TSN2201 Computer Networks

LECTURE 11-TRANSPORT LAYER
(Process-to-Process Delivery,
UDP & TCP)

Outline

- Transport Layer Services
- Process-to Process Delivery
- Connectionless Vs Connection-oriented Service
- User Datagram Packet (UDP) Protocol
- Transmission Control Protocol (TCP)
- Stream Control Transmission Protocol (SCTP)

Transport-Layer Services

- Transport layer is located between the network layer and the application layer.
- The transport layer is responsible for providing services to the application layer;
- It receives services from the network layer.

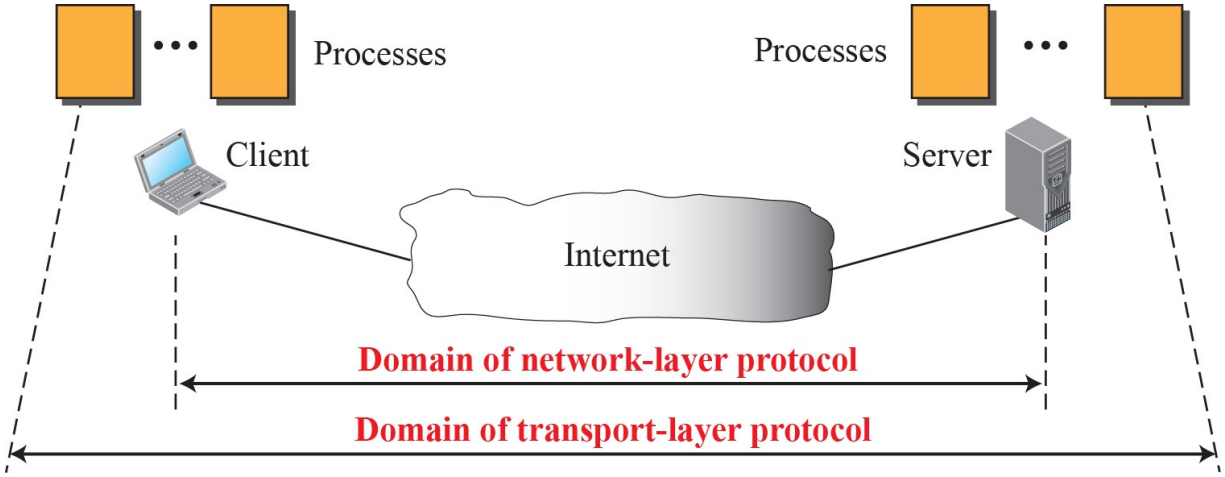
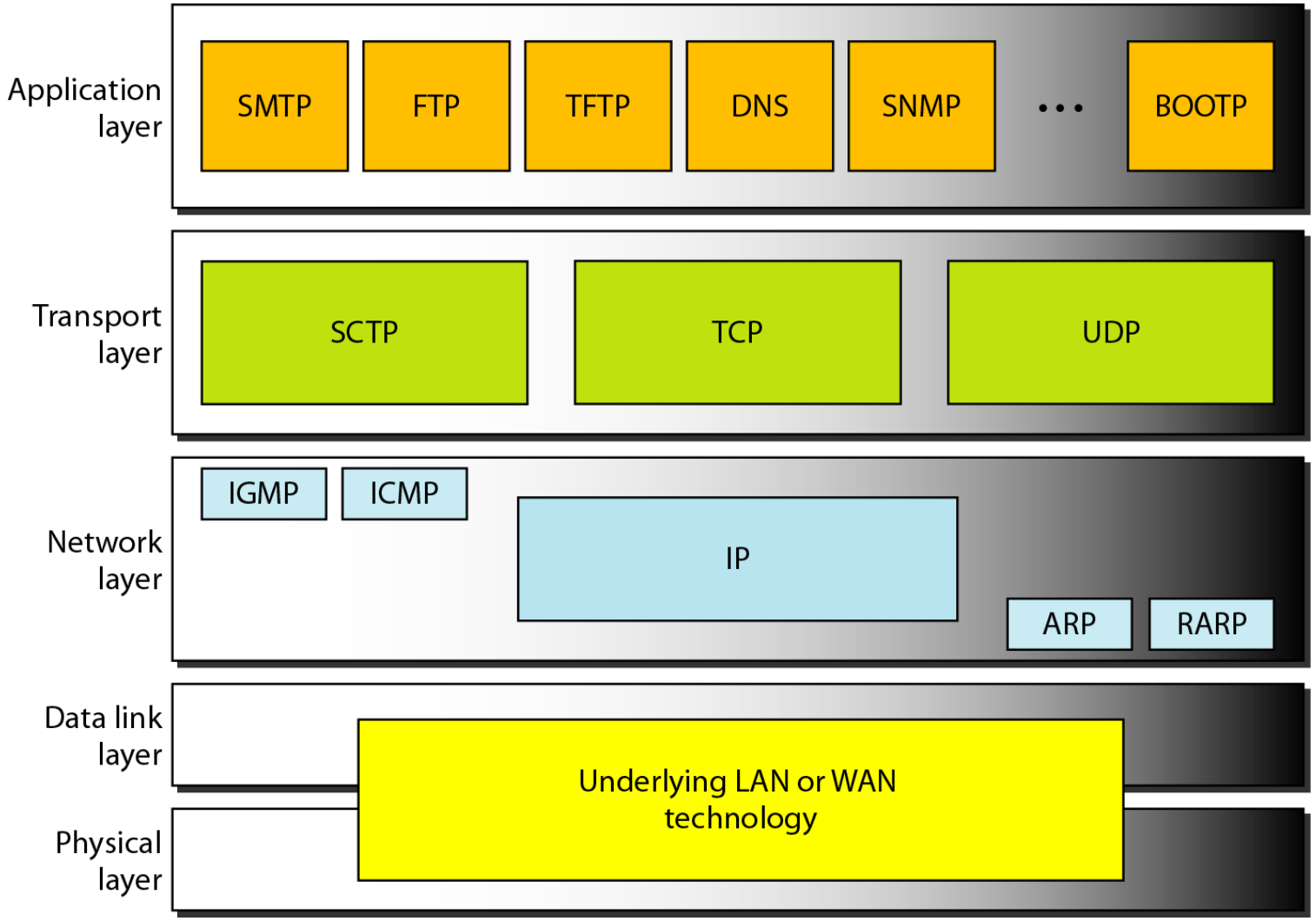


Figure 1: ***Network layer versus transport layer***

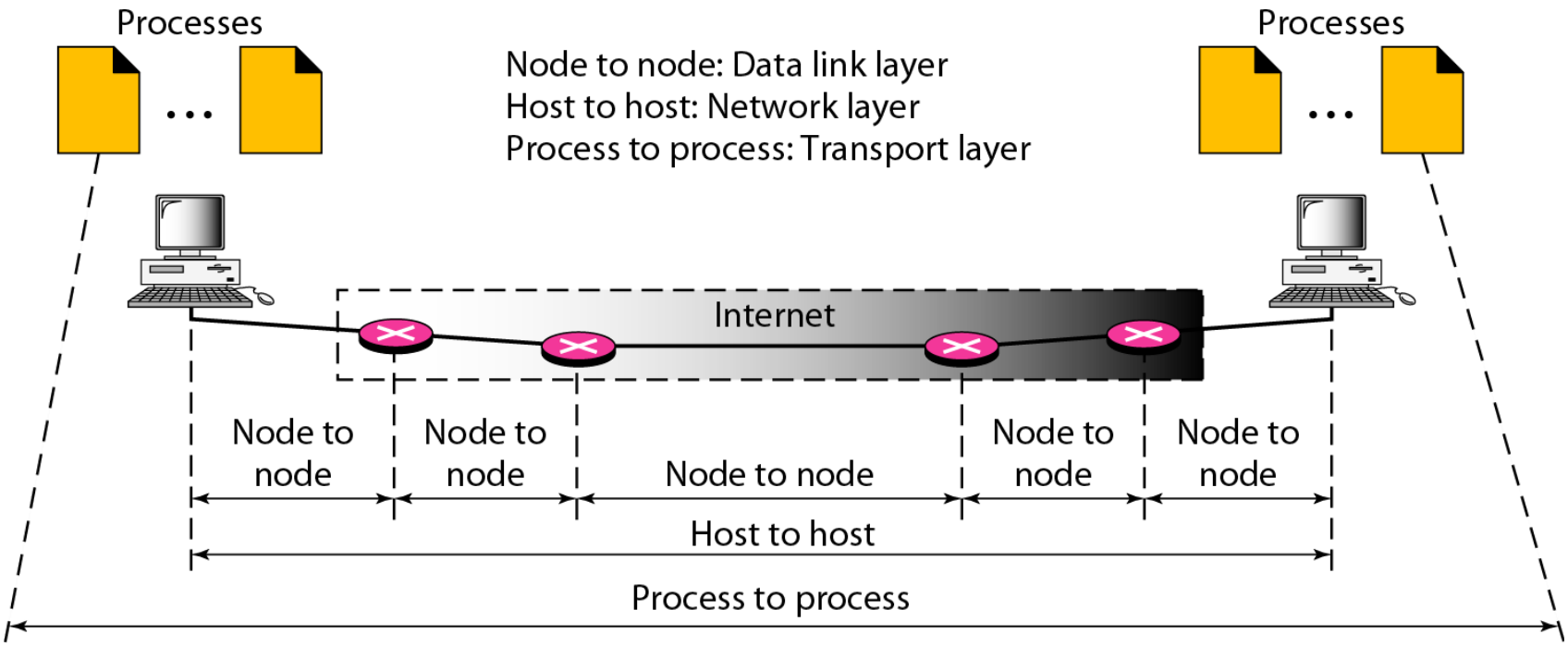
Figure 2: ***Position of UDP, TCP, and SCTP in TCP/IP suite***



PROCESS-TO-PROCESS DELIVERY

- *The transport layer is responsible for process-to-process delivery—the delivery of a packet, part of a message, from one process to another.*
- *Two processes communicate in a client/server relationship.*

Figure 3: *Types of data deliveries*



Addressing

- At the transport layer, we use a transport layer address called a **port number** to choose among multiple processes running in the destination host.

  - The ***destination port number*** is needed for ***delivery***; the ***source port number*** is needed for the ***reply***.

- In the Internet model, the port numbers are 16-bit integers between 0 and 65,535. The client program defines itself with a (temporary) port number, chosen randomly by the transport layer software running on the client host.

- Universal port numbers called well-known port numbers are used for servers.

- In Fig.4, the Daytime client process uses an ephemeral (temporary) port number 52,000 and the Daytime server process uses the well-known (permanent) port number 13.

