

Lecture 6:

Transmission Control Protocol

CSE 123: Computer Networks
Alex C. Snoeren



Project 1a due MONDAY

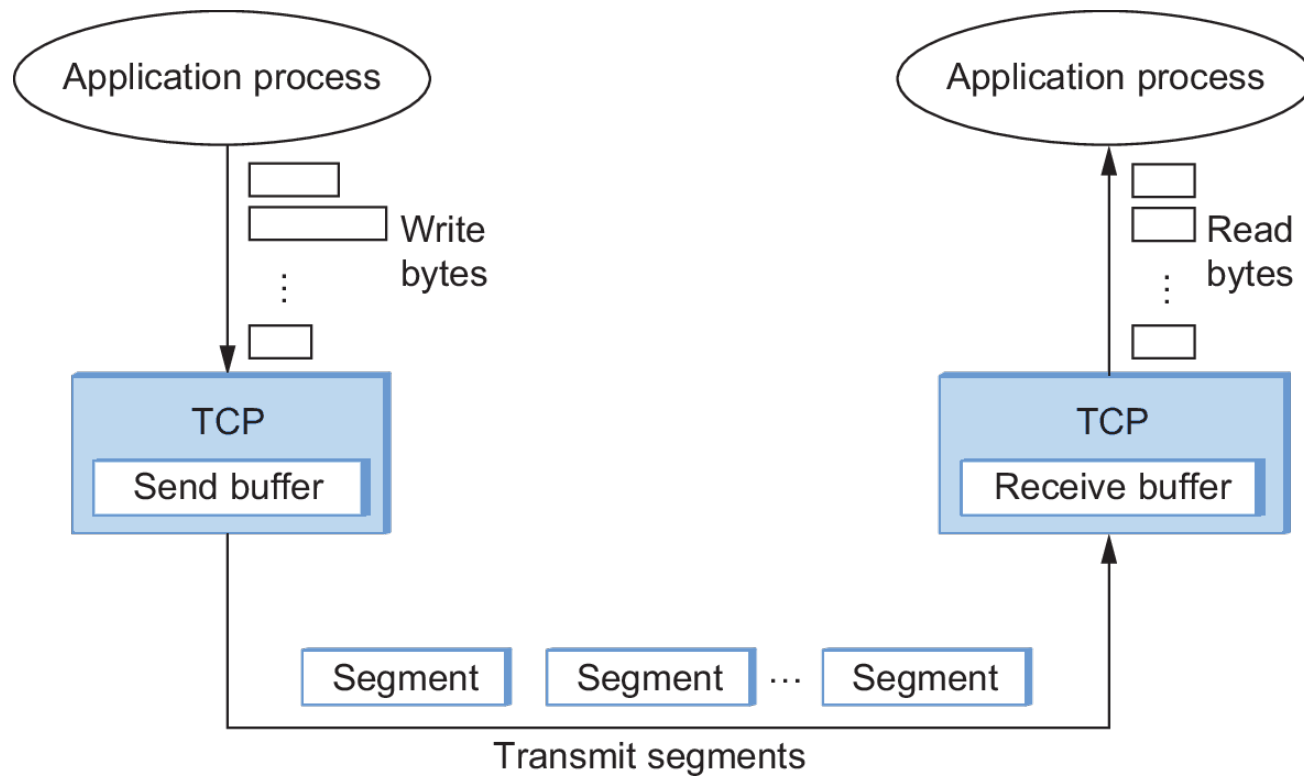


Lecture 6 Overview

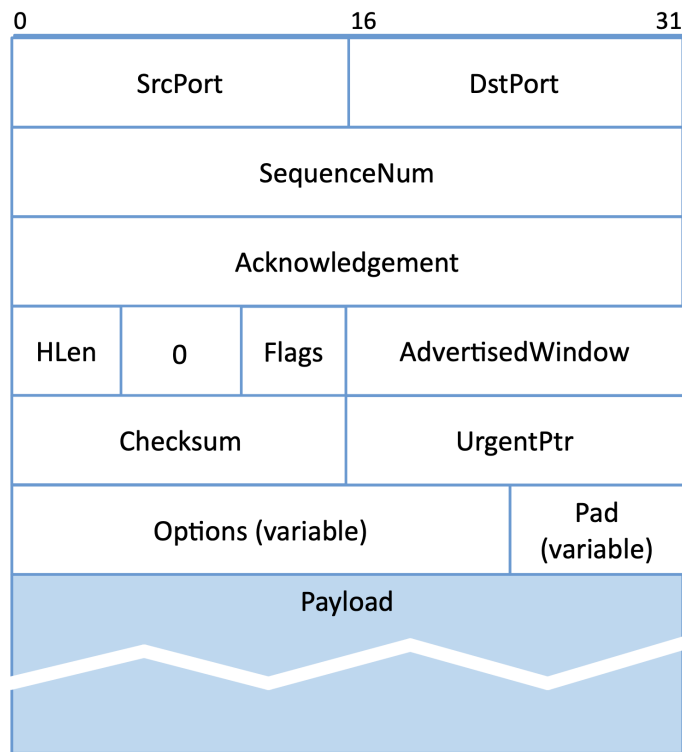


- Transport Control Protocol
 - ◆ Delivery model and header format
 - ◆ Connection establishment
 - ◆ Connection teardown
- Congestion control introduction
