Lecture 24: Networking (cont'd)

CS 105

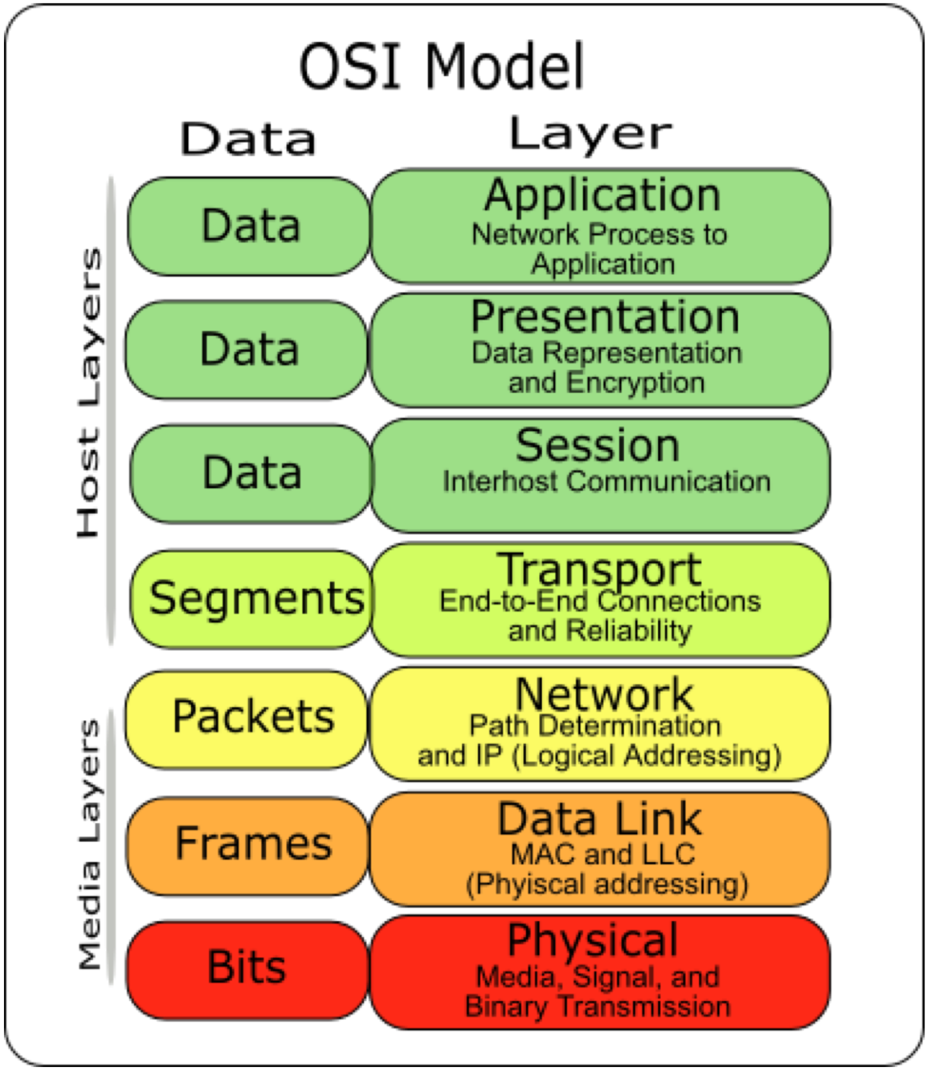
April 22, 2019

OSI Model

IP

Frames

Wires



OSI Model
Data
Layer
Host Layers
Data
Application
Network Process to Application
Data
Presentation
Data Representation and Encryption
Data
Session
Interhost Communication
Segments
Transport
End-to-End Connections and Reliability
Media Layers
Packets
Network
Path Determination and IP (Logical Addressing)
Frames
Data Link
MAC and LLC (Phyiscal addressing)
Bits
Physical
Media, Signal, and Binary Transmission

OSI Layers

Transport	data transfer, logical communication between processes on hosts	???
Network	path determination across multiple network segments, routing, logical addressing	IP
Link	decides whose turn it is to talk, finds physical device on network	Ethernet, 802.11
Physical	exchanges bits on the media	wires, signal encoding

Should the network guarantee packet delivery?

Transport Layer Protocols

User Datagram Protocol (UDP)