Figure 4: ***Port numbers***

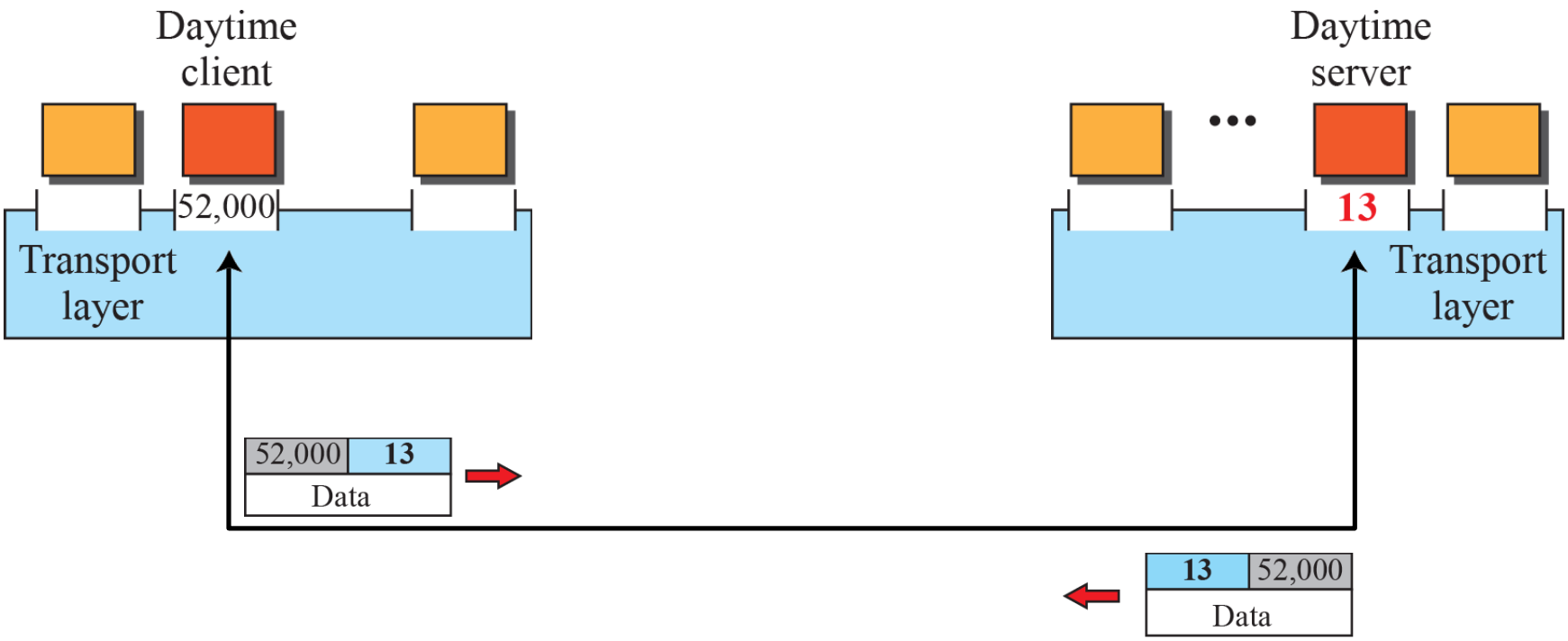
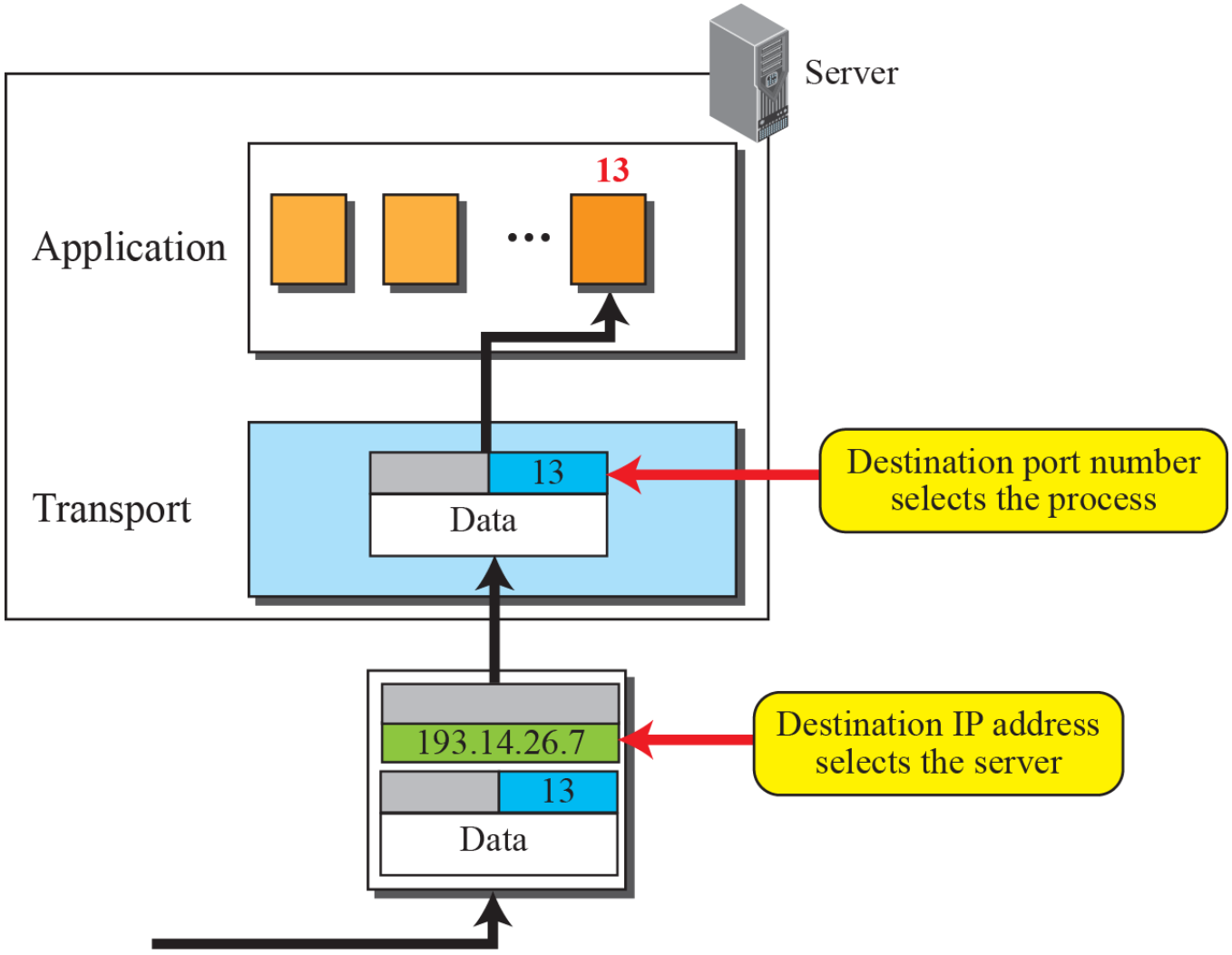


Figure 5: *IP addresses versus port numbers*



ICANN ranges for Port Numbers

- The ICANN (**Internet Corporation for Assigned Names and Numbers)** has divided the port numbers into three ranges: well known, registered and dynamic (private).

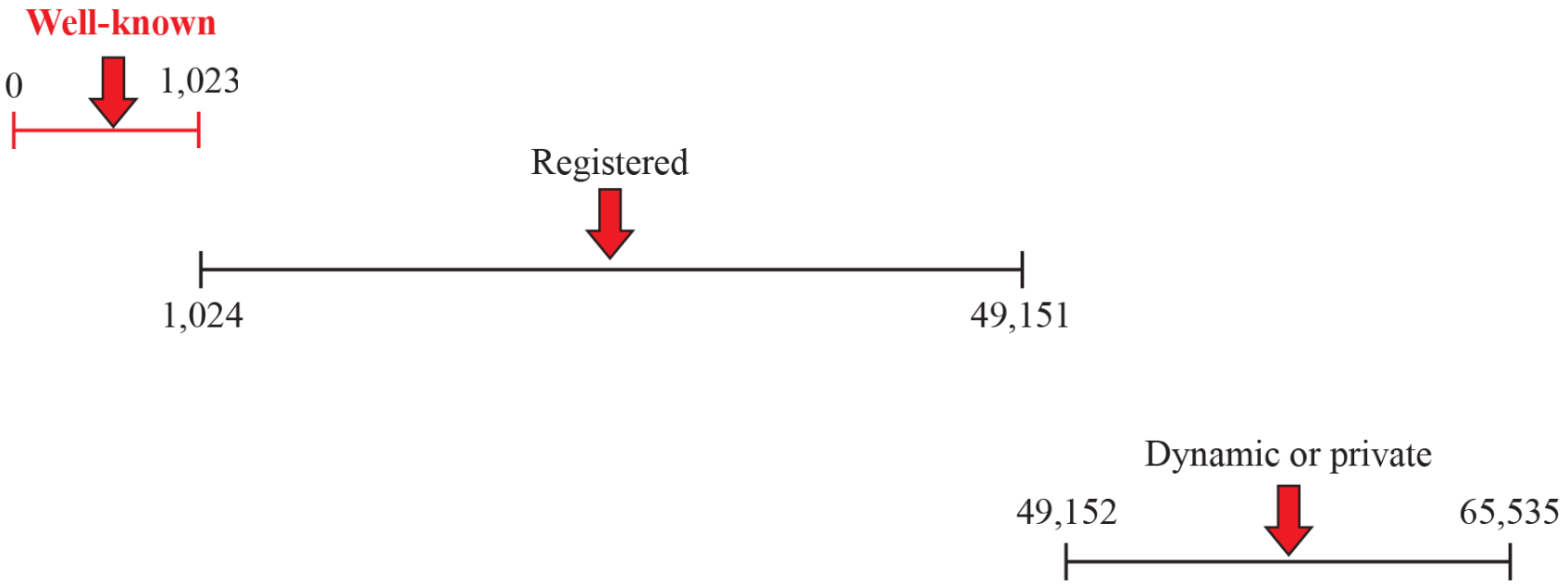


Figure 6: ***ICANN Ranges***

Socket Address

- Process-to-process delivery needs two identifiers: **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.

