Internet Protocols EBU5403
Live Lecture B1/B2/B3

Module organiser: Richard Clegg
(r.clegg@qmul.ac.uk)

Michael Chai (michael.chai@qmul.ac.uk)

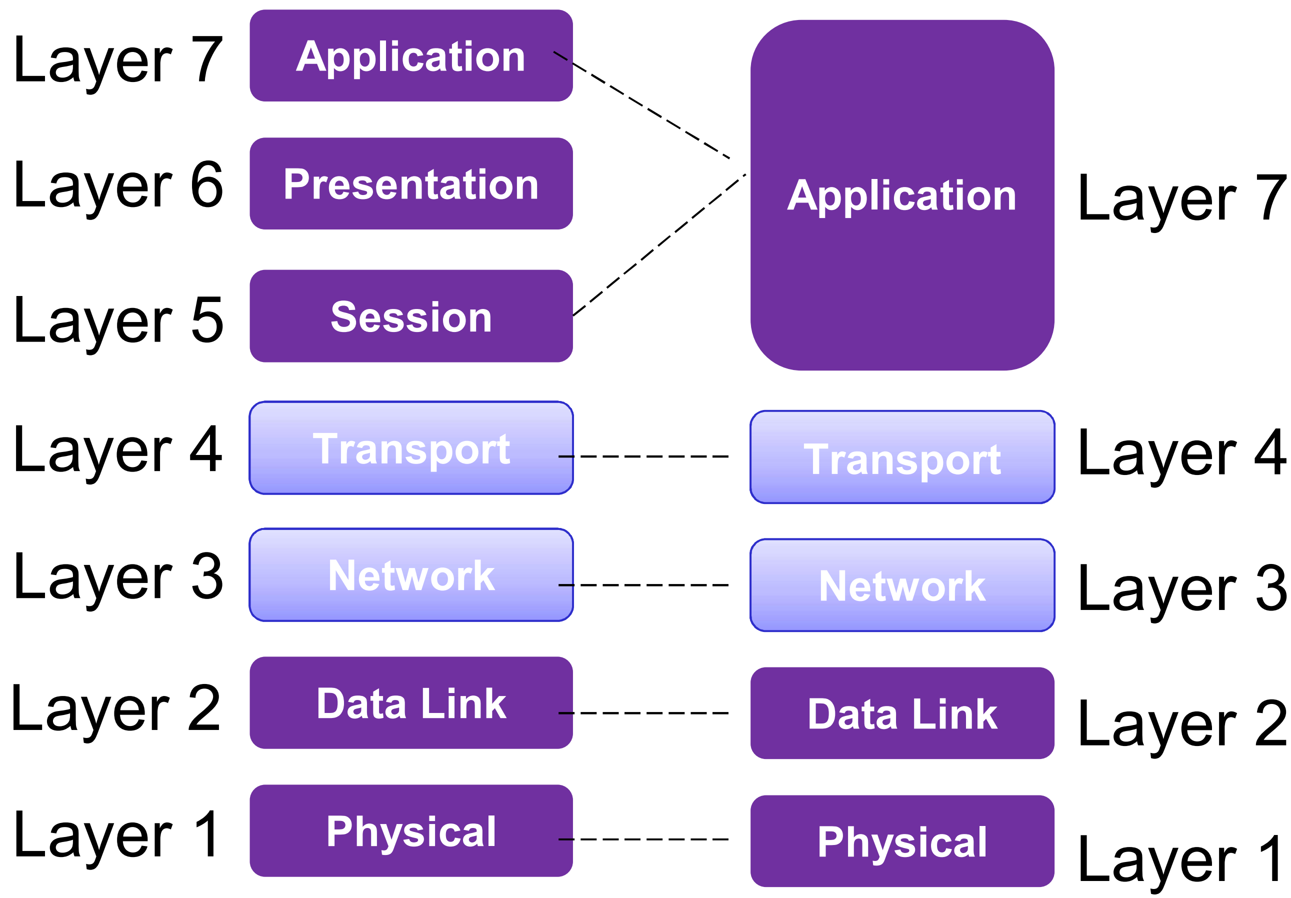
Cunhua Pan(c.pan@qmul.ac.uk)

	Part 1	Part 2	Part 3	Part 4
Ecommerce + Telecoms 1	Richard Clegg		Cunhua Pan	
Telecoms 2	Michael Chai			

Structure of course

- Part A
  - Introduction to IP Networks
  - The Transport layer (part I)
- Part B
  - The Transport layer (part II)
  - The Network layer (part I)
  - Class test
- Part C
  - The Network layer (part II)
  - The Data link layer (part I)
  - Router lab tutorial (assessed lab work after this week)
- Part D
  - The Data link layer (part II)
  - Network management and security
  - Class test

Layers in this life lecture

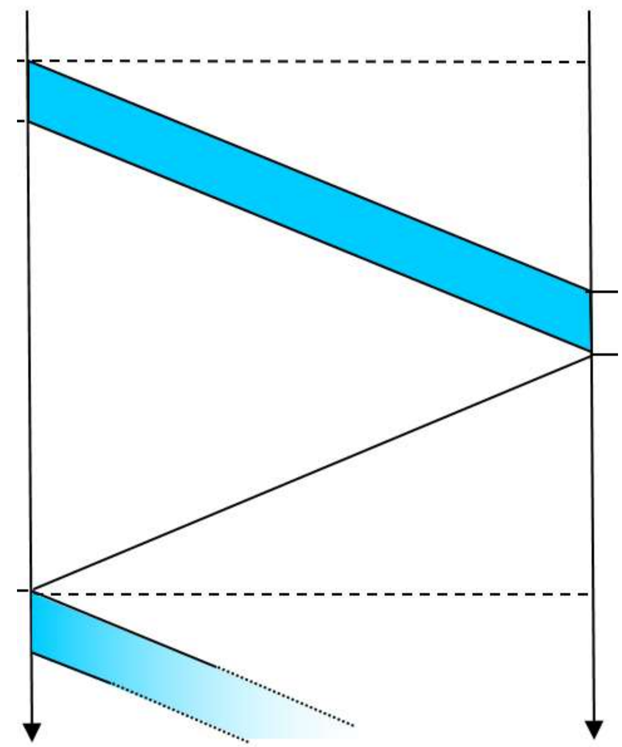


Reminder of lecture contents

- Lecture B1
  - Stop and wait vs Go-back-N and Selective Repeat
  - TCP RTT estimate (Jacobsen-Karel)
  - TCP fast retransmit
- Lecture B2
  - TCP flow control
  - TCP start up/shut down
  - TCP congestion control
- Lecture B3
  - Data plane and control plane overview
  - Forwarding + longest prefix match

