Overview of TCP/IP
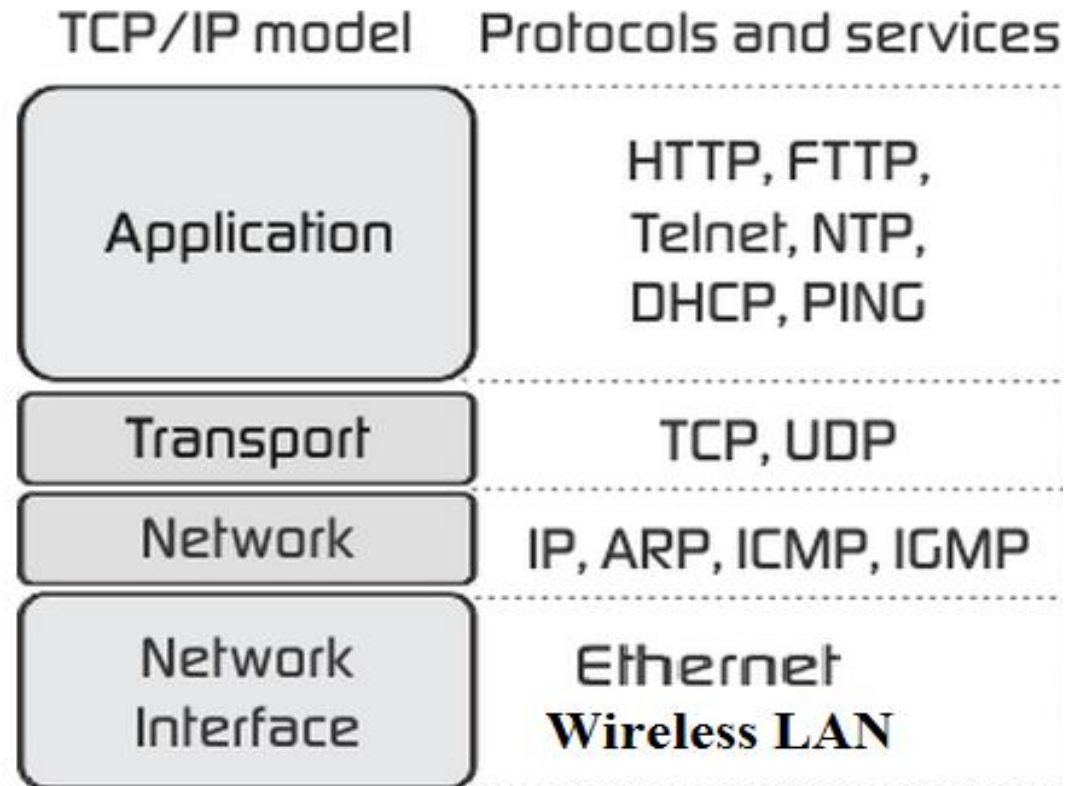
Beulah A.

AP/CSE

Architecture of TCP/IP

- TCP
  - Telnet, SMTP, FTP
  - HTTP
- UDP
  - DNS
  - SNMP

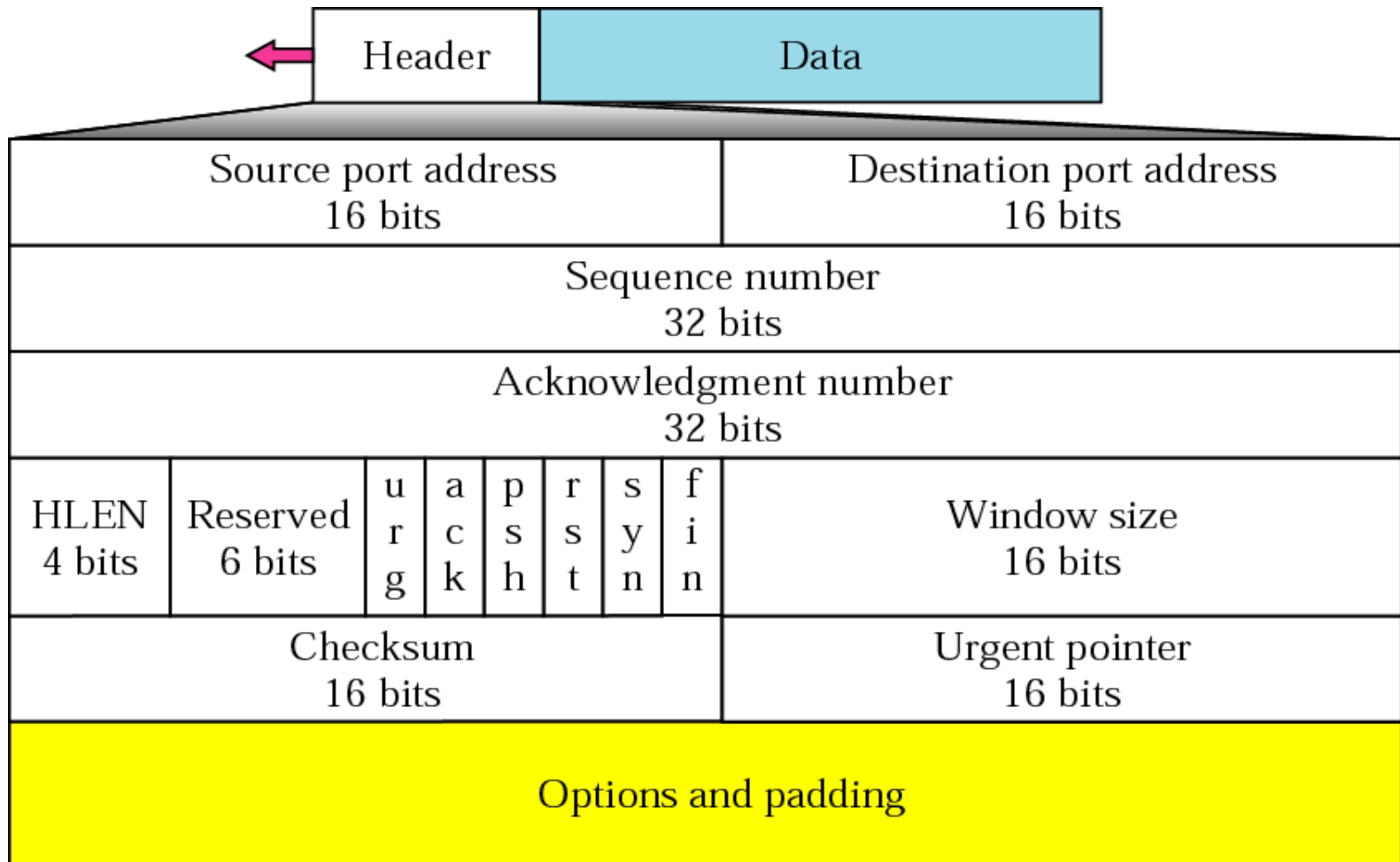| TCP/IP model	Protocols and services
Application	HTTP, FTTP, Telnet, NTP, DHCP, PING
Transport	TCP, UDP
Network	IP, ARP, ICMP, IGMP
Network Interface	Ethernet **Wireless LAN**

