Bharat Ratna Puratchi Thalaivar Dr.MGR Government Arts and Science College, Palacode – 636808

B.Sc. COMPUTER SCIENCE SEMESTER

UNIT – I

Introduction - Network Hardware - Network Software - Reference Models: OSI Reference model. Physical Layer: Guided Transmission media - Wireless Transmission- Public Switched Telephone Network - The Mobile Telephone System.

UNIT – II

Data Link Layer: Data Link layer Design Issues - Error Detection and Correction - Elementary protocols - Sliding Window Protocols - MAC sub layer: Channel allocation problem - Multiple access protocols.

UNIT – III

Network Layer: Network Layer Design Issues – Routing Algorithms – Congestion control algorithm - Quality of service - Internetworking.

UNIT – IV

Transport Layer: Transport services – Elements of transport protocols - congestion control – Internet Transport protocol – UDP – TCP.

UNIT – V

Application layers: Domain name system – Electronic mail - The World Wide Web. Network Security: Cryptography – Symmetric, Public Key algorithms.

TEXT BOOK

1. David J.Wetherall, Andrew S.Tanenbaum,"Computer Networks", 5 th Edition, pearsonEducation, 2012.

REFERENCE BOOKS

1. B.A. Forouzan , " Data Communication and Networking ", 4 th Edition, Tata McGraw Hill,2007.

2. B.A. Forouzan, FirouzMosharraf,"Computer network – A Top down Approach", Tata McGraw Hill,2012.

3. A.Leon&M.Leon,"Introduction to Information Technology", 1th Edition, Vijay Nicole Publications,2013.

UNIT – I

INTRODUCTION

Each of the past three centuries has been dominated by a single technology.

- ☞ The 18th century was the era of the great mechanical systems accompanying the
- ☞ Industrial Revolution.
- ☞ The 19th century was the age of the steam engine.
- ☞ During the 20th century, the key technology was information gathering,
- ☞ processing, and distribution.

As a result of rapid technological progress, these areas are rapidly converging and the differences between collecting, transporting, storing, and processing information are quickly disappearing.

The idea that within twenty years equally powerful computers smaller than postage stamps would be mass produced by the millions was pure science fiction. The merging of computers and communications has had a profound influence on the way computer systems are organized. The concept of the "computer center" as a room with a large computer to which users bring their work for processing is now totally obsolete. The old model of a single computer serving all of the organization's computational needs has been replaced by one in which a large number of separate but interconnected computers do the job. These systems are called computer networks.

USES OF COMPUTER NETWORKS

Business Applications

Many companies have a substantial number of computers. Put in slightly more general form, the issue here is resource sharing, and the goal is to make all programs, equipment, and especially data available to anyone on the network without regard to the physical location of the resource and the user. An obvious and widespread example is having a group of office workers share a common printer. None of the individuals really needs a private printer, and a high-volume networked printer is often cheaper, faster, and easier to maintain than a large collection of individual printers

Home Applications

In 1977, Ken Olsen was president of the Digital Equipment Corporation, then the number two computer vendor in the world (after IBM). Probably the biggest reason now is for Internet access.
Some of the more popular uses of the Internet for home users are as follows:

- Access to remote information.
- Person-to-person communication.
- Interactive entertainment.
- Electronic commerce.

Mobile Users

Mobile computers, such as notebook computers and personal digital assistants (PDAs), are one of the fastest growing segments of the computer industry. Many owners of these computers have desktop machines back at the office and want to be connected to their home base even when away from home or en route. Since having a wired connection is impossible in cars and airplanes, there is a lot of interest in wireless networks.

Although wireless networking and mobile computing are often related, they are not identical. Even notebook computers are sometimes wired. For example, if a traveler plugs a notebook computer into the telephone jack in a hotel room, he has mobility without a wireless network.

Social Issues

The widespread introduction of networking has introduced new social, ethical, and political problems.. A popular feature of many networks is newsgroups or bulletin boards whereby people can exchange messages with like-minded individuals. As long as the subjects are restricted to technical topics or hobbies like gardening, not too many problems will arise.

NETWORK HARDWARE

There is no generally accepted taxonomy into which all computer networks fit, but two dimensions stand out as important: transmission technology and scale. Broadly speaking, there are two types of transmission technology that are in widespread use. They are as follows:

- Broadcast links.
- Point-to-point links.

Broadcast networks have a single communication channel that is shared by all the machines on the network. Short messages, called packets in certain contexts, sent by any machine are received by all the others. An address field within the packet specifies the intended recipient. Upon receiving a packet, a machine checks the address field. If the packet is intended for the receiving machine, that machine processes the packet; if the packet is intended for some other machine, it is just ignored.

PERSONAL AREA NETWORKS

PANs (Personal Area Networks) let devices communicate over the range of a person. A common example is a wireless network that connects a computer with its peripherals. Almost every computer has an attached monitor, keyboard, mouse, and printer. Without using wireless, this connection must be done with cables.

So many new users have a hard time finding the right cables and plugging them into the right little holes (even though they are usually color coded) that most computer vendors offer the option of sending a technician to the user's home to do it. To help these users, some companies got together to design a short-range wireless network called Bluetooth to connect these components without wires. The idea is that if your devices have Bluetooth, then you need no cables. You just put them down, turn them on, and they work together. For many people, this ease of operation is a big plus.

LOCAL AREA NETWORKS

Local area networks, generally called LANs, are 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 distinguished from other kinds of networks by three characteristics:
(1) Their size,
(2) Their transmission technology, and
(3) Their topology.

LANs are restricted in size, which means that the worst-case transmission time is bounded and known in advance. Knowing this bound makes it possible to use certain kinds of designs that would not otherwise be possible. It also simplifies network management.

METROPOLITAN AREA NETWORKS

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.** This system grew from earlier community antenna systems used in areas with poor over-the-air television reception. In these early systems, a large antenna was placed on top of a nearby hill
and signal was then piped to the subscribers' houses. At first, these were locally-designed, ad hoc systems. Then 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. Often these channels were highly specialized, such as all news, all sports, all cooking, all gardening, and so on.

WIDE AREA NETWORKS

A wide area network, or WAN, spans a large geographical area, often a country or continent. It contains a collection of machines intended for running user (i.e., application) programs.. The hosts are connected by a communication subnet, or just subnet for short. The hosts are owned by the customers (e.g., people's personal computers), whereas the communication subnet is typically owned and operated by a telephone company or Internet service provider. **The job of the subnet is to carry messages from host to host, just as the telephone system carries words from speaker to listener.**

Wireless Networks

Digital wireless communication is not a new idea. As early as 1901, the Italian physicist Guglielmo Marconi demonstrated a ship-to-shore wireless telegraph, using Morse Code (dots and dashes are binary, after all). Modern digital wireless systems have better performance, but the basic idea is the same. To a first approximation, wireless networks can be divided into three main categories:

- **System interconnection.**
- **Wireless LANs.**
- **Wireless WANs.**

System interconnection is all about interconnecting the components of a computer using short-range radio. Almost every computer has a monitor, keyboard, mouse, and printer connected to the main unit by cables. So many new users have a hard time plugging all the cables into the right little holes (even though they are usually color coded) that most computer vendors offer the option of sending a technician to the user's home to do it. Consequently, some companies got together to design a short-range wireless network called Bluetooth to connect these components without wires. Bluetooth also allows digital cameras, headsets, scanners, and other devices to connect to a computer by merely being brought within range. No cables, no driver installation, just put them down, turn them on, and they work. For many people, this ease of operation is a big plus.

Home Networks

Home networking is on the horizon. The fundamental idea is that in the future most homes will be set up for networking. Every device in the home will be capable of communicating with every other device, and all of them will be accessible over the Internet. This is one of those visionary concepts that nobody asked for (like TV remote controls or mobile phones), but once they arrived nobody can imagine how they lived without them. Many devices are capable of being networked. Some of the more obvious categories (with examples) are as follows:

1. Computers (desktop PC, notebook PC, PDA, shared peripherals).
2. Entertainment (TV, DVD, VCR, camcorder, camera, stereo, MP3).
3. Telecommunications (telephone, mobile telephone, intercom, fax).
4. Appliances (microwave, refrigerator, clock, furnace, airco, lights).
5. Telemetry (utility meter, smoke/burglar alarm, thermostat, babycam).

Internetworks

Many networks exist in the world, often with different hardware and software. People connected to one network often want to communicate with people attached to a different one. The fulfillment of this desire requires that different, and frequently incompatible networks, be connected, sometimes by means of machines called gateways to make the connection and provide the necessary translation, both in terms of hardware and software. A collection of interconnected networks is called an internetwork or internet. These terms will be used in a generic sense, in contrast to the worldwide Internet (which is one specific internet), which we will always capitalize.

A common form of internet is a collection of LANs connected by a WAN. The only real technical distinction between a subnet and a WAN in this case is whether hosts are present. If the system within the gray area contains only routers, it is a subnet; if it contains both routers and hosts, it is a WAN. The real differences relate to ownership and use.

Subnets, networks, and internetworks are often confused. Subnet makes the most sense in the context of a wide area network, where it refers to the collection of routers and communication lines

owned by the network operator. As an analogy, the telephone system consists of telephone switching offices connected to one another by high-speed lines, and to houses and businesses by low-speed lines. These lines and equipment, owned and managed by the telephone company, form the subnet of the telephone system. The telephones themselves (the hosts in this analogy) are not part of the subnet. The combination of a subnet and its hosts forms a network.

NETWORK SOFTWARE

The first computer networks were designed with the hardware as the main concern and the software as an afterthought. This strategy no longer works. Network software is now highly structured. In the following sections we examine the software structuring technique in some detail. The method described here forms the keystone of the entire book and will occur repeatedly later on.

Protocol Hierarchies

To reduce their design complexity, most networks are organized as a stack of layers or levels, each one built upon the one below it. The number of layers, the name of each layer, the contents of each layer, and the function of each layer differ from network to network. The purpose of each layer is to offer certain services to the higher layers, shielding those layers from the details of how the offered services are actually implemented. In a sense, each layer is a kind of virtual machine, offering certain services to the layer above it.

Design Issues for the Layers

Some of the key design issues that occur in computer networks are present in several layers..

- Every layer needs a mechanism for identifying senders and receivers. Since a network normally has many computers, some of which have multiple processes, a means is needed for a process on one machine to specify with whom it wants to talk. As a consequence of having multiple destinations, some form of addressing is needed in order to specify a specific destination.
- Another set of design decisions concerns the rules for data transfer. In some systems, data only travel in one direction; in others, data can go both ways.
- The protocol must also determine how many logical channels the connection corresponds to and what their priorities are.
- Many networks provide at least two logical channels per connection, one for normal data and one for urgent data.
- Error control is an important issue because physical communication circuits are not perfect.
- Many error-detecting and error-correcting codes are known, but both ends of the connection must agree on which one is being used.

Connection-Oriented and Connectionless Services

Layers can offer two different types of service to the layers above them: connection-oriented and connectionless.

. Connection-oriented service is modeled after the telephone system. To talk to someone, you pick up the phone, dial the number, talk, and then hang up. Similarly, to use a connection-oriented network service, the service user first establishes a connection, uses the connection, and then releases the connection.

The essential aspect of a connection is that it acts like a tube: the sender pushes objects (bits) in at one end, and the receiver takes them out at the other end. In most cases the order is preserved so that the bits arrive in the order they were sent. In some cases when a connection is established, the sender,

receiver, and subnet conduct a negotiation about parameters to be used, such as maximum message size, quality of service required, and other issues.

Typically, one side makes a proposal and the other side can accept it, reject it, or make a counterproposal. In contrast, connectionless service is modeled after the postal system. Each message (letter) carries the full destination address, and each one is routed through the system independent of all the others. Normally, when two messages are sent to the same destination, the first one sent will be the first one to arrive. However, it is possible that the first one sent can be delayed so that the second one arrives first.

Service Primitives

A service is formally specified by a set of primitives (operations) available to a user process to access the service. These primitives tell the service to perform some action or report on an action taken by a peer entity. If the protocol stack is located in the operating system, as it often is, the primitives are normally system calls. These calls cause a trap to kernel mode, which then turns control of the machine over to the operating system to send the necessary packets.

Primitive	Meaning
LISTEN	Block waiting for an incoming connection
CONNECT	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Send a message to the peer
DISCONNECT	Terminate a connection

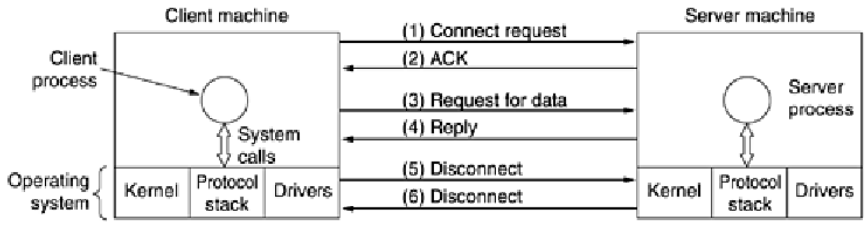
These primitives might be used as follows.

First, the server executes LISTEN to indicate that it is prepared to accept incoming connections. A common way to implement LISTEN is to make it a blocking system call. After executing the primitive, the server process is blocked until a request for connection appears.

Next, the client process executes CONNECT to establish a connection with the server. The CONNECT call needs to specify who to connect to, so it might have a parameter giving the server's address. The operating system then typically sends a packet to the peer asking it to connect. The client process is suspended until there is a response. When the packet arrives at the server, it is processed by the operating system there.

When the system sees that the packet is requesting a connection, it checks to see if there is a listener. If so, it does two things: unblocks the listener and sends back an acknowledgement. The arrival of this acknowledgement then releases the client. At this point the client and server are both running and they have a connection established.. If a connection request arrives and there is no listener, the result is undefined.

In some systems the packet may be queued for a short time in anticipation of a LISTEN.



The obvious analogy between this protocol and real life is a customer (client) calling a company's customer service manager. The service manager starts out by being near the telephone in case it rings. Then the client places the call. When the manager picks up the phone, the connection is established. The next step is for the server to execute RECEIVE to prepare to accept the first request.

Normally, the server does this immediately upon being released from the LISTEN, before the acknowledgement can get back to the client. The RECEIVE call blocks the server. Then the client executes SEND to transmit its request (3) followed by the execution of RECEIVE to get the reply. The arrival of the request packet at the server machine unblocks the server process so it can process the request. After it has done the work, it uses SEND to return the answer to the client (4).

The arrival of this packet unblocks the client, which can now inspect the answer. If the client has additional requests, it can make them now. If it is done, it can use DISCONNECT to terminate the connection. Usually, an initial DISCONNECT is a blocking call, suspending the client and sending a packet to the server saying that the connection is no longer needed (5). When the server gets the packet, it also issues a DISCONNECT of its own, acknowledging the client and releasing the connection. When the server's packet (6) gets back to the client machine, the client process is released and the connection is broken. In a nutshell, this is how connection-oriented communication works. Of course, life is not so simple. Many things can go wrong here.

The Relationship of Services to Protocols

Services and protocols are distinct concepts, although they are frequently confused. This distinction is so important, however, that we emphasize it again here. A service is a set of primitives (operations) that a layer provides to the layer above it. The service defines what operations the layer is prepared to perform on behalf of its users, but it says nothing at all about how these operations are implemented.

A service relates to an interface between two layers, with the lower layer being the service provider and the upper layer being the service user. A protocol, in contrast, is a set of rules governing the format and meaning of the packets, or messages that are exchanged by the peer entities within a layer. Entities use protocols to implement their service definitions. They are free to change their protocols at will, provided they do not change the service visible to their users. In this way, the service and the protocol are completely decoupled.

REFERENCE MODELS

Reference models give a conceptual framework that standardizes communication between heterogeneous **networks**.We will discuss two important network architectures: the OSI reference model and the TCP/IP reference model. Although the *protocols* associated with the OSI model are not used any more, the *model* itself is actually quite general and still valid, and the features discussed at each layer are still very important.

The TCP/IP model has the opposite properties: the model itself is not of much use but the protocols are widely used. For this reason we will look at both of them in detail. Also, sometimes you can learn more from failures than from successes.

The OSI Reference Model

This model is based on a proposal developed by the International Standards Organization (ISO) as a first step toward international standardization of the protocols used in the various layers (Day and Zimmermann, 1983). It The model is called the ISO **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. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:

1. A layer should be created where a different abstraction is needed.
2. Each layer should perform a well-defined function.

3. The function of each layer should be chosen with an eye toward defining internationally standardized protocols.
4. The layer boundaries should be chosen to minimize the information flow across the interfaces.
5. The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity and small enough that the architecture does not become unwieldy.

The Physical Layer

The **physical layer** is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit it is received by the other side as a 1 bit, not as a 0 bits.

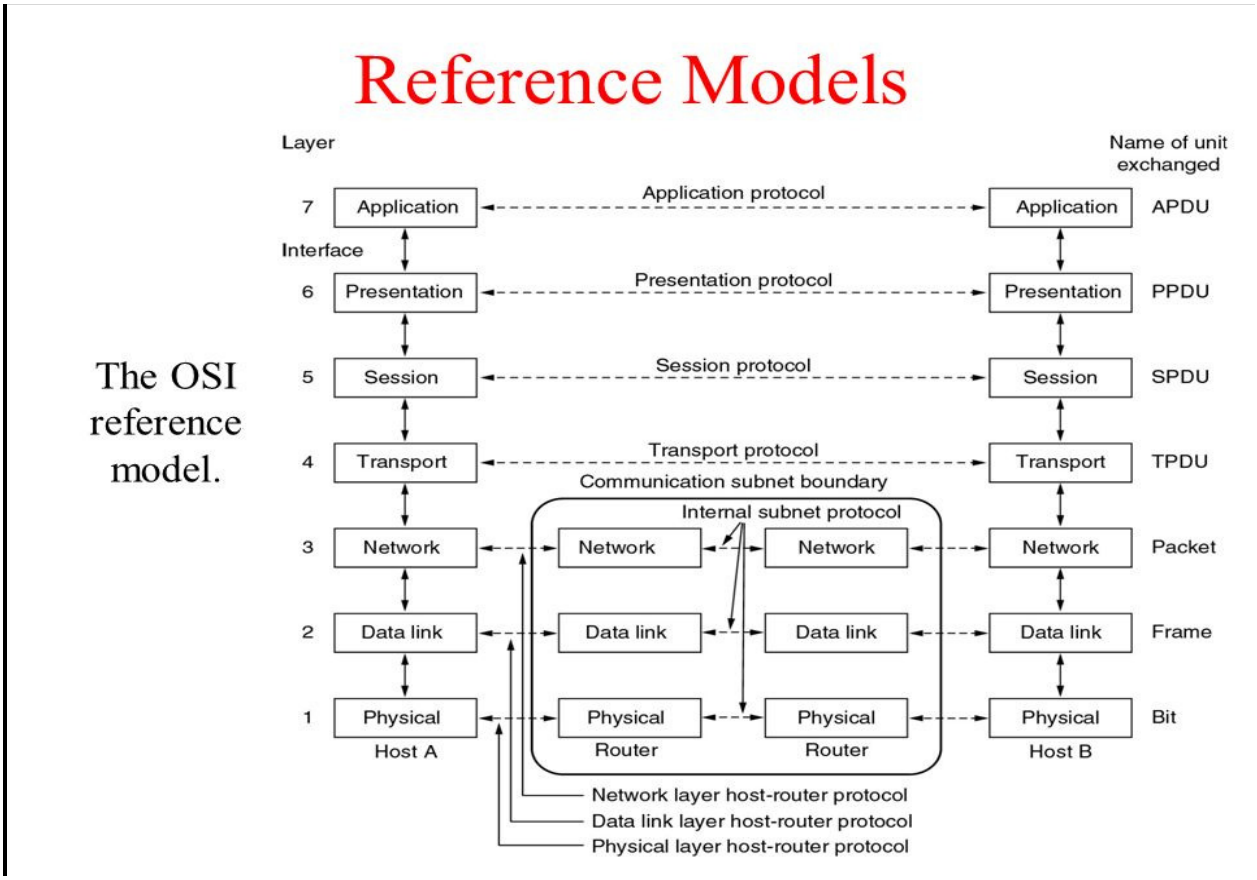


Figure 1-20. The OSI reference model.

Typical questions here are what electrical signals should be used to represent a 1 and a 0, how many nanoseconds a bit lasts, whether transmission may proceed simultaneously in both directions, how the initial connection is established, how it is torn down when both sides are finished, how many pins the network connector has, and what each pin is used for.

The Data Link Layer

The main task of the data link layer is to transform a raw transmission facility into a line that appears free of undetected transmission errors. It does so by masking the real errors so the network layer does not see them.

Broadcast networks have an additional issue in the data link layer: how to control access to the shared channel. A special sublayer of the data link layer, the medium access control sublayer, deals with this problem.

The Network Layer

The network layer controls the operation of the subnet. A key design issue is determining how packets are routed from source to destination.

Routes can be based on static tables that are ‘‘wired into’’ the network and rarely changed, or more often they can be updated automatically to avoid failed components.

They can also be determined at the start of each conversation, for example, a terminal session, such as a login to a remote machine. Finally, they can be highly dynamic, being determined anew for each packet to reflect the current network load.

The Transport Layer

The basic function of the transport layer is to accept data from above it, split it up into smaller units if need be, pass these to the network layer, and ensure that the pieces all arrive correctly at the other end.

The transport layer also determines what type of service to provide to the session layer, and, ultimately, to the users of the network. The most popular type of transport connection is an error-free point-to-point channel that delivers messages or bytes in the order in which they were sent.

The transport layer is a true end-to-end layer; it carries data all the way from the source to the destination. In other words, a program on the source machine carries on a conversation with a similar program on the destination machine, using the message headers and control messages.

The Session Layer

The session layer allows users on different machines to establish sessions between them.

Sessions offer various services, including dialog control (keeping track of whose turn it is to transmit), token management (preventing two parties from attempting the same critical operation simultaneously), and synchronization (check pointing long transmissions to allow them to pick up from where they left off in the event of a crash and subsequent recovery).

The Presentation Layer

Unlike the lower layers, which are mostly concerned with moving bits around, the presentation layer is concerned with the syntax and semantics of the information transmitted.

In order to make it possible for computers with different internal data representations to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used ‘‘on the wire.’’ The presentation layer manages these abstract data structures and al-lows higher-level data structures (e.g., banking records) to be defined and exchanged.

The Application Layer

The application layer contains a variety of protocols that are commonly needed by users. One widely used application protocol is HTTP (Hyper Text Transfer Protocol), which is the basis for the World Wide Web. When a browser wants a Web page, it sends the name of the page it wants to the server hosting the page using HTTP. The server then sends the page back. Other application protocols are used for file transfer, electronic mail, and network news.

THE PHYSICAL LAYER

In this chapter we will look at the lowest layer depicted in the hierarchy. It defines the mechanical, electrical, and timing interfaces to the network. We will begin with a theoretical analysis of data transmission, only to discover that Mother. Nature puts some limits on what can be sent over a channel. Then we will cover three kinds of transmission media: guided (copper wire and fiber optics), wireless (terrestrial radio), and satellite.

This material will provide background information on the key transmission technologies used in modern networks. The remainder of the chapter will be devoted to three examples of communication systems used in practice for wide area computer networks: the (fixed) telephone system, the mobile phone system, and the cable television system. All three use fiber optics in the backbone, but they are organized differently and use different technologies for the last mile.

GUIDED TRANSMISSION MEDIA

The purpose of the physical layer is to transport bits from one machine to an-other. Various physical media can be used for the actual transmission. Each one has its own niche in terms of bandwidth, delay, cost, and ease of installation and maintenance. Media are roughly grouped into guided media, such as copper wire and fiber optics, and unguided media, such as terrestrial wireless, satellite, and lasers through the air. We will look at guided media in this section, and unguided media in the next sections.

Magnetic Media

One of the most common ways to transport data from one computer to another is to write them onto magnetic tape or removable media (e.g., recordable DVDs), physically transport the tape or disks to the destination machine, and read them back in again.

Although this method is not as sophisticated as using a geosynchronous communication satellite, it is often more cost effective, especially for applications in which high bandwidth or cost per bit transported is the key factor.

A simple calculation will make this point clear. An industry-standard Ultrium tape can hold 800 gigabytes. A box 60 × 60 × 60 cm can hold about 1000 of these tapes, for a total capacity of 800 terabytes, or 6400 terabits (6.4 petabits). A box of tapes can be delivered anywhere in the United States in 24 hours by Federal Express and other companies.

Twisted Pairs

Although the bandwidth characteristics of magnetic tape are excellent, the delay characteristics are poor.

Transmission time is measured in minutes or hours, not milliseconds. For many applications an online connection is needed. One of the oldest and still most common transmission media is **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. A signal is usually carried as the difference in voltage between the two wires in the pair. This provides better immunity to external noise because the noise tends to affect both wires the same, leaving the differential unchanged.

Twisted pairs can be used for transmitting either analog or digital information. The bandwidth depends on the thickness of the wire and the distance traveled, but several megabits/sec can be achieved for a few kilometers in many cases. Due to their adequate performance and low cost, twisted pairs are widely used and are likely to remain so for years to come. Different LAN standards may use the twisted pairs differently. For example,100-Mbps Ethernet uses two (out of the four) pairs, one pair for each direction

1-Gbps Ethernet uses all four pairs in both directions simultaneously; this requires the receiver to factor out the signal that is transmitted locally.

Some general terminology is now in order. Links that can be used in both directions at the same time, like a two-lane road, are called full-duplex links. In contrast, links that can be used in either direction, but only one way at a time, like a single-track railroad line. are called half-duplex links. A third category consists of links that allow traffic in only one direction, like a one-way street. They are called simplex link

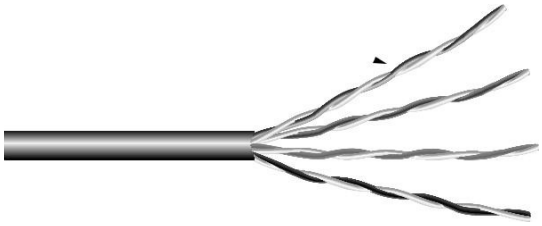


Figure 2-1. Category UTP cable with four twisted pairs.

Through Category, these wiring types are referred to as UTP (Unshielded Twisted Pair) as they consist simply of wires and insulators. In contrast to these, Category cables have shielding on the individual twisted pairs, as well as around the entire cable (but inside the plastic protective sheath).

Coaxial Cable

Another common transmission medium is the **coaxial cable** (known to its many friends as just ‘‘coax’’ and pronounced ‘‘co-ax’’). It has better shielding and greater bandwidth than unshielded twisted pairs, so it can span longer distances at higher speeds. Two kinds of coaxial cable are widely used. One kind, 50-ohm cable, is commonly used when it is intended for digital transmission from the start. The other kind, 75-ohm cable, is commonly used for analog transmission and cable television. This distinction is based on historical, rather than technical, factors (e.g., early dipole antennas had an impedance of 300 ohms, and it was easy to use existing 4:1 impedance-matching transformers). Starting in the mid-1990s, cable TV operators began to provide Internet access over cable, which has made 75-ohm cable more important for data communication.

A coaxial cable consists of a 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. A cutaway view of a coaxial cable is shown in Fig. 2-

4.

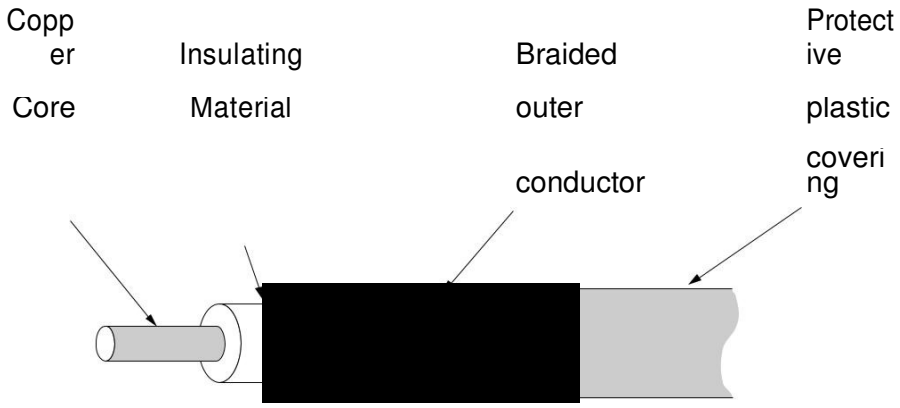


Figure 2-2. A coaxial cable

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 and length. Modern cables have a bandwidth of up to a few GHz. Coaxial cables used to be widely used within the telephone system for long-distance lines but have now largely been replaced by fiber optics on long-haul routes. Coax is still widely used for cable television and metropolitan area networks, however.

Power Lines

The telephone and cable television networks are not the only sources of wiring that can be reused for data communication. There is a yet more common kind of wiring: electrical power lines. Power lines deliver electrical power to houses, and electrical wiring within houses distributes the power to electrical outlets.

The use of power lines for data communication is an old idea. Power lines have been used by electricity companies for low-rate communication such as re-mote metering for many years, as well in the home to control devices (e.g., the X10 standard).

The convenience of using power lines for networking should be clear. Simply plug a TV and a receiver into the wall, which you must do anyway because they need power, and they can send and receive movies over the electrical wiring. This configuration is shown in Fig. 2-5. There is no other plug or radio. The data signal is superimposed on the low-frequency power signal (on the active or ‘‘hot’’ wire) as both signals use the wiring at the same time.

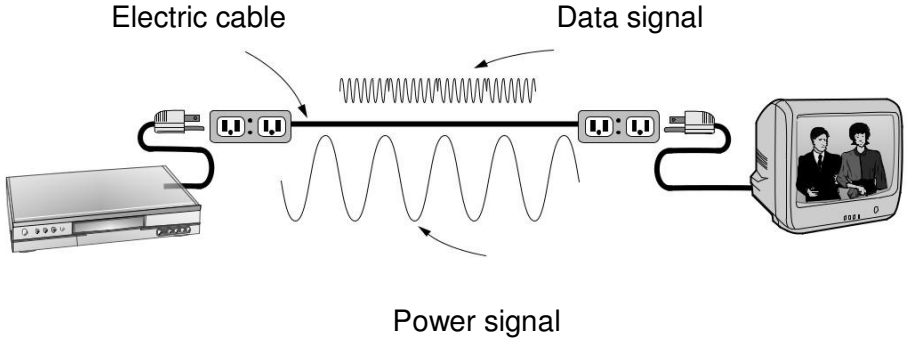


Figure 2-3. A network that uses household electrical wiring.

The difficulty with using household electrical wiring for a network is that it was designed to distribute power signals. This task is quite different than distributing data signals, at which household wiring does a horrible job. Electrical signals are sent at 50–60 Hz and the wiring attenuates the much higher frequency (MHz) signals needed for high-rate data communication. The electrical properties of the wiring vary from one house to the next and change as appliances are turned on and off, which causes data signals to bounce around the wiring. Transient currents when appliances switch on and off create electrical noise over a wide range of frequencies.