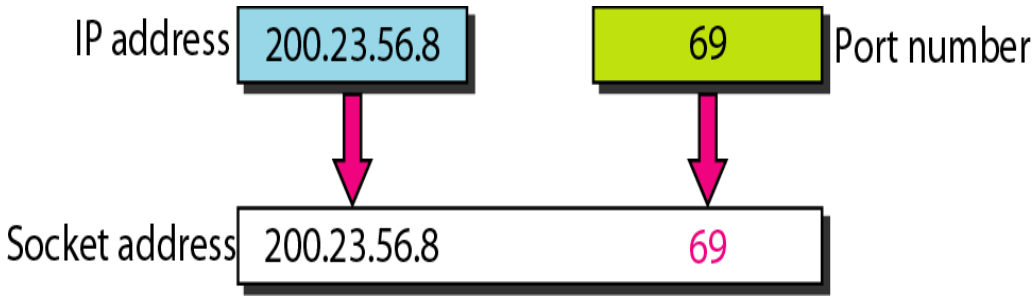


Figure 7: *Socket Address*

Encapsulation and decapsulation

Figure 8: ***Encapsulation and decapsulation***

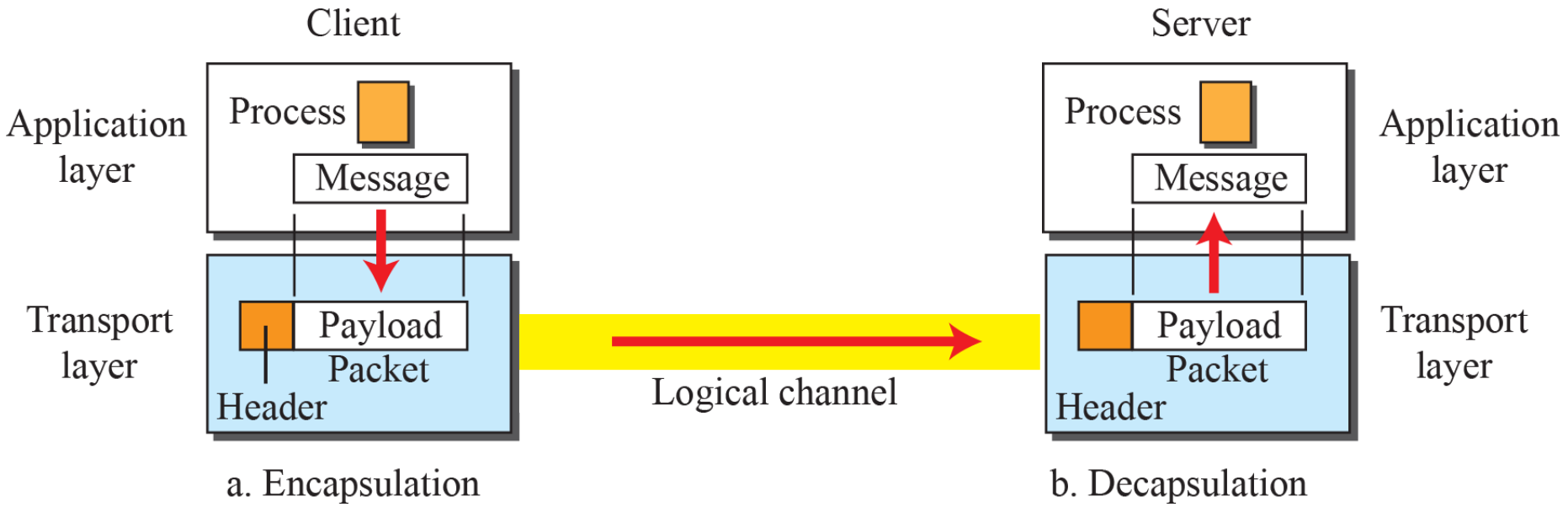
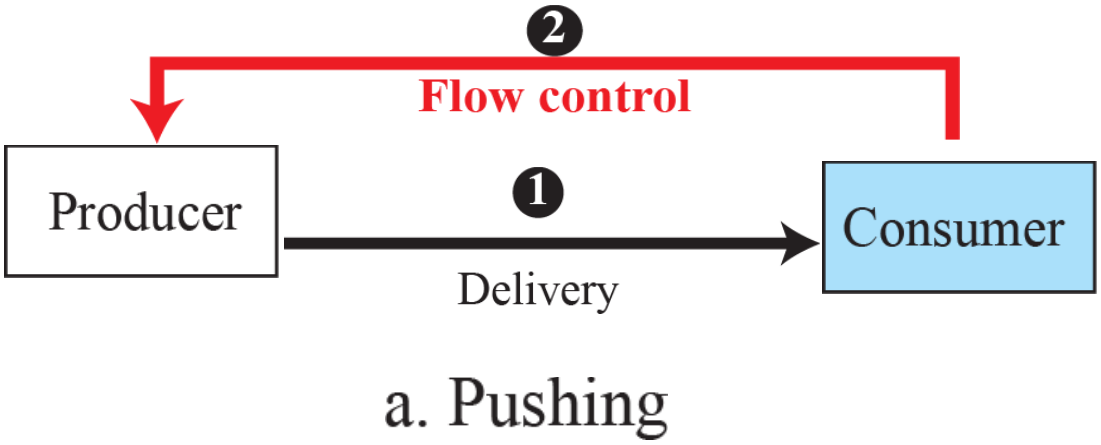
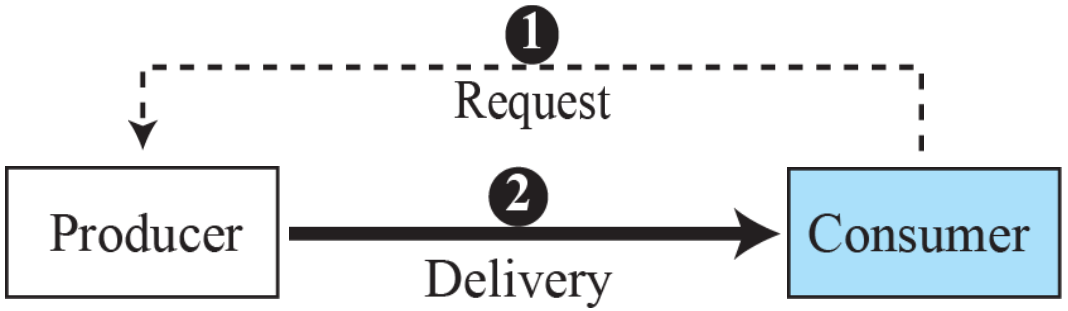


Figure 10 : *Pushing or pulling*



a. Pushing



b. Pulling

Figure 11: ***Flow control at the transport layer***

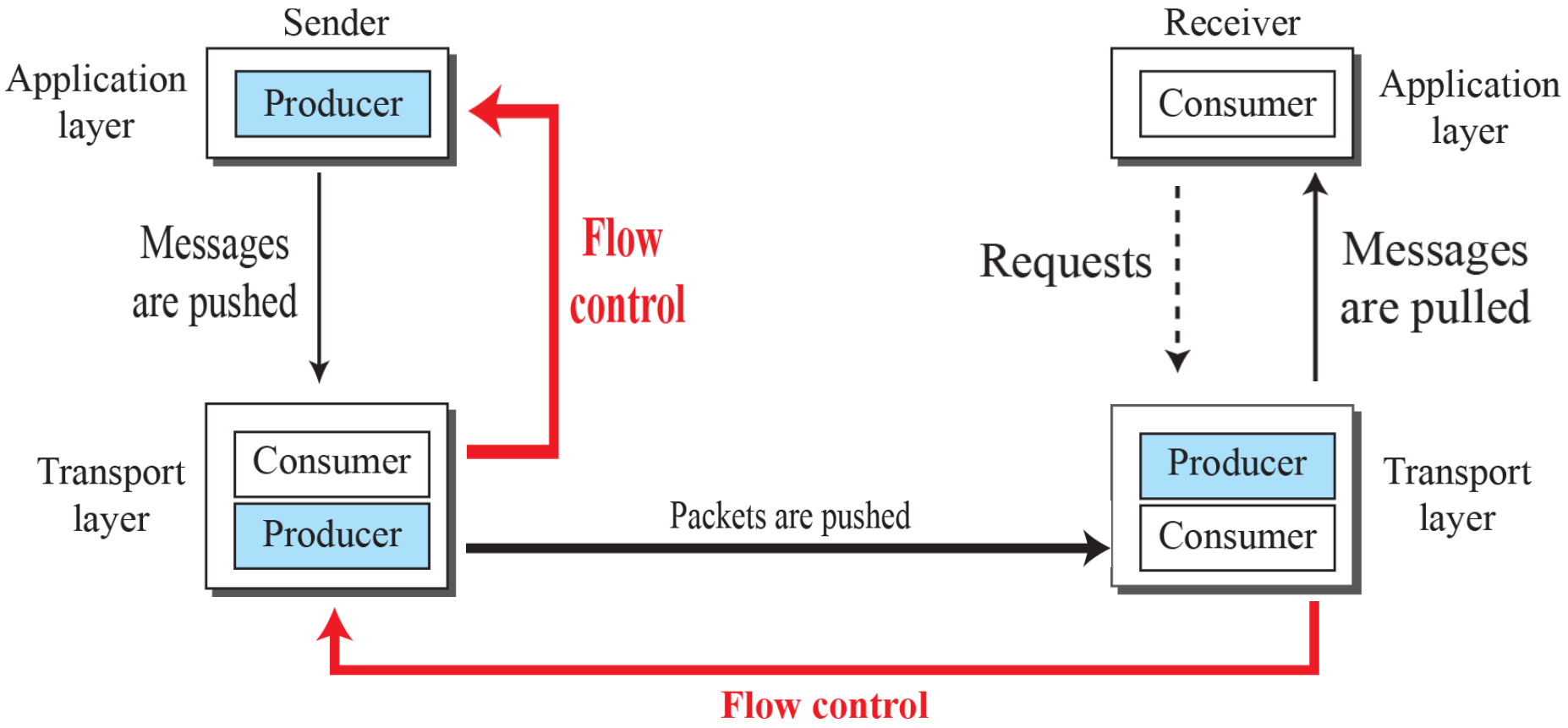
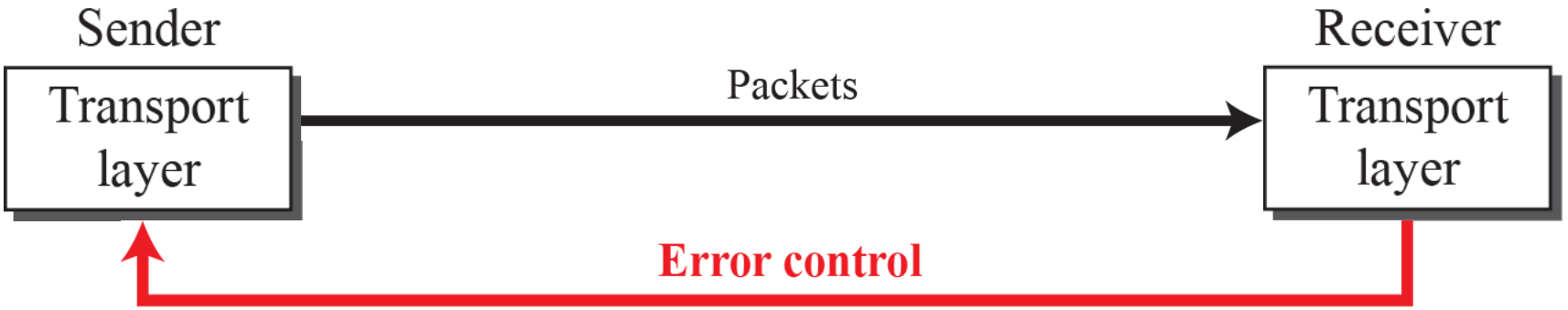


