Module 2
Circuit and Packet Switching

NETWORK AND COMMUNICATION

Mr. A. Swaminathan VIT Chennai

Theory Class 7

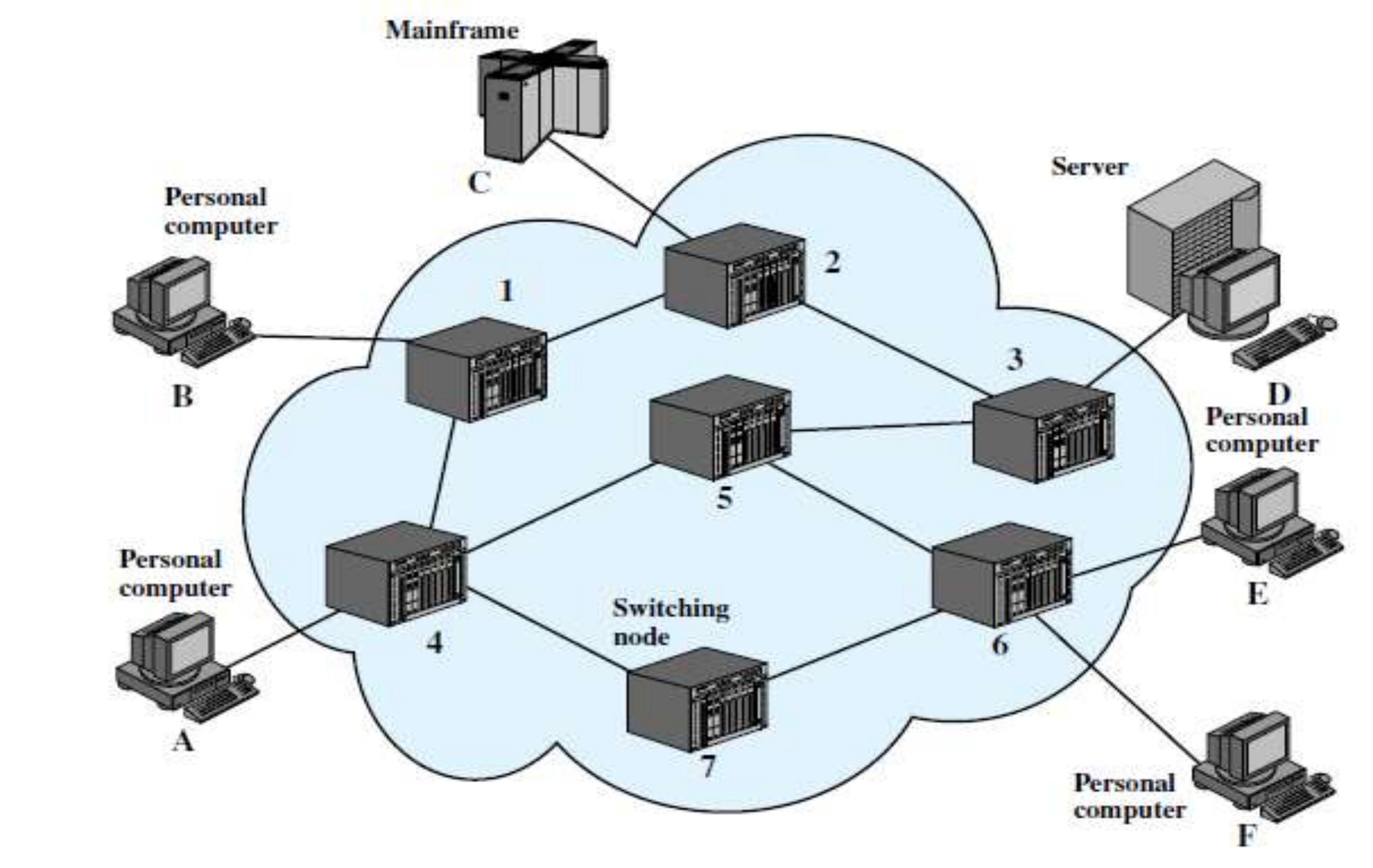
Overview

- Switch
- Types of switching
- Switching Facility
- Station
- Nodes
- Disadvantages and Advantages
- Applications

Switching

- Series of inter linked nodes
- Device that connect multiple communication lines together for effective communication
- To make point to point connection (mesh)
- It finds a route to the destination
- How to allocate bandwidth

Switching



Source: Data Communications and Networking – Behrouz A. Forouzan

Types of switching

1. Circuit Switching
2. Packet Switching

Circuit Switching

- Used in public telephone networks
- Voice communication
- Dedicated path two stations using one or more switches
- It consists of 3 phases
  - Connection Establishment (link to link)
  - Data Transfer
  - Disconnect the Communication

Properties of circuit switching

- Switch close
- Switch open
- Physical layer
- Developed for voice but applicable for data
- In efficiency
- Delay
  - Long initial delay (to find the destination route)
  - Low data delay

Circuit Switching - Disadvantages

- No packets
- Purpose of design - Voice
- Dedicated resource
- Ideal
- Fixed data rate

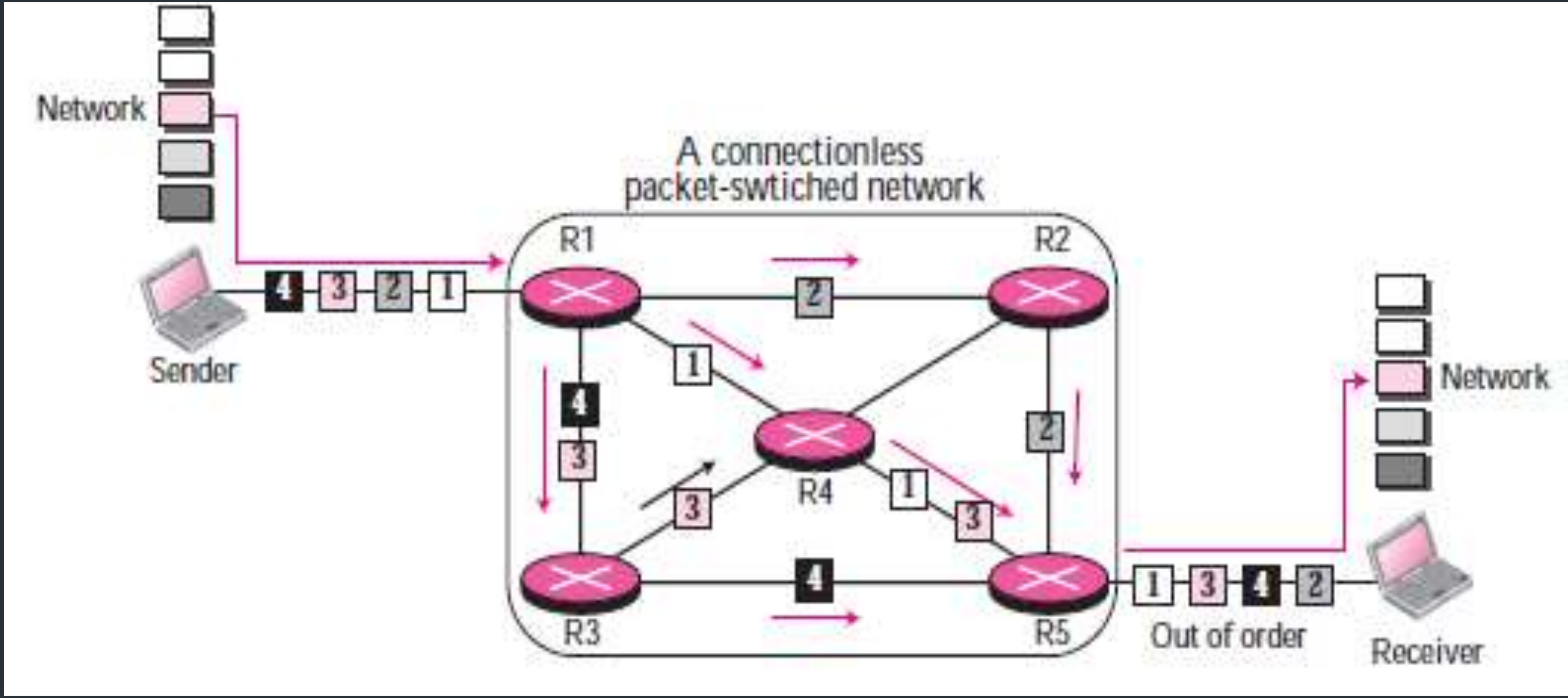
Packet Switching

- Short packets are transferred from S->D
- Ex. 100 bytes (data) – 10 x 10 (Packets)
- Packets consists
  - Control information – Routing address/source/destination
  - User data
- Approaches
  - Connection less service
  - Connection oriented service

Packet Switching Advantages

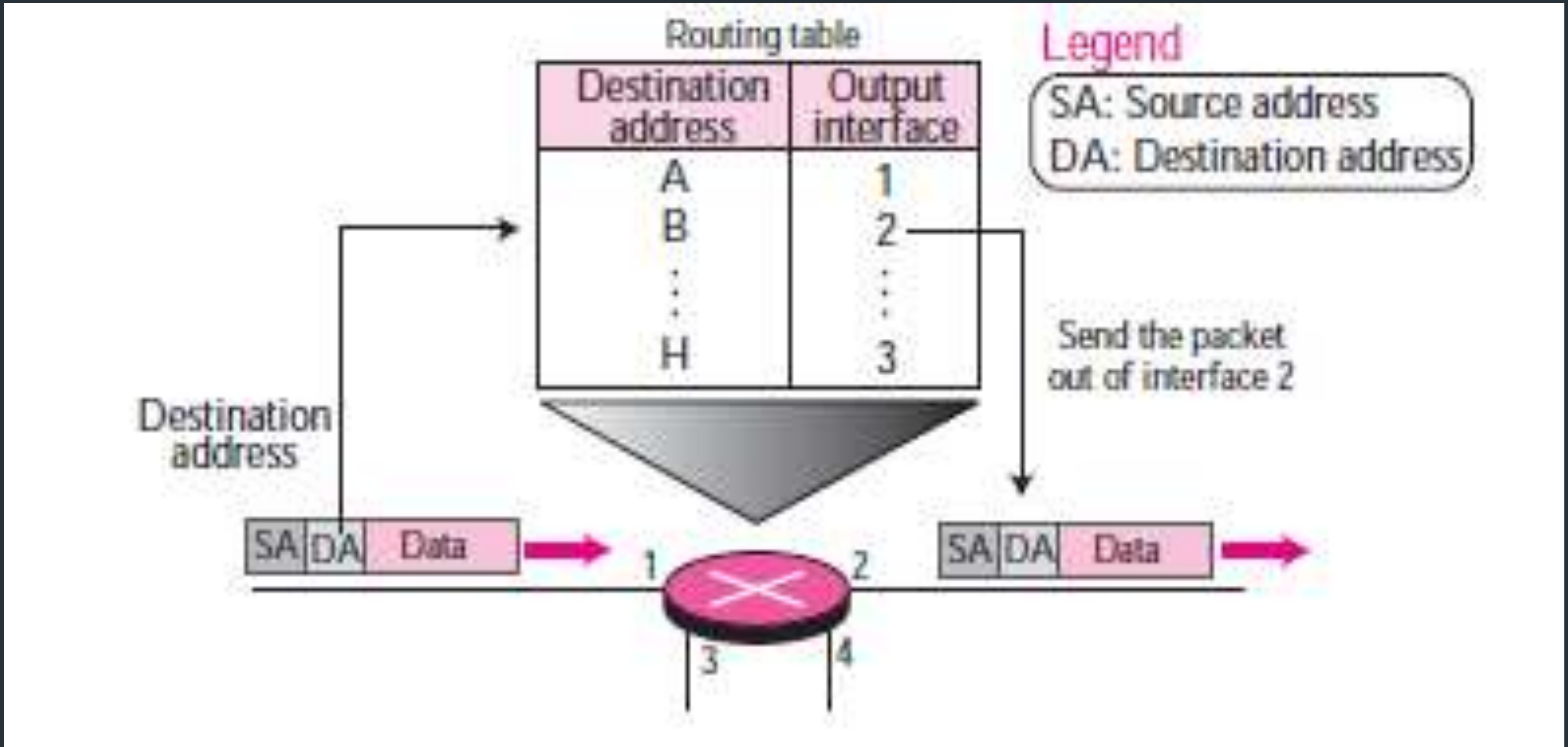
- Station
- Packets: breaks messages into packets
- Sequentially send the packets
- One by one
- Routing: Stream of packets (message) S->D
- Network layer
- Store and forward
- Types
  - Datagram
  - Virtual circuit approach

Connection less Packet Switching



Source: Data Communications and Networking – Behrouz A. Forouzan

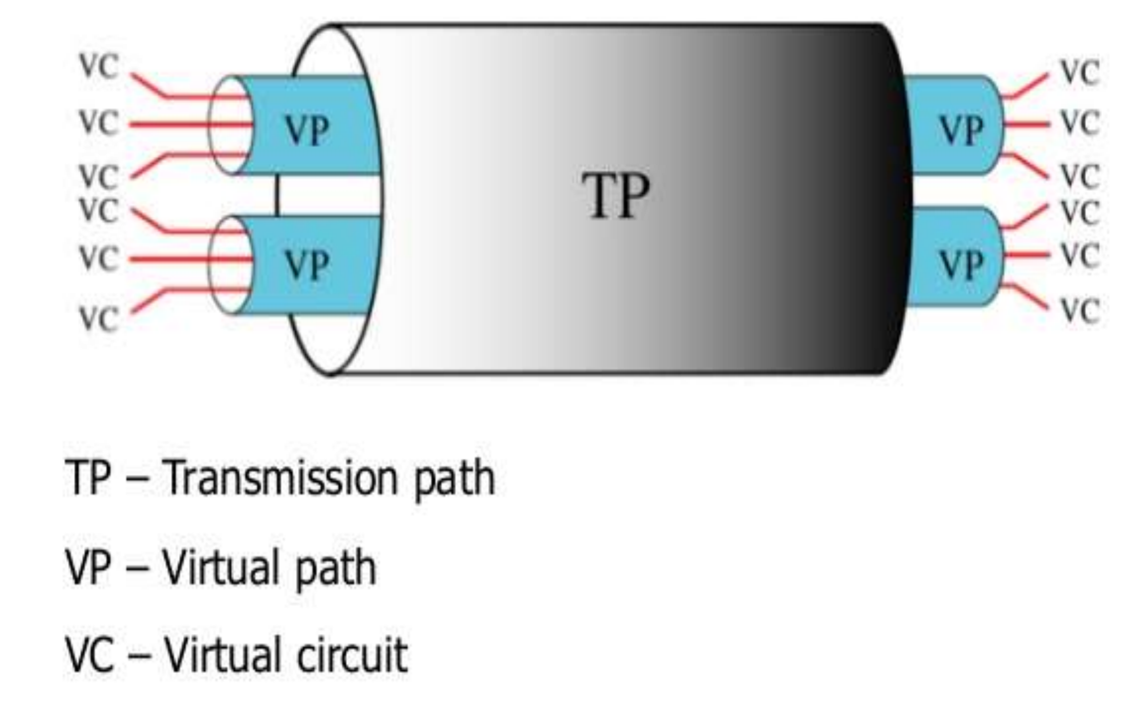
Forwarding Process in a Router (Connectionless Service)



