TCP

Computer Networks(CS31204)

Prof. Sudip Misra

Department of Computer Science and Engineering

Indian Institute of Technology Kharagpur

Email: smisra@sit.iitkgp.ernet.in

Website: http://cse.iitkgp.ac.in/~smisra/

Research Lab: cse.iitkgp.ac.in/~smisra/swan/

Introduction

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

Services

- Process-to-Process Communication
- Stream Delivery Service
- Full-Duplex Communication
- Multiplexing and Demultiplexing
- Connection-Oriented Service
- Reliable Service

Numbering System in TCP

- TCP software keeps track of the segments being transmitted or received, there is no field for a segment number value in the segment header.
- There are two fields, called the *sequence number* and the *acknowledgment number*. These two fields refer to a byte number.
- TCP numbers all data bytes (octets) that are transmitted in a connection.
- Numbering is independent in each direction. When TCP receives bytes of data from a process, TCP stores them in the sending buffer and numbers them.
- The numbering does not necessarily start from 0.
- TCP chooses an arbitrary number between 0 and 232 - 1 for the number of the first byte.

Sequence Number

After the bytes have been numbered, TCP assigns a sequence number to each segment that is being sent. The sequence number, in each direction, is defined as follows:

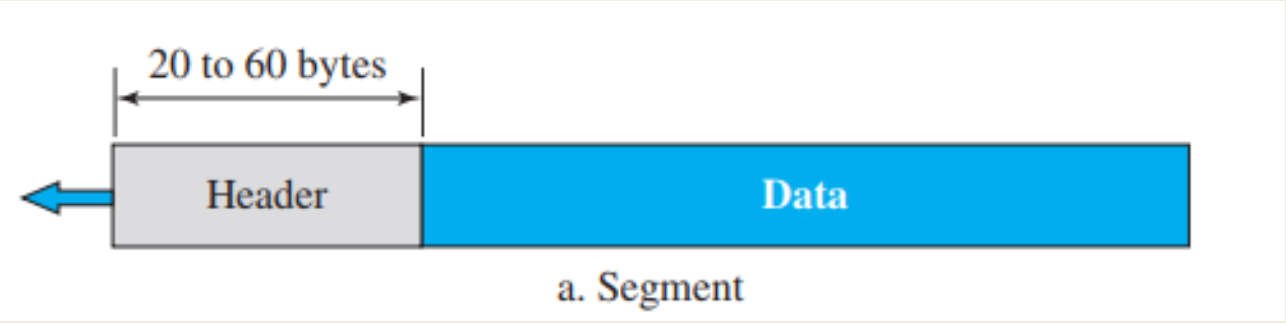
- The sequence number of the first segment is the ISN (initial sequence number), which is a random number.
- The sequence number of any other segment is the sequence number of the previous segment plus the number of bytes (real or imaginary) carried by the previous segment.

Acknowledgement Number

- Each party also uses an acknowledgment number to confirm the bytes it has received.
- The acknowledgment number defines the number of the next byte that the party expects to receive.
- In acknowledgment number is cumulative, which means that the party takes the number of the last byte that it has received, safe and sound, adds 1 to it, and announces this sum as the acknowledgment number.
- The term *cumulative* here means that if a party uses 5643 as an acknowledgment number, it has received all bytes from the beginning up to 5642.
- The party has received 5642 bytes, because the first byte number does not have to be 0.