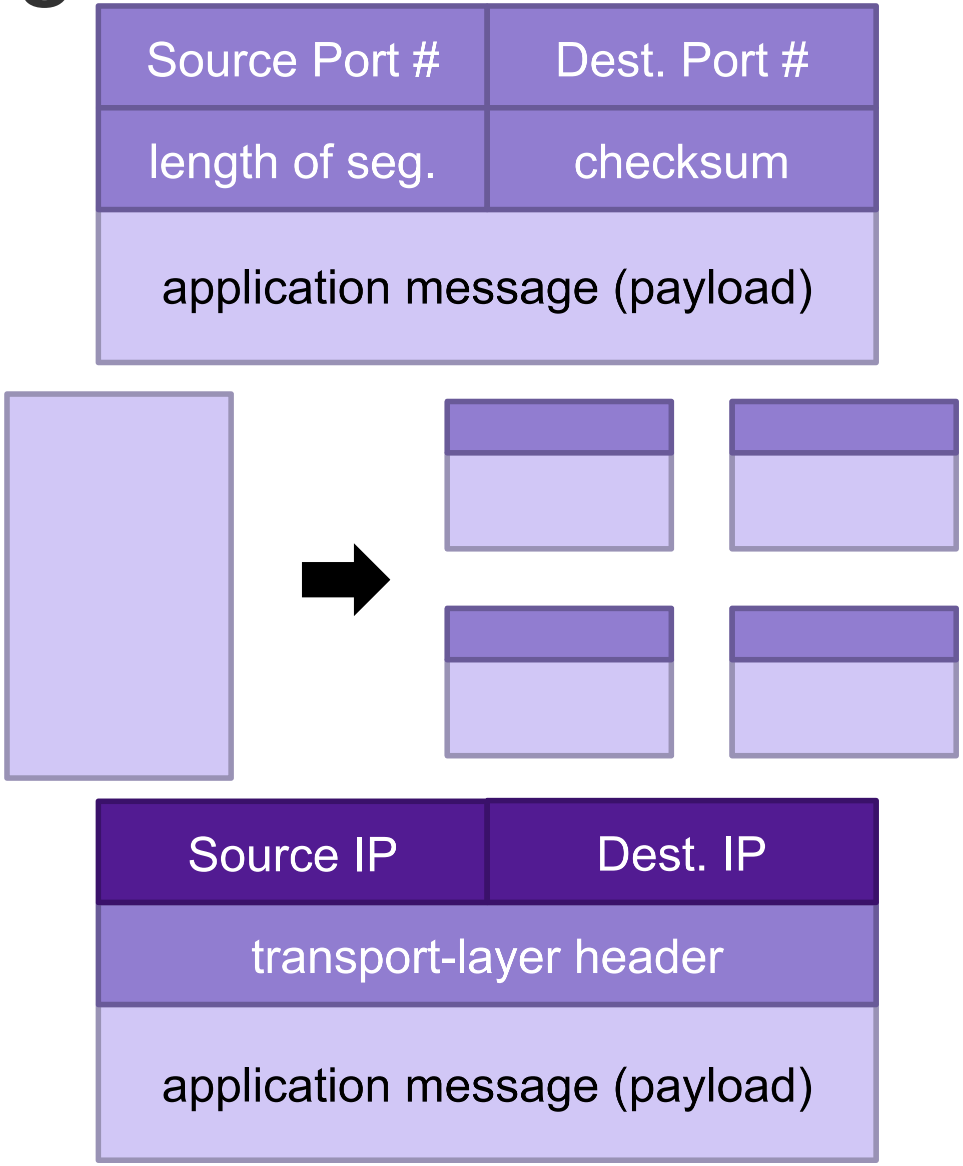
- **unreliable, unordered delivery**
- connectionless
- best-effort, segments might be lost, delivered out-of-order, duplicated
- reliability (if required) is the responsibility of the app

Transmission Control Protocol (TCP)

- **reliable, inorder delivery**
- connection setup
- flow control
- congestion control

UDP: creating a segment

- Sending application:
  - specifies IP address and port
  - uses socket bound to source port
- Transport Layer (UDP):
  - breaks application message into smaller chunks
  - adds transport-layer header to each message to form a segment
  - header size = 8 bytes
- Network Layer (IP):
  - adds network-layer header to each datagram



Source Port #

Dest. Port #

length of seg.

checksum

application message (payload)

Source IP

Dest. IP

transport-layer header

application message (payload)

UDP: tradeoffs

- fast:
  - no connection setup
  - no rate-limiting
- simple:
  - no connection state
  - small header (8 bytes)

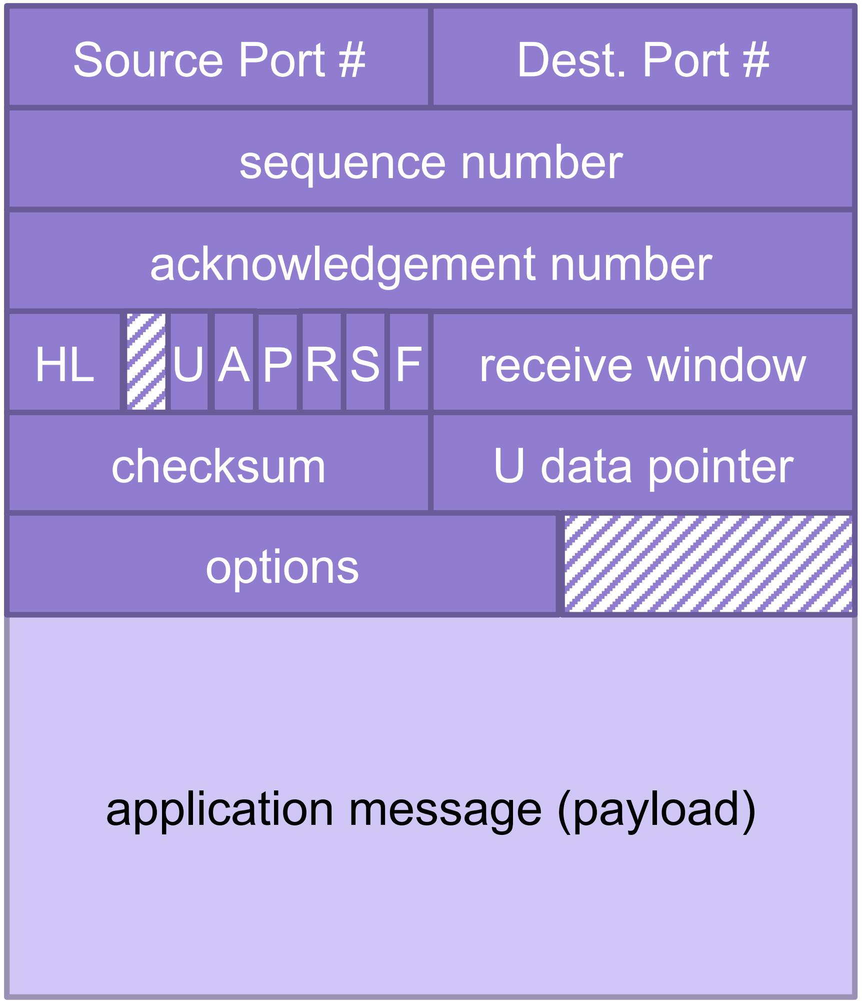
- (possibly) extra work for applications
  - reordering
  - duplicate suppression
  - handle missing packets

