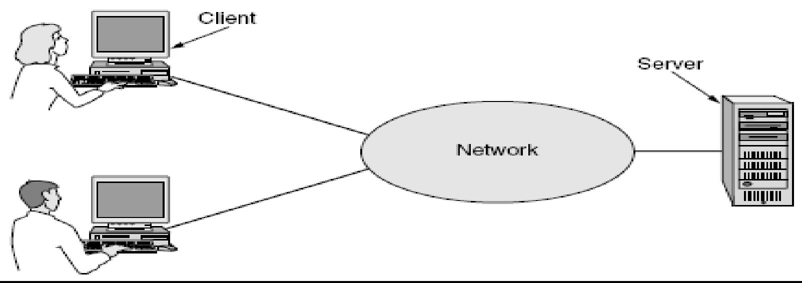
UNIT-1

The term ‘‘computer network’’ to mean a collection of autonomous computers interconnected by a single technology. Two computers are said to be interconnected if they are able to exchange information.

The key distinction is that in a distributed system, a collection of independent computers appears to its users as a single coherent system. Often a layer of software on top of the operating system, called **middleware**, is responsible for implementing this model. A well-known example of a distributed system is the **World Wide Web**.

USES OF COMPUTER NETWORKS

1. **Business Applications:** 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 or the user. Networks called **VPNs** (**Virtual Private Networks**) may be used to join the individual networks at different sites into one extended network. In this model, the data are stored on powerful computers called **servers**. Often these are centrally housed and maintained by a system administrator. In contrast, the employees have simpler machines, called **clients**, on their desks, with which they access remote data, for example, to include in spreadsheets they are constructing.



2. **Home Applications:** Internet access provides home users with **connectivity** to remote computers. Much of this information is accessed using the client-server model, but there is different, popular model for accessing information that goes by the name of **peer-to-peer** communication. In this form, individuals who form a loose group can communicate with others in the group. Every person can, in principle, communicate with one or more other people; there is no fixed division into clients and servers.

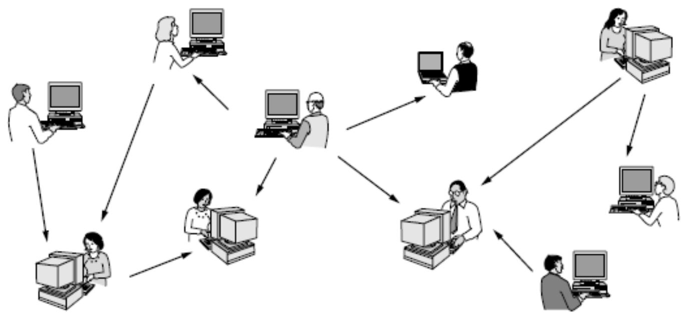


Figure 1-3. In a peer-to-peer system there are no fixed clients and servers.

Our fourth category is entertainment. Our last category is **ubiquitous computing**, in which computing is embedded into everyday life,

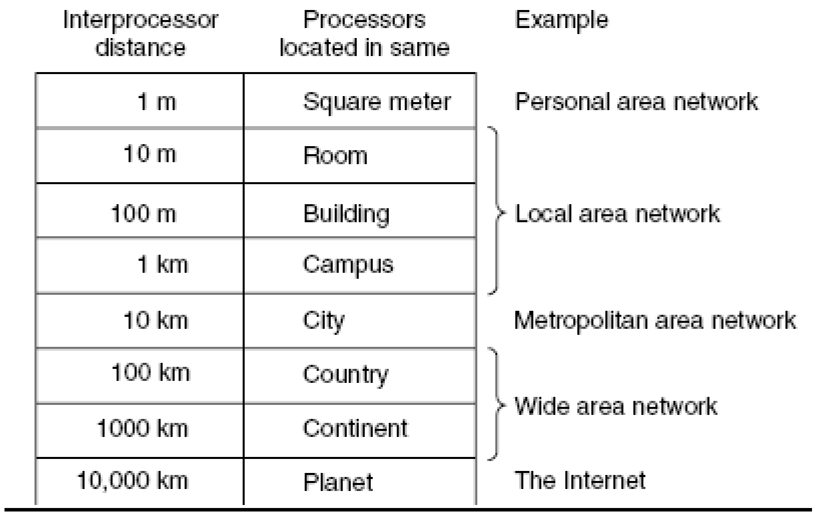
3. **Mobile Users:** **Connectivity** to the Internet enables many of these mobile uses. Since having a wired connection is impossible in cars, boats, and airplanes, there is a lot of interest in wireless networks. Cellular networks operated by the telephone companies are one familiar kind of wireless network that blankets us with coverage for mobile phones. Wireless **hotspots** based on the 802.11 standard are another kind of wireless network for mobile computers.
4. **Social Issues:** **Phishing** messages masquerade as originating from a trustworthy party, documents. It can be difficult to prevent computers from impersonating people on the Internet. This problem has led to the development of **CAPTCHAs**, in which a computer asks a person to solve a short recognition task.

NETWORK HARDWARE

Broadly speaking, there are two types of transmission technology that are in widespread use: **broadcast** links and **point-to-point** links. Point-to-point transmission with exactly one sender and exactly one receiver is sometimes called **unicasting**.

In contrast, on a **broadcast network**, the communication channel is shared by all the machines on the network; A wireless network is a common example of a broadcast link, with communication shared over a coverage region that depends on the wireless channel and the transmitting machine.

Broadcast systems usually also allow the possibility of addressing a packet to *all* destinations by using a special code in the address field. When a packet with this code is transmitted, it is received and processed by every machine on the network. This mode of operation is called **broadcasting**. Some broadcast systems also support transmission to a subset of the machines, which known as **multicasting**. An alternative criterion for classifying networks is by scale. Distance is important as a classification metric because different technologies are used at different scales.



Interprocessor distance	Processors located in same	Example
1 m	Square meter	Personal area network
10 m	Room	Local area network
100 m	Building	Local area network
1 km	Campus	Local area network
10 km	City	Metropolitan area network
100 km	Country	Wide area network
1000 km	Continent	Wide area network
10,000 km	Planet	The Internet

Classification of interconnected processors by scale.

1. <u>Personal Area Networks:</u> PANs (**Personal Area Networks**) devices communicate over the range of a person. A common example is a wireless network that connects a computer with its peripherals. Another example is Bluetooth which uses master-slave paradigm.

2. <u>Local Area Networks:</u> A LAN is a privately owned network that operates within and nearby a single building like a home, office or factory. LANs are widely used to connect personal computers and consumer electronics to let them share resources and exchange information.

3. <u>Metropolitan Area Networks:</u> A **MAN** (**Metropolitan Area Network**) covers a city. The best-known examples of MANs are the cable television networks available in many cities.

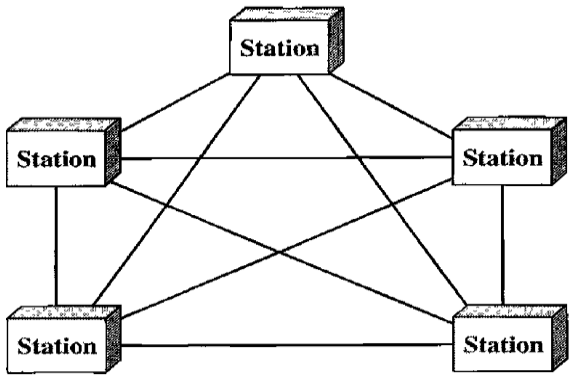
4. <u>Wide Area Networks:</u> A **WAN** (**Wide Area Network**) spans a large geographical area, often a country or continent. In most WANs, the subnet consists of two distinct components: transmission lines and switching elements. **Transmission lines** move bits between machines. They can be made of copper wire, optical fiber, or even radio links. **Switching elements**, or just **switches**, are specialized computers that connect two or more transmission lines.

5. <u>Internetworks:</u> A collection of interconnected networks is called an **internetwork** or **internet**. The general name for a machine that makes a connection between two or more networks and provides the necessary translation, both in terms of hardware and software, is a **gateway**. Gateways are distinguished by the layer at which they operate in the protocol hierarchy.

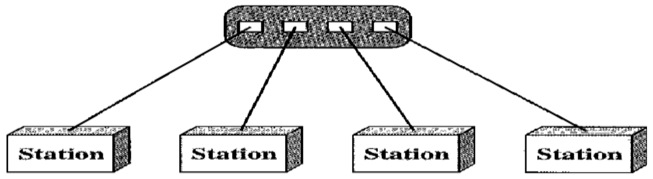
NETWORK TOPOLOGIES:

The term *physical topology* refers to the way in which a network is laid out physically.1or more devices connect to a link; two or more links form a topology. The topology of a network is the geometric representation of the relationship of all the links and linking devices (usually called nodes) to one another. There are four basic topologies possible: mesh, star, bus, and ring.

