# Syllabus

**Module II (10 Hours)** Syllabus

- Transport Layer Protocols: Introduction to transport layer, Multiplexing and de-multiplexing, Principles of Reliable data transfer - Stop-and-wait and Go-back-N design and evaluation, Connection oriented transport TCP, Connectionless transport UDP, Principles of congestion control -efficiency and fairness


- Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education, 1 st Edition (2011).
- James F Kurose and Keith W Ross, "Computer Networking: A Top - Down Approach", Pearson Education; 6 th Edition (2017)

# Transport Layer

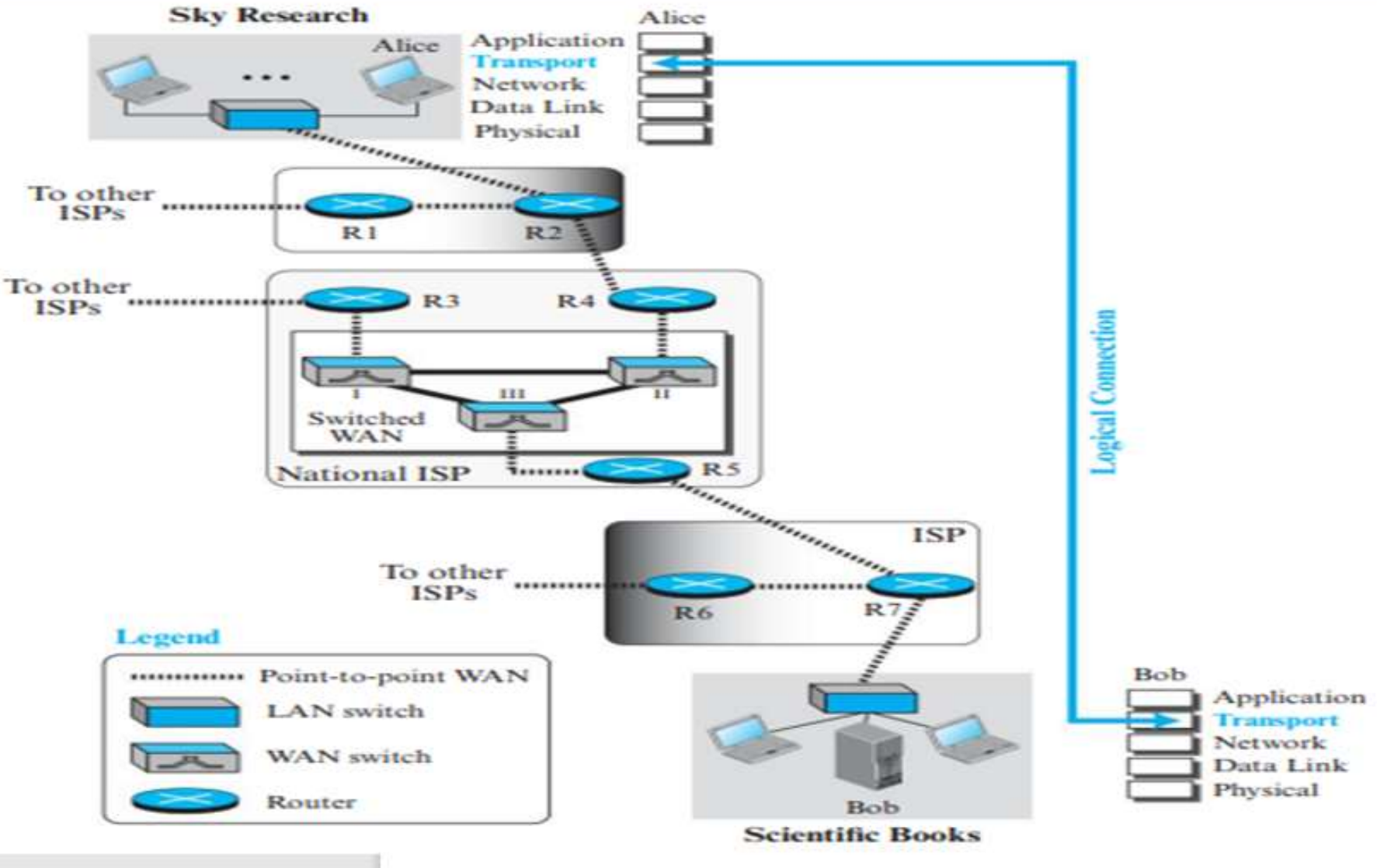
- The transport layer in the TCP/IP suite is located between the application layer and the network layer.
- It provides services to the application layer and receives services from the network layer.
- The transport layer is the heart of the TCP/IP protocol suite; it is the end-to-end logical vehicle for transferring data from one point to another in the Internet.

# Transport Layer - INTRODUCTION

- The transport layer is located between the application layer and the network layer.
- It provides a process-to-process communication between two application layers, one at the local host and the other at the remote host.

# Transport Layer - INTRODUCTION

Figure 2.1 *Logical connection at the transport layer*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Process-to-Process Communication

