Lecture 8:
Transport layer

Reading 6.2, 6.3, 6.4, 6.5

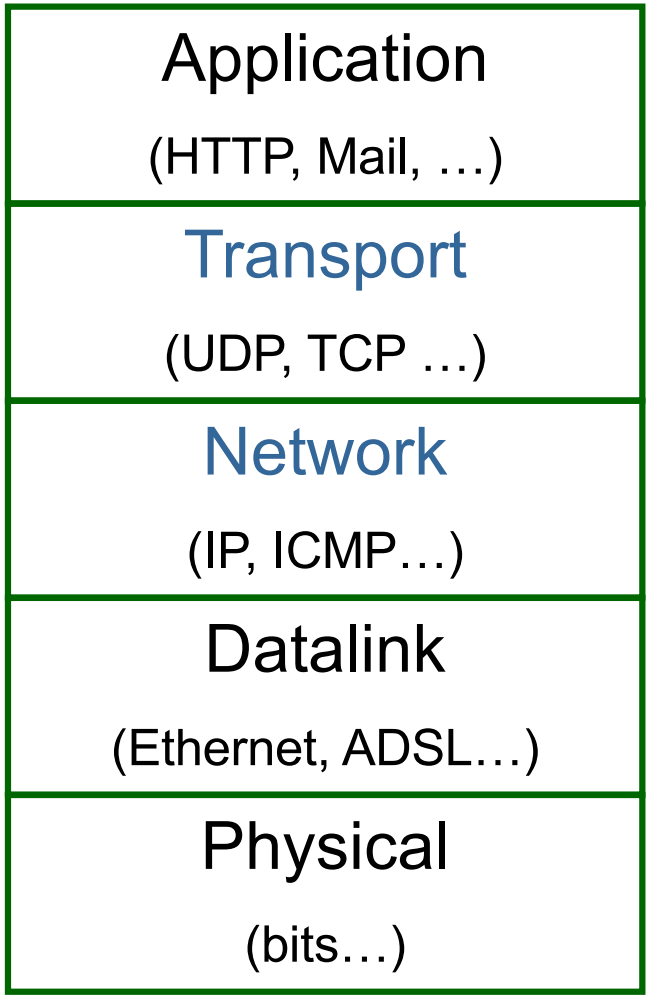
Computer Networks, Tanenbaum

ONE LOVE. ONE FUTURE.

Contents

- Principles of transport layer
- UDP protocol
- TCP protocol

Transport layer in OSI model



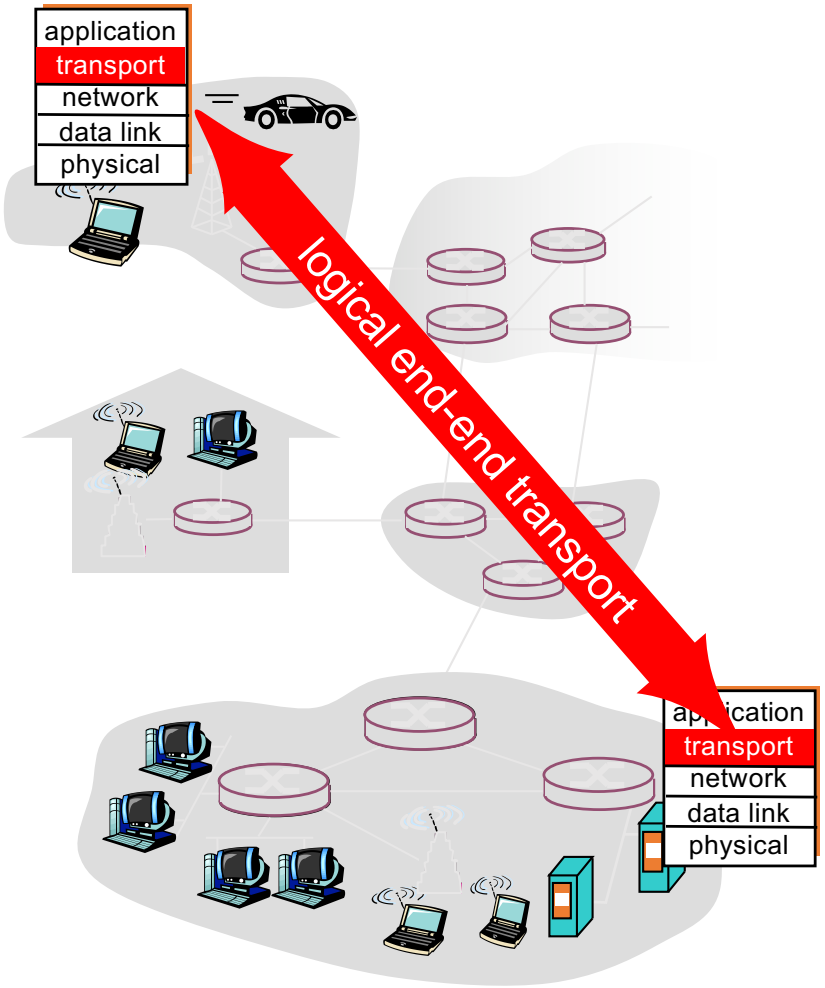
Support applications

Transferring data between applications/processes

Routing and forwarding data between hosts

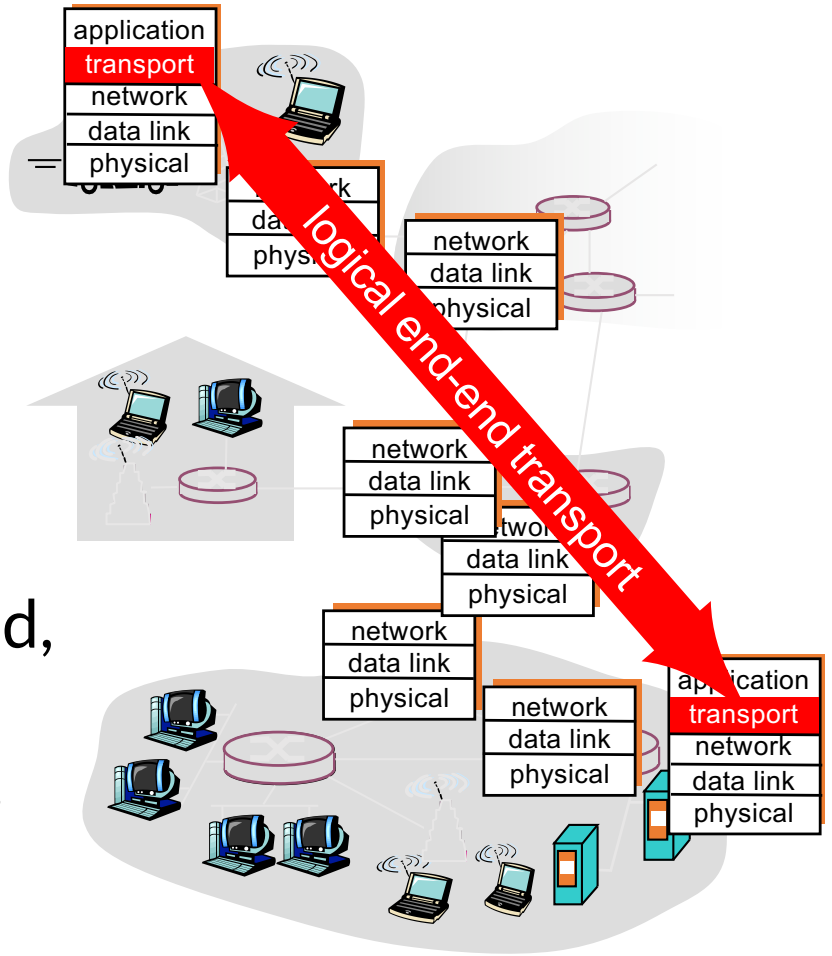
Principle of transport layer (1)

- Get applications/processes to communicate to each other
- Sender:
  - Receives data from application
  - Place data in segments and give to network layer
  - If the data size is too big, it is divided into many segments
- Receiver:
  - Receives segments from network layer
  - Reconstitute data from segments and deliver to the application



Principle of transport layer (2)

- Transport layer is installed in end systems
  - Not installed in routers, switches…
- Two kinds of transport layer services
  - Reliable, connection-oriented, e.g TCP
  - Not reliable, connectionless, e.g. UDP



Why two kinds of service?