Figure 12: ***Error control at the transport layer***



Connectionless Vs Connection-Oriented

- A transport layer protocol can be connectionless or connection-oriented.
- In a ***connectionless*** service,
  - the packets are sent from one party to another with no need for connection establishment or connection release.
  - The packets are not numbered; they may be delayed or lost or may arrive out of sequence.
  - ***User datagram Protocol*** (***UDP***) is connectionless unreliable protocol.
- In a ***connection-oriented*** service,
  - a connection is first established between the sender and receiver before data transfer.
  - At the end, the connection is released.
  - ***Transmission Control Protocol*** (***TCP***) is a connection-oriented reliable protocol.

Table 1: Some 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 PROTOCOL (UDP)

- *The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol.*
- *It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.*
- *UDP is a very simple protocol using a minimum of overhead (advantage)*

User Datagram (UDP) Format…

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

- Figure 13 shows the format of a user datagram.

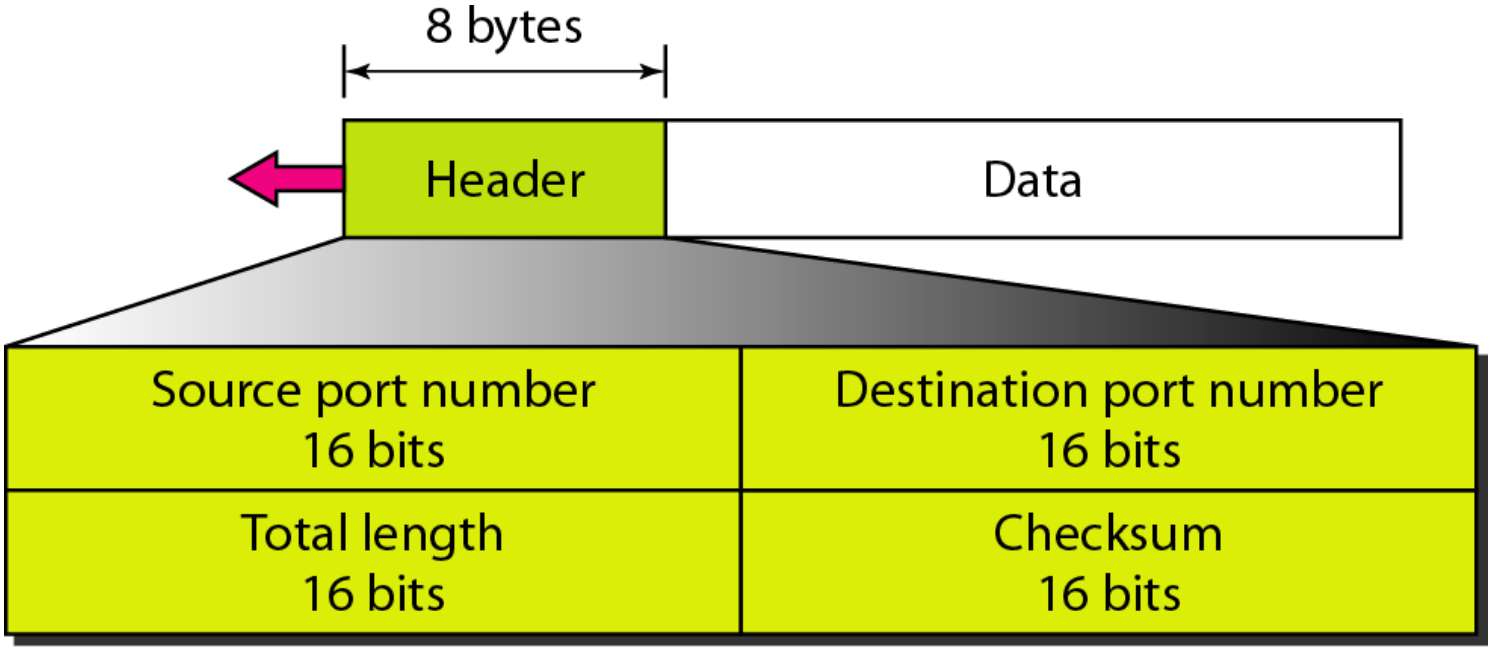
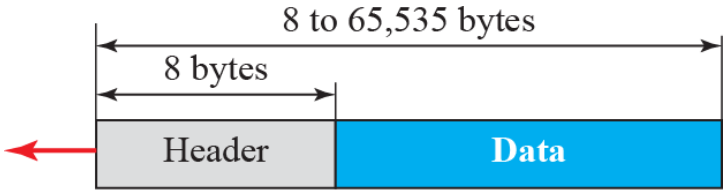
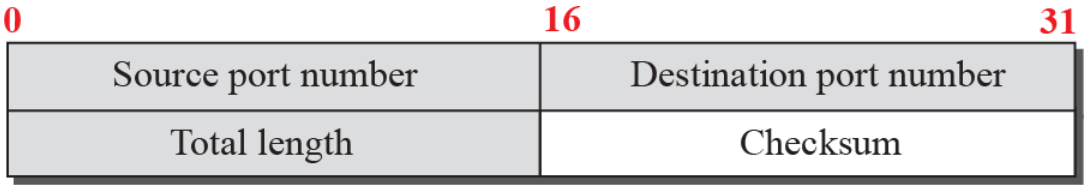


Figure 13 *User datagram format*

User Datagram (UDP) Format



a. UDP user datagram



b. Header format

Figure 14 ***User datagram packet and Header format***

- The first two fields in the header define the source and destination port numbers.
- The third field in the header defines the total length of the user datagram, header plus data. The 16 bits can define a total length of 0 to 65,535 bytes.
- However, the total length needs to be less because a UDP user datagram is stored in an IP datagram with the total length of 65,535 bytes.
- The last field in the header can carry the optional checksum.

Example 1

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

CB84000D001C001C

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

Solution

a. The source port number is the first four hexadecimal digits $(CB84)_{16}$ or $(52100)_{10}$

b. The destination port number is the second four hexadecimal digits $(000D)_{16}$ or $(13)_{10}$.

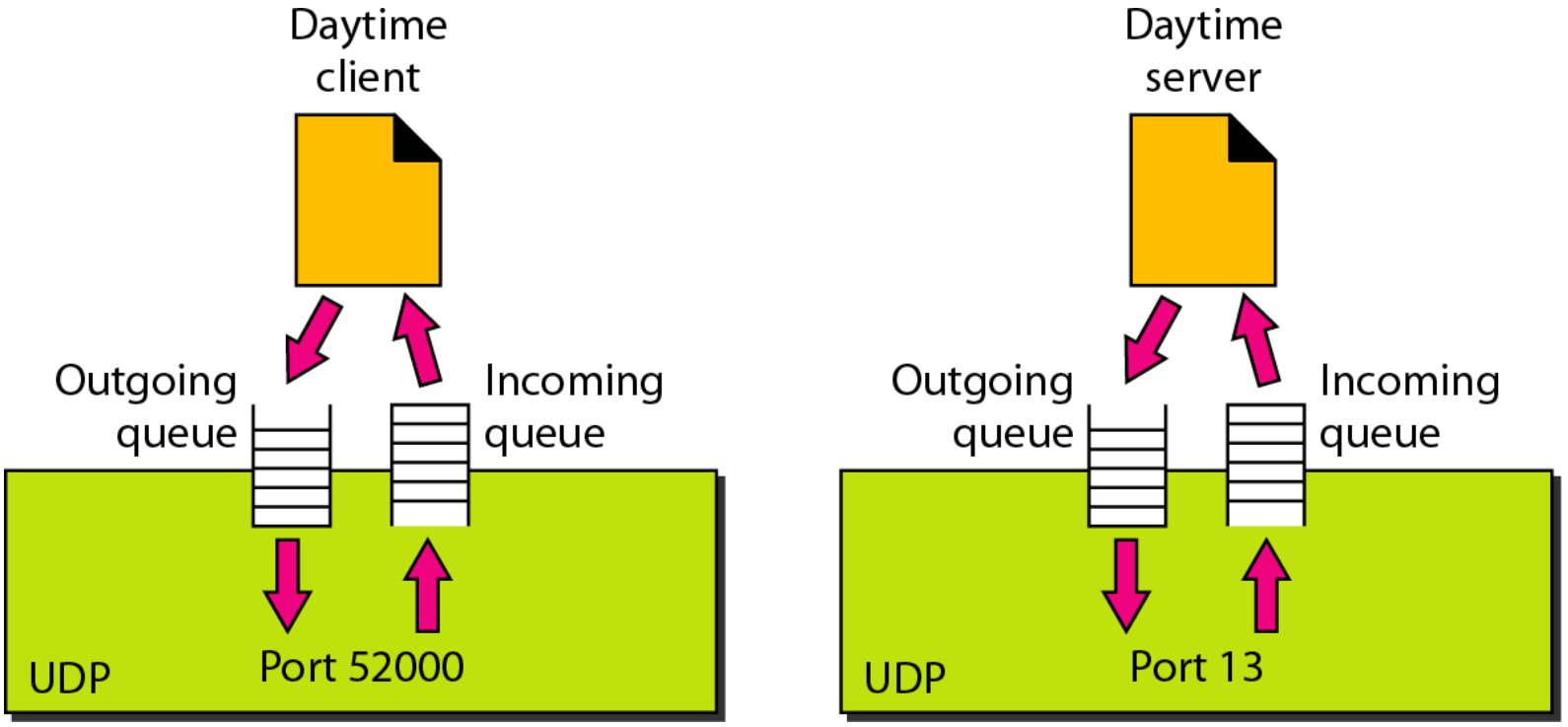
c. The third four hexadecimal digits $(001C)_{16}$ define the length of the whole UDP packet as 28 bytes.

d. The length of the data is the length of the whole packet minus the length of the header, or $28 - 8 = 20$ bytes.

e. Since the destination port number is 13 (well-known port), the packet is from the client to the server.

f. The client process is the Daytime (see Table 1).