Transport Protocols by Application

Application	Application-Level Protocol	Transport Protocol
Name Translation	DNS	Typically UDP
Routing Protocol	RIP	Typically UDP
Network Management	SNMP	Typically UDP
Remote File Server	NFS	Typically UDP
Streaming multimedia	(proprietary)	UDP or TCP
Internet telephony	(proprietary)	UDP or TCP
Remote terminal access	Telnet	TCP
File Transfer	(S)FTP	TCP
Email	SMTP	TCP
Web	HTTP(S)	TCP

TCP Segment Format

- S,A,F flags used to manage connection



Source Port # | Dest. Port #

sequence number

acknowledgement number

HL | ▨ | U | A | P | R | S | F | receive window

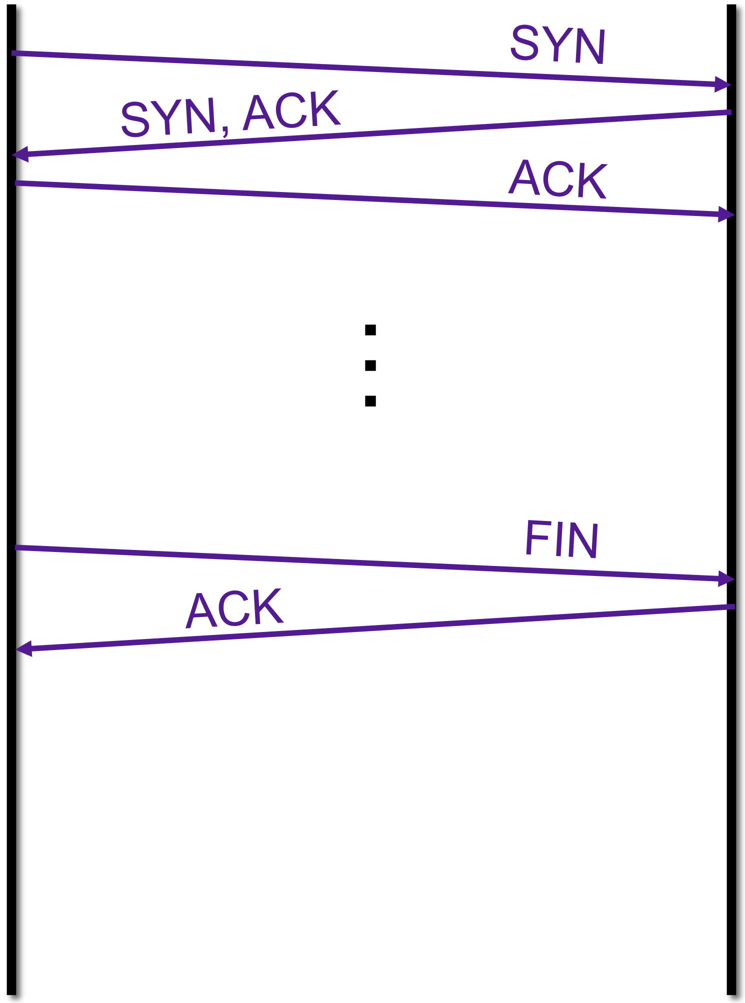
checksum | U data pointer

options | ▨

application message (payload)