Mesh In a mesh topology, every device has a dedicated point-to-point link to every other device. The term dedicated means that the link carries traffic only between the two devices it connects. we need *n(n* -1) /2 duplex-mode links.

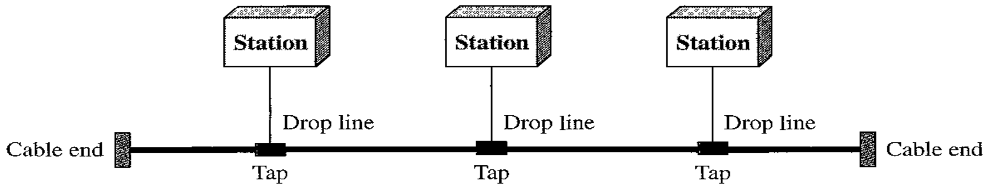


Star Topology In a star topology, each device has a dedicated point-to-point link only to a central controller, usually called a hub. The devices are not directly linked to one another. A star topology is less expensive than a mesh topology. In a star, each device needs only one link and one I/O port to connect it to any number of others. Other advantages include robustness. If one link fails, only that link is affected. All other links remain active.



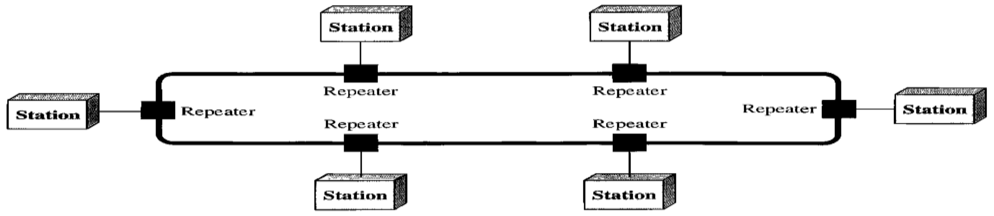
One big disadvantage of a star topology is the dependency of the whole topology on one single point, the hub. If the hub goes down, the whole system is dead.

A **bus topology,** on the other hand, is multipoint. One long cable acts as a **backbone** to link all the devices in a network



Nodes are connected to the bus cable by drop lines and taps. A drop line is a connection running between the device and the main cable. A tap is a connector that either splices into the main cable or punctures the sheathing of a cable to create a contact with the metallic core.

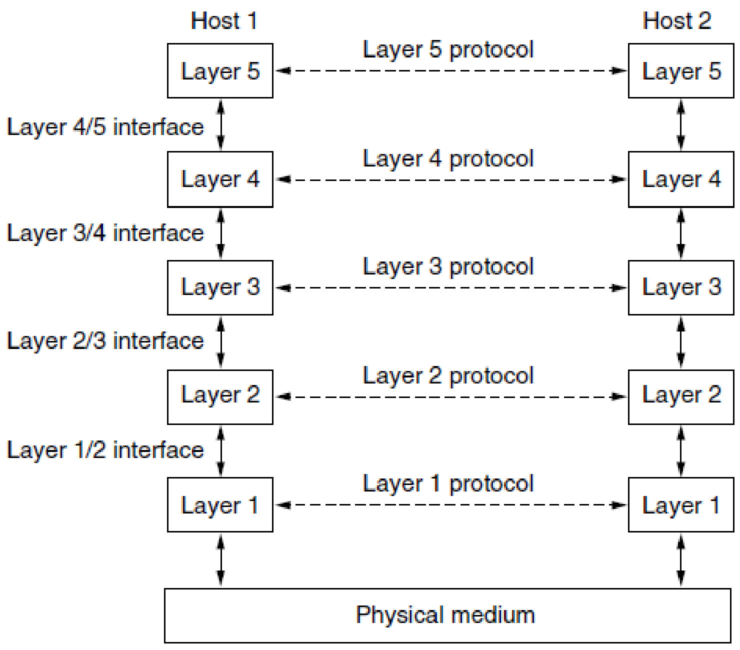
Ring Topology In a ring topology, each device has a dedicated point-to-point connection with only the two devices on either side of it. A signal is passed along the ring in one direction, from device to device, until it reaches its destination. Each device in the ring incorporates a repeater. When a device receives a signal intended for another device, its repeater regenerates the bits and passes them along.



NETWORK SOFTWARE:

1. 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 while shielding those layers from the details of how the offered services are actually implemented.

A five-layer network is illustrated below.. The entities comprising the corresponding layers on different machines are called **peers**. The peers may be software processes, hardware devices, or even human beings. In other words, it is the peers that communicate by using the protocol to talk to each other.



Between each pair of adjacent layers is an **interface**. The interface defines which primitive operations and services the lower layer makes available to the upper one.

A set of layers and protocols is called network **architecture**. The specification of architecture must contain enough information to allow an implementer to write the program or build the hardware for each layer so that it will correctly obey the appropriate protocol. A list of the protocols used by a certain system, one protocol per layer, is called a **protocol stack**.

2 Design Issues for the Layers:

Reliability is the design issue of making a network that operates correctly even though it is made up of a collection of components that are themselves unreliable. One mechanism for finding errors in received information uses codes for **error detection**. Information that is incorrectly received can then be retransmitted until it is received correctly. More powerful codes allow for **error correction**, where the correct message is recovered from the possibly incorrect bits that were originally received.

A second design issue concerns the evolution of the network. Over time, networks grow larger and new designs emerge that need to be connected to the existing network. The key structuring mechanism used to support change by dividing the overall problem and hiding implementation details: **protocol layering**. Since there are many computers on the network, every layer needs a mechanism for identifying the senders and receivers that are involved in a particular message. This mechanism is called **addressing** or **naming**, in the low and high layers, respectively. Designs that continue to work well when the network gets large are said to be **scalable**.

A third design issue is resource allocation. Networks provide a service to hosts from their underlying resources, such as the capacity of transmission lines. To do this well, they need mechanisms that divide their resources so that one host does not interfere with another too much. Many designs share network bandwidth dynamically, according to the short term needs of hosts, rather than by giving each host a fixed fraction of the bandwidth that it may or may not use. This design is called **statistical multiplexing**, meaning sharing based on the statistics of demand.

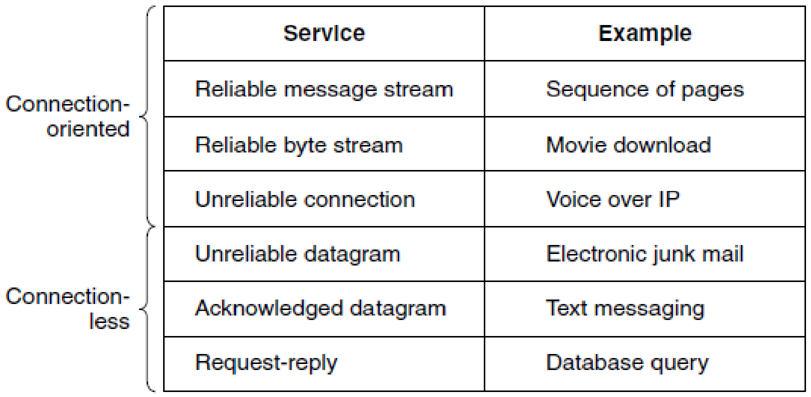
The last major design issue is to secure the network by defending it against different kinds of threats. One of the threats we have mentioned previously is that of eavesdropping on communications. Mechanisms that provide **confidentiality** defend against this threat, and they are used in multiple layers. Mechanisms for **authentication** prevent someone from impersonating someone else. Other mechanisms for **integrity** prevent surreptitious changes to messages, such as altering. All of these designs are based on cryptography.

3. Connection-Oriented Versus Connectionless Service:

Connection-oriented service is modeled after the telephone system. Similarly, to use a connection-oriented network service, the service user first establishes a connection, uses the connection, and then releases the connection. A **circuit** is another name for a connection with associated resources, such as a fixed bandwidth.

In contrast to connection-oriented service, **connectionless** service is modeled after the postal system. Each message (letter) carries the full destination address, and each one is routed through the intermediate nodes inside the system independent of all the subsequent messages..

When the intermediate nodes receive a message in full before sending it on to the next node, this is called **store-and-forward switching**. The alternative, in which the onward transmission of a message at a node starts before it is completely received by the node, is called **cut-through switching**.