Figure 15 *Queues in UDP*



UDP Operation/Applications

- *UDP* is useful for a process that requires simple request-response communication with no concern for flow and error control.
- *UDP* is useful for a process with internal flow and error control mechanisms (such as Trivial File Transfer Control [TFTP]).
- *UDP* is a suitable transport protocol for multicasting.
- *UDP* is used for management processes such as SNMP.
- *UDP* is used in some route updating protocols such as Routing Information protocol (RIP).

Example 2

- A client-server application such as DNS uses the services of UDP because a client needs to send a short request to a server and to receive a quick response from it.

- The request and response can each fit in one user datagram.

- Since only one message is exchanged in each direction, the connectionless feature is not an issue; the client or server does not worry that messages are delivered out of order.

Example 3

- A client-server application such as SMTP, which is used in electronic mail, cannot use the services of UDP because a user can send a long e-mail message, which may include multimedia (images, audio, or video).

- If the application uses UDP and the message does not fit in one single user datagram, the message must be split by the application into different user datagrams.
    - Here the connectionless service may create problems.
    - The user datagrams may arrive and be delivered to the receiver application out of order.
    - The receiver application may not be able to reorder the pieces.

- This means the connectionless service has a disadvantage for an application program that sends long messages.

Example 4

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

TCP

- Transmission Control Protocol (TCP) is a connection-oriented, reliable protocol.
- TCP explicitly defines connection establishment, data transfer, and connection teardown phases to provide a connection-oriented service.
- TCP uses a combination of GBN and SR protocols to provide reliability.

TCP Features

TCP is a connection-oriented protocol; it creates a virtual connection between two TCPs to send data. In addition,

- TCP uses flow and error control mechanisms at the transport level.
- TCP uses parity bits, a checksum, or a cyclic redundancy check (CRC) for tackling transmission errors
- TCP uses sequencing to handle duplicates and out-of-order delivery
- TCP uses retransmission to handle lost packets
- TCP uses some techniques to avoid replay
- TCP uses flow control to prevent data overrun
- TCP uses techniques to avoid congestion

Figure 16: ***Stream delivery***

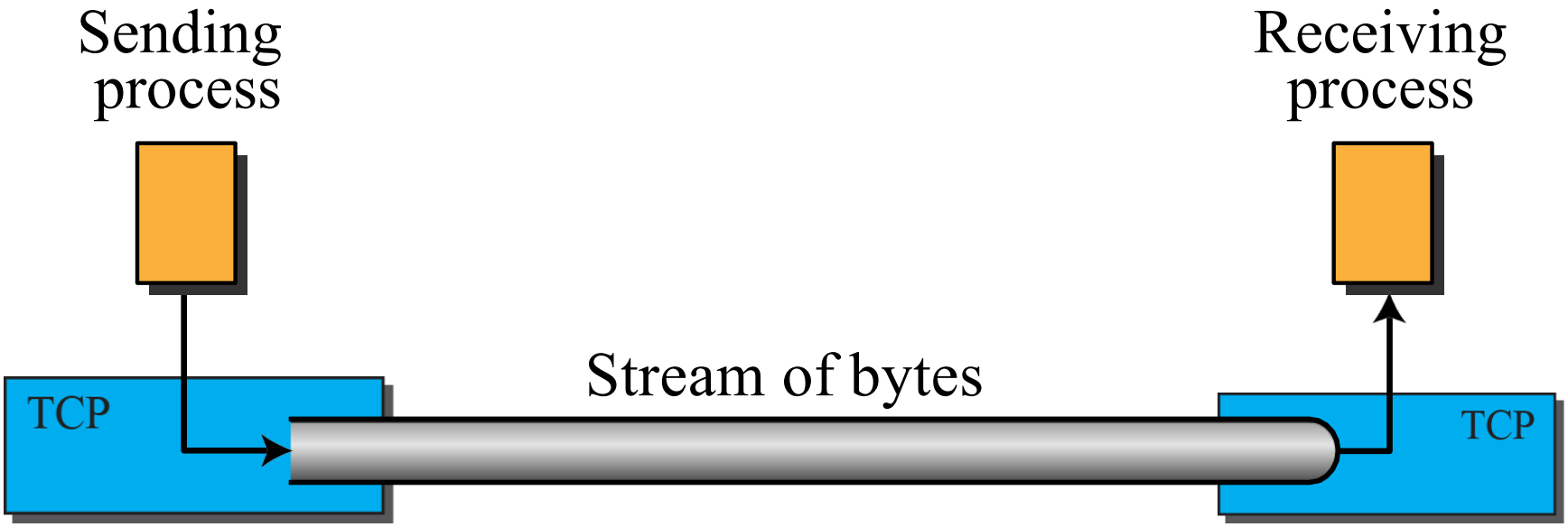


Figure 17: Sending and receiving buffers

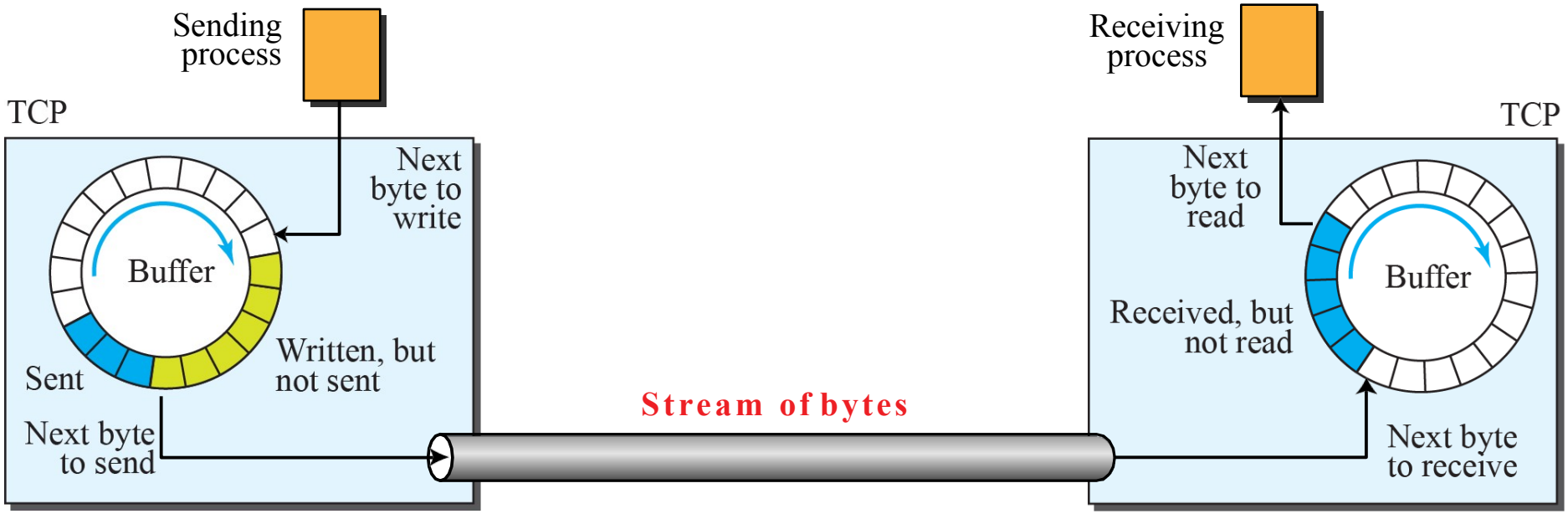
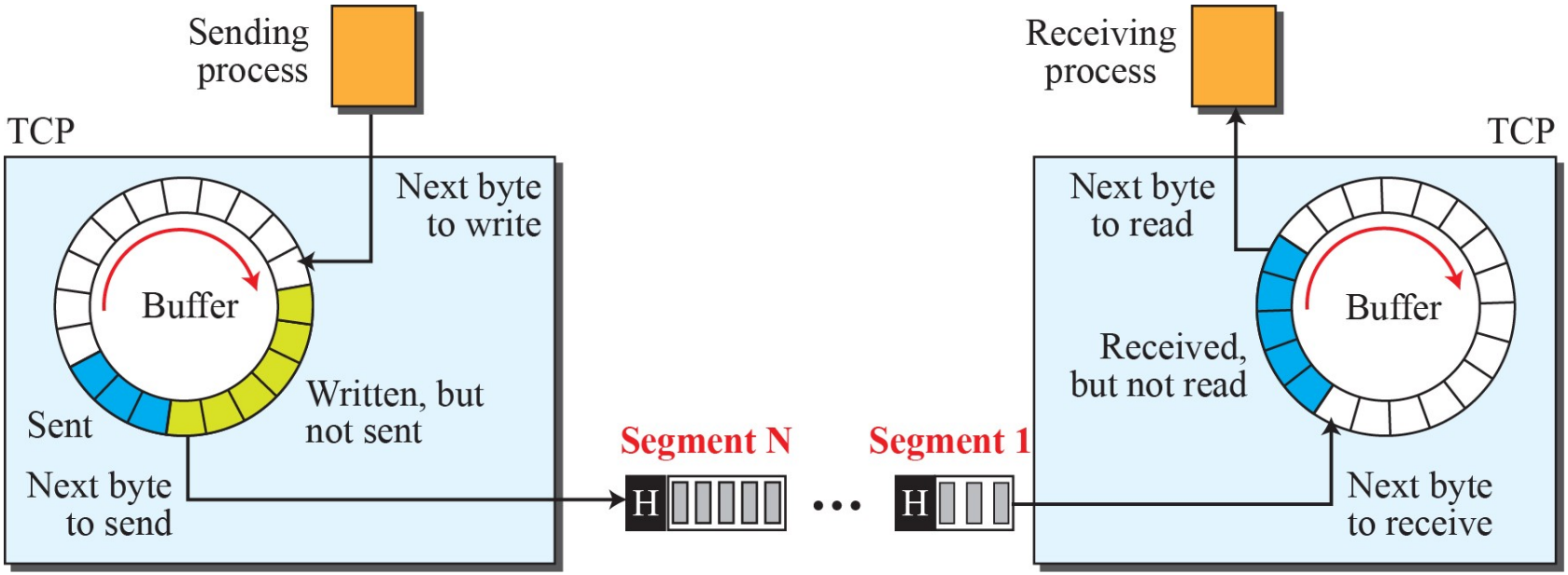


Figure 18: TCP segments



Note

The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.