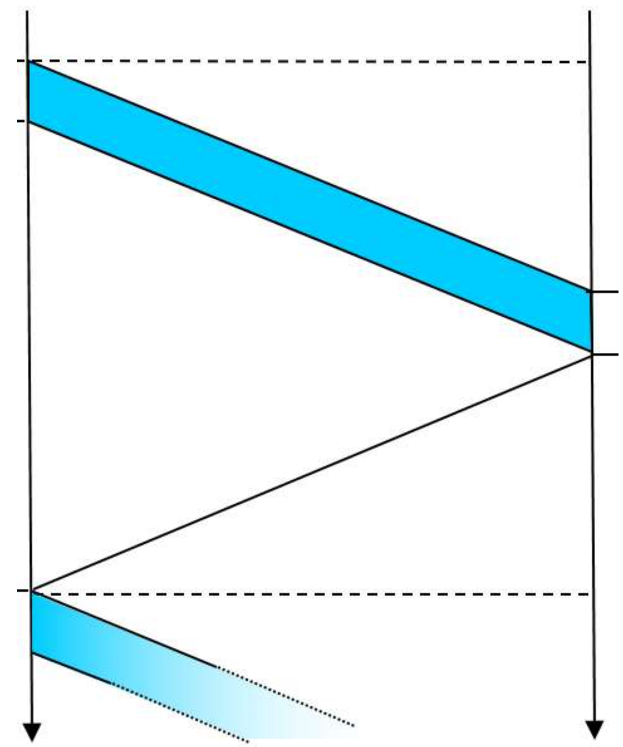
Link utilisation

- Define link utilisation:
  A. The time it takes between the sender finishing sending and the sender getting an ACK from the receiver.
  B. The time it takes to send a packet.
  C. The proportion of the time that the link is used for sending data.
  D. The proportion of the time that the link is empty.



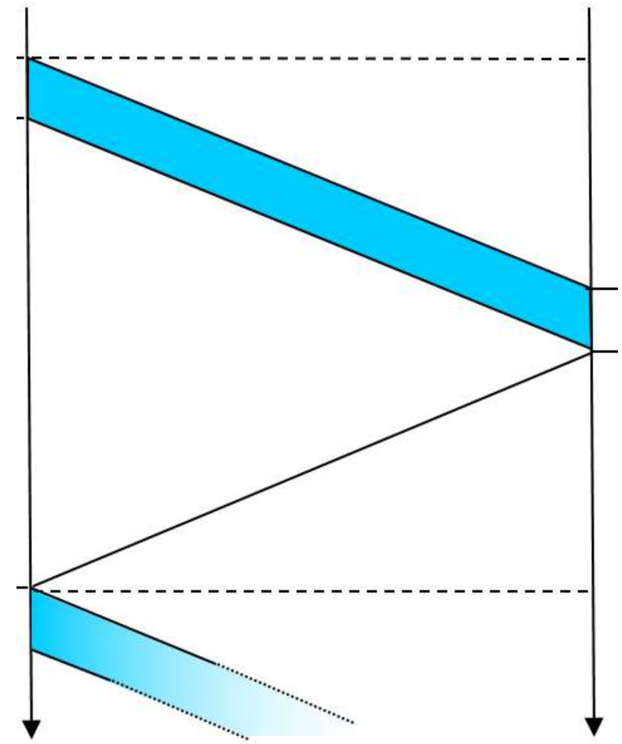
Link utilisation

- Define link utilisation:
  A. The time it takes between the sender finishing sending and the sender getting an ACK from the receiver. (This is RTT).
  B. The time it takes to send a packet. (This is transmission delay)
  C. The proportion of the time that the link is used for sending data.
  D. The proportion of the time that the link is empty. (This is 1 – utilisation).



Link utilisation

- Consider data transmission in rdt 3.0. Packets of length 1500 bits are sent over a link which is 100Mb/s. The delay is 1ms.
- What is the utilisation (approx):
  A. 0.00744
  B. 0.00515
  C. 0.00015
  D. 1.5



Link utilisation

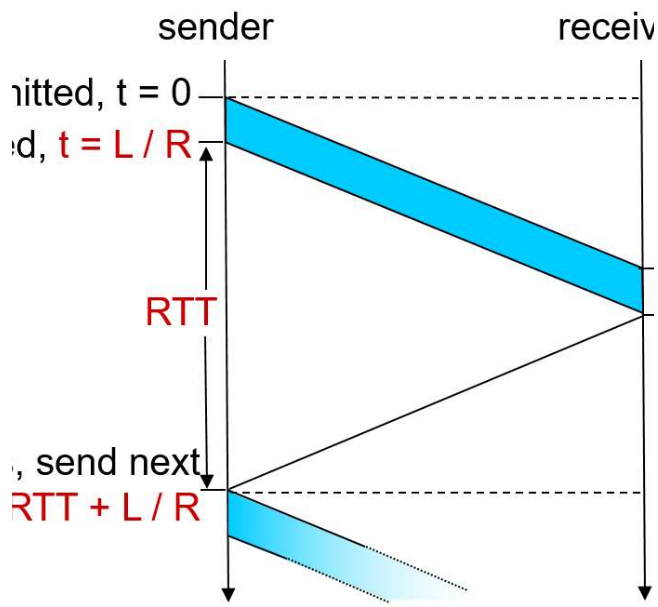
- Consider data transmission in rdt 3.0. Packets of length 1500 bits are sent over a link which is 100Mb/s. The delay is 1ms.
- What is the utilisation (approx):
  A. 0.00744
  B. 0.00515
  C. 0.00015
  D. 1.5

  L=1500bits R=100,000,000 bits/sec

  L/R= 0.000015 seconds

  RTT= 0.002 seconds

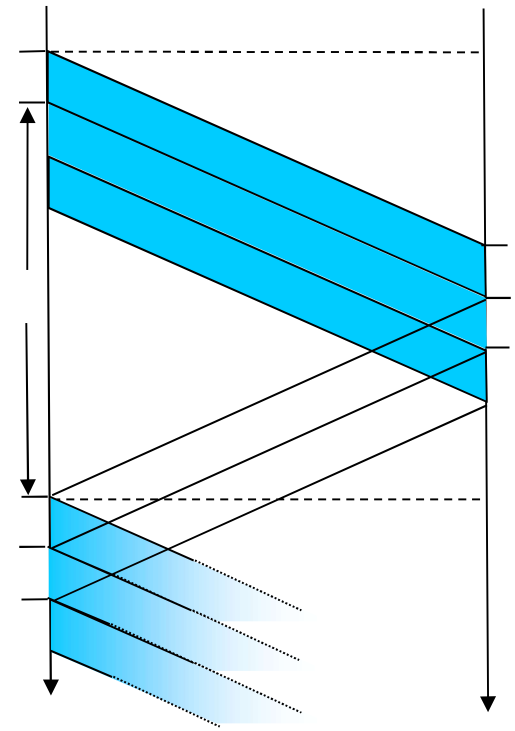
  U = (L/R) / (L/R + RTT)= 0.000015/0.002015 = 0.00744..