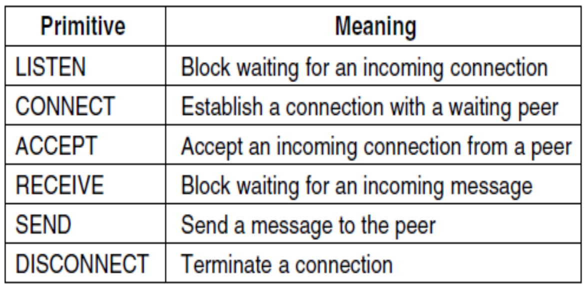
	Service	Example
Connection-oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Movie download
	Unreliable connection	Voice over IP
Connection-less	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Text messaging
	Request-reply	Database query

Six different types of service

4. Service Primitives:

A service is formally specified by a set of **primitives** (operations) available to user processes 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.

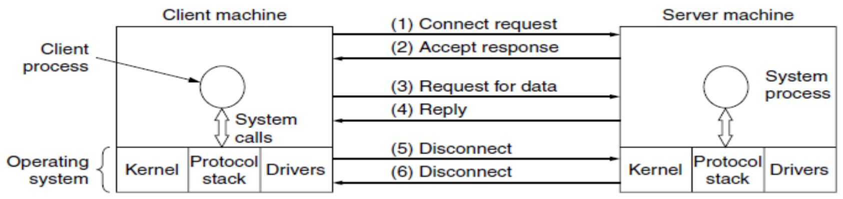
The set of primitives available depends on the nature of the service being provided. The primitives for connection-oriented service are different from those of connectionless service.



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

<u>Six service primitives that provide a simple connection-oriented service</u>

These primitives might be used for a request-reply interaction in a client-server environment.



A simple client-server interaction using acknowledged datagrams

4 <u>REFERENCE MODELS:</u>

1. 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 was revised in 1995 (Day, 1995). 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. We will just call it the **OSI model** for short.

The OSI model has seven layers. The principles that were applied to arrive at the seven layers can be briefly summarized as follows:

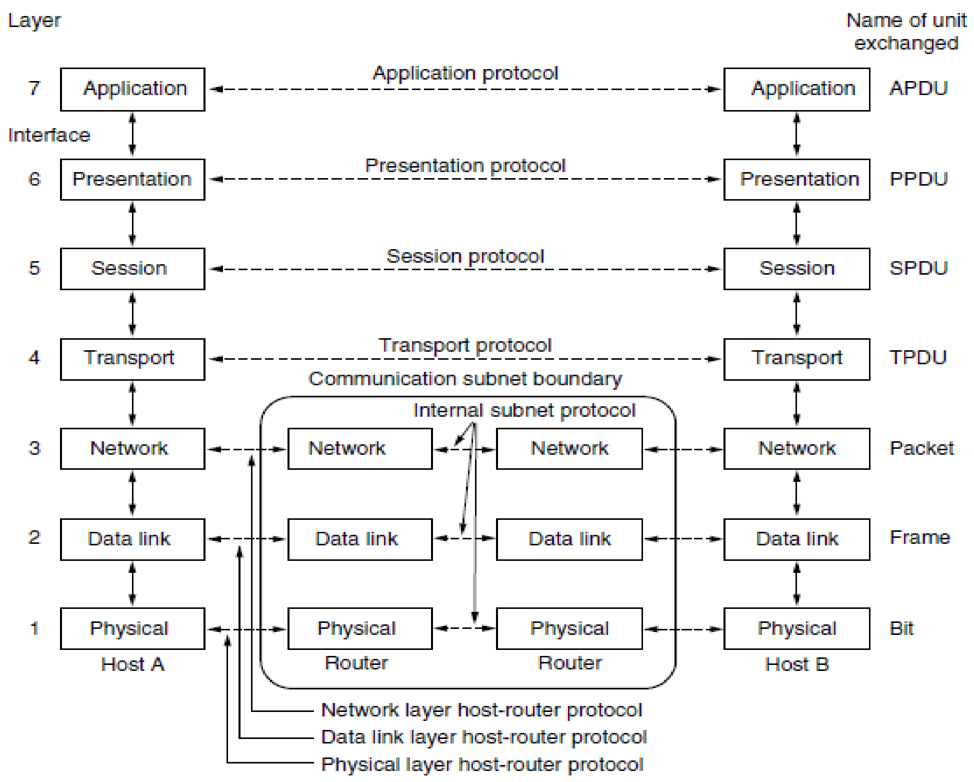
1. A layer should be created where a different abstraction is needed.

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.



Physical Layer

The physical layer coordinates the functions required to carry a bit stream over a physical medium. It deals with the mechanical and electrical specifications of the interface and transmission medium.

The physical layer is also concerned with the following:

- Physical characteristics of interfaces and medium
- Representation of bits
- Data rate
- Synchronization of bits
- Line configuration
- Physical topology
- Transmission mode

Data Link Layer

The data link layer transforms the physical layer, a raw transmission facility, to a reliable link. It makes the physical layer appear error-free to the upper layer.

Other responsibilities of the data link layer include the following:

- Framing
- Physical addressing
- Flow control
- Error control
- Access control

Network Layer

The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links). Other responsibilities of the network layer include the following:

- Logical addressing
- Routing

Transport Layer

The transport layer is responsible for process-to-process delivery of the entire message. A process is an application program running on a host. Other responsibilities of the transport layer include the following:

- Service-point addressing
- Segmentation and reassembly
- Connection control
- Flow control
- Error control

Session Layer

The services provided by the first three layers (physical, data link, and network) are not sufficient for some processes. The session layer is the network dialog controller. It establishes, maintains, and synchronizes the interaction among communicating systems.

Specific responsibilities of the session layer include the following:

- Dialog control
- Synchronization

Presentation Layer

The presentation layer is concerned with the syntax and semantics of the information exchanged between two systems.

Specific responsibilities of the presentation layer include the following:

- Translation
- Encryption
- Compression

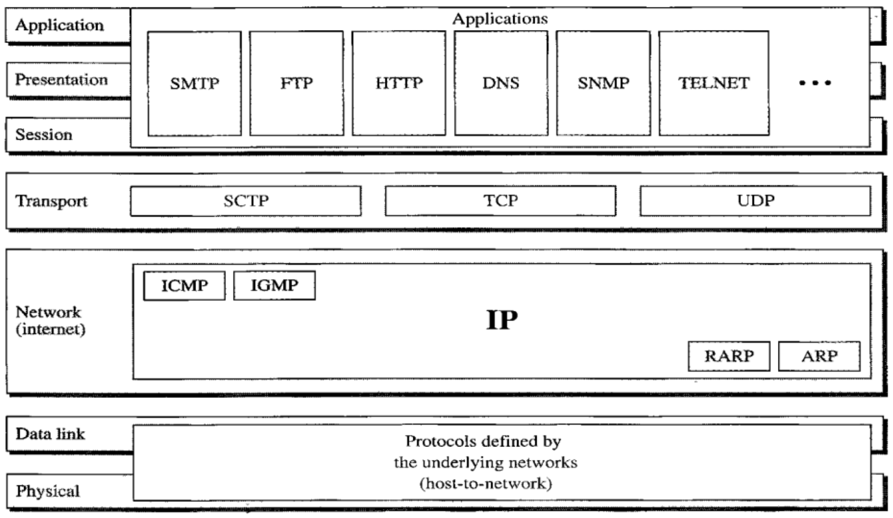
Application Layer

The application layer enables the user, whether human or software, to access the network. It provides user interfaces and support for services such as electronic mail, remote file access and transfer, shared database management, and other types of distributed information services.

Specific services provided by the application layer include the following:

- Network virtual terminal
- File transfer, access, and management
- Mail services
- Directory services

2.TCP/IP PROTOCOL SUITE:



Physical and Data Link Layers

At the physical and data link layers, *TCP/IP* does not define any specific protocol. It supports all the standard and proprietary protocols. A network in a *TCP/IP* internetwork can be a local-area network or a wide-area network.

Network Layer

At the network layer *TCP/IP* supports the Internetworking Protocol. IP, in turn, uses four supporting protocols: ARP, RARP, ICMP, and IGMP.

Transport Layer

Traditionally the transport layer was represented in *TCP/IP* by two protocols: TCP and UDP. UDP and TCP are transport level protocols responsible for delivery of a message from a process to another process.