- Requirements from application layer are various
- Some applications need transport service with 100% fiability such as mail, web…
  - Should use TCP transport service
- Some applications need to transmit data as fast as possible, with some fault tolerance, e.g. VoIP, Video Streaming
  - Should use UDP transport service

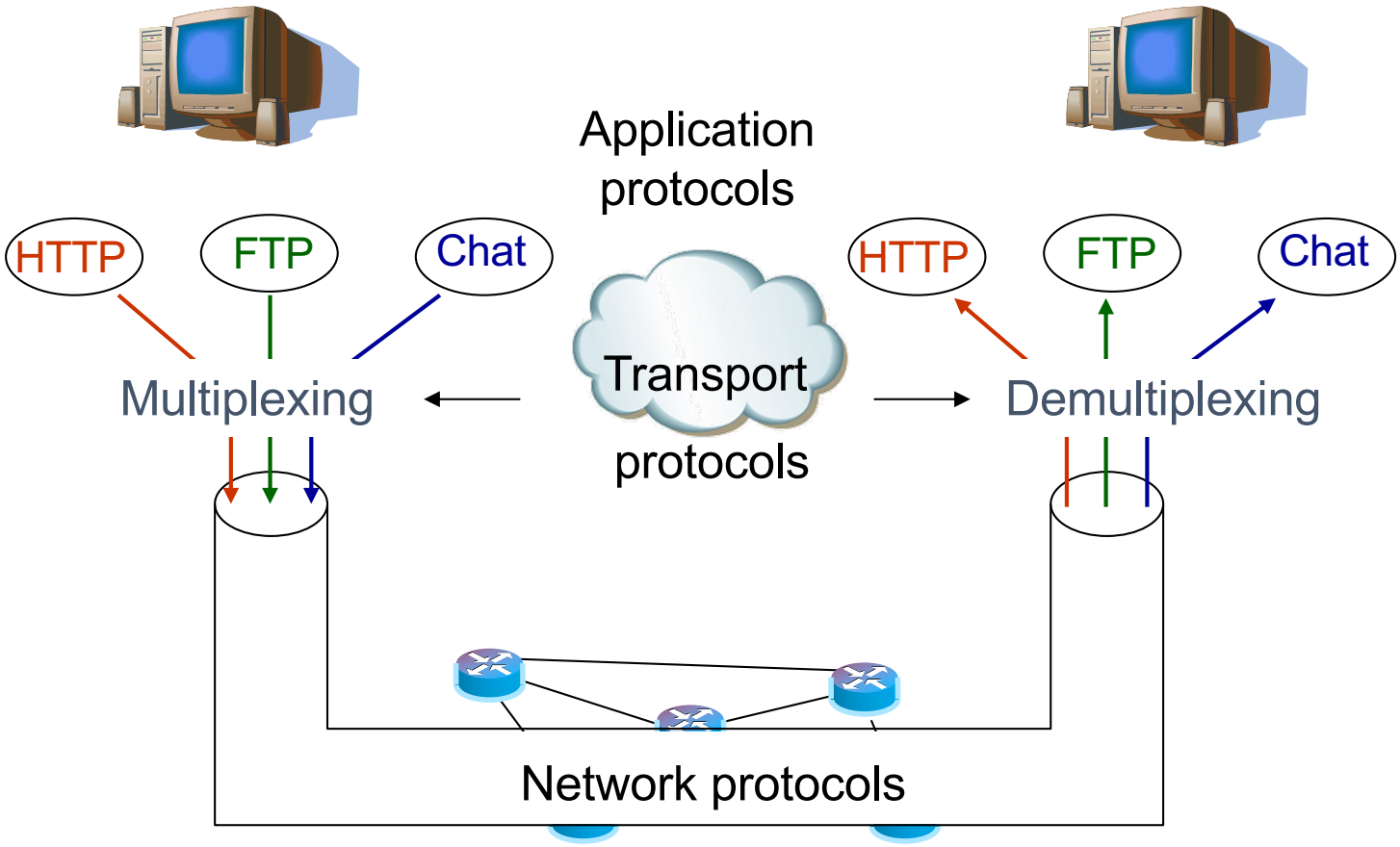
Applications and transport services

Application	Application protocols	Transport protocols
e-mail	SMTP	TCP
remote terminal access	Telnet	TCP
Web	HTTP	TCP
file transfer	FTP	TCP
streaming multimedia	Specific protocols (e.g. RealNetworks)	TCP or UDP
Internet telephony	Specific protocols (e.g., Vonage,Dialpad)	Usually UDP

Functionalities

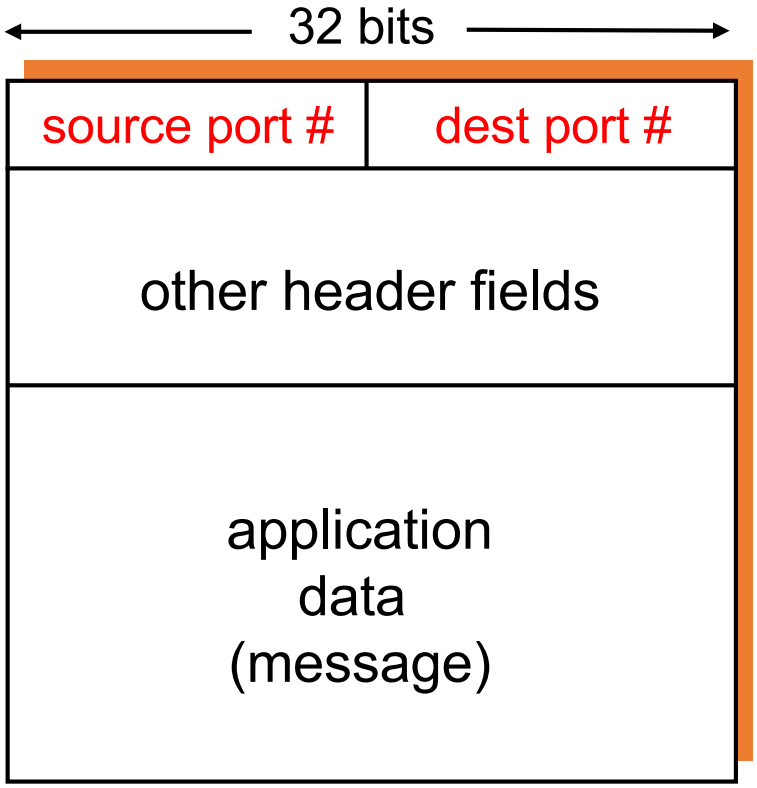
MUX/DEMUX

Mux/Demux


Application protocols
HTTP
FTP
Chat
HTTP
FTP
Chat
Multiplexing
Transport protocols
Demultiplexing
Network protocols

How does it Mux/Demux?