Segment

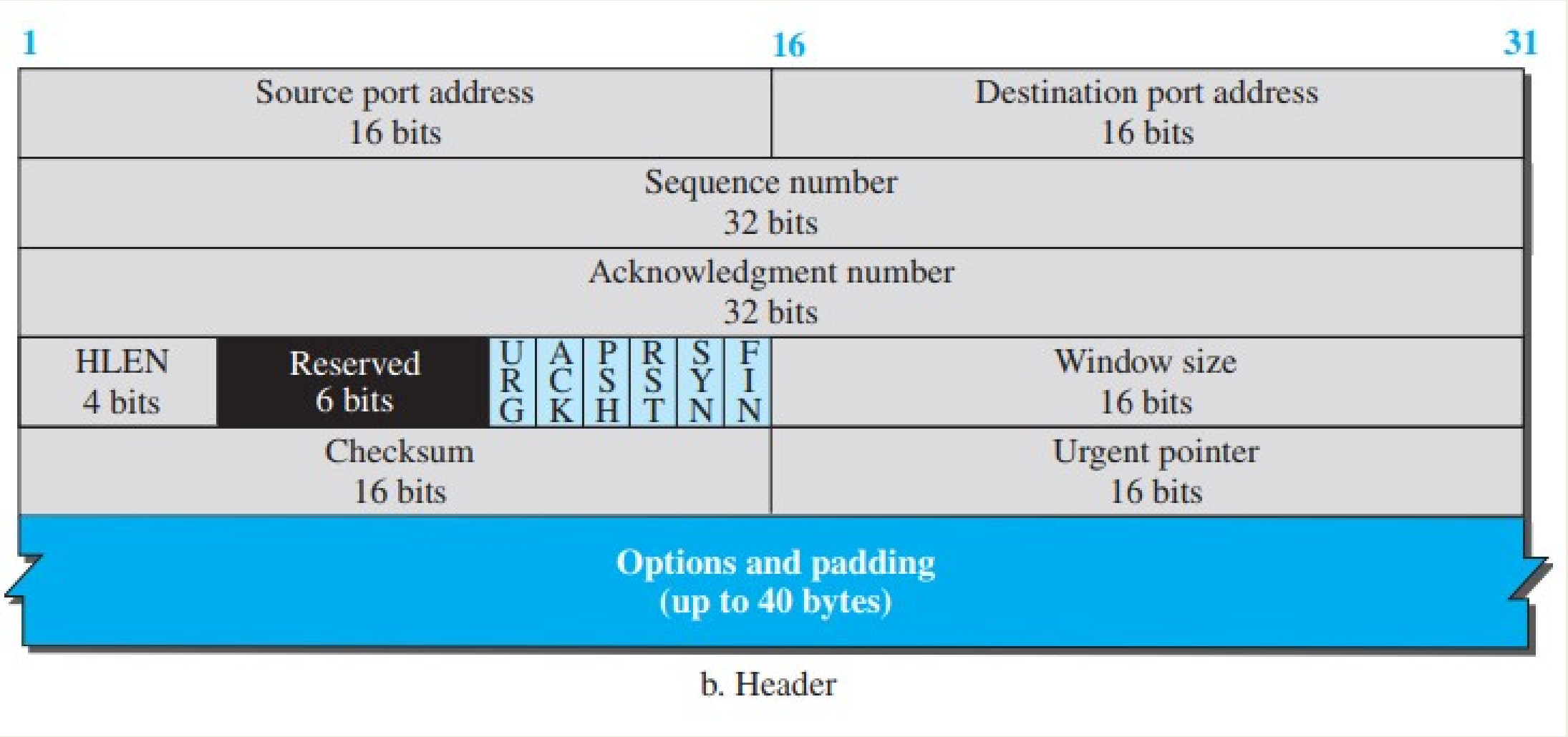
- The segment consists of a header of 20 to 60 bytes, followed by data from the application program.
- The header is 20 bytes if there are no options and up to 60 bytes if it contains options.



a. Segment

Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

TCP Header



b. Header

Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

TCP Header: Contd.

Source port address.

- This is a 16-bit field that defines the port number of the application program in the host that is sending the segment.

Destination port address.

- This is a 16-bit field that defines the port number of the application program in the host that is receiving the segment.

Sequence number.

- This 32-bit field defines the number assigned to the first byte of data contained in this segment.
- TCP is a stream transport protocol.
- To ensure connectivity, each byte to be transmitted is numbered.
- The sequence number tells the destination which byte in this sequence is the first byte in the segment.
- During connection establishment (discussed later) each party uses a random number generator to create an **initial sequence number** (ISN), which is usually different in each direction.

TCP Header: Contd.

Acknowledgment number.

- This 32-bit field defines the byte number that the receiver of the segment is expecting to receive from the other party.
- If the receiver of the segment has successfully received byte number x from the other party, it returns x + 1 as the acknowledgment number.
- Acknowledgment and data can be piggybacked together.

Header length.

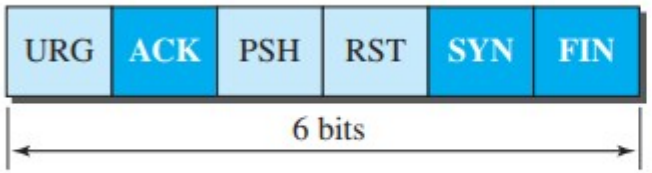
- This 4-bit field indicates the number of 4-byte words in the TCP header.
- The length of the header can be between 20 and 60 bytes.
- Therefore, the value of this field is always between 5 (5 × 4 = 20) and 15 (15 × 4 = 60).

TCP Header: Contd.

Control.

- This field defines 6 different control bits or flags.

- These bits enable flow control, connection establishment and termination, connection abortion, and the mode of data transfer in TCP.



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

Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

TCP Header: Contd.

Window size.

- This field defines the window size of the sending TCP in bytes.
- The length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes.
- This value is normally referred to as the receiving window (rwnd) and is determined by the receiver.

Checksum.

- This 16-bit field contains the checksum.
- The use of the checksum for TCP is mandatory.
- For the TCP pseudo-header, the value for the protocol field is 6.

Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

TCP Header: Contd.

Window size.