Source: Data Communications and Networking – Behrouz A. Forouzan

Connection oriented Packet switching

- Virtual Connection
- Destination Address
- Flow label, Virtual Path Identifier
- Forwarding decision based on packet label
- Three-phase process
- Setup
- Data Transfer
- Teardown



Source: Data Communications and Networking – Behrouz A. Forouzan

References

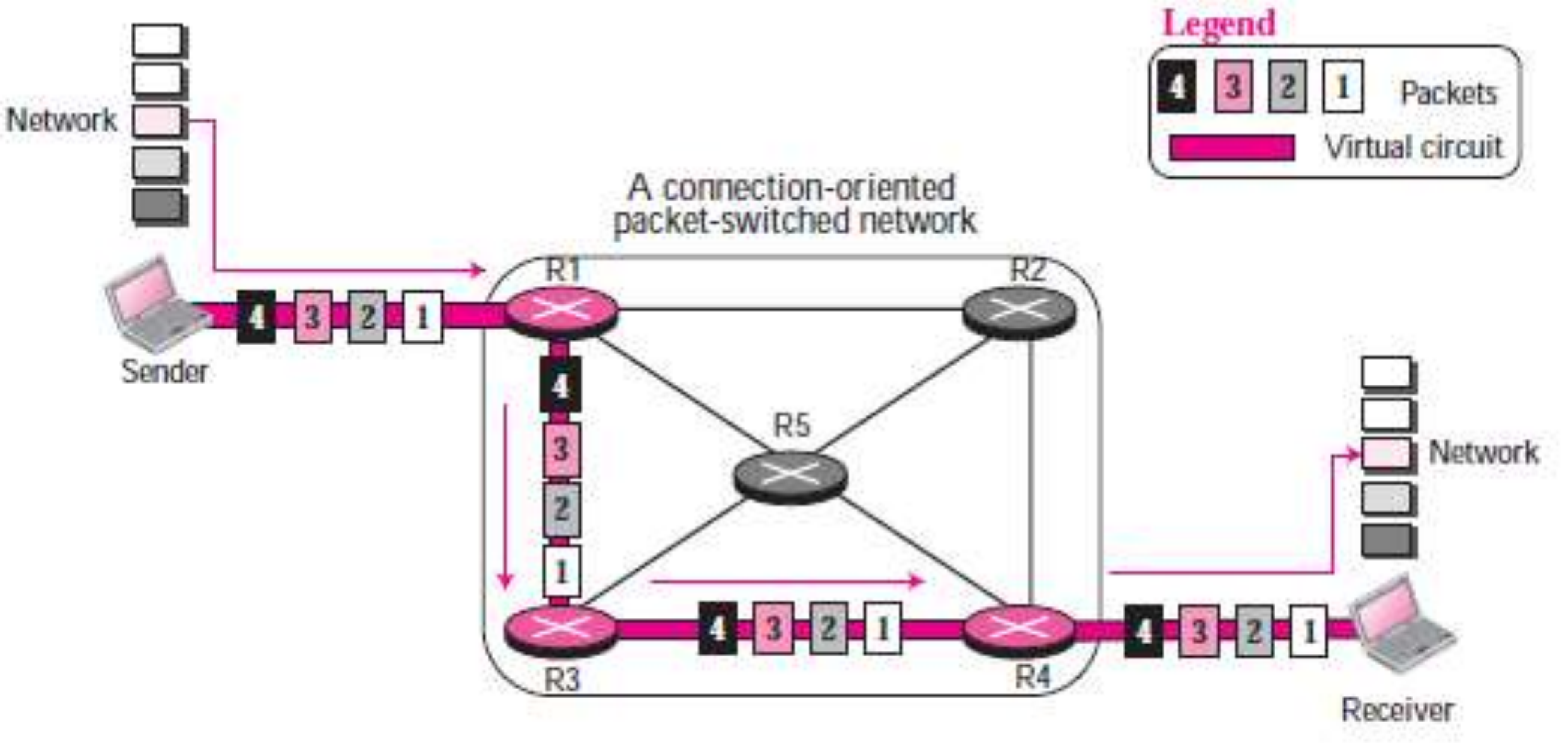
- Forouzan Behrouz, A. "Data Communication and networking." (2008).
- Peterson, Larry L., and Bruce S. Davie. *Computer networks: a systems approach*. Elsevier, 2007.
- Stallings, William. *Data and computer communications*. Pearson Education India, 2007.
- TCP/IP Protocol Suite, Behrouz A. Forouzan, McGraw-Hill Education, 4 Ed., 2010
- Web Links as mentioned in source
- https://www.eecs.yorku.ca/course_archive/2015-16/W/3214/CSE3214_10_PacketDelay_2016_posted.pdf

Theory Class 8

Overview

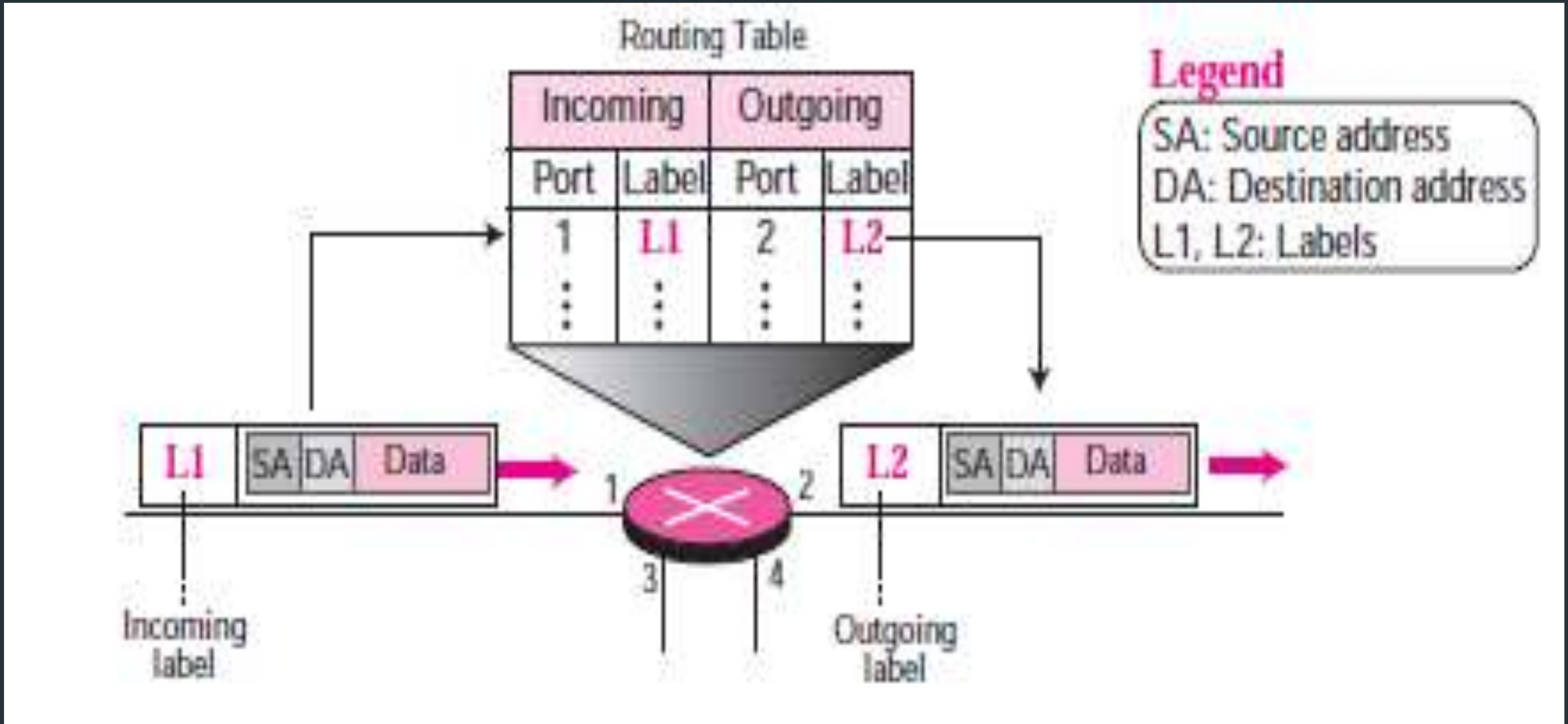
- Connection oriented packet switching (Refer: Theory_Class_6)
- Connectionless packet switching
- Advantages
- Datagram
- Virtual Circuit Switch
- Types of VCS
- Summary

Connection oriented Packet Switching



Source: Data Communications and Networking – Behrouz A. Forouzan

Forwarding Process in a Router (Connection-Oriented Service)



Source: Data Communications and Networking – Behrouz A. Forouzan

Advantages of Packet Switching over Circuit Switching

- Circuit switching less suited for data and other nonvoice transmission
- Line is often idle and resource wasted
- Fixed data rate – sender & receiver
- Flexibility is less compared to packets
- breaks messages into packets
- Sequentially send the packets by one by one
- Routing: Stream of packets (message) S->D
- Network layer
- Store and forward

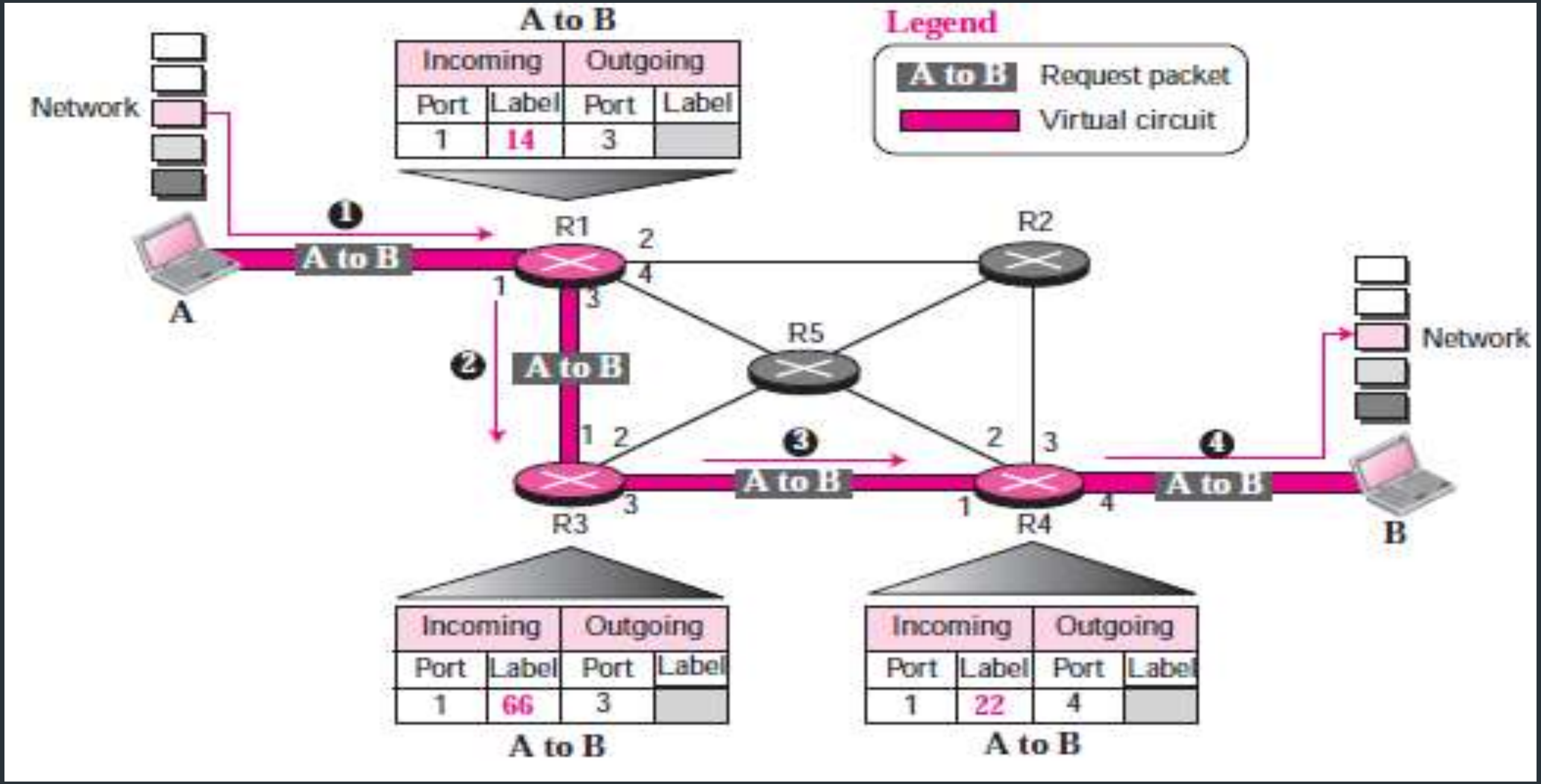
Datagram

- Does not bother about flow of packets
- Out of order
- Packet missing
- Receiver –
  - Reorder
  - Recover
  - Ex. Internet

Virtual Circuit Switching

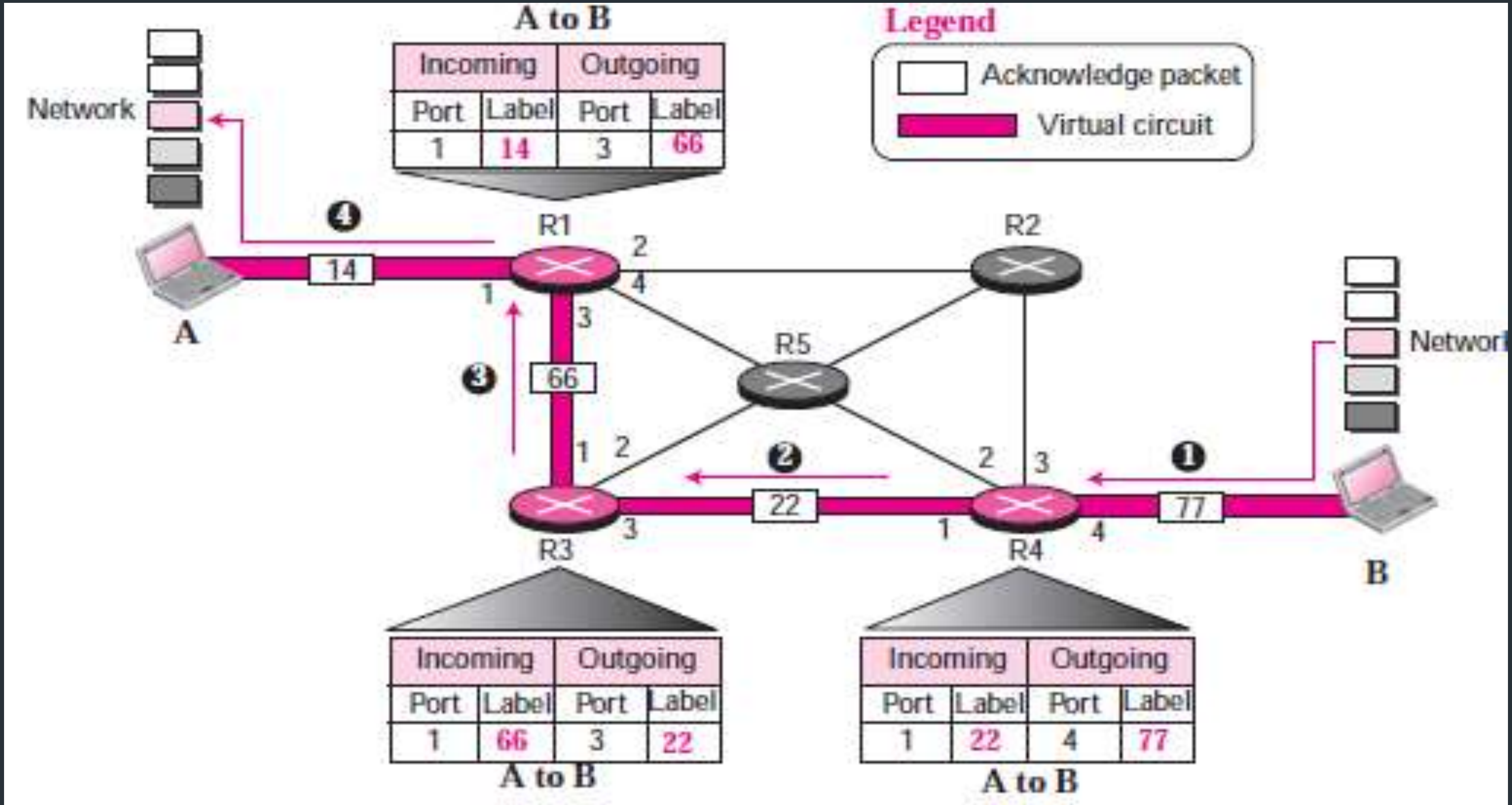
- Hybrid of Circuit switch and Datagram
- Single route is chosen
- Datalink layer
- VCS
  - Request Packet
  - Acknowledgment Packet
  - Teardown Packet
- Types
  - SVC
  - PVC