- How to distinguish applications running in the same hosts?
  - Use an identifier called port number (16 bits)
  - Each process is assigned a port
- Socket: A pair of IP address and port
  - Socket identifies an unique application process all over the world



TCP/UDP segment format

UDP
User Datagram Protocol

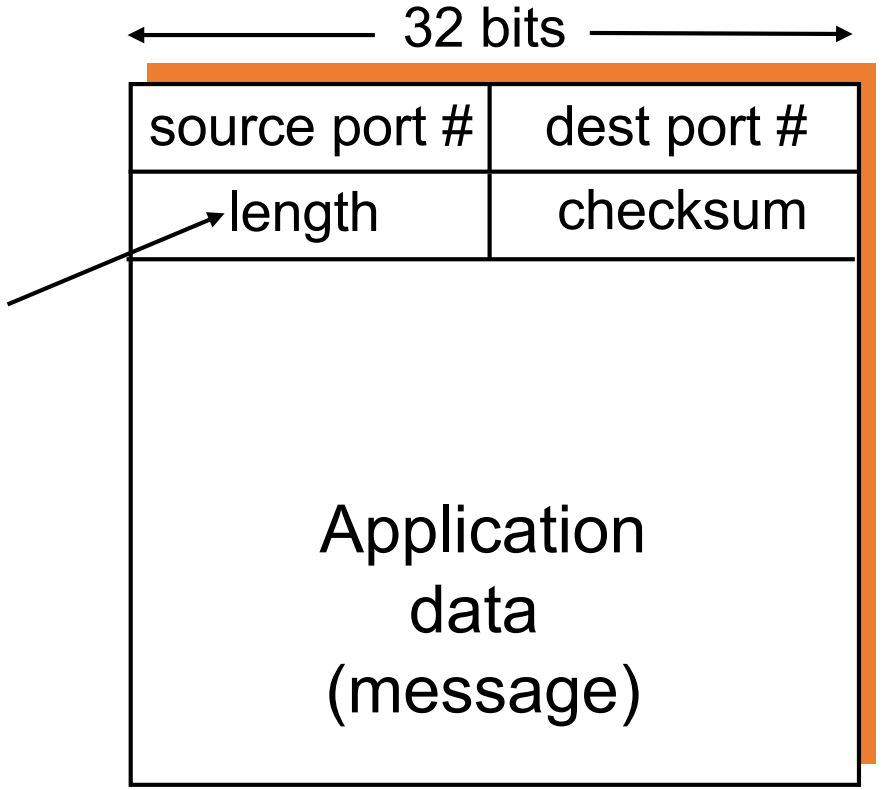
“Best effort” protocols

- Why UDP?
  - No need to establish connection (cause delay)
  - Simple
  - Small header
  - No congestion control → send data as fast as possible

- Main functionality of UDP?
  - MUX/DEMUX
  - Detect error by checksum

Datagram format

- Data unit in UDP is called datagram



Issues of UDP

- No congestion control
  - Cause overload of the Internet
- No reliability
  - Applications have to implement themselves mechanisms to control errors

TCP
Transmission Control Protocol

TCP segment structure

Connection management

Flow control

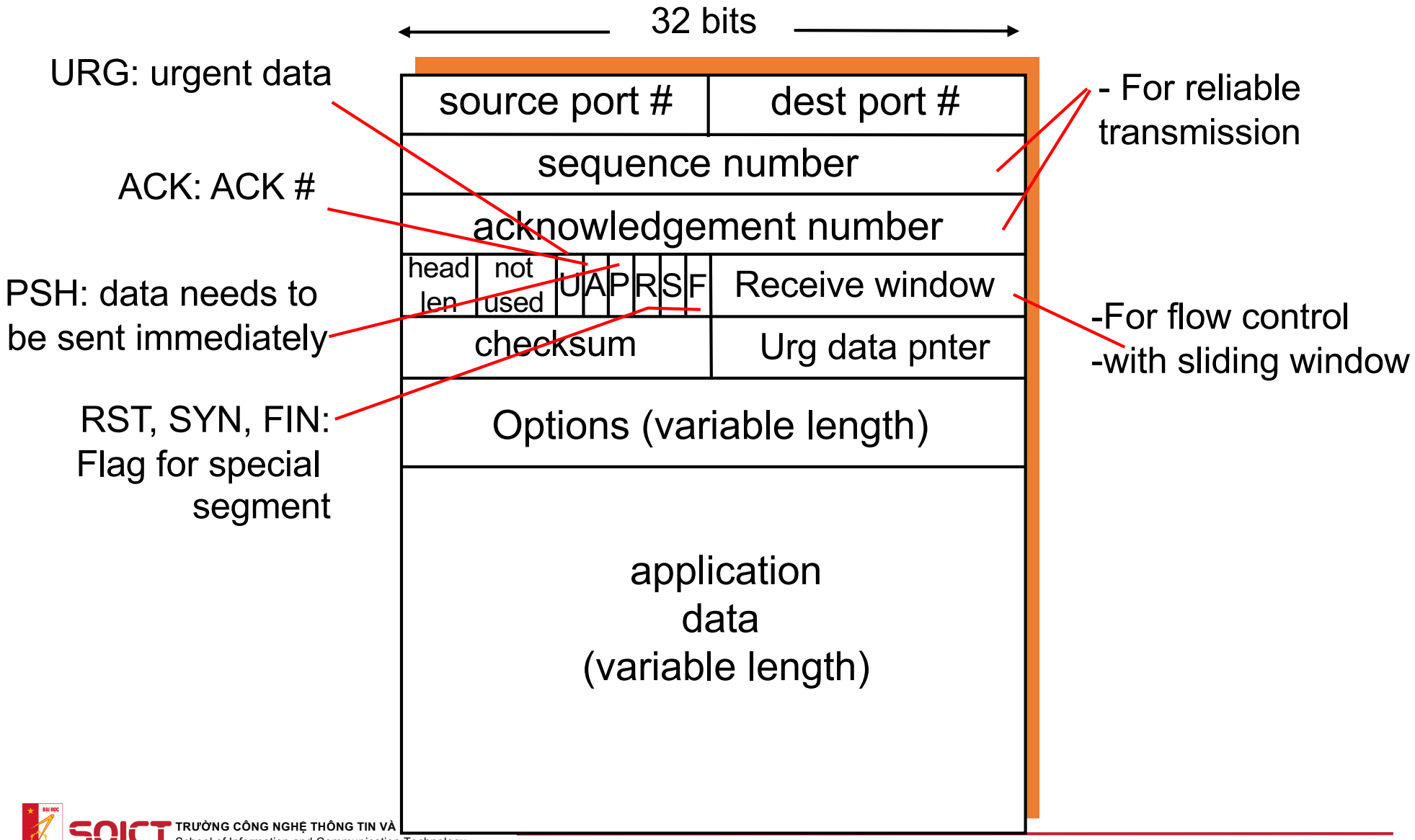
Congestion control

ONE LOVE. ONE FUTURE.

Overview of TCP

- Connection oriented
  - 3 steps hand-shake
- Data transmission in stream of byte, reliable
  - Use buffer
- Transmit data in pipeline
  - Increase the performance
- Flow control
  - Sliding windows
- Congestion control
  - Detect congestion and solve

TCP segment


32 bits
URG: urgent data
ACK: ACK #
PSH: data needs to be sent immediately
RST, SYN, FIN: Flag for special segment
source port #
dest port #
sequence number
acknowledgement number
head len
not used
U
A
P
R
S
F
Receive window
checksum
Urg data pnter
Options (variable length)
application data (variable length)
- For reliable transmission
-For flow control
-with sliding window

How TCP provide reliable service?

