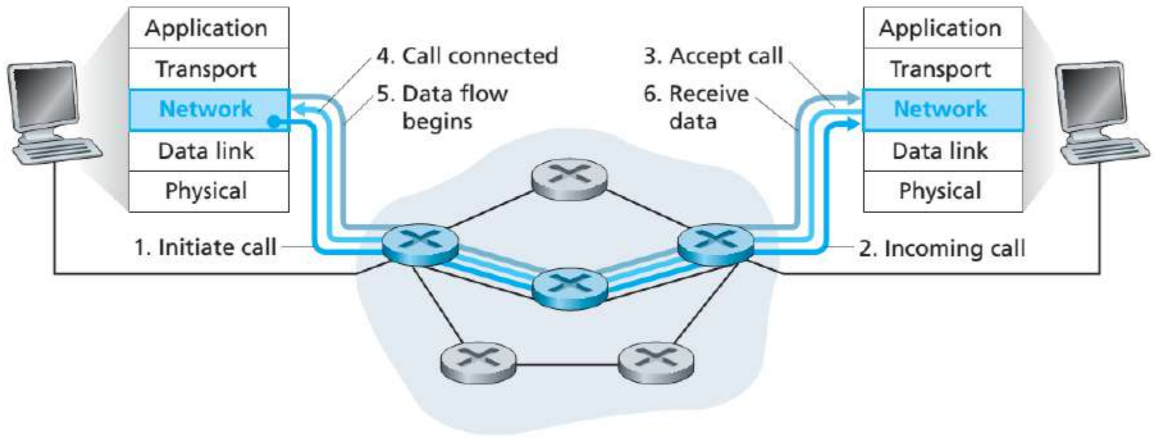


Fig. 3 Virtual-Circuit Setup

- There are three identifiable phases in a virtual circuit:
  1. **VC setup:**
     - During the setup phase, the sending transport layer contacts the network layer, specifies the receiver's address, and waits for the network to set up the VC.
     - The network layer determines the path between sender and receiver, that is, the series of links and routers through which all packets of the VC will travel.
     - The network layer also determines the VC number for each link along the path.
     - Finally, the network layer adds an entry in the forwarding table in each router along the path. During VC setup, the network layer may also reserve resources (for example, bandwidth) along the path of the VC.
  2. **Data transfer:**
     - As shown in Figure 3, once the VC has been established, packets can begin to flow along the VC.
  3. **VC teardown:**
     - This is initiated when the sender (or receiver) informs the network layer of its desire to terminate the VC.
     - The network layer will then typically inform the end system on the other side of the network of the call termination and update the forwarding tables in each of the packet routers on the path to indicate that the VC no longer exists.

Datagram Network (Connection-Less Service)

- In connection less service, packets are injected into the subnet individually and routed independently of each other.
- No advance setup is needed. In this context, the packets are frequently called datagrams (in analogy with telegrams) and the subnet is called a **datagram subnet**.
- Suppose that the process P1 in Figure 4 has a long message for P2. It hands the message to the transport layer with instructions to deliver it to process P2 on host H2.

- Let us assume that the message is four times longer than the maximum packet size, so the network layer has to break it into four packets, 1, 2, 3, and 4 and sends each of them in turn to router A using some point-to-point protocol, for example, PPP.

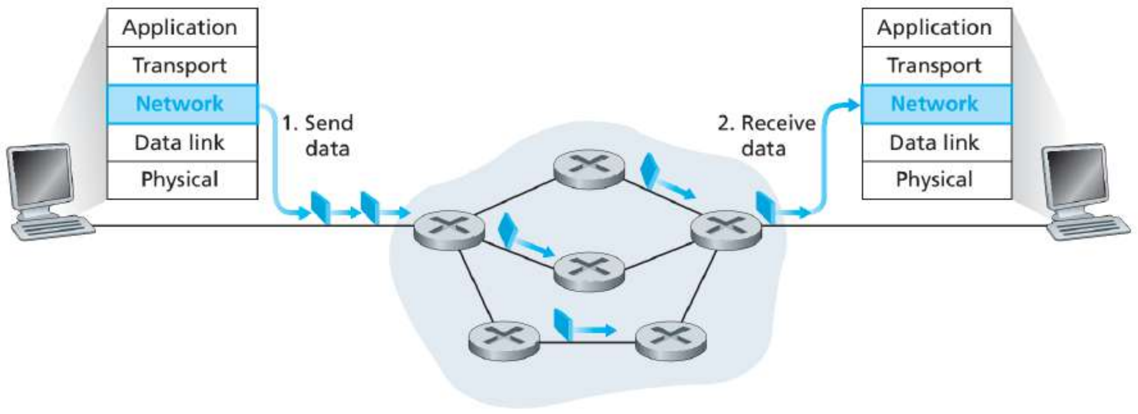


Fig. 4 Datagram Network

- At this point the carrier takes over. Every router has an internal table telling it where to send packets for each possible destination. Each table entry is a pair consisting of a destination and the outgoing line to use for that destination.
- Only directly-connected lines can be used.

Datagram Network vs. Virtual Circuit Network

Issue	Datagram	Virtual Circuit
Connection Setup	None	Required
Addressing	Packet contains full source and destination address	Packet contains short virtual circuit number identifier.
State Information	None other than router table containing destination network	Each virtual circuit number entered to table on setup, used for routing.
Routing	Packets routed independently	Route established at setup, all packets follow same route.
Effect of Router Failure	Only on packets lost during crash	All virtual circuits passing through failed router terminated.
Congestion Control	Difficult since all packets routed independently router resource requirements can vary.	Simple by pre-allocating enough buffers to each virtual circuit at setup, since maximum number of circuits fixed.

Router architecture

- Routers have four components:
  1. Input ports
  2. Switching fabric
  3. Output ports
  4. Routing processor

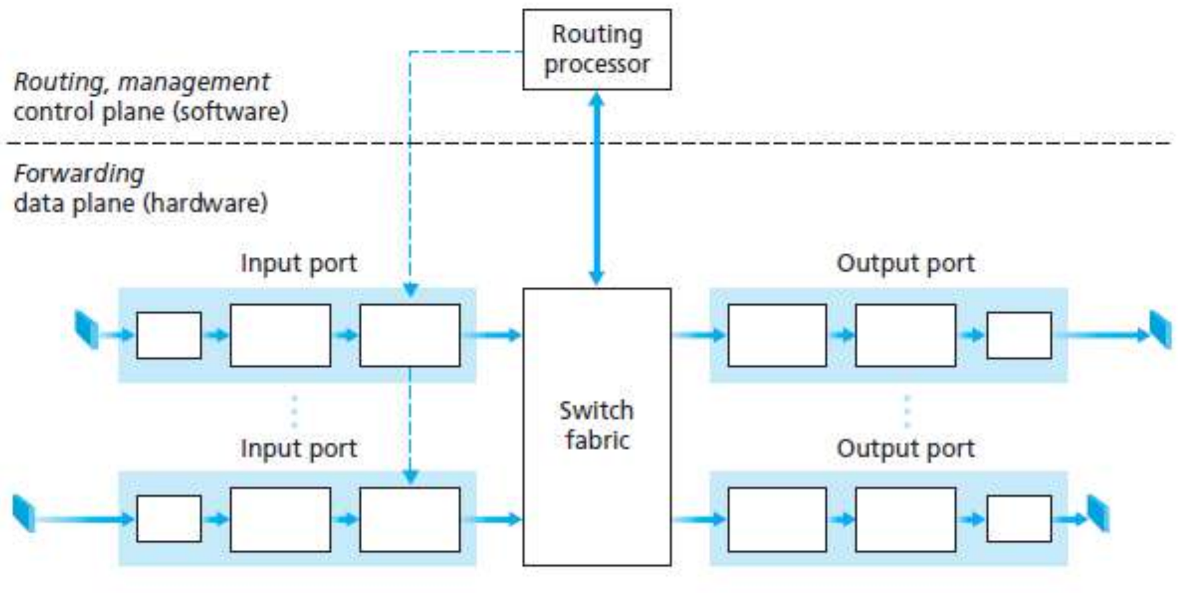


Fig. 5 Router architecture

Input ports

- An input port performs several key functions.
- It performs the physical layer function of terminating an incoming physical link at a router; this is shown in the leftmost box of the input port and the rightmost box of the output port in Figure 5.
- An input port also performs link-layer functions needed to interoperate with the link layer at the other side of the incoming link; this is represented by the middle boxes in the input and output ports.
- Perhaps most crucially, the lookup function is also performed at the input port; this will occur in the rightmost box of the input port. It is here that the forwarding table is consulted to determine the router output port to which an arriving packet will be forwarded via the switching fabric.
- Control packets (for example, packets carrying routing protocol information) are forwarded from an input port to the routing processor.

Switching fabric

- The switching fabric connects the router's input ports to its output ports.
- This switching fabric is completely contained within the router - a network inside of a network router!

Output ports

- An output port stores packets received from the switching fabric and transmits these packets on the outgoing link by performing the necessary link-layer and physical-layer functions.
- When a link is bidirectional (that is, carries traffic in both directions), an output port will typically be paired with the input port for that link on the same line card.

Routing processor

- The routing processor executes the routing protocols, maintains routing tables and attached link state information and computes the forwarding table for the router.
- It also performs the network management functions.

Types of switching fabrics

- Three types of switching fabrics
    1. Switching via memory
    2. Switching via a bus
    3. Switching via an interconnection network

Switching via memory

- The simplest, earliest routers were traditional computers, with switching between input and output ports being done under direct control of the CPU (routing processor).
- An input port with an arriving packet first signalled the routing processor via an interrupt. The packet was then copied from the input port into processor memory.
- The routing processor then extracted the destination address from the header, looked up the appropriate output port in the forwarding table, and copied the packet to the output port's buffers.
- In this scenario, if the memory bandwidth is such that B packets per second can be written into, or read from, memory, then the overall forwarding throughput must be less than B/2.
- Note also that two packets cannot be forwarded at the same time, even if they have different destination ports, since only one memory read/write over the shared system bus can be done at a time.

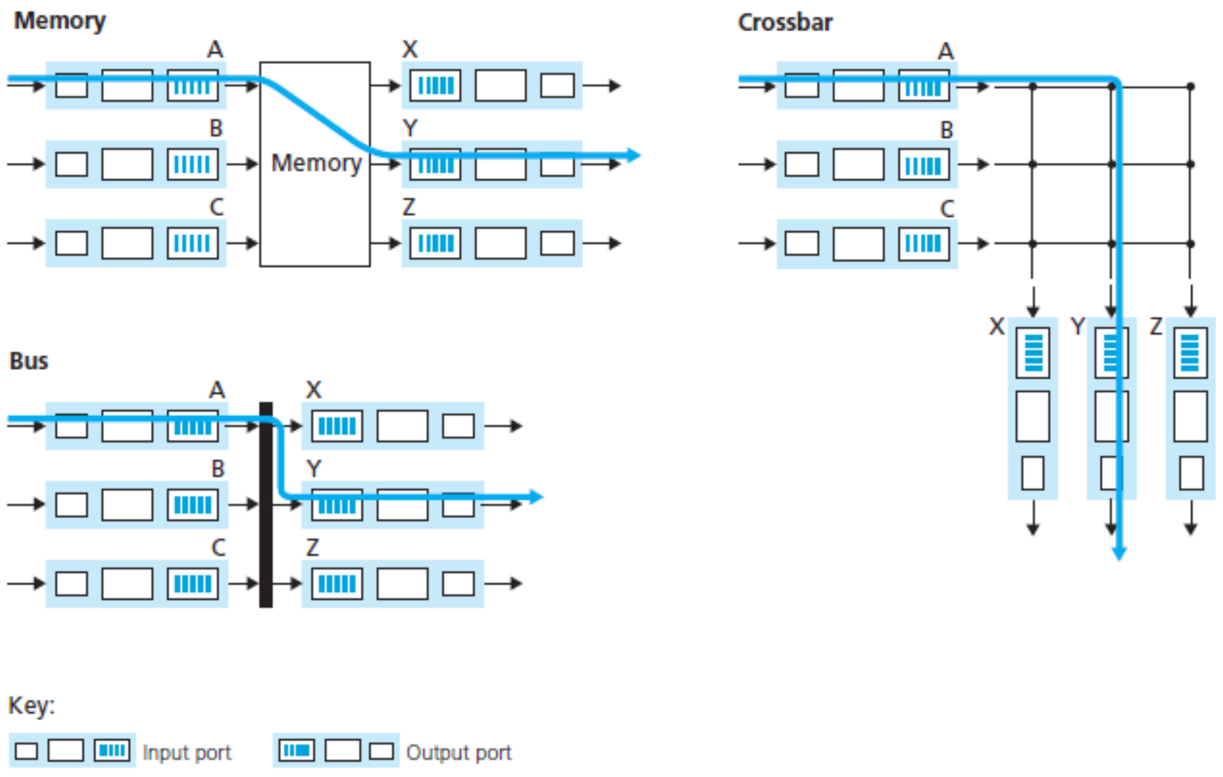


Fig. 6 Three switching techniques

Switching via a bus

- In this approach, an input port transfers a packet directly to the output port over a shared bus, without intervention by the routing processor.

- This is typically done by having the input port pre-pend a switch-internal label (header) to the packet indicating the local output port to which this packet is being transferred and transmitting the packet onto the bus.
- The packet is received by all output ports, but only the port that matches the label will keep the packet.
- The label is then removed at the output port, as this label is only used within the switch to cross the bus.
- If multiple packets arrive to the router at the same time, each at a different input port, all but one must wait since only one packet can cross the bus at a time. Because every packet must cross the single bus, the switching speed of the router is limited to the bus speed.

Switching via an interconnection network

- One way to overcome the bandwidth limitation of a single, shared bus is to use a more sophisticated interconnection network, such as those that have been used in the past to interconnect processors in a multiprocessor computer architecture.
- A crossbar switch is an interconnection network consisting of 2N buses that connect N input ports to N output ports, as shown in Figure 6.
- Each vertical bus intersects each horizontal bus at a crosspoint, which can be opened or closed at any time by the switch fabric controller (whose logic is part of the switching fabric itself).
- When a packet arrives from port A and needs to be forwarded to port Y, the switch controller closes the crosspoint at the intersection of busses A and Y, and port A then sends the packet onto its bus, which is picked up (only) by bus Y.
- Note that a packet from port B can be forwarded to port X at the same time, since the A-to-Y and B-to-X packets use different input and output busses.
- Thus, unlike the previous two switching approaches, crossbar networks are capable of forwarding multiple packets in parallel.
- However, if two packets from two different input ports are destined to the same output port, then one will have to wait at the input, since only one packet can be sent over any given bus at a time.

IPv4 datagram format

- **Version number**: These 4 bits specify the IP protocol version of the datagram. It determines how to interpret the header. Currently the only permitted values are 4 (0100) or 6 (0110).
- **Header length:** Specifies the length of the IP header, in 32-bit words.
- **Type of service:** The type of service (TOS) bits were included in the IPv4 header to allow different types of IP datagrams (for example, datagrams particularly requiring low delay, high throughput, or reliability) to be distinguished from each other.
- **Datagram length:** This is the total length of the IP datagram (header plus data), measured in bytes.
- **Identifier:** Uniquely identifies the datagram. It is incremented by 1 each time a datagram is sent. All fragments of a datagram contain the same identification value. This allows the destination host to determine which fragment belongs to which datagram.
- **Flags:** In order for the destination host to be absolutely sure it has received the last fragment of the original datagram, the last fragment has a flag bit set to 0, whereas all the other fragments have this flag bit set to 1.
- **Fragmentation offset:** When fragmentation of a message occurs, this field specifies the offset, or position, in the overall message where the data in this fragment goes. It is specified in units of 8 bytes (64 bits).