Fiber Optics

In this section we will study fiber optics to learn how that transmission technology works. In the ongoing race between computing and communication, communication may yet win because of fiber optic networks. The implication of this would be essentially infinite bandwidth and a new conventional wisdom that computers are hopelessly slow so that networks should try to avoid computation at all costs, no matter how much bandwidth that wastes

This change will take a while to sink in to a generation of computer scientists and engineers taught to think in terms of the low Shannon limits imposed by copper.

Fiber optics are used for long-haul transmission in network backbones, high-speed LANs (although so far, copper has always managed catch up eventually), and high-speed Internet access such as FttH (Fiber to the Home).

This transmission system would leak light and be useless in practice were it not for an interesting principle of physics. When a light ray passes from one medium, fused silica to air the ray is refracted (bent) at the silica/air boundary, as shown in. Here we see a light ray incident on the boundary at an angle α_1 emerging at an angle $\beta 1$

The amount of refraction depends on the properties of the two media (in particular, their indices of refraction). For angles of incidence above a certain critical value, the light is refracted back into the silica; none of it escapes into the air. Thus, a light ray incident at or above the critical angle is trapped inside the fiber, and can propagate for many kilometers with virtually no loss.

the fiber's diameter is reduced to a few wavelengths of light the fiber acts like a wave guide and the light can propagate only in a straight line, without bouncing, yielding a single-mode fiber. Single-mode fibers are more ex-pensive but are widely used for longer distances.

WIRELESS TRANSMISSION

Our age has given rise to information junkies: people who need to be on-line all the time. For these mobile users, twisted pair, coax, and fiber optics are of no use. They need to get their hits of data for their laptop, notebook, shirt pocket, palmtop, or wristwatch computers without being tethered to the terrestrial communication infrastructure. For these users, wireless communication is the answer. In the following sections, we will look at wireless communication in general, as it has many other important applications besides providing connectivity to users who want to surf the Web from the beach.

The Electromagnetic Spectrum

When electrons move, they create electromagnetic waves that can propagate through space (even in a vacuum). These waves were predicted by the British physicist James Clerk Maxwell in 1865 and first observed by the German physicist Heinrich Hertz in 1887. **The** number of oscillations per second of a wave is called its frequency, f, and is measured in Hz (in honor of Heinrich Hertz). The distance between

two consecutive maxima (or minima) is called the wavelength, which is universally designated by the Greek letter l (lambda).

When an antenna of the appropriate size is attached to an electrical circuit, the electromagnetic waves can be broadcast efficiently and received by a receiver some distance away. All wireless communication is based on this principle. In vacuum, all electromagnetic waves travel at the same speed, no matter what their frequency.

This speed, usually called the speed of light, c, is approximately 3 x 108 m/sec, or about 1 foot (30 cm) per nanosecond. (A case could be made for redefining the foot as the distance light travels in a vacuum in 1 nsec rather than basing it on the shoe size of some long-dead king.) In copper or fiber the speed slows to about 2/3 of this value and becomes slightly frequency dependent. The speed of light is the ultimate speed limit. No object or signal can ever move faster than it.

The fundamental relation between f, l, and c (in vacuum) is

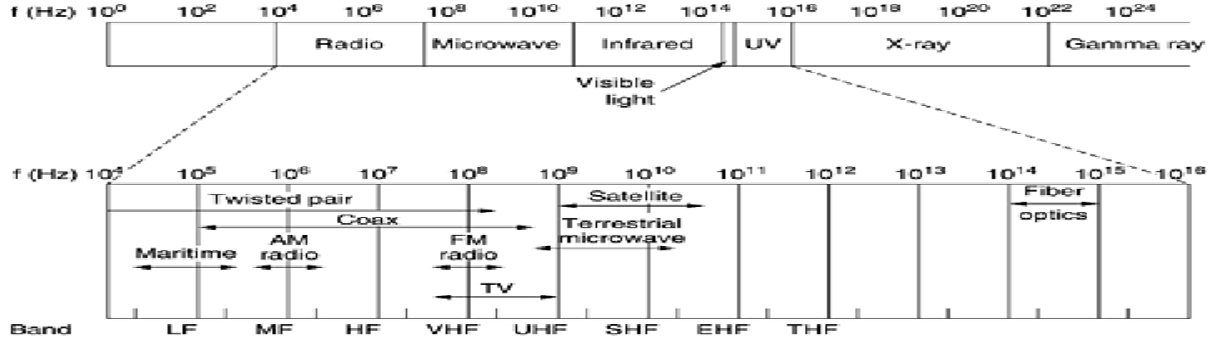
Equation 2

$$\lambda f = c$$

Since c is a constant, if we know f, we can find l, and vice versa. As a rule of thumb, when l is in meters and f is in MHz, lf 300. For example, 100-MHz waves are about 3 meters long, 1000-MHz waves are 0.3-meters long, and 0.1-meter waves have a frequency of 3000 MHz..

The radio, microwave, infrared, and visible light portions of the spectrum can all be used for transmitting information by modulating the amplitude, frequency, or phase of the waves. Ultraviolet light, X-rays, and gamma rays would be even better, due to their higher frequencies, but they are hard to produce and modulate, do not propagate well through buildings, and are dangerous to living things.

The electromagnetic spectrum and its uses for communication.



The amount of information that an electromagnetic wave can carry is related to its bandwidth. With current technology, it is possible to encode a few bits per Hertz at low frequencies, but often as many as 8 at high frequencies, so a coaxial cable with a 750 MHz bandwidth can carry several gigabits/sec

If we solve Eq. (2-2) for f and differentiate with respect to l, we get

$$\frac{df}{d\lambda} = -\frac{c}{\lambda^2}$$

If we now go to finite differences instead of differentials and only look at absolute values, we get

Equation 2

$$\Delta f = \frac{c\Delta\lambda}{\lambda^2}$$

Thus, given the width of a wavelength band, Dl, we can compute the corresponding frequency band, Df, and from that the data rate the band can produce. The wider the band, the higher the data rate. In frequency hopping spread spectrum, the transmitter hops from frequency to frequency hundreds of times

per second. It is popular for military communication because it makes transmissions hard to detect and next to impossible to jam.

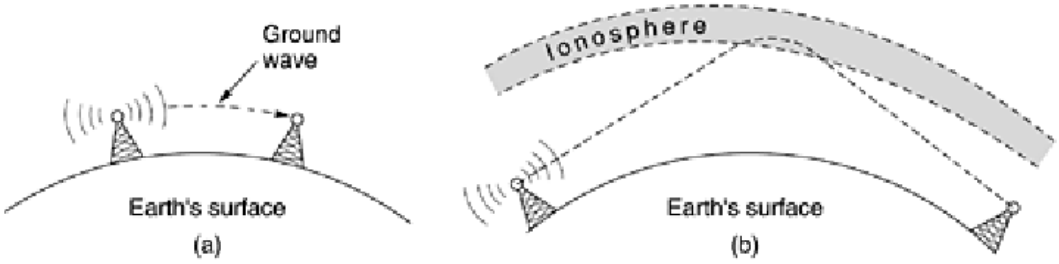
It also offers good resistance to multipath fading because the direct signal always arrives at the receiver first. Reflected signals follow a longer path and arrive later. By then the receiver may have changed frequency and no longer accepts signals on the previous frequency, thus eliminating interference between the direct and reflected signals. In recent years, this technique has also been applied commercially—both 802.11 and Bluetooth use it, for example.

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. Sometimes omnidirectional radio is good, but sometimes it is bad

In the VLF, LF, and MF bands, radio waves follow the ground,. These waves can be detected for perhaps 1000 km at the lower frequencies, less at the higher ones. AM radio broadcasting uses the MF band, which is why the ground waves from Boston AM radio stations cannot be heard easily in New York. Radio waves in these bands pass through buildings easily, which is why portable radios work indoors. The main problem with using these bands for data communication is their low bandwidth



In the HF and VHF bands, the ground waves tend to be absorbed by the earth. However, the waves that reachthe ionosphere, a layer of charged particles circling the earth at a height of 100 to 500 km, are refracted by it andsent back to earth). Under certain atmospheric conditions, the signals can bounceseveral times. Amateur radio operators (hams) use these bands to talk long distance. The military alsocommunicate in the HF and VHF bands.

Microwave Transmission

Above 100 MHz, the waves travel in nearly straight lines and can therefore be narrowly focused. Concentrating all the energy into a small beam by means of a parabolic antenna (like the familiar satellite TV dish) gives a much higher signal-to-noise ratio, but the transmitting and receiving antennas must be accurately aligned with each other.

In addition, this directionality allows multiple transmitters lined up in a row to communicate with multiple receivers in a row without interference, provided some minimum spacing rules are observed. Before fiber optics, for decades these microwaves formed the heart of the long-distance telephone transmission system. In fact, MCI, one of AT&T's first competitors after it was deregulated, built its entire system with microwave communications going from tower to tower tens of kilometers apart. Even the company's name reflected this (MCI stood for Microwave Communications, Inc.).

MCI has since gone over to fiber and merged with WorldCom. Since the microwaves travel in a straight line, if the towers are too far apart, the earth will get in the way (think about a San Francisco to Amsterdam link). Consequently, repeaters are needed periodically. The higher the towers are, the farther apart they can be. The distance between repeaters goes up very roughly with the square root of the tower height.

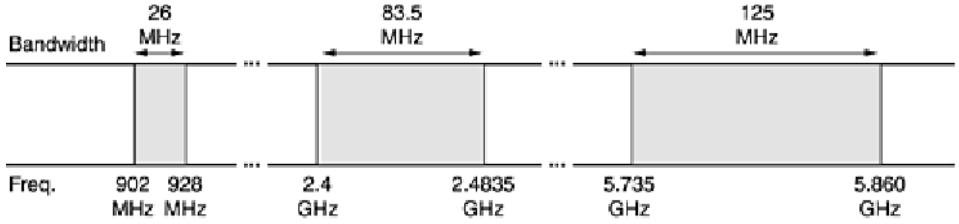
For 100-meter-high towers, repeaters can be spaced 80 km apart. Unlike radio waves at lower frequencies, microwaves do not pass through buildings well. In addition, even though the beam may be well focused at the transmitter, there is still some divergence in space. Some waves may be refracted off low-lying atmospheric layers and may take slightly longer to arrive than the direct waves. The delayed waves may arrive out of phase with the direct wave and thus cancel the signal. This effect is called multipath fading and is often a serious problem. It is weather and frequency dependent.

The Politics of the Electromagnetic Spectrum

To prevent total chaos, there are national and international agreements about who gets to use which frequencies. Since everyone wants a higher data rate, everyone wants more spectrum. National governments allocate spectrum for AM and FM radio, television, and mobile phones, as well as for telephone companies, police, maritime, navigation, military, government, and many other competing users.

Worldwide, an agency of ITU-R (WARC) tries to coordinate this allocation so devices that work in multiple countries can be manufactured. However, countries are not bound by ITU-R's recommendations, and the FCC (Federal Communication Commission), which does the allocation for the United States, has occasionally rejected ITU-R's recommendations (usually because they required some politically-powerful group giving up some piece of the spectrum).

Figure 2-13. The ISM bands in the United States.



Infrared and Millimeter Waves

Unguided infrared and millimeter waves are widely used for short-range communication. The remote controls used on televisions, VCRs, and stereos all use infrared communication. 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 radio.

Lightwave Transmission

Unguided optical signaling has been in use for centuries. Coherent optical signaling using lasers is inherently unidirectional, so each building needs its own laser and its own photodetector. This scheme offers very high bandwidth and very low cost. It is also relatively easy to install and, unlike microwave, does not require an FCC license.

The laser's strength, a very narrow beam, is also its weakness here. Aiming a laser beam 1-mm wide at a target the size of a pin head 500 meters away requires the marksmanship of a latter-day Annie Oakley. Usually, lenses are put into the system to defocus the beam slightly. A disadvantage is that laser beams cannot penetrate rain or thick fog, but they normally work well on sunny days.

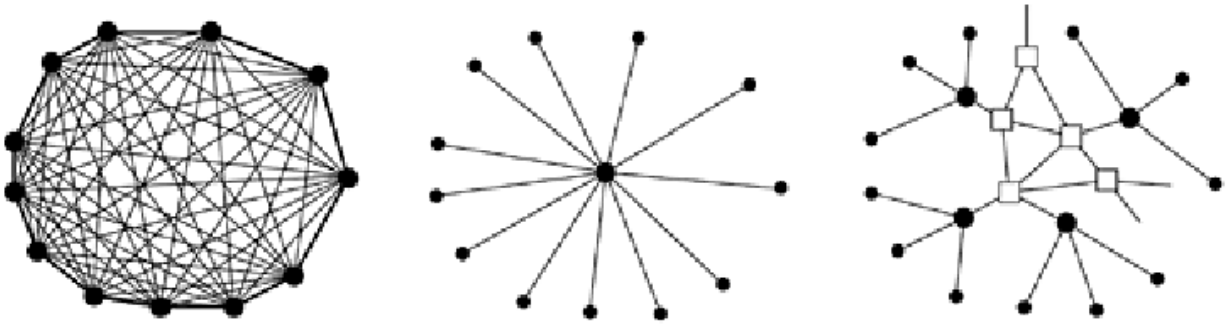
THE PUBLIC SWITCHED TELEPHONE NETWORK

When two computers owned by the same company or organization and located close to each other need to communicate, it is often easiest just to run a cable between them. LANs work this way. However, when the distances are large or there are many computers or the cables have to pass through a public road or

other public right of way, the costs of running private cables are usually prohibitive These facilities, especially the PSTN (Public Switched Telephone Network), were usually designed many years ago, with a completely different goal in mind: transmitting the human voice in a more-or-less recognizable form. Their suitability for use in computer-computer communication is often marginal at best, but the situation is rapidly changing with the introduction of fiber optics and digital technology.

Structure of the Telephone System

Soon after Alexander Graham Bell patented the telephone in 1876, there was an enormous demand for his new invention. The initial market was for the sale of telephones, which came in pairs. It was up to the customer to string a single wire between them. The electrons returned through the earth. If a telephone owner wanted to talk to n other telephone owners, separate wires had to be strung to all n houses. Within a year, the cities were covered with wires passing over houses and trees in a wild jumble. It became immediately obvious that the model of connecting every telephone to every other telephone, as shown in Fig was not going to work.



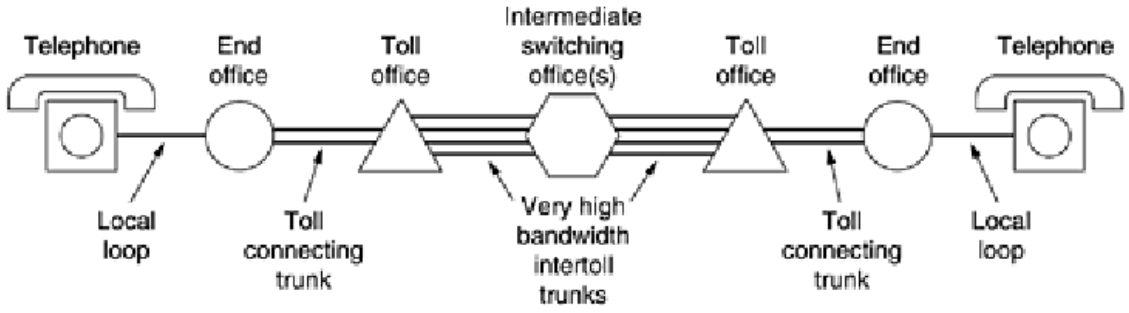
(a) (b) (c)

To his credit, Bell saw this and formed the Bell Telephone Company, which opened its first switching office (in New Haven, Connecticut) in 1878. The company ran a wire to each customer's house or office. To make a call, the customer would crank the phone to make a ringing sound in the telephone company office to attract the attention of an operator, who would then manually connect the caller to the callee by using a jumper cable. The model of a single switching office is illustrated in Figure .(b)

Pretty soon, Bell System switching offices were springing up everywhere and people wanted to make long distance calls between cities, so the Bell system began to connect the switching offices. The original problem soon returned: to connect every switching office to every other switching office by means of a wire between them quickly became unmanageable, so second-level switching offices were invented. After a while, multiple second level offices were needed, as illustrated in Fig. (c).

Eventually, the hierarchy grew to five levels. By 1890, the three major parts of the telephone system were in place: the switching offices, the wires between the customers and the switching and the long-distance connections between the switching offices.. The following description is highly simplified but gives the essential flavor nevertheless. Each telephone has two copper wires coming out of it that go directly to the telephone company's nearest end office (also called a local central office). The distance is typically 1 to 10 km, being shorter in cities than in rural areas.

A typical circuit route for a medium-distance call.



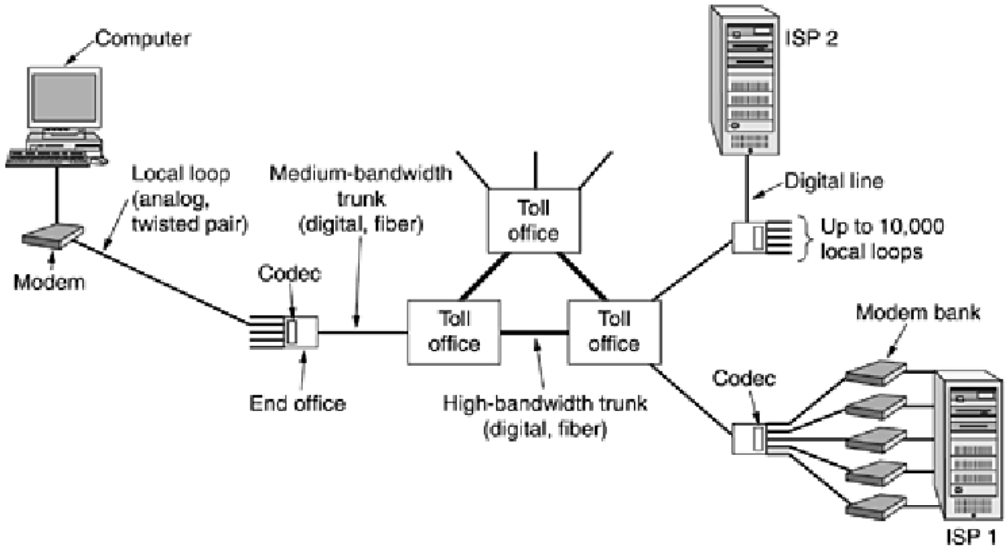
In summary, the telephone system consists of three major components:

1. **Local loops (analog twisted pairs going into houses and businesses).**
2. **Trunks (digital fiber optics connecting the switching offices).**
3. **Switching offices (where calls are moved from one trunk to another).**

The Local Loop: Modems, ADSL, and Wireless

The main parts of the system are illustrated in Fig. Here we see the local loops, the trunks, and the toll offices and end offices, both of which contain switching equipment that switches calls. An end office has up to 10,000 local loops . With the advent of competition for local service, this system was no longer tenable because multiple companies wanted to own the end office code. Also, the number of codes was basically used up, so complex mapping schemes had to be introduced.

. The use of both analog and digital transmission for a computer to computer call. Conversion is done by the modems and codecs.



Let us begin with the part that most people are familiar with:

- The two-wire local loop coming from a telephone company end office into houses and small businesses.
- The local loop is also frequently referred to as the "last mile," although the length can be up to several miles.
- It has used analog signaling for over 100 years and is likely to continue doing so for some years to come, due to the high cost of converting to digital..

Modems

Due to the problems just discussed, especially the fact that both attenuation and propagation speed are frequency dependent, it is undesirable to have a wide range of frequencies in the signal.

Unfortunately, the square waves used in digital signals have a wide frequency spectrum and thus are subject to strong attenuation and delay distortion. These effects make baseband (DC) signaling unsuitable except at slow speeds and over short distances.

To get around the problems associated with DC signaling, especially on telephone lines, AC signaling is used. A continuous tone in the 1000 to 2000-Hz range, called a sine wave carrier, is introduced. Its amplitude, frequency, or phase can be modulated to transmit information.

- In amplitude modulation, two different amplitudes are used to represent 0 and 1, respectively.
- In frequency modulation, also known as frequency shift keying, two (or more) different tones are used. (The term keying is also widely used in the industry as a synonym for modulation.)
- In the simplest form of phase modulation, the carrier wave is systematically shifted 0 or 180 degrees at uniformly spaced intervals.

The following Figure illustrates the three forms of modulation. In Fig. (a) one of the amplitudes is nonzero and one is zero. In Fig. (b) two frequencies are used. In Fig. (c) a phase shift is either present or absent at each bit boundary. A device that accepts a serial stream of bits as input and produces a carrier modulated by one (or more) of these methods (or vice versa) is called a modem (for modulator-demodulator). The modem is inserted between the (digital) computer and the (analog) telephone system.

Figure (a) A binary signal. (b) Amplitude modulation. (c) Frequency modulation. (d) Phase modulation.

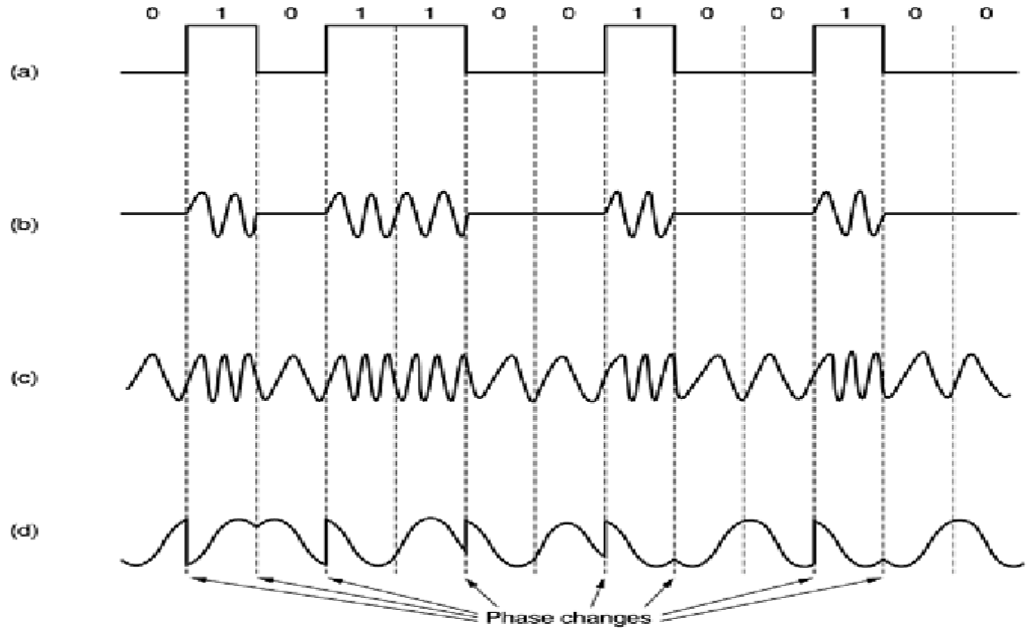
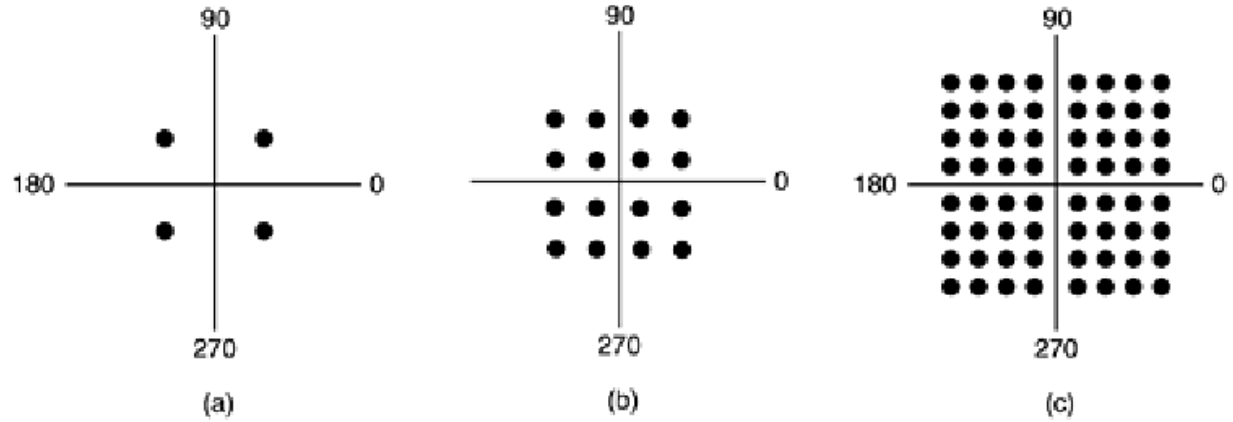


Figure. (a) QPSK. (b) QAM-16. (c) QAM-64.

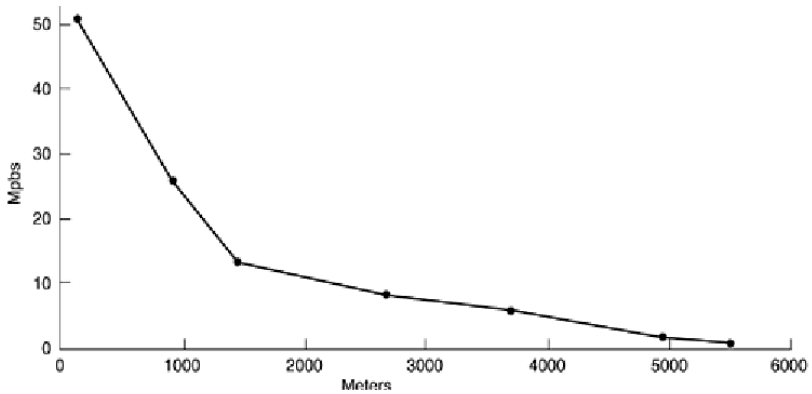


In Fig. (b) we see a different modulation scheme, in which four amplitudes and four phases are used, for a total of 16 different combinations. **This modulation scheme can be used to transmit 4 bits per symbol. It is called QAM-16 (Quadrature Amplitude Modulation).** Sometimes the term 16-QAM is used instead. QAM-16 can be used, for example, to transmit 9600 bps over a 2400-baud line.
Figure (c) is yet another modulation scheme involving amplitude and phase. It allows 64 different combinations, so 6 bits can be transmitted per symbol. It is called QAM-64. Higher-order QAMs also are used. Diagrams such as those of Figure which show the legal combinations of amplitude and phase, are called constellation diagrams. Each high-speed modem standard has its own constellation pattern and can talk only to other modems that use the same one

Digital Subscriber Lines

When the telephone industry finally got to 56 kbps, it patted itself on the back for a job well done. Meanwhile, the cable TV industry was offering speeds up to 10 Mbps on shared cables, and satellite companies were planning to offer upward of 50 Mbps. As Internet access became an increasingly important part of their business, the telephone companies (LECs) began to realize they needed a more competitive product. Initially, there were many overlapping offerings, all under the general name of xDSL (Digital Subscriber Line), for various x.

Bandwidth versus distance over category 3 UTP for DSL.



Trunks and Multiplexing

Economies of scale play an important role in the telephone system. It costs essentially the same amount of money to install and maintain a high-bandwidth trunk as a low-bandwidth trunk between two switching offices. Consequently, telephone companies have developed elaborate schemes for multiplexing many conversations over a single physical trunk.

These multiplexing schemes can be divided into two basic categories:

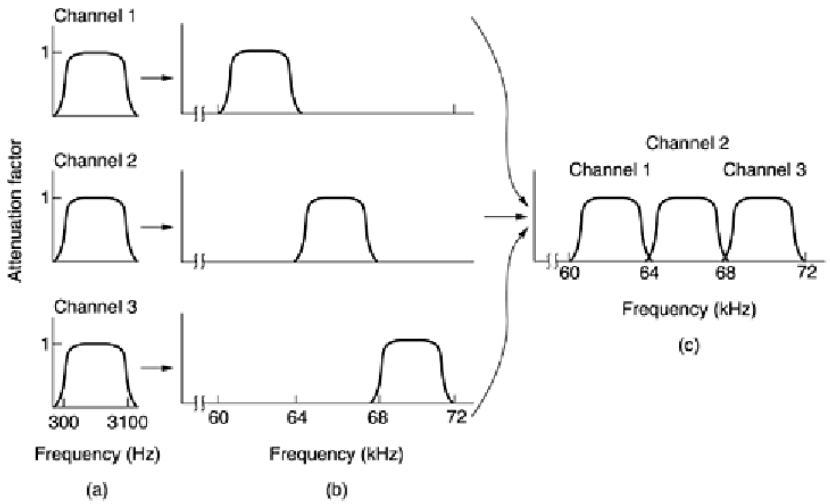
- **FDM (Frequency Division Multiplexing)**
- **DM (Time Division Multiplexing).**
- **In FDM, the frequency spectrum is divided into frequency bands, with each user having exclusive possession of some band.**
- **In TDM, the users take turns (in a round-robin fashion), each one periodically getting the entire bandwidth for a little burst of time.**

AM radio broadcasting provides illustrations of both kinds of multiplexing. The allocated spectrum is about 1 MHz, roughly 500 to 1500 kHz. Different frequencies are allocated to different logical channels (stations), each operating in a portion of the spectrum, with the interchannel separation great enough to prevent interference. This system is an example of frequency division multiplexing. In addition (in some countries), the individual stations have two logical subchannels: music and advertising. These two alternate in time on the same frequency, first a burst of music, then a burst of advertising, then more music, and so on. This situation is time division multiplexing.
Then we will turn to TDM, and end with an advanced TDM system used for fiber optics (SONET).

Frequency Division Multiplexing

The following Figure shows how three voice-grade telephone channels are multiplexed using FDM. Filters limit the usable bandwidth to about 3100 Hz per voice-grade channel. When many channels are multiplexed together, 4000 Hz is allocated to each channel to keep them well separated. First the voice channels are raised in frequency, each by a different amount. This overlap means that a strong spike at the edge of one channel will be felt in the adjacent one as nonthermal noise.

Frequency division multiplexing. (a) The original bandwidths. (b) The bandwidths raised in frequency. (c) The multiplexed channel.

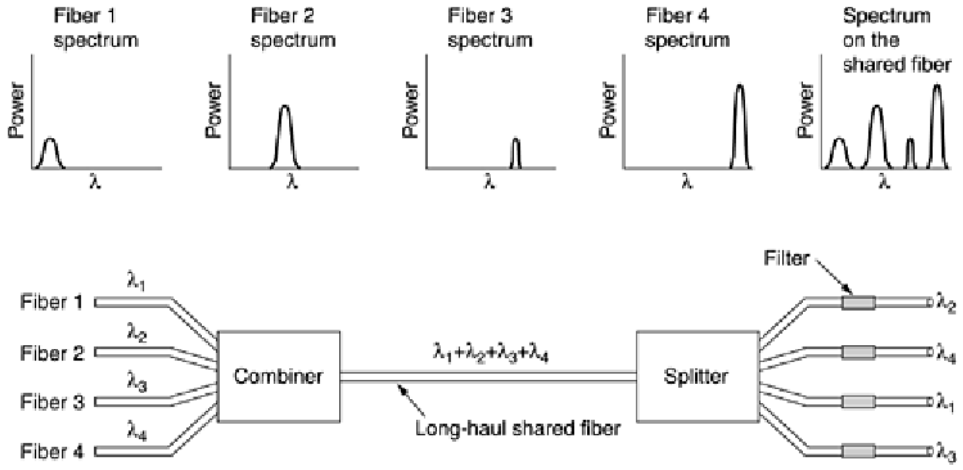


1. The FDM schemes used around the world are to some degree standardized.
2. A widespread standard is twelve 4000-Hz voice channels multiplexed into the 60 to 108 kHz band. This unit is called a group.
3. The 12-kHz to 60- kHz band is sometimes used for another group. Many carriers offer a 48- to 56-kbps leased line service to customers, based on the group.
4. Five groups (60 voice channels) can be multiplexed to form a supergroup.
5. The next unit is the mastergroup, which is five supergroups (CCITT standard) or ten supergroups (Bell system).
6. Other standards of up to 230,000 voice channels also exist.

