

# Introduction to Computer Networks

Lecture 6: Transport Layer

Dr. Amal Elnahas

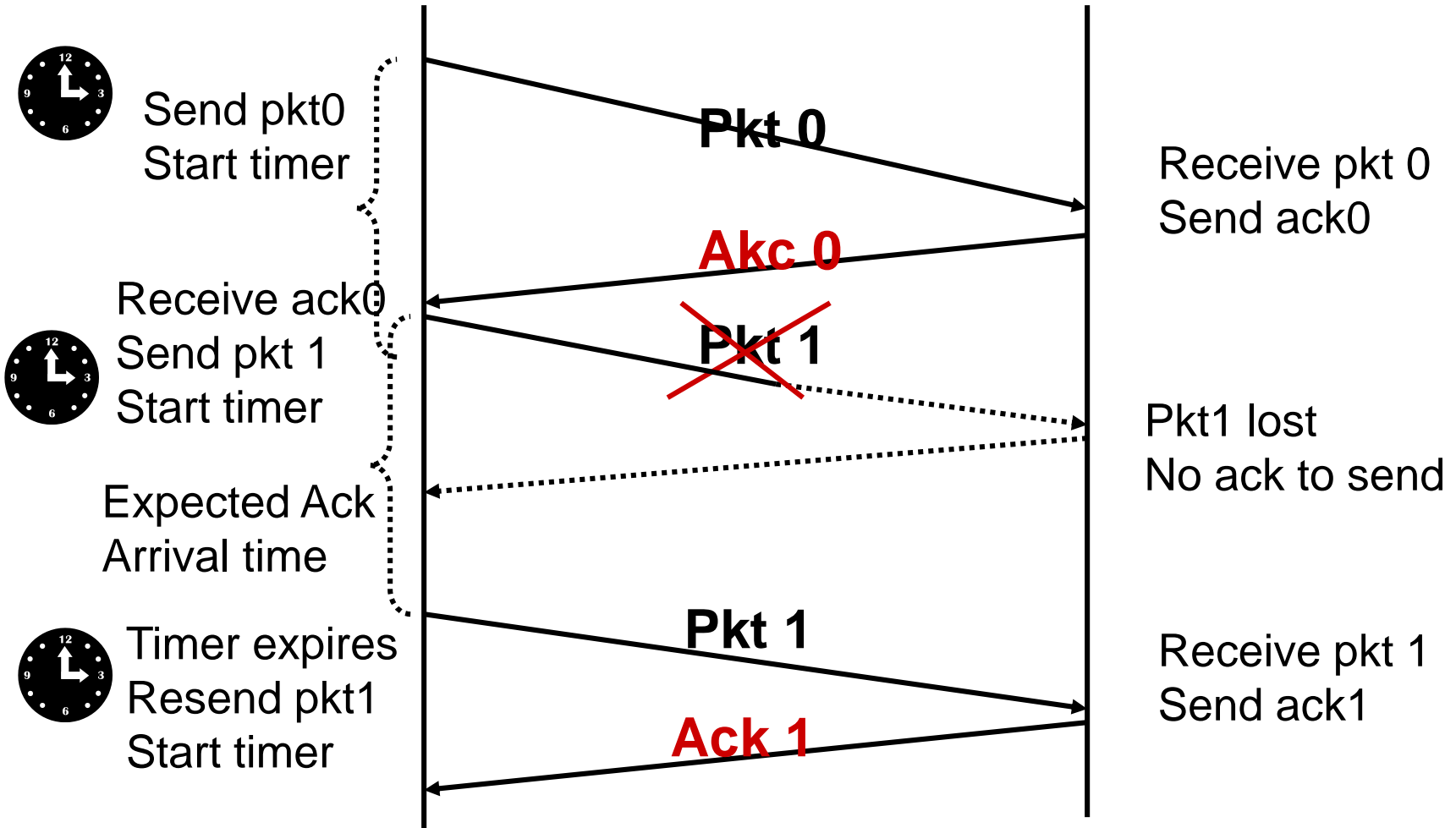
# TCP Reliability

- Reliability:
  - No duplication
  - No data loss
- Unreliable channel: (corrupted pkts)
  - Errors can occur; checksum to detect bit errors
- How to recover from errors:
  - **Positive acknowledgements (ACKs)**: receiver explicitly tells sender that pkt received OK
  - **Negative acknowledgements (NAKs)**: receiver explicitly tells sender that pkt had errors
  - sender retransmits pkt on receipt of NAK
  - Protocols based on this concepts are known as “Automatic Repeat reQuest (ARQ)” protocols

# Major Problems

- What if ACK or NACK are corrupted/lost?
  - Sender starts timer whenever transmitting
  - Receiver sends acknowledgement when data arrives (Positive Acknowledgment with Retransmission technique (PAR))
  - Sender retransmits if timer expires before acknowledgement arrives
- NACK not used, receiver ACKs last packet received OK so far
- Add a sequence number to each segment

# PAR: Packet Loss

**Sender****Network****Receiver**


# Pipelining

- Modifications:

- Allow sender to send multiple packets at once, but keep track of receiving the ack for each:

 associate each packet with a sequence number and timer

- Receiver must handle multiple packets (from multiple sources):

 Tell the sender a limit on the maximum number of pkts that can be sent without acknowledgment (Flow Control)