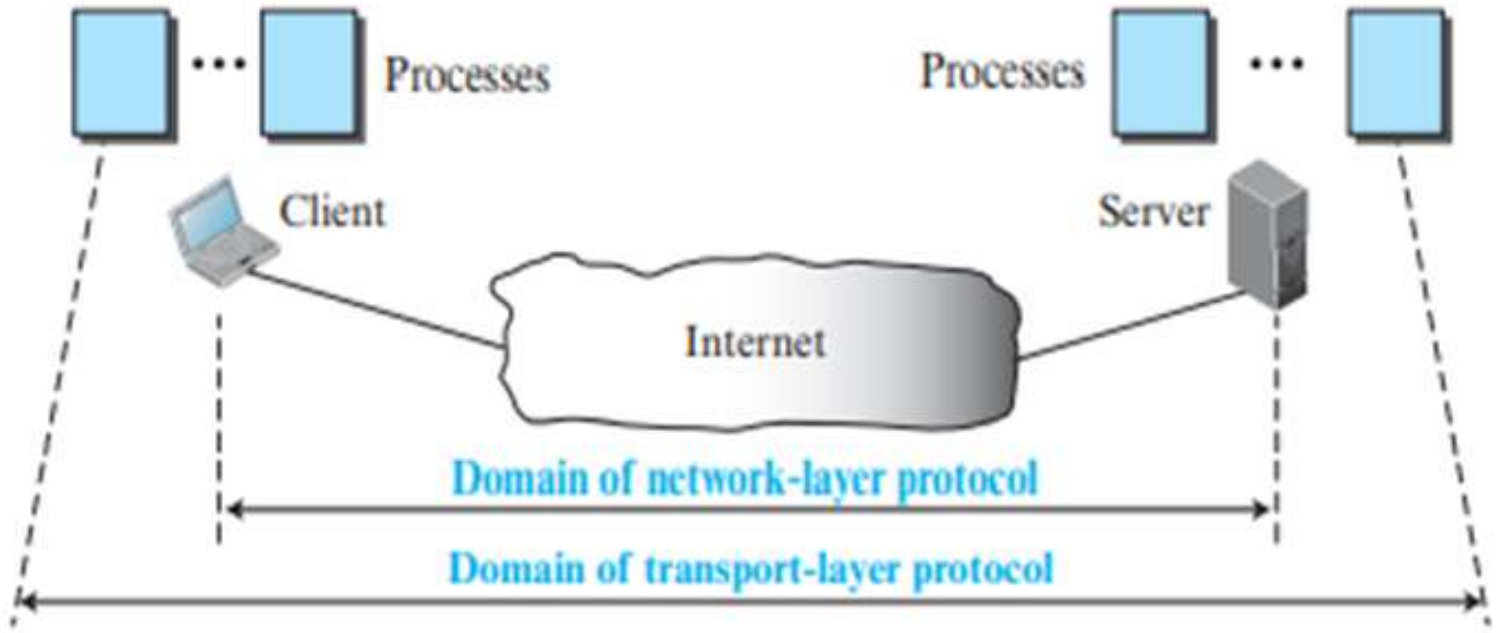
- The first duty of a transport-layer protocol is to provide process-to-process communication.
- A process is an application-layer entity (running program) that uses the services of the transport layer.
- We need to understand the difference between host-to-host communication and process-to-process communication.
- 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.

# Transport-Layer Services

## Process-to-Process Communication

- Figure 2.2 shows the domains of a network layer and a transport layer.

Figure 2.2 *Network layer versus transport layer*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

Addressing: Port Numbers

- 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.

# Transport-Layer Services

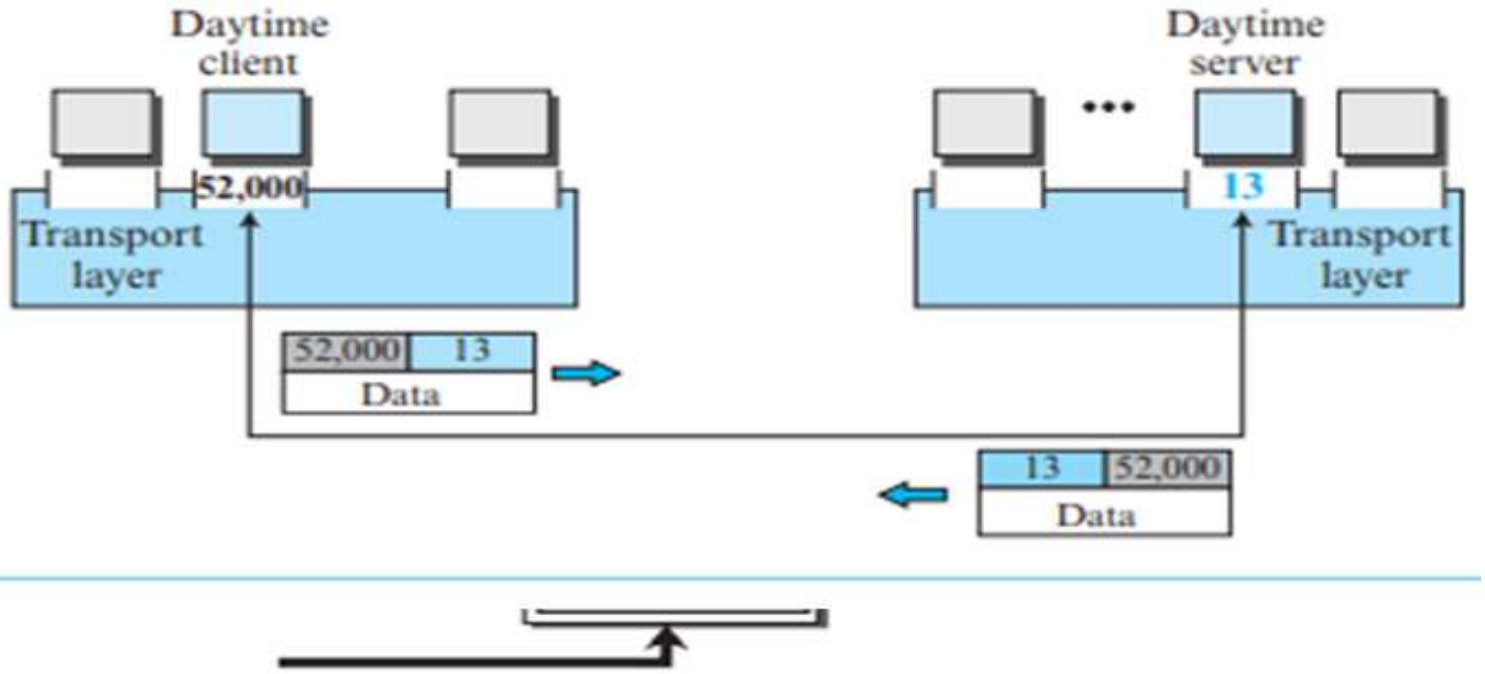
Addressing: Port Numbers

- An ephemeral port number is recommended to be greater than 1,023 for client/server programs to work properly.
- The server process must also define itself with a port number.
- TCP/IP has decided to use universal port numbers for servers; these are called well-known port numbers.
- Every client process knows the well-known port number of the corresponding server process.

# Transport-Layer Services

## Addressing: Port Numbers

Figure 2.3 *Port numbers*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

- The client process can use an ephemeral (temporary) port number, 52,000, to identify itself, the server process must use the well-known (permanent) port number 13("daytime" service using either tcp/ip or udp/ip).