Example 5

- Suppose a TCP connection is transferring a file of 5000 bytes. The first byte is numbered 10,001. What are the sequence numbers for each segment if data are sent in five segments, each carrying 1000 bytes?

SOLUTION:

The following shows the sequence number for each segment:

Segment 1	→	**Sequence Number: 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)**

Note

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

The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive.

The acknowledgment number is cumulative.

Segment

A packet in TCP is called a segment.

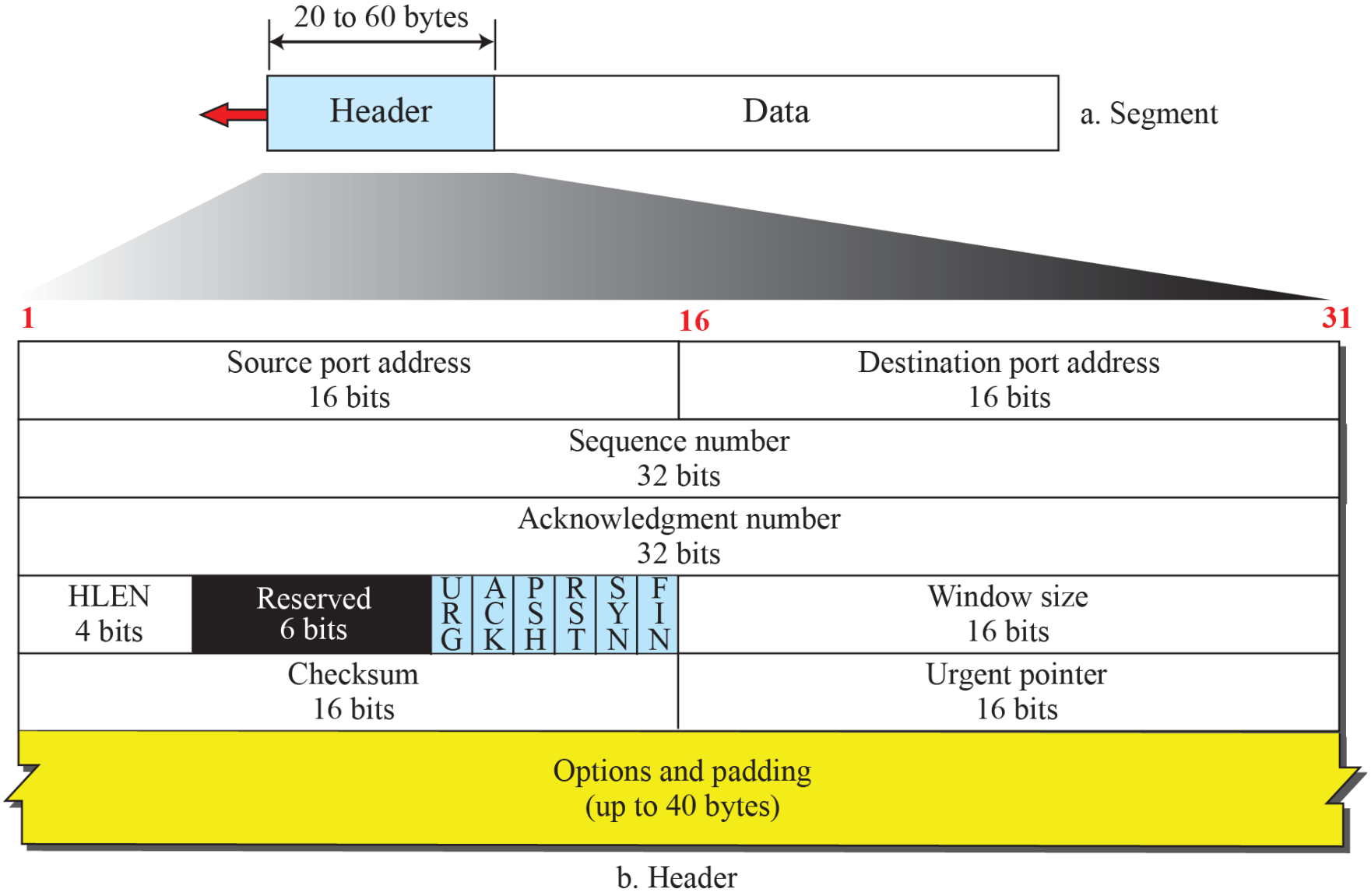
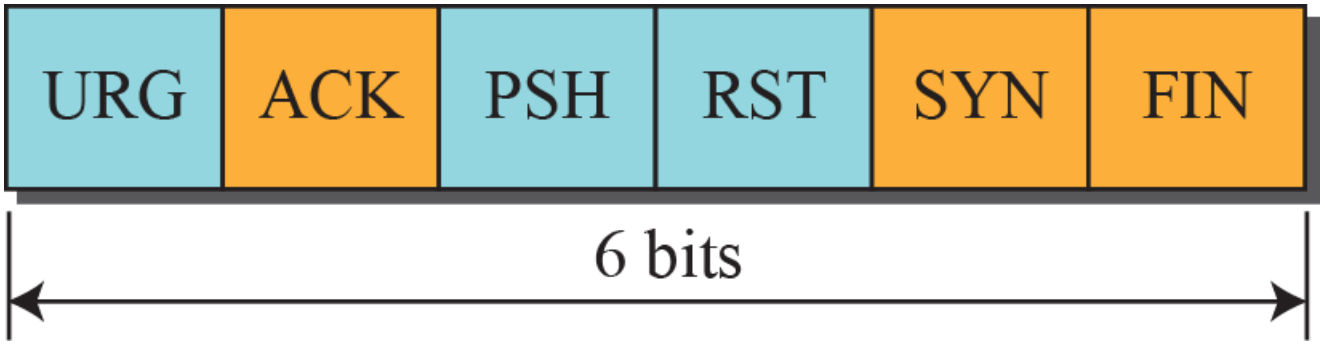


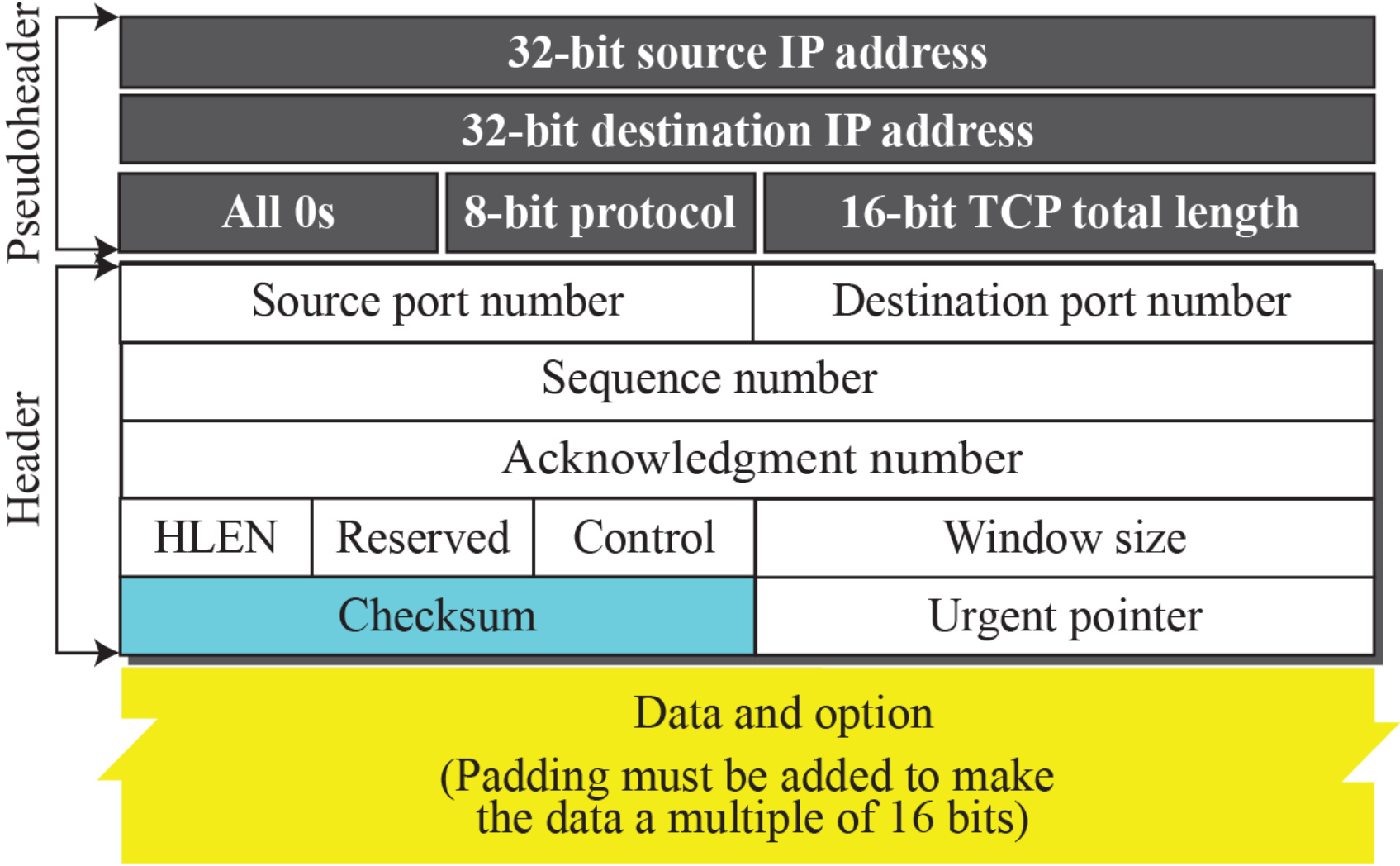
Figure 19: TCP segment format

Figure 20: ***Control field***



URG: Urgent pointer is valid
ACK: Acknowledgment is valid
PSH: Request for push
RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: Terminate the connection

Figure 21: ***Pseudoheader added to the TCP datagram***



TCP Connection

- TCP is connection-oriented.
  - It establishes a virtual path between source and destination.
  - All the segments belonging to a message are then sent over this virtual path.
- In TCP, transmission requires three phases:
  - connection establishment,
  - data transfer, and
  - connection termination.
- TCP transmits data in full-duplex mode.
  - The connection establishment in TCP is called three-way handshaking.
- After connection is established, bidirectional data transfer can take place.
  - The client and server can both send data and acknowledgements.
- Any of two parties involved in exchanging data (client or server) can close the connection, although it is usually initiated by the client.