Unit II   Beulah A.   30-Jan-17

Terminologies of TCP/IP

- TCP
- IP
- HTTP
- SMTP
- MIME
- FTP
- SNMP

- ARP
- RARP
- BOOTP
- Routers
- DNS
- IP Address
- ICMP
- IGMP

Overview of TCP Operations

- TCP Segment

- IP datagram

- Port address

- Data Encapsulation

TCP Segment

▸ 20 or 60 byte header

Header	Data

Source port address 16 bits							Destination port address 16 bits	
Sequence number 32 bits								
Acknowledgment number 32 bits								
HLEN 4 bits	Reserved 6 bits	u r g	a c k	p s h	r s t	s y n	f i n	Window size 16 bits
Checksum 16 bits							Urgent pointer 16 bits	
Options and padding								

Unit II   Beulah A.   30-Jan-17

Numbering System

- Byte Number

- Sequence Number

- Acknowledgment Number

Byte Number

▸ Each byte should be numbered

▸ Random number – 1057

▸ Data contains 6000 bytes

▸ 1057 – 7056

Unit II   Beulah A.   30-Jan-17

Sequence Number

▸ Sequence number for each segment is the number of the first byte carried in that segment

▸ Imagine a TCP connection is transferring a file of 6000 bytes. The first byte is numbered 10010. What are the sequence numbers for each segment if data are sent in five segments with the first four segments carrying 1000 bytes and the last segment carrying 2000 bytes?

Unit II   Beulah A.   30-Jan-17

Sequence Number

▸ The following shows the sequence number for each segment:

Segment 1 ==> sequence number: 10,010

(range: 10,010 to 11,009)

Segment 2 ==> sequence number: 11,010

(range: 11,010 to 12,009)

Segment 3 ==> sequence number: 12,010

(range: 12,010 to 13,009)

Segment 4 ==> sequence number: 13,010
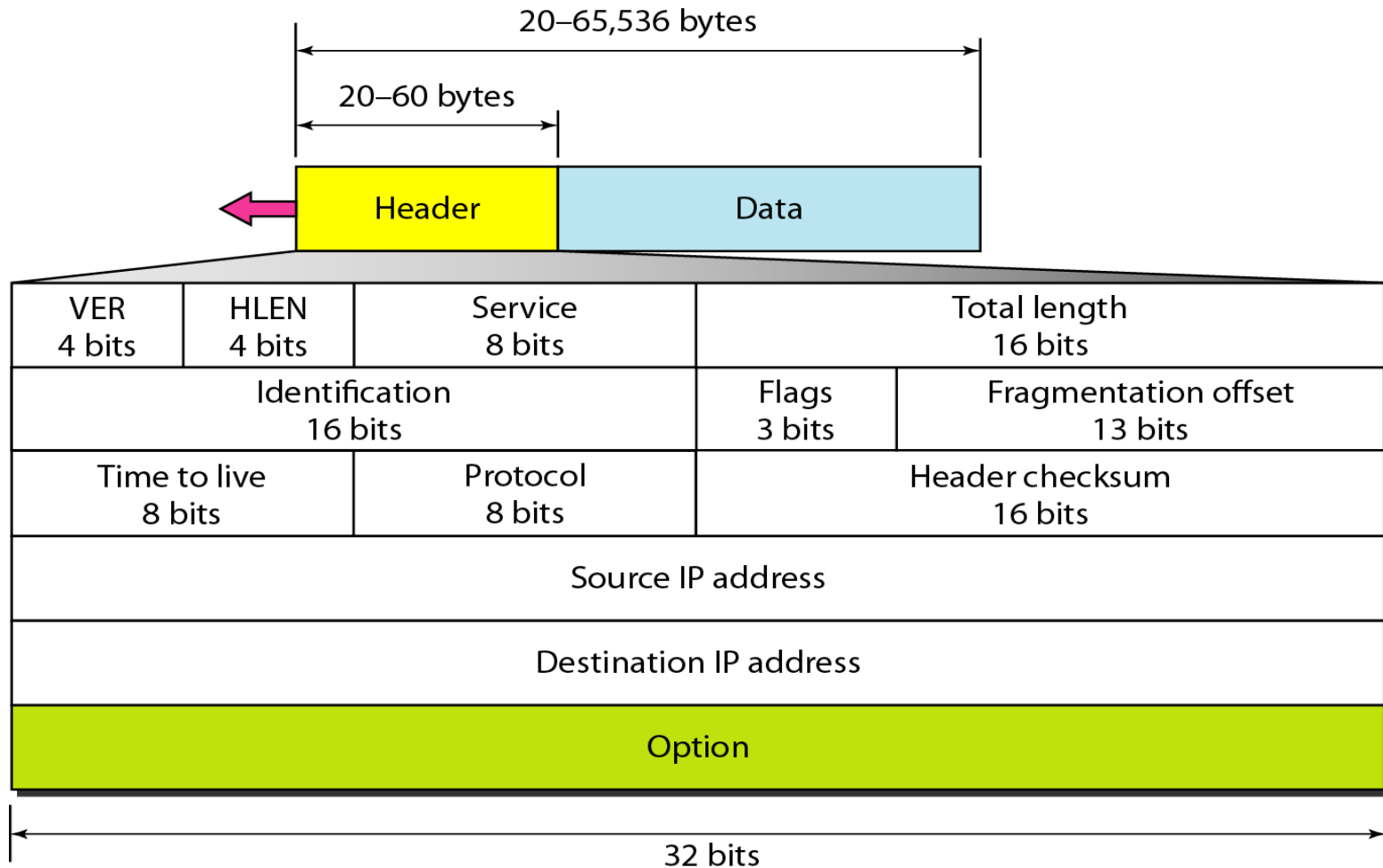
(range: 13,010 to 14,009)

Segment 5 ==> sequence number: 14,010

(range: 14,010 to 16,009)