Wavelength Division Multiplexing

For fiber optic channels, a variation of frequency division multiplexing is used. It is called WDM (Wavelength Division Multiplexing). The basic principle of WDM on fibers is depicted in Fig.. Here four fibers come together at an optical combiner, each with its energy present at a different wavelength. The four beams are combined onto a single shared fiber for transmission to a distant destination. At the far end, the beam is split up over as many fibers as there were on the input side. Each output fiber contains a short, specially-constructed core that filters out all but one wavelength. The resulting signals can be routed to their destination or recombined in different ways for additional multiplexed transport.

. Wavelength division multiplexing.



Time Division Multiplexing

WDM technology is wonderful, but there is still a lot of copper wire in the telephone system, so let us turn back to it for a while.

Although FDM is still used over copper wires or microwave channels, it requires analog circuitry and is not amenable to being done by a computer.

In contrast, TDM can be handled entirely by digital electronics, so it has become far more widespread in recent years.

Unfortunately, it can only be used for digital data. Since the local loops produce analog signals, a conversion is needed from analog to digital in the end office, where all the individual local loops come together to be combined onto outgoing trunks. Computer data sent over a modem are also analog, so the following description also applies to them.

The analog signals are digitized in the end office by a device called a codec (coder-decoder), producing a series of 8- bit numbers.

The codec makes 8000 samples per second because the Nyquist theorem says that this is sufficient to capture all the information from the 4-kHz telephone channel bandwidth.

At a lower sampling rate, information would be lost; at a higher one, no extra information would be gained. This technique is called PCM (Pulse Code Modulation).

PCM forms the heart of the modern telephone system. As a consequence, virtually all time intervals within the telephone system are multiples of 125 μsec.

When digital transmission began emerging as a feasible technology, CCITT was unable to reach agreement on an international standard for PCM.

Switching

The phone system is divided into two principal parts:

- ☞ outside plant (the local loops and trunks, since they are physically outside the switching offices)

- ☞ inside plant (the switches), which are inside the switching offices. We have just looked at the outside plant.

Two different switching techniques are used nowadays:

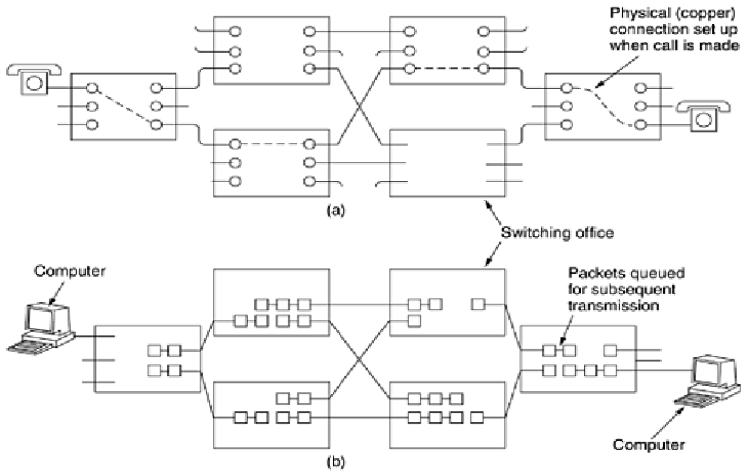
- ☞ circuit switching
- ☞ packet switching.

Circuit Switching

When you or your computer places a telephone call, the switching equipment within the telephone system seeks out a physical path all the way from your telephone to the receiver's telephone. This technique is called circuit switching and is shown schematically

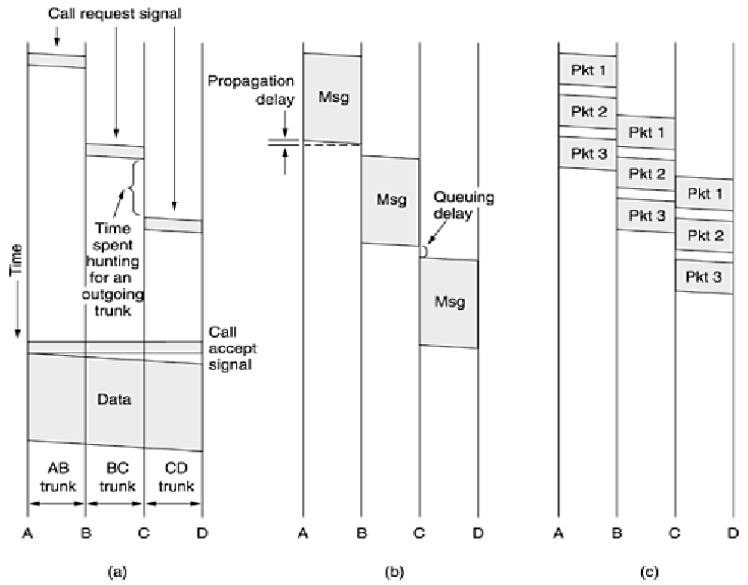
Each of the six rectangles represents a carrier switching office (end office, toll office, etc.). In this example, each office has three incoming lines and three outgoing lines. When a call passes through a switching office, a physical connection is (conceptually) established between the line on which the call came in and one of the output lines, as shown by the dotted lines.

Figure (a) Circuit switching. (b) Packet switching.



The model shown in Fig (a) is highly simplified, of course, because parts of the physical path between the two telephones may, in fact, be microwave or fiber links onto which thousands of calls are multiplexed.

Figure Timing of events in (a) circuit switching, (b) message switching, (c) packet switching.



The alternative to circuit switching is packet switching, shown in Fig..With this technology, individual packets are sent as need be, with no dedicated path being set up in advance. It is up to each packet to find its way to the destination on its own. An important property of circuit switching is the need to set up an end-to-end path before any data can be sent. The elapsed time between the end of dialing and the start of ringing can easily be 10 sec, more on long-distance or international calls. During this time interval, the telephone system is hunting for a path, as shown in Fig.

Message Switching

An alternative switching strategy is message switching, illustrated in Fig (b). When this form of switching is used, no physical path is established in advance between sender and receiver. Instead, when the sender has a block of data to be sent, it is stored in the first switching office (i.e., router) and then forwarded later, one hop at a time. Each block is received in its entirety, inspected for errors, and then retransmitted. A network using this technique is called a store-and-forward network.

The first electromechanical telecommunication systems used message switching, namely, for telegrams. The message was punched on paper tape (off-line) at the sending office, and then read in and transmitted over a communication line to the next office along the way, where it was punched out on paper tape. An operator there tore the tape off and read it in on one of the many tape readers, one reader per outgoing trunk. Such a switching office was called a torn tape office. Paper tape is long gone and message switching is not used any more.

Packet Switching

With message switching, there is no limit at all on block size, which means that routers (in a modern system) must have disks to buffer long blocks. It also means that a single block can tie up a router-router line for minutes, rendering message switching useless for interactive traffic. Packet-switching networks place a tight upper limit on block size, allowing packets to be buffered in router main memory instead of on disk.

- packet-switching networks are well suited for handling interactive traffic.:
- the first packet of a multi packet message can be forwarded before the second one has fully arrived, reducing delay and improving throughput. For these reasons, computer networks are usually packet switched, occasionally circuit switched, but never message switched.

Circuit switching and packet switching differ in many respects. To start with, circuit switching requires that a circuit be set up end to end before communication begins. Packet switching does not require any advance setup. The first packet can just be sent as soon as it is available. The result of the

connection setup with circuit switching is the reservation of bandwidth all the way from the sender to the receiver. All packets follow this path. Among other properties, having all packets follow the same path means that they cannot arrive out of order. With packet switching there is no path, so different packets can follow different paths, depending on network conditions at the time they are sent. They may arrive out of order. Packet switching is more fault tolerant than circuit switching

THE MOBILE TELEPHONE SYSTEM

The traditional telephone system, even if it someday gets multi gigabit end-to-end fiber, will still not be able to satisfy a growing group of users: people on the go. People now expect to make phone calls and to use their phones to check email and surf the Web from airplanes, cars, swimming pools, and while jogging in the park. Consequently, there is a tremendous amount of interest in wireless telephone.

Mobile phones (sometimes called **cell phones**) have gone through three distinct generations, widely called **1G**, **2G**, and **3G**. The generations are:

1. Analog voice.
2. Digital voice.
3. Digital voice and data (Internet, email, etc.).

(Mobile phones should not be confused with **cordless phones** that consist of a base station and a handset sold as a set for use within the home. These are never used for networking, so we will not examine them further.)

First-Generation (1G) Mobile Phones: Analog Voice

- Enough about the politics and marketing aspects of mobile phones. Now let us look at the technology, starting with the earliest system. Mobile radiotelephones were used sporadically for maritime and military communication during the early decades of the 20th century.
- This system used a single large transmitter on top of a tall building and had a single channel, used for both sending and receiving. In the 1960s, **IMTS** (**Improved Mobile Telephone System**) was installed. It, too, used a high-powered (200-watt) transmitter on top of a hill but it had two frequencies, one for sending and one for receiving, so the push-to-talk button was no longer needed.

Advanced Mobile Phone System

All that changed with **AMPS** (**Advanced Mobile Phone System**), invented by Bell Labs and first installed in the United States in 1982. It was also used in England, where it was called TACS, and in Japan, where it was called MCS-L1. AMPS was formally retired in 2008, but we will look at it to understand the con-text for the 2G and 3G systems that improved on it.

In all mobile phone systems, a geographic region is divided up into **cells**, which is why the devices are sometimes called cell one. In AMPS, the cells are typically 10 to 20 km across; in digital systems, the cells are smaller. Each cell uses some set of frequencies not used by any of its neighbors.

The key idea that gives cellular systems far more capacity than previous systems is the use of relatively small cells and the reuse of transmission frequencies in nearby (but not adjacent) cells. Whereas an IMTS system 100 km across can have only one call on each frequency, an AMPS system might have 100 10-km cells in the same area and be able to have 10 to 15 calls on each frequency, in widely separated cells. Thus, the cellular design increases the system capacity by at least an order of magnitude, more as the cells get smaller.

Channels

AMPS uses FDM to separate the channels. The system uses 832 full-duplex channels, each consisting of a pair of simplex channels. This arrangement is known as **FDD** (**Frequency Division Duplex**).

The 832 simplex channels from 824 to 849 MHz are used for mobile to base station transmission, and 832 simplex channels from 869 to 894 MHz are used for base station to mobile transmission. Each of these simplex channels is 30 kHz wide.

The 832 channels are divided into four categories. Control channels (base to mobile) are used to manage the system. Paging channels (base to mobile) alert mobile users to calls for them. Access channels (bidirectional) are used for call setup and channel assignment

Call Management

Each mobile telephone in AMPS has a 32-bit serial number and a 10-digit telephone number in its programmable read-only memory. The telephone number is represented as a 3-digit area code in 10 bits and a 7-digit subscriber number in 24 bits.

When a phone is switched on, it scans a preprogrammed list of 21 control channels to find the most powerful signal. The phone then broadcasts its 32-bit serial number and 34-bit telephone number. Like all the control information in AMPS, this packet is sent in digital form, multiple times, and with an error code, even though the voice channels themselves are analog.

Second-Generation (2G) Mobile Phones: Digital Voice

The first generation of mobile phones was analog; the second generation is digital. Switching to digital has several advantages. It provides capacity gains by allowing voice signals to be digitized and compressed.

It improves security by al-lowing voice and control signals to be encrypted. This in turn deters fraud and eavesdropping, whether from intentional scanning or echoes of other calls due to RF propagation. Finally, it enables new services such as text messaging.

Just as there was no worldwide standardization during the first generation, there was also no worldwide standardization during the second, either. Several different systems were developed, and three have been widely deployed. **D-AMPS** (**Digital Advanced Mobile Phone System**) is a digital version of AMPS that coexists with AMPS and uses TDM to place multiple calls on the same frequency channel. It is described in International Standard IS-54 and its successor IS-136.

Also, the name **PCS** (**Personal Communications Services**) is sometimes used in the marketing literature to indicate a second-generation (i.e., digital) system. Originally it meant a mobile phone using the 1900 MHz band, but that distinction is rarely made now.

GSM—The Global System for Mobile Communications

GSM started life in the 1980s as an effort to produce a single European 2G standard. The task was assigned to a telecommunications group called (in French) Groupe Speciale´ Mobile.

The first GSM systems were deployed starting in 1991 and were a quick success. It soon became clear that GSM was going to be more than a European success, with uptake stretching to countries as far away as Australia, so GSM was renamed to have a more worldwide appeal.

GSM and the other mobile phone systems we will study retain from 1G systems a design based on cells, frequency reuse across cells, and mobility with handoffs as subscribers move. It is the details that differ. Here, we will briefly discuss some of the main properties of GSM. However, the printed GSM standard is over 5000 [sic] pages long.

Third-Generation (3G) Mobile Phones: Digital Voice and Data

The first generation of mobile phones was analog voice, and the second generation was digital voice. The third generation of mobile phones, or **3G** as it is called, is all about digital voice *and* data.

A number of factors are driving the industry. First, data traffic already exceeds voice traffic on the fixed network and is growing exponentially, whereas voice traffic is essentially flat.

Many industry experts expect data traffic to dominate voice on mobile devices as well soon. Second, the telephone, entertainment, and computer industries have all gone digital and are rapidly converging. Many people are drooling over lightweight, portable devices that act as a telephone, music and video player, email terminal, Web interface, gaming machine, and more, all with worldwide wireless connectivity to the Internet at high bandwidth.

ITU tried to get a bit more specific about this vision starting back around 1992. It issued a blueprint for getting there called **IMT-2000**, where IMT stood for **International Mobile Telecommunications** .The basic services that the IMT-2000 network was supposed to provide to its users are:

1. High-quality voice transmission.
2. Messaging (replacing email, fax, SMS, chat, etc.).
3. Multimedia (playing music, viewing videos, films, television, etc.).
4. Internet access (Web surfing, including pages with audio and video).

Additional services might be video conferencing, tale presence, group game playing, and m-commerce (waving your telephone at the cashier to pay in a store).

. First, CDMA can improve capacity by taking advantage of small periods when some transmitters are silent. CDMA, by simply not transmitting one user lowers the interference for other users, and it is likely that some fraction of users will not be transmitting in a busy cell at any given time. Thus CDMA takes advantage of expected silences to allow a larger number of simultaneous calls.

Another improvement over the basic CDMA scheme we described earlier is to allow different users to send data at different rates. This trick is accomplished naturally in CDMA by fixing the rate at which chips are transmitted and assigning users chip sequences of different lengths

. Users tend to notice this, but it is inevitable occasionally with the current design. Hard handoff is the norm with FDM designs to avoid the cost of having the mobile transmit or receive on two frequencies simultaneously.

Figure 1. Soft handoff (a) before, (b) during, and (c) after.

Much has been written about 3G, most of it praising it as the greatest thing since sliced bread. Meanwhile, many operators have taken cautious steps in the direction of 3G by going to what is sometimes called **2.5G**, although 2.1G might be more accurate. One such system is **EDGE** (**Enhanced Data rates for GSM Evolution**), which is just GSM with more bits per symbol.

The trouble is, more bits per symbol also means more errors per symbol, so EDGE has nine different schemes for modulation and error correction, differing in terms of how much of the bandwidth is devoted to fixing the errors introduced by the higher speed.

EDGE is one step along an evolutionary path that is defined from GSM to WCDMA. Similarly, there is an evolutionary path defined for operators to upgrade from IS-95 to CDMA2000 networks.

Even though 3G networks are not fully deployed yet, some researchers regard 3G as a done deal. These people are already working on 4G systems under the name of **LTE** (**Long Term Evolution**).

----------UNIT I COMPLETED----------

UNIT – II

THE DATA LINK LAYER

The design principles for layer 2, the data link layer deals with the algorithms for achieving reliable, efficient communication between two adjacent machines at the data link layer. By adjacent, we mean that the two machines are connected by a communication channel that acts conceptually like a wire .The essential property of a channel that makes it "wirelike" is that the bits are delivered in exactly the same order in which they are sent.

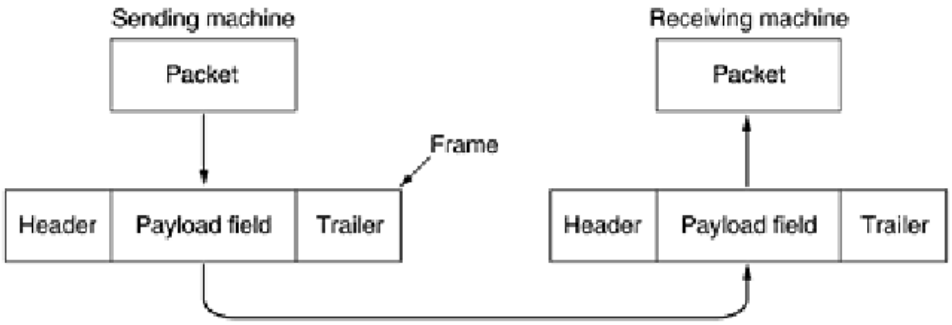
Furthermore, they have only a finite data rate, and there is a nonzero propagation delay between the time a bit is sent and the time it is received. These limitations have important implications for the efficiency of the data transfer.

DATA LINK LAYER DESIGN ISSUES

The data link layer has a number of specific functions it can carry out. These functions include

1. Providing a well-defined service interface to the network layer.
2. Dealing with transmission errors.
3. Regulating the flow of data so that slow receivers are not swamped by fast senders.

To accomplish these goals, the data link layer takes the packets it gets from the network layer and encapsulates them into frames for transmission. Each frame contains a frame header, a payload field for holding the packet, and a frame trailer. Frame management forms the heart of what the data link layer does.



Services Provided to the Network Layer

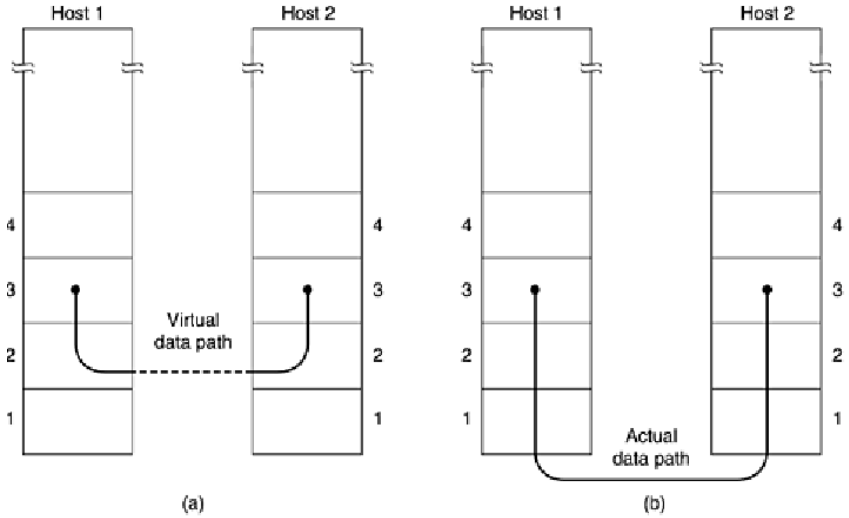
The function of the data link layer is to provide services to the network layer.

- The principal service is transferring data from the network layer on the source machine to the network layer on the destination machine.
- On the source machine is an entity, call it a process, in the network layer that hands some bits to the data link layer for transmission to the destination.
- The job of the data link layer is to transmit the bits to the destination machine so they can be handed over to the network layer there, as shown in Fig.
- The actual transmission follows the path of Fig. (b), but it is easier to think in terms of two data link layer processes communicating using a data link protocol

The data link layer can be designed to offer various services. The actual services offered can vary from system to system. Three reasonable possibilities that are commonly provided are

1. Unacknowledged connectionless service.
2. Acknowledged connectionless service.
3. Acknowledged connection-oriented service.

Figure 3-2. (a) Virtual communication. (b) Actual communication.



Unacknowledged connectionless service consists of having the source machine send independent frames to the destination machine without having the destination machine acknowledge them. No logical connection is established beforehand or released afterward. If a frame is lost due to noise on the line, no attempt is made to detect the loss or recover from it in the data link layer. This class of service is appropriate when the error rate is very low so that recovery is left to higher layers.

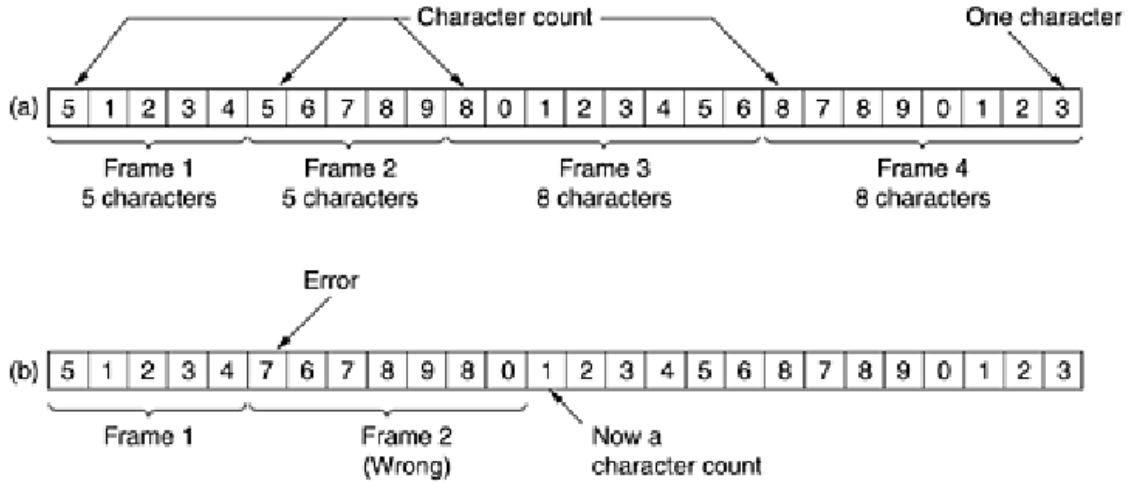
It is also appropriate for real-time traffic, such as voice, in which late data are worse than bad data. Most LANs use unacknowledged connectionless service in the data link layer. The next step up in terms of reliability is acknowledged connectionless service. When this service is offered, there are still no logical connections used, but each frame sent is individually acknowledged. In this way, the sender knows whether a frame has arrived correctly. If it has not arrived within a specified time interval, it can be sent again. This service is useful over unreliable channels, such as wireless systems. The network layer can always send a packet and wait for it to be acknowledged.

If the acknowledgement is not forthcoming before the timer expires, the sender can just send the entire message again. The trouble with this strategy is that frames usually have a strict maximum length imposed by the hardware and network layer packets do not. If the average packet is broken up into, say, 10 frames, and 20 percent of all frames are lost, it may take a very long time for the packet to get through. If individual frames are acknowledged and retransmitted, entire packets get through much faster. On reliable channels, such as fiber, the overhead of a heavyweight data link protocol may be unnecessary, but on wireless channels, with their inherent unreliability, it is well worth the cost.

Since it is too risky to count on timing to mark the start and end of each frame, other methods have been devised. There are four methods:

1. Character count.
2. Flag bytes with byte stuffing.
3. Starting and ending flags, with bit stuffing.
4. Physical layer coding violations.

A character stream. (a) Without errors. (b) With one error.



Framing

To provide service to the network layer, the data link layer must use the ser-vice provided to it by the physical layer. What the physical layer does is accept a raw bit stream and attempt to deliver it to the destination. If the channel is noisy, as it is for most wireless and some wired links, the physical layer will add some redundancy to its signals to reduce the bit error rate to a tolerable level. However, the bit stream received by the data link layer is not guaranteed to be error free.

Some bits may have different values and the number of bits received may be less than, equal to, or more than the number of bits transmitted. It is up to the data link layer to detect and, if necessary, correct errors.

The usual approach is for the data link layer to break up the bit stream into discrete frames, compute a short token called a checksum for each frame, and include the checksum in the frame when it is transmitted .Breaking up the bit stream into frames is more difficult than it at first appears. A good design must make it easy for a receiver to find the start of new frames while using little of the channel bandwidth.

Error Control

Having solved the problem of marking the start and end of each frame, the next problem is how to make sure all frames are eventually delivered to the network layer at the destination and in the proper order.

Suppose that the sender just kept outputting frames without regard to whether they were arriving properly. This might be fine for unacknowledged connectionless service, but would most certainly not be fine for reliable, connection-oriented service.

The usual way to ensure reliable delivery is to provide the sender with some feedback about what is happening at the other end of the line. Typically, the protocol calls for the receiver to send back special control frames bearing positive or negative acknowledgements about the incoming frames.

If the sender receives a positive acknowledgement about a frame, it knows the frame has arrived safely. On the other hand, a negative acknowledgement means that something has gone wrong, and the frame must be transmitted again.

Flow Control

Another important design issue that occurs in the data link layer (and higher layers as well) is what to do with a sender that systematically wants to transmit frames faster than the receiver can accept them. This situation can easily occur when the sender is running on a fast (or lightly loaded) computer and the receiver is running on a slow (or heavily loaded) machine. The sender keeps pumping the frames

out at a high rate until the receiver is completely swamped. Even if the transmission is error free, at a certain point the receiver will simply be unable to handle the frames as they arrive and will start to lose some. Clearly, something has to be done to prevent this situation.
Two approaches are commonly used.

- In the first one, feedback-based flow control, the receiver sends back information to the sender giving it permission to send more data or at least telling the sender how the receiver is doing.
- In the second one, rate based flow control, the protocol has a built-in mechanism that limits the rate at which senders may transmit data, without using feedback from the receiver.

ERROR DETECTION AND CORRECTION

The telephone system has three parts:

- The switches,
- The interoffice trunks,
- The local loops**.**

The first two are now almost entirely digital in most developed countries. The local loops are still analog twisted copper pairs and will continue to be so for years due to the enormous expense of replacing them. While errors are rare on the digital part, they are still common on the local loops. Furthermore, wireless communication is becoming more common, and the error rates here are orders of magnitude worse than on the interoffice fiber trunks. The conclusion is: transmission errors are going to be with us for many years to come.

Error-Correcting Codes

Network designers have developed two basic strategies for dealing with errors.

- One way is to include enough redundant information along with each block of data sent, to enable the receiver to deduce what the transmitted data must have been
- . The other way is to include only enough redundancy to allow the receiver to deduce that an error occurred, but not which error, and have it request a retransmission.
- The former strategy uses error correcting codes and the latter uses error-detecting codes.
- The use of error-correcting codes is often referred to as forward error correction.

Error-Detecting Codes

Error-correcting codes are widely used on wireless links, which are notoriously noisy and error prone when compared to copper wire or optical fibers. Without error-correcting codes, it would be hard to get anything through. However, over copper wire or fiber, the error rate is much lower, so error detection and retransmission is usually more efficient there for dealing with the occasional error.
The algorithm for computing the checksum is as follows:

1. Let r be the degree of G(x). Append r zero bits to the low-order end of the frame so it now contains m + r bits and corresponds to the polynomial x M r (x).

2. Divide the bit string corresponding to G(x) into the bit string corresponding to x M r (x), using modulo 2 division.

3. Subtract the remainder (which is always r or fewer bits) from the bit string corresponding to x M r (x) using modulo 2 subtraction. The result is the check summed frame to be transmitted. Call its polynomial T(x).

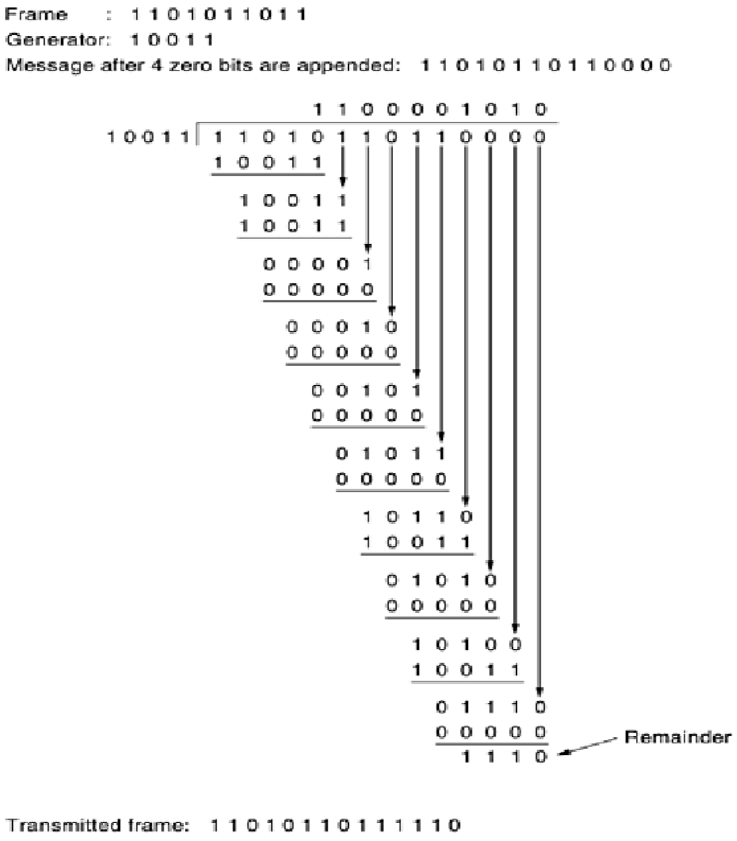


Figure 1. Calculation of the polynomial code checksum.

ELEMENTARY DATA LINK PROTOCOLS

Consider a protocol that is as simple as it can be. Data are transmitted in one direction only. Both the transmitting and receiving network layers are always ready. Processing time can be ignored. Infinite buffer space is available.

To introduce the subject of protocols, we will begin by looking at three protocols of increasing complexity. For interested readers, a simulator for these and subsequent protocols is available via the Web (see the preface). Before we look at the protocols, it is useful to make explicit some of the assumptions underlying the model of communication. The physical layer process and some of the data link layer process run on dedicate hardware called a NIC (Network Interface Card).

The rest of the link layer process and the network layer process run on the main CPU as part of the operating system, with the soft-ware for the link layer process often taking the form of a **device driver**.

A Utopian Simplex Protocol

As an initial example we will consider a protocol that is as simple as it can be because it does not worry about the possibility of anything going wrong. Data are transmitted in one direction only. Both the transmitting and receiving network layers are always ready. Processing time can be ignored. Infinite buffer space is available. And best of all, the communication channel between the data link layers never damages or loses frames.