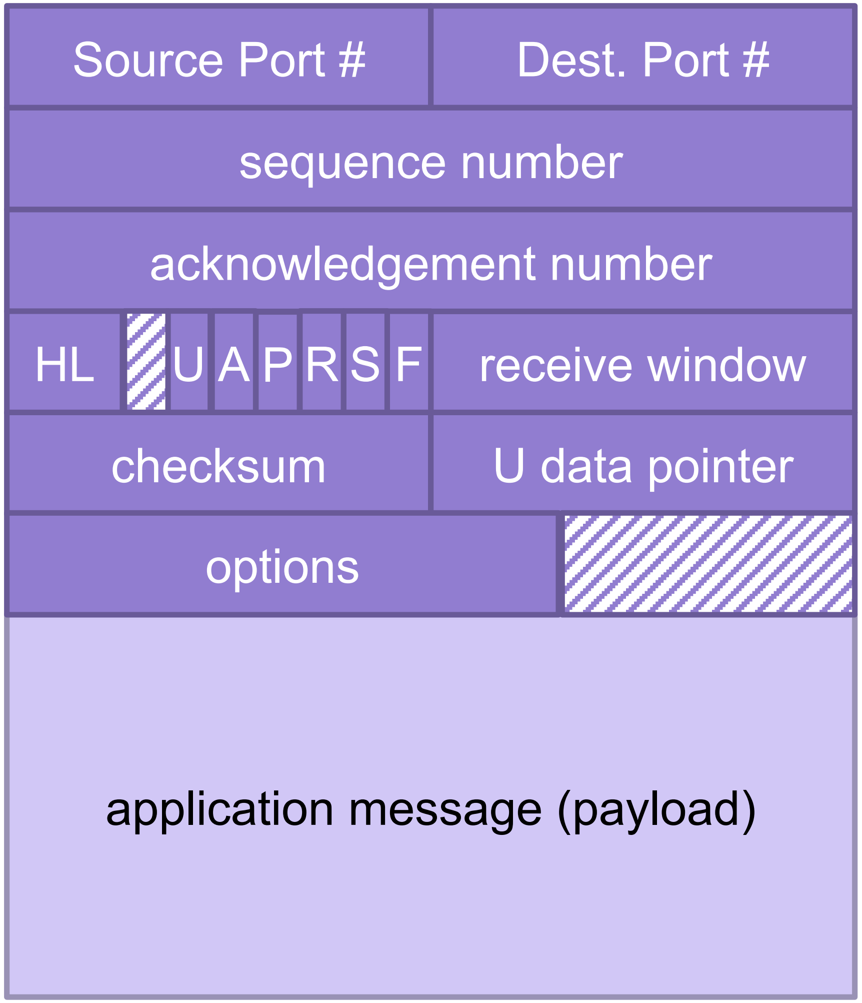
TCP Connections

- TCP is connection-oriented
- A connection is initiated with a three-way handshake
  - Recall: server will typically create a new socket to handle the new connection
- FIN works (mostly) like SYN but to teardown a connection



TCP Segment Format

- S,A,F flags used to manage connection
- sequence number and acknowledgement number (+ A flag) are used to implement reliable transport



Source Port # | Dest. Port #

sequence number

acknowledgement number

HL | U | A | P | R | S | F | receive window

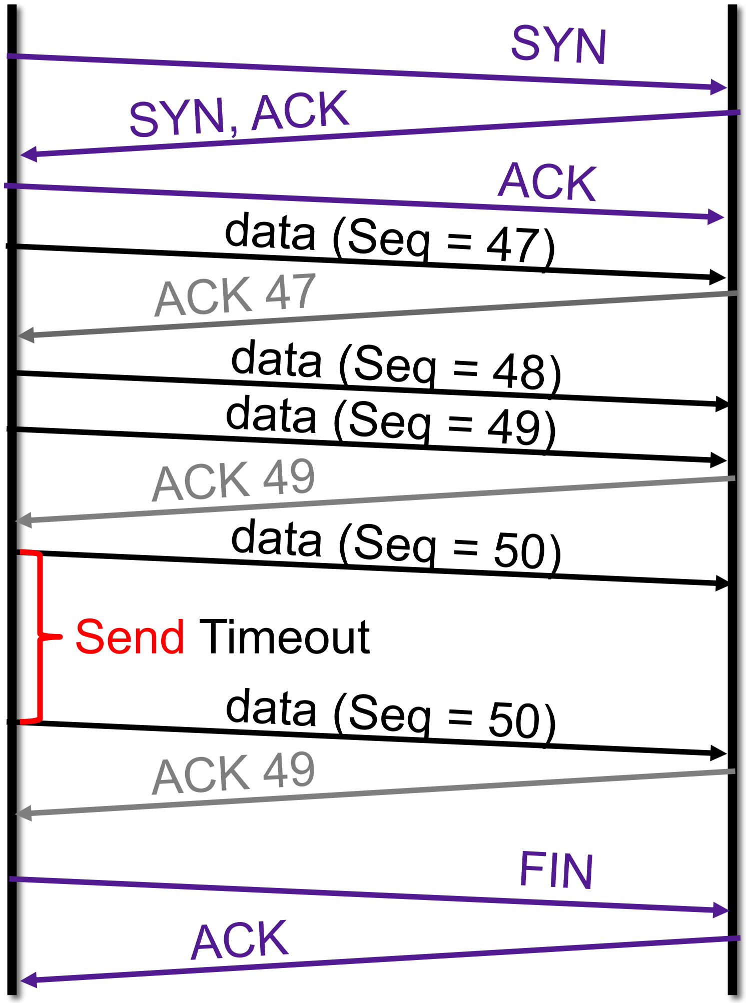
checksum | U data pointer

options

application message (payload)