**Pipelined protocols (Sliding Window protocols)**

# Sliding Window Protocol (wind=2)

Send pkt1,  
Limit=1

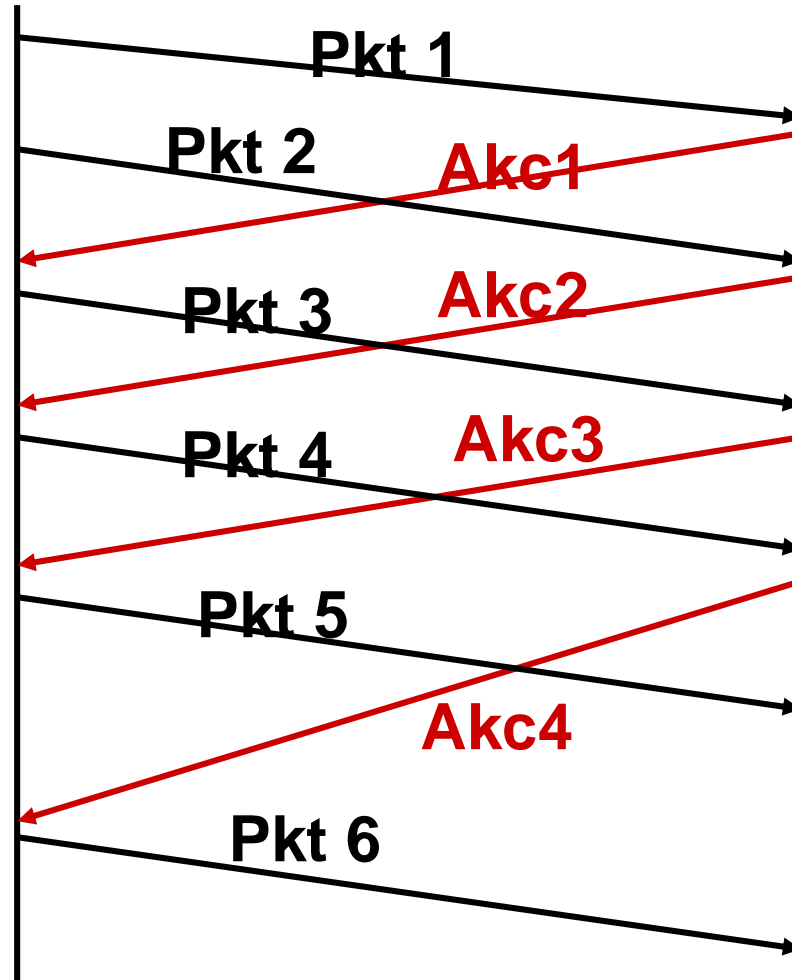
Send pkt2,  
**Stop** (limit=0)

Send pkt 3,  
Limit=0

Send pkt 4,  
Limit=0

Send pkt 5,  
Limit=0

Send pkt 6,  
Limit=0



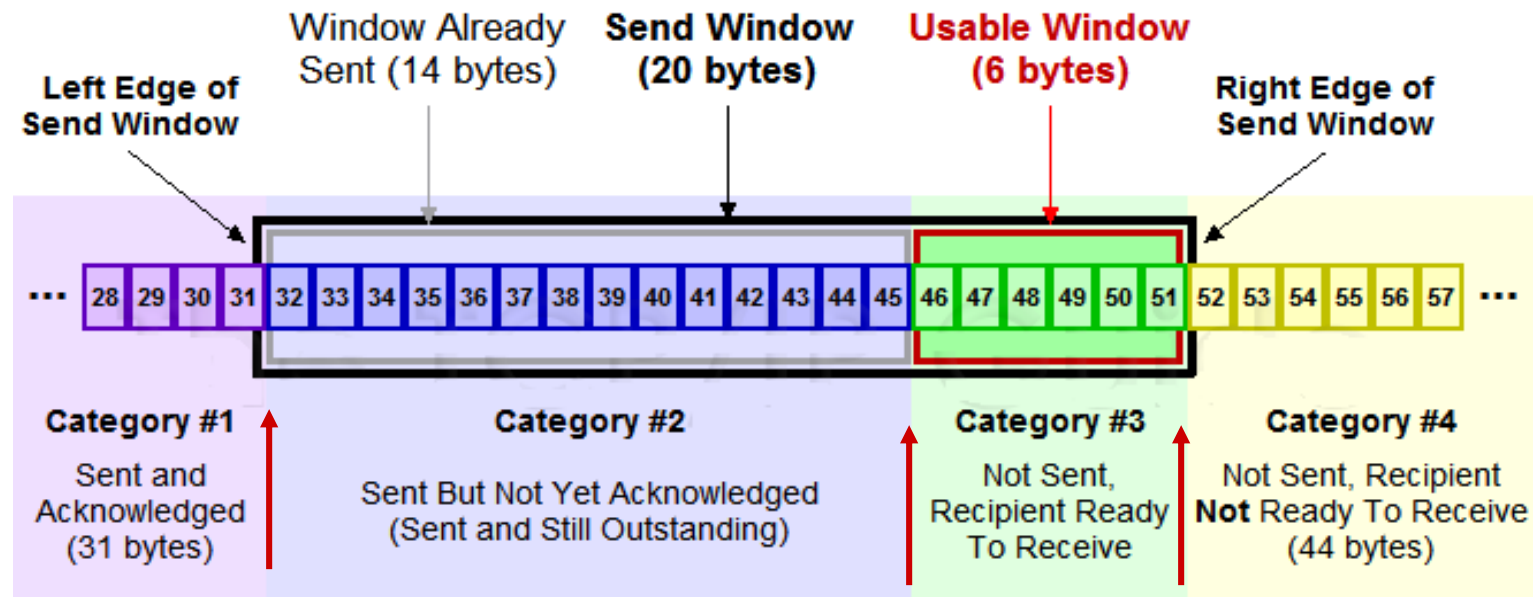
Send ack1

Send ack2

Send ack3,

# Sliding Window Protocol

- Receiver decides the maximum number of bytes sender can send without receiving acknowledgment, “**window size**”
- Sender maintains 3 pointers per connection to keep track of bytes sent, acknowledged, or not yet acknowledged
- Window moves forward as acknowledgment arrive



# Congestion Control

- Congestion:
  - Too many sources send too much data too fast for the network to handle
- Congestion control:
  - Reduce transmission rate to match the slowest (bottleneck) link on the path
- Congestion Indication:
  - Loss of packets (buffer overflow at routers)
  - High delay (queuing delay in routers buffers)
- Loss of packet Indication:
  - Receiving duplicate ACK for previous pkt
  - Timer expires



# Sliding Window: Congestion Control

- Sender maintains a congestion window (cwnd), in addition to the receiver's window (rwnd) advertised in ACK

$$\text{Allowed-window} = \min(\text{cwnd}, \text{rwnd})$$

- If no congestion: Allowed-window = rwnd
- Packet loss is interpreted as congestion occurrence: reduce congestion window size.