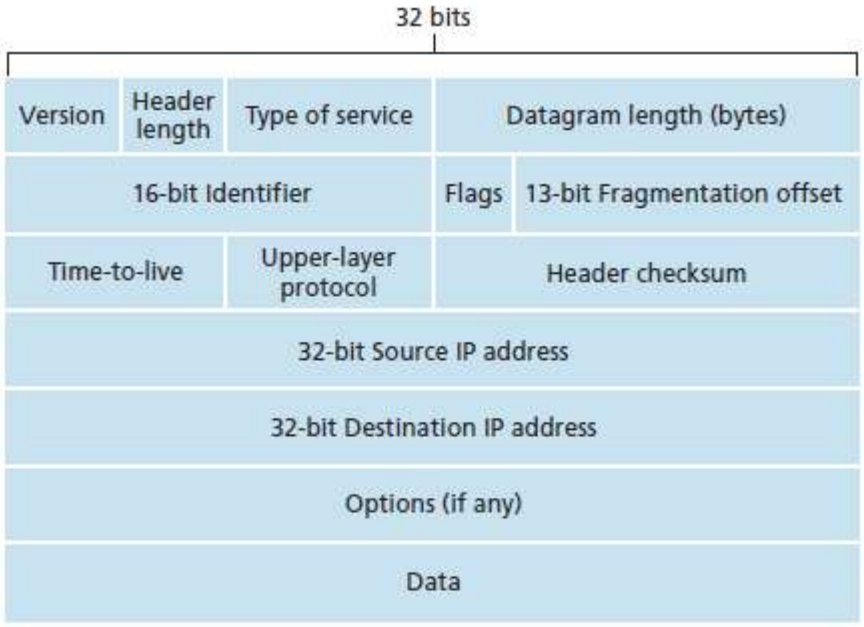


Fig. 7 IPv4 datagram format

- **Time-to-live:** Specifies how long the datagram is allowed to “live” on the network. Each router decrements the value of the TTL field (reduces it by one) prior to transmitting it. If the TTL field drops to zero, the datagram is assumed to have taken too long a route and is discarded.
- **Protocol**: This field is used only when an IP datagram reaches its final destination. The value of this field indicates the specific transport-layer protocol to which the data portion of this IP datagram should be passed. For example, a value of 6 indicates that the data portion is passed to TCP, while a value of 17 indicates that the data is passed to UDP.
- **Header checksum:** The header checksum aids a router in detecting bit errors in a received IP datagram.
- **Source and destination IP addresses:** When a source creates a datagram, it inserts its IP address into the source IP address field and inserts the address of the ultimate destination into the destination IP address field.
- **Options:** The options fields allow an IP header to be extended.
- **Data (payload):** The data to be transmitted in the datagram, either an entire higher-layer message or a fragment of one.

IP addressing: Introduction

- **IP address:** It is 32-bit identifier for host, router interface
- **Interface:** It is a connection between host/router and physical link.
  - A router’s typically have multiple interfaces
  - A host typically has one or two interfaces
- There is an IP addresses associated with each interface.
- **Subnets:** To determine the subnets, detach each interface from its host or router, creating islands of isolated networks, with interfaces terminating the end points of the isolated networks. Each of these isolated networks is called a subnet.
- **Subnet part**: high order bits defines subnet
- **Host part:** low order bits defines host.

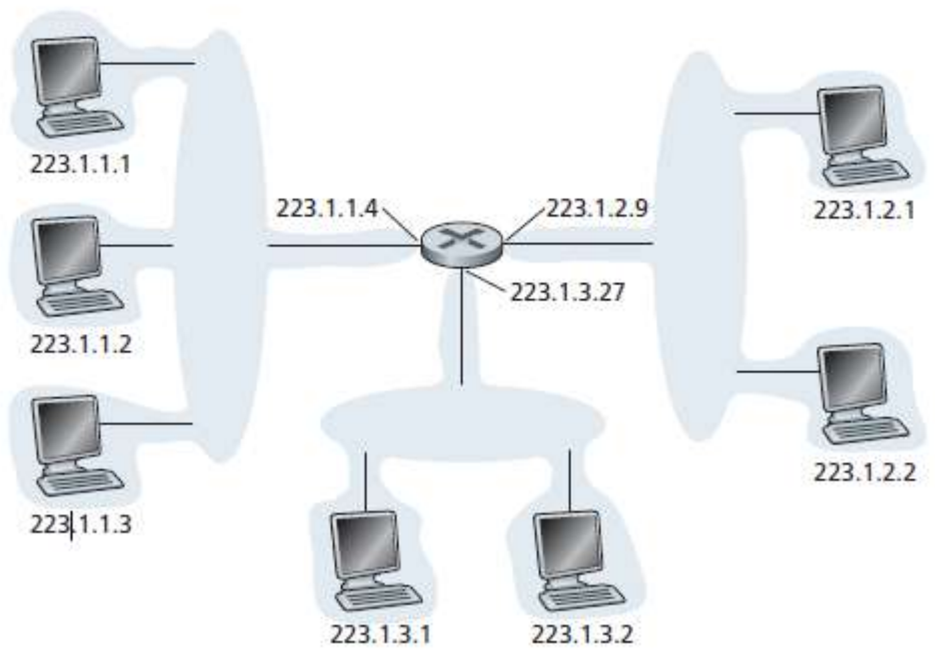
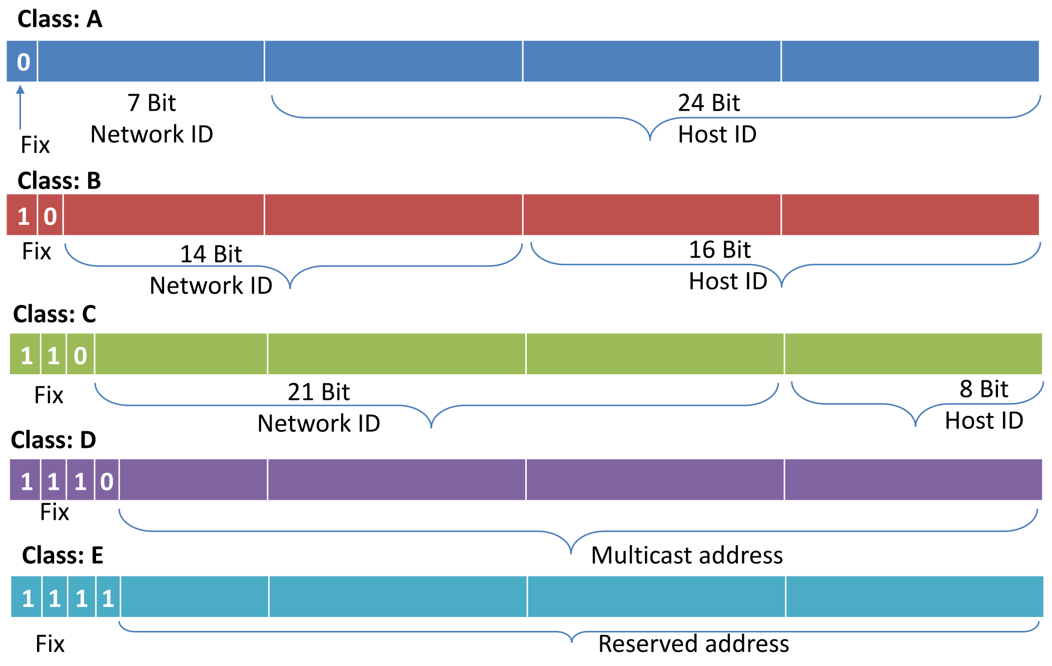


Fig. 8 Interface addresses and subnets

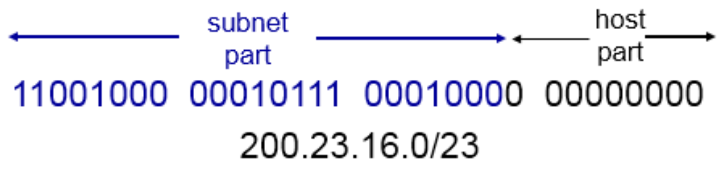
Classification of IP Addresses (Classful Addressing)



CIDR (Classless Inter-Domain Routing)

- Originally, IP addresses were assigned in four major address classes, A through D.

- Each of these classes allocates one portion of the 32-bit IP address format to identify a network gateway - the first 8 bits for class A, the first 16 for class B, and the first 24 for class C. The remainder identify hosts on that network.
- More than 16 million in class A, 65,535 in class B and 254 in class C. (Class D addresses identify multicast domains.)
- To illustrate the problems with the class system, consider that one of the most commonly used classes was Class B.
- An organization that needed more than 254 host machines (500 hosts) would often get a Class B license, even though it would have far fewer than 65,534 hosts.
- This resulted in most of the block of addresses allocated going unused.
- CIDR reduced the problem of wasted address space by providing a new and more flexible way to specify network addresses in routers.
- A single IP address can be used to designate many unique IP addresses with CIDR.
- A CIDR IP address looks like a normal IP address except that it ends with a slash followed by a number, called the IP network prefix. CIDR addresses reduce the size of routing tables and make more IP addresses available within organizations.



IP Addressing Summary:

Class	Leading bits	Size of *network number* bit field	Size of *rest* bit field	Number of networks	Addresses per network	Total addresses in class	Start address	End address	Default subnet mask in dot-decimal notation	CIDR notation
Class A	0	8	24	128 (2^7)	16,777,216 (2^{24})	2,147,483,648 (2^{31})	0.0.0.0	127.255.255.255	255.0.0.0	/8
Class B	10	16	16	16,384 (2^{14})	65,536 (2^{16})	1,073,741,824 (2^{30})	128.0.0.0	191.255.255.255	255.255.0.0	/16
Class C	110	24	8	2,097,152 (2^{21})	256 (2^8)	536,870,912 (2^{29})	192.0.0.0	223.255.255.255	255.255.255.0	/24
Class D (multicast)	1110	not defined	not defined	not defined	not defined	268,435,456 (2^{28})	224.0.0.0	239.255.255.255	not defined	not defined
Class E (reserved)	1111	not defined	not defined	not defined	not defined	268,435,456 (2^{28})	240.0.0.0	255.255.255.255	not defined	not defined

DHCP: Dynamic Host Configuration Protocol

- Dynamic Host Configuration Protocol is a protocol for assigning dynamic IP addresses to devices on a network.

- With dynamic addressing, a device can have a different IP address every time it connects to the network.
- In some systems, the device's IP address can even change while it is still connected. It allows reuse of addresses (only hold address while connected "on"). It also support mobile users who want to join network.

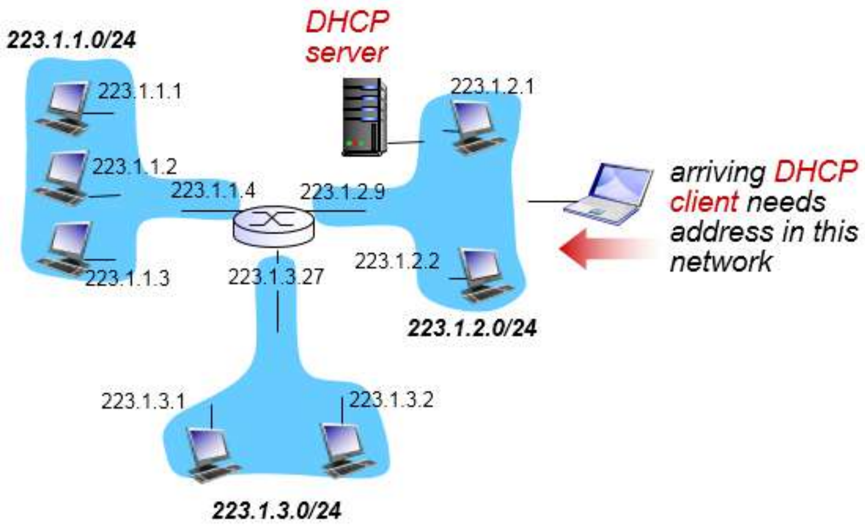


Fig. 9 DHCP client-server scenario

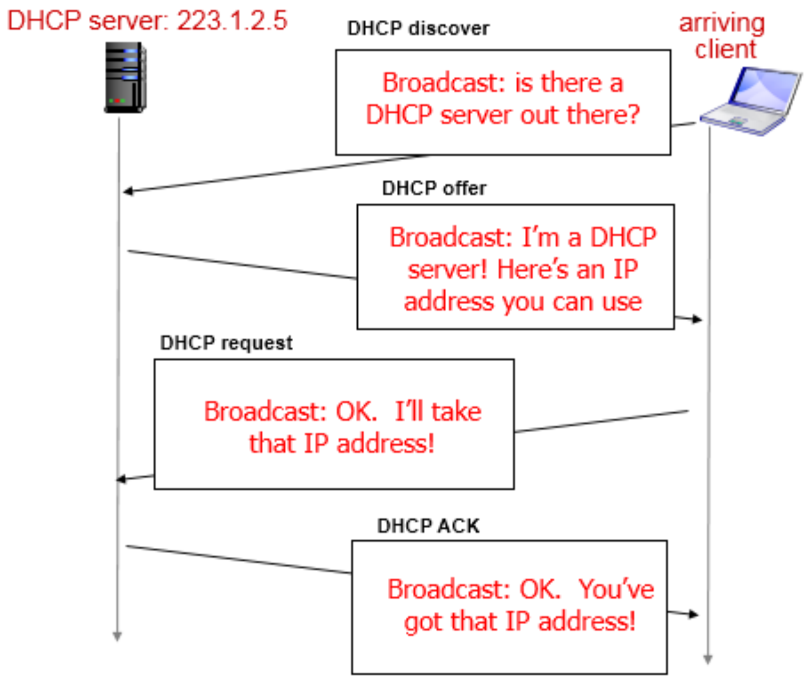

DHCP server: 223.1.2.5
DHCP discover
arriving client
Broadcast: is there a DHCP server out there?
DHCP offer
Broadcast: I'm a DHCP server! Here's an IP address you can use
DHCP request
Broadcast: OK. I'll take that IP address!
DHCP ACK
Broadcast: OK. You've got that IP address!

Fig. 10 DHCP client-server interaction

DHCP server discovery

- The first task of a newly arriving host is to find a DHCP server with which to interact.
- This is done using a DHCP discover message, which a client sends within a UDP packet to port 67.

- The UDP packet is encapsulated in an IP datagram. But to whom should this datagram be sent? The host doesn’t even know the IP address of the network to which it is attaching.
- Given this, the DHCP client creates an IP datagram containing its DHCP discover message along with the broadcast destination IP address of 255.255.255.255 and a “this host” source IP address of 0.0.0.0.
- The DHCP client passes the IP datagram to the link layer, which then broadcasts this frame to all nodes attached to the subnet.

DHCP server offer(s)

- A DHCP server receiving a DHCP discover message responds to the client with a DHCP offer message that is broadcast to all nodes on the subnet, again using the IP broadcast address of 255.255.255.255.
- Since several DHCP servers can be present on the subnet, the client may find itself in the enviable position of being able to choose from among several offers.
- Each server offer message contains the transaction ID of the received discover message, the proposed IP address for the client, the network mask, and an IP address lease time - the amount of time for which the IP address will be valid.

DHCP request

- The newly arriving client will choose from among one or more server offers and respond to its selected offer with a DHCP request message, echoing back the configuration parameters.

DHCP ACK

- The server responds to the DHCP request message with a DHCP ACK message, confirming the requested parameters.

Network Address Translation (NAT)

- The Internet has grown larger than anyone ever imagined it could be.
- Although the exact size is unknown, the current estimate is that there are about 100 million hosts and more than 350 million users actively on the Internet.
- In fact, the rate of growth has been such that the Internet is effectively doubling in size each year.
- So what does the size of the Internet have to do with NAT? For a computer to communicate with other computers and Web servers on the Internet, it must have an IP address.
- An IP address is a unique 32-bit number that identifies the location of your computer on a network.
- When IP addressing first came out, everyone thought that there were sufficiently of addresses to cover any need. Theoretically, you could have 4,294,967,296 unique addresses (232). The actual number of available addresses is smaller (somewhere between 3.2 and 3.3 billion) because of the way that the addresses are separated into classes, and because some addresses are set aside for multicasting, testing or other special uses.
- With the explosion of the Internet and the increase in home networks and business networks, the number of available IP addresses is simply not enough.
- The obvious solution is to redesign the address format to allow for more possible addresses. This is being developed (called IPv6), but will take several years to implement because it requires modification of the entire infrastructure of the Internet.
- This is where NAT (RFC 1631) comes to the rescue.
- Network Address Translation allows a single device, such as a router, to act as an agent between the Internet (or "public network") and a local (or "private") network.

- This means that only a single, unique IP address is required to represent an entire group of computers.