Request packet in VCS



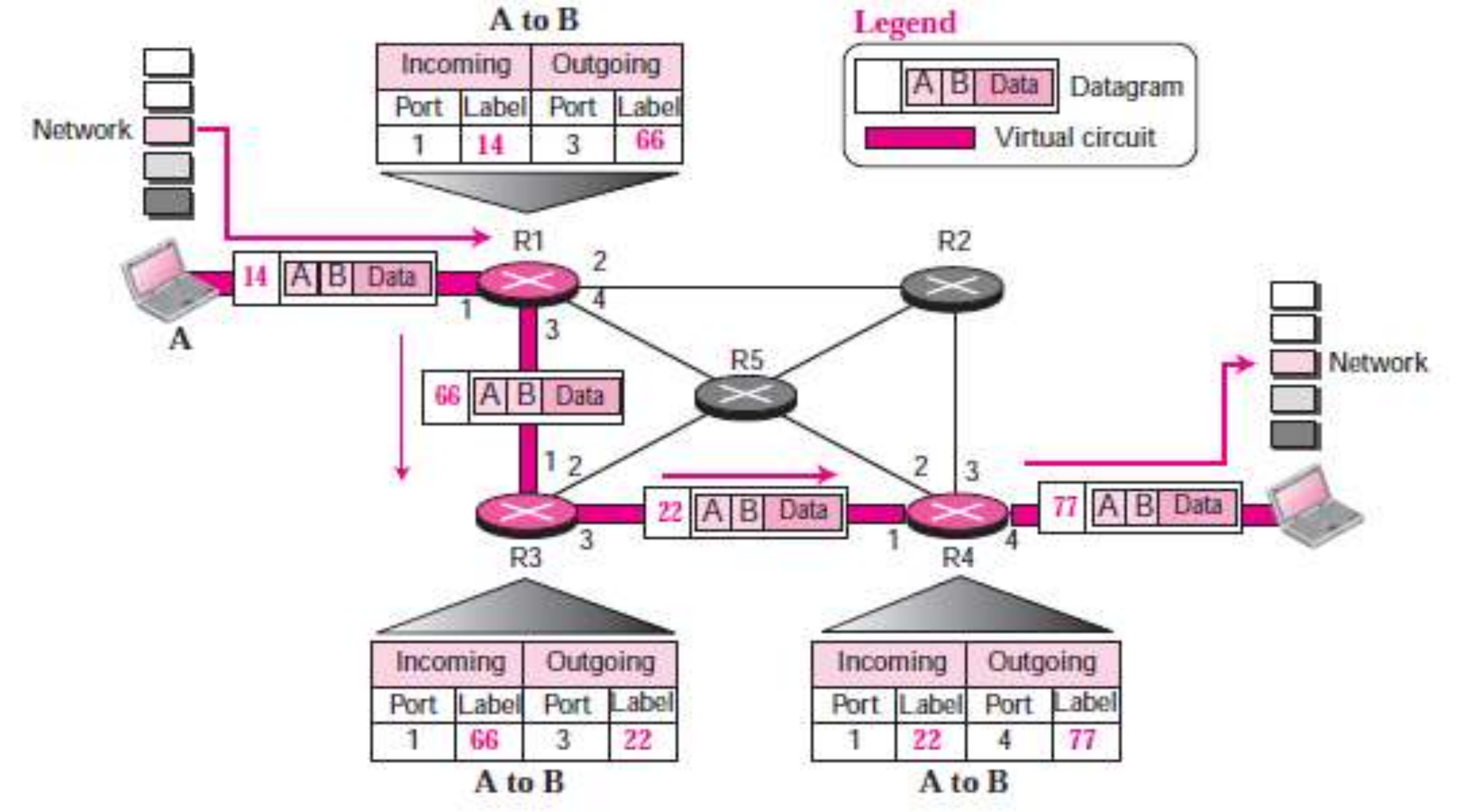
Source: Data Communications and Networking – Behrouz A. Forouzan

Acknowledgment Packet in VCN



Source: Data Communications and Networking – Behrouz A. Forouzan

Data Transfer Phase in VCN



Source: Data Communications and Networking – Behrouz A. Forouzan

Types of Virtual Circuit Switching

Switched Virtual Circuit

- Temporary
- It is created whenever is needed
- It exist only for the duration
- Specific exchanges

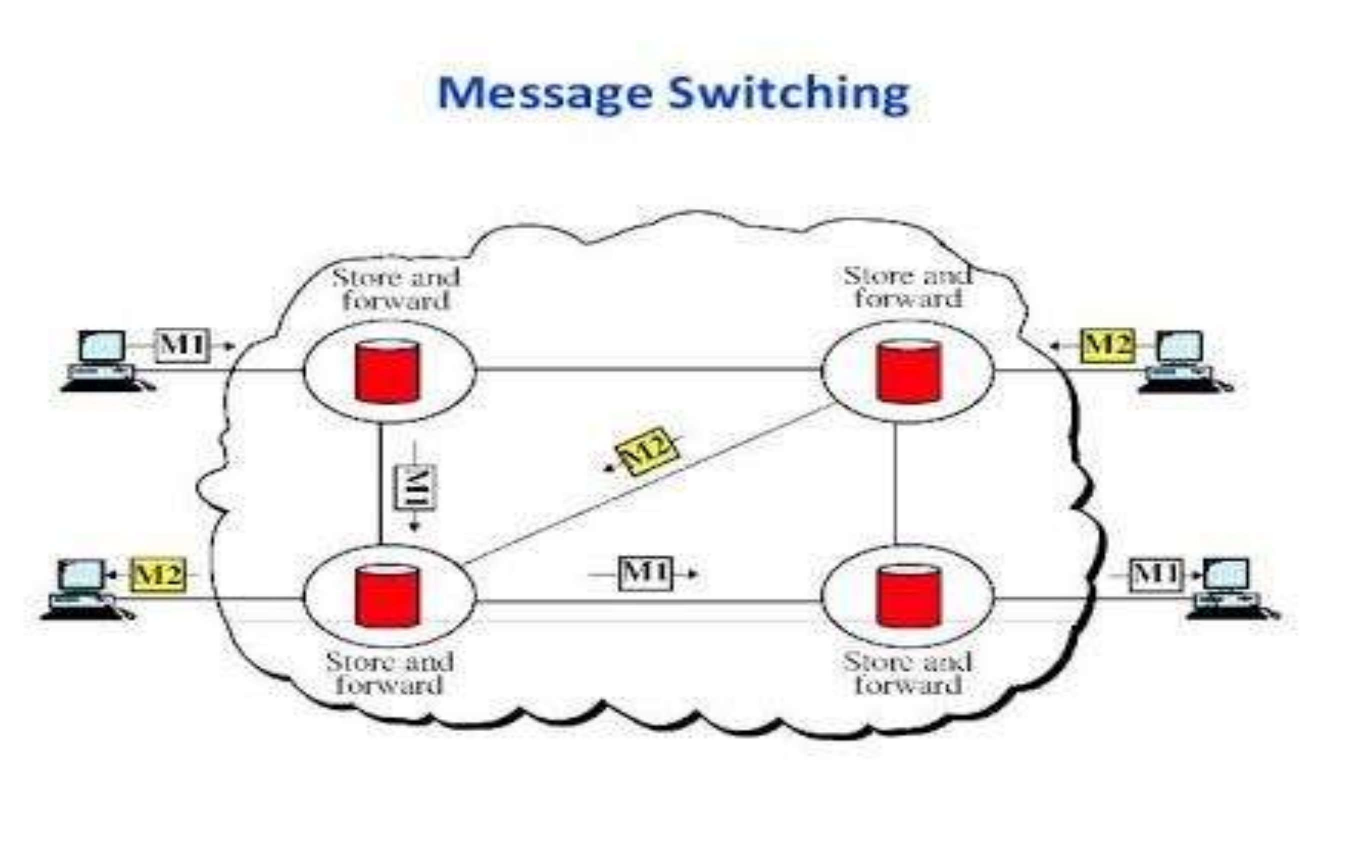
Permanent Virtual Circuit

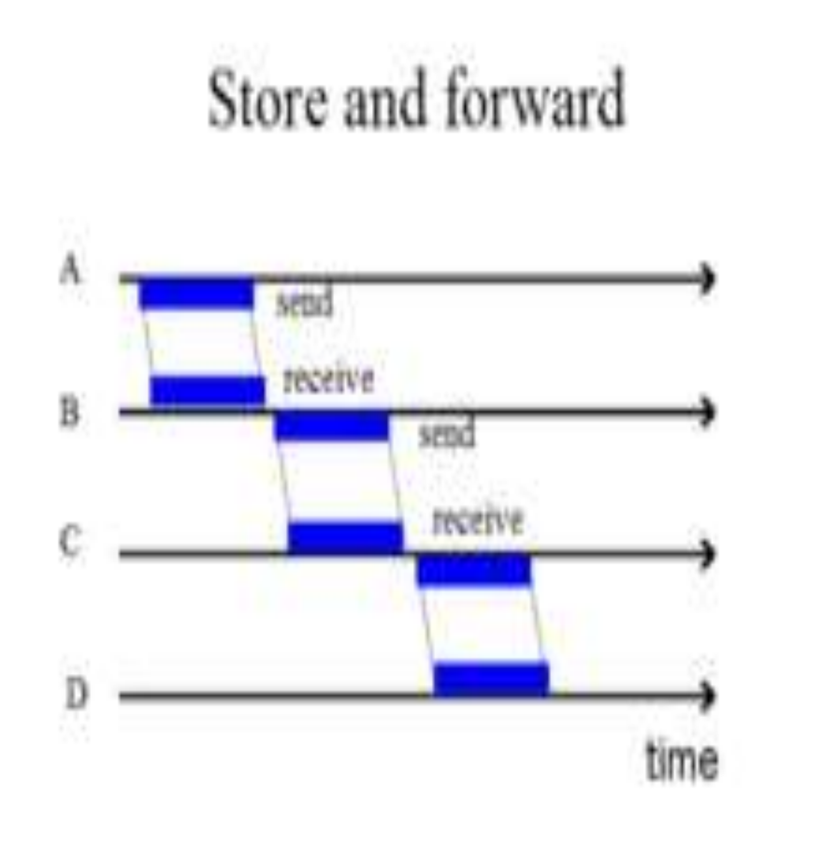
- Permanent
- Leased lines
- Continuous basis
- Virtual circuit for two users

Message Switching

- Data is routed in its entirety from the source node to the destination node,
- one hope at a time.
- Every intermediate switch in the network stores the whole message.
  - If the entire network's resources are engaged
  - network becomes blocked,
  - the message-switched network stores and delays the message until ample resources become available for effective transmission of the message.
  - Before the advancements in packet switching, message switching acted as an efficient substitute for circuit switching.

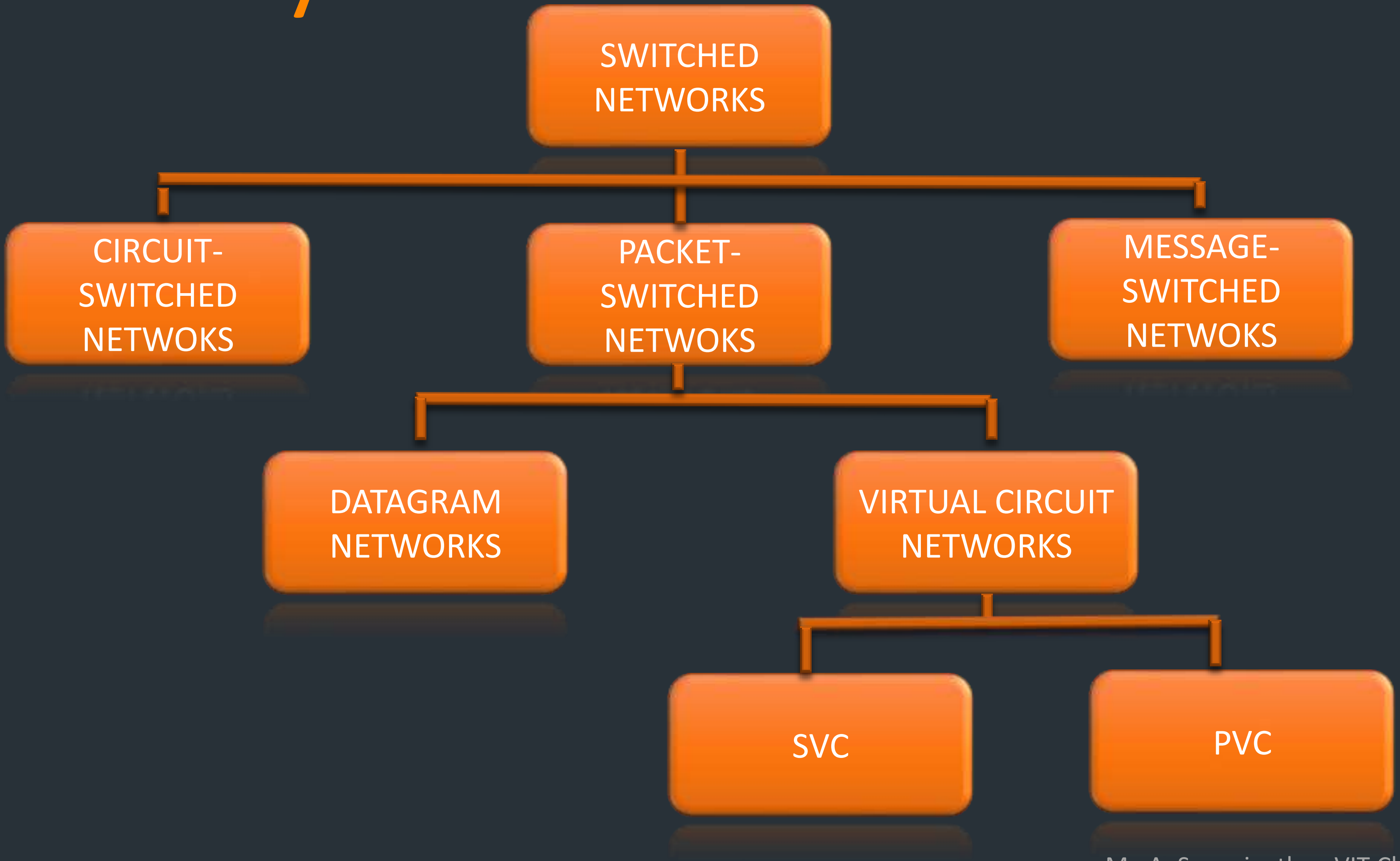
Message Switching





Source:
https://www.google.com/search?q=message+switched+network&sxsrf=ALeKk00uLjFaICh0OUVMIJ7MUtmKauT25g:1595391267165&tbm=isch&source=iu&ictx=1&fir=HgL74QZOj7NtHM%252C8TsE3Lod-0W1vM%252C_&vet=1&usg=AI4_-kR_Mk8vW4MVQ-oQYnzevSpgvQmAfQ&sa=X&ved=2ahUKEwiV9YbO_9_qAhWCbSsKHV35CDAQ_h0wAHoECAUQBA&biw=1366&bih=657

