COMPUTER NETWORKS MODULE 5.1

MS. JINCY J FERNANDEZ

ASST. PROF, CSE

RSET

MODULE 5

Transport Layer

INTRODUCTION

- Responsible for the delivery of a message from one process to another.
- A process is an application program running on a host.
- The transport layer header include port numbers
- Transport layer protocol can be connectionless or connection oriented
- A message is normally divided into transmittable segments.

TRANSPORT LAYER SERVICES

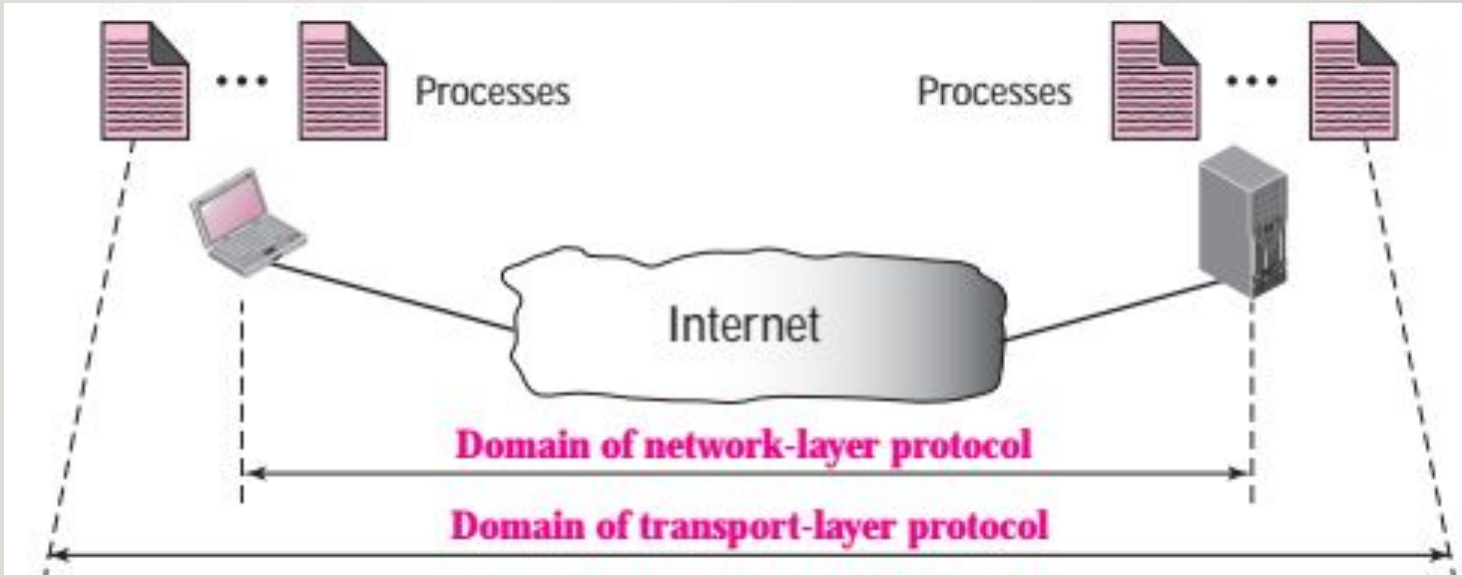
- Services provided to the upper layers.
- Ultimate goal of transport layer is to provide efficient, reliable and cost-effective services to its users which are processes running on the application layer.
- For this we have a transport entity. It is located in the OS kernel or is a separate user process or on the NIC.

TRANSPORT LAYER SERVICES

- Process-to-Process Communication
- Addressing: Port Numbers
- Encapsulation and Decapsulation
- Multiplexing and Demultiplexing
- Flow Control
- Error Control
- Congestion Control
- Connectionless and Connection-Oriented Services

PROCESS – TO – PROCESS COMMUNICATION

- Process is an application layer entity – running program.
- Transport layer is responsible for delivering the message to the appropriate process.

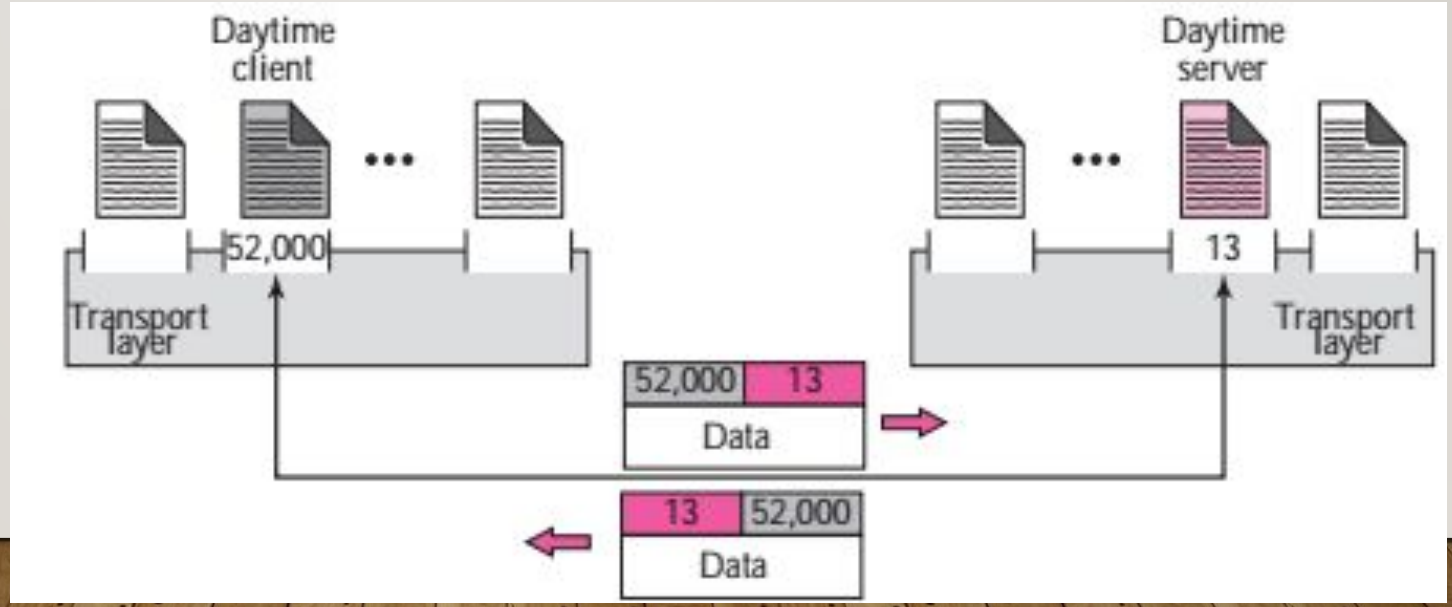


ADDRESSING: PORT NUMBERS

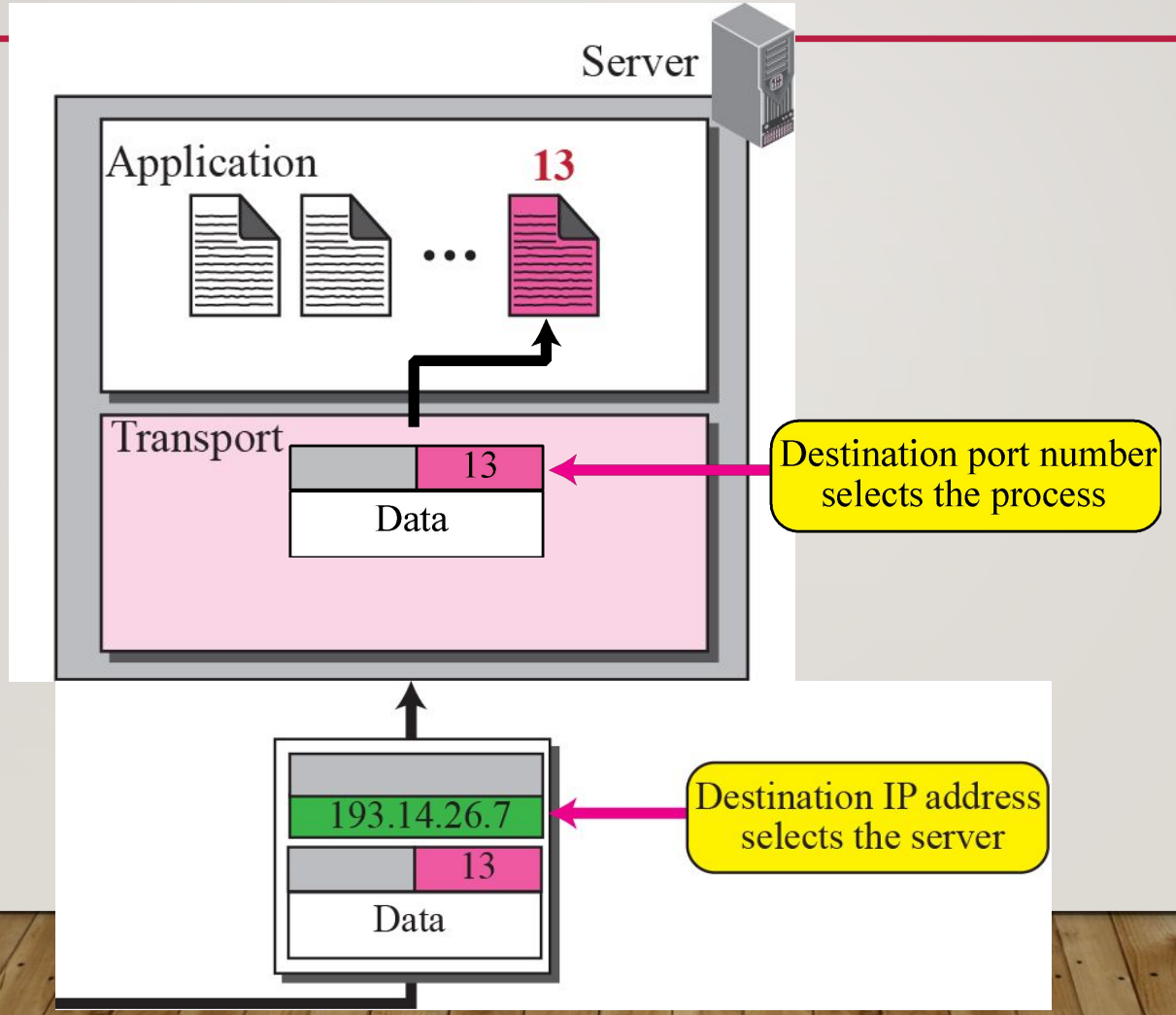
- Process-to-process communication usually follows the client-server paradigm.
- Hosts are identified by their IP addresses.
- Processes are defined by ports.
- Ports are identified by port numbers.
- In TCP/IP the port numbers are identifiers between 0 – 65,535.

ADDRESSING-PORT NUMBERS

- Clients use ephemeral port numbers: short lived.
- Servers use well-known port numbers.
- Every client process knows the well known port number of the corresponding server process.

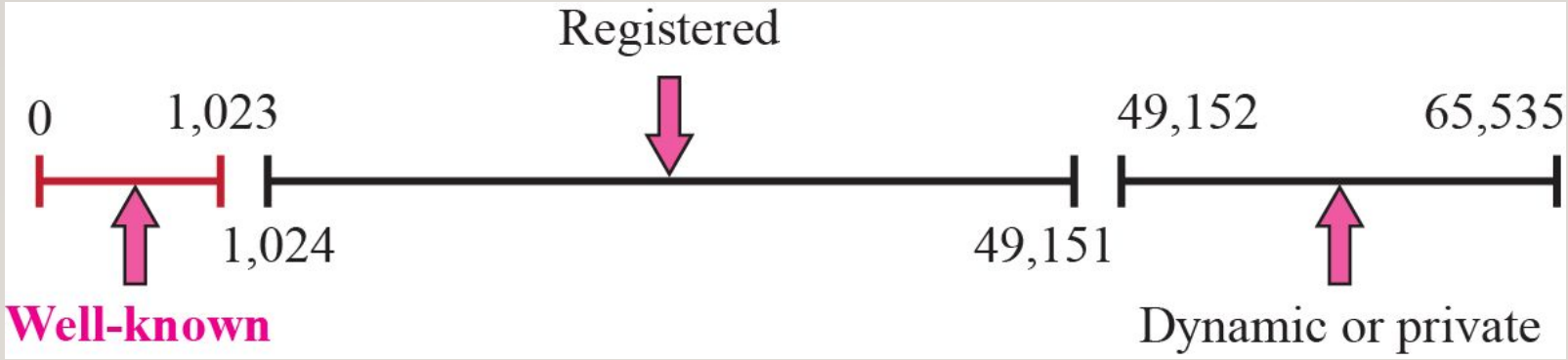


ADDRESSING- IP ADDRESSES VERSUS PORT NUMBERS



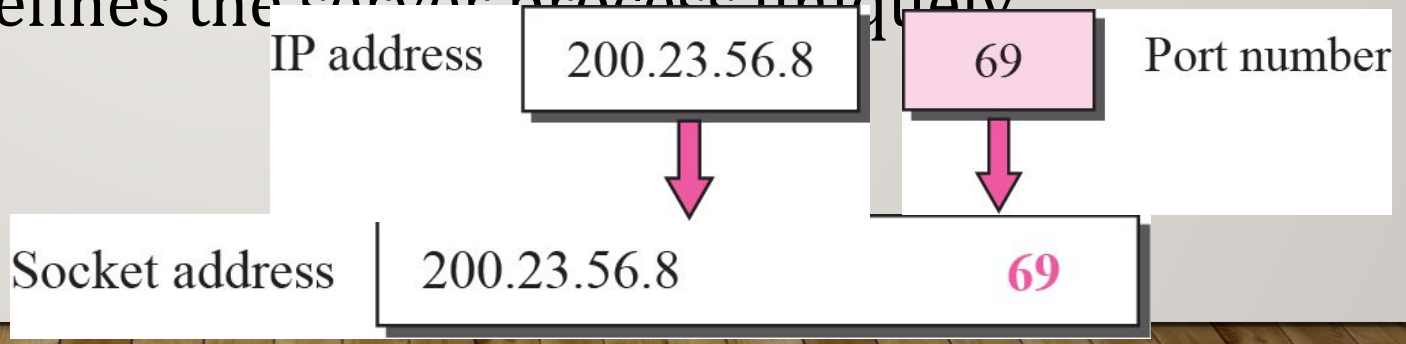
ADDRESSING- ICANN RANGES

- Well known ports; controlled by ICANN. Eg: DNS-53, HTTP-80 etc.
- Registered ports.: not controlled by ICANN, but registered with ICANN. Eg: used by companies.
- Dynamic ports: neither controlled or registered with ICANN; can be used as temporary or private port numbers. Eg: assigned to normal users.