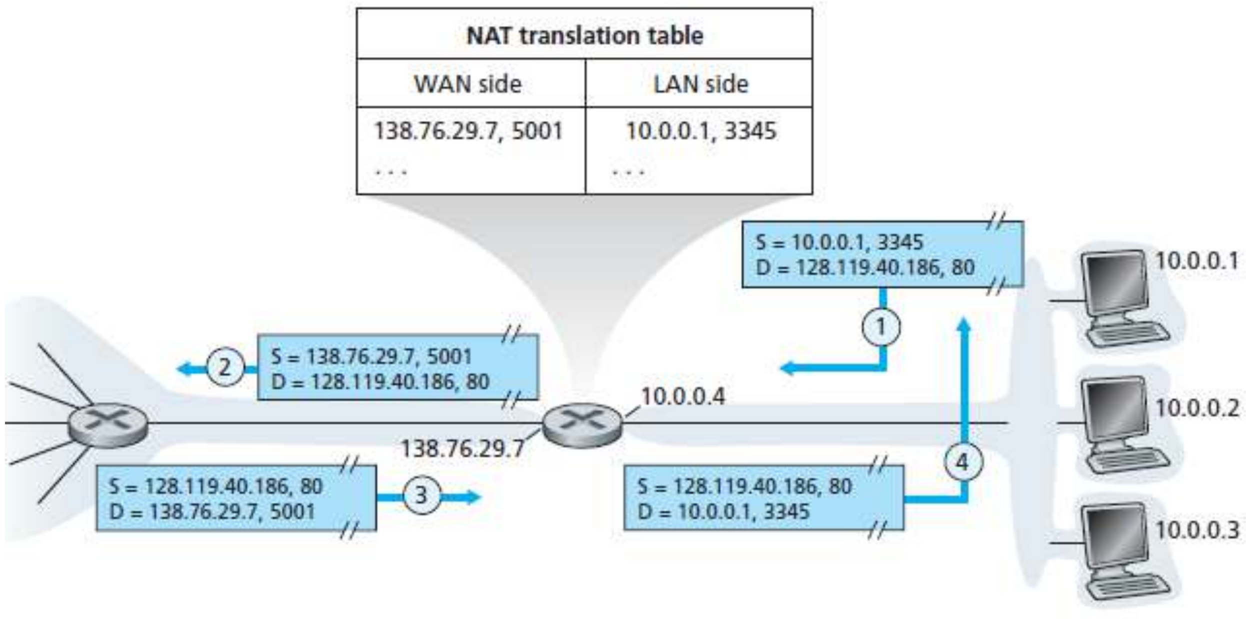


Fig. 11 Network address translation

ICMP: Internet Control Message Protocol

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

Message Type	Description
Destination unreachable	Packet could not be delivered
Time exceeded	Time to live field hit 0
Parameter problem	Invalid header field
Source quench	Choke packet
Redirect	Teach a router about geography
Echo	Ask a machine if it is alive
Echo reply	Yes, I am alive
Timestamp request	Same as Echo request, but with timestamp
Timestamp reply	Same as Echo reply, but with timestamp

- The DESTINATION UNREACHABLE message is used when the subnet or a router cannot locate the destination or when a packet with the DF bit cannot be delivered because a ''small-packet'' network stands in the way.
- The TIME EXCEEDED message is sent when a packet is dropped because its counter has reached zero.
- The PARAMETER PROBLEM message indicates that an illegal value has been detected in a header field.

- This problem indicates a bug in the sending host's IP software or possibly in the software of a router transited.
- The SOURCE QUENCH message was formerly used to throttle hosts that were sending too many packets. When a host received this message, it was expected to slow down.
- The REDIRECT message is used when a router notices that a packet seems to be routed wrong. It is used by the router to tell the sending host about the probable error.
- The ECHO and ECHO REPLY messages are used to see if a given destination is reachable and alive.
- Upon receiving the ECHO message, the destination is expected to send an ECHO REPLY message back.
- The TIMESTAMP REQUEST and TIMESTAMP REPLY messages are similar, except that the arrival time of the message and the departure time of the reply are recorded in the reply.

IPv6 Datagram Format

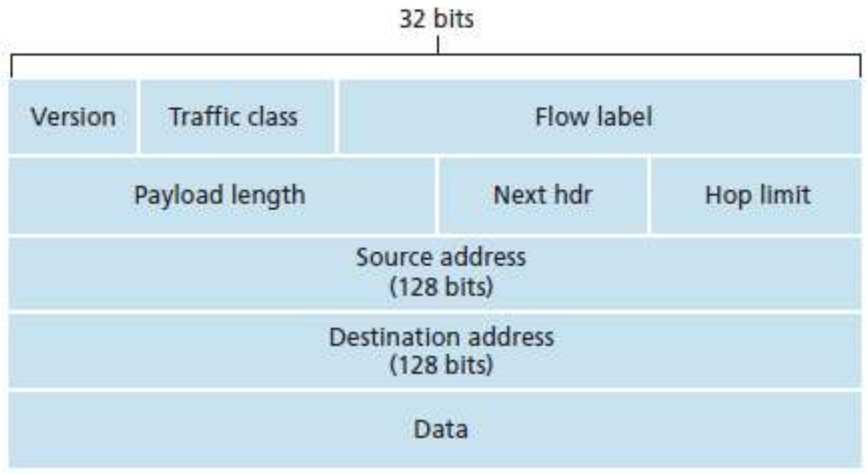


Fig. 12 IPv6 datagram format

- **Version:** The size of the Version field is 4 bits. The Version field shows the version of IP and is set to 6.
- **Traffic Class**: The size of Traffic Class field is 8 bits. Traffic Class field is similar to the IPv4 Type of Service (ToS) field. The Traffic Class field indicates the IPv6 packet’s class or priority.
- **Flow Label**: The size of Flow Label field is 20 bits. The Flow Label field provide additional support for real-time datagram delivery and quality of service features. The purpose of Flow Label field is to indicate that this packet belongs to a specific sequence of packets between a source and destination and can be used to prioritized delivery of packets for services like voice.
- **Payload Length**: The size of the Payload Length field is 16 bits. The Payload Length field shows the length of the IPv6 payload, including the extension headers and the upper layer protocol data
- **Next Header**: The size of the Next Header field is 8 bits. The Next Header field shows either the type of the first extension (if any extension header is available) or the protocol in the upper layer such as TCP, UDP, or ICMPv6.
- **Hop Limit**: The size of the Hop Limit field is 8 bits The Hop Limit field shows the maximum number of routers the IPv6 packet can travel. This Hop Limit field is similar to IPv4 Time to Live (TTL) field.
- **Source Address:** The size of the Source Address field is 128 bits. The Source Address field shows the IPv6 address of the source of the packet.

- **Destination Address:** The size of the Destination Address field is 128 bits. The Destination Address field shows the IPv6 address of the destination of the packet.
- **Data:** The data to be transmitted in the datagram, either an entire higher-layer message or a fragment of one.

Difference between IPv4 and IPv6

IPv4	IPv6
• IPv4 addresses are 32 bit length.	• IPv6 addresses are 128 bit length.
• Fragmentation is done by sender and forwarding routers.	• Fragmentation is done only by sender.
• No packet flow identification.	• Packet flow identification is available within the IPv6 header using the Flow Label field.
• Checksum field is available in header	• No checksum field in header.
• Options fields are available in header.	• No option fields, but Extension headers are available.
• Address Resolution Protocol (ARP)is available to map IPv4 addresses to MAC addresses.	• Address Resolution Protocol (ARP) is replaced with Neighbour Discovery Protocol.
• Broadcast messages are available.	• Broadcast messages are not available.
• Manual configuration (Static) of IP addresses or DHCP (Dynamic configuration) is required to configure IP addresses.	• Auto-configuration of addresses is available.

The Link-State (LS) Routing Algorithm (Dijkstra's algorithm)

- Dijkstra's algorithm computes the least-cost path from one node (the source, which we will refer to as u) to all other nodes in the network.
- Dijkstra's algorithm is iterative and has the property that after the k^{th} iteration of the algorithm, the least-cost paths are known to k destination nodes, and among the least-cost paths to all destination nodes, these k paths will have the k smallest costs.
- Let us define the following notation:
  - D(v): cost of the least-cost path from the source node to destination v as of this iteration of the algorithm.
  - p(v): previous node (neighbor of v) along the current least-cost path from the source to v.
  - N' : subset of nodes; v is in N' if the least-cost path from the source to v is definitively known.
- The global routing algorithm consists of an initialization step followed by a loop.
- The number of times the loop is executed is equal to the number of nodes in the network.
- Upon termination, the algorithm will have calculated the shortest paths from the source node u to every other node in the network.
- As an example, let's consider the network in Figure 13 and compute the least-cost paths from u to all possible destinations.

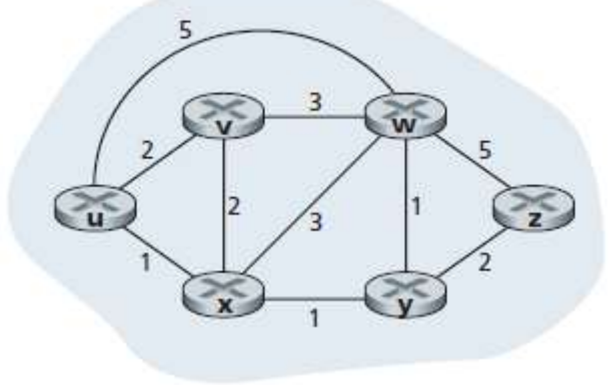


Fig. 13 Abstract graph model of a computer network

```
Initialization:
  N' = {u}
  for all nodes v
    if v is a neighbor of u
      then D(v) = c(u,v)
    else D(v) = ∞

Loop
  find w not in N' such that D(w) is a minimum
  add w to N'
  update D(v) for each neighbor v of w and not in N':
       D(v) = min( D(v), D(w) + c(w,v) )
  /* new cost to v is either old cost to v or known
   least path cost to w plus cost from w to v */
until N'= N
```

- Let’s consider the few first steps in detail.
- In the initialization step, the currently known least-cost paths from u to its directly attached neighbours, v, x, and w, are initialized to 2, 1, and 5, respectively. Note in particular that the cost to w is set to 5 (even though we will soon see that a lesser-cost path does indeed exist) since this is the cost of the direct (one hop) link from u to w. The costs to y and z are set to infinity because they are not directly connected to u.
- In the first iteration, we look among those nodes not yet added to the set N’ and find that node with the least cost as of the end of the previous iteration. That node is x, with a cost of 1, and thus x is added to the set N’. Line 12 of the LS algorithm is then performed to update D(v) for all nodes v, yielding the results shown in the second line (Step 1) in below table. The cost of the path to v is unchanged. The cost of the path to w (which was 5 at the end of the initialization) through node x is found to have a cost of 4. Hence this lower-cost path is selected and w’s predecessor along the shortest path from u is set to x. Similarly, the cost to y (through x) is computed to be 2, and the table is updated accordingly.
- In the second iteration, nodes v and y are found to have the least-cost paths (2), and we break the tie arbitrarily and add y to the set N’ so that N’ now contains u, x, and y. The cost to the remaining

nodes not yet in N', that is, nodes v, w, and z are updated via line 12 of the LS algorithm, yielding the results shown in the third row in the below table.

- And so on. . . .
- When the LS algorithm terminates, we have, for each node, its predecessor along the least-cost path from the source node.
- For each predecessor, we also have its predecessor, and so in this manner we can construct the entire path from the source to all destinations.
- The forwarding table in a node, say node u, can then be constructed from this information by storing, for each destination, the next-hop node on the least-cost path from u to the destination.
- Figure 14. Shows the resulting least-cost paths for u for the network in Figure 13.

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

Table: Running the link-state algorithm on the network in Figure 13

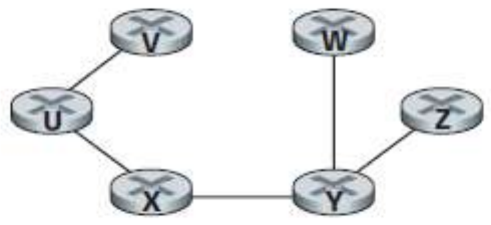


Fig. 14 Least cost path for nodule u

The Distance-Vector (DV) Routing Algorithm

- Distance-vector (DV) algorithm is iterative, asynchronous, and distributed.
- It is distributed in that each node receives some information from one or more of its directly attached neighbours, performs a calculation, and then distributes the results of its calculation back to its neighbours.
- It is iterative in that this process continues on until no more information is exchanged between neighbours.
- The algorithm is asynchronous in that it does not require all of the nodes to operate in lockstep with each other.
- Let dx(y) be the cost of the least-cost path from node x to node y. Then the least costs are related by the celebrated Bellman-Ford equation:

$$d_x(y) = \min_v\{c(x,v) + d_v(y)\}$$

- where the $\min_v$ in the equation is taken over all of x's neighbours. Indeed, after traveling from x to v, if we then take the least-cost path from v to y, the path cost will be c(x,v) + dv(y).

- Since we must begin by traveling to some neighbour v, the least cost from x to y is the minimum of c(x,v) + dv(y) taken over all neighbours v.

```
Initialization:
   for all destinations y in N:
      $D_x(y)$ = c(x,y)    /* if y is not a neighbor then c(x,y) = ∞ */
   for each neighbor w
      $D_w(y)$ = ? for all destinations y in N
   for each neighbor w
      send distance vector $\mathbf{D}_x$ = [$D_x(y)$: y in N] to w

loop
   wait (until I see a link cost change to some neighbor w or
         until I receive a distance vector from some neighbor w)

   for each y in N:
      $D_x(y) = \min_v\{c(x,v) + D_v(y)\}$

   if $D_x(y)$ changed for any destination y
      send distance vector $\mathbf{D}_x$ = [$D_x(y)$: y in N] to all neighbors

forever
```

- Figure 15 illustrates the operation of the DV algorithm for the simple three node network shown at the top of the figure.
- The operation of the algorithm is illustrated in a synchronous manner, where all nodes simultaneously receive distance vectors from their neighbours, compute their new distance vectors, and inform their neighbours if their distance vectors have changed.
- The leftmost column of the figure displays three initial routing tables for each of the three nodes.
- For example, the table in the upper-left corner is node x’s initial routing table.
- Within a specific routing table, each row is a distance vector - specifically, each node’s routing table includes its own distance vector and that of each of its neighbours.
- Thus, the first row in node x’s initial routing table is $D_x = [D_x(x), D_x(y), D_x(z)] = [0, 2, 7]$.
- The second and third rows in this table are the most recently received distance vectors from nodes y and z, respectively.
- Because at initialization node x has not received anything from node y or z, the entries in the second and third rows are initialized to infinity.
- After initialization, each node sends its distance vector to each of its two neighbours.
- This is illustrated in Figure 15 by the arrows from the first column of tables to the second column of tables.
- For example, node x sends its distance vector D_x = [0, 2, 7] to both nodes y and z. After receiving the updates, each node recomputes its own distance vector.
- For example, node x computes

$$D_x(x) = 0$$
$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} = \min\{2 + 0, 7 + 1\} = 2$$
$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} = \min\{2 + 1, 7 + 0\} = 3$$

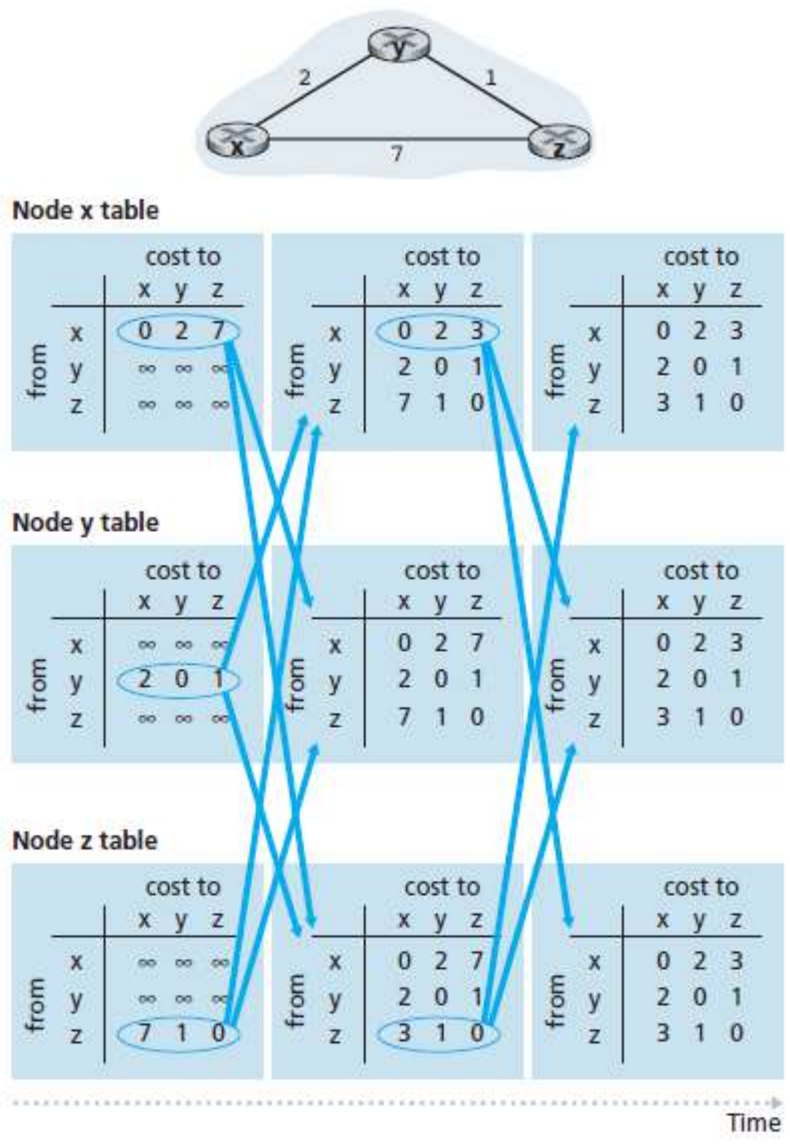


Fig. 15 Distance-vector (DV) algorithm

- The second column therefore displays, for each node, the node's new distance vector along with distance vectors just received from its neighbours.
- Note, that node x's estimate for the least cost to node z, Dx(z), has changed from 7 to 3.
- Also note that for node x, neighbouring node y achieves the minimum in line 14 of the DV algorithm; thus at this stage of the algorithm, we have at node x that v*(y) =y and v*(z) = y.
- After the nodes recomputes their distance vectors, they again send their updated distance vectors to their neighbours (if there has been a change).
- This is illustrated in Figure 15 by the arrows from the second column of tables to the third column of tables.
- Note that only nodes x and z send updates: node y's distance vector didn't change so node y doesn't send an update.

