COMPUTER NETWORK

GATE 2023

Section 10: Computer Networks

1. Concept of layering
    1. OSI
    2. TCP/IP Protocol Stacks.
2. Basics of packet, circuit and virtual circuit-switching.
3. Data link layer
    1. Framing
    2. Error detection
    3. Medium Access Control
    4. Ethernet bridging.
4. Routing protocols
    1. Shortest path
    2. Flooding
    3. Distance vector
    4. Link state routing.
5. Fragmentation and IP addressing, IPv4, CIDR notation, Basics of IP support protocols (ARP, DHCP, ICMP), Network Address Translation (NAT).
6. Transport layer
    1. Flow control and congestion control,
    2. UDP
    3. TCP
    4. Sockets.
7. Application layer protocols
    1. DNS
    2. SMTP
    3. HTTP
    4. FTP
    5. Email.

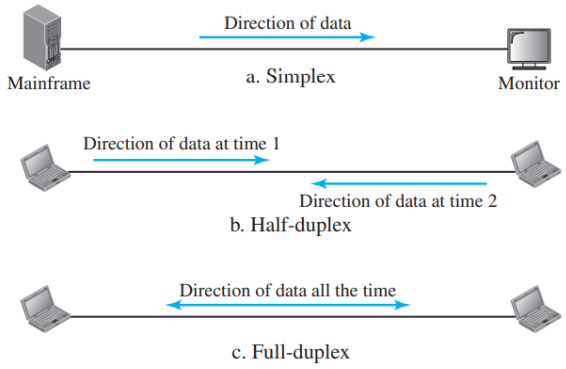
INTRODUCTION

1) Telecommunication – Tele Greek word means far, and communication means transfer of data
2) Data - information presented in whatever form is agreed upon by the parties creating and using the data
3) Data communications are the exchange of data between two devices via some form of transmission medium such as a wire cable
4) The effectiveness of a data communications system depends on four fundamental characteristics:
   Delivery - The system must deliver data to the correct destination.
   Accuracy - The system must deliver the data accurately
   Timeliness - The system must deliver data in a timely manner
   Jitter - Jitter refers to the variation in the packet arrival time

- Components :

1) Message. The message is the information (data) to be communicated. Popular forms of information include text, numbers, pictures, audio, and video.
2) Sender. The sender is the device that sends the data message. It can be a computer, workstation, telephone handset, video camera, and so on.
3) Receiver. The receiver is the device that receives the message. It can be a computer, workstation, telephone handset, television, and so on.
4) Transmission medium. The transmission medium is the physical path by which a message travels from sender to receiver. Some examples of transmission media include twisted-pair wire, coaxial cable, fiber-optic cable, and radio waves.
5) Protocol. A protocol is a set of rules that govern data communications. It represents an agreement between the communicating devices. Without a protocol, two devices may be connected but not communicating, just as a person speaking French cannot be understood by a person who speaks only Japanese

- Data flow :



1.2) Networks :

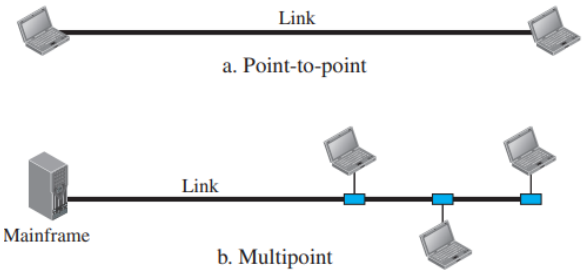
A network is the interconnection of a set of devices capable of communication. In this definition, a device can be a host (or an end system as it is sometimes called) such as a large computer, desktop, laptop, workstation, cellular phone, or security system. A device in this definition can also be a connecting device such as a router, which connects the network to other networks, a switch, which connects devices together, a modem (modulator-demodulator), which changes the form of data, and so on.

⇨ Network criteria :

1) Performance : Performance can be measured in many ways, including transit time and response time. Transit time is the amount of time required for a message to travel from one device to another. Response time is the elapsed time between an inquiry and a response.
2) Reliability : Reliability is measured by the frequency of failure, the time it takes a link to recover from a failure, and the network's robustness in a catastrophe.
3) Security : protecting data from unauthorized access

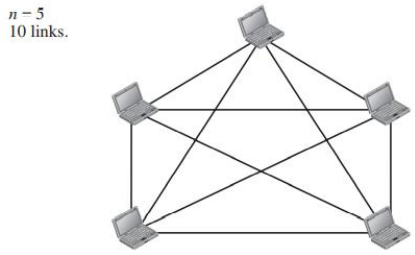
Baseband differ from broadband in homogeneous data. In baseband you can send only one type of data at time, whereas in broadband you can send text, audio, video, etc,… simultaneously. In baseband we have notation like 10base5 or 10base2. Here 10 means 10mbps bandwidth channel, base means baseband and 5 means 500m length of channel.

⇨ Physical structure : type of connection : 1) point to point 2) multipoint

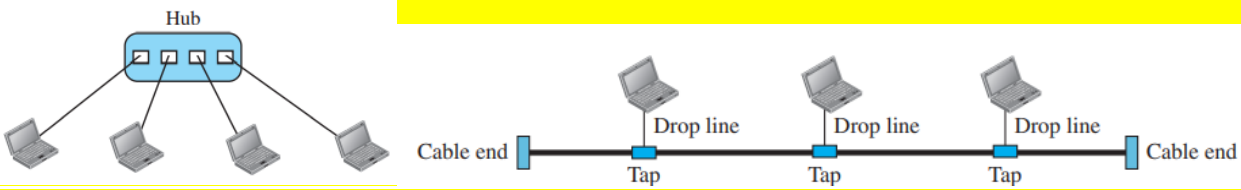


Physical topology : The term physical topology refers to the way in which a network is laid out physically.

1) *Mesh Topology* : n(n-1)/2 nodes in simplex and n(n-1) nodes in duplex



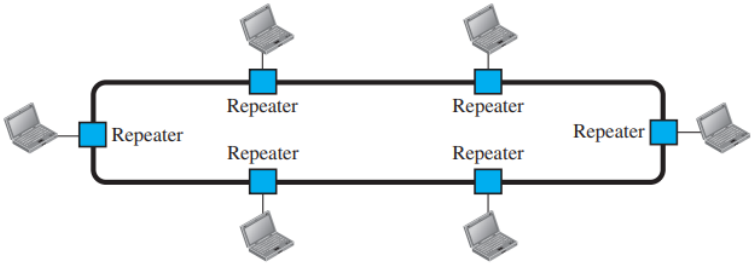
2) *Star* : In a star topology, each device has a dedicated point-to-point link only to a central controller, usually called a hub.



3) Bus : multipoint. In bus topology, we use repeater for long data transmission in between two bus topologies, here we use baseband as channel.

Problem : Standard Ethernet cable 10base5 are used for connecting similar LAN's in bus topology. The number of repeaters required to have a length of 3000 m of total LAN are _____.
Answer : 500m channel is required by one bus topology and total length is 3000m so 6 bus topology is required in between these topology we use 5 repeater. So answer is 5.

4) Ring : each device has a dedicated point-to-point connection with only the two devices on either side of it.



- **Network type** :

⇨ LAN's : A local area network (LAN) is usually privately owned and connects some hosts in a single office, building, or campus.

⇨ Wide Area Network : a WAN has a wider geographical span, spanning a town, a state, a country, or even the world.

1) Point-to-Point WAN : A point-to-point WAN is a network that connects two communicating devices through a transmission media (cable or air)
2) Switched WAN : A switched WAN is a network with more than two ends.

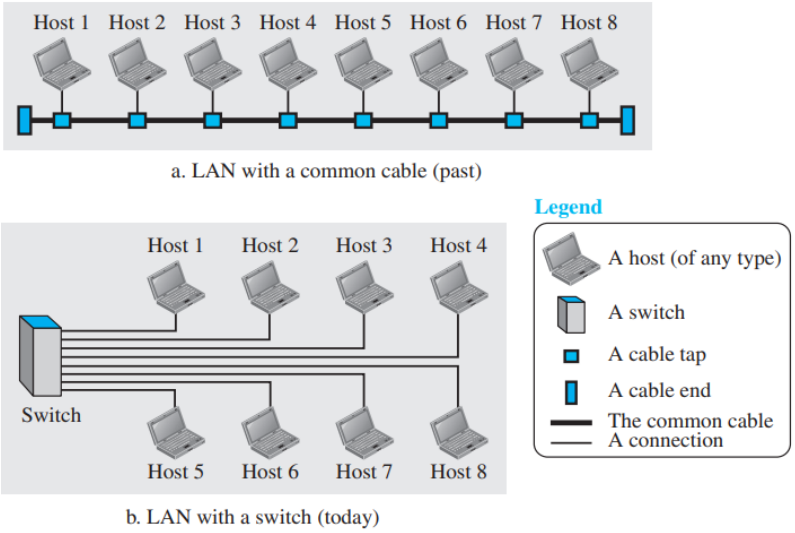


Figure 1.9 *A point-to-point WAN*

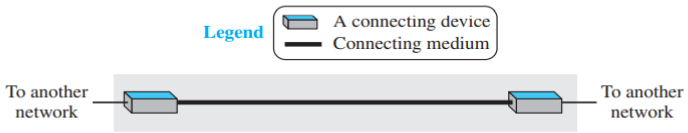
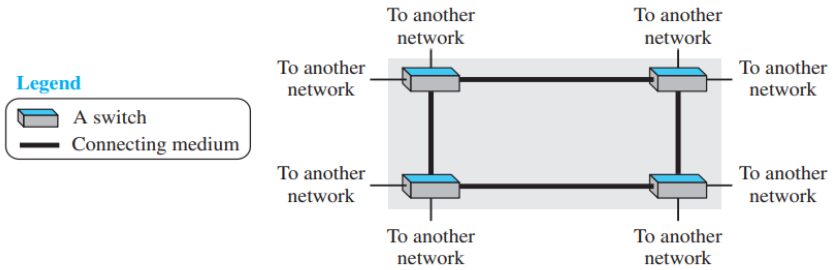


Figure 1.10 *A switched WAN*



- When two or more networks are connected, they make an internetwork, or internet.

Figure 1.13 *A circuit-switched network*

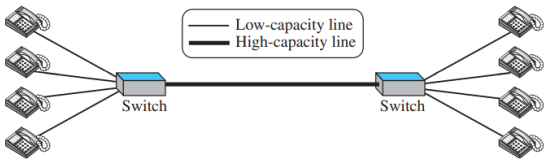
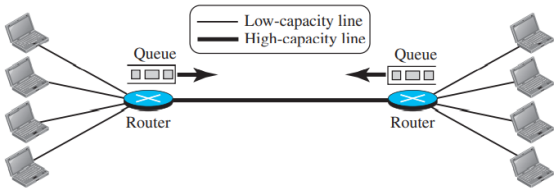
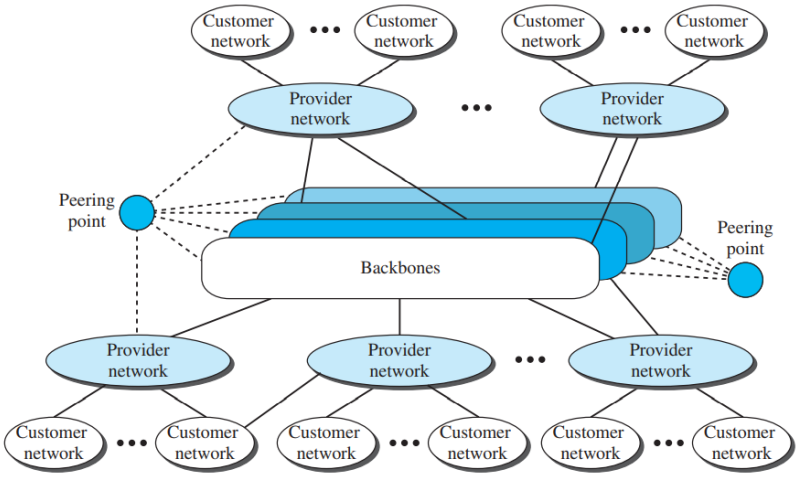


Figure 1.14 *A packet-switched network*



- Authorities made a decision to split TCP into two protocols: Transmission Control Protocol (TCP) and Internet Protocol (IP). IP would handle datagram routing while TCP would be responsible for higher level functions such as segmentation, reassembly, and error detection.

Figure 1.15 *The Internet today*



- **Network Model** :

Protocol : In data communication and networking, a protocol defines the rules that both the sender and receiver and all intermediate devices need to follow to be able to communicate effectively.

When communication is simple, we may need only one simple protocol; when the communication is complex, we may need to divide the task between different layers, in which case we need a protocol at each layer, or **protocol layering.**

⇨ Principle of protocol layering :

1) First Principle : The first principle dictates that if we want bidirectional communication, we need to make each layer so that it is able to perform two opposite tasks, one in each direction.
2) Second Principle : The second principle that we need to follow in protocol layering is that the two objects under each layer at both sites should be identical.

Figure 2.2 *A three-layer protocol*

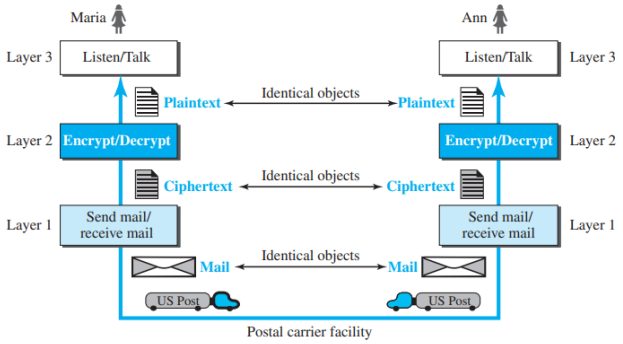
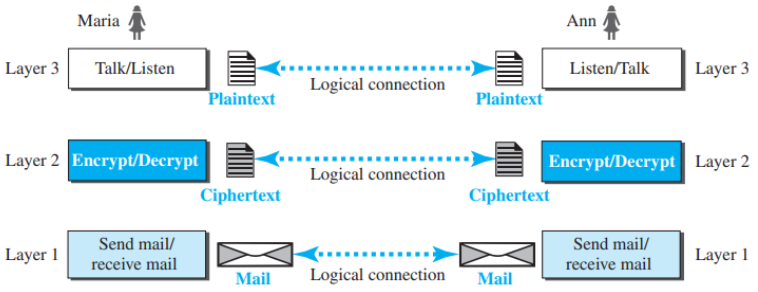


Figure 2.3 *Logical connection between peer layers*



- ⇨ Logical Connection : Layer to layer connection
- ⇨ TCP/IP PROTOCOL SUITE : It is a **hierarchical protocol** made up of interactive modules, each of which provides a specific functionality. The term hierarchical means that each upper level protocol is supported by the services provided by one or more lower level protocols. The original TCP/IP protocol suite was defined as four software layers built upon the hardware. Today, however, TCP/IP is thought of as a five-layer model.

Figure 2.4 *Layers in the TCP/IP protocol suite*

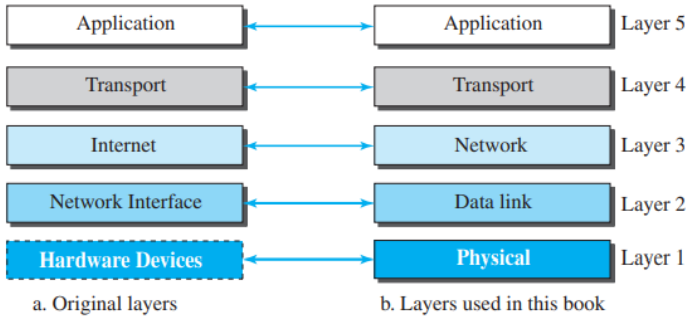


Figure 2.5 *Communication through an internet*

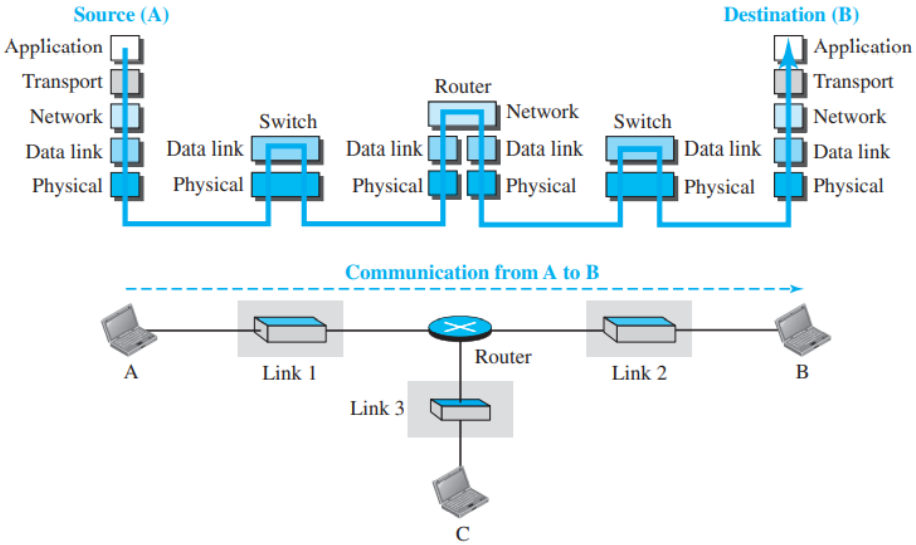
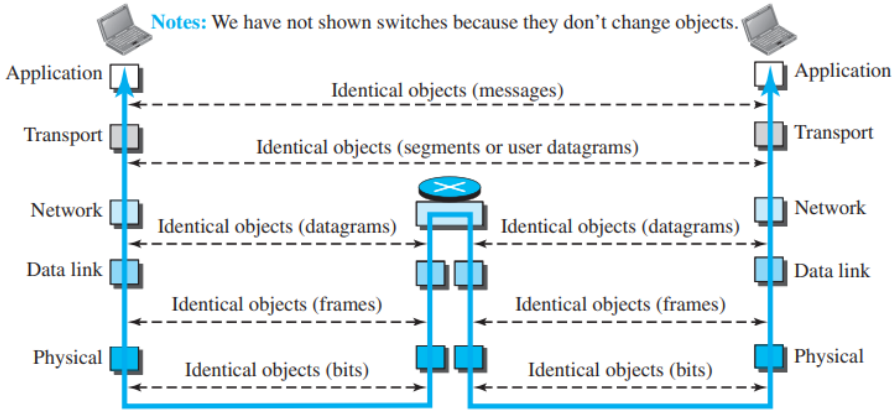


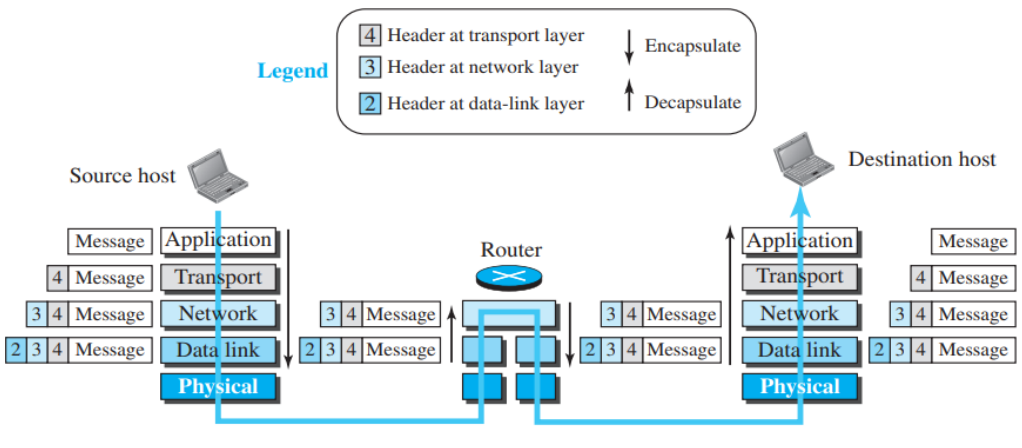
Figure 2.7 *Identical objects in the TCP/IP protocol suite*



1) Physical Layer : We can say that the physical layer is responsible for carrying individual bits in a frame across the link. there is another, hidden layer, the transmission media, under the physical layer. Two devices are connected by a transmission medium (cable or air). We need to know that the transmission medium does not carry bits; it carries electrical or optical signal.
2) Data-link Layer : We have seen that an internet is made up of several links (LANs and WANs) connected by switch.
3) Network Layer : The network layer is responsible for creating a connection between the source computer and the destination computer. The network layer in the Internet includes the main protocol, Internet Protocol (IP), that defines the format of the packet, called a datagram at the network layer. IP also defines the format and the structure of addresses used in this layer.
4) Transport Layer : The logical connection at the transport layer is also end-to-end. The transport layer at the source host gets the message from the application layer, encapsulates it in a transport layer packet (called a segment or a user datagram in different protocols) and sends it, through the logical (imaginary) connection, to the transport layer at the destination host.
5) Application Layer : the logical connection between the two application layers is end-to-end. The two application layers exchange messages between each other as though there were a bridge between the two layers.

⇨ Encapsulation and Decapsulation :

Figure 2.8 *Encapsulation/Decapsulation*



1) Encapsulation at Source : At the application layer, the data to be exchanged is referred to as a **message**. The transport layer takes the message as the payload, the load that the transport layer should take care of. It adds the transport layer header to the payload, which contains the identifiers of the source and destination application programs that we want to communicate plus some more information that is needed for the end-to-end delivery of the message. All together they called **segment**. The network layer takes the transport-layer packet as data or payload and adds its own header to the payload. The header contains the addresses of the source and destination hosts and some more information used for error checking of the header . All together they called **datagram**. the data-link layer takes the network-layer packet as data or payload and adds its own header, which contains the link-layer addresses of the host or the next hop (the router). The result is the link-layer packet, which is called a **frame**.
2) Encap/Decapsu at router : After the set of bits are delivered to the data-link layer, this layer decapsulates the datagram from the frame and passes it to the network layer. The network layer only inspects the source and destination addresses in the datagram header and consults its forwarding table to find the next hop to which the datagram is to be delivered. The data-link layer of the next link encapsulates the datagram in a frame and passes it to the physical layer for transmission. Hop means link not router.
3) Decap at destination : it is reverse of encap at source.

Figure 2.9 *Addressing in the TCP/IP protocol suite*

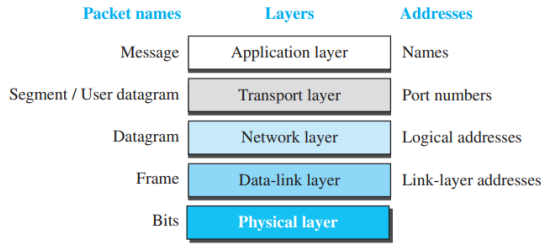
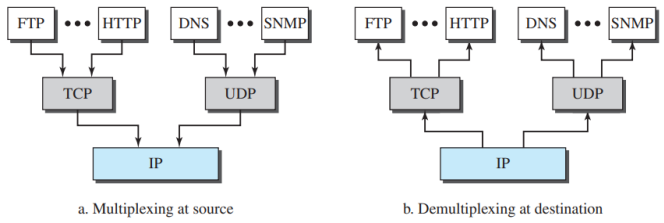


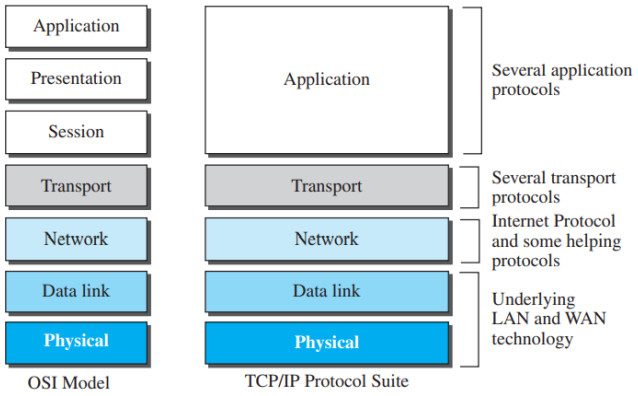
Figure 2.10 *Multiplexing and demultiplexing*



The physical address, also known as the link address, is the address of a node as defined by its LAN or WAN. The IP address uniquely defines a host on the Internet. The port address identifies a process on a host. A specific address is a user-friendly address.

- OSI Model (Open system interconnection) : An open system is a set of protocols that allows any two different systems to communicate regardless of their underlying architecture. The purpose of the OSI model is to show how to facilitate communication between different systems without requiring changes to the logic of the underlying hardware and software. The OSI model is not a protocol; it is a model for understanding and designing a network architecture that is flexible, robust, and interoperable.
- ISO is the organization; OSI is the model. ///// International Organization for Standardization (ISO)

Figure 2.12 *TCP/IP and OSI model*



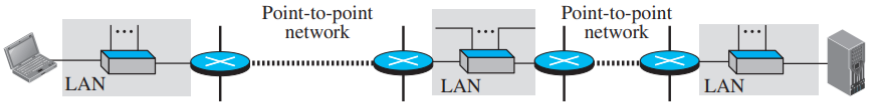
The OSI model appeared after the TCP/IP protocol suite.

Reason of Failure of OSI model : 1) lot of time and money had been spent on the suite; changing it would cost a lot. 2) actual protocols for these two layers were not fully defined, nor were they fully described, and the corresponding software was not fully developed. 3) It did not show a high enough level of performance to entice the Internet authority to switch from the TCP/IP protocol suite to the OSI model.

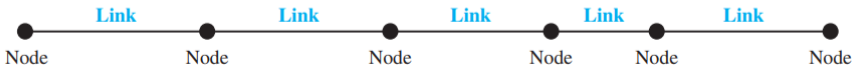
DATA LINK LAYER

Introduction

1) Nodes and links : Communication at the data-link layer is node-to-node. A data unit from one point in the Internet needs to pass through many networks (LANs and WANs) to reach another point. Theses LANs and WANs are connected by routers. It is customary to refer to the two end hosts and the routers as nodes and the networks in between as links. Figure 9.2 is a simple representation of links and nodes when the path of the data unit is only six nodes.



a. A small part of the Internet

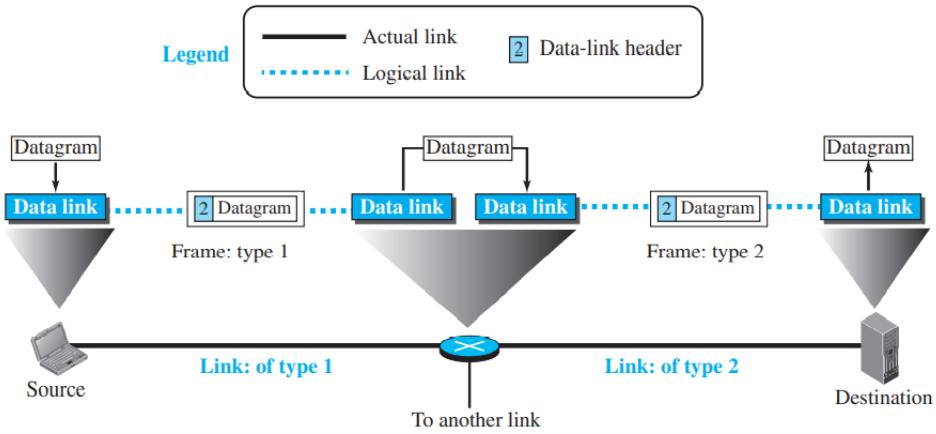


b. Nodes and links

The first node is the source host; the last node is the destination host. The other four nodes are four routers. The first, the third, and the fifth links represent the three LANs; the second and the fourth links represent the two WANs.

2) Services :

Figure 9.3 *A communication with only three nodes*



The duty scope of the data-link layer is node-to-node.

a) Framing : The data-link layer at each node needs to encapsulate the datagram (packet received from the network layer) in a frame before sending it to the next node. The node also needs to decapsulate the datagram from the frame received on the logical channel.
A packet at the data-link layer is normally called a frame.

b) Flow control : If the producer produces items that cannot be consumed, accumulation of items occurs. The sending data-link layer at the end of a link is a producer of frames; the receiving data-link layer at the other end of a link is a consumer. If the rate of produced frames is higher than the rate of consumed frames, frames at the receiving end need to be buffered while waiting to be consumed (processed).

c) Error Control : Since electromagnetic signals are susceptible to error, a frame is susceptible to error. The error needs first to be detected. After detection, it needs to be either corrected at the receiver node or discarded and retransmitted by the sending node.

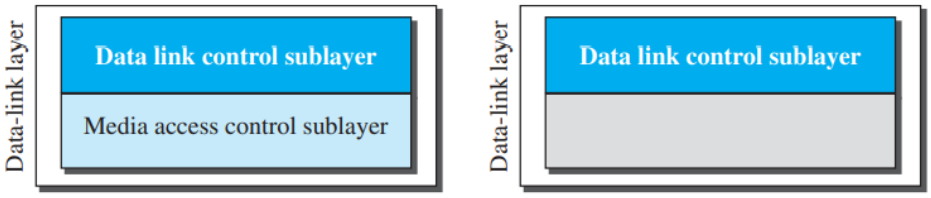
d) Congestion Control : Although a link may be congested with frames, which may result in frame loss, most data-link-layer protocols do not directly use a congestion control to alleviate congestion,

3) Although two nodes are physically connected by a transmission medium such as cable or air, we need to remember that the data-link layer controls how the medium is used.

- we can have a point-to-point link or a broadcast link. In a point-to-point link, the link is dedicated to the two devices; in a broadcast link, the link is shared between several pairs of devices. For example, when two friends use the traditional home phones to chat, they are using a point-to-point link; when the same two friends use their cellular phones, they are using a broadcast link (the air is shared among many cell phone users).

4) To better understand the functionality of and the services provided by the link layer, we can divide the data-link layer into two sublayers: **data link control (DLC)** and **media access control (MAC)**.

- The data link control sublayer deals with all issues common to both point-to-point and broadcast links; the media access control sublayer deals only with issues specific to broadcast links.

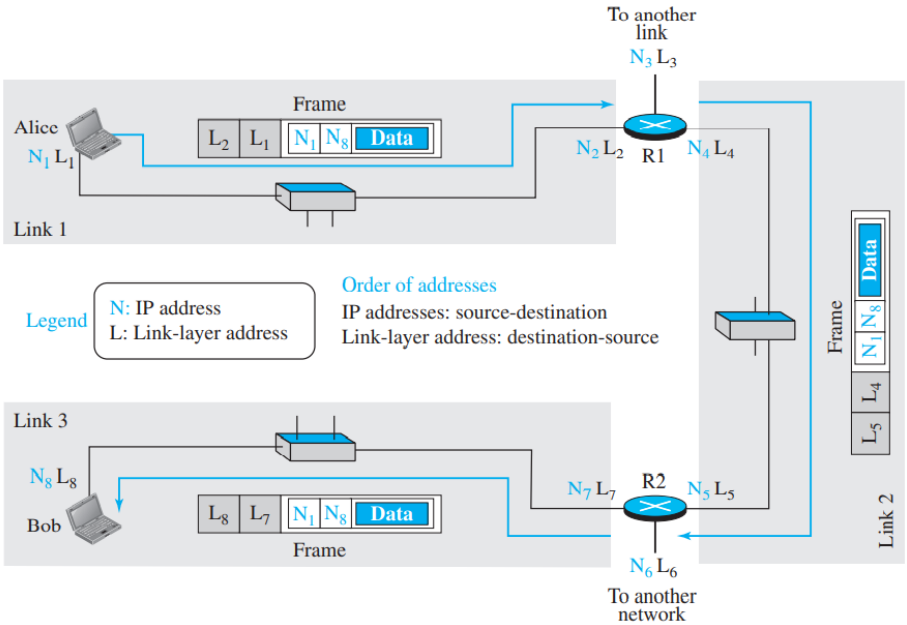


a. Data-link layer of a broadcast link

b. Data-link layer of a point-to-point link

5) Link Layer Addressing : A link-layer address is sometimes called a link address, sometimes a physical address, and sometimes a MAC address.

Figure 9.5 *IP addresses and link-layer addresses in a small internet*



a) Three Types of Addresses :

Unicast and multicast address Checking scheme is different for router and mac address.

Unicast Address : Each host or each interface of a router is assigned a unicast address. Unicasting means one-to-one communication. A frame with a unicast address destination is destined only for one entity in the link. Example 9.1 As we will see in Chapter 13, the unicast link-layer addresses in the most common LAN, Ethernet, are 48 bits (six bytes) that are presented as 12 hexadecimal digits separated by colons; for example, the following is a link-layer address of a computer. Second digit should be even.

A**2**:34:45:11:92:F1

Multicast Address : Some link-layer protocols define multicast addresses. Multicasting means one-to-many communication. However, the jurisdiction is local (inside the link), The second digit, however, needs to be an odd number in hexadecimal. Multicast address can be destination address but not a source address.

A**3**:34:45:11:92:F1

Broadcast Address : Some link-layer protocols define a broadcast address. Broadcasting means one-to-all communication. A frame with a destination broadcast address is sent to all entities in the link. the broadcast link-layer addresses in the most common LAN, Ethernet, are 48 bits, all 1s, that are presented as 12 hexadecimal digits separated by colons.

FF:FF:FF:FF:FF:FF

b) Unicast and multicast in MAC address :

If the LSB of the 1st byte is 0 ⇒ Unicast

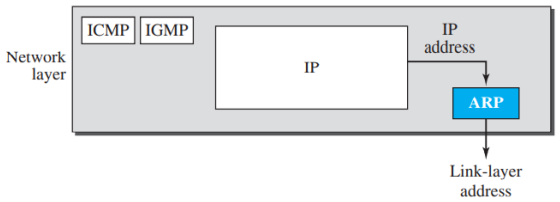
If the LSB of the 1st byte is 1 ⇒ Multicast

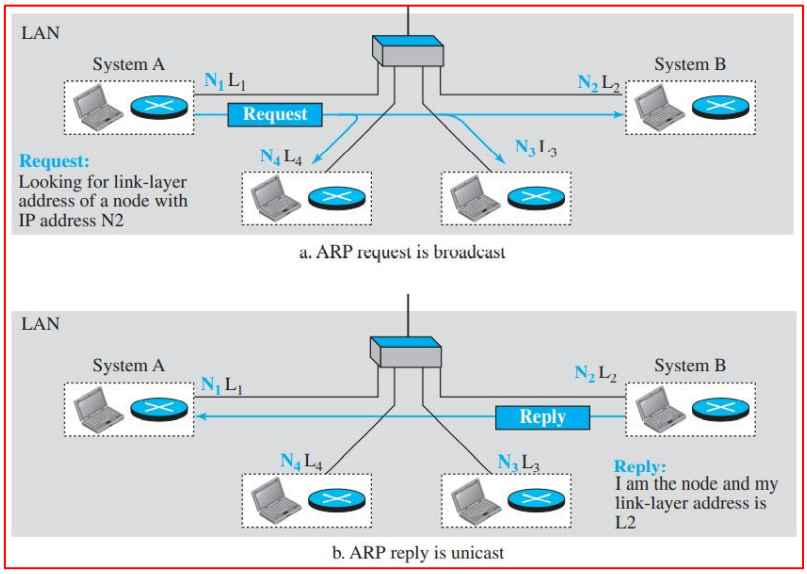
Remember it is 0 and 1 bit of 1st byte.

The range of multicast MAC Address lie between 01-00-5E-00-00-00 to 01-00-5E-7F-FF-FF. Remember this is for multicast not for unicast.

6) Address Resolution Protocol (ARP) :

The last router knows the IP address of the destination host. However, the IP address of the next node is not helpful in moving a frame through a link; we need the link-layer address of the next node. This is the time when the Address Resolution Protocol (ARP) becomes helpful. ARP accepts an IP address from the IP protocol, maps the address to the corresponding link-layer address, and passes it to the data-link layer.





Questions : A question that is often asked is this: If system A can broadcast a frame to find the link layer address of system B, why can't system A send the datagram for system B using a broadcast frame?

Answer : Let us assume that there are 20 systems connected to the network (link): system A, system B, and 18 other systems. We also assume that system A has 10 datagrams to send to system B in one second.

a. Without using ARP, system A needs to send 10 broadcast frames. Each of the 18 other systems need to receive the frames, decapsulate the frames, remove the datagram and pass it to their network-layer to find out the datagrams do not belong to them. This means processing and discarding 180 broadcast frames.

b. Using ARP, system A needs to send only one broadcast frame. Each of the 18 other systems need to receive the frames, decapsulate the frames, remove the ARP message and pass the message to their ARP protocol to find that the frame must be discarded. This means processing and discarding only 18 (instead of 180) broadcast frames. After system B responds with its own data-link address, system A can store the link-layer address in its cache memory.

Figure 9.8 *ARP packet*

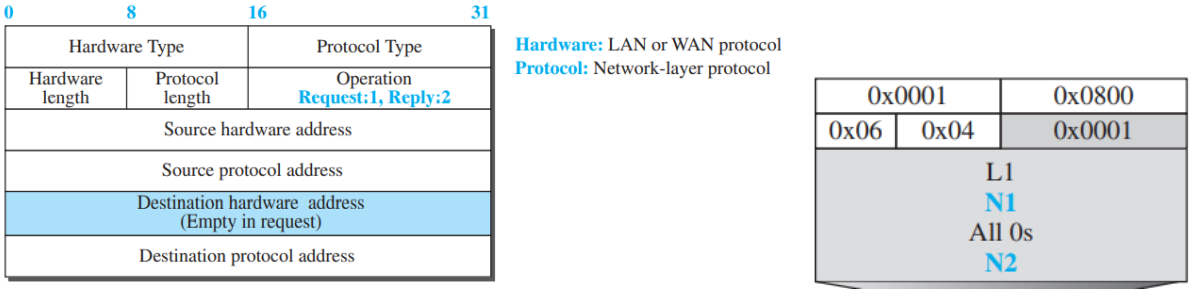
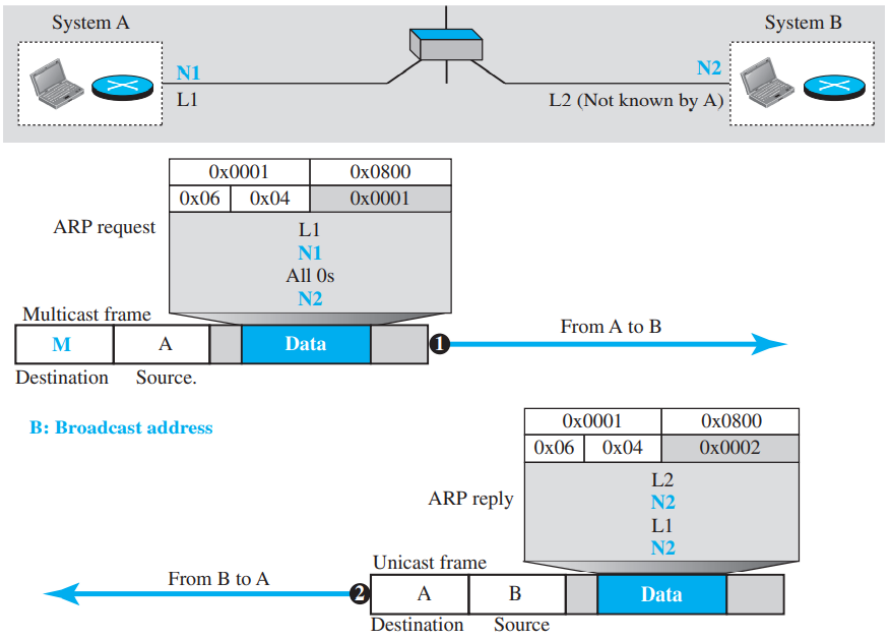


Figure 9.9 *Example 9.4*

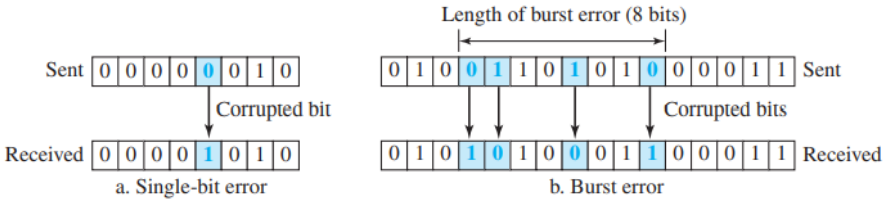


If a host is trying to send a packet to a device on a remote LAN segment, but there are currently no mappings in its ARP cache in that case it will send an ARP request for the MAC address of the default gateway. Remote host means device which is not in mapping.

Error Detection and Correction

1) Whenever bits flow from one point to another, they are subject to unpredictable changes because of interference.

Figure 10.1 *Single-bit and burst error*



2) The central concept in detecting or correcting errors is redundancy. To be able to detect or correct errors, we need to send some extra bits with our data. These redundant bits are added by the sender and removed by the receiver. Their presence allows the receiver to detect or correct corrupted bits.
3) In error detection, we are only looking to see if any error has occurred. The answer is a simple yes or no. We are not even interested in the number of corrupted bits. A single-bit error is the same for us as a burst error.
4) In error correction, we need to know the exact number of bits that are corrupted and, more importantly, their location in the message. The number of errors and the size of the message are important factors.
5) Redundancy is achieved through various coding schemes. The sender adds redundant bits through a process that creates a relationship between the redundant bits and the actual data bits.

- **Block Coding**:

In block coding, we divide our message into blocks, each of k bits, called **datawords**. We add r redundant bits to each block to make the length n = k + r. The resulting n-bit blocks are called **codewords.**

The sender creates codewords out of datawords by using a generator that applies the rules and procedures of encoding (discussed later). Each codeword sent to the receiver may change during transmission. If the received codeword is the same as one of the valid codewords, the word is accepted; the corresponding dataword is extracted for use. If the received codeword is not valid, it is discarded. However, if the codeword is corrupted during transmission but the received word still matches a valid codeword, the error remains undetected.

An error-detecting code can detect only the types of errors for which it is designed; other types of errors may remain undetected

1) *Hamming Distance* : The Hamming distance between two words (of the same size) is the number of differences between the corresponding bits.

For example, if the codeword 00000 is sent and 01101 is received, 3 bits are in error and the Hamming distance between the two is d(00000, 01101) = 3.

To guarantee the <u>detection</u> of up to p errors in all cases, the minimum Hamming distance in a block code must be dmin = p + 1. d>=2p+1 when d is distance and p is number of error bits. This formula is for <u>correction</u>. For n bit error detection, we need n parity bits.

Hamming encoding algorithm :

Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15	
Parity bit coverage	p1	✓		✓		✓		✓		✓		✓		✓		✓		✓		✓		...
	p2		✓	✓			✓	✓			✓	✓			✓	✓			✓	✓		
	p4				✓	✓	✓	✓					✓	✓	✓	✓					✓	
	p8								✓	✓	✓	✓	✓	✓	✓	✓						
	p16																✓	✓	✓	✓	✓	

Example :

Consider A 12-bit even parity Hamming code word containing 8 bits of data and 4 parity bits is read from memory. The format of the Hamming code word is shown in below figure:

Bit position	1	2	3	4	5	6	7	8	9	10	11	12
Encoded data bits	p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8

What was the original 8-bit data word that was written into memory if the 12-bit encoded data bits $p_1p_2d_1p_4d_2d_3d_4p_8d_5d_6d_7d_8$ = 000011101010?