Summary



SWITCHED NETWORKS
CIRCUIT-SWITCHED NETWOKS
PACKET-SWITCHED NETWOKS
MESSAGE-SWITCHED NETWOKS
DATAGRAM NETWORKS
VIRTUAL CIRCUIT NETWORKS
SVC
PVC

Delays in Packet Switching

- Transmission Delay
- Propagation Delay
- Queuing Delay
- Processing Delay

Transmission Delay

- Time taken to put a packet onto link.
- time to send out / absorb all of the packet bits
- “store-and-forward” delay
- on the order of 10^{-6}seconds to 10^{-3} seconds

$$d_{transmission}\ [sec] = \frac{L\ [bit]}{R\ [bps]}$$

packet size [bits]

link transmission rate [bps]

Propagation delay

- Time taken by the first bit to travel from sender to receiver end of the link.
- depends on physical medium of the link
- on the order of 10^{-6}seconds

$$d_{propagation}\,[sec] = \frac{d\,[m]}{s\,[m/sec]}$$

distance between source and destination [m]

propagation speed of medium [m/s]

Processing Delay

- time it takes routers to process the packet header
- time required to process a packet – to check for bit errors, to determine output links, etc.
  - at source prior to sending,
  - at any intermediate router, and
  - at destination prior to delivering to application
- on the order of 10^{-6}seconds or less – often negligible

Queuing Delay

- time spent waiting in a queue at any point along the route depends on intensity and nature of traffic arriving at queue(s)

- on the order of 10^{-6} seconds to 10^{-3} seconds

  Average Queuing delay = (N-1)L/(2*R)

  where

  N = no. of packets

  L=size of packet

  R=bandwidth

Delay (Cont.)

For M hops and N packets,

Total delay =

M*(Transmission delay + propagation delay)

+

(M-1)*(Processing delay + Queuing delay)

+

(N-1)*(Transmission delay)

For N connecting link in the circuit,

Transmission delay = N*L/R

Propagation delay = N*(d/s)

References

- Forouzan Behrouz, A. "Data Communication and networking." (2008).
- Peterson, Larry L., and Bruce S. Davie. *Computer networks: a systems approach*. Elsevier, 2007.
- Stallings, William. *Data and computer communications*. Pearson Education India, 2007.
- TCP/IP Protocol Suite, Behrouz A. Forouzan, McGraw-Hill Education, 4 Ed., 2010
- Web Links as mentioned in source
- https://www.eecs.yorku.ca/course_archive/2015-16/W/3214/CSE3214_10_PacketDelay_2016_posted.pdf

Theory Class 9

Overview

- Problems based on
  - Transmission time
  - Propagation
  - Link utilization
  - Network delay

Problem 1

If the message size is 1 kb and bandwidth is 1 Mbps calculate the transmission time(T_t)?

Solution:

Given data: message size = 1kb

$= 1 \times 10^3$ bits

Bandwidth = 1×10^6 bits

Formula : T_t= Length of the message / Bandwidth

$T_t = 1 \times 10^3 / 1 \times 10^6$ bits

$= 1 \times 10^{-3}$ ms

Problem 2

If the distance between the sender and receiver is 3 Kms and velocity is 2*10^8 m/sec calculate the propagation time?

- Tp = d/s
- =3*10^3 / 2*10^8
- =1.5x10^-5
- = 1.5x10^-5 x 10^-6/10^-6
- = 15 µsec

Problem 3

What is the propagation time if the distance between the two nodes is 12000 Kms? Assume that the propagation speed to be 2.4*10^8 m/s in cable?

Tp=distance/speed

Propagation time = distance / propagation speed

= 12000x1000 / 2.4*10^8

= (5000 x 1000) / 10^8 sec

= (5000 x 1000 x 10^{-3}) / 10^8 x 10^{-3}

= (5000 x 1000 x 10^{-3} x10^3) / 10^8

= 50 ms

Problem 4

If the link utilization is to be 50% and propagation time is 10 m/sec calculate the transmission time?

- Formula: η = Transmission time / total time

$= T_t \,/\, T_t+T_p$

$50\,/100 = T_t \,/\, T_t+T_p$

$T_t + 2 \text{ x } T_p = 2\ T_p$

$T_t = 2/\ T_t+T_p$

$= 2 \text{ x } 10 \text{ m}$

$= 20 \text{ m sec}$

Problem 5

If the link utilization is 75% and RTr is 10 m/sec calculate the transmission time?

$\eta = T_t / T_t + 2 \times Tp$

$75 / 100 = T_t \ / T_t \ + R_{TT}$

$¾ = T_t \ / T_t \ + 10$

$3 T_t + 30 = 4 T_t$

$T_t \ = 30$ m sec

Problem 6

If the message size, bandwidth and RTT is L, B and RTT respectively calculate the link utilization of the sender?

- $\eta = T_t / T_t + 2 \times Tp$
- = (L/B)/(L/B)+R
- = L/L+BR
- = $\eta \times B$

Example Problem 7

Example [network delay – one packet, one hop]

Suppose that a user at one end sends a 1-Mbit file to a remote server on the other end over a data link operating at 64 kbps. Assume that we are using a fiber optic link with a propagation rate of the speed of light, approximately 3×10^8 m/sec, and that the distance is 4800 km. Ignore any processing or queuing delays. Compute the overall network delay, i.e. time to transmit the file?

Example Problem 8

Example [network delay – one packet, one hop]

For the same problem, now suppose that we have a 1-Gbps link. Compute the overall network delay, i.e. time to transmit the file, in this case?

Example Problem 9

Example [network delay – one packet, multiple hops]

A message needs to be transmitted over a path that involves two intermediate switches. For simplicity assume that the propagation delay and the bit rate of the transmission lines are the same, and ignore any queueing delay. Compute the overall end-to-end message delay in case of datagram packet switching?

Problem 10

Assume:

N = number of hops

L = message length [bits]

R = data rate [bps]

P = packet size [bits] (payload + header)

H = packet overhead

dpropagation = propagation delay per hop [sec]

Compute end-to-end delay for datagram switching, assuming:

N=4,

L=3200,

R=9600,

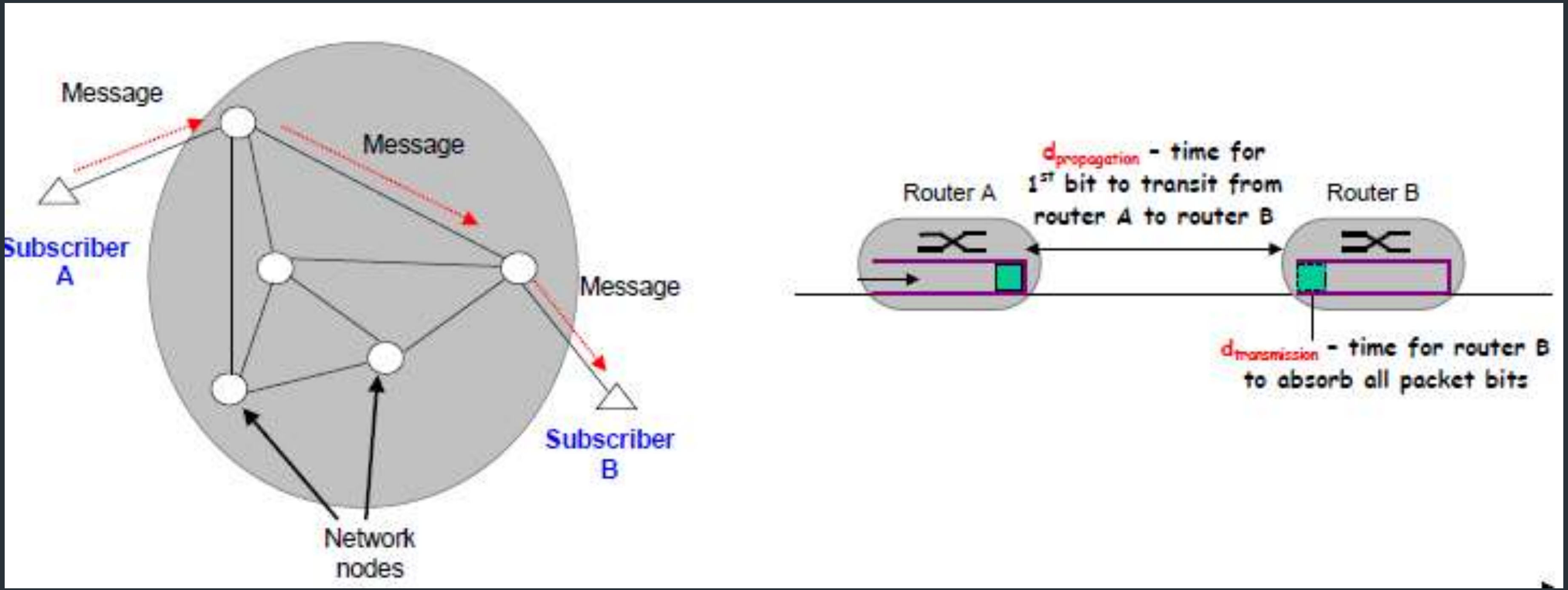
P=1024,

H=16,

dpropagation=0.001.

Ignore any queuing or processing delay.

Propagation Delay



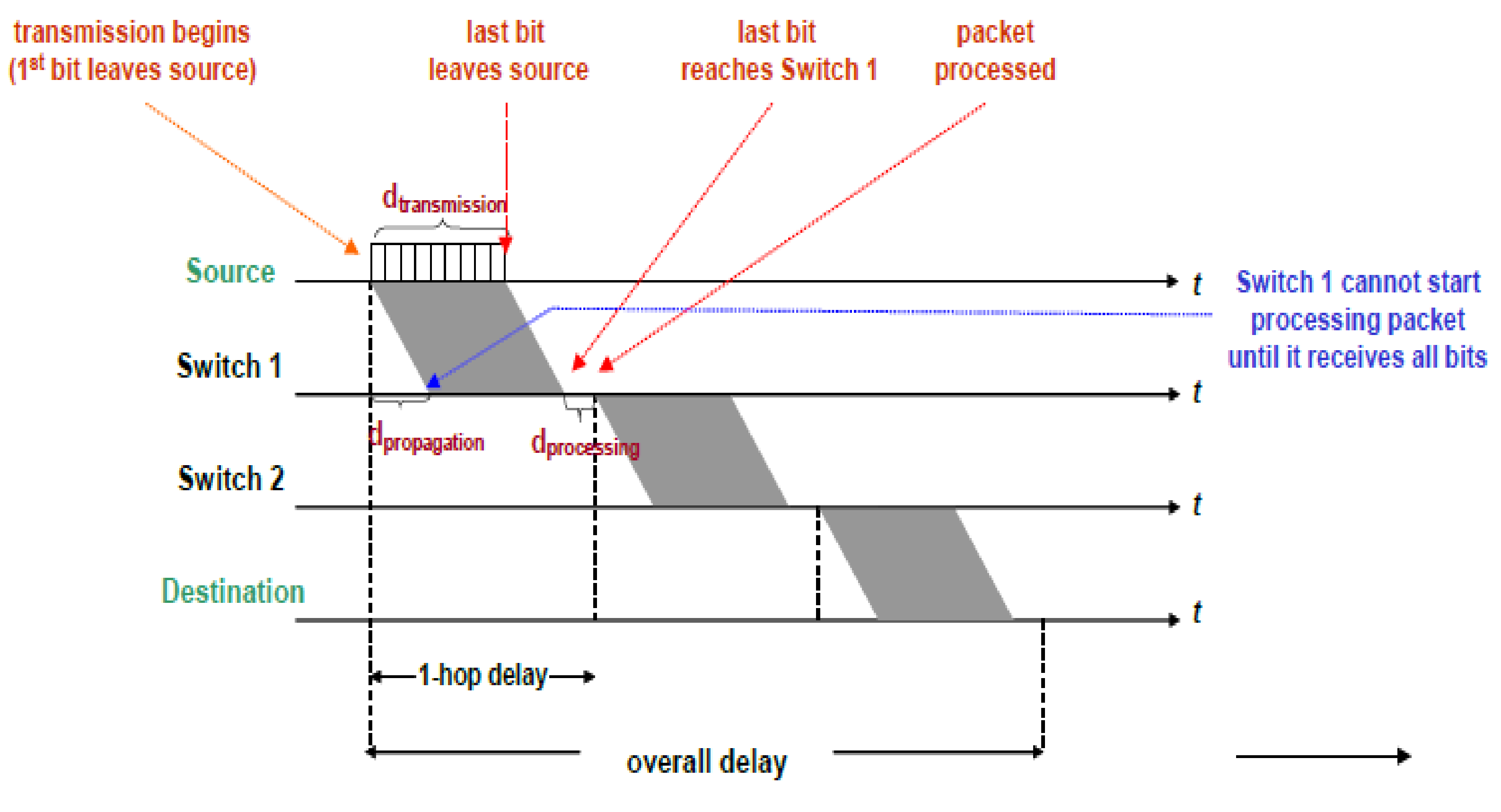
Source: Data Communications and Networking – Behrouz A. Forouzan

Propagation Delay

Delay in case of datagram packet switching

2 intermediate routers ⇒ 3 hops between source and destination

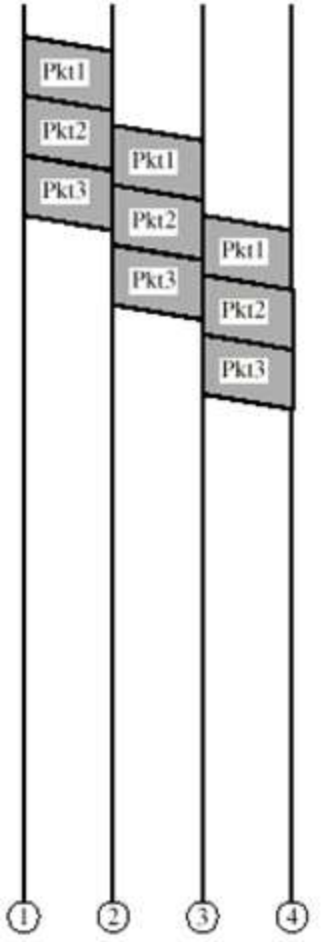
$$d_{total} = 3*d_{propagation} + 3*d_{transmission} + 3*d_{processing}$$