# Congestion Window Maintenance

- TCP connections probe for available bandwidth
- Increase the congestion window until loss occurs
- When loss is detected decrease window, then begin probing (increasing) again
- The congestion window grows in two phases:
  - Slow start — Ramp up transmission rate until loss occurs
  - Congestion avoidance — Keep connection close to sustainable bandwidth
- A window size threshold (bytes transmitted) distinguishes between slow start and congestion avoidance phases

# Phase I: Slow-Start

Begin

    cwnd = 1 // (one segment)

    repeat

        if receive ACK

            cwnd++

    until (loss or cwnd > threshold)

    if (cwnd > threshold)

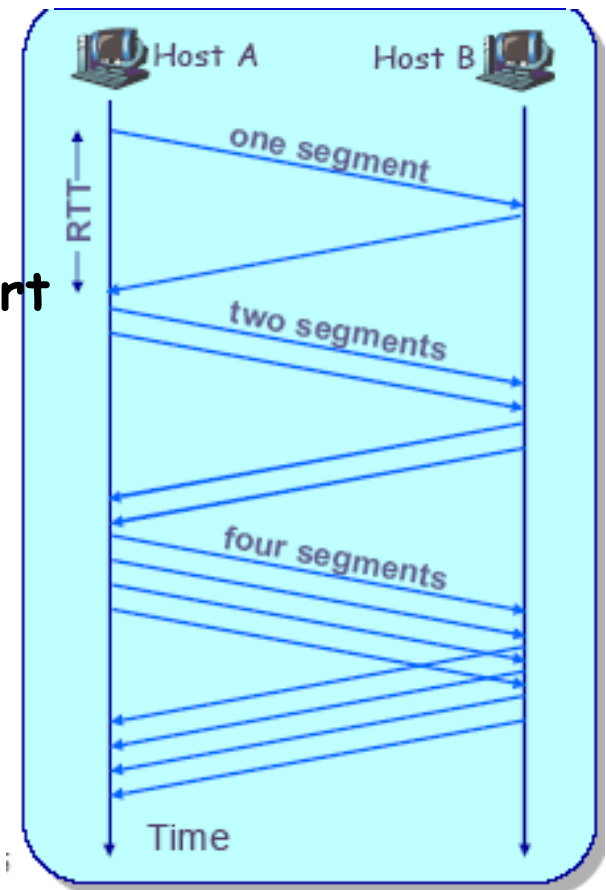
        repeat

            cwnd++ when cwnd segments ACKed

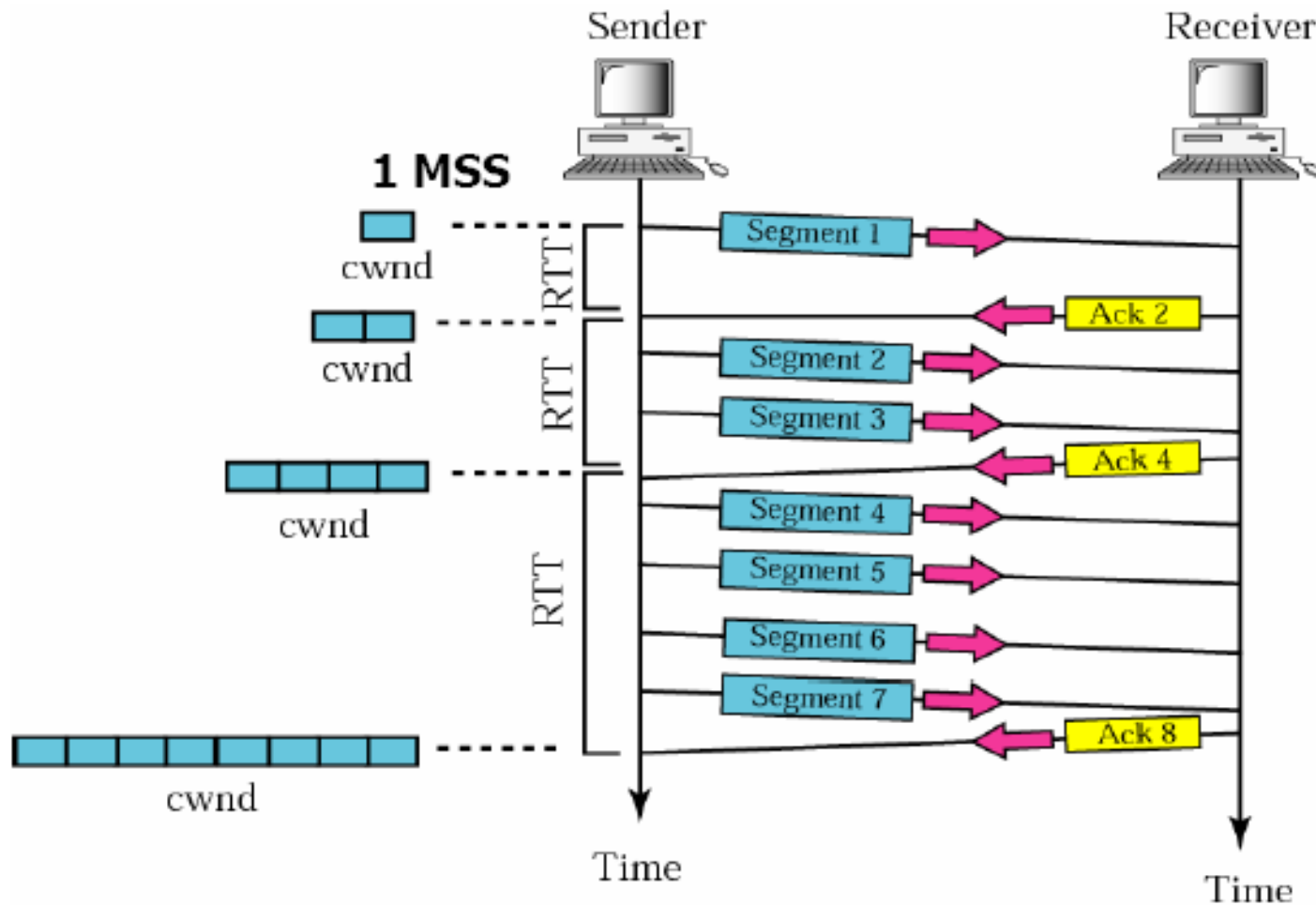
        until (loss)