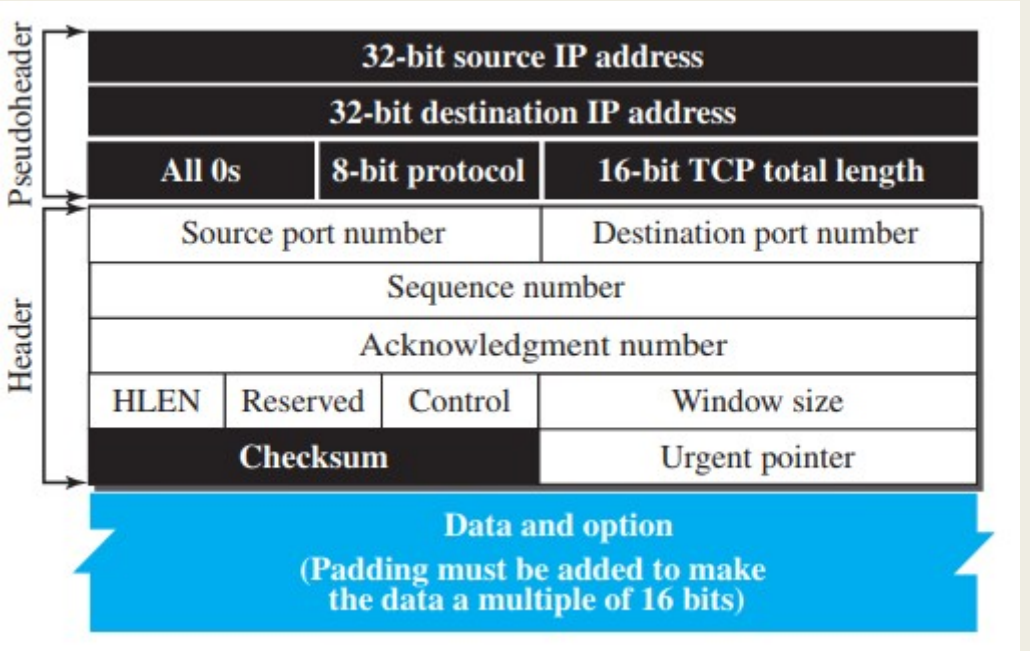
- This field defines the window size of the sending TCP in bytes.
- The length of this field is 16 bits, which means that the maximum size of the window is 65,535 bytes.
- This value is normally referred to as the receiving window (rwnd) and is determined by the receiver.

Checksum.

- This 16-bit field contains the checksum.
- The use of the checksum for TCP is mandatory.
- For the TCP pseudo-header, the value for the protocol field is 6.

TCP Header: Contd.

- Keep the end-to-end feature of TCP and avoid replication of data available in the IP header.
- **The pseudo-header consists of parts of the IP header.**
- It covers relevant fields of the IP header that are static (do not change in the routing of packets).



Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

TCP Header: Contd.

Urgent pointer.

- This 16-bit field, which is valid only if the urgent flag is set, 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.

Options.

- There can be up to 40 bytes of optional information in the TCP header.

Computing TCP Checksum

- Prepends a psuedo-header before the TCP segment
- Pads octets of zeros to make (psuedo-header + TCP segment + pad length) multiple of 16
- Checksum computed on this entire thing
- Psuedo-header and pad octets are not transmitted, just used for calculating the checksum
- Receiver will do the same and compare checksum

Basic Data Transfer

- Connection established
- All transmission involves TCP segments
- Application generates data
- TCP software puts application data in send buffer
- TCP segments are formed from the data in the send buffer
  - Can be of different sizes, formed at different times
  - TCP header added
- Sender sends the TCP segments one by one as they are formed subject to send window size
- Timer set for each segment set, sender waits for acknowledgments
- If timeout, retransmit. If ack received, change window and transmit more segments if possible

Basic Data Transfer

- Receiver sends acknowledgments by sending another TCP segment
  - Can be an acknowledgements segment (ACK flag set and a valid acknowledgement no. field) with no data (how to know this?)
  - Acknowledgements can be piggybacked on TCP segments carrying data in the other direction
  - Acknowledgement specifies next sequence no. the receiver expects
  - Cumulative positive acknowledgement, no NAK