Acknowledgment Number

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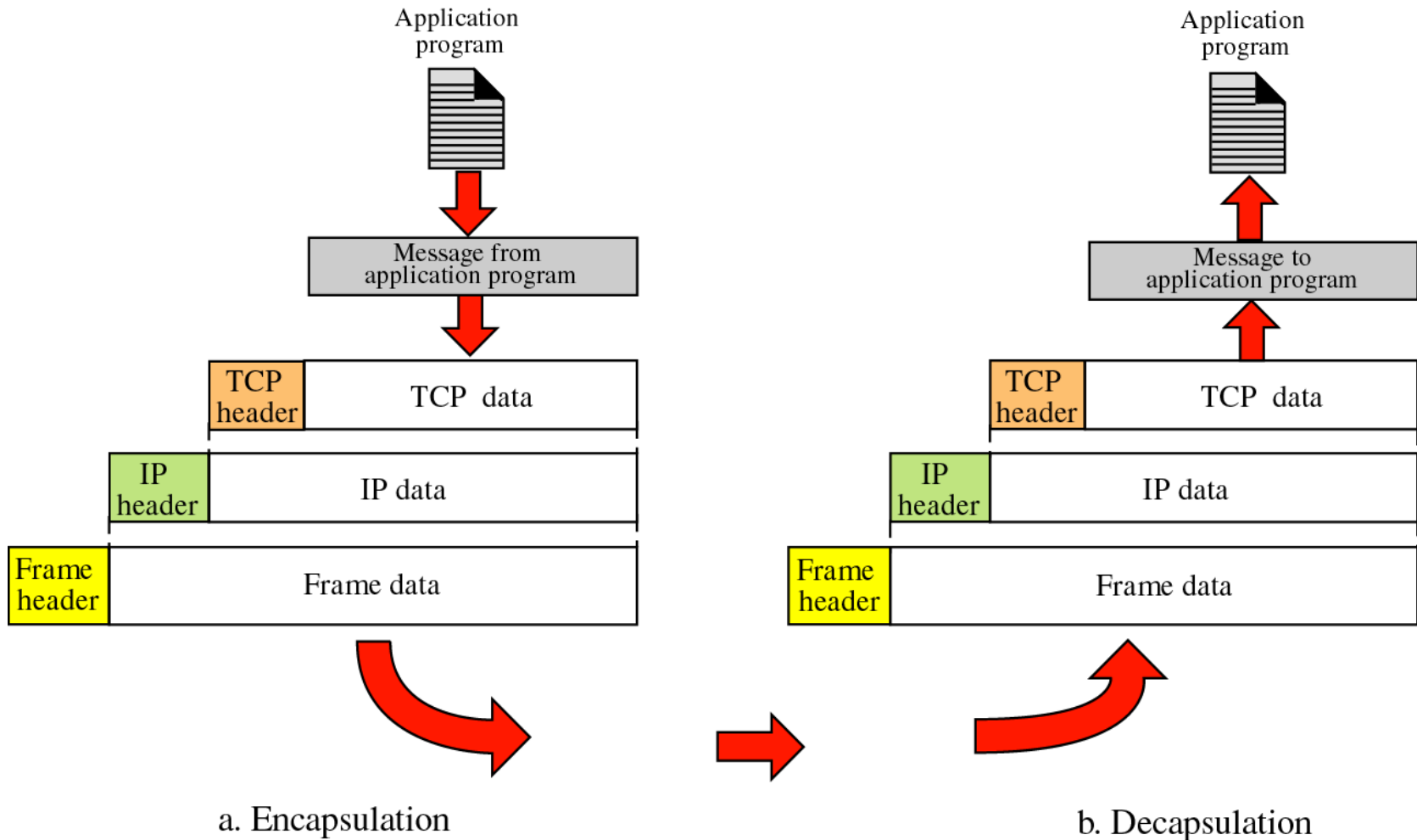
▶ The acknowledgment number is cumulative.

Unit II Beulah A. 30-Jan-17

IP Datagram

20–65,536 bytes

20–60 bytes

Header	Data

VER 4 bits	HLEN 4 bits	Service 8 bits	Total length 16 bits	
Identification 16 bits			Flags 3 bits	Fragmentation offset 13 bits
Time to live 8 bits		Protocol 8 bits	Header checksum 16 bits	
Source IP address				
Destination IP address				
Option				

32 bits

Port Nos.

Port	Protocol	Description
7	Echo	Echoes a received datagram back to the sender
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	FTP, Data	File Transfer Protocol (data connection)
21	FTP, Control	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Server
67	BOOTP	Bootstrap Protocol
79	Finger	Finger
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

Unit II   Beulah A.   30-Jan-17

Data Encapsulation

a. Encapsulation

b. Decapsulation

Application Layer Protocols of TCP

- SMTP
- FTP
- Telnet

Unit II   Beulah A.   30-Jan-17

TCP/IP vs OSI/ISO Model

OSI Layer		
Application	**Applications**	
Presentation	SMTP · FTP · HTTP · DNS · SNMP · TELNET · ...	
Session		

Transport	SCTP · TCP · UDP

Network (internet)	ICMP · IGMP · **IP** · RARP · ARP

Data link · Physical	Protocols defined by the underlying networks (host-to-network)
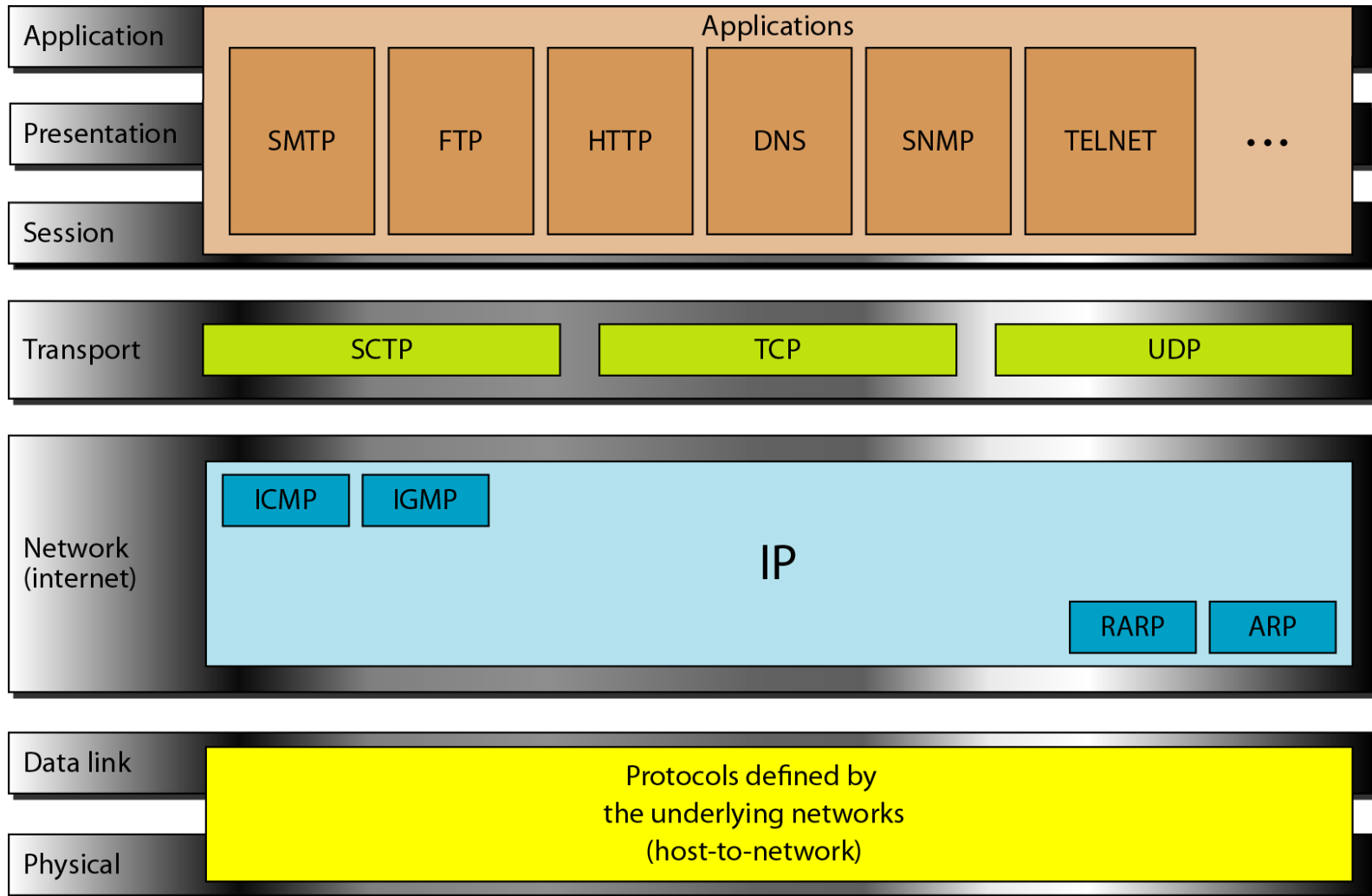
Adaptation of TCP Window

