CS348: Computer Networks

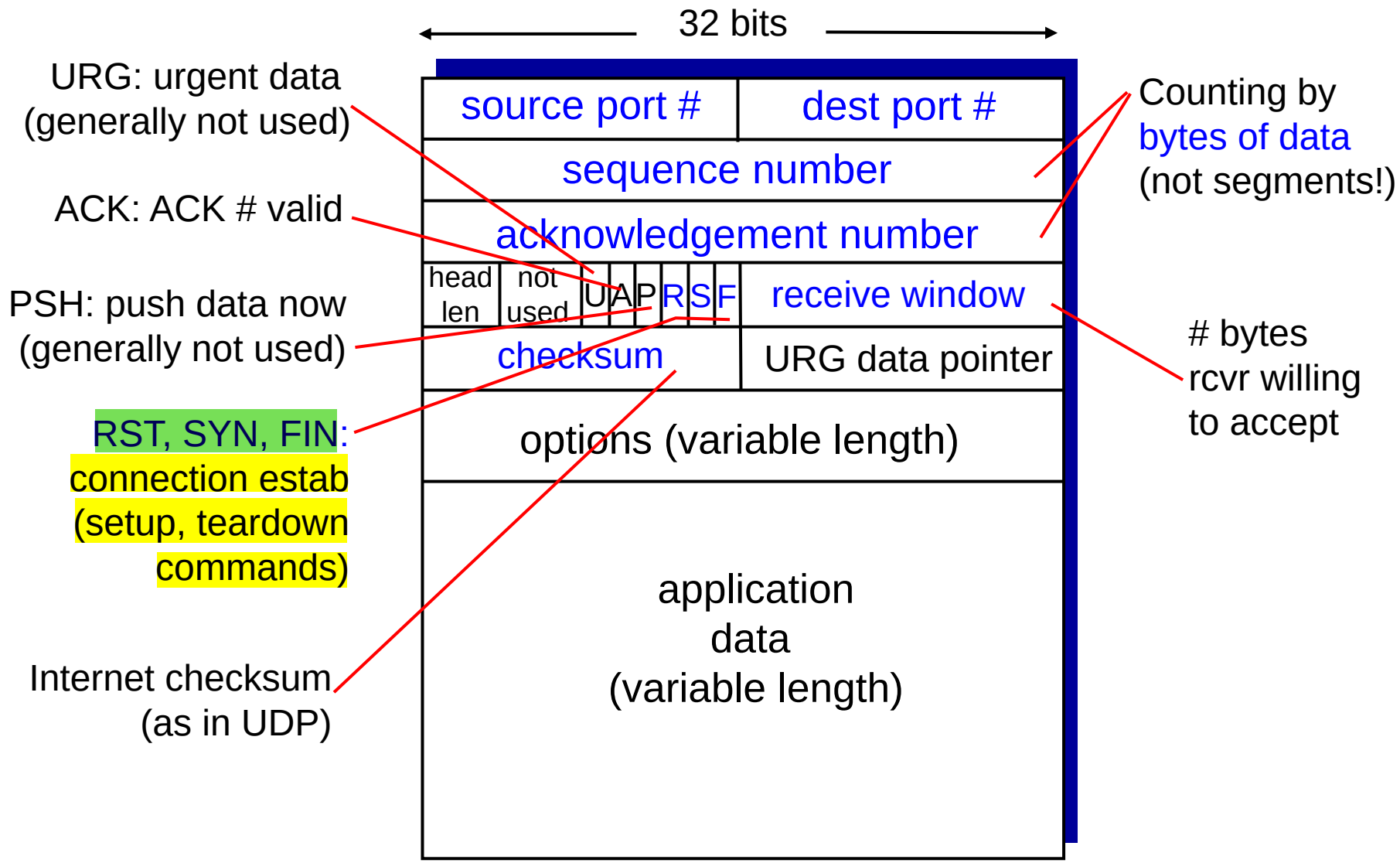
TCP Introduction

Dr. Manas Khatua
Assistant Professor
Dept. of CSE, IIT Guwahati
E-mail: manaskhatua@iitg.ac.in

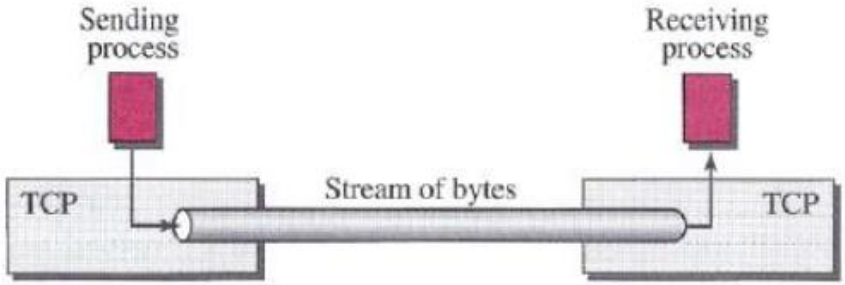
TCP: Overview

- connection-oriented:
  - handshaking (exchange of control msgs)
  - initialize sender, receiver state before data exchange
  - TCP connection is not end-to-end TDM/FDM circuit or virtual circuit (as only end systems maintain connection state)
- point-to-point:
  - one sender, one receiver
- reliable, in-order, *byte steam:*
  - error and loss control
  - no "message boundaries"
- pipelined:
  - TCP congestion and flow control set window size
- flow controlled:
  - sender will not overwhelm the receiver
- full duplex data:
  - bi-directional data flow in same connection
  - MSS: maximum segment size (max amount of app. layer data in a segment)