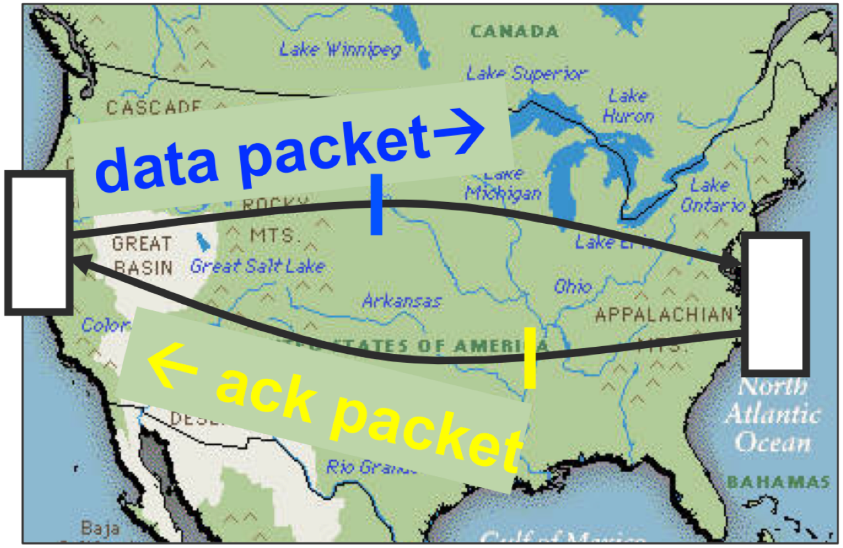
Reliable Transport

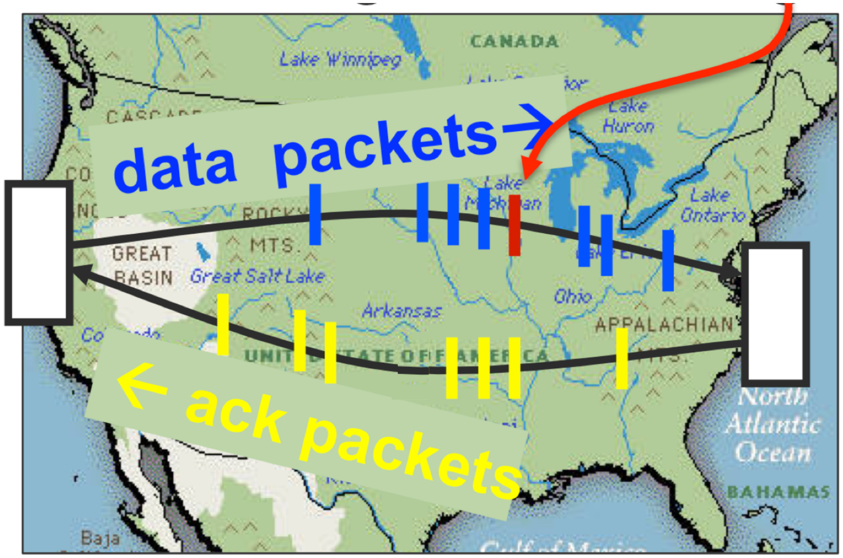
- Each SYN segment will include a randomly chosen sequence number
- Sequence number of each segment is incremented by data length
- Receiver sends ACK segments acknowledging latest sequence number received
- Sender maintains copy of all sent but unacknowledged segments; resends if ACK does not arrive within timeout
- Timeout is dynamically adjusted to account for round-trip delay



Pipelined Protocols

- Pipelining allows sender to send multiple "in-flight", yet-to-be-acknowledged packets
  - increases throughput
  - needs buffering at sender and receiver
- what if a packet in the middle goes missing?
- how big should the window be?