The protocol consists of two distinct procedures, a sender and a receiver. The sender runs in the data link layer of the source machine, and the receiver runs in the data link layer of the destination machine. No sequence numbers or acknowledgements are used here, so *MAX SEQ* is not needed. The only event

type possible is *frame arrival* (i.e., the arrival of an undamaged frame). The sender is in an infinite while loop just pumping data out onto the line as fast as it can. The body of the loop consists of three actions: go fetch a packet from the (always obliging) network layer, construct an outbound frame using the variable *s*, and send the frame on its way.

A Simplex Stop-and-Wait Protocol for an Error-Free Channel

Protocols in which the sender sends one frame and then waits for an acknowledgement before proceeding are called **stop-and-wait**. Now we will tackle the problem of preventing the sender from flooding the receiver with frames faster than the latter is able to process them. This situation can easily happen in practice so being able to prevent it is of great importance.

As in protocol 1, the sender starts out by fetching a packet from the network layer, using it to construct a frame, and sending it on its way. But now, unlike in protocol 1, the sender must wait until an acknowledgement frame arrives before looping back and fetching the next packet from the network layer. The sending data link layer need not even inspect the incoming frame as there is only one possibility. The incoming frame is always an acknowledgement.

The only difference between *receiver1* and *receiver2* is that after delivering a packet to the network layer, *receiver2* sends an acknowledgement frame back to the sender before entering the wait loop again. Because only the arrival of the frame back at the sender is important, not its contents, the receiver need not put any particular information in it.

A Simplex Stop-and-Wait Protocol for a Noisy Channel

Let us consider the normal situation of a communication channel that makes errors. Frames may be either damaged or lost completely. However, we assume that if a frame is damaged in transit, the receiver hardware will detect this when it computes the checksum.

To see what might go wrong, remember that the goal of the data link layer is to provide error-free, transparent communication between network layer processes. The network layer on machine *A* gives a series of packets to its data link layer, which must ensure that an identical series of packets is delivered to the net-work layer on machine *B* by its data link layer. In particular, the network layer on *B* has no way of knowing that a packet has been lost or duplicated, so the data link layer must guarantee that no combination of transmission errors, however unlikely, can cause a duplicate packet to be delivered to a network layer.

Consider the following scenario:

1. The network layer on *A* gives packet 1 to its data link layer. The packet is correctly received at *B* and passed to the network layer on *B*. *B* sends an acknowledgement frame back to *A*.
2. The acknowledgement frame gets lost completely. It just never ar-rives at all. Life would be a great deal simpler if the channel mangle and lost only data frames and not control frames, but sad to say, the channel is not very discriminating.
3. The data link layer on *A* eventually times out. Not having received an acknowledgement, it (incorrectly) assumes that its data frame was lost or damaged and sends the frame containing packet 1 again.
4. The duplicate frame also arrives intact at the data link layer on *B* and is unwittingly passed to the network layer there. If *A* is sending a file to *B*, part of the file will be duplicated (i.e., the copy of the file made by *B* will be incorrect and the error will not have been detected). In other words, the protocol will fail.

SLIDING WINDOW PROTOCOLS

In the previous protocols, data frames were transmitted in one direction only. In most practical situations, there is a need to transmit data in both directions. One way of achieving full-duplex data transmission is to run two instances of one of the previous protocols, each using a separate link for simplex data traffic (in different directions). Each link is then comprised of a ‘‘forward’’ channel (for data) and a ‘‘reverse’’ channel (for acknowledgements). In both cases the capacity of the reverse channel is almost entirely wasted.

A better idea is to use the same link for data in both directions. After all, in protocols 2 and 3 it was already being used to transmit frames both ways, and the reverse channel normally has the same capacity as the forward channel.

A One-Bit Sliding Window Protocol

Before tackling the general case, let us examine a sliding window protocol with a window size of 1. Such a protocol uses stop-and-wait since the sender transmits a frame and waits for its acknowledgement before sending the next one. *Next frame to send* tells which frame the sender is trying to send. Similarly, *frame expected* tells which frame the receiver is expecting. In both cases, 0 and 1 are the only possibilities.

Under normal circumstances, one of the two data link layers goes first and transmits the first frame. In other words, only one of the data link layer programs should contain the *to physical layer* and *start timer* procedure calls outside the main loop.

The starting machine fetches the first packet from its network layer, builds a frame from it, and sends it. When this (or any) frame arrives, the receiving data link layer checks to see if it is a duplicate, just as in protocol 3. If the frame is the one expected, it is passed to the network layer and the receiver’s window is slid up.

The acknowledgement field contains the number of the last frame received without error. If this number agrees with the sequence number of the frame the sender is trying to send, the sender knows it is done with the frame stored in *buff-er* and can fetch the next packet from its network layer. If the sequence number disagrees, it must continue trying to send the same frame. Whenever a frame is received, a frame is also sent back.

A Protocol Using Go-Back-N

Until now we have made the tacit assumption that the transmission time required for a frame to arrive at the receiver plus the transmission time for the acknowledgement to come back is negligible. Sometimes this assumption is clearly false. In these situations the long round-trip time can have important implications for the efficiency of the bandwidth utilization. As an example, consider a 50-kbps satellite channel with a 500-msec round-trip propagation delay.

The problem described here can be viewed as a consequence of the rule requiring a sender to wait for an acknowledgement before sending another frame. If we relax that restriction, much better efficiency can be achieved. Basically, the solution lies in allowing the sender to transmit up to *w* frames before blocking, in-stead of just 1.

For smaller window sizes, the utilization of the link will be less than 100% since the sender will be blocked sometimes. We can write the utilization as the fraction of time that the sender is not blocked:

$$\text{link utilization } \delta \frac{w}{1 + 2BD}$$

This value is an upper bound because it does not allow for any frame processing time and treats the acknowledgement frame as having zero length, since it is usually short. The equation shows the need for having a large window *w* when-ever the bandwidth-delay product is large. This technique of keeping multiple frames in flight is an example of pipelining. Pipelining frames over an unreliable communication channel raises some serious issues. First, what happens if a frame in the middle of a long stream is damaged or lost? Large numbers of succeeding frames will arrive at the receiver before the sender even finds out that anything is wrong.

A Protocol Using Selective Repeat

The go-back-n protocol works well if errors are rare, but if the line is poor it wastes a lot of bandwidth on retransmitted frames. An alternative strategy, the selective repeat protocol, is to allow the receiver to accept and buffer the frames following a damaged or lost one.

In this protocol, both sender and receiver maintain a window of outstanding and acceptable sequence numbers, respectively. The sender's window size starts out at 0 and grows to some predefined maximum. The receiver's window, in contrast, is always fixed in size and equal to the predetermined maximum. The receiver has a buffer reserved for each sequence number within its fixed window. Associated with each buffer is a bit (*arrived*) telling whether the buffer is full or empty. Whenever a frame arrives, its sequence number is checked by the function *between* to see if it falls within the window. If so and if it has not already been received, it is accepted and stored.

This action is taken without regard to whether or not the frame contains the next packet expected by the network layer. Of course, it must be kept within the data link layer and not passed to the network layer until all the lower-numbered frames have already been delivered to the network layer in the correct order.

The protocol should operate correctly despite this disaster. The sender eventually times out and re-transmits frame 0. When this frame arrives at the receiver, a check is made to see if it falls within the receiver's window. It is essential that the timeout associated with the auxiliary timer be appreciably shorter than the timeout used for timing out data frames. This condition is required to ensure that a correctly received frame is acknowledged early enough that the frame's retransmission timer does not expire and retransmit the frame.

THE MEDIUM ACCESS CONTROL SUBLAYER

Network links can be divided into two categories: those using point-to-point connections and those using broadcast channels. The MAC sub layer is the bottom part of the data link layer, so logically we should have studied it examining all the point-to-point protocols. it is easier to understand protocols involving multiple parties after two-party protocols.

THE CHANNEL ALLOCATION PROBLEM

The central theme of this chapter is how to allocate a single broadcast channel among competing users. The channel might be a portion of the wireless spectrum in a geographic region, or a single wire or optical fiber to which multiple nodes are connected. It does not matter. In both cases, the channel connects each user to all other users and any user who makes full use of the channel interferes with other users who also wish to use the channel.

Static Channel Allocation

The traditional way of allocating a single channel, such as a telephone trunk, among multiple competing users is to chop up its capacity by using one of the multiplexing schemes. If there are N users, the bandwidth is divided into N equal-sized portions, with each user being assigned one portion. Since each user has a private frequency band, there is now no interference among users. When there is only a small and constant number of users, each of which has a steady stream or a heavy load of traffic, this division is a simple and efficient allocation mechanism.

A wireless example is FM radio stations. Each station gets a portion of the FM band and uses it most of the time to broadcast its signal.

The poor performance of static FDM can easily be seen with a simple queueing theory calculation. Let us start by finding the mean time delay, T, to send a frame onto a channel of capacity C bps. We assume that the frames arrive randomly with an average arrival rate of λ frames/sec, and that the frames vary in length with an average length of $1/\mu$ bits. With these parameters, the service rate of the channel is μC frames/sec. A standard queueing theory result is

$$T = \frac{1}{\mu C - \lambda}$$

(For the curious, this result is for an ''M/M/1'' queue. It requires that the randomness of the times between frame arrivals and the frame lengths follow an exponential distribution, or equivalently be the result of a Poisson process.) Precisely the same arguments that apply to FDM also apply to other ways of statically dividing the channel. If we were to use time division multiplexing (TDM) and allocate each user every Nth time slot, if a user does not use the allocated.

Assumptions for Dynamic Channel Allocation

1. **Independent Traffic**. The model consists of N independent **stations** (e.g., computers, telephones), each with a program or user that generates frames for transmission. The expected number of frames generated in an interval of length t is $\lambda \Delta t$, where λ is a constant (the arrival rate of new frames). Once a frame has been generated, the station is blocked and does nothing until the frame has been successfully transmitted.

2. **Single Channel**. A single channel is available for all communication. All stations can transmit on it and all can receive from it. The stations are assumed to be equally capable, though protocols may assign them different roles (e.g., priorities).

3. **Observable Collisions**. If two frames are transmitted simultaneously, they overlap in time and the resulting signal is garbled. This event is called a **collision**. All stations can detect that a collision has occurred. A collided frame must be transmitted again later. No errors other than those generated by collisions occur.

4. **Continuous or Slotted Time**. Time may be assumed continuous, in which case frame transmission can begin at any instant. Alternatively, time may be slotted or divided into discrete intervals (called slots). Frame transmissions must then begin at the start of a slot. A slot may contain 0, 1, or more frames, corresponding to an idle slot, a successful transmission, or a collision, respectively.

5. **Carrier Sense or No Carrier Sense**. With the carrier sense assumption, stations can tell if the channel is in use before trying to use it. No station will attempt to use the channel while it is sensed as busy. If there is no carrier sense, stations cannot sense the channel before trying to use it. They just go ahead and transmit. Only later can they determine whether the transmission was successful.

MULTIPLE ACCESS PROTOCOLS

Many algorithms for allocating a multiple access channel are known. In the following sections, we will study a small sample of the more interesting ones and give some examples of how they are commonly used in practice.

ALOHA

- ✓ In this case, ‘‘pristine’’ can be interpreted as ‘‘not having a working telephone sys-tem.’’ This did not make life more pleasant for researcher Norman Abramson and his colleagues at the University of Hawaii who were trying to connect users on re-mote islands to the main computer in Honolulu. Stringing their own cables under the Pacific Ocean was not in the cards, so they looked for a different solution.

- ✓ The one they found used short-range radios, with each user terminal sharing the same upstream frequency to send frames to the central computer.
- ✓ It included a simple and elegant method to solve the channel allocation problem. Their work has been extended by many researchers since then (Schwartz and Abramson, 2009). Although Abramson’s work, called the ALOHA system, used ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single shared channel.

Pure ALOHA

- ✓ The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent. There will be collisions, of course, and the colliding frames will be damaged. Senders need some way to find out if this is the case.
- ✓ In the ALOHA system, after each station has sent its frame to the central computer, this computer rebroadcasts the frame to all of the stations. A sending station can thus listen for the broadcast from the hub to see if its frame has gotten through. In other systems, such as wired
- ✓ LANs, the sender might be able to listen for collisions while transmitting.
- ✓ If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems

in which multiple users share a common channel in a way that can lead to conflicts are known as **contention** systems.

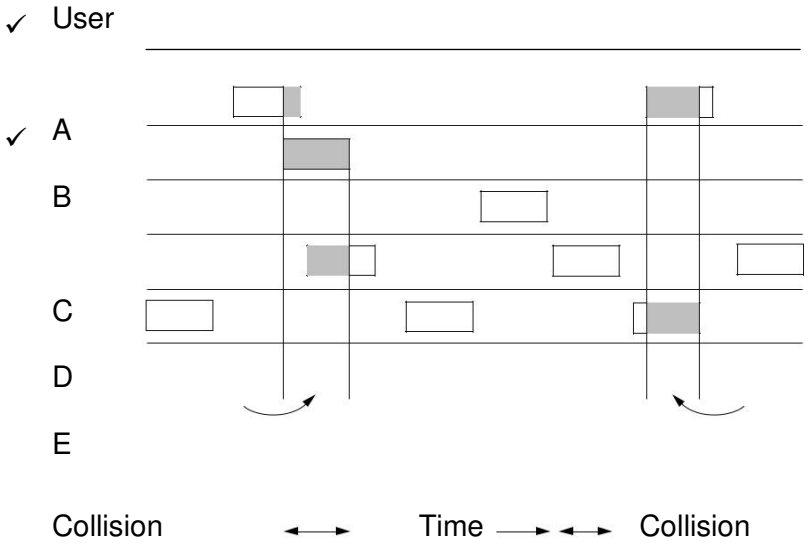


Figure 1. In pure ALOHA, frames are transmitted at completely arbitrary times.

A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. . The station then transmits a frame containing the line over the shared channel to the central computer and checks the channel to see if it was successful. If so, the user sees the reply and goes back to typing. If not, the user continues to wait while the station retransmits the frame over and over until it has been successfully sent.

Slotted ALOHA

Soon after ALOHA came onto the scene, Roberts (1972) published a method for doubling the capacity of an ALOHA system. His proposal was to divide time into discrete intervals called **slots**, each interval corresponding to one frame.

In Roberts' method, which has come to be known as **slotted ALOHA**—in contrast to Abramson's **pure ALOHA**—a station is not permitted to send when-ever the user types a line. Instead, it is required to wait for the beginning of the next slot. Thus, the continuous time ALOHA is turned into a discrete time one. This halves the vulnerable period.

The probability that it will avoid a collision is e^{-G}, which is the probability that all the other stations are silent in that slot. The probability of a collision is then just $1 - e^{-G}$. The probability of a transmission requiring exactly k attempts (i.e., $k - 1$ collisions followed by one success) is

$$P_k = e^{-G}(1 - e^{-G})^{k-1}$$

The expected number of transmissions, E, per line typed at a terminal.

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**. A number of them have been proposed, and they were long ago analyzed in detail. These networks can achieve a much better utilization than $1/e$. In this section, we will discuss some protocols for improving performance.

Persistent and Non persistent CSMA

- The first carrier sense protocol that we will study here is called **1-persistent CSMA** (**Carrier Sense Multiple Access**). That is a bit of a mouthful for the simplest CSMA scheme. 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 idle, the stations sends its data. Otherwise, if the channel is busy, the station just waits until it becomes idle. Then the station 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.
- This chance depends on the number of frames that fit on the channel, or the **bandwidth-delay product** of the channel. If only a tiny fraction of a frame fits on the channel, which is the case in most LANs since the propagation delay is small, the chance of a collision happening is small. The larger the bandwidth-delay product, the more important this effect becomes, and the worse the
- performance of the protocol.
- Even so, this protocol has better performance than pure ALOHA because both stations have the decency to desist from interfering with the third station's frame. Exactly the same holds for
- slotted ALOHA.
- A second carrier sense protocol is **non persistent CSMA**. In this protocol, a conscious attempt is made to be less greedy than in the previous one. As before, a station senses the channel when it wants to send a frame, and 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.

CSMA with Collision Detection

- Persistent and non persistent CSMA protocols are definitely an improvement over ALOHA because they ensure that no station begins to transmit while the channel is busy. However, if two stations sense the channel to be idle and begin transmitting simultaneously, their signals will still collide.
- Another improvement is for the stations to quickly detect the collision and abruptly stop transmitting, (rather than finishing them) since they are irretrievably garbled anyway. This strategy saves time and bandwidth.
- This protocol, known as **CSMA/CD** (**CSMA with Collision Detection**), is the basis of the classic Ethernet LAN, so it is worth devoting some time to looking at it in detail. It is important to realize that collision detection is an analog process.
- The station's hardware must listen to the channel while it is transmitting. If the signal it reads back is different from the signal it is putting out, it knows that a collision is occurring.

COLLISION-FREE PROTOCOLS

- In the protocols to be described, we assume that there are exactly N stations, each programmed with a unique address from 0 to $N - 1$. It does not matter that some stations may be inactive part of the time. We also assume that propagation delay is negligible.
- Most of these protocols are not currently used in major systems, but in a rapidly changing field, having some protocols with excellent properties available for future systems is often a good thing.

A Bit-Map Protocol

In our first collision-free protocol, the **basic bit-map method**, each contention period consists of exactly *N* slots. If station 0 has a frame to send, it transmits a 1 bit during the slot 0.

No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 bit during slot 1, but only if it has a frame queued. Protocols like this in which the desire to transmit is broadcast before the actual transmission are called **reservation protocols** because they reserve channel ownership in advance and prevent collisions.

Let us briefly analyze the performance of this protocol. For convenience, we will measure time in units of the contention bit slot, with data frames consisting of *d* time units.

The channel efficiency at low load is easy to compute. The overhead per frame is *N* bits and the amount of data is *d* bits, for an efficiency of $d / (d + N)$.

Token Passing

- The essence of the bit-map protocol is that it lets every station transmit a frame in turn in a predefined order. Another way to accomplish the same thing is to pass a small message called a **token** from one station to the next in the same predefined order.

- The token represents permission to send. If a station has a frame queued for transmission when it receives the token, it can send that frame before it passes the token to the next station. If it has no queued frame, it simply passes the token.

- In a **token ring** protocol, the topology of the network is used to define the order in which stations send. The stations are connected one to the next in a single ring.

- The performance of token passing is similar to that of the bit-map protocol, though the contention slots and frames of one cycle are now intermingled.

Binary Countdown

- A problem with the basic bit-map protocol, and by extension token passing, is that the overhead is 1 bit per station, so it does not scale well to networks with thousands of stations. We can do better than that by using binary station ad-dresses with a channel that combines transmissions.
- All addresses are assumed to be the same length. The bits in each ad-dress position from different stations are BOOLEAN OR together by the channel when they are sent at the same time. We will call this protocol **binary count-down**.
- It implicitly assumes that the trans-mission delays are negligible so that all stations see asserted bits essentially instantaneously.

LIMITED-CONTENTION PROTOCOLS

- Each strategy can be rated as to how well it does with respect to the two important performance measures, delay at low load and channel efficiency at high load.
- Under conditions of light load, contention (i.e., pure or slotted ALOHA) is preferable due to its low delay (since collisions are rare). As the load increases, contention becomes increasingly less attractive because the overhead associated with channel arbitration becomes greater.
- Such protocols, which we will call **limited-contention protocols**, do in fact exist, and will conclude our study of carrier sense networks.

The Adaptive Tree Walk Protocol

This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through $N / 2$ and one from the rest. The proc-ess was repeated recursively until the infected soldiers were determined.

. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4's turn during slot 2.

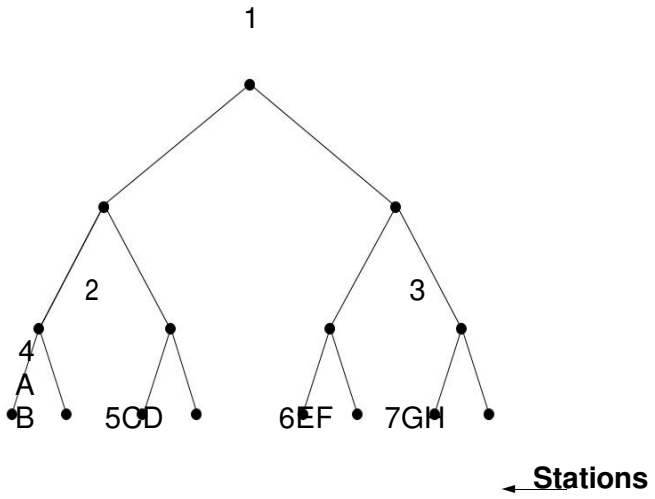


Figure 1 The tree for eight stations

In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node's left and right children. If a bit slot is idle or if only one station transmits in it, the searching of its node can stop because all ready stations have been located. (Were there more than one, there would have been a collision.) When the load on the system is heavy, it is hardly worth the effort to dedicate slot 0 to node 1 because that makes sense only in the unlikely event that precisely one station has a frame to send.

WIRELESS LAN PROTOCOLS

- Such a LAN is an example of a broadcast channel. It also has somewhat different properties than a wired LAN, which leads to different MAC protocols. In this section, we will examine some of these protocols. In Sec. 4.4, we will look at 802.11 (WiFi) in detail.
- A common configuration for a wireless LAN is an office building with access points (APs) strategically placed around the building. The APs are wired together using copper or fiber and provide connectivity to the stations that talk to them.
- A naive approach to using a wireless LAN might be to try CSMA: just listen for other transmissions and only transmit if no one else is doing so. The trouble is, this protocol is not really a good way to think about wireless because what matters for reception is interference at the receiver, not at the sender.
- To see the nature of the problem, consider where four wireless stations are illustrated. For our purposes, it does not matter which are APs and which are laptops. The radio range is such that *A* and *B* are within each other's range and can potentially interfere with one another. *C* can also potentially interfere with both *B* and *D*, but not with *A*.

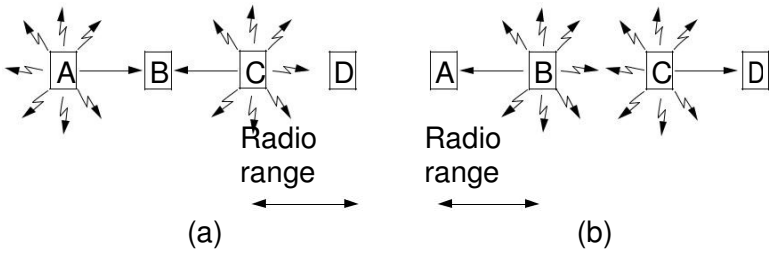
- 

Figure 4-11. A wireless LAN. (a) *A* and *C* are hidden terminals when trans-mitting to *B*. (b) *B* and *C* are exposed terminals when transmitting to *A* and *D*.

- First consider what happens when *A* and *C* transmit to *B*, as depicted in Fig. 4-11(a). If *A* sends and then *C* immediately senses the medium, it will not hear *A* because *A* is out of range. Thus *C* will falsely conclude that it can transmit to *B*. If *C* does start transmitting, it will interfere at *B*, wiping out the frame from *A*. (We assume here that no CDMA-type scheme is used to provide multiple channels, so collisions garble the signal and destroy both frames.) We want a MAC protocol that will prevent this kind of collision from happening because it wastes bandwidth.
- The problem of a station not being able to detect a potential competitor for the medium because the competitor is too far away is called the **hidden terminal problem**.
- Now let us look at a different situation: *B* transmitting to *A* at the same time that *C* wants to transmit to *D,* as shown in Fig. 4-11(b). If *C* senses the medium, it will hear a transmission and falsely conclude that it may not send to *D* (shown as a dashed line). In fact, such a transmission would cause bad reception only in the zone between *B* and *C*, where neither of the intended receivers is located. We want a MAC protocol that prevents this kind of deferral from happening
- because it wastes bandwidth. The problem is called the **exposed terminal problem**.
- The difficulty is that, before starting a transmission, a station really wants to know whether there is radio activity around the receiver.
- An early and influential protocol that tackles these problems for wireless LANs is **MACA** (**Multiple Access with Collision Avoidance**) (Karn, 1990). The basic idea behind it is for the sender to stimulate the receiver into outputting a short frame, so stations nearby can detect this transmission and avoid transmitting for the duration of the upcoming (large) data frame. This technique is used instead of carrier sense.

----------UNIT II COMPLETED----------

UNIT III

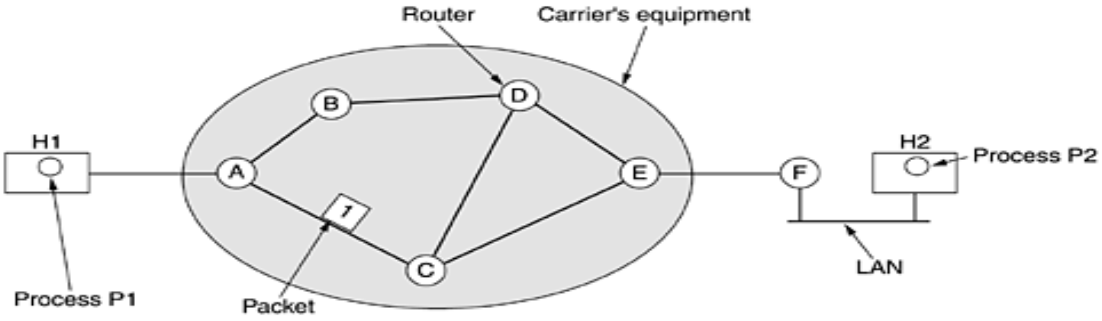
THE NETWORK LAYER

The network layer is concerned with getting packets from the source all the way to the destination.

NETWORK LAYER DESIGN ISSUES

Store-and-Forward Packet Switching

The major components of the system are the carrier's equipment (routers connected by transmission lines), shown inside the shaded oval, and the customers' equipment, shown outside the oval. Host *H1* is directly connected to one of the carrier's routers, *A*, by a leased line. In contrast, *H2* is on a LAN with a router, *F*, owned and operated by the customer. This router also has a leased line to the carrier's equipment. We have shown *F* as being outside the oval because it does not belong to the carrier, but in terms of construction, software, and protocols, it is probably no different from the carrier's routers.



This equipment is used as follows. A host with a packet to send transmits it to the nearest router, either on its own LAN or over a point-to-point link to the carrier. The packet is stored there until it has fully arrived so the checksum can be verified. Then it is forwarded to the next router along the path until it reaches the destination host, where it is delivered. This mechanism is store-and-forward packet switching

Services Provided to the Transport Layer

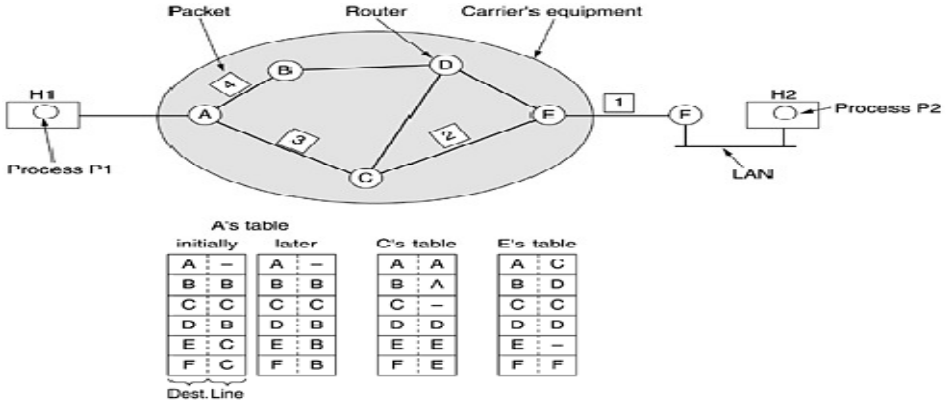
The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is what kind of services the network layer provides to the transport layer. The network layer services have been designed with the following goals in mind.

- The services should be independent of the router technology.
- The transport layer should be shielded from the number, type, and topology of the routers present.
- The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

Implementation of Connectionless Service

If connectionless service is offered, 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**. If connection-oriented service is used, 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**), in analogy with the physical circuits set up by the telephone

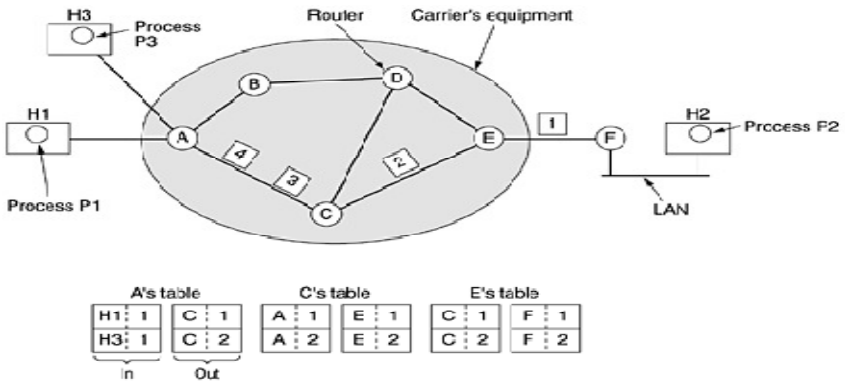
system, and the subnet is called a **virtual-circuit subnet**. In this section we will examine datagram subnets; in the next one we will examine virtual-circuit subnets.



A has only two outgoing lines—to *B* and *C*—so every incoming packet must be sent to one of these routers, even if the ultimate destination is some other router. *A*'s initial routing table is shown in the figure under the label "initially."

As they arrived at *A*, packets 1, 2, and 3 were stored briefly (to verify their checksums). Then each was forwarded to *C* according to *A*'s table. Packet 1 was then forwarded to *E* and then to *F*. When it got to *F*, it was encapsulated in a data link layer frame and sent to *H2* over the LAN. Packets 2 and 3 follow the same route. However, something different happened to packet 4. When it got to *A* it was sent to router *B*, even though it is also destined for *F*. For some reason, *A* decided to send packet 4 via a different route than that of the first three. Perhaps it learned of a traffic jam somewhere along the *ACE* path and updated its routing table, as shown under the label "later." **The algorithm that manages the tables and makes the routing decisions is called the routing algorithm.**

Implementation of Connection-Oriented Service



Host *H1* has established connection 1 with host *H2*. It is remembered as the first entry in each of the routing tables. The first line of *A*'s table says that if a packet bearing connection identifier 1 comes in from *H1*, it is to be sent to router *C* and given connection identifier 1. Similarly, the first entry at *C* routes the packet to *E*, also with connection identifier 1. Now let us consider what happens if *H3* also wants to establish a connection to *H2*. It chooses connection identifier and tells the subnet to establish the virtual circuit.

Comparison of Virtual-Circuit and Datagram Networks

Both virtual circuits and datagrams have their supporters and their detractors. We will now attempt to summarize both sets of arguments.

Issue	Datagram network	Virtual-circuit network
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed Independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC

Figure 1 Comparison of datagram and virtual-circuit networks.