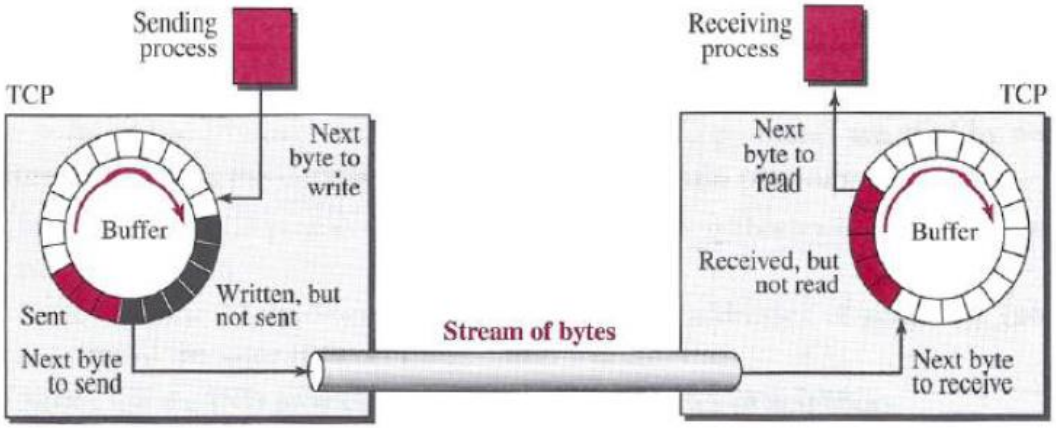
TCP segment structure



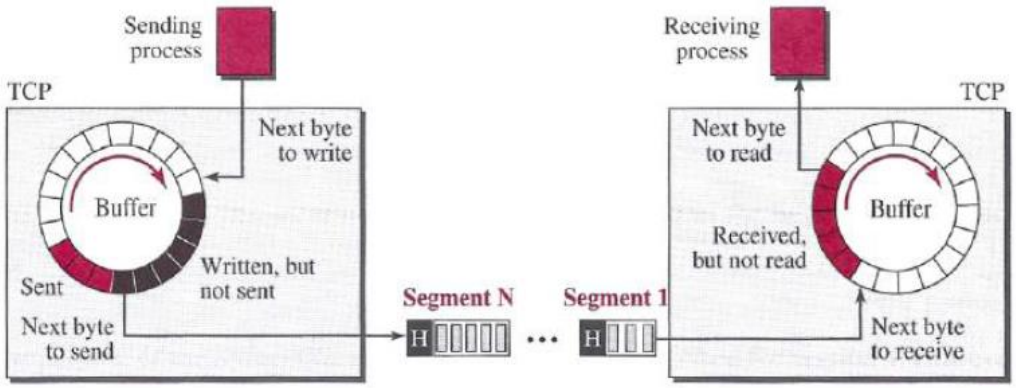
TCP - Stream Delivery Service



- Each TCP endpoint has its own sending and receiving buffer
- Sending and Receiving buffer may not necessarily write / read data at the same rate



Cont…



- Segmentation and Reassembly
  - TCP usually determines the maximum segment size (MSS) based on the MTU of Layer 3 (IP layer).
  - Note: for UDP, the assumption was that the UDP segments are small in size
- No segment number in TCP. But, TCP uses SEQ and ACK numbers
  - These are byte numbers, but not segment numbers
- Number is independent in each direction
- For 1st byte: arbitrary number in [0, 2^{32}- 1], as TCP uses 32-bit seq#

TCP Seq #, ACK

sequence numbers:

- byte stream "number" of first byte in segment's data

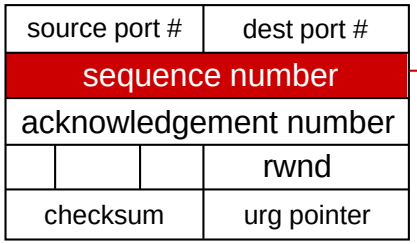
acknowledgements:

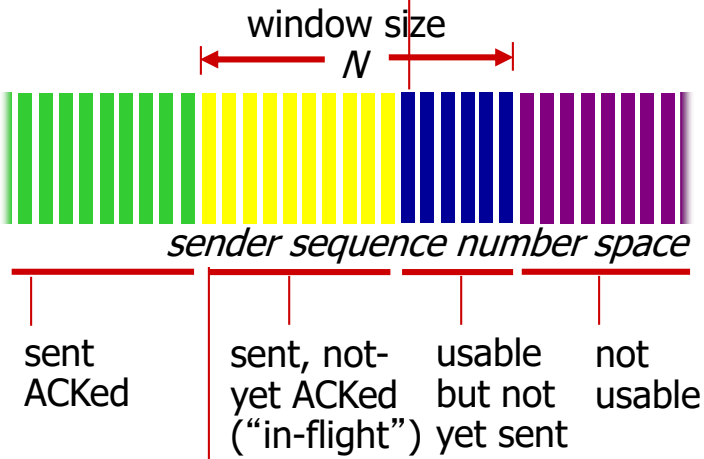
- seq # of next byte expected from other side
- cumulative ACK

Q: how receiver handles out-of-order segments

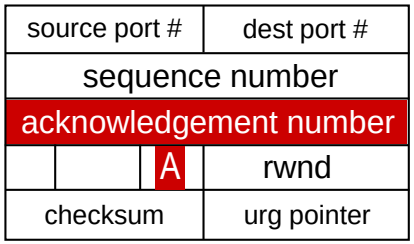
- Ans: TCP spec doesn't say! -- up to implementer

outgoing segment from sender





incoming segment to sender



Cont…

- **ACK in TCP** can be described as a hybrid of GBN and SR

- TCP is similar to GBN because both protocols have a limit on the number of unACK'd packets that the sender can send into the network.

- However, TCP is different from GBN because GBN requires the retransmission of every unACK'd packet when packets are lost, but TCP only retransmits the oldest unACK'd one.

- TCP is similar to SR because, when packets are lost due to congestion, the protocols do not require the sender to retransmit EVERY unACK'd packet sent by the sender. The sender just retransmits the oldest unACK'd packet.

- TCP is different from SR because SR requires individual acknowledgement of each packet that was sent by the receiver; but rather than selectively ACKing every packet, TCP sends an ACK for the next packet that it is expecting and buffers the ones that it has received so far, even if they're out of order

Cont…

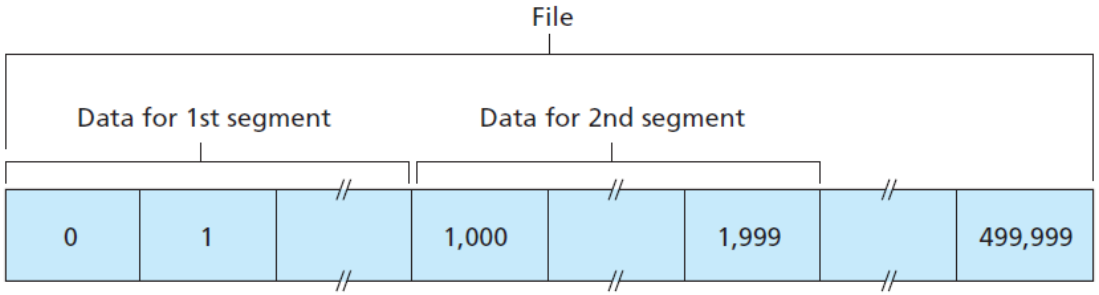
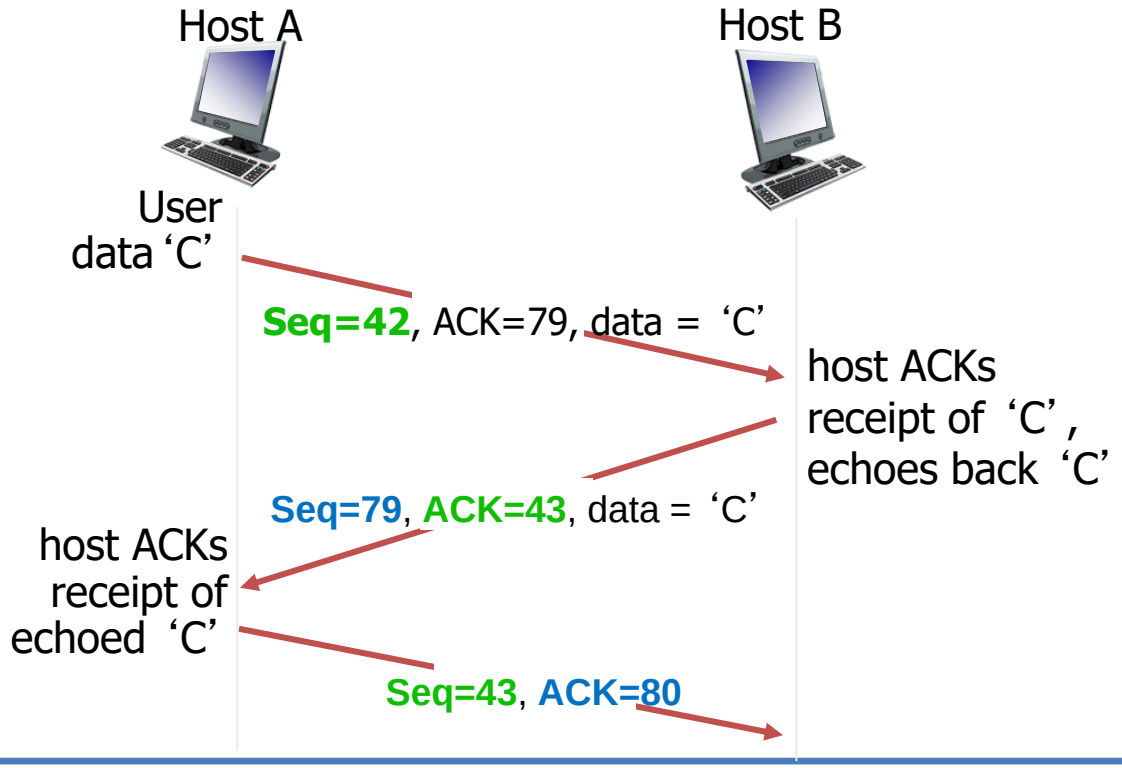