- In order to assure if data arrives to destination:
  - Seq. #
  - Ack
- TCP cycle life:
  - Connection establishing
    - 3 steps
  - Data transmission
  - Close connection

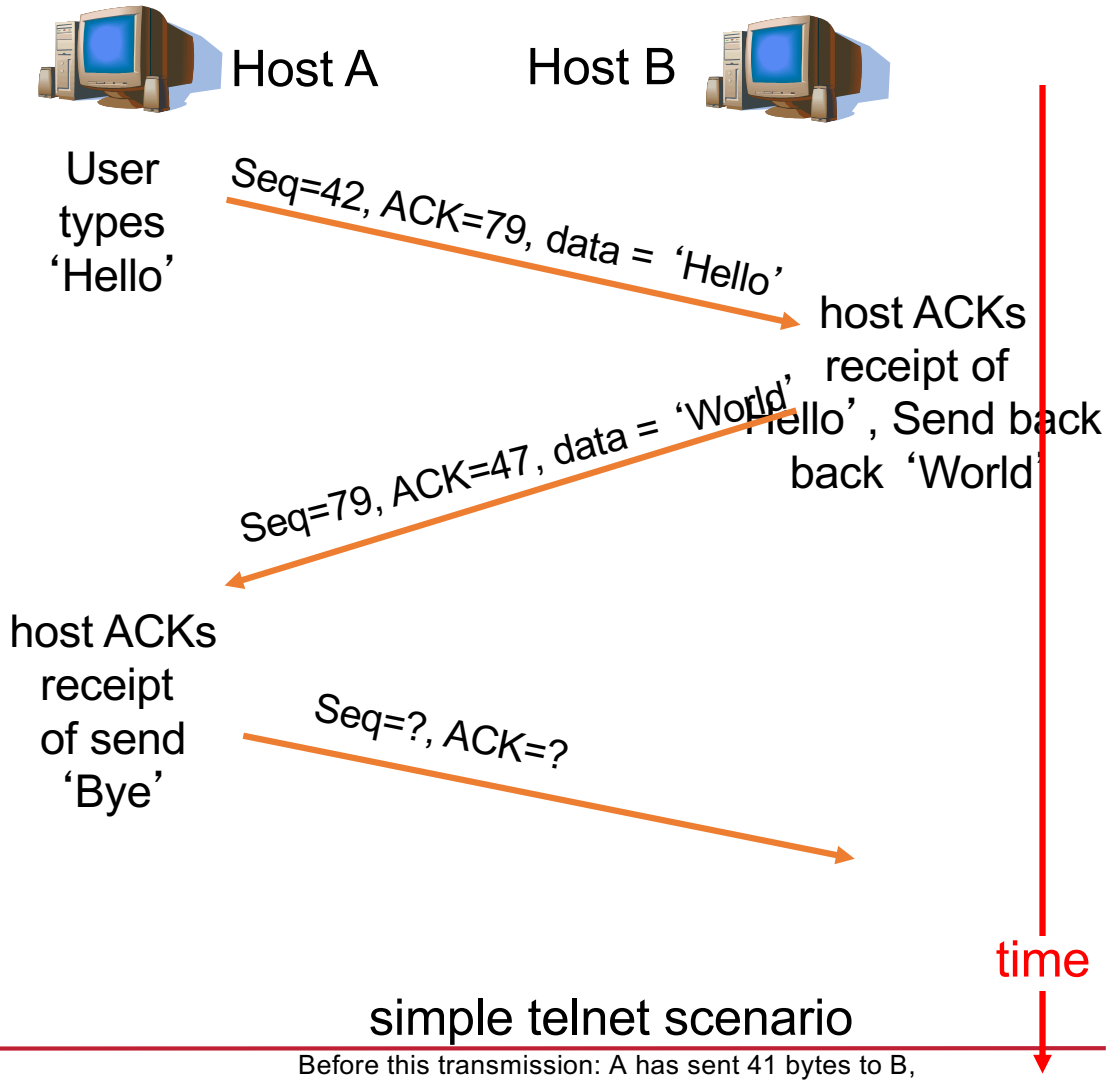
Acknowledgement in TCP

Seq. #:

- Index of the first byte of the segment in the data stream

ACK:

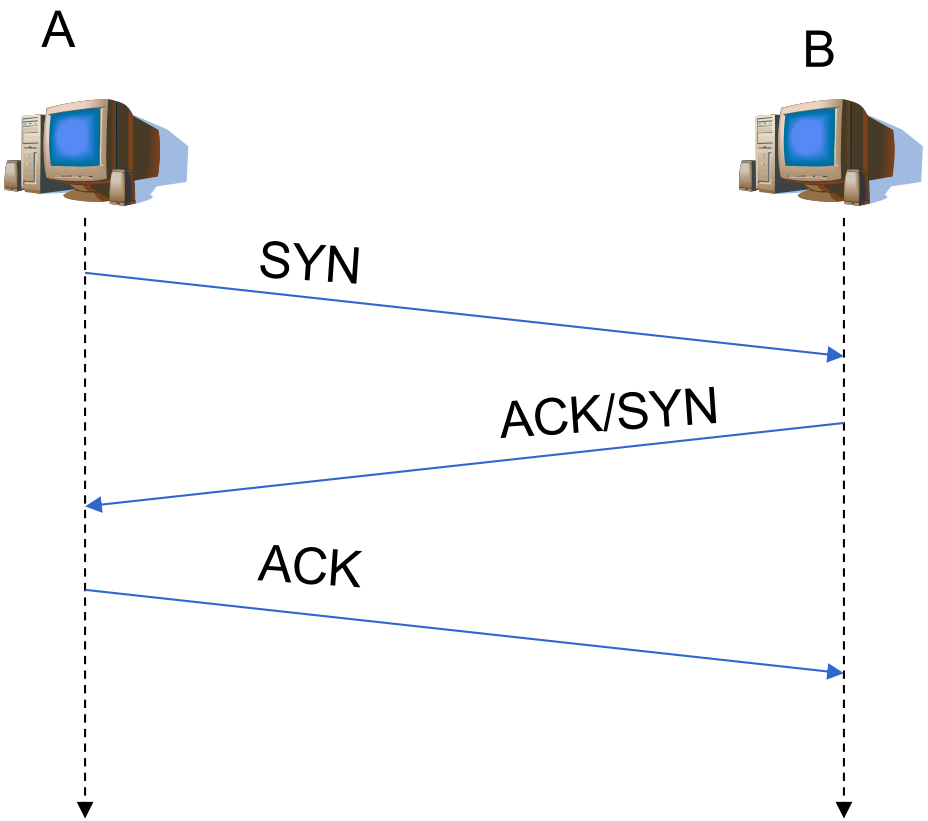
- The index of the first byte expected to receive from the other-side
- Implicitly to confirm that the ACK senders have received well previous bytes



simple telnet scenario

Before this transmission: A has sent 41 bytes to B, and B has sent 78 bytes to A

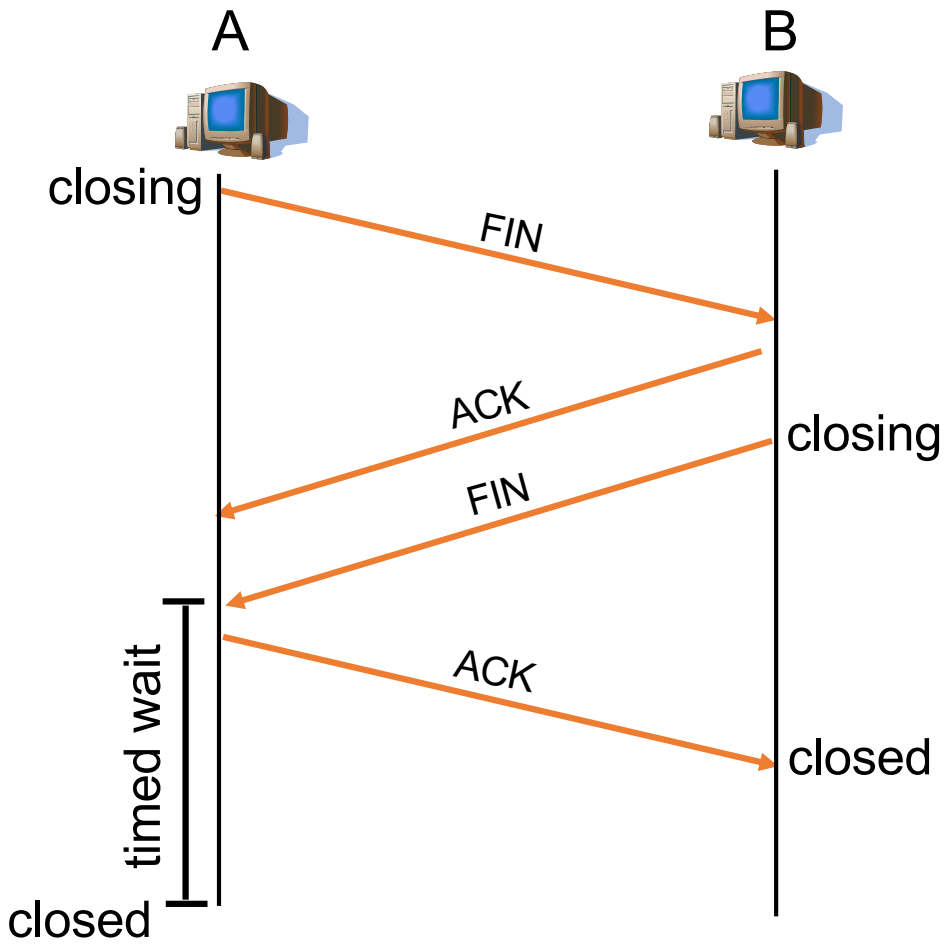
Connection establishing in TCP : 3 steps