Source: Data Communications and Networking – Behrouz A. Forouzan

Network Delay

Example[network delay –multiple packets, multiplehops]]



Source: Data Communications and Networking – Behrouz A. Forouzan

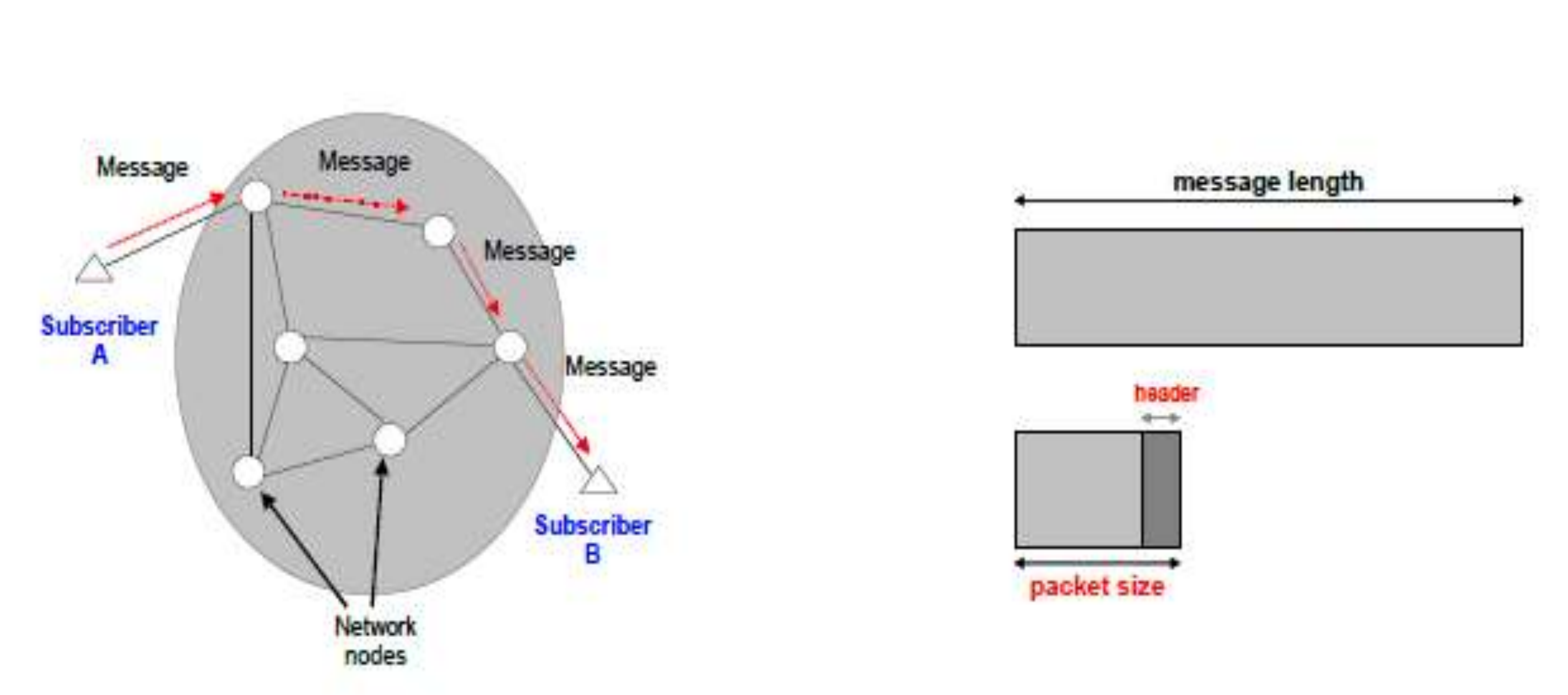
Solution

There are P – H = 1024 – 16 = 1008 data bits per packet.

A message of 3200 bits require four packets:

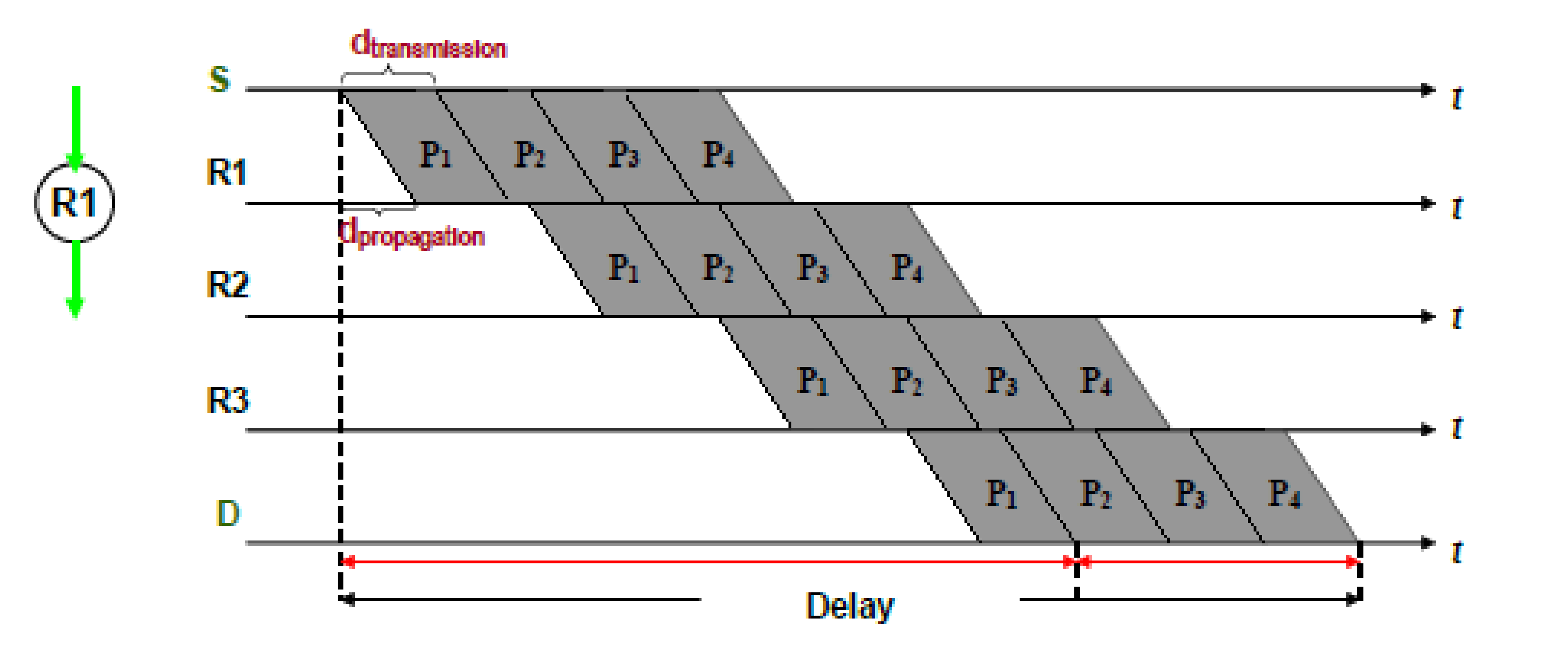
Np = 4.

(3200 bits/1008 bits/packet = 3.17 packets)



Source: Data Communications and Networking – Behrouz A. Forouzan

Solution



d = D1 + D2 + D3 + D4, where

D1 = Time to transmit entire 1st packet over all hops

D2 = Time to transmit entire 2nd packet

D3 = Time to transmit entire 3rd packet

D4 = Time to transmit entire 4th packet

time to absorb the rest of the message

Source: Data Communications and Networking – Behrouz A. Forouzan

Solution

$$D1 = N^*d_{propagation} + 4^*T =$$
$$= 4^*d_{propagation} + 4^*P/R =$$
$$= 4 \text{ x } 0.001 + 4^*1024/9600 =$$
$$= 0.427$$

$$D2 = D3 = D4 = T =$$
$$= (P/R) =$$
$$= (1024/9600) =$$
$$= 0.107$$

$$d = 0.427 + 3^*0.107 =$$
$$= 0.748 \text{ sec}$$

References

- Forouzan Behrouz, A. "Data Communication and networking." (2008).
- Peterson, Larry L., and Bruce S. Davie. *Computer networks: a systems approach*. Elsevier, 2007.
- Stallings, William. *Data and computer communications*. Pearson Education India, 2007.
- TCP/IP Protocol Suite, Behrouz A. Forouzan, McGraw-Hill Education, 4 Ed., 2010
- Web Links as mentioned in source
- https://www.eecs.yorku.ca/course_archive/2015-16/W/3214/CSE3214_10_PacketDelay_2016_posted.pdf

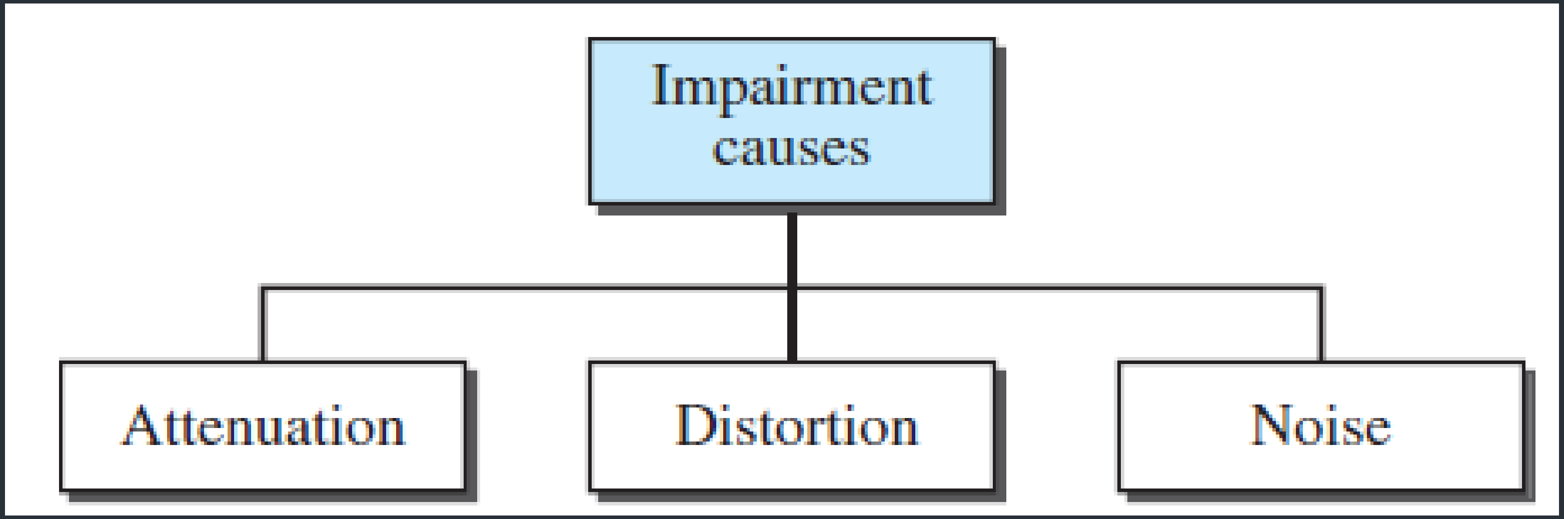
Theory Class 10

Overview

- Transmission impairment
- Attenuation
- Decibel
- Problems
- Noise
- Noise Types
- Type of channels
  - Noiseless channel
  - Noisy Channel
- Signal to Noise ration
- Data rate
- Performance
- Problems

Transmission Impairment

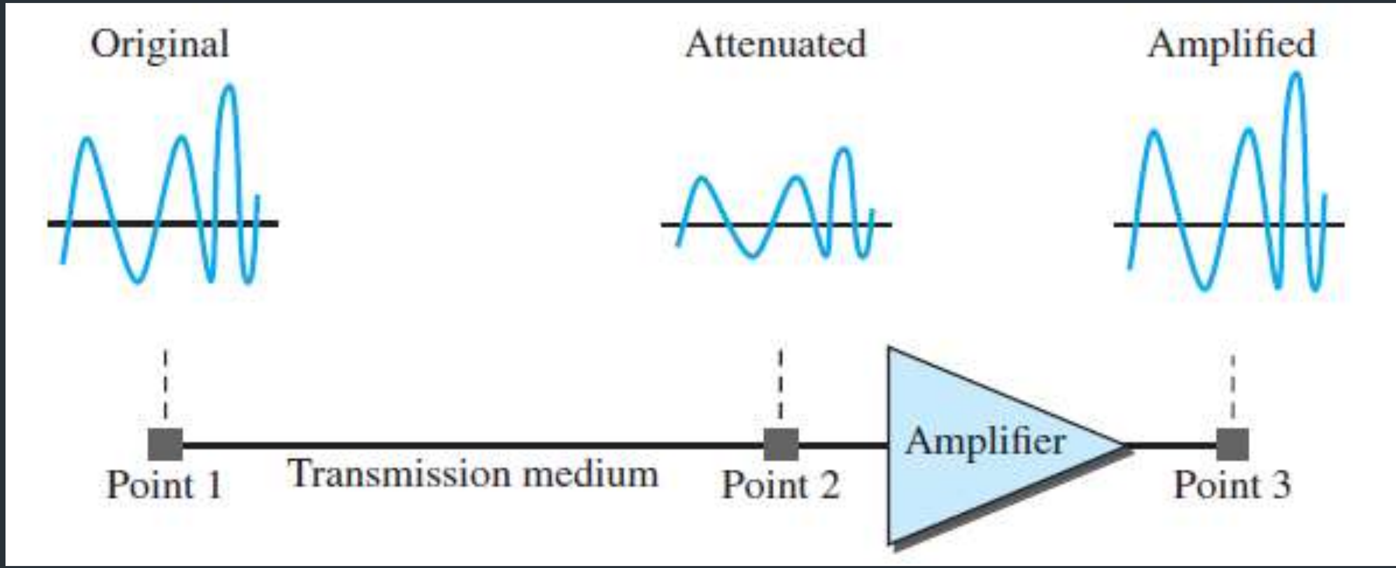
- The signal at the beginning of the medium is not the same as the signal at the end of the medium.
- What is sent is not what is received



Source: Data Communications and Networking – Behrouz A. Forouzan

Attenuation

- Loss of energy.
- When a signal travels through a medium, it loses some of its energy in overcoming the resistance of the medium.
- Some of the electrical energy in the signal is converted to heat.
- To compensate for this loss, amplifiers are used to amplify the signal.



Source: Data Communications and Networking – Behrouz A. Forouzan

Decibel

- The relative strengths of two signals or one signal at two different points.
- Decibel is negative if a signal is attenuated and positive if a signal is amplified.

$$\mathrm{dB} = 10\log_{10}\frac{P_2}{P_1}$$

Source: Data Communications and Networking – Behrouz A. Forouzan

Exercise 1

- Assume a signal travels through a transmission medium and its power is reduced to one-half. Compute the attenuation (loss of power).

Solution – Exercise 1

$P_2 = 0.5 * P_1$

$$10 \log_{10} (P_2/P_1) = 10 \log 0.5 * P_1/P_1$$
$$= 10 \log 0.5$$
$$= 10*(-0.3)$$
$$= -3 \text{ dB}$$

A loss of 3dB is equivalent to losing one-half of the power.