# Transport-Layer Services

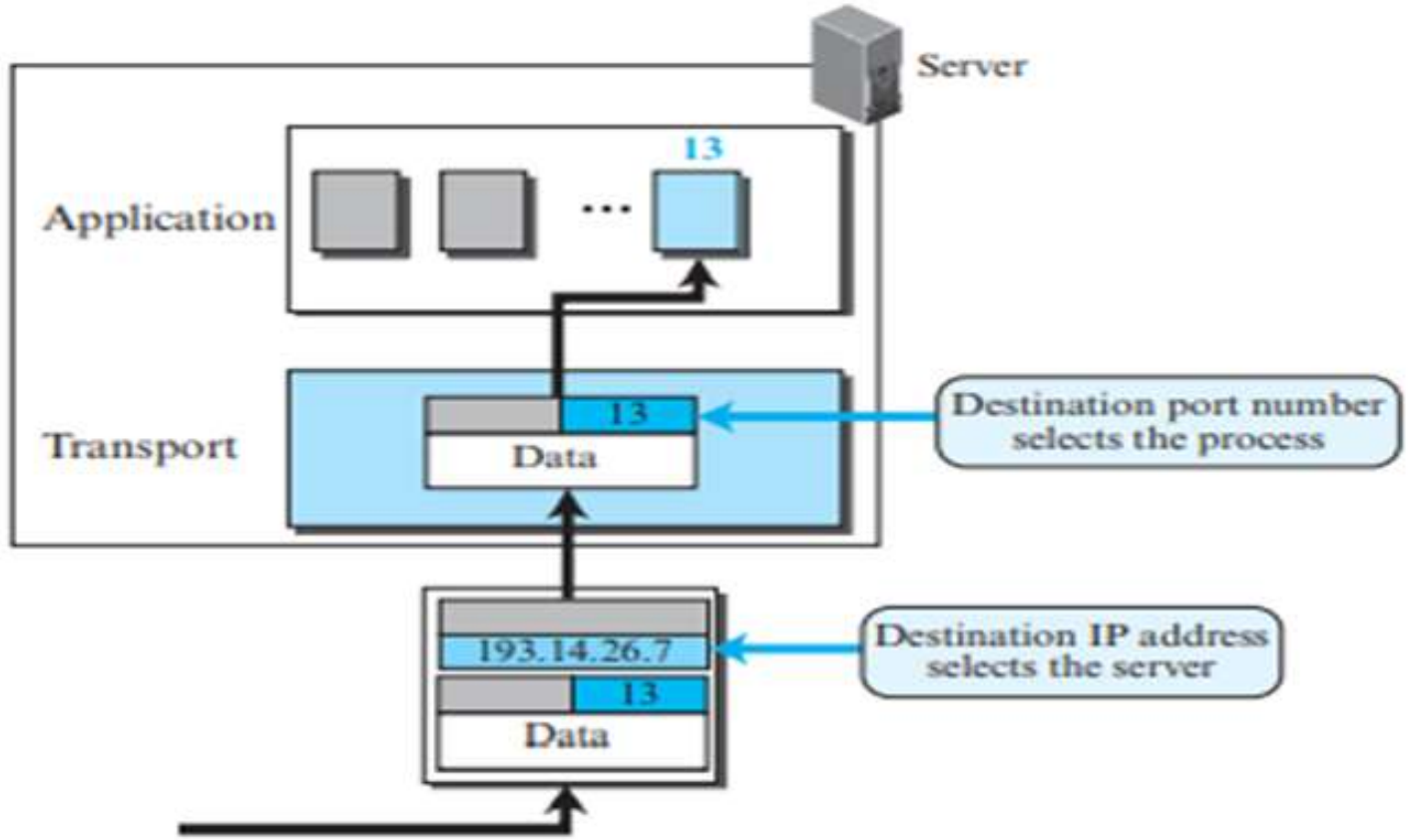
## Addressing: Port Numbers

- IP addresses and port numbers play different roles in selecting the final destination of data.
- The destination IP address defines the host among the different hosts in the world.
- After the host has been selected, the port number defines one of the processes on this particular host.

# Transport-Layer Services

## Addressing: Port Numbers

Figure 2.4 *IP addresses versus port numbers*



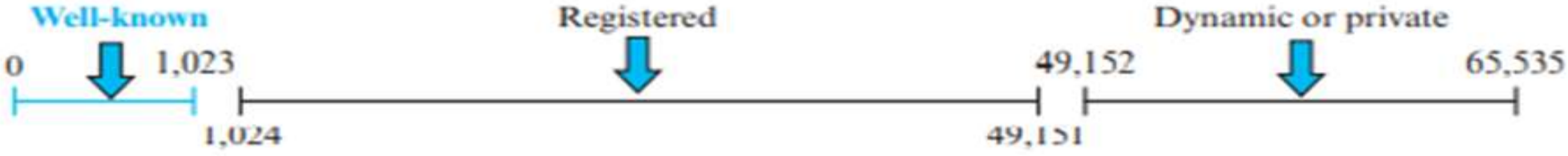

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Addressing: Port Numbers

- Port numbers into three ranges: well-known, registered, and dynamic (or private)
- Well-known ports: The ports ranging from 0 to 1,023 are assigned and controlled by ICANN.
- Registered ports: The ports ranging from 1,024 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 2.5 *ICANN ranges* The Internet Corporation for Assigned Names and Numbers




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

Example 2.1

- In UNIX, the well-known ports are stored in a file called /etc/services.
- Each line in this file gives the name of the server and the well-known port number.
- We can use the grep utility to extract the line corresponding to the desired application.
- The following shows the port for TFTP.
- Note that TFTP can use port 69 on either UDP or TCP.
    - $grep tftp/etc/services
    - tftp 69/tcp
    - tftp 69/udp

Trivial File Transfer Protocol - Used to transfer configurations to and from network devices

# Transport-Layer Services

Example 2.1

- SNMP uses two port numbers (161 and 162), each for a different purpose.
  - $grep snmp/etc/services
  - snmp161/tcp#Simple Net Mgmt Proto
  - snmp161/udp#Simple Net Mgmt Proto
  - snmptrap162/udp#Traps for SNMP

Traps(port no 162) are a part of the SNMP protocol and are used to monitor network devices in real time and detect critical events and errors

# Transport-Layer Services

## Socket Addresses

- A transport-layer protocol in the TCP suite needs both the IP address and the port number, at each end, to make a connection. The combination of an IP address and a port number is called a socket address.
- The client socket address defines the client process uniquely just as the server socket address defines the server process uniquely (see Figure 2.6).