Application Layer

The *application layer* in TCP/IP is equivalent to the combined session, presentation, and application layers in the OSI model

3. A Comparison of the OSI and TCP/IP Reference Models:

The OSI and TCP/IP reference models have much in common. Both are based on the concept of a stack of independent protocols. Three concepts are central to the OSI model:

1. Services.

2. Interfaces.

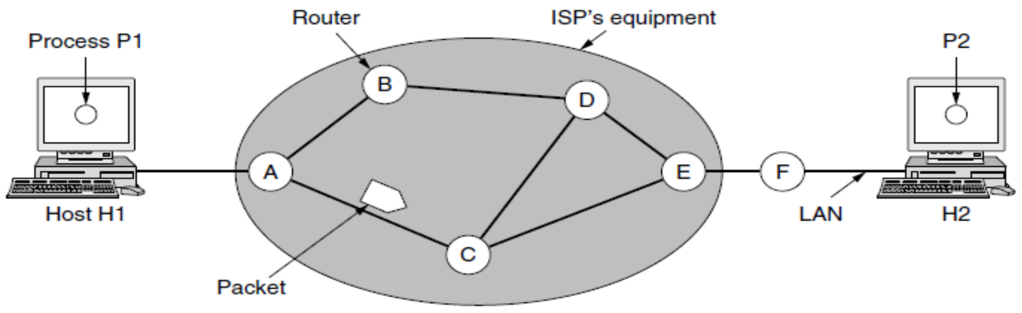
3. Protocols.

NETWORK LAYER

NETWORK LAYER DESIGN ISSUES:

1. Store-and-Forward Packet Switching:

The major components of the network are the ISP's equipment (routers connected by transmission lines), and the customers' equipment. Host *H1* is directly connected to one of the ISP's routers, *A*, perhaps as a home computer that is plugged into a DSL modem. In contrast, *H2* is on a LAN, which might be an office Ethernet, with a router, *F*, owned and operated by the customer. This router has a leased line to the ISP's equipment.



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 ISP. The packet is stored there until it has fully arrived and the link has finished its processing by verifying the checksum. 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.

2. Services Provided to the Transport Layer

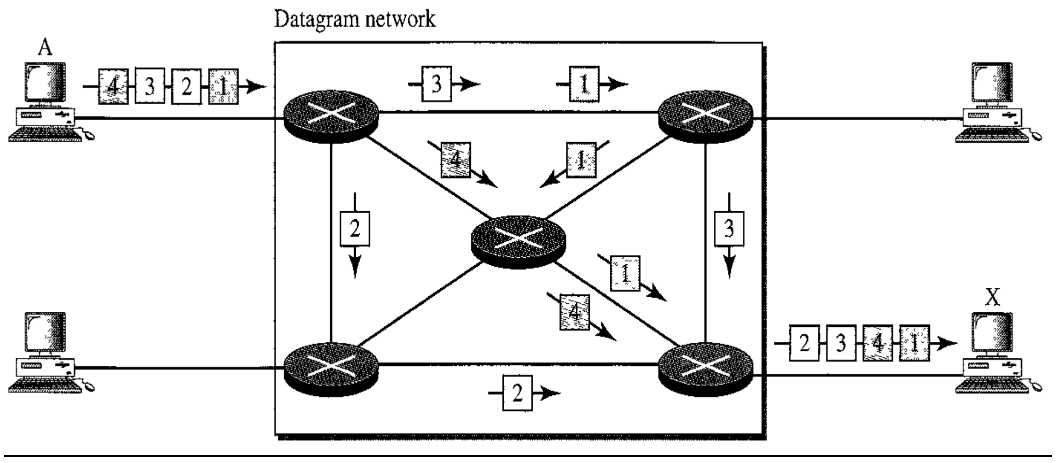
The network layer provides services to the transport layer at the network layer/transport layer interface. An important question is precisely what kind of services the network layer provides to the transport layer. The services need to be carefully designed with the following goals in mind:

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

3. Implementation of Connectionless Service:

Two different organizations are possible, depending on the type of service offered. If connectionless service is offered, packets are injected into the network individually and routed independently of each other. No advance setup is needed. In this context, the packets are frequently called **datagrams** and the network is called a **datagram network**. Let us now see how a datagram network works. Suppose that the process *P1* in Fig. 5-2 has a long message for *P2*. It hands the message to the transport layer, with instructions to deliver it to process *P2* on host *H2*. The transport layer code runs on *H1*, typically within the operating system. It prepends a transport header to the front of the message and hands the result to the network layer, probably just another procedure within the operating system.

Datagram switching is normally done at the network layer.



The datagram networks are sometimes referred to as connectionless networks. The term *connectionless* here means that the switch (packet switch) does not keep information about the connection state. There are no setup or teardown phases. Each packet is treated the same by a switch regardless of its source or destination.

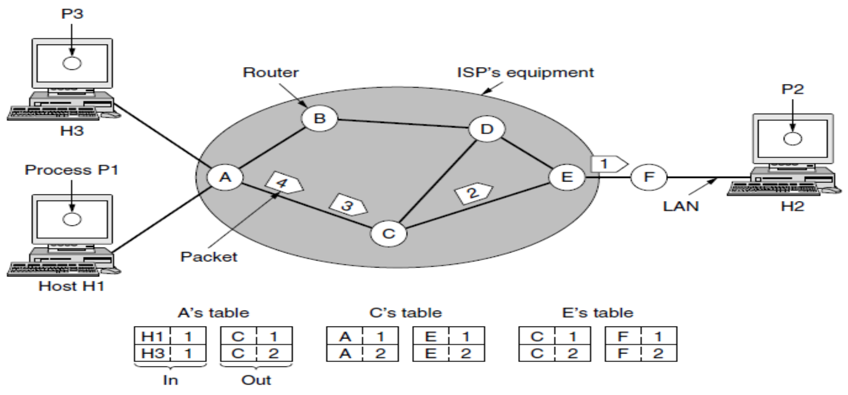
Routing Table:

In this type of network, each switch has a routing table which is based on the destination address. The routing tables are dynamic and are updated periodically. The destination addresses and the corresponding forwarding output ports are recorded in the tables. This is different from the table of a circuit switched network in which each entry is created when the setup phase is completed and deleted when the teardown phase is over. The algorithm that manages the tables and makes the routing decisions is called the **routing algorithm**.

4. Implementation of Connection-Oriented Service

For connection-oriented service, we need a virtual-circuit network. The idea behind virtual circuits is to avoid having to choose a new route for every packet sent. Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers. That route is used for all traffic flowing over the connection. When the connection is released, the virtual circuit is also terminated. With connection-oriented service, each packet carries an identifier telling which virtual circuit it belongs to.

As an example, consider the situation shown. Here, host *H1* has established connection 1 with host *H2*. This connection 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.



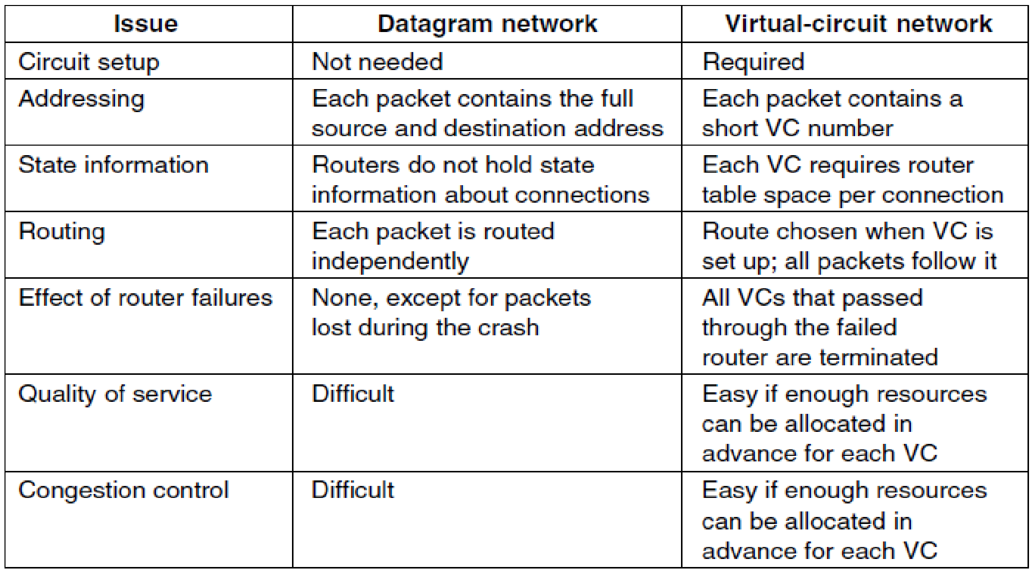
Routing within a virtual-circuit network

Let us consider what happens if *H3* also wants to establish a connection to *H2*. It chooses connection identifier 1 and tells the network to establish the virtual circuit. This leads to the second row in the tables. Note that we have a conflict here because although *A* can easily distinguish connection 1 packets from *H1* from connection 1 packets from *H3*, *C* cannot do this. For this reason, *A* assigns a different connection identifier to the outgoing traffic for the second connection. Avoiding conflicts of this kind is why routers need the ability to replace connection identifiers in outgoing packets.