(a) 01011010

(b) 01111010

(c) 01101010

(d) 00111010

Solution:

Received code is 000011101010. We perform the parity check for even parity.

Bit position	1	2	3	4	5	6	7	8	9	10	11	12	
Encoded data bits	P1	P2	D1	P4	D2	D3	D4	P8	D5	D6	D7	D8	
12-bit encoded data bits	0	0	0	0	1	1	1	0	1	0	1	0	
Parity Check for P1	0		0		1		1		1		1		Pass as number of 1's even
Parity Check for P2		0	0			1	1			0	1		Fail as number of 1's odd
Parity Check for P4				0	1	1	1					0	Fail as number of 1's odd
Parity Check for P8								0	1	0	1	0	Pass as number of 1's even

The parity check for parity bit P2 and P4 fails; therefore there is error in the bit position 2 + 4 = 6. So the correct the original 8-bit data word that was written into memory = 01011010.

2) *Linear Block Coding* : a linear block code is a code in which the exclusive OR (addition modulo-2) of two valid codewords creates another valid codeword.

Minimum Distance for Linear Block Codes : It is simple to find the minimum Hamming distance for a linear block code. The minimum Hamming distance is the number of 1s in the nonzero valid codeword with the smallest number of 1s.

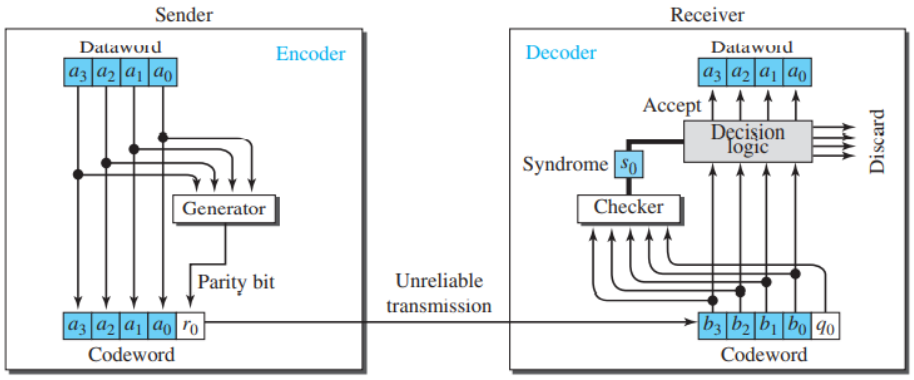
Example : The code in Table 10.1 is a linear block code because the result of XORing any codeword with any other codeword is a valid codeword. For example, the XORing of the second and third codewords creates the fourth one.

Table 10.1 *A code for error detection in Example 10.1*

Dataword	*Codeword*	*Dataword*	*Codeword*
00	000	10	101
01	011	11	110

3) *Parity Check Code* : Perhaps the most familiar error-detecting code is the parity-check code. This code is a linear block code. In this code, a k-bit dataword is changed to an n-bit codeword where n = k + 1. The extra bit, called the **parity bit**, is selected to make the total number of 1s in the codeword even.

Figure 10.4 *Encoder and decoder for simple parity-check code*



r0 = a3 + a2+a1+a0 (mod2)

If the number of 1s is even, the result is 0; if the number of 1s is odd, the result is 1. In both cases, the total number of 1s in the codeword is even.

S0 is called syndrome, s0 = b3+b2+b1+b0+q0 (mod 2)

If the syndrome is 0, there is no detectable error in the received codeword; the data portion of the received codeword is accepted as the dataword; if the syndrome is 1, the data portion of the received codeword is discarded. The dataword is not created.

A parity-check code can detect an odd number of errors.

4) *Cyclic Codes* : Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword. For example, if 1011000 is a codeword and we cyclically left-shift, then 0110001 is also a codeword. If the degree of polynomial is n then we have to append n zeros in divison.
cyclic codes called the cyclic redundancy check (CRC), which is used in networks such as LANs and WANs.

Figure 10.5 *CRC encoder and decoder*

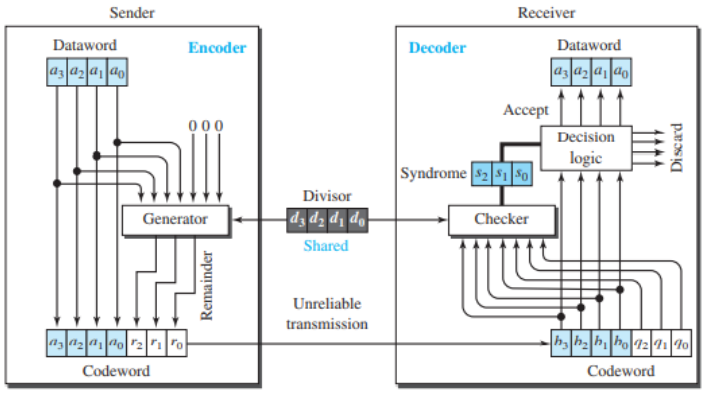


Figure 10.7 *Division in the CRC decoder for two cases*

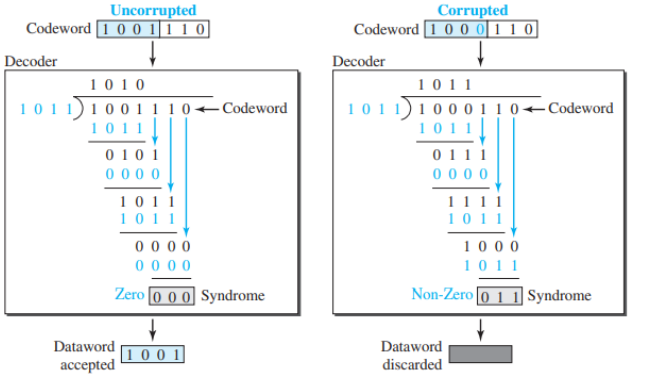
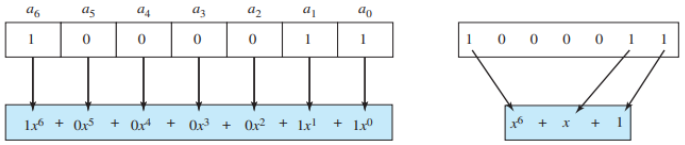
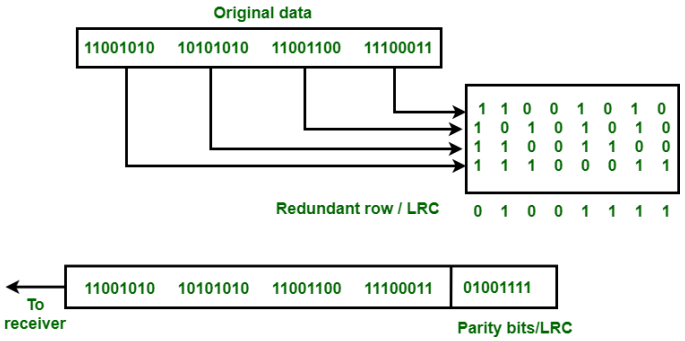


Figure 10.8 *A polynomial to represent a binary word*

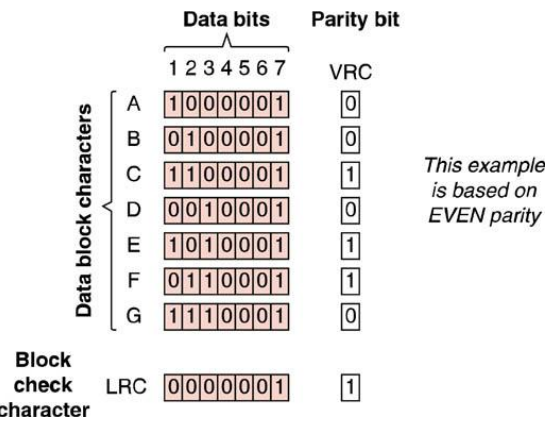


$x^{23} + x^3 + 1$. Here the bit pattern is 24 bits in length (three 1s and twenty-one 0s) while the polynomial is just three terms.



- **All burst errors with $L \le r$ will be detected.**
- **All burst errors with $L = r + 1$ will be detected with probability $1 - (1/2)^{r-1}$.**
- **All burst errors with $L > r + 1$ will be detected with probability $1 - (1/2)^r$.**

Longitudinal redundancy check.



We can summarize the criteria for a good polynomial generator:

A good polynomial generator needs to have the following characteristics:

1. **It should have at least two terms.**
2. **The coefficient of the term x^0 should be 1.**
3. **It should not divide $x^t + 1$, for t between 2 and $n - 1$.**
4. **It should have the factor $x + 1$.**

A generator that contains a factor of x + 1 can detect all odd-numbered errors.

5) Checksum : Checksum is an error-detecting technique that can be applied to a message of any length. In the Internet, the checksum technique is mostly used at the network and transport layer rather than the data-link layer.

Note :

1) Accuracy of error detection : *CRC > Checksum > Parity Check*
2) LRC can detect upto 3 bit of error and can only detect and correct all single bit error.
3) Every hamming code satisfy the following condition : $d+r+1<=2^r$
   d = number of data bits, r = number of parity bits or control bit
4) CRC polynomial codes use a field where, if X divided by G remainder C, then (X+C) divided by G should give remainder 0.
5) $G(x) = x^5$, will not able to detect error in datacode $x^{15} + x^6$ because x^5 is factor of this datacode.

//End of introduction now part 1 : Data link control

Data Link Control (DLC)

The data link control (DLC) deals with procedures for communication between two adjacent nodes—node-to-node communication—no matter whether the link is dedicated or broadcast. Data link control functions include framing and flow and error control.

1) **Framing** : Frame can be of fixed or variable size. In fixed-size framing, there is no need for defining the boundaries of the frames; the size itself can be used as a delimiter.

- Variable sized framing : we need a way to define the end of one frame and the beginning of the next.

Figure 11.1 *A frame in a character-oriented protocol*

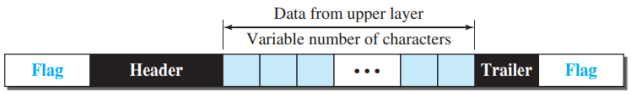


Figure 11.3 *A frame in a bit-oriented protocol*

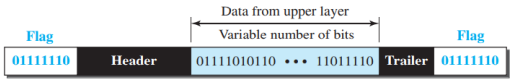
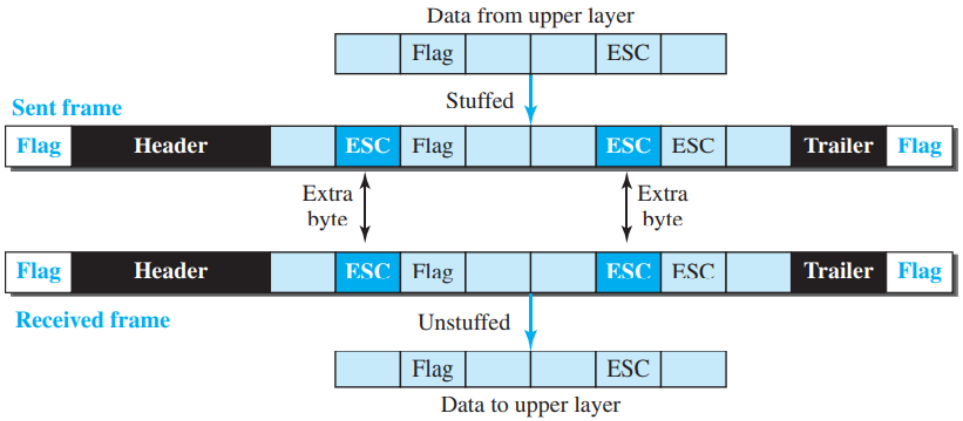
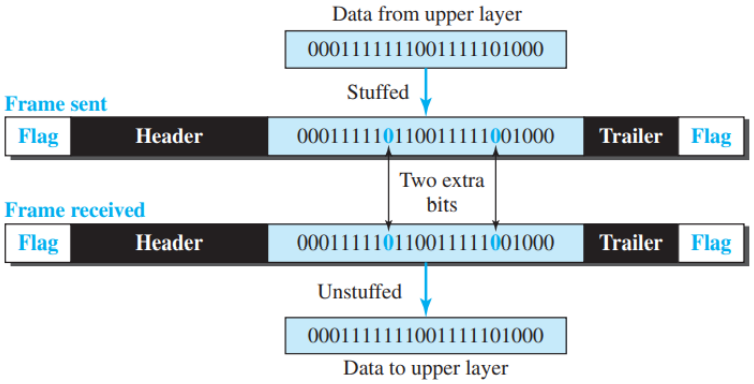


Figure 11.2 *Byte stuffing and unstuffing*



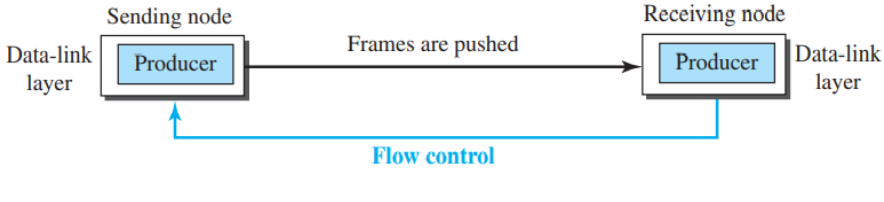
Byte stuffing is the process of adding one extra byte whenever there is a flag or escape character in the text.

Figure 11.4 *Bit stuffing and unstuffing*



Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 0111110 for a flag.

Figure 11.5 *Flow control at the data-link layer*



A buffer is a set of memory locations that can hold packets at the sender and receiver. The flow control communication can occur by sending signals from the consumer to the producer. When the buffer of the receiving data-link layer is full, it informs the sending data-link layer to stop pushing frames.

- Data link protocol : Simple, Stop-and-Wait, Go-Back-N, and Selective-Repeat.

1) **Simple protocol** : Our first protocol is a simple protocol with neither flow nor error control. We assume that the receiver can immediately handle any frame it receives.

Figure 11.7 *Simple protocol*

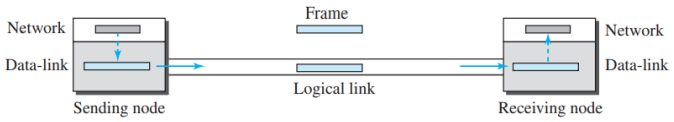


Figure 11.8 *FSMs for the simple protocol*

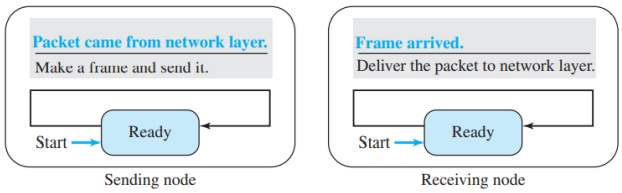
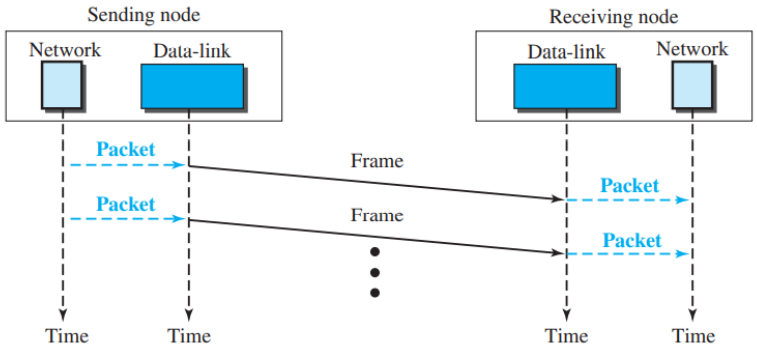


Figure 11.9 *Flow diagram for Example 11.2*



2) **Stop and wait protocol** : uses both flow and error control.

Figure 11.10 *Stop-and-Wait protocol*

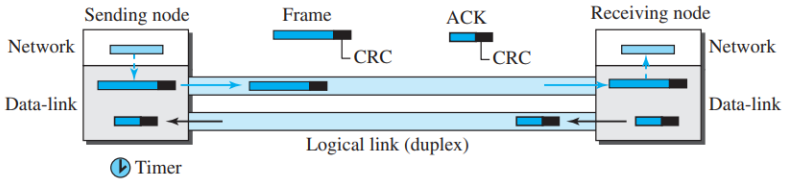


Figure 11.11 *FSM for the Stop-and-Wait protocol*

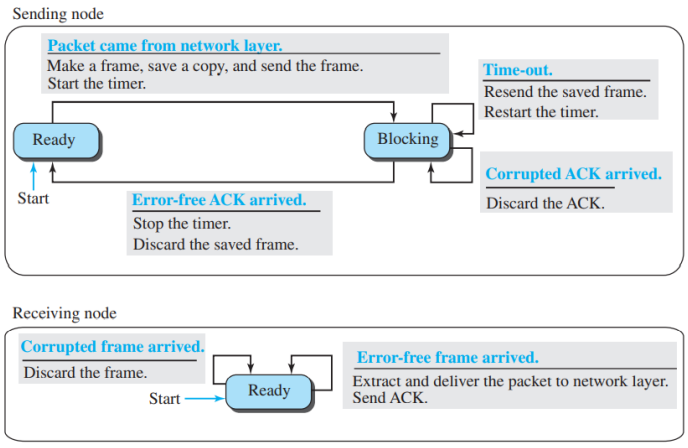
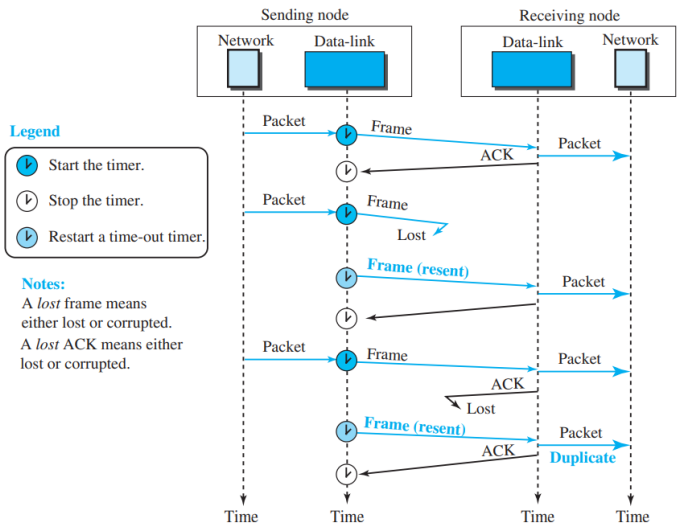
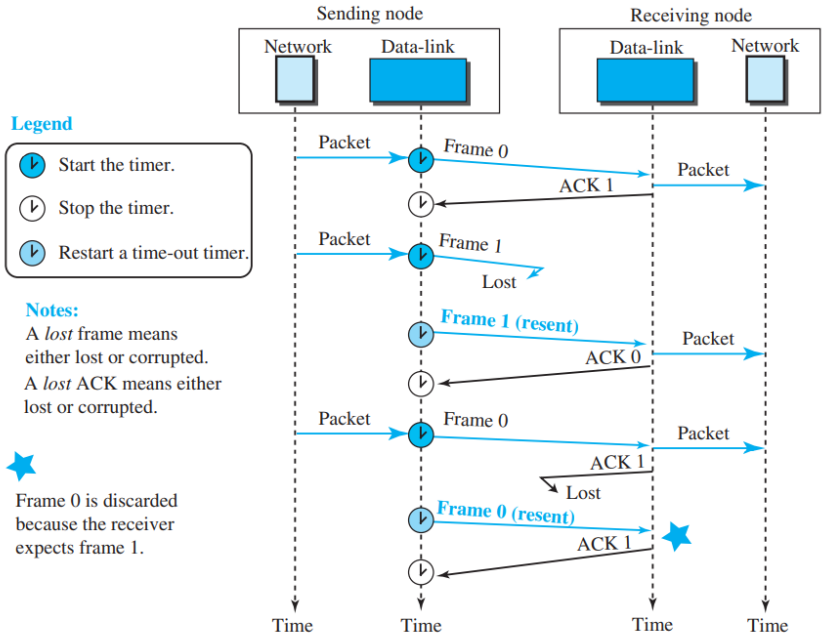


Figure 11.12 *Flow diagram for Example 11.3*



⇨ Problem and solution to stop and wait : Adding sequence numbers and acknowledgment numbers can prevent duplicates. The first frame is sent and acknowledged. The second frame is sent, but lost. After time-out, it is resent. The third frame is sent and acknowledged, but the acknowledgment is lost. The frame is resent. Sequence starts with 0,1,0,1... and acknowledgement starts with 1,0,1,0,….

Figure 11.13 *Flow diagram for Example 11.4*



HDLC : High-level Data Link Control (HDLC) is a bit-oriented protocol for communication over point-to-point and multipoint links. It implements the Stop-and-Wait protocol we discussed earlier.

HDLC provides two common transfer modes that can be used in different configurations: normal response mode (NRM) and asynchronous balanced mode (ABM). In normal response mode (NRM), the station configuration is unbalanced. We have one primary station and multiple secondary stations.

To provide the flexibility necessary to support all the options possible in the modes and configurations just described, HDLC defines three types of frames: information frames (I-frames), supervisory frames (S-frames), and unnumbered frames (U-frames).

Figure 11.14 *Normal response mode*

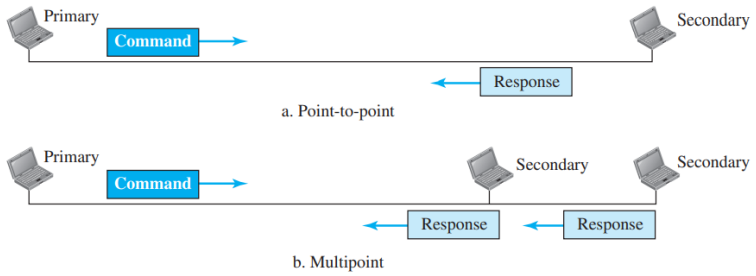


Figure 11.15 *Asynchronous balanced mode*

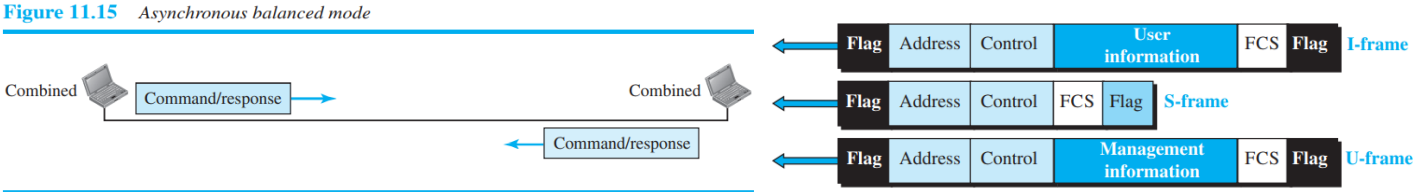
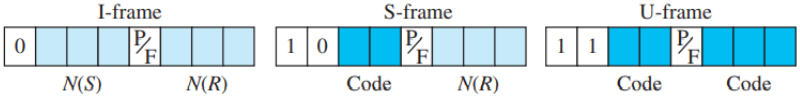


Figure 11.17 *Control field format for the different frame types*



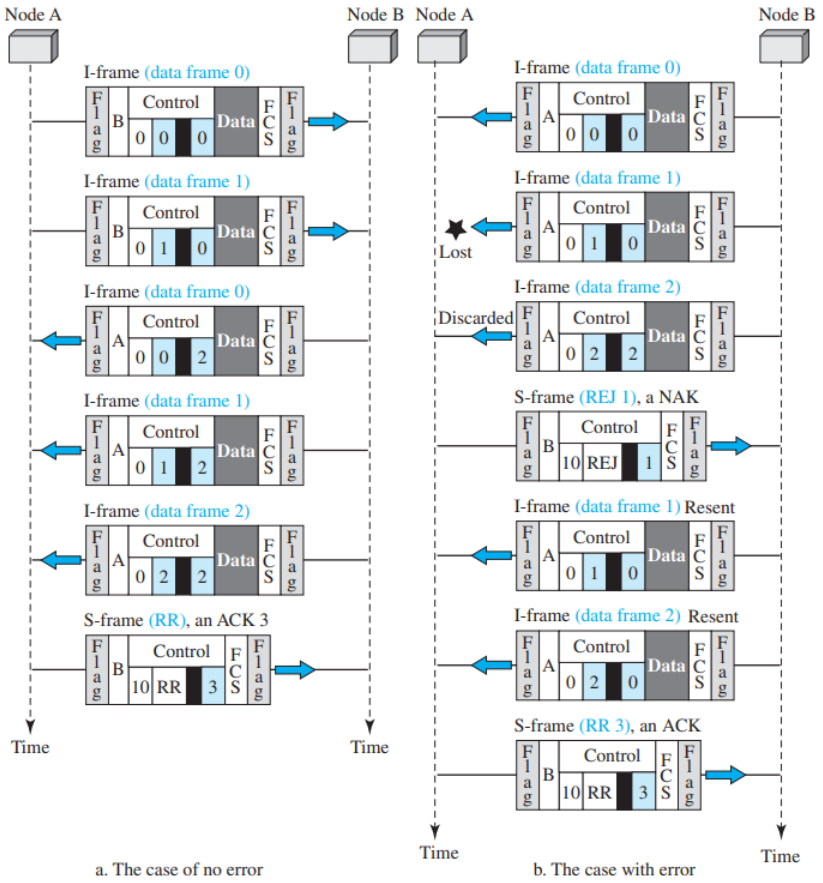
a) Control Field for I-Frames : I-frames are designed to carry user data from the network layer. In addition, they can include flow- and error-control information (piggybacking). The subfields in the control field are used to define these functions. The first bit defines the type. If the first bit of the control field is 0, this means the frame is an I-frame. The next 3 bits, called N(S), define the sequence number of the frame. Note that with 3 bits, we can define a sequence number between 0 and 7. The last 3 bits, called N(R), correspond to the acknowledgment number when piggybacking is used. The single bit between N(S) and N(R) is called the P/F bit. The P/F field is a single bit with a dual purpose. It has meaning only when it is set (bit = 1) and can mean poll or final. It means poll when the frame is sent by a primary station to a secondary (when the address field contains the address of the receiver). It means final when the frame is sent by a secondary to a primary (when the address field contains the address of the sender).

b) Control Field for S-Frames : The last 3 bits, called N(R), correspond to the acknowledgment number (ACK) or negative acknowledgment number (NAK), depending on the type of S-frame. The 2 bits called code are used to define the type of S-frame itself.
   1) Receive ready (RR). If the value of the code subfield is 00,
   2) Receive not ready (RNR). If the value of the code subfield is 10
   3) Reject (REJ). If the value of the code subfield is 01
   4) Selective reject (SREJ). If the value of the code subfield is 11,

c) Control Field : for U-Frames Unnumbered frames are used to exchange session management and control information between connected devices. Unlike S-frames, U-frames contain an information field, but one used for system management information, not user data. As with S-frames, however, much of the information carried by U-frames is contained in codes included in the control field. U-frame codes are divided into two sections: a 2-bit prefix before the P/ F bit and a 3-bit suffix after the P/F bit. Together, these two segments (5 bits) can be used to create up to 32 different types of U-frames.

The informational frames are used to carry data frames. Supervisory frames are used only to transport control information for flow and error control. Unnumbered frames are reserved for system management and provide connection-oriented service.

Figure 11.19 *Example of piggybacking with and without error*



One of the most common protocols for point-to-point access is the Point-to-Point Protocol (PPP) This belong to data link layer. PPP uses only one type of frame, but allows multiplexing of different payloads to achieve a kind of connection-oriented service authentication. Encapsulating different packets in a frame allows PPP to move to different states to provide necessary services.

MEDIA ACCESS CONTROL (MAC)

Two features give this method its name. First, there is no scheduled time for a station to transmit. Transmission is random among the stations. That is why these methods are called random access. Second, no rules specify which station should send next. Stations complete with one another to access the medium. That is why these methods are also called contention methods.

- ⇨ In a random-access method, each station has the right to the medium without being controlled by any other station. However, if more than one station tries to send, there is an access conflict—collision—and the frames will be either destroyed or modified.

each station follows a procedure that answers the following questions:

- ❑ When can the station access the medium?
- ❑ What can the station do if the medium is busy?
- ❑ How can the station determine the success or failure of the transmission?
- ❑ What can the station do if there is an access conflict?

1) **ALOHA** : ALOHA, the earliest random-access method, was developed at the University of Hawaii in early 1970. It was designed for a radio (wireless) LAN, but it can be used on any shared medium.

Important terms :

- Pure ALOHA dictates that when the time-out period passes, each station waits a random amount of time before resending its frame. The randomness will help avoid more collisions. We call this time the **backoff time TB**.
- Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames. After a maximum number of retransmission attempts Kmax, a station must give up and try later.
- **Vulnerable time** : Let us find the vulnerable time, the length of time in which there is a possibility of collision.
- Throughput : Let us call G the average number of frames generated by **the system** (includes all host) in one frame transmission time. i.e. frames generated in Tfr.

a) Pure ALOHA :

Figure 12.3 *Procedure for pure ALOHA protocol*

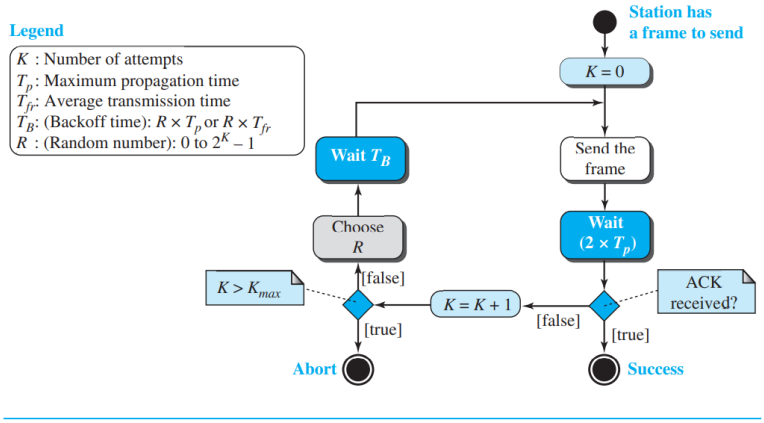
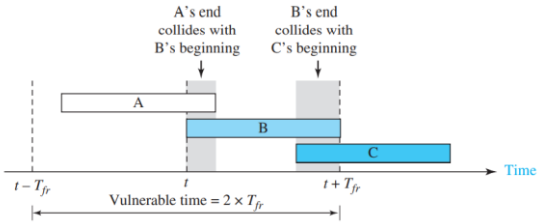


Figure 12.4 *Vulnerable time for pure ALOHA protocol*



Example 12.1

The stations on a wireless ALOHA network are a maximum of 600 km apart. If we assume that signals propagate at 3 × 108 m/s, we find Tp = (600 × 103) / (3 × 108) = 2 ms. For K = 2, the range of R is {0, 1, 2, 3}. This means that TB can be 0, 2, 4, or 6 ms, based on the outcome of the random variable R.

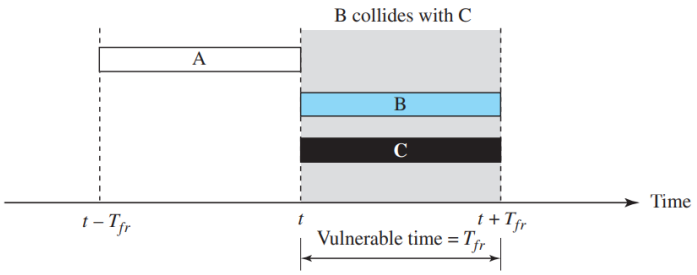
ALOHA vulnerable time = 2 x Tfr, S = average number of successful transmissions.

The throughput for pure ALOHA is $S = G \times e^{-2G}$.
The maximum throughput $S_{max} = 1/(2e) = 0.184$ when $G = (1/2)$.

b) Slotted ALOHA : Pure ALOHA has a vulnerable time of 2 × Tfr. This is so because there is no rule that defines when the station can send. A station may send soon after another station has started or just before another station has finished. Slotted ALOHA was invented to improve the efficiency of pure ALOHA. In slotted ALOHA we divide the time into slots of Tfr seconds and force the station to send only at the beginning of the time slot.

Slotted ALOHA vulnerable time = Tfr

Figure 12.6 *Vulnerable time for slotted ALOHA protocol*



The throughput for slotted ALOHA is $S = G \times e^{-G}$.
The maximum throughput $S_{max} = 0.368$ when $G = 1$.

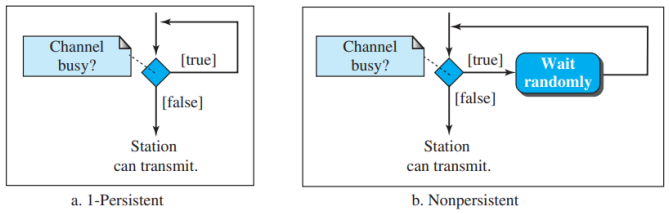
Example : A group of N stations share a 100 Kbps pure ALOHA channel. Each station outputs a 1000 byte frame on average once every 50 seconds. The maximum value of N is _____.
Answer : Total data produced by N station in one second = N x 1000 x 8 /50. Now, with pure ALOHA useful bandwidth is 0.184 x 100kbps which should be equal to data produced by N network so by equating we get 115.

2) CSMA (Carrier sense multiple access) : sense before transmit

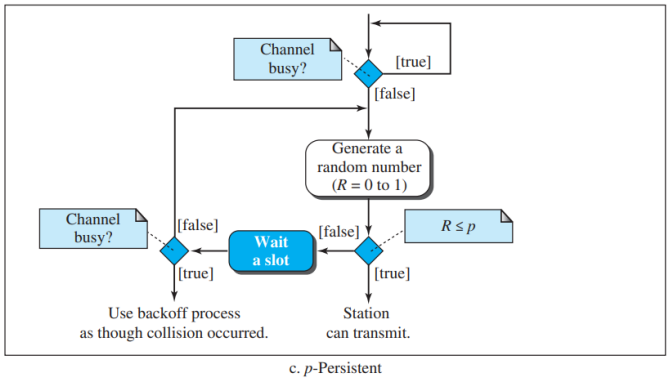
The vulnerable time for CSMA is the propagation time Tp. This is the time needed for a signal to propagate from one end of the medium to the other.

Persistence Methods : What should a station do if the channel is busy? What should a station do if the channel is idle? Three methods have been devised to answer these questions: the 1-persistent method, the nonpersistent method, and the p-persistent method.

Figure 12.10 *Flow diagram for three persistence methods*

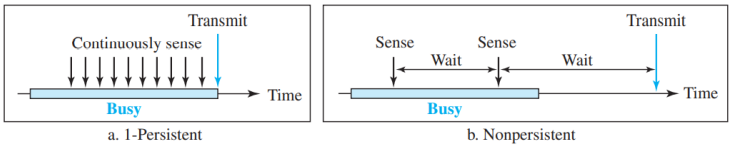


a. 1-Persistent b. Nonpersistent

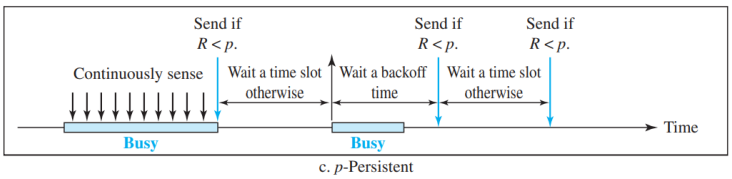


c. *p*-Persistent

Figure 12.9 *Behavior of three persistence methods*



a. 1-Persistent b. Nonpersistent



c. *p*-Persistent

Example : In p persistent CSMA network there are 6 system in a slot. The probability of each station sending the data is 0.7 and to avoid collision only 2 stations should transmit the data at a time. What is the probability that channel is collision free?
Answer : This question was asked in test series your analogy is correct but remember that only 2 station should transmit the data at a time means at most 2 stations not exactly 2 station. So you have to consider 0, 1, 2 all cases. Apart from that everything was correct.

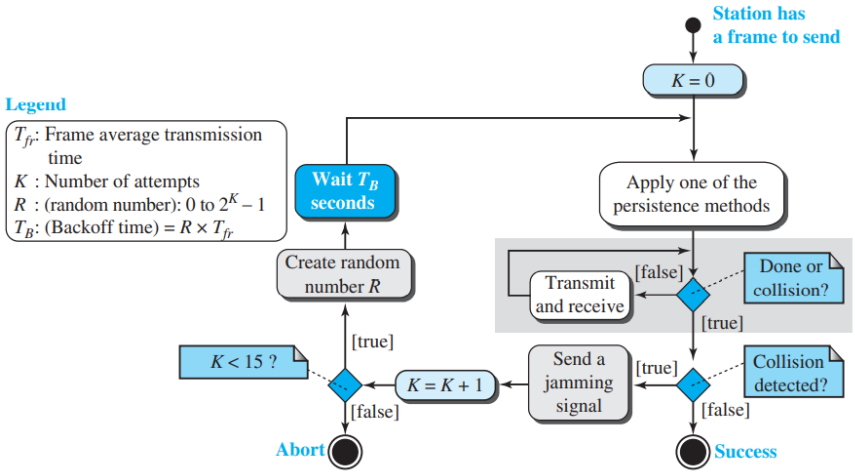
3) **CSMA/CD** : The CSMA method does not specify the procedure following a collision. Carrier sense multiple access with collision detection (CSMA/CD) augments the algorithm to handle the collision. In this method, a station monitors the medium after it sends a frame to see if the transmission was successful. If so, the station is finished. If, however, there is a collision, the frame is sent again.

the frame transmission time Tfr must be at least two times the maximum propagation time Tp.

Example 12.5 A network using CSMA/CD has a bandwidth of 10 Mbps. If the maximum propagation time (including the delays in the devices and ignoring the time needed to send a jamming signal, as we see later) is 25.6 μs, what is the minimum size of the frame?

Solution The minimum frame transmission time is Tfr = 2 × Tp = 51.2 μs. This means, in the worst case, a station needs to transmit for a period of 51.2 μs to detect the collision. The minimum size of the frame is 10 Mbps × 51.2 μs = 512 bits or 64 bytes. This is actually the minimum size of the frame for Standard Ethernet, as we will see later in the chapter.

Figure 12.13 *Flow diagram for the CSMA/CD*



if n collision then 2^n-1 possibility.

Value of random number depend upon 1) if Host A and host B collide and if A first sends message then it will have random number 0 and if host B send frame after receiving frame A then it will have random number 1.

short jamming signal to make sure that all other stations become aware of the collision.

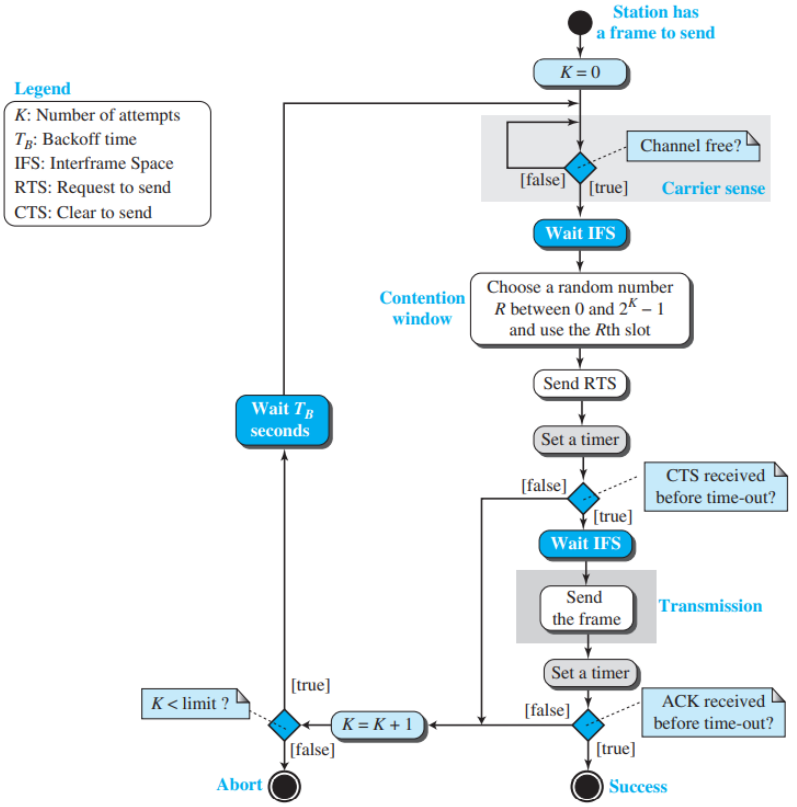
Energy Level : We can say that the level of energy in a channel can have three values: zero, normal, and abnormal. At the zero level, the channel is idle. At the normal level, a station has successfully captured the channel and is sending its frame. At the abnormal level, there is a collision and the level of the energy is twice the normal level. A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode.

Throughput : The throughput of CSMA/CD is greater than that of pure or slotted ALOHA. The maximum throughput occurs at a different value of G and is based on the persistence method and the value of p in the p-persistent approach. For the 1-persistent method, the maximum throughput is around 50 percent when G = 1. For the nonpersistent method, the maximum throughput can go up to 90 percent when G is between 3 and 8.

4) **CSMA/CA** : Carrier sense multiple access with collision avoidance (CSMA/CA) was invented for wireless networks. Collisions are avoided through the use of CSMA/CA's three strategies: the interframe space, the contention window, and acknowledgments.

- When an idle channel is found, the station does not send immediately. It waits for a period of time called the **interframe space** or IFS.
- The contention window is an amount of time divided into slots. A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential backoff strategy. This means that it is set to one slot the first time and then doubles each time the station cannot detect an idle channel after the IFS time
- With all these precautions, there still may be a collision resulting in destroyed data. In addition, the data may be corrupted during the transmission. The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

Figure 12.15 *Flow diagram of CSMA/CA*



.16 *Contention window*

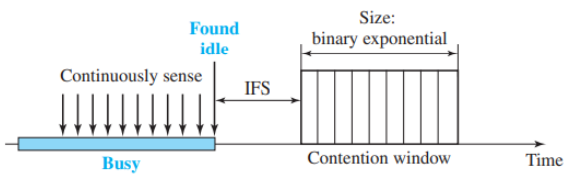
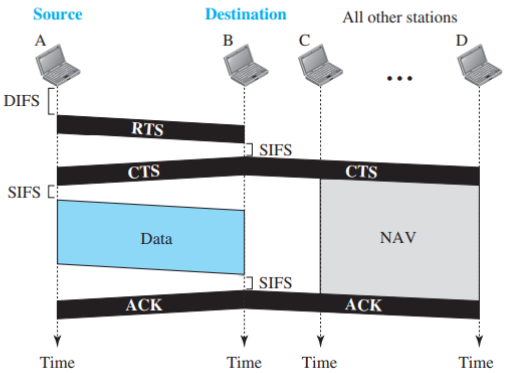


Figure 12.17 *CSMA/CA and NAV*



DIFS : DCF interframe space, SIFS : short interframe space, RTS : Ready to send, CTS : Clear to send, NAV : Network allocation vector = When a station sends an RTS frame, it includes the duration of time that it needs to occupy the channel. The stations that are affected by this transmission create a timer called a **network allocation vector** (NAV) that shows how much time must pass before these stations are allowed to check the channel for idleness. Each time a station accesses the system and sends an RTS frame, other stations start their NAV.