Figure 3.30 ♦ Dividing file data into TCP segments

- 1st seg. SEQ#=0
- 2nd seg. SEQ#=1000
- 3rd seg. SEQ#=2000
- So on.

simple telnet scenario



TCP Connection

- TCP is connection-oriented (although IP is connectionless)
- TCP connection is logical, not physical.

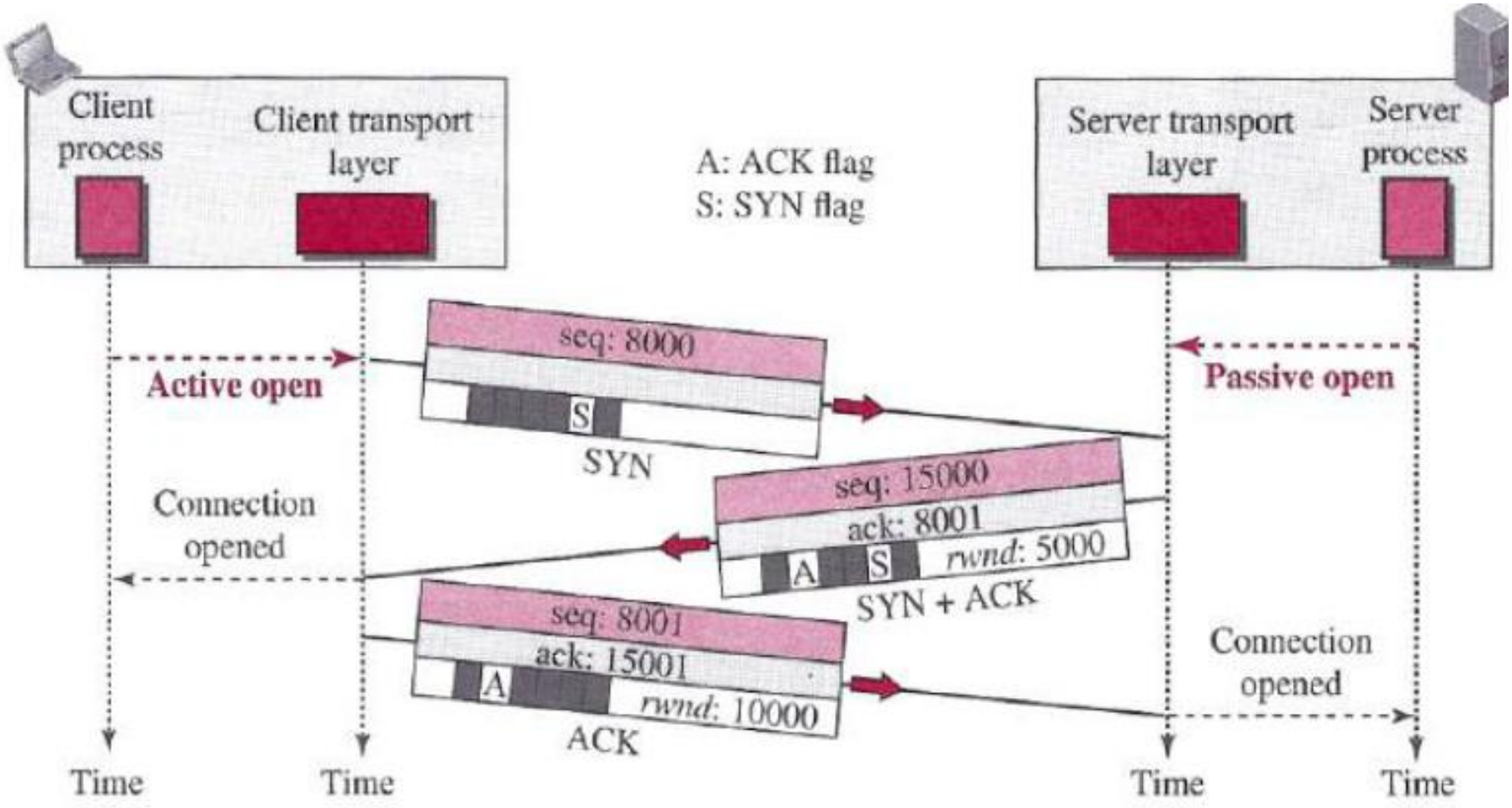
- TCP operates in full-duplex mode
- TCP uses three-way-handshaking
  - SYN
  - ACK+SYN
  - ACK