- After receiving the updates, the nodes then recomputes their distance vectors and update their routing tables, which are shown in the third column.
- The process of receiving updated distance vectors from neighbours, recomputing routing table entries, and informing neighbours of changed costs of the least-cost path to a destination continues until no update messages are sent.
- At this point, since no update messages are sent, no further routing table calculations will occur and the algorithm will enter a quiescent state; that is, all nodes will be performing the wait in Lines 10–11 of the DV algorithm.
- The algorithm remains in the quiescent state until a link cost changes.

Comparison of (Difference between) LS and DV Routing Algorithms

Distance Vector Protocol	**Link state protocol**
Entire routing table is sent as an update	Updates are incremental & entire routing table is not sent as update
Distance vector protocol send periodic update at every 30 or 90 second	Updates are triggered not periodic
Update are broadcasted	Updates are multicasted
Updates are sent to directly connected neighbour only	Update are sent to entire network & to just directly connected neighbour
Routers don't have end to end visibility of entire network.	Routers have visibility of entire network of that area only.
It is prone to routing loops	No routing loops

The Count to Infinity problem

- Distance vector routing works in theory but has a serious drawback in practice.
- Consider a router whose best route to destination X is large.
- If on the next exchange neighbour A suddenly reports a short delay to X, the router just switches over to using line to A to send traffic to X.
- Suppose A is down initially and all the other routers know this. In other words, they have all recorded the delay to A as infinity.
- When A comes up, the other routers learn about it via the vector exchanges.
- At the time of the first exchange, B learns that its left neighbour has zero delay to A.
- B now makes an entry in its routing table that A is one hop away to the left.
- All the other routers still think that A is down. At this point, the routing table entries for A are as shown in the second row of Figure 16 (a).
- On the next exchange, C learns that B has a path of length 1 to A, so it updates its routing table to indicate a path of length 2, but D and E do not hear the good news until later.
- Clearly, the good news is spreading at the rate of one hop per exchange.

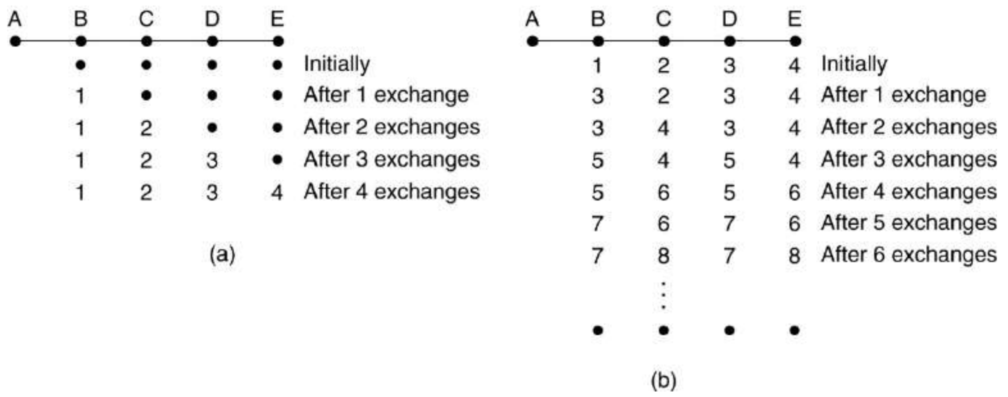
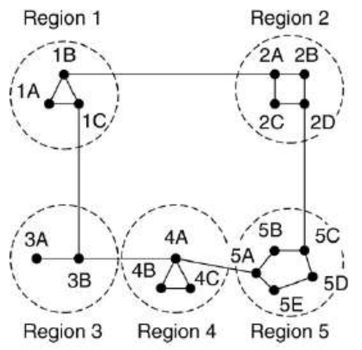


Fig. 16: The Count to infinity problem

- Now let us consider the situation of Figure 16 (b), in which all the lines and routers are initially up. Routers B, C, D, and E have distances to A of 1, 2, 3, and 4, respectively.
- Suddenly A goes down, or alternatively, the line between A and B is cut, which is effectively the same thing from B's point of view.
- At the first packet exchange, B does not hear anything from A.
- Fortunately, C says: Do not worry; I have a path to A of length 2.
- Little does B know that C's path runs through B itself. For all B knows, C might have ten lines all with separate paths to A of length 2.
- As a result, B thinks it can reach A via C, with a path length of 3. D and E do not update their entries for A on the first exchange.
- On the second exchange, C notices that each of its neighbours claims to have a path to A of length 3.
- It picks one of them at random and makes its new distance to A 4, as shown in third row of Figure 16(b).
- Subsequent exchanges produce the history shown in the rest of Figure 16(b).
- From this figure, it should be clear why bad news travels slowly: no router ever has a value more than one higher than the minimum of all its neighbours.
- Gradually, all routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinity.
- For this reason, it is wise to set infinity to the longest path plus 1.
- Not entirely surprisingly, this problem is known as the count-to-infinity problem.

Hierarchical Routing

- As networks grow in size, the router routing tables grow proportionally.
- Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them.
- At a certain point the network may grow to the point where it is no longer feasible for every router to have an entry for every other router, so the routing will have to be done hierarchically, as it is in the telephone network.
- When hierarchical routing is used, the routers are divided into what called **regions**, with each router knowing all the details about how to route packets to destinations within its own region, but knowing nothing about the internal structure of other regions.



(a)

Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

Fig. 2: Hierarchical Routing

- Figure 17 gives a quantitative example of routing in a two-level hierarchy with five regions. The full routing table for router 1A has 17 entries, as shown in Figure 17 (b).
- When routing is done hierarchically, as in Figure 17 (c), there are entries for all the local routers as before, but all other regions have been condensed into a single router, so all traffic for region 2 goes via the 1B -2A line, but the rest of the remote traffic goes via the 1C -3B line. Hierarchical routing has reduced the table from 17 to 7 entries.
- As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase.
- Unfortunately, these gains in space are not free. There is a penalty to be paid, and this penalty is in the form of increased path length.
- For example, the best route from 1A to 5C is via region 2, but with hierarchical routing all traffic to region 5 goes via region 3, because that is better for most destinations in region 5.
- When a single network becomes very large, an interesting question is: How many levels should the hierarchy have? For example, consider a subnet with 720 routers.
- If there is no hierarchy, each router needs 720 routing table entries. If the subnet is partitioned into 24 regions of 30 routers each, each router needs 30 local entries plus 23 remote entries for a total of 53 entries.

Broadcast Routing

- In some applications, hosts need to send messages to many or all other hosts.
- For example, a service distributing weather reports, stock market updates, or live radio programs might work best by broadcasting to all machines and letting those that are interested read the data.
- Sending a packet to all destinations simultaneously is called broadcasting.
- **First** broadcasting method that requires no special features from the subnet is for the source to simply send a distinct packet to each destination.

- Not only is the method wasteful of bandwidth, but it also requires the source to have a complete list of all destinations. In practice this may be the only possibility, but it is the least desirable of the methods.

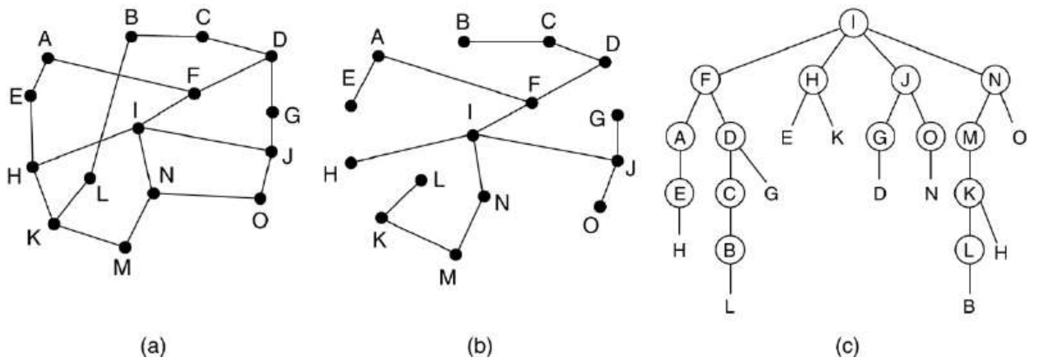


Fig. 3: (a) A subnet. (b) A sink tree. (c) The tree built by reverse path forwarding

- Flooding is another Second method. Although flooding is ill-suited for ordinary point-to-point communication, for broadcasting it might rate serious consideration, especially if none of the methods described below are applicable.
- The problem with flooding as a broadcast technique is the same problem it has as a point-to-point routing algorithm: it generates too many packets and consumes too much bandwidth.
- A third algorithm is multi destination routing.
- If this method is used, each packet contains either a list of destinations or a bit map indicating the desired destinations.
- When a packet arrives at a router, the router checks all the destinations to determine the set of output lines that will be needed.
- The router generates a new copy of the packet for each output line to be used and includes in each packet only those destinations that are to use the line.
- A fourth broadcast algorithm makes explicit use of the sink tree for the router initiating the broadcast-or any other convenient spanning tree for that matter.
- A spanning tree is a subset of the subnet that includes all the routers but contains no loops.
- If each router knows which of its lines belong to the spanning tree, it can copy an incoming broadcast packet onto all the spanning tree lines except the one it arrived on.
- Fifth broadcast algorithm is reverse path forwarding, is remarkably simple once it has been pointed out.
- When a broadcast packet arrives at a router, the router checks to see if the packet arrived on the line that is normally used for sending packets to the source of the broadcast.
- If so, there is an excellent chance that the broadcast packet itself followed the best route from the router and is therefore the first copy to arrive at the router.
- This being the case, the router forwards copies of it onto all lines except the one it arrived on.

Multicast Routing

- Sending a message to a group is called multicasting, and its routing algorithm is called multicast routing.
- Multicasting requires group management. Some way is needed to create and destroy groups, and to allow processes to join and leave groups.

- To do multicast routing, each router computes a spanning tree covering all other routers.
- For example, in Figure (a) we have two groups, 1 and 2.
- Some routers are attached to hosts that belong to one or both of these groups, as indicated in the figure.
- A spanning tree for the leftmost router is shown in Figure (b).
- When a process sends a multicast packet to a group, the first router examines its spanning tree and prunes it, removing all lines that do not lead to hosts that are members of the group.
- In our example, Figure (c) shows the pruned spanning tree for group 1.

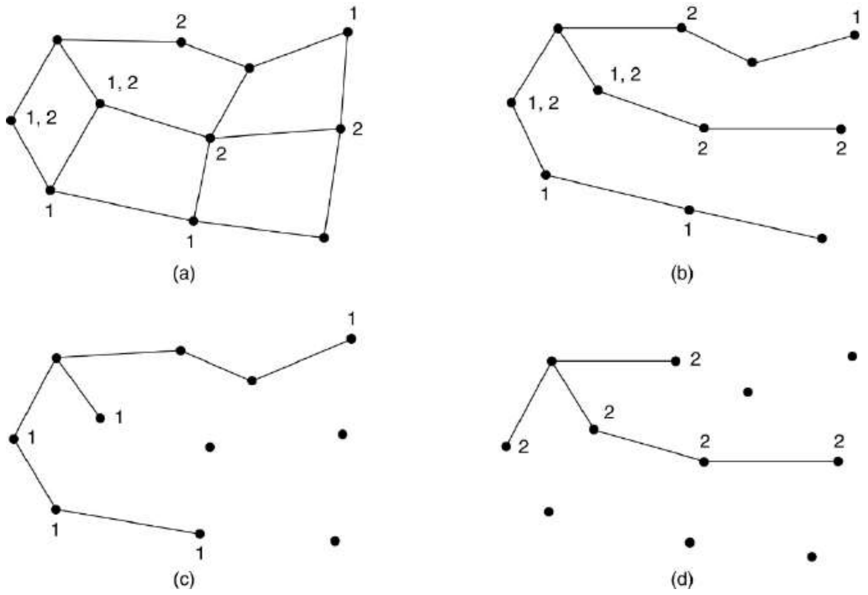


Fig. 49: (a) A network. (b) A spanning tree for the leftmost router

- Similarly, Figure (d) shows the pruned spanning tree for group 2. Multicast packets are forwarded only along the appropriate spanning tree.

Intra-AS Routing

- It is also known as interior gateway protocols (IGP)
- Most common intra-AS routing protocols:
    1. RIP: Routing Information Protocol
    2. OSPF: Open Shortest Path First
    3. IGRP: Interior Gateway Routing Protocol

RIP (Routing Information Protocol)

- The Routing Information Protocol (RIP) defines a way for routers, which connect networks using the Internet Protocol (IP), to share information about how to route traffic among networks.
- Each RIP router maintains a routing table, which is a list of all the destinations (networks) it knows how to reach, along with the distance to that destination.
- RIP uses a distance vector algorithm to decide which path to put a packet on to get to its destination.
- It stores in its routing table the distance for each network it knows how to reach, along with the address of the "next hop" router - another router that is on one of the same networks - through which a packet has to travel to get to that destination.

- If it receives an update on a route and the new path is shorter, it will update its table entry with the length and next-hop address of the shorter path; if the new path is longer, it will wait through a "hold-down" period to see if later updates reflect the higher value as well, and only update the table entry if the new, longer path is stable.
- Using RIP, each router sends its entire routing table to its closest neighbours every 30 seconds. (The neighbours are the other routers to which this router is connected directly - that is, the other routers on the same network segments this router is on.)
- The neighbours in turn will pass the information on to their nearest neighbours, and so on, until all RIP hosts within the network have the same knowledge of routing paths, a state known as convergence.

OSPF (Open Shortest Path First)

- The Internet is made up of a large number of **Autonomous Systems** (**AS**).
- A routing algorithm within an AS is called an interior gateway protocol; an algorithm for routing between AS is called an exterior gateway protocol.
- Many of the ASes in the Internet are themselves large and nontrivial to manage.
- OSPF allows them to be divided into numbered areas, where an area is a network or a set of contiguous networks.
- Areas do not overlap but need not be exhaustive, that is, some routers may belong to no area. An area is a generalization of a subnet.
- Every AS has a backbone area, called area 0.
- All areas are connected to the backbone, possibly by tunnels, so it is possible to go from any area in the AS to any other area in the AS via the backbone.
- Each router that is connected to two or more areas is part of the backbone. As with other areas, the topology of the backbone is not visible outside the backbone.