Basic Concepts

- E.g. HTTP (used by web services) typically uses TCP
  - Reliable transport between client and server required
- TCP
  - Steam oriented, not transaction oriented
  - Packet Loss
    - → congestion
    - → slow down transmission
- Well known – TCP guesses quite often wrong in wireless and mobile networks
  - Packet loss due to transmission errors
  - Packet loss due to change of network
- Result
  - Severe performance degradation

Client   Server

TCP SYN
TCP SYN/ACK   } Connection setup
TCP ACK
HTTP request
HTTP response   } Data transmission

>15 s no data

Connection release
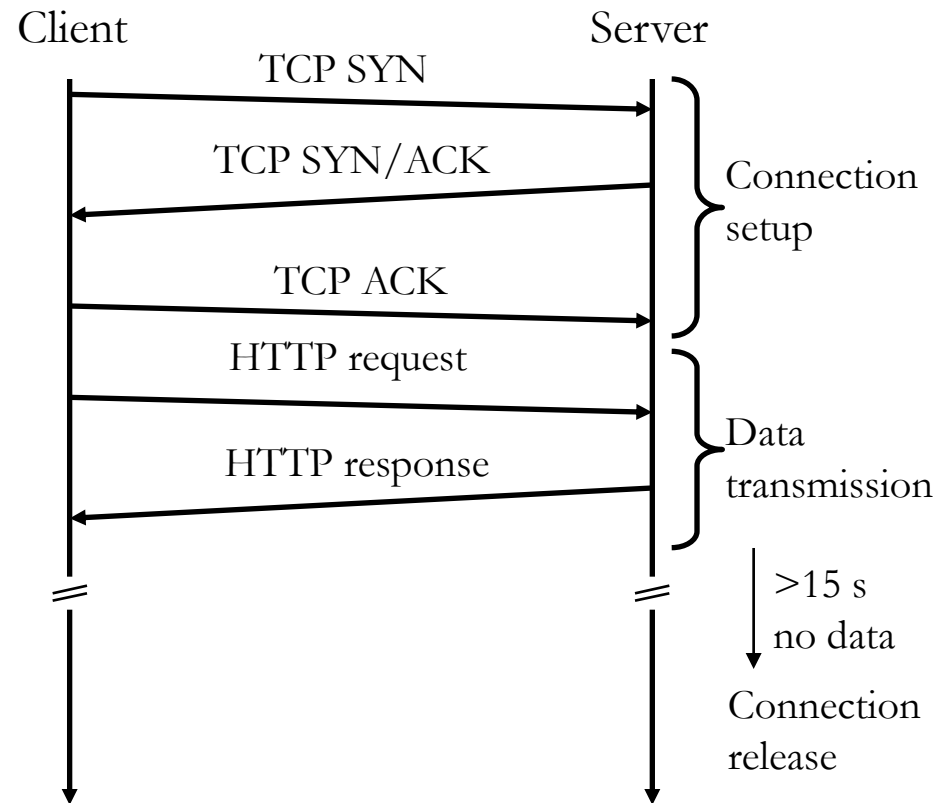
Additive Increase

- Additive Increase is a reaction to perceived available capacity.

- Linear Increase → For each "cwnd's worth" of packets sent, increase cwnd by 1 packet.

- In practice, cwnd is incremented exponentially for each arriving ACK.

Source   Destination

Unit II   Beulah A.   30-Jan-17
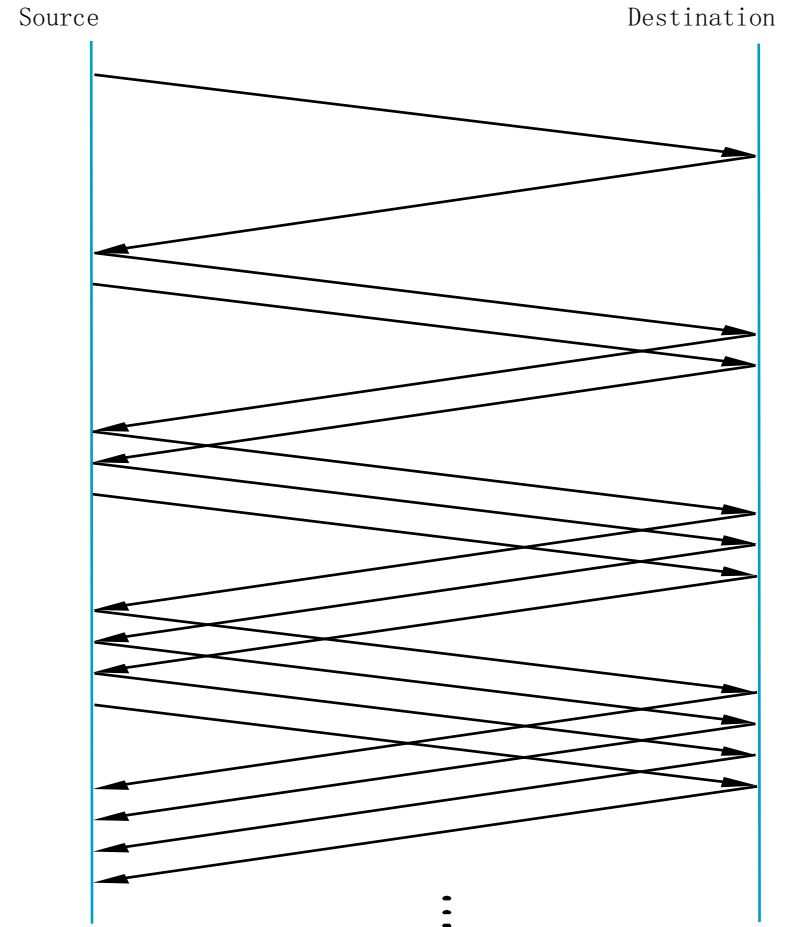
Silly Window Syndrome

- If a receiver with this problem is unable to process all incoming data, it requests that its sender reduce the amount of data they send at a time.

- MSS/2

- If the receiver continues to be unable to process all incoming data, the window becomes smaller and smaller, sometimes to the point that the data transmitted is smaller than the packet header, making data transmission extremely inefficient.

- The name of this problem is due to the window size shrinking to a "silly" value.