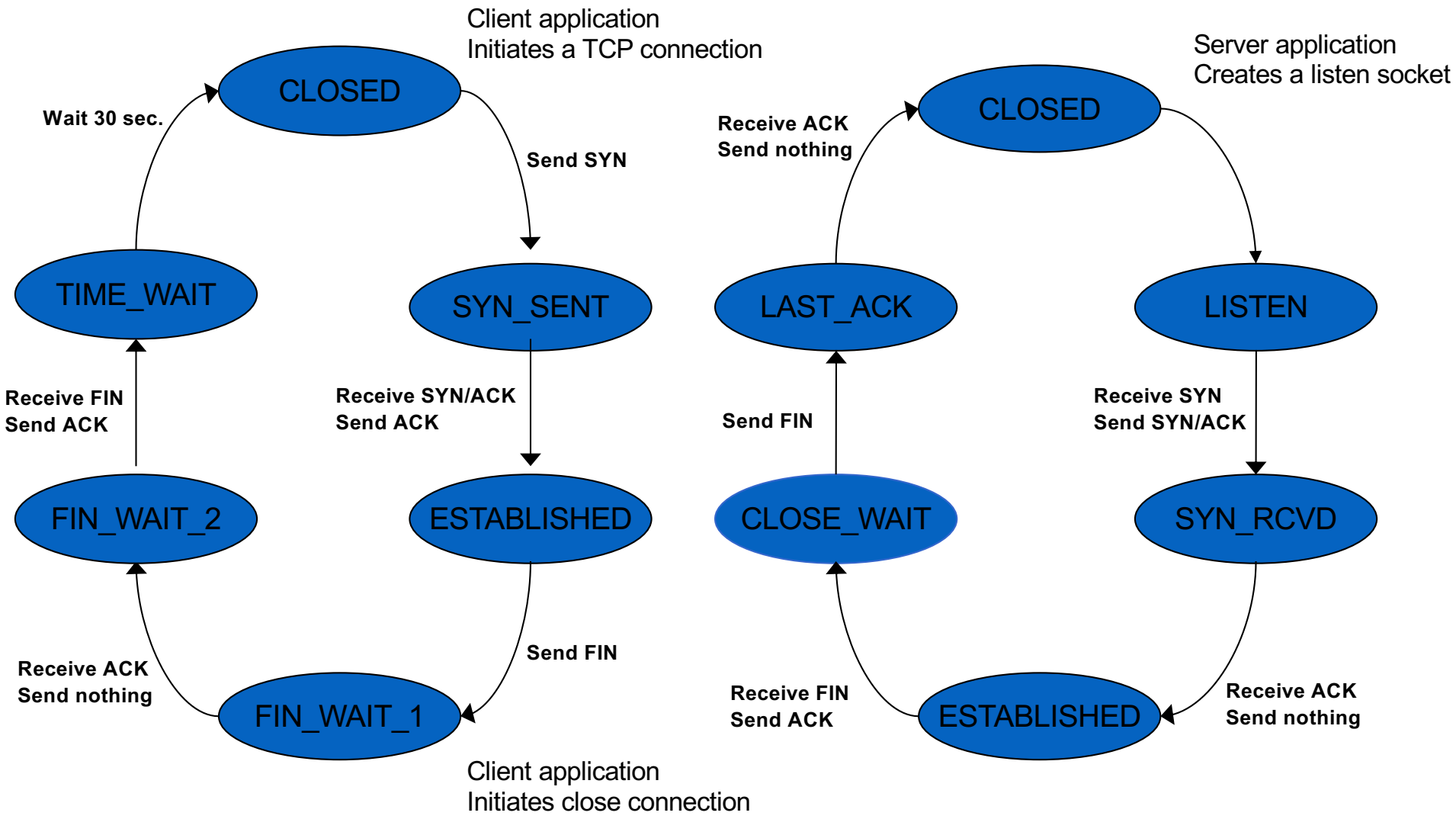
- Step 1: A sends SYN to B
  - Indicate initial value of seq # of A
  - No data
- Step 2: B receives SYN, replies by SYNACK
  - B initiates the buffer on its side
  - Indicate initial value of seq. # of B
- Step 3: A receives SYNACK, replies ACK, maybe with data.

Close connection

- Step 1: Send FIN to B
- Step 2: B receives FIN, replies ACK, closes the connection and sends FIN.
- Step 3: A receives FIN, replies ACK, go to “waiting”.
- Step 4: B receives ACK. close connection