Exercise 2

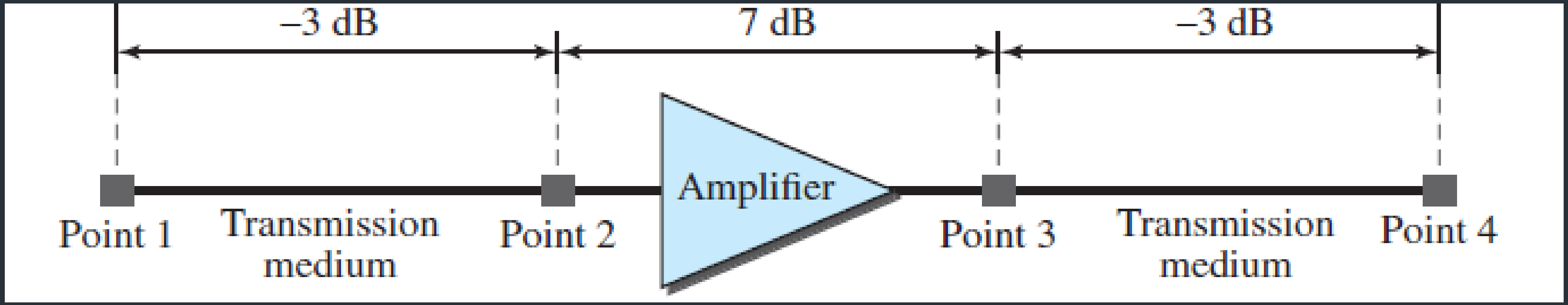
- A signal travels through an amplifier, and its power is increased 10 times. Compute the amplification (Gain of power).

Solution – Exercise 2

$$10 \log_{10} (P_2/P_1) = 10 \log_{10} (10P_1/P_1)$$
$$= 10 \log_{10} 10$$
$$= 10 (1)$$
$$= 10 \text{ dB}$$

Exercise 3

- Compute the resultant decibel of the signal that travels from point 1 to point 4.



Source: Data Communications and Networking – Behrouz A. Forouzan

Solution – Exercise 3

Decibel Value = -3 + 7 -3 = 1

The signal has gained in power

Exercise 4

- Signal power in milliwatts

$$dB_m = 10 \log_{10} P_m$$

- Calculate the power of a signal if its $dB_m = -30$.

Solution – Exercise 4

$dB_m = 10 \log_{10} P_m$

$-30 = 10 \log_{10} P_m$

$\log_{10} P_m = -3$

$P_m = 10^{-3}$ mW

Exercise 5

- The loss in a cable is usually defined in decibels per kilometer (dB/km).
- Assume the signal at the beginning of a cable with −0.3 dB/km has a power of 2 mW. Compute the power of the signal at 5 km.

Solution – Exercise 5

The loss in the cable in decibels = 5 * (-0.3)

= -1.5dB

Power Calculation

$10 \log_{10} P_2/P_1 = -1.5$

$P_2/P_1 = 10^{-0.15}$

$= 0.71$

$P_2 = 0.71P_1$

$= 0.7 * 2mW$

$=1.4mW$

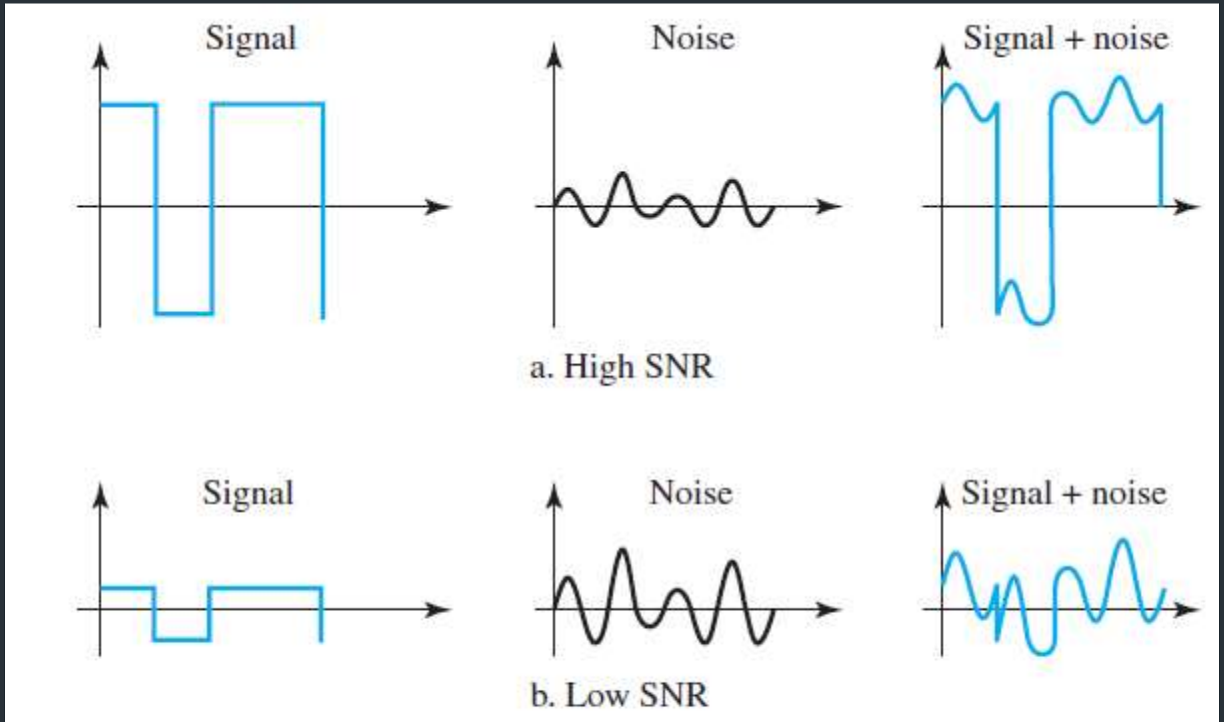
Noise

- Thermal Noise
- Induced Noise
- Cross Talk
- Impulse Noise

Signal-to-Noise Ratio (SNR)

$$\text{SNR} = \frac{\text{average signal power}}{\text{average noise power}}$$

$$\text{SNR}_{\text{dB}} = 10 \log_{10} \text{SNR}$$



Source: Data Communications and Networking – Behrouz A. Forouzan

Exercise 6

- The power of a signal is 10 mW and the power of the noise is 1 μW.
  - Compute the values of SNR and SNR_{dB}
- Compute the values of SNR and SNR_{dB} for a noiseless channel.

Solution – Exercise 6

SNR = (10,000 μw)/(1 μw) = 10,000

SNR_{dB}= 10 log $_{10}$ 10,000

=10 log $_{10}$ 10^4

=40

Noiseless Channel

SNR = (Signal Power)/0 = ∞

SNR_{dB}= 10 log 10 ∞

= ∞

Data Rate Limits

- how fast we can send data, in bits per second, over a channel.
- Data rate depends on three factors:
  - **The bandwidth available**
  - **The level of the signals we use**
  - **The quality of the channel (the level of noise)**
- Nyquist for a noiseless channel
- Shannon for a noisy channel

Noiseless Channel: Nyquist bit Rate

- BitRate = 2 * bandwidth * $\log_2 L$
- bandwidth is the bandwidth of the channel
- L is the number of signal levels used to represent data
- BitRate is the bit rate in bits per second

Exercise 7

- Consider a noiseless channel with a bandwidth of 3000 Hz transmitting a signal with two signal levels. Compute the maximum bit rate.

Solution – Exercise 7

$$\text{BitRate} = 2 * 3000 * \log_2 2$$
$$= 6000 \text{ bps}$$

Exercise 8

- Consider a noiseless channel transmitting a signal with four signal levels (for each level, we send 2 bits). Compute the maximum bit rate.

Solution – Exercise 8

Bitrate = $2 * 3000 * \log_2 4$

$= 12000$ bps

Exercise 9

- We need to send 265 kbps over a noiseless channel with a bandwidth of 20 kHz. How many signal levels do we need?

Solution – Exercise 9

$265{,}000 = 2 * 20{,}000 * \log_2 L$

$\log_2 L = 6.625$

$L = 2^{6.625}$

$= 98.7$ signal levels

Noisy Channel: Shannon Capacity

To determine the theoretical highest data rate for a noisy channel

$$\text{Capacity} = \text{bandwidth} \times \log_2(1 + \text{SNR})$$

- bandwidth is the bandwidth of the channel
- SNR is the signal-to noise ratio
- capacity is the capacity of the channel in bits per second

Exercise 10 & 11

- Consider an extremely noisy channel in which the value of the signal-to-noise ratio is almost zero. For this channel the capacity *C* is ______.
- A telephone line normally has a bandwidth of 3000 Hz (300 to 3300 Hz) assigned for data communications. The signal-to-noise ratio is usually 3162. For this channel the capacity is ___________.

Solution – Exercise 10 & 11

$C = B \log_2 (1+SNR)$ $= B \log_2 (1+0)$ $= B \log_2 1$ $= B * 0$ $= 0$	$C = B \log_2(1+SNR)$ $= 3000 \log_2(1+3162)$ $= 3000 * 11.62$ $= 34{,}860$ bps

Exercise 12

- The signal-to-noise ratio is often given in decibels. Assume that SNR_{dB} = 36 and the channel bandwidth is 2 MHz. The theoretical channel capacity is _________________.

Solution – Exercise 12

$SNRdB = 10 \log_{10} SNR$

$SNR = 10^{SNRdB/10}$

$SNR = 10^{3.6} = 3981$

$C = B \log_2(1 + SNR)$

$= 2 * 10^6 * \log_2 3982$

$= 24$ Mbps

Performance

- Bandwidth
- Throughput
- Latency (Delay)
  - Transmission delay
  - Propagation delay
  - Queuing delay
- Jitter

Exercise 13

- A Network with bandwidth of 10 Mbps can pass only an average of 12,000 frames per minute with each frame carrying an average of 10,000bits. Calculate the throughput of the given network.

Solution – Exercise 13

Throughput= (12,000 * 10,000) / 60

= 2 Mbps

The throughput is almost one-fifth of the bandwidth.

References

- Forouzan Behrouz, A. "Data Communication and networking." (2008).
- Peterson, Larry L., and Bruce S. Davie. *Computer networks: a systems approach*. Elsevier, 2007.
- Stallings, William. *Data and computer communications*. Pearson Education India, 2007.
- TCP/IP Protocol Suite, Behrouz A. Forouzan, McGraw-Hill Education, 4 Ed., 2010
- Web Links as mentioned in source
- https://www.eecs.yorku.ca/course_archive/2015-16/W/3214/CSE3214_10_PacketDelay_2016_posted.pdf

Theory Class 11

Overview

- What is Socket?
- Socket Programming API (C language)
  - Socket
  - Bind
  - Listen
  - Accept
  - Connect
  - Send, Recv
  - Close
- Example

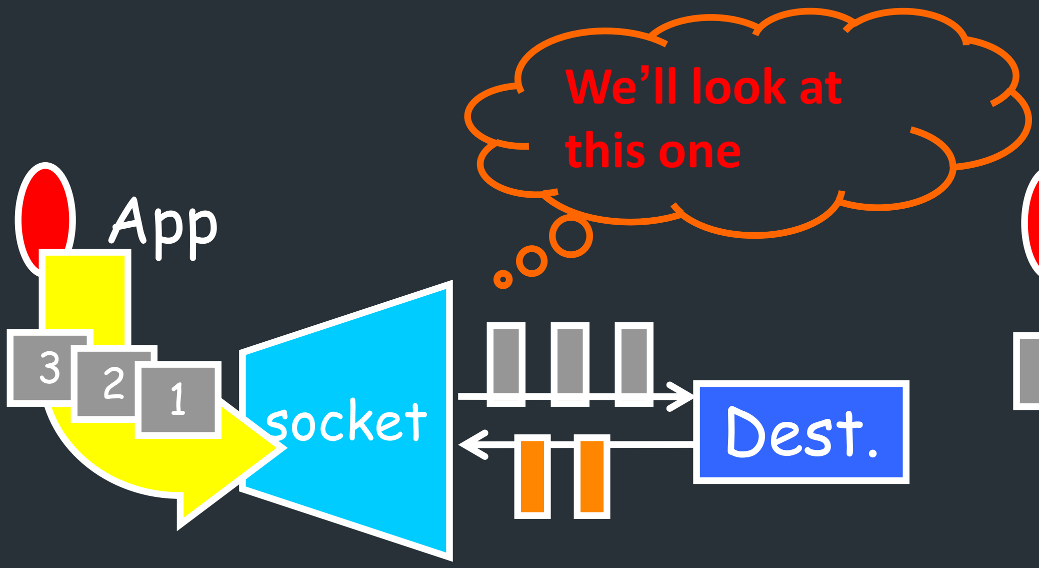
What is a socket?

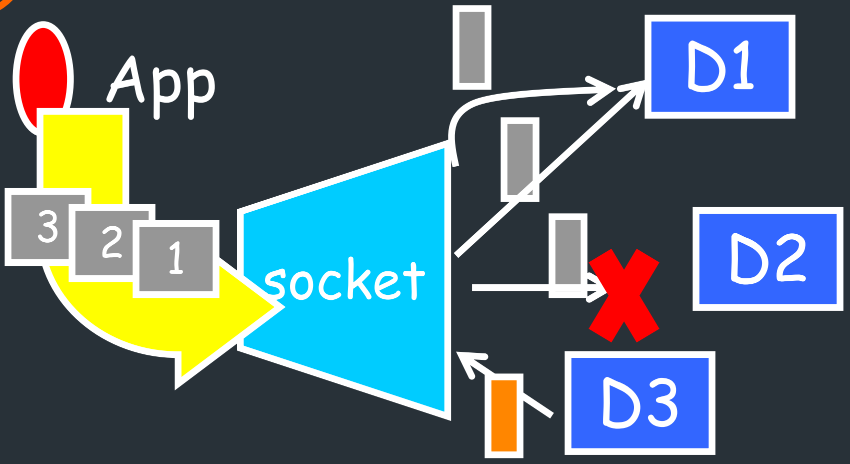
- An interface between application and network
  - The application creates a socket
  - The socket *type* dictates the style of communication
    - reliable vs. best effort
    - connection-oriented vs. connectionless

- Once configured, the application can
  - pass data to the socket for network transmission
  - receive data from the socket (transmitted through the network by some other host)