 - ◆ Bandwidth Allocation
 - ◆ Fairness
 - ◆ Congestion Collapse

TCP Delivery



TCP Header



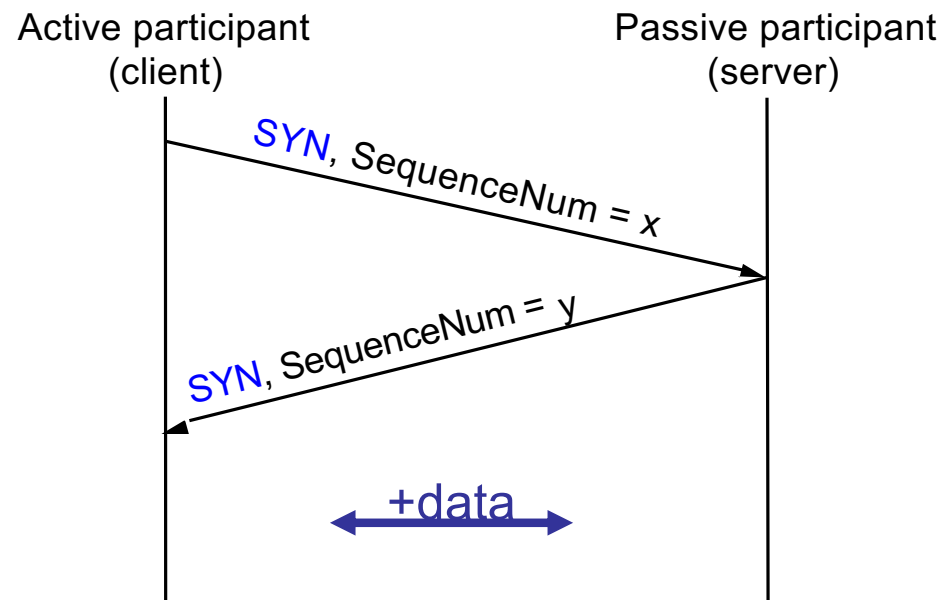
- Every segment has a header
 - ◆ Ports duplex like UDP
 - ◆ Seq/Ack/AdvWin are sliding window
 - ◆ Checksum actually computed
- Can be variable length
 - ◆ Options fields, header length
- Flags are used for connection management