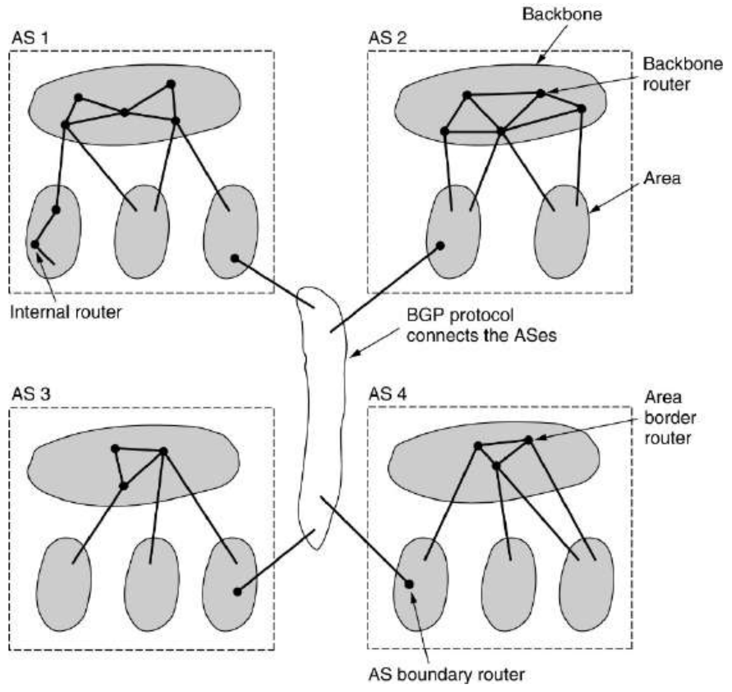


Fig. 5: The relation between ASes, backbones, and areas in OSPF

- Within an area, each router has the same link state database and runs the same shortest path algorithm.
- Its main job is to calculate the shortest path from itself to every other router in the area, including the router that is connected to the backbone, of which there must be at least one.
- A router that connects to two areas needs the databases for both areas and must run the shortest path algorithm for each one separately.
- This algorithm forces a star configuration on OSPF with the backbone being the hub and the other areas being spokes. Packets are routed from source to destination ''as is.''
- They are not encapsulated or tunneled, unless going to an area whose only connection to the backbone is a tunnel. Figure shows part of the Internet with ASes and areas.

OSPF distinguishes four classes of routers:

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

Area border router (ABR)

- An area border router (ABR) is a router that connects one or more areas to the main backbone network.
- It is considered a member of all areas it is connected to.
- An ABR keeps multiple copies of the link-state database in memory, one for each area to which that router is connected.

Autonomous system boundary router (ASBR)

- An autonomous system boundary router (ASBR) is a router that is connected to more than one Routing protocol and that exchanges routing information with routers in other protocols.
- ASBRs typically also run an exterior routing protocol (e.g., BGP), or use static routes, or both.
- An ASBR is used to distribute routes received from other, external ASs throughout its own autonomous system.

Internal router (IR)

- An internal router is a router that has OSPF neighbour relationships with interfaces in the same area. An internal router has all its interfaces in a single area.

Backbone router (BR)

- The backbone routers accept information from the area border routers in order to compute the best route from each backbone router to every other router.
- This information is propagated back to the area border routers, which advertise it within their areas.

Comparison between RIP OSPF and BGP

RIP	OSPF	BGP
RIP is intra domain routing protocol used with in the autonomous system	OSPF is also intra domain routing protocol used with in the autonomous system	It is inter domain routing protocol used between the autonomous system
RIP is used for Small networks with maximum number of hops 16	OSPF is used in large autonomous system with no limitation	The BGP protocol is used for very large-scale networks
RIP uses Distance Vector	OSPF uses Link State	BGP uses Path Vector

RIP send entire routing update to all directly connected interface	OSPF send multicast Hello packet to the neighbours, to create session	BGP send Open packet to the neighbours to create session
RIP use Bellman ford Algorithm	OSPF use Dijikstra Algorithm	BGP use Path-Vector Routing

Consider a router that interconnects three subnets: Subnet 1, Subnet 2, and Subnet 3. Suppose all of the interfaces in each of these three subnets are required to have the prefix 23.1.17/24. Also suppose that Subnet 1 is required to support at least 60 interfaces, Subnet 2 is to support at least 90 interfaces, and Subnet 3 is to support at least 12 interfaces. Provide three network addresses (of the form a.b.c.d/x) that satisfy these constraints.

- For Subnet1 we have to support at least 60 interfaces and 2^6 >= 60 so the prefix for subnet1 is 32-6 = 26 for **subnet1 = 23.1.17.x/26**
- For Subnet2 we have to support at least 90 interfaces and 2^7 >= 90 so the prefix for subnet2 is 32-7 = 25 , and so **subnet2 = 23.1.17.y/25**
- For Subnet3 we have to support at least 12 interfaces and 2^4 >= 12 so the prefix for subnet3 is 32-4 = 28 , and so **subnet3 = 23.1.17.z/28**
- Now find the values for x,y and z.
  - subnet 1 **23.1.17.0/26**
  - subnet 2 **23.1.17.128/25**
  - subnet 3 **23.1.17.64/28**

Suppose datagrams are limited to 1,500 bytes (including header) between source Host A and destination Host B. Assuming a 20-byte IP header, how many datagrams would be required to send an MP3 consisting of 5 million bytes? Explain how you computed your answer.

Given: IP Header size = 20 bytes
Datagram Size = 1500 bytes
We know: TCP Header size = 20 bytes
So to find the data contain in each datagram we need to deduct IP and TCP Header that is
1500 - 20 - 20 = 1460 bytes
Each datagram can carry maximum 1460 bytes.
So we number of datagrams required to send 5 million bytes =5000000 / 1460 = 3424.66
So we need **3425** datagrams to carry 5 million bytes.

The Services Provided by the Link Layer

- **Framing**: Almost all link-layer protocols encapsulate each network-layer datagram within a link-layer frame before transmission over the link. The structure of the frame is specified by the link-layer protocol.
- **Link access:** A medium access control (MAC) protocol specifies the rules by which a frame is transmitted onto the link. For point-to-point links that have a single sender at one end of the link and a single receiver at the other end of the link, the MAC protocol is simple the sender can send a frame whenever the link is idle. The more interesting case is when multiple nodes share a single broadcast link - the so-called multiple access problem.
- **Reliable delivery:** When a link-layer protocol provides reliable delivery service, it guarantees to move each network-layer datagram across the link without error. A link-layer reliable delivery service can be achieved with acknowledgments and retransmissions.
- **Error detection and correction:** The link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa. Such bit errors are introduced by signal attenuation and electromagnetic noise. Because there is no need to forward a datagram that has an error, many link-layer protocols provide a mechanism to detect such bit errors. This is done by having the transmitting node include error-detection bits in the frame, and having the receiving node perform an error check.

Where is the link layer implemented?

- It is implemented in each and every host.
- The link layer is implemented in a network adapter, also sometimes known as a network interface card (NIC).

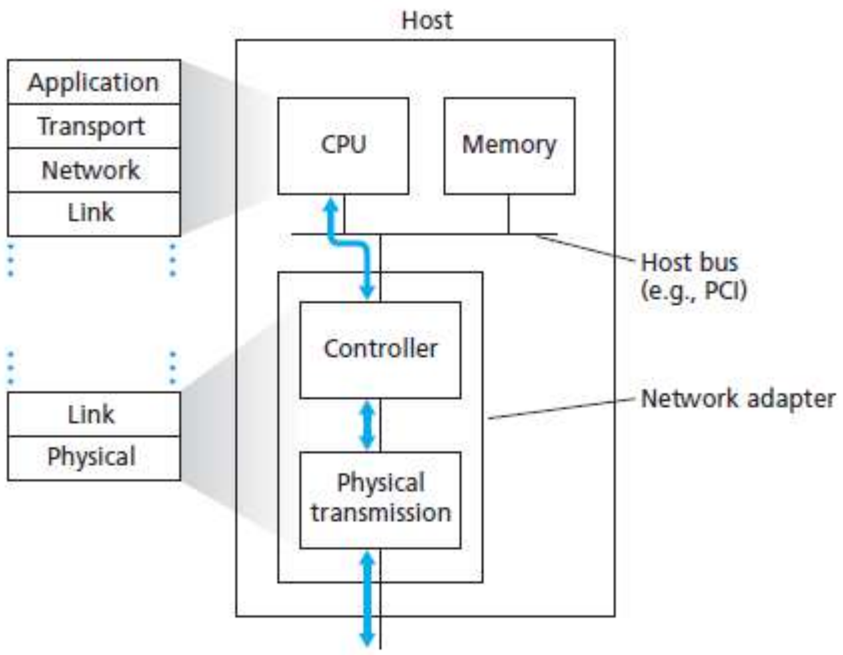


Fig. 1: Network adapter: its relationship to other host components and to protocol stack functionality

- Figure 1 shows a network adapter attaching to a host's bus (e.g., a PCI or PCI-X bus), where it looks much like any other I/O device to the other host components.
- Figure 1 also shows that while most of the link layer is implemented in hardware, part of the link layer is implemented in software that runs on the host's CPU.

- The software components of the link layer implement higher-level link layer functionality such as assembling link-layer addressing information and activating the controller hardware.
- On the receiving side, link-layer software responds to controller interrupts (e.g., due to the receipt of one or more frames), handling error conditions and passing a datagram up to the network layer.
- Thus, the link layer is a combination of hardware and software-the place in the protocol stack where software meets hardware.

Error Detection and Correction Techniques

- Techniques for error detection
    1. parity checks
    2. checksum methods
    3. cyclic redundancy checks

Parity checks

- In this technique, a redundant bit called parity bit, is appended to every data unit so that the number of 1s in the unit including the parity becomes even.
- Blocks of data from the source are subjected to a check bit or Parity bit generator form, where a parity of 1 is added to the block if it contains an odd number of 1's and 0 is added if it contains an even number of 1's.
- At the receiving end the parity bit is computed from the received data bits and compared with the received parity bit.
- This scheme makes the total number of 1's even, that is why it is called even parity checking.

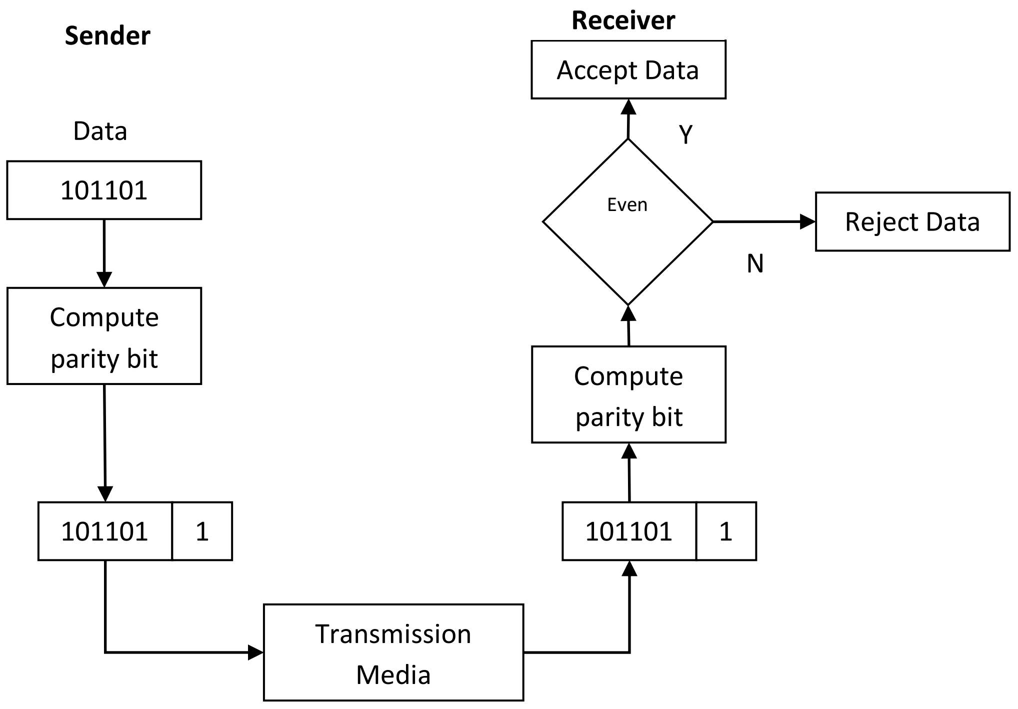


Fig. 2: Even parity checking scheme

Performance

- A receiver can detect all single bit errors in each code word.
- Errors in more than one bit cannot be detected.

Two-dimension Parity Check

- Performance can be improved by using two-dimensional parity check, which organizes the block of bits in the form of a table.

- Parity check bits are calculated for each row, which is equivalent to a simple parity check bit.
- Parity check bits are also calculated for all columns then both are sent along with the data.
- At the receiving end these are compared with the parity bits calculated on the received data.

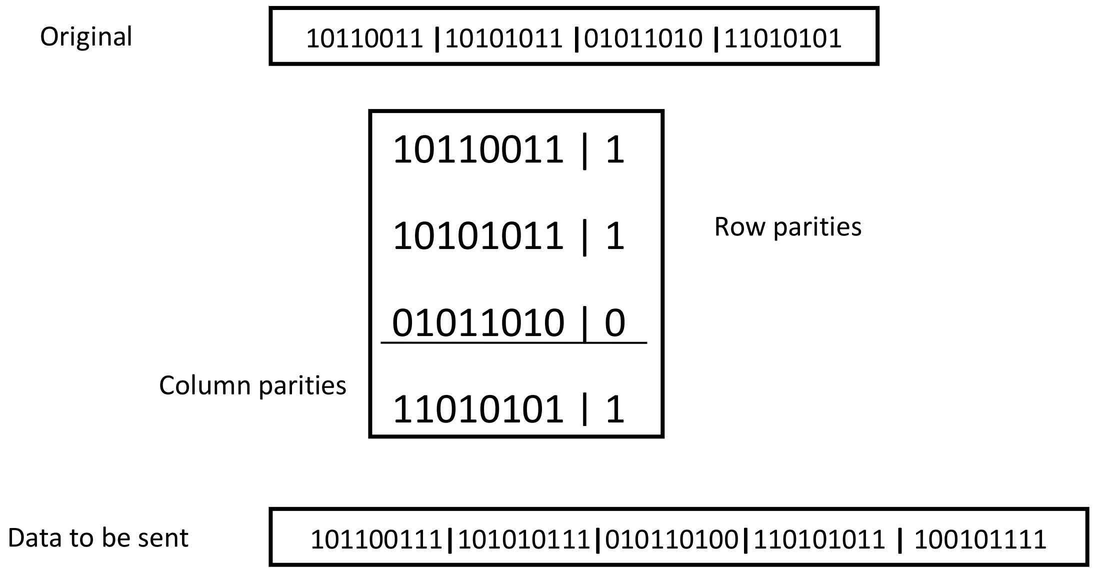


Fig. 3: Two-dimensional parity check

Performance

- Two- Dimension Parity Checking increases the likelihood of detecting burst errors.
- 2-D Parity check of n bits can detect a burst error of n bits.
- A burst error of more than n bits is also detected by 2-D Parity check with a high-probability.
- If two bits in one data unit are damaged and two bits in exactly same position in another data unit are also damaged, the 2-D Parity check checker will not detect an error.

Checksum

- Here, the data is divided into k segments each of m bits.
- In the sender's end the segments are added using 1's complement arithmetic to get the sum.
- The sum is complemented to get the checksum.
- The checksum segment is sent along with the data segments.
- At the receiver's end, all received segments are added using 1's complement arithmetic to get the sum. The sum is complemented.
- If the result is zero, the received data is accepted; otherwise discarded.

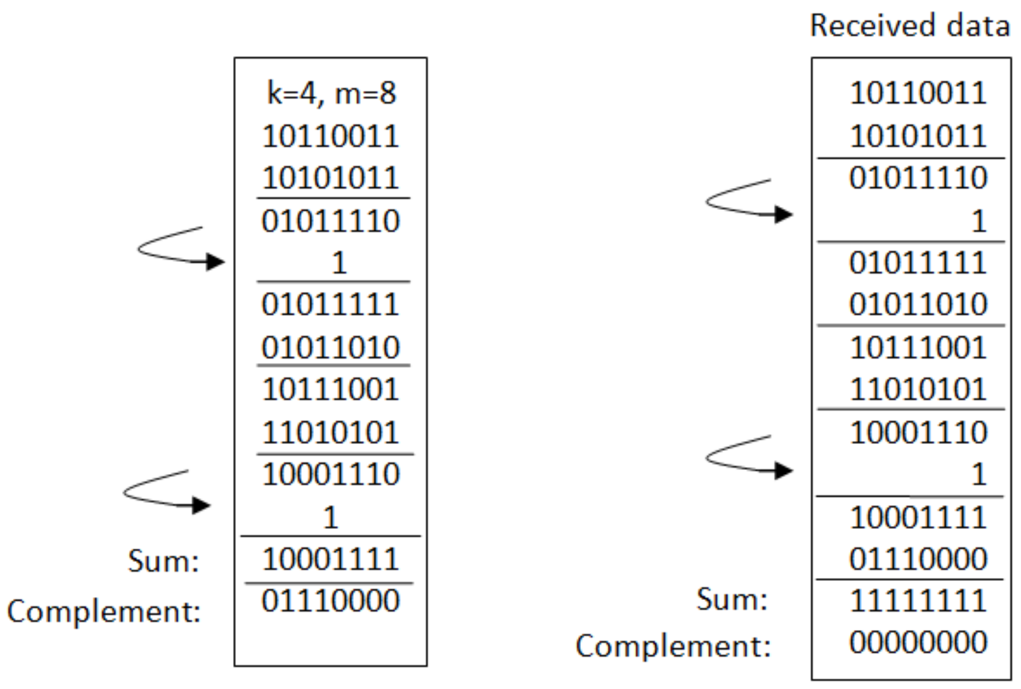


Fig. 4: Checksum

Performance

- The checksum detects all errors involving an odd number of bits.
- It also detects most errors involving even number of bits.