⇨ Controlled Access : In controlled access, the stations consult one another to find which station has the right to send. A station cannot send unless it has been authorized by other stations.

1) Reservation : In the reservation method, a station needs to make a reservation before sending data. Time is divided into intervals. In each interval, a reservation frame precedes the data frames sent in that interval. If there are N stations in the system, there are exactly N reservation

minislots in the reservation frame. Each minislot belongs to a station. When a station needs to send a data frame, it makes a reservation in its own minislot. The stations that have made reservations can send their data frames after the reservation frame.

Figure 12.18 *Reservation access method*

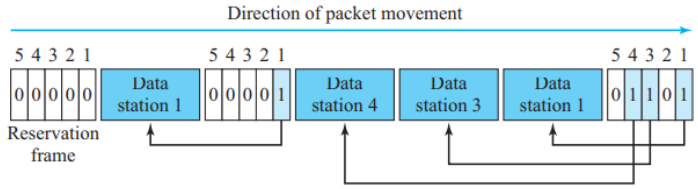
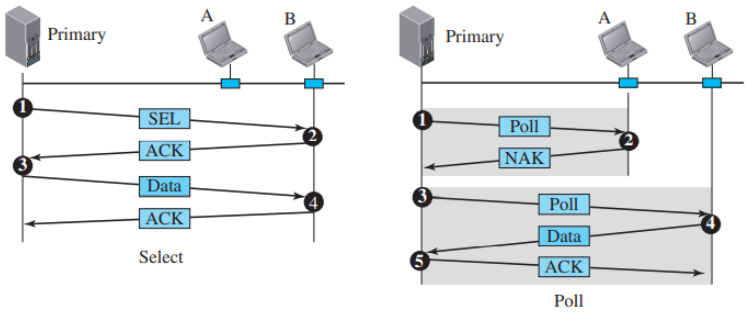


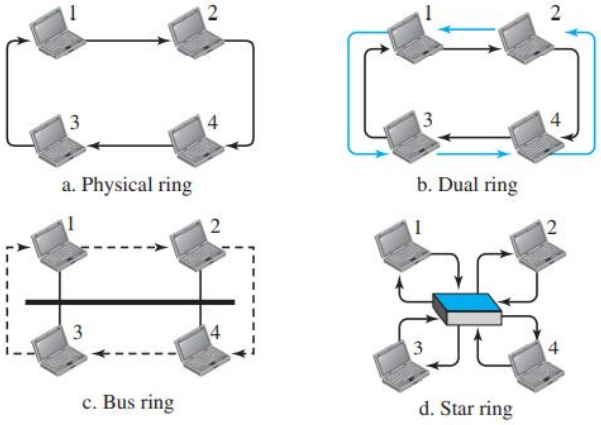
Figure 12.19 *Select and poll functions in polling-access method*



2) Polling : Polling works with topologies in which one device is designated as a primary station and the other devices are secondary stations. All data exchanges must be made through the primary device even when the ultimate destination is a secondary device. The primary device controls the link; the secondary devices follow its instructions.
The select function is used whenever the primary device has something to send.
The poll function is used by the primary device to solicit transmissions from the secondary devices.

3) Token Passing : In the token-passing method, the stations in a network are organized in a logical ring. In other words, for each station, there is a predecessor and a successor. In this method, a special packet called a **token** circulates through the ring. The possession of the token gives the station the right to access the channel and send its data.

Figure 12.20 *Logical ring and physical topology in token-passing access method*



a) Physical Ring : pre-processor doesn't have address of next successor, if the medium fails, the whole system fails.

b) Dual ring : uses a second (auxiliary) ring which operates in the reverse direction compared with the main ring. The second ring is for emergencies only, The high-speed Token Ring

networks called FDDI (Fiber Distributed Data Interface) and CDDI (Copper Distributed Data Interface) use this topology.

c) Bus ring : They, however, make a logical ring, because each station knows the address of its successor. The Token Bus LAN, standardized by IEEE, uses this topology.

d) Star ring : adding and removing stations from the ring is easier. This topology is still used in the Token Ring LAN designed by IBM.

Channelization (or channel partition, as it is sometimes called) is a multiple-access method in which the available bandwidth of a link is shared in time, frequency, or through code, among different stations.

Figure 12.21 *Frequency-division multiple access (FDMA)*

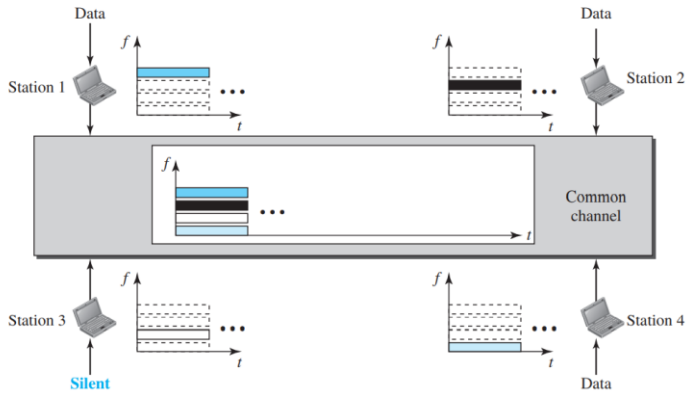


Figure 12.22 *Time-division multiple access (TDMA)*

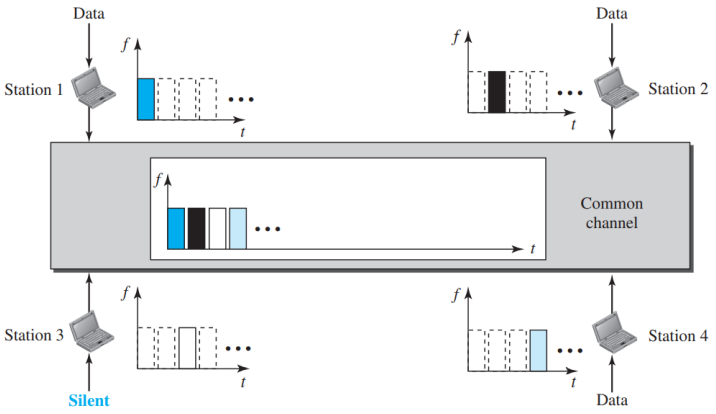
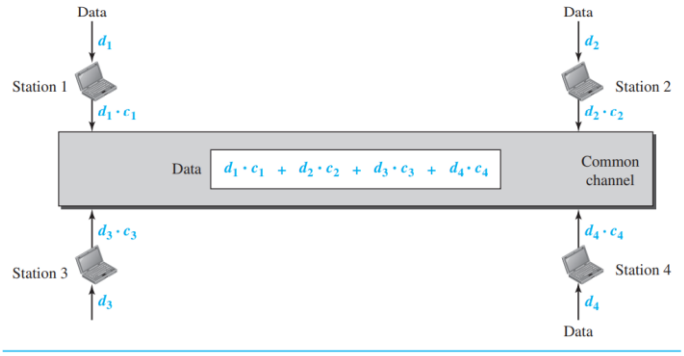
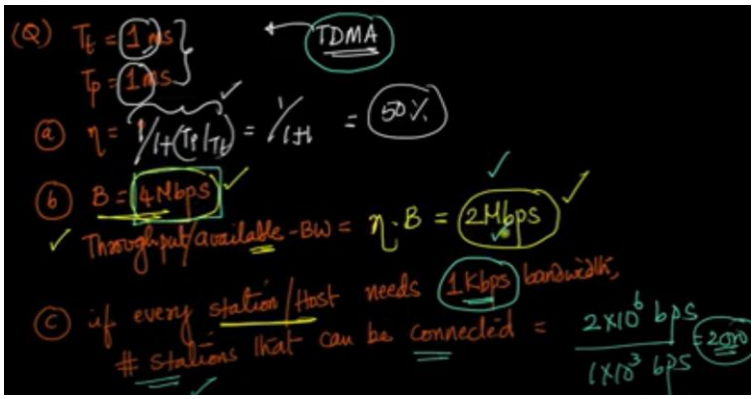


Figure 12.23 *Simple idea of communication with code*



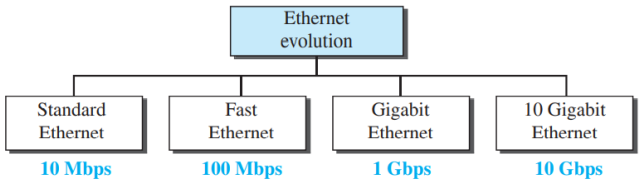
$$\text{data} = (d_1 \cdot c_1 + d_2 \cdot c_2 + d_3 \cdot c_3 + d_4 \cdot c_4) \cdot c_1$$
$$= d_1 \cdot c_1 \cdot c_1 + d_2 \cdot c_2 \cdot c_1 + d_3 \cdot c_3 \cdot c_1 + d_4 \cdot c_4 \cdot c_1 = 4 \times d_1$$



- **Ethernet Protocol :**

IEEE Project 802 : Before we discuss the Ethernet protocol and all its generations, we need to briefly discuss the IEEE standard that we often encounter in text or real life. In 1985, the Computer Society of the IEEE started a project, called Project 802, to set standards to enable intercommunication among equipment from a variety of manufacturers. Project 802 does not seek to replace any part of the OSI model or TCP/IP protocol suite. Instead, it is a way of specifying functions of the physical layer and the data-link layer of major LAN protocols.

Figure 13.2 *Ethernet evolution through four generations*



- **Connecting devices :** Connecting devices can operate in different layers of the Internet model. We discuss three kinds of connecting devices: hubs, link-layer switches, and routers. Hubs today operate in the first layer of the Internet model. Link-layer switches operate in the first two layers. Routers operate in the first three layers.

Hub : A hub is a device that operates only in the physical layer. Signals that carry information within a network can travel a fixed distance before attenuation endangers the integrity of the data. A repeater receives a signal and, before it becomes too weak or corrupted, regenerates and retimes the original bit pattern. The repeater then sends the refreshed signal.

Today, however, Ethernet LANs use star topology. In a star topology, a repeater is a multiport device, often called a hub, that can be used to serve as the connecting point and at the same time function as a repeater.

Bridged Ethernet : The first step in the Ethernet evolution was the division of a LAN by bridges. Bridges have two effects on an Ethernet LAN: They raise the bandwidth hence throughput and they separate collision domains. Hub on the otherhand is not collision domain separated device.

Link Layer device (Switch) : One may ask what the difference in functionality is between a link-layer switch and a hub. A link-layer switch has filtering capability. It can check the destination address of a frame and can decide from which outgoing port the frame should be sent. the link-layer switch can check the MAC addresses (source and destination) contained in the frame. A link-layer switch does not change the link-layer (MAC) addresses in a frame. If switch receives a frame that contains a source MAC address which doesn't match with source MAC addresses present in given MAC address table so switch will perform following actions : (i) Switch adds the source MAC address and port number in MAC address table. (ii) If destination address is not known, the switch broadcasts the frame out all port except the port through which the frame received.

A transparent switch is a switch in which the stations are completely unaware of the switch's existence. If a switch is added or deleted from the system, reconfiguration of the stations is unnecessary.

Loop Problem : Transparent switches work fine as long as there are no redundant switches in the system. Systems administrators, however, like to have redundant switches (more than one switch between a pair of LANs) to make the system more reliable. If a switch fails, another switch takes over until the failed one is repaired or replaced. Redundancy can create loops in the system, which is very undesirable.

Spanning Tree Algorithm : To solve the looping problem, the IEEE specification requires that switches use the spanning tree algorithm to create a loopless topology. In a switched LAN, this means creating a topology in which each LAN can be reached from any other LAN through one path only (no loop). We cannot change the physical topology of the system because of physical

connections between cables and switches, but we can create a logical topology that overlays the physical one.

A ROUTER is a three-layer device; it operates in the physical, data-link, and network layers. As a physical-layer device, it regenerates the signal it receives. As a link-layer device, the router checks the physical addresses (source and destination) contained in the packet. As a network-layer device, a router checks the network-layer addresses. A router changes the link-layer addresses in a packet.

A **FIREWALL** is a device (usually a router or a computer) installed between the internal network of an organization and the rest of the Internet. For example, a firewall may filter all incoming packets destined for a specific host or a specific server such as HTTP. A firewall can be used to deny access to a specific host or a specific service in the organization. A firewall is usually classified as a **packet-filter firewall** or a **proxy-based firewall**. A packet-filter firewall filters at the network or transport layer. A proxy firewall filters at the application layer.

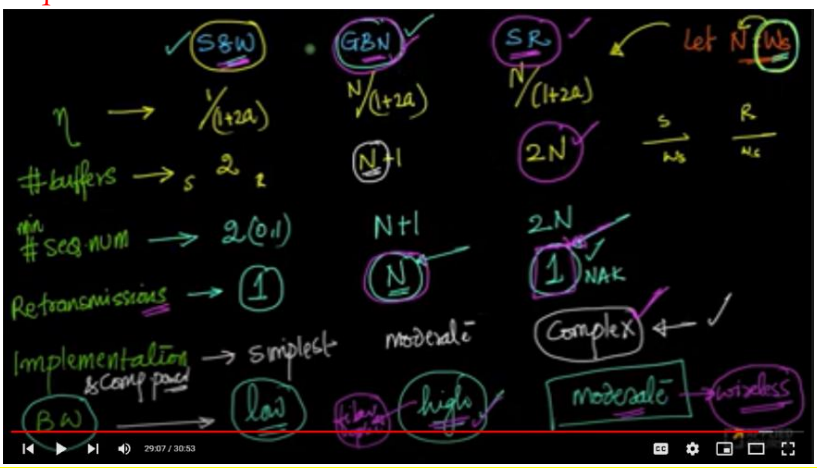
The packet-filter firewall is based on the information available in the network layer and transport layer headers (IP and TCP/UDP). However, sometimes we need to filter a message based on the information available in the message itself (at the application layer). As an example, assume that an organization wants to implement the following policies regarding its web pages: only those Internet users who have previously established business relations with the company can have access; access to other users must be blocked. In this case, a packet-filter firewall is not feasible because it cannot distinguish between different packets arriving at TCP port 80 (HTTP). Testing must be done at the application level (using URLs).

- Businesses with a public website that customers use must make their web server accessible from the internet. Doing so means putting their entire internal network at high risk. To prevent this, an organization could pay a hosting firm to host the website or their public servers on a firewall, but this would affect performance. So instead, the public servers are hosted on a network that is separate and isolated.
- A DMZ (Demilitarized) network provides a buffer between the internet and an organization's private network. The DMZ is isolated by a security gateway, such as a firewall, that filters traffic between the DMZ and a LAN. The default DMZ server is protected by another security gateway that filters traffic coming in from external networks.
- It is ideally located between two firewalls, and the DMZ firewall setup ensures incoming network packets are observed by a firewall—or other security tools—before they make it through to the servers hosted in the DMZ. This means that even if a sophisticated attacker is able to get past the first firewall, they must also access the hardened services in the DMZ before they can do damage to a business.
- If an attacker is able to penetrate the external firewall and compromise a system in the DMZ, they then also have to get past an internal firewall before gaining access to sensitive corporate data. A highly skilled bad actor may well be able to breach a secure DMZ, but the resources within it should sound alarms that provide plenty of warning that a breach is in progress.

Note :

1) The number of bits used for addressing giga Ethernet is 48 bits.
2) Separately calculate transmission time.

3) In 3G network, W-CDMA is also known as UMTS. The minimum spectrum allocation required for W-CDMA is 5MHz.



4)

5) Remember in link utilization do not put your answer in terms of percentage if not mentioned and link utilization formula also include window size which is nothing but sender window size not sender + receiver window size. So, if sequence number is given do not forget to subtract 1 in case of GobackN and divide by half in case of selective repeat.

Fairness And Utilization As Numbers:

Node throughputs: x_i - data bits transmitted by the i-th node in T units of time $(i = 0, 1, \ldots, n)$

Channel capacity: C - maximal number of bits transmittable in a unit of time.

Utilization: $U = \dfrac{x_0 + x_1 + \cdots + x_n}{C \times T} \quad (0 \leq U \leq 1)$

Fairness: $F = \dfrac{(x_0 + x_1 + \cdots + x_n)^2}{n(x_0^2 + x_1^2 + \cdots + x_n^2)} - \dfrac{1}{n} \quad (0 \leq F \leq 1)$

effici. >= Ws/(1+2Tp/Tt)

Questions on data-link layer :

1) In Ethernet when Manchester encoding is used, the bit rate is:

Half the baud rate. *Same as the baud rate.*

Twice the baud rate. *None of the above*

Answer : Basically, there is a relation between bit rate and baud rate which is:

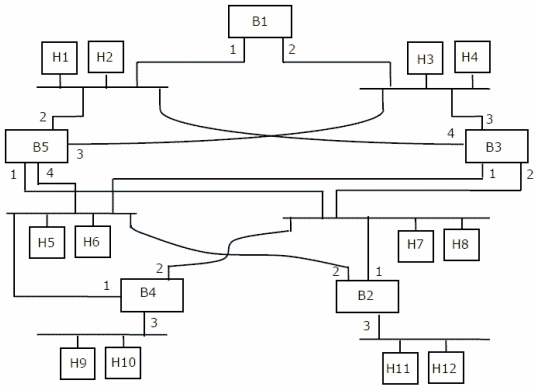
BaudRate = Bitrate * c * (1/r)

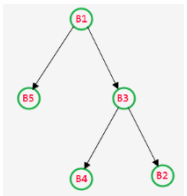
c = Case factor usually = 1 (if nothing mentioned in question)

r = data element per signal element,

In Manchester signal or differential Manchester, a data bit is encoded in two signal elements so r = 1/2 because for half duration level is up and for half of bit duration its below so two signal element per data element.

2) Consider the diagram shown below where a number of LANs are connected by (transparent) bridges. In order to avoid packets looping through circuits in the graph, the bridges organize themselves in a spanning tree. First, the root bridge is identified as the bridge with the least serial number. Next, the root sends out (one or more) data units to enable the setting up of the spanning tree of shortest paths from the root bridge to each bridge. Each bridge identifies a port (the root port) through which it will forward frames to the root bridge. Port conflicts are always resolved in favour of the port with the lower index value. When there is a possibility of multiple bridges forwarding to the same LAN (but not through the root port), ties are broken as follows: bridges closest to the root get preference and between such bridges, the one with the lowest serial number is preferred.





For the given connection of LANs by bridges, which one of the following choices represents the depth first traversal of the spanning tree of bridges?

Answer : Quick info out of this question. Self-learning bridges forward frames to all interface except incoming interface.

Basic idea :

- Define a root bridge having smallest id (all of its interfaces are active)
- Each bridge computes the shortest path to the root bridge, and notes this port (root port)
- For LAN segment, determine which bridge provides shortest path to root (designated bridge). Port connecting LAN segment to bridge is called designated port.
- Disable all port that is not a root port or a designated port.

Now back to our problem :

First we select B1 as it as smallest id as root. Then we have two choices B5 and B3 we choose B3 because smallest id. Now first we assign root ports. Talking about B3 so there are two closed path out of which 3 no. path is chosen. Similar for B5 there are two path out of which no.2 is chosen as root port. Now talking about B2 so there are again two path one through B3 and other through B5. We choose smallest id so no.1 of the B2 is chosen as root port. Now for B4 you cannot choose no.2 because it is chosen by B3 so you choose no.1 as root port.

By looking at the definition of designated port. In selecting designated port we select host and then we find shortest path to root bridge. and now everything is simple. Port 1 and 2 of the B1 becomes designated port. And so on you can do others too.

3) Let host H1 send out a broadcast ping packet. Which of the following options represents the correct forwarding table on B3 ?

a.

Hosts	Port
H1, H2, H3, H4	3
H5, H6, H9, H10	1
H7, H8, H11, H12	2

b.

Hosts	Port
H1, H2	4
H3, H4	3
H5, H6	1
H7, H8, H9, H10, H11, H12	2

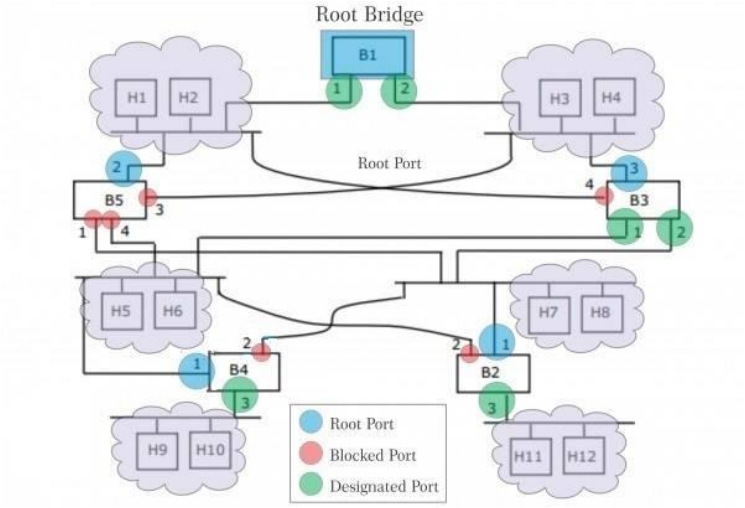
c.

Hosts	Port
H3, H4	3
H5, H6, H9, H10	1
H1, H2	4
H7, H8, H11, H12	2

d.

Hosts	Port
H1, H2, H3, H4	3
H5, H7, H9, H10	1
H7, H8, H11, H12	4

Answer : For full explanation watch this…https://www.youtube.com/embed/JgApAnUQ1Ss



4) Assume 10 Mbps ethernet and two stations A and B on its same segment. The RTT between two nodes is 650 bit times. A and B start transmitting frame and collision occurs and sends 30 bit times jam signal. Find the time at which both nodes A and B sense an idle channel (in µsec) _____.

Answer : Your answer was 355/10Mbps which is correct but some little modification. You have to consider jam signals propagation delay because you have sent jam signal so it will also take time to reach at other station. So correct answer should be 355+325 = 680/10 = 68µsec.

5) Which of the following statements is TRUE about CSMA/CD:

A. IEEE 802.11 wireless LAN runs CSMA/CD protocol

B. Ethernet is not based on CSMA/CD protocol

C. CSMA/CD is not suitable for a high propagation delay network like satellite network

D. There is no contention in a CSMA/CD network

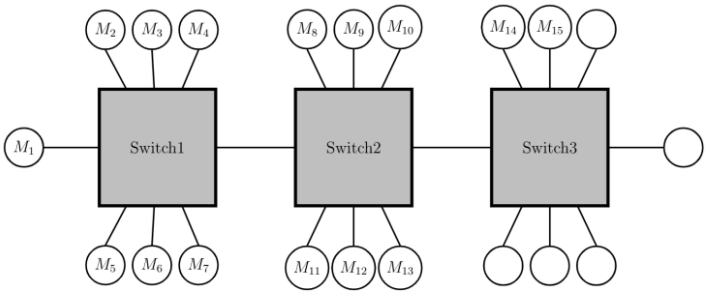
Answer : CSMA/CD was used in early days, 802.3 not in 802.11. There will be contention in this protocol. Ethernet is based on csma/cd early in 1980.

6) What is the distance of the following code : 000000, 010101, 011001, 000111, 111111.

Answer : Hamming distance is the minimum hamming distance between any two code which occurs between code 2 and 3 which is 2. Remember minimum.

7) Consider that 15 machines need to be connected in a LAN using 8 port Ethernet switches. Assume that these switches do not have any separate uplink ports. The minimum number of switches needed is

Answer : 3 switches



8) 1000 airline reservation stations are competing for the use of a single slotted ALOHA channel. The average station makes 36 requests per hour. A slot is 100 μsec. What is the approximate total channel load?

Answer : Each terminal makes one request every 3600 sec / 36 request = 100 sec.

Total load is 1000 requests per 100 sec or 10 requests per sec.

There are 1 sec/100 μsec = 1000000 μsec / 100 μsec = 10000 slots in one second.

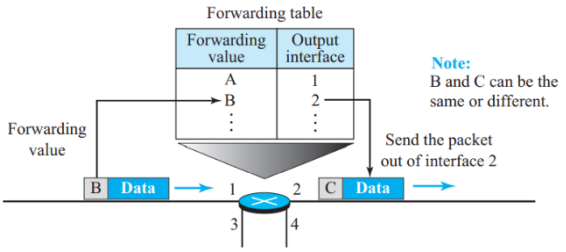
Hence, G=10/10000=1/1000=0.1%

NETWORK LAYER

Services provided by network layer :

- Packetizing : The first duty of the network layer is definitely packetizing: encapsulating the payload (data received from upper layer) in a network-layer packet at the source and decapsulating the payload from the network-layer packet at the destination.
- Routing : The network layer is responsible for routing the packet from its source to the destination. A physical network is a combination of networks (LANs and WANs) and routers that connect them. This means that there is more than one route from the source to the destination. The network layer is responsible for finding the best one among these possible routes. The network layer needs to have some specific strategies for defining the best route.
- Forwarding : If routing is applying strategies and running some routing protocols to create the decision-making tables for each router, forwarding can be defined as the action applied by each router when a packet arrives at one of its interfaces. The decision-making table a router normally uses for applying this action is sometimes called the forwarding table and sometimes the routing table.
- Error control : Although error control also can be implemented in the network layer, the designers of the network layer in the Internet ignored this issue for the data being carried by the network layer. The designers of the network layer, however, have added a checksum field to the datagram to control any corruption in the header, but not in the whole datagram. This checksum may prevent any changes or corruptions in the header of the datagram.

Figure 18.2 *Forwarding process*



- Flow control : Flow control regulates the amount of data a source can send without overwhelming the receiver. If the upper layer at the source computer produces data faster than the upper layer at the destination computer can consume it, the receiver will be overwhelmed with data. To control the flow of data, the receiver needs to send some feedback to the sender to inform the latter that it is overwhelmed with data. However, NL(Network layer does not use any flow control mechanism). IP does not guarantee that all data will be delivered or that the data that are delivered will arrive in the proper order. That work is done by transport layer.
- Congestion control : Congestion may occur if the number of datagrams sent by source computers is beyond the capacity of the network or routers. In this situation, some routers may drop some of the datagrams. However, as more datagrams are dropped, the situation may become worse because, due to the error control mechanism at the upper layers, the sender may send duplicates of the lost packets.
- Quality of Service : As the Internet has allowed new applications such as multimedia communication (in particular real-time communication of audio and video), the quality of service (QoS) of the communication has become more and more important.
- Security : Another issue related to communication at the network layer is security. Security was not a concern when the Internet was originally designed because it was used by a small

number of users at universities for research activities; other people had no access to the Internet.

Circuit Switching :

In this approach, there is a dedicated route between sender and receiver. Before the link is determined in the circuit switching approach, the dedicated route will continue until the connection is eliminated. It is reliable and slow. In circuit switching, ***Time for setup and Teardown = 3*** *x* ***(Propagation time + Transmission Time)***

Message Switching :

Message Switching is an approach in which a message is sent as a whole unit and routed by the intermediate hub at which it is saved and delivered. There is no installation of a dedicated route between the sender and receiver in the message switching approach.

The destination location is added to the message. It supports flexible routing as the message is routed by the intermediate hub based on its data.

Packet Switching :

Packet switching is a switching approach in which the message is transmitted in one go, but it is split into the lower item, and they are sent separately. The message divided into lower elements are called packets, and these packets are provided with a specific number to recognize their series at the receiving end.

Although in data communication switching techniques are divided into two broad categories, circuit switching and packet switching, only packet switching is used at the network layer because the unit of data at this layer is a packet. Circuit switching is mostly used at the physical layer; the electrical switch mentioned earlier is a kind of circuit switch. Switches in a circuit-switched network process connection establishment and tear-down messages, whereas switches in a packet-switched network do not. That's why Circuit switching uses more resources than packet switching.

we infer that a kind of switching occurs at the network layer. A router, in fact, is a switch that creates a connection between an input port and an output port (or a set of output ports), just as an electrical switch connects the input to the output to let electricity flow. Today, a packet-switched network can use two different approaches to route the packets: the datagram approach and the virtual circuit approach. There is also a cut-through switching which forwards packets without any error checking.

In computer networking, **source routing**, also called **path addressing**, allows a sender of a packet to partially or completely specify the route of the packet takes through the network. In contrast, in **non-source routing protocols**, routers in the network determine the path based on the packet's destination.

1) Datagram Approach: Connectionless Service

Figure 18.3 *A connectionless packet-switched network*

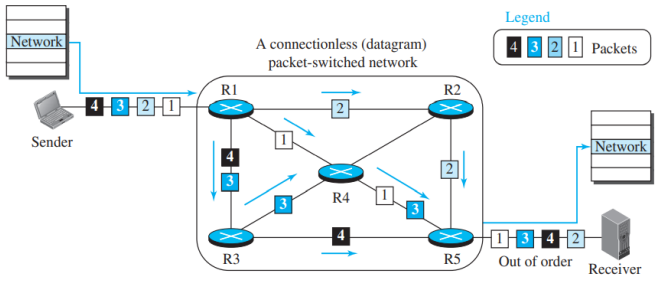
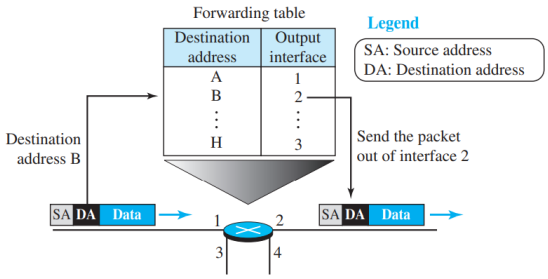


Figure 18.4 *Forwarding process in a router when used in a connectionless network*



the network-layer protocol treats each packet independently, with each packet having no relationship to any other packet. **In the datagram approach, the forwarding decision is based on the destination address of the packet.**

2) Virtual-Circuit Approach: Connection-Oriented Service

Figure 18.5 *A virtual-circuit packet-switched network*

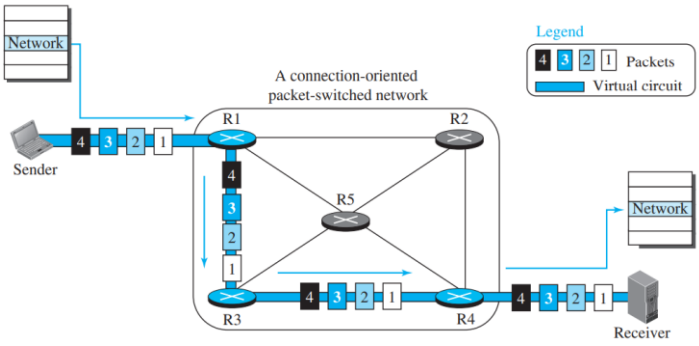
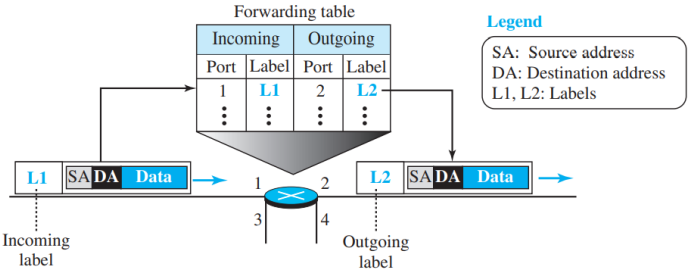


Figure 18.6 *Forwarding process in a router when used in a virtual-circuit network*



To create a connection-oriented service, a three-phase process is used: setup, data transfer, and teardown. In the setup phase, the source and destination addresses of the sender and receiver are used to make table entries for the connection-oriented service. In the teardown phase, the source and destination inform the router to delete the corresponding entries. Data transfer occurs between these two phases. **In the virtual-circuit approach, the forwarding decision is based on the label of the packet.**

Figure 18.7 *Sending request packet in a virtual-circuit network*

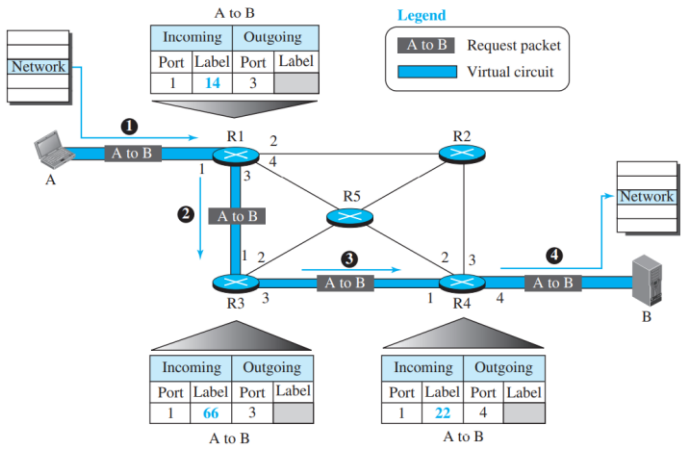
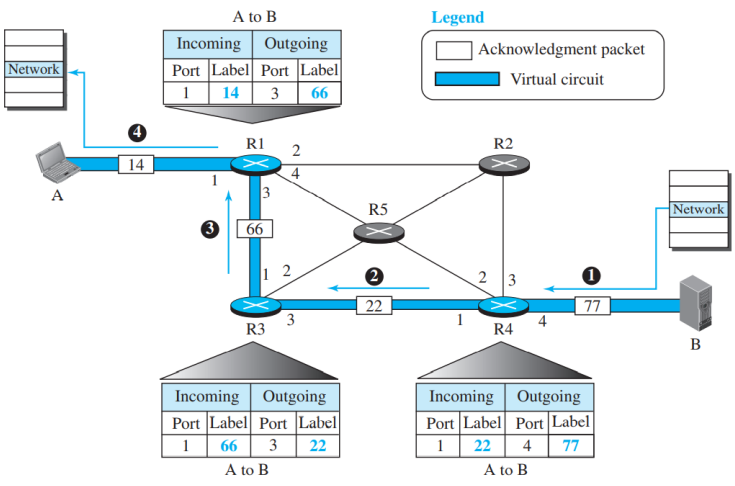


Figure 18.8 *Sending acknowledgments in a virtual-circuit network*



Teardown Phase : In the teardown phase, source A, after sending all packets to B, sends a special packet called a teardown packet. Destination B responds with a confirmation packet. All routers delete the corresponding entries from their tables.

NETWORK-LAYER PERFORMANCE : The performance of a network can be measured in terms of delay, throughput, and packet loss. Congestion control is an issue that can improve the performance.

1) Transmission Delay : Delay(tr) = (Packet length) / (Transmission rate)
2) Propagation Delay : Delay(pg) = (Distance) / (Propagation speed)
3) Processing Delay : The processing delay is the time required for a router or a destination host to receive a packet from its input port, remove the header, perform an error detection procedure, and deliver the packet to the output port (in the case of a router) or deliver the packet to the upper-layer protocol (in the case of the destination host).
   Delay(pr)= Time required to process a packet in a router or a destination host.
4) Queuing Delay : The queuing delay for a packet in a router is measured as the time a packet waits in the input queue and output queue of a router.
   Delay(qu) = The time a packet waits in input and output queues in a router.
5) Total Delay : Total delay = (n + 1) (Delaytr + Delaypg + Delaypr) + (n) (Delayqu)
   Note that if we have n routers also note that source and destination are not routers, we have (n + 1) links. Therefore, we have (n + 1) transmission delays related to n routers and the source, (n + 1) propagation delays related to (n + 1) links, (n + 1) processing delays related to n routers and the destination, and only n queuing delays related to n routers.

Throughput : Throughput at any point in a network is defined as the number of bits passing through the point in a second, which is actually the transmission rate of data at that point. Throughput = minimum {TR1, TR2, . . . TRn}

Figure 18.10 *Throughput in a path with three links in a series*

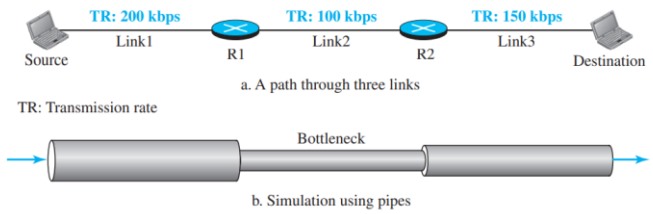
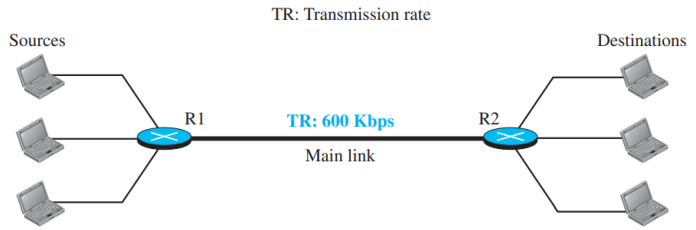


Figure 18.12 *Effect of throughput in shared links*



When the load is much less than the capacity of the network, the delay is at a minimum. This minimum delay is composed of propagation delay and processing delay, both of which are negligible. However, when the load reaches the network capacity, the delay increases sharply because we now need to add the queuing delay to the total delay. Note that the delay becomes infinite when the load is greater than the capacity.

we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal).

1) Open-Loop Congestion Control : In open-loop congestion control, policies are applied to prevent congestion before it happens. In these mechanisms, congestion control is handled by either the source or the destination.
   a) Retransmission Policy
   b) Window Policy
   c) Acknowledgment Policy
   d) Discarding Policy
   e) Admission Policy

2) Closed-Loop Congestion Control : Closed-loop congestion control mechanisms try to alleviate congestion after it happens. Several mechanisms have been used by different protocols.
   a) Backpressure : The technique of backpressure refers to a congestion control mechanism in which a congested node stops receiving data from the immediate upstream node or nodes.
   b) Choke Packet : A choke packet is a packet sent by a node to the source to inform it of congestion.
   c) Implicate signalling : The source guesses that there is congestion somewhere in the network from other symptoms.
   d) Explicit Signalling : In the explicit-signalling method, the signal is included in the packets that carry data.

IPv4 ADDRESSING :

The identifier used in the IP layer of the TCP/IP protocol suite to identify the connection of each device to the Internet is called the Internet address or IP address. An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a host or a router to the Internet.

An address space is the total number of addresses used by the protocol. If a protocol uses b bits to define an address, the address space is 2^b because each bit can have two different values (0 or 1). IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than four billion).

Figure 18.16 *Three different notations in IPv4 addressing*

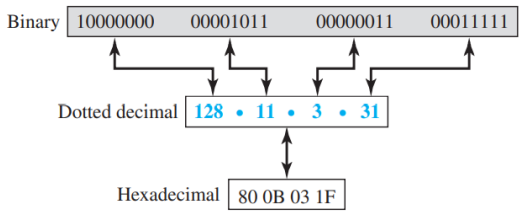
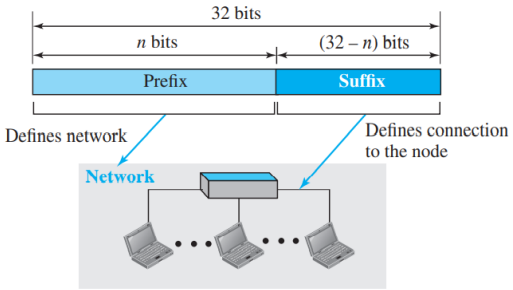
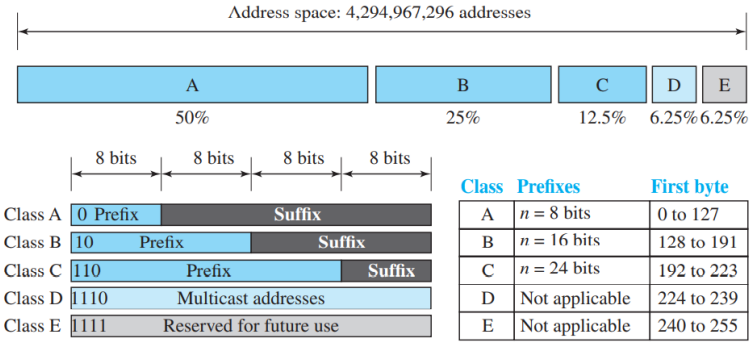


Figure 18.17 *Hierarchy in addressing*



1) Classful Addressing :

Figure 18.18 *Occupation of the address space in classful addressing*



Class	Prefixes	First byte
A	$n = 8$ bits	0 to 127
B	$n = 16$ bits	128 to 191
C	$n = 24$ bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

Class	1st octet of IP address	Default Subnet Mask	Network / Host	Number of networks	Maximum nodes in a network
A	1 - 126	255.0.0.0	N.H.H.H	126	16,777,214
B	128 - 191	255.255.0.0	N.N.H.H	16,384	65,534
C	192 - 223	255.255.255.0	N.N.N.H	2,097,152	254
D	224 - 239				
E	240 - 254				

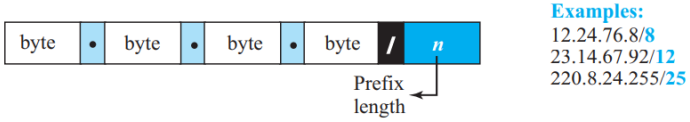
Address Depletion : The reason that classful addressing has become obsolete is address depletion. Since the addresses were not distributed properly, the Internet was faced with the problem of the addresses being rapidly used up, resulting in no more addresses available for organizations and individuals that needed to be connected to the Internet.

Subnetting and Supernetting : To alleviate address depletion, two strategies were proposed and, to some extent, implemented: subnetting and supernetting. In subnetting, a class A or class B block is divided into several subnets. Each subnet has a larger prefix length than the original network. supernetting was devised to combine several class C blocks into a larger block to be attractive to organizations that need more than the 256 addresses available in a class C block. This idea did not work either because it makes the routing of packets more difficult.

2) **Classless Addressing** : In classless addressing, variable-length blocks are used that belong to no classes. We can have a block of 1 address, 2 addresses, 4 addresses, 128 addresses, and so on

The notation is informally referred to as slash notation and formally as classless interdomain routing or CIDR (pronounced cider) strategy.

Figure 18.20 *Slash notation (CIDR)*



Examples:
12.24.76.8/8
23.14.67.92/12
220.8.24.255/25

a) The number of addresses in the block is found as $N = 2^{32-n}$.

To find the first address, we keep the n leftmost bits and set the (32 − n) rightmost bits all to 0s.

To find the last address, we keep the n leftmost bits and set the (32 − n) rightmost bits all to 1s.

b) A computer can easily find the address mask because it is the complement of $(2^{32-n} - 1)$. The address mask is a 32-bit number in which the n leftmost bits are set to 1s and the rest of the bits (32 − n) are set to 0's.

The number of addresses in the block N = NOT (mask) + 1.

The first address in the block = (Any address in the block) AND (mask).

The last address in the block = (Any address in the block) OR [(NOT (mask)].

Block Allocation : An ISP has requested a block of 1000 addresses. Since 1000 is not a power of 2, 1024 addresses are granted. The prefix length is calculated as n = 32 − log21024 = 22. An available block, 18.14.12.0/22, is granted to the ISP. It can be seen that the first address in decimal is 302,910,464, which is divisible by 1024.

first address = (prefix in decimal) × 2^{32-n} = (prefix in decimal) × N

- ⇨ **Subnetting** : More levels of hierarchy can be created using subnetting. An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a subnetwork (or subnet) the process is called subnetting.
- ⇨ *Designing Subnets* :
  Total number of addresses granted to the organization is N, the prefix length is n, the assigned number of addresses to each subnetwork is N_{sub}, prefix length for each subnetwork is n_{sub}.
  - ❑ The number of addresses in each subnetwork should be a power of 2.
  - ❑ The prefix length for each subnetwork should be found using the following formula: first address = (prefix in decimal) × 2^{32-n} = (prefix in decimal) × N, $n_{sub} = 32 - \log_2 N_{sub}$.