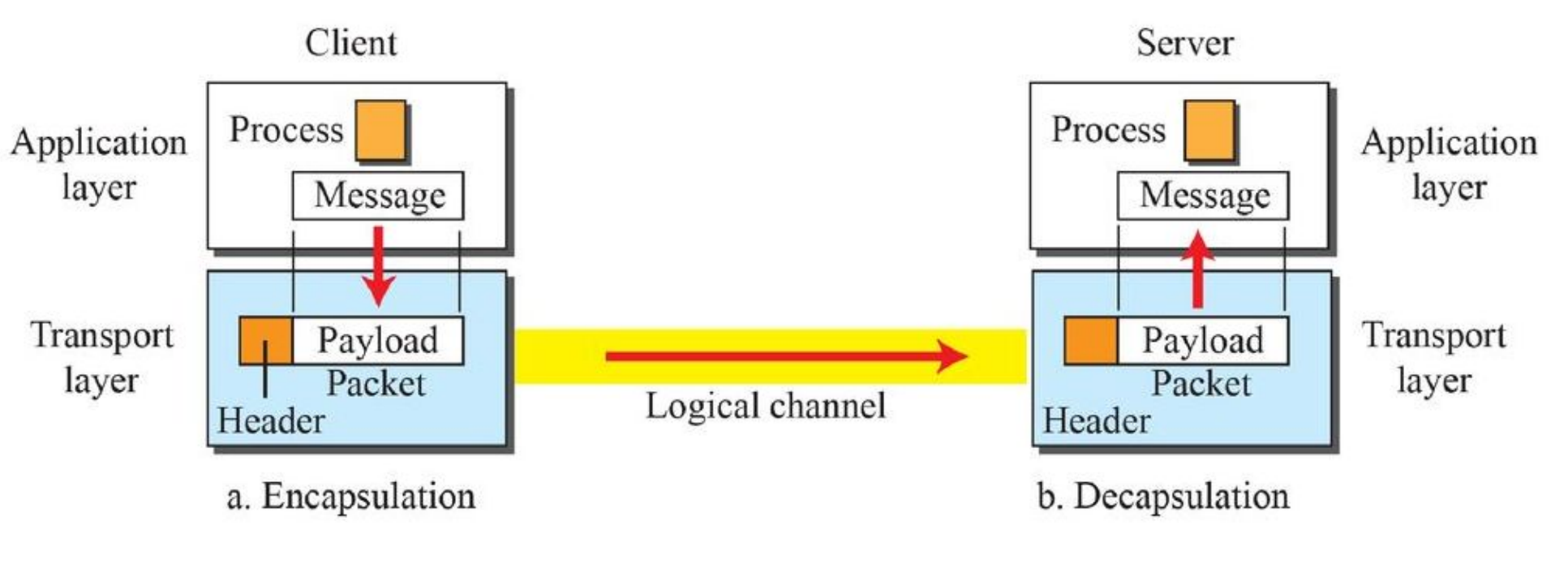


ADDRESSING- SOCKET ADDRESS

- Transport layer in TCP suite needs both IP address and the port number to make a connection.
- The combination of IP address and port number is called a socket address.
- The client socket address defines the client process uniquely.
- The server socket address defines the server process uniquely



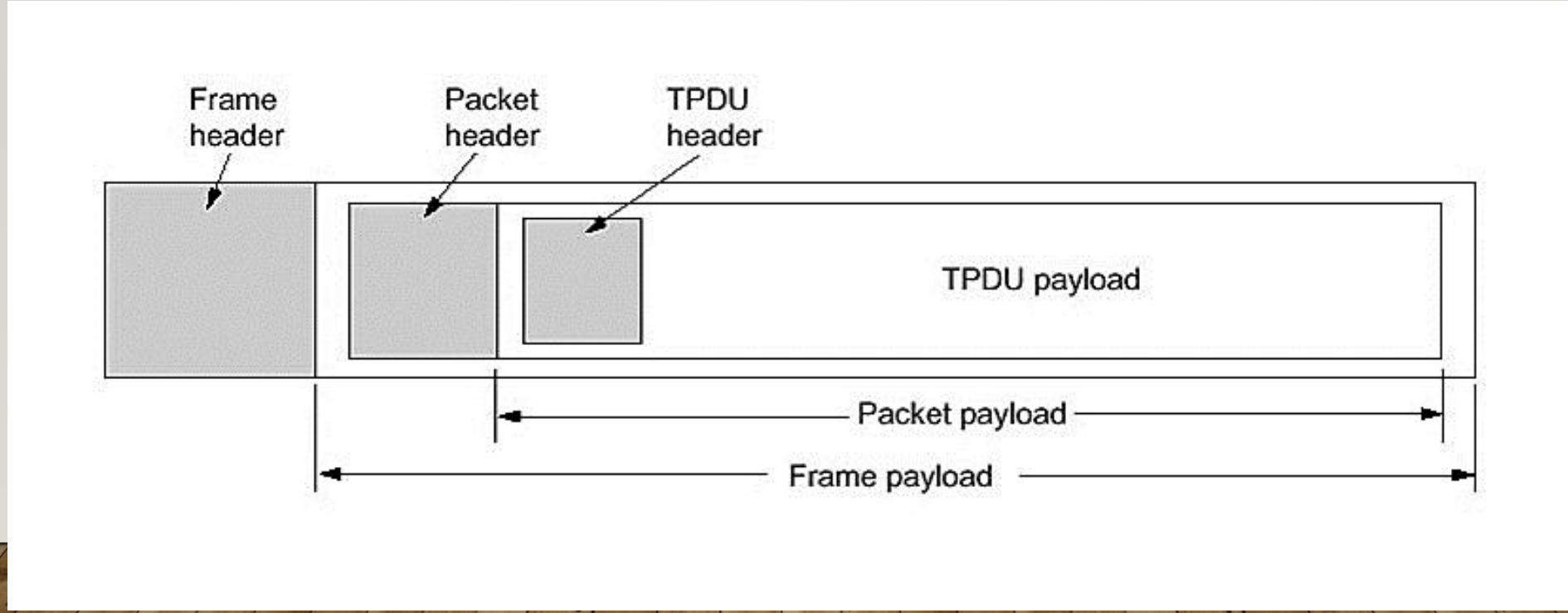
ENCAPSULATION AND DECAPSULATION



User datagram or Segment or Packet

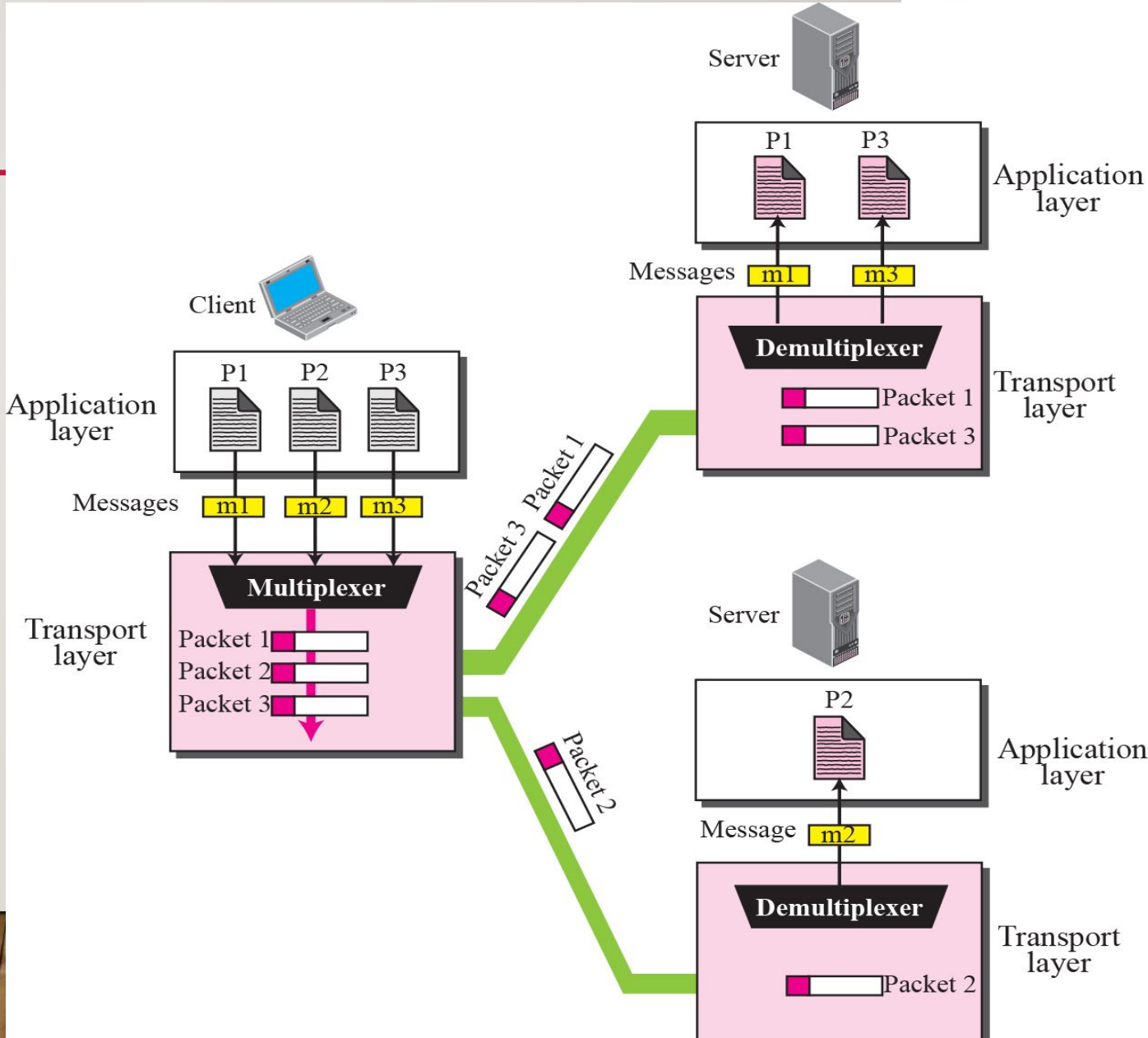
TPDU

- Transport protocol data unit- TPDU



MULTIPLEXING AND DEMULTIPLEXING

- When an entity accepts items from more than one source multiplexing.
- When an entity delivers items to more than one source demultiplexing.
- Transport layer at the source performs multiplexing; transport layer at the destination performs demultiplexing.



FLOW CONTROL

- When the producer produces items faster than what can be consumed by the consumer, the consumer will have to drop some packets.
- Pushing and Pulling.
- Pushing: If the sender delivers items whenever they are produced without a prior request from the consumer.
- Pulling: If the producer delivers the items after the consumer has requested them.

ERROR CONTROL

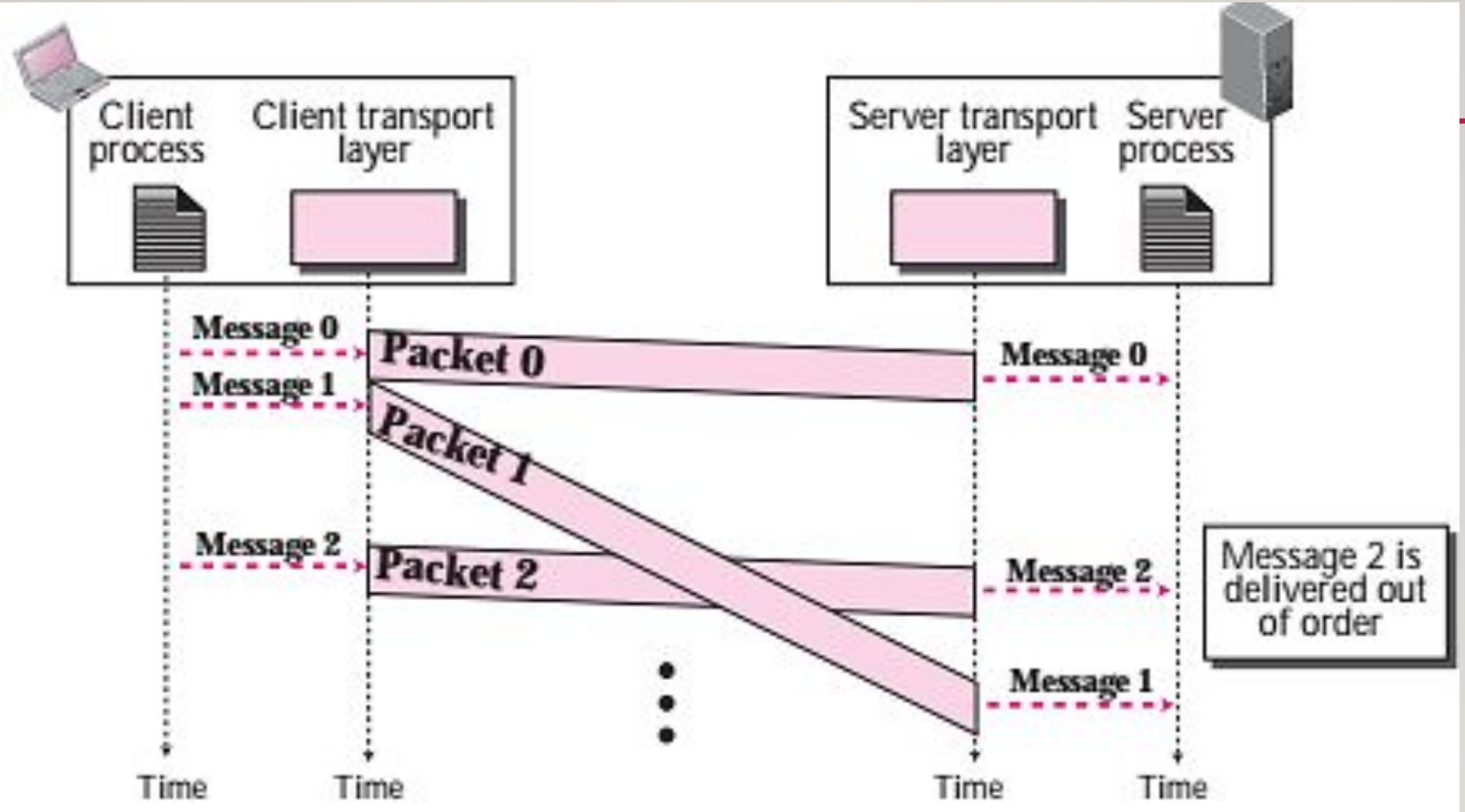
- In the internet, network layer is unreliable, so it is essential to make the transport layer reliable.
- Detect and discard corrupted packets
- Keep track of lost and discarded packets and resend them
- Recognize duplicate packets and discard them
- Buffering out-of-order packets until the missing packets arrive
  - Sequence numbers
  - Acknowledgements
  - Time-out

CONGESTION CONTROL

- Congestion occurs if the load on the network (the number of packets sent to the network) is greater than the *capacity* of the network (the number of packets a network can handle).
- Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity
- Open-loop congestion control
  - Policies are applied to prevent congestion before it happens
- Closed-loop congestion control
  - Try to alleviate congestion after it happens