Figure 2.6 *Socket address*

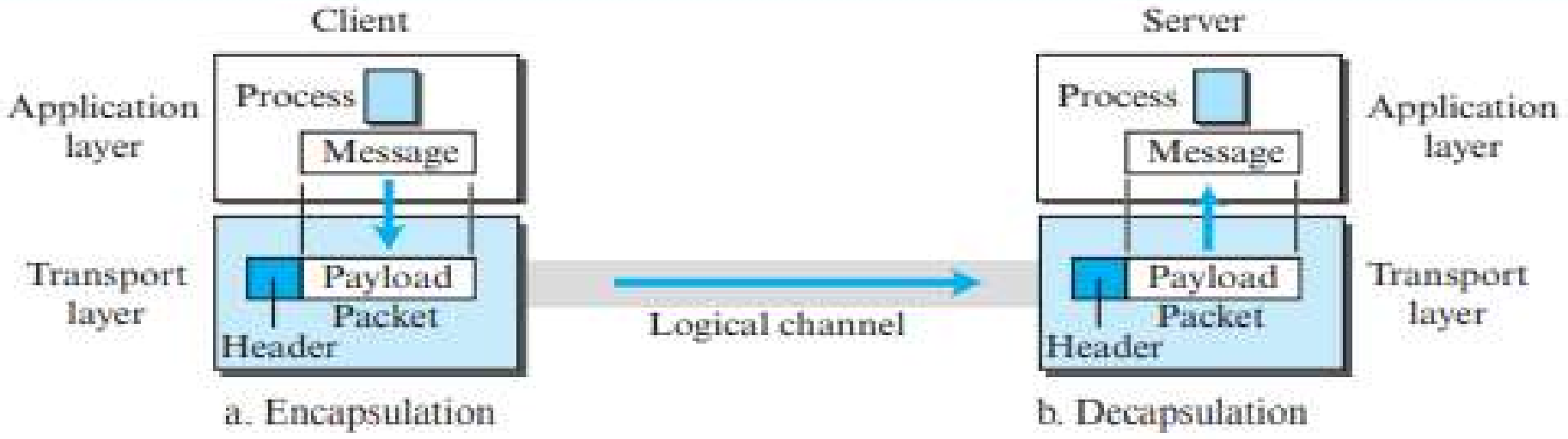

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Encapsulation and Decapsulation

- To send a message from one process to another, the transport-layer protocol encapsulates and decapsulates.
- The packets at the transport layers in the Internet are called user datagrams, segments, or packets, depending on what transport-layer protocol we use.

Figure 2.7 *Encapsulation and decapsulation*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Multiplexing and Demultiplexing

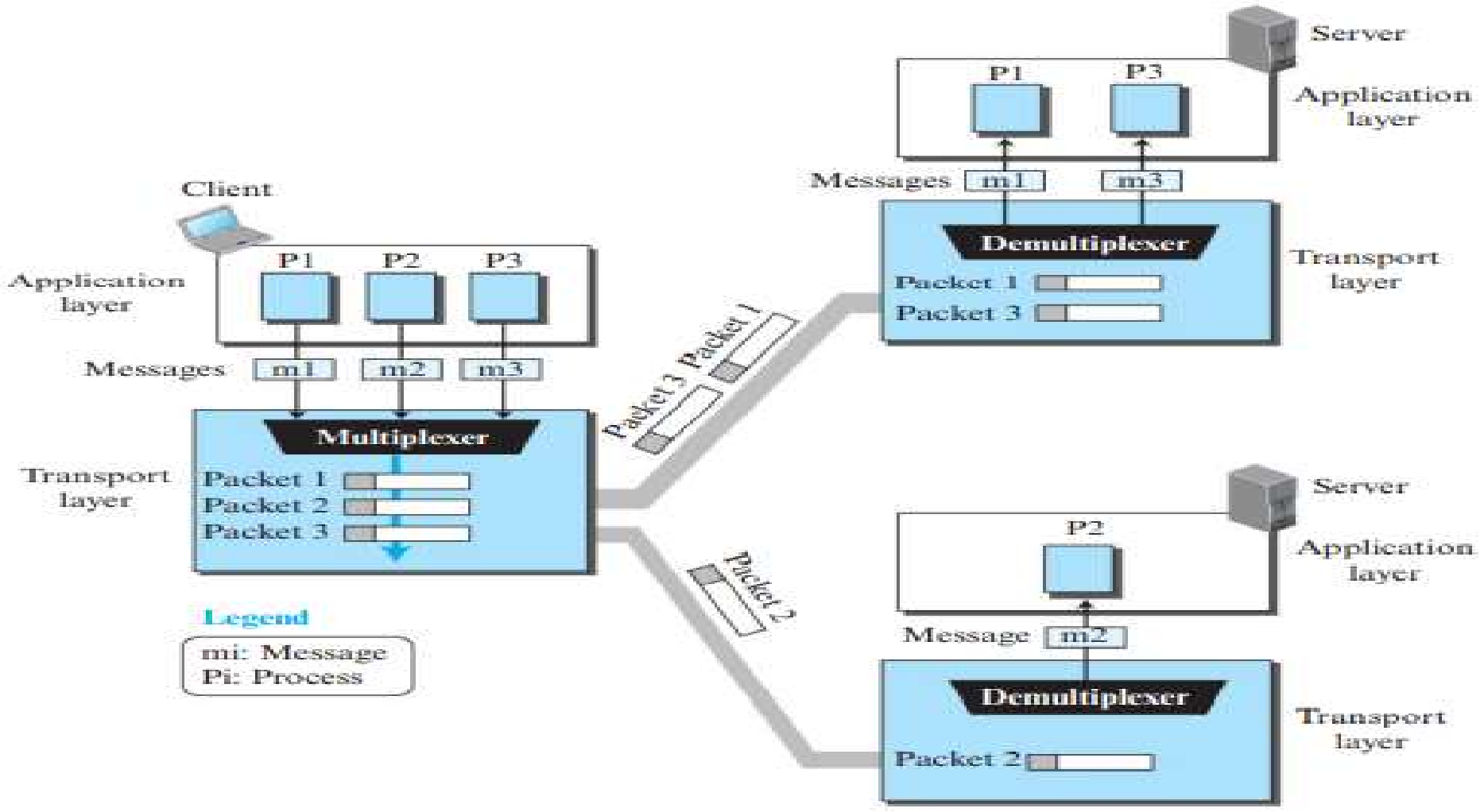
- Whenever an entity accepts items from more than one source, this is referred to as multiplexing (many to one); whenever an entity delivers items to more than one source, this is referred to as demultiplexing (one to many).

- The transport layer at the source performs multiplexing; the transport layer at the destination performs demultiplexing (Figure 2.8).

# Transport-Layer Services

## Multiplexing and Demultiplexing

Figure 2.8 *Multiplexing and demultiplexing*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Flow Control at Transport Layer

- In communication at the transport layer, we are dealing with four entities: sender process, sender transport layer, receiver transport layer, and receiver process.