- The above repeats until connection is terminated

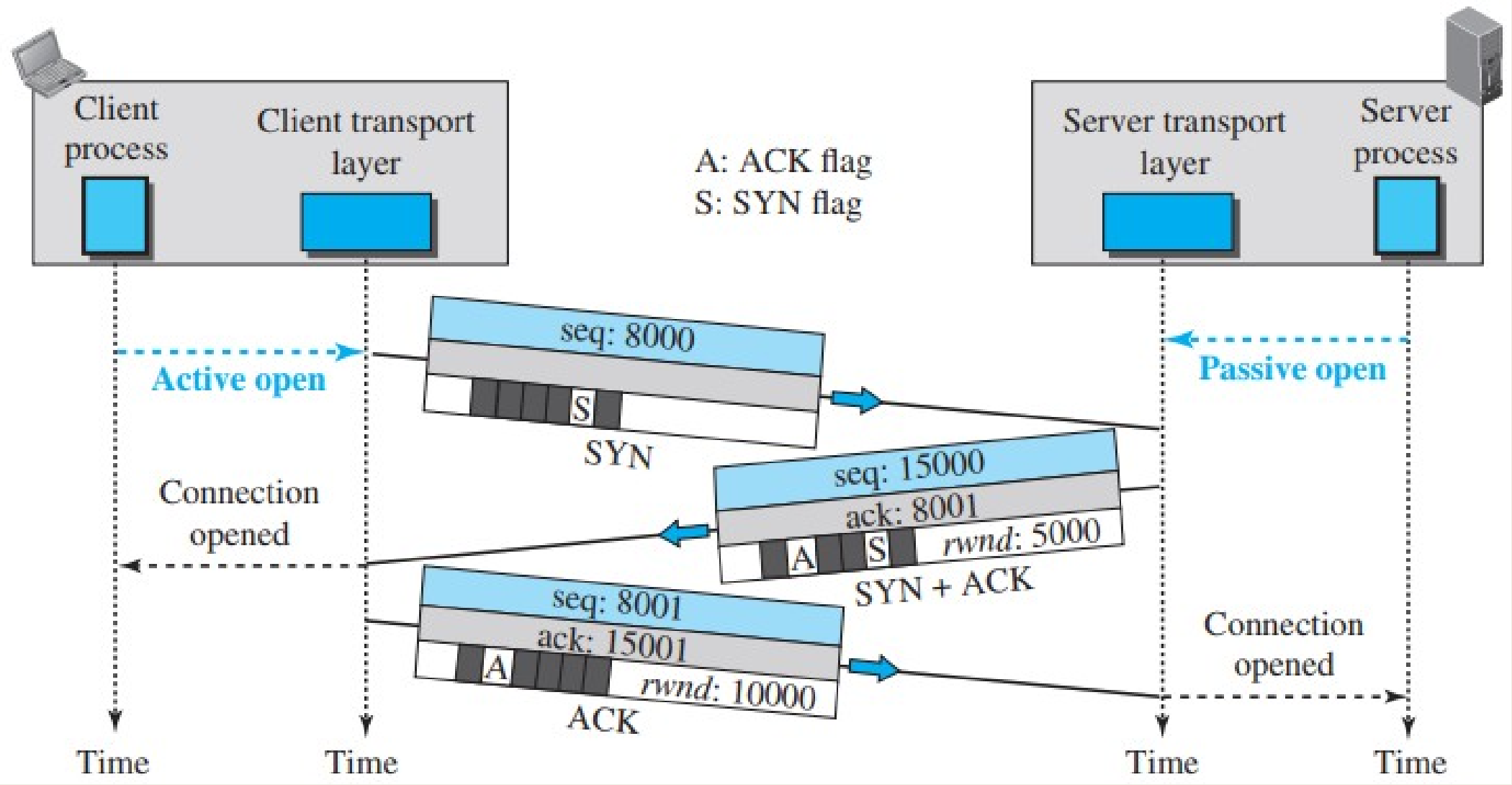
TCP Connection

- TCP establishes a logical path between the source and destination.
- All of the segments belonging to a message are then sent over this logical path.
- Using a single logical connection for the entire message facilitates the acknowledgment process as well as retransmission of damaged or lost frames.
- TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself.
- If a segment is lost or corrupted, it is retransmitted. Unlike TCP, IP is unaware of this retransmission.
- If a segment arrives out of order, TCP holds it until the missing segments arrive; IP is unaware of this reordering.