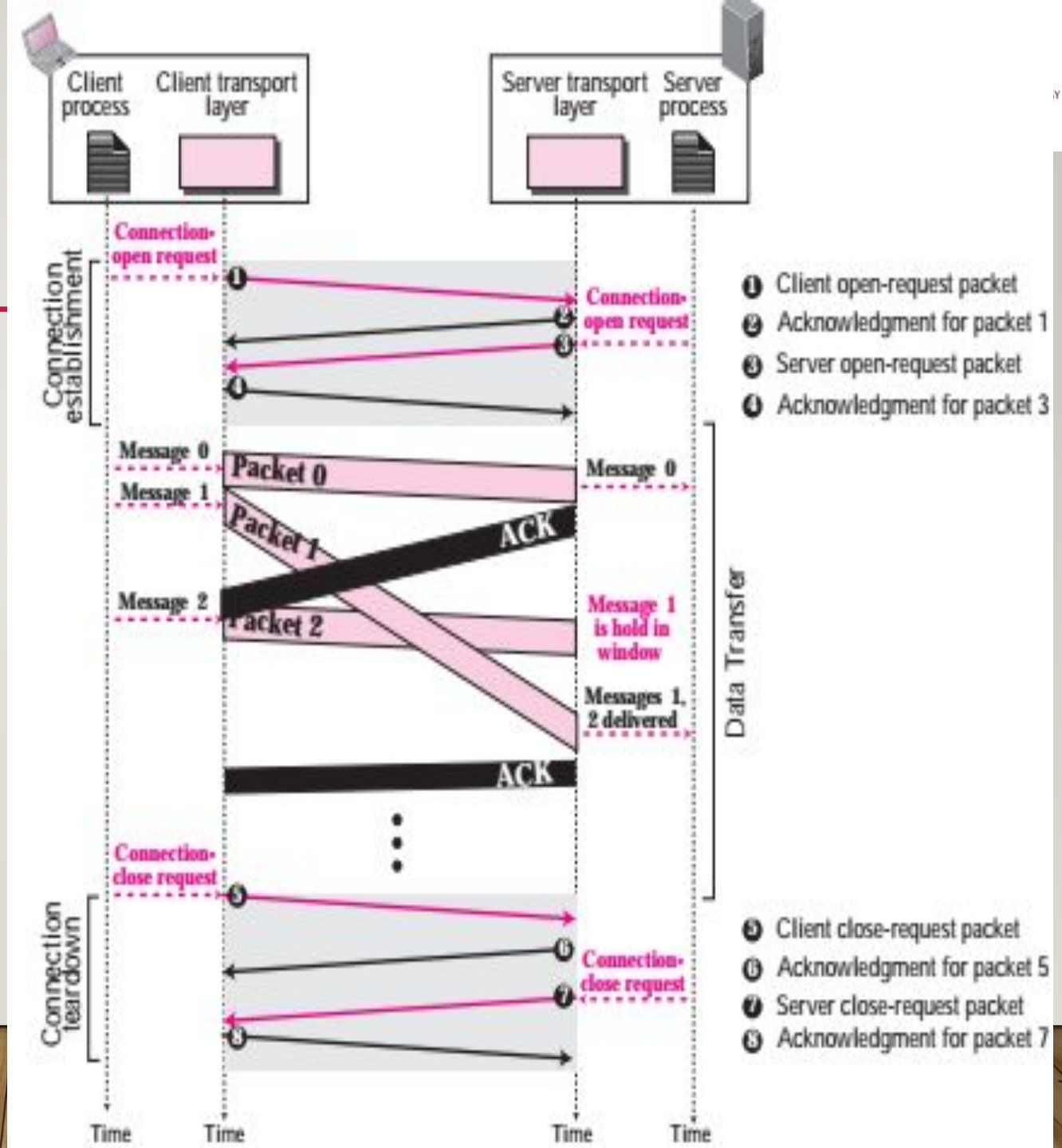
CONNECTIONLESS SERVICE

- The source process (application program) divides its message into chunks of data
- Chunks are delivered to the connectionless transport protocol one by one
- The transport layer treats each chunk as a single unit without any relation between the chunks
- No dependency between the packets at the transport layer
- The packets may arrive out of order at the destination and will be delivered out of order to the server process


Client process
Client transport layer
Server transport layer
Server process
Message 0
Packet 0
Message 0
Message 1
Packet 1
Message 2
Packet 2
Message 2
Message 2 is delivered out of order
Message 1
Time
Time
Time
Time

CONNECTION-ORIENTED SERVICE

- Connection established
- Data transferred
- Connection teared down



THANK YOU!!!

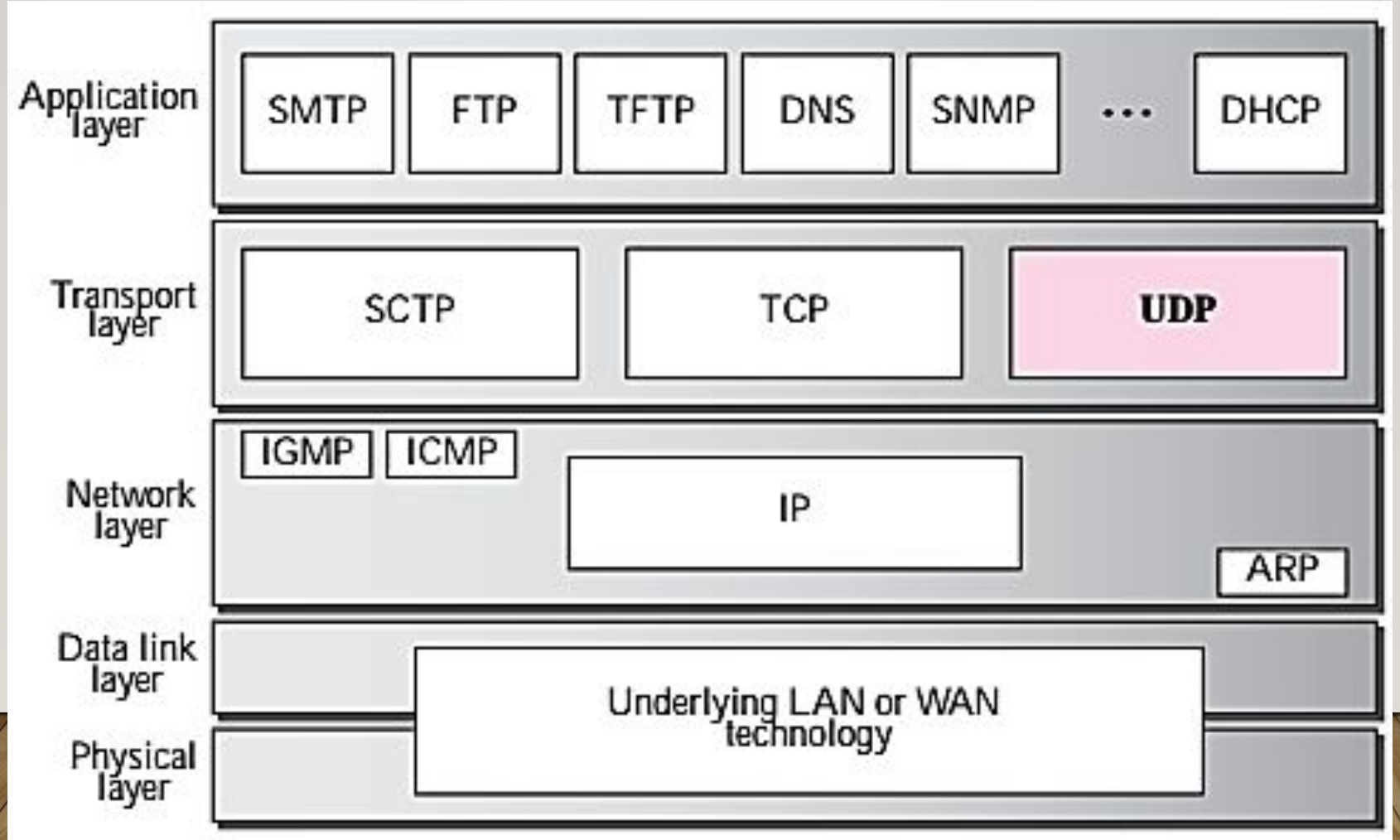
COMPUTER NETWORKS MODULE 5.2

MS. JINCY J FERNANDEZ

ASST. PROF, CSE

RSET

TRANSPORT LAYER PROTOCOLS



UDP

- User Datagram Protocol.
- UDP is a connectionless unreliable protocol in transport layer.
- Provides process-to-process communication using port numbers.
- Does not provide flow control, congestion control and does not use acknowledgements.
- Error control provided to some extent.
- If UDP detects an error in a received packet, it silently drops it.

WHY UDP?

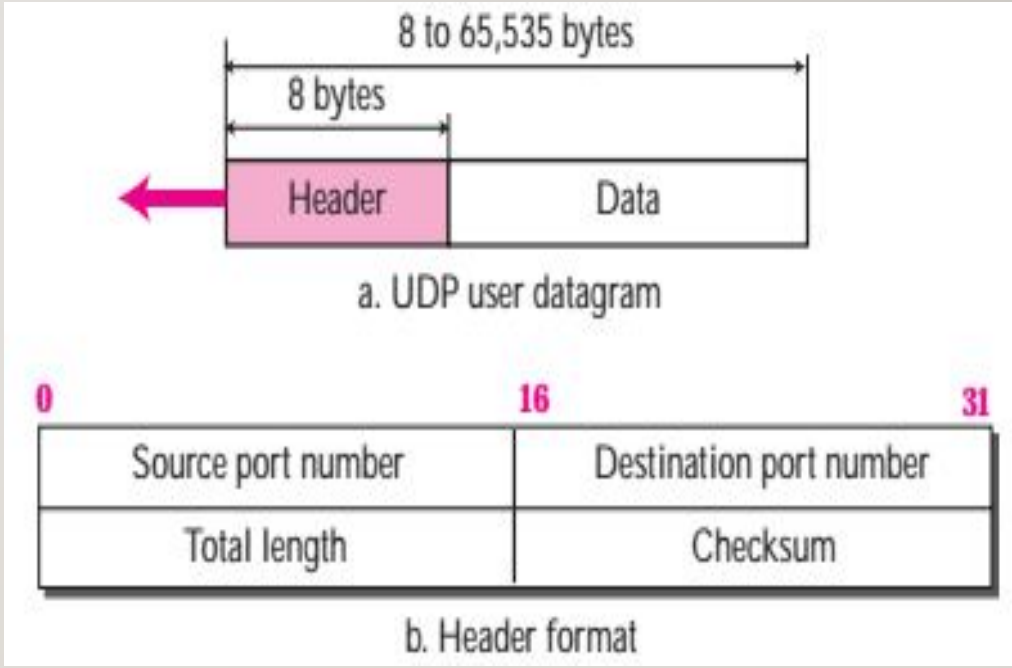
- Simple protocol.
- Minimum overhead.
- A process can use UDP to send a small message without giving importance to reliability.
- Sending small messages using UDP requires only less interaction between sender and receiver than using TCP.

WELL KNOWN PORTS USED WITH UDP AND TCP

Port	*Protocol*	*UDP*	*TCP*	*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, 21	FTP		√	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, 162	SNMP		√	Simple Network Management Protocol

USER DATAGRAM

- UDP packets are called user datagram.
- It has a fixed size header of 8 bytes.
- Source port number- used by process running on source host.
- Destination port number- used by process running on destination host.
- Total length- defines the total length of the user datagram header plus data.
- Checksum- used to detect errors over the entire user datagram.



USER DATAGRAM

- Qn. Let the content of a UDP header in hexadecimal format is 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?

UDP SERVICES

1. Process-to-process Communication
   - Done using sockets – combination of IP addresses & port numbers
2. Connectionless Service
   - User datagram sent by UDP should be an independent datagram.
   - User datagrams are not numbered.
   - No connection establishment and no connection termination.
   - Each request must be small enough to fit into one user datagram.
   - Only 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.

UDP SERVICES

3. No Flow Control

- No window mechanisms.
- Receiver may overflow with incoming packets.

4. No Congestion Control

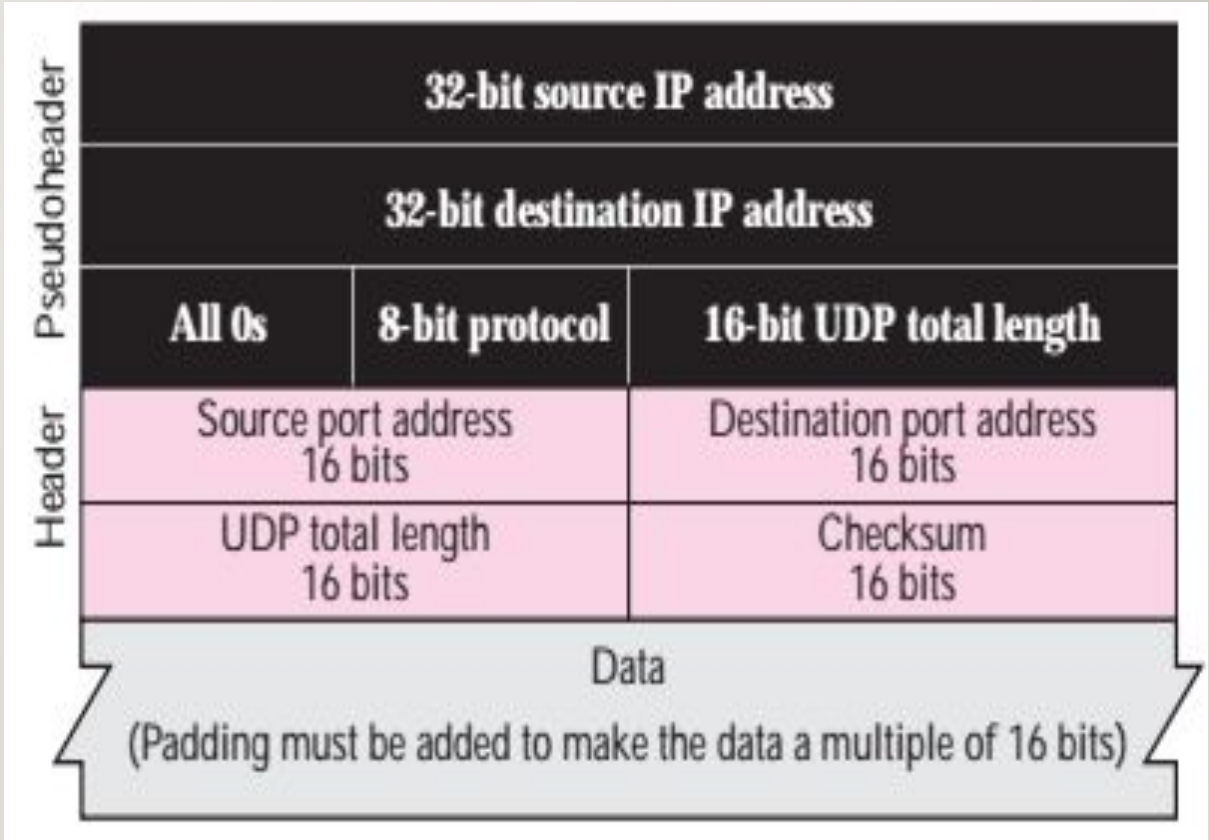
- UDP assumes that packets sent are small and sporadic and won't create any congestion.

5. Error Control – only through checksum

- Sender does not know if a message has been lost or duplicated.
- When the receiver detects an error through the checksum, the user datagram is silently discarded.

ERROR CONTROL-CHECKSUM

- UDP checksum calculation includes three sections: a pseudo header, the UDP header, and the data coming from the application layer.
- To calculate the checksum a pseudoheader (part of the IP header) is prepended to actual header.
- Calculate checksum on data, header & pseudoheader and insert.
- The sender of a UDP packet can choose not to calculate the checksum. Then insert all 0s.