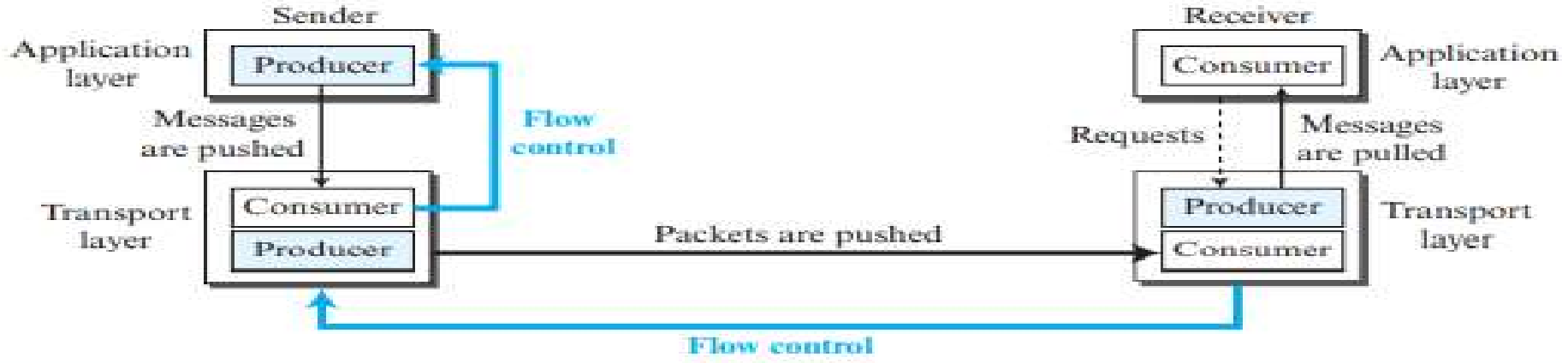
- The sending process at the application layer is only a producer. It produces message chunks and pushes them to the transport layer.
- The sending transport layer has a double role: it is both a consumer and a producer. It consumes the messages pushed by the producer. It encapsulates the messages in packets and pushes them to the receiving transport layer.

# Transport-Layer Services

## Flow Control at Transport Layer

- The receiving transport layer also has a double role, it is the consumer for the packets received from the sender and the producer that decapsulates the messages and delivers them to the application layer.
- The last delivery is normally a pulling delivery; the transport layer waits until the application-layer process asks for messages.

Figure 2.9 *Flow control at the transport layer*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Flow Control - 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.
- When the buffer of the sending transport layer is full, it informs the application layer to stop passing chunks of messages; when there are some vacancies, it informs the application layer that it can pass message chunks again.
- When the buffer of the receiving transport layer is full, it informs the sending transport layer to stop sending packets. When there are some vacancies, it informs the sending transport layer that it can send packets again.

# Transport-Layer Services

Error Control

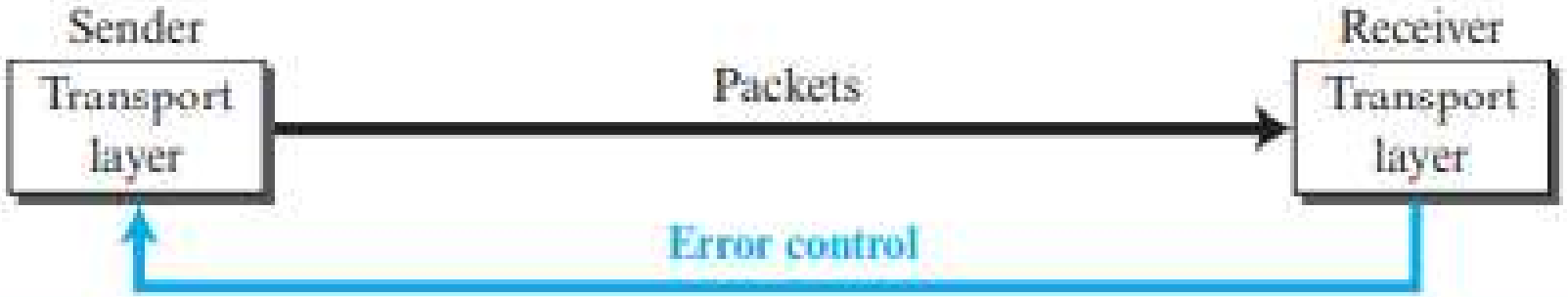
- In the Internet, since the underlying network layer (IP) is unreliable, we need to make the transport layer reliable if the application requires reliability.
- Reliability can be achieved to add error control services to the transport layer.
- Error control at the transport layer is responsible for
  1. Detecting and discarding corrupted packets.
  2. Keeping track of lost and discarded packets and resending them.
  3. Recognizing duplicate packets and discarding them.
  4. Buffering out-of-order packets until the missing packets arrive.

# Transport-Layer Services

## 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.
- Figure 2.10 shows the error control between the sending and receiving transport layer.

Figure 2.10 *Error control at the transport layer*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Error Control - 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.
- When a packet is corrupted or lost, the receiving transport layer can inform the sending transport layer to resend that packet using the sequence number.
- The receiving transport layer can also detect duplicate packets if two received packets have the same sequence number.
- The out-of-order packets can be recognized by observing gaps in the sequence numbers.

# Transport-Layer Services

Error Control - Sequence Numbers

- Packets are numbered sequentially.
- However, because we need to include the sequence number of each packet in the header, we need to set a limit.
- If the header of the packet allows m bits for the sequence number, the sequence numbers range from 0 to $2^m - 1$.
- For example, if m is 4, the only sequence numbers are 0 through 15, inclusive.
- However, we can wrap around the sequence. The sequence numbers are modulo $2^m$.
- For error control, the sequence numbers are modulo $2^m$, where m is the size of the sequence number field in bits.

# Transport-Layer Services

Error Control - Acknowledgment and timer

- We can use both positive and negative signals as error control, but we discuss only positive signals, which are more common at the transport layer.
- The receiver side can send an acknowledgment (ACK) for each of a collection of packets that have arrived safe and sound.
- The receiver can simply discard the corrupted packets. 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.

# Transport-Layer Services

## Combination of Flow and Error Control

- Flow control requires the use of two buffers, one at the sender site and the other at the receiver site.
- Error control requires the use of sequence and acknowledgment numbers by both sides.
- These two requirements can be combined if we use two numbered buffers, one at the sender, one at the receiver.
- 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.

# Transport-Layer Services

Combination of Flow and Error Control

- 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.