❑ The starting address in each subnetwork should be divisible by the number of addresses in that subnetwork. This can be achieved if we first assign addresses to larger subnetworks.

Example 18.5

An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 subblocks of addresses to use in its three subnets: one subblock of 10 addresses, one subblock of 60 addresses, and one subblock of 120 addresses. Design the subblocks.

Solution

There are $2^{32-24} = 256$ addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24. To satisfy the third requirement, we assign addresses to subblocks, starting with the largest and ending with the smallest one.

a. The number of addresses in the largest subblock, which requires 120 addresses, is not a power of 2. We allocate 128 addresses. The subnet mask for this subnet can be found as $n_1 = 32 - \log_2 128 = 25$. The first address in this block is 14.24.74.0/25; the last address is 14.24.74.127/25.

b. The number of addresses in the second largest subblock, which requires 60 addresses, is not a power of 2 either. We allocate 64 addresses. The subnet mask for this subnet can be found as $n_2 = 32 - \log_2 64 = 26$. The first address in this block is 14.24.74.128/26; the last address is 14.24.74.191/26.

c. The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2 either. We allocate 16 addresses. The subnet mask for this subnet can be found as $n_3 = 32 - \log_2 16 = 28$. The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28.

If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.208. The last address is 14.24.74.255. We don't know about the prefix length yet. Figure 18.23

Figure 18.23 *Solution to Example 18.5*

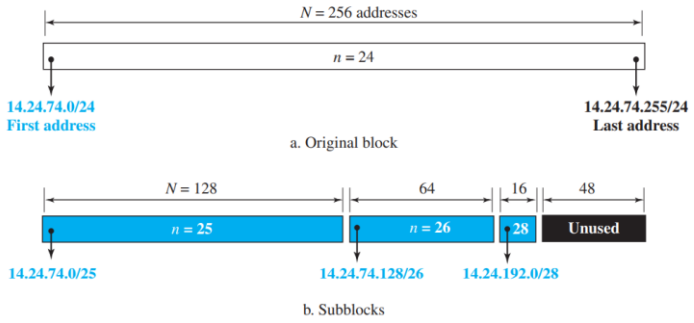
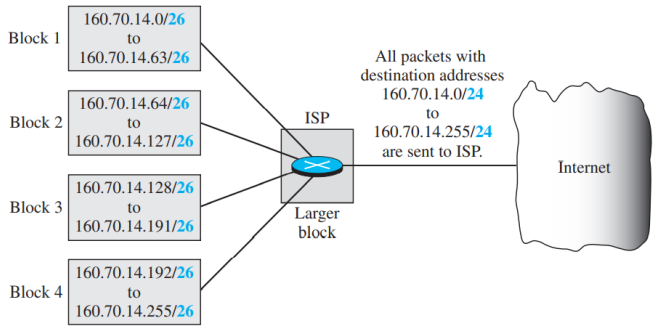


Figure 18.24 *Example of address aggregation*



Special Address :

This-host Address : The only address in the block 0.0.0.0/32 is called the this-host address. It is used whenever a host needs to send an IP datagram but it does not know its own address to use as the source address.

Limited-broadcast Address : The only address in the block 255.255.255.255/32 is called the limited-broadcast address. It is used whenever a router or a host needs to send a datagram to all devices in a network.

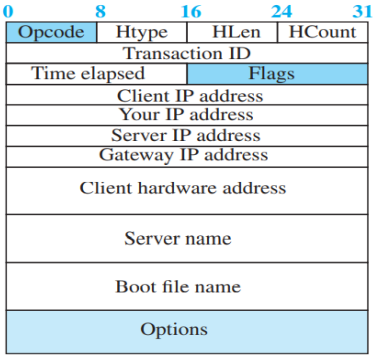
Loopback Address : The block 127.0.0.0/8 is called the loopback address. A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host.

Private Addresses : Four blocks are assigned as private addresses: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, and 169.254.0.0/16. These addresses cannot be used for WAN.

Multicast Addresses : The block 224.0.0.0/4 is reserved for multicast addresses.

DHCP (Dynamic Host configuration protocol) : ISP can receive a block of addresses directly from ICANN and a small organization can receive a block of addresses from an ISP. After a block of addresses are assigned to an organization, However, address assignment in an organization can be done automatically using the Dynamic Host Configuration Protocol (DHCP). DHCP is an application-layer program, using the client-server paradigm, that actually helps TCP/IP at the network layer. Also called plug and play protocol. You can have multiple DHCP in same network.

Figure 18.25 *DHCP message format*



Fields:
Opcode: Operation code, request (1) or reply (2)
Htype: Hardware type (Ethernet, ...)
HLen: Length of hardware address
HCount: Maximum number of hops the packet can travel
Transaction ID: An integer set by the client and repeated by the server
Time elapsed: The number of seconds since the client started to boot
Flags: First bit defines unicast (0) or multicast (1); other 15 bits not used
Client IP address: Set to 0 if the client does not know it
Your IP address: The client IP address sent by the server
Server IP address: A broadcast IP address if client does not know it
Gateway IP address: The address of default router
Server name: A 64-byte domain name of the server
Boot file name: A 128-byte file name holding extra information
Options: A 64-byte field with dual purpose described in text

Option format

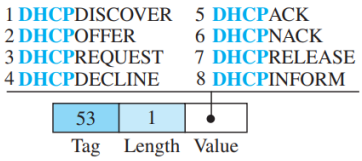


Figure 18.27 *Operation of DHCP*

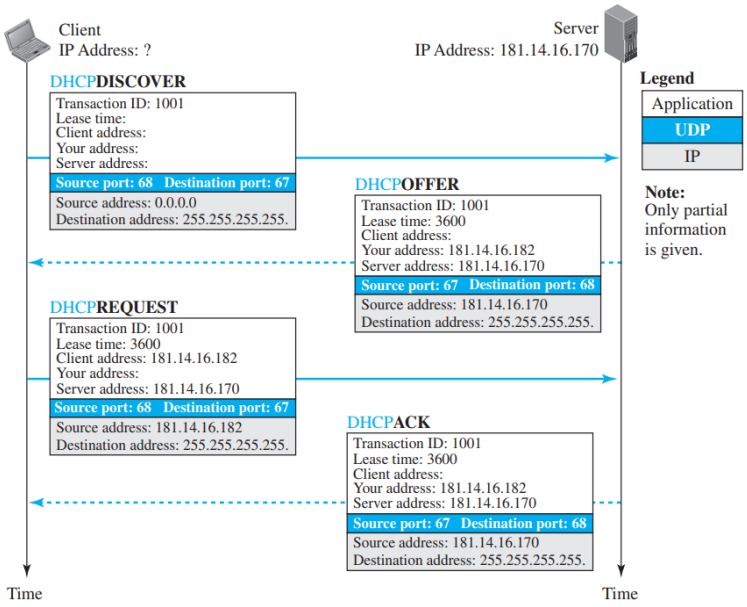
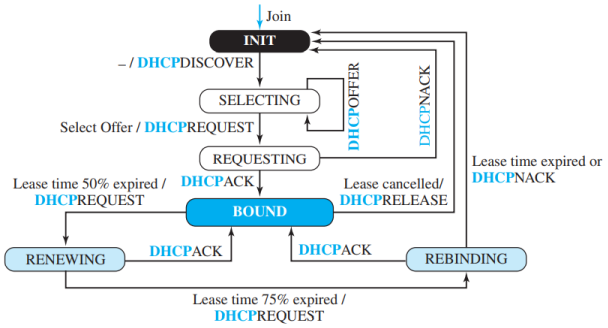


Figure 18.28 *FSM for the DHCP client*



Finally, the selected server responds with a DHCPACK message to the client if the offered IP address is valid. If the server cannot keep its offer (for example, if the address is offered to another host in between), the server sends a DHCPNACK message and the client needs to repeat the process. This message is also broadcast to let other servers know that the request is accepted or rejected.

A technology that can provide the mapping between the private and universal addresses, and at the same time support virtual private networks is **Network Address Translation** (NAT).

Address Translation : All of the outgoing packets go through the NAT router, which replaces the source address in the packet with the global NAT address. All incoming packets also pass through the NAT router, which replaces the destination address in the packet (the NAT router global address) with the appropriate private address. NAT causes routing table to includes less information.

Figure 18.30 *Address translation*

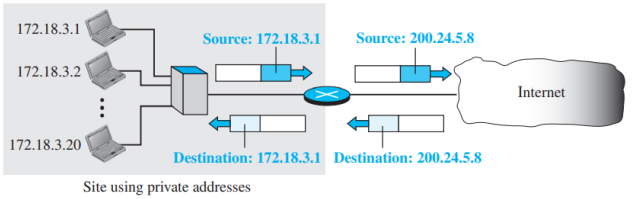
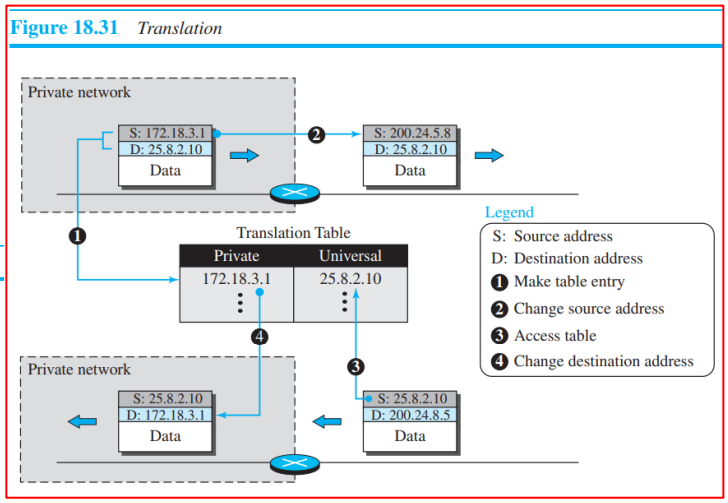


Figure 18.31 *Translation*



The reader may have noticed that translating the source addresses for an outgoing packet is straightforward. But how does the NAT router know the destination address for a packet coming from the Internet?

Using One IP Address : In its simplest form, a translation table has only two columns: the private address and the external address (destination address of the packet). When the router translates the source address of the outgoing packet, it also makes note of the destination address— where the packet is going. When the response comes back from the destination, the router uses the source address of the packet (as the external address) to find the private address of the packet.

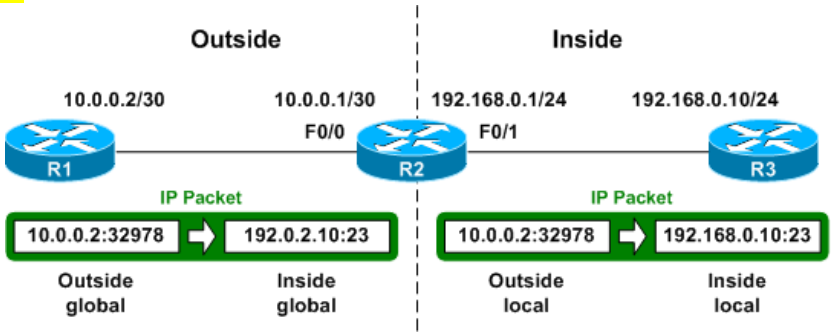
Using a Pool of IP Addresses : The use of only one global address by the NAT router allows only one private-network host to access a given external host. To remove this restriction, the NAT router can use a pool of global addresses. For example, instead of using only one global address (200.24.5.8), the NAT router can use four addresses (200.24.5.8, 200.24.5.9, 200.24.5.10, and 200.24.5.11).

Using Both IP Addresses and Port Addresses : To allow a many-to-many relationship between private-network hosts and external server programs, we need more information in the translation table. For example, suppose two hosts inside a private network with addresses 172.18.3.1 and 172.18.3.2 need to access the HTTP server on external host 25.8.3.2.

Table 18.1 *Five-column translation table*

Private address	*Private port*	*External address*	*External port*	*Transport protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
⋮	⋮	⋮	⋮	⋮

NAT Address Types :



Inside global : The address of the inside host as seen from the outside
Inside local : The address of the inside host as seen from the inside

Outside local : The address of the outside host as seen from the inside
Outside global : The address of the outside host as seen from the outside

Disadvantages of NAT :

- NAT slows down packet handing.
- End to end connection is lost (can be solved with static NAT sometime).

FORWARDING OF IP PACKETS : forwarding means to place the packet in its route to its destination. When IP is used as a connectionless protocol, forwarding is based on the destination address of the IP datagram; when the IP is used as a connection-oriented protocol, forwarding is based on the label attached to an IP datagram.

1) Forwarding Based on Destination Address : Unfortunately, the destination address in the packet gives no clue about the network address. To solve the problem, we need to include the mask (/n) in the table. In other words, a classless forwarding table needs to include four pieces of information: the mask, the network address, the interface number, and the IP address of the next router.

Example : Make a forwarding table for router R1 using the configuration in Figure 18.33

Figure 18.33 *Configuration for Example 18.7*

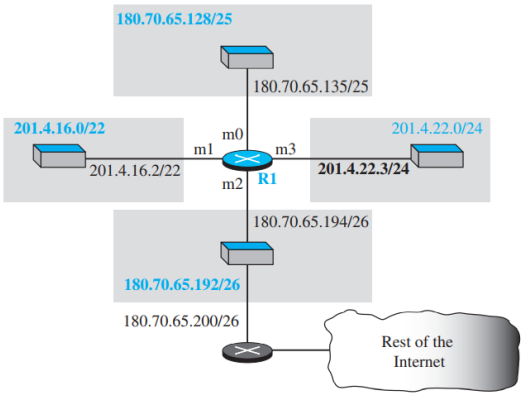


Table 18.2 *Forwarding table for router R1 in Figure 18.33*

Network address/mask	*Next hop*	*Interface*
180.70.65.192/**26**	—	m2
180.70.65.128/**25**	—	m0
201.4.22.0/**24**	—	m3
201.4.16.0/**22**	—	m1
Default	180.70.65.200	m2

Example 18.9 : Show the forwarding process if a packet arrives at R1 => in Figure 18.33 with the destination address 180.70.65.140.

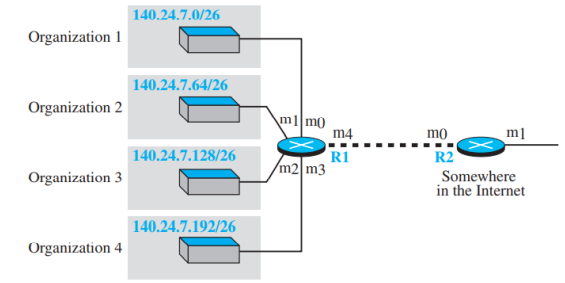
Solution : The router performs the following steps:

1. The first mask (/26) is applied to the destination address. The result is 180.70.65.128, which does not match the corresponding network address.

2. The second mask (/25) is applied to the destination address. The result is 180.70.65.128, which matches the corresponding network address. The next-hop address and the interface number m0 are extracted for forwarding the packet.

- The increased size of the table results in an increase in the amount of time needed to search the table. To alleviate the problem, the idea of address aggregation was designed.

Figure 18.34 *Address aggregation*



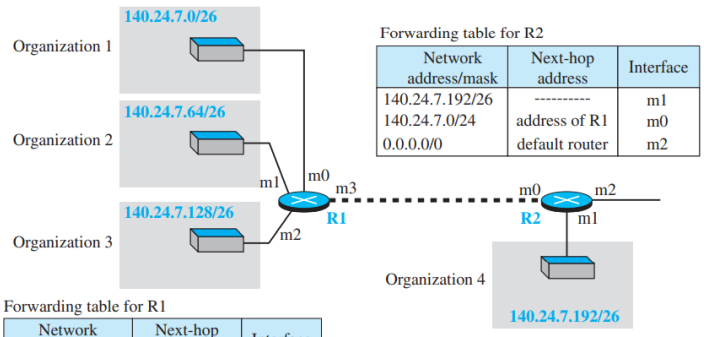
Forwarding table for R1

Network address/mask	Next-hop address	Interface
140.24.7.0/26	----------	m0
140.24.7.64/26	----------	m1
140.24.7.128/26	----------	m2
140.24.7.192/26	----------	m3
0.0.0.0/0	address of R2	m4

Forwarding table for R2

Network address/mask	Next-hop address	Interface
140.24.7.0/24	----------	m0
0.0.0.0/0	default router	m1

Figure 18.35 *Longest mask matching*



Forwarding table for R2

Network address/mask	Next-hop address	Interface
140.24.7.192/26	----------	m1
140.24.7.0/24	address of R1	m0
0.0.0.0/0	default router	m2

Forwarding table for R1

Network address/mask	Next-hop address	Interface
140.24.7.0/26	----------	m0
140.24.7.64/26	----------	m1
140.24.7.128/26	----------	m2
0.0.0.0/0	default router	m3

Hierarchical Routing : Internet is divided into backbone and national ISPs. National ISPs are divided into regional ISPs, and regional ISPs are divided into local ISPs. If the forwarding table has a sense of hierarchy like the Internet architecture, the forwarding table can decrease in size.

Figure 18.36 *Hierarchical routing with ISPs*

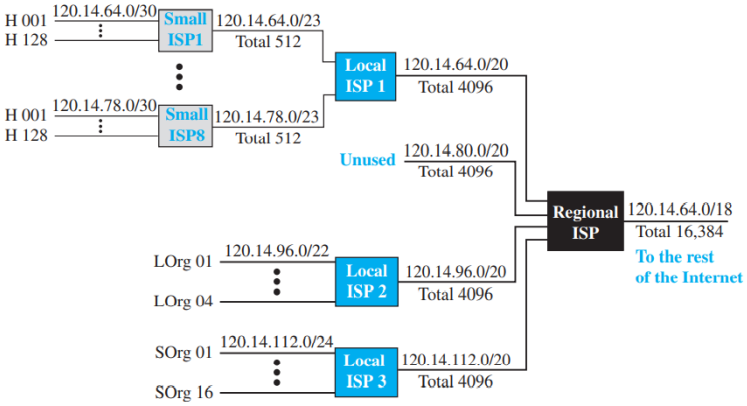
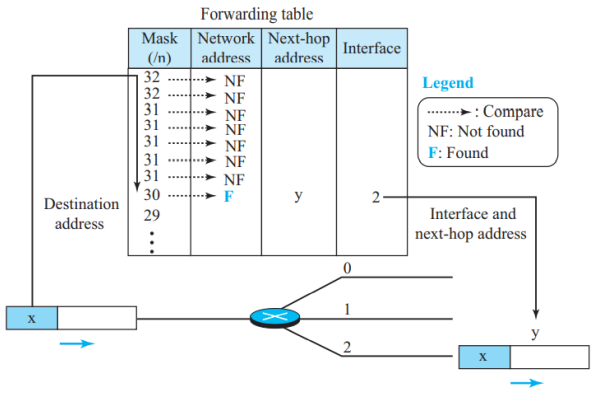


Figure 18.37 *Example 18.11: Forwarding based on destination address*



Geographical Routing : To decrease the size of the forwarding table even further, we need to extend hierarchical routing to include geographical routing. We must divide the entire address space into a few large blocks.

2) Forwarding Based on Label:

In the 1980s, an effort started to somehow change IP to behave like a connection-oriented protocol in which the routing is replaced by switching. As we discussed earlier in the chapter, in a connectionless network (datagram approach), a router forwards a packet based on the destination address in the header of the packet. On the other hand, in a connection-oriented network (virtual-circuit approach), a switch forwards a packet based on the label attached to the packet. Routing is normally based on searching the contents of a table; switching can be done by accessing a table using an index. In other words, routing involves searching; switching involves accessing.

Multi-Protocol Label Switching (MPLS) : During the 1980s, several vendors created routers that implement switching technology. Later IETF approved a standard that is called Multi-Protocol Label Switching. In this standard, some conventional routers in the Internet can be replaced by MPLS routers, which can behave like a router and a switch. When behaving like a router, MPLS can forward the packet based on the destination address; when behaving like a switch, it can forward a packet based on the label.

Figure 18.39 *MPLS header added to an IP packet*

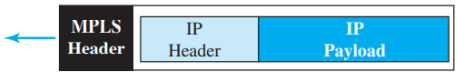
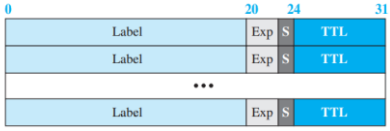


Figure 18.40 *MPLS header made of a stack of labels*



The following is a brief description of each field:

❑ **Label**. This 20-bit field defines the label that is used to index the forwarding table in the router. ❑ **Exp**. This 3-bit field is reserved for experimental purposes. ❑ **S.** The one-bit stack field defines the situation of the subheader in the stack. When the bit is 1, it means that the header is the last one in the stack. ❑ **TTL**. This 8-bit field is similar to the TTL field in the IP datagram. Each visited router decrements the value of this field. When it reaches zero, the packet is discarded to prevent looping.

Network-Layer Protocols : The main protocol, Internet Protocol version 4 (IPv4), is responsible for packetizing, forwarding, and delivery of a packet at the network layer. The Internet Control Message Protocol version 4 (ICMPv4) helps IPv4 to handle some errors that may occur in the network-layer delivery. The Internet Group Management Protocol (IGMP) is used to help IPv4 in multicasting. The Address Resolution Protocol (ARP) is used to glue the network and data-link layers in mapping network-layer addresses to link-layer addresses.

Figure 19.1 *Position of IP and other network-layer protocols in TCP/IP protocol suite*

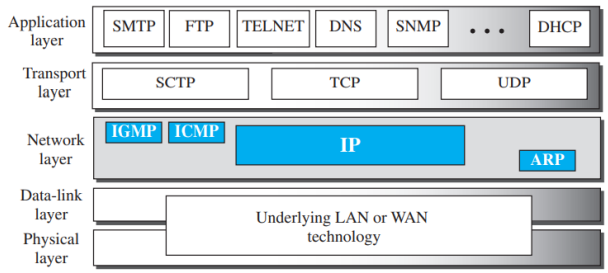
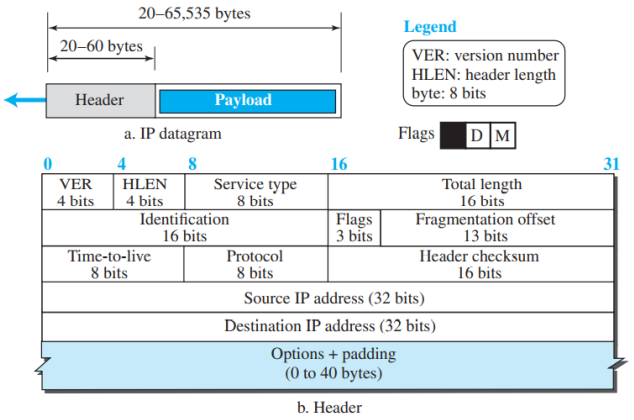


Figure 19.2 *IP datagram*



- IPv4 is an unreliable datagram protocol—a best-effort delivery service. The term best-effort means that IPv4 packets can be corrupted, be lost, arrive out of order, or be delayed, and may create congestion for the network. If reliability is important, IPv4 must be paired with a reliable transport-layer protocol such as TCP.

1) Datagram formate : (header especially)

❑ **Version Number**. The 4-bit version number (VER) field defines the version of the IPv4 protocol, which, obviously, has the value of 4.

❑ **Header Length**. The 4-bit header length (HLEN or IHL) field defines the total length of the datagram header in 4-byte words. The IPv4 datagram has a variable-length header. When a device receives a datagram, it needs to know when the header stops and the data, which is encapsulated in the packet, starts. The receiver needs to multiply the value of this field by 4 to find the total length.

❑ **Service Type**. In the original design of the IP header, this field was referred to as type of service (TOS), which defined how the datagram should be handled.

❑ **Total Length**. This 16-bit field defines the total length (header plus data) of the IP datagram in bytes.

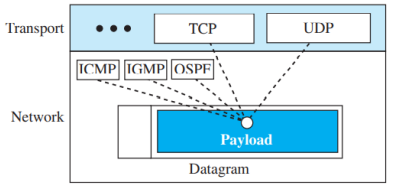
Length of data = total length − (HLEN) × 4

❑ **Identification, Flags, and Fragmentation Offset**. These three fields are related to the fragmentation of the IP datagram when the size of the datagram is larger than the underlying network can carry.

❑ The **time-to-live (TTL)** field is used to control the maximum number of hops (routers) visited by the datagram. When a source host sends the datagram, it stores a number in this field. This value is approximately two times the maximum number of routers between any two hosts. Each router that processes the datagram decrements this number by one.

❑ **Protocol**. In TCP/IP, the data section of a packet, called the payload, carries the whole packet from another protocol. the value of this field helps to define to which protocol the payload should be delivered. TCP uses byte stream that is every byte that is send using TCP is numbered.

Figure 19.3 *Multiplexing and demultiplexing using the value of the protocol field*



Some protocol values	
ICMP	01
IGMP	02
TCP	06
UDP	17
OSPF	89

❑ **Header checksum**. IP is not a reliable protocol; it does not check whether the payload carried by a datagram is corrupted during the transmission.

❑ **Source and Destination Addresses**. These 32-bit source and destination address fields define the IP address of the source and destination respectively.

❑ **Options**. A datagram header can have up to 40 bytes of options. Options can be used for network testing and debugging

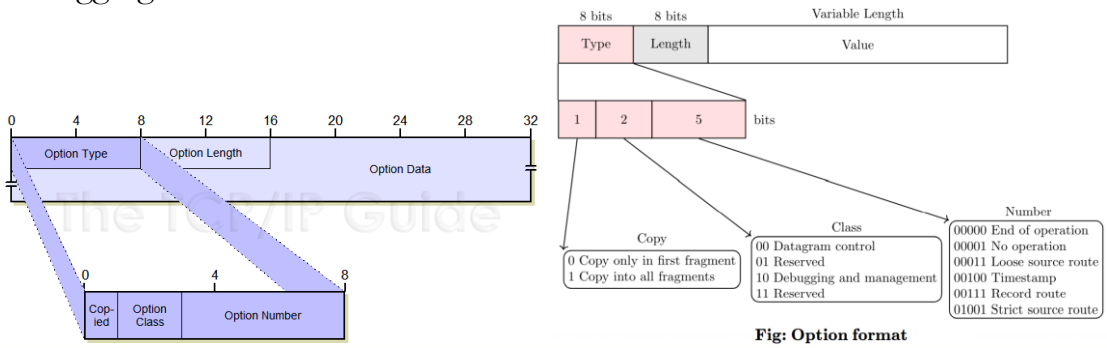


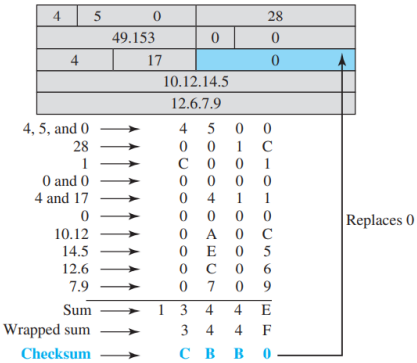
Fig: Option format

Subfield Name	Size (bytes)	Description
Option Type	1	***Option Type:*** This 8-bit field is divided into three "sub-subfields", according to the following format: **Sub-Subfield Name** / **Size (bytes)** / **Description** ***Copied*** / 1/8 1 bit) / ***Copied Flag:*** This bit is set to 1 if the option is intended to be copied into all fragments when a datagram is fragmented; it is cleared to 0 if the option should not be copied into fragments. ***Option Class*** / 2/8 (2 bits) / ***Option Class:*** Specifies one of four potential values that indicate the general category into which the option belongs. In fact, only two of the values are used: 0 is for *Control* options, and 2 for *Debugging and Measurement*. ***Option Number*** / 5/8 (5 bits) / ***Option Number:*** Specifies the kind of option. 32 different values can be specified for each of the two option classes. Of these, a few are more commonly employed. See below for more information on the specific options.
Option Length	0 or 1	***Option Length:*** For variable-length options, indicates the size of the entire option, including all three subfields shown here, in bytes.
Option Data	0 or Variable	***Option Data:*** For variable-length options, contains data to be sent as part of the option.

Options are divided into two broad categories: single-byte options and multiple-byte options. 1) **Single-Byte Options** : There are two single-byte options. *No Operation* : A no-operation option is a 1-byte option used as a filler between options. *End of Option* : An end-of-option option is a 1-byte option used for padding at the end of the option field. It, however, can only be used as the last option. 2) **Multiple-Byte Options** : There are four multiple-byte options. 1) Record Route : A record route option is used to record the Internet routers that handle the datagram. It can list up to **nine** router addresses. It can be used for debugging and management purposes. 2) Strict Source Route : A strict source route option is used by the source to predetermine a route for the datagram as it travels through the Internet. 3) Loose Source Route : A loose source route option is similar to the strict source route, but it is less rigid. Each router in the list must be visited, but the datagram can visit other routers as well. 4) Timestamp : A timestamp option is used to record the time of datagram processing by a router.

❑ **Payload**. Payload, or data, is the main reason for creating a datagram.

Figure 19.4 *Example of checksum calculation in IPv4*



Note that the calculation of wrapped sum and checksum can also be done as follows in hexadecimal:

Wrapped Sum = Sum mod FFFF
Checksum = FFFF − Wrapped Sum

Example 19.1

An IPv4 packet has arrived with the first 8 bits as $(01000010)_2$. The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 leftmost bits $(0100)_2$ show the version, which is correct. The next 4 bits $(0010)_2$ show an invalid header length $(2 \times 4 = 8)$. The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission.

Example 19.2

In an IPv4 packet, the value of HLEN is $(1000)_2$. How many bytes of options are being carried by this packet?

Solution

The HLEN value is 8, which means the total number of bytes in the header is 8×4, or 32 bytes. The first 20 bytes are the base header, the next 12 bytes are the options.

Example 19.3

In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is $(0028)_{16}$. How many bytes of data are being carried by this packet?

Solution

The HLEN value is 5, which means the total number of bytes in the header is 5×4, or 20 bytes (no options). The total length is $(0028)_{16}$ or 40 bytes, which means the packet is carrying 20 bytes of data $(40 - 20)$.

Example 19.4 An IPv4 packet has arrived with the first few hexadecimal digits as shown. $(45000028000100000102 \ldots)_{16}$. How many hops can this packet travel before being dropped? The data belong to what upper-layer protocol?

Solution : To find the time-to-live field, we skip 8 bytes (16 hexadecimal digits). The time-to-live field is the ninth byte, which is (01)16. This means the packet can travel only one hop. The protocol field is the next byte (02)16, which means that the upper-layer protocol is IGMP.

2) **Fragmentation** : The value of the MTU differs from one physical network protocol to another. For example, the value for a LAN is normally 1500 bytes, but for a WAN it can be larger or smaller. In order to make the IP protocol independent of the physical network, the designers decided to make the maximum length of the IP datagram equal to 65,535 bytes. This makes transmission more efficient if one day we use a link-layer protocol with an MTU of this size. However, for other physical networks, we must divide the datagram to make it possible for it to pass through these networks. This is called fragmentation. MTU is head + data or payload. MTU payload should be multiple of 8 if it is not then select some multiple of 8 smaller than mtu payload.

Figure 19.5 *Maximum transfer unit (MTU)*

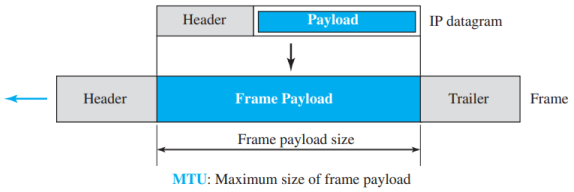
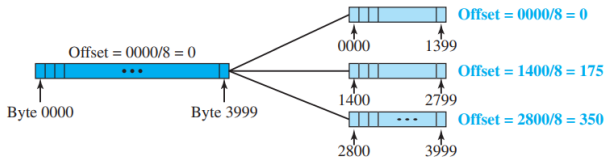


Figure 19.6 *Fragmentation example*



❑ **Identification** : The 16-bit identification field identifies a datagram originating from the source host. When a datagram is fragmented, the value in the identification field is copied into all fragments. all fragments having the same identification value should be assembled into one datagram.

Flags field - RDM: The leftmost bit is reserved (not used). The second bit (D bit) is called the do not fragment bit. If its value is 1, the machine must not fragment the datagram. If it cannot pass the datagram through any available physical network, it discards the datagram and sends an ICMP error message to the source host (discussed later). If its value is 0, the datagram can be fragmented if necessary. The third bit (M bit) is called the more fragment bit. If its value is 1, it means the datagram is not the last fragment; there are more fragments after this one. If its value is 0, it means this is the last or only fragment. The 13-bit fragmentation offset field shows the relative position of this fragment with respect to the whole datagram. It is the offset of the data in the original datagram measured in units of 8 bytes. Use of offset is to get first address of packet, by mul it by 8.

Figure 19.7 *Detailed fragmentation example*

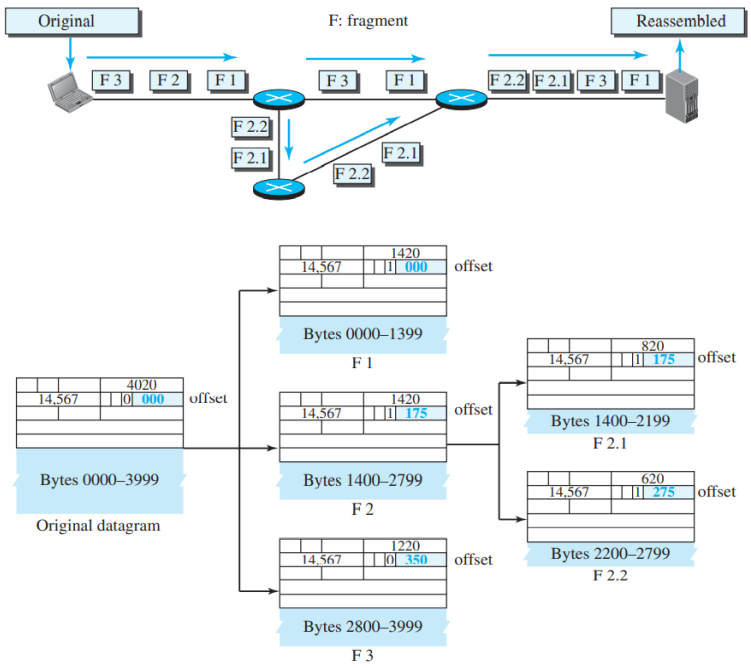
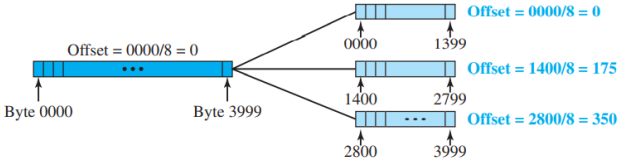


Figure 19.6 *Fragmentation example*



Example 19.10 A packet has arrived in which the offset value is 100, the value of HLEN is 5, and the value of the total length field is 100. What are the numbers of the first byte and the last byte? *Solution* : The first byte number is 100 × 8 = 800. The total length is 100 bytes, and the header length is 20 bytes (5 × 4), which means that there are 80 bytes in this datagram. If the first byte number is 800, the last byte number must be 879.

Packet Sniffing : An intruder may intercept an IP packet and make a copy of it.

Packet Modification : The attacker intercepts the packet, changes its contents, and sends the new packet to the receiver. The receiver believes that the packet is coming from the original sender.

IP Spoofing : An attacker can masquerade as somebody else and create an IP packet that carries the source address of another computer. Above attacks can be reduced by following method :

❑ Defining Algorithms and Keys, ❑ Packet Encryption, ❑ Data Integrity, ❑ Origin Authentication.

ICMPv4 (Internet message control protocol) : The IPv4 has no error-reporting or error-correcting mechanism. What happens if something goes wrong? What happens if a router must discard a datagram because it cannot find a route to the final destination, or because the time-to-live field has a zero value? What happens if the final destination host must discard the received fragments of a datagram because it has not received all fragments within a predetermined time limit? The Internet Control Message Protocol version 4 (ICMPv4) has been designed to compensate for the above two deficiencies.

The final destination host can reassemble the original datagram from the fragments received (if none of them is lost) using the following strategy: a. The first fragment has an offset field value of zero. b. Divide the length of the first fragment by 8. The second fragment has an offset value equal to that result. c. Divide the total length of the first and second fragment by 8. The third fragment has an offset value equal to that result. d. Continue the process. The last fragment has its M bit set to 0. e. Continue the process. The last fragment has a more bit value of 0.

Figure 19.8 *General format of ICMP messages*

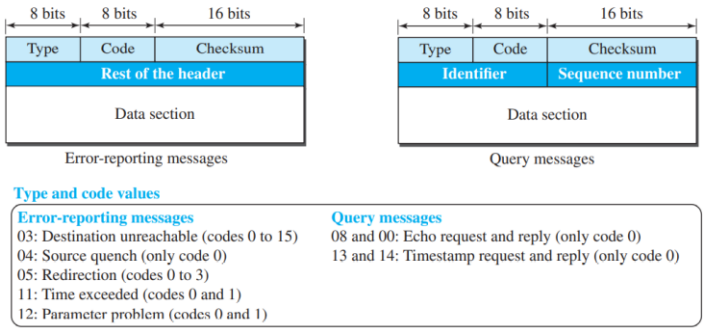
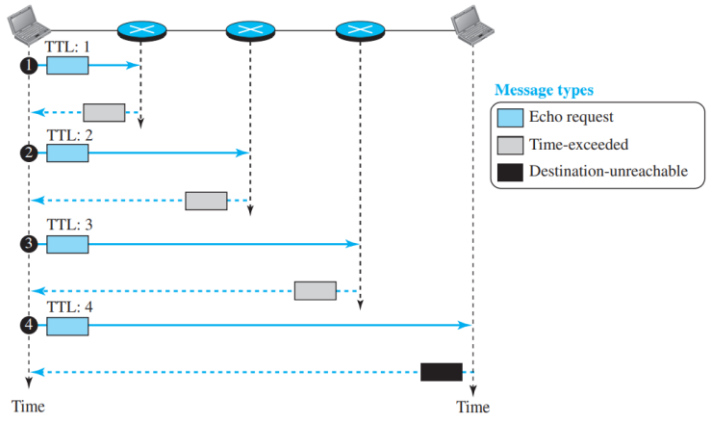


Figure 19.10 *Use of ICMPv4 in traceroute*



The following are important points about ICMP error messages:

❑ See figure 19.8 rounded rectangle it contains for what error ICMP message should be generated.

❑ No ICMP error message will be generated in response to a datagram carrying an ICMP error message.

❑ No ICMP error message will be generated for a fragmented datagram that is not the first fragment.

❑ No ICMP error message will be generated for a datagram having a multicast address.

❑ No ICMP error message will be generated for a datagram having a special address such as 127.0.0.0 or 0.0.0.0.

- Note that all error messages contain a data section that includes the IP header of the original datagram plus the first 8 bytes of data in that datagram. The original datagram header is added to give the original source, which receives the error message, information about the datagram itself.

Destination Unreachable : A host is unreachable. This may happen, for example, when we use the HTTP protocol to access a web page, but the server is down. The message "destination host is not reachable" is created and sent back to the source.

Source Quench : Which informs the sender that the network has encountered congestion and the datagram has been dropped; the source needs to slow down sending more datagrams.

Redirection Message : The redirection message (type 5) is used when the source uses a wrong router to send out its message.

Parameter Problem : A parameter problem message (type 12) can be sent when either there is a problem in the header of a datagram (code 0) or some options are missing or cannot be interpreted (code 1)

Query Messages : The echo request (type 8) and the echo reply (type 0) pair of messages are used by a host or a router to test the liveliness of another host or router. The timestamp request (type 13) and the timestamp reply (type 14) pair of messages are used to find the round-trip time between two devices or to check whether the clocks in two devices are synchronized.

Debugging tools : Ping : We can use the ping program to find if a host is alive and responding. We use ping here to see how it uses ICMP packets. The source host sends ICMP echo-request messages; the destination, if alive, responds with ICMP echo-reply messages.

Ad-hoc network (WANET) - wireless ad hoc network are there in which hosts are not fixed, any host can leave or join the network on demand. So, following all these protocols will help in reducing overhead for routers. It does not rely on a pre-existing infrastructure, such as routers in wired networks or access points in wireless networks.

The traceroute : Program gets help from two error-reporting messages: time-exceeded and destination-unreachable. Traceroute works by sending packets with gradually increasing value, starting with value of 16. The first router receives the packet, decrements the value and drops the packet because it then has value zero. The router sends an Time Exceeded message back to the source. The next set of packets are given a value of so the first router forwards the packets, but the second router drops them and replies with Time Exceeded. Proceeding in this way, traceroute uses the returned Time Exceeded messages to build a list of routers that packets traverse, until the destination is reached and returns an ICMP Echo Reply message. The technique used by traceroute to identify these hosts is by progressively querying routers about the next router on the path to B using ICMP packets, starting with the first router.

Unicast Routing : In unicast routing, a packet is routed, hop by hop, from its source to its destination by the help of forwarding tables. The source host needs no forwarding table because it delivers its packet to the default router in its local network. The destination host needs no forwarding table either because it receives the packet from its default router in its local network. In other words, there are several routes that a packet can travel from the source to the destination; what must be determined is which route the packet should take.

Figure 20.1 *An internet and its graphical representation*

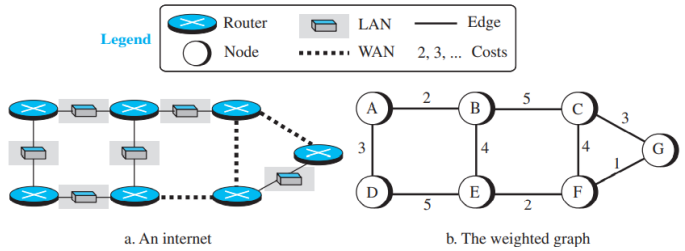
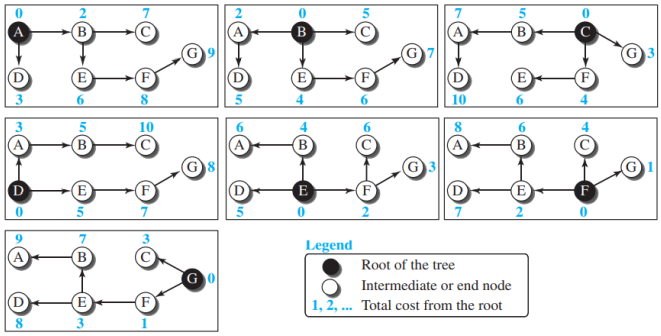


Figure 20.2 *Least-cost trees for nodes in the internet of Figure 20.1*



1) Least-Cost Routing : the source router chooses a route to the destination router in such a way that the total cost for the route is the least cost among all possible routes.