End

**Slow-start**  
(expon.  
growth)



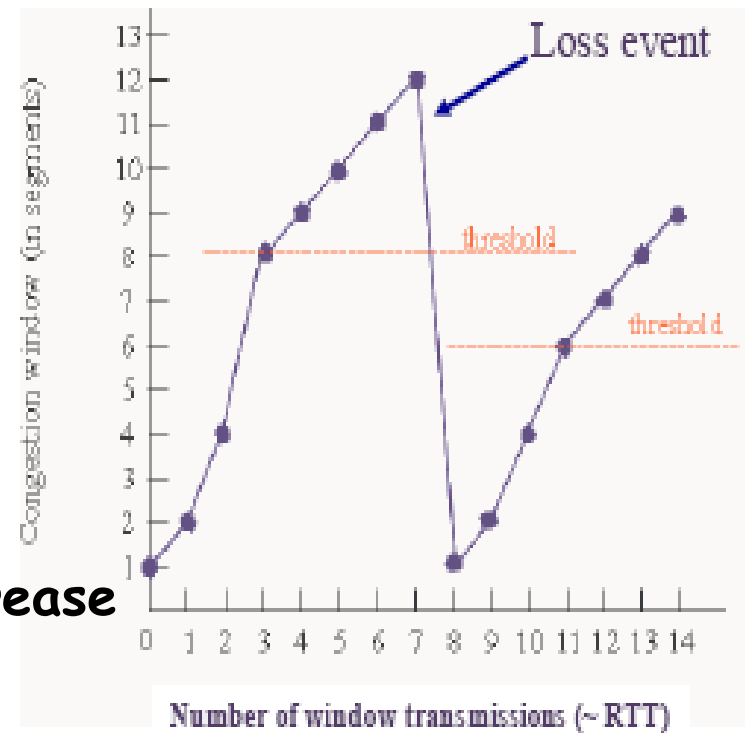
# Slow Start



## Phase II: Congestion Avoidance

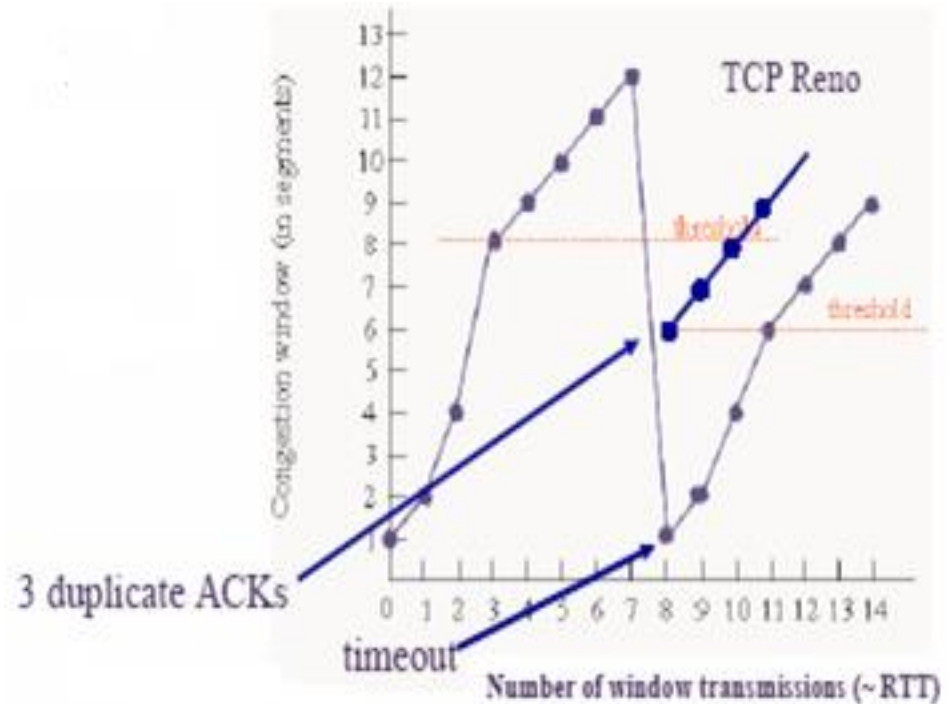
```
Begin
  if (loss)
    set threshold to (cwnd/2)
    set cwnd to 1
    perform "slow-start"
  endif
End
```

Additive Increase/Multiplicative Decrease  
(AIMD)