Connection Establishment



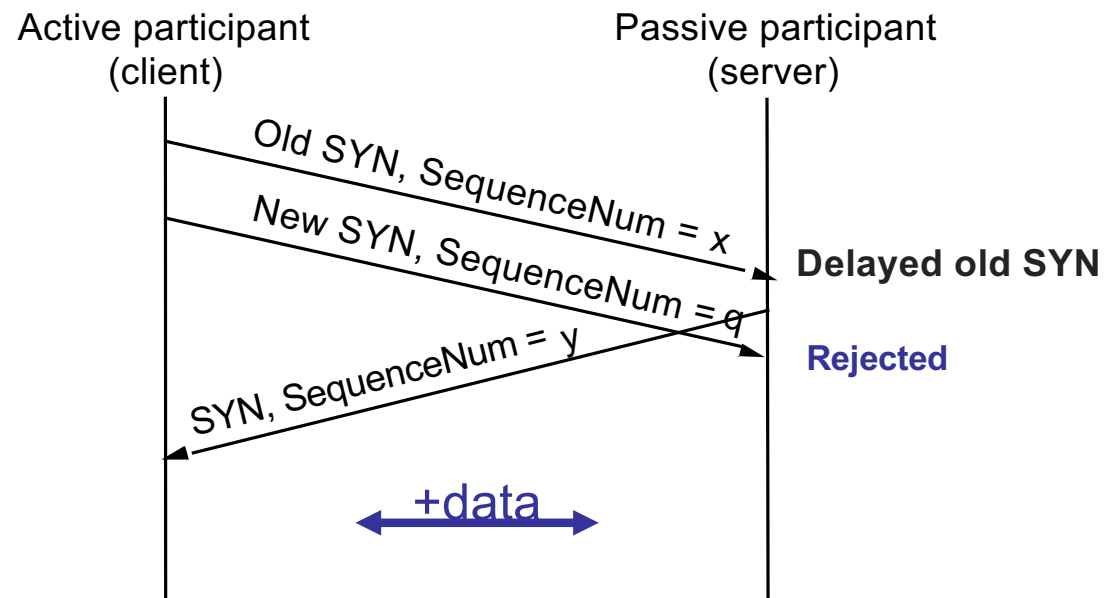
- Both sender and receiver must be ready before we start to transfer the data
 - ◆ Sender and receiver need to agree on a set of parameters
 - ◆ Most important: sequence number space in each direction
 - ◆ Lots of other parameters: e.g., the advertised window size
- Handshake protocols: setup state between two oblivious endpoints
 - ◆ Need to deal with **delayed** and **reordered** packets

Two-way handshake?



What's wrong here?

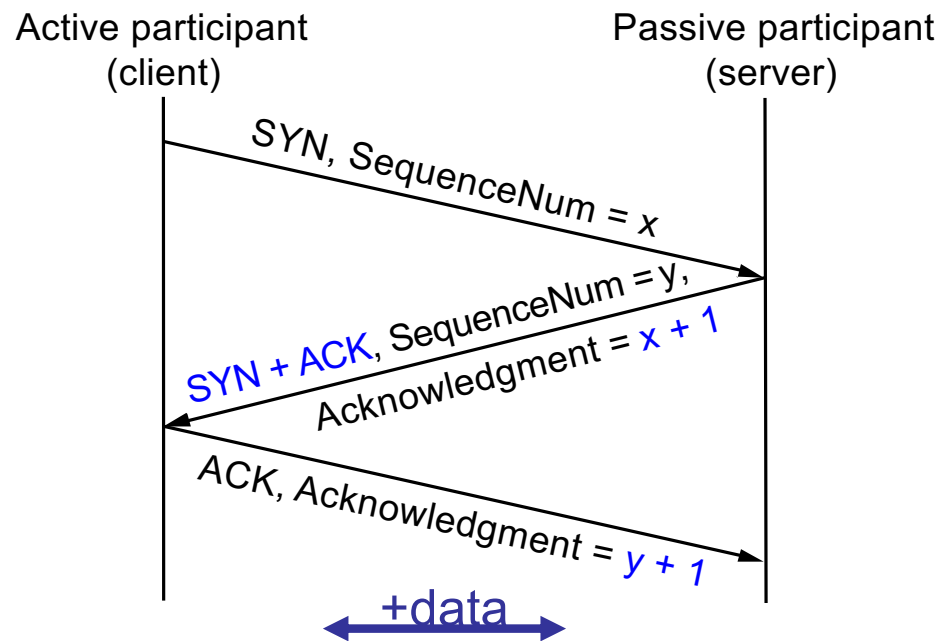
Two-way handshake?