- In TCP, connection-oriented transmission requires three phases: connection establishment, data transfer, and connection termination.

Connection Establishment

- The connection establishment in TCP is called **three-way handshaking.**
- The server program tells its TCP that it is ready to accept a connection.
- This request is called a passive open.
- Although the server TCP is ready to accept a connection from any machine in the world, it cannot make the connection itself.
- The client program issues a request for an active open.
- A client that wishes to connect to an open server tells its TCP to connect to a particular server.
- TCP can now start the three-way handshaking process.

Connection Establishment



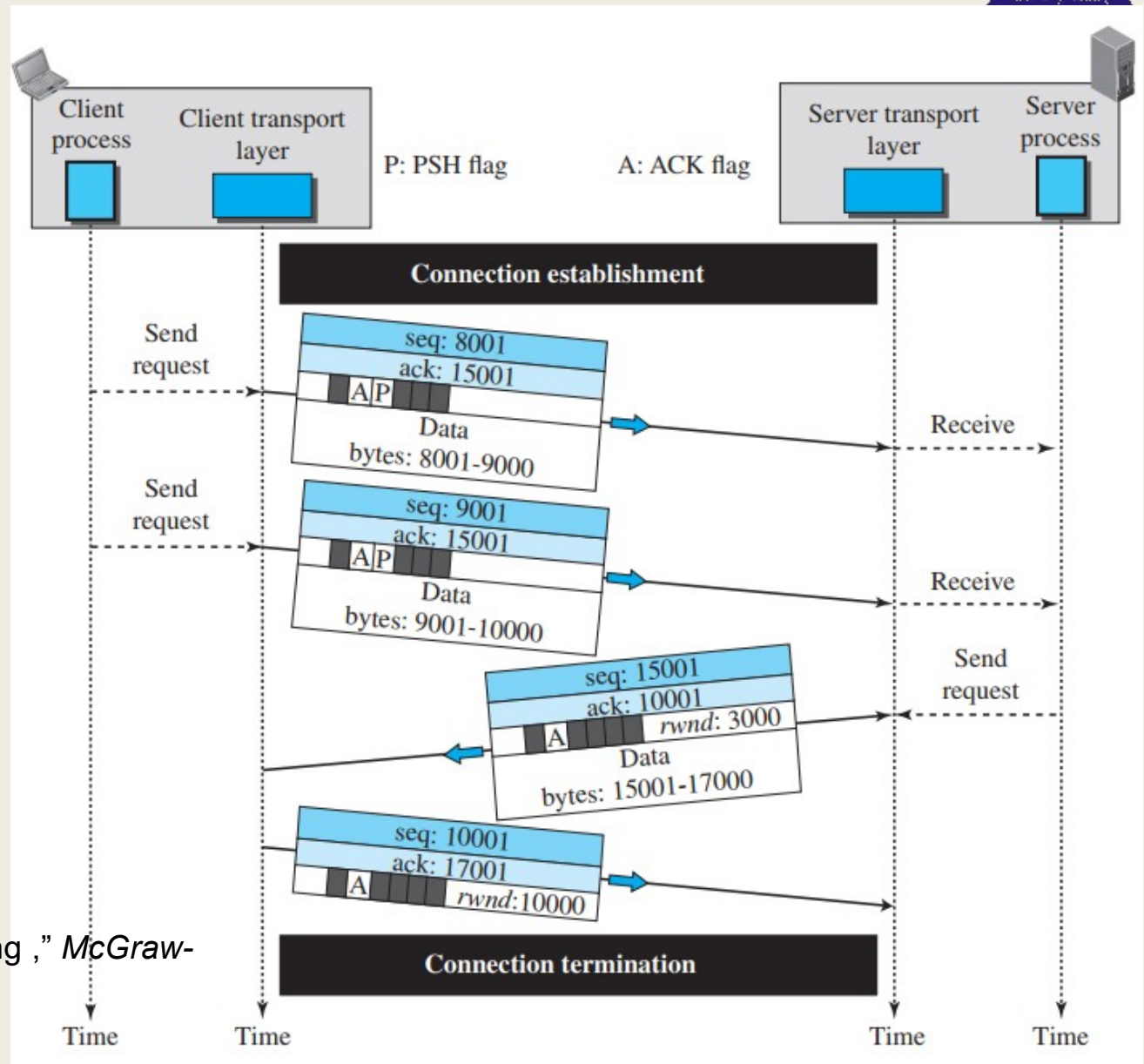
Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