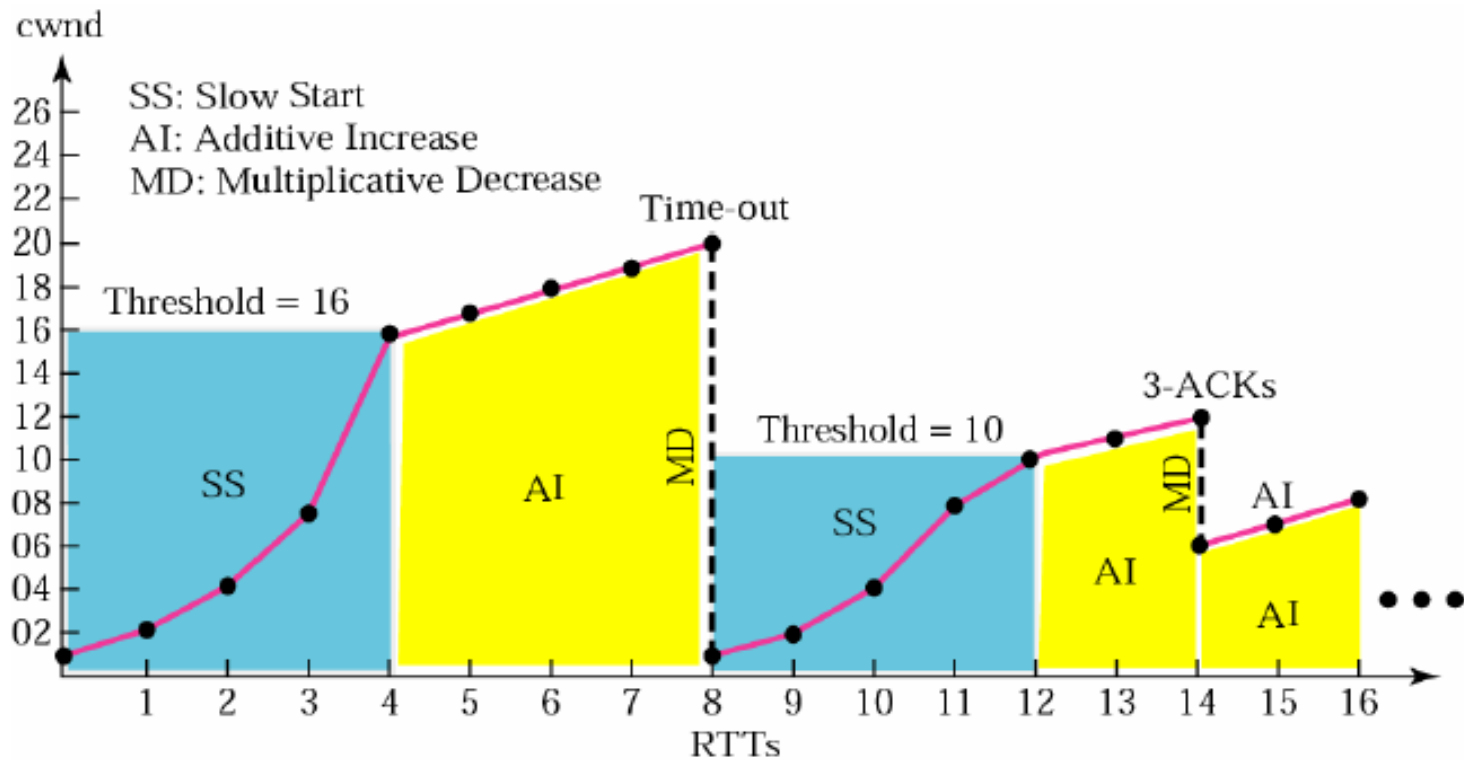


# Congestion Avoidance Refinement

- Important remark:
  - Timeout is a more alarming indicator of congestion than 3 ACKs (why?)
- When receiving 3 duplicate ACKs:
  - Set new threshold to  $\text{cwnd}/2$
  - Set new cwnd to new threshold
  - Increase cwnd linearly



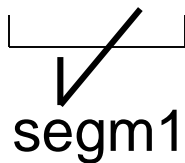
# Example



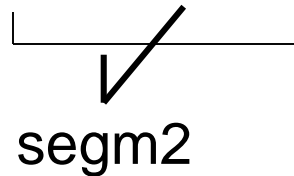
# TCP Sliding Window Protocol

- TCP uses cumulative acknowledgment:
- The ACK number is the number of the next expected byte to be received
- Ex: Sending segments with bytes ID as follows:

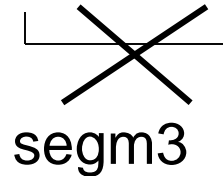
20,21



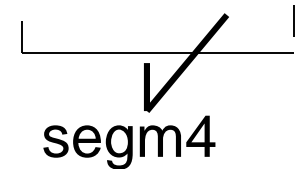
22,23,24



25,26



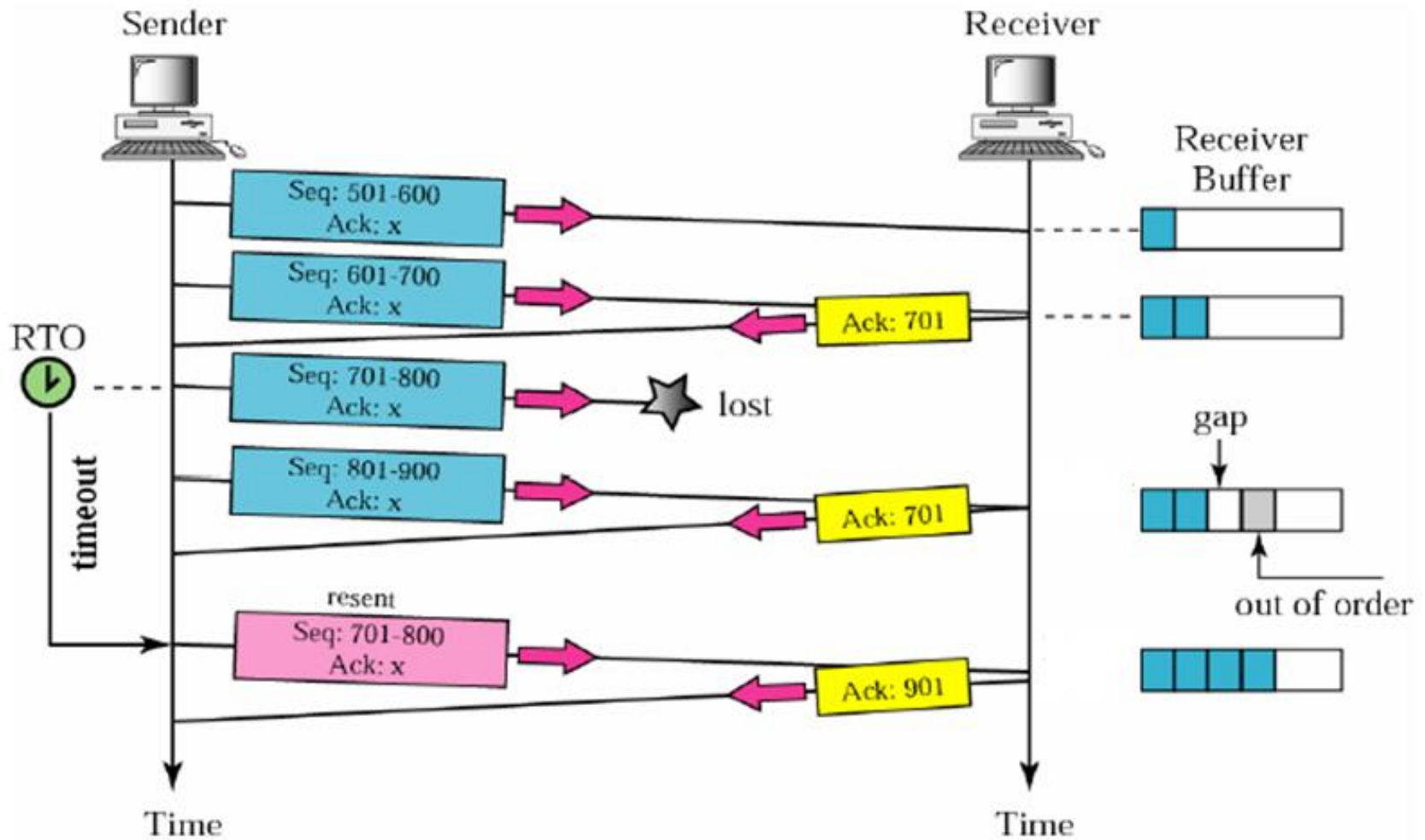
27,28,29



- If segment 1, 2 and 4 are received, segment 3 is lost:
  - ACK is sent to acknowledge up to byte 24 (ACK(25))
  - Segment 4 is buffered



# Lost segment

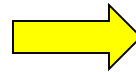


# When Receiver Sends Acknowledgment?

## Event at TCP Receiver

## TCP Receiver Action

In-order segment arrival with expected seq.#; all previous data ACKed



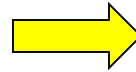
Delayed ACK: wait 500 msec for next segment to send; if not, send ACK

In-order segment arrival with expected seq.#; one delayed ACK pending



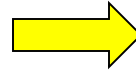
Immediately send single cumulative ACK for both segments

Out-of-order segment arrival (higher seq. num); gap detected



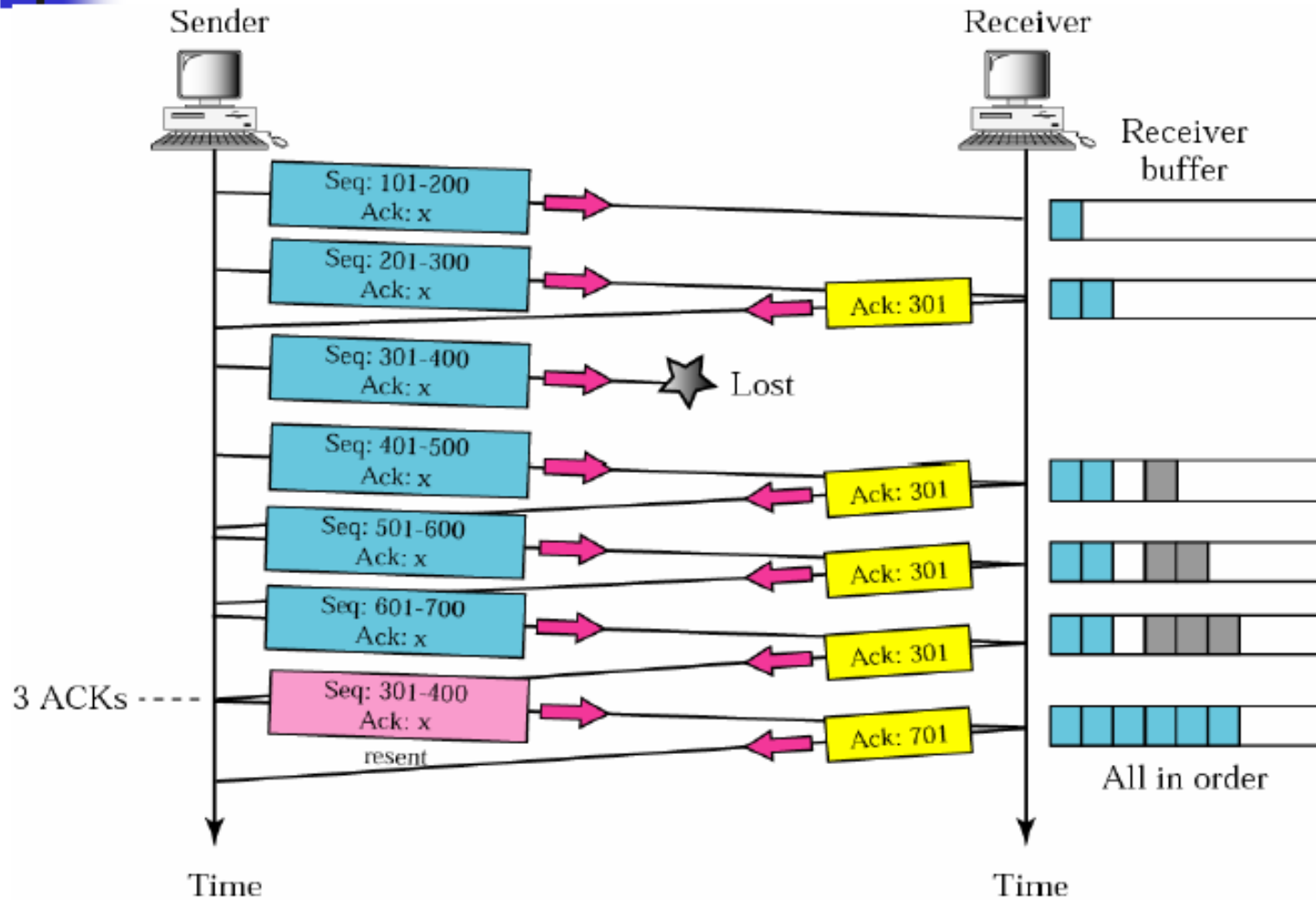
Send duplicate ACK for expected segment sequence number