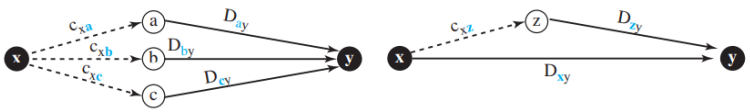
Routing Algorithm :

1) *Distance-Vector Routing* : In distance-vector routing, the first thing each node creates is its own least-cost tree with the rudimentary information it has about its immediate neighbors.

Bellman-Ford Equation : The following shows the general case in which Dij is the shortest distance and cij is the cost between nodes i and j. Dxy = min{(cxa + Day), (cxb + Dby), (cxc + Dcy), …}

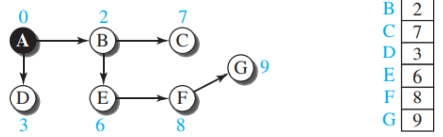
Dxy = min{Dxy, (cxz + Dzy)}

Figure 20.3 *Graphical idea behind Bellman-Ford equation*



a. General case with three intermediate nodes b. Updating a path with a new route

Figure 20.4 *The distance vector corresponding to a tree*



a. Tree for node A b. Distance vector for node A

Distance Vectors : The concept of a distance vector is the rationale for the name distance-vector routing. A least-cost tree is a combination of least-cost paths from the root of the tree to all destinations. All the updates of distance vectors happen at the same time so old table values should be considered. **Shortest path algo** is being applied at every node simultaneously. In both DV and LS.

Figure 20.5 *The first distance vector for an internet*

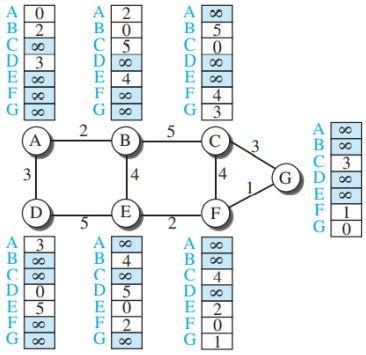
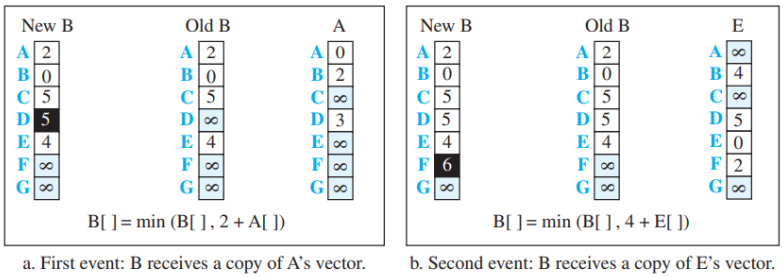


Figure 20.6 *Updating distance vectors*



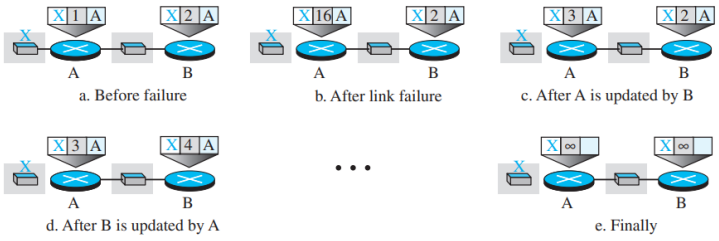
a. First event: B receives a copy of A's vector. b. Second event: B receives a copy of E's vector.

Note:
X[ ]: the whole vector

Problem : Count to Infinity: For a routing protocol to work properly, if a link is broken (cost becomes infinity), every other router should be aware of it immediately, but in distance-vector routing, this takes some time. The problem is referred to as count to infinity.

Solution : 1) Two-Node Loop :

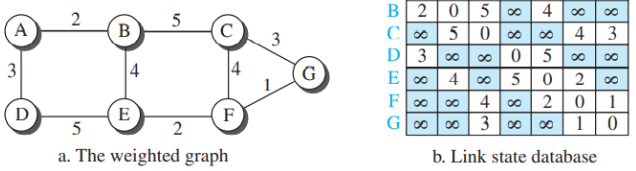
Figure 20.7 *Two-node instability*



2) Split Horizon : One solution to instability is called split horizon. In this strategy, instead of flooding the table through each interface, each node sends only part of its table through each interface.
3) In the poison reverse strategy B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."
4) Three-Node Instability The two-node instability can be avoided using split horizon combined with poison reverse. However, if the instability is between three nodes, stability cannot be guaranteed.

2) **Link-State Routing :** A routing algorithm that directly follows our discussion for creating least-cost trees and forwarding tables is link-state (LS) routing. This method uses the term link-state to define the characteristic of a link (an edge) that represents a network in the internet.

Figure 20.8 *Example of a link-state database*

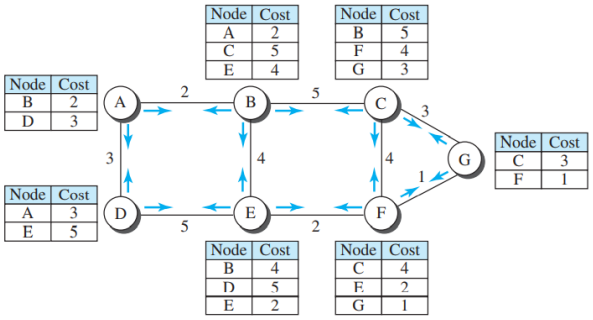


a. The weighted graph

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

b. Link state database

Figure 20.9 *LSPs created and sent out by each node to build LSDB*



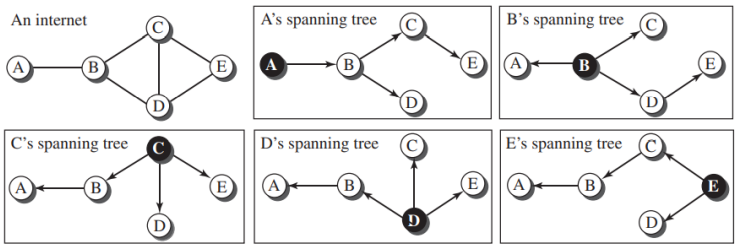
To create a least-cost tree with this method, each node needs to have a complete map of the network, which means it needs to know the state of each link. The collection of states for all links is called the link-state database (LSDB).

Now the question is how each node can create this LSDB that contains information about the whole internet. This can be done by a process called flooding. Each node can send some greeting messages to all its immediate neighbors (those nodes to which it is connected directly) to collect two pieces of information for each neighboring node: the identity of the node and the cost of the link. The combination of these two pieces of information is called the LS packet (LSP).

```
1  Dijkstra's Algorithm ( )
2  {
3      // Initialization
4      Tree = {root}                    // Tree is made only of the root
```

```
5    for (y = 1 to N)                    // N is the number of nodes
6    {
7        if (y is the root)
8            D[y] = 0                    // D[y] is shortest distance from root to node y
9        else if (y is a neighbor)
10           D[y] = c[root][y]           // c[x][y] is cost between nodes x and y in LSDB
11       else
12           D[y] = ∞
13   }
14   // Calculation
15   repeat
16   {
17       find a node w, with D[w] minimum among all nodes not in the Tree
18       Tree = Tree ∪ {w}               // Add w to tree
19       // Update distances for all neighbors of w
20       for (every node x, which is a neighbor of w and not in the Tree)
21       {
22           D[x] = min{D[x], (D[w] + c[w][x])}
23       }
24   } until (all nodes included in the Tree)
25 } // End of Dijkstra
```

Figure 20.11 *Spanning trees in path-vector routing*



3) **Path-Vector Routing :** Both link-state and distance-vector routing are based on the least-cost goal. However, there are instances where this goal is not the priority. For example, assume that there are some routers in the internet that a sender wants to prevent its packets from going through. For example, a router may belong to an organization that does not provide enough security or it may belong to a commercial rival of the sender which might inspect the packets for obtaining information. Least-cost routing does not prevent a packet from passing through an area when that area is in the least-cost path. Path vector routing assume that there is one node in each autonomous system that acts on behalf of the entire autonomous system.

Figure 20.12 *Path vectors made at booting time*

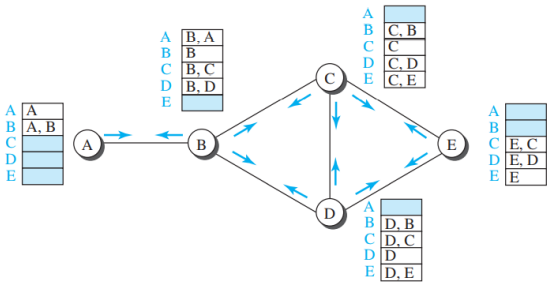


Figure 20.13 *Updating path vectors*

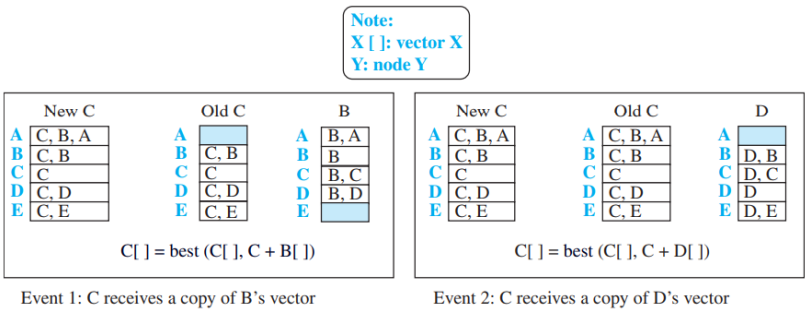
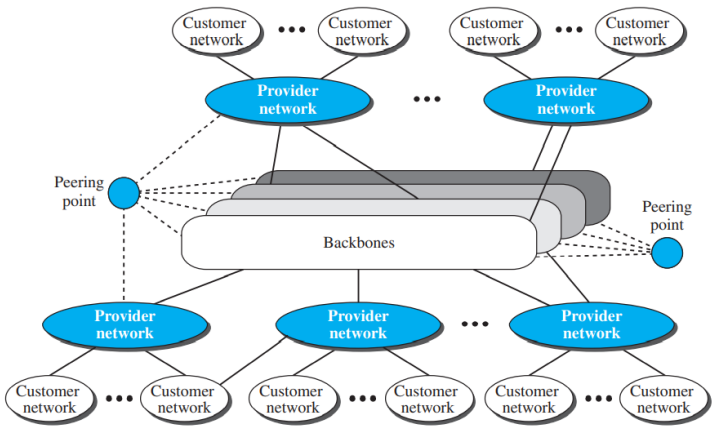


Table 20.3 *Path-vector algorithm for a node*

```
1  Path_Vector_Routing ( )
2  {
3     // Initialization
4     for (y = 1 to N)
5     {
6        if (y is myself)
7           Path[y] = myself
8        else if (y is a neighbor)
9           Path[y] = myself + neighbor node
10       else
11          Path[y] = empty
12    }
13    Send vector {Path[1], Path[2], …, Path[y]} to all neighbors
14    // Update
15    repeat (forever)
16    {
17       wait (for a vector Path_w from a neighbor w)
18       for (y = 1 to N)
19       {
20          if (Path_w includes myself)
21             discard the path                    // Avoid any loop
22          else
23             Path[y] = best {Path[y], (myself + Path_w[y])}
24       }
25       If (there is a change in the vector)
26          Send vector {Path[1], Path[2], …, Path[y]} to all neighbors
27    }
28 } // End of Path Vector
```

Figure 20.14 *Internet structure*



⇨ **UNICAST ROUTING PROTOCOLS :** we discuss three common protocols used in the Internet: Routing Information Protocol (RIP), based on the distance-vector algorithm, Open Shortest Path First (OSPF), based on the link-state algorithm, and Border Gateway Protocol (BGP), based on the path-vector algorithm.

A protocol is more than an algorithm.

Any of these three entities (backbone, provider network, or customer network) can be called an Internet Service Provider or ISP. They provide services, but at different levels.

Hierarchical Routing : It is obvious that routing in the Internet cannot be done using a single protocol for two reasons: a scalability problem and an administrative issue. Scalability problem means that the size of the forwarding tables becomes huge, searching for a destination in a forwarding table becomes time-consuming, and updating creates a huge amount of traffic. The administrative issue is related to the Internet structure described in Figure 20.14. As the figure shows, each ISP is run by an administrative authority. The administrator needs to have control in its system.

Anonymous System : Presently, the two common intradomain routing protocols are RIP and OSPF; the only interdomain routing protocol is BGP. Each AS can run a routing protocol that meets its needs, but the global Internet runs a global protocol to glue all ASs together.

❑ Stub AS. A stub AS has only one connection to another AS. The data traffic can be either initiated or terminated in a stub AS; the data cannot pass through it. (eg. Costumer network)

❑ Multihomed AS. A multihomed AS can have more than one connection to other ASs, but it does not allow data traffic to pass through it.

❑ Transient AS. A transient AS is connected to more than one other AS and also allows the traffic to pass through. The provider networks and the backbone are good examples of transient ASs.

- **Routing Information Protocol (RIP)** : Intradomain routing protocols based on the distance-vector routing algorithm

the number of hops, which means the number of networks (subnets) a packet needs to travel through from the source router to the final destination host. In RIP, the maximum cost of a path can be 15, which means 16 is considered as infinity (no connection). For this reason, RIP can be used only in autonomous systems in which the diameter of the AS is not more than 15 hops.

Figure 20.15 *Hop counts in RIP*

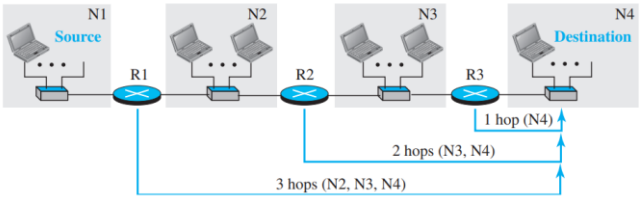


Figure 20.16 *Forwarding tables*

Forwarding table for R1

Destination network	Next router	Cost in hops
N1	——	1
N2	——	1
N3	R2	2
N4	R2	3

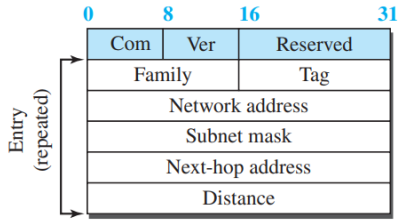
Forwarding table for R2

Destination network	Next router	Cost in hops
N1	R1	2
N2	——	1
N3	——	1
N4	R3	2

Forwarding table for R3

Destination network	Next router	Cost in hops
N1	R2	3
N2	R2	2
N3	——	1
N4	——	1

Figure 20.17 *RIP message format*

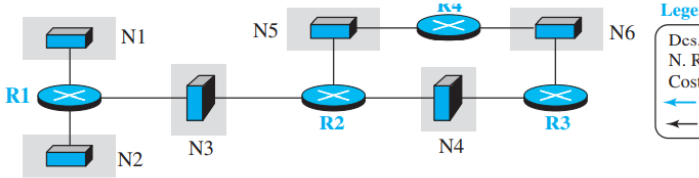


Fields

Com: Command, request (1), response (2)
Ver: Version, current version is 2
Family: Family of protocol, for TCP/IP value is 2
Tag: Information about autonomous system
Network address: Destination address
Subnet mask: Prefix length
Next-hop address: Address length
Distance: Number of hops to the destination

RIP has two types of messages: request and response. A request message is sent by a router that has just come up or by a router that has some time-out entries. A request message can ask about specific entries or all entries. A response (or update) message can be either solicited or unsolicited. A solicited response message is sent only in answer to a request message. It contains information about the destination specified in the corresponding request message. An unsolicited response message, on the other hand, is sent periodically, every 30 seconds or when there is a change in the forwarding table.

Figure 20.18 *Example of an autonomous system using RIP*



R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1

R2

Des.	N. R.	Cost
N3	——	1
N4	——	1
N5	——	1

R3

Des.	N. R.	Cost
N4	——	1
N6	——	1

R4

Des.	N. R.	Cost
N5	——	1
N6	——	1

Forwarding tables after all routers booted

New R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1
N4	R2	2
N5	R2	2

Old R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1

R2 Seen by R1

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	R2	2

New R3

Des.	N. R.	Cost
N3	R2	2
N4	——	1
N5	R2	2
N6	——	1

Old R3

Des.	N. R.	Cost
N4	——	1
N6	——	1

R2 Seen by R3

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	R2	2

New R4

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	——	1
N6	——	1

Old R4

Des.	N. R.	Cost
N5	——	1
N6	——	1

R2 Seen by R4

Des.	N. R.	Cost
N3	R2	2
N4	R2	2
N5	R2	2

Changes in the forwarding tables of R1, R3, and R4 after they receive a copy of R2's table

Final R1

Des.	N. R.	Cost
N1	——	1
N2	——	1
N3	——	1
N4	R2	2
N5	R2	2
N6	R2	3

Final R2

Des.	N. R.	Cost
N1	R1	2
N2	R1	2
N3	——	1
N4	——	1
N5	——	1
N6	R3	2

Final R3

Des.	N. R.	Cost
N1	R2	3
N2	R2	3
N3	R2	2
N4	——	1
N5	R2	2
N6	——	1

Final R4

Des.	N. R.	Cost
N1	R2	3
N2	R2	3
N3	R2	2
N4	R2	2
N5	——	1
N6	——	1

Forwarding tables for all routers after they have been stablized

❑ Three performance of RIP : 1) Update Messages. 2) Convergence of Forwarding Tables 3) Robustness.

- **Open Shortest Path First (OSPF) :** is also an intradomain, classless routing protocol like RIP, but it is based on the link-state routing protocol. we described earlier in the chapter. OSPF is an open protocol, which means that the specification is a public document.

Figure 20.19 *Metric in OSPF*

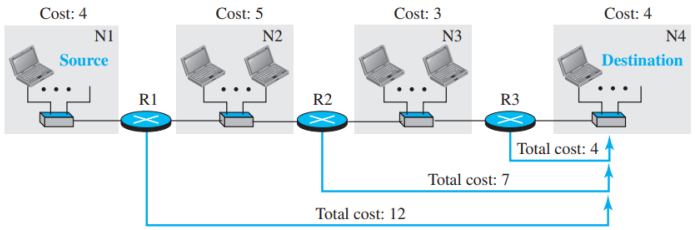
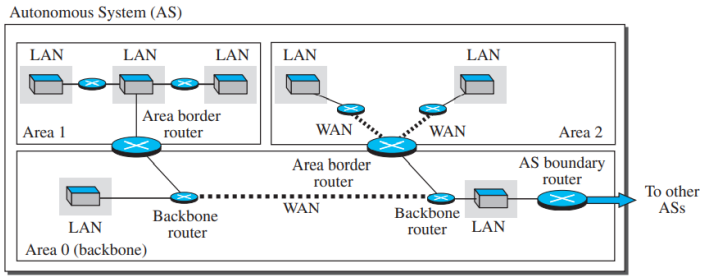


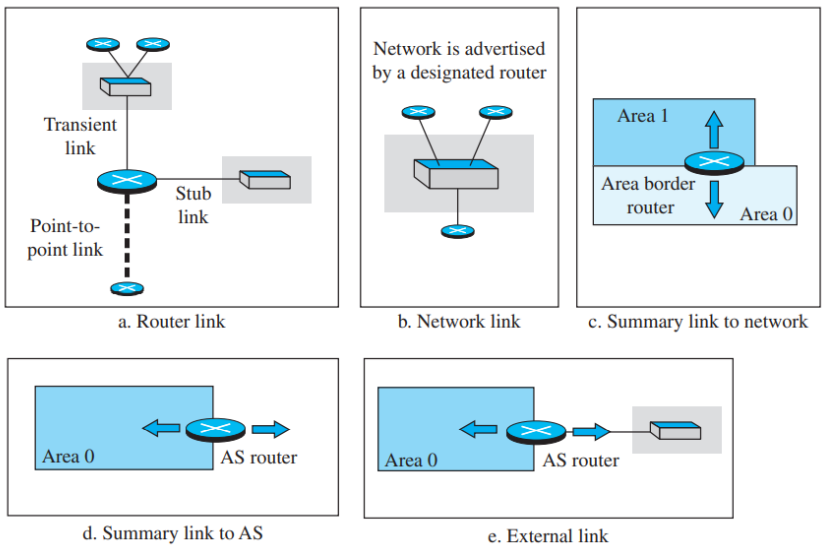
Figure 20.21 *Areas in an autonomous system*



Each OSPF router can create a forwarding table after finding the shortest-path tree between itself and the destination using Dijkstra's algorithm, described earlier in the chapter.

The formation of shortest-path trees in OSPF requires that all routers flood the whole AS with their LSPs to create the global LSDB. Although this may not create a problem in a small AS, it may have created a huge volume of traffic in a large AS. To prevent this, the AS needs to be divided into small sections called areas. Each area acts as a small independent domain for flooding LSPs. In other words, OSPF uses another level of hierarchy in routing: the first level is the autonomous system, the second is the area.

Figure 20.22 *Five different LSPs*



❑ Router link. A router link advertises the existence of a router as a node. In addition to giving the address of the announcing router, this type of advertisement can define one or more types of links that connect the advertising router to other entities.

❑ Network link. A network link advertises the network as a node. However, since a network cannot do announcements itself (it is a passive entity), one of the routers is assigned as the designated router and does the advertising.

❑ Summary link to network. This is done by an area border router; it advertises the summary of links collected by the backbone to an area or the summary of links collected by the area to the backbone.

❑ Summary link to AS. This is done by an AS router that advertises the summary links from other ASs to the backbone area of the current AS, information which later can be disseminated to the areas so that they will know about the networks in other ASs.

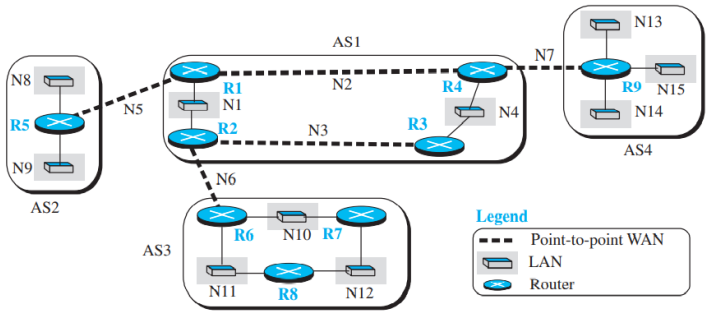
❑ External link. This is also done by an AS router to announce the existence of a single network outside the AS to the backbone area to be disseminated into the areas.

OSPF Messages : The hello message (type 1) is used by a router to introduce itself to the neighbors and announce all neighbors that it already knows. The database description message (type 2) is normally sent in response to the hello message to allow a newly joined router to acquire the full LSDB. The link-state request message (type 3) is sent by a router that needs information about a specific LS. The link-state update message (type 4) is the main OSPF message used for building the LSDB. This message, in fact, has five different versions (router link, network link, summary link to network, summary link to AS border router, and external link), as we discussed before. The link-state acknowledgment message (type 5) is used to create reliability in OSPF; each router that receives a link-state update message needs to acknowledge it.

- **Border Gateway Protocol Version 4 (BGP4) :** The Border Gateway Protocol version 4 (BGP4) is the only interdomain routing protocol used in the Internet today. BGP4 is based on the path-vector algorithm we described before, but it is tailored to provide information about the reachability of networks in the Internet.

AS2, AS3, and AS4 are stub autonomous systems; AS1 is a transient one.

Figure 20.24 *A sample internet with four ASs*



To enable each router to route a packet to any network in the internet, we first install a variation of BGP4, called external BGP (eBGP), on each border router (the one at the edge of each AS which is connected to a router at another AS). We then install the second variation of BGP, called internal BGP (iBGP), on all routers. This means that the border routers will be running three routing protocols (intradomain, eBGP, and iBGP), but other routers are running two protocols (intradomain and iBGP). We discuss the effect of each BGP variation separately.

BGP performance can be compared with RIP. BGP speakers exchange a lot of messages to create forwarding tables, but BGP is free from loops and count-to-infinity. The same weakness we mention for RIP about propagation of failure and corruption also exists in BGP.

The maximum cost of a path in an autonomous system (AS) which uses Routing Information Protocol (RIP) for intra-AS routing is 15.

- **Multicasting :**

Figure 21.1 *Unicasting*

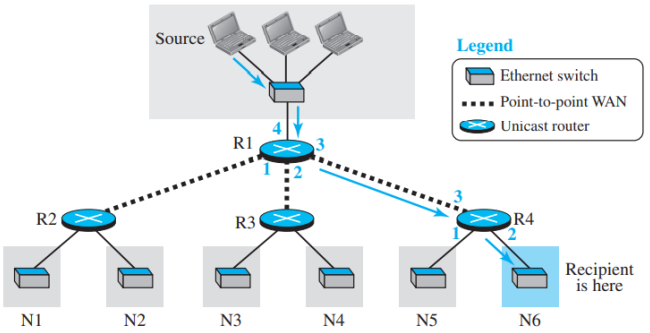
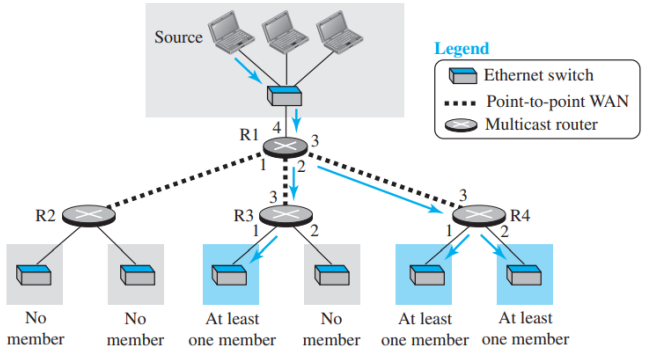


Figure 21.2 *Multicasting*



Multicasting starts with a single packet from the source that is duplicated by the routers. The destination address in each packet is the same for all duplicates. Note that only a single copy of the packet travels between any two routers. In multiple unicasting, several packets start from the source. If there are three destinations, for example, the source sends three packets, each with a different unicast destination address. Note that there may be multiple copies traveling between two routers. For example, when a person sends an e-mail message to a group of people, this is multiple unicasting. The e-mail application software creates replicas of the message, each with a different destination address, and sends them one by one.

Multicasting is more efficient than multiple unicasting. In Figure 21.3, we can see how multicasting requires less bandwidth than multiple unicasting. In multiple unicasting, some of the links must handle several copies. 2. In multiple unicasting, the packets are created by the source with a relative delay between packets. If there are 1,000 destinations, the delay between the first and the last packet may be unacceptable. In multicasting, there is no delay because only one packet is created by the source.

- **Broadcasting :** Broadcasting means one-to-all communication: a host sends a packet to all hosts in an internet. Broadcasting in this sense is not provided at the Internet level for the obvious reason that it may create a huge volume of traffic and use a huge amount of bandwidth. Partial broadcasting, however, is done in the Internet. For example, some peer-to-peer applications may use broadcasting to access all peers.

NOTE :

1) The lower end layer device cannot work at higher end layer as per rules of networking model. Since router is used for connecting different networks in network layer, it can also in data link layer for connecting LANs.
2) It is possible for computer to have multiple IP addresses. The usual reason for a device to have more than one IP address is that it is present on more than one separate subnet, or network.
3) OSPF needs to perform reliable multicasting because it needs to talk to multiple possible neighbours on the same network segment. Now, TCP does not support multicast and UDP is not reliable Therefore, OSPF implements its own transport mechanism that allows both for reliability (acknowledgements and retransmissions of lost segments) and multicasting, bypassing both TCP and UDP.

Questions on Network Layer **:**

1) In a packet switching network, packets are routed from source to destination along a single path having two intermediate nodes. If the message size is 24 bytes and each packet contains a header of 3 bytes, then the optimum packet size is :

Answer : As we know in packet switching- dividing message into packets decrease the transmission time due to pipelined transmission. there are two station and between two station two node are there means router. so optimum packet size will be the packet size that decreases the overall transmission time. As pipelining transmission will be done. So, let the size of optimal packet be x+3. which means there are 24/x number of packets that we have to transmit. After first packets crosses the 1st router the station will send other packet because of pipelining. There are three links in between so 3*transmission time. Transmission time is equal to length of data/ bandwidth lets call L/BW and L here is (x+3). so now only last will take whole transmission time. means (24/x)-1 * (3+x) + 3 * (3+x) = TT. now we take derivative to minimise it and we will get 6 as the answer.

2) A node X on a 10 Mbps network is regulated by a token bucket. The token bucket is filled at a rate of 2 Mbps. Token bucket is initially filled with 16 megabits. The maximum duration taken by X to transmit at full rate of 10 Mbps is _____ secs.

Answer : T = Capacity/(Rout-Rin). Here X wants to transmit means he wants to empty the token bucket so, Rout = 10. Capacity is given as 16. So, answer is 2.

3)

If probability of frame reaching safely is 0.1 then mean number of transmissions of a frame to make it success is _____.

10 — Correct Option

Solution :
10
The mean number of transmissions before successful transmission is 1/p, where p is the probability of frame reaching safely. Therefore required number of transmissions are 1/0.1 = 10.

4) Consider an IP packet with a length of 4,500 bytes that includes a 20-byte IPv4 header and 40-byte TCP header. The packet is forwarded to an IPv4 router that supports a Maximum Transmission Unit (MTU) of 600 bytes. Assume that the length of the IP header in all the outgoing fragments of this packet is 20 bytes. Assume that the fragmentation offset value stored in the first fragment is 0. The fragmentation offset value stored in the third fragment is

Answer : IP packet is in network layer so it will contain both IP and TCP header. We will subtract only IP. Because header gets added while exiting stage not while entering and header gets removed while entering.

Packet Length $= 4500B$
IP Payload $= 4500 - 20 = 4480B$

MTU $= 600B$
MTU Payload $= 600B - 20B = 580B$
But payload should be multiple of 8 so number nearest to 580 and multiple of 8 is 576, so MTU payload $= 576B$
IP Packet size $= 576 + 20B = 596B$
Size of Offset $= \frac{576}{8} = 72$

1^{st} fragment offset $= 0$
2^{nd} fragment offset $= 72$
3^{rd} fragment offset $= 144$

5) Consider the following three statements about link state and distance vector routing protocols, for a large network with 500 network nodes and 4000 links.

[S1]: The computational overhead in link state protocols is higher than in distance vector protocols.
[S2]: A distance vector protocol (with split horizon) avoids persistent routing loops, but not a link state protocol.
[S3]: After a topology change, a link state protocol will converge faster than a distance vector protocol.
Which one of the following is correct about S1, S2 and S3 ?

Answer :
The computational overhead in link state protocols is higher than in distance vector protocols. Bcz LSR is based upon global knowledge whereas DVR is based upon Local info . Persistent looping can be avoided with the help of split horizon in DVR. But there is no concept of persistent looping in LSR, in LSR only temporary loop exist and can automatically solved by system or router. S2 is false. And, after a topology change, a link state protocol will converge faster than a distance vector protocol. S3 is true.

6) A link of capacity 100 Mbps is carrying traffic from a number of sources. Each source generates an on-off traffic stream; when the source is on, the rate of traffic is 10Mbps, and when the source is off, the rate of traffic is zero. The duty cycle, which is the ratio of on-time to off-time, is 1:2. When there is no buffer at the link, the minimum number of sources that can be multiplied on the link so that link capacity is not wasted and no data loss occurs in S1. Assuming that all sources are synchronized and that the link is provided with a large buffer, the maximum number of sources that can be multiplied so that no data loss occurs is S2. The values of S1 and S2 are, respectively,

Answer : Since there is no buffer and constraint given is there should not be any lost, and no wastage of capacity as well. Since data should not be lost, we calculate for the extreme case when all sources are on(that is transmitting) 10Mbps x n-station <= 100 Mbps
n = 10 station.
In next part of the question it is given that the link is provided with large buffer and we are asked to find out the maximum number of stations.
1/3 x 10 x n <= 100
n = 30 station.

7) In a packet switching network, packets are routed from source to destination along a single path having two intermediate nodes. If the message size is 24 bytes and each packet contains a header of 3 bytes, then the optimum packet size is:

Answer : There are 3 links in total between source and destination. We need to calculate minimum transmission time. Let x be the payload size of message so there are 24/x number of packets so total optimal packet size is x+3. Now Total time = total data / BW. Consider bandwidth is 1. Total data = 3(x+3) + (k-1)(x+3). Here k represents total number of packets. After solving and differentiating we get x = 6. Means optimum packet size is 6+3=9 bytes.

8) In a network of LANs connected by bridges, packets are sent from one LAN to another through intermediate bridges. Since more than one path may exist between two LANs, packets may have to be routed through multiple bridges. Why is the spanning tree algorithm used for bridge-routing?

Answer : To avoid infinite loop.

9) A group of 15 routers is interconnected in a centralized complete binary tree with a router at each tree node. Router communicates with router by sending a message to the root of the tree. The root then sends the message back down to router . The mean number of hops per message, assuming all possible router pairs are equally likely is

Answer : Make binary tree with 4 level because it is only complete binary tree with 15 nodes. Average number of hops message goes to root = 3x8 + 2x4 + 1x2 + 0x1 / 15 = 2.267. Now it is given that root then sends message back down to router. So now total average hops message = 2 x 2.267 = 4.534

10) Most important

An IP packet originally has a size of 7000 bytes including 20-byte header and 6980-bytes payload. To reach the destination, the route goes through three networks, A, B, and C. Network A is the one where the sender is directly connected, and Network C is where the receiver is directly connected. Network A has an MTU of 5000 bytes. Network B has an MTU of 3000 bytes. Network C has an MTU of 1000 bytes. Assume that there is no other traffic and no congestion in the network, also network does not re-order the packets. Which of the following statement(s) is/are TRUE about this network?

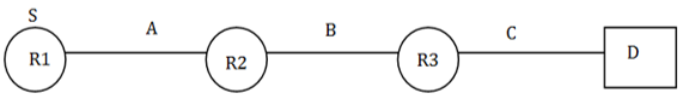
(a) The total number of fragments received by the receiver is 10.

(b) The total length of last fragment received by the receiver is 52 bytes

(c) The fragment offset of the last packet received by the receiver is 866

(d) the total number of bytes received by the transport layer of receiver is 7200 bytes

Answer :



Packet size = 7000 bytes

Payload = 6980 bytes

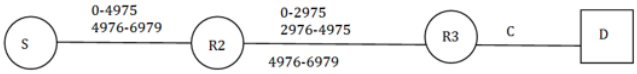
Network A → MTU (5000)

Excluding header MTU = 5000 – 20 = 4980

4980 is not divided by 8.

So, 4976 is divisible by 8.

Network A $\frac{6980}{4976}$ = 1.40 ≅ 2 fragments



Network B has MTU →3000 bytes (including header)

network B has ⇒ 2980 ⇒MTU has 2976

So only fragment 0 – 4975 is divided

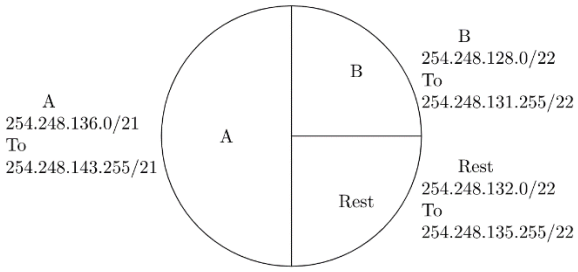
If question ask you about last IP fragment do not forget to add IP header into fragment.

Questions on IP addressing :

1) A n Internet Service Provider (ISP) has the following chunk of CIDR-based IP addresses available with it: 245.248.128.0/20. The ISP wants to give half of this chunk of addresses to Organization A, and a quarter to Organization B, while retaining the remaining with itself. Which of the following is a valid allocation of addresses to A and B?

245.248.136.0/21 and 245.248.128.0/22
245.248.132.0/22 and 245.248.132.0/21
245.248.128.0/21 and 245.248.128.0/22
245.248.136.0/24 and 245.248.132.0/2

Answer :



2) Which of the following IP address can be used in WAN?

10.0.0.1
15.1.5.6
172.16.0.10
None

Answer : 15.1.5.6 as addresses from 10.0.0.1 to 172.16.5.6 is private network thus cannot be used for WAN.

3) Which of the following can be used as both Source and Destination IP?

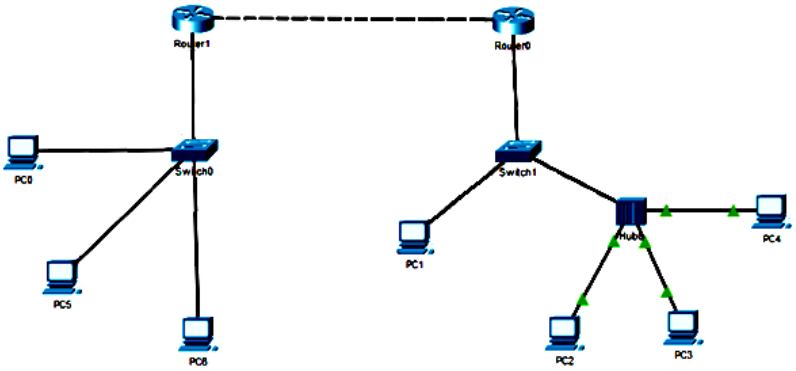
198.168.1.255
127.0.0.1
10.0.0.1
255.255.255.255

Answer : This is type of question where you have to remember all the things no other option.
192.168.1.255 is a Direct Broadcast Address (DBA), DBA can't be used as Source Address.
127.0.0.1 Loopback address again can't be used as the source address.
255.255.255.255 is Limited Broadcast Address, which cannot be used as DBA.
10.0.0.1 can be both used as both source and destination IP address.

4) Every host in an IPv4 network has a resolution 1-second real-time clock with battery backup. Each host needs to generate up to unique 50 identifiers per second. Assume that each host has a globally unique IPv4 address. Design a globally unique ID for this purpose. After what period (in seconds) will the identifiers generated by a host wrap around ?

Answer : In worst case scenario can be that all 2^{32} host are present on the network each generating 1000 packets simultaneously in 1 second. So, total packet produced in 1 second = 2^{32} x 2^{10}(approx. of 1000) = 2^{42}.Now, we can distinguish 2^{50} packets, after that wrap around(so warp around time will be when 2^{50} identifiers are used). 2^{42} takes 1 second, 2^{50} will take 256 seconds.

Consider the following network:

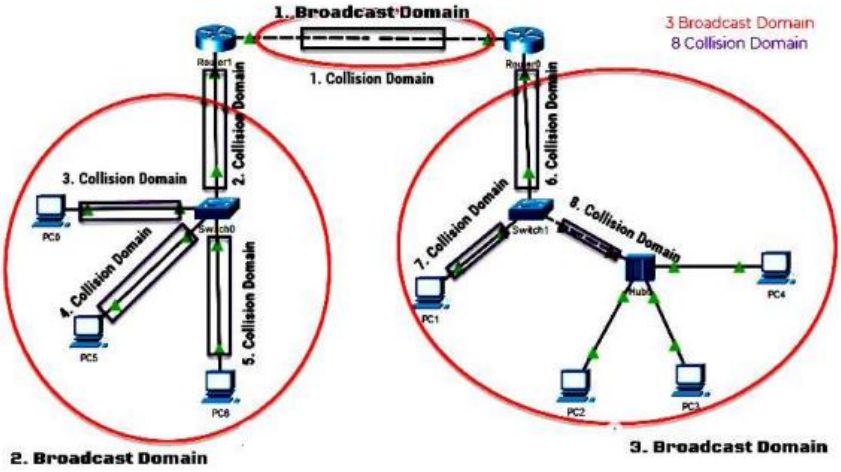


The number of broadcast domain and collision domain in the above network respectively is

(a) 2, 8 (b) 3, 6 (c) 3, 8 (d) 2, 6

Answer: (c)

Solution:



TRANSPORT LAYER

Introduction

It provides a process-to-process communication between two application layers, one at the local host and the other at the remote host.

1) **Services** :

a) *Process-to-Process Communication* : The network layer is responsible for communication at the computer level (host-to-host communication). A network-layer protocol can deliver the message only to the destination computer. However, this is an incomplete delivery. The message still needs to be handed to the correct process. This is where a transport-layer protocol takes over. A transport-layer protocol is responsible for delivery of the message to the appropriate process.

Figure 23.2 *Network layer versus transport layer*

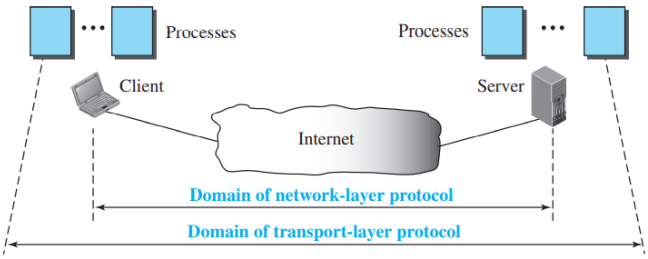
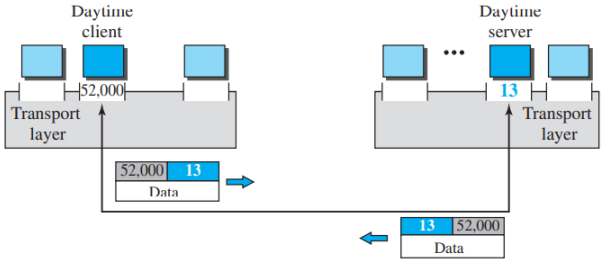


Figure 23.3 *Port numbers*



b) *Addressing* : **Port Numbers** : A process on the local host, called a client, needs services from a process usually on the remote host, called a server.

The local host and the remote host are defined using IP addresses. To define the processes, we need second identifiers, called **port numbers**. In the TCP/IP protocol suite, the port numbers are integers between 0 and 65,535 (16 bits). The client program defines itself with a port number, called the **ephemeral port number**. The word ephemeral means "short-lived" and is used because the life of a client is normally short. An ephemeral port number is recommended to be greater than 1023 for some client/server programs to work properly.

Figure 23.4 *IP addresses versus port numbers*

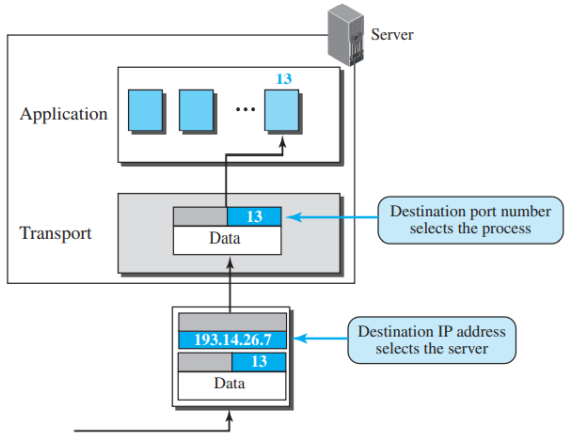
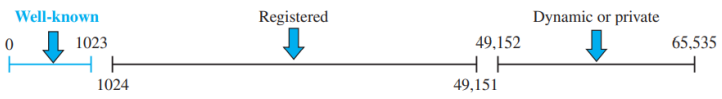


Figure 23.5 *ICANN ranges*



- ***Well-known ports.*** The ports ranging from 0 to 1023 are assigned and controlled by ICANN. These are the well-known ports.
- ***Registered ports.*** The ports ranging from 1024 to 49,151 are not assigned or controlled by ICANN. They can only be registered with ICANN to prevent duplication.
- ***Dynamic ports.*** The ports ranging from 49,152 to 65,535 are neither controlled nor registered. They can be used as temporary or private port numbers.