Data Transfer

- Bidirectional data transfer take place.
- The acknowledgments are piggybacked with the data.
- The data segments sent by the client have the PSH (push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received.
- sending TCP uses a buffer to store the stream of data coming from the sending application program. The sending TCP can select the segment size.
- The receiving TCP also buffers the data when they arrive and delivers them to the application program when the application program is ready or when it is convenient for the receiving TCP.

Data Transfer

- The client sends 2,000 bytes of data in two segments.
- The server then sends 2,000 bytes in one segment.
- The client sends one more segment.
- The first three segments carry both data and acknowledgment, but the last segment carries only an acknowledgment because there is no more
  data to be sent.
- The data segments sent by the client have the PSH (push) flag set so that the server TCP knows to deliver data to the server process as soon as they are received.



Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

Urgent Data

- Each byte of data has a position in the stream.
- There are occasions in which an application program needs to send *urgent* bytes, some bytes that need to be treated in a special way by the application at the other end.
- The solution is to send a segment with the URG bit set.
- The sending application program tells the sending TCP that the piece of data is urgent.
- The sending TCP creates a segment and inserts the urgent data at the beginning of the segment.

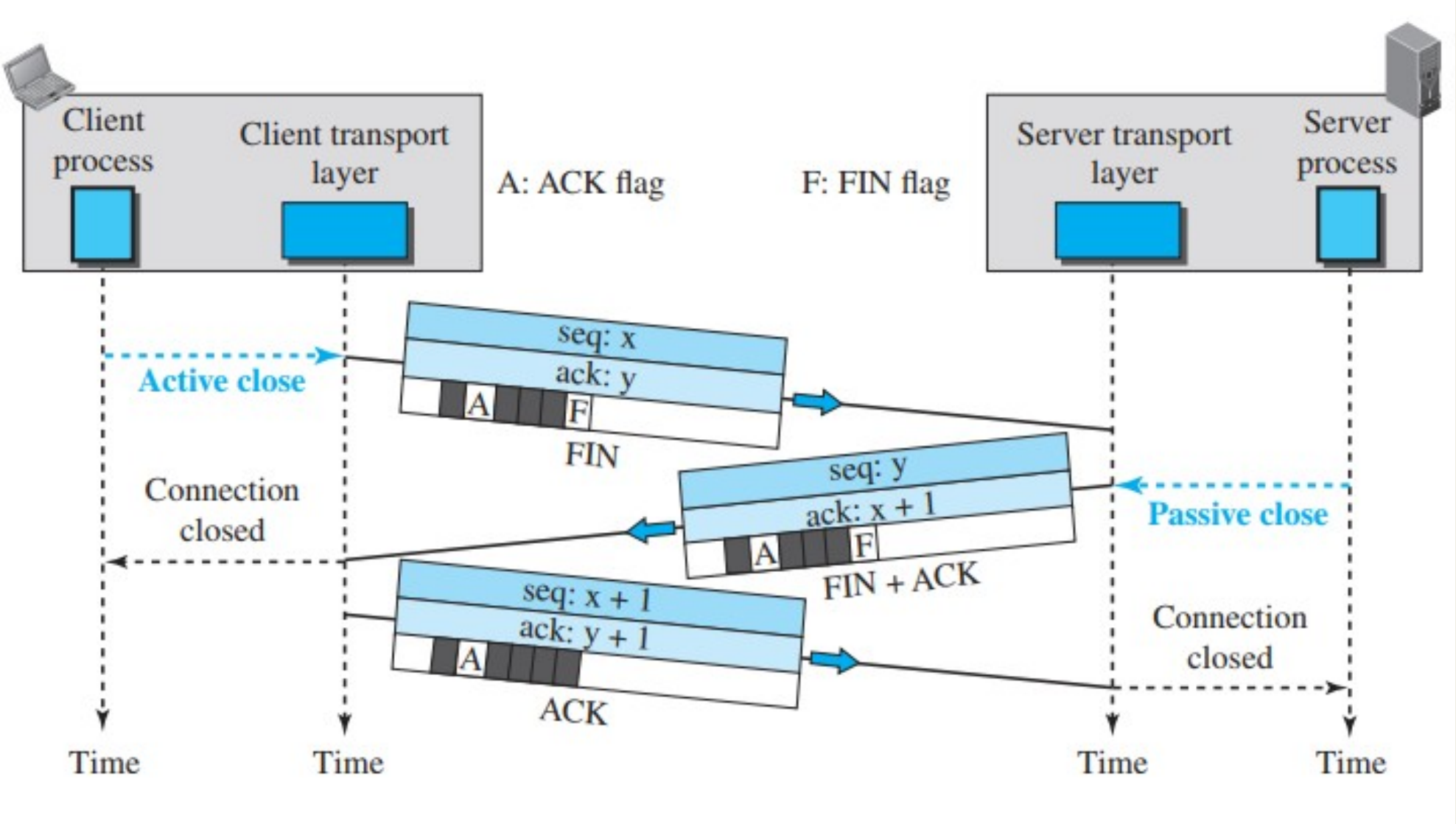
Connection Termination

Either of the two parties involved in exchanging data (client or server) can close the connection.

Ways:

- three-way handshaking
- four-way handshaking with a half-close option.