Pipelining to increase utilisation

Instead of sending packets one after another packets are sent in batches of N. The link is not full and no losses are experienced. What is effect on utilisation

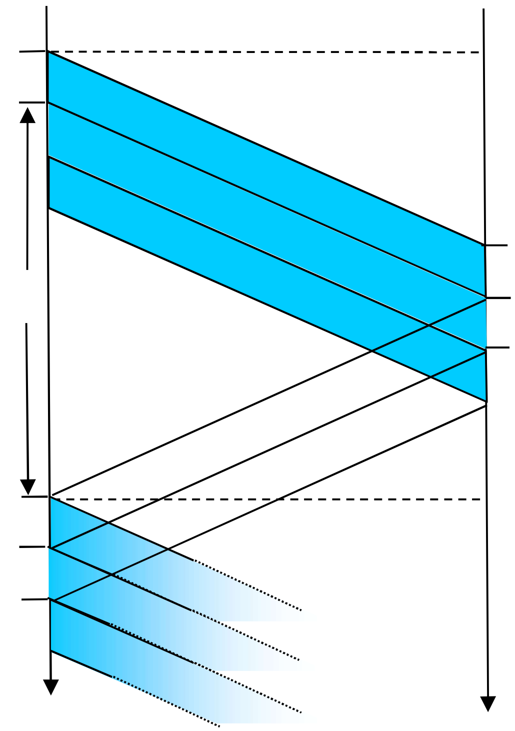
A. Increases by a factor of N
B. Stays the same
C. Becomes three times as much
D. Decreases to 1/N



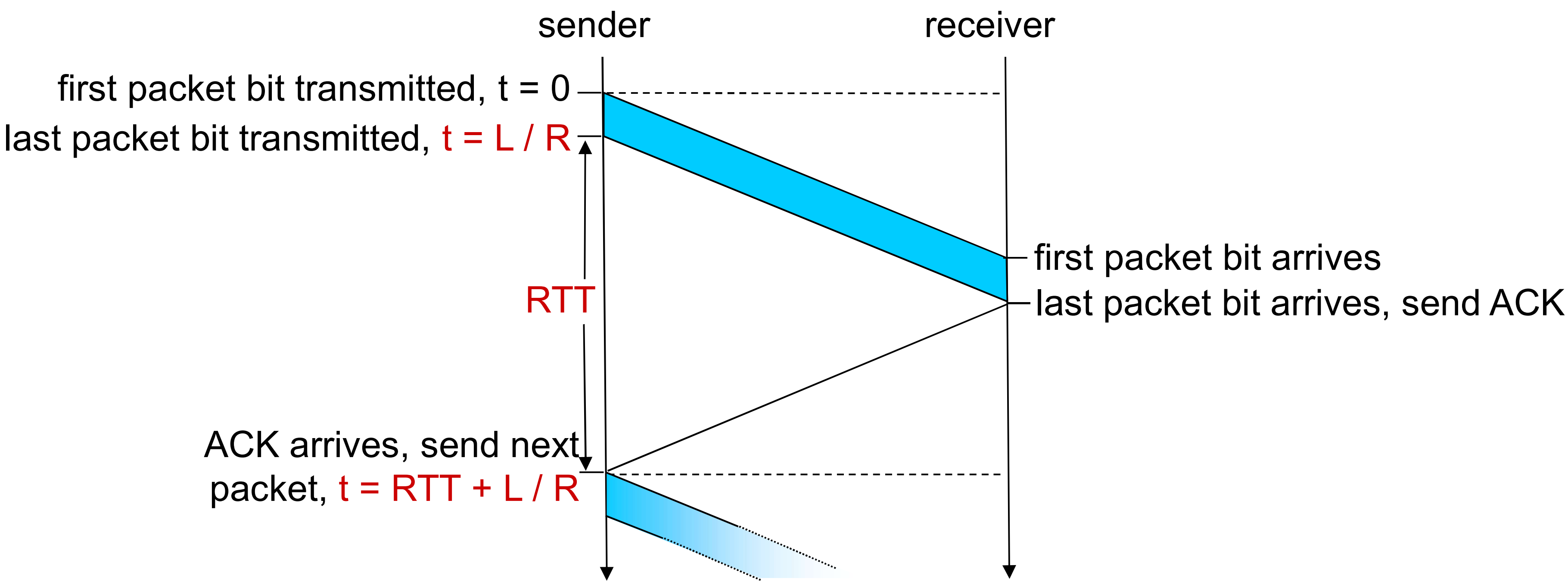
Pipelining to increase utilisation

Instead of sending packets one after another packets are sent in batches of N. The link is not full and no losses are experienced. What is effect on utilisation