- Nagel's Algorithm

Adaptive Retransmission

▸ TCP achieves reliability by retransmitting segments after a Timeout

▸ Choosing the value of the Timeout

  ▸ Set time out as a function as RTT

  ▸ If too small, retransmit unnecessarily

  ▸ If too large, poor throughput

  ▸ Make this adaptive, to respond to changing congestion delays in Internet

Adaptive Retransmission
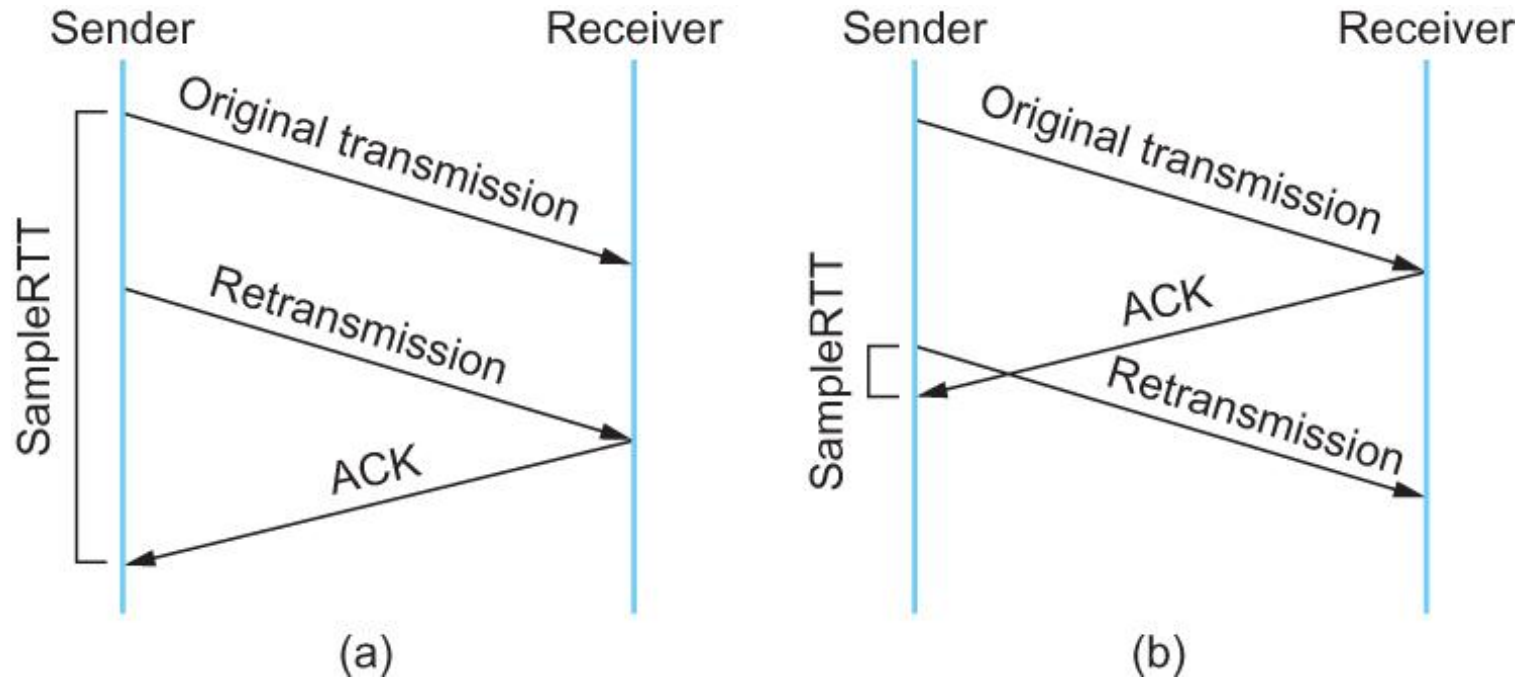
Associating the ACK with (a) original transmission versus (b) retransmission