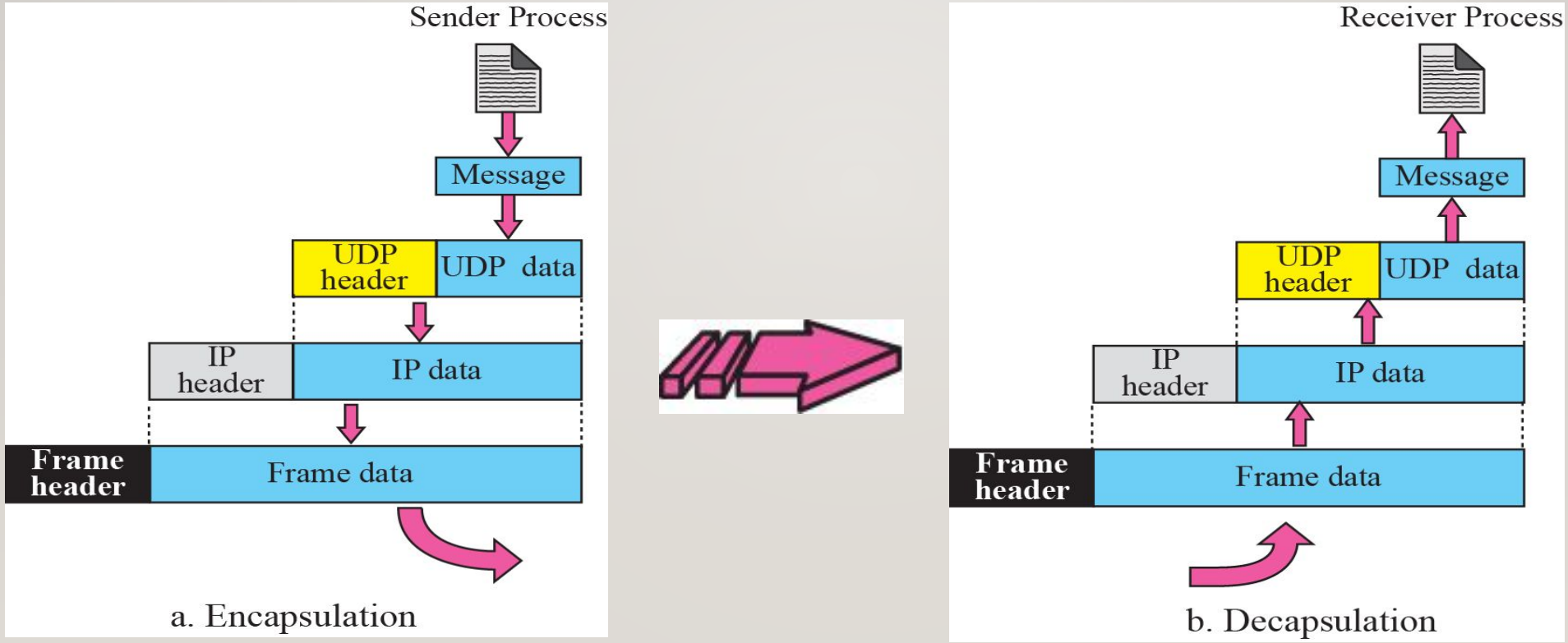
CHECKSUM CALCULATION OF A UDP DATAGRAM

153.18.8.105			
171.2.14.10			
All 0s	17	15	
1087		13	
15		All 0s	
T	E	S	T
I	N	G	Pad

10011001	00010010	153.18
00001000	01101001	8.105
10101011	00000010	171.2
00001110	00001010	14.10
00000000	00010001	0 and 17
00000000	00001111	15
00000100	00111111	1087
00000000	00001101	13
00000000	00001111	15
00000000	00000000	0 (checksum)
01010100	01000101	T and E
01010011	01010100	S and T
01001001	01001110	I and N
01000111	00000000	G and 0 (padding)
10010110	**11101011**	Sum
01101001	**00010100**	Checksum

UDP SERVICES

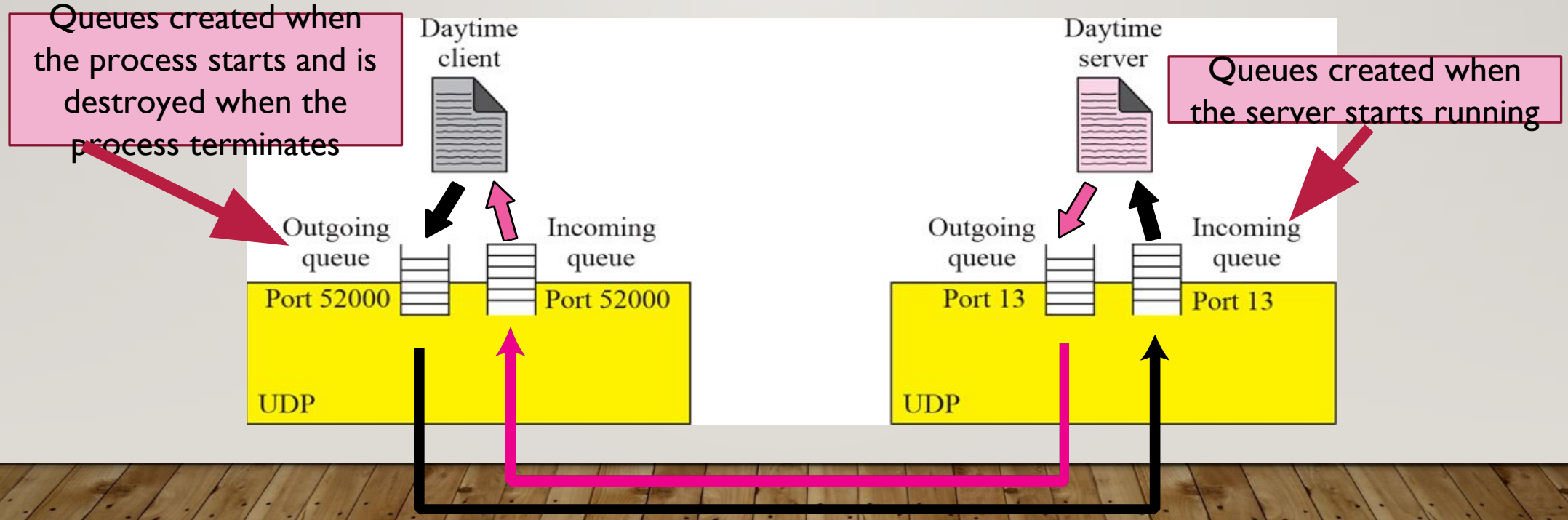
6. Encapsulation and Decapsulation



UDP SERVICES

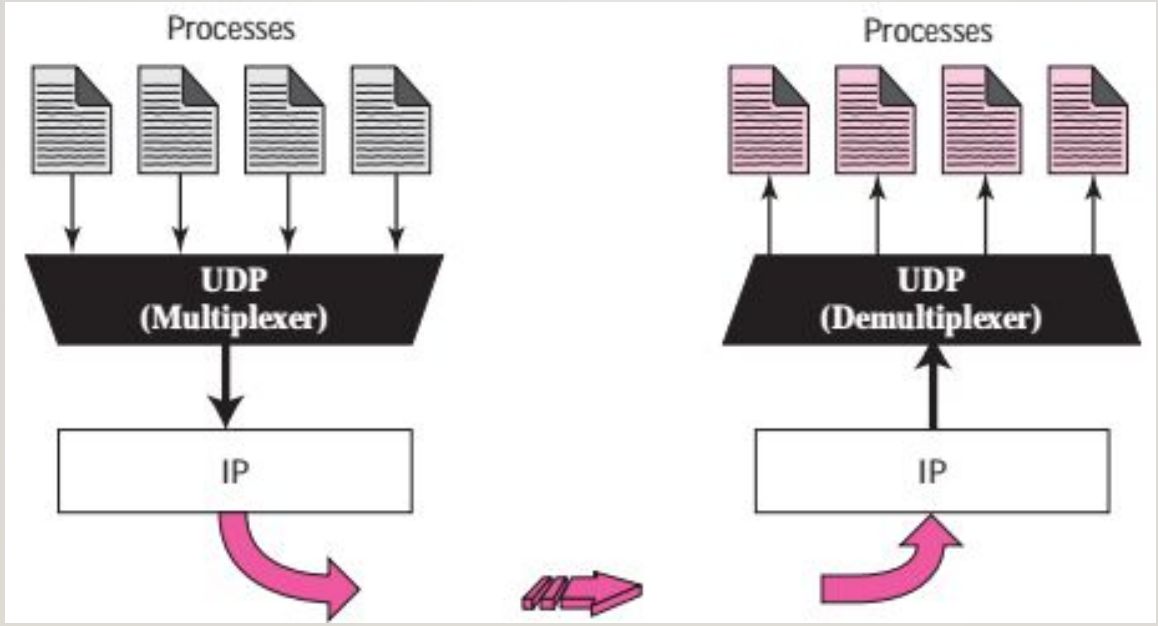
7. Queuing

- Queues are associated with ports.



UDP SERVICES

8. Multiplexing and Demultiplexing



APPLICATIONS

- Suitable for multicasting applications.
- Used for management processes such as SNMP.
- Used for route updating protocols like RIP.
- Used for real time applications.
- Used in applications with internal flow and error control mechanisms like TFTP.

TCP

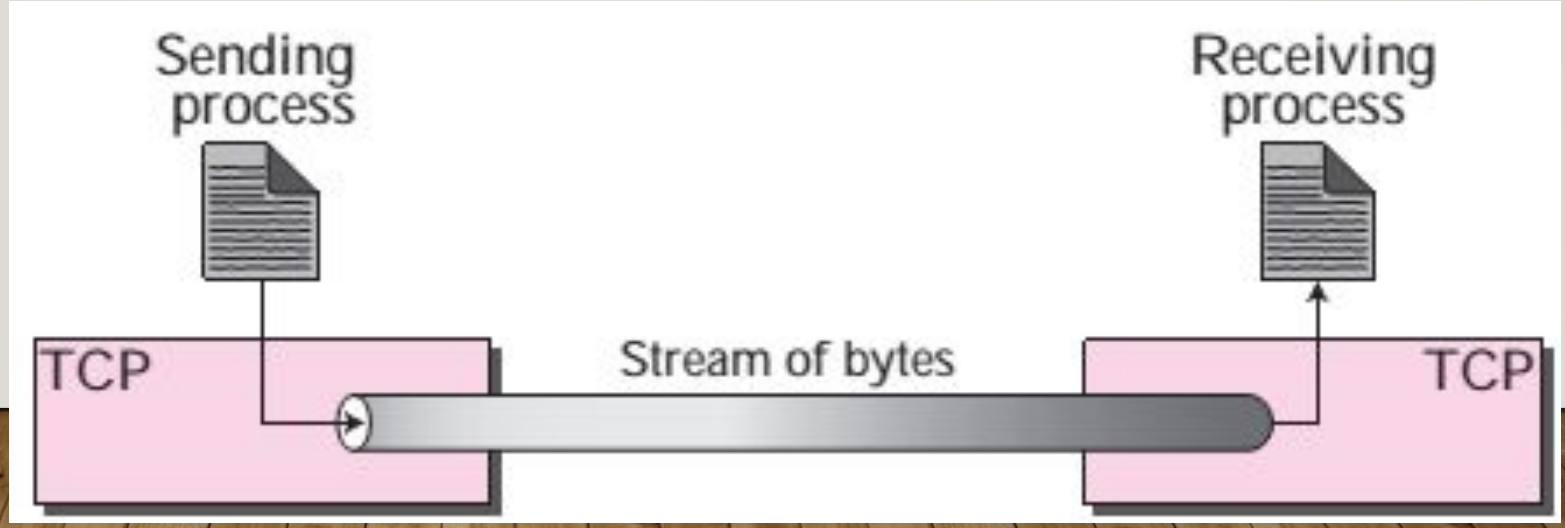
- Transmission Control Protocol.
- Connection oriented reliable protocol of transport layer.
- Ensures stream delivery service.
- Provides all most all transport layer services.

TCP SERVICES

1. Process-to-process Communication
   - Done using sockets – combination of IP addresses & port numbers

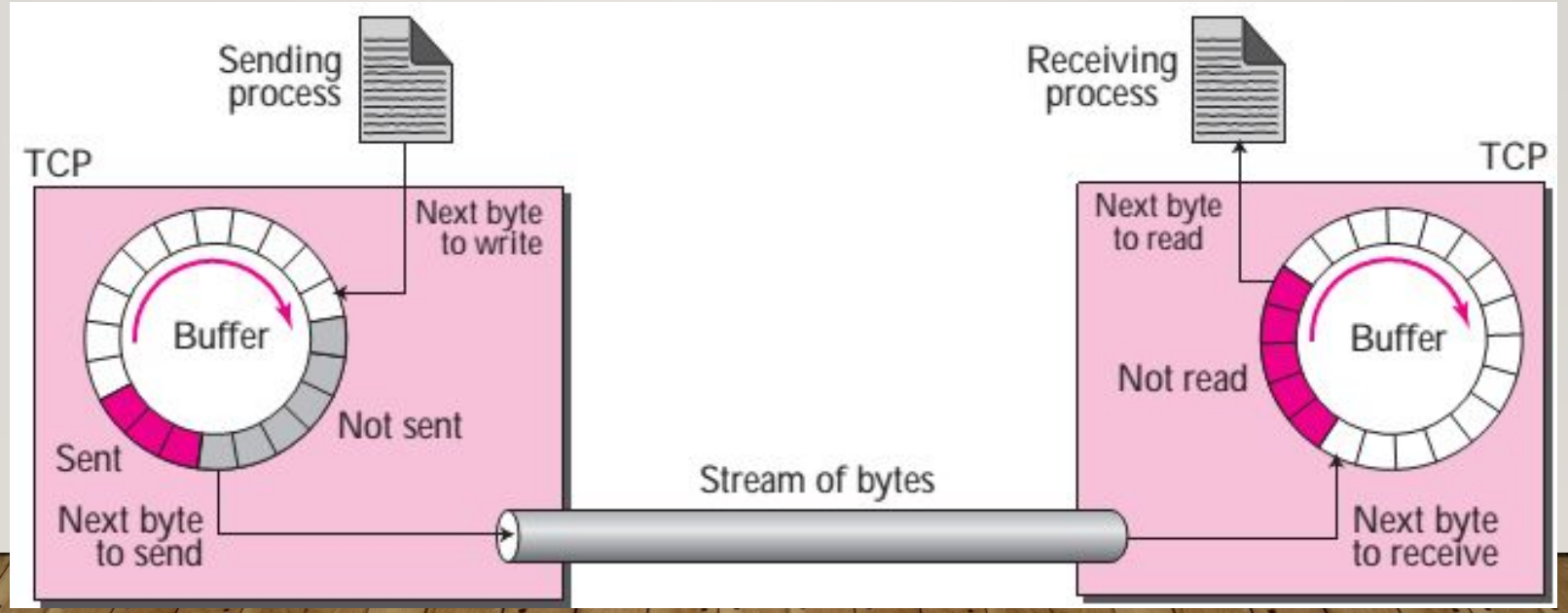
TCP SERVICES

2. Stream Delivery Service:
   - UDP had messages with pre-defined boundaries.
   - TCP is a stream-oriented protocol.
   - TCP 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.