A. **Increases by a factor of N**
B. Stays the same
C. Becomes three times as much
D. Decreases to 1/N

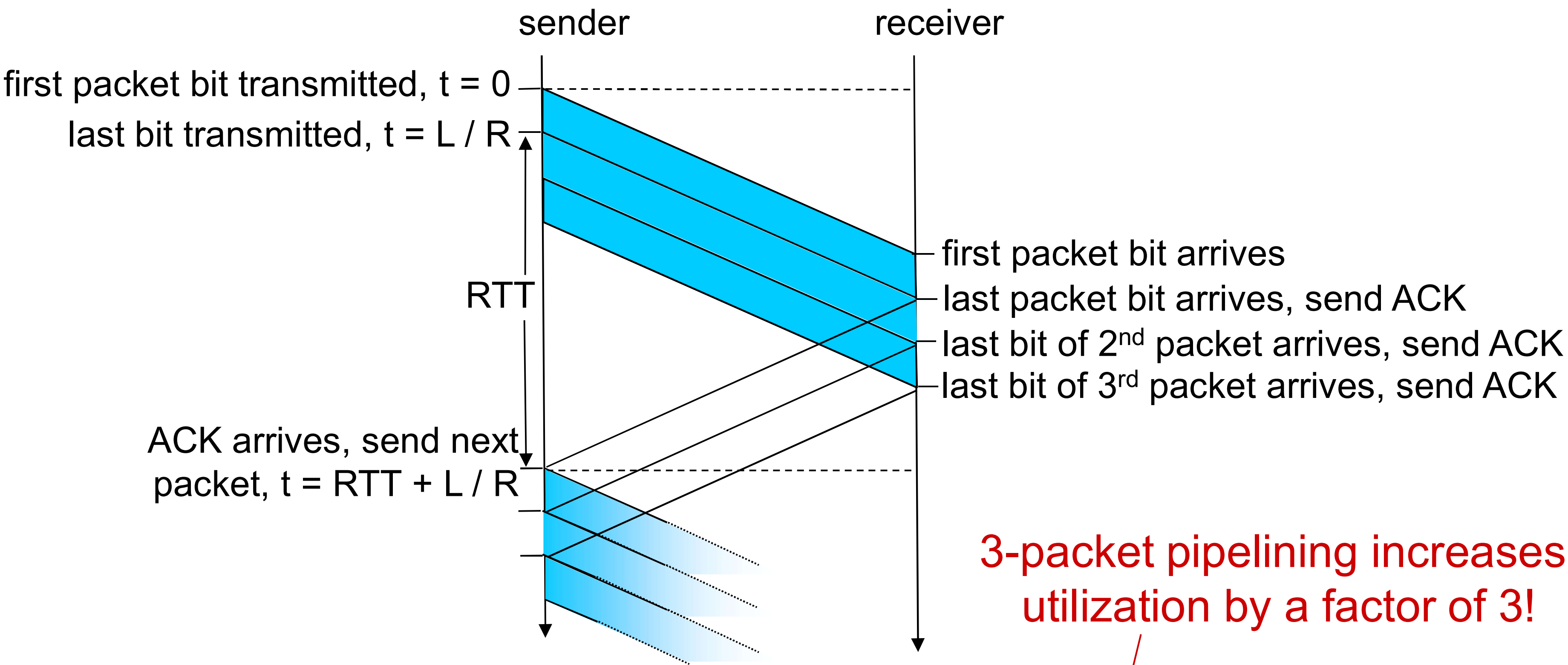


rdt3.0: stop-and-wait operation



$$U_{sender} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

Pipelining: increased utilization



3-packet pipelining increases utilization by a factor of 3!

$$U_{sender} = \frac{3L / R}{RTT + L / R} = \frac{.0024}{30.008} = 0.00081$$

Pipelined protocols: overview

Go-back-N:

- sender can have up to N unacked packets in pipeline
- receiver only sends *cumulative ACK*
  - doesn't ACK packet if there's a gap
- sender has timer for oldest unacked packet
  - when timer expires, retransmit *all* unacked packets

Selective Repeat:

- sender can have up to N packets which it has not seen ACK for in "pipeline"
- Receiver sends *individual ACK* for each packet
- sender maintains timer for each packet with no ACK
  - when timer expires, retransmit only that unacked packet

Go back N

- Which of these statements is true for go-back-N receiver if it gets packets 1,2,3,5
  A) Should send ACKs 1,2,3,5
  B) Should send packets 1,2,3,5 to the application
  C) Should send no packets to application because of missing data
  D) Should discard packet 5.

Go back N

- Which of these statements is true for go-back-N receiver if it gets packets 1,2,3,5

  A) Should send ACKs 1,2,3,5

     No ACK if there is a gap

  B) Should send packets 1,2,3,5 to the application

     Packet 5 would be out of order so don't send.

  C) Should send no packets to application because of missing data

     Apart from anything else it does not know 4 is missing when it receives 1,2 and 3.

  D) Should discard packet 5.

Selective repeat

- Which is true for selective repeat receiver if it gets packets 1,2,3,5
  A) Should send ACKs 1,2,3,5
  B) Should send packets 1,2,3,5 to the application
  C) Should send no packets to application because of missing data
  D) Should discard packet 5.

Selective repeat

- Which of these statements is true for go-back-N receiver if it gets packets 1,2,3,5

  A) Should send ACKs 1,2,3,5

  Yes – it acknowledges every packet

  B) Should send packets 1,2,3,5 to the application

  Packet 5 would be out of order so don't send.

  C) Should send no packets to application because of missing data

  Apart from anything else it does not know 4 is missing when it receives 1,2 and 3.

  D) Should discard packet 5.

  Keeps it in buffer and waits for 4.

Advantages each scheme

- Go Back N
  - Simple to implement
  - Only a single timer
  - One ACK can acknowledge many packets
- Selective repeat
  - Minimises resending of packets (better use of network)
  - Individual ACKs give more information to sender

Jacobsen-Karel Algorithm

```
EstimatedRTT = (1- α)*EstimatedRTT + α*SampleRTT
```

- $\alpha = 0.125$ (1/8)
- The previous estimate of RTT was 80ms.
- A new sample arrives with RTT = 160ms.
- What is the new estimated RTT?

Jacobsen-Karel Algorithm

$\texttt{EstimatedRTT} = (1-\alpha)*\texttt{EstimatedRTT} + \alpha*\texttt{SampleRTT}$

- $\alpha = 0.125$ (1/8)
- The previous estimate of RTT was 80ms.
- A new sample arrives with RTT = 160ms.
- What is the new estimated RTT?

- **$(1-\alpha) = 7/8$**
- **$\texttt{EstimatedRTT} = 7/8*80 + 1/8 * 160\texttt{ms}$**
- **$\texttt{EstimatedRTT} = 90\texttt{ms}$**

TCP flow control

Which statements about flow control are true:

- A) The receive window (rwnd) is sent over the network to the sender.
- B) The receive window measures how much congestion is on the network
- C) Flow control reduces network congestion
- D) Sender sees rwnd=5000bytes. It has 4 UNACK'd packets of 1500bytes. It can send 1 more.

TCP flow control

Which statements about flow control are true:

- A) The receive window (rwnd) is sent over the network to the sender.