Figure 22: ***Connection establishment using three-way handshaking***

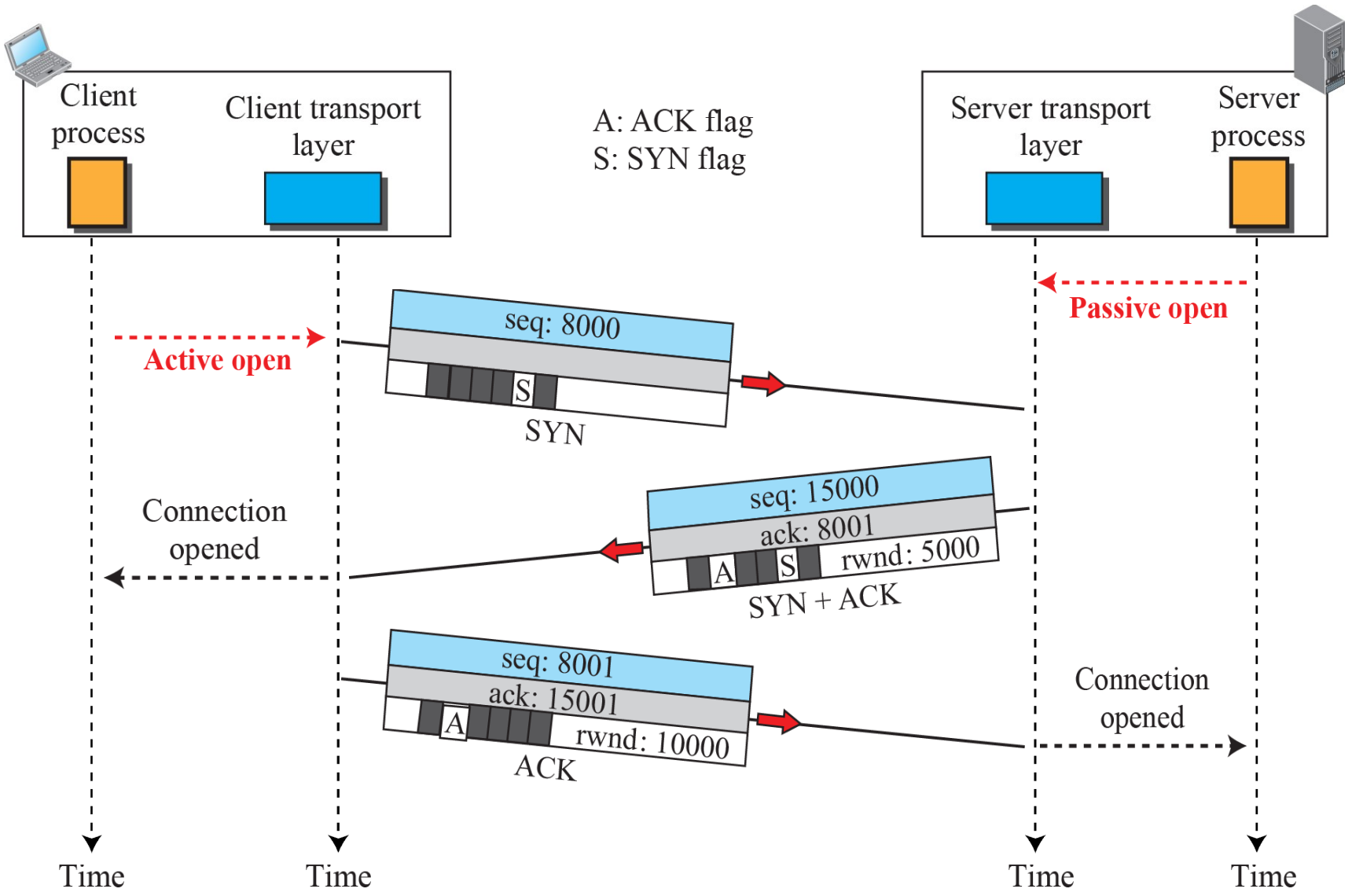


Figure 23: ***Data transfer***

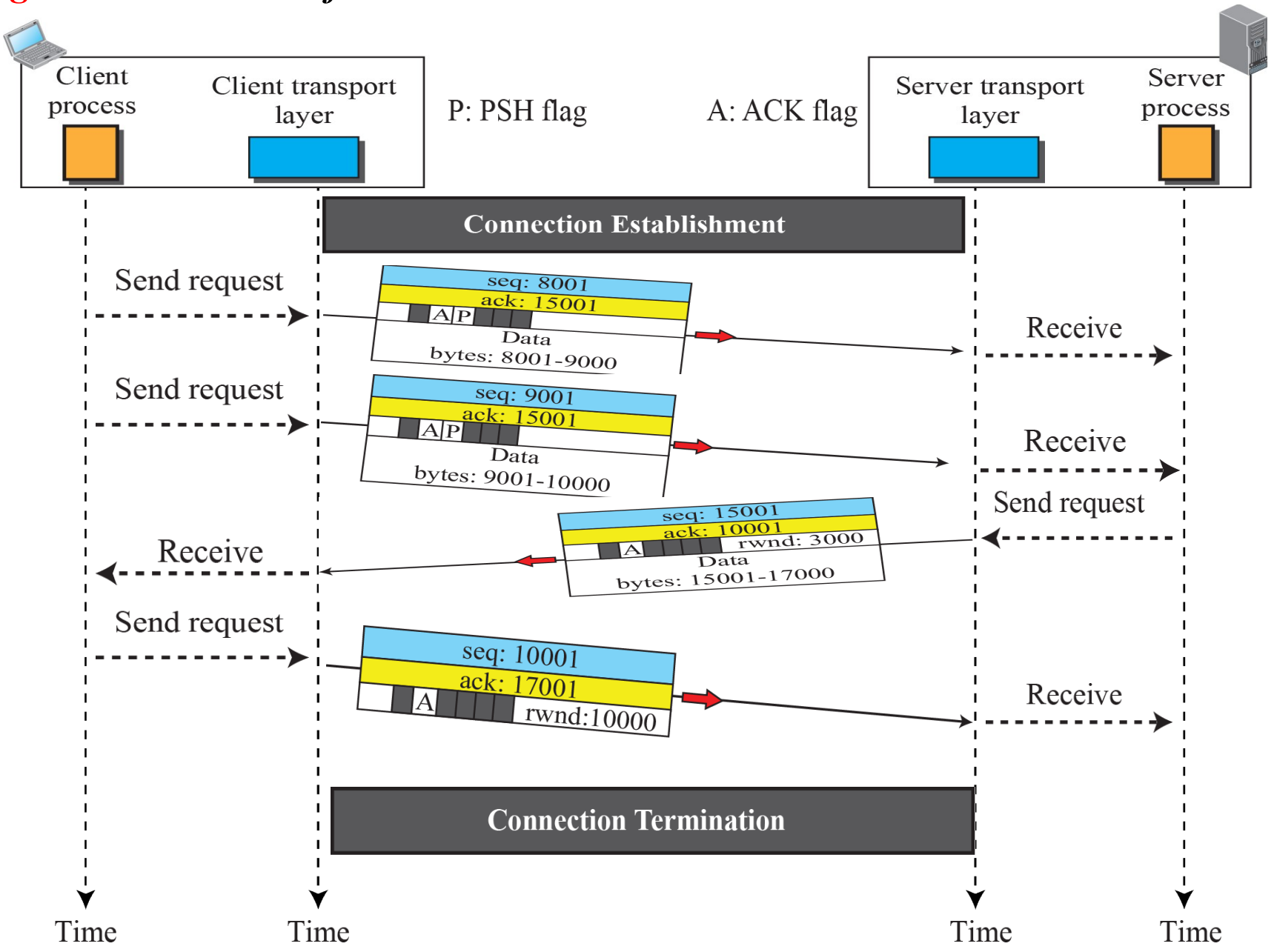
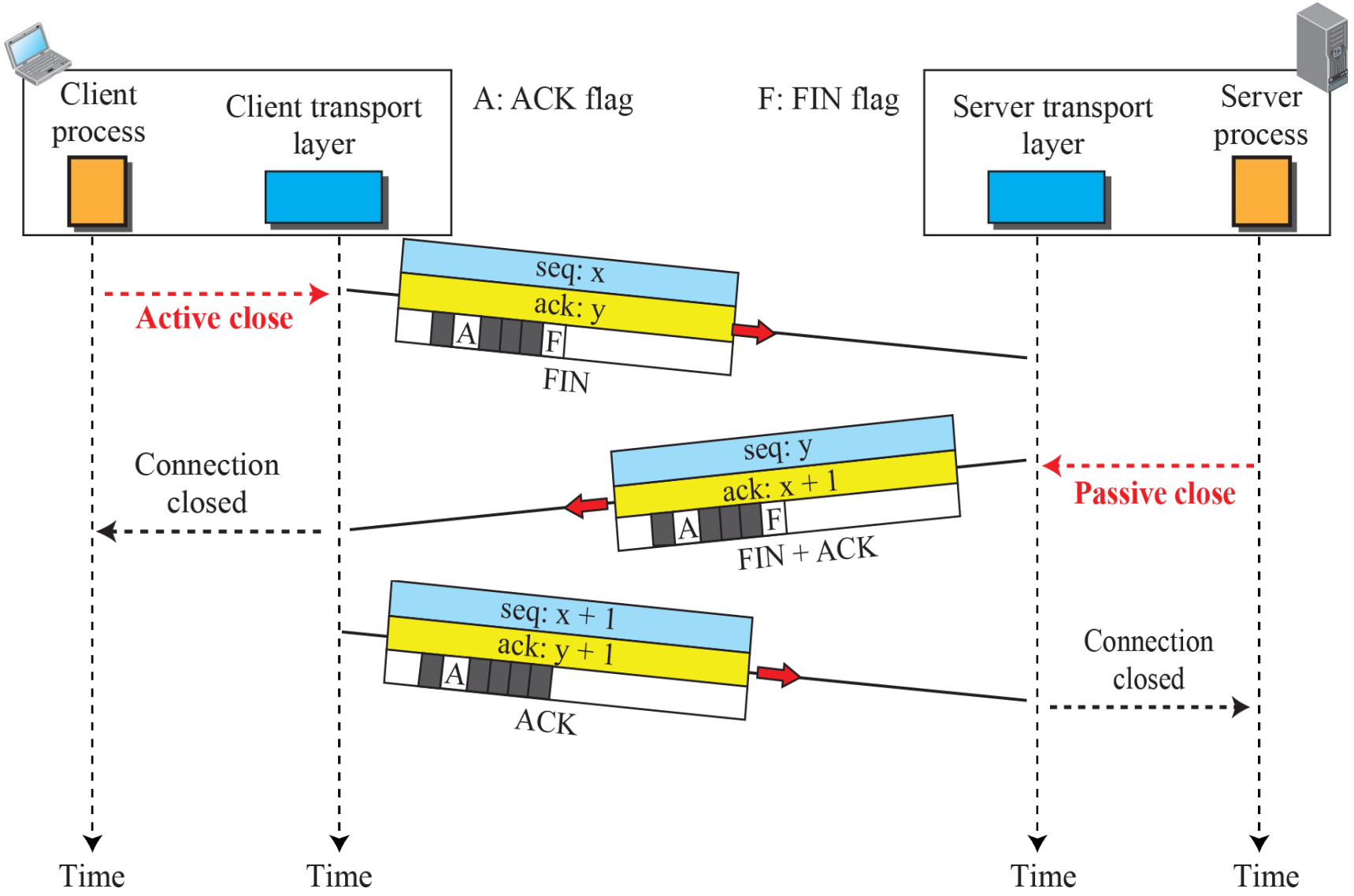