In some contexts, this process is called **label switching**. An example of a connection-oriented network service is **MPLS** (**MultiProtocol Label Switching**). It is used within ISP networks in the Internet, with IP packets wrapped in an MPLS header having a 20-bit connection identifier or label. MPLS is often hidden from customers, with the ISP establishing long-term connections for large amounts of traffic, but it is increasingly being used to help when quality of service is important but also with other ISP traffic management tasks.

5. Comparison of Virtual-Circuit and Datagram Networks

Both virtual circuits and datagrams have their supporters and their detractors. The major issues are listed.



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
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

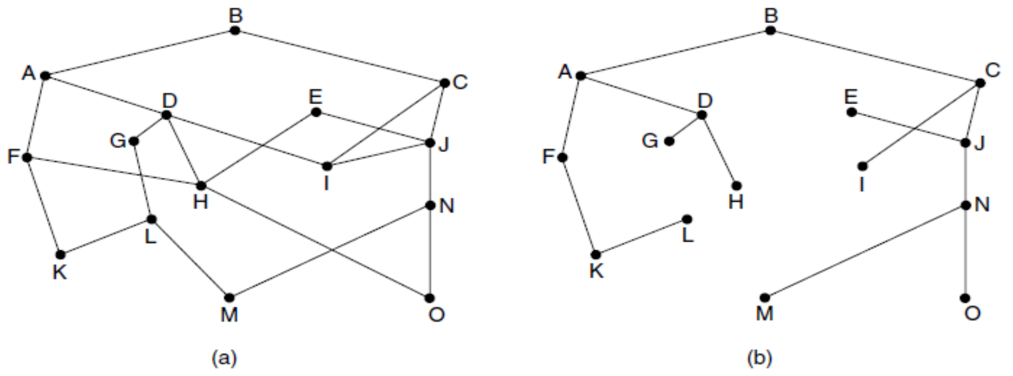
ROUTING ALGORITHMS:

Routing algorithms can be grouped into two major classes: non-adaptive and adaptive. **Non-adaptive algorithms** do not base their routing decisions on any measurements or estimates of the current topology and traffic. This procedure is called **static routing. Adaptive algorithms**, in contrast, change their routing decisions to reflect changes in the topology, and sometimes changes in the traffic as well. This procedure is called **dynamic routing**

1. The Optimality Principle:

It states that if router *J* is on the optimal path from router *I* to router *K*, then the optimal path from *J* to *K* also falls along the same route.

As a direct consequence of the optimality principle, the set of optimal routes from all sources to a given destination form a tree rooted at the destination. Such a tree is called a **sink tree** where the distance metric is the number of hops. The goal of all routing algorithms is to discover and use the sink trees for all routers.

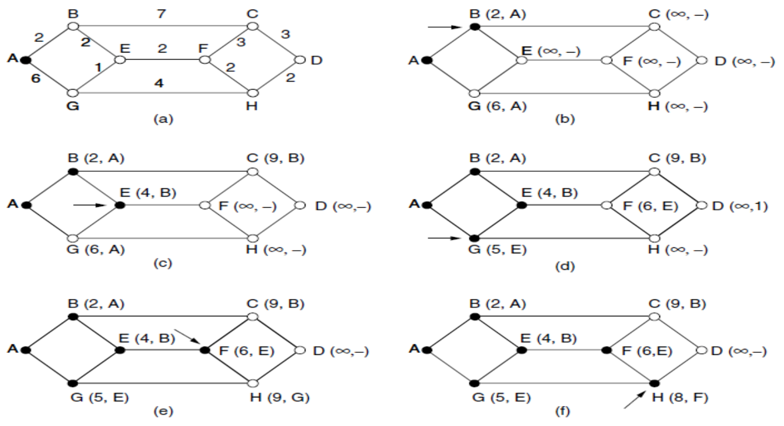


(a) A network. (b) A sink tree for router B

If we allow all of the possible paths to be chosen, the tree becomes a more general structure called a **DAG** (**Directed Acyclic Graph**). DAGs have no loops.

2. Shortest Path Algorithm

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. One way of measuring path length is the number of hops. Using this metric, the paths *ABC* and *ABE* in Figure are equally long. Another metric is the geographic distance in kilometers, in which case *ABC* is clearly much longer than *ABE* (assuming the figure is drawn to scale).



The first six steps used in computing the shortest path from A to D. The arrows indicate the working node.

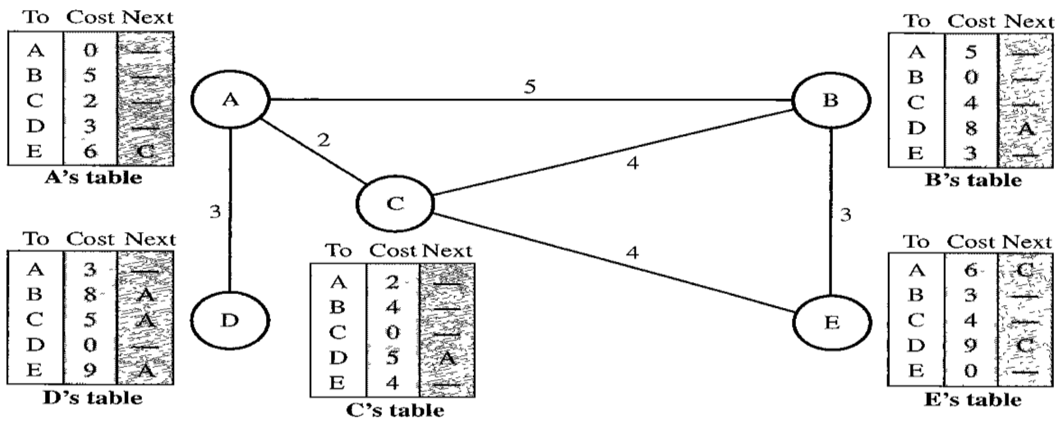
3. Flooding

When a routing algorithm is implemented, each router must make decisions based on local knowledge, not the complete picture of the network. 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 is not practical for sending most packets, but it does have some important uses. First, it ensures that a packet is delivered to every node in the network. Second, flooding is tremendously robust. Even if large numbers of routers are blown to bits flooding will find a path if one exists, to get a packet to its destination.

4. Distance Vector Routing

In **distance vector routing**, the least-cost route between any two nodes is the route with minimum distance. In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node. The table at each node also guides the packets to the desired node by showing the next stop in the route (next-hop routing).



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

A's table

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

B's table

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

D's table

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

C's table

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

E's table

In distance vector routing, each node shares its routing table with its immediate neighbors periodically and when there is a change.

Updating

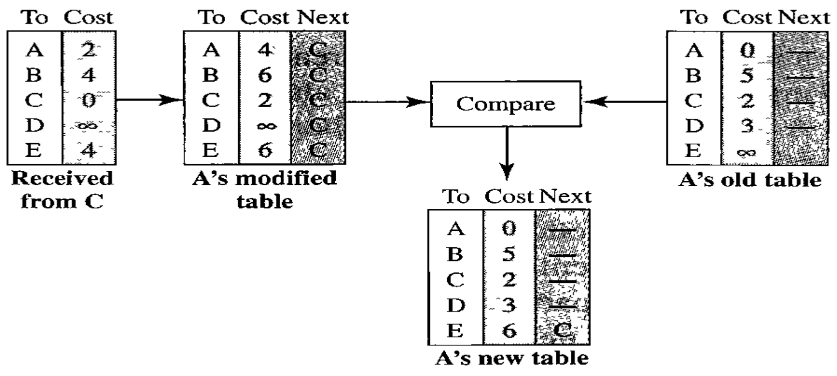
When a node receives a two-column table from a neighbor, it needs to update its routing table. Updating takes three steps:

1. The receiving node needs to add the cost between itself and the sending node to each value in the second column. The logic is clear. If node C claims that its distance to a destination is *x* mi, and the distance between A and C is *y* mi, then the distance between A and that destination, via C, is *x* + *y* mi.

2. The receiving node needs to add the name of the sending node to each row as the third column if the receiving node uses information from any row. The sending node is the next node in the route.

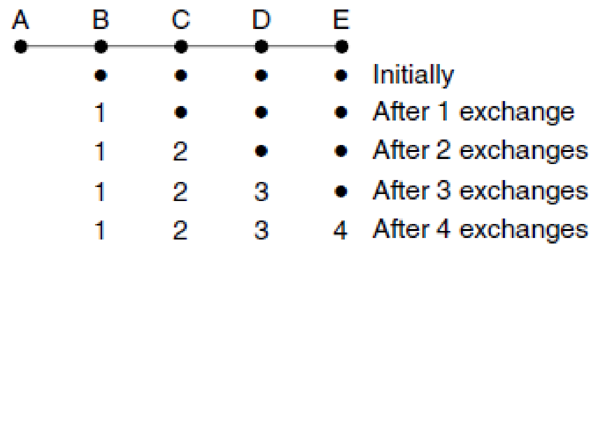
3. The receiving node needs to compare each row of its old table with the corresponding row of the modified version of the received table.