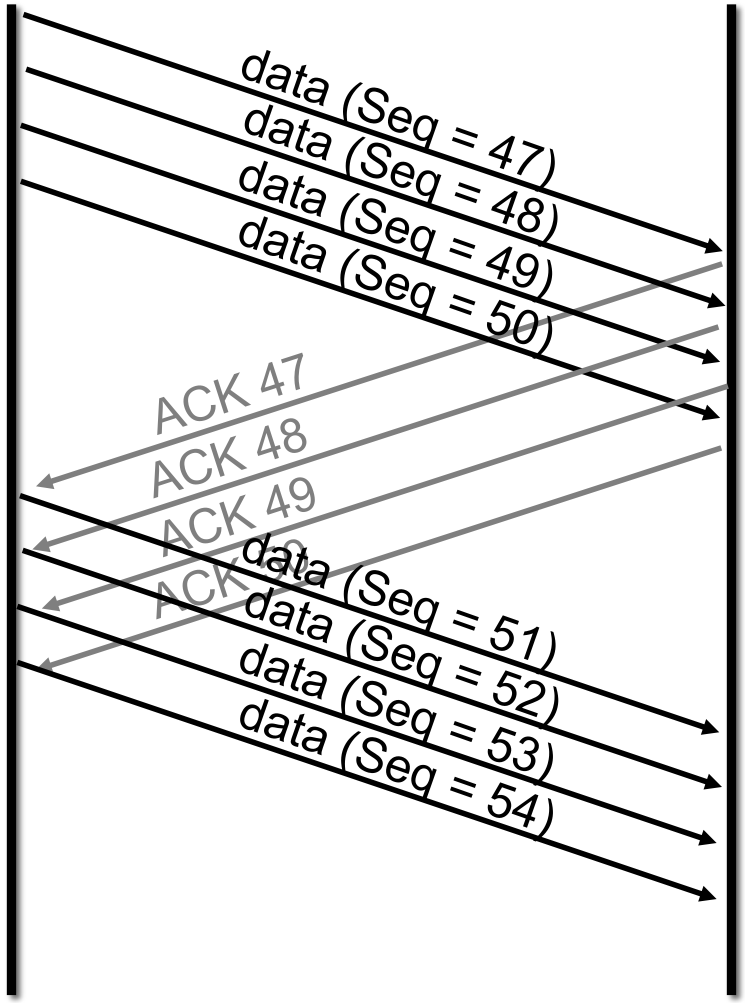




Example: Window Size = 4

- sender can have up to 4 unacknowledged messages

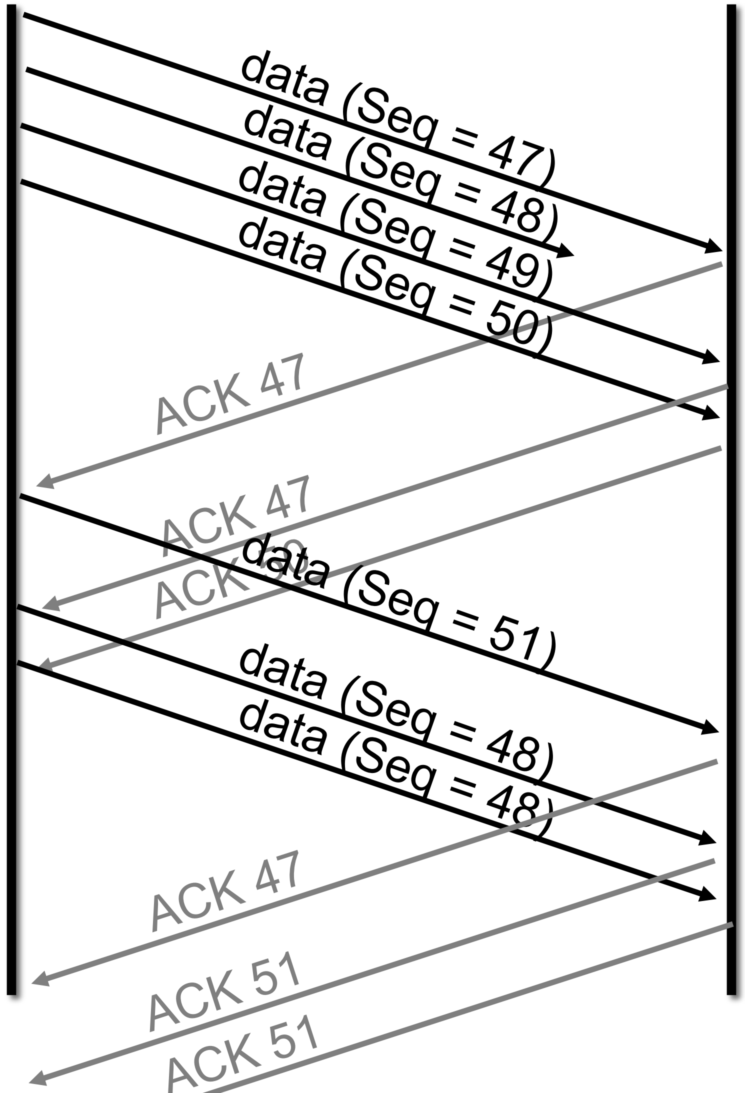
- when ACK for first message is received, it can send another message



TCP Fast Retransmit

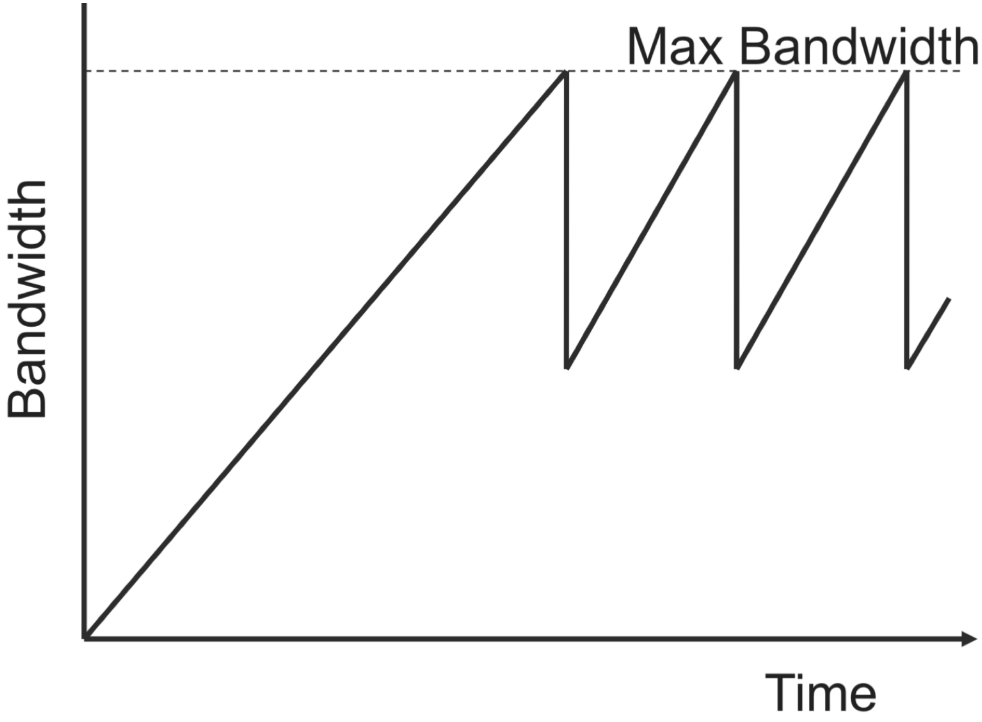
- Receiver always acks the last id it successfully received