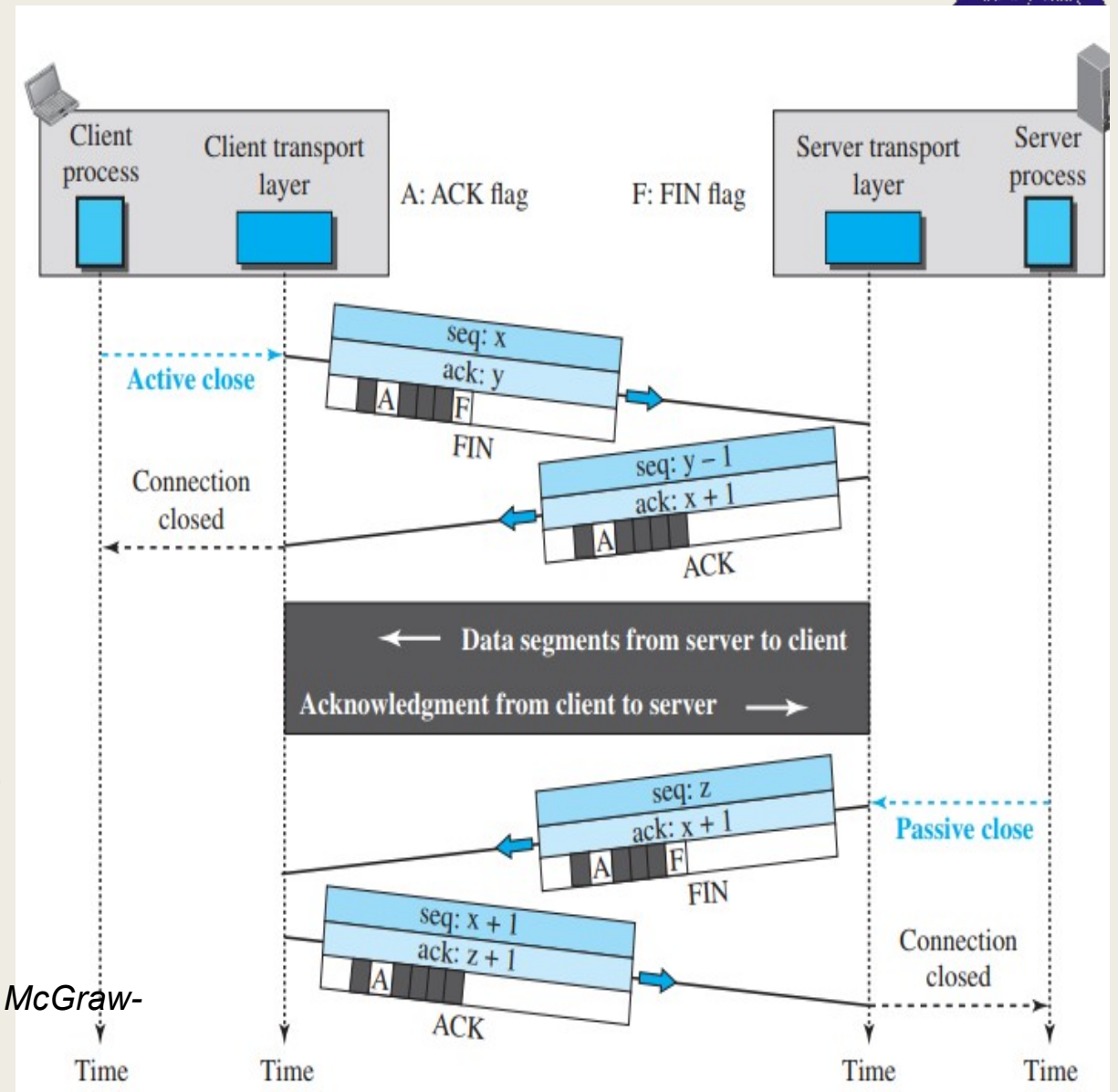
Three Way Handshaking



Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

Half Close

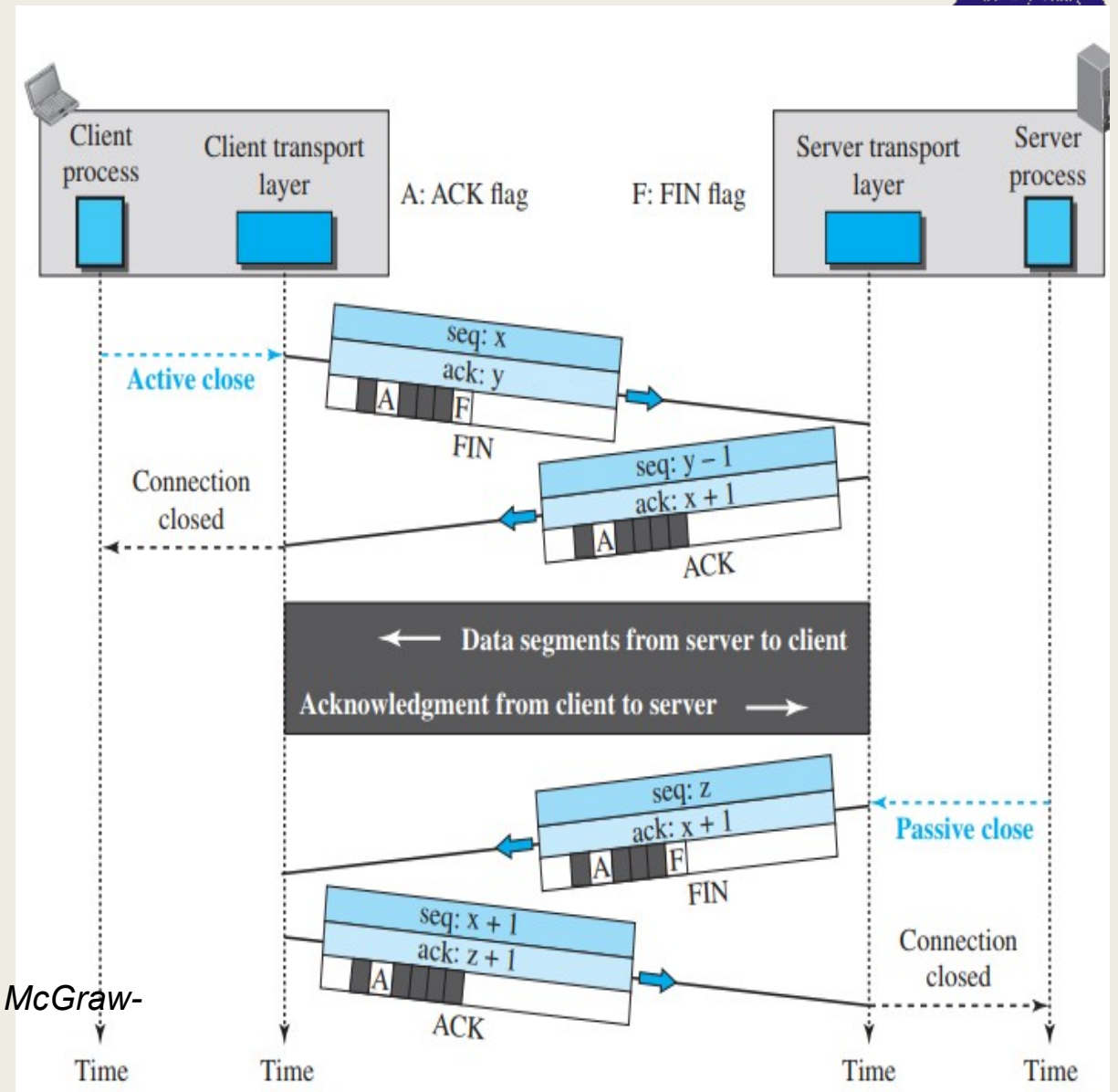
- In TCP, one end can stop sending data while still receiving data. This is called a ***halfclose.***
- Either the server or the client can issue a half-close request.
- The client half-closes the connection by sending a FIN segment.
- The server accepts the half-close by sending the ACK segment.
- The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



Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

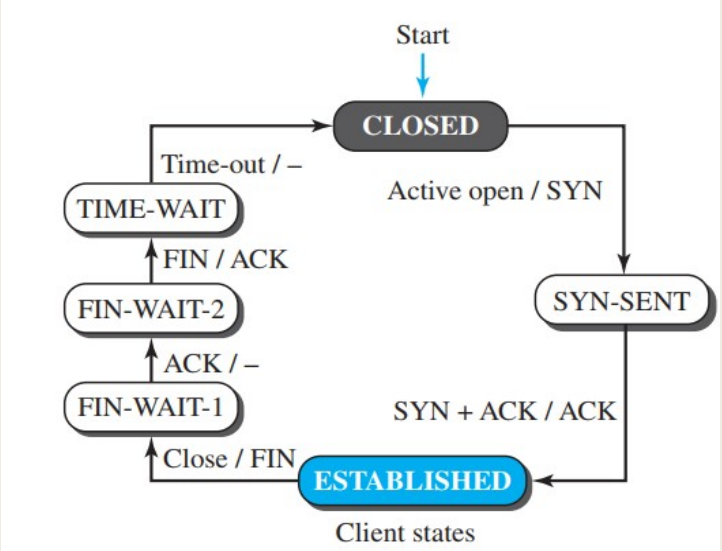
Half Close

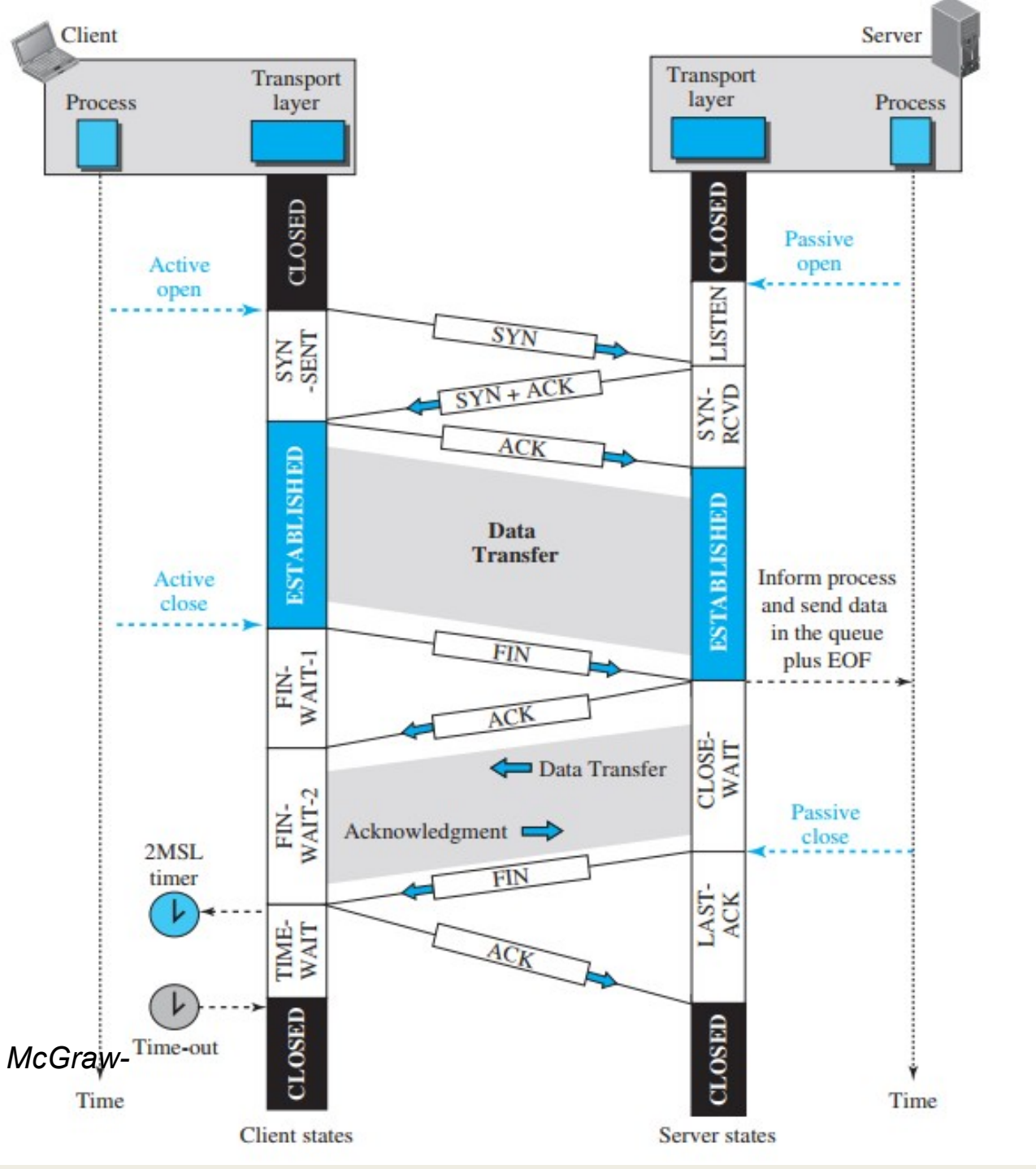
- In TCP, one end can stop sending data while still receiving data. This is called a ***halfclose.***
- Either the server or the client can issue a half-close request.
- The client half-closes the connection by sending a FIN segment.
- The server accepts the half-close by sending the ACK segment.
- The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



Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

Contd.





Source: B. A. Forouzan, " Data Communications and Networking ," *McGraw-Hill Forouzan Networking Series*,5E.

Thank You!!!