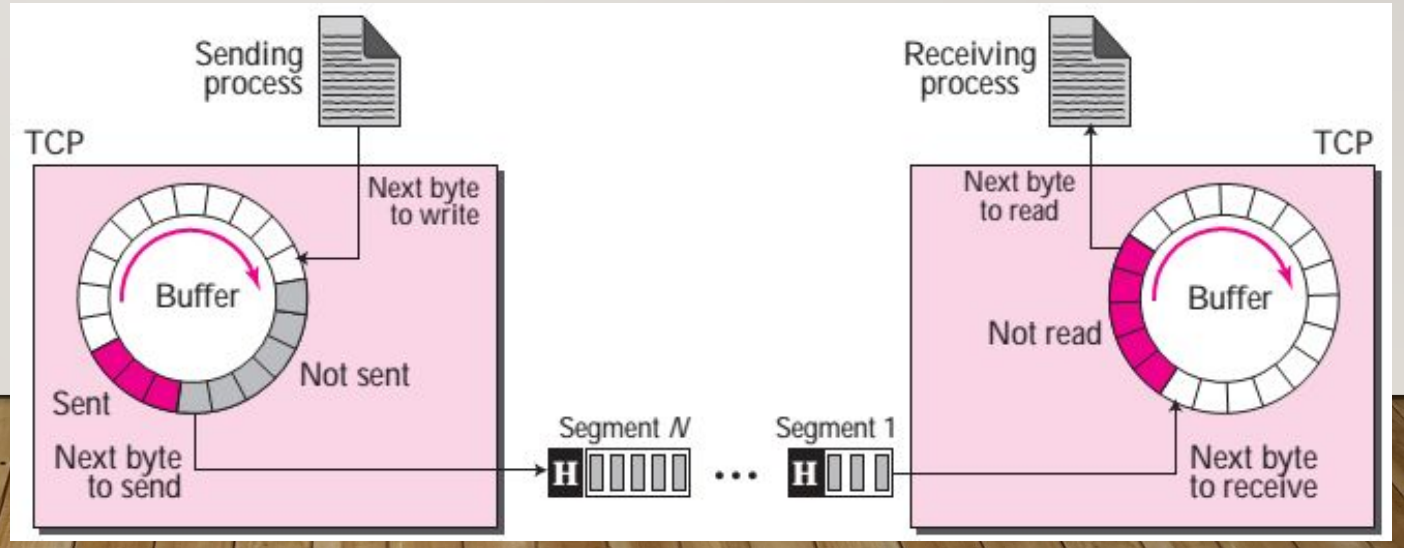
TCP SERVICES

3. Sending and Receiving Buffers:
   - Since the sending and receiving process may not operate at the same rate, buffers are used for storage.



TCP SERVICES

4. Segments:
   - TCP groups bytes together into packets called segments.
   - TCP adds a header to each segment and delivers the segment to the network layer for transmission.
   - Segments are encapsulated in an IP datagram and transmitted.



TCP SERVICES

5. Full Duplex Communication
   - Segments move in both directions.
6. Connection Oriented Service
   - TCPs at the sender and the receiver establishes a virtual connection between them before sending the data and terminates the connection after sending the data.
7. Multiplexing and Demultiplexing
   - Sender performs multiplexing and receiver performs demultiplexing.
8. Reliable Service
   - Acknowledgements used to ensure safe arrival of data.

TCP FEATURES

- Numbering system
  - Byte Number
  - Sequence Number
  - Acknowledgement Number

TCP FEATURES

- Byte Number
  - TCP numbers all data bytes that are transmitted in a connection.
  - When TCP receives bytes of data from a process, TCP stores them in the sending buffer and numbers them.
  - TCP chooses an arbitrary number between 0 and $2^{32} - 1$ for the number of the first byte.
  - E.g. if byte number = 1057 and total data to be sent is 6000 bytes, then the bytes are numbered from 1057 to 7056.
  - Used for flow control and error control.

TCP FEATURES

- Sequence Number
  - TCP assigns a sequence number to each segment.
  - The sequence number for each segment is the number of the first byte of data carried in that segment.

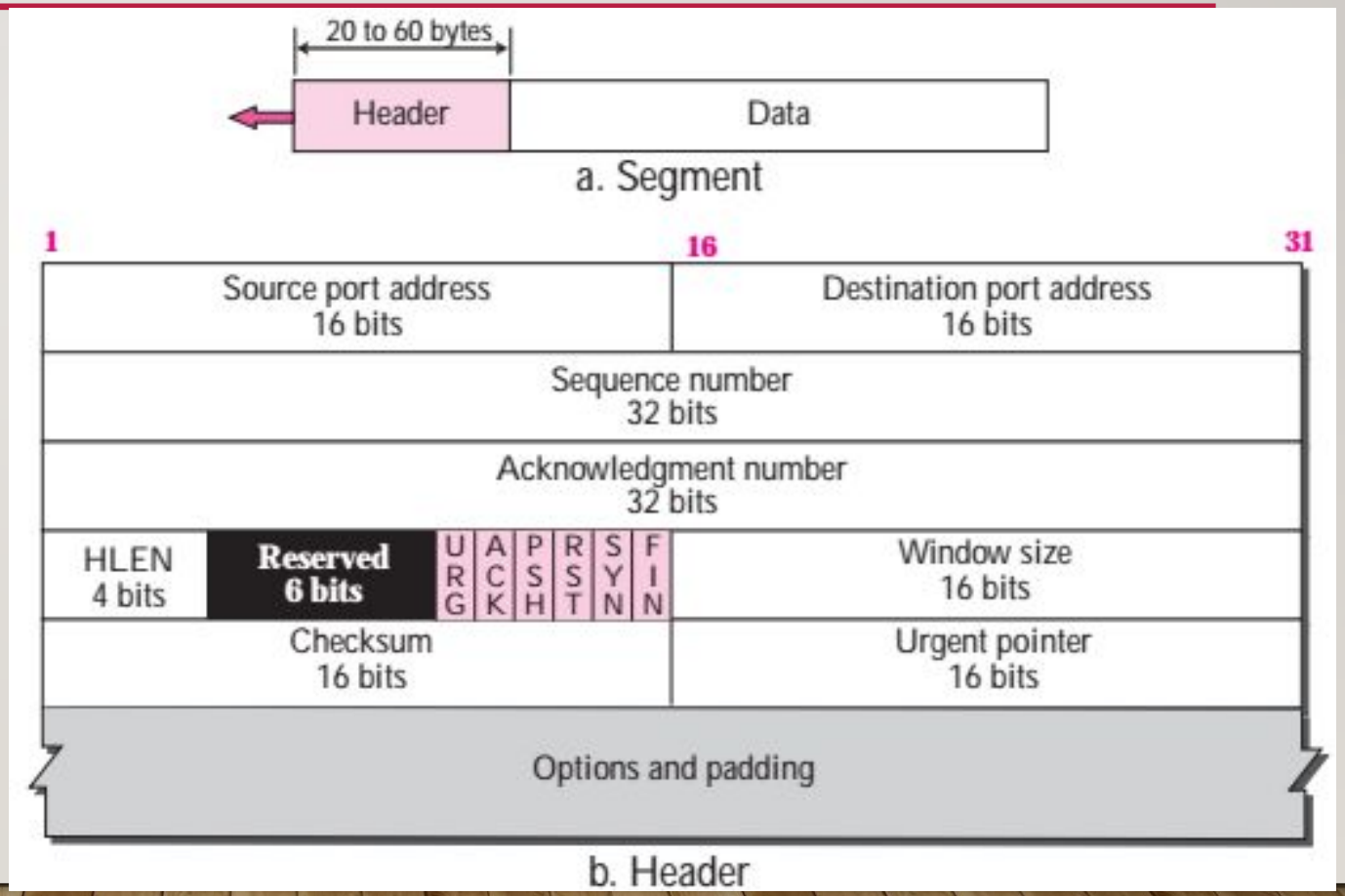
Segment 1	→	**Sequence Number:**	**10,001**	**Range:**	**10,001**	to	**11,000**
Segment 2	→	**Sequence Number:**	**11,001**	**Range:**	**11,001**	to	**12,000**
Segment 3	→	**Sequence Number:**	**12,001**	**Range:**	**12,001**	to	**13,000**
Segment 4	→	**Sequence Number:**	**13,001**	**Range:**	**13,001**	to	**14,000**
Segment 5	→	**Sequence Number:**	**14,001**	**Range:**	**14,001**	to	**15,000**

TCP FEATURES

- Acknowledgment number
  - Acknowledgment number is used by a party to confirm the bytes it has received.
  - The acknowledgment number defines the number of the next byte that the party expects to receive.
  - The acknowledgment number is cumulative.
  - If a party uses 5,641 as an acknowledgment number, it has received all bytes from the beginning up to 5,640.

TCP SEGMENT FORMAT

- A packet in TCP is called a segment.
- Consists of header and data.



FIELDS IN THE SEGMENT HEADER FORMAT

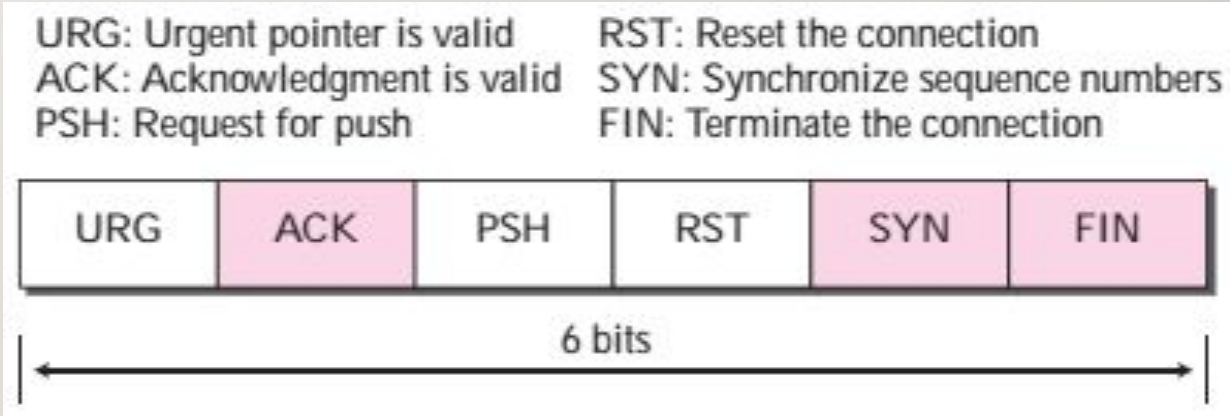
- Source port number
  - 16-bit field that defines the port number of the host that is sending the segment.
- Destination port number
  - 16-bit field that defines the port number of the host that is receiving the segment.
- Sequence number
  - 32-bit field defines the number assigned to the first byte of data contained in this segment.
  - During connection establishment each party uses a random number generator to create an initial sequence number (ISN).The ISN is usually different in each direction.

FIELDS IN THE SEGMENT HEADER FORMAT

- Acknowledgement number
  - 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party.
  - If the receiver has successfully received byte number *x*, it returns *x* + 1 as the acknowledgment number .
  - Acknowledgment and data can be piggybacked together.
- Header length
  - 4-bit field indicates the number of 4-byte words in the TCP header.
  - The length of the header can be between 20 and 60 bytes.
- Reserved
  - Reserved for future use.

FLAGS

- URG- it is set to 1 if urgent pointer is in use.
- ACK- it is set to 1 to indicate that the segment contains an acknowledgement.
- PSH- it is set to 1 to indicate pushed data.
- RST- it is set to 1 to indicate a connection is being reset, rejected or refused.
- SYN- it is set to 1 to indicate that a new connection is to be established.
- FIN- it is set to 1 to indicate that a connection is been released.



FIELDS IN THE SEGMENT HEADER FORMAT

- Window size
  - Defines the window size of the sending TCP in bytes.
  - The length of this field is 16 bits, so the maximum size of the window is 65,535 bytes.
  - This value is referred to as the receiving window and is determined by the receiver.
- Urgent pointer
  - 16-bit field, which is valid only if the urgent flag is set. It is used when the segment contains urgent data.
  - It defines a value that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment.

FIELDS IN THE SEGMENT HEADER FORMAT

- Checksum
  - Calculation is same as that in UDP.
- Options
  - There can be up to 40 bytes of optional information.

PHASES IN A TCP CONNECTION

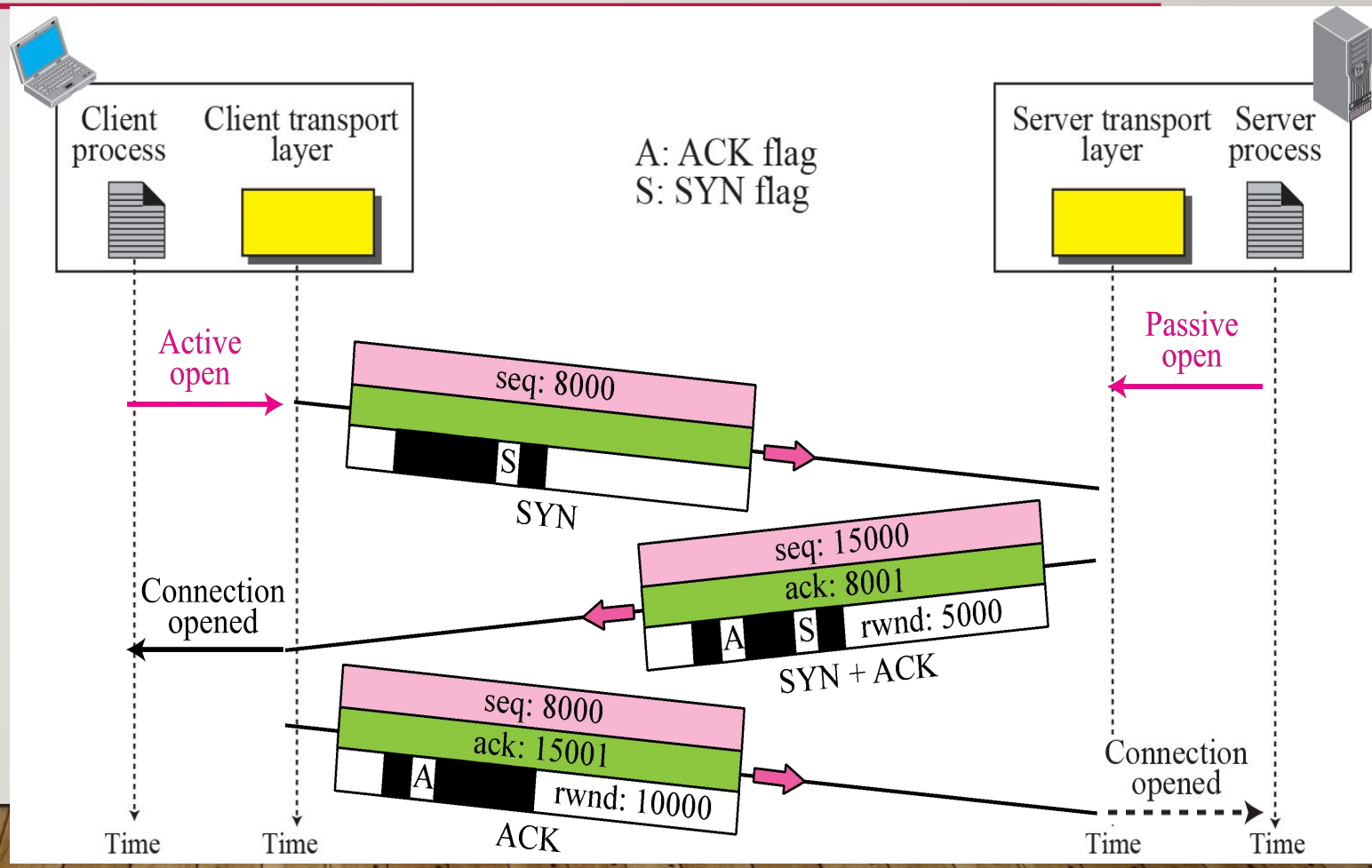
- TCP is a connection oriented protocol.
- It establishes a virtual path between the source and the destination.
- All the segments are sent along this virtual path.
- Transmission is done in three phases:
  - Connection Establishment
  - Data Transfer
  - Connection Termination

CONNECTION ESTABLISHMENT

- TCP transmits data in full-duplex mode.
- Each party must initialize communication and get approval from the other party before data is transferred.
- Connection establishment in TCP is called three-way handshaking.