- Sender detects loss without waiting for timeout, resends missing packet



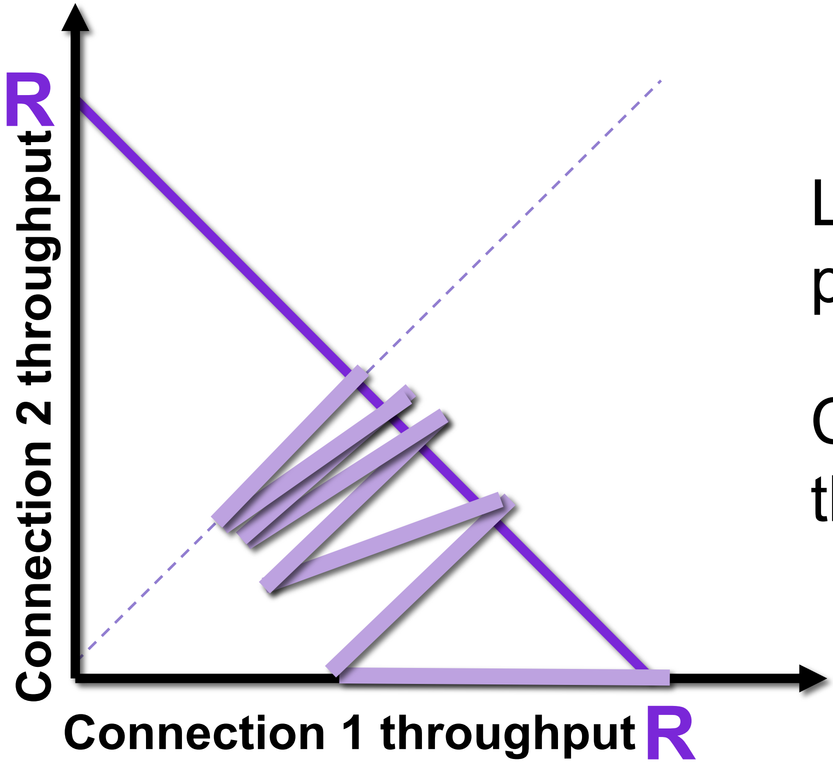
TCP Congestion Control

- TCP operates under a principle of additive increase-multiplicative decrease
  - window size++ every RTT if no packets lost
  - window size/2 if a packet is dropped



TCP Fairness

- Goal: if k TCP sessions share same bottleneck link of bandwidth R, each should have average rate of R/k



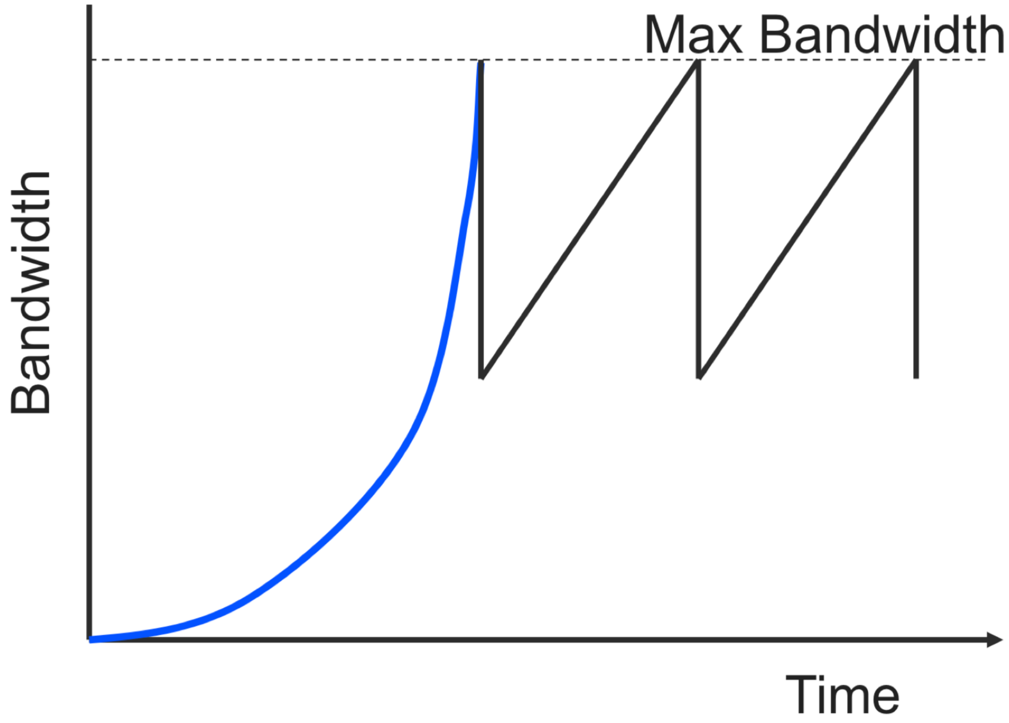
Loss: decreases throughput proportional to current bandwidth

Congestion avoidance: increases throughput linearly (evenly)

TCP Slow Start

- Problem: linear increase takes a long time to build up a decent window size, and most transactions are small