Cyclic Redundancy Checks (CRC)

- CRC is the most powerful and easy to implement technique.
- CRC is based on *binary division*.
- In CRC, a sequence of redundant bits, are appended to the end of data unit so that the resulting data unit becomes exactly divisible by a second, predetermined binary number.
- At the destination, the incoming data unit is divided by the same number.
- If at this step there is no remainder, the data unit is assumed to be correct and is therefore accepted.
- A remainder indicates that the data unit has been damaged in transit and therefore must be rejected.
- The binary number, which is (r+1) bit in length, can also be considered as the coefficients of a polynomial, called *Generator Polynomial*.

Performance

- CRC is a very effective error detection technique.
- If the divisor is chosen according to the previously mentioned rules, its performance can be summarized as follows
- CRC can detect all single-bit errors
- CRC can detect all double-bit errors (three 1's)
- CRC can detect any odd number of errors (X+1)
- CRC can detect all burst errors of less than the degree of the polynomial.

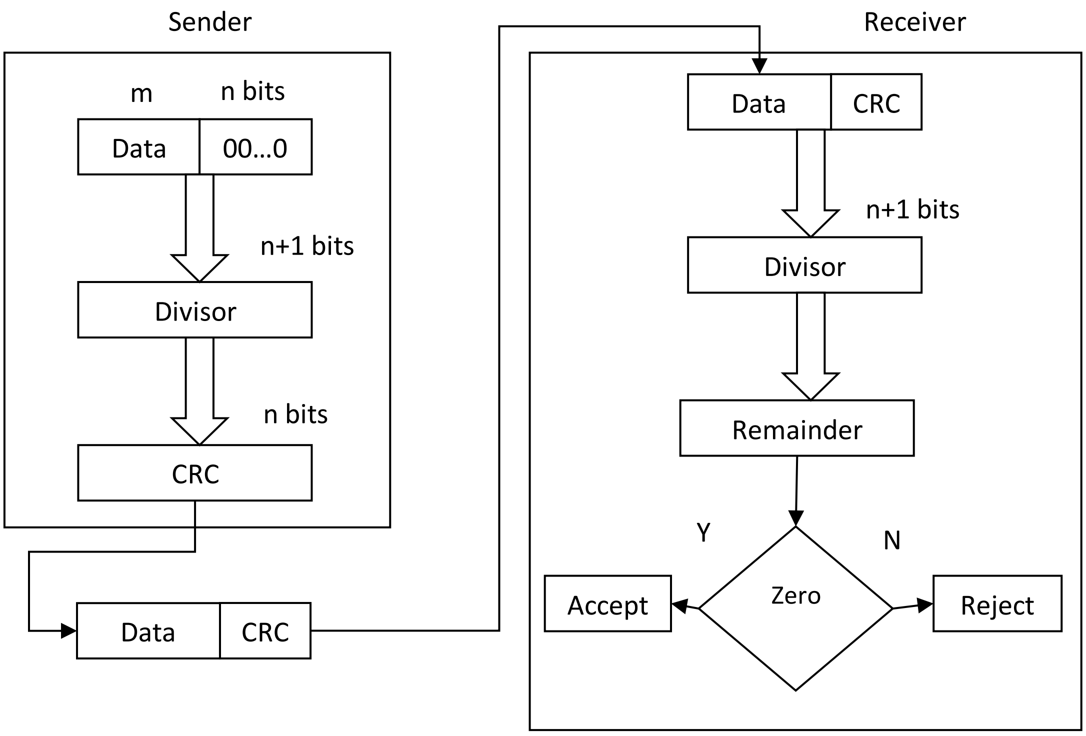


Fig. 5: Basic scheme for Cyclic Redundancy Check

Frame : 1 1 0 1 0 1 1 0 1 1

Generator: 1 0 0 1 1

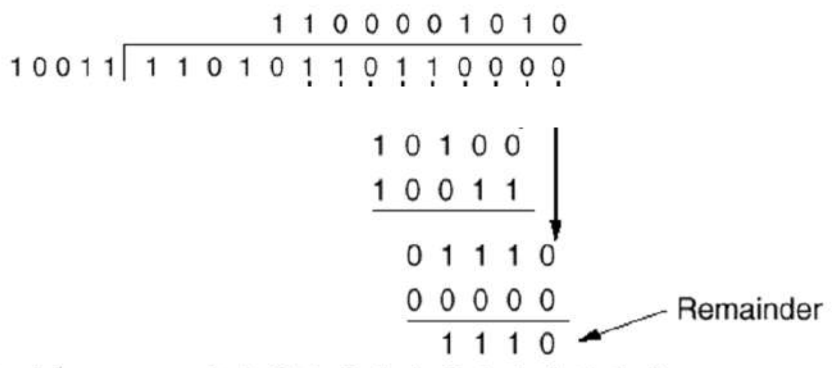
Message after appending 4 zero bits: 1 1 0 1 0 1 1 0 0 0 0

```
                  1 1 0 0 0 0 1 0 1 0
         ____________________________
1 0 0 1 1| 1 1 0 1 0 1 1 0 1 1 0 0 0 0
           1 0 0 1 1
           ---------
             1 0 0 1 1
             1 0 0 1 1
             ---------
               0 0 0 0 1
               0 0 0 0 0
               ---------
```

Frame : 1 1 0 1 0 1 1 0 1 1

Generator: 1 0 0 1 1

Message after appending 4 zero bits: 1 1 0 1 0 1 1 0 0 0 0



Transmitted frame: 1 1 0 1 0 1 1 0 1 1 1 1 1 0

Multiple access links and protocols

- There are two types of network links:
    1. A **point-to-point link** consists of a single sender at one end of the link and a single receiver at the other end of the link.
    2. A **broadcast link**, can have multiple sending and receiving nodes all connected to the same, single, shared broadcast channel. The term broadcast is used here because when any one node transmits a frame, the channel broadcasts the frame and each of the other nodes receives a copy.
- **Multiple access problem**: How to coordinate the access of multiple sending and receiving to a shared broadcast channel.
- In broadcast link all nodes are capable of transmitting frames, more than two nodes can transmit frames at the same time.
- When this happens, all of the nodes receive multiple frames at the same time; that is, the transmitted frames collide at all of the receivers.
- When there is a collision, none of the receiving nodes can make any sense of any of the frames that were transmitted; in a sense, the signals of the colliding frames become inextricably tangled together.
- Thus, all the frames involved in the collision are lost, and the broadcast channel is wasted during the collision interval.
- If many nodes want to transmit frames frequently, many transmissions will result in collisions, and much of the bandwidth of the broadcast channel will be wasted.
- In order to ensure that the broadcast channel performs useful work when multiple nodes are active, it is necessary to somehow coordinate the transmissions of the active nodes.
- This coordination job is the responsibility of the **multiple access protocol**.

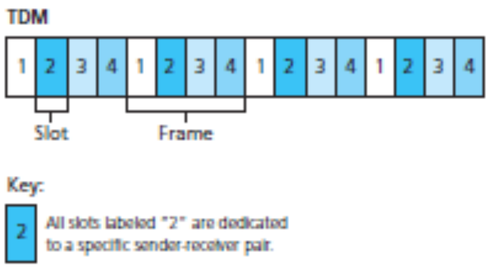
Multiple Access Protocols

- Categories of Multiple Access Protocol:
    1. channel partitioning protocols
        - ➢ divide channel into smaller “pieces” (time slots, frequency, code)
        - ➢ allocate piece to node for exclusive use
        - ➢ Examples of channel partitioning protocols
            1. TDMA: Time Division Multiple Access
            2. FDMA: Frequency Division Multiple Access

3. CDMA: Code Division Multiple Access
2. random access protocols
   - channel not divided and allow collisions
   - "recover" from collisions
   - examples of random access MAC (Medium Access Control) protocols
     1. ALOHA
     2. slotted ALOHA
     3. CSMA, CSMA/CD, CSMA/CA
3. taking-turns protocols
   - nodes take turns but nodes with more to send can take longer turns
   - examples of taking-turns protocols
     1. polling
     2. token passing

TDMA: Time Division Multiple Access

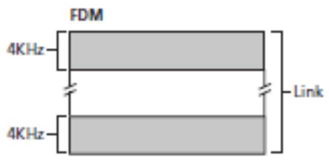
- Suppose the channel supports N nodes and that the transmission rate of the channel is R bps.
- TDM divides time into time frames and further divides each time frame into N time slots.
- Each time slot is then assigned to one of the N nodes. Whenever a node has a packet to send, it transmits the packet's bits during its assigned time slot in the revolving TDM frame.
- Typically, slot sizes are chosen so that a single packet can be transmitted during a slot time.


TDM
1
2
3
4
Slot
Frame
Key:
2
All slots labeled "2" are dedicated to a specific sender-receiver pair.

- TDM is attractive because it eliminates collisions and is perfectly fair: Each node gets a dedicated transmission rate of R/N bps during each frame time.
- However, it has two major drawbacks. First, a node is limited to an average rate of R/N bps even when it is the only node with packets to send.
- A second drawback is that a node must always wait for its turn in the transmission sequence again, even when it is the only node with a frame to send.

FDMA: Frequency Division Multiple Access

- FDM divides the R bps channel into different frequencies (each with a bandwidth of R/N) and assigns each frequency to one of the N nodes.
- FDM thus creates N smaller channels of R/N bps out of the single, larger R bps channel.
- FDM shares both the advantages and disadvantages of TDM.
- It avoids collisions and divides the bandwidth fairly among the N nodes.
- FDM also shares a principal disadvantage with TDM - a node is limited to a bandwidth of R/N, even when it is the only node with packets to send.



CDMA: Code Division Multiple Access

- While TDM and FDM assign time slots and frequencies, respectively to the nodes, CDMA assigns a different code to each node.
- Each node then uses its unique code to encode the data bits it sends.
- If the codes are chosen carefully, CDMA networks have the wonderful property that different nodes can transmit simultaneously and yet have their respective receivers correctly receive a sender's encoded data bits in spite of interfering transmissions by other nodes.
- CDMA has been used in military systems for some time and now has widespread civilian use, particularly in cellular telephony.
- Because CDMA's use is so tightly tied to wireless channels.
- It will suffice to know that CDMA codes, like time slots in TDM and frequencies in FDM, can be allocated to the multiple access channel users.

Pure ALOHA and Slotted ALOHA Protocol

Pure ALOHA

- Pure ALOHA allows users to transmit whenever they have data to be sent.
- Senders wait to see if a collision occurred (after whole message has been sent).
- If collision occurs, each station involved waits a **random amount of time** then tries again.
- Systems in which multiple users share a common channel in a way that can lead to conflicts are widely known as **contention systems**.

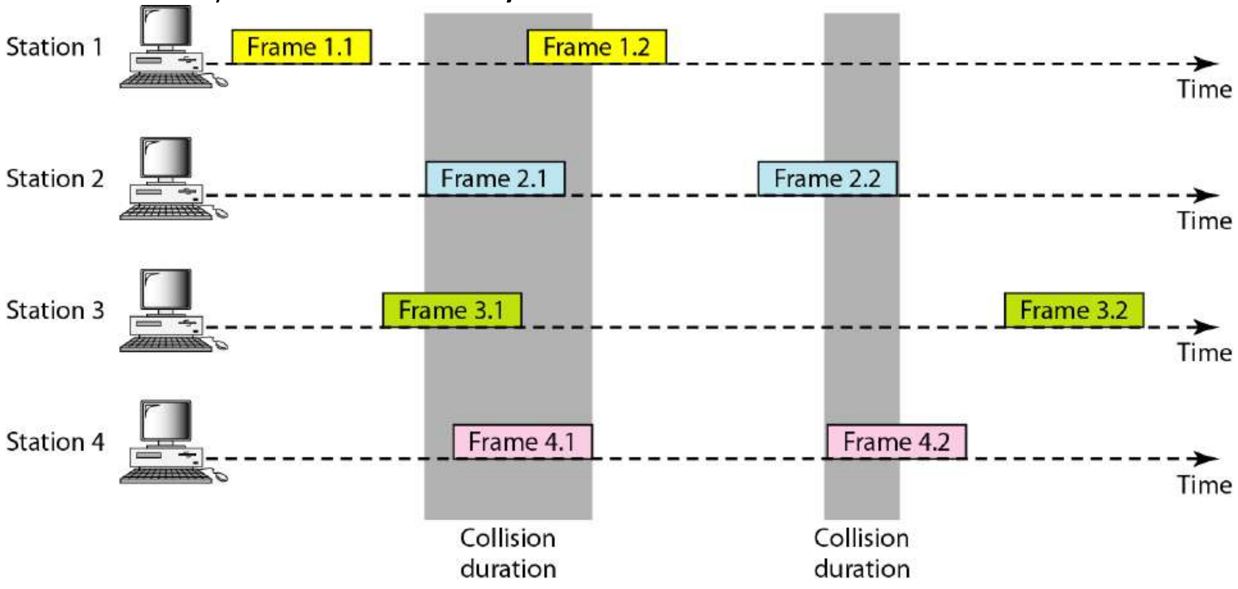


Fig. 6: In pure ALOHA, frames are transmitted at completely arbitrary times

- Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled.
- If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed and both will have to be retransmitted later.

What is the efficiency of an ALOHA channel?

- Let the **"frame time"** denote the amount of time needed to transmit the standard, fixed-length frame.
- Also assume that the infinite population of users generates new frames with mean N frames per frame time.

- If N > 1, the user community is generating frames at a higher rate than the channel can handle, and nearly every frame will suffer a collision. For reasonable throughput we would expect 0 < N < 1.
- Let us further assume that the probability of k transmission attempts per frame time, old and new combined, is also Poisson, with mean G per frame time.
- Under all loads, the throughput, S, is just the offered load, G, times the probability, P_0, of a transmission succeeding-that is, $S = GP_0$, where P_0 is the probability that a frame does not suffer a collision.
- The probability that k frames are generated during a given frame time is given by the Poisson distribution: $\Pr[k] = \frac{G^k e^{-G}}{k!}$
- The maximum throughput occurs at G = 0.5, with S = 1/2e, which is about **0.184**.

Slotted ALOHA

- Slotted ALOHA was invented to improve the efficiency of pure ALOHA as chances of collision in pure ALOHA are very high.
- In 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.

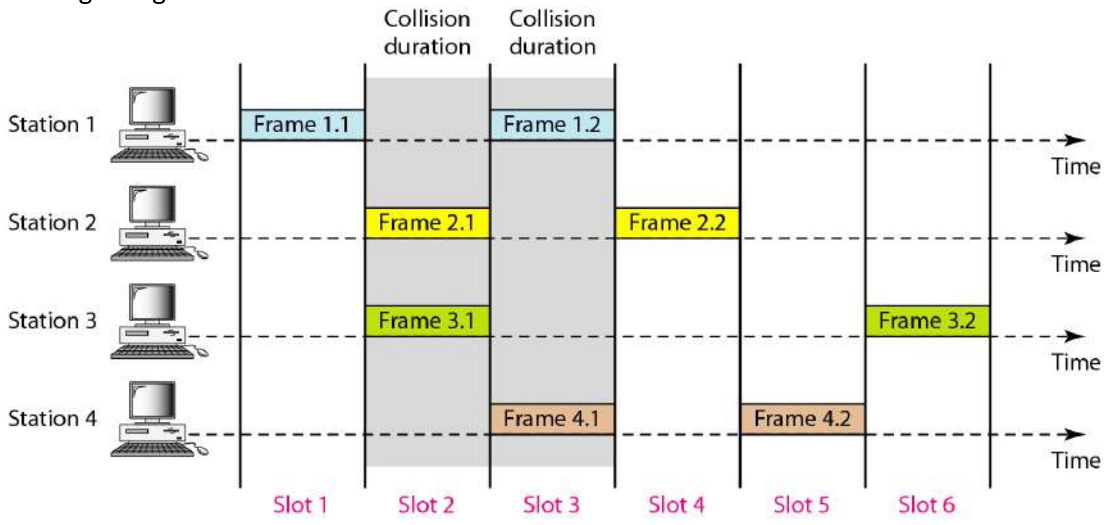


Fig. 7: Frames transmitted in Slotted ALOHA

What is the efficiency of an ALOHA channel?