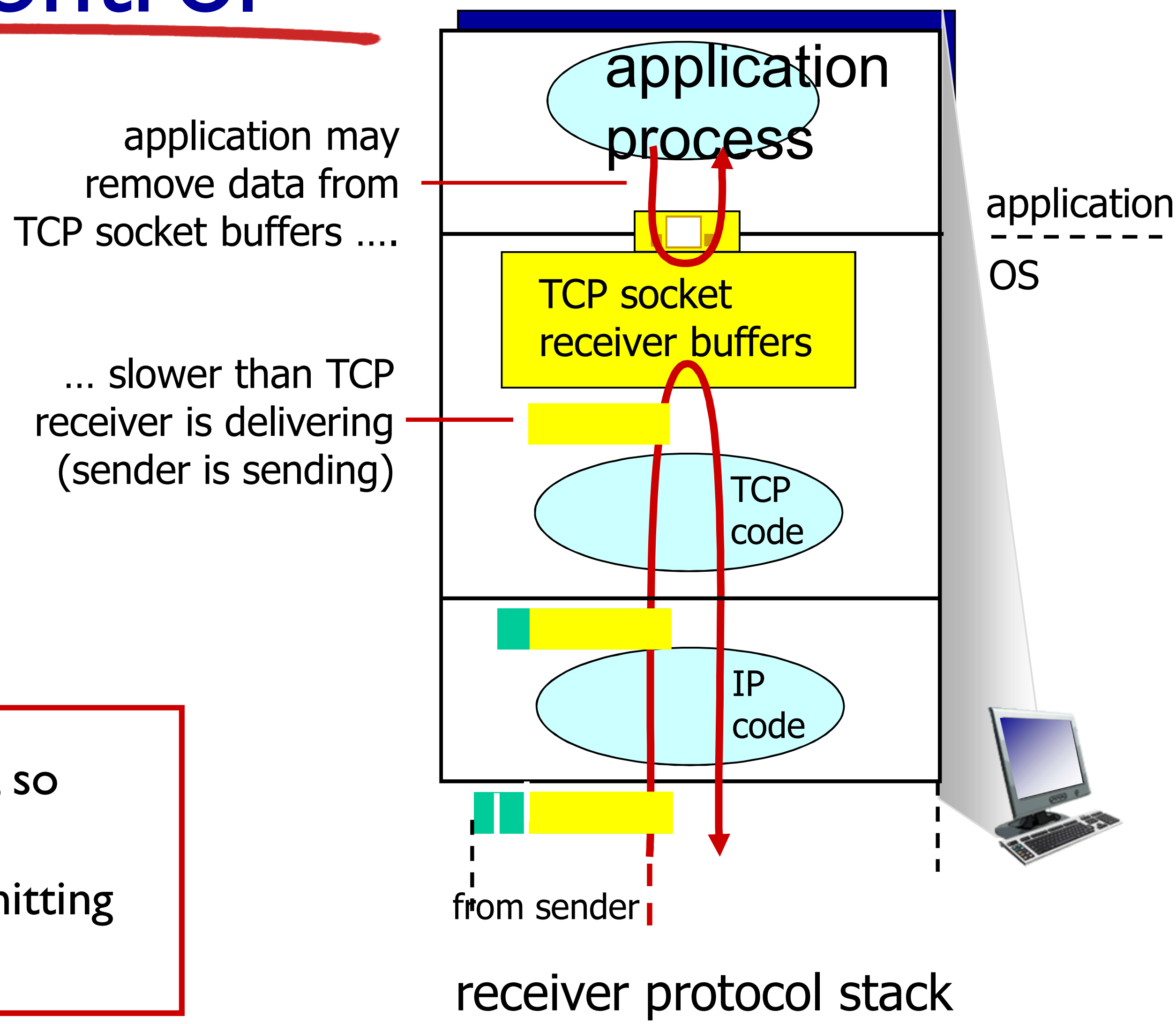
  Yes – it is in the TCP header.
- B) The receive window measures how much congestion is on the network

  No, it measures the receive buffer
- C) Flow control reduces network congestion

  Not deliberately, cannot see network congestion
- D) Sender sees rwnd=5000bytes. It has 4 UNACK'd packets of 1500bytes. It can send 1 more.

  No – 6000 bytes already send

TCP flow control

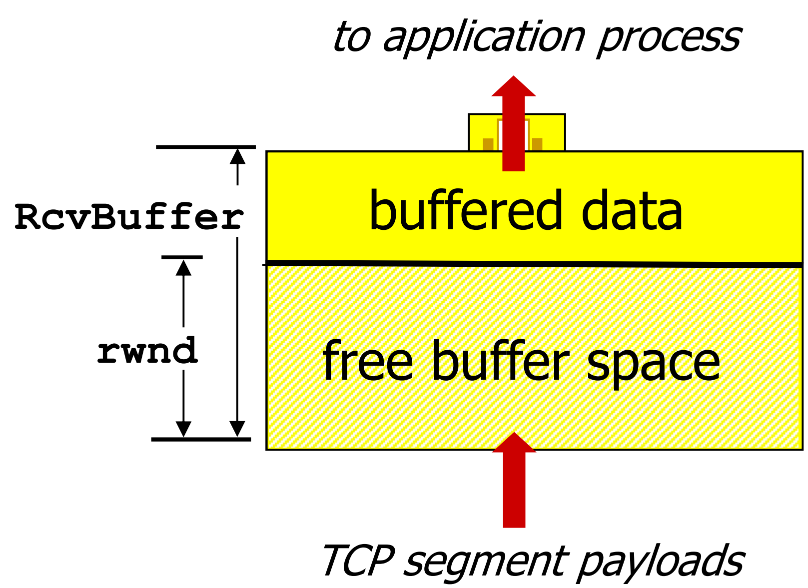


flow control

receiver controls sender, so sender will not overflow receiver buffer by transmitting too much, too fast

TCP flow control

- receiver "advertises" free buffer space by including **`rwnd`** value in TCP header of receiver-to-sender segments
  - **`RcvBuffer`** size set via socket options (typical default is 4096 bytes)
  - many operating systems autoadjust **`RcvBuffer`**
- sender limits amount of unacked ("in-flight") data to receiver's **`rwnd`** value
- guarantees receive buffer will not overflow



receiver-side buffering

`rwnd` = receive window

TCP handshaking

- What is the sequence for the TCP three way handshake to open a connection?