- Solution: allow window size to increase exponentially until first loss

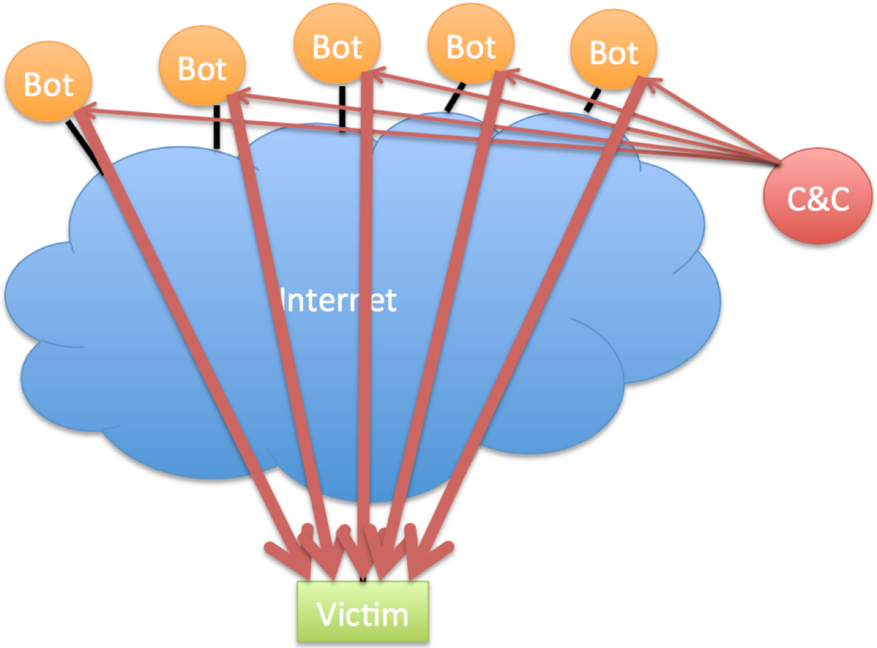


TCP Summary

- Reliable, in-order message delivery
- Connection-oriented, three-way handshake
- Transmission window for better throughput
  - timeouts based on link parameters (e.g., RTT, variance)
- Congestion control
  - Linear increase, exponential backoff
- Fast adaptation
  - Exponential increase in the initial phase

Denial of Service Attacks

- a **SYN flood** is a simple denial of service attack
- implemented by sending SYN messages (first message of TCP handshake) and suppressing ACK messages (third message of handshake)



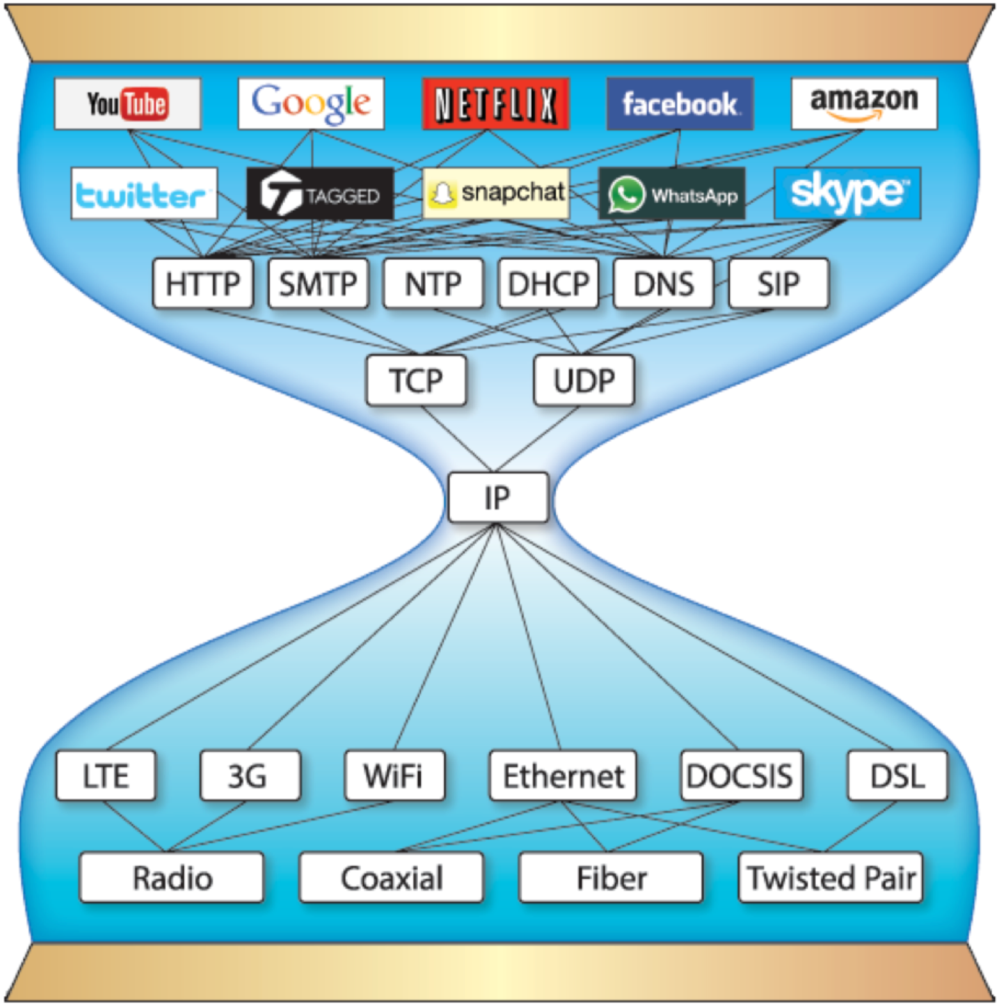
Example DDoS Attack

vs

Dyn

SM

Continuing up the Network Stack…


YouTube
Google
NETFLIX
facebook
amazon
twitter
TAGGED
snapchat
WhatsApp
skype
HTTP
SMTP
NTP
DHCP
DNS
SIP
TCP
UDP
IP
LTE
3G
WiFi
Ethernet
DOCSIS
DSL
Radio
Coaxial
Fiber
Twisted Pair