- Pure ALOHA → $S = Ge^{-2G}$
- Slotted ALOHA → $S = Ge^{-G}$

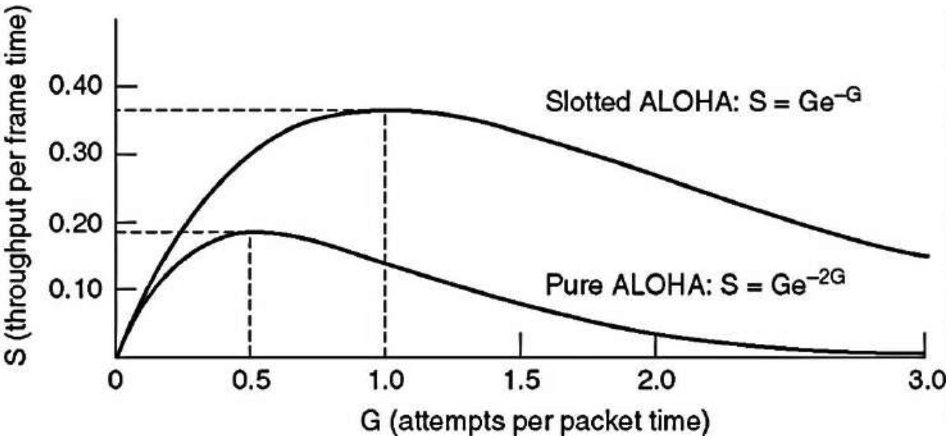


Fig. 8: Throughput versus offered traffic for ALOHA systems

- As seen in figure both graphs have the same shape. If G is small so is S, which means that if few frames are generated few frames will be transmitted successfully.
- As G increases so does S but up to a certain point. As G continues to increase S approaches to 0 which means that if more frames are generated there will be more collisions and the success rate will fall to 0.
- Similarly for pure ALOHA the maximum occurs at G=0.5 for which S = 1/2e = 0.184 which means the rate of successful transmissions is approximately 18.4%.
- As seen from the graph the maximum for slotted ALOHA occurs at G=1 for which S=1/e=0.368. In other words the rate of successful transmissions is approximately 0.368 frames per slot time or 37% of the time will be spent on successful transmissions.

Advantages of Slotted Aloha

- A single active node can continuously transmit at full rate of channel.
- Slotted ALOHA is also highly decentralized, because each node detects collisions and independently decides when to retransmit.
- Slotted ALOHA is also an extremely simple protocol.

Disadvantages of Slotted Aloha

- When there are multiple active nodes, a certain fraction of the slots will have collisions and will therefore be wasted.
- idle slots
- nodes may be able to detect collision in less than time to transmit packet
- clock synchronization

CSMA (Carrier Sense Multiple Access) Protocols

- Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called carrier sense protocols.
- There are various CSMA protocol:

1. 1-persistent CSMA
2. Non persistent CSMA
3. p-persistent CSMA
4. CSMA/CD (CSMA with Collision Detection)

1-persistent CSMA

- When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment.
- If the channel is busy, the station waits until it becomes idle.
- When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again.
- The protocol is called 1-persistent because the station transmits with a probability of 1 when it finds the channel idle.
- The propagation delay has an important effect on the performance of the protocol.
- There is a small chance that just after a station begins sending, another station will become ready to send and sense the channel.
- If the first station's signal has not yet reached the second one, the latter will sense an idle channel and will also begin sending, resulting in a collision.
- The longer the propagation delay, the more important this effect becomes, and the worse the performance of the protocol.
- Even if the propagation delay is zero, there will still be collisions.
- If two stations become ready in the middle of a third station's transmission, both will wait politely until the transmission ends and then both will begin transmitting exactly simultaneously, resulting in a collision.
- If they were not so impatient, there would be fewer collisions.

Non persistent CSMA

- In this protocol, a conscious attempt is made to be less greedy than in the previous one.
- Before sending, a station senses the channel. If no one else is sending, the station begins doing so itself.
- However, if the channel is already in use, the station does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission.
- Instead, it waits a random period of time and then repeats the algorithm. Consequently, this algorithm leads to better channel utilization but longer delays than 1-persistent CSMA

P-persistent CSMA

- It applies to slotted channels.
- When a station becomes ready to send, it senses the channel.
- If it is idle, it transmits with a probability p.
- With a probability $q=1-p$, it defers until the next slot.
- If that slot is also idle, it either transmits or defers again, with probabilities p and q.
- This process is repeated until either the frame has been transmitted or another station has begun transmitting.
- In the latter case, the unlucky station acts as if there had been a collision (i.e., it waits a random time and starts again).
- If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm.
- Figure shows the computed throughput versus offered traffic for all three protocols.

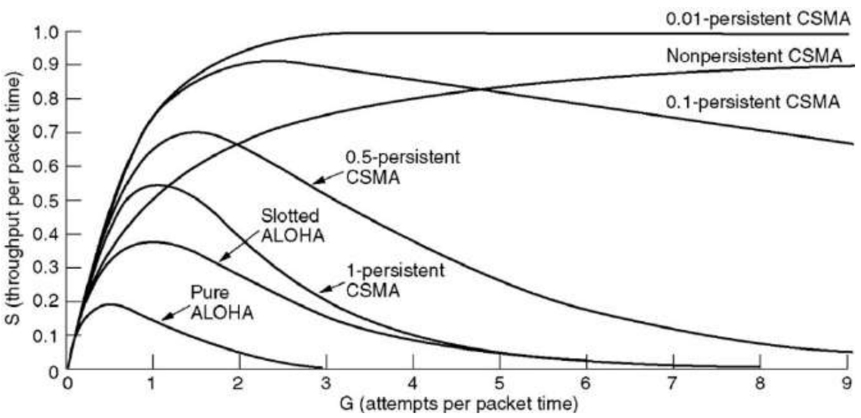


Fig. 9: Comparison of the channel utilization versus load for various random access protocols

CSMA/CD (CSMA with Collision Detection)

- If two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately.
- Rather than finish transmitting their frames, which are irretrievably garbled anyway, they should abruptly stop transmitting as soon as the collision is detected.
- Quickly terminating damaged frames saves time and bandwidth.
- This protocol, known as CSMA/CD (CSMA with Collision Detection) is widely used on LANs in the MAC sublayer.

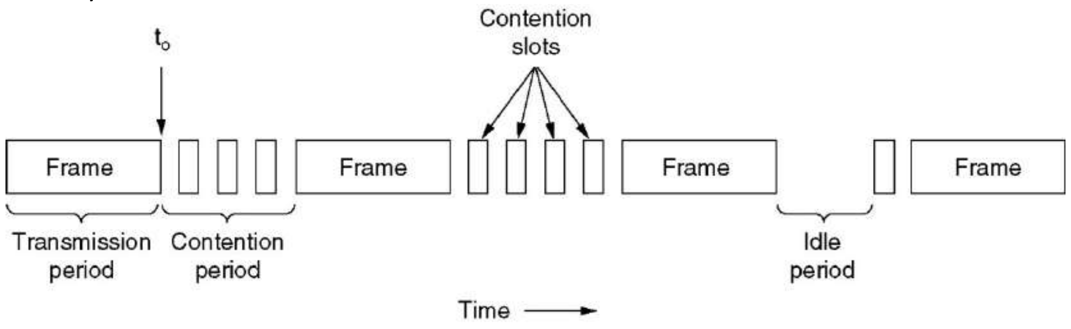


Fig. 10: CSMA/CD can be in one of three states: contention, transmission, or idle

- CSMA/CD, as well as many other LAN protocols, uses the conceptual model of Figure.
- At the point marked t_0, a station has finished transmitting its frame.
- Any other station having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision.
- Collisions can be detected by looking at the power or pulse width of the received signal and comparing it to the transmitted signal.
- After a station detects a collision, it aborts its transmission, waits a random period of time, and then tries again, assuming that no other station has started transmitting in the meantime.
- Therefore, model for CSMA/CD will consist of alternating contention and transmission periods, with idle periods occurring when all stations are quiet.

Taking turns MAC protocols

Polling

- The polling protocol requires one of the nodes to be designated as a master node.
- The master node polls each of the nodes in a round-robin fashion.

- In particular, the master node first sends a message to node 1, saying that it (node 1) can transmit up to some maximum number of frames.
- After node 1 transmits some frames, the master node tells node 2 it (node 2) can transmit up to the maximum number of frames.
- The master node can determine when a node has finished sending its frames by observing the lack of a signal on the channel.
- The procedure continues in this manner, with the master node polling each of the nodes in a cyclic manner.
- The polling protocol eliminates the collisions and empty slots that plague random access protocols.
- This allows polling to achieve a much higher efficiency.
- The first drawback is that the protocol introduces a polling delay—the amount of time required to notify a node that it can transmit.
- The second drawback, which is potentially more serious, is that if the master node fails, the entire channel becomes inoperative.

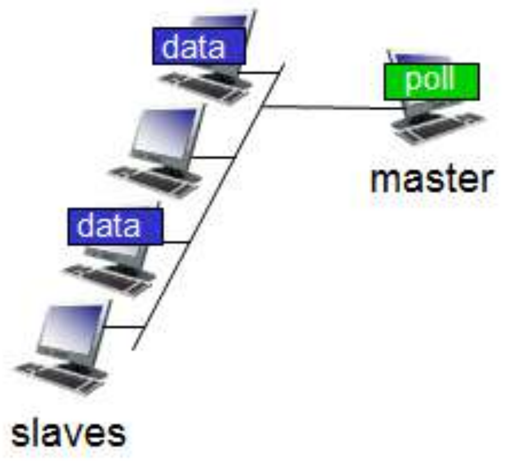


Fig. 11: Polling

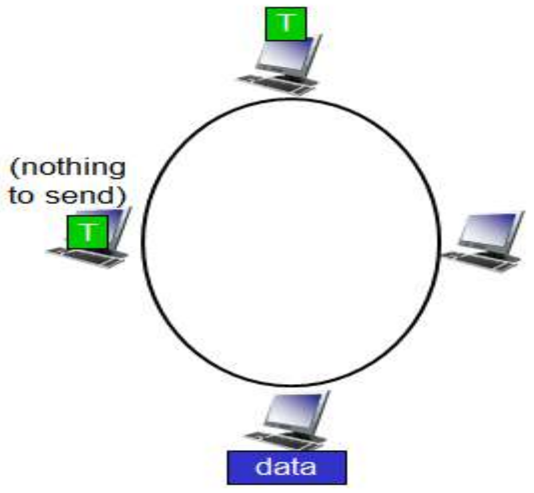


Fig. 12: Token Passing

Token Passing

- In taking-turns protocol there is no master node.
- A small, special-purpose frame known as a token is exchanged among the nodes in some fixed order.
- For example, node 1 might always send the token to node 2, node 2 might always send the token to node 3, and node N might always send the token to node 1.
- When a node receives a token, it holds onto the token only if it has some frames to transmit; otherwise, it immediately forwards the token to the next node.
- If a node does have frames to transmit when it receives the token, it sends up to a maximum number of frames and then forwards the token to the next node.
- Token passing is decentralized and highly efficient. But it has its problems as well. For example, the failure of one node can crash the entire channel. Or if a node accidentally neglects to release the token, then some recovery procedure must be invoked to get the token back in circulation.

Ethernet

- **Ethernet** is one of the widely used local area network (LAN) technology.
    1. Switched Ethernet
    2. Fast Ethernet
    3. Gigabit Ethernet

Switched Ethernet

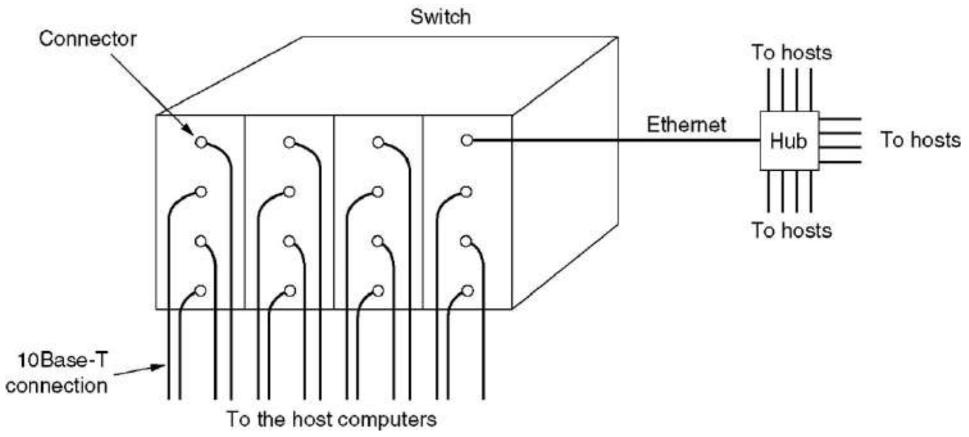


Fig. 1: A simple example of switched Ethernet

- Switched Ethernet gives dedicated 10 Mbps bandwidth on each of its ports.
- On each of the ports one can connect either a thick/thin segment or a computer.
- In Switched Ethernet, the collision domain is separated.
- The hub is replaced by a switch, which functions as a fast bridge.
- It can recognize the destination address of the received frame and can forward the frame to the port to which the destination station is connected.
- The other ports are not involved in the transmission process.
- The switch can receive another frame from another station at the same time and can route this frame to its own final destination.
- In this case, both the physical and logical topologies are star.
- The throughput can be further increased on switched Ethernet by using full-duplex technique, which uses separate wire pairs for transmitting and receiving.
- Thus a station can transmit and receive simultaneously, effectively doubling the throughput to 20 Mbps on each port.

Fast Ethernet

- The 802.u or the fast Ethernet, as it is commonly known, was approved by the IEEE 802 Committee.
- The fast Ethernet uses the same frame format, same CSMA/CD protocol and same interface as the 802.3, but uses a data transfer rate of 100 Mbps instead of 10 Mbps.
- However, fast Ethernet is based entirely on 10-Base-T, because of its advantages (Although technically 10-BASE-5 or 10-BASE-2 can be used with shorter segment length).
- IEEE has designed two categories of Fast Ethernet: 100Base-X and 100Base-T4.
- 100Base-X uses two-wire interface between a hub and a station while 100Base-T4 uses four-wire interface.
- 100-Base-X itself is divided into two: 100Base-TX and 100base-FX

Name	Cable	Max. segment	Advantages
100Base-T4	Twisted pair	100 m	Uses category 3 UTP
100Base-TX	Twisted pair	100 m	Full duplex at 100 Mbps
100Base-FX	Fiber optics	2000 m	Full duplex at 100 Mbps; long runs

- Upgrade the data rate to 100 Mbps.
- Make it compatible with Standard Ethernet.
- Keep the same 48-bit address.
- Keep the same frame format.
- Keep the same minimum and maximum frame lengths.

Gigabyte Ethernet

- The technology is based on fiber optic cable. Multi-mode fiber is able to transmit at gigabit rate to at least 580 meters and with single-mode runs exceeding 3 km.
- Fiber optic cabling is costly. In order to reduce the cost of cabling, the 802.3z working group also proposed the use of twisted pair or cable or coaxial cable for distances up to 30 meters.
- At gigabit speed, two stations 200 meters apart will not detect a collision, when both simultaneously send 64-byte frames.
- This inability to detect collision leads to network instability.
- A mechanism known as *carrier extension* has been proposed for frames shorter than 512 bytes.
- The number of repeater hops is also restricted to only one in place of two for 100 Base-T.

Name	Cable	Max. segment	Advantages
1000Base-SX	Fiber optics	550 m	Multimode fiber (50, 62.5 microns)
1000Base-LX	Fiber optics	5000 m	Single (10 μ) or multimode (50, 62.5 μ)
1000Base-CX	2 Pairs of STP	25 m	Shielded twisted pair
1000Base-T	4 Pairs of UTP	100 m	Standard category 5 UTP

- Upgrade the data rate to 1 gbps.
- Make it compatible with Standard or Fast Ethernet.
- Use the same 48-bit address.
- Use the same frame format.
- Keep the same minimum and maximum frame lengths.
- To support auto-negotiation as defined in Fast Ethernet.

Virtual LANs

- A VLAN is a switched network that is logically segmented by functions, project teams, or applications without regard to the physical location of users.
- For example, several end stations might be grouped as a department, such as engineering or accounting.
- When the end stations are physically located close to one another, you can group them into a LAN segment.
- If any of the end stations are in different buildings (not the same physical LAN segment), you can then group them into a VLAN.