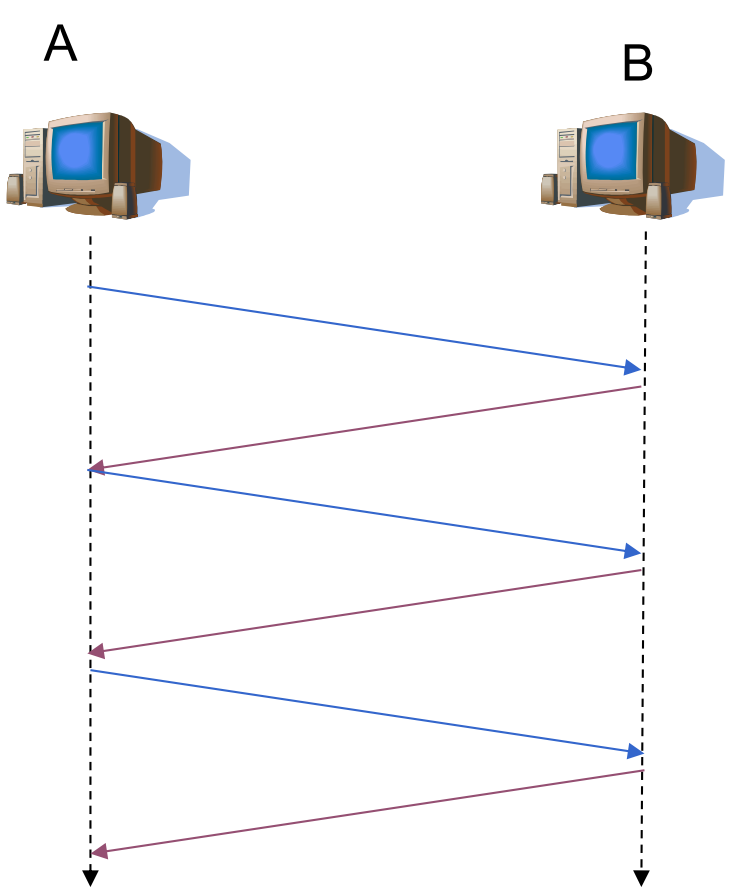


Symplified life cycle of TCP

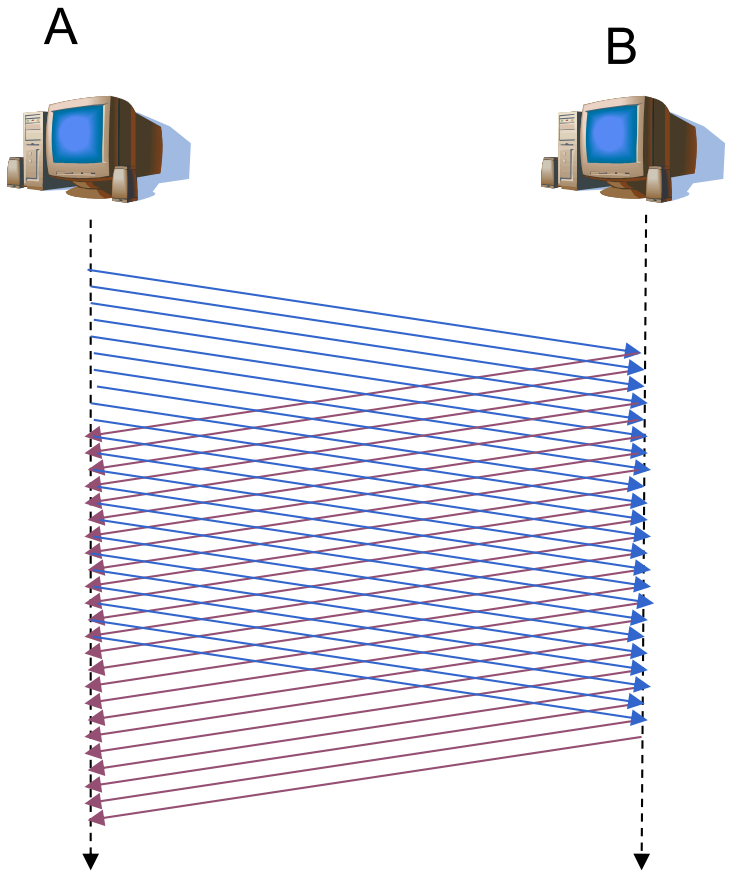


Flow control in TCP

Flow control(1)



Slow

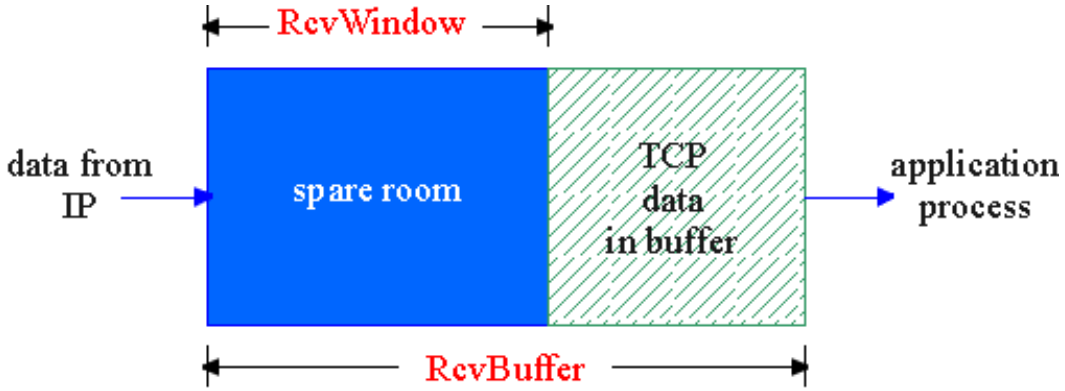


Overload

Flow control (2)

- Control the amount of data to be sent
  - Assure the best efficiency
  - Avoid overloading the receiver.
- TCP uses sliding window for flow control
- Two windows
  - Rwnd: Receive window on receiver side
  - CWnd: Congestion window on sender side
- The maximum amount of data to be sent should be min(Rwnd, Cwnd)

Flow control TCP

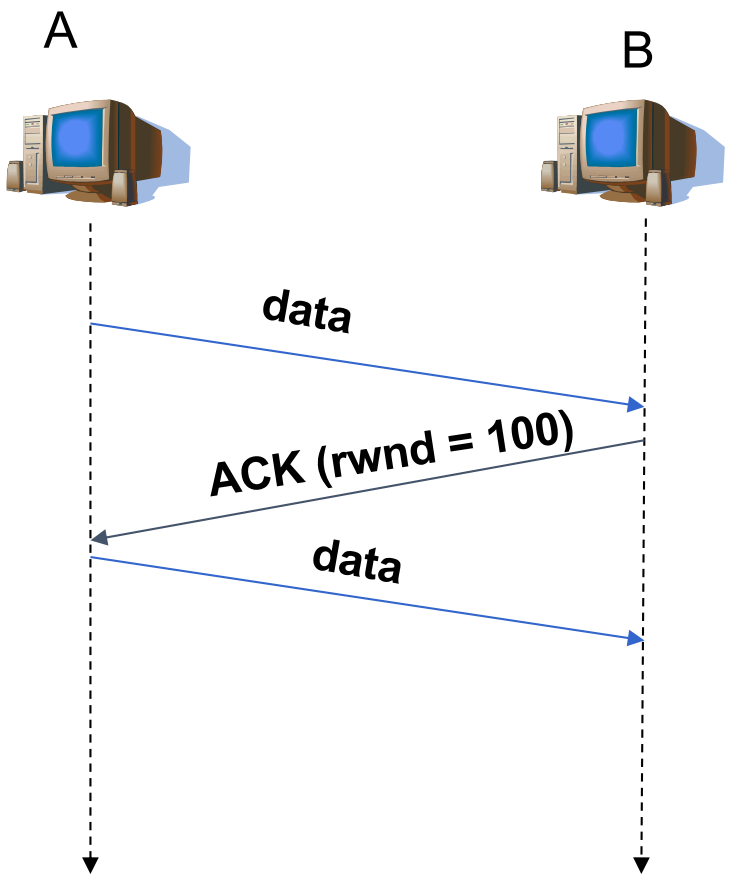


- Size of free buffer

= Rwnd

= **RcvBuffer-[LastByteRcvd - LastByteRead]**

Information exchanged on Rwnd

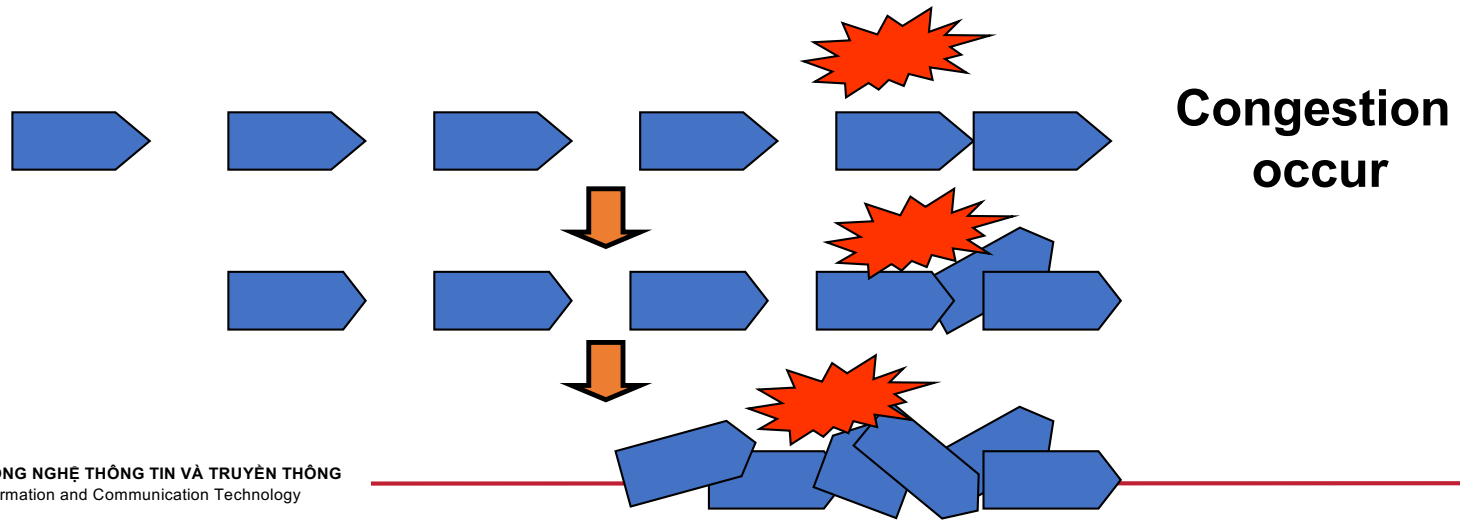


- Receiver inform regularly to senders the value of `Rwnd` in acknowledgment segments