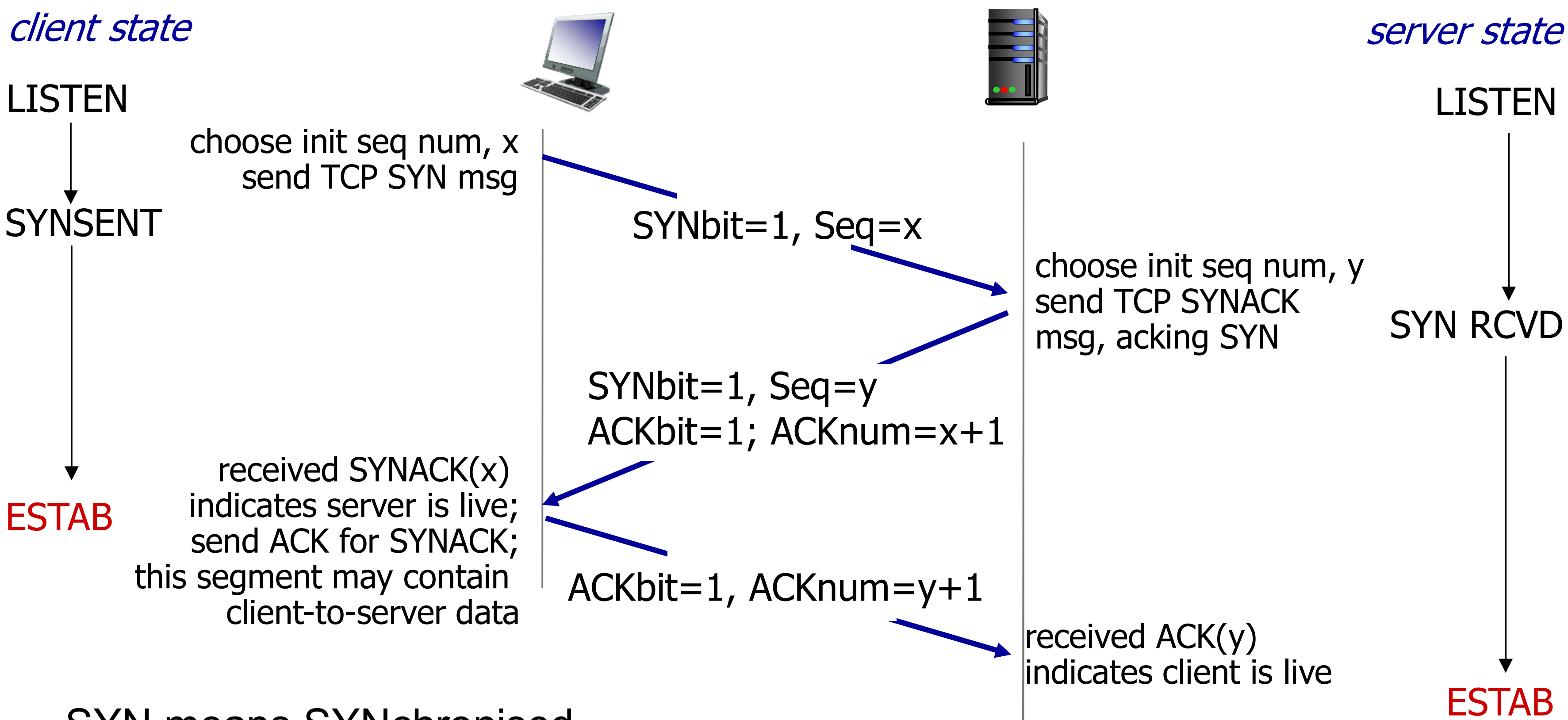
- To close a connection the sequence is FIN, FIN-ACK, ACK. Why might the FIN-ACK be split into two packets?

TCP handshaking

- What is the sequence for the TCP three way handshake to open a connection?
  - SYN SYN-ACK ACK
- To close a connection the sequence is FIN, FIN-ACK, ACK. Why might the FIN-ACK be split into two packets?

The closing computer may have data left to send.