Arrival of segment that Partially or completely fills the gap

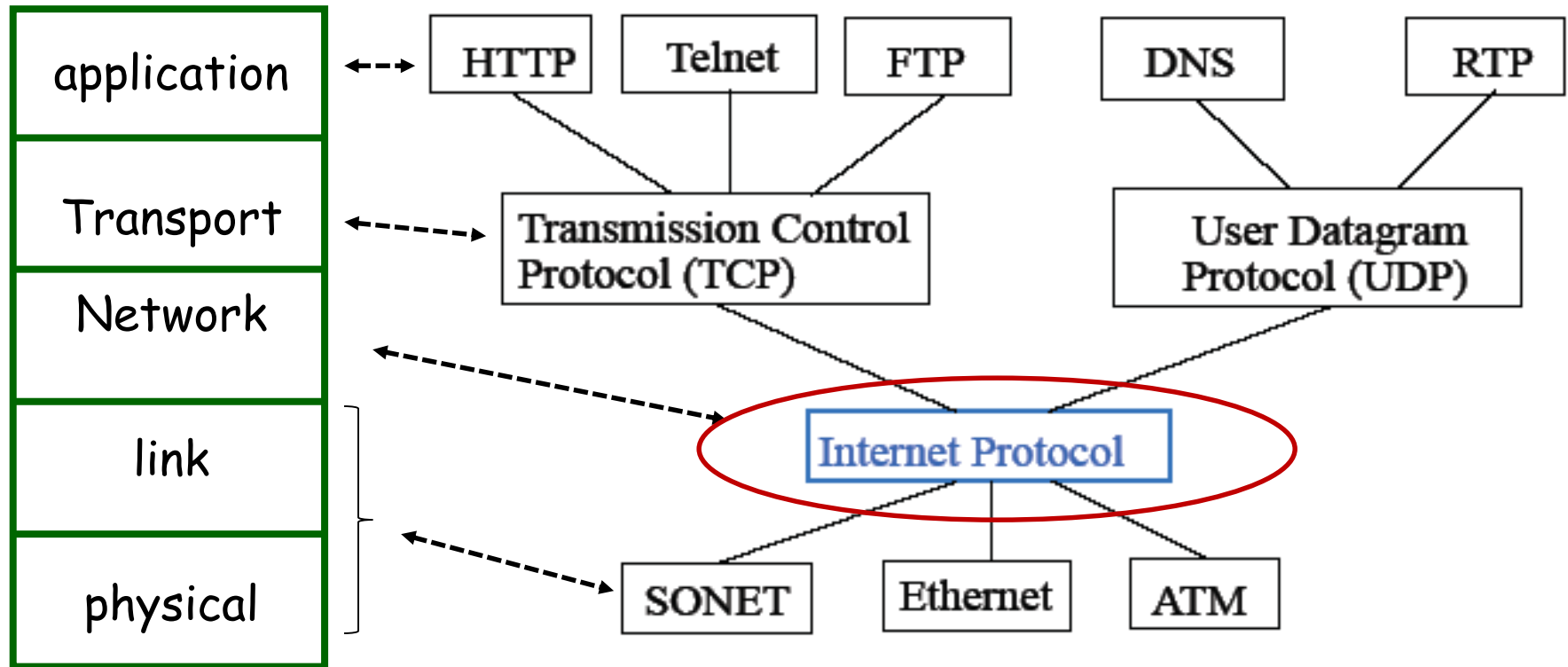


Immediate ACK if segment Starts at lower end of gap

# Fast Retransmission

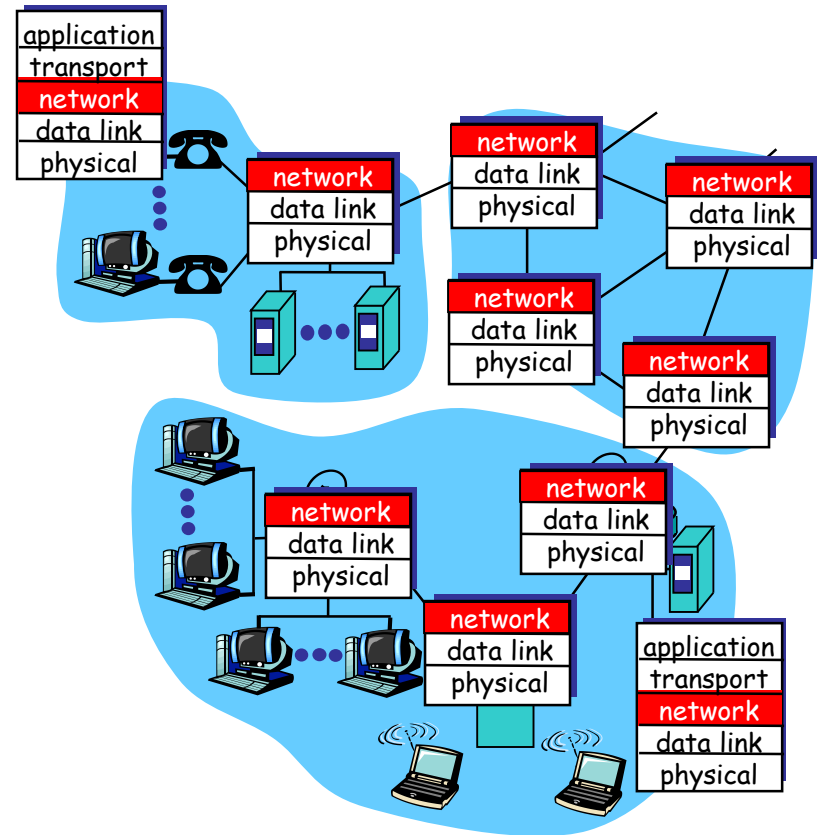


## Where Are We Now?

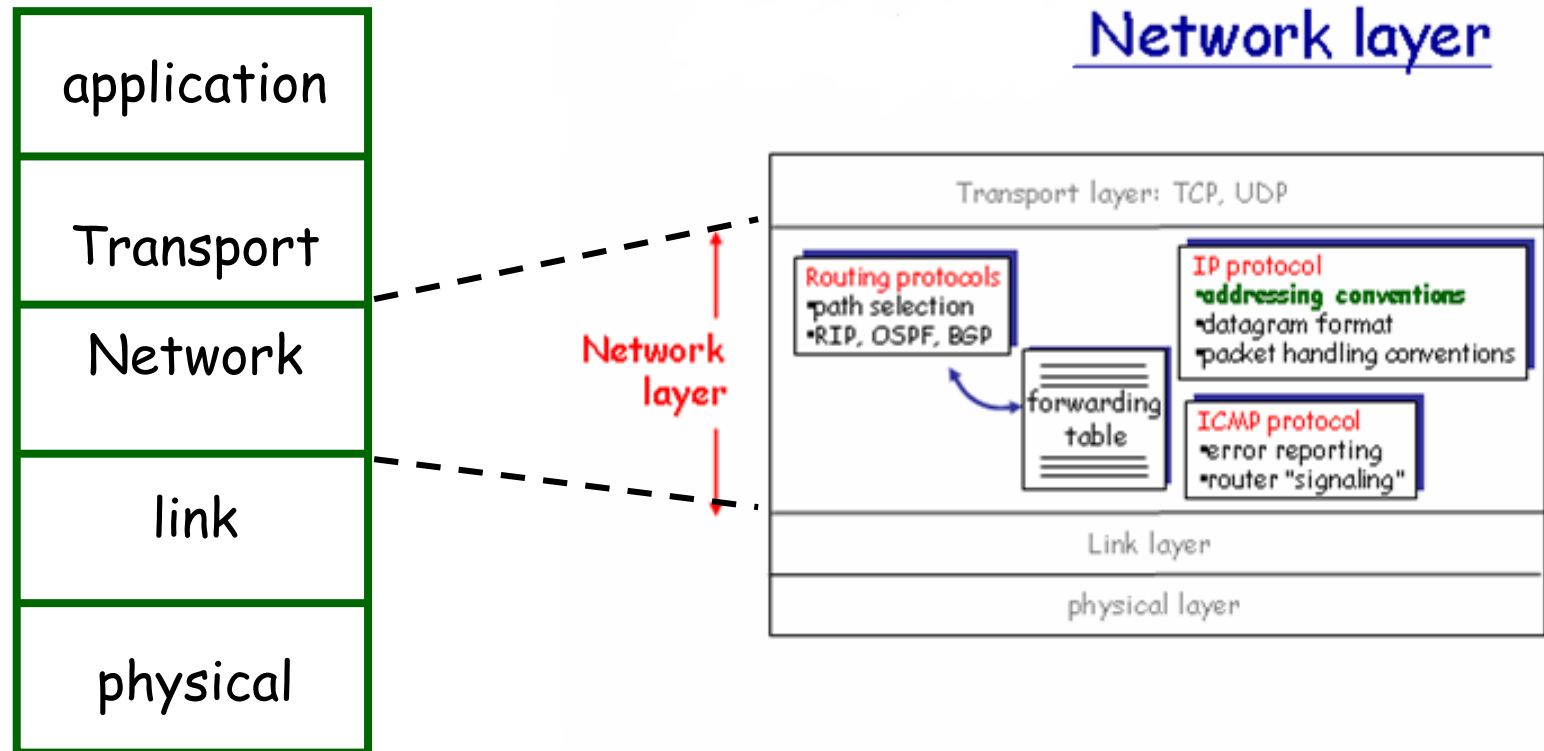


# Network layer

- Move packets from sending to receiving host
- Network layer functions:
  - Forwarding
    - Moves pkt from an input link to the output link
  - Routing:
    - Determines routing path from source to destination (routing algorithm)



# Network Layer in the Internet



# IP Addressing Conventions

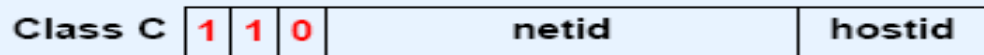
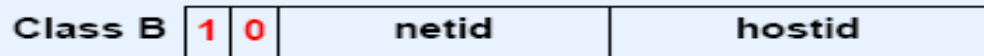
- 32 bits (4 bytes) unique value for each host
- Address composed of 2 parts (2-level hierarchy):
  - **Prefix** (network ID): identifies network to which host attaches
  - **Suffix** (host ID): identifies host on that network



- Dotted-decimal notation: Each byte is written in decimal in MSB order, separated by decimals. Example:

32-bit address	<u>10011000</u>	<u>00000001</u>	<u>00110110</u>	<u>00110000</u>			
Dotted decimal representation	152	.	1	.	54	.	48

# Original IPv4 Address Classes



## Three Principle Classes



## Other (seldom used) Classes

class				
A	NetID	hostID	hostID	hostID
B	NetID	NetID	hostID	hostID
C	NetID	NetID	NetID	hostID



# Special IP Addresses

For Address of Type...	If Network part is...	And Host part is ...	Then this means ...
--	Anything other than all 0's or all 1's	All 0's	The address of the whole network
Destination	Anything other than all 0's or all 1's	All 1's	Broadcast address for the specified network
Destination	All 1's	All 1's	Broadcast address for same network as originating host
Destination	127 (Class A, all 1's)	Anything	"This computer" (source of the packet)



loopback - never sent on network

# Important Remarks

- Classful addresses are self-identifying
- Consequences
  - Can determine boundary between prefix and suffix from the address itself
  - No additional state needed to store boundary information
  - Both hosts and routers benefit