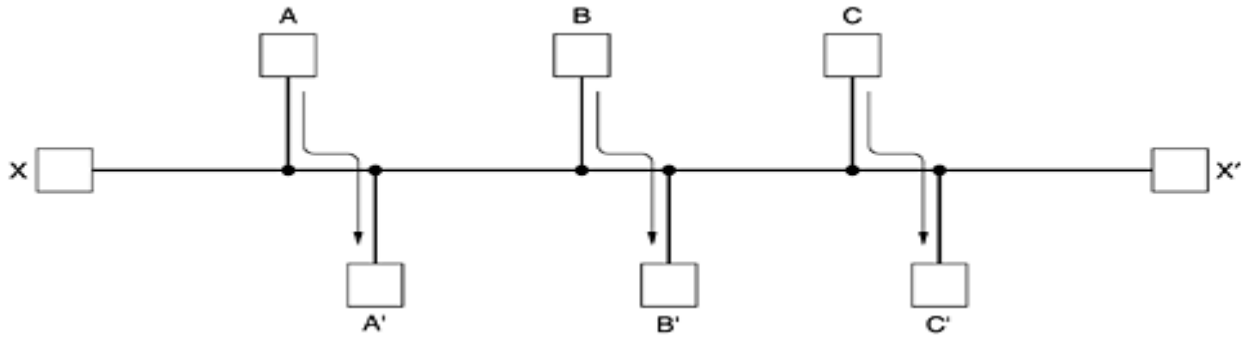
Inside the network, several trade-offs exist between virtual circuits and datagrams. One trade-off is setup time versus address parsing time. Using virtual circuits requires a setup phase, which takes time and

consumes resources. Using virtual circuits requires a setup phase, which takes time and consumes resources.

ROUTING ALGORITHMS

The main function of the network layer is routing packets from the source machine to the destination machine.

The **routing algorithm** is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagrams internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. If the subnet uses virtual circuits internally, routing decisions are made only when a new virtual circuit is being set up. Thereafter, data packets just follow the previously-established route. The latter case is sometimes called **session routing**



Routing algorithms can be grouped into two major classes:

- nonadaptive
- adaptive.

Nonadaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from *I* to *J* (for all *I* and *J*) is computed in advance, off-line, and downloaded to the routers when the network is booted. This procedure is sometimes called static routing.

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get their information (e.g., locally, from adjacent routers, or from all routers), when they change the routes

The Optimality Principle

Before we get into specific algorithms, it may be helpful to note that one can make a general statement about optimal routes without regard to network topology or traffic. This statement is known as the **optimalty principle**.

If we allow all of the possible paths to be chosen, the tree be-comes a more general structure called a **DAG** (**Directed Acyclic Graph**). DAGs have no loops. We will use sink trees as a convenient shorthand for both cases. Both cases also depend on the technical assumption that the paths do not interfere with each other so, for example, a traffic jam on one path will not cause another path to divert.

Links and routers can go down and come back up during operation, so different routers may have different ideas about the current topology. Also, we have quietly finessed the issue of whether each router has to individually acquire the information on which to base its sink tree computation or whether this information is collected by some other means.

Shortest Path Routing

The concept of a **shortest path** deserves some explanation. One way of measuring path length is the number of hops. The idea is to build a graph of the network, with each node of the graph representing a router and each edge of the graph representing a communication line, or link. To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.

Having examined each of the nodes adjacent to *A*, we examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent. This one becomes the new working node.

Several algorithms for computing the shortest path between two nodes of a graph are known. This one is due to Dijkstra (1959) and finds the shortest paths between a source and all destinations in the network. Each node is labeled (in parentheses) with its distance from the source node along the best known path. The final path is copied into the output variable, *path*, the path is thus reversed. The two reversal effects cancel, and the answer is produced in the correct order.

Flooding

A simple local technique is **flooding**, in which every incoming packet is sent out on every outgoing line except the one it arrived on.

Flooding obviously generates vast numbers of duplicate packets, in fact, an infinite number unless some measures are taken to damp the process. One such measure is to have a hop counter contained in the header of each packet that is decremented at each hop, with the packet being discarded when the counter reaches zero. Ideally, the hop counter should be initialized to the length of the path from source to destination. If the sender does not know how long the path is, it can initialize the counter to the worst case, namely, the full diameter of the net-work.

A better technique for damming the flood is to have routers keep track of which packets have been flooded, to avoid sending them out a second time. One way to achieve this goal is to have the source router put a sequence number in each packet it receives from its hosts. Each router then needs a list per source router telling which sequence numbers originating at that source have already been seen. If an incoming packet is on the list, it is not flooded. Flooding can also be used as a metric against which other routing algorithms can be compared. Flooding always chooses the shortest path because it chooses every possible path in parallel.

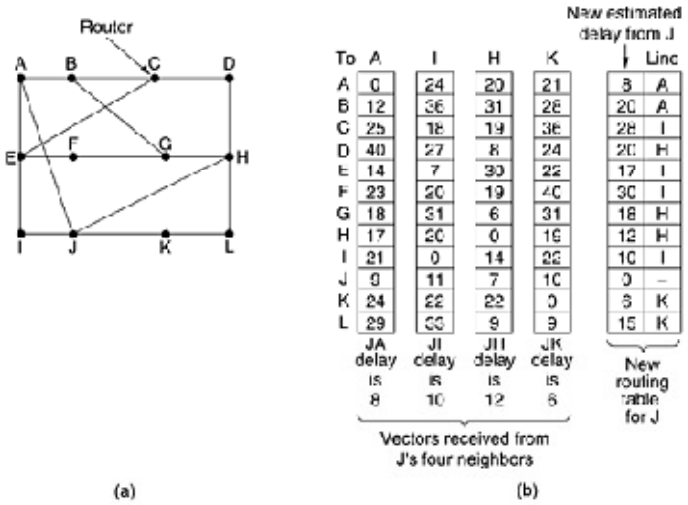
Distance Vector Routing

Two dynamic algorithm are

1 .**Distance vector routing**

2. Link State Routing

Distance vector routing algorithms operate by having each router maintain a table (i.e, a vector) giving the best nown distance to each destination and which line to use to get there. These tables are updated by exchanging information with the neighbors. The distance vector routing algorithm is sometimes called by other names, most commonly the distributed **Bellman-Ford** routing algorithm and the **Ford-Fulkerson** algorithm,



Link State Routing

The idea behind link state routing is simple and can be stated as five parts. Each router must do the following:

- ☞ Discover its neighbors and learn their network addresses.
- ☞ Measure the delay or cost to each of its neighbors.
- ☞ Construct a packet telling all it has just learned.
- ☞ Send this packet to all other routers.
- ☞ Compute the shortest path to every other router.

Learning about the Neighbors

When a router is booted, its first task is to learn who its neighbors are. It accomplishes this goal by sending a special HELLO packet on each point-to-point line. The router on the other end is expected to send back a reply giving its name. These names must be globally unique because when a distant router later hears that three routers are all connected to *F*, it is essential that it can determine whether all three mean the same *F*.

Setting Link Costs

The link state routing algorithm requires each link to have a distance or cost metric for finding shortest paths. The cost to reach neighbors can be set automatically, or configured by the network operator. A common choice is to make the cost inversely proportional to the bandwidth of the link.

If the network is geographically spread out, the delay of the links may be factored into the cost so that paths over shorter links are better choices. The most direct way to determine this delay is to send over the line a special ECHO packet that the other side is required to send back immediately.

Building Link State Packets

The information needed for the exchange has been collected, the next step is for each router to build a packet containing all the data. The packet starts with the identity of the sender, followed by a sequence number and age (to be de-scribed later) and a list of neighbors.

Distributing the Link State Packets

The fundamental idea is to use flooding to distribute the link state packets to all routers. To keep the flood in check, each packet contains a sequence number that is incremented for each new packet sent.

This algorithm has a few problems, but they are manageable. First, if the sequence numbers wrap around, confusion will reign. The solution here is to use a 32-bit sequence number. With one link state packet per second, it would take 137 years to wrap around, so this possibility can be ignored. The records where the packet originated, its sequence number and age, and the data. In addition, there are send and acknowledgement flags for each of *B*'s three links The send flags mean that the packet must be sent on the indicated link. The acknowledgement flags mean that it must be acknowledged there.

Computing the New Routes

Every link is, in fact, represented twice, once for each direction. The different directions may even have different costs. The shortest-path computations may then find different paths from router *A* to *B* than from router *B* to *A*. Compared to distance vector routing, link state routing requires more memory and computation. For a network with *n* routers, each of which has *k* neighbors, the memory required to store the input data is proportional to *kn*, which is at least as large as a routing table listing all the destinations.

The state routing is widely used in actual networks, so a few words about some example protocols are in order. Many ISPs use the IS-IS (Intermediate System-Intermediate System) link state protocol. It was designed for an early network called DECnet, later adopted by ISO for use with the OSI protocols and then modified to handle other protocols as well, most notably, IP. OSPF (Open Shortest Path First) is the other main link state protocol.

Hierarchical Routing

The hierarchical routing is used, the routers are divided into what we will call **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.

Broadcast Routing

- Sending a packet to all destinations simultaneously is called **broadcasting**; various methods have been proposed for doing it. One broadcasting method that requires no special features from the subnet is for the source to simply send a distinct packet to each destination.
- An improvement is multi destination routing, in which each packet contains either a list of destinations or a bit map indicating the desired destinations.
- The network bandwidth is therefore used more efficiently. However, this scheme still requires the source to know all the destinations, plus it is as much work for a router to determine where to send one multi destination packet as it is for multiple distinct packets. last broadcast algorithm improves on the behavior of reverse path for-warding. It makes explicit use of the sink or any other convenient spanning tree—for the router initiating the broadcast.
- A spanning tree is a subset of the network that includes all the routers but contains no loops. Sink trees are spanning trees. 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. The only problem is that each router must have knowledge of some spanning tree for the method to be applicable.

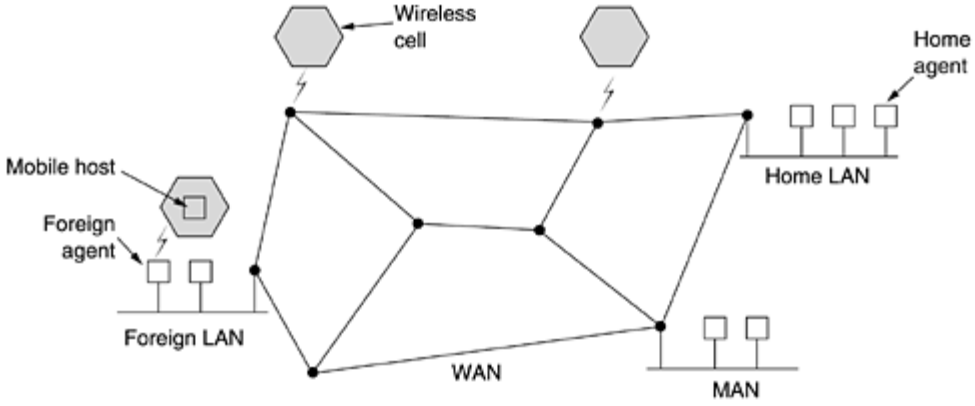
Multicast Routing

Sending a message to such a group is called **multicasting**, and its routing algorithm is called **multicast routing**.

It was one way of doing multicast routing. Multicast routing schemes build on the broadcast routing schemes we have al-ready studied, sending packets along spanning trees to deliver the packets to the members of the group while making efficient use of bandwidth. The simplest one can be used if link state routing is used and each router is aware of the complete topology, including which hosts belong to which groups. Each router can then construct its own pruned spanning tree for each sender to the group in question by constructing a sink tree for the sender as usual and then removing all links that do not connect group members to the sink node. MOSPF (Multicast OSPF) is an example of a link state protocol that works in this way

Routing for Mobile Hosts

Millions of people have portable computers nowadays, and they generally want to read their e-mail and access their normal file systems wherever in the world they may be. These mobile hosts introduce a new complication: to route a packet to a mobile host, the network first has to find it. The subject of incorporating mobile hosts into a network is very young, but in this section we will sketch some of the issues and give a possible solution



the world is divided up (geographically) into small units. Let us call them areas, where an area is typically a LAN or wireless cell. Each area has one or more **foreign agents**, which are processes that keep track of all mobile hosts visiting the area. In addition, each area has a **home agent**, which keeps track of hosts whose home is in the area, but who are currently visiting another area.

Anycast Routing

The source sends to a single destination (called **unicast**), to all destinations (called broadcast), and to a group of destinations (called multicast). Another delivery model, called **anycast** is sometimes also useful. In anycast, a packet is delivered to the nearest member of a group. Schemes that find these paths are called **anycast routing**. we will not have to devise new routing schemes for anycast because regular distance vector and link state routing can produce anycast routes.

Routing in Ad Hoc Networks

- Networks of nodes that just happen to be near each other are called **ad hoc networks** or **MANETs** (**Mobile Ad hoc NETworks**). The hosts are mobile but the routers are fixed. An even more extreme case is one in which the routers themselves are mobile.
- Many, many routing algorithms for ad hoc networks have been proposed. ad hoc networks have been little used in practice compared to mobile networks, it is unclear which of these protocols are most useful. As an ex-ample, we will look at one of the most popular routing algorithms, **AODV** (**Ad hoc On-demand Distance Vector**) (Perkins and Royer, 1999). It is a relative of the distance vector algorithm that has been adapted to work in a mobile environment, in which nodes often have limited bandwidth and battery lifetimes.
    1. **Route Discovery**
    2. **Route Maintenance**

Route Discovery

- In AODV, routes to a destination are discovered on demand, that is, only when a somebody wants to send a packet to that destination. This saves much work that would otherwise be wasted when the topology changes before the route is used. At any instant, the topology of an ad hoc network can be described by a graph of connected nodes. Two nodes are connected (i.e., have an arc between them in the graph) if they can communicate directly using their radios.
- In a large network, the algorithm generates many broadcasts, even for destinations that are close by. To reduce overhead, the scope of the broadcasts is limited using the IP packet's *Time to live* field. This field is initialized by the sender and decremented on each hop.

Route Maintenance

The algorithm needs to be able to deal with periodically; each node broadcasts a *Hello* message. Each of its neighbors is expected to respond to it. If no response is forthcoming, the broadcaster knows that neighbor has moved out of range or failed and is no longer connected to it. This information is used to purge routes that no longer work. For each possible destination, each node, *N*, keeps track of its active neighbors that have fed it a packet for that destination during the last *T* seconds. When any of *N*'s neighbors becomes unreachable, it checks its routing table to see which destinations have routes using the now-gone neighbor.

There are many other ad hoc routing schemes. DSR (Dynamic Source Routing) A different strategy based on geography is explored by GPSR (Greedy Perimeter State-less Routing). If all nodes know their geographic positions, forwarding to a destination can proceed without route computation by simply heading in the right direction and circling back to escape any dead ends. Which protocols win out will depend on the kinds of ad hoc networks that prove useful in practice.

CONGESTION CONTROL ALGORITHMS

When too many packets are present in (a part of) the subnet, performance degrades. This situation is called congestion.

The network and transport layers share the responsibility for handling congestion. Since congestion occurs within the network, it is the network layer that directly experiences it and must ultimately determine what to do with the excess packets.

There are different methods of congestion control,

- Approaches to Congestion Control
- Traffic-Aware Routing
- Admission Control
- Traffic Throttling
- Load Shedding

Approaches to Congestion Control

The presence of congestion means that the load is greater than the resources (in a part of the network) can handle. Two solutions come to mind: increase the resources or decrease the load. The most basic way to avoid congestion is to build a network that is well matched to the traffic that it carries. If there is a low-bandwidth link on the path along which most traffic is directed, congestion is likely.

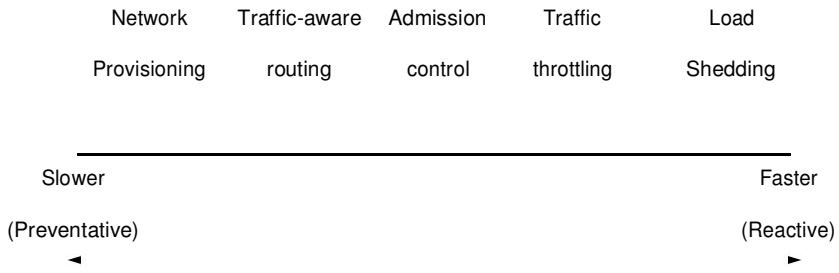


Figure 1 Timescales of approaches

More often, links and routers that are regularly heavily utilized are upgraded at the earliest opportunity. This is called **provisioning** and happens on a time scale of months, driven by long-term traffic trends. To make the most of the existing network capacity, routes can be tailored to traffic patterns that change during the day as network users wake and sleep in different time zones. Some local radio stations have helicopters flying around their cities to report on road congestion to make it possible for their mobile listeners to route their packets (cars) around hotspots. This is called **traffic-aware routing.**

Traffic-Aware Routing

Traffic-aware routing was used in the early Internet according to this model. Including queuing delay in the weight used for the shortest path calculation will make *EI* more attractive. After the new routing tables have been installed, most of the East-West traffic will now go over *EI*, loading this link. Attempts to include load but change weights within a narrow range only slow down routing oscillations. Two techniques can contribute to a successful solution. The first is multipath routing, in which there can be multiple paths from a source to a destination.

The Internet routing protocols do not generally adjust their routes depending on the load. Instead, adjustments are made outside the routing protocol by slowly changing its inputs. This is called traffic engineering.

Admission Control

It was used in virtual-circuit networks to keep congestion at bay is **admission control**. do not set up a new virtual circuit unless the network can carry the added traffic without becoming congest-ed. Thus, attempts to set up a virtual circuit may fail. This is better than the alter-native, as letting more people in when the network is busy just makes matters worse. By analogy, in the telephone system, when a switch gets overloaded it practices admission control by not giving dial tones.

Admission control can also be combined with traffic-aware routing by considering routes around traffic hotspots as part of the setup procedure.

Traffic is often described in terms of its rate and shape. The problem of how to describe it in a simple yet meaningful way is difficult because traffic is typically bare the average rate is only.

Traffic Throttling

In the Internet and many other computer networks, senders adjust their trans-missions to send as much traffic as the network can readily deliver. In this setting, the network aims to operate just before the onset of congestion. . The term **congestion avoidance** is sometimes used to contrast this operating point with the one in which the network.

The queueing delay inside routers directly captures any congestion experienced by packets. It should be low most of time, but will jump when there is a burst of traffic that generates a backlog. To maintain a good estimate of the queueing delay, d, a sample of the instantaneous queue length, s, can be made periodically and d updated according to

$$d_{new} = \alpha d_{old} + (1 - \alpha)s$$

where the constant α determines how fast the router forgets recent history. This is called an EWMA (Exponentially Weighted Moving Average).

There are three methods of Traffic Throttling,

- Choke Packets
- Explicit Congestion Notification
- Hop-by-Hop Backpressure

Choke Packets

The most direct way to notify a sender of congestion is to tell it directly. In this approach, the router selects a congested packet and sends a **choke packet** back to the source host, giving it the destination found in the packet.

The original packet may be tagged (a header bit is turned on) so that it will not generate any more choke packets farther along the path and then forwarded in the usual way. To avoid increasing load on the network during a time of congestion, the router may only send choke packets at a low rate.

Explicit Congestion Notification

The network delivers the packet, the destination can note that there is congestion and inform the sender when it sends a reply packet. Instead of generating additional packets to warn of congestion, a router can tag any packet it forwards (by setting a bit in the packet's header) to signal that it is experiencing congestion.

Hop-by-Hop Backpressure

It was the high speeds or over long distances, many new packets may be transmitted after congestion has been signaled because of the delay before the signal takes effect.

Load Shedding

Load shedding is a fancy way of saying that when routers are being inundated by packets that they cannot handle, they just throw them away. The term comes from the world of electrical power generation, where it refers to the practice of utilities intentionally blacking out certain areas to save the entire grid from collapsing on hot summer days when the demand for electricity greatly exceeds the supply.

The preferred choice may depend on the type of applications that use the network. The packets are more important than regular data packets because they establish routes; if they are lost, the net-work may lose connectivity.

Random Early Detection

A popular algorithm for doing this is called **RED** (**Random Early Detection**) (Floyd and Jacobson, 1993). The motivation for this idea is that most Internet hosts do not yet get conges-tion signals from routers in the form of ECN.

QUALITY OF SERVICE

Quality of service (QoS) refers to a network's ability to achieve maximum bandwidth and deal with other network performance elements like latency, error rate and uptime. Quality of service also involves controlling and managing network resources by setting priorities for specific types of data (video, audio, files) on the network.

Quality of service mechanisms let a network with less capacity meet application requirements just as well at a lower cost. Moreover, over provisioning is based on expected traffic. All bets are off if the traffic pattern changes too much. With quality of service mechanisms, the net-work can honor the performance guarantees that it makes even when traffic spikes, at the cost of turning down some requests.

Four issues must be addressed to ensure quality of service:

1. What applications need from the network.
2. How to regulate the traffic that enters the network.
3. How to reserve resources at routers to guarantee performance.
4. Whether the network can safely accept more traffic.

Application Requirements

A stream of packets from a source to a destination is called a **flow**. A flow might be all the packets of a connection in a connection-oriented network, or all the packets sent from one process to another process in a connectionless network.

The network requirements are less demanding than application requirements in those cases that the application can improve on the service provided by the network. In particular, networks do not need to be lossless for reliable file transfer, and they do not need to deliver packets with identical delays for audio and video play out. Some amount of loss can be repaired with retransmissions, and some amount of jitter can be smoothed by buffering packets at the receiver.

Several common applications and the stringency of their network requirements are listed in Fig. 1.

Application	Bandwidth	Delay	Jitter	Loss
Email	Low	Low	Low	Medium
File sharing	High	Low	Low	Medium
Web access	Medium	Medium	Low	Medium
Remote login	Low	Medium	Medium	Medium
Audio on demand	Low	Low	High	Low
Video on demand	High	Low	High	Low
Telephony	Low	High	High	Low
Videoconferencing	High	High	High	Low

Figure 1. Stringency of applications' quality-of-service requirements.

The applications differ in their bandwidth needs, with email, audio in all forms, and remote login not needing much, but file sharing and video in all forms needing a great deal. More interesting are the delay requirements. File transfer applications, including email and video, are not delay sensitive.

Traffic Shaping

Traffic shaping is a technique for regulating the average rate and business of a flow of data that enters the network. The goal is to allow applications to transmit a wide variety of traffic that suits their needs, including some bursts, yet have a simple and useful way to describe the possible traffic patterns to the network.

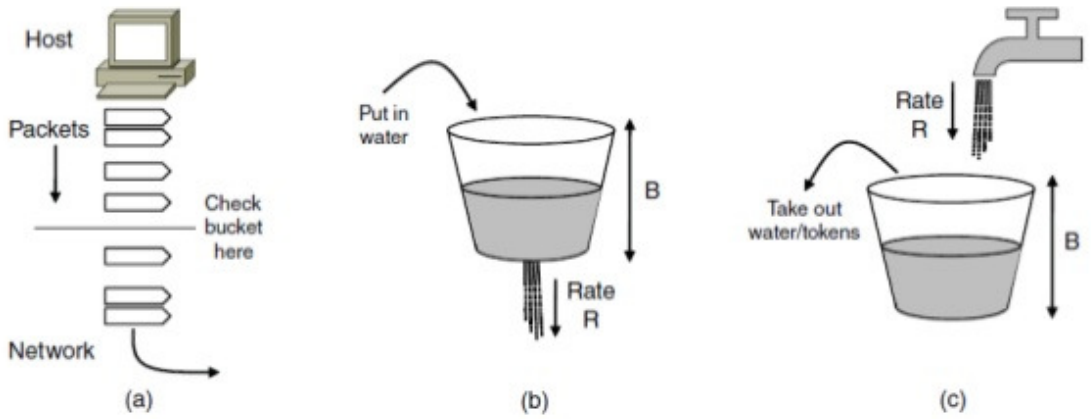
Traffic shaping reduces congestion and thus helps the network live up to its promise. However, to make it work, there is also the issue of how the provider can tell if the customer is following the

agreement and what to do if the customer is not. Packets in excess of the agreed pattern might be dropped by the network, or they might be marked as having lower priority. Monitoring a traffic flow is called **traffic policing**.

Leaky and Token Buckets

The leaky bucket is an algorithm based on an analogy of how a bucket with a leak will overflow if either the average rate at which water is poured in exceeds the rate at which the bucket leaks or if more water than the capacity of the bucket is poured in all at once, and how the water leaks from the bucket at an (almost) constant rate.

Traffic Shaping (1)



(a) Shaping packets. (b) A leaky bucket. (c) A token bucket

Try to imagine a bucket with a small hole in the bottom. No matter the rate at which water enters the bucket, the outflow is at a constant rate, *R*, when there is any water in the bucket and zero when the bucket is empty. Also, once the bucket is full to capacity *B*, any additional water enter-ing it spills over the sides and is lost.

This bucket can be used to shape or police packets entering the network. , each host is connected to the network by an interface containing a leaky bucket. To send a packet into the network, it must be possible to put more water into the bucket. If a packet arrives when the bucket is full, the packet must either be queued until enough water leaks out to hold it or be discarded.

The former might happen at a host shaping its traffic for the network as part of the operating system. The latter might happen in hardware at a provider network interface that is policing traffic entering the network. This technique was proposed by Turner (1986) and is called the **leaky bucket algorithm**.

Leaky and token buckets limit the long-term rate of a flow but allow short-term bursts up to a maximum regulated length to pass through unaltered and without suffering any artificial delays. Leaky and token buckets are easy to implement. We will now describe the operation of a token bucket. Even though we have described water flowing continuously into and out of the bucket, real implementations must work with discrete quantities. A token bucket is implemented with a counter for the level of the bucket.

Packet Scheduling

Algorithms that allocate router resources among the packets of a flow and between competing flows are called **packet scheduling algorithms**. Three different kinds of resources can potentially be reserved for different flows:

1. Bandwidth.
2. Buffer space.
3. CPU cycles.

- The first one, bandwidth, is the most obvious. If a flow requires 1 Mbps and the outgoing line has a capacity of 2 Mbps, trying to direct three flows through that line is not going to work. Thus, reserving bandwidth means not oversubscribing any output line.
- The purpose of the buffer is to absorb small bursts of traffic as the flows contend with each other. If no buffer is available, the packet has to be discarded since there is no place to put it. For good quality of service, some buffers might be reserved for a specific flow so that flow does not have to compete for buffers with other flows. Up to some maximum value, there will always be a buffer available when the flow needs one.
- Packet scheduling algorithms allocate bandwidth and other router resources by determining which of the buffered packets to send on the output line next. We already described the most straightforward scheduler when explaining how routers work.
- Each router buffers packets in a queue for each output line until they can be sent, and they are sent in the same order that they arrived. This algorithm is known as **FIFO** (**First-In First-Out**), or equivalently **FCFS** (**First-Come First-Serve**).
- FIFO routers usually drop newly arriving packets when the queue is full. Since the newly arrived packet would have been placed at the end of the queue, this behavior is called **tail drop.**
- Many packet scheduling algorithms have been devised that provide stronger isolation between flows and thwart attempts at interference (Bhatti and Crowcroft, 2000). One of the first ones was the **fair queueing** algorithm devised by Nagle (1987). The essence of this algorithm is that routers have separate queues, one for each flow for a output line.

Admission Control

We first saw admission control used to control congestion, which is a performance guarantee, albeit a weak one. The guarantees we are considering now are stronger, but the model is the same. The user offers a flow with an accompanying QoS requirement to the network.

The network then decides whether to accept or reject the flow based on its capacity and the commitments it has made to other flows. If it accepts, the network reserves capacity in advance at routers to guarantee QoS when traffic is sent on the new flow.

The reservations must be made at all of the routers along the route that the packets take through the network. Any routers on the path without reservations might become congested, and a single congested router can break the QoS guarantee. Many routing algorithms find the single best path between each source and each destination and send all traffic over the best path.

Integrated Services

The generic name for this work is **integrated services**. It was aimed at both unit cast and multicast applications. An example of the former is a single user streaming a video clip from a news site.

In many multicast applications, groups can change membership dynamically, for example, as people enter a video conference and then get bored and switch to a soap opera or the croquet channel.

RSVP—The Resource reservation Protocol

The main part of the integrated services architecture that is visible to the users of the network is **RSVP**. This protocol is used for making the reservations; other protocols are used for sending the data.

In its simplest form, the protocol uses multicast routing using spanning trees, as discussed earlier. Each group is assigned a group address. To send to a group, a sender puts the group's address in its packets. The standard multicast routing algorithm then builds a spanning tree covering all group members.

The routing algorithm is not part of RSVP. The only difference from normal multicasting is a little extra information that is multicast to the group periodically to tell the routers along the tree to maintain certain data structures in their memories.

Differentiated Services

Differentiated services can be offered by a set of routers forming an administrative domain (e.g., an ISP or a telco). The administration defines a set of service classes with corresponding forwarding rules. If a customer subscribes to differentiated services, customer packets entering the domain are marked with the class to which they belong. This information is carried in the *Differentiated ser-vices* field of IPv4 and IPv6 packets .

The Flow-based algorithms have the potential to offer good quality of service to one or more flows because they reserve whatever resources are needed along the route.

Two method of services,

- **Expedited Forwarding**
- **Assured Forwarding**

Expedited Forwarding

The choice of service classes is up to each operator, but since packets are often forwarded between networks run by different operators, IETF has defined some network-independent service classes. The simplest class is **expedited for-warding**.

The idea behind expedited forwarding is very simple. Two classes of service are available: regular and expedited. The vast majority of the traffic is expected to be regular, but a limited fraction of the packets are expedited. The expedited packets should be able to transit the network as though no other packets were present. In this way they will get low loss, low delay and low jitter service—just what is needed for VoIP.

If the packets pass through a corporate network or ISP that sup-ports expedited service, they will receive preferential treatment. If the network does not support expedited service, no harm is done.

Assured Forwarding

A elaborate scheme for managing the service classes is called **assured forwarding**. It is described in RFC 2597. Assured forwarding specifies that there shall be four priority classes, each class having its own resources.

The first step is to classify the packets into one of the four priority classes. As before, this step might be done on the sending host (as shown in the figure) or in the ingress router, and the rate of higher-priority packets may be limited by the operator as part of the service offering.

The next step is to determine the discard class for each packet. This is done by passing the packets of each priority class through a traffic policer such as a token bucket. The weights double from one class to the next higher class, the higher class will get twice the bandwidth. Within a priority class, packets with a higher discard class can be preferentially dropped by running an algorithm such as RED (Random Early Detection),

INTERNETWORKING

Two or more networks are connected to form an **internetwork**, or more simply an **internet**.

It would be much simpler to join networks together if everyone used a single networking technology, and it is often the case that there is a dominant kind of network, such as Ethernet. If there will always be different networks, it would be simpler if we did not need to interconnect them.

The purpose of joining all these networks is to allow users on any of them to communicate with users on all the other ones. When you pay an ISP for Internet service, you may be charged depending on the bandwidth of your line, but what you are really paying for is the ability to exchange packets with any other host that is also connected to the Internet.

Networks Differ

Networks can differ in many ways. Some of the differences, such as different modulation techniques or frame formats, are internal to the physical and data link layers. It is papering over these differences that makes internetworking more difficult than operating within a single network.

Many specific differences may have to be accommodated as well. How do we multicast a packet to a group with some members on a network that does not support multicast.