TCP 3-way handshake

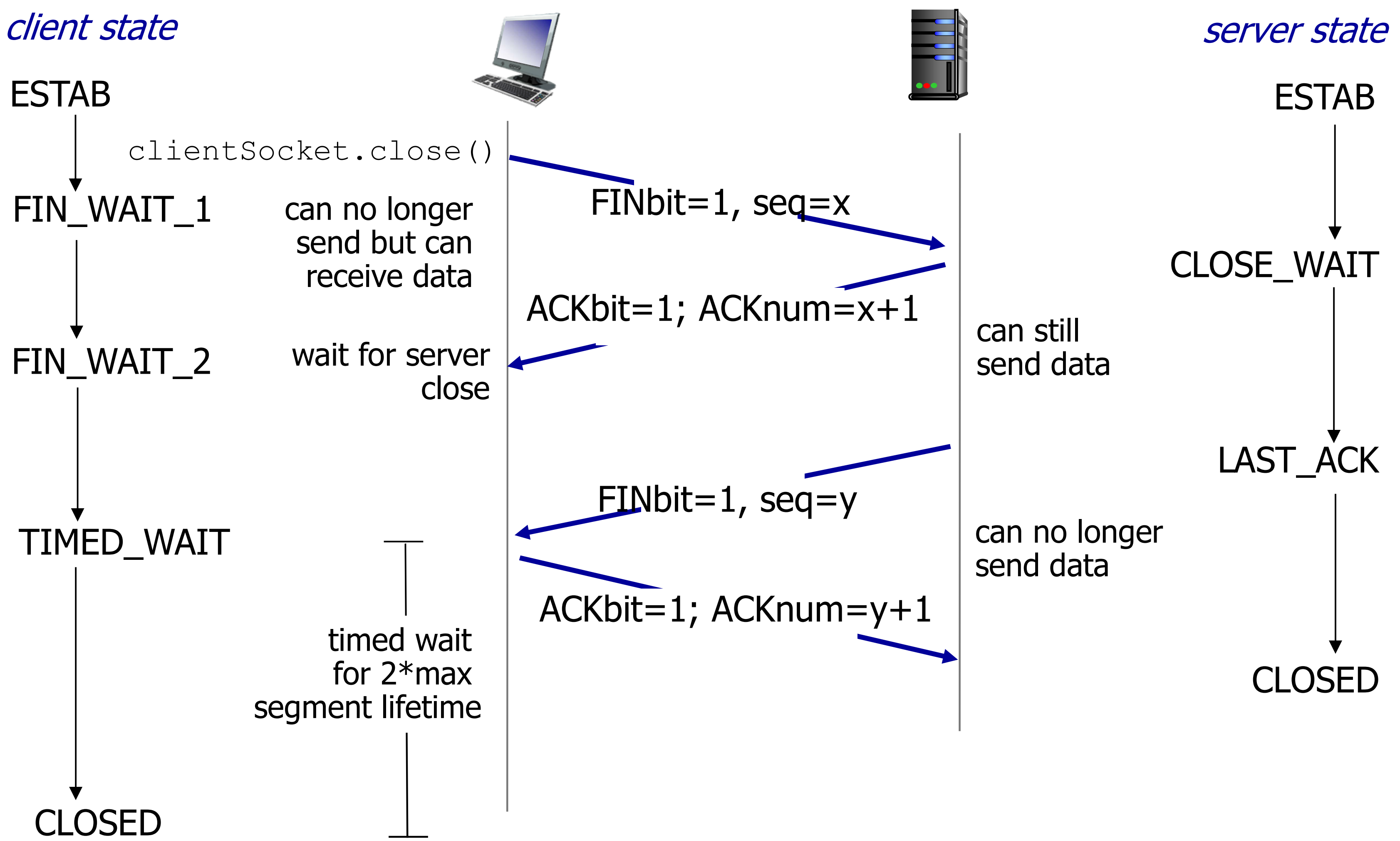


SYN means SYNchronised

SYNbit = bit in TCP header saying this is SYN packet

ACKbit = bit in TCP header saying this is ACK packet

TCP: closing a connection



TCP congestion control

Which statements about congestion control are true:

- A) The congestion window (cwnd) is sent over the network to the sender.
- B) AIMD means that cwnd increases only slowly when there is no loss

C) Congestion control has problems because of loss

- D) Congestion control reduces network congestion

TCP congestion control

Which statements about congestion control are true:

- A) The congestion window (cwnd) is sent over the network to the sender.

  No, it is calculated by the sender.
- B) AIMD means that cwnd increases only slowly when there is no loss

  Yes – it increases slowly decreases quickly.
- C) Congestion control has problems because of loss

  Loss is fundamental to most congestion control
- D) Congestion control reduces network congestion

  Yes – this is its very aim.

TCP mechanisms

Name these three TCP mechanisms

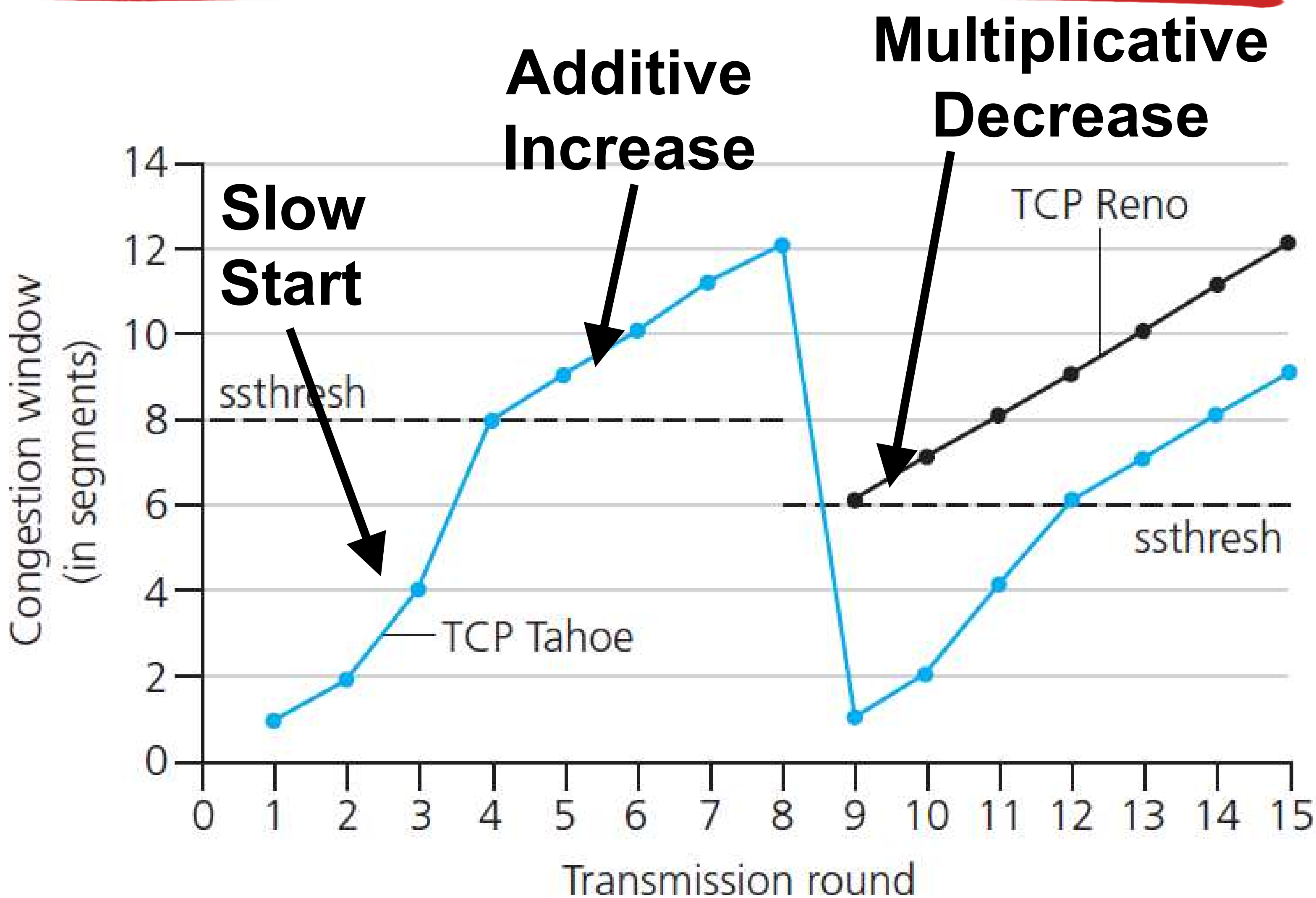
- When I see four Acknowledgement packets with the same number I retransmit a packet.
- When cwnd is small I double cwnd every RTT where no loss is experienced.
- When I think there is no congestion I increase the window size slowly, when I think there is congestion I decrease the window size rapidly