- **a.** If the next-node entry is different, the receiving node chooses the row with the smaller cost. If there is a tie, the old one is kept.
- **b.** If the next-node entry is the same, the receiving node chooses the new row. For example, suppose node C has previously advertised a route to node X with distance Suppose that now there is no path between C and X; node C now advertises this route with a distance of infinity. Node A must not ignore this value even though its old entry is smaller. The old route does not exist anymore. The new route has a distance of infinity.

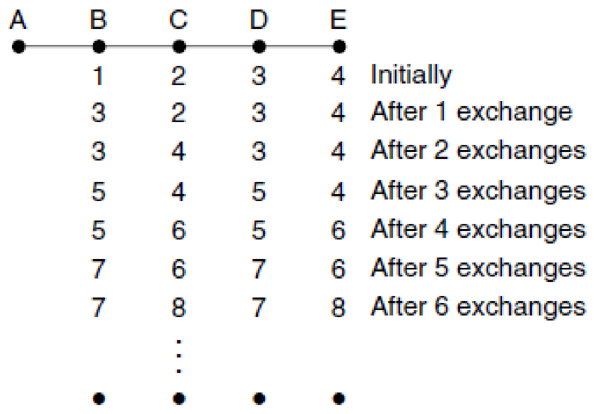


The Count-to-Infinity Problem

The settling of routes to best paths across the network is called **convergence**. Distance vector routing is useful as a simple technique by which routers can collectively compute shortest paths, but it has a serious drawback in practice.


A B C D E
• • • • Initially
1 • • • After 1 exchange
1 2 • • After 2 exchanges
1 2 3 • After 3 exchanges
1 2 3 4 After 4 exchanges

(a)


A B C D E
1 2 3 4 Initially
3 2 3 4 After 1 exchange
3 4 3 4 After 2 exchanges
5 4 5 4 After 3 exchanges
5 6 5 6 After 4 exchanges
7 6 7 6 After 5 exchanges
7 8 7 8 After 6 exchanges
• • • •

(b)

When *A* comes up, the other routers learn about it via the vector exchanges. For simplicity, we will assume that there is a gigantic going somewhere that is struck periodically to initiate a vector exchange at all routers simultaneously. At the time of the first exchange, *B* learns that its left-hand neighbor has zero delay to *A*. *B* now makes an entry in its routing table indicating that *A* is one hop away to the left. All the other routers still think that *A* is down. At this point, the routing table entries for *A* are as shown in the second row of figure (a). On the next exchange, *C* learns that *B* has a path of length 1 to *A*, so it updates its routing table to indicate a path of length 2, but *D* and *E* do not hear the good news until later. Clearly, the good news is spreading at the rate of one hop per exchange. In a network whose longest path is of length *N* hops, within *N* exchanges everyone will know about newly revived links and routers.

Now let us consider the situation of figure(b), in which all the links and routers are initially up. Routers *B*, *C*, *D*, and *E* have distances to *A* of 1, 2, 3, and 4 hops, respectively. Suddenly, either *A* goes down or the link between *A* and *B* is cut (which is effectively the same thing from *B*'s point of view).

At the first packet exchange, *B* does not hear anything from *A*. Fortunately, *C* says ''Do not worry; I have a path to *A* of length 2.'' Little does *B* suspect that *C*'s path runs through *B* itself. For all *B* knows, *C* might have ten links all with separate paths to *A* of length 2. As a result, *B* thinks it can reach *A* via *C*, with a path length of 3. *D* and *E* do not update their entries for *A* on the first exchange.

On the second exchange, *C* notices that each of its neighbors claims to have a path to *A* of length 3. It picks one of them at random and makes its new distance to *A* 4, as shown in the third row of figure(b). Subsequent exchanges produce the history shown in the rest of figure(b).

From this figure, it should be clear why bad news travels slowly: no router ever has a value more than one higher than the minimum of all its neighbors. Gradually, all routers work their way up to infinity, but the number of exchanges required depends on the numerical value used for infinity. For this reason, it is wise to set infinity to the longest path plus 1. This problem is known as the **count-to-infinity** problem.

5. Link State Routing

. The idea behind link state routing is fairly simple and can be stated as five parts. Each router must do the following things to make it work:

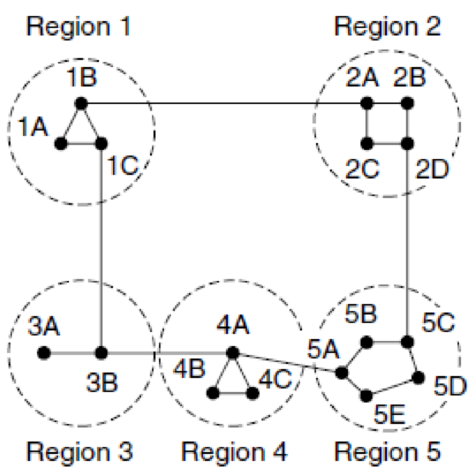
1. Discover its neighbors and learn their network addresses.
2. Set the distance or cost metric to each of its neighbors.
3. Construct a packet telling all it has just learned.
4. Send this packet to and receive packets from all other routers.
5. Compute the shortest path to every other router.

Variants of link state routing called IS-IS and OSPF are the routing algorithms that are most widely used inside large networks and the Internet today.

6. Hierarchical Routing

When hierarchical routing is used, the routers are divided into what we will call **regions**. Each router knows all the details about how to route packets to destinations within its own region but knows nothing about the internal structure of other regions. When different networks are interconnected, it is natural to regard each one as a separate region to free the routers in one network from having to know the topological structure of the other ones.

For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on, until we run out of names for aggregations.



(a)

Full table for 1A

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

(b)

Hierarchical table for 1A

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

(c)

7. Broadcast Routing

Sending a packet to all destinations simultaneously is called **broadcasting**. Various methods have been proposed for doing it.

An improvement is **multidestination routing**, in which each packet contains either a list of destinations or a bit map indicating the desired destinations. The idea for **reverse path forwarding** is elegant and remarkably simple once it has been pointed out. When a broadcast packet arrives at a router, the router checks to see if the packet arrived on the link that is normally used for sending packets *toward* the source of the broadcast. If so, there is an excellent chance that the broadcast packet itself followed the best route from the router and is therefore the first copy to arrive at the router.

Our last broadcast algorithm improves on the behavior of reverse path forwarding. It makes explicit use of the sink tree—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. This method makes excellent use of bandwidth, generating the absolute minimum number of packets necessary to do the job.

8. Multicast Routing

Optimal Routing: Shortest Path Trees

The process of optimal Interdomain routing eventually results in the finding of the *shortest path tree*. The root of the tree is the source, and the leaves are the potential destinations. The path from the root to each destination is the shortest path.

In multicast routing, each involved router needs to construct a shortest path tree for each group. When a router receives a multicast packet, the situation is different from when it receives a unicast packet. A multicast packet may have destinations in more than one network. Forwarding of a single packet to members of a group requires a shortest path tree. Two approaches have been used to solve the problem: source-based trees and group-shared trees.

9. Routing for Mobile Hosts:

Millions of people use computers while on the go, from truly mobile situations with wireless devices in moving cars, to nomadic situations in which laptop computers are used in a series of different locations. We will use the term **mobile hosts** to mean either category, as distinct from stationary hosts that never move. These mobile hosts introduce a new complication: to route a packet to a mobile host, the network first has to find it.

Each host is associated with a home network, and a home agent is associated with the home network. Each host (or more precisely, each interface) is assigned an IP address that is topologically correct for its home network. This address is said to be the host’s permanent address. From the perspective of a given host, any network other than its home network is said to be a foreign network. When a host is connected to a foreign network it is considered to be a foreign host in that network. A foreign agent is associated with each network, whose job is to keep track of the foreign hosts in that network. When a host is connected to a foreign network, it is assigned an IP address that is topologically correct for that foreign network. This IP address is said to be the host’s foreign address or care-of-address (COA).

Data Transfer: CN sends an IP packet with MN as a destination address and CN as a source address. The internet, not having information on the current location of MN, routes the packet to the router responsible for the home network of MN. This is done using the standard routing mechanisms of the internet. The HA now intercepts the packet, knowing that MN is currently not in its home network. The packet is not forwarded into the subnet as usual, but encapsulated and tunneled to the COA. A new header is put in front of the old IP header showing the COA as new destination and HA as source of the encapsulated packet (step 2).