Congestion control in TCP

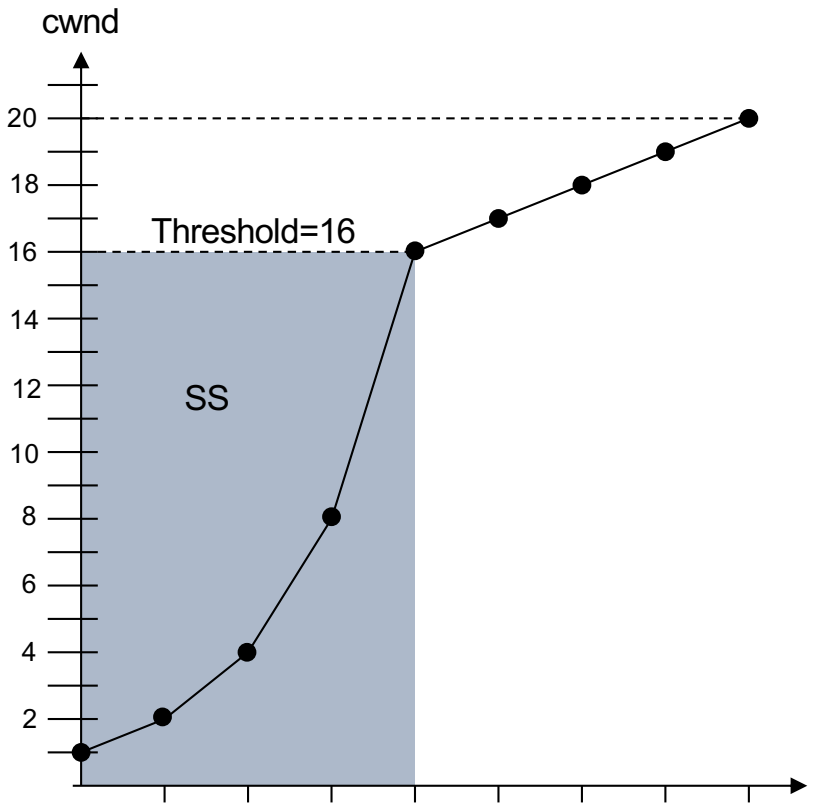
Overview of Congestion control

- When congestion happens?
  - Too many pairs of senders-receivers in the network
  - High traffic
- Consequence of congestion
  - Packet loss
  - Reduce of throughput, increase of delay
  - Network situation become worst with reliable protocol such as TCP.



Principles of congestion control

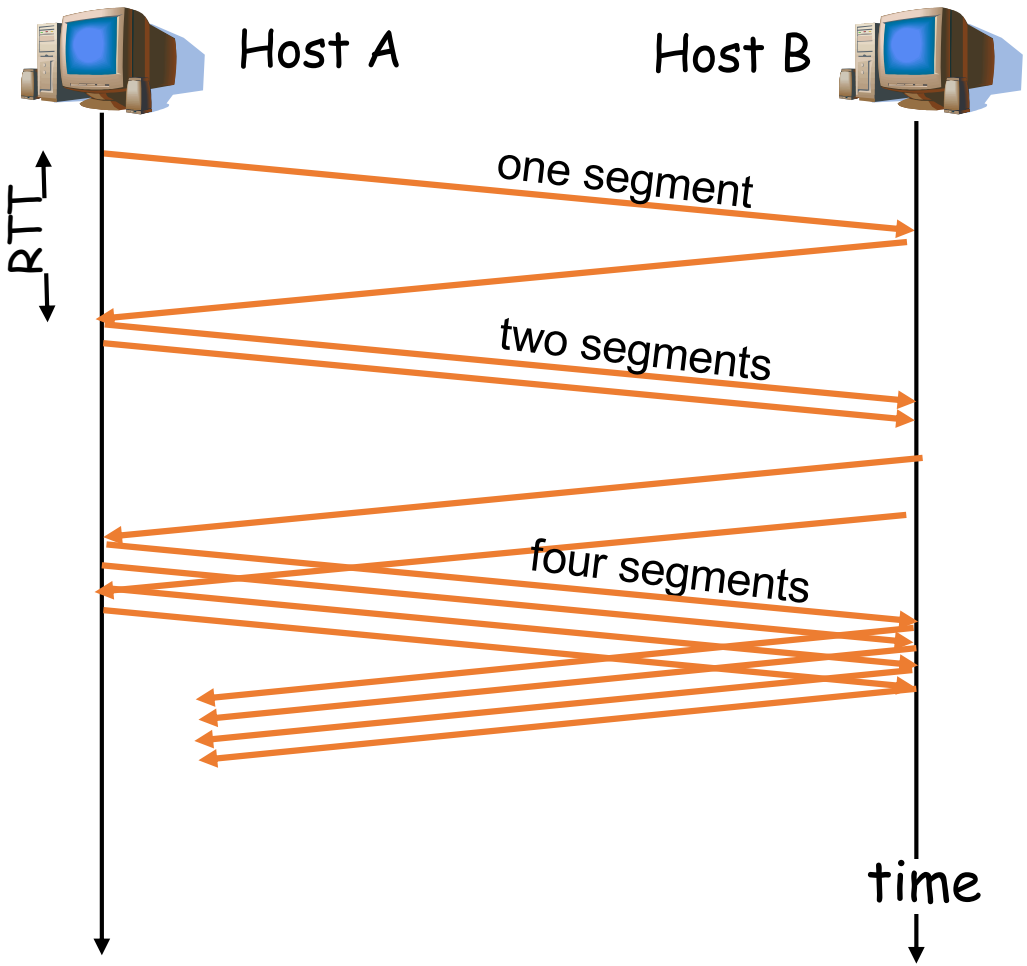
- Slow-start
  - Increases the transmission speed in exponential order
  - Increase until a threshold
- Congestion avoidance
  - Increase the transmission speed in linear order until congestion is detected
- How to detect the congestion?
  - By packets lost?



TCP Slow Start (1)