TCP mechanisms

Name these three TCP mechanisms

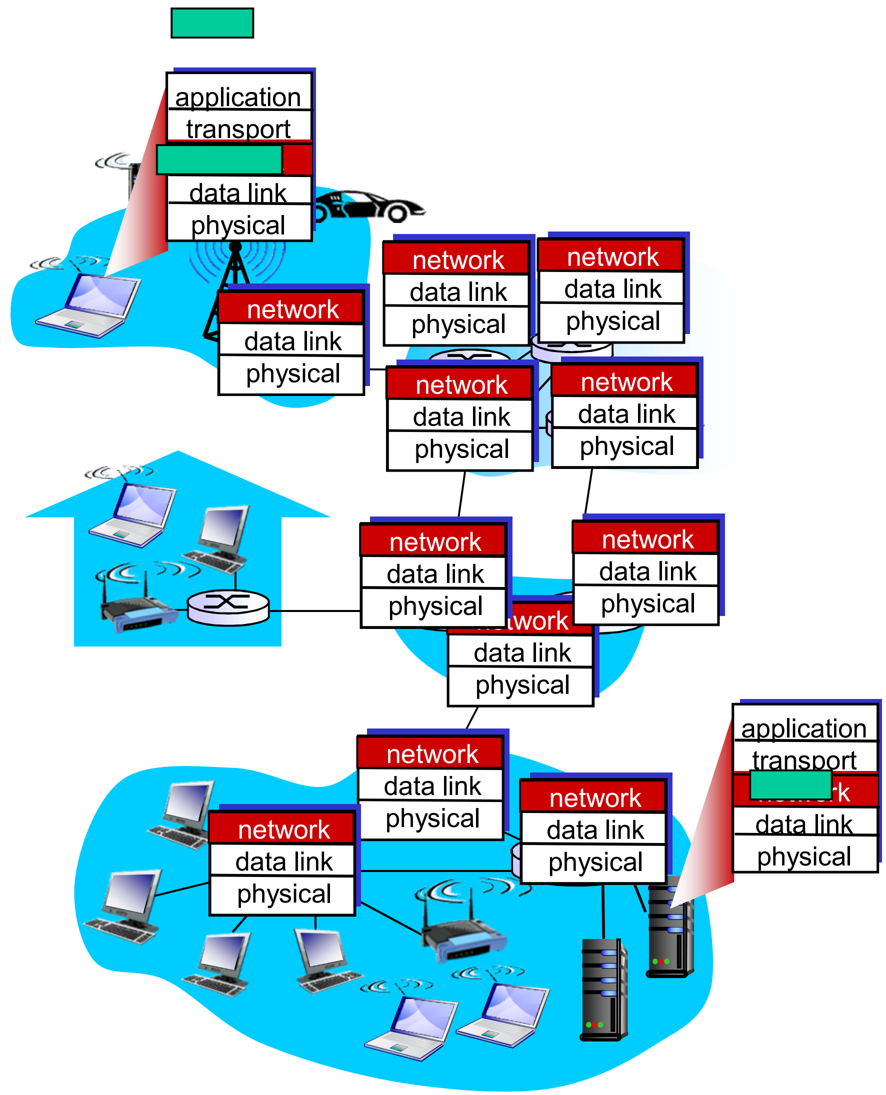
- When I see four Acknowledgement packets with the same number I retransmit a packet.
- Fast retransmit
- When cwnd is small I double cwnd every RTT where no loss is experienced.
- Slow start
- When I think there is no congestion I increase the window size slowly, when I think there is congestion I decrease the window size rapidly
- AIMD (Additive Increase Multiplicative Decrease)

TCP: switching from slow start to Congestion Avoidance



Network layer

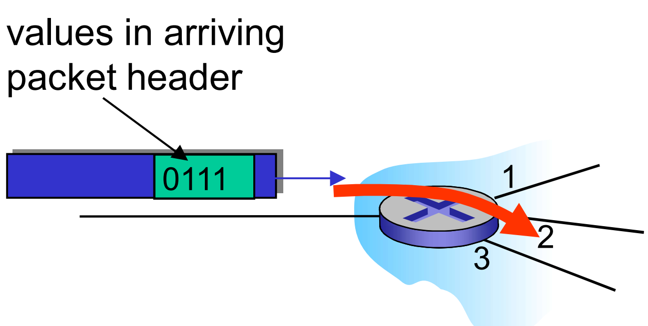
- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on receiving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- router examines header fields in all IP datagrams passing through it



Network layer: data plane, control plane

Data plane

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port
- forwarding function



Control plane

- network-wide logic
- determines how datagram is routed among routers along end-end path from source host to destination host
- two control-plane approaches:
  - *traditional routing algorithms:* implemented in routers
  - *software-defined networking (SDN)*: implemented in (remote) servers

Data plane or control plane

- Finding the correct route for a packet

- Forwarding a packet using Longest Prefix Match

- An SDN controller

- A router

Data plane or control plane

- Finding the correct route for a packet
  - Control plane
- Forwarding a packet using Longest Prefix Match
  - Data plane – takes place locally using forwarding table
- An SDN controller
  - Control plane (it is in the name)
- A traditional router
  - Both (a router has forwarding and routing)

Longest prefix matching which interface do the packets go to

Destination Address Range	Link
11001000 00010111 01111*** *********	A
11001000 00010111 01111000 *********	B
11001000 00010111 00011*** *********	C
otherwise	D

Dest: 11001000 00010111 01111000 10101010

Dest: 11001000 00010111 01111100 10101010

Dest: 11001000 00010111 00111000 10101010

Longest prefix matching which interface do the packets go to

Destination Address Range	Link
11001000 00010111 01111*** **********	A
11001000 00010111 01111000 **********	B
11001000 00010111 00011*** **********	C
otherwise	D

```
Dest: 11001000  00010111  01111000  10101010
B
Dest: 11001000  00010111  01111100  10101010
A
Dest: 11001000  00010111  00111000  10101010
D
```

Longest prefix matching

- we'll see *why* longest prefix matching is used shortly, when we study addressing
- longest prefix matching: often performed using ternary content addressable memories (TCAMs) specialised very high speed memory
  - *content addressable:* present address to TCAM: retrieve address in one clock cycle, regardless of table size
  - Cisco Catalyst: can up ~1M routing table entries in TCAM

What have we learned

- Throughput and pipelined protocols
  - Stop and wait vs Go-back-N and Selective Repeat
- TCP mechanisms
  - TCP RTT estimate (Jacobsen-Karel)
  - TCP fast retransmit
  - TCP flow control
  - TCP start up/shut down
  - TCP congestion control
- Network layer
  - Data plane and control plane overview
  - Forwarding + longest prefix match