Figure 23.6 *Socket address*

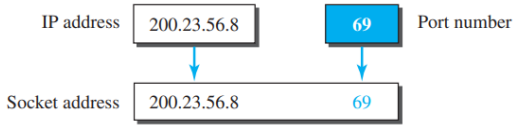
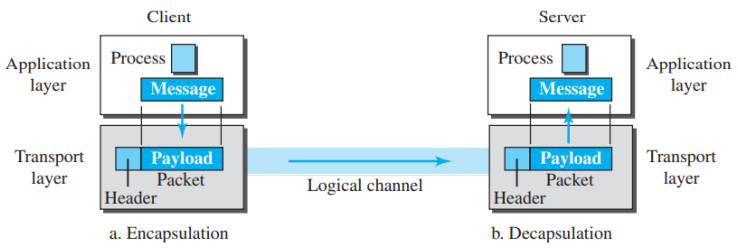


Figure 23.7 *Encapsulation and decapsulation*



c) *Multiplexing and Demultiplexing* : Whenever an entity accepts items from more than one source, this is referred to as **multiplexing (many to one)**; whenever an entity delivers items to more than one source, this is referred to as **demultiplexing (one to many)**. The transport layer at the source performs multiplexing; the transport layer at the destination performs demultiplexing.

The client process P2 needs to send a request to the corresponding server process running at another server. The transport layer at the client site accepts three messages from the three processes and creates three packets. It acts as a multiplexer. The packets 1 and 3 use the same logical channel to reach the transport layer of the first server. When they arrive at the server, the transport layer does the job of a demultiplexer and distributes the messages to two different processes. The transport layer at the second server receives packet 2 and delivers it to the corresponding process. Note that we still have demultiplexing although there is only one message.

Figure 23.8 *Multiplexing and demultiplexing*

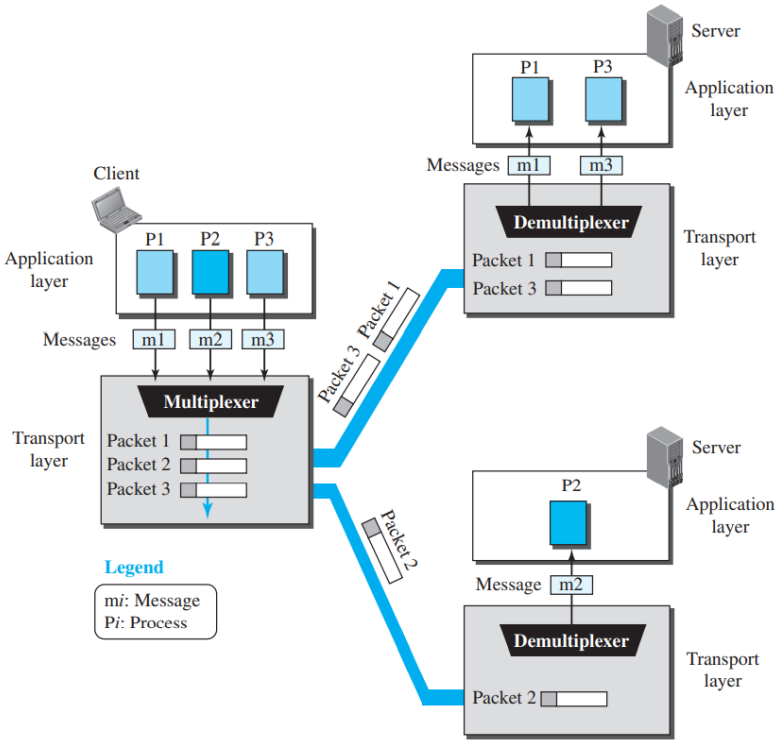
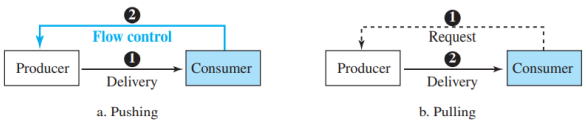


Figure 23.9 *Pushing or pulling*



d) *Flow control* : Whenever an entity produces items and another entity consumes them, there should be a balance between production and consumption rates.

i) Pushing or Pulling : Delivery of items from a producer to a consumer can occur in one of two ways: pushing or pulling. If the sender delivers items whenever they are produced–without a prior request from the consumer–the delivery is referred to as **pushing**. If the producer delivers the items after the consumer has requested them, the delivery is referred to as **pulling**. Figure 23.9 shows these two types of delivery.

ii) Flow Control at Transport Layer :

Figure 23.10 *Flow control at the transport layer*

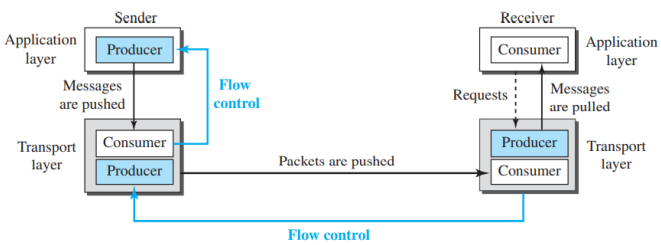
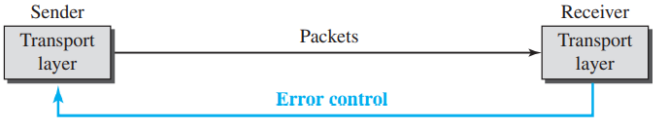


Figure 23.11 *Error control at the transport layer*



iii) *Buffers* : Although flow control can be implemented in several ways, one of the solutions is normally to use two buffers: one at the sending transport layer and the other at the receiving transport layer. A buffer is a set of memory locations that can hold packets at the sender and receiver. The flow control communication can occur by sending signals from the consumer to the producer.

e) *Error Control* : Error control, unlike flow control, involves only the sending and receiving transport layers. We are assuming that the message chunks exchanged between the application and transport layers are error free.

i) Sequence Numbers : Error control requires that the sending transport layer knows which packet is to be resent and the receiving transport layer knows which packet is a duplicate, or which packet has arrived out of order. This can be done if the packets are numbered. We can add a field to the transport-layer packet to hold the sequence number of the packet.

For error control, the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.

ii) Acknowledgment : The sender can detect lost packets if it uses a timer. When a packet is sent, the sender starts a timer. If an ACK does not arrive before the timer expires, the sender resends the packet.

f) *Combination of Flow and Error Control* : We have discussed that flow control requires the use of two buffers, one at the sender site and the other at the receiver site. At the sender, when a packet is prepared to be sent, we use the number of the next free location, x, in the buffer as the sequence number of the packet. When the packet is sent, a copy is stored at memory location x, awaiting the acknowledgment from the other end. When an acknowledgment related to a sent packet arrives, the packet is purged and the memory location becomes free. At the receiver, when a packet with sequence number y arrives, it is stored at the memory location y until the application layer is ready to receive it. An acknowledgment can be sent to announce the arrival of packet y.

Figure 23.12 *Sliding window in circular format*

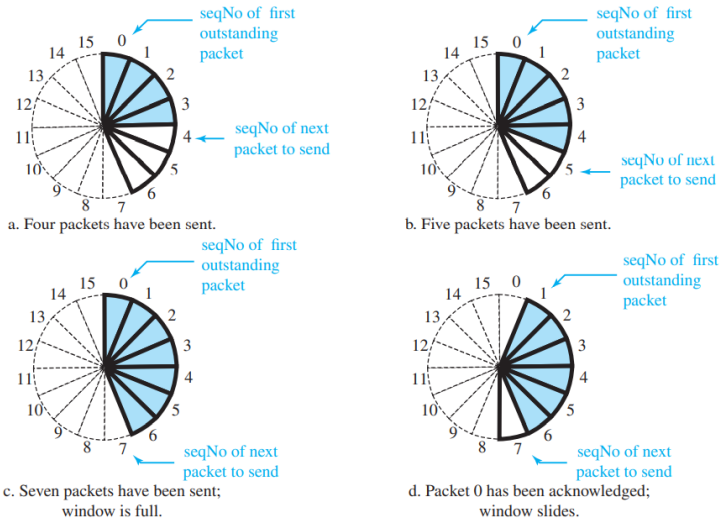
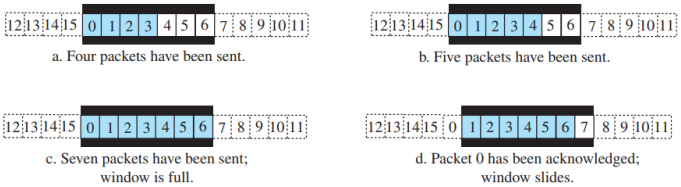


Figure 23.13 *Sliding window in linear format*



Sliding Window : Since the sequence numbers use modulo 2^m, a circle can represent the sequence numbers from 0 to $2^m - 1$. Sequence number is random number selected by computer.

2) Connectionless and Connection-Oriented Protocols :

At the transport layer, we are not concerned about the physical paths of packets (we assume a logical connection between two transport layers). Connectionless service at the transport layer means independency between packets; connection-oriented means dependency.

A) Connectionless Service : The transport layer treats each chunk as a single unit without any relation between the chunks. When a chunk arrives from the application layer, the transport layer encapsulates it in a packet and sends it.

Figure 23.14 *Connectionless service*

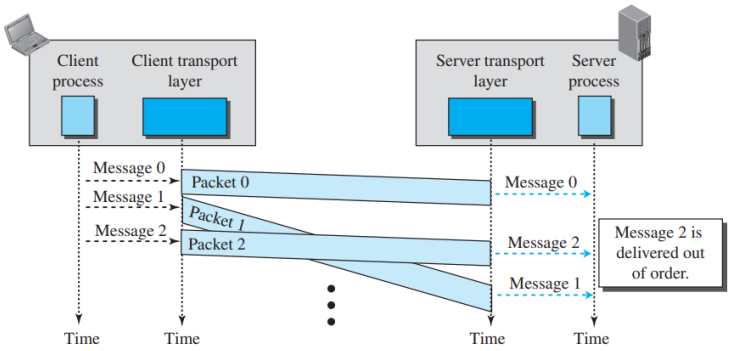
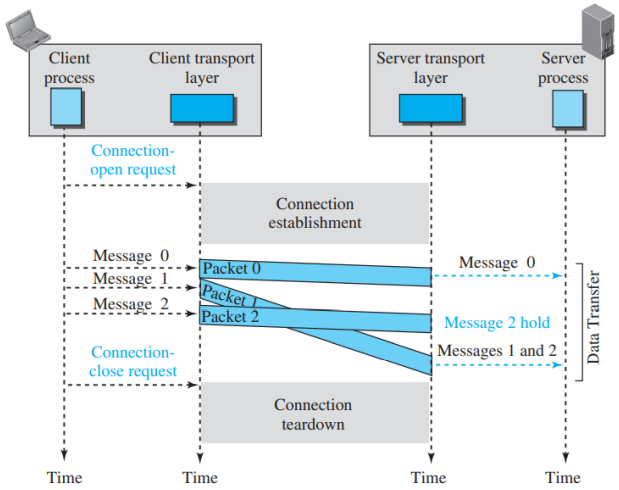


Figure 23.15 *Connection-oriented service*

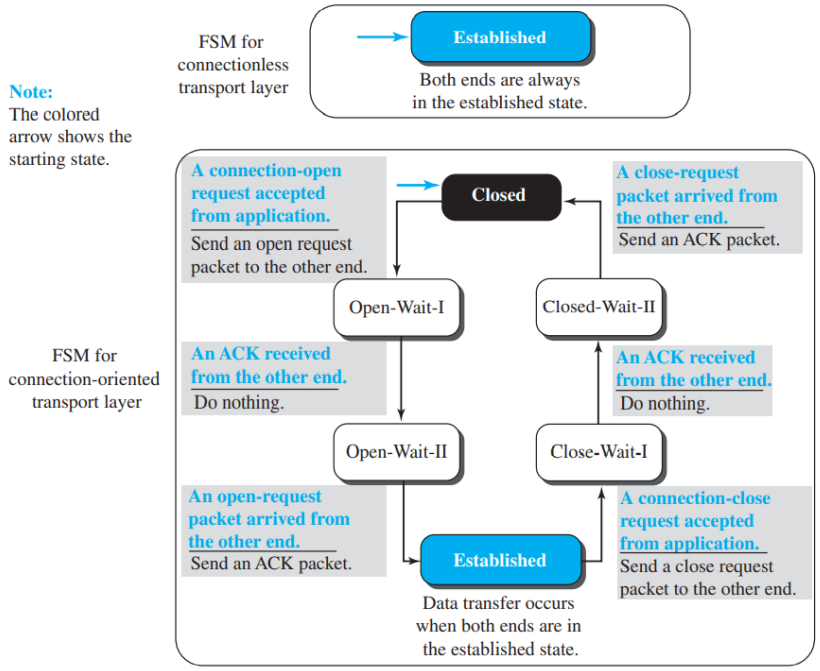


The situation would be worse if one of the packets were lost. Since there is no numbering on the packets, the receiving transport layer has no idea that one of the messages has been lost. It just delivers two chunks of data to the server process. The above two problems arise from the fact that the two transport layers do not coordinate with each other.

We can say that no flow control, error control, or congestion control can be effectively implemented in a connectionless service.

B) **Connection-oriented Services** : In a connection-oriented service, the client and the server first need to establish a logical connection between themselves. The data exchange can only happen after the connection establishment. After data exchange, the connection needs to be teardown (Figure 23.15).

Figure 23.16 *Connectionless and connection-oriented service represented as FSMs*



- **TRANSPORT LAYER PROTOCOLS** :

The TCP/IP protocol uses a transport-layer protocol that is either a modification or a combination of some of these protocols.

1) **Simple protocol** : Our first protocol is a simple **connectionless protocol** with neither flow nor error control. We assume that the receiver can immediately handle any packet it receives. In other words, the receiver can never be overwhelmed with incoming packets. Figure 23.17 shows the layout for this protocol.

Figure 23.17 *Simple protocol*

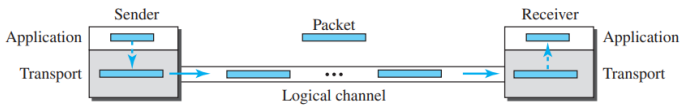


Figure 23.18 *FSMs for the simple protocol*

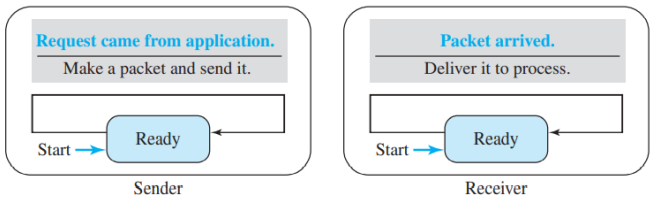
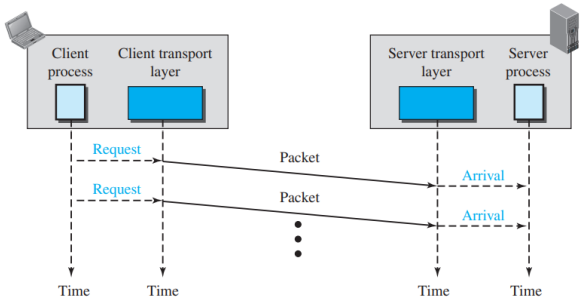


Figure 23.19 *Flow diagram for Example 23.3*



2) **Stop and wait protocol** : Which uses both flow and error control. Both the sender and the receiver use a sliding window of size 1. The sender sends one packet at a time and waits for an acknowledgment before sending the next one. To detect corrupted packets, we need to add a checksum to each data packet. When a packet arrives at the receiver site, it is checked. If its checksum is incorrect, the packet is corrupted and silently discarded.

Figure 23.20 *Stop-and-Wait protocol*

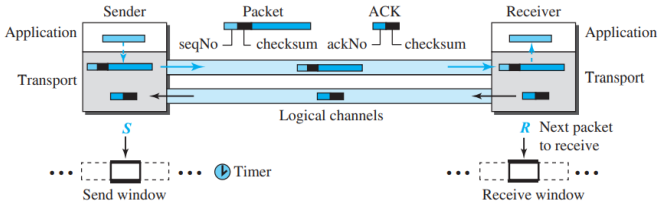


Figure 23.21 *FSMs for the Stop-and-Wait protocol*

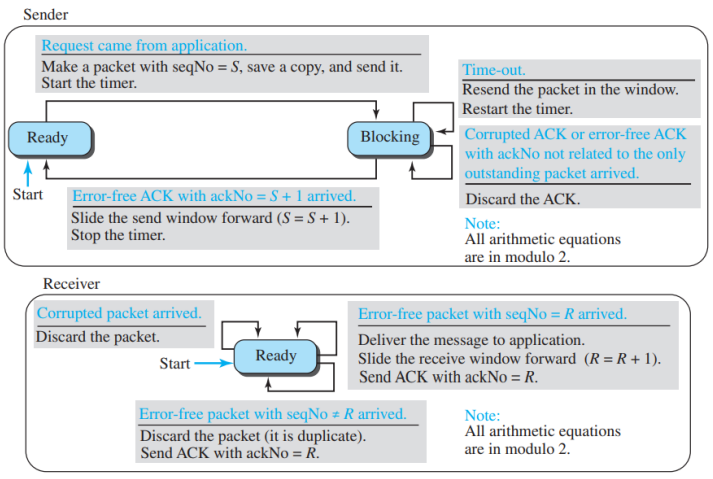
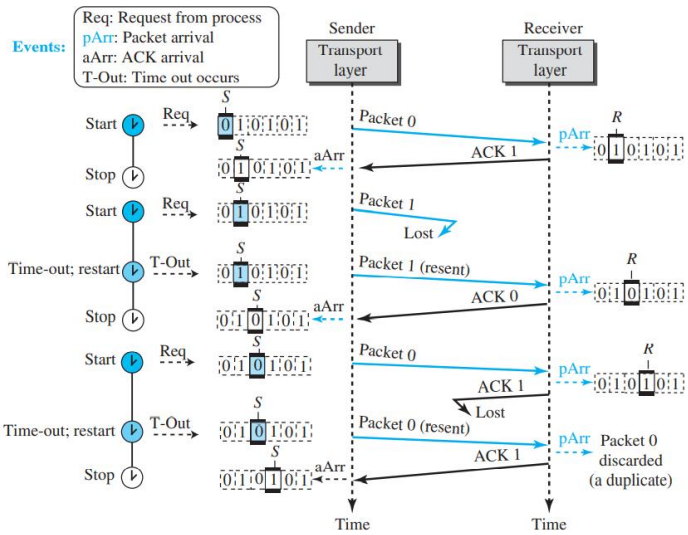


Figure 23.22 *Flow diagram for Example 23.4*



- ⇨ Acknowledgement numbers and sequence numbers : For example, if packet 0 has arrived safe and sound, the receiver sends an ACK with acknowledgment 1 (meaning packet 1 is expected next). If packet 1 has arrived safe and sound, the receiver sends an ACK with acknowledgment 0 (meaning packet 0 is expected).
- ⇨ **Efficiency :** The Stop-and-Wait protocol is very inefficient if our channel is thick and long. By thick, we mean that our channel has a large bandwidth (high data rate); by long, we mean the round-trip delay is long. The product of these two is called the **bandwidth-delay product**. We can think of the channel as a pipe. The bandwidth-delay product then is the volume of the pipe in bits. The pipe is always there. It is not efficient if it is not used. The bandwidth-delay product is a measure of the number of bits a sender can transmit through the system while waiting for an acknowledgment from the receiver.

Question on stop and wait :

1) Assume that, in a Stop-and-Wait system, the bandwidth of the line is 1 Mbps, and 1 bit takes 20 milliseconds to make a round trip. What is the bandwidth-delay product? If the system data packets are 1,000 bits in length, what is the utilization percentage of the link?

Solution : The bandwidth-delay product is $(1 \times 106) \times (20 \times 10{-}3) = 20{,}000$ bits. The system can send 20,000 bits during the time it takes for the data to go from the sender to the receiver and the acknowledgment to come back. However, the system sends only 1,000 bits. We can say that the link utilization is only 1,000/20,000, or 5 percent. For this reason, in a link with a high bandwidth or long delay, the use of Stop-and-Wait wastes the capacity of the link.

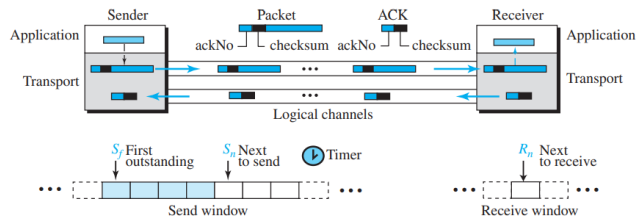
2) What is the utilization percentage of the link in question 1 if we have a protocol that can send up to 15 packets before stopping and worrying about the acknowledgments?

Solution : The bandwidth-delay product is still 20,000 bits. The system can send up to 15 packets or 15,000 bits during a round trip. This means the utilization is 15,000/20,000, or 75 percent. Of course, if there are damaged packets, the utilization percentage is much less because packets have to be resent……..END

Pipelining : In networking and in other areas, a task is often begun before the previous task has ended. This is known as pipelining. There is no pipelining in the Stop-and-Wait protocol because a sender must wait for a packet to reach the destination and be acknowledged before the next packet can be sent. Pipelining improves the efficiency of the transmission if the number of bits in transition is large with respect to the bandwidth-delay product.

3) **GO-BACK-N protocol** : To improve the efficiency of transmission (to fill the pipe), multiple packets must be in transition while the sender is waiting for acknowledgment. In other words, we need to let more than one packet be outstanding to keep the channel busy while the sender is waiting for acknowledgment. The key to Go-back-N is that we can send several packets before receiving acknowledgments, but the receiver can only buffer one packet. We keep a copy of the sent packets until the acknowledgments arrive.

Figure 23.23 *Go-Back-N protocol*



- As we mentioned before, the sequence numbers are modulo 2^m, where m is the size of the sequence number field in bits.
- An acknowledgment number in this protocol is cumulative and defines the sequence number of the next packet expected. For example, if the acknowledgment number (ackNo) is 7, it means all packets with sequence number up to 6 have arrived, safe and sound, and the receiver is expecting the packet with sequence number 7.

Figure 23.24 *Send window for Go-Back-N*

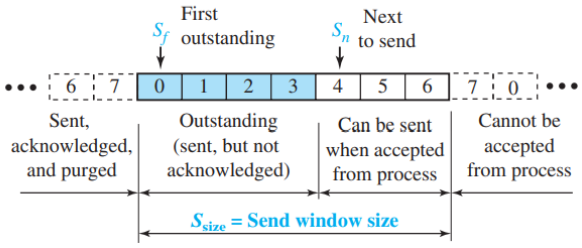


Figure 23.25 *Sliding the send window*

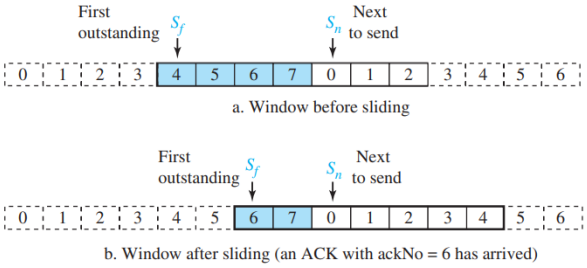
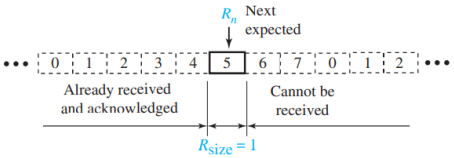
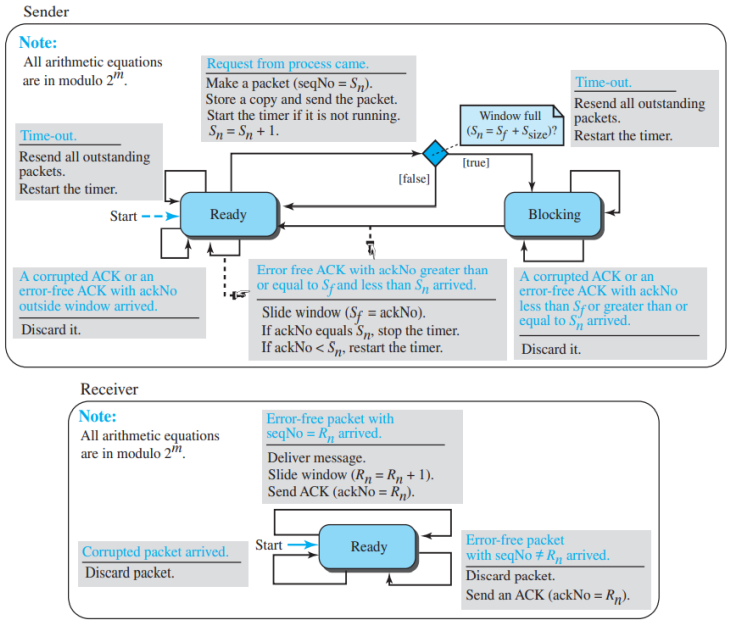


Figure 23.26 *Receive window for Go-Back-N*



The receive window is an abstract concept defining an imaginary box of size 1 with a single variable R_n. The window slides when a correct packet has arrived; sliding occurs one slot at a time.

Figure 23.27 *FSMs for the Go-Back-N protocol*



- **Timers** : Although there can be a timer for each packet that is sent, in our protocol we use only one. The reason is that the timer for the first outstanding packet always expires first. We resend all outstanding packets when this timer expires.
- *Resending packets* : When the timer expires, the sender resends all outstanding packets. For example, suppose the sender has already sent packet 6 (Sn = 7), but the only timer expires. If Sf = 3, this means that packets 3, 4, 5, and 6 have not been acknowledged; the sender goes back and resends packets 3, 4, 5, and 6. That is why the protocol is called Go-Back-N. On a time-out, the machine goes back N locations and resends all packets.
- *Send Window Size* : We can now show why the size of the send window must be less than 2^m. As an example, we choose m = 2, which means the size of the window can be $2^m - 1$, or 3. Figure 23.28 compares a window size of 3 against a window size of 4. This shows that the size of the send window must be less than 2^m.

In the Go-Back-*N* protocol, the size of the send window must be less than 2^m; the size of the receive window is always 1.

Figure 23.28 *Send window size for Go-Back-N*

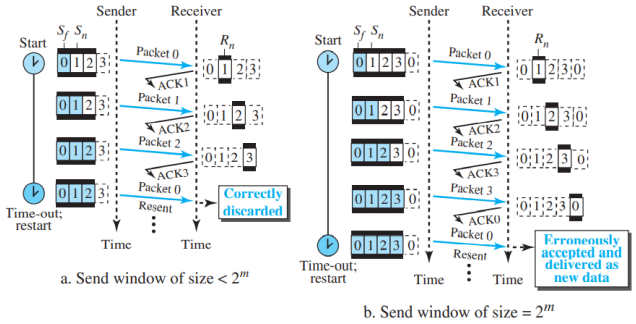


Figure 23.29 *Flow diagram for Example 23.7*

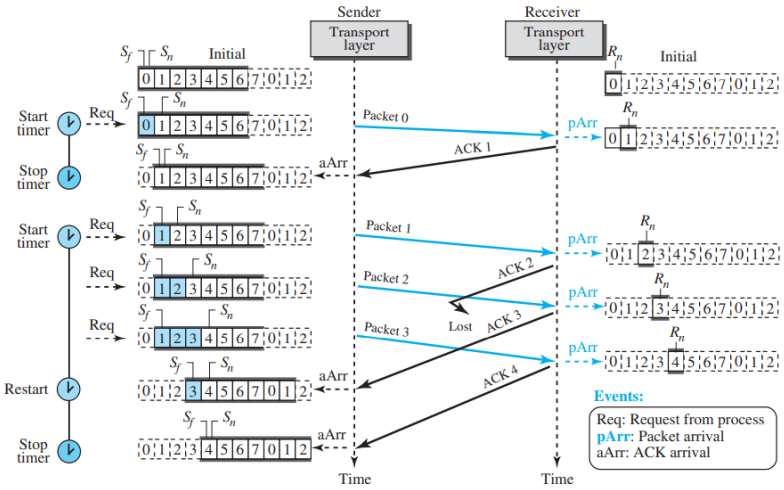
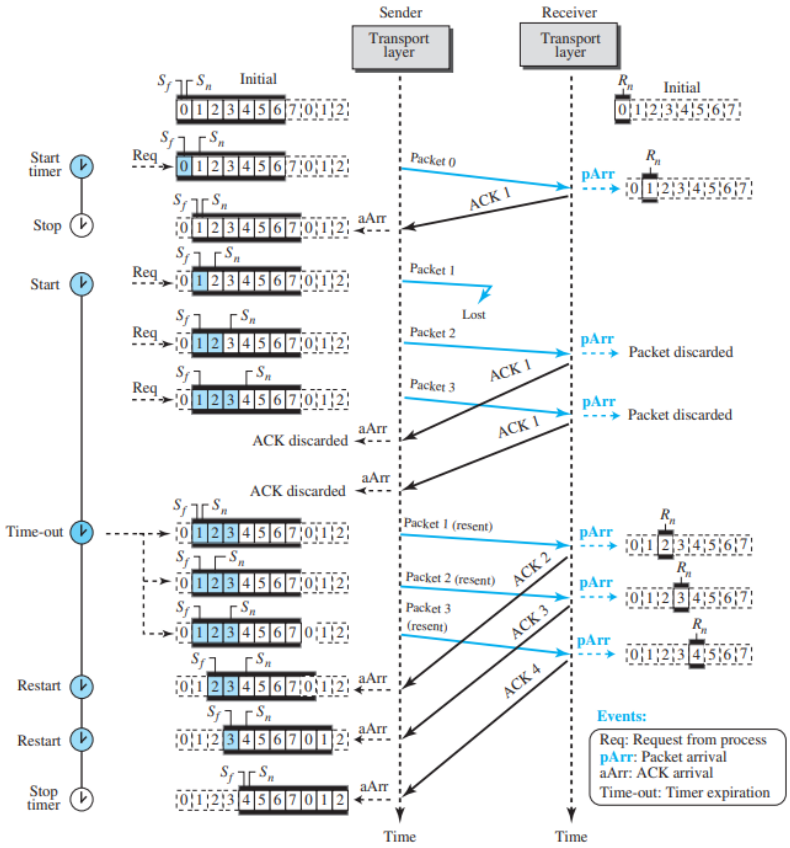
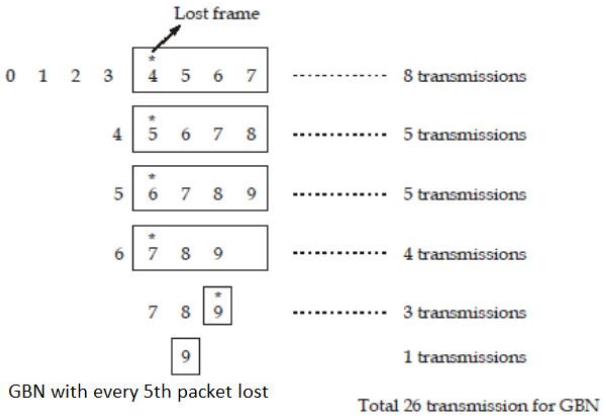


Figure 23.30 *Flow diagram for Example 23.8*





4) **Selective Repeat protocol** : which, as the name implies, resends only selective packets, those that are actually lost. The send window maximum size can be 2^{m-1}. For example, if m = 4, the sequence numbers go from 0 to 15, but the maximum size of the sender window is just 8 (it is 15 in the Go-Back-N Protocol).

Figure 23.31 *Outline of Selective-Repeat*

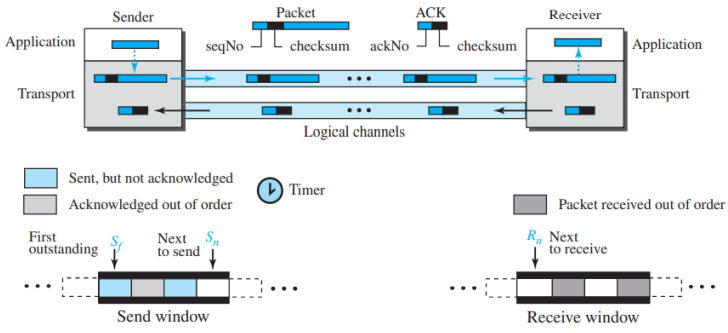


Figure 23.32 *Send window for Selective-Repeat protocol*

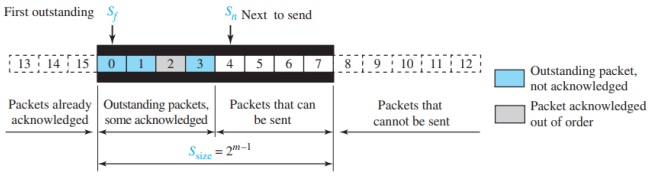
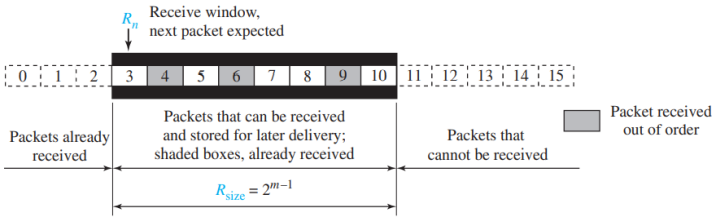


Figure 23.33 *Receive window for Selective-Repeat protocol*



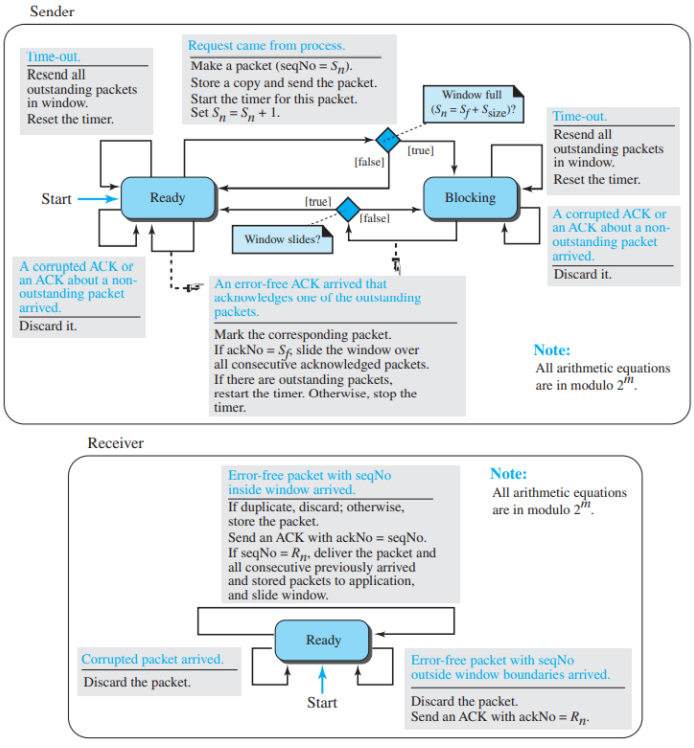
⇨ **Timer** : Theoretically, Selective-Repeat uses one timer for each outstanding packet. When a timer expires, only the corresponding packet is resent. In other words, GBN treats outstanding packets as a group; SR treats them individually. However, most transport-layer protocols that implement SR use only a single timer. For this reason, we use only one timer.

⇨ Acknowledgement : In the Selective-Repeat protocol, an acknowledgment number defines the sequence number of the error-free packet received.

Example 23.9 Assume a sender sends 6 packets: packets 0, 1, 2, 3, 4, and 5. The sender receives an ACK with ackNo = 3. What is the interpretation if the system is using GBN or SR?

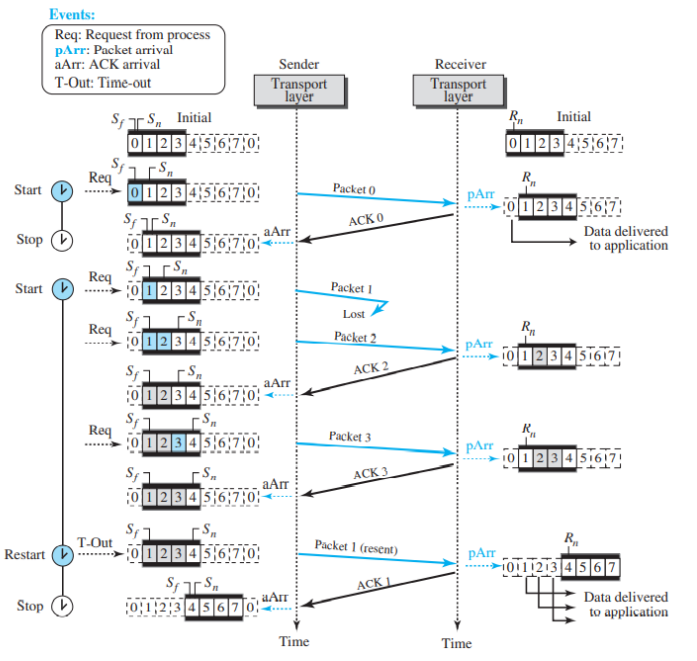
Solution : If the system is using GBN, it means that packets 0, 1, and 2 have been received uncorrupted and the receiver is expecting packet 3. If the system is using SR, it means that packet 3 has been received uncorrupted; the ACK does not say anything about other packets.

Figure 23.34 *FSMs for SR protocol*



This example is similar to Example 23.8 (Figure 23.30) in which packet 1 is lost. We show how Selective-Repeat behaves in this case. Figure 23.35 shows the situation.

Figure 23.35 *Flow diagram for Example 23.10*



- ⇨ Window Sizes : We can now show why the size of the sender and receiver windows can be at most one-half of 2^m. For an example, we choose m = 2, which means the size of the window is $2^m/2$ or $2^{(m-1)} = 2$. Figure 23.36 compares a window size of 2 with a window size of 3.
- ⇨ **Piggybacking** : The four protocols we discussed earlier in this section are all unidirectional: data packets flow in only one direction and acknowledgments travel in the other direction. In real life, data packets are normally flowing in both directions: from client to server and from server to client. This means that acknowledgments also need to flow in both directions. A technique called piggybacking is used to improve the efficiency of the bidirectional protocols. When a packet is carrying data from A to B, it can also carry acknowledgment feedback about arrived packets from B; when a packet is carrying data from B to A, it can also carry acknowledgment feedback about the arrived packets from A.

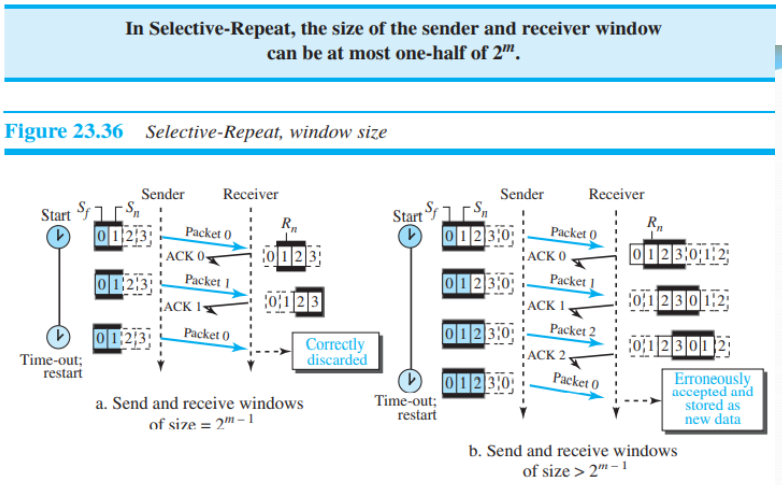


Figure 23.36 *Selective-Repeat, window size*

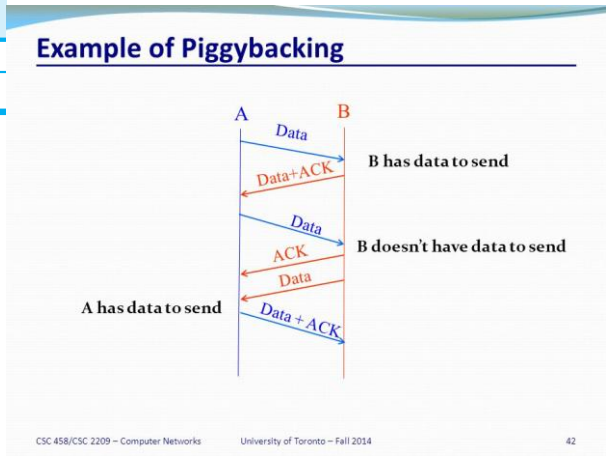
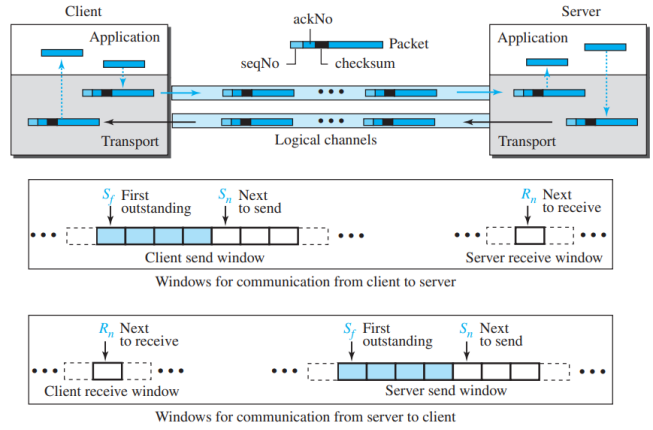
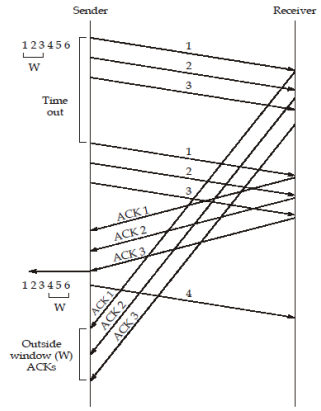


Figure 23.37 *Design of piggybacking in Go-Back-N*



SR allowed out-of-order correct packet delivery.
Consider following situation



This situation possible in SR and GBN both.

Note : In case of SR and Go-back-N it is possible to receive a ACK for a packet that falls outside of its current window. Consider above image.

Transport Layer Protocol

Services Each protocol provides a different type of service and should be used appropriately.
UDP: UDP is an unreliable, connectionless transport-layer protocol used for its simplicity and efficiency in applications where error control can be provided by the application-layer process.
TCP : TCP is a reliable connection-oriented protocol that can be used in any application where reliability is important.

SCTP : SCTP is a new transport-layer protocol that combines the features of UDP and TCP.