# Transport-Layer Services

Sliding Window

- The sequence numbers used modulo $2^m$, a circle can represent the sequence numbers from 0 to $2^m - 1$.
- The buffer is represented as a set of slices, called the sliding window, that occupies part of the circle at any time.
- At the sender site, when a packet is sent, the corresponding slice is marked.
- When all the slices are marked, it means that the buffer is full and no further messages can be accepted from the application layer.
- When an acknowledgment arrives, the corresponding slice is unmarked.

# Transport-Layer Services

## Sliding Window

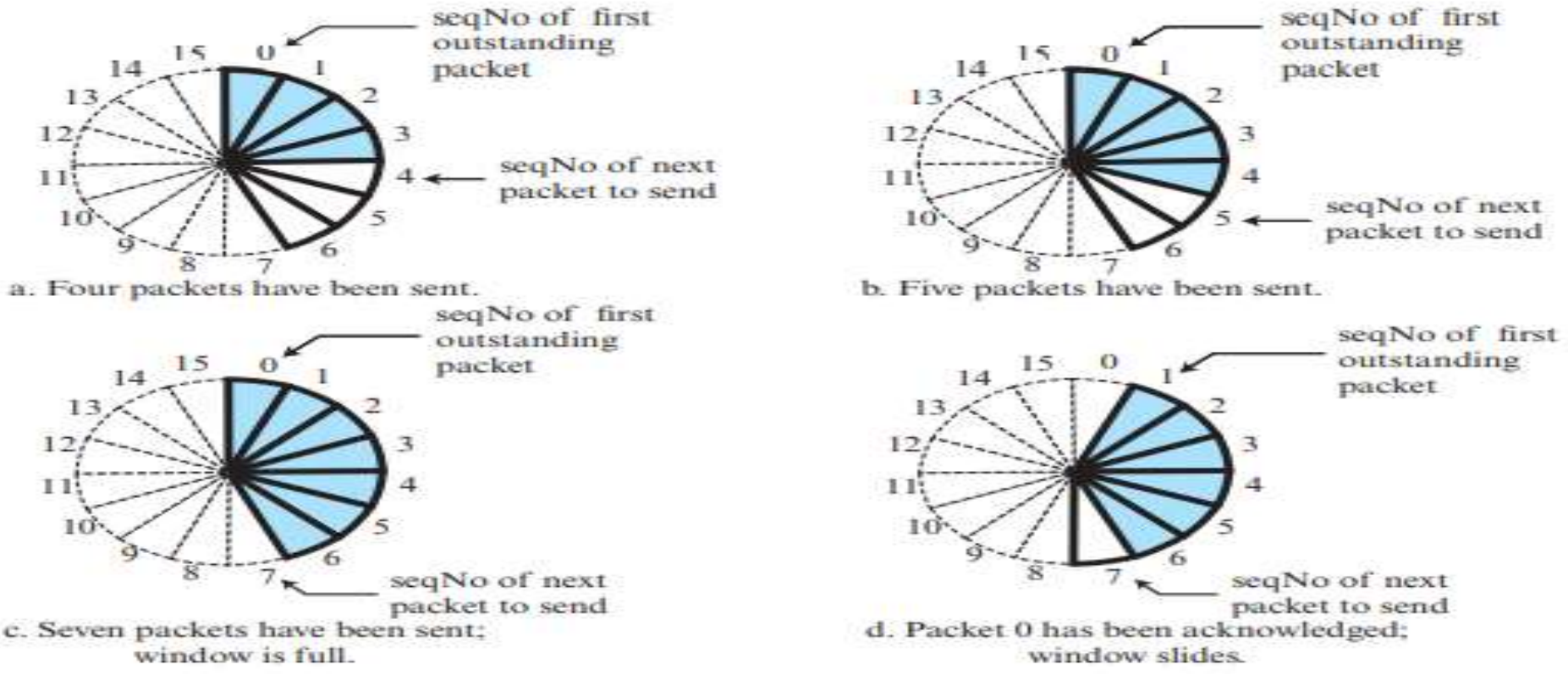
- If some consecutive slices from the beginning of the window are unmarked, the window slides over the range of the corresponding sequence numbers to allow more free slices at the end of the window.

# Transport-Layer Services

## Sliding Window

- Figure 2.11 shows the sliding window at the sender. The sequence numbers are in modulo 16 (m = 4) and the size of the window is 7.
- Note that the sliding window is just an abstraction.

Figure 2.11 *Sliding window in circular format*



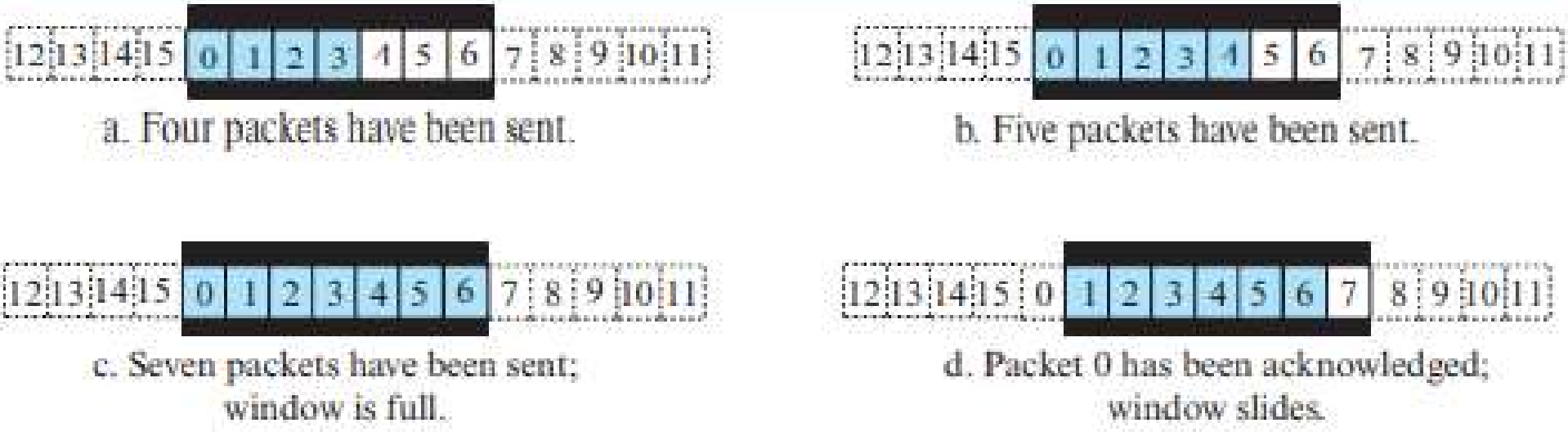

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Sliding Window

- Most protocols show the sliding window using linear representation.