Item	Some Possibilities
Service offered	Connectionless versus connection oriented

Addressing	Different sizes, flat or hierarchical
Broadcasting	Present or absent (also multicast)
Packet size	Every network has its own maximum
Ordering	Ordered and unordered delivery
Quality of service	Present or absent; many different kinds
Reliability	Different levels of loss
Security	Privacy rules, encryption, etc.
Parameters	Different timeouts, flow specifications, etc.
Accounting	By connect time, packet, byte, or not at all

Figure 1. Some of the many ways networks can differ.

These kinds of differences can be papered over, with some effort. For ex-ample, a gateway joining two networks might generate separate packets for each destination in lieu of better network support for multicasting. A large packet might be broken up, sent in pieces, and then joined back together. Receivers might buffer packets and deliver them in order. Networks also can differ in large respects that are more difficult to reconcile.

Networks Can Be Connected

This approach has been tremendously successful, and the layer they proposed was eventually separated into the TCP and IP proto-cols. Almost four decades later, IP is the foundation of the modern Internet.

IP provides a universal packet format that all routers recognize and that can be passed through almost every network. IP has extended its reach from computer networks to take over the telephone network. It also runs on sensor networks and other tiny devices that were once presumed too resource-constrained to support it.

The source accepts data from the transport layer and generates a packet with the common net-work. The router now examines the IP address in the packet and looks up this address in its routing table. Based on this address, it decides to send the packet to the second router next. . Arguably, even IP has only worked so well by serving as a kind of lowest common denominator.

Tunneling

Multi-protocol routers have to understand both IPv4 and IPv6 packets. In effect, the en-tire trip from one multiprotocol router to the other is like a hop over a single link. The solution to this problem is a technique called **tunneling**.

Tunneling is widely used to connect isolated hosts and networks using other networks. The network that results is called an **overlay** since it has effectively been overlaid on the base network.

Deployment of a network protocol with a new feature is a common reason, as our ‘‘IPv6 over IPv4’’

Internetwork Routing

Networks run by different operators lead to bigger problems. First, the operators may have different ideas about what is a good path through the network. One operator may want the route with the least delay, while another may want the most inexpensive route.

The internet may be much larger than any of the networks that comprise it. It may therefore require routing algorithms that scale well by using a hierarchy, even if none of the individual networks need to use a hierarchy.

The two levels are usually not strictly hierarchical, as highly suboptimal paths might result if a large international network and a small regional network were both abstracted to be a single network.

Packet Fragmentation

Each network or link imposes some maximum size on its packets. These limits have various causes, among them

1. Hardware (e.g., the size of an Ethernet frame).
2. Operating system (e.g., all buffers are 512 bytes).
3. Protocols (e.g., the number of bits in the packet length field).
4. Compliance with some (inter)national standard.
5. Desire to reduce error-induced retransmissions to some level.
6. Desire to prevent one packet from occupying the channel too long.

Hosts usually prefer to transmit large packets because this reduces packet overheads such as bandwidth wasted on header bytes. An obvious internetworking problem appears when a large packet wants to travel through a network whose maximum packet size is too small.

This packet size is called the **Path MTU** (**Path Maximum Transmission Unit**). Even if the source did know the path MTU, packets are routed independently in a connectionless network such as the Internet. This routing means that paths may suddenly change, which can unexpectedly change the path MTU.

The alternative solution to the problem is to allow routers to break up packets into **fragments**, sending each fragment as a separate network layer packet. The overhead can be higher than with transparent fragmentation because fragment headers are now carried over some links where they may not be needed. But the real problem is the existence of fragments in the first place.

TCP and IP are typically implemented together (as ‘‘TCP/IP’’) to be able to pass this sort of information. Even if this is not done for other protocols, fragmentation has still been moved out of the network and into the hosts.

----------UNIT III COMPLETED----------

UNIT IV

THE TRANSPORT LAYER

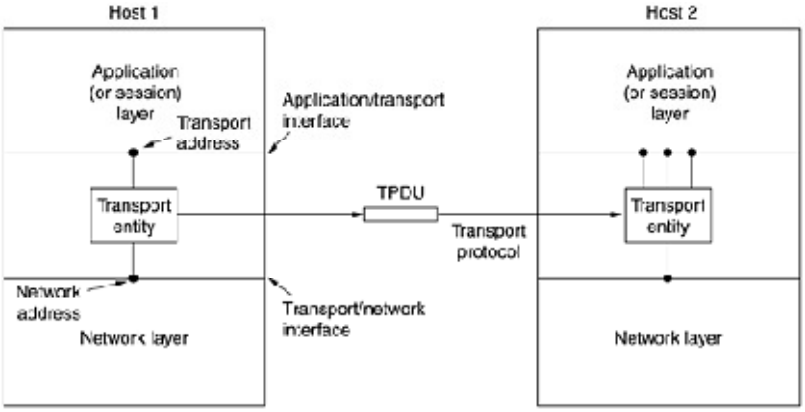
The transport layer is the heart of the protocol hierarchy. The transport layer builds on the network layer to provide data transport from a process on a source machine to a process on a destination machine with a desired level of reliability that is independent of the physical networks currently in use. It provides the abstractions that applications need to use the network.

THE TRANSPORT SERVICE

Services Provided to the Upper Layers

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

The transport entity can be located in the operating system kernel, in a library package bound into network applications, in a separate user process, or even on the network interface card.



There are two types of network service

- Connection-oriented
- Connectionless

Similarly, there are also two types of transport service. The connection-oriented transport service is similar to the connection-oriented network service in many ways.

In both cases, connections have three phases:

- Establishment
- Data transfer
- Release.

- Addressing and flow control are also similar in both layers. Furthermore, the connectionless transport service is also very similar to the connectionless network service.

- The bottom four layers can be seen as the transport service provider, whereas the upper layer(s) are the transport service user.

Transport Service Primitives

- ✓ To allow users to access the transport service, the transport layer must provide some operations to application programs, that is, a transport service interface. Each transport service has its own interface.
- ✓ The transport service is similar to the network service, but there are also some important differences.
- ✓ The main difference is that the network service is intended to model the service offered by real networks. Real networks can lose packets, so the network service is generally unreliable.
- ✓ The (connection-oriented) transport service, in contrast, is reliable.

As an example, consider two processes connected by pipes in UNIX. They assume the connection between them is perfect. They do not want to know about acknowledgements, lost packets, congestion, or anything like that. What they want is a 100 percent reliable connection. Process A puts data into one end of the pipe, and process B takes it out of the other.

A second difference between the network service and transport service is whom the services are intended for. The network service is used only by the transport entities. Consequently, the transport service must be convenient and easy to use.

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

Table:.1 - The primitives for a simple transport service.

Eg: Consider an application with a server and a number of remote clients.

1. The server executes a “LISTEN” primitive by calling a library procedure that makes a System call to block the server until a client turns up.
2. When a client wants to talk to the server, it executes a “CONNECT” primitive, with “CONNECTION REQUEST” TPDU sent to the server.
3. When it arrives, the TE unblocks the server and sends a “CONNECTION ACCEPTED” TPDU back to the client.
4. When it arrives, the client is unblocked and the connection is established. Data can now be exchanged using “SEND” and “RECEIVE” primitives.
5. When a connection is no longer needed, it must be released to free up table space within the 2 transport entries, which is done with “DISCONNECT” primitive by sending “DISCONNECTION REQUEST” TPDU. This disconnection can b done either by asymmetric variant (connection is released, depending on other one) or by symmetric variant (connection is released, independent of other one).

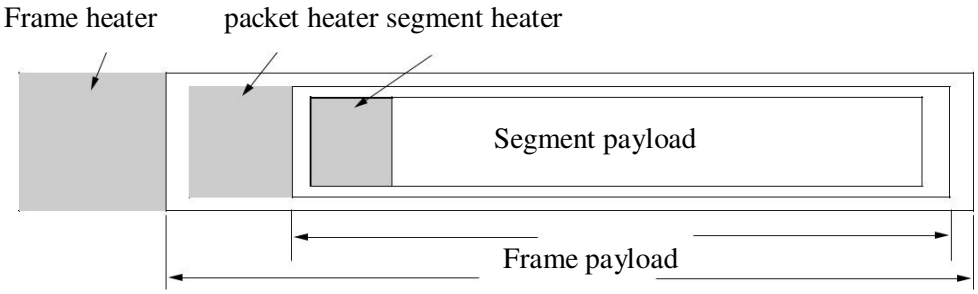


Figure 2 - Nesting of TPDUs, packets, and frames

- ✓ The term segment for messages sent from transport entity to transport entity.
- ✓ TCP, UDP and other Internet protocols use this term. Segments (exchanged by the transport layer) are contained in packets (exchanged by the network layer).
- ✓ These packets are contained in frames(exchanged by the data link layer).When a frame arrives, the data link layer processes the frame header and, if the destination address matches for local delivery, passes the contents of the frame payload field up to the network entity.
- ✓ The network entity similarly processes the packet header and then passes the contents of the packet payload up to the transport entity.

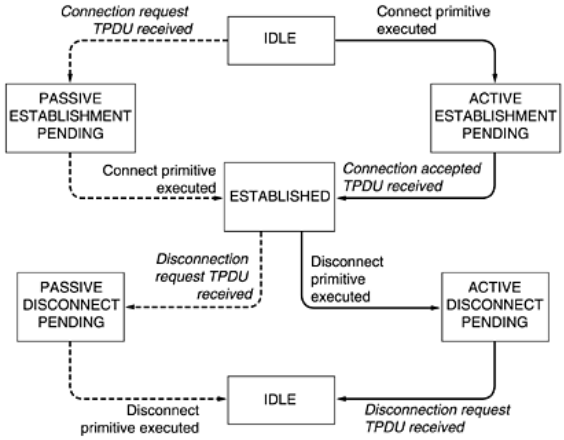


Figure 3 - A state diagram for a simple connection management scheme.

Each transition is triggered by some event, either a primitive executed by the local transport user or an incoming packet. For simplicity, we assume here that each TPDU is separately acknowledged. We also assume that a symmetric disconnection model is used, with the client going first. Please note that this model is quite unsophisticated. We will look at more realistic models later on.

Berkeley Sockets

These primitives are socket primitives used in Berkley UNIX for TCP.

The socket primitives are mainly used for TCP. These sockets were first released as part of the Berkeley UNIX 4.2BSD software distribution in 1983. They quickly became popular. The primitives are now widely used for Internet programming on many operating systems, especially UNIX -based systems, and there is a socket-style API for Windows called ‘‘winsock.’’

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Figure 4.4 - The socket primitives for TCP.

The first four primitives in the list are executed in that order by servers.

- ✓ The SOCKET primitive creates a new endpoint and allocates table space for it within the transport entity. The parameter includes the addressing format to be used, the type of service desired and the protocol. Newly created sockets do not have network addresses.
- ✓ The BIND primitive is used to connect the newly created sockets to an address. Once a server has bound an address to a socket, remote clients can connect to it.
- ✓ The LISTEN call, which allocates space to queue incoming calls for the case that several clients try to connect at the same time.
- ✓ The server executes an ACCEPT primitive to block waiting for an incoming connection.

Some of the client side primitives are. Here, too, a socket must first be created

- ✓ The CONNECT primitive blocks the caller and actively starts the connection process. When it¬ completes, the client process is unblocked and the connection is established.
- ✓ Both sides can now use SEND and RECEIVE to transmit and receive data over the full-duplex connection.
- ✓ Connection release with sockets is symmetric. When both sides have executed a CLOSE primitive, the¬ connection is released.

ELEMENTS OF TRANSPORT PROTOCOLS

The transport service is implemented by a transport protocol used between the two transport entities. The transport protocols resemble the data link protocols. Both have to deal with error control, sequencing, and flow control, among other issues. The difference transport protocol and data link protocol depends upon the environment in which they are operated.

These differences are due to major dissimilarities between the environments in which the two protocols operate, as shown in Fig.

At the data link layer, two routers communicate directly via a physical channel, whether wired or wireless, whereas at the transport layer, this physical channel is replaced by the entire network. This difference has many important implications for the protocols.

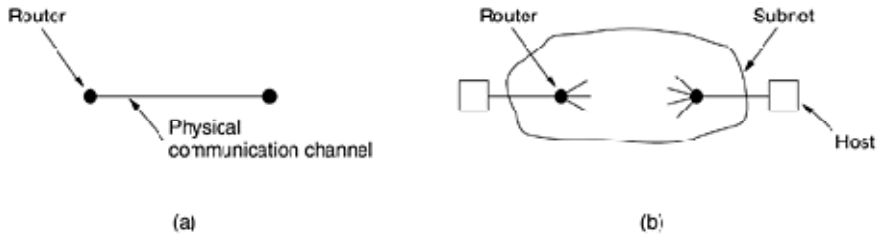


Figure (a) Environment of the data link layer. (b) Environment of the transport layer.

In the data link layer, it is not necessary for a router to specify which router it wants to talk to. In the transport layer, explicit addressing of destinations is required.

In the transport layer, initial connection establishment is more complicated, as we will see. Difference between the data link layer and the transport layer is the potential existence of storage capacity in the subnet.

The elements of transport protocols are:

1. ADDRESSING
2. Connection Establishment.
3. Connection Release.
4. Error control and flow control
5. Multiplexing.

Addressing

When an application (e.g., a user) process wishes to set up a connection to a remote application process, it must specify which one to connect to. The method normally used is to define transport addresses to which processes can listen for connection requests. In the Internet, these endpoints are called ports.

There are two types of access points.

- ✓ TSAP (Transport Service Access Point) to mean a specific endpoint in the transport layer.
- ✓ The analogous endpoints in the network layer (i.e., network layer addresses) are not surprisingly called NSAPs (Network Service Access Points). IP addresses are examples of NSAP

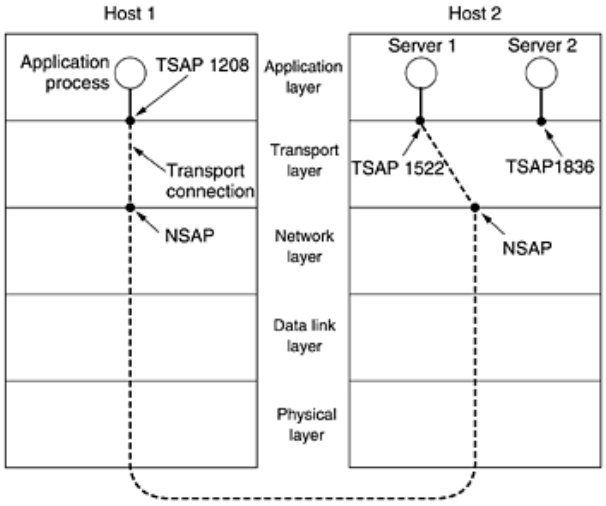


Fig : TSAP and NSAP network connections

Application processes, both clients and servers, can attach themselves to a local TSAP to establish a connection to a remote TSAP. These connections run through NSAPs on each host. The purpose of having TSAPs is that in some networks, each computer has a single NSAP, so some way is needed to distinguish multiple transport endpoints that share that NSAP.

A possible scenario for a transport connection is as follows:

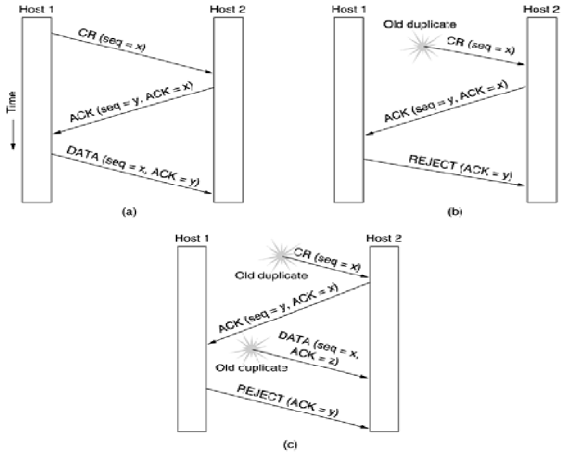
1. A mail server process attaches itself to TSAP 1522 on host 2 to wait for an incoming call. How a process attaches itself to a TSAP is outside the networking model and depends entirely on the local operating system. A call such as our LISTEN might be used, for example.
2. 2. An application process on host 1 wants to send an email message, so it attaches itself to TSAP 1208 and issues a CONNECT request. The request specifies TSAP 1208 on host 1 as the source and TSAP 1522 on host 2 as the destination. This action ultimately results in a transport connection being established between the application process and the server.
3. The application process sends over the mail message.
4. 4. The mail server responds to say that it will deliver the message.
5. 5. The transport connection is released.

Connection Establishment

1. With packet lifetimes bounded, it is possible to devise a fool proof way to establish connections safely. Packet lifetime can be bounded to a known maximum using one of the following techniques: *Normal operation.*

- ✓ Restricted subnet design
- ✓ Putting a hop counter in each packet
- ✓ Time stamping in each packet

Using a 3-way hand shake, a connection can be established. This establishment protocol doesn't require both sides to begin sending with the same sequence number.



- ➢ The first technique includes any method that prevents packets from looping, combined with some way of bounding delay including congestion over the longest possible path. It is difficult, given that internets may range from a single city to international in scope.

- The second method consists of having the hop count initialized to some appropriate value and decremented each time the packet is forwarded. The network protocol simply discards any packet whose hop counter becomes zero.
- The third method requires each packet to bear the time it was created, with the routers agreeing to¬ discard any packet older than some agreed-upon time.

In fig (A) Tomlinson (1975) introduced the three-way handshake.

This establishment protocol involves one peer checking with the other that the connection request is¬ indeed current. Host 1 chooses a sequence number, x , and sends a CONNECTION REQUEST segment containing it to host 2. Host 2replies with an ACK segment acknowledging x and announcing its own initial sequence number, y.

In fig (B) the first segment is a delayed duplicate CONNECTION REQUEST from an old connection.

This segment arrives at host 2 without host 1's knowledge. Host 2 reacts to this segment by sending¬ host1an ACK segment, in effect asking for verification that host 1 was indeed trying to set up a new connection.
When host 1 rejects host 2's attempt to establish a connection, host 2 realizes that it was tricked by a¬ delayed duplicate and abandons the connection. In this way, a delayed duplicate does no damage.

In fig (C) previous example, host 2 gets a delayed CONNECTION REQUEST and replies to it.

At this point, it is crucial to realize that host 2 has proposed using y as the initial sequence number for¬ host 2 to host 1 traffic, knowing full well that no segments containing sequence number y or acknowledgements to y are still in existence.

When the second delayed segment arrives at host 2, the fact that z has been acknowledged rather than y¬ tells host 2 that this, too, is an old duplicate.

Connection Release

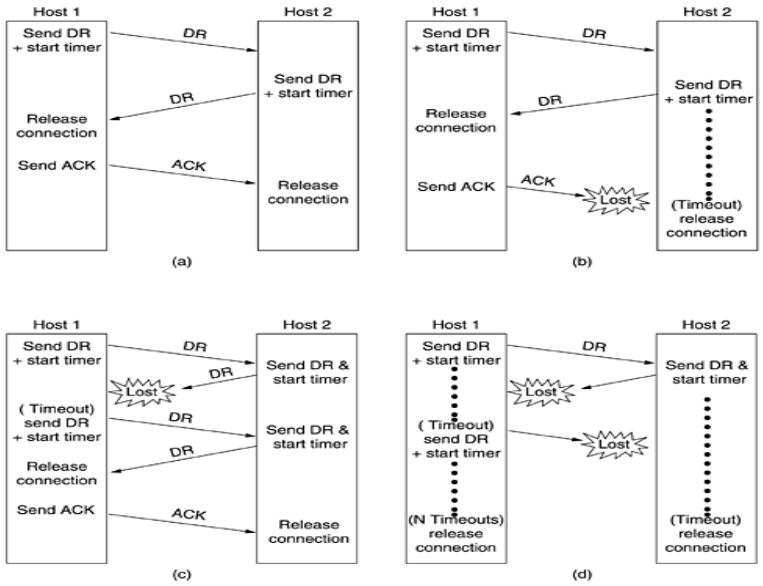
A connection is released using either asymmetric or symmetric variant. But, the improved protocol for releasing a connection is a 3-way handshake protocol.

There are two styles of terminating a connection:

1) Asymmetric release and

2) Symmetric release.

Asymmetric release is the way the telephone system works: when one party hangs up, the connection is broken. Symmetric release treats the connection as two separate unidirectional connections and requires each one to be released separately.



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

Fig 1 : One of the user sends a DISCONNECTION REQUEST TPDU in order to initiate connection release When it arrives, the recipient sends back a DR-TPDU, too, and starts a timer. When this DR arrives, the original sender sends back an ACKTPDU and releases the connection.

Fig 2 : Initial process is done in the same way as in fig-(a). If the final ACK-TPDU is lost, the situation is saved by the timer. When the timer is expired, the connection is released.

Fig 3: If the second DR is lost, the user initiating the disconnection will not receive the expected response, and will timeout and starts all over again.

Fig 4: except that all repeated attempts to retransmit the DR is assumed to be failed due to lost TPDUs. After 'N' entries, the sender just gives up and releases the connection.

FLOW CONTROL AND BUFFERING

Flow control is done by having a sliding window on each connection to keep a fast transmitter from over running a slow receiver. Buffering must be done by the sender, if the network service is unreliable. The sender buffers all the TPDUs sent to the receiver. The buffer size varies for different TPDUs.

They are:

a) Chained Fixed-size Buffers

b) Chained Variable-size Buffers

c) One large Circular Buffer per Connection

(a). Chained Fixed-size Buffers:

If most TPDUs are nearly the same size, the buffers are organized as a pool of identical size buffers, with one TPDU per buffer.

(b). Chained Variable-size Buffers:

This is an approach to the buffer-size problem. i.e., if there is wide variation in TPDU size, from a few characters typed at a terminal to thousands of characters from file transfers, some problems may occur:

- If the buffer size is chosen equal to the largest possible TPDU, space will be wasted whenever a short TPDU arrives.
- If the buffer size is chosen less than the maximum TPDU size, multiple buffers will be needed for long TPDUs.

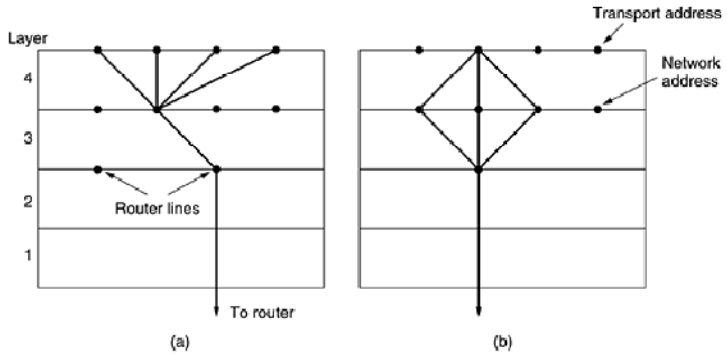
To overcome these problems, we employ variable-size buffers.

(c). One large Circular Buffer per Connection: A single large circular buffer per connection is dedicated when all connections are heavily loaded.

1. Source Buffering is used for low band width bursty traffic
2. Destination Buffering is used for high band width smooth traffic.
3. Dynamic Buffering is used if the traffic pattern changes randomly.

Multiplexing

In networks that use virtual circuits within the subnet, each open connection consumes some table space in the routers for the entire duration of the connection. If buffers are dedicated to the virtual circuit in each router as well, a user who left a terminal logged into a remote machine, there is need for multiplexing. There are 2 kinds of multiplexing,



(a) Upward multiplexing. (b) Downward multiplexing.

(a). UP-WARD MULTIPLEXING:

In the below figure, all the 4 distinct transport connections use the same network connection to the remote host. When connect time forms the major component of the carrier's bill, it is up to the transport layer to group port connections according to their destination and map each group onto the minimum number of port connections.

(b). DOWN-WARD MULTIPLEXING:

- If too many transport connections are mapped onto the one network connection, the performance will be poor.
- If too few transport connections are mapped onto one network connection, the service will be expensive.

The possible solution is to have the transport layer open multiple connections and distribute the traffic among them on round-robin basis, as indicated in the below with 'k' network connections open, the effective band width is increased by a factor of 'k'.

CONGESTION CONTROL

If the transport entities on many machines send too many packets into the net-work too quickly, the network will become congested, with performance degraded as packets are delayed and lost. lost. Controlling congestion to avoid this problem is the combined responsibility of the network and transport layers. Congestion oc-curs at routers, so it is detected at the network layer.

Desirable Bandwidth Allocation

That is, we must specify the state in which a good congestion control algorithm will operate the network. The goal is more than to simply avoid congestion. It is to find a good al-location of bandwidth to the transport entities that are using the network.

Efficiency and Power

An efficient allocation of bandwidth across transport entities will use all of the network capacity that is available. They should usually get less than 20 Mbps for good performance. The reason is that the traffic is often bursty.

Max-Min Fairness

This sounds like a simple question to answer—give all the senders an equal fraction of the bandwidth but it involves several considerations.

Perhaps the first consideration is to ask what this problem has to do with congestion control. After all, if the network gives a sender some amount of bandwidth to use, the sender should just use that much bandwidth.

IP routers often have all connections competing for the same bandwidth. In this situation, it is the congestion control mechanism that is allocating bandwidth to the competing connections.

Convergence

A final criterion is that the congestion control algorithm converge quickly to a fair and efficient allocation of bandwidth.

A good congestion control algorithm should rapidly converge to the ideal operating point, and it should track that point as it changes over time. If the convergence is too slow, the algorithm will never be close to the changing operating point. If the algorithm is not stable, it may fail to converge to the right point in some cases, or even oscillate around the right point.

Regulating the Sending Rate

The way that a transport protocol should regulate the sending rate depends on the form of the feedback returned by the network. Different network layers may return different kinds of feedback. The feedback may be explicit or implicit, and it may be precise or imprecise.

For example, if XCP tells senders the rate to use, the senders may simply use that rate. In the other cases, however, some guesswork is involved.XCP tells senders the rate to use, the senders may simply use that rate. In the other cases, however, some guesswork is involved. In the absence of a congestion signal, the senders should decrease their rates. When a congestion signal is given, the senders should decrease their rates. The way in which the rates are increased or decreased is given by a **control law**. These laws have a major effect on performance.

Protocol	Signal	Explicit?	Precise?
XCP	Rate to use	Yes	Yes
TCP with ECN	Congestion warning	Yes	No
FAST TCP	End-to-end delay	No	Yes
Compound TCP	Packet loss & end-to-end delay	No	Yes
CUBIC TCP	Packet loss	No	No
TCP	Packet loss	No	No

Figure 1. Signals of some congestion control protocols.

The case of binary congestion feedback and concluded that **AIMD** (**Additive Increase Multiplicative Decrease**) is the appropriate control law to arrive at the efficient and fair operating point. To argue this case, they constructed a graphical argument for the simple case of two connections competing for the bandwidth of a single link.

Wireless Issues

Transport protocols such as TCP that implement congestion control should be independent of the underlying network and link layer technologies. That is a good theory, but in practice there are issues with wireless networks. The main issue is that packet loss is often used as a congestion signal, including by TCP.

Wireless networks lose packets all the time due to transmission errors. the loss rate for fast TCP connections is very small; 1% is a moderate loss rate, and by the time the loss rate reaches 10% the connection has effectively stopped working. However, for wireless networks such as 802. uses a stop-

and-wait protocol to deliver each frame, retrying transmissions multiple times if need be before reporting a packet loss to the higher layer.

This strategy is feasible because congestion control algorithms must already handle the case of new users entering the network or existing users changing their sending rates. Even though the capacity of wired links is fixed, the changing behavior of other users presents itself as variability in the bandwidth that is available to a given user.

THE INTERNET TRANSPORT PROTOCOLS: UDP

The Internet has two main protocols in the transport layer, a connectionless protocol and a connection oriented one. The protocols complement each other. The connectionless protocol is UDP. It does almost nothing beyond sending packets between applications, letting applications build their own protocols on top as needed.

The connection-oriented protocol is TCP. It does almost everything. It makes connections and adds reliability with retransmissions, along with flow control and congestion control, all on behalf of the applications that use it. Since UDP is a transport layer protocol that typically runs in the operating system and protocols that use UDP typically run in user s pace, these uses might be considered applications.

INTROUCTION TO UDP

- ✓ The Internet protocol suite supports a connectionless transport protocol called UDP (User Datagram Protocol). UDP provides a way for applications to send encapsulated IP datagrams without having to establish a connection.
- ✓ UDP transmits segments consisting of an 8-byte header followed by the pay-load. The two ports serve to identify the end-points within the source and destination machines.
- ✓ When a UDP packet arrives, its payload is handed to the process attached to the destination port. This attachment occurs when the BIND primitive. Without the port fields, the transport layer would not know what to do with each incoming packet. With them, it delivers the embedded segment to the correct application.

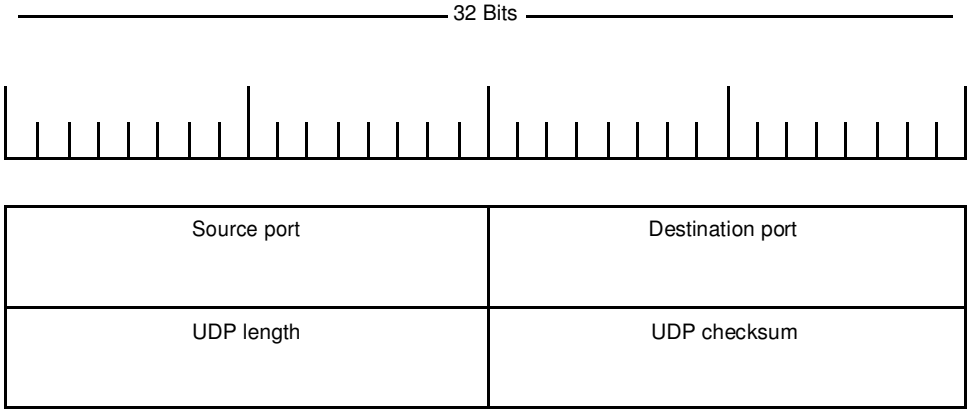


Figure 1 The UDP header.

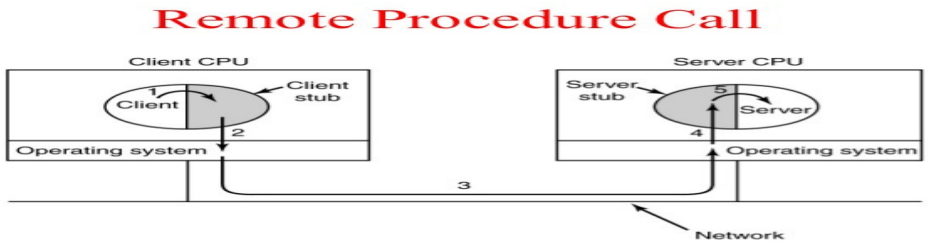
Source port, destination port: Identifies the end points within the source and destination machines.

UDP length: Includes 8-byte header and the data.

UDP checksum: Includes the UDP header, the UDP data padded out to an even number of bytes if need be. It is an optional field .

REMOTE PROCEDURE CALL

- In a certain sense, sending a message to a remote host and getting a reply back is like making a function call in a programming language. This is to arrange request-reply interactions on networks to be cast in the form of procedure calls.
- For example, just imagine a procedure named get IP address (host name) that works by sending a UDP packet to a DNS server and waiting or the reply, timing out and trying again if one is not forthcoming quickly enough. In this way, all the details of networking can be hidden from the programmer.
- RPC is used to call remote programs using the procedural call. When a process on machine 1 calls a procedure on machine 2, the calling process on 1 is suspended and execution of the called procedure takes place on 2.
- Information can be transported from the caller to the call in the parameters and can come back in the procedure result. No message passing is visible to the application programmer. This technique is known as RPC (Remote Procedure Call) and has become the basis for many networking applications.