Figure 24: ***Connection termination using three-way handshaking***

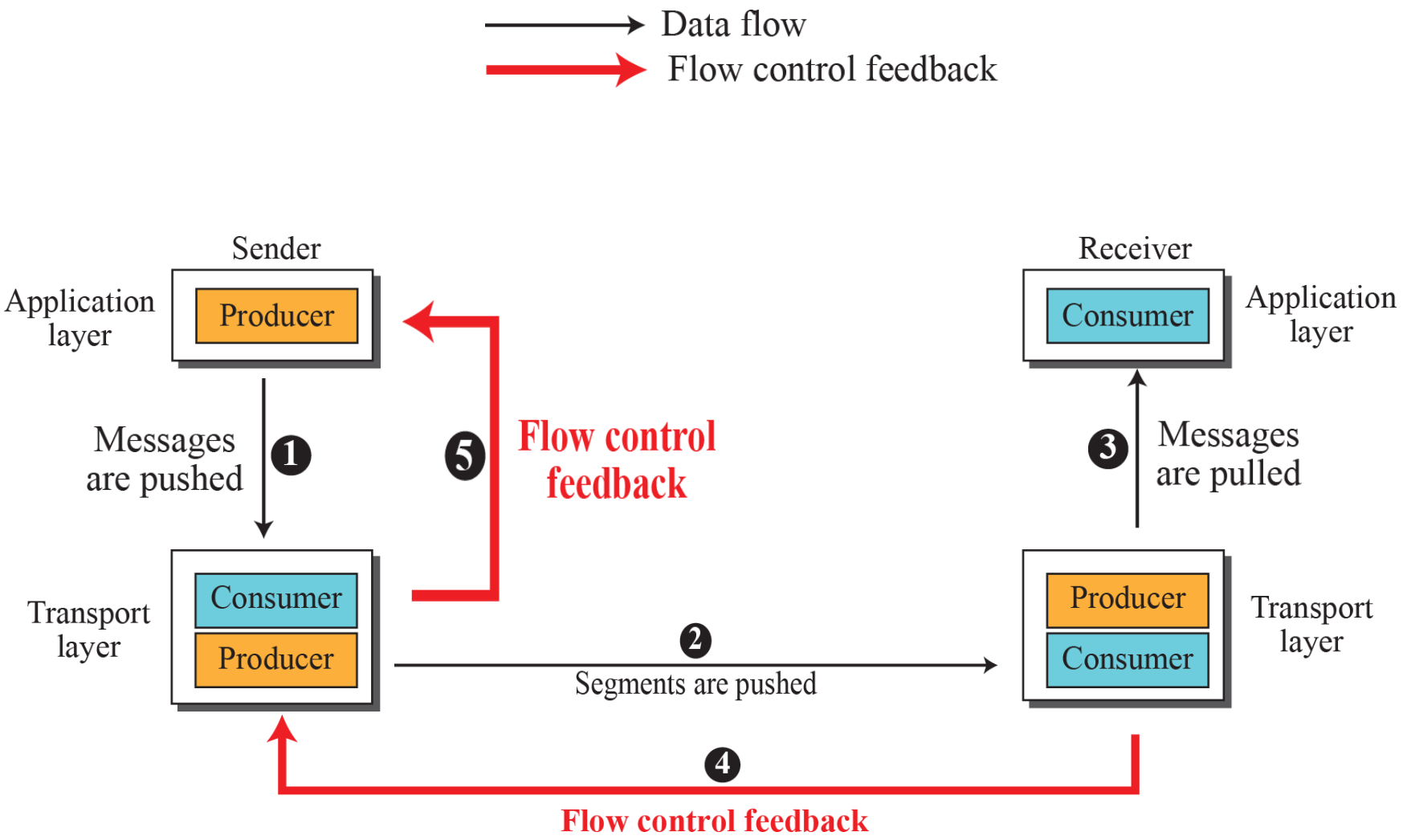


Flow Control

- ***Flow control balances the rate a producer creates data with the rate a consumer can use the data.***

- ***TCP separates flow control from error control.***

Figure 25: ***Data flow and flow control feedbacks in TCP***



Error Control

- ***TCP is a reliable transport-layer protocol.***

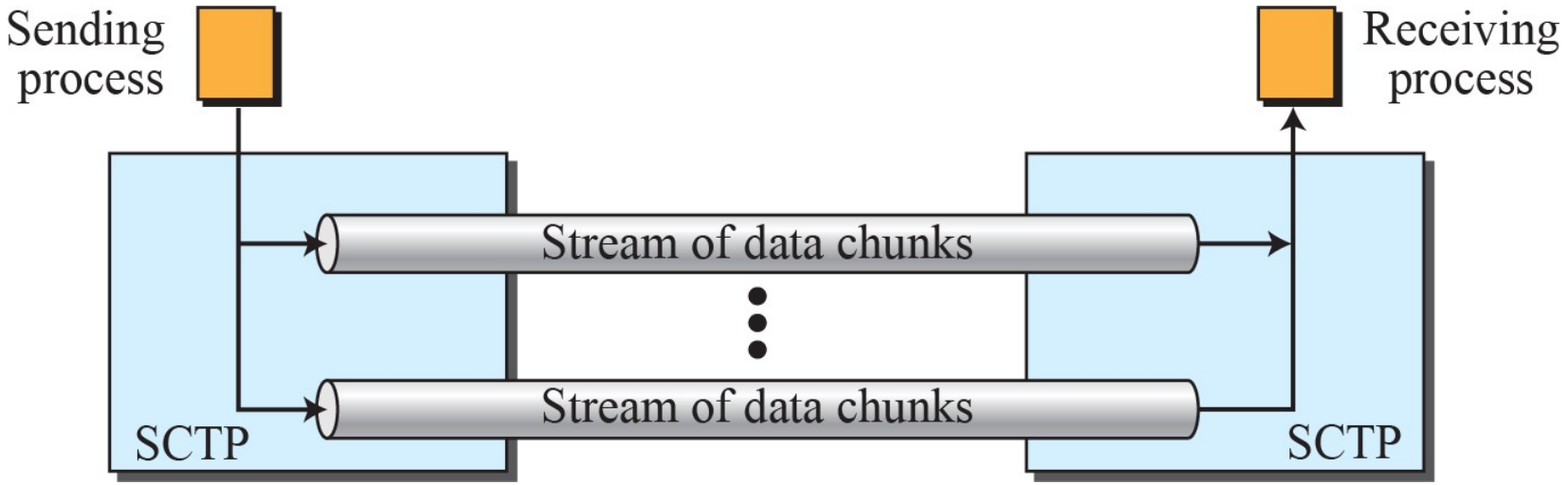
- ***This means that an application program that delivers a stream of data to TCP relies on TCP to deliver the entire stream to the application program on the other end in order, without error, and without any part lost or duplicated.***

SCTP

Stream Control Transmission Protocol (SCTP) is a new transport-layer protocol designed to combine some features of UDP and TCP in an effort to create a protocol for multimedia communication.

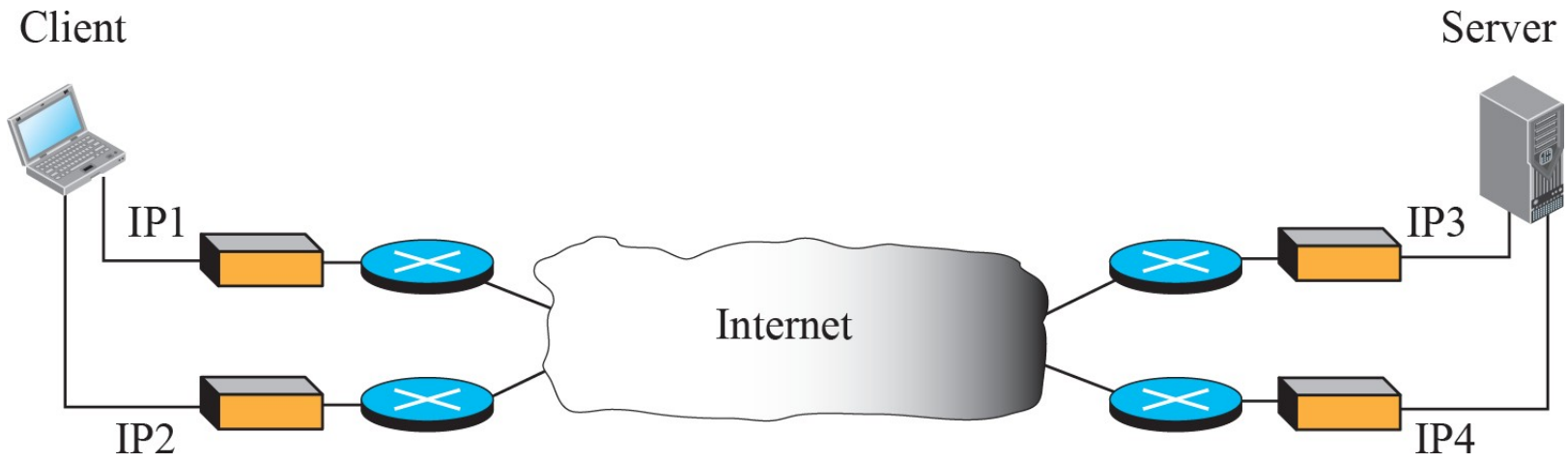
SCTP Services…

Figure 26 : ***Multiple-stream concept***



SCTP Services

Figure 27 : ***Multihoming concept***



Acknowledgement

Slides adapted from the text book,

B.A.Forouzan, "Data Communications and Networking" Fifth Edition, McGraw-Hill