Figure 2.12 *Sliding window in linear format*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Congestion Control

- An important issue in a packet-switched network, such as the Internet, is congestion.
- Congestion in a network may occur 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 that control the congestion and keep the load below the capacity.

# Transport-Layer Services

## Congestion Control

- Congestion in a network or internetwork occurs because routers and switches have queues—buffers that hold the packets before and after processing.
- A router, for example, has an input queue and an output queue for each interface.
- If a router cannot process the packets at the same rate at which they arrive, the queues become overloaded and congestion occurs.
- Congestion at the transport layer is actually the result of congestion at the network layer, which manifests itself at the transport layer.

# Transport-Layer Services

## Connectionless and Connection-Oriented Services

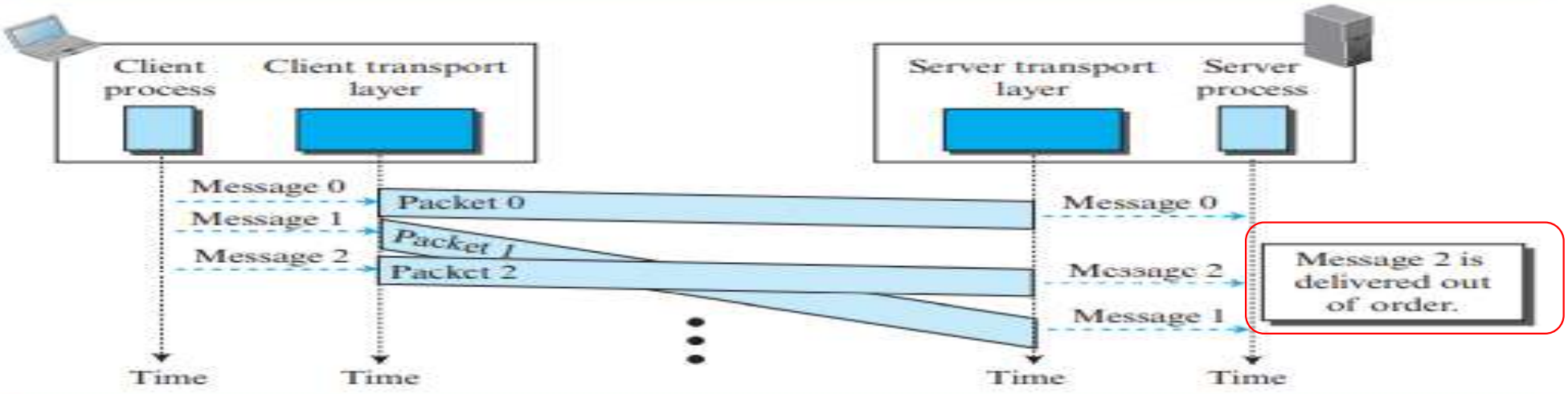
- A transport-layer protocol can provide two types of services: connectionless and connection-oriented.
- At the transport layer, we are not concerned about the physical paths of packets (we assume a logical connection between two transport layers).
- Connectionless service at the transport layer means independency between packets; connection-oriented means dependency.

# Transport-Layer Services

## Connectionless Service

- Assume that a client process has three chunks of messages to send to a server process.
- The chunks are handed over to the connectionless transport protocol in order.
- However, since there is 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

Figure 2.13 *Connectionless service*



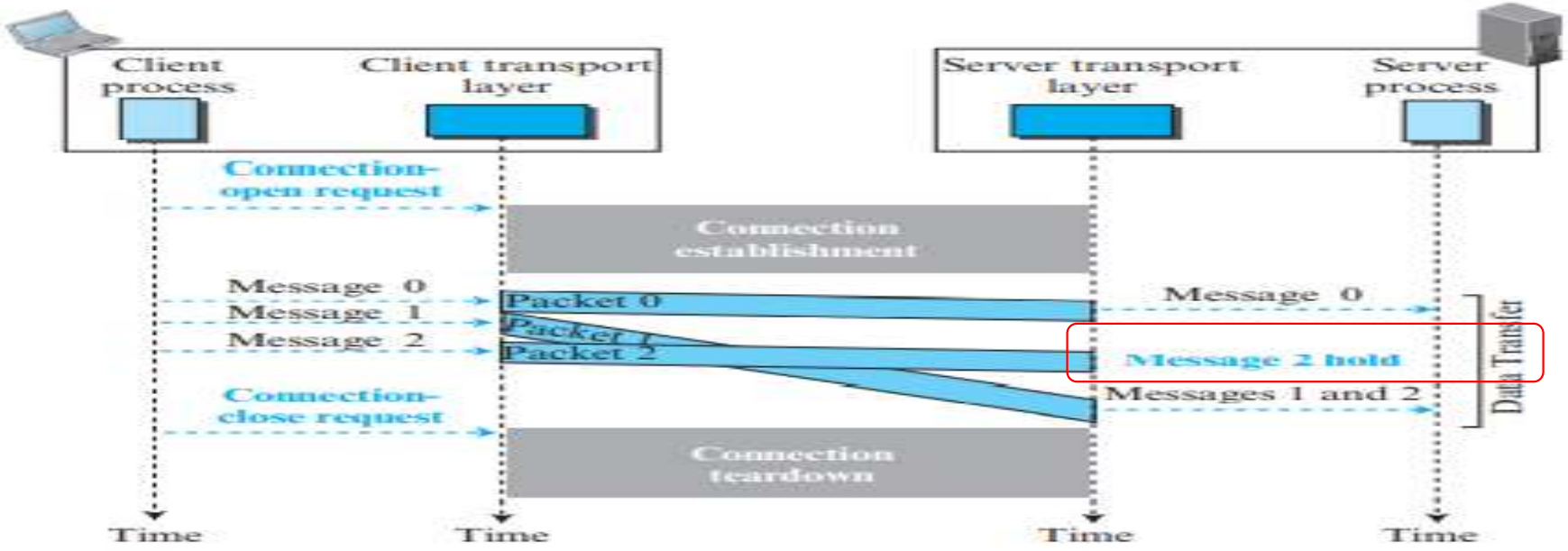

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Transport-Layer Services

## Connection-Oriented Service

- 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 torn down (Figure 2.14).
- We can implement flow control, error control, and congestion control in a connection oriented protocol.