Figure 24.1 *Position of transport-layer protocols in the TCP/IP protocol suite*

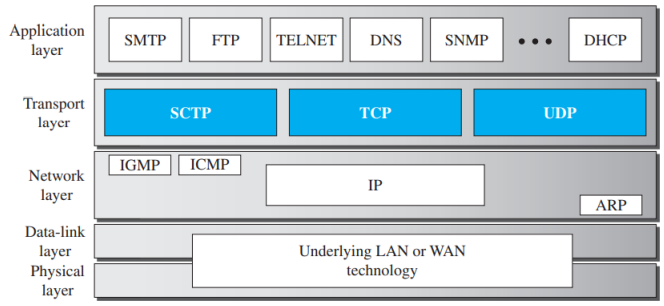


Table 24.1 *Some well-known ports used with UDP and TCP*

Port	*Protocol*	*UDP*	*TCP*	*SCTP*	*Description*
7	Echo	√	√	√	Echoes back a received datagram
9	Discard	√	√	√	Discards any datagram that is received
11	Users	√	√	√	Active users
13	Daytime	√	√	√	Returns the date and the time
17	Quote	√	√	√	Returns a quote of the day
19	Chargen	√	√	√	Returns a string of characters
20	FTP-data		√	√	File Transfer Protocol
21	FTP-21		√	√	File Transfer Protocol
23	TELNET		√	√	Terminal Network
25	SMTP		√	√	Simple Mail Transfer Protocol
53	DNS	√	√	√	Domain Name Service
67	DHCP	√	√	√	Dynamic Host Configuration Protocol
69	TFTP	√	√	√	Trivial File Transfer Protocol
80	HTTP		√	√	HyperText Transfer Protocol
111	RPC	√	√	√	Remote Procedure Call
123	NTP	√	√	√	Network Time Protocol
161	SNMP-server	√			Simple Network Management Protocol
162	SNMP-client	√			Simple Network Management Protocol

Port Numbers : Port numbers provide end-to-end addresses at the transport layer and allow multiplexing and demultiplexing at this layer, just as IP addresses do at the network layer.

- **UDP (User datagram protocol) :**

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

UDP packets, called user datagrams, have a fixed-size header of 8 bytes made of four fields, each of 2 bytes (16 bits).

Example 24.1

The following is the content of a UDP header in hexadecimal format.

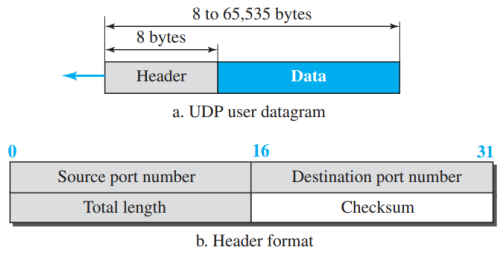
CB84000D001C001C

a. What is the source port number?
b. What is the destination port number?
c. What is the total length of the user datagram?
d. What is the length of the data?
e. Is the packet directed from a client to a server or vice versa?
f. What is the client process?

Solution

a. The source port number is the first four hexadecimal digits $(CB84)_{16}$, which means that the source port number is 52100.
b. The destination port number is the second four hexadecimal digits $(000D)_{16}$, which means that the destination port number is 13.
c. The third four hexadecimal digits $(001C)_{16}$ define the length of the whole UDP packet as 28 bytes.
d. The length of the data is the length of the whole packet minus the length of the header, or $28 - 8 = 20$ bytes.
e. Since the destination port number is 13 (well-known port), the packet is from the client to the server.
f. The client process is the Daytime (see Table 24.1).

Figure 24.2 *User datagram packet format*



1) UDP Services:

a) Process-to-Process Communication : UDP provides process-to-process communication using socket addresses, a combination of IP addresses and port numbers.

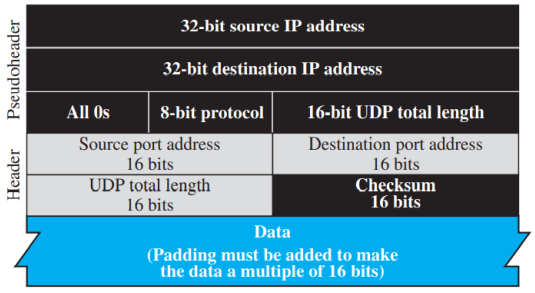
b) Connectionless services : One of the ramifications of being connectionless is that the process that uses UDP cannot send a stream of data to UDP and expect UDP to chop them into different, related user datagrams. Instead each request must be small enough to fit into one user datagram. Only those processes sending short messages, messages less than 65,507 bytes (65,535 minus 8 bytes for the UDP header and minus 20 bytes for the IP header), can use UDP.

c) UDP is a very simple protocol. There is no flow control, and hence no window mechanism.

d) There is no error control mechanism in UDP except for the checksum. This means that the sender does not know if a message has been lost or duplicated.

e) Checksum :

Figure 24.3 *Pseudoheader for checksum calculation*



Example 24.2

What value is sent for the checksum in each one of the following hypothetical situations?

a. The sender decides not to include the checksum.
b. The sender decides to include the checksum, but the value of the sum is all 1s.
c. The sender decides to include the checksum, but the value of the sum is all 0s.

Solution

a. The value sent for the checksum field is all 0s to show that the checksum is not calculated.
b. When the sender complements the sum, the result is all 0s; the sender complements the result again before sending. The value sent for the checksum is all 1s. The second complement operation is needed to avoid confusion with the case in part a.
c. This situation never happens because it implies that the value of every term included in the calculation of the sum is all 0s, which is impossible; some fields in the pseudoheader have nonzero values.

UDP is an example of the connectionless simple protocol we discussed earlier with the exception of an optional checksum added to packets for error detection.

Example 24.3 A client-server application such as DNS (see Chapter 26) uses the services of UDP because a client needs to send a short request to a server and to receive a quick response from it. The request and response can each fit in one user datagram. Since only one message is exchanged in each direction, the connectionless feature is not an issue; the client or server does not worry that messages are delivered out of order.

Example 24.4 A client-server application such as SMTP (see Chapter 27), which is used in electronic mail, cannot use the services of UDP because a user might send a long e-mail message, which could include multimedia (images, audio, or video). If the application uses UDP and the message does not fit in one user datagram, the message must be split by the application into different user datagrams. Here the connectionless service may create problems. The user datagrams may arrive and be delivered to the receiver application out of order. The receiver application may not be able to reorder the pieces. This means the connectionless service has a disadvantage for an application program that sends long messages. In SMTP, when we send a message, we do not expect to receive a response quickly (sometimes no response is required). This means that the extra delay inherent in connection-oriented service is not crucial for SMTP.

Example 24.5 Assume we are downloading a very large text file from the Internet. We definitely need to use a transport layer that provides reliable service. We don't want part of the file to be missing or corrupted when we open the file. The delay created between the deliveries of the parts is not an overriding concern for us; we wait until the whole file is composed before looking at it. In this case, UDP is not a suitable transport layer.

Example 24.6 Assume we are using a real-time interactive application, such as Skype. Audio and video are divided into frames and sent one after another. If the transport layer is supposed to resend a corrupted or lost frame, the synchronizing of the whole transmission may be lost. The viewer suddenly sees a blank screen and needs to wait until the second transmission arrives. This is not tolerable. However, if each small part of the screen is sent using a single user datagram, the receiving UDP can easily ignore the corrupted or lost packet and deliver the rest to the application program. That part of the screen is blank for a very short period of time, which most viewers do not even notice.

Typical Applications : The following shows some typical applications that can benefit more from the services of UDP than from those of TCP.

❑ UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control. It is not usually used for a process such as FTP that needs to send bulk data (see Chapter 26).

❑ UDP is suitable for a process with internal flow- and error-control mechanisms. For example, the Trivial File Transfer Protocol (TFTP) process includes flow and error control. It can easily use UDP.

❑ UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software but not in the TCP software.

❑ UDP is used for management processes such as SNMP (see Chapter 27).

❑ UDP is used for some route updating protocols such as **Routing Information Protocol** (RIP) (see Chapter 20).

❑ UDP is normally used for interactive real-time applications that cannot tolerate uneven delay between sections of a received message.

❑A connected UDP socket cannot be used to communicate with multiple peers simultaneously.

- **TRANSMISSION CONTROL PROTOCOL** : Transmission Control Protocol (TCP) is a connection-oriented, reliable protocol. TCP explicitly defines connection establishment, data transfer, and connection teardown phases to provide a connection-oriented service. TCP uses a combination of GBN and SR protocols to provide reliability. To achieve this goal, TCP uses checksum (for error detection), retransmission of lost or corrupted packets, cumulative and selective acknowledgments, and timers.

1) Services :

a) *Process-to-Process Communication* : As with UDP, TCP provides process-to-process communication using port numbers.

b) *Stream Delivery Service* : TCP, unlike UDP, is a stream-oriented protocol. TCP, on the other hand, allows the sending process to deliver data as a stream of bytes and allows the receiving process to obtain data as a stream of bytes. TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their bytes across the Internet.

c) *Buffers* : The white section contains empty chambers that can be filled by the sending process (producer). The coloured area holds bytes that have been sent but not yet acknowledged.

Figure 24.5 *Sending and receiving buffers*

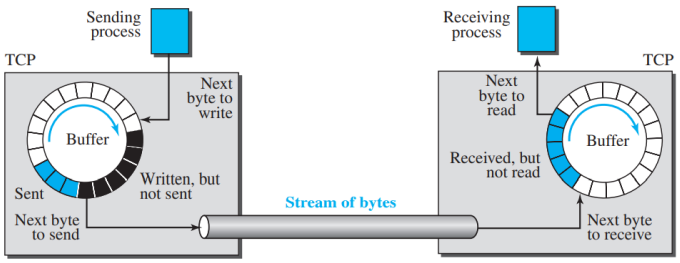
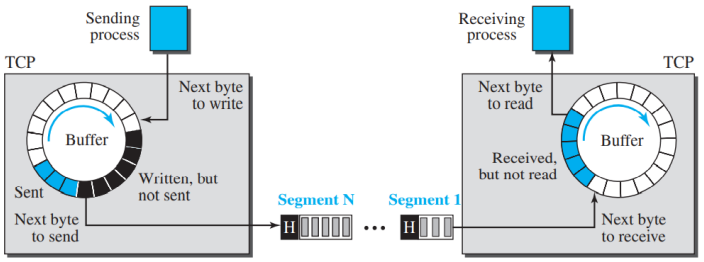


Figure 24.6 *TCP segments*



d) Full-Duplex Communication : TCP offers full-duplex service, where data can flow in both directions at the same time.

e) Multiplexing and Demultiplexing : Like UDP, TCP performs multiplexing at the sender and demultiplexing at the receiver. However, since TCP is a connection-oriented protocol, a connection needs to be established for each pair of processes.

2) Features of TCP :

a) *Numbering system* : Although the TCP software keeps track of the segments being transmitted or received, there is no field for a segment number value in the segment header. Instead, there are two fields, called the sequence number and the acknowledgment number.

1) **Byte number** : The numbering does not necessarily start from 0. Instead, TCP chooses an arbitrary number between 0 and $2^{32} - 1$ for the number of the first byte. For example, if the number happens to be 1057 and the total data to be sent is 6000 bytes, the bytes are numbered from 1057 to 7056.

2) **Sequence number** : The sequence number, in each direction, is defined as follows:
   1. The sequence number of the first segment is the ISN (initial sequence number), which is a random number.
   2. The sequence number of any other segment is the sequence number of the previous segment plus the number of bytes (real or imaginary) carried by the previous segment.

3) **Acknowledgement numbers** : The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive. The acknowledgment number is cumulative.
   The term cumulative here means that if a party uses 5643 as an acknowledgment number, it has received all bytes from the beginning up to 5642. Note that this does not mean that the party has received 5642 bytes, because the first byte number does not have to be 0.

Example 24.7

Suppose a TCP connection is transferring a file of 5000 bytes. The first byte is numbered 10001. What are the sequence numbers for each segment if data are sent in five segments, each carrying 1000 bytes?

Solution

The following shows the sequence number for each segment:

Segment 1	→	Sequence Number:	10001	**Range:**	10001	to	11000
Segment 2	→	Sequence Number:	11001	**Range:**	11001	to	12000
Segment 3	→	Sequence Number:	12001	**Range:**	12001	to	13000
Segment 4	→	Sequence Number:	13001	**Range:**	13001	to	14000
Segment 5	→	Sequence Number:	14001	**Range:**	14001	to	15000

The value in the sequence number field of a segment defines the number assigned to the first data byte contained in that segment.

4) **Segment :**

Figure 24.7 *TCP segment format*

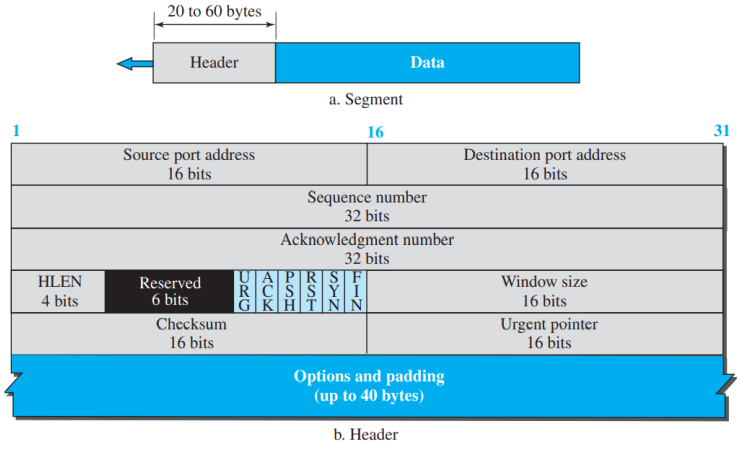


Figure 24.9 *Pseudoheader added to the TCP datagram*

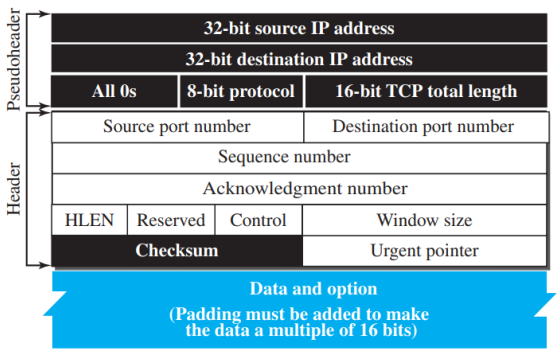
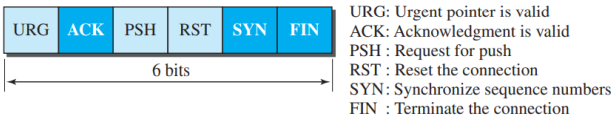


Figure 24.8 *Control field*



The use of the checksum in TCP is mandatory, whereas in case of UDP is not mandatory.

5) **A TCP Connection** : TCP is connection-oriented. As discussed before, a connection-oriented transport protocol establishes a logical path between the source and destination. In TCP, connection-oriented transmission requires three phases: **connection establishment, data transfer, and connection termination.**

Connection Establishment :

a) Three-Way Handshaking : 1. A SYN segment cannot carry data, but it consumes one sequence number. 2. A SYN + ACK segment cannot carry data, but it does consume one sequence number. 3. An ACK segment, if carrying no data, consumes no sequence number.

b) **SYN Flooding Attack** : The connection establishment procedure in TCP is susceptible to a serious security problem called SYN flooding attack. If, during this short period of time, the number of SYN segments is large, the server eventually runs out of resources and may be unable to accept connection requests from valid clients. This SYN flooding attack belongs to a group of security attacks known as a denial of service attack, in which an attacker monopolizes a system with so many service requests that the system overloads and denies service to valid requests.

One recent strategy is to postpone resource allocation until the server can verify that the connection request is coming from a valid IP address, by using what is called a cookie. Cookie is not piece of code it is just a string. SCTP, the new transport-layer protocol that we discuss later, uses this strategy.

Figure 24.10 *Connection establishment using three-way handshaking*

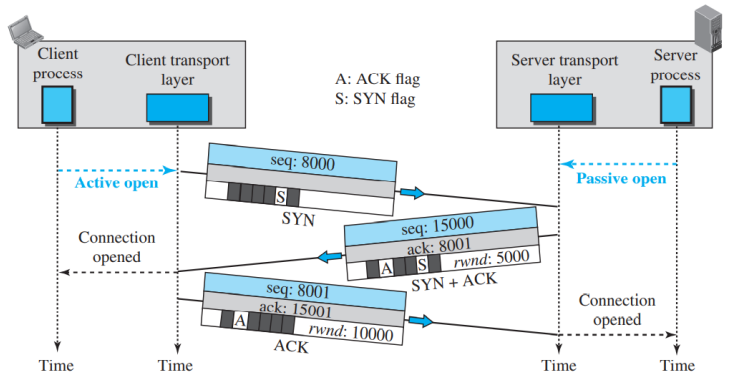
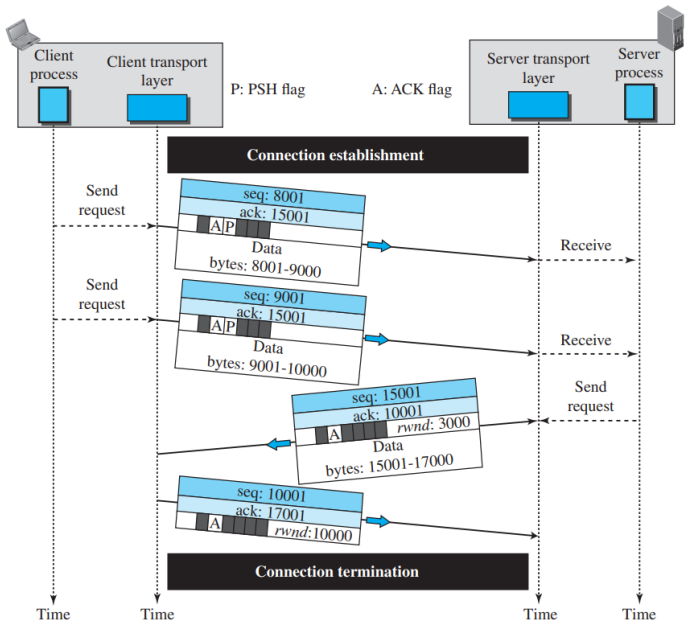


Figure 24.11 *Data transfer*



Data transfer : After connection is established, bidirectional data transfer can take place. The client and server can send data and acknowledgments in both directions.

a) Pushing Data : The sending TCP can select the segment size. The receiving TCP also buffers the data when they arrive and delivers them to the application program when the application

program is ready or when it is convenient for the receiving TCP. This type of flexibility increases the efficiency of TCP.

b) Urgent Data : TCP is a stream-oriented protocol. This means that the data is presented from the application program to TCP as a stream of bytes. Each byte of data has a position in the stream. However, there are occasions in which an application program needs to send urgent bytes, some bytes that need to be treated in a special way by the application at the other end.

Example : For example, if the segment sequence number is 15000 and the value of the urgent pointer is 200, the first byte of urgent data is the byte 15000 and the last byte is the byte 15200. The rest of the bytes in the segment (if present) are nonurgent.

Connection termination : 1) Three-Way Handshaking : 1. The FIN segment consumes one sequence number if it does not carry data. 2. If it does not carry data, it consumes only one sequence number because it needs to be acknowledged. 3. This segment cannot carry data and consumes no sequence numbers.

2) Half-Close : In TCP, one end can stop sending data while still receiving data. This is called a half close. The data transfer from the client to the server stops. The client half-closes the connection by sending a FIN segment. The server accepts the half-close by sending the ACK segment. The server, however, can still send data. When the server has sent all of the processed data, it sends a FIN segment, which is acknowledged by an ACK from the client. After half-closing the connection, data can travel from the server to the client and acknowledgments can travel from the client to the server. The client cannot send any more data to the server.

Figure 24.13 *Half-close*

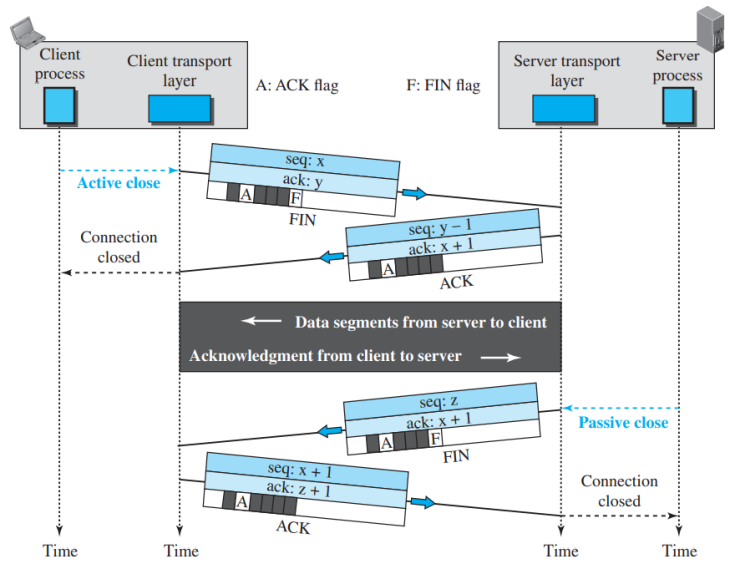
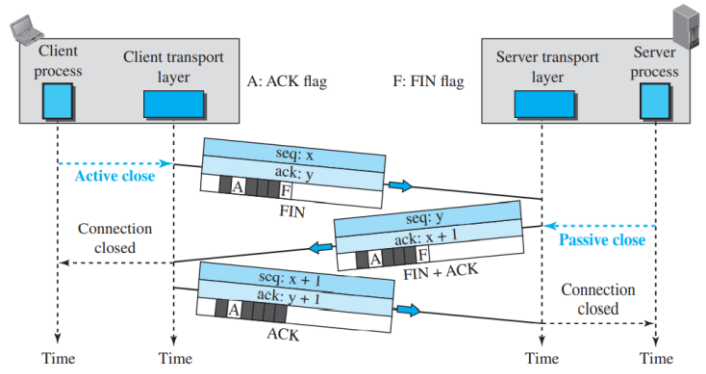


Figure 24.12 *Connection termination using three-way handshaking*



The FIN + ACK segment consumes only one sequence number if it does not carry data.

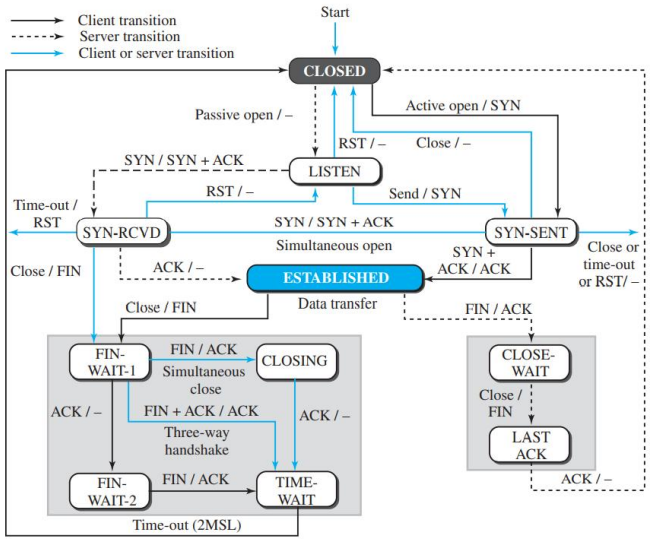
State Transition Diagram : To keep track of all the different events happening during connection establishment, connection termination, and data transfer.

The rounded-corner rectangles represent the states. The transition from one state to another is shown using directed lines. Each line has two strings separated by a slash. The first string is the input, what TCP receives. The second is the output, what TCP sends. The dotted black lines in the figure represent the transition that a server normally goes through; the solid black lines show the transitions that a client normally goes through. However, in some situations, a server transitions through a solid line or a client transitions through a dotted line. The coloured lines show special situations. Note that the rounded-corner rectangle marked ESTABLISHED is in fact two sets of states, a set for the client and another for the server, that are used for flow and error control, as explained later in the chapter.

A Half-Close Scenario : The client process issues an active open command to its TCP to request a connection to a specific **socket address**. TCP sends a SYN segment and moves to the SYN-SENT state. After receiving the SYN + ACK segment, TCP sends an ACK segment and goes to the ESTABLISHED state. Data are transferred, possibly in both directions, and acknowledged. When the client process has no more data to send, it issues a command called an active close. The TCP sends a FIN segment and goes to the FIN-WAIT-1 state. When it receives the ACK segment, it goes to the FIN-WAIT-2 state. When the client receives a FIN segment, it sends an ACK segment and goes to the TIME-WAIT state. The client remains in this state for 2 MSL seconds (see TCP timers later in the chapter). When the corresponding timer expires, the client goes to the CLOSED state.

The server process issues a passive open command. The server TCP goes to the LISTEN state and remains there passively until it receives a SYN segment. The TCP then sends a SYN + ACK segment and goes to the SYN-RCVD state, waiting for the client to send an ACK segment. After receiving the ACK segment, TCP goes to the ESTABLISHED state, where data transfer can take place. TCP remains in this state until it receives a FIN segment from the client signifying that there are no more data to be exchanged and the connection can be closed. The server, upon receiving the FIN segment, sends all queued data to the server with a virtual EOF marker, which means that the connection must be closed. It sends an ACK segment and goes to the CLOSEWAIT state, but postpones acknowledging the FIN segment received from the client until it receives a passive close command from its process. After receiving the passive close command, the server sends a FIN segment to the client and goes to the LASTACK state, waiting for the final ACK. When the ACK segment is received from the client, the server goes to the CLOSE state. Figure 24.16 shows the same scenario with states over the time line.

Figure 24.14 *State transition diagram*



The state marked *ESTABLISHED* in the FSM is in fact two different sets of states that the client and server undergo to transfer data.

Table 24.2 *States for TCP*

State	*Description*
CLOSED	No connection exists
LISTEN	Passive open received; waiting for SYN
SYN-SENT	SYN sent; waiting for ACK
SYN-RCVD	SYN + ACK sent; waiting for ACK
ESTABLISHED	Connection established; data transfer in progress
FIN-WAIT-1	First FIN sent; waiting for ACK
FIN-WAIT-2	ACK to first FIN received; waiting for second FIN
CLOSE-WAIT	First FIN received, ACK sent; waiting for application to close
TIME-WAIT	Second FIN received, ACK sent; waiting for 2MSL time-out
LAST-ACK	Second FIN sent; waiting for ACK
CLOSING	Both sides decided to close simultaneously

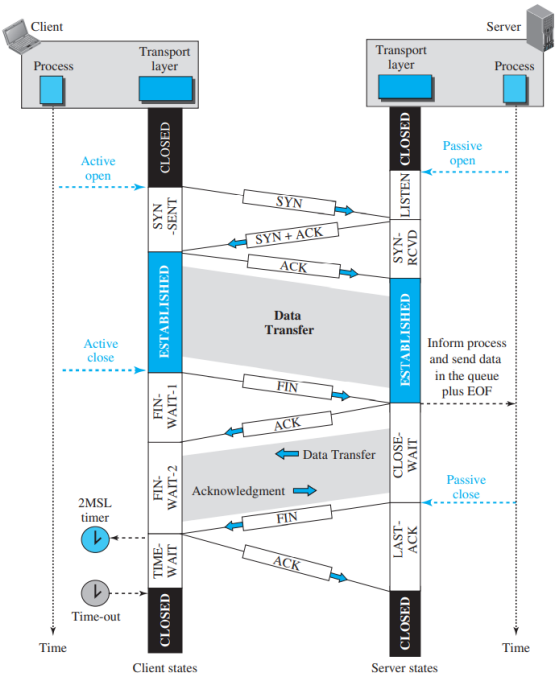
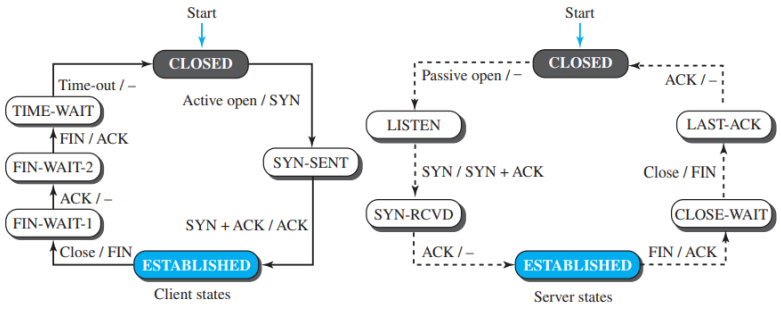


Figure 24.15 *Transition diagram with half-close connection termination*

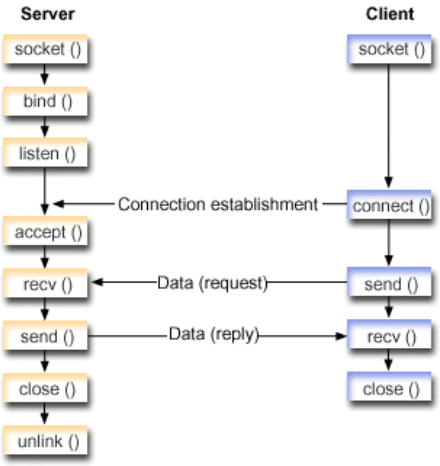


Socket programming :

socket() : creates a new socket of a certain socket type, identified by an integer number, and allocates system resources to it.

bind() : is typically used on the server side, and associates a socket with a socket address structure, i.e. a specified local port number and IP address.

listen() : is used on the server side, and causes a bound TCP socket to enter listening state. listen() marks the socket referred to by sockfd as a passive socket, that is, as a socket that will be used to accept incoming connection requests using accept().



connect() : is used on the client side, and assigns a free local port number to a socket. In case of a TCP socket, it causes an attempt to establish a new TCP connection. When connect() is called by client, following three-way handshake happens to establish the connection in TCP. 1) The client requests a connection by sending a SYN (synchronize) message to the server. 2) The server acknowledges this request by sending SYN-ACK back to the client. 3) The client responds with an ACK, and the connection is established.

accept() : It is used only with TCP sockets, because it is connection oriented and it helps to accept the request for connection.

Order in which a server process invokes the function calls - blind, Listen, accept, revc.

Windows in TCP : TCP uses two windows (send window and receive window) for each direction of data transfer, which means four windows for a bidirectional communication.

Send Window : The window size is 100 bytes, but later we see that the send window size is dictated by the receiver.

1. The window size in SR is the number of packets, but the window size in TCP is the number of bytes.

2. TCP can store data received from the process and send them later, but we assume that the sending TCP is capable of sending segments of data as soon as it receives them from its process.
3. Another difference is the number of timers.

Receive Window : The window size is 100 bytes. in practice, the window should never shrink.

1. the receive window size, normally called rwnd, can be determined as: rwnd = buffer size - number of waiting bytes to be pulled
2. The second difference is the way acknowledgments are used in the TCP protocol.

Figure 24.17 *Send window in TCP*

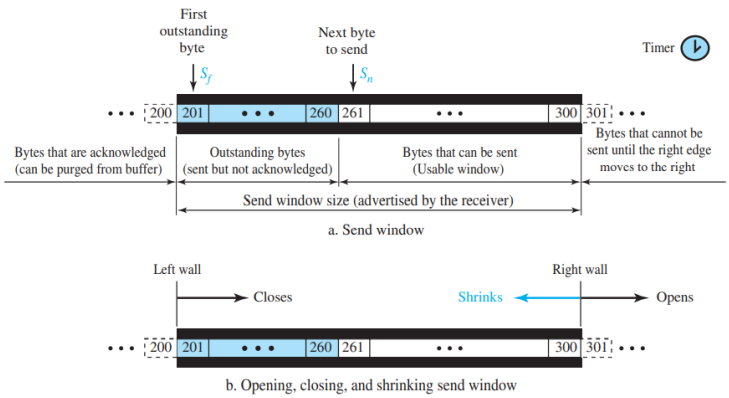
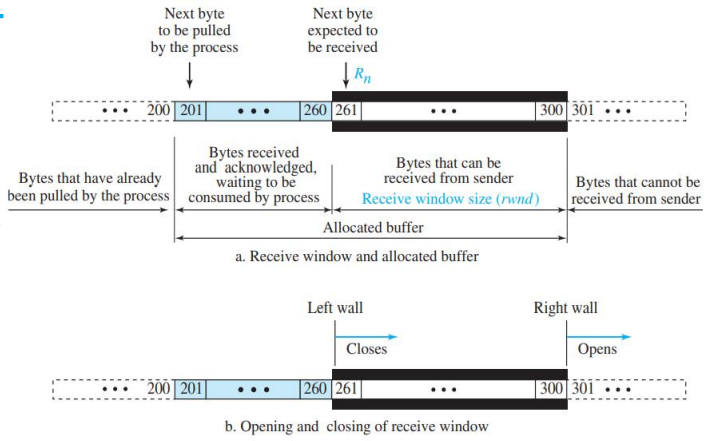
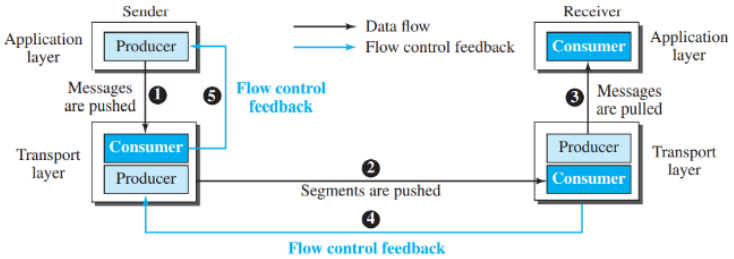


Figure 24.18 *Receive window in TCP*



Advertised Window gives the amount of free space available in the receive buffer.
e.g if maximum receiver window size is 1000 bytes and till 500 are received then 500 bytes are free space hence advertised window is 500.

Figure 24.19 *Data flow and flow control feedbacks in TCP*



Flow Control : flow control balances the rate a producer creates data with the rate a consumer can use the data.

a) Opening and Closing Windows : The opening, closing, and shrinking of the send window is controlled by the receiver. The send window closes (moves its left wall to the right) when a new acknowledgment allows it to do so. The send window opens (its right wall moves to the right) when the receive window size (rwnd) advertised by the receiver allows it to do so (new ackNo + new rwnd > last ackNo + last rwnd). The send window shrinks in the event this situation does not occur.

Figure 24.20 *An example of flow control*

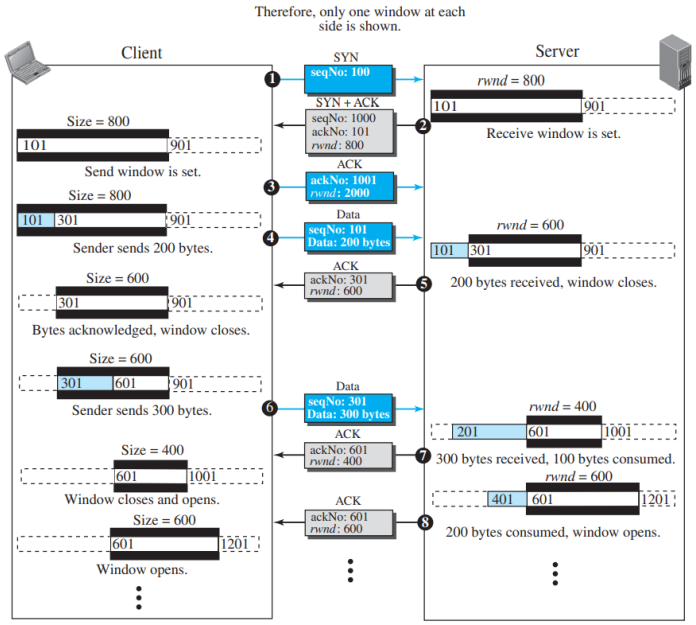
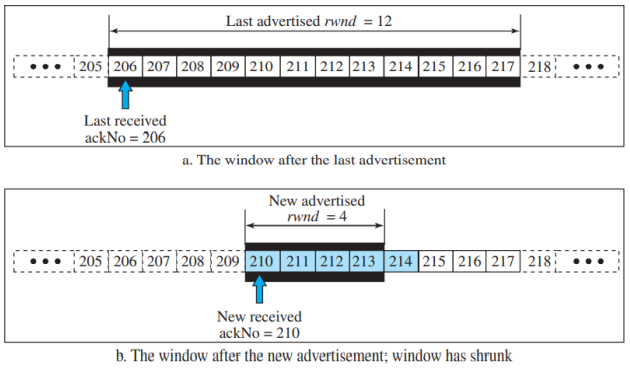


Figure 24.21 *Example 24.8*



Shrinking of window : The inequality is a mandate for the receiver to check its advertisement. However, note that the inequality is valid only if Sf < Sn; we need to remember that all calculations are in modulo 2^{32}.

new ackNo + new rwnd ≥ last ackNo + last rwnd

Window shutdown : The receiver can temporarily shut down the window by sending a rwnd of 0. This can happen if for some reason the receiver does not want to receive any data from the sender for a while. even when the window is shut down by an order from the receiver, the sender can always send a segment with 1 byte of data. This is called **probing** and is used to prevent a deadlock.

Silly Window Syndrome : A serious problem can arise in the sliding window operation when either the sending application program creates data slowly or the receiving application program consumes data slowly, or both. Any of these situations results in the sending of data in very small segments, which reduces the efficiency of the operation. This problem is called the **silly window syndrome**.

A) Syndrome Created by the Sender : The sending TCP may create a silly window syndrome if it is serving an application program that creates data slowly.

Nagle found an elegant solution. **Nagle's algorithm** is simple:

1. The sending TCP sends the first piece of data it receives from the sending application program even if it is only 1 byte.
2. After sending the first segment, the sending TCP accumulates data in the output buffer and waits until either the receiving TCP sends an acknowledgment or until enough data have accumulated to fill a maximum-size segment. At this time, the sending TCP can send the segment.
3. Step 2 is repeated for the rest of the transmission. Segment 3 is sent immediately if an acknowledgment is received for segment 2, or if enough data have accumulated to fill a maximum-size segment.

B) Syndrome Created by the Receiver : The receiving TCP may create a silly window syndrome if it is serving an application program that consumes data slowly.

Clark's solution is to send an acknowledgment as soon as the data arrive, but to announce a window size of zero until either there is enough space to accommodate a segment of maximum size or until at least half of the receive buffer is empty.

The second solution is to delay sending the acknowledgment. This means that when a segment arrives, it is not acknowledged immediately. The receiver waits until there is a decent amount of space in its incoming buffer before acknowledging the arrived segments. The delayed acknowledgment prevents the sending TCP from sliding its window. After the sending TCP has sent the data in the window, it stops. This kills the syndrome. It now defines that the acknowledgment should not be delayed by more than 500 ms.

Error Control : Error control in TCP is achieved through the use of three simple tools: checksum, acknowledgment, and time-out.

a) Checksum : If a segment is corrupted, as detected by an invalid checksum, the segment is discarded by the destination TCP and is considered as lost. TCP uses a 16-bit checksum that is mandatory in every segment.

b) ACK segments do not consume sequence numbers and are not acknowledged.

1. Cumulative Acknowledgment (ACK) : TCP was originally designed to acknowledge receipt of segments cumulatively. The receiver advertises the next byte it expects to receive, ignoring all segments received and stored out of order
2. Selective Acknowledgment : A SACK does not replace an ACK, but reports additional information to the sender. A SACK reports a block of bytes that is out of order, and also a block of bytes that is duplicated, i.e., received more than once

c) Retransmission : When the retransmission timer expires or when the sender receives three duplicate ACKs for the first segment in the queue, that segment is retransmitted.

1. Retransmission after RTO : The sending TCP maintains one retransmission time-out (RTO) for each connection. When the timer matures, i.e. times out, TCP resends the segment in the front of the queue (the segment with the smallest sequence number) and restarts the timer.
2. Retransmission after Three Duplicate ACK Segments : This feature is called fast retransmission. In this version, if three duplicate acknowledgments (i.e., an original ACK plus three exactly identical copies) arrive for a segment, the next segment is retransmitted without waiting for the time-out.

Data may arrive out of order and be temporarily stored by the receiving TCP, but TCP guarantees that no out-of-order data are delivered to the process. **TCP can best be modeled as a Selective-Repeat protocol.**

Data Transfer in TCP :

Figure 24.22 *Simplified FSM for the TCP sender side*

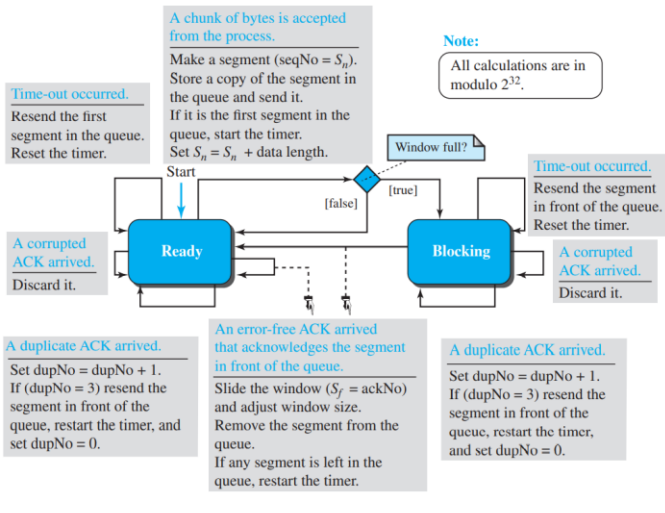
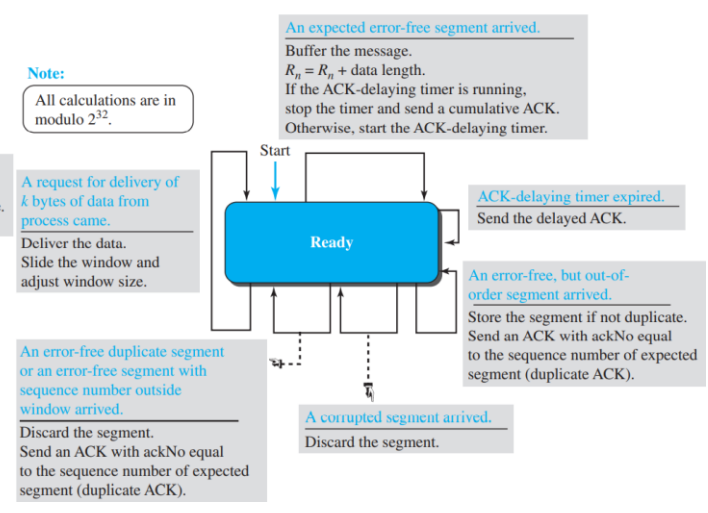


Figure 24.23 *Simplified FSM for the TCP receiver side*



TCP Congestion Control :

a) Congestion Window : To control the number of segments to transmit, TCP uses another variable called a congestion window, cwnd, whose size is controlled by the congestion situation in the network (as we will explain shortly). The cwnd variable and the rwnd variable together define the size of the send window in TCP. The first is related to the congestion in the middle (network); the second is related to the congestion at the end. The actual size of the window is the minimum of these two.

Actual window size = minimum (rwnd, cwnd)

b) Congestion Detection : The TCP sender uses the occurrence of two events as signs of congestion in the network: time-out and receiving three duplicate ACKs.

1. The first is the time-out. If a TCP sender does not receive an ACK for a segment or a group of segments before the time-out occurs, it assumes that the corresponding segment or segments are lost and the loss is due to congestion.
2. Another event is the receiving of three duplicate ACKs (four ACKs with the same acknowledgment number). Recall that when a TCP receiver sends a duplicate ACK, it is the sign that a segment has been delayed, but sending three duplicate ACKs is the sign of a missing segment, which can be due to congestion in the network. However, the congestion in the case of three duplicate ACKs can be less severe than in the case of time-out. When a receiver sends three duplicate ACKs, it means that one segment is missing, but three segments have been received. The network is either slightly congested or has recovered from the congestion.