The foreign agent now decapsulates the packet, i.e., removes the additional header, and forwards the original packet with CN as source and MN as destination to the MN (step 3). Again, for the MN mobility is not visible. It receives the packet with the same sender and receiver address as it would have done in the home network

Sending packets from the mobile node (MN) to the CN is comparatively simple. The MN sends the packet as usual with its own fixed IP address as source and CN's address as destination (step 4). The router with the FA acts as default router and forwards the packet in the same way as it would do for any other node in the foreign network. As long as CN is a fixed node the remainder is in the fixed internet as usual. If CN were also a mobile node residing in a foreign network, the same mechanisms as described in steps 1 through 3 would apply now in the other direction..

10. Routing in Ad Hoc Networks:

Routing in Mobile Ad hoc networks is an important issue as these networks do not have fixed infrastructure and routing requires distributed and cooperative actions from all nodes in the network. MANET's provide point to point routing similar to Internet routing. The major difference between routing in MANET and regular internet is the route discovery mechanism. Another major difference in the routing is the network address. In internet routing, the network address (IP address) is hierarchical containing a network ID and a computer ID on that network. In contrast, for most MANET's the network address is simply an ID of the node in the network and is not hierarchical. The routing protocol must use the entire address to decide the next hop.

Types of MANET Routing Algorithms:

1. Based on the information used to build routing tables:

• ***Shortest distance algorithms*****:** algorithms that use distance information to build routing tables.

• ***Link state algorithms*****:** algorithms that use connectivity information to build a topology graph that is used to build routing tables.

2. Based on when routing tables are built:

• ***Proactive algorithms*****:** maintain routes to destinations even if they are not needed. Some of the examples are Destination Sequenced Distance Vector (DSDV), Wireless Routing Algorithm (WRP), Global State Routing (GSR), Source-tree Adaptive Routing (STAR), Cluster-Head Gateway Switch Routing (CGSR), Topology Broadcast Reverse Path Forwarding (TBRPF), Optimized Link State Routing (OLSR) etc.

- Always maintain routes:- Little or no delay for route determination
- Consume bandwidth to keep routes up-to-date
- Maintain routes which may never be used

Advantages: low route latency, State information, QoS guarantee related to connection set-up or other real-time requirements

Disadvantages: high overhead (periodic updates) and route repair depends on update frequency

• ***Reactive algorithms*****:** maintain routes to destinations only when they are needed. Examples are Dynamic Source Routing (DSR), Ad hoc-On demand distance Vector (AODV), Temporally ordered Routing Algorithm (TORA), Associativity-Based Routing (ABR) etc

 only obtain route information when needed

 Advantages: no overhead from periodic update, scalability as long as there is only light traffic and low mobility.

 Disadvantages: high route latency, route caching can reduce latency

• ***Hybrid algorithms*****:** maintain routes to nearby nodes even if they are not needed and maintain routes to far away nodes only when needed. Example is Zone Routing Protocol (ZRP).

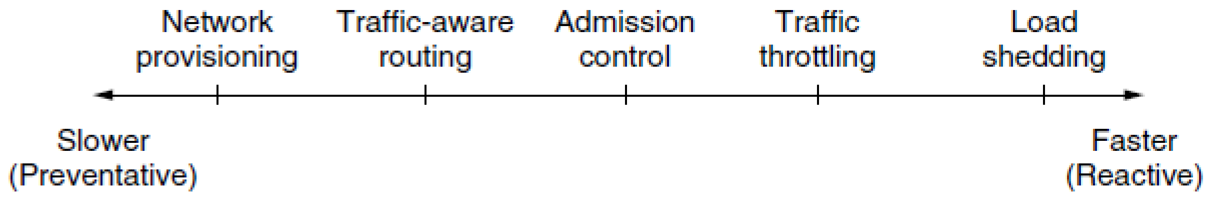
CONGESTION CONTROL ALGORITHMS

Too many packets present in the network cause packet delay and loss that degrades performance. This situation is called **congestion**. The network and transport layers share the responsibility for handling congestion. Unless the network is well designed, it may experience a **congestion collapse**, in which performance plummets as the offered load increases beyond the capacity.

It is worth pointing out the difference between congestion control and flow control, as the relationship is a very subtle one. Congestion control has to do with making sure the network is able to carry the offered traffic. It is a global issue, involving the behavior of all the hosts and routers. Flow control, in contrast, relates to the traffic between a particular sender and a particular receiver. Its job is to make sure that a fast sender cannot continually transmit data faster than the receiver is able to absorb it.

1. Approaches to Congestion Control

The presence of congestion means that the load is greater than the resources can handle. Two solutions come to mind: increase the resources or decrease the load.



Timescales of approaches to congestion control

QUALITY OF SERVICE

An easy solution to provide good quality of service is to build a network with enough capacity for whatever traffic will be thrown at it. The name for this solution is **over provisioning**. The resulting network will carry application traffic without significant loss and, assuming a decent routing scheme, will deliver packets with low latency. Performance doesn’t get any better than this. Quality of service mechanisms let a network with less capacity meet application requirements just as well at a lower cost. Moreover, overprovisioning is based on expected traffic. All bets are off if the traffic pattern changes too much. With quality of service mechanisms, the network 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.

1. 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 needs of each flow can be characterized by four primary parameters: bandwidth, delay, jitter, and loss. Together, these determine the **QoS** (**Quality of Service**) the flow requires.

To accommodate a variety of applications, networks may support different categories of QoS. An influential example comes from ATM networks, which were once part of a grand vision for networking but have since become a niche technology. They support:

1. Constant bit rate (e.g., telephony).
2. Real-time variable bit rate (e.g., compressed videoconferencing).
3. Non-real-time variable bit rate (e.g., watching a movie on demand).
4. Available bit rate (e.g., file transfer).

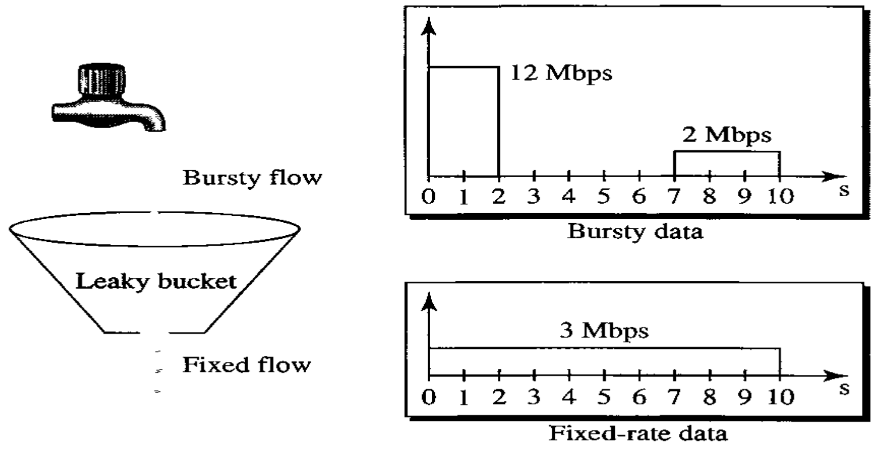
2. Traffic Shaping

Traffic shaping is a technique for regulating the average rate and burstiness 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.

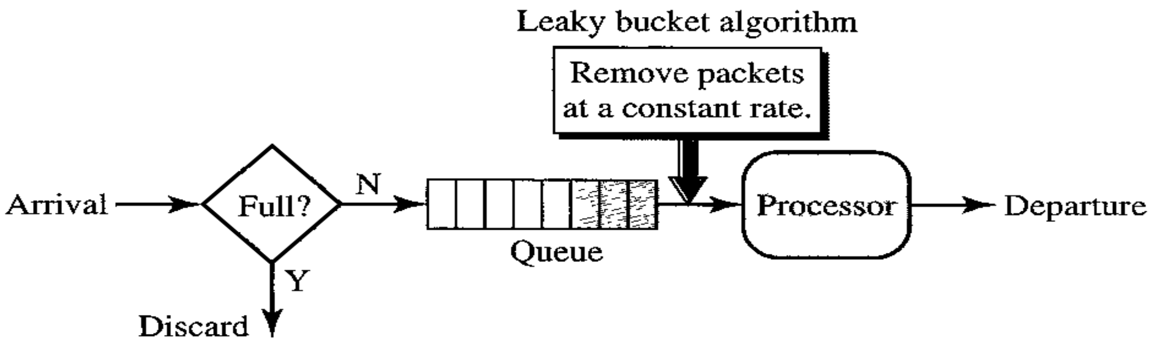
Two techniques can shape traffic: leaky bucket and token bucket.