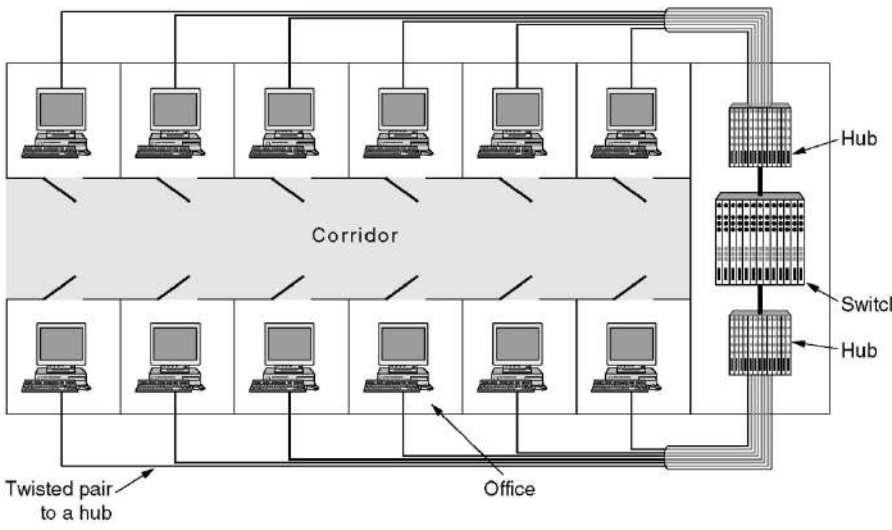


Fig. 14: VLAN

- VLANs provide the following features:

Simplification of end-station moves, adds and changes

- When an end station is physically moved to a new location, its attributes can be reassigned from a network management station through Simple Network Management Protocol (SNMP) or through the user interface menus.
- When an end station is moved within the same VLAN, it retains its previously assigned attributes in its new location. When an end station is moved to a different VLAN, the attributes of the new VLAN are applied to the end station.

Controlled traffic activity

- VLANs allow ports on the same or different switches to be grouped so that traffic is confined to members of only that group.
- This feature restricts broadcast, unicast, and multicast traffic (flooding) only to ports included in a certain VLAN.
- The management domain is a group of VLANs that are managed by a single administrative authority.

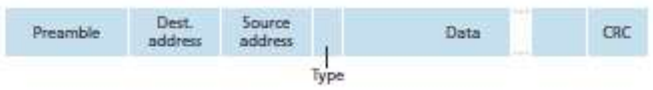
Workgroup and network security

- You can increase security by segmenting the network into distinct broadcast domains.
- To this end, VLANs can restrict the number of users in a broadcast domain.
- You can also control the size and composition of the broadcast domain by controlling the size and composition of a VLAN.

Components

- Networks that have VLANs contain one or more of the following components:
  - Switches that logically segment connected end stations
  - Routers that provide VLAN communications between workgroups
  - Transport protocols that carry VLAN traffic across shared LAN and ATM backbones
  - Interoperability with previously installed LAN systems

Ethernet frame structure

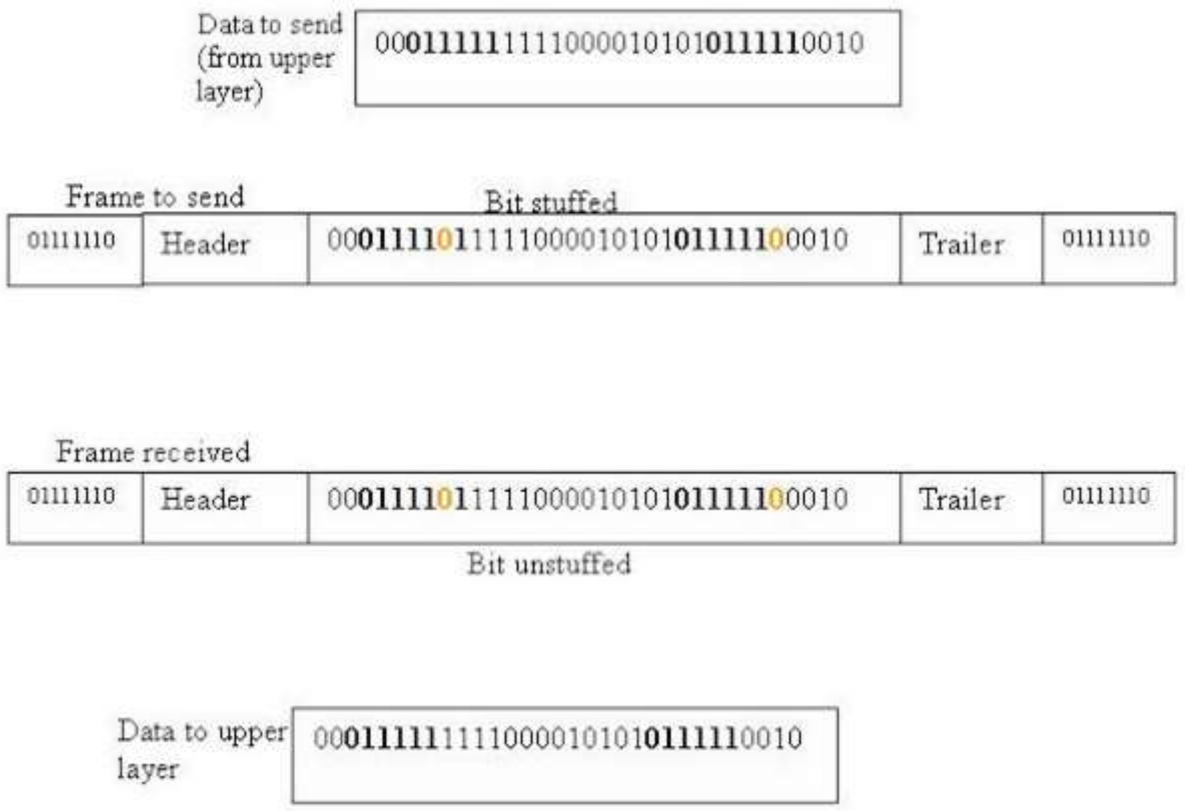


- **Data field (46 to 1,500 bytes):** This field carries the IP datagram. The maximum transmission unit (MTU) of Ethernet is 1,500 bytes. This means that if the IP datagram exceeds 1,500 bytes, then the host has to fragment the datagram. The minimum size of the data field is 46 bytes. This means that if the IP datagram is less than 46 bytes, the data field has to be "stuffed" to fill it out to 46 bytes. When stuffing is used, the data passed to the network layer contains the stuffing as well as an IP datagram.
- **Destination address (6 bytes):** This field contains the MAC address of the destination adapter, BB-BB-BB-BB-BB-BB. When adapter B receives an Ethernet frame whose destination address is either BB-BB-BB-BB-BB-BB or the MAC broadcast address, it passes the contents of the frame's data field to the network layer; if it receives a frame with any other MAC address, it discards the frame.
- **Sources address (6 bytes):** This field contains the MAC address of the adapter that transmits the frame onto the LAN, in this example, AA-AA-AA-AA-AA-AA.
- **Type field (2 bytes):** The type field permits Ethernet to multiplex network-layer protocols. To understand this, we need to keep in mind that hosts can use other network-layer protocols besides IP. In fact, a given host may support multiple network-layer protocols using different protocols for different applications. For this reason, when the Ethernet frame arrives at adapter B, adapter B needs to know to which network-layer protocol it should pass (that is, demultiplex) the contents of the data field.
- **Cyclic redundancy checks (CRC) (4 bytes):** The purpose of the CRC field is to allow the receiving adapter, adapter B, to detect bit errors in the frame.
- **Preamble (8 bytes)**: The Ethernet frame begins with an 8-byte preamble field. Each of the first 7 bytes of the preamble has a value of 10101010; the last byte is 10101011. The first 7 bytes of the preamble serve to "wake up" the receiving adapters and to synchronize their clocks to that of the sender's clock. Why should the clocks be out of synchronization? Keep in mind that adapter A aims to transmit the frame at 10 Mbps, 100 Mbps, or 1 Gbps, depending on the type of Ethernet LAN. However, because nothing is absolutely perfect, adapter A will not transmit the frame at exactly the target rate; there will always be some drift from the target rate, a drift which is not known a priori by the other adapters on the LAN. A receiving adapter can lock onto adapter A's clock simply by locking onto the bits in the first 7 bytes of the preamble. The last 2 bits of the eighth byte of the preamble (the first two consecutive 1s) alert adapter B that the "important stuff" is about to come.

Bit and Byte Stuffing

Bit Stuffing

- In a bit-oriented protocol, the data to send is a series of bits.
- In order to distinguish frames, most protocols use a bit pattern of 8-bit length (01111110) as flag at the beginning and end of each frame.
- Here also cause the problem of appearance of flag in the data part to deal with this an extra bit added.
- This method is called bit stuffing. In bit stuffing, if a 0 and five successive 1 bits are encountered, an extra 0 is added.
- The receiver node removes the extra-added zero.
- This process is illustrate below,

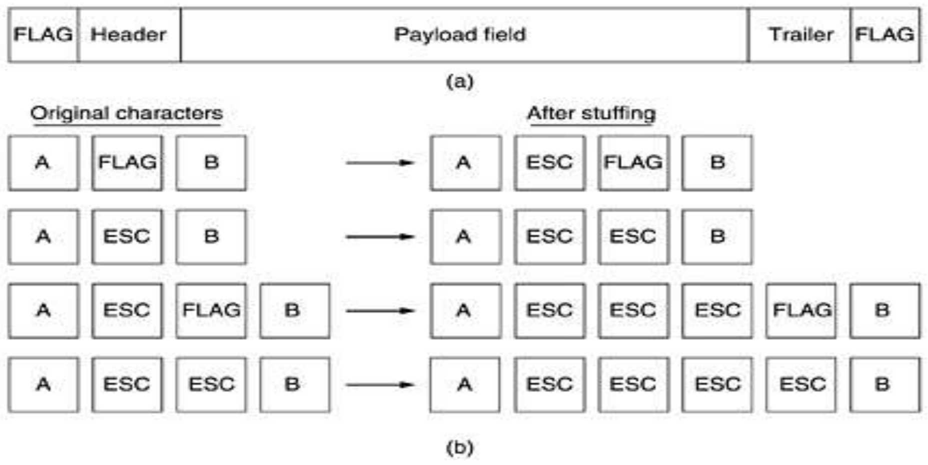


Byte Stuffing

- The second framing solves the problem of resynchronization by having each frame start and end with special bytes.
- A flag byte is used to separate the frame as both the starting and ending delimiter, as shown in figure.
- In this way, if the receiver ever loses synchronization, it can just search for the flag byte to find the end of the current frame.
- Two consecutive flag bytes indicate the end of one frame and start of the next one.
- It may easily happen that the flag byte's bit pattern occurs in the data.
- To solve this problem is to have the sender's data link layer insert a special escape byte (ESC) just before each "accidental" flag byte in the data.
- The data link layer on the receiving end removes the escape byte before the data are given to the network layer.
- This technique is called *byte stuffing* or *character stuffing*.

Disadvantage of using this framing method is that it is closely tied to the use of 8-bit characters.

- Not all character codes use 8-bit characters.
- E.g., UNICODE uses 16-bit characters



Self-Learning Properties of Link Layer Switches

- A switch has the wonderful property that its table is built automatically, dynamically, and autonomously, without any intervention from a network administrator or from a configuration protocol. In other words, switches are self-learning.
- This capability is accomplished as follows:
  1. The switch table is initially empty.
  2. For each incoming frame received on an interface, the switch stores in its table
     - the MAC address in the frame's source address field,
     - the interface from which the frame arrived
     - the current time.

     In this manner the switch records in its table the LAN segment on which the sender resides. If every host in the LAN eventually sends a frame, then every host will eventually get recorded in the table.
  3. The switch deletes an address in the table if no frames are received with that address as the source address after some period of time (the aging time).
- In this manner, if a PC is replaced by another PC (with a different adapter), the MAC address of the original PC will eventually be purged from the switch table.
- Let's walk through the self-learning property for the uppermost switch in Figure 15.
- Suppose at time 9:39 a frame with source address 01-12-23-34-45-56 arrives from interface 2.
- Suppose that this address is not in the switch table. Then the switch adds a new entry to the table.
- Continuing with this same example, suppose that the aging time for this switch is 60 minutes, and no frames with source address 62-FE-F7-11-89-A3 arrive to the switch between 9:32 and 10:32. Then at time 10:32, the switch removes this address from its table.

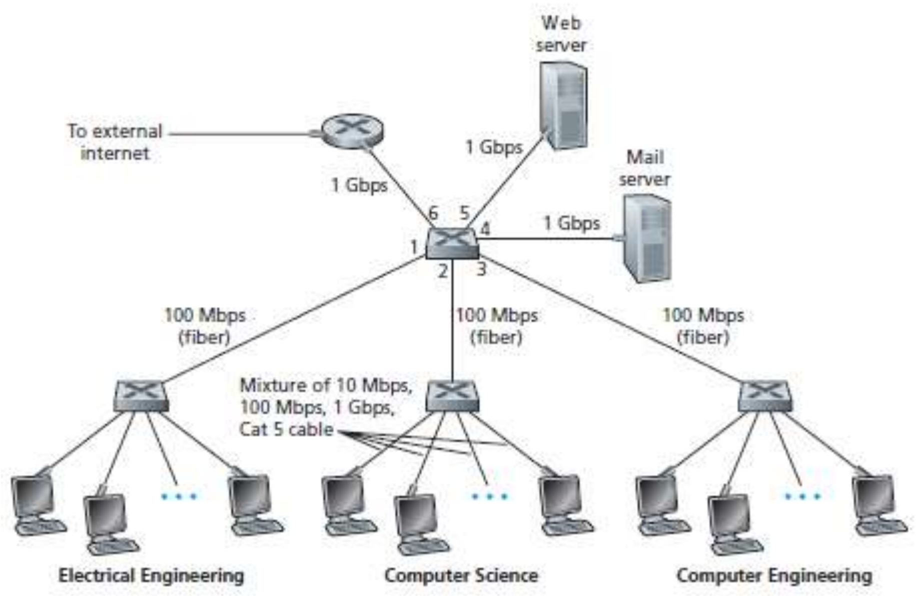


Fig. 15: An institutional network connected together by four switches

Address	Interface	Time
62-FE-F7-11-89-A3	1	9:32
7C-BA-B2-B4-91-10	3	9:36
....	....	....

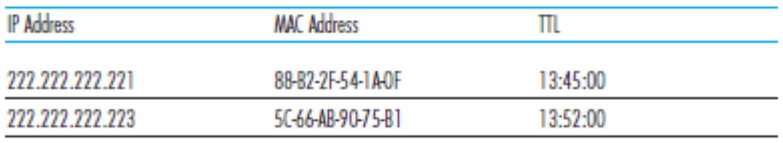
ARP (Address Resolution Protocol)

- Address Resolution Protocol (ARP) is a protocol for mapping an Internet Protocol address (IP address) to a physical machine address that is recognized in the local network.
- For example, in IP Version 4, the most common level of IP in use today, an address is 32 bits long.
- In an Ethernet local area network, however, addresses for attached devices are 48 bits long. (The physical machine address is also known as a Media Access Control or MAC address.)
- A table, usually called the ARP cache, is used to maintain a correlation between each MAC address and its corresponding IP address.
- ARP provides the protocol rules for making this correlation and providing address conversion in both directions.

How ARP works

- When an incoming packet destined for a host machine on a particular local area network arrives at a gateway, the gateway asks the ARP program to find a physical host or MAC address that matches the IP address.
- The ARP program looks in the ARP cache and, if it finds the address, provides it so that the packet can be converted to the right packet length and format and sent to the machine.
- If no entry is found for the IP address, ARP broadcasts a request packet in a special format to all the machines on the LAN to see if one machine knows that it has that IP address associated with it.
- A machine that recognizes the IP address as its own returns a reply so indicating.

- ARP updates the ARP cache for future reference and then sends the packet to the MAC address that replied.
- A possible ARP table in any node is as per below



IP Address	MAC Address	TTL
222.222.222.221	88-B2-2F-54-1A-0F	13:45:00
222.222.222.223	5C-66-AB-90-75-B1	13:52:00

- Since protocol details differ for each type of local area network, there are separate ARP Requests for Comments (RFC) for Ethernet, ATM, Fiber Distributed-Data Interface, HIPPI, and other protocols.