Improvement in TCP Performance

Improvement in TCP Performance

▸ Traditional TCP

  ▸ Slow Start

  ▸ Fast Retransmit and recovery

▸ Popular TCP Congestion Control Algorithms

  ▸ TCP Tahoe

  ▸ TCP Reno
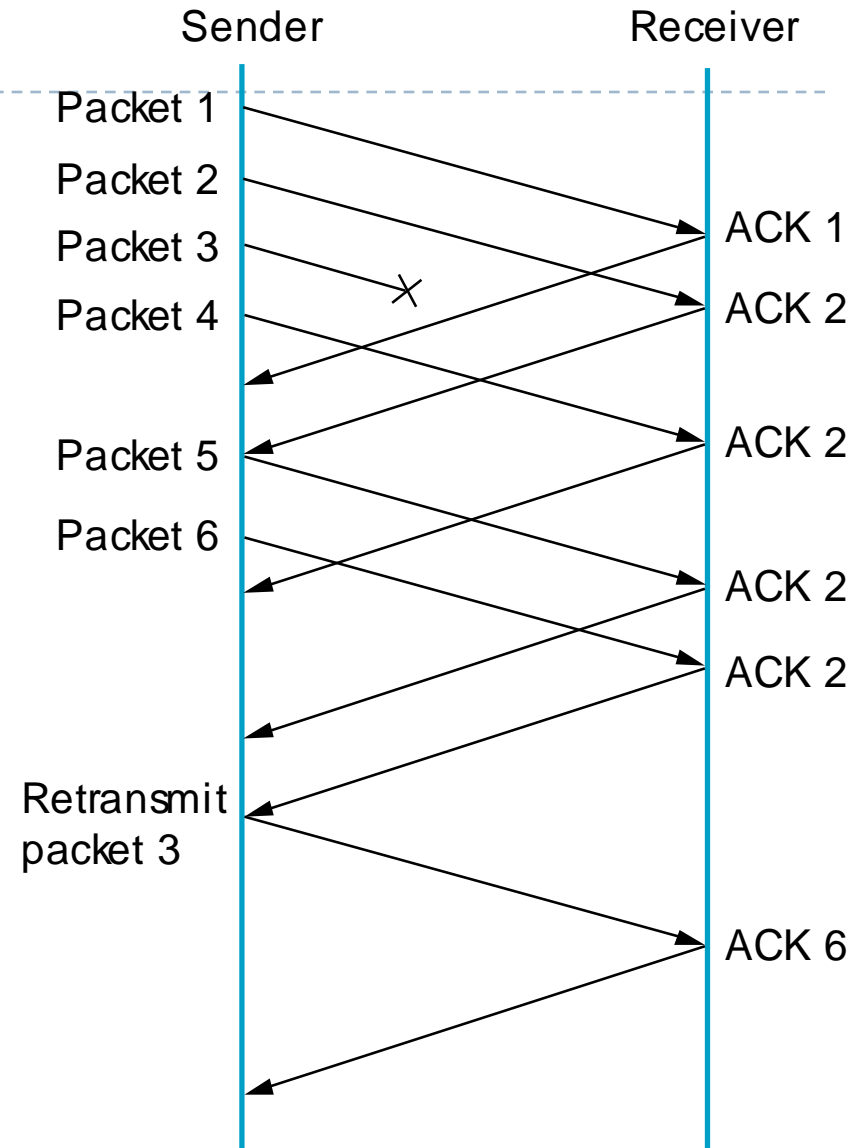
  ▸ TCP SACK

  ▸ TCP Vegas

▸ TCP in Mobile Networks

Unit II   Beulah A.   30-Jan-17

TCP Slow-Start algorithm

- Sender calculates a congestion window for a receiver
- Start with a congestion window size equal to one segment
- Exponential increase of the congestion window up to the congestion threshold, then linear increase
- Missing acknowledgement causes the reduction of the congestion threshold to one half of the current congestion window
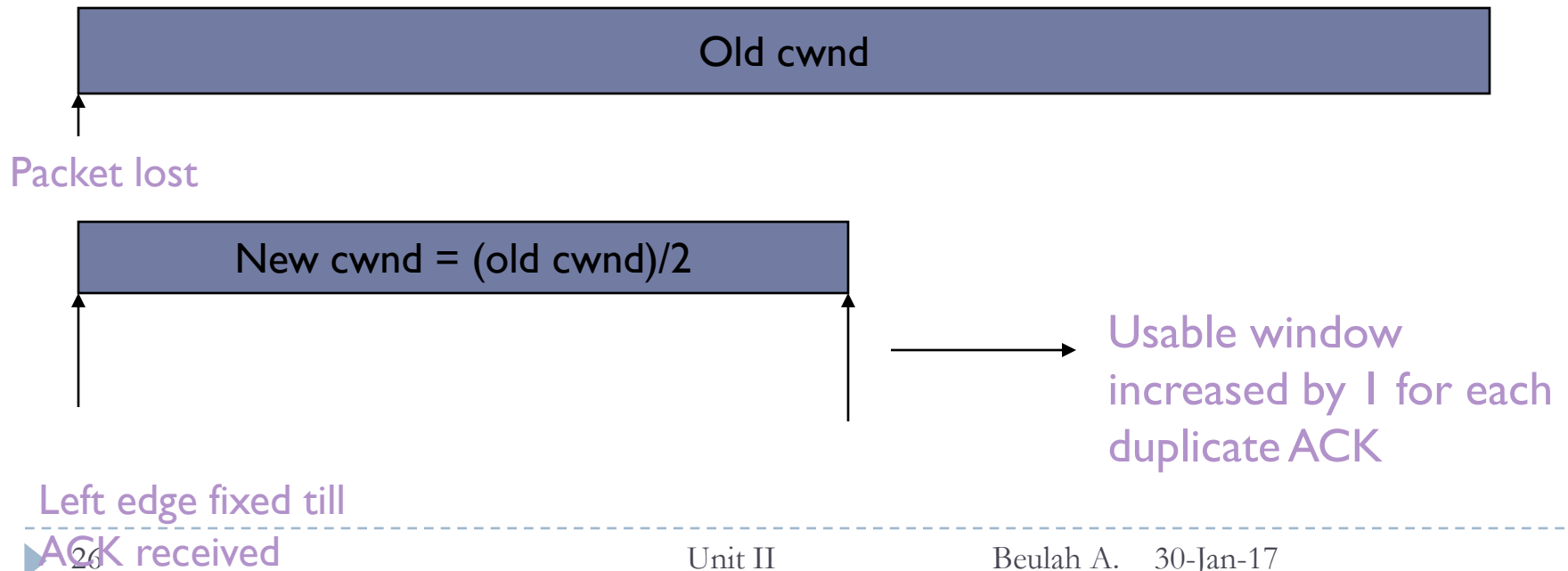- Congestion window starts again with one segment

Host A   Host B

RTT

one segment

two segments

four segments

time

Unit II   Beulah A.   30-Jan-17
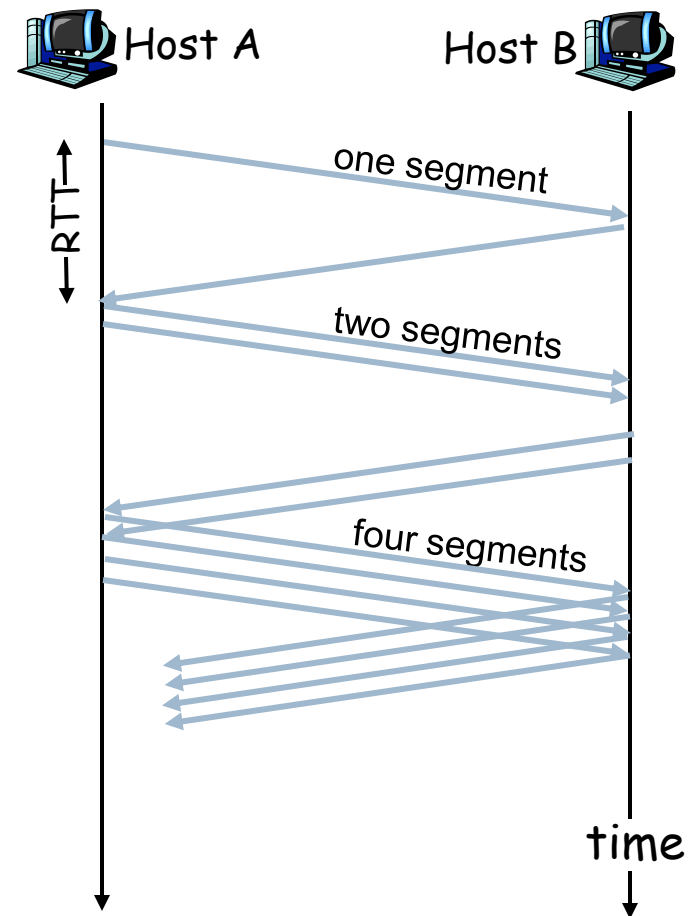
Fast Retransmit

- Coarse timeouts remained a problem, and Fast retransmit was added with TCP.

- Since the receiver responds every time a packet arrives, this implies the sender will see duplicate ACKs.

- Use duplicate ACKs to signal lost packet.

- Upon receipt of *three duplicate ACKs,* the TCP Sender retransmits the lost packet.

Sender   Receiver

Packet 1
Packet 2   ACK 1
Packet 3   ACK 2
Packet 4
   ACK 2
Packet 5
Packet 6   ACK 2
   ACK 2
Retransmit packet 3
   ACK 6

Fast Recovery

‣ Fast recovery was added with TCP.

‣ When fast retransmit detects three duplicate ACKs, start the recovery process from congestion

‣ After Fast Retransmit, half the cwnd and commence *recovery from this point using linear additive increase.*