THREE-WAY HANDSHAKING

- Step 1:
  - Client sends SYN segment for synchronization of sequence numbers.
  - Random number as the first sequence number (ISN) and send it to server.
  - This segment does not contain any acknowledgement number.
  - A SYN segment cannot carry data, but it consumes one sequence number.



THREE-WAY HANDSHAKING

- Step 2:
  - Server sends the second segment, a SYN+ACK segment with two flag bits set.
  - Uses this segment to initialize a sequence number for numbering the byte sent from the server to the client.
  - Server also acknowledge the receipt of SYN segment from the client by setting the ACK flag.
  - Needs to define the receive window, rwnd.
  - A SYN+ACK segment cannot carry data.

THREE-WAY HANDSHAKING

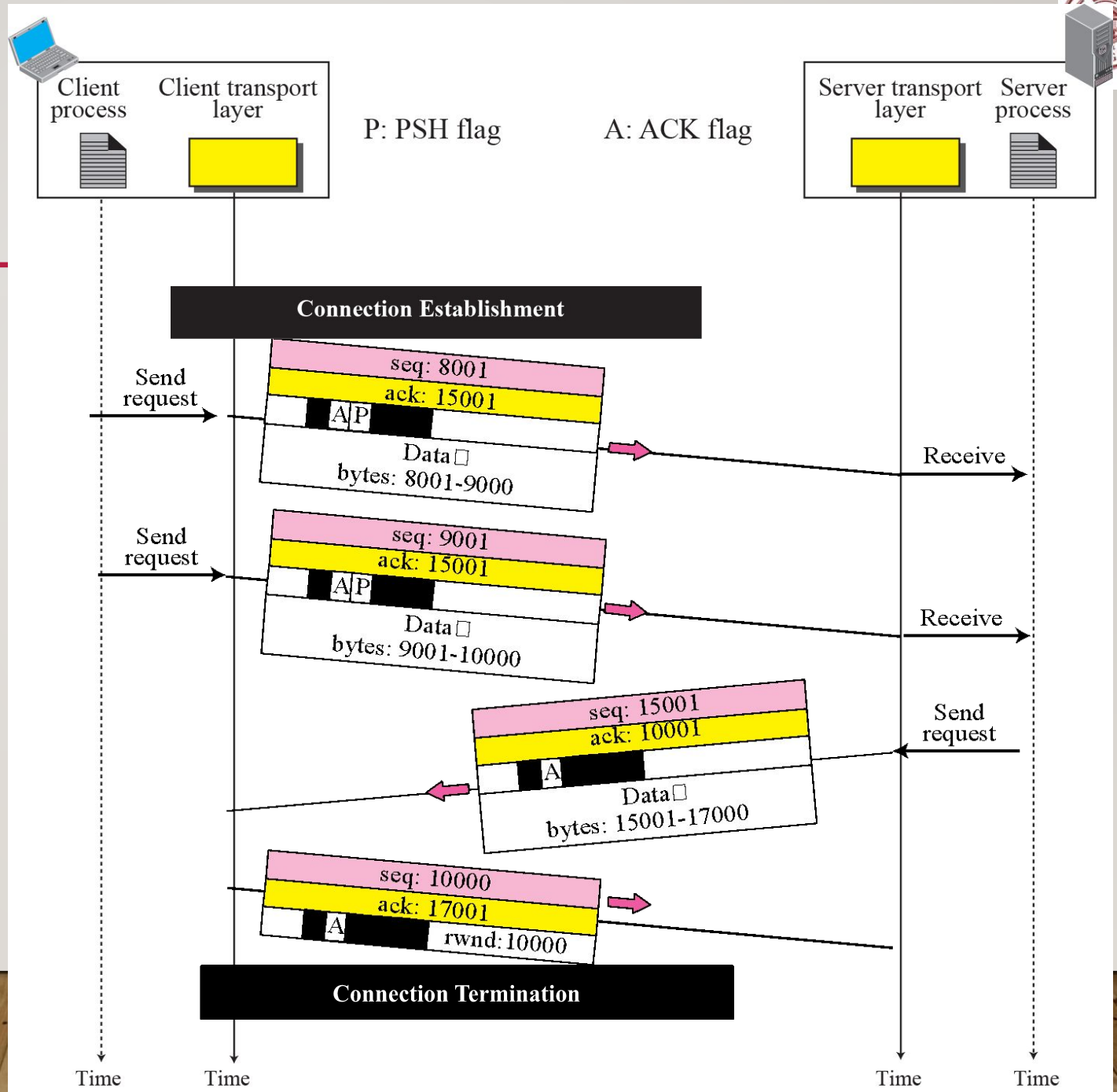
- Step 3:
  - Client sends the third segment an ACK segment.
  - An ACK segment does not consume any sequence number if it does not carry data.
  - Some implementations allow this segment to carry first chunk of data from the client.

THREE-WAY HANDSHAKING- SYN FLOODING ATTACK

- Connection establishment procedure in TCP is susceptible to a serious security problem called SYN flooding attack.
- When one or more malicious attackers send a large number of SYN segments to a server pretending that each of them is coming from a different client by faking the source IP addresses in the datagrams.
- The server, assuming that the clients are issuing an active open, allocates the necessary resources.
- TCP server then sends the SYN+ACK segments to the fake clients, which are lost.
- Server eventually runs out of resources and may be unable to accept connection requests from valid clients.

DATA TRANSFER

- After connection is established, bidirectional data transfer can take place.



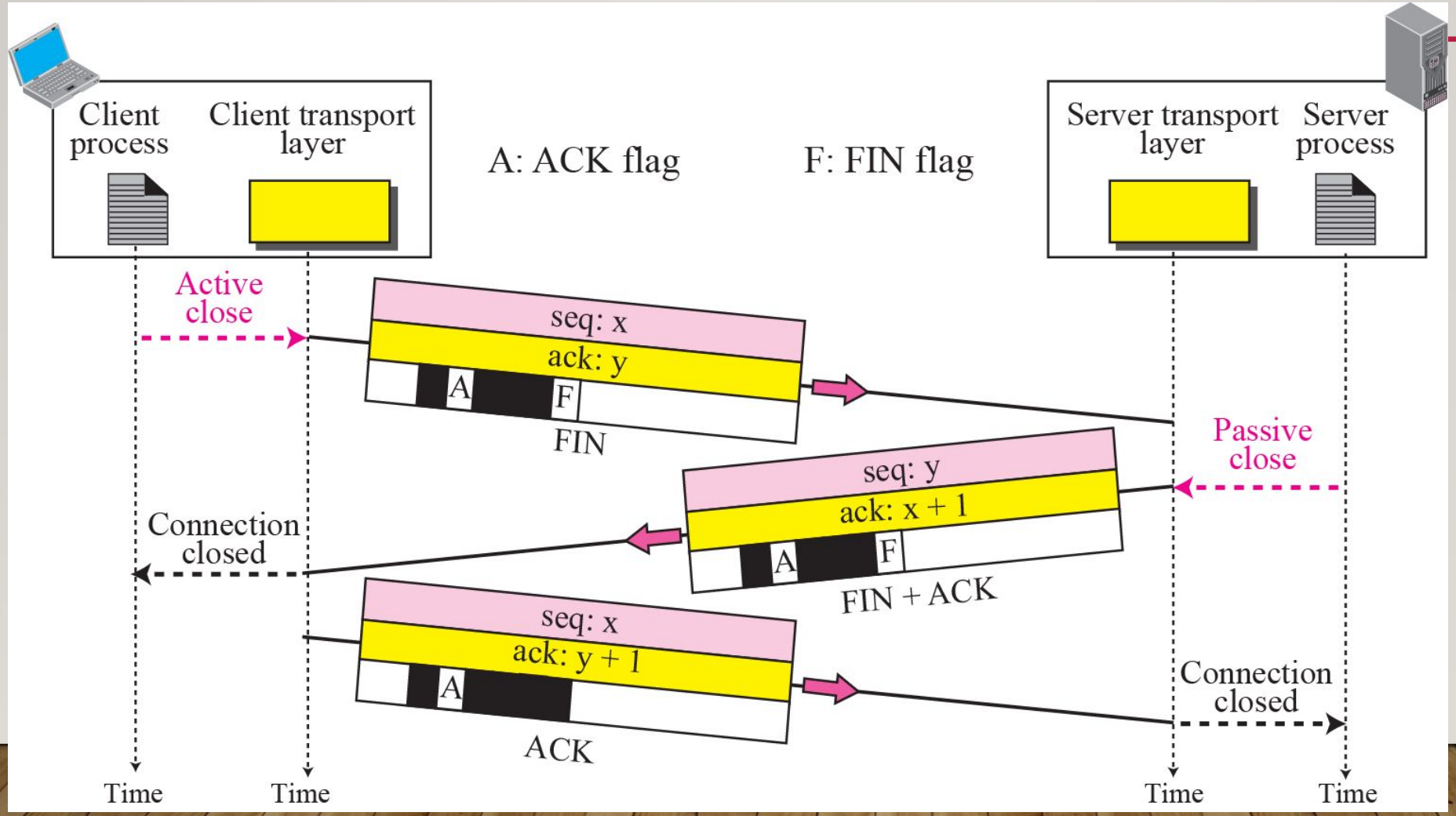
DATA TRANSFER- URGENT DATA

- Used when an application program needs to send urgent bytes.-> send a segment with URG bit set.
- Urgent data is inserted at the beginning of the segment.
- Rest of the segment can contain normal data from the buffer.
- Urgent pointer field in the header defines the end of the urgent data (lasts byte of the urgent data)

CONNECTION TERMINATION

- Any of the two parties can close a connection
- Three-way handshaking
  - Both ends agree and stop sending data.
- Four-way handshaking with a half-close option
  - One end stop sending data, but still is able to receive data

THREE-WAY HANDSHAKING- TERMINATION



THREE-WAY HANDSHAKING- TERMINATION

- Step 1:
  - Client TCP sends the first segment, a FIN segment in which FIN flag is set.
  - A FIN segment can include the last chunk of data sent by the client or it can be just a control segment.
  - FIN segment consumes one sequence number if it does not carry data.

THREE-WAY HANDSHAKING- TERMINATION

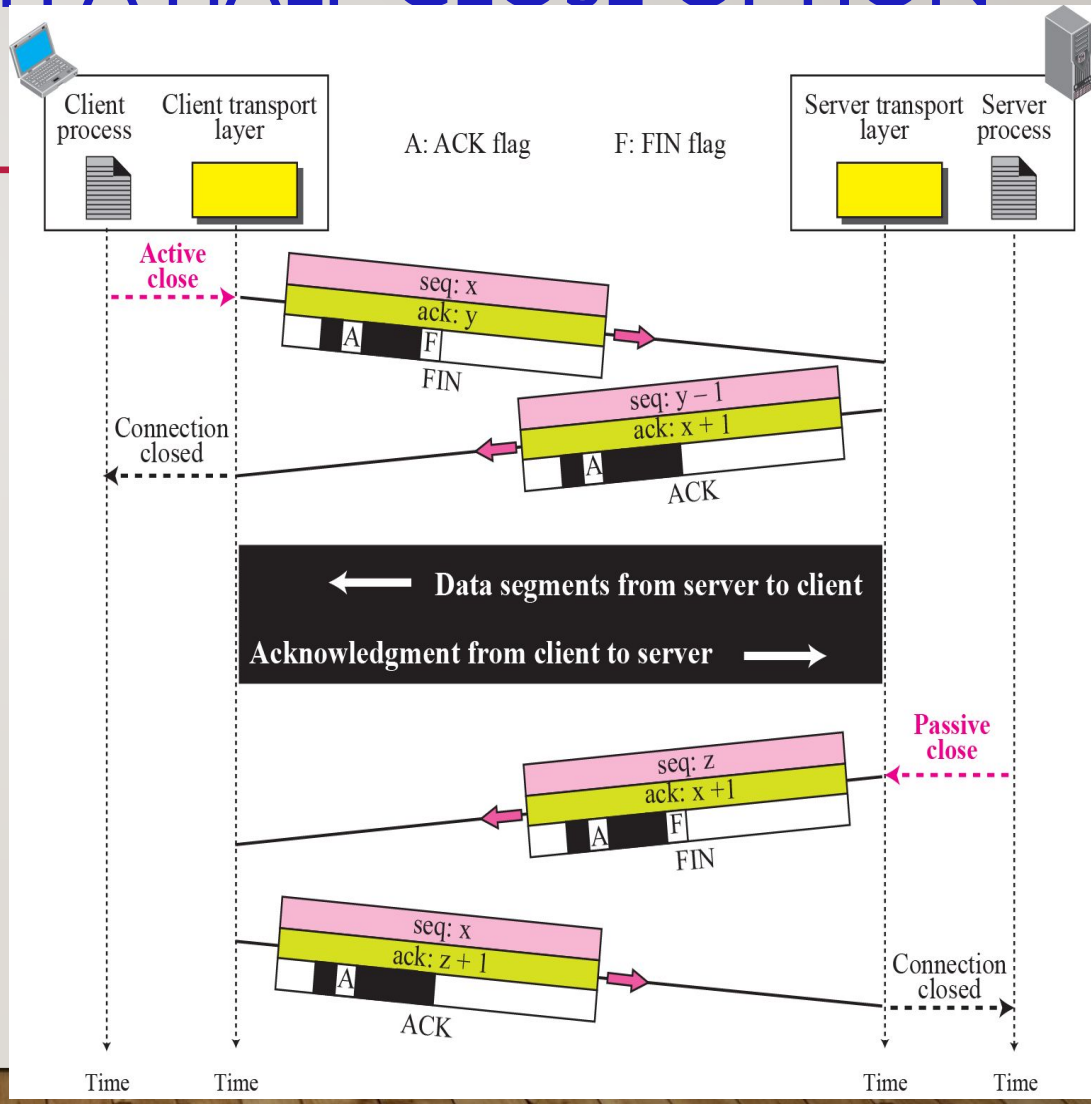
- Step 2:
  - Server TCP, after receiving the FIN segment sends the second segment, a FIN+ACK segment to confirm the receipt of the FIN segment from the client and at the same time to announce the closing of the connection.
  - FIN+ACK segment consumes one sequence number if it does not carry data.

THREE-WAY HANDSHAKING- TERMINATION

- Step 3:
  - Client TCP, sends the last segment, an ACK segment to confirm the receipt of the FIN segment from the TCP server.
  - This segment contains the acknowledgement number.
  - This segment does not carry data and consumes no sequence numbers.