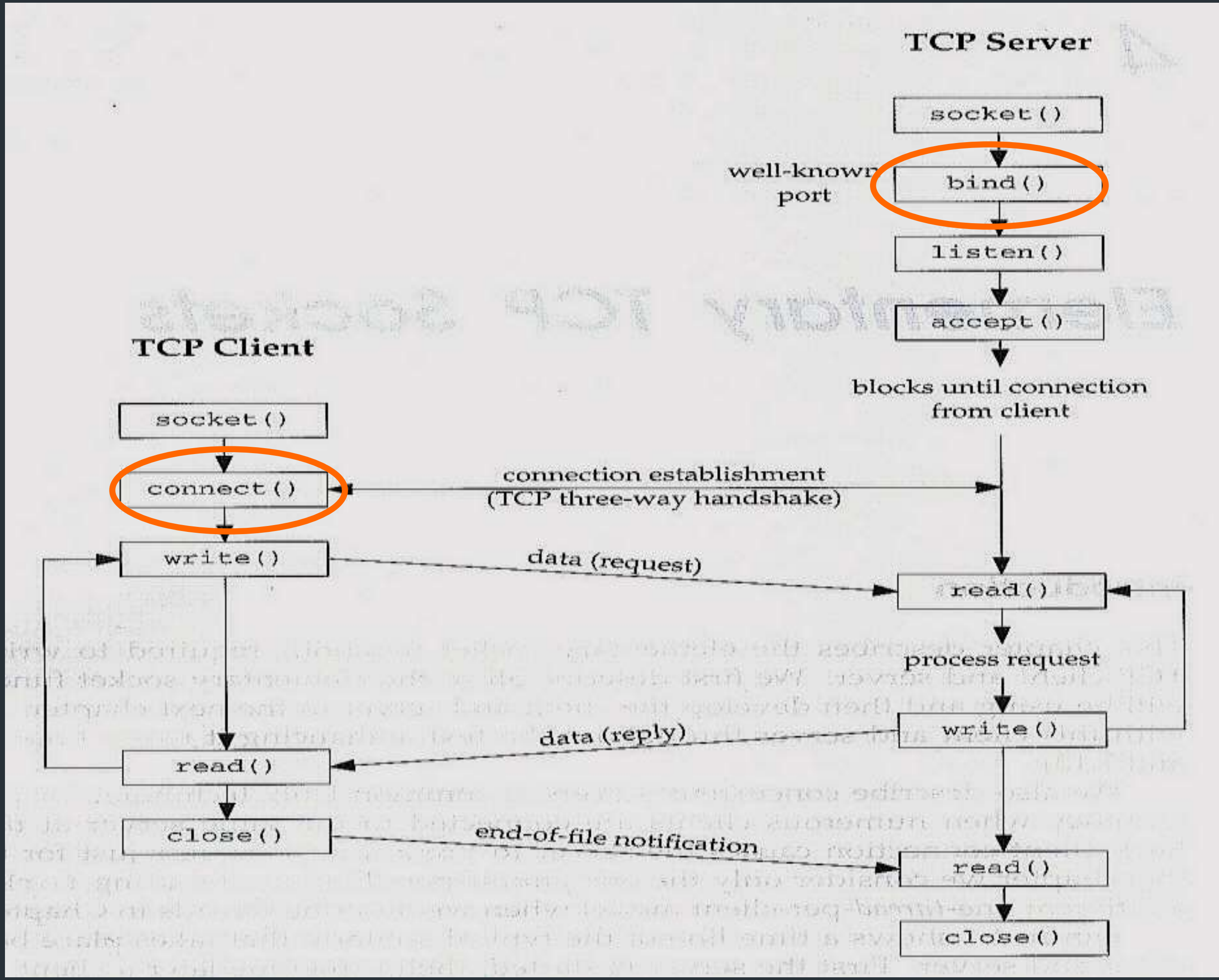
Two essential types of sockets

- SOCK_STREAM
  - TCP sockets
  - reliable delivery
  - in-order guaranteed
  - connection-oriented
- SOCK_DGRAM
  - UDP sockets
  - unreliable delivery
  - no order guarantees
  - no notion of “connection”

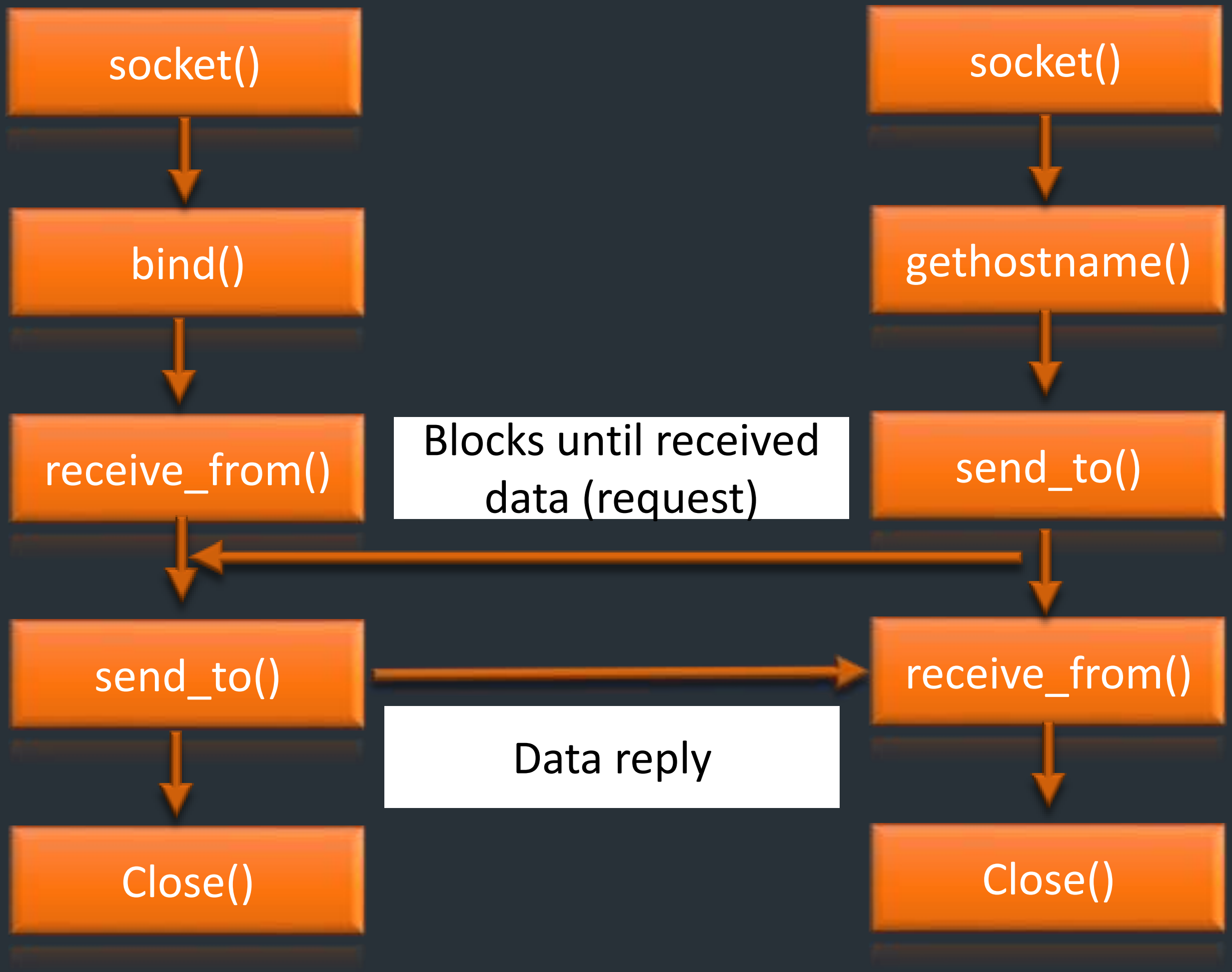




Socket Creation



Interaction of Server – Client in UDP



Socket Creation in C: socket

- int s = socket(domain, type, protocol);

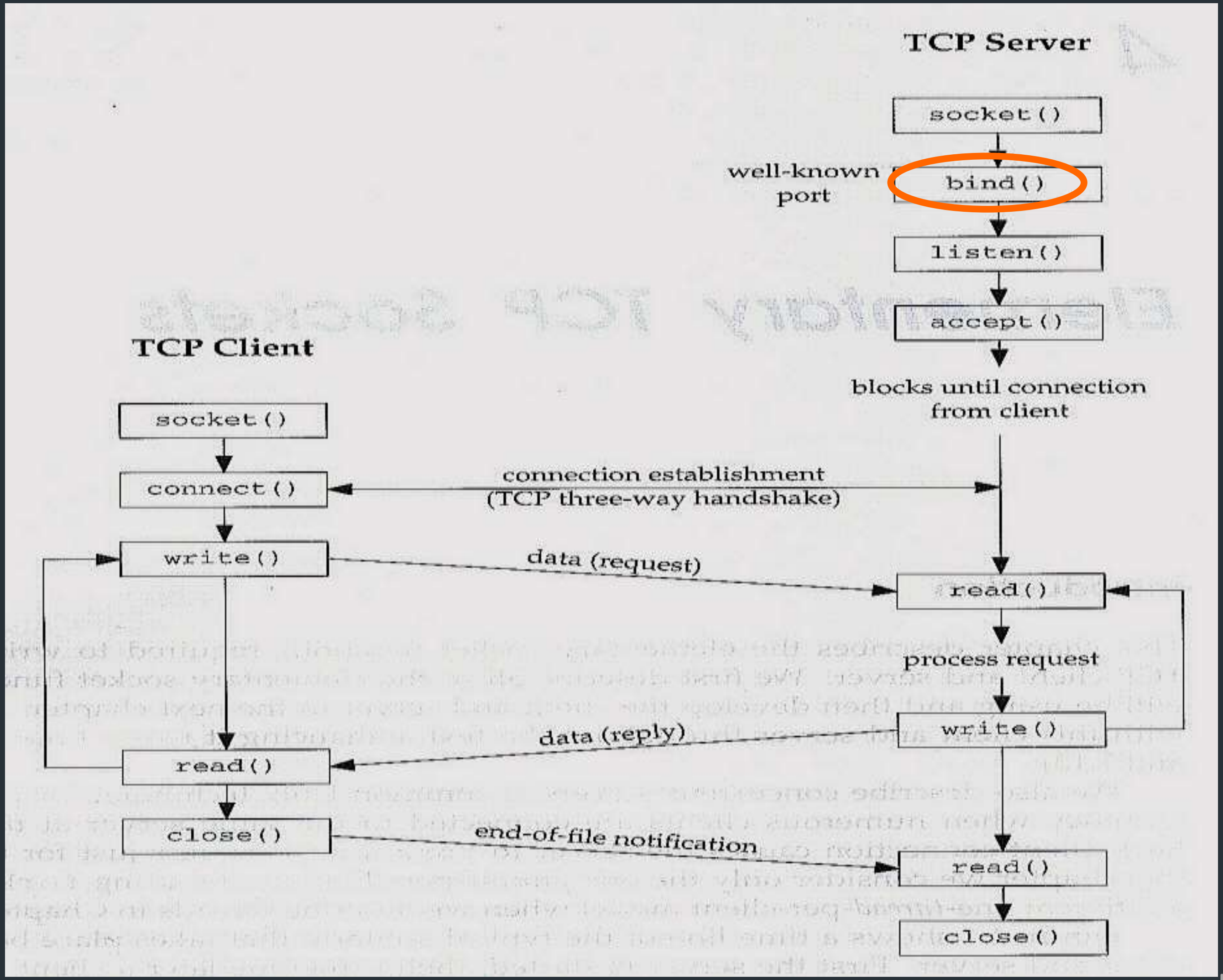
where

- s: socket descriptor, an integer (like a file-handle)
- domain: integer, communication domain
  - e.g., AF_INET (IPv4 protocol)
  - **Note. We'll use AF_INET**
- type: communication type
  - SOCK_STREAM: reliable, 2-way, connection-based service
  - SOCK_DGRAM: unreliable, connectionless
  - **Note. We'll use SOCK_STREAM**
- protocol: e.g., TCP or UDP
  - use IPPROTO_TCP or IPPROTO_UDP to send/receive TCP or UDP packets
  - **Note. We'll use IPPROTO_TCP**

Socket Creation in C: socket

```
/* sample code to create a socket */

hSocket=socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
```

Binds a socket to an address



Addresses, Ports and Sockets

- Like apartments and mailboxes
  - You are the application
  - Street address of your apartment building is the IP address
  - Your mailbox is the port
  - The post-office is the network
  - The socket is the key that gives you access to the right mailbox
- Q: How do you choose which port a socket connects to?

Addresses, Ports and Sockets

- Choose a port number that is registered for general use, from 1024 to 49151
  - Do not use ports 1 to 1023. These ports are reserved for use by the Internet Assigned Numbers Authority (IANA)
  - Avoid using ports 49152 through 65535. These are dynamic ports that operating systems use randomly. If you choose one of these ports, you risk a potential port conflict

The bind function

- associates a port for use by the socket
- int status = bind(sock, & addrport, size)

Where

- status: return status, 0 if successful, -1 otherwise
- sock: socket being used
- addrport: address structure
- size: the size (in bytes) of the addr port structure

Bind is non-blocking: returns immediately

The struct sockaddr

- The sockaddr_in structure has four parts:
  - sin_family: address family (e.g., **AF_INET IP addresses)**
  - sin_port: port number
  - sin_addr: IP-address
  - sin_zero: // un-used

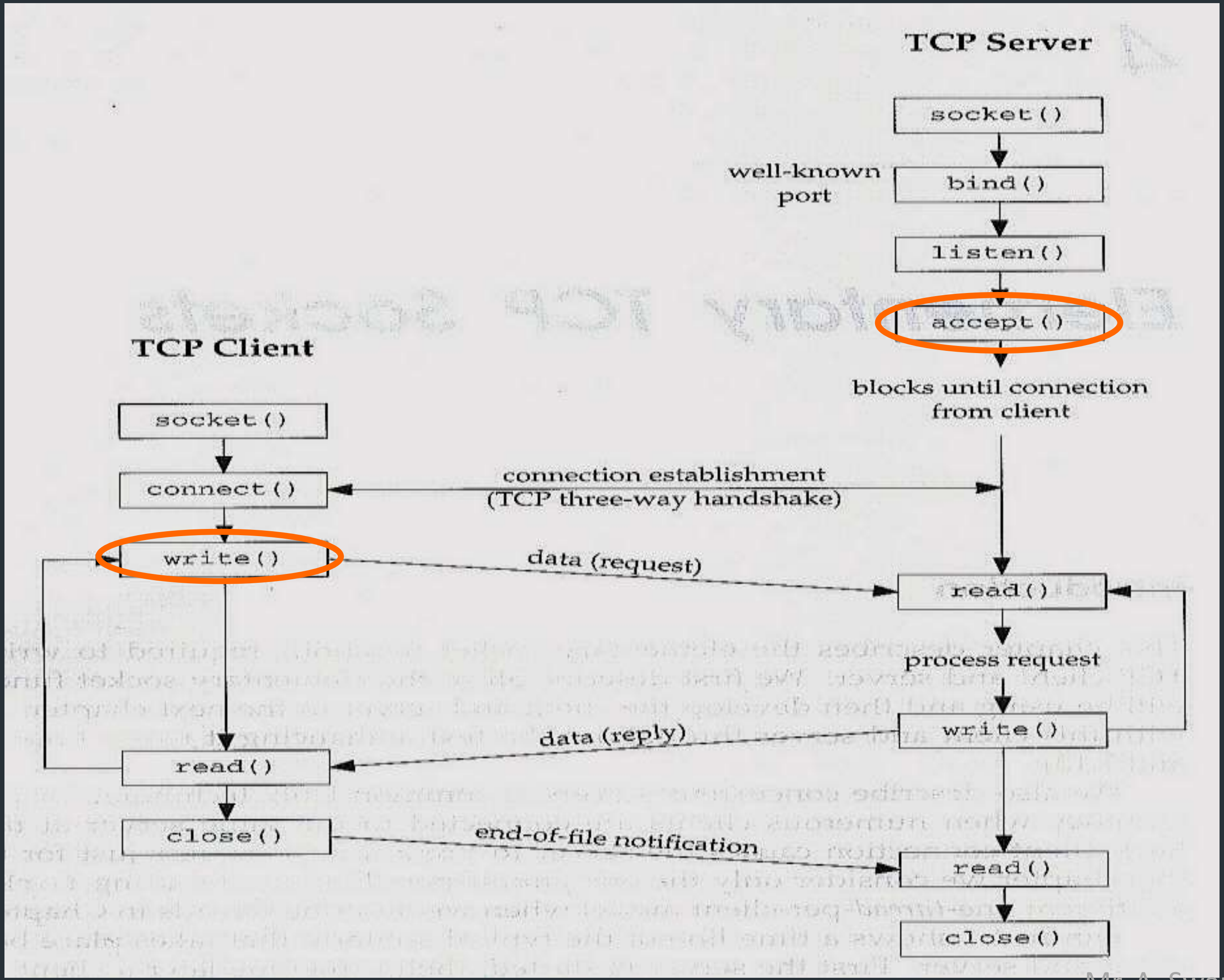
- #include <netinet/in.h> // All pointers to socket address structures are often cast to pointers to this type before use in various functions and system calls:

struct sockaddr
{
unsigned short *sa_family*;
// address family, AF_xxx
 	
char *sa_data*[14];
// 14 bytes of protocol address
};
	// IPv4 AF_INET sockets:
	
	struct sockaddr_in
	{
	short *sin_family*;
	// e.g. AF_INET, AF_INET6
	unsigned short *sin_port*;
	// e.g. htons(3490)
	struct in_addr *sin_addr*;
	// see struct in_addr, below
	char *sin_zero*[8];
	// zero this if you want to
	};
		struct in_addr
		{
		unsigned long *s_addr*;
		// load with inet_pton()
		};