Figure 2.14 *Connection-oriented service*




[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

# Multiplexing and Demultiplexing

- How a receiving host directs an incoming transport-layer segment to the appropriate socket.
- Each transport-layer segment has a set of fields in the segment for this purpose.
- At the receiving end, the transport layer examines these fields to identify the receiving socket and then directs the segment to that socket.
- This job of delivering the data in a transport-layer segment to the correct socket is called demultiplexing.
- The job of gathering data chunks at the source host from different sockets, encapsulating each data chunk with header information (that will later be used in demultiplexing) to create segments, and passing the segments to the network layer is called multiplexing.

# Multiplexing and Demultiplexing

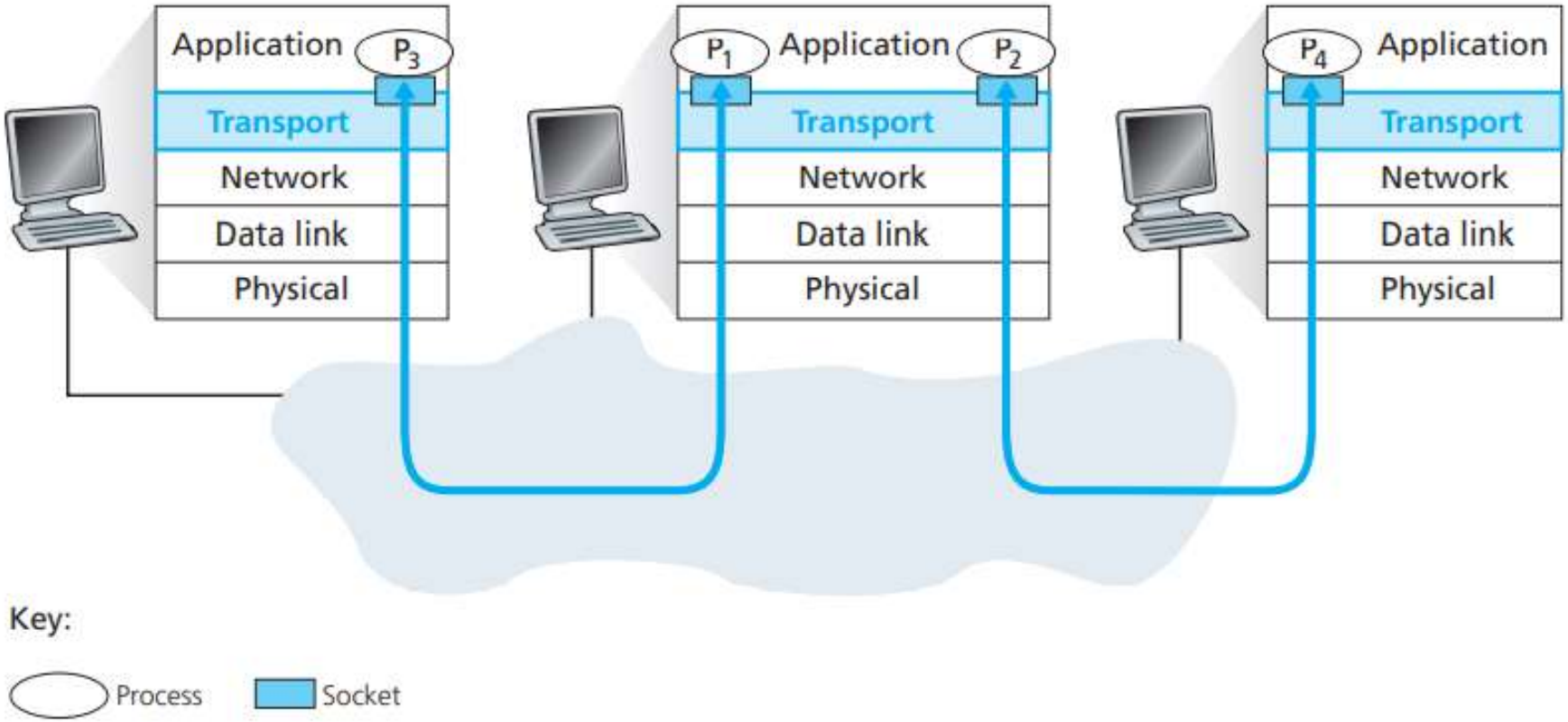



**Figure 2.15** ♦ Transport-layer multiplexing and demultiplexing

James F Kurose and Keith W Ross, "Computer Networking: A Top - Down Approach", Pearson Education; 6 th Edition (2017)

# Multiplexing and Demultiplexing

- Transport-layer multiplexing requires (1) that sockets have unique identifiers, and (2) that each segment have special fields that indicate the socket to which the segment is to be delivered.
- These special fields, illustrated in Figure 2.16, are the source port number field and the destination port number field.

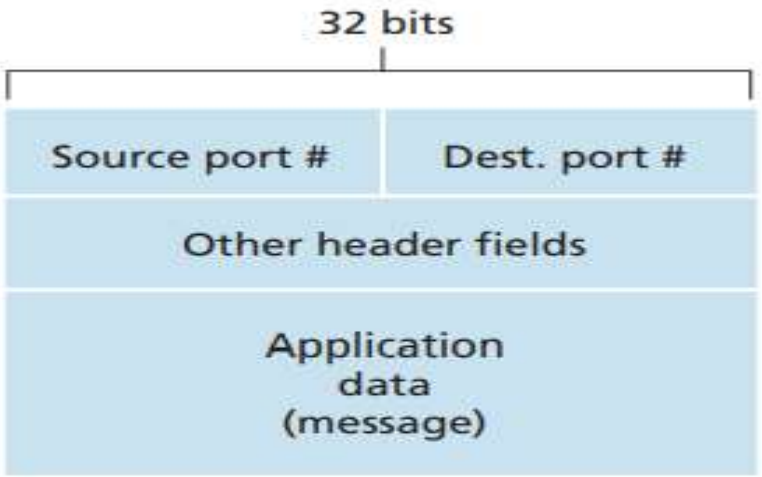



**Figure 2.16** ♦ Source and destination port-number fields in a transport-layer segment

# Multiplexing and Demultiplexing

- Each port number is a 16-bit number, ranging from 0 to 65535.
- The port numbers ranging from 0 to 1023 are called well-known port numbers and are restricted, which means that they are reserved for use by well-known application protocols such as HTTP (which uses port number 80) and FTP (which uses port number 21).
- The list of well-known port numbers is given in RFC 1700 and is updated at http://www.iana.org [RFC 3232].

- Connectionless Multiplexing and Demultiplexing – UDP [REFER]
- Connection-Oriented Multiplexing and Demultiplexing – TCP [REFER]