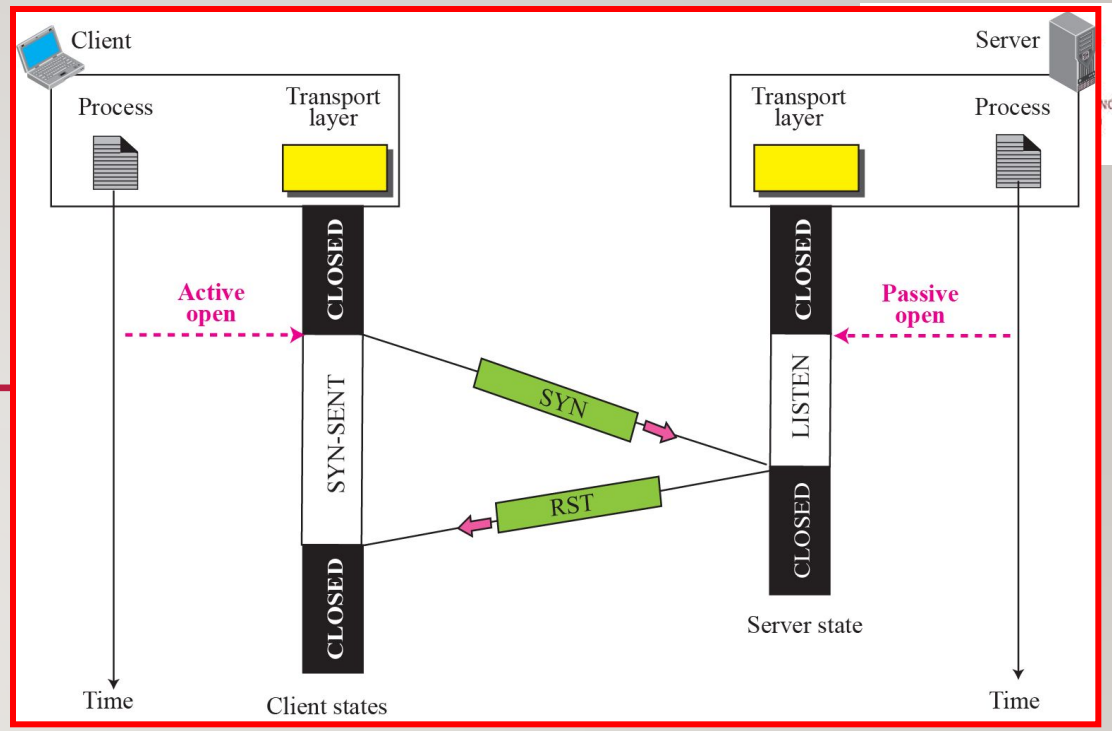
FOUR-WAY HANDSHAKING WITH A HALF-CLOSE OPTION

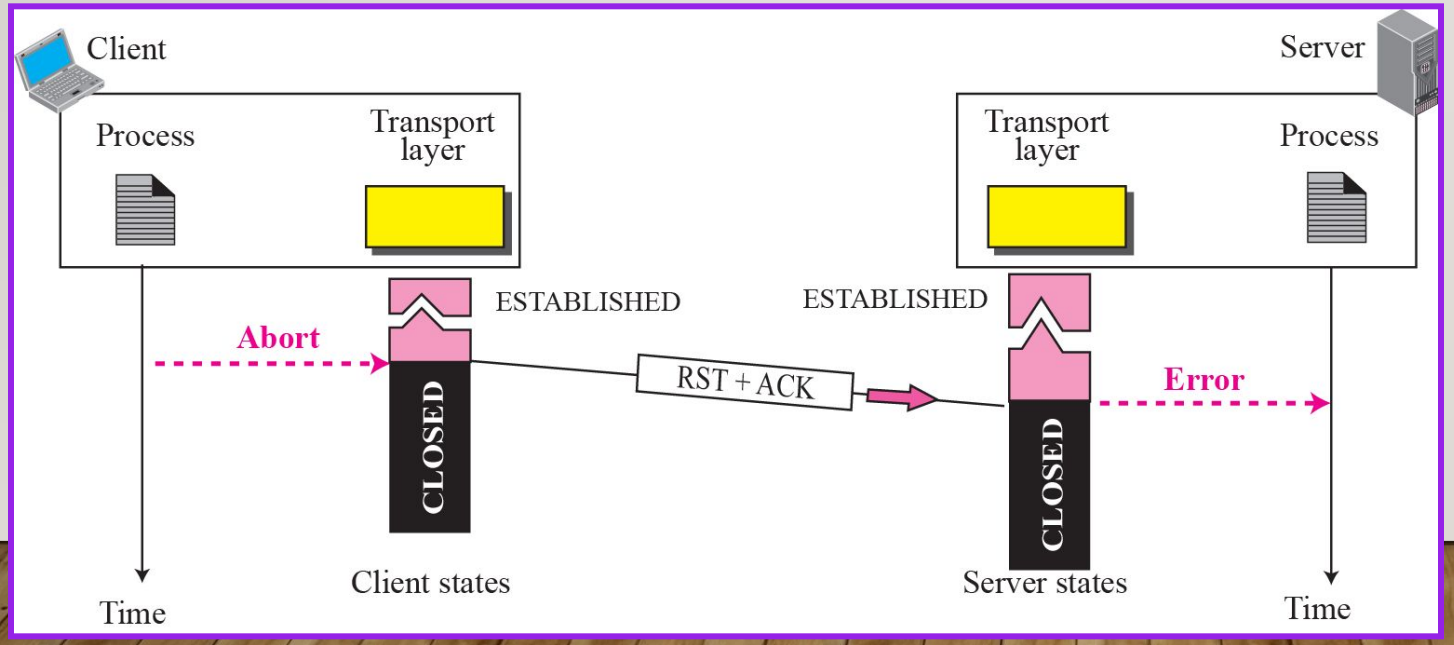
- Half Close:
  - One end can stop sending data while still receiving data.
  - Either the server or client can issue a half close request.
  - E.g. client half closes the connection by ending a FIN segment.
  - Server accepts the half close by sending the ACK segment.
  - When server has sent all the data, it sends a FIN segment, which is acknowledged



CONNECTION RESET

- TCP at one end may
  - Deny a connection request
  - Abort an existing connection or terminate an idle

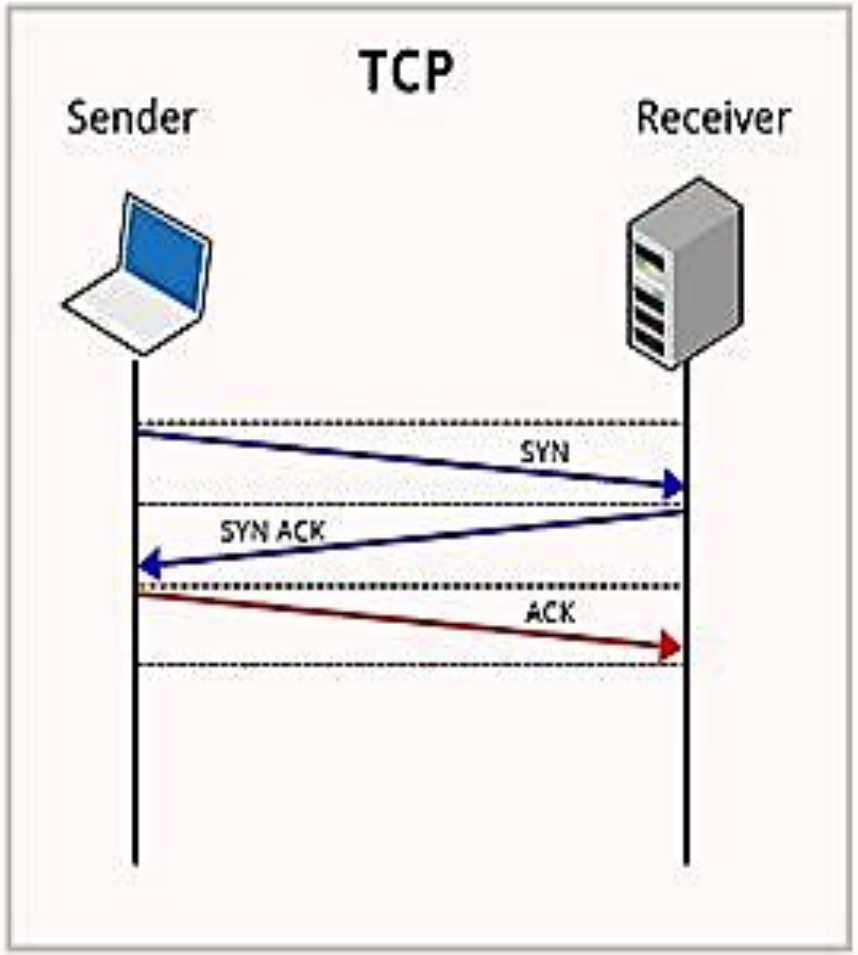


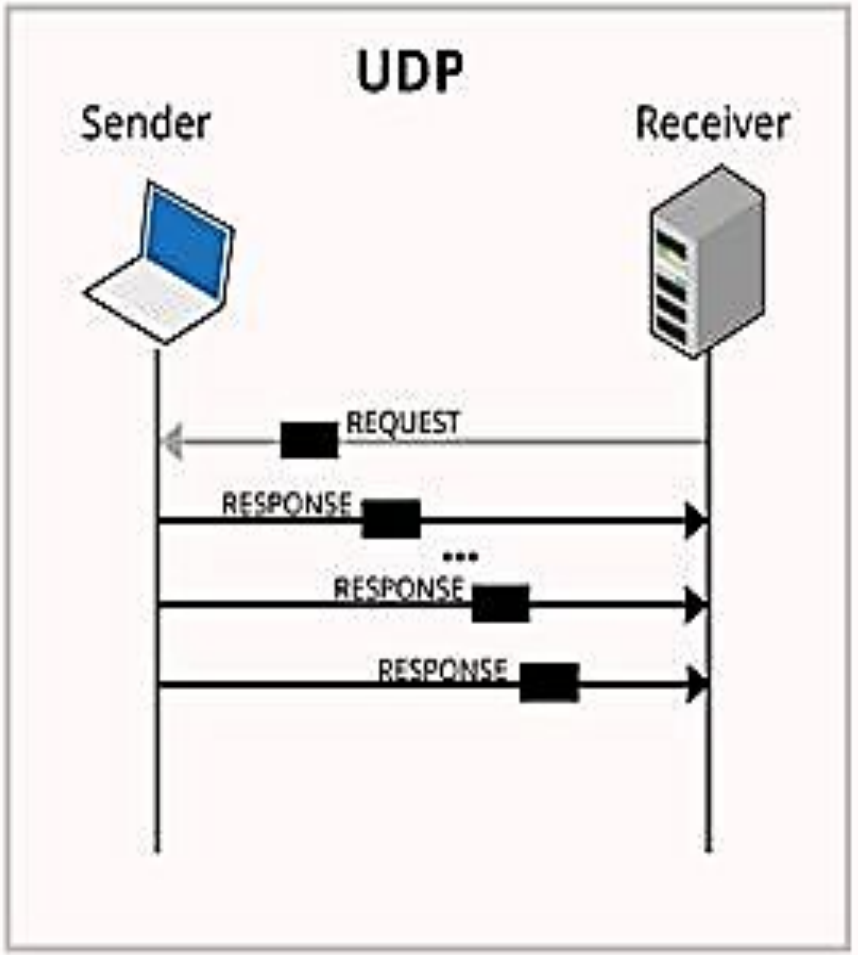


APPLICATIONS

- Bootstrap Protocol
- Dynamic Host Configuration Protocol (DHCP)
- Domain Name System (DNS)
- E-mail
- File Transfer Protocol (FTP)
- IP filtering and network address translation
- Secure Shell (SSH) applications such as PuTTY
- Remote Desktop (RDP) applications such as Microsoft Remote Desktop Client
- Secure File Transfer Protocol (SFTP)
- Secure Copy Protocol (SCP) applications such as WinSCP

TCP Vs UDP Communication





TCP	UDP
Secure	Unsecure
Connection-Oriented	Connectionless
Slow	Fast
Guaranteed transmittion	No Guarantee
Used by crtical applications	Used by real-time applications
Packet reorder mechanism	No reorder mechanism
Flow control	No flow control
Error Checking	No Error Checkin
20 Bytes Header	8 Bytes Header
Acknowledgement Mechanism	No Acknowledgement
Three-way handshake (SYN, SYN-ACK, ACK)	No hanshake
DNS, HTTP, HTTPs, FTP, SMTP, Telnet,SNMP	DNS, DHCP, TFTP, SNMP, RIP, VOIP

THANK YOU!!!

COMPUTER NETWORKS MODULE 5.3

MS. JINCY J FERNANDEZ

ASST. PROF, CSE

RSET

TCP RETRANSMISSION POLICY

- Retransmission resending the packets over the network that have been either lost or damaged to provide reliable communication.
- Retransmission means the data packets have been lost, which leads to a lack of acknowledgment. This lack of acknowledgment triggers a timer to timeout.
- **The sender sets the timeout period for an ACK. The timeout period can be of two types:**
  - **Too short:** If the timeout period is too short, then the retransmissions will be wasted.
  - **Too long:** If the timeout period is too long, then there will be an excessive delay when the packet is lost.

TCP RETRANSMISSION POLICY

- In modern implementations, a segment is retransmitted on two occasions: when a retransmission timer expires or when the sender receives three duplicate ACKs.
- No retransmission occurs for segments that do not consume sequence numbers.
- No retransmission timer is set for an ACK segment.

TCP RETRANSMISSION POLICY

- Retransmission After RTO:
  - TCP maintains one retransmission time-out (RTO) timer for all outstanding (sent, but not acknowledged) segments.
  - When the timer matures, the earliest outstanding segment is retransmitted.
  - no time-out timer is set for a segment that carries only an Acknowledgment
  - The value of RTO is dynamic in TCP and is updated based on the round-trip time (RTT) of segments. Round trip time is the time required for the packet to travel from the source to the destination and then come back again.
  - The RTT can vary depending upon the network's characteristics, i.e., if the network is congested, it means that the RTT is very high.

TCP RETRANSMISSION POLICY

- Retransmission After Three Duplicate ACK Segments:
  - Retransmission of a segment using RTO is sufficient if the value of RTO is not very large.
  - Sometimes, one segment is lost, and the receiver receives so many out-of-order segments that they cannot be saved (limited buffer size).
  - Most implementations today follow the three-duplicate-ACKs rule and retransmit the missing segment immediately.

TCP RETRANSMISSION POLICY

- Out of Order segments
  - When a segment is delayed, lost, or discarded, the segments following that segment arrive out of order.
  - Originally, TCP was designed to discard all out-of-order segments, resulting in the retransmission of the missing segment and the following segments.
  - Most implementations today do not discard the out-of-order segments. They store them temporarily and flag them as out-of-order segments until the missing segment arrives.
  - The out-of-order segments are not delivered to the process. TCP guarantees that data are delivered to the process in order..

TCP CONGESTION CONTROL

- Congestion occurs if the transport entities on many machines send too many packets into the network too quickly.
- Degrade the performance of the network as packets are delayed and lost.
- Controlling congestion is the combined responsibility of the network and transport layers.
- Congestion occurs at routers; so it is detected at the network layer.
- Congestion is caused by traffic sent into the network by the transport layer.
- Effective way to control congestion is for the transport layers to send packets into the network more slowly

TCP CONGESTION CONTROL

- TCP is an end-to-end protocol that uses the service of IP.
- The congestion in the router is in the IP territory and should be taken care of by IP.
- TCP uses a congestion window (cwnd), and a congestion policy that avoid congestion
- The size of cwnd is controlled by the congestion situation in the network.
- cwnd and rwnd together define the size of the send window in TCP.
- cwnd related to the congestion in the middle.
- rwnd related to the congestion at the end.
- Actual size of the send window= minimum(cwnd, rwnd)

TCP CONGESTION CONTROL- CONGESTION DETECTION

- TCP sender uses the occurrence of two events as signs of congestion in the network: time out and receiving three duplicate ACKs.
- Timeout: If a TCP sender does not receive an ACK for a segment or a group of segments before the timeout occurs, it assumes that corresponding segment or group of segments are lost and the loss is due to congestion.
- Three duplicate ACKs: 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.