Example(Server)


```
// first, create and fill in values for the sockaddr_in structure Address
struct sockaddr_in Address;
/* create Address stucture */

Address.sin_family = AF_INET;
/* AF_INET represents the address family INET for Internet sockets. */

Address.sin_port = htons(nHostPort);
/* The function htons() converts from host byte order to network byte
   order*/

Address.sin_addr.s_addr = INADDR_ANY;
/* INADDR_ANY allows us to work without knowing the IP address of the
   machine the client program is running  on (very convenient) */

// next, bind the socket to the port
if( bind(hServerSocket, (struct sockaddr *) &Address, sizeof(Address)) == -1)
   {
        printf("\nCould not connect to host\n");
        return -1;
   }
```

Connection setup



Connection setup: listen & accept

- The listen function prepares a bound socket to accept incoming connections
- int status = listen(sock, queuelen)

where

  - status: return value, 0 if listening, -1 if error
  - sock: socket being used
  - queuelen: number of active participants that can “wait” for a connection

- Example code:

```
if (listen(hServerSocket, 1) == -1)
{
      printf("\nCould not listen\n");
      return -1;
}
```

Connection setup: listen & accept

- Use the accept function to accept a connection request from a remote host
- The function returns a socket corresponding to the accepted connection
- int s = accept(sock, &cliaddr, &addrlen)
  – s: new socket used for data-transfer
  – sock: original socket being listened on (e.g., server)
  – cliaddr: address structure of the active participant (e.g., client)
    - The accept function updates/returns the sockaddr structure with the client's address information
  – addrlen: size (in bytes) of the client sockaddr structure
    - The accept function updates/returns this value

accept is blocking: waits for connection before returning

- Example code:

```
hSocket = accept(hServerSocket, (struct sockaddr *) &Cliaddr, (socklen_t *)
    &AddressSize);
/* socklen_t is socket address length type, defined in sys/socket.h; in our example
    code it is being cast from a pointer to an integer */
```

Client

create socket and connect to remote host

- First, the client must create a socket (socket call as before) and fills in its address structure
- Then, the client connects to the remote host
  - The connect function is used by a client program to establish communication with a remote entity
- int status = connect(sock, &servaddr, addrlen);

where

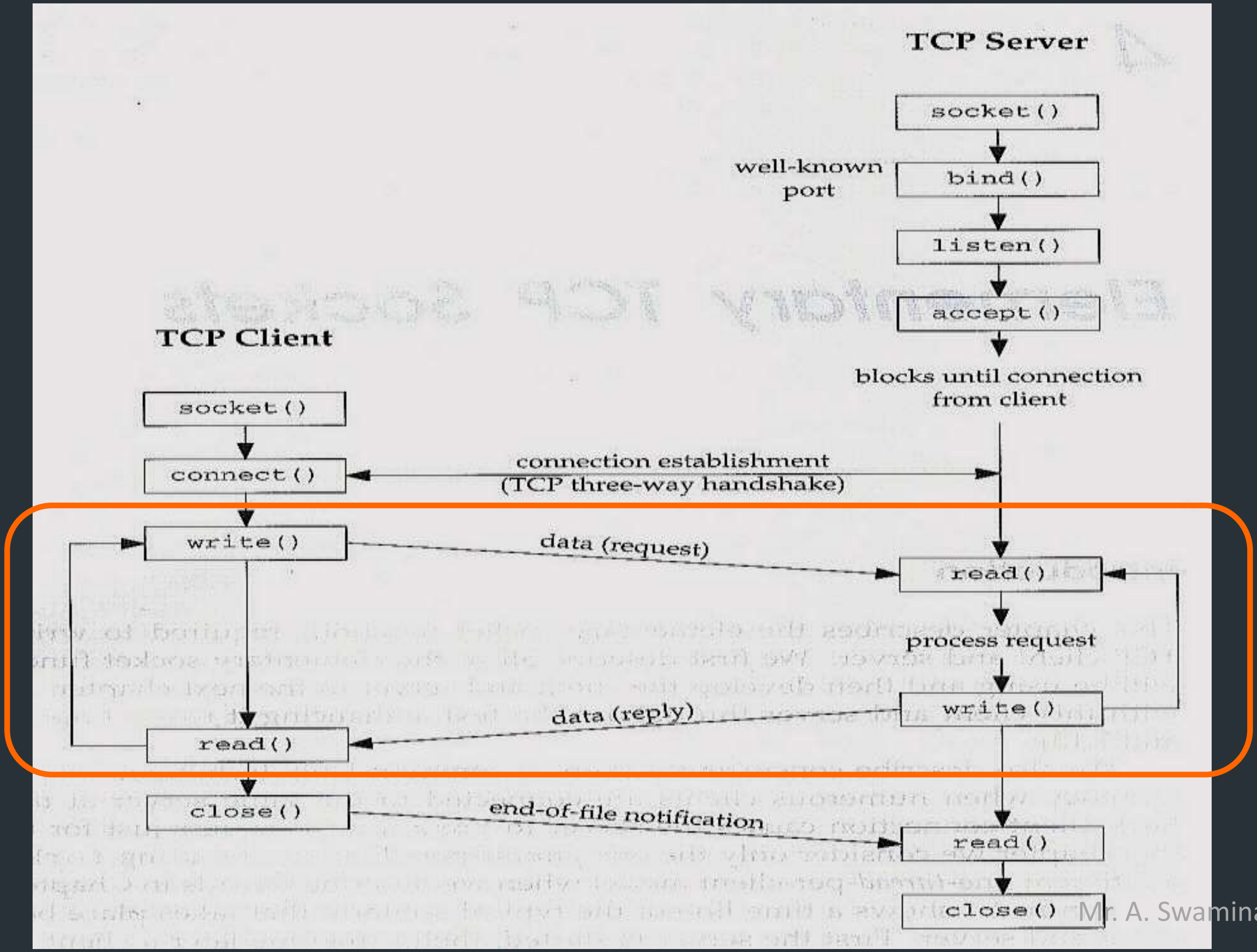
- status: return value, 0 if successful connect, -1 otherwise
- sock: client's socket to be used in connection
- servaddr: server's address structure
- addrlen: size (in bytes) of the servaddr structure

connect is blocking

- Example code:

```
if(connect(hSocket, (struct sockaddr*) &Address, sizeof(Address)) == -1)
    {
        printf("\nCould not connect to host\n");
    }
```

Sending/Receiving Data



```
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>
main()
{int sd,sd2,nsd,clilen,sport,len,port;
char sendmsg[20],rcvmsg[20];
struct sockaddr_in servaddr, cliaddr;
printf("enter server port\n");
scanf("%d",&sport);
printf("%d\n",sport);
sd=socket(AF_INET,SOCK_STREAM,0);
if(sd<0)
printf("cant create\n");
else
printf("Socket is created\n");
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);
servaddr.sin_port=htons(sport);
sd2=bind(sd,(struct sockaddr*)&servaddr,sizeof(servaddr));
if(sd2<0)
printf("cant bind");
else
printf("Binded\n");
listen(sd,5);
clilen=sizeof(cliaddr);
nsd=accept(sd,(struct sockaddr*)&cliaddr,&clilen);
if(nsd<0)
printf("cant accept");
else
printf("Accept");}
```

```
#include<stdio.h>
#include<sys/types.h>
#include<netinet/in.h>
#include<string.h>
main()
{
int csd,cport,len;
char sendmsg[20],rcvmsg[20];
struct sockaddr_in servaddr, cliaddr;
printf("Enter server port\n");
scanf("%d",&cport);
printf("%d\n",cport);
csd=socket(AF_INET,SOCK_STREAM,0);
if(csd<0)
printf("cant create\n");
else
printf("Socket is created\n");
servaddr.sin_family=AF_INET;
servaddr.sin_addr.s_addr=htonl(INADDR_ANY);//inet_addr("");
servaddr.sin_port=htons(cport);
if(connect(csd,(struct sockaddr*)&servaddr,sizeof(servaddr))<0)
printf("cant connect");
else
printf("Connected");
}
```

Sending / Receiving Data

- Send data
- int count = send(int s, const void * msg, int len, unsigned int falgs);

Where:

  - count: number of bytes transmitted (-1 if error)
  - sock: socket being used
  - buf: buffer to be transmitted
  - len: length of buffer (in bytes) to transmit
  - flags: special options, usually just 0

- Receive data
- int count = recv(int s, void *buf, int len, unsigned int flags);

Where:

  - count: number of bytes received (-1 if error)
  - sock: socket being used
  - buf: stores received bytes
  - len: number of bytes received
  - flags: special options, usually just 0

Example (Client/Server)

Mr. A. Swaminathan VIT Chennai

CLIENT

…

// write a message to the server

n = send(sock,buffer,strlen(buffer),0);

// do some processing

…

// read a message from the server

n = recv(sock,buffer,255,0);

SERVER

// after the client executed a write(), it will read

n = recv(newsock,buffer,255,0);

// do some processing

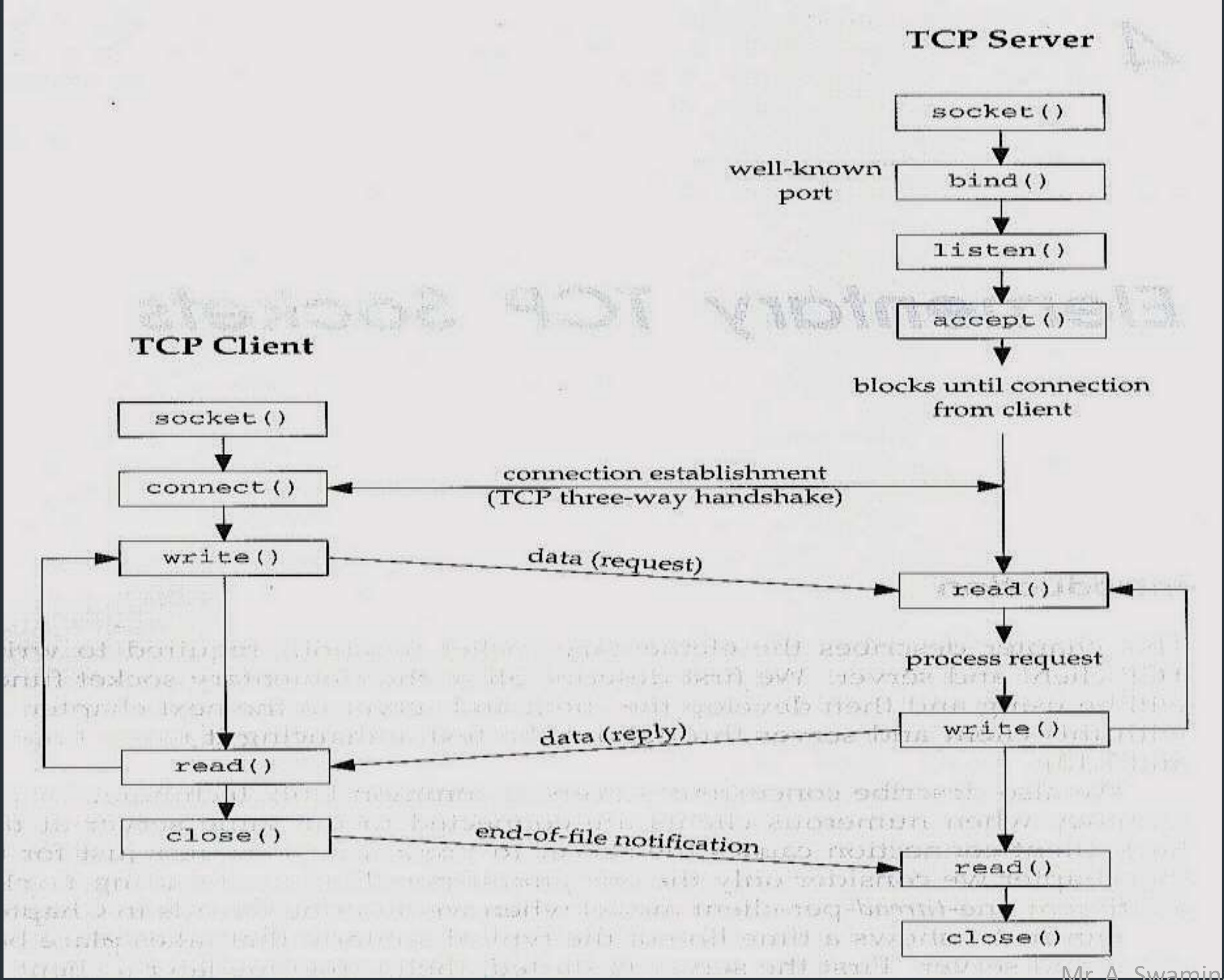
…

// send the result to the client

n = send(newsock,resp_msg,strlen(resp_msg),0);

close

- When finished using a socket, the socket should be closed:
- status = close(s);
  - status: return value, 0 if successful, -1 if error
  - s: the file descriptor (socket being closed)


TCP Server
socket()
well-known port
bind()
listen()
accept()
blocks until connection from client
TCP Client
socket()
connect()
connection establishment (TCP three-way handshake)
write()
data (request)
read()
process request
write()
data (reply)
read()
close()
end-of-file notification
read()
close()

References

- Forouzan Behrouz, A. "Data Communication and networking." (2008).
- Peterson, Larry L., and Bruce S. Davie. *Computer networks: a systems approach*. Elsevier, 2007.
- Stallings, William. *Data and computer communications*. Pearson Education India, 2007.
- TCP/IP Protocol Suite, Behrouz A. Forouzan, McGraw-Hill Education, 4 Ed., 2010
- Web Links as mentioned in source
- https://www.eecs.yorku.ca/course_archive/2015-16/W/3214/CSE3214_10_PacketDelay_2016_posted.pdf

Thank You

Queries?