Steps in making a remote procedure call. The stubs are shaded.

Step 1 is the client calling the client stub. This call is a local procedure call, with the parameters pushed onto the stack in the normal way.

Step 2 is the client stub packing the parameters into a message and making a system call to send the message. Packing the parameters is called marshaling.

Step 3 is the operating system sending the message from the client machine to the server machine. **Step 4** is the operating system passing the incoming packet to the server stub.

Step 5 is the server stub calling the server procedure with the un marshaled parameters. The reply traces the same path in the other direction.

The key item to note here is that the client procedure, written by the user, just makes a normal (i.e., local) procedure call to the client stub, which has the same name as the server procedure. Since the client procedure and client stub are in the same address space, the parameters are passed in the usual way.

Real-Time Transport Protocols

Client-server RPC is one area in which UDP is widely used. was **RTP** (**Real-time Transport Protocol**) born. It is described in RFC 3550 and is now in widespread use for multimedia applications. We will describe two aspects of real-time transport.

- The first is the RTP protocol for transporting audio and video data in packets.
- The second is the processing that takes place, mostly at the receiver, to play out the audio and video at the right time.

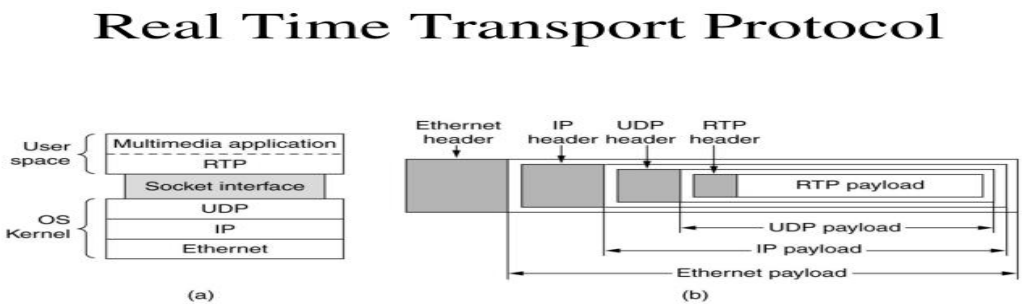


Figure 1. (a) The position of RTP in the protocol stack. (b) Packet nesting.

- The multimedia application consists of multiple audio, video, text, and possibly other streams. These are fed into the RTP library, which is in user space along with the application.
- This library multiplexes the streams and encodes them in RTP packets, which it stuffs into a socket. On the operating sys-tem side of the socket, UDP packets are generated to wrap the RTP packets and handed to IP for transmission over a link such as Ethernet.
- The multimedia application eventually receives multi-media data from the RTP library. It is responsible for playing out the media.

RTP—The Real-time Transport Protocol

- The basic function of RTP is to multiplex several real-time data streams onto a single stream of UDP packets. The UDP stream can be sent to a single destination (unicasting) or to multiple destinations (multicasting).
- The RTP format contains several features to help receivers work with multi-media information. Each packet sent in an RTP stream is given a number one higher than its predecessor.
- RTP provides a header field in which the source can specify the encoding but is otherwise not involved in how encoding is done.

RTCP—The Real-time Transport Control Protocol

- RTP has a little sister protocol (little sibling protocol?) called **RTCP** (**Real-time Transport Control Protocol**). The first function can be used to provide feedback on delay, variation in delay or jitter, bandwidth, congestion, and other network properties to the sources.
- RTCP also handles inter stream synchronization. The problem is that different streams may use different clocks, with different granularities and different drift rates. RTCP can be used to keep them in sync.

THE INTERNET TRANSPORT PROTOCOLS: TCP

It was specifically designed to provide a reliable end-to end byte stream over an unreliable network. It was designed to adapt dynamically to properties of the inter network and to be robust in the face of many kinds of failures.

Each machine supporting TCP has a TCP transport entity, which accepts user data streams from local processes, breaks them up into pieces not exceeding 64kbytes and sends each piece as a separate IP datagram. When these datagrams arrive at a machine, they are given to TCP entity, which reconstructs the original byte streams. It is up to TCP to time out and retransmits them as needed, also to reassemble datagrams into messages in proper sequence.

The different issues to be considered are:

- The TCP Service Model
- The TCP Protocol
- The TCP Segment Header
- The Connection Establishment
- Connection Release
- Connection Management Modeling
- TCP sliding window
- TCP Timer Management.
- TCP Congestion Control

The TCP Service Model

- ✓ TCP service is obtained by having both the sender and receiver create end points called SOCKETS.
- ✓ Each socket has a socket number(address)consisting of the IP address of the host, called a "PORT" (= TSAP)
- ✓ To obtain TCP service a connection must be explicitly established between a socket on the sending machine and a socket on the receiving machine
- ✓ All TCP connections are full duplex and point to point i.e., multicasting or broadcasting is not supported.
- ✓ A TCP connection is a byte stream, not a message stream i.e., the data is delivered as chunks.
- ✓ **Sockets:** A socket may be used for multiple connections at the same time. In other words, 2 or more connections may terminate at same socket.

The TCP Protocol

- ✓ A key feature of TCP, and one which dominates the protocol design, is that every byte on a TCP connection has its own 32-bit sequence number.
- ✓ When the Internet began, the lines between routers were mostly 56-kbps leased lines, so a host blasting away at full speed took over 1 week to cycle through the sequence numbers.
- ✓ The basic protocol used by TCP entities is the sliding window protocol.
- ✓ When a sender transmits a segment, it also starts a timer.

- ✓ When the segment arrives at the destination, the receiving TCP entity sends back a segment (with data if any exist, otherwise without data) bearing an acknowledgement number equal to the next sequence number it expects to receive.
- ✓ If the sender's timer goes off before the acknowledgement is received, the sender transmits the segment again.

The TCP Segment Header

Every segment begins with a fixed-format, 20-byte header. The fixed header may be followed by header options. After the options, if any, up to 65,535 - 20 - 20 = 65,495 data bytes may follow, where the first 20 refer to the IP header and the second to the TCP header. Segments without any data are legal and are commonly used for acknowledgements and control messages.

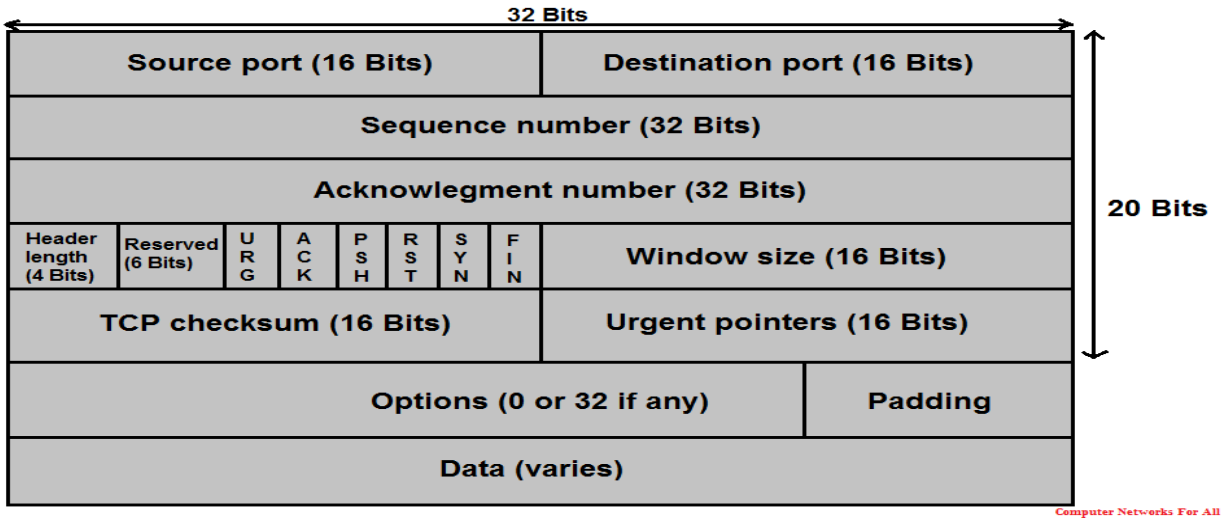


Fig 1: The TCP Header

Source Port, Destination Port : Identify local end points of the connections

Sequence number: Specifies the sequence number of the segment Acknowledgement Number: Specifies the next byte expected.

TCP header length: Tells how many 32-bit words are contained in TCP header URG: It is set to 1 if URGENT pointer is in use, which indicates start of urgent data.

ACK: It is set to 1 to indicate that the acknowledgement number is valid.

PSH: Indicates pushed data

RST: It is used to reset a connection that has become confused due to reject an invalid segment or refuse an attempt to open a connection.

FIN: Used to release a connection.

SYN: Used to establish connections.

TCP Connection Establishment

- To establish a connection, one side, say, the server, passively waits for an incoming connection by executing the LISTEN and ACCEPT primitives in that order, either specifying a specific source or nobody in particular.
- some process is listening to the port, that process is given the incoming TCP segment. It can either accept or reject the connection. If it accepts, an acknowledgement segment is sent back.
- One way to defend against this attack is to use **SYN cookies**. Instead of remembering the sequence number, a host chooses a cryptographically generated sequence number, puts it on the outgoing segment, and forgets it. If the three-way handshake completes, this sequence number (plus 1) will be returned to the host.
- This procedure allows the host to check that an acknowledged sequence number is correct without having to remember the sequence number separately.

TCP Connection Release

- Although TCP connections are full duplex, to understand how connections are released it is best to think of them as a pair of simplex connections.
- Each simplex connection is released independently of its sibling. To release a connection, either party can send a TCP segment with the FIN bit set, which means that it has no more data to transmit.
- When the FIN is acknowledged, that direction is shut down for new data. Data may continue to flow indefinitely in the other direction, however.
- When both directions have been shut down, the connection is released.
- Normally, four TCP segments are needed to release a connection, one FIN and one ACK for each direction. However, it is possible for the first ACK and the second FIN to be contained in the same segment, reducing the total count to three.

TCP Connection Management Modeling

The steps required establishing and release connections can be represented in a finite state machine with the 11 states listed in Fig. 1. In each state, certain events are legal. When a legal event happens, some action may be taken. If some other event happens, an error is reported.

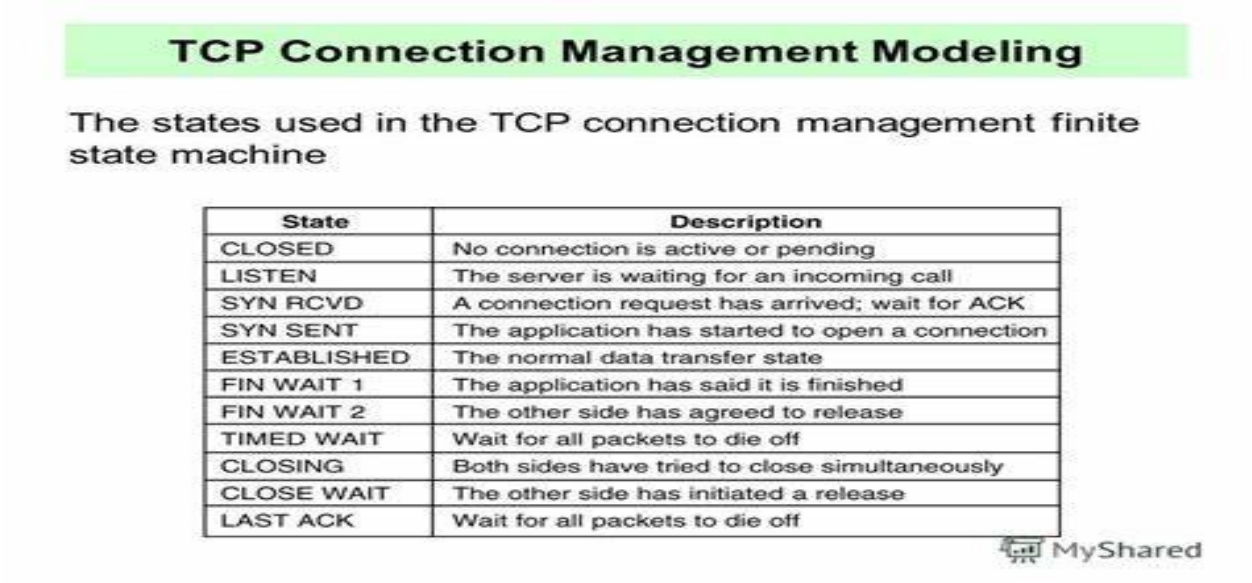


Fig 1:The states used in the TCP connection management finite state machine.

TCP Connection management from server's point of view:

1. The server does a LISTEN and settles down to see who turns up.
2. When a SYN comes in, the server acknowledges it and goes to the SYNRCVD state
3. When the servers SYN is itself acknowledged the 3-way handshake is complete and server goes to the ESTABLISHED state. Data transfer can now occur.
4. When the client has had enough, it does a close, which causes a FIN to arrive at the server [dashed box marked passive close].
5. The server is then signaled.
6. When it too, does a CLOSE, a FIN is sent to the client.
7. When the client's acknowledgement shows up, the server releases the connection and deletes the connection record.

TCP Sliding Window

- This is one of the problems that ruin the TCP performance, which occurs when data are passed to the sending TCP entity in large blocks, but an interactive application on the receiving side reads 1 byte at a time.

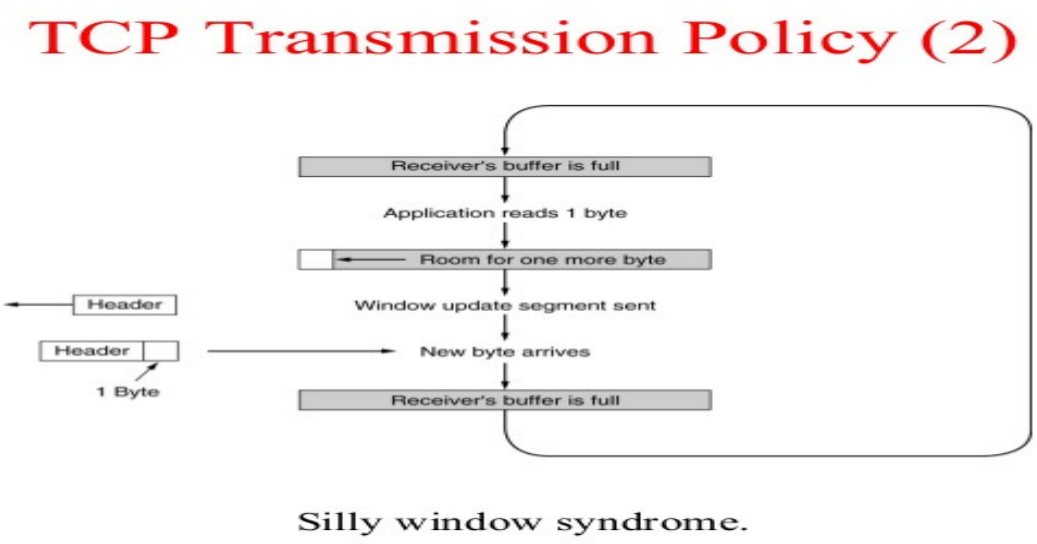


Figure 1 Silly window syndrome

- .Initially the TCP buffer on the receiving side is full and the sender knows this(win=0) .
- Then the interactive application reads 1 character from tcp stream.
- Now, the receiving TCP sends a window update to the sender saying that it is all right to send 1 byte. The sender obligates and sends 1 byte.
- The buffer is now full, and so the receiver acknowledges the 1 byte segment but sets window to zero. This behavior can go on forever.

TCP TIMER MANAGEMENT:

The most important of these is the **RTO** (**Retransmission TimeOut**)

TCP uses 3 kinds of timers:

1. Retransmission timer
2. Persistence timer
3. Keep-Alive timer.

1. Retransmission timer: When a segment is sent, a timer is started. If the segment is acknowledged before the timer expires, the timer is stopped. If on the other hand, the timer goes off before the acknowledgement comes in, the segment is retransmitted and the timer is started again. The algorithm that constantly adjusts the time-out interval, based on continuous measurements of n/w performance was proposed by JACOBSON and works as follows:

for each connection, TCP maintains a variable RTT, that is the best current estimate of the round trip time to the destination inn question.

When a segment is sent, a timer is started, both to see how long the acknowledgement takes and to trigger a retransmission if it takes too long.

If the acknowledgement gets back before the timer expires, TCP measures how long the measurements took say M It then updates RTT according to the formula

RTT = αRTT + (1-α) M

2. Persistence timer:

 The receiver sends an acknowledgement with a window size of '0' telling the sender to wait later, the receiver updates the window, but the packet with the update is lost now both the sender and receiver are waiting for each other to do something

 when the persistence timer goes off, the sender transmits a probe to the receiver the response to the probe gives the window size.

 if it is still zero, the persistence timer is set again and the cycle repeats.

 if it is non zero, data can now be sent.

3. Keep-Alive timer: When a connection has been idle for a long time, this timer may go off to cause one side to check if other side is still there. If it fails to respond, the connection is terminated.

TCP CONGESTION CONTROL:

- The network layer detects congestion when queues grow large at routers and tries to manage it, if only by dropping packets. It is up to the transport layer to receive congestion feedback from the network layer and slow down the rate of traffic that it is sending into the network.
- In the Internet, TCP plays the main role in controlling congestion, as well as the main role in reliable transport.
- TCP congestion control is based on implementing this approach using a window and with packet loss as the binary signal.

  Different congestion control algorithms used by TCP are:
  - ✓ RTT variance Estimation.
  - ✓ Exponential RTO back-off } Re-transmission Timer Management
  - ✓ Karn's Algorithm
  - ✓ Slow Start
  - ✓ Dynamic window sizing on congestion
  - ✓ Fast Retransmit } Window Management
  - ✓ Fast Recovery

The Future of TCP

- ➢ TCP has been used for many applications and extended over time to give good performance over a wide range of networks.
- ➢ The first one is that TCP does not provide the transport semantics that all applications want.
- ➢ This has led to proposals for new protocols that would provide a slightly different interface.
- ➢ The second issue is congestion control. You may have expected that this is a solved problem after our deliberations and the mechanisms that have been developed over time.

----------UNIT IV COMPLETED----------

UNIT V

THE APPLICATION LAYER

The application layer **is a** layer in the Open Systems Interconnection .The application layer is a component within a application that controls the communication method to other devices. It's an abstraction layer service that masks the rest of the application from the transmission process. The application layer relies on all the layers below it to complete its process.

DNS—THE DOMAIN NAME SYSTEM

The essence of DNS is the invention of a hierarchical, domain-based naming scheme and a distributed database system for implementing this naming scheme.
To map a name onto an IP address, an application program calls a library procedure called the **resolver.**

The DNS Name Space

Managing a large and constantly changing set of names is a nontrivial problem. . The leaves of the tree represent domains that have no sub domains (but do contain machines, of course). A leaf domain may contain a single host, or it may represent a company and contain thousands of hosts.

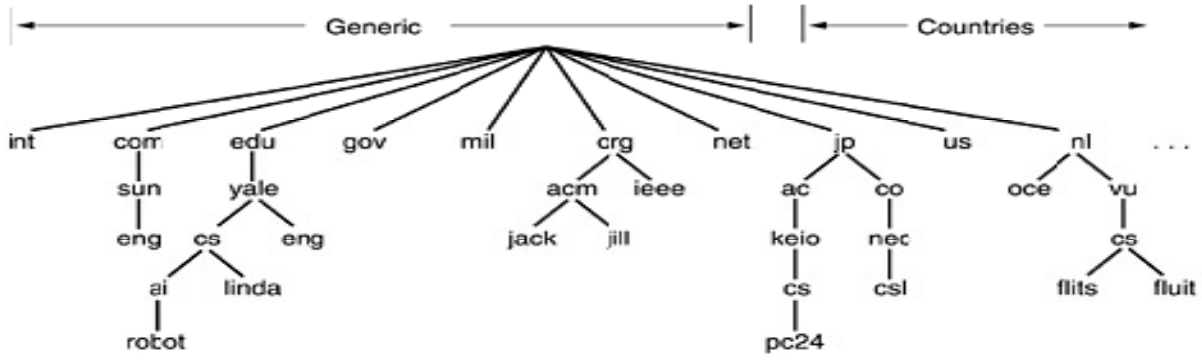


Fig 1 :A portion of the Internet domain name space.

The top-level domains come in two flavors: generic and countries. The original generic domains were *com (commercial), edu* (educational institutions), *gov* (the U.S. Federal Government), *int* (certain international organizations), *mil* (the U.S. armed forces), *net* (network providers), and *org* (nonprofit organizations).

The practice of registering a domain only to turn around and sell it off to an interested party at a much higher price even has a name. It is called **cybersquatting**. Many companies that were slow off the mark when the Internet era began found their obvious domain names already taken when they tried to acquire them.

Domain Resource Records

Every domain, whether it is a single host or a top-level domain, can have a set of **resource records** associated with it.

A resource record is a five tuple. Although they are encoded in binary for efficiency, in most expositions, resource records are presented as ASCII text, one line per resource record. The format we will use is as follows:

Domain - name Time to live Class Type Value

The *Domain_name* tells the domain to which this record applies. Normally, many records exist for each domain and each copy of the database holds information about multiple domains.

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Fig 1.The principal DNS resource record types for IPv4.

Another important record type is the *NS* record. It specifies a name server for the domain or sub domain. This is a host that has a copy of the database for a domain.

Name Servers

A single name server could contain the entire DNS database and respond to all queries about it. In practice, this server would be so overloaded as to be useless. To avoid the problems associated with having only a single source of information, the DNS name space is divided into non overlapping **zones**. One possible way to divide the name space of Fig1. Each circled zone contains some part of the tree.

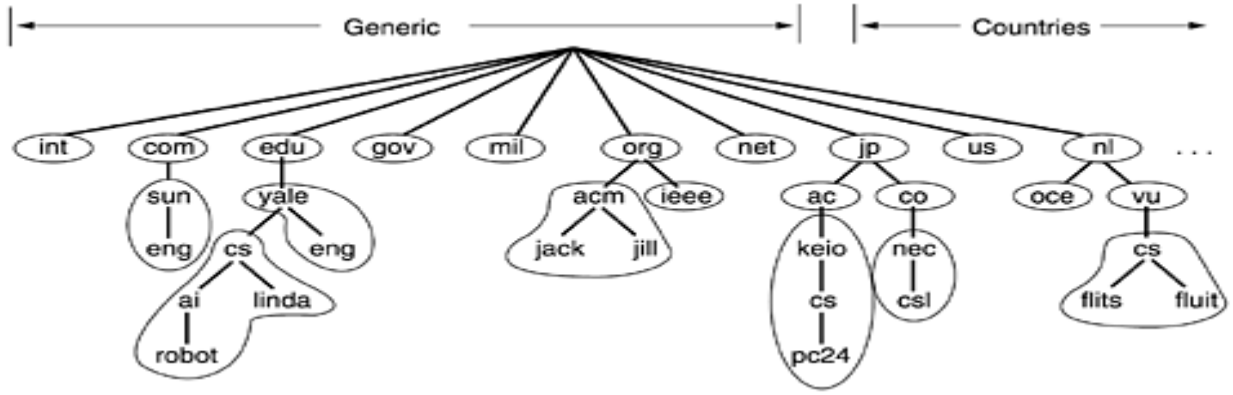


Figure 1. Part of the DNS name space divided into zones (which are circled).

Each zone is also associated with one or more name servers. These are hosts that hold the database for the zone. Normally, a zone will have one primary name server, which gets its information from a file on its disk, and one or more secondary name servers, which get their information from the primary name server.

The process of looking up a name and finding an address is called name resolution. When a resolver has a query about a domain name, it passes the query to a local name server. An authoritative record is one that comes from the authority that manages the record and is thus always correct. Authoritative records are in contrast to cached records, which may be out of date.

ELECTRONIC MAIL

Electronic mail, or more commonly **email**, has been around for over three decades. Faster than mail, email has been a popular application since the early days of the Internet.

The email protocols have evolved during the period of their use, too. The first email systems simply consisted of file transfer protocols, with the convention that the first line of each message (i.e., file) contained the recipient's address. Electronic mail, or **e-mail**, as it is known to its many fans, has been around for over two decades

1.Architecture and Services

They normally consist of two subsystems

- The **user agents**, which allow people to read and send e-mail,
- the **message transfer agents**, which move the messages from the source to the destination.

E-mail systems support five basic functions

- **Composition** refers to the process of creating messages and answers. Although any text editor can be used for the body of the message, the system itself can provide assistance with addressing and the numerous header fields attached to each message
- **Transfer** refers to moving messages from the originator to the recipient
- **Reporting** has to do with telling the originator what happened to the message
- **Displaying** incoming messages is needed so people can read their e-mail. Sometimes conversion is required or a special viewer must be invoked,
- **Disposition** is the final step and concerns what the recipient does with the message after receiving it.
- Most systems allow users to create **mailboxes** to store incoming e-mail. Commands are needed to create and destroy mailboxes, inspect the contents of mailboxes, insert and delete messages from mailboxes, and so on.
- Most systems allow users to create **mailboxes** to store incoming e-mail. Commands are needed to create and destroy mailboxes, inspect the contents of mailboxes, insert and delete messages from mailboxes, and so on. Corporate managers often need to send a message to each of their subordinates, customers, or suppliers. This gives rise to the idea of a **mailing list**, which is a list of e-mail addresses. When a message is sent to the mailing list, identical copies are delivered to everyone on the list.

2.The User Agent

- E-mail systems have two basic parts, as we have seen: the user agents and the message transfer agents. In this section we will look at the user agents. A user agent is normally a program (sometimes called a mail reader) that accepts a variety of commands for composing, receiving, and replying to messages, as well as for manipulating mailboxes. Some user agents have a fancy menu- or icon-driven interface that requires a mouse, 2.whereas others expect 1-character commands from the keyboard.

- ***Sending E-mail***

To send an e-mail message, a user must provide the message, the destination address, and possibly some other parameters. The message can be produced with a free-standing text editor, a word processing program, or possibly with a specialized text editor built into the user agent. The destination address must be in a format that the user agent can deal with

- ***Reading E-mail***

Typically, when a user agent is started up, it looks at the user's mailbox for incoming e-mail before displaying anything on the screen.

3. Message Formats

Messages sent by the user agent must be placed in a standard format to be handled by the message transfer agents. First we will look at basic ASCII email using RFC 5322, which is the latest revision of the original Internet message format as described in RFC 822.

RFC 5322—The Internet Message Format

The original RFC 822 was de-signed decades ago and did not clearly distinguish the envelope fields from the header fields.

Although it has been revised to RFC 5322, completely redoing it was not possible due to its widespread usage. In normal usage, the user agent builds a message and passes it to the message transfer agent, which then uses some of the header fields to construct the actual envelope, a somewhat old-fashioned mixing of message and envelope.

The principal header fields related to message transport are listed in Fig.1. The *To:* field gives the DNS address of the primary recipient. Having multiple recipients is also allowed. The *Cc:* field gives the addresses of any secondary recipients. In terms of delivery, there is no distinction between the primary and secondary recipients.

Header	Meaning
To:	Email address(es) of primary recipient(s)
Cc:	Email address(es) of secondary recipient(s)
Bcc:	Email address(es) for blind carbon copies

From:	Person or people who created the message
Sender:	Email address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Figure 1. RFC 5322 header fields related to message transport.

The next two fields, *From:* and *Sender:*, tell who wrote and sent the message, respectively. These need not be the same. For example, a business executive may write a message, but her assistant may be the one who actually transmits it.

MIME—The Multipurpose Internet Mail Extensions

The basic idea of MIME is to continue to use the RFC 822 format, but to add structure to the message body and define encoding rules for non-ASCII messages. It is widely used for mail messages that are sent across the Internet, as well as to describe content for other applications such as Web browsing. Any message not containing a *MIME-Version:* header is assumed to be an English plaintext message (or at least one using only ASCII characters) and is processed as such.

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Figure 1 Message headers added by MIME.

The *Content-Description:* header is an ASCII string telling what is in the message. This header is needed so the recipient will know whether it is worth decoding and reading the message

4.Message Transfer

The message transfer agents replay messages from the originator to the recipient. The mail transfer is done with the SMTP protocol.

The simplest way to move messages is to establish a transport connection from the source machine to the destination machine and then just transfer the message. This is how SMTP originally worked.

SMTP—The Simple Mail Transfer Protocol

SMTP is a simple ASCII protocol. After establishing the TCP connection to port 25, the sending machine, operating as the client, waits for the receiving machine, operating as the server, to talk first. The basic SMTP works well, but it is limited in several respects. It does not include authentication. This means that the *FROM* command in the example could give any sender address that it pleases.

Mail Submission

Originally, user agents ran on the same computer as the sending message transfer agent. In this setting, all that is required to send a message is for the user agent to talk to the local mail server, using the dialog that we have just described.. Mail transfer agents run on ISP and company servers. They are always connected to the Internet. This difference means that a user agent in Boston may need to contact its regular mail server in Seattle to send a mail message because the user is traveling.

Message Transfer

Once the sending mail transfer agent receives a message from the user agent, it will deliver it to the receiving mail transfer agent using SMTP. To do this, the sender uses the destination address.

5.Final Delivery

Our mail message is almost delivered. It has arrived at Bob's mailbox. All that remains is to transfer a copy of the message to Bob's user agent for display. The mail transfer agent simply wrote new messages to the end of the mailbox file, and the user agent simply checked the mailbox file for new mail.

IMAP—The Internet Message Access Protocol

One of the main protocols that is used for final delivery is **IMAP** (**Internet Message Access Protocol**). Version 4 of the protocol is defined in RFC 3501. To use IMAP, the mail server runs an IMAP server that listens to port 143. The user agent runs an IMAP client.

Webmail

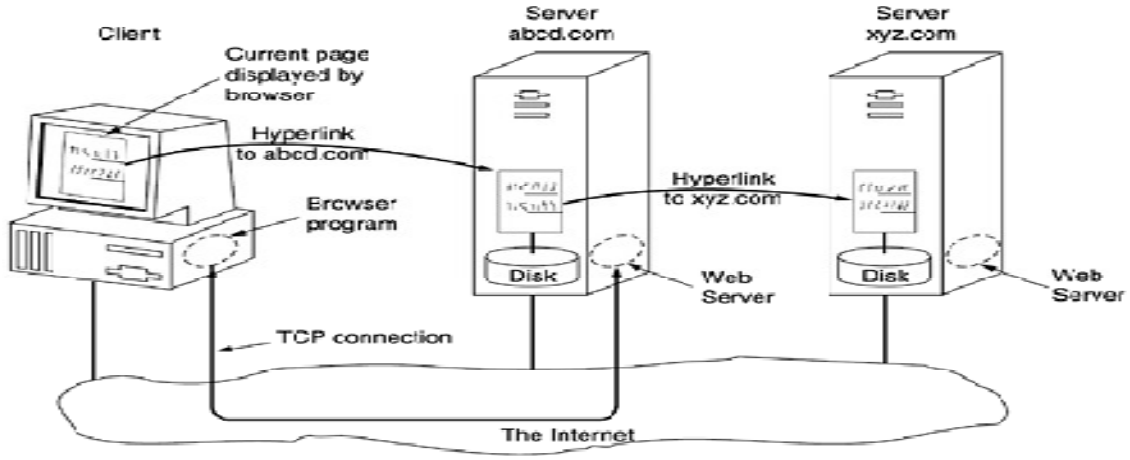
An increasingly popular alternative to IMAP and SMTP for providing email service is to use the Web as an interface for sending and receiving mail. Widely used **Webmail** systems include Google Gmail, Microsoft Hotmail and Yahoo! Mail. Webmail is one example of software (in this case, a mail user agent) that is provided as a service using the Web.