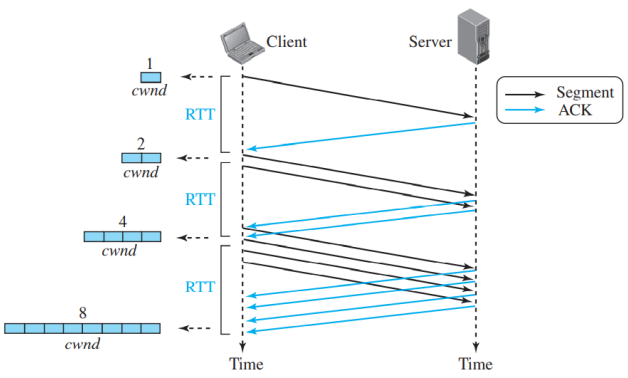
A very interesting point in TCP congestion is that the TCP sender uses only one feedback from the other end to detect congestion: ACKs. The lack of regular, timely receipt of ACKs, which results in a time-out, is the sign of a strong congestion; the receiving of three duplicate ACKs is the sign of a weak congestion in the network.

c) Congestion Policies : TCP's general policy for handling congestion is based on three algorithms: slow start, congestion avoidance, and fast recovery.

1. Slow Start : Exponential Increase The slow-start algorithm is based on the idea that the size of the congestion window (cwnd) starts with one maximum segment size (MSS), but it increases one MSS each time an acknowledgment arrives.

In the slow-start algorithm, the size of the congestion window increases exponentially until it reaches a threshold. In other words, **it will increase by ack sent by server and not by squaring**. Cwnd increase by 1 MSS per Ack and doubles per RTT.

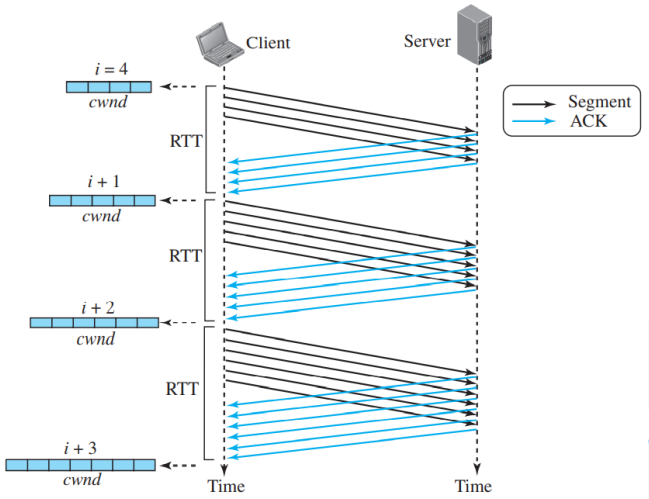
Figure 24.29 *Slow start, exponential increase*



Start	→	$cwnd = 1 \rightarrow 2^0$
After 1 RTT	→	$cwnd = cwnd + 1 = 1 + 1 = 2 \rightarrow 2^1$
After 2 RTT	→	$cwnd = cwnd + 2 = 2 + 2 = 4 \rightarrow 2^2$
After 3 RTT	→	$cwnd = cwnd + 4 = 4 + 4 = 8 \rightarrow 2^3$

2. congestion avoidance : which increases the cwnd additively instead of exponentially.

Figure 24.30 *Congestion avoidance, additive increase*



Start	→	$cwnd = i$
After 1 RTT	→	$cwnd = i + 1$
After 2 RTT	→	$cwnd = i + 2$
After 3 RTT	→	$cwnd = i + 3$

In the congestion-avoidance algorithm, the size of the congestion window increases additively until congestion is detected.

3. Fast Recovery : This algorithm is also an additive increase, but it increases the size of the congestion window when a duplicate ACK arrives (after the three duplicate ACKs that trigger the use of this algorithm).

d) Policy Transition : We discussed three congestion policies in TCP. Taho TCP, Reno TCP, and New Reno TCP.

1. **Taho TCP** : The early TCP, known as Taho TCP, used only two different algorithms in their congestion policy: slow start and congestion avoidance. Carefully see it is not threshold/2.

Figure 24.31 *FSM for Taho TCP*

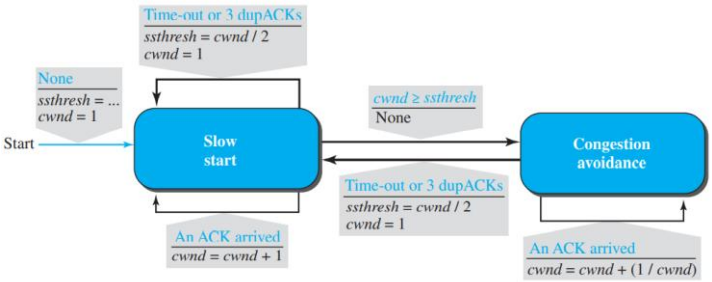
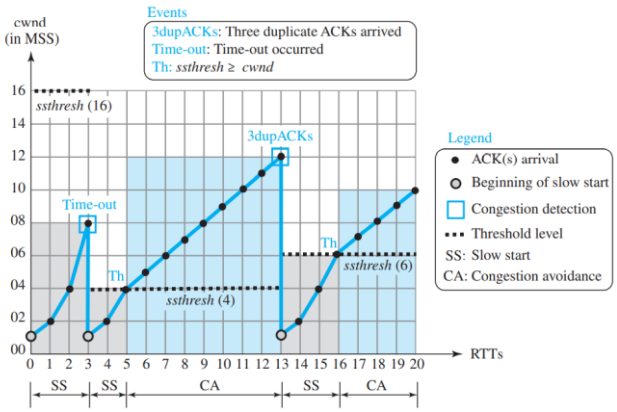


Figure 24.32 *Example 24.9*



2. **Reno TCP** :

Figure 24.33 *FSM for Reno TCP*

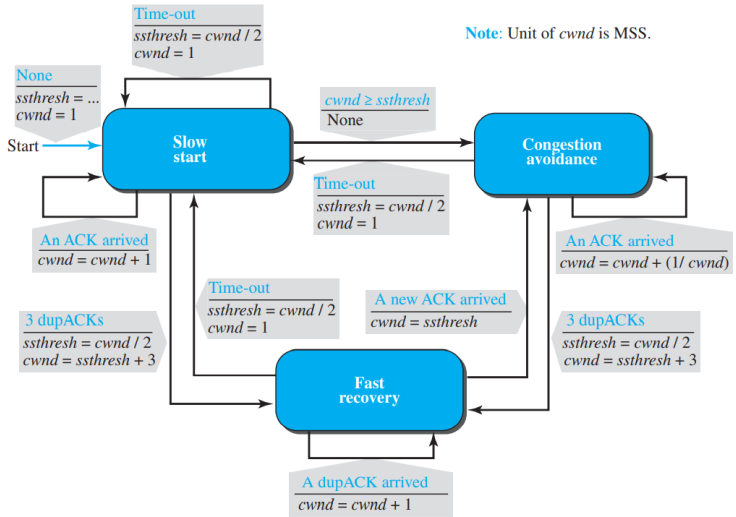
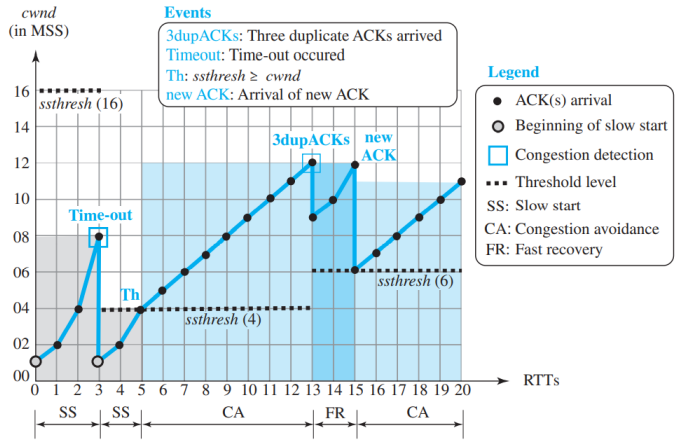
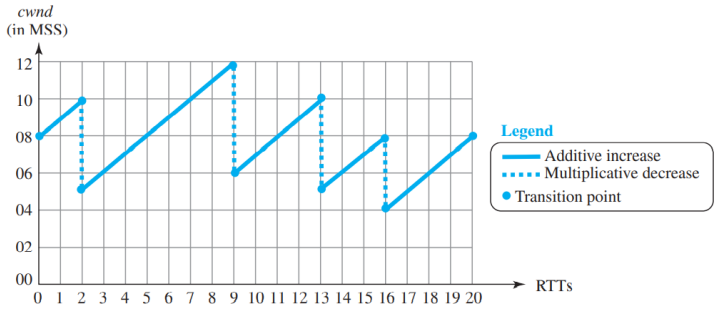


Figure 24.34 *Example 24.10*



congestion window size, after it passes the initial slow-start state, follows a saw tooth pattern called additive increase, multiplicative decrease (AIMD).

Figure 24.35 *Additive increase, multiplicative decrease (AIMD)*



Example 24.11

If MSS = 10 KB (kilobytes) and RTT = 100 ms in Figure 24.35, we can calculate the throughput as shown below.

$W_{max} = (10 + 12 + 10 + 8 + 8) / 5 = 9.6$ MSS

Throughput = $(0.75\ W_{max} / RTT) = 0.75 \times 960$ kbps / 100 ms = 7.2 Mbps

TCP Throughput : throughput = (0.75) Wmax / RTT, Wmax is the average of window sizes when the congestion occurs.

TCP Timers :

1. Retransmission Timer : To retransmit lost segments, TCP employs one retransmission timer (for the whole connection period) that handles the retransmission time-out (RTO), the waiting time for an acknowledgment of a segment.
   a) Round-Trip Time (RTT = 2Tp) :
   - ❑ Measured RTT : how long it takes to send a segment and receive an acknowledgment for it. This is the measured RTT.
   - ❑ Smoothed RTT : weighted average of RTTM and the previous RTTS.

Initially	→	**No value**
After first measurement	→	$\mathbf{RTT_S = RTT_M}$
After each measurement	→	$\mathbf{RTT_S = (1 - \alpha)\ RTT_S + \alpha \times RTT_M}$

The value of α is implementation-dependent, but it is normally set to 1/8. In other words, the new RTT_S is calculated as 7/8 of the old RTT_S and 1/8 of the current RTT_M.

- ❑ ***RTT Deviation.*** Most implementations do not just use RTT_S; they also calculate the RTT deviation, called RTT_D, based on the RTT_S and RTT_M, using the following formulas. (The value of β is also implementation-dependent, but is usually set to 1/4.)

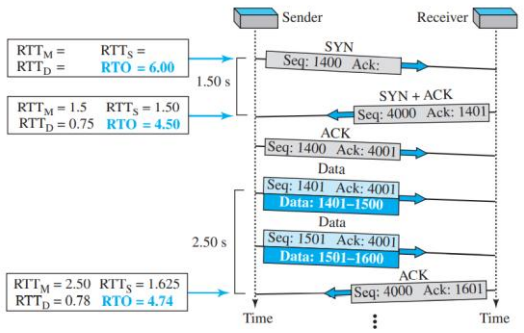
Initially	→	**No value**
After first measurement	→	$\mathbf{RTT_D = RTT_M / 2}$
After each measurement	→	$\mathbf{RTT_D = (1 - \beta)\ RTT_D + \beta \times \lvert RTT_S - RTT_M \rvert}$

2. Retransmission Time-out (RTO) The value of RTO is based on the smoothed roundtrip time and its deviation. Most implementations use the following formula to calculate the RTO:
   Original → Initial value
   After any measurement → RTO = RTTS + 4 × RTTD

Example :

When the first data segment is sent, a new RTT measurement starts. Note that the sender does not start an RTT measurement when it sends the ACK segment, because it does not consume a sequence number and there is no time-out. No RTT measurement starts for the second data segment because a measurement is already in progress. The arrival of the last ACK segment is used to calculate the next value of RTTM. Although the last ACK segment acknowledges both data segments (cumulative), its arrival finalizes the value of RTTM for the first segment. The values of these variables are now as shown below.

Figure 24.36 *Example 24.12*



$\mathbf{RTT_M = 2.5}$

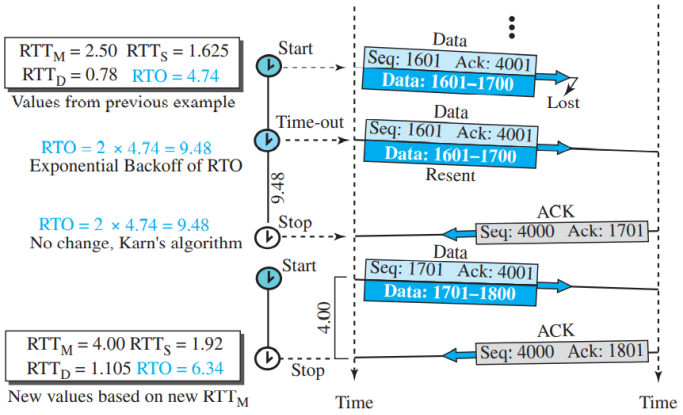
$\mathbf{RTT_S = (7/8) \times (1.5) + (1/8) \times (2.5) = 1.625}$

$\mathbf{RTT_D = (3/4) \times (0.75) + (1/4) \times |1.625 - 2.5| = 0.78}$

$\mathbf{RTO = 1.625 + 4 \times (0.78) = 4.74}$

- Karn's algorithm is simple. Do not consider the round-trip time of a retransmitted segment in the calculation of RTTs. Do not update the value of RTTs until you send a segment and receive an acknowledgment without the need for retransmission.
- Exponential Backoff : . The value of RTO is doubled for each retransmission. So, if the segment is retransmitted once, the value is two times the RTO. If it is retransmitted twice, the value is four times the RTO, and so on.

Figure 24.37 *Example 24.13*



The first segment in the figure is sent, but lost. The RTO timer expires after 4.74 seconds. The segment is retransmitted and the timer is set to 9.48, twice the previous value of RTO. This time an ACK is received before the time-out. We wait until we send a new segment and receive the ACK for it before recalculating the RTO (Karn's algorithm).

- **TIME-WAIT Timer** : The TIME-WAIT (2MSL) timer is used during connection termination. The maximum segment lifetime (MSL) is the amount of time any segment can exist in a network before being discarded. When client sends FIN segment and it will get received by receiver by sending ACK packet and before this process it has sent some data but because of delay it received after FIN segment and ACK segment at the receiver side. So now receiver know that the sequence number of this segment is lower than last segment so it will except but if it is higher than last sequence number, it will consider as some malicious data and it will reject. But consider the case where before sending ACK for delayed segment the receiver has sent FIN segment so now connection is closed but still sender wait for some time called **time-wait time.** So that ack of some previous segment can still be received at client. But it will discard segment of ack higher than last sent segment. This will protect it from malicious data.
- Deadlock and persistence timer : If the receiving TCP announce a window size of zero, the sending TCP stops transmitting segments until the receiving TCP sends an ACK segment announcing a nonzero window size. This ACK segment can be lost. ACK segments are not acknowledged nor retransmitted in TCP. If this acknowledgment is lost, the receiving TCP thinks that it has done its job and waits for the sending TCP to send more segments. There is no retransmission timer for a segment containing only an acknowledgment. The sending TCP has not received an acknowledgment and waits for the other TCP to send an acknowledgment advertising the size of the window. Both TCP's might continue to wait for each other forever (a deadlock). To remove deadlock, we use persistence timer.
- Keepalive timer : Suppose that a client opens a TCP connection to a server, transfers some data, and becomes silent. Perhaps the client has crashed. In this case, the connection remains open forever. The time-out is usually 2 hours. If the server does not hear from the client after

2 hours, it sends a probe segment. If there is no response after 10 probes, each of which is 75 seconds apart, it assumes that the client is down and terminates the connection.

Example :
Assume the TCP round trip time RTT is currently 30 ms and the following ACK's come in after 26, 32 and 24 ms respectively. What is the new RTT estimate (in ms) using basic algorithm ($\alpha = 0.9$)?
Answer : RTTM = 26, RTTS = 30, $\alpha = 0.9$
RTTS = RTTS(α)+RTTM(1- α)
I know formula contradict but Always remember small part of ack RTT i.e. RTTM and big part of current RTT.

Note :

- Maintaining connection semantics between two directly connected nodes is done by network layer.
- Recovering lost packets between two directly connected nodes is done by transport layer.
- Recovering lost packets between two nodes separated by multiple hops is done by data link control.
- Arbitration is done between multiple nodes attached to a single medium to resolve conflicts.
- When correctness of data is given more importance, we use TCP and when time is more important than correctness of data UDP is preferred.
- There will be no stalling if time to sent entire window >= time for first ack to arrive back. i.e. W*S/B>=RTT. Where S is segment size and other are usual.

Questions on Transport Layer :

1) Which of the following functionality must be implemented by a transport protocol over and above the network protocol?

A. Recovery from packet losses
B. Detection of duplicate packets
C. Packet delivery in the correct order
D. End to end connectivity

Answer : Recovery from packet losses is not by network layer. Only end to end connectivity is important in term of complete data transmission and that is why transport protocol is important. Because data is sent from network A to B but if data is not sent to correct process, service is incomplete.

2) Match the following :

Field	Length in bits
P. UDP Header's Port Number	I. 48
Q. Ethernet MAC Address	II. 8
R. IPv6 Next Header	III. 32
S. TCP Header's Sequence Number	IV. 16

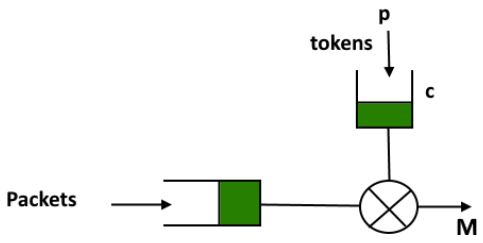
Answer : P-IV, Q-I, R-II, S-III

3) For a host machine that uses the token bucket algorithm for congestion control, the token bucket has a capacity of 1 megabyte and the maximum output rate is 20 megabytes per second. Tokens arrive at a rate to sustain output at a rate of 10 megabytes per second. The token bucket is currently full and the machine needs to send 12 megabytes of data. The minimum time required to transmit the data is _____ seconds.

Answer : Token bucket is a congestion control algorithm for data transfer. It takes tokens to synchronize between the rate of incoming and outgoing data.
According to the token bucket algorithm, the minimum time required to send 1 MB of data or the maximum rate of data transmission is
given by:

$S = C / (M - P)$

Where,
M = Maximum burst rate,
P = Rate of arrival of a token,
C = capacity of the bucket



Using the above formula for the given question we can say that:
M = 20 MB
P = 10 MB
C = 1 MB
S = 1 / (20- 10) = 0.1 sec
Since, the bucket is initially full, it already has 1 MB to transmit so it will be transmitted instantly. So, we are left with only (12 - 1), i.e. 11 MB of data to be transmitted. Time required to send the 11 MB will be 11 * 0.1 = 1.1 sec

4) The maximum payload of a TCP segment is:

65,535
65,515
65,495
65,475

Answer : it says payload so you need to subtract 40(20+20) headers from it. 20 headers from IP and 20 headers from TCP.

5) Assume that you have made a request for a web page through your web browser to a web server. Initially the browser cache is empty. Further, the browser is configured to send HTTP requests in non-persistent mode. The web page contains text and five very small images. The minimum number of TCP connections required to display the web page completely in your browser is _____ . Note - This question was Numerical Type

Answer : Total 6 entities are there so; we need 6 connections. Here if persistent is given then the answer would be 1. Persistent = for all files one connection. And non-persistent = for all files separate connection. Note if RTT is to find out then first it will take compulsory 2RTT of time for TCP connection and header file of html and then 1RTT for files (text and images).

6) Which of the following statements are true ? (a) Three broad categories of Networks are (i) Circuit Switched Networks (ii) Packet Switched Networks (iii) Message Switched Networks (b) Circuit Switched Network resources need not be reserved during the setup phase. (c) In packet switching there is no resource allocation for packets. Code :

(a) and (b) only
(a) and (c) only
(b) and (c) only
(a), (b) and (c)

Answer : In packet switching there is no resource allocation for packets"--> This is also false.
Packet switching is of 2 types--> 1. Datagram
2. Virtual Circuit Switching
Virtual circuit is connection-oriented simply meaning that there is a reservation of resources like buffers, CPU, bandwidth, etc. for the time in which the newly setup VC is going to be used by a data transfer session. So, NONE of the options given are correct, but if we consider only datagram (which is not connection oriented) then we can go with option C.

7) Consider an instance of TCP's Additive Increase Multiplicative Decrease (AIMD) algorithm where the window size at the start of the slow start phase is 2 MSS and the threshold at the start of the first transmission is 8 MSS. Assume that a timeout occurs during the fifth transmission. Find the congestion window size at the end of the tenth transmission.

A. 8 MSS
C. 7 MSS
B. 14 MSS
D. 10 MSS

Answer : 2→4→8(threshold)→9→10(fifth transmission so we do AIMD and new threshold becomes 10/2 = 5 and we start from 2 MSS)→4→5→6→7→8(this is tenth transmission) which is required in question). 8 is answer.

8) Which of the following statements are TRUE?
S1: TCP handles both congestion and flow control
S2: UDP handles congestion but not flow control
S3: Fast retransmit deals with congestion but not flow control
S4: Slow start mechanism deals with both congestion and flow control

Answer :
(S1) TCP handles both congestion and flow control True. It uses congestion window for congestion control & Advertisement window for flow control

(S2) UDP handles congestion but not flow control UDP does not handle congestion but also not handle flow control.
(S3) Fast retransmit deals with congestion but not flow control Yes. Fast Retransmit is technique for detecting out of Order Datagram & Sending it. It is congestion control technique and has no relation with Flow control
(S4) Slow start mechanism deals with both congestion and flow control False. It has nothing to do with Flow control. Flow control is taken care by Advertisement window. Slow start is way Sender tries to gauge network capacity !

9) Assume that the bandwidth for a connection is 1048560 bits/sec. Let A be the value of RTT in milliseconds (rounded off to the nearest integer) after which the window scale option is needed. Let B be the maximum possible window size with window scale option. Then the values of A and B are

Answer : In TCP when the bandwidth delay product increases beyond 64K receiver window scaling is needed. The bandwidth delay product is the maximum amount of data on the network circuit at any time and is measured as RTT * Bandwidth. This is not the time for sending data rather just the time for sending data without acknowledgement. So, here, we have bandwidth delay product = (1048560/8) Byte * α = 64K, α = (64 K * 8) / 1048560 = 0.5s = 500milliseconds. When window scaling happens, a 14bit shift count is used in header. So, the maximum possible window size gets increased from 2^{16}- 1 to (2^{16} - 1) * 2^{14}.

10) Consider a long-lived session with an end-to-end bandwidth of 1Gbps (10^{-9} bits-per-second). The session starts with a sequence number of 1234. The minimum time (in seconds, rounded to the closet integer) before this sequence number can be used again is

Answer : This question asks for warp around time. So total 32 bits will generate 2^{32} byte of number we divide this by time to get time as 34.35

11) Consider a TCP connection with following specifications
Maximum transfer rate: 10 Mbps
Maximum segment size: 2000 bytes.
Round trip propagation delay: 160 msec
What is the maximum window size (in MSS) that this TCP connection can achieve?

Answer :

Let W denote the max window size measured in segments. Then, W * MSS/RTT = 10Mbps, as packets will be dropped if the maximum sending rate exceeds link capacity. Thus, we have
W * 2000 * 8/160 msec = 10Mbps, then W is 100 segments.

Correct answer is 100

Questions on Mics of computer network :

1) A firewall is to be configured to allow hosts in a private network to freely open TCP connections and send packets on open connections. However, it will only allow external hosts to send packets on existing open TCP connections or connections that are being opened (by internal hosts) but not allow them to open TCP connections to hosts in the private network. To achieve this the minimum capability of the firewall should be that of

A combinational circuit *A pushdown automaton with one stack*
A finite automaton *A pushdown automaton with two stacks*

Answer : A combinational circuit⇒ Not possible, because we need memory in Firewall, Combinational ckt has none.

A finite automaton⇒ We need infinite memory, there is no upper limit on Number of TCP ckt so Not this.

A pushdown automaton with one stack⇒ Stack is infinite. Suppose we have 2 connections, we have pushed details of those on stack we cannot access the details of connection which was pushed first, without popping it off. So, Big NO.

A pushdown automaton with two stacks⇒ This is TM. It can do everything our normal computer can do so Yes. A firewall can be created out of TM.

2) How many bytes of data can be sent in 15 seconds over a serial link with baud rate of 9600 in asynchronous mode with odd parity and two stop bits in the frame?

10,000 bytes *15,000 bytes*
12,000 bytes *27,000 bytes*

Answer : Baud rate = Number of symbols sent per unit time.

When by "symbol" we mean a "bit", then we say Bit rate = Number of bits sent per unit time.

Given that it is aynchronous mode of transmission, then along with per byte, you have to send some extra bit like start, stop bit and parity bits, etc (start and stop bit are compulsory).

1 bit for start bit, 8 bits for data, 1 bit for parity, 2 bits for stop bits.

(9600 * 15)/12 = 12000 bytes.

3) What will be the total minimum bandwidth of the channel required for 7 channels of 400 kHz bandwidth multiplexed together with each guard band of 20 kHz?

2800 khz *3600 khz*
2600 khz *2920 khz*

Answer : For seven channels, we need at least six guard bands.

the required bandwidth is at least $7 \times 400 + 6 \times 20 = 2920$ khz

APPLICATION LAYER

If you got time then read whole application layer

1) DNS (Domain name system)

The client/server programs can be divided into two categories : those that can be directly used by the user, such as email, and those that support other application programs. The domain name system (DNS) is a supporting program that is used by other programs such as e-mail.

Note : DNS is commonly used by other application layer protocols including HTTP, SMTP and FTP to translate user-supplied hostnames to IP addresses. DNS uses UDP for name and queries. DNS primarily uses the UDP on port number 53 to serve request. We can configure multiple DNS servers for a client and they will be used in sequential order upon resolution failure.

A user of an e-mail program may know the email address of the recipient; however, the IP protocol needs the IP address.

1. The browser extracts the hostname, www.someschool.edu, from the URL and passes the hostname to the client side of the DNS application.
2. The DNS client sends a query containing the hostname to a DNS server.
3. The DNS client eventually receives a reply, which includes the IP address for the hostname.
4. Once the browser receives the IP address from DNS, it can initiate a TCP connection to the HTTP server process located at port 80 at that IP address.

2) Hyper Text Transfer Protocol (HTTP) :

HTTP is implemented in two programs: a client program and a server program.

HTTP is said to be a stateless protocol

Stateless protocol is a communications protocol in which no information is retained by either sender or receiver.(Note: The remembered information is called as the state)

Non-Persistent and Persistent Connections: In many Internet applications, the client and server Communicate for an extended period of time, with the client making a series of requests and the Server responding to each of the requests. Depending on the application and on how the application is being used, the series of requests may be made back-to-back, periodically at regular intervals. When this client-server interaction is taking place over TCP, the application developer needs to make an important decision - should each request/response pair be sent over a separate TCP connection known as non-persistent connections or should all of the requests and their corresponding responses be sent over the same TCP connection known as persistent connections.

Note: The default mode of HTTP uses persistent connections with pipelining.

2.1) URL : Uniform Resource Locator, This URL consists of three parts: the protocol (http), the DNS name of the host (www.cs.washington.edu), and the path name (index.html). When a user clicks on a hyperlink, the browser carries out a series of steps:

a) The browser determines the URL (by seeing what was selected).
b) Before DNS request computer will check if IP is present in cache if it is not there then step c
c) The browser asks DNS for the IP address of the server www.cs.washington.edu.
d) DNS replies with 128.208.3.88
e) The browser makes a TCP connection to 128.208.3.88 on port 80, the well-known port for the HTTP protocol.

f) It sends over an HTTP request asking for the page /index.html

g) The www.cs.washington.edu server sends the page as an HTTP response, for example, by sending the file /index.html.

h) If the page includes URLs that are needed for display, the browser fetches the other URLs using the same process. In this case, the URLs include multiple embedded images also fetched from www.cs.washington.edu, an embedded video from www.youtube.com, and a script from www.google-analytics.com

i) The browser displays the page /index.html

j) The TCP connections are released if there are no other requests to the same servers for a short period.

2.2) Cookies : The World Wide Web was originally designed as a stateless entity. A client sends a request; a server Responds. Their relationship is over. Today the Web has other functions; some are listed here.

a) Some websites need to allow access to registered clients only.

b) Websites are being used as electronic stores that allow users to browse through the store.

c) Some websites are used as portals: the user selects the Web pages he wants to see. Eg: Yahoo etc.

d) Some websites are just advertising

For these purposes, the cookie mechanism was devised.

1. When a server receives a request from a client, it stores information about the client in a file or a string.
2. The server includes the cookie in the response that it sends to the client.
3. When the client receives the response, the browser stores the cookie in the cookie directory, which is sorted by the domain server name. Cookies enables a web server to link subsequent HTTP requests to the same web browser.

Domain	Path	Content	Expires	Secure
toms-casino.com	/	CustomerID=297793521	15-10-10 17:00	Yes
jills-store.com	/	Cart=1-00501;1-07031;2-13721	11-1-11 14:22	No
aportal.com	/	Prefs=Stk:CSCO+ORCL;Spt:Jets	31-12-20 23:59	No
sneaky.com	/	UserID=4627239101	31-12-19 23:59	No

Figure 7-22. Some examples of cookies.

Cookies are just strings, not executable programs.

Note:

A cookie could contain a virus, but since cookies are treated as data, there is no official way for the virus to actually run and do damage.

If Expires field is absent, the browser discards the cookie when it exits. Such a cookie is called a non-persistent cookie. If a time and date are supplied, the cookie is said to be a persistent cookie and is kept until it expires.

2.3) HTTP : The Hypertext Transfer Protocol (HTTP) is a request-response protocol used mainly to access data on the World Wide Web.

HTTP functions as a combination of FTP and SMTP. It is similar to FTP because it transfers files and uses the services of TCP, but it uses only one connection. There is no separate control connection; only data are transferred between the client and the server. HTTP is like SMTP because

the data transferred between the client and the server look like SMTP Messages. In addition, the format of the messages is controlled by MIME-like headers. Unlike SMTP, the HTTP Messages (ASCII) are not destined to be read by humans; they are read and interpreted by the HTTP server and HTTP client (browser).
SMTP messages are stored and forwarded, but HTTP messages are delivered immediately.
HTTP uses the services of TCP on well-known port 80. HTTP itself is a stateless protocol.

Table 27.1 *Methods*

Method	*Action*
GET	Requests a document from the server
HEAD	Requests information about a document but not the document itself
POST	Sends some information from the client to the server
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request
CONNECT	Reserved
OPTION	Inquires about available options

Connect: Connect through a proxy
Delete: Remove the Web page
Note : HTTP uses ASCII characters. A client can directly connect to a server using TELNET, which logs into port 80.

Persistent Versus Non-Persistent Connection:
HTTP 1.0 specified a non-persistent connection, while a persistent connection is the default in version

1. **Non-persistent Connection**: In a non-persistent connection, one TCP connection is made for each request/response. a. The client opens a TCP connection and sends a request.
   b. The server sends the response and closes the connection.
   c. The client reads the data until it encounters an end-of-file marker; it then closes the connection. For N different pictures in different files, the connection must be opened and closed N times and the server needs N different buffers.
2. **Persistent Connection**: In a persistent connection, the server leaves the connection open for more requests after sending a response. The server can close the connection at the request of a client or if a time-out has been reached. It is also possible to pipeline requests.

2.4) Proxy Server : HTTP supports proxy servers. A proxy server is a computer that keeps copies of responses to recent requests. The HTTP client sends a request to the proxy server. The proxy server checks its cache. If the response is not stored in the cache, the proxy server sends the request to the corresponding server. Incoming responses are sent to the proxy server and stored for future requests from other clients.
The proxy server reduces the load on the original server, decreases traffic, and improves latency. However, to use the proxy server, the client must be configured to access the proxy instead of the target server.
Network latency is an expression of how much time it takes for a packet of data to get from one designated point to another.

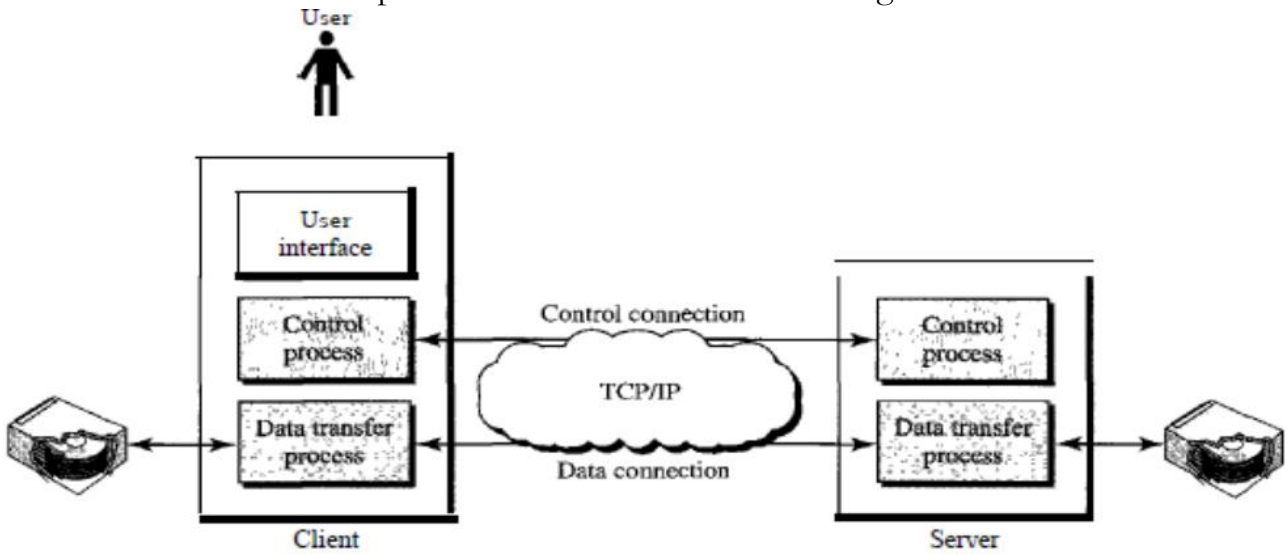
3) FTP (Files Transfer Protocol) :

File Transfer Protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another.

FTP differs from other client/server applications in that it establishes two connections between the hosts. One connection is used for data transfer, the other for control information (commands and responses). Separation of commands and data transfer makes FTP more efficient.

The control connection remains connected during the entire interactive FTP session. The data connection is opened and then closed for each file transferred. The data connection, on the other hand, needs more complex rules than control line due to the variety of data types transferred.

The client has three components: user interface, client control process, and the client data transfer process. The server has two components: the server control process and the server data transfer process. The control connection is made between the control processes. The data connection is made between the data transfer processes as shown below in the figure:



NOTE : FTP uses the same approach as SMTP to communicate across the control connection. FTP uses port 20 for data transfer and port 21 for control signal.

File transfer in FTP means one of three things (commands):

- RETR A file is to be copied from the server to the client
- STOR file is to be copied from the client to the server
- LIST A list of directory or file names is to be sent from the server to the client

In FTP, before downloading the file, authentication is required and that will be done using control commands like USER, PASS.

The client must define the type of file to be transferred, the structure of the data, and the transmission mode. In passive mode of FTP, the client initiates path data and control connections, while in active mode client initiates the control connection and then the saver initiates the data connection.

- HTTP sends request and response header lines into the same TCP connection that carries the transferred file itself (In-band).
- FTP uses two parallel TCP connections to transfer a file, a control connection and a data connection (Out-band). FTP uses Telnet protocol for Control info on a TCP connection and another TCP connection for data exchange.

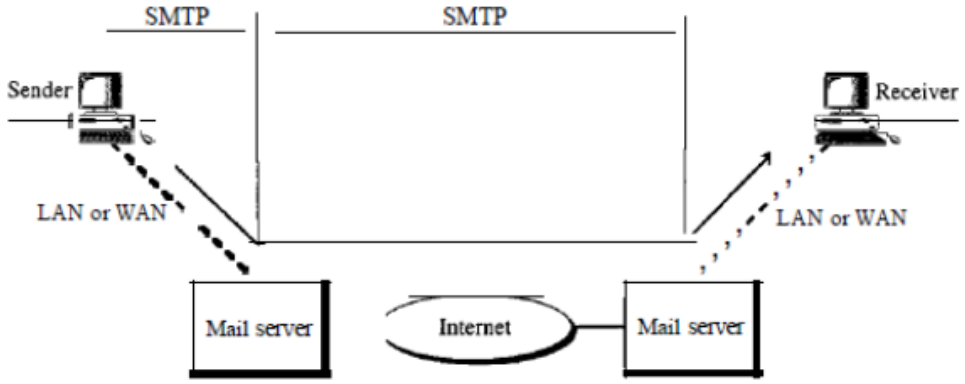
3.1) TELNET : It is the standard TCP/IP protocol for virtual terminal service as proposed by ISO. TELNET Enables the establishment of a connection to a remote system in such a way that the local

terminal appears to be a terminal at the remote system. TELNET is a general-purpose client/server application program.

4) Simple Mail Transfer Protocol (SMTP) :

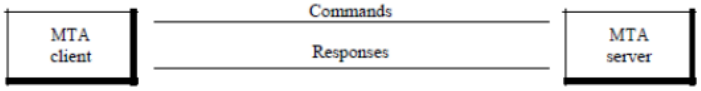
The actual mail transfer is done through message transfer Agents (MTA). To send mail, a system must have the client MTA, and to receive mail, a system must have a server MTA.

Figure 26.16 *SMTP range*



SMTP is used two times, between the sender and the sender's mail server and between the two mail servers. Email can travel through single SMTP server also. SMTP uses commands and responses to transfer messages between an MTA client and an MTA Server.

26.17 *Commands and responses*



Command	Required	Description
HELO	*	Identifies server
MAIL	*	Initiates the mail transaction
RCPT	*	Identifies mail recipient
DATA	*	Initiates mail data transfer
RSET	*	Aborts the current mail transaction
NOOP	*	Receiver returns OK. Used to test the server connection
QUIT	*	Close the connection
VRFY	*	Verify recipient exists
SEND		Delivers message to one or more terminals
SOML		Delivers message to one or more terminals or mailboxes
SAML		Delivers message to one or more terminals and mailboxes
EXPN		Expand mailing list addresses
HELP		Requests Help info from receiver
TURN		Asks receiver to take role as server

Note: We use TELNET to log into port 25 (the well-known port for SMTP).
If you are using web based email, you browser would use SMTP to send email messages to a mail server.

HTTP V/S SMTP

HTTP is mainly a pull protocol—someone loads information on a Web server and users use HTTP to pull the information from the server at their convenience.
SMTP is primarily a push protocol—the sending mail server pushes the file to the receiving mail server.

5) POP3 :

Post Office Protocol, version 3 (POP3) is simple and limited in functionality. The client POP3 software is installed on the recipient computer; the server POP3 software is installed on the mail server.

POP3 begins when the user agent (the client) opens a TCP connection to the mail server (the server) on port 110.

A user agent using POP3 can often be configured (by the user) to "download and delete" from the mailbox or to "download and keep" at the mailbox.

POP3 progresses through three phases: authorization, transaction, and update. During transaction phase during this phase, the user agent can mark messages for deletion, remove deletion marks, and obtain mail statistics. POP3 protocol will allow to download mails from single device.

6) **IMAP4** :

IMAP4 is similar to POP3, but it has more features; IMAP4 is more powerful and more Complex. POP3 does not allow the user to organize her mail on the server; the user cannot have different folders on the server. In addition, POP3 does not allow the user to partially check the contents of the mail before downloading. IMAP4 provides the following extra functions:

1. A user can check the e-mail header prior to downloading.
2. A user can search the contents of the e-mail for a specific string of characters prior to downloading.
3. A user can partially download e-mail. This is especially useful if bandwidth is limited and the e-mail contains multimedia with high bandwidth requirements.
4. A user can create, delete, or rename mailboxes on the mail server.
5. A user can create a hierarchy of mailboxes in a folder for e-mail storage.

7) **SIP (Session initiation protocol)** : Session initiation protocol establishes, manages and terminates multimedia sessions.

Questions on application layer and communication :

1) Provide the best matching between the entries in the two columns given in the table below :

I.	Proxy Server	a.	Firewall
II.	Kazaa, DC++	b.	Caching
III.	Slip	c.	P2P
IV.	DNS	d.	PPP

Answer : i. Proxy Server Proxy Server and Firewall can be combined. a. Firewall
ii. Kazaa, DC++ These are P2P application. c. P2P
iii. Slip P2P Slip is a predecessor of PPP. d. PPP
iv. DNS DNS responses are often called b. Caching

2) Let us consider a statistical time division multiplexing of packets. The number of sources is 10. In a time unit, a source transmits a packet of 1000 bits. The number of sources sending data for the first 20-time units is 6, 9, 3, 7, 2, 2, 2, 3, 4, 6, 1, 10, 7, 5, 8, 3, 6, 2, 9, 5 respectively. The output capacity of multiplexer is 5000 bits per time unit. Then the average number of backlogged of packets per time unit during the given period is

Answer : The capacity of multiplexer is 5000 bits per time unit. This means there are 5 packets per unit time since each source transmits a packet of 1000 bits in a unit time. If the no. of packets

transmitted is larger than 5 then the extra packets are backlogged. This means gets added to the next number and further backlog is calculated.
First is 6 in sequence and we have only 5 packets to transmit so 1 packet will be backlogged and it will get added to next number in sequence so now second number is 9+1=10. Similarly following procedure, we get following backlog.
Total number of backlogged = 1 + 5 + 3 + 5 + 2 + 0 + 0 + 0 + 0 + 1 + 0 + 5 + 7 + 7 + 10 + 8 + 9 + 6 + 10 + 10
Avg backlog per time unit = 89/time unit = 89/20 = 4.45 is the answer.

3)

A user wants to retrieve 8 images from a Web page. The user clicks a base HTML file to get the base HTML files for 8 images. Suppose that the size of base HTML page and all images are 2 KB. Assume that the data rate is 2 Mbps. The round-trip time (RTT) for sending control or HTTP requests is 100 milliseconds. What is the overall retrieve time (in sec) in the case of non-persistent HTTP with no parallel TCP connection? Assume that 1K = 10^3 and 1M = 10^6. _____ (Rounded off to three decimal places)

Answer: 1.872

Solution:

In non persistent case, total 2RTT + file transmission time is required for getting one object (1 RTT for TCP connection, 1 RTT for request/reply and transmit time).

Time for getting one HTML file and 8 objects without parallel connection is that 2 RTT + time to get raw page (HTML page) + 2 RTT + time to get first object + 2 RTT + time to get second object + ...+ 2 RTT + time to get eighth object. Total time (non-persistent without parallel connection)

= 9 * (2 RTT + time to get one object) = 9 * (2 * 100 + 8) = 1872 milliseconds