Leaky Bucket

If a bucket has a small hole at the bottom, the water leaks from the bucket at a constant rate as long as there is water in the bucket. The rate at which the water leaks does not depend on the rate at which the water is input to the bucket unless the bucket is empty. The input rate can vary, but the output rate remains constant. Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic. Bursty chunks are stored in the bucket and sent out at an average rate.



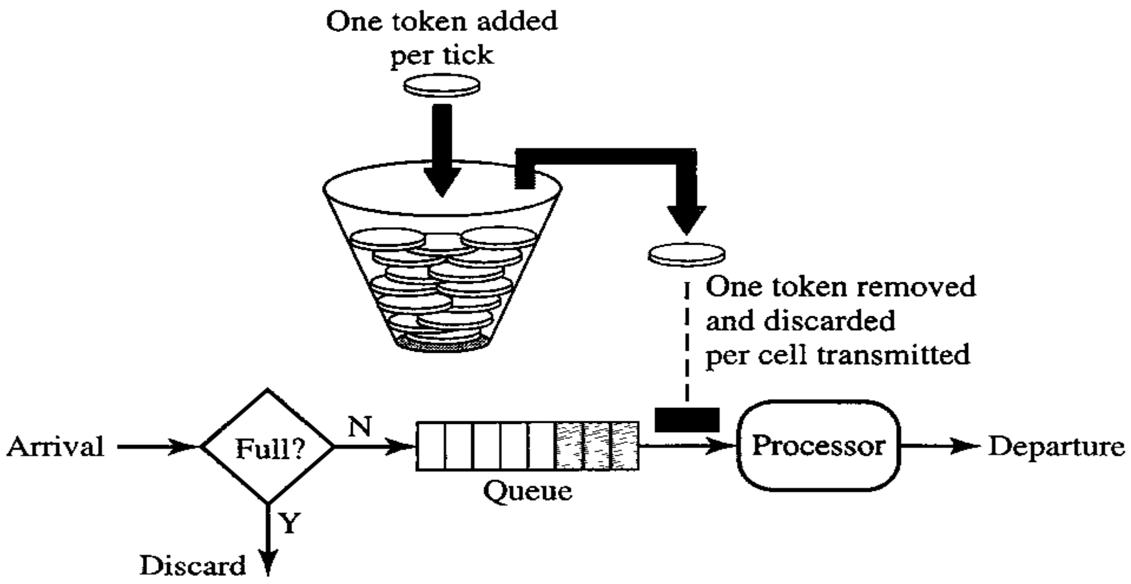
<u>Leaky bucket implementation</u>



Token Bucket

The leaky bucket is very restrictive. It does not credit an idle host. On the other hand, the token bucket algorithm allows idle hosts to accumulate credit for the future in the form of tokens. For each tick of the clock, the system sends *n* tokens to the bucket. The system removes one token for every cell (or byte) of data sent. In other words, the host can send bursty data as long as the bucket is not empty.

The token bucket can easily be implemented with a counter. The token is initialized to zero. Each time a token is added, the counter is incremented by 1. Each time a unit of data is sent, the counter is decremented by 1. When the counter is zero, the host cannot send data.



3. 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.

Packet scheduling algorithms allocate bandwidth and other router resources by determining which of the buffered packets to send on the output line next. 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**. It is intuitive, and you may be wondering what alternatives exist.

- FIFO Queuing
- Priority Queuing
- Weighted Fair Queuing

4. Admission Control

QoS guarantees are established through the process of 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. QoS guarantees for new flows may still be accommodated by choosing a different route for the flow that has excess capacity. This is called **QoS routing**.

Because many parties may be involved in the flow negotiation, flows must be described accurately in terms of specific parameters that can be negotiated. A set of such parameters is called a **flow specification**. Typically, the sender (e.g., the video server) produces a flow specification proposing the parameters it would like to use. As the specification propagates along the route, each router examines it and modifies the parameters as need be. The modifications can only reduce the flow, not increase it. When it gets to the other end, the parameters can be established.

INTEGRATED SERVICES

Two models have been designed to provide quality of service in the Internet: Integrated Services and Differentiated Services. Both models emphasize the use of quality of service at the network layer (IP), although the model can also be used in other layers such as the data link.

Integrated Services, sometimes called IntServ, is *a flow-based* QoS model, which means that a user needs to create a flow, a kind of virtual circuit, from the source to the destination and inform all routers of the resource requirement.

Signaling

The reader may remember that IP is a connectionless, datagram, packet-switching protocol. How can we implement a flow-based model over a connectionless protocol? The solution is a signaling protocol to run over IP that provides the signaling mechanism for making a reservation. This protocol is called Resource Reservation Protocol (RSVP) .

Flow Specification

When a source makes a reservation, it needs to define a flow specification. A flow specification has two parts: Rspec (resource specification) and Tspec (traffic specification). Rspec defines the resource that the flow needs to reserve (buffer, bandwidth, etc.). Tspec defines the traffic characterization of the flow.

Admission

After a router receives the flow specification from an application, it decides to admit or deny the service. The decision is based on the previous commitments of the router and the current availability of the resource.

RSVP

In the Integrated Services model, an application program needs resource reservation. A virtual-circuit network needs a signaling system to set up the virtual circuit before data traffic can start. The Resource Reservation Protocol (RSVP) is a signaling protocol to help IP create a flow and consequently make a resource reservation.

Receiver-Based Reservation

In RSVP, the receivers, not the sender, make the reservation. This strategy matches the other multicasting protocols.

RSVP Messages

RSVP has several types of messages. However, for our purposes, we discuss only two of them: Path and Resv.

Path Messages: The path is needed for the reservation. RSVP uses *Path* messages. A Path message travels from the sender and reaches all receivers in the multicast path. On the way, a Path message stores the necessary information for the receivers. A Path message is sent in a multicast environment; a new message is created when the path diverges.

Resv Messages: After a receiver has received a Path message, it sends a *Resv* message. The Resv message travels toward the sender (upstream) and makes a resource reservation on the routers that support RSVP. If a router does not support RSVP on the path, it routes the packet based on the best-effort delivery methods.

Reservation Merging

In RSVP, the resources are not reserved for each receiver in a flow; the reservation is merged. In Rc3 requests a 2-Mbps bandwidth while Rc2 requests a 1-Mbps bandwidth. Router R3, which needs to make a bandwidth reservation, merges the two requests. The reservation is made for 2 Mbps, the larger of the two, because a 2-Mbps input reservation can handle both requests.

Problems with Integrated Services

There are at least two problems with Integrated Services that may prevent its full implementation in the Internet: scalability and service-type limitation.

Scalability

The Integrated Services model requires that each router keep information for each flow. As the Internet is growing every day, this is a serious problem.

Service-Type Limitation

The Integrated Services model provides only two types of services, guaranteed and control-load. Those opposing this model argue that applications may need more than these two types of services.

<u>DIFFERENTIATED SERVICES</u>

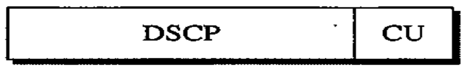
Differentiated Services (DS or Diffserv) was introduced by the IETF (Internet Engineering Task Force) to handle the shortcomings of Integrated Services. Two fundamental changes were made:

1. The main processing was moved from the core of the network to the edge of the network. This solves the scalability problem. The routers do not have to store information about flows. The applications, or hosts, define the type of service they need each time they send a packet.

2. The per-flow service is changed to per-class service. The router routes the packet based on the class of service defined in the packet, not the flow. This solves the service-type limitation problem.

DS Field

In Diffserv, each packet contains a field called the DS field. The value of this field is set at the boundary of the network by the host or the first router designated as the boundary router. IETF proposes to replace the existing TOS (type of service) field in IPv4 or the class field in IPv6 by the DS field.



The DS field contains two subfields: DSCP and CU. The DSCP (Differentiated Services Code Point) is a 6-bit subfield that defines the per-hop behavior (PHB). The 2-bit CU (currently unused) subfield is not currently used.

The Diffserv capable node (router) uses the DSCP 6 bits as an index to a table defining the packet-handling mechanism for the current packet being processed.

Per-Hop Behavior

The Diffserv model defines per-hop behaviors (PHBs) for each node that receives a packet. So far three PHBs are defined: DE PHB, EF PHB, and AF PHB.

DE PHB The DE PHB (default PHB) is the same as best-effort delivery, which is compatible with TOS.

EF PHB The EF PHB (expedited forwarding PHB) provides the following services:

- Low loss
- Low latency
- Ensured bandwidth

This is the same as having a virtual connection between the source and destination.

AF PHB The AF PHB (assured forwarding PHB) delivers the packet with a high assurance as long as the class traffic does not exceed the traffic profile of the node. The users of the network need to be aware that some packets may be discarded.