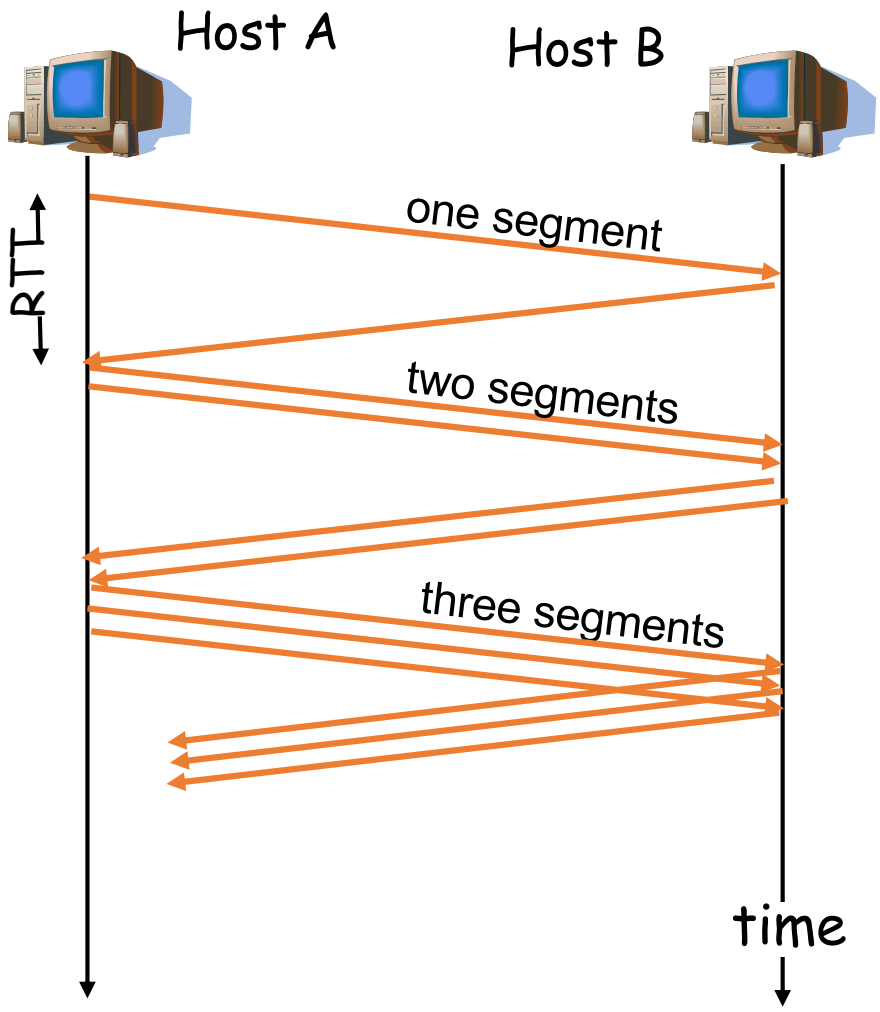
- Main idea
  - Cwnd: congestion window = data size that source can send to destination without waiting for ACK
    - Rwnd (sliding windows in flow control): maximum data size to be sent without ACK according to flow control
    - Actual data size can be sent without ACK = min (Cwnd, Rwnd)
  - Initiate cwnd =1 MSS (Maximum segment size)
  - Increase cwnd =+1 MSS after each reception of a ACK packet from the receiver.
  - Increase slowly but the speed increase in exponential order
- Increase until a threshold: ssthresh
- After that TCP move to congestion avoidance period

TCP Slow Start (2)



Congestion avoidance

- Main idea
  - Increase cwnd in additional order until cwnd reaches to congestion
- After each RTT, cwnd =cwnd + 1 MSS



TCP reaction in congestion situation (1)

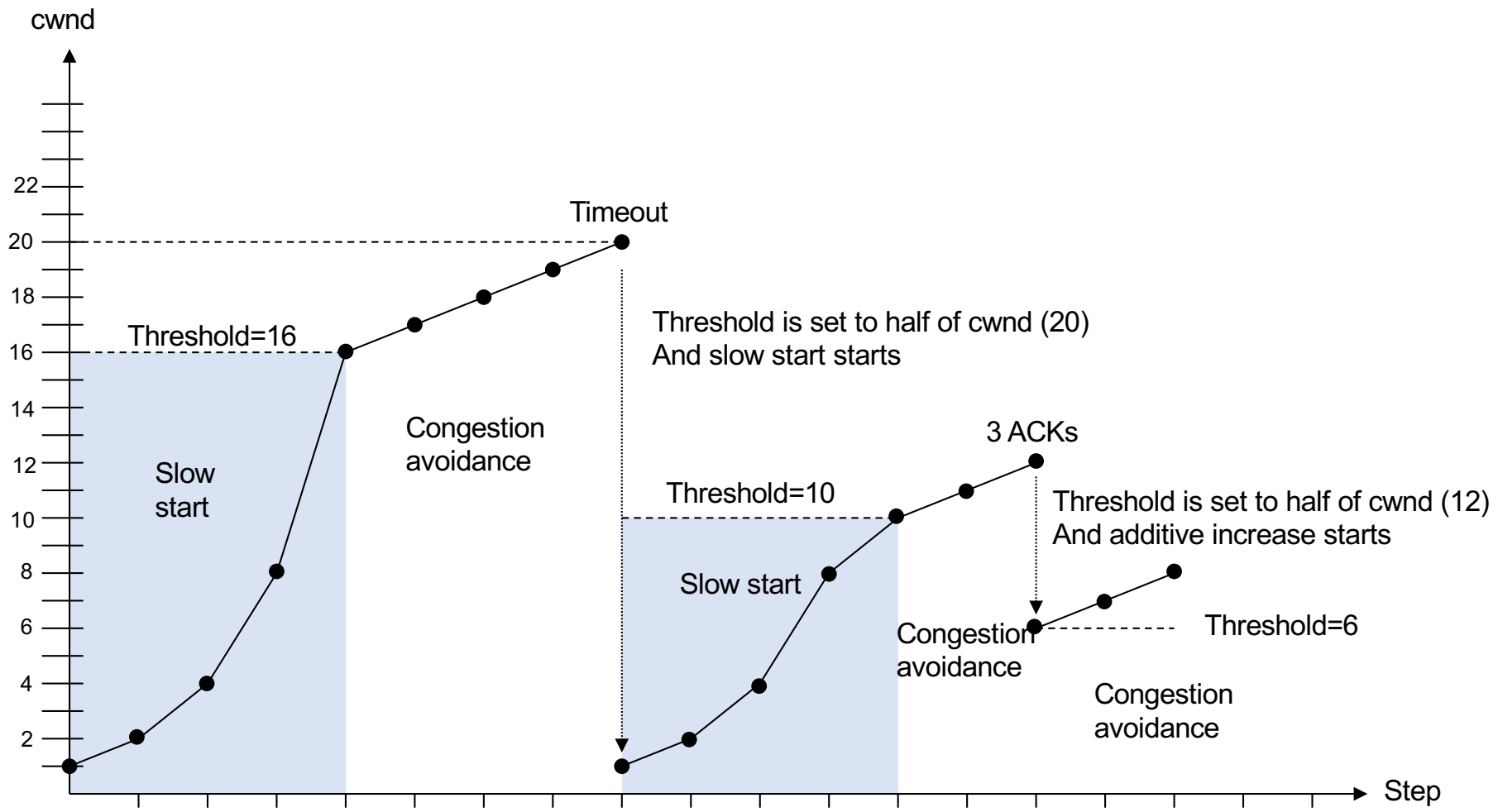
- Reduce the transmission speed
- How to detect the congestion?
  - If there are some re-transmits ➔ There might be congestion
- When the source node need to re-transmit data?
  - Timeout!
  - When it receives multiple ACK for the same segment

TCP reaction in congestion situation(2)

- When sender reach timeout but still does not receive ACK for a segment
  - TCP sets ssthresh = ½ current cwnd
  - TCP sets cwnd =1 MSS
  - TCP move to slow start phase

- If sender receives 3 identical ACK
  - TCP sets ssthresh = ½ current cwnd
  - TCP move to “congestion avoidance”

Congestion control – illustration



Exercise

- Assume that we need transmit 1 file
  - File size *O* =100KB over TCP connection
  - S is the size of each TCP segment, *S* = 536 byte
  - *RTT* = 100 *ms*.
- Assume that the congestion window size of TCP is fixed with value W.

  What is the minimum transmission time? If the transmission speed is
  - R = 10 Mbit/s;
  - R= 100 Mbits/s.

Exercise

- Assume that we need transmit 1 file
  - File size O =100KB over TCP connection
  - S is the size of each TCP segment, *S* = 536 byte
  - *RTT* = 100 *ms*.
- Assume that the congestion window of TCP works according to slow-start mechanism (ssthreshold = infinity).
- What is the size of the congestion window when the whole file is transmitted.
- How much of time is required for transmitting the file? If R = 10 Mbit/s; R= 100 Mbits/s.