THE WORLD WIDE WEB

In 1994, CERN and M.I.T. signed an agreement setting up the **World Wide Web Consortium** (sometimes abbreviated as **W3C**), an organization devoted to further developing the Web, standardizing protocols, and encouraging interoperability between sites

1.Architectural Overview

.This process can be repeated indefinitely. The idea of having one page point to another, now called **hypertext**, was invented by a visionary M.I.T. professor of electrical engineering, Vannevar Bush, in 1945 (Bush, 1945). This was long before the Internet was invented. A dynamic page may present itself differently each time it is displayed. For example, the front page for an electronic store may be different for each visitor.



The Client Side

When a user clicks on a hyperlink, the browser carries out a series of steps in order to fetch the page pointed to. Suppose that a user is browsing the Web and finds a link on Internet telephony that points to ITU's home page, which is *http://www.itu.org/home/index.html*. Let us trace the steps that occur when this link is selected

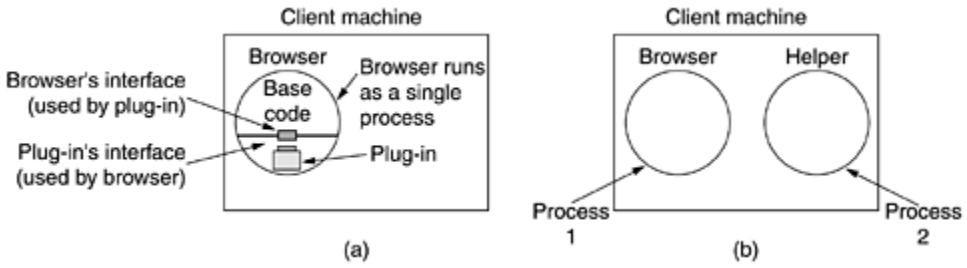
1. The browser determines the URL (by seeing what was selected).
2. The browser asks DNS for the IP address of *www.itu.org*.
3. DNS replies with 156.106.192.32.
4. The browser makes a TCP connection to port 80 on 156.106.192.32.
5. It then sends over a request asking for file */home/index.html*.
6. The *www.itu.org* server sends the file */home/index.html*.
7. The TCP connection is released.
8. The browser displays all the text in */home/index.html*.
9. The browser fetches and displays all images in this file.

Many browsers display which step they are currently executing in a status line at the bottom of the screen. In this way, when the performance is poor, the user can see if it is due to DNS not responding, the server not responding, or simply network congestion during page transmission.

To be able to display the new page (or any page), the browser has to understand its format. To allow all browsers to understand all Web pages, Web pages are written in a standardized language called HTML, which describes Web pages.

In addition to having ordinary text (not underlined) and hypertext (underlined), Web pages can also contain icons, line drawings, maps, and photographs.

There are two possibilities: plug-ins and helper applications. A **plug-in** is a code module that the browser fetches from a special directory on the disk and installs as an extension to itself .Because plug-ins run inside the browser, they have access to the current page and can modify its appearance. After the plug-in has done its job (usually after the user has moved to a different Web page), the plug-in is removed from the browser's memory.



Each browser has a set of procedures that all plug-ins must implement so the browser can call the plug-in. For example, there is typically a procedure the browser's base code calls to supply the plug-in with data to display. This set of procedures is the plugin's interface and is browser specific.

The other way to extend a browser is to use a **helper application**. This is a complete program, running as a separate process. Since the helper is a separate program, it offers no interface to the browser and makes no use of browser services. Instead, it usually just accepts the name of a scratch file where the content file has been stored, opens the file, and displays the contents.

The Server Side

A Web server is similar to the server of Fig. 6-6. That server, like a real Web server, is given the name of a file to look up and return. In both cases, the steps that the server performs in its main loop are:

1. Accept a TCP connection from a client (a browser).
2. Get the name of the file requested.
3. Get the file (from disk).
4. Return the file to the client.
5. Release the TCP connection.

The next step for building a faster server is to make the server multithreaded. In one design, the server consists of a front-end module that accepts all incoming requests and k processing modules, as shown below The $k + 1$ threads all belong to the same process so the processing modules all have access to the cache within the process' address space. When a request comes in, the front end accepts it and builds a short record describing it. It then hands the record to one of the processing modules. In another possible design, the front end is eliminated and each processing module tries to acquire its own requests, but a locking protocol is then required to prevent conflicts.

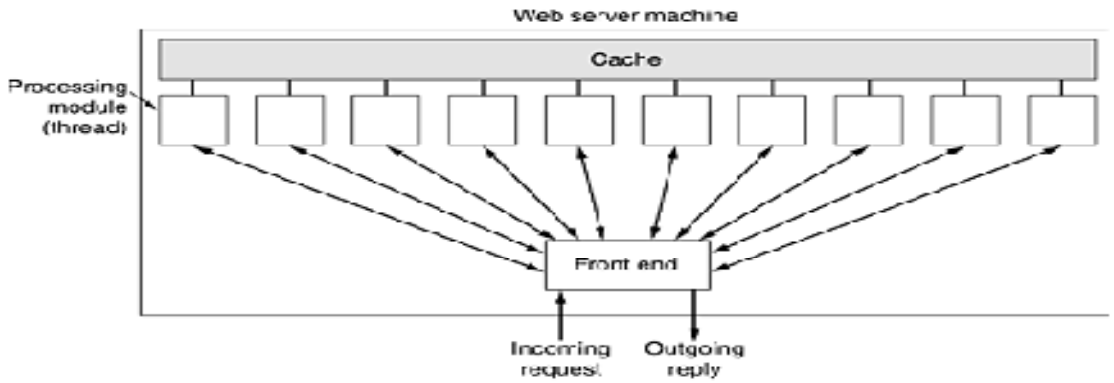


Figure 1 A multithreaded Web server with a front end and processing modules.

The processing module first checks the cache to see if the file needed is there. If so, it updates the record to include a pointer to the file in the record. If it is not there, the processing module starts a disk operation to read it into the cache .The advantage of this scheme is that while one or more processing modules are blocked waiting for a disk operation to complete (and thus consuming no CPU time), other modules can be actively working on other requests.

Modern Web servers do more than just accept file names and return files. In fact, the actual processing of each request can get quite complicated. For this reason, in many servers each processing module performs a series of steps. The front end passes each incoming request to the first available module, which then carries it out using some subset of the following steps, depending on which ones are needed for that particular request.

1. Resolve the name of the Web page requested.
2. Authenticate the client.
3. Perform access control on the client.
4. Perform access control on the Web page.
5. Check the cache.
6. Fetch the requested page from disk.
7. Determine the MIME type to include in the response.
8. Take care of miscellaneous odds and ends.
9. Return the reply to the client.
10. Make an entry in the server log.

2. Static Web Pages

The basis of the Web is transferring Web pages from server to client. In the simplest form, Web pages are static. That is, they are just files sitting on some server that present themselves in the same way each time they are fetched and viewed. Just because they are static does not mean that the pages are inert at the browser, however. A page containing a video can be a static Web page.

As mentioned earlier, the lingua franca of the Web, in which most pages are written, is HTML. The home pages of teachers are usually static HTML pages.

The home pages of companies are usually dynamic pages put together by a Web design company. In this section, we will take a brief look at static HTML pages as a foundation for later material.

HTML—The HyperText Markup Language

HTML (**HyperText Markup Language**) was introduced with the Web. It allows users to produce Web pages that include text, graphics, video, pointers to other Web pages, and more.

HTML is a markup language, or language for describing how documents are to be formatted. The term ‘‘markup’’ comes from the old days when copyeditors actually marked up documents to tell the print in those days, a human being—which fonts to use, and so on.

Markup languages thus contain explicit commands for formatting. For example, in HTML, **means start boldface mode, and** means leave boldface mode. LaTeX and TeX are other examples of markup languages that are well known to most academic authors.

HTML provides various mechanisms for making lists, including nested lists.

3.Dynamic Web Documents

It is proving so successful that it is rivaling traditional application software. Of course, the fact that these applications are offered for free by large providers helps.

Server-Side Dynamic Web Page Generation

A user fills in a form and clicks on the *submit* button, a message is sent to the server indicating that it contains the contents of a form, along with the fields the user filled in. This message is not the name of a file to return. What is needed is that the message is given to a program or script to process. Usually, the processing involves using the user-supplied information to look up a record in a database on the server's disk and generate a custom HTML page to send back to the client.

The traditional way to handle forms and other interactive Web pages is a system called the **CGI** (**Common Gateway Interface**). It is a standardized interface to allow Web servers to talk to back-end programs and scripts that can accept input (e.g., from forms) and generate HTML pages in response.

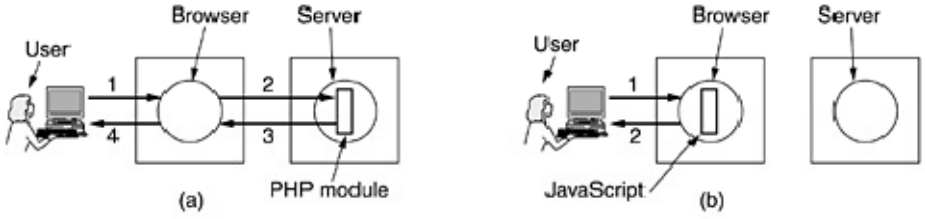
Client-Side Dynamic Web Page Generation

CGI, PHP, JSP, and ASP scripts solve the problem of handling forms and interactions with databases on the server. They can all accept incoming information from forms, look up information in one or more databases, and generate HTML pages with the results. What none of them can do is respond to mouse movements or interact with users directly.

JavaScript is a scripting language, *very* loosely inspired by some ideas from the Java programming language. This difference does not mean that JavaScript is better than PHP. Their uses are completely different. PHP (and, by implication, JSP and ASP) are used when interaction with a remote database is needed. JavaScript is used when the interaction is with the user at the client computer. It is certainly possible (and common) to have HTML pages that use both PHP and JavaScript, although they cannot do the same work or own the same button, of course.

JavaScript is a full-blown programming language, with all the power of C or Java. It has variables, strings, arrays, objects, functions, and all the usual control structures. It also has a large number of facilities specific for Web pages, including the ability to manage windows and frames, set and get cookies, deal with forms, and handle hyperlinks.

(a) Server-side scripting with PHP. (b) Client-side scripting with JavaScript.



4.HTTP—The HyperText Transfer Protocol

The transfer protocol used throughout the World Wide Web is **HTTP** (**HyperText Transfer Protocol**). It specifies what messages clients may send to servers and what responses they get back in return. The value of using TCP is that neither browsers nor servers have to worry about lost messages, duplicate messages, long messages, or acknowledgements. All of these matters are handled by the TCP implementation.

Methods:

Although HTTP was designed for use in the Web, it has been intentionally made more general than necessary with an eye to future object-oriented applications. For this reason, operations, called **methods**, other than just requesting a Web page are supported

- The *GET* method requests the server to send the page (by which we mean object, in the most general case, but in practice normally just a file).
- The *HEAD* method just asks for the message header, without the actual page. This method can be used to get a page's time of last modification, to collect information for indexing purposes, or just to test a URL for validity.
- The *PUT* method is the reverse of *GET*: instead of reading the page, it writes the page. This method makes it possible to build a collection of Web pages on a remote server. The body of the request contains the page. It may be encoded using MIME, in which case the lines following the *PUT* might include *Content-Type* and authentication headers, to prove that the caller indeed has permission to perform the requested operation.
- *DELETE* does what you might expect: it removes the page. As with *PUT*, authentication and permission play a major role here
- The *TRACE* method is for debugging. It instructs the server to send back the request. This method is useful when requests are not being processed correctly and the client wants to know what request the server actually got.
- The *CONNECT* method is not currently used. It is reserved for future use.
- The *OPTIONS* method provides a way for the client to query the server about its properties or those of a specific file.

Every request gets a response consisting of a status line, and possibly additional information (e.g., all or part of a Web page). The status line contains a three-digit status code telling whether

the request was satisfied, and if not, why not.

5.The Mobile Web

The Web is used from most every type of computer, and that includes mobile phones. Browsing the Web over a wireless network while mobile can be very useful. It also presents technical problems because much Web content was designed for flashy presentations on desktop computers with broadband connectivity. In this section we will describe how Web access from mobile devices, or the **mobile Web**, is being developed.

Compared to desktop computers at work or at home, mobile phones present several difficulties for Web browsing:

1. Relatively small screens preclude large pages and large images.
2. Limited input capabilities make it tedious to enter URLs or other lengthy input.
3. Network bandwidth is limited over wireless links, particularly on cellular (3G) networks, where it is often expensive too.
4. Connectivity may be intermittent.
5. Computing power is limited, for reasons of battery life, size, heat dissipation, and cost.

These difficulties mean that simply using desktop content for the mobile Web is likely to deliver a frustrating user experience.

6 .Web Search

To perform a Web search in the traditional manner, the user directs her brow-ser to the URL of a Web search site. The major search sites include Google, Yahoo!, and Bing. Next, the user submits search terms using a form.

To finish our description of the Web, we will discuss what is arguably the most successful Web application: search. In 1998, Sergey Brin and Larry Page, then graduate students at Stanford, formed a startup called Google to build a better Web search engine.

They were armed with the then-radical idea that a search algorithm that counted how many times each page was pointed to by other pages was a better measure of its importance than how many times it contained the key words being sought.

For instance, many pages link to the main Cisco page, which makes this page more important to a user searching for ‘‘Cisco’’ than a page out-side of the company that happens to use the word ‘‘Cisco’’ many times.

NETWORK SECURITY

- Network security is a specialized field in computer networking that involves securing a computer network infrastructure.
- Network security is an over-arching term that describes that the policies and procedures implemented by a network administrator to avoid and keep track of unauthorized access, exploitation, modification, or denial of the network and network resources.
- Computers are an integral part of everyday operations. Organizations depend on them, A computer system failure will have a critical impact on the organization.
- The objective of security management is to eliminate or minimize computer vulnerability to destruction, modification, or disclosure.
- Computer security itself is a term that has many meanings and related terms.
- Subjects such as authentication and access controls must be addressed in a broad terms of computer security.

- Thus the network security refer to the protection of the multiple computers and other devices that are connected together. Related to these two terms are two others, information security and information assurance.

CRYPTOGRAPHY

A cryptographic key is a string of bits used by a cryptographic algorithm to transform plain text into **cipher** text or vice versa. This key remains private and ensures secure communication. A cryptographic key is the core part of cryptographic operations. Many cryptographic systems include pairs of operations, such as encryption and decryption..

Substitution Ciphers

In a **substitution cipher**, each letter or group of letters is replaced by another letter or group of letters to disguise it. One of the oldest known ciphers is the **Caesar cipher**, attributed to Julius Caesar. With this method, *a* becomes *D*, *b* be-comes *E*, *c* becomes *F*, . . . , and *z* becomes *C*. For example, *attack* becomes *DWWDFN*. In our examples, plaintext will be given in lowercase letters, and cipher text in uppercase letters.

A slight generalization of the Caesar cipher allows the cipher text alphabet to be shifted by *k* letters, instead of always three. In this case, *k* becomes a key to the general method of circularly shifted alphabets. The Caesar cipher may have fooled Pompey, but it has not fooled anyone since. The general system of symbol-for-symbol substitution is called a **mono alphabetic substitution cipher.**

Transposition Ciphers

Substitution ciphers preserve the order of the plaintext symbols but disguise them. **Transposition ciphers**, in contrast, reorder the letters but do not disguise them. depicts a common transposition cipher, the columnar transposition. The cipher is keyed by a word or phrase not containing any repeated letters. In this example, MEGABUCK is the key. The purpose of the key is to order the columns, with column 1 being under the key letter closest to the start of the alpha-bet, and so on.

Some transposition ciphers accept a fixed-length block of input and produce a fixed-length block of output. These ciphers can be completely described by giving a list telling the order in which the characters are to be output.

One-Time Pads

Constructing an unbreakable cipher is actually quite easy; the technique has been known for decades. First choose a random bit string as the key. Then convert the plaintext into a bit string, for example, by using its ASCII representation. Finally, compute the XOR (eXclusive OR) of these two strings, bit by bit. The resulting cipher text cannot be broken because in a sufficiently large sample of cipher text, each letter will occur equally often, as will every digram, every tri-gram, and so on. This method, known as the **one-time pad**, is immune to all present and future attacks, no matter how much computational power the intruder has. The reason derives from information theory: there is simply no information in the message because all possible plaintexts of the given length are equally likely.

Quantum Cryptography

Interestingly, there may be a solution to the problem of how to transmit the one-time pad over the network, and it comes from a very unlikely source: quantum mechanics. This area is still experimental, but initial tests are promising. If it can be perfected and be made efficient, virtually all cryptography will eventually be done using one-time pads since they are provably secure. Below we will briefly explain

how this method, **quantum cryptography**, works. In particular, we will describe a protocol called **BB84** after its authors and publication year (Bennet and Brassard, 1984).

A user, Alice, wants to establish a one-time pad with a second user, Bob. Alice and Bob are called **principals**, the main characters in our story. Quantum cryptography is based on the fact that light comes in little packets called **photons**, which have some peculiar properties.

Two Fundamental Cryptographic Principles

Although there are many different cryptographic systems in the pages ahead, two principles underlying all of them are important to understand.

Redundancy

The first principle is that all encrypted messages must contain some redundancy, that is, information not needed to understand the message.. Thinking they are being very efficient, TCP's programmers decide that ordering messages should consist of a 16-byte customer name followed by a 3-byte data field (1 byte for the quantity and 2 bytes for the product number). The last 3 bytes are to be encrypted using a very long key known only by the customer and TCP.

Freshness

The second cryptographic principle is that measures must be taken to ensure that each message received can be verified as being fresh, that is, sent very recently. This measure is needed to prevent active intruders from playing back old messages. If no such measures were taken, our ex-employee could tap TCP's phone line and just keep repeating previously sent valid messages.

SYMMETRIC-KEY ALGORITHMS

A secret **key algorithm** (sometimes called a **symmetric algorithm**) is a **cryptographic algorithm** that uses the same **key** to encrypt and decrypt data.

Modern cryptography uses the same basic ideas as traditional cryptography (transposition and substitution), but its emphasis is different. Traditionally, cryptographers have used simple algorithms.). A software implementation is programmed as a loop with at least eight iterations, each one per-forming S-box-type substitutions on sub blocks of the 64- to 256-bit data block.

DES—The Data Encryption Standard

DES (**Data Encryption Standard**), was widely adopted by the industry for use in security products. It is no longer secure in its original form, but in a modified form it is still useful. We will now explain how DES works.

DES has been enveloped in controversy since the day it was launched. It was based on a cipher developed and patented by IBM, called Lucifer, except that IBM's cipher used a 128-bit key instead of a 56-bit key.

Triple DES

This design immediately gives rise to two questions. First, why are only two keys used, instead of three? Second, why is **EDE** (**Encrypt Decrypt Encrypt**) used, instead of **EEE** (**Encrypt Encrypt Encrypt**)? The reason that two keys are used is that even the most paranoid of cryptographers believe that 112 bits is considered a feature, not a bug.)

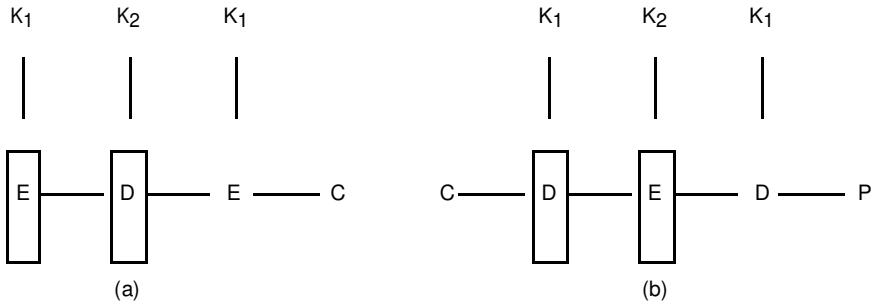


Figure1 (a) Triple encryption using DES. (b) Decryption.

Going to 168 bits would just add the unnecessary overhead of managing and transporting another key for little real gain. reason for encrypting, decrypting, and then encrypting again is backward compatibility with existing single-key DES systems. Both the encryption and decryption functions are mappings between sets of 64-bit numbers. From a crypto-graphic point of view, the two mappings are equally strong. By using EDE, how-ever, instead of EEE, a computer using triple encryption can speak to one using single encryption by just setting $K_1 = K_2$.

AES—The Advanced Encryption Standard

As DES began approaching the end of its useful life, even with triple DES, **NIST** (**National Institute of Standards and Technology**), the agency of the U.S. Dept.

All over the world were invited to submit proposals for a new standard, to be called **AES** (**Advanced Encryption Standard**). The bake-off rules were:

1. The algorithm must be a symmetric block cipher.
2. The full design must be public.
3. Key lengths of 128, 192, and 256 bits must be supported.
4. Both software and hardware implementations must be possible.
5. The algorithm must be public or licensed on nondiscriminatory terms.

Cipher Modes

Despite all this complexity, AES (or DES, or any block cipher for that matter) is basically a mono alphabetic substitution cipher using big characters (128-bit characters for AES and 64-bit characters for DES). Whenever the same plaintext block goes in the front end, the same cipher text block comes out the back end. If you encrypt the plaintext *abcdefgh* 100 times with the same DES key, you get the same cipher text 100 times.

Electronic Code Book Mode

To see how this mono alphabetic substitution cipher property can be used to partially defeat the cipher, we will use (triple) DES because it is easier to depict 64-bit blocks than 128-bit blocks, but AES has exactly the same problem.

The straightforward way to use DES to encrypt a long piece of plaintext is to break it up into consecutive 8-byte (64-bit) blocks and encrypt them one after another with the same key. The last piece of plaintext is padded out to 64 bits, if need be. This technique is known as **ECB mode** (**Electronic Code Book mode**) in analogy with old-fashioned code books where each plaintext word was listed, followed by its cipher text (usually a five-digit decimal number).

Cipher Block Chaining Mode

To thwart this type of attack, all block ciphers can be chained in various ways so that replacing a block the way Leslie did will cause the plaintext decrypted starting at the replaced block to be garbage. One way of chaining is **cipher block chaining**. In this method.

The each plaintext block is XORed with the previous cipher text block before being encrypted. Consequently, the same plaintext block no longer maps onto the same cipher text block, and the encryption is no longer a big mono alphabetic substitution cipher. The first block is XORed with a randomly chosen **IV** (**Initialization Vector**), which is transmitted (in plain-text) along with the cipher text.

Cipher Feedback Mod

The cipher block chaining has the disadvantage of requiring an entire 64-bit block to arrive before decryption can begin. For byte-by-byte encryption, **cipher feedback mode** using (triple) DES .For AES, the idea is exactly the same, only a 128-bit shift register is use., The state of the encryption machine is shown after bytes 0 through 9 have been encrypted and sent. When plaintext byte 10 arrives, as illustrated the DES algorithm operates on the 64-bit shift register to generate a 64-bit cipher text.

A problem with cipher feedback mode is that if one bit of the ciphertext is accidentally inverted during transmission, the 8 bytes that are decrypted while the bad byte is in the shift register will be corrupted. Once the bad byte is pushed out of the shift register, correct plaintext will once again be generated.

Stream Cipher Mode

Nevertheless, applications exist in which having a 1-bit transmission error mess up 64 bits of plaintext is too large an effect. For these applications, a fourth option, **stream cipher mode**, exists. It works by encrypting an initialization vector, using a key to get an output block. The output block is then encrypted, using the key to get a second output block

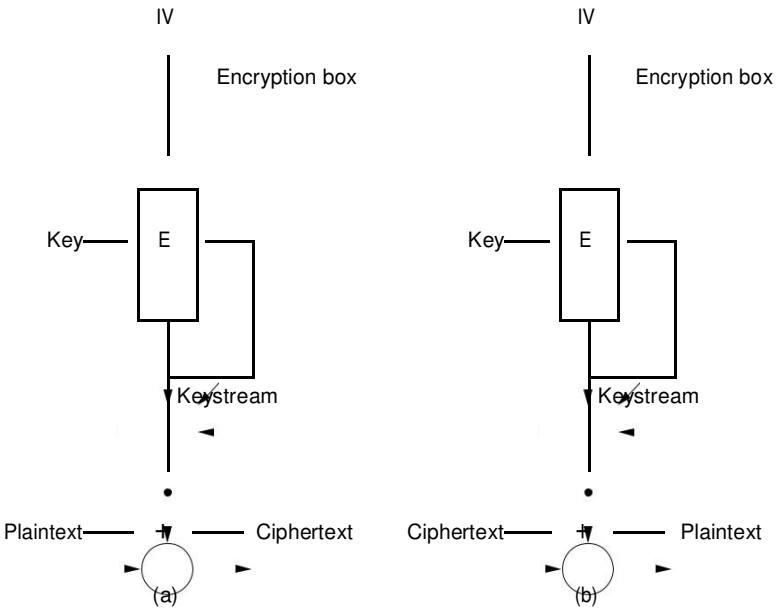


Figure 1A stream cipher. (a) Encryption. (b) Decryption.

This block is then encrypted to get a third block, and so on. The (arbitrarily large) sequence of output blocks, called the **keystream**, is treated like a one-time pad and XORed with the plaintext to get the ciphertext.Note that the IV is used only on the first step. After that, the output is encrypted. Also note that the keystream is independent of the data, so it can be computed in advance, if need be, and is completely insensitive to transmission errors.

Decryption occurs by generating the same keystream at the receiving side. Since the keystream depends only on the IV and the key, it is not affected by transmission errors in the ciphertext. Thus, a 1-bit error in the transmitted cipher-text generates only a 1-bit error in the decrypted plaintext.

Counter Mode

One problem that all the modes except electronic code book mode have is that random access to encrypted data is impossible. For example, suppose a file is transmitted over a network and then stored on disk in encrypted form. This might be a reasonable way to operate if the receiving computer is a notebook computer that might be stolen. Storing all critical files in encrypted form greatly reduces the damage due to secret information leaking out in the event that the computer falls into the wrong hands.

However, disk files are often accessed in non sequential order, especially files in databases. With a file encrypted using cipher block chaining, accessing a ran-dom block requires first decrypting all the blocks ahead of it, an expensive proposition.

Other Ciphers

AES (Rijndael) and DES are the best-known symmetric-key cryptographic algorithms, and the standard industry choices, if only for liability reasons. (No one will blame you if you use AES in your product and AES is cracked, but they will certainly blame you if you use a nonstandard cipher and it is later broken.) How-ever, it is worth mentioning that numerous other symmetric-key ciphers have been devised. Some of these are embedded inside various products.

Cipher	Author	Key length	Comments
DES	IBM	56 bits	Too weak to use now
RC4	Ronald Rivest	1–2048 bits	Caution: some keys are weak
RC5	Ronald Rivest	128–256 bits	Good, but patented
AES (Rijndael)	Daemen and Rijmen	128–256 bits	Best choice
Serpent	Anderson, Biham, Knudsen	128–256 bits	Very strong
Triple DES	IBM	168 bits	Good, but getting old
Twofish	Bruce Schneier	128–256 bits	Very strong; widely used

Figure 1. Some common symmetric-key cryptographic algorithms.

Cryptanalysis

The first development is **differential cryptanalysis** (Biham and Shamir, 1997). This technique can be used to attack any block cipher. It works by beginning with a pair of plaintext blocks differing in only a small number of bits and watching carefully what happens on each internal iteration as the encryption proceeds. In many cases, some bit pat-terns are more common than others, which can lead to probabilistic attacks.

The second development worth noting is **linear cryptanalysis** (Matsui, 1994). It can break DES with only 2^{43} known plaintexts. It works by XORing certain bits in the plaintext and ciphertext together and examining the result. When done repeatedly, half the bits should be 0s and half should be 1s. The third development is using analysis of electrical power consumption to find secret keys. Computers typically use around 3 volts to represent a 1 bit and 0 volts to represent a 0 bit.

PUBLIC-KEY ALGORITHMS

Public key cryptography (PKC) is an encryption technique that uses a paired public and private key (or asymmetric key) algorithm for secure data communication. A message sender uses a recipient's public key to encrypt a message. To decrypt the sender's message, only the recipient's private key may be used.

RSA

The only catch is that we need to find algorithms that indeed satisfy all three requirements. Due to the potential advantages of public-key cryptography, many researchers are hard at work.

Its major disadvantage is that it requires keys of at least 1024 bits for good security (versus 128 bits for symmetric-key algorithms), which makes it quite slow.

The RSA method is based on some principles from number theory. We will now summarize how to use the method; for details, consult the paper.

1. Choose two large primes, p and q (typically 1024 bits).
2. Compute $n = p \times q$ and $z = (p - 1) \times (q - 1)$.
3. Choose a number relatively prime to z and call it d.
4. Find e such that $e \times d = 1 \bmod z$.

With these parameters computed in advance, we are ready to begin encryption. Divide the plaintext (regarded as a bit string) into blocks, so that each plaintext message, P, falls in the interval $0 \le P < n$. Do that by grouping the plaintext into blocks of k bits, where k is the largest integer for which $2^k < n$ is true.

5.

To encrypt a message, P, compute $C = P^e \pmod{n}$. To decrypt C, compute $P = C^d \pmod{n}$. It can be proven that for all P in the specified range, the en-cryption and decryption functions are inverses. To perform the encryption, you need e and n. To perform the decryption, you need d and n. Therefore, the public key consists of the pair (e, n) and the private key consists of (d, n).

Other Public-Key Algorithms

Although RSA is widely used, it is by no means the only public-key algorithm known. The first public-key algorithm was the knapsack algorithm (Merkle and Hellman, 1978). The idea here is that someone owns a large number of objects, each with a different weight. The owner encodes the message by secretly

selecting a subset of the objects and placing them in the knapsack. The total weight of the objects in the knapsack is made public, as is the list of all possible objects and their corresponding weights. The list of objects in the knapsack is kept secret. With certain additional restrictions, the problem of figuring out a possible list of objects with the given weight was thought to be computationally infeasible and formed the basis of the public-key algorithm..

Other public-key schemes are based on the difficulty of computing discrete logarithms. Algorithms that use this principle have been invented by El Gamal (1985) and Schnorr (1991).

A few other schemes exist, such as those based on elliptic curves (Menezes and Vanstone, 1993), but the two major categories are those based on the difficulty of factoring large numbers and computing discrete logarithms modulo a large prime. T