Old cwnd

Packet lost

New cwnd = (old cwnd)/2

Usable window increased by 1 for each duplicate ACK

Left edge fixed till ACK received

Popular TCP Congestion Control Algs

▸ Comparison

  ▸ Tahoe: Slow start, fast retransmit

  ▸ Reno: Tahoe + fast recovery

  ▸ New-Reno: Reno with modified fast recovery

  ▸ SACK: Reno + selective ACKs

  ▸ Vegas: Modified Slow start, and retransmission

TCP Tahoe

▶ Slow start, fast retransmit

▶ Fast retransmit improves channel utilization
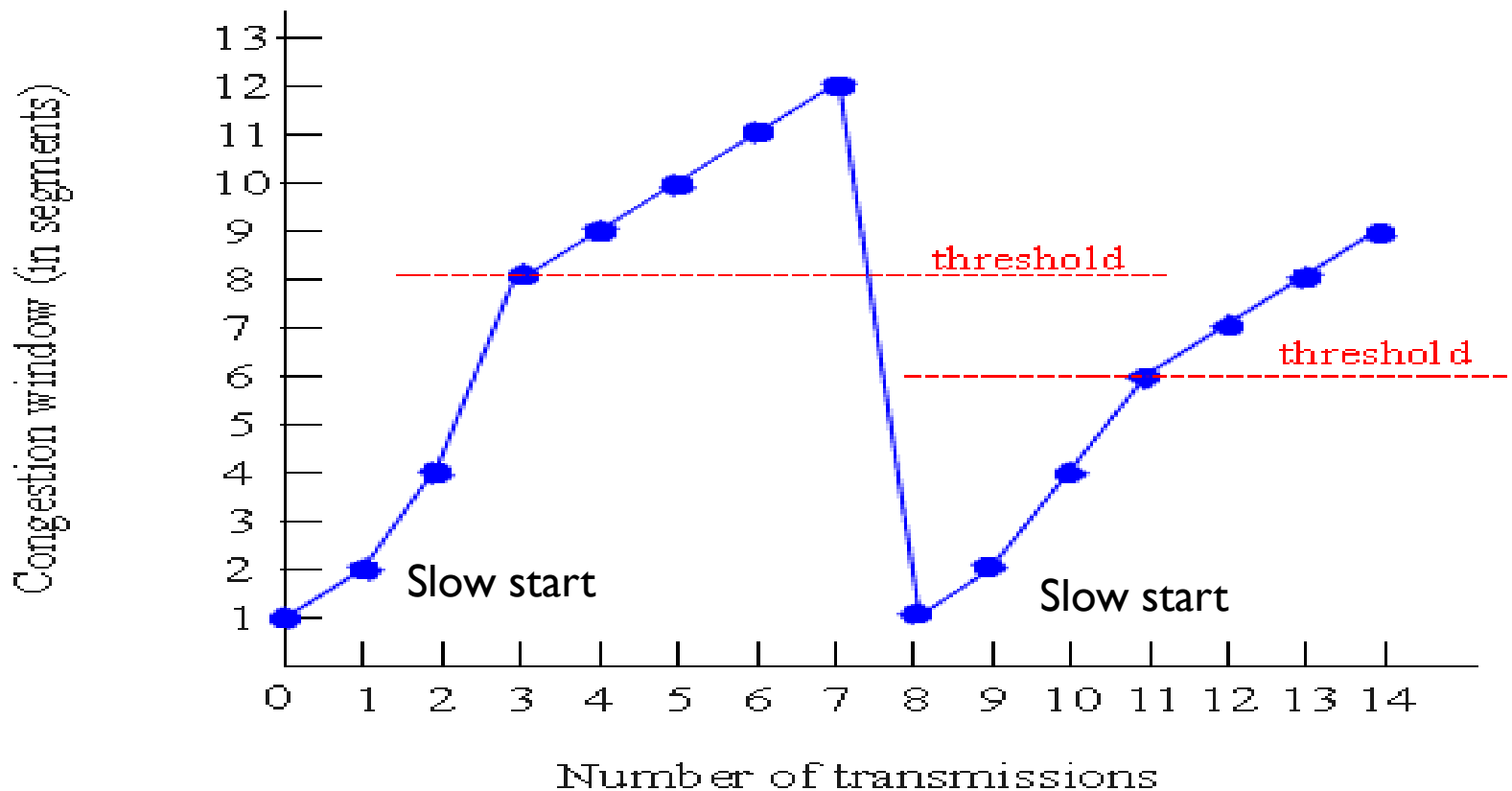
TCP Tahoe

Congestion Avoidance

/* slowstart is over   */

/* Congwin > threshold */

Until (loss event) {

  every w segments ACKed:

    Congwin++

  }

threshold = Congwin/2

Congwin = 1

perform slowstart

TCP Reno

▸ Tahoe + fast recovery

Initially:

    cwnd = 1;
    ssthresh = 64K;

For each newly ACKed segment:

    if (cwnd < ssthresh)
        /* slow start*/
        cwnd = cwnd + 1;
    else
        /* congestion avoidance; cwnd increases (approx.)
            by 1 per RTT */
        cwnd += 1;

Triple-duplicate ACKs:

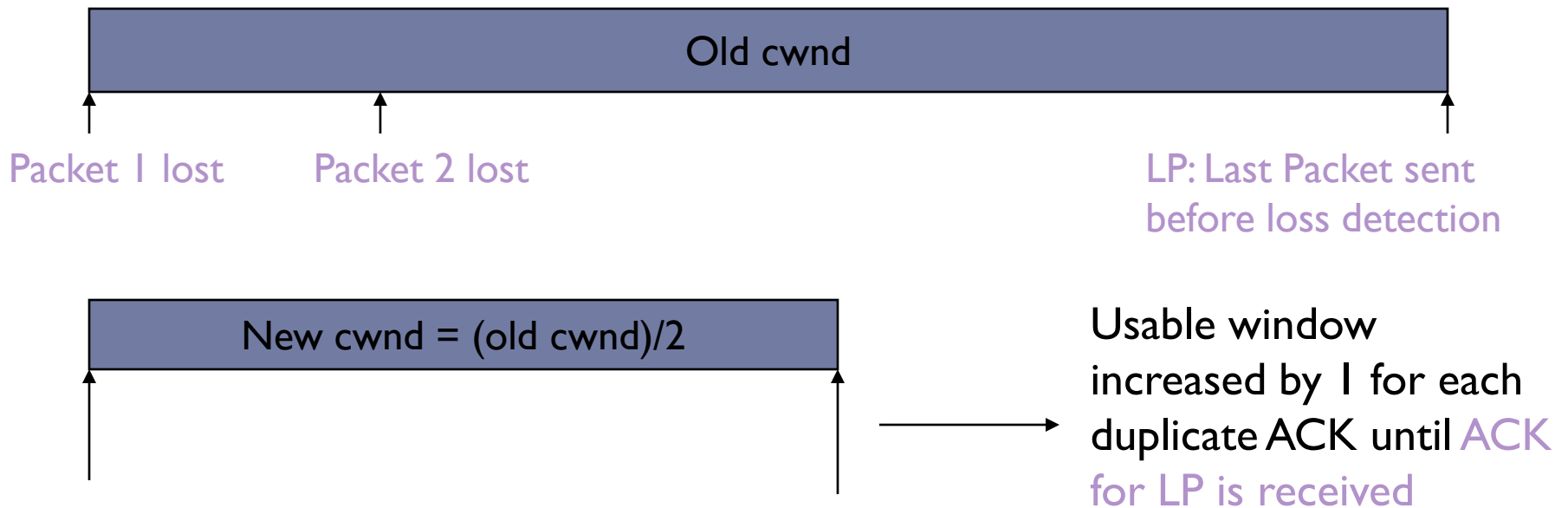   /* multiplicative decrease */
    cwnd = ssthresh = cwnd/2;
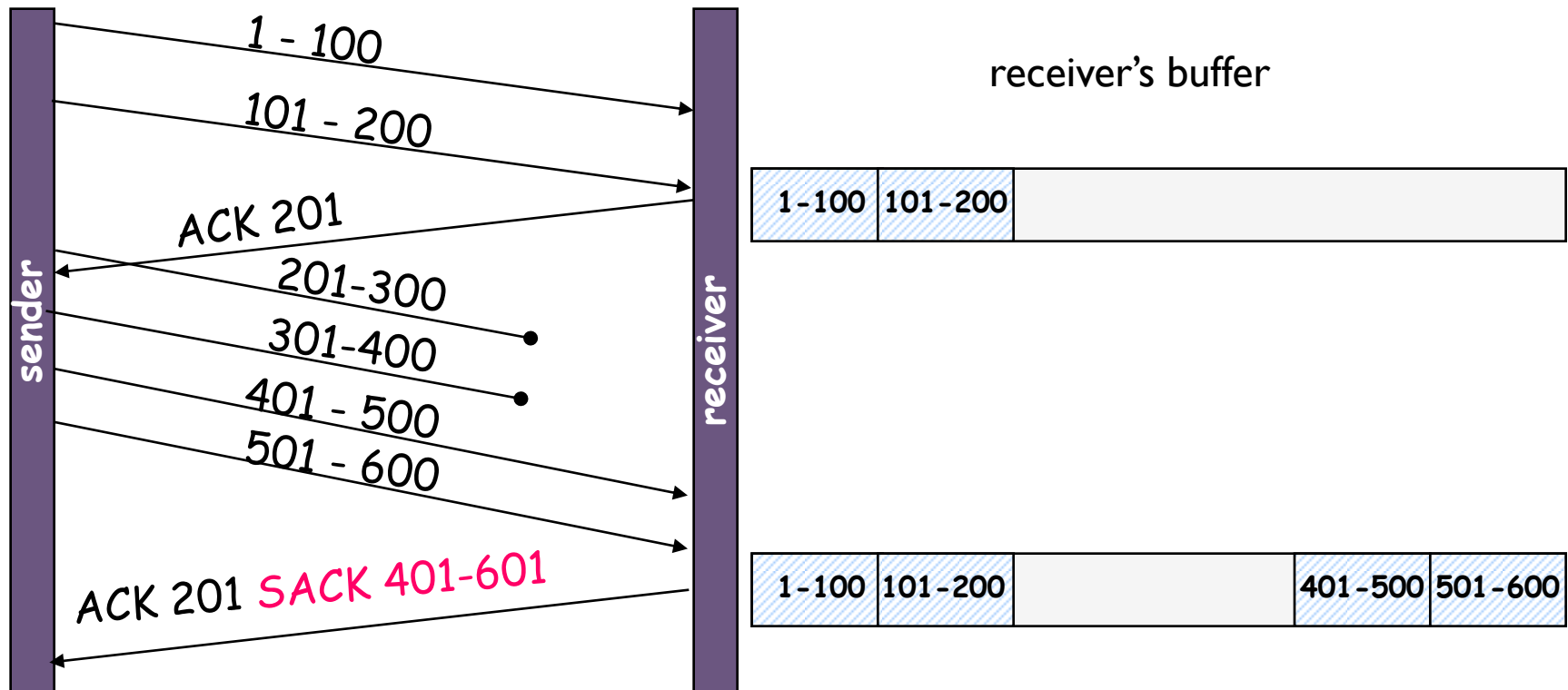
Timeout:

    ssthresh = cwnd/2;
    cwnd = 1;

Unit II   Beulah A.   30-Jan-17

TCP New-Reno

▸ New-Reno continues with fast recovery if a *partial ACK* is received

Old cwnd

↑ Packet 1 lost   ↑ Packet 2 lost   ↑ LP: Last Packet sent before loss detection

New cwnd = (old cwnd)/2

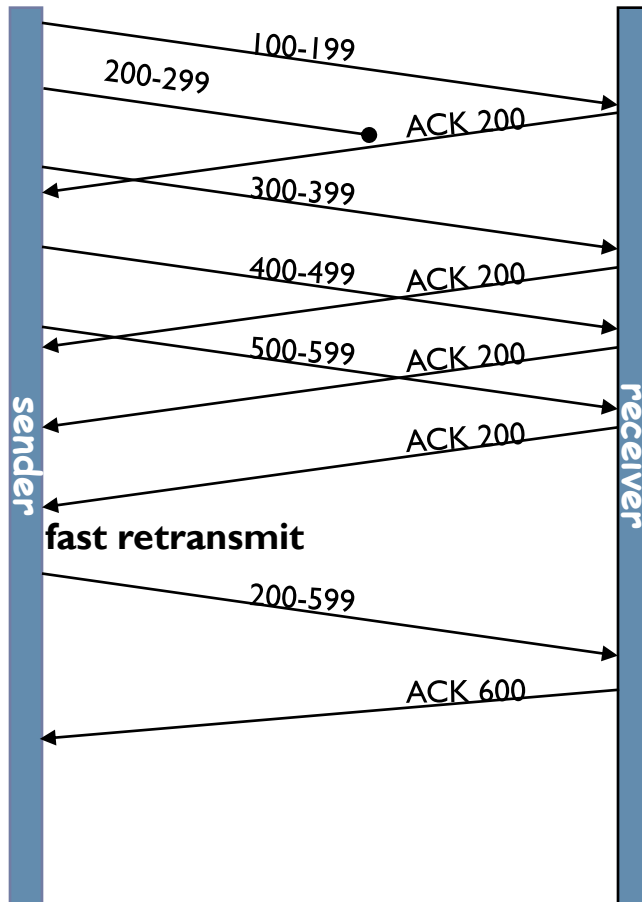→ Usable window increased by 1 for each duplicate ACK until ACK for LP is received
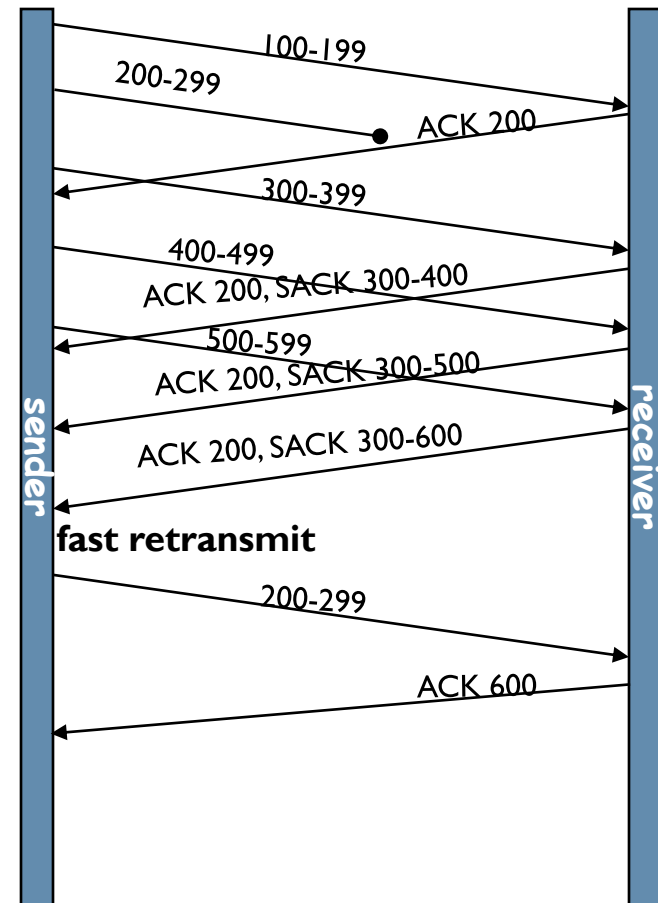
TCP SACK

▸ Reno + selective ACKs

TCP SACK

TCP without SACK

TCP with SACK

TCP Vegas

▶ There are three ways proposed in Vegas to increase delivery throughput and decrease packet loss.

  ▶ Modified Slow-Start Mechanism

    ▶ Cwnd is allowed exponential growth only every other RTT. (doubles the size of cwnd every 2 RTT time while there are no losses).

  ▶ New Congestion Avoidance Mechanism

    ▶ Control the size of cwnd by observing the variation of RTT

    ▶ Congestion in network (high RTT)

  ▶ New Retransmission Mechanism

    ▶ Sender does retransmission after a dupACK received, if RTT estimate > timeout (without waiting for 3 dupACKs)

Summary

- Transmission Control Protocol(TCP)
  - Basic concepts of TCP
  - TCP header
- Adaptation of TCP window
  - Silly window syndrome
  - Adaptive retransmision
- Improvements in TCP (Wired TCP)
  - TCP Tahoe
  - TCP Reno
  - TCP New-Reno
  - TCP SACK
  - TCP Vegas

Test your understanding

▶ Why do congestion occur in a network?

▶ How does slow start help improve the performance of TCP?

References

▸ Prasant Kumar Pattnaik, Rajib Mall, "Fundamentals of Mobile Computing", PHI Learning Pvt. Ltd, New Delhi – 2012.

▸ Jochen H. Schller, "Mobile Communications", Second Edition, Pearson Education, New Delhi, 2007.

▸ Behrouz A. Forouzan, "Data communication and Networking", Fourth Edition,Tata McGraw – Hill, 2011.