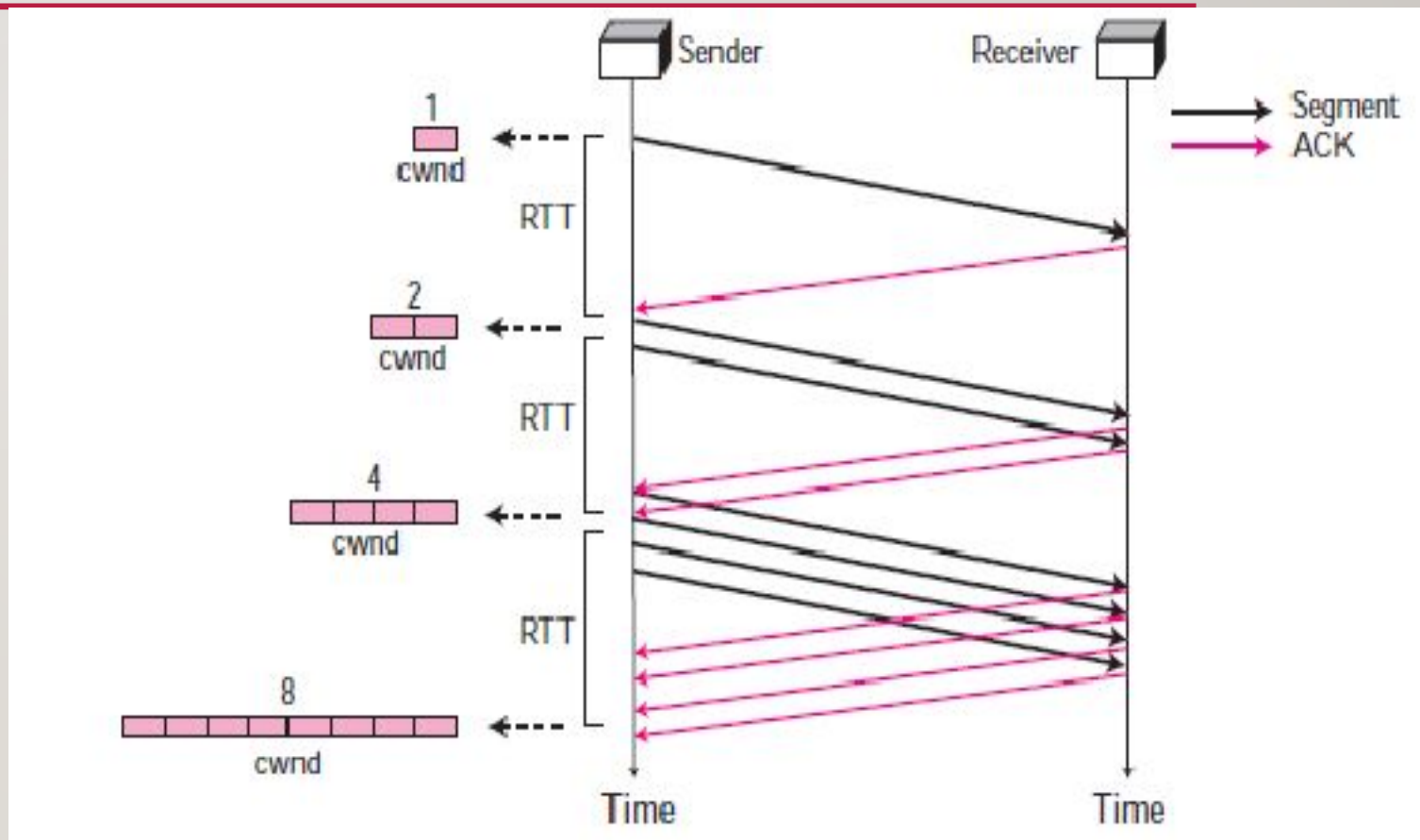
TCP CONGESTION CONTROL- CONGESTION POLICIES

- Slow start: Exponential Increase
  - In this phase, after every RTT the congestion window size increments exponentially.
  - The algorithm starts slowly but grows exponentially.
  - Assume that each segment is of same size and carries MSS bytes.

Start	
After 1 RTT	
After 2 RTT	
After 3 RTT	

TCP CONGESTION CONTROL- CONGESTION POLICIES

- Slow start: Exponential Increase



TCP CONGESTION CONTROL- CONGESTION POLICIES

- Slow start: Exponential Increase
  - A slow start cannot continue indefinitely; use a threshold to stop.
  - Sender keeps track of the threshold.
  - When the size of the window in bytes reaches the threshold, slow start stops.
  - The size of the congestion window increases exponentially until it reaches the threshold.
  - Threshold = Maximum number of TCP segments that receiver window can accommodate / 2

    = (Receiver window size / Maximum Segment Size) / 2

TCP CONGESTION CONTROL- CONGESTION POLICIES

- Congestion avoidance: Additive Increase
  - With slow start algorithm, the size of the congestion window increases exponentially.
  - To avoid congestion before it happens, slow down this exponential growth.
  - The size of the congestion window increases additively instead of exponentially.
  - When the size of the congestion window reaches the slow start threshold, slow start phase stops and additive phase begins.

TCP CONGESTION CONTROL- CONGESTION POLICIES

- Congestion avoidance: Additive Increase

Initially cwnd = i

After 1 RTT, cwnd = i+1

2 RTT, cwnd = i+2

3 RTT, cwnd = i+3

Start	cwnd=1
After 1 RTT	cwnd=cwnd+1= 1+1 =2
After 2 RTT	cwnd= cwnd+1 =2+1=3
After 3 RTT	cwnd= cwnd+1=3+1=4

TCP CONGESTION CONTROL- CONGESTION POLICIES

- Congestion Detection Phase : multiplicative decrement
  - If congestion occurs, the congestion window size is decreased.
  - If a time-out occurs
    - a. It sets the value of the threshold to one-half of the current window size.
    - b. It sets *cwnd* to the size of one segment.
    - c. It starts the slow-start phase again.
  - If three ACKs are received,
    - It sets the value of the threshold to one-half of the current window size.
    - It sets *cwnd* to the value of the threshold
    - It starts the congestion avoidance phase.

THANK YOU!!!

COMPUTER NETWORKS MODULE 5.4

MS. JINCY J FERNANDEZ

ASST. PROF, CSE

RSET

APPLICATION LAYER

- The application layer enables the user, whether human or software, to access the network.
- It provides user interfaces and support for services such as electronic mail, file access and transfer, access to system resources, surfing the world wide web, and network management.
- The application layer is responsible for providing services to the user.

SERVICES OF APPLICATION LAYER

- **Network Virtual terminal:**
  - An application layer allows a user to log on to a remote host. To do so, the application creates a software emulation of a terminal at the remote host. The user's computer talks to the software terminal, which in turn, talks to the host. The remote host thinks that it is communicating with one of its own terminals, so it allows the user to log on.
- **File Transfer, Access, and Management (FTAM):**
  - An application allows a user to access files in a remote computer, to retrieve files from a computer and to manage files in a remote computer.

SERVICES OF APPLICATION LAYER

- **Addressing:**
  - To obtain communication between client and server, there is a need for addressing. When a client made a request to the server, the request contains the server address and its own address. The server response to the client request, the request contains the destination address, i.e., client address. To achieve this kind of addressing, DNS is used.
- **Mail Services:**
  - An application layer provides Email forwarding and storage.
- **Directory Services:**
  - An application contains a distributed database that provides access for global information about various objects and services.

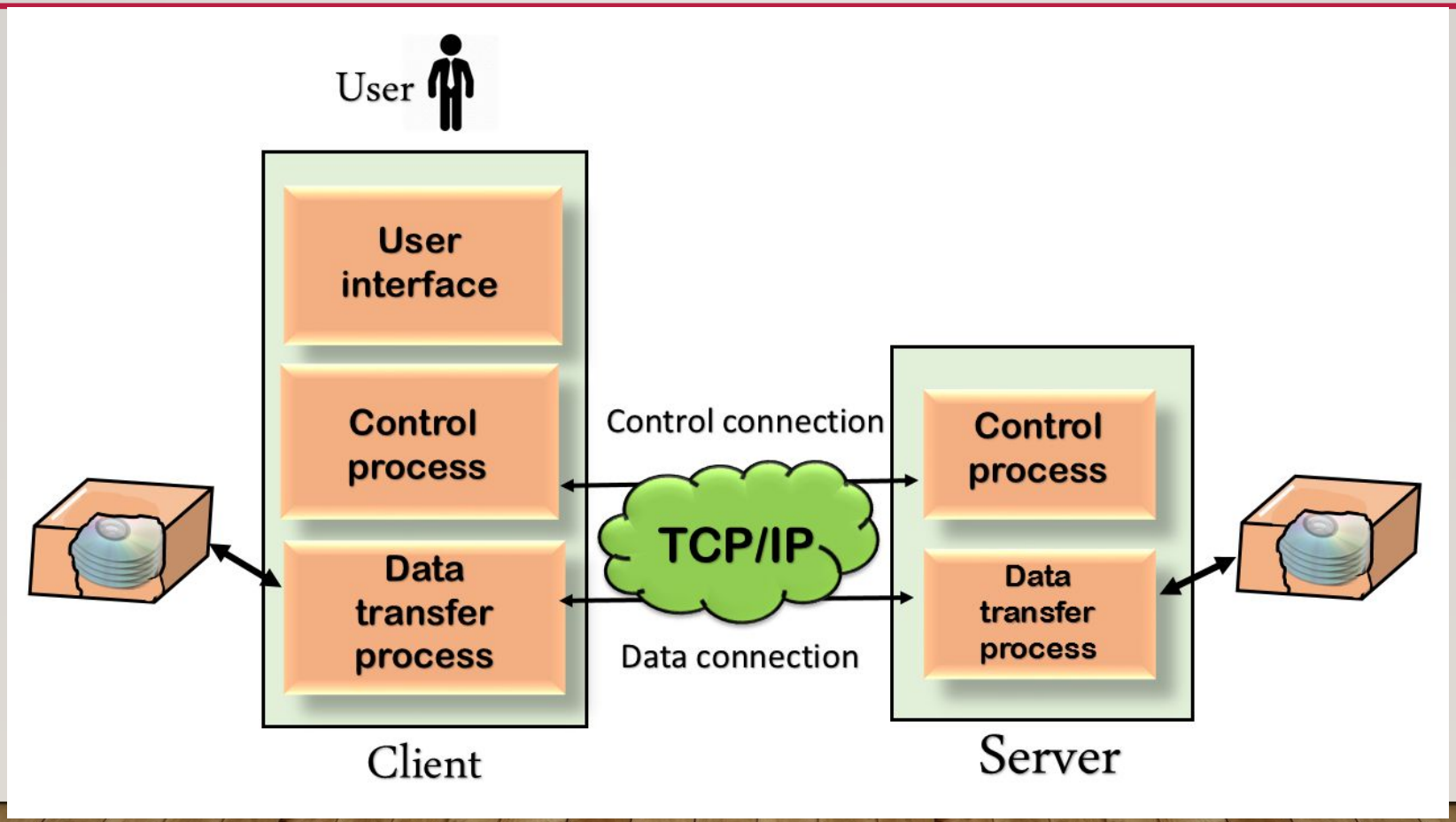
FILE TRANSFER PROTOCOL

- Transferring files from one computer to another is one of the most common tasks expected from a networking or internetworking environment.
- File Transfer Protocol (FTP) is the standard mechanism provided by *TCP/IP* for copying a file from one host to another.
- Transferring files from one system to another seems simple and straightforward.
- Problems exist
  - Two systems may use different file name conventions.
  - Two systems may have different ways to represent text and data.
  - Two systems may have different directory structures.

FILE TRANSFER PROTOCOL

- FTP differs from other client/server applications in that it establishes two connections between the hosts.
  - One connection is used for data transfer.
  - Other for control information (commands and responses).
- The control connection uses very simple rules of communication.
- The data connection needs more complex rules due to the variety of data types transferred.
- FTP uses two well-known TCP ports: Port 21 is used for the control connection, and port 20 is used for the data connection.

FILE TRANSFER PROTOCOL



FILE TRANSFER PROTOCOL

- The client has three components:
  - User interface, Client control process, Client data transfer process.
- The server has two components: server control process and server data transfer process.
- The control connection is made between the control processes.
- The data connection is made between the data transfer processes.
- The control connection remains connected during the entire interactive FTP session.
- The data connection is opened and then closed for each file transferred.
- when a user starts an FTP session, the control connection opens.
- While the control connection is open, the data connection can be opened and closed multiple times if several files are transferred.

FILE TRANSFER PROTOCOL

- Communication is achieved through commands and responses.
- Each command or response is only one short line, no need to worry about file format or file structure..
- A file is to be copied from the server to the client. This is called *retrieving.* It is done under the supervision of the RETR command,
- A file is to be copied from the client to the server. This is called *storing.* It is done under the supervision of the STOR command.
- A list of directory or file names is to be sent from the server to the client. This is done under the supervision of the LIST command. FTP treats a list of directory or file names as a file. It is sent over the data connection.

FILE TRANSFER PROTOCOL

Command	Argument(s)	Description
ABOR		Abort the previous command
CDUP		Change to parent directory
CWD	Directory name	Change to another directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
MKD	Directory name	Create a new directory
PASS	User password	Password
PASV		Server chooses a port
PORT	port identifier	Client chooses a port
PWD		Display name of current directory
QUIT		Log out of the system
RETR	File name(s)	Retrieve files; files are transferred from server to client
RMD	Directory name	Delete a directory
RNFR	File name (old)	Identify a file to be renamed
RNTO	File name (new)	Rename the file
STOR	File name(s)	Store files; file(s) are transferred from client to server
STRU	**F**, **R**, or **P**	Define data organization (**F**: file, **R**: record, or **P**: page)
TYPE	**A**, **E**, **I**	Default file type (**A**: ASCII, **E**: EBCDIC, **I**: image)
USER	User ID	User information
MODE	**S**, **B**, or **C**	Define transmission mode (**S**: stream, **B**: block, or **C**: compressed

FILE TRANSFER PROTOCOL

- File Type
  - FTP can transfer one of the following file types across the data connection: an ASCII file, EBCDIC file, or image file.
  - The ASCII file is the default format for transferring text files. Each character is encoded using 7-bit ASCII. The sender transforms the file from its own representation into ASCII characters, and the receiver transforms the ASCII characters to its own representation.
  - EBCDIC developed by IBM
  - The image file is the default format for transferring binary files.

FILE TRANSFER PROTOCOL

- Data Structure:
  - FTP can transfer a file across the data connection by using one of the following interpretations about the structure of the data: file structure, record structure, and page structure.
  - In the file structure format, the file is a continuous stream of bytes.
  - In the record structure, the file is divided into records. This can be used only with text files.
  - In the page structure, the file is divided into pages, with each page having a page number and a page header. The pages can be stored and accessed randomly or sequentially.

THANK YOU!!!