

CS 372 Lecture #23

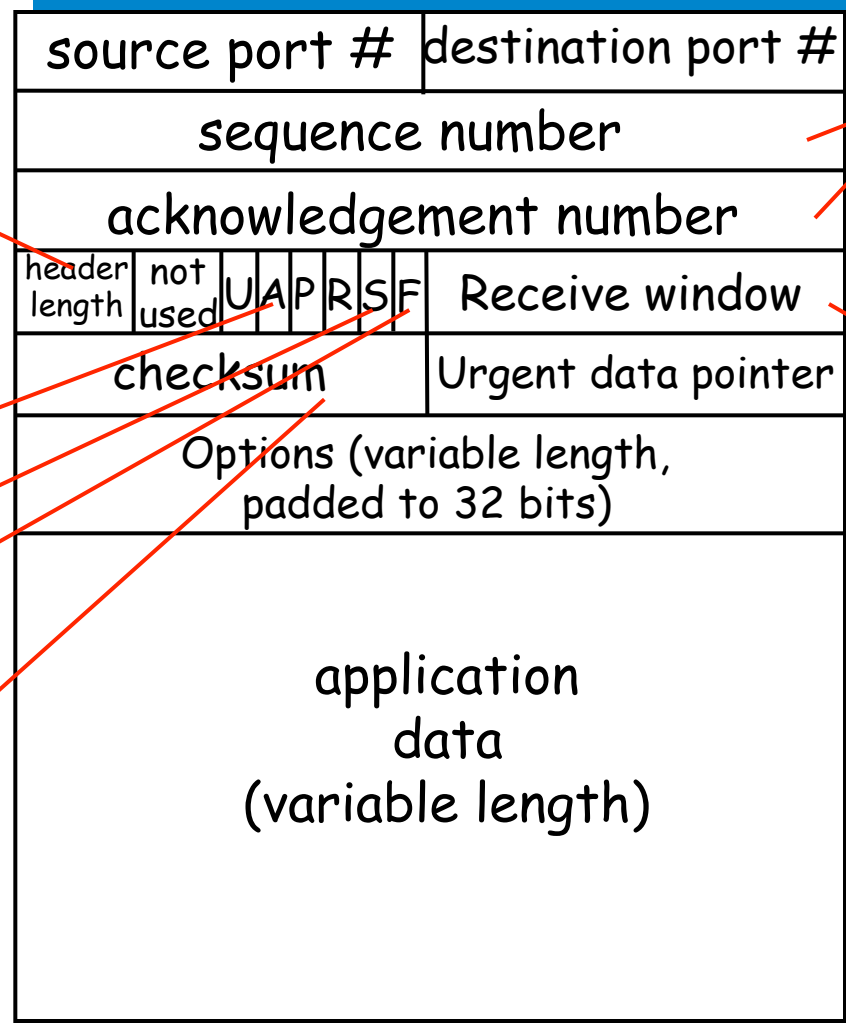
Reliable data transfer with TCP

- connection setup/teardown
- fairness
- Wrap up TCP/UDP
- Wrap up transport layer

Note: Many of the lecture slides are based on presentations that accompany *Computer Networking: A Top Down Approach*, 6th edition, by Jim Kurose & Keith Ross, Addison-Wesley, 2013.

TCP segment structure

← 32 bits →



4-bit header size.
Number of 32-bit
“lines” (minimum=5,
maximum=15)

counting
by bytes
of data
(not segments!)

bytes
receiver
is willing
to accept

ACK
SYN
FIN

Internet
checksum
(as in UDP)

TCP 3-way handshake

client state

LISTEN

↓
SYN SENT

↓
ESTAB

Choose initial
sequence number (x).
Send SYN message.



SYN bit=1, Seq#=x



Choose initial
sequence number (y).
Send SYNACK
message.

server state

LISTEN

↓
SYN RCVD

↓
ESTAB

SYN bit=1, Seq#=y
ACK bit=1, ACK#=x+1

Received SYNACK(x+1)
indicates server is live.
Send ACK.
Segment may contain
client-to-server data.

ACK bit=1, ACK#=y+1

received ACK(y+1)
indicates client is live.

TCP: closing a connection

client state

ESTAB

`clientSocket.close()`

FIN_WAIT_1

Can no longer
send, but can
receive data

FIN_WAIT_2

Wait for
server close.

TIMED_WAIT

Timed wait
(in case ACK
is lost)

CLOSED



FIN bit=1, Seq#=x

ACK bit=1, ACK#=x+1

FIN bit=1, Seq#=y

ACK bit=1, ACK#=y+1

Send ACK
message.
Can still
send data.

Can no longer
send data.

server state

ESTAB

CLOSE_WAIT

LAST_ACK

CLOSED

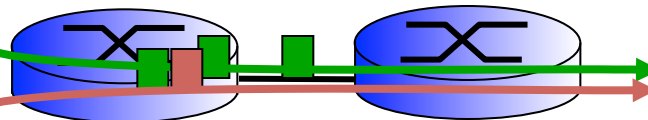
TCP Fairness

Fairness goal: if n TCP sessions share the same bottleneck link of bandwidth R , each should have average rate of R/n

TCP connection 1



TCP connection 2



bottleneck
router
capacity R

- TCP is fair to each connection
- Not necessarily fair to each host
 - application can open multiple parallel connections between two hosts
 - web browsers do this

Compare UDP and TCP

- UDP provides
 - connectionless service
 - *end-to-end best-effort (unreliable)* delivery
- TCP provides
 - connection-oriented service
 - *end-to-end reliable byte stream* delivery
- Both
 - use IP (network layer) for delivery to destination
 - send to / receive from multiple applications
 - de-multiplex to destination application protocol ports

Compare UDP and TCP

- **UDP** is simple and fast
 - 8-byte header
- **TCP** is complicated
 - 20-byte header (minimum)
 - adds a huge amount of overhead to achieve reliability
 - especially for acknowledgements and congestion control
 - amazingly ... it works
 - depends on extremely fast and reliable hardware in the network core.

- TCP connection setup/teardown
 - 3-way handshake
 - closing a connection
- TCP “fairness”
- Transport layer protocols
 - provide application-to-application delivery
 - accept data from: / deliver data to: application layer processes
 - depend on network layer protocols to carry segments through the network core