- Let, an application program (i.e. client) wants to make a connection with another application program (i.e. server) using TCP

- The process starts with the server.
  - Passive open (server process informs transport layer of the server that it is ready)
  - Active open (client process issues request to client transport layer)
  - Now client transport layer starts three-way-handshaking

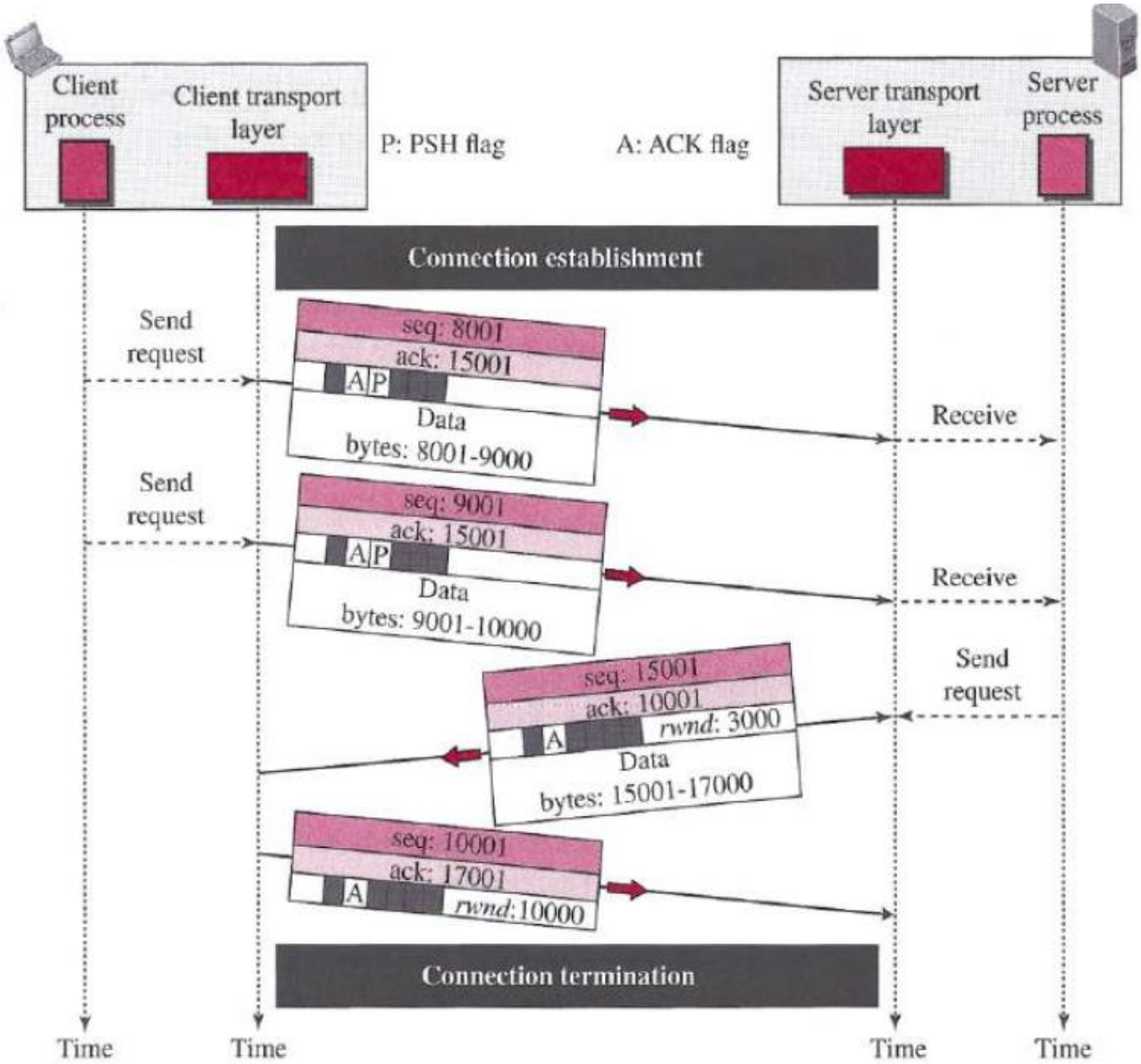
Connection Creation

- ACK and SYN flags are used



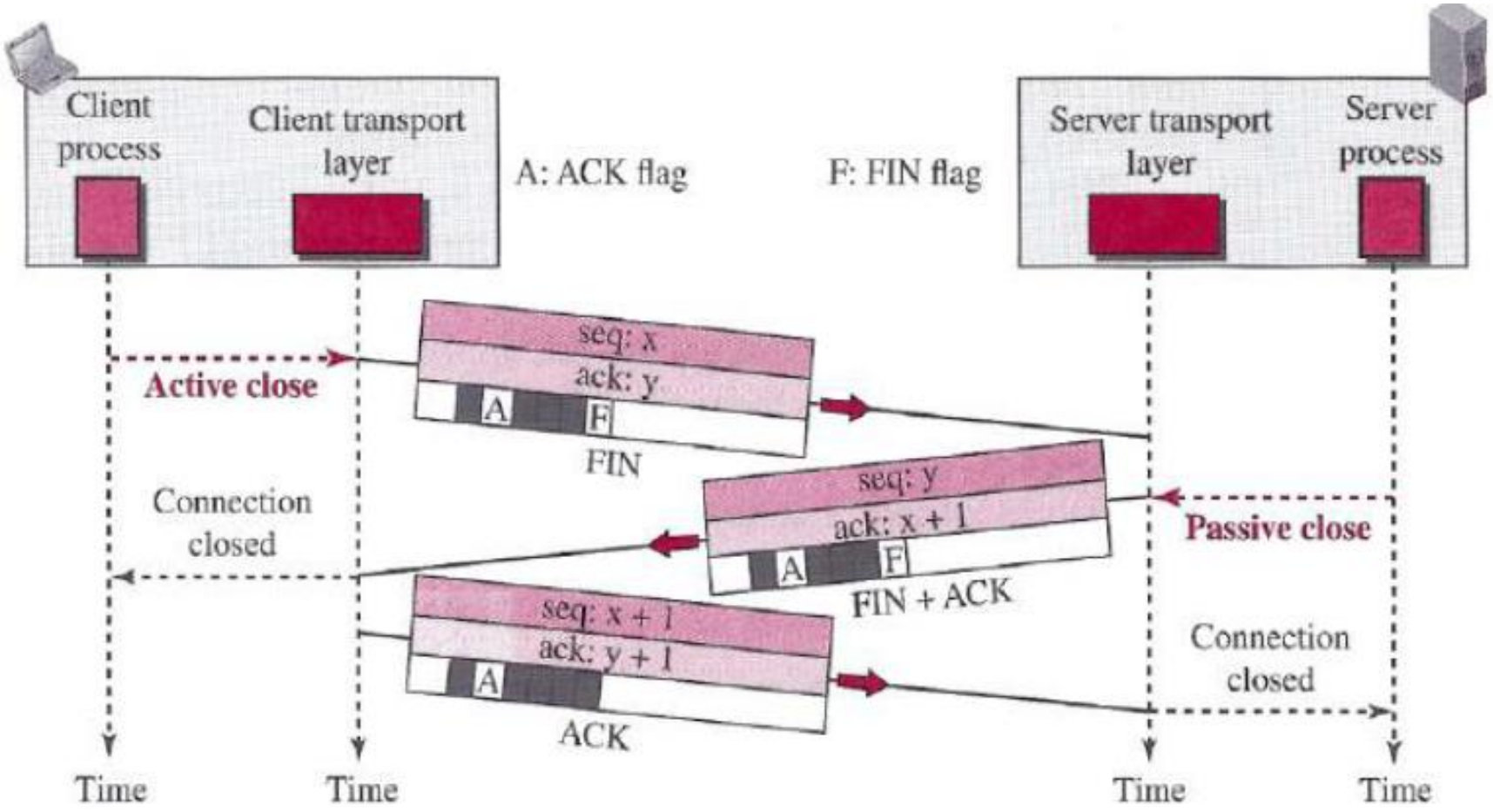
Data Transfer

ACK and/or PSH flags are used



Connection Termination

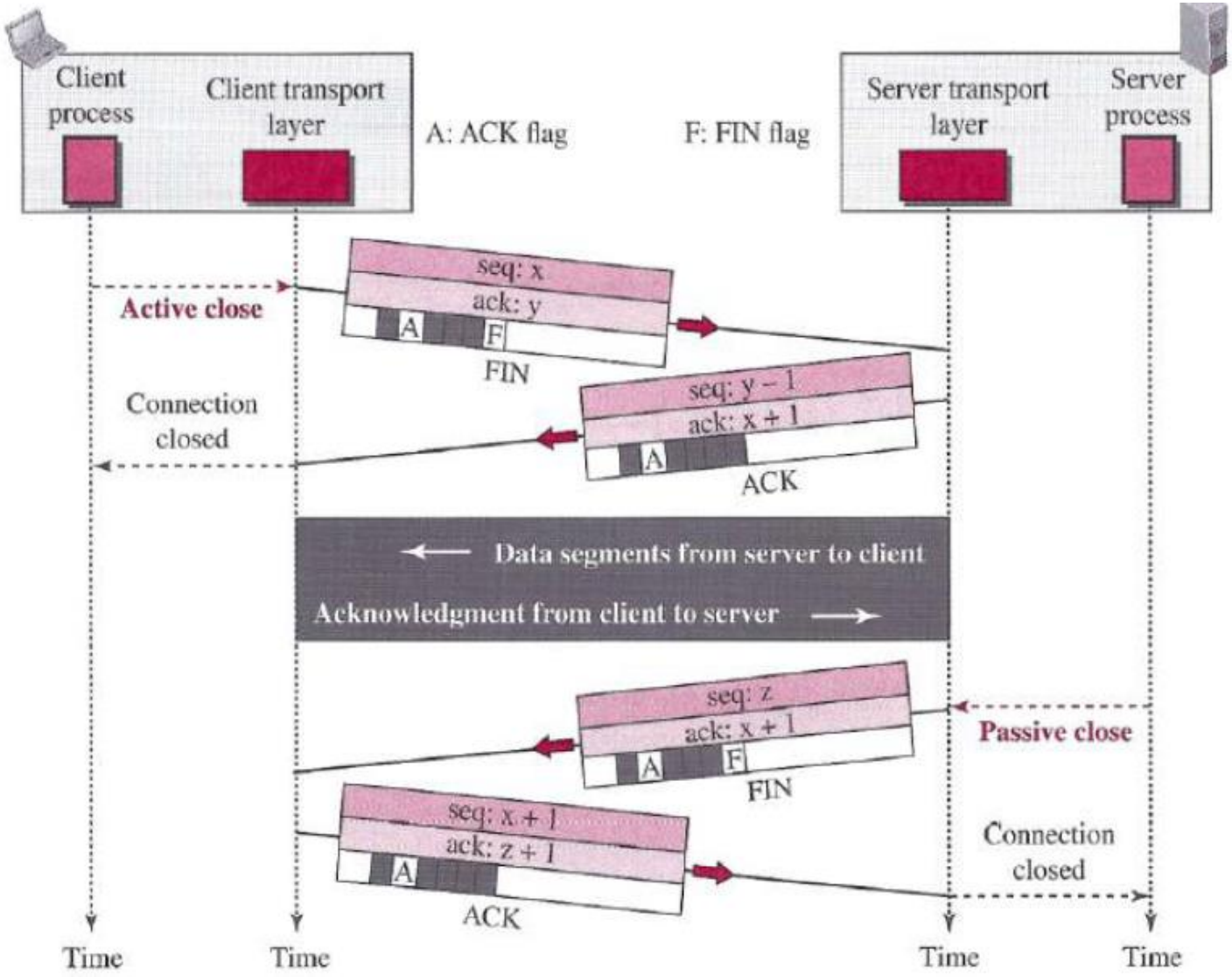
- ACK and FIN flags are used



Half-close Connection

Example:

Sorting at server



Full Scenario

Create Connection

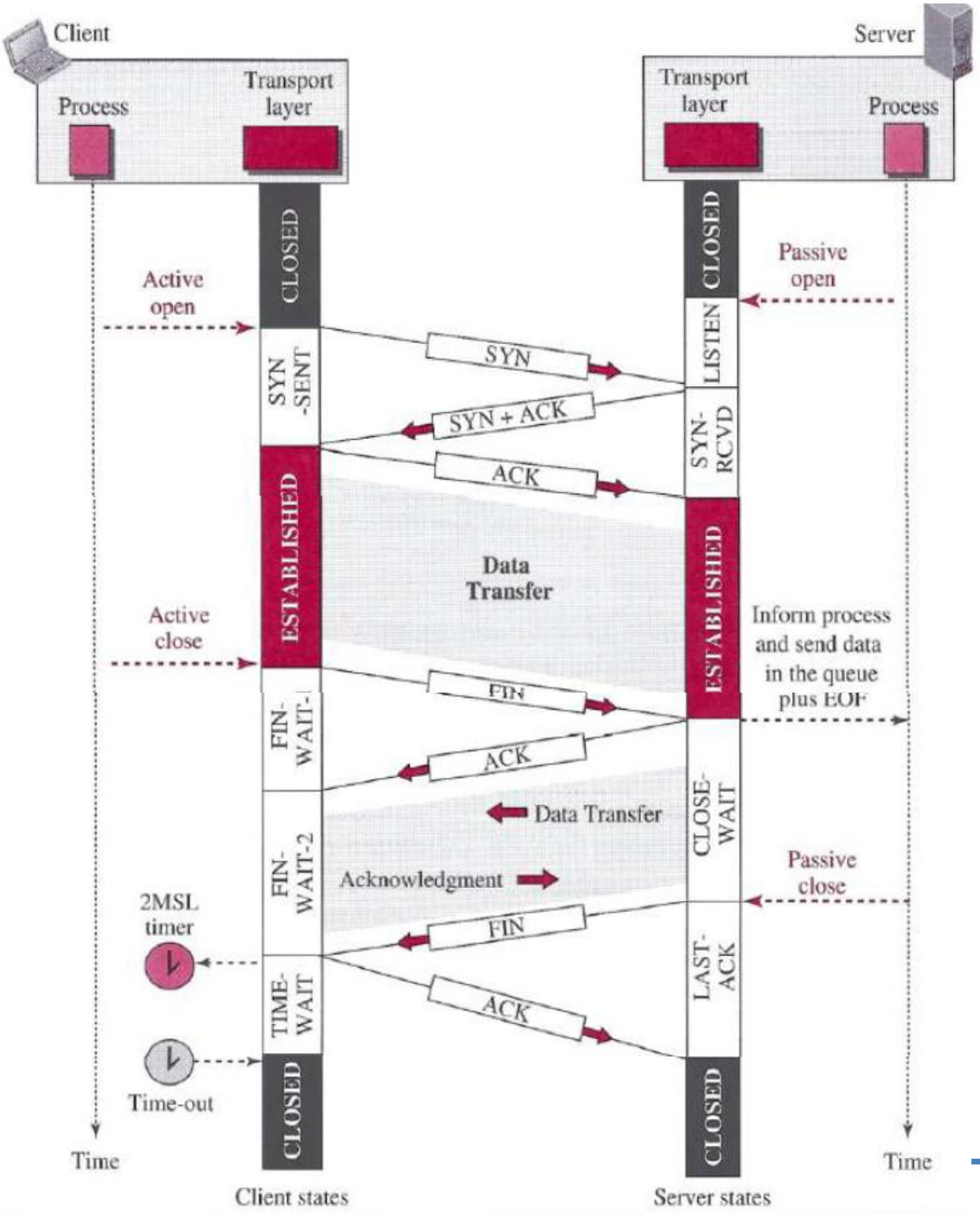
Data Transfer
in both direction

Half-close

Receive Response

Close other half

Closed status



PSH, RST, URG flags

- PUSH (PSH) flag
  - means sending TCP must not wait for the window to be filled, and then send the segment
  - informs the receiving TCP to deliver the received segment immediately to application program

- RESET (RST) flag
  - means it is telling the sender "I don't have socket for that segment. Please don't resend the segment"
  - It is required when a host receives TCP SYN segment with a destination port (say 80) but the destination is not accepting any connection on that port (may be Web server is not running at port 80)

- URGENT (URG) flag
  - when this bit is set, the Urgent Pointer is also set (in the TCP header Options field: 16 bit).
  - URG pointer tell how many bytes of the data is urgent in the segment that has arrived.
  - Example:
    - if the data size is 100 bytes and only first 50 bytes is urgent, the urgent pointer will have a value of 50

RTT Estimation & Timeout in TCP

- TCP uses a timeout/retransmit mechanism to recover from lost segments.
- The **timeout** should be larger than the connection's **round-trip time** (**RTT**)
- **Q**: How should the RTT be estimated in the first place?
- **Q**: Should a timer be associated with each and every unACKed segment?
- The base RTT (*SampleRTT*) for a segment is
  - the amount of time between the timestamps when the segment is sent and when an ACK for the segment is received.
- But, the *SampleRTT* values will fluctuate from segment to segment due to
  - congestion in the routers,
  - varying load on the end systems.
- **Solution**:
  - TCP maintains an average of the *SampleRTT* values (called *EstimatedRTT*)
  - Exponentially weighted moving average (EWMA)

$$\text{EstimatedRTT} = (1 - \alpha) \cdot \text{EstimatedRTT} + \alpha \cdot \text{SampleRTT}$$

The recommended value of α is $= 1/8$

Cont…

- It is also valuable to have a measure of the variability of the RTT.

DevRTT = (1 – β) . DevRTT + β . | SampleRTT – EstimatedRTT | The recommended value of β is 1/4.

- So, the DevRTT will be
  - Small for little fluctuation
  - Large for lot of fluctuation

- It is desirable to set, Timeout = *EstimatedRTT* + some margin.
- The margin should be
  - large when there is a lot of fluctuation in the *SampleRTT* values;
  - small when there is little fluctuation in the *SampleRTT* values

So, Timeout_Interval = EstimatedRTT + 4 . DevRTT

- *Question:* How long the receiver waits before sending a stand-alone ACK to acknowledge data?
  - *Delayed ACK* was invented to reduce the number of *ACKs* required to acknowledge the segments
  - A host may delay sending an ACK response by up to 500 ms.
  - However, a stand-alone ACK is sent if 2 packets of data arrive before the delayed ACK timer expires.

TCP Applications

- Major Internet applications rely on TCP
  - World Wide Web (HTTP)
  - E-mail (SMTP, IMAP, POP)
  - File Transfer Protocol (FTP)
  - Secure Shell (SSH)
  - Telnet

Thanks!

Content of this PPT are taken from:

1) **Computer Networks: A Top Down Approach**, by J.F. Kuros and K.W. Ross, 6th Eds, 2013, Pearson Education.

2) **Data Communications and Networking**, by B. A. Forouzan , 5th Eds, 2012, McGraw-Hill.

3) **Chapter 3 : Transport Layer,** PowerPoint slides of "Computer Networking: A Top Down Approach", 6th Eds, J.F. Kurose, K.W. Ross