Three-Way Handshake

- Opens both directions for transfer

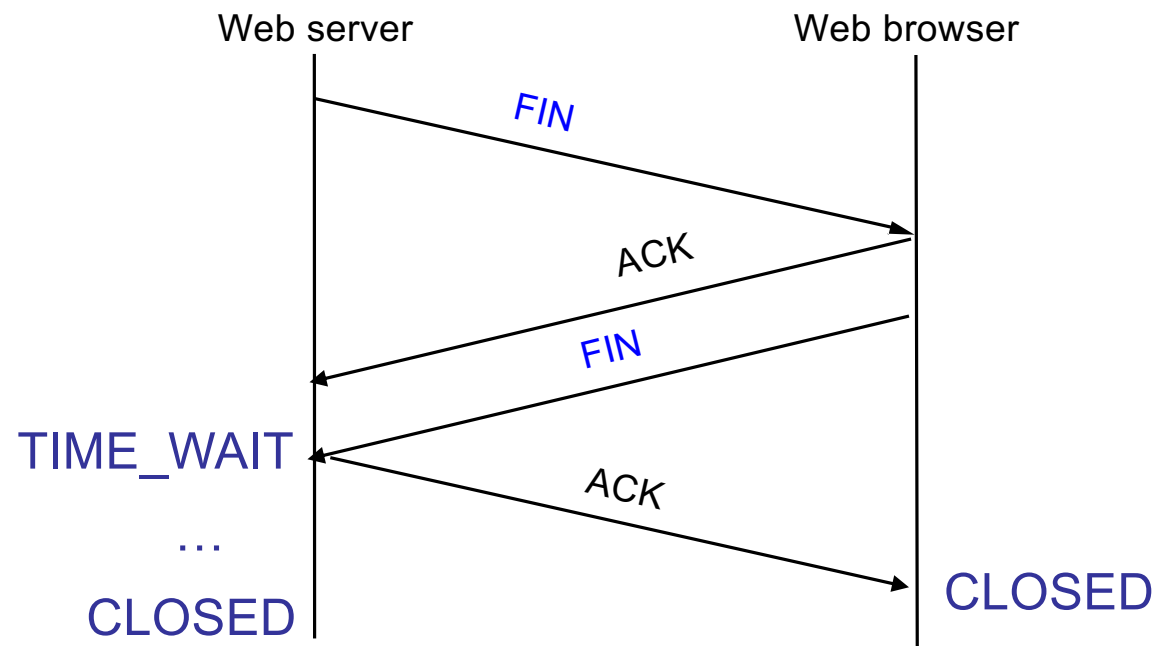


Connection Teardown



- Orderly release by sender and receiver when done
 - ◆ Delivers all pending data and “hangs up”
- Cleans up state in sender and receiver
- TCP provides a “symmetric” close
 - ◆ Both sides shutdown independently

TCP Connection Teardown





The TIME_WAIT State

- We wait $2*MSL$ (maximum segment lifetime of 120 seconds) before completing the close
 - ♦ Why?
- ACK might have been lost and so FIN will be resent
 - ♦ Could interfere with a subsequent connection
- Real life: Abortive close
 - ♦ Don't wait for $2*MSL$, simply send **Reset** packet (**RST**)
 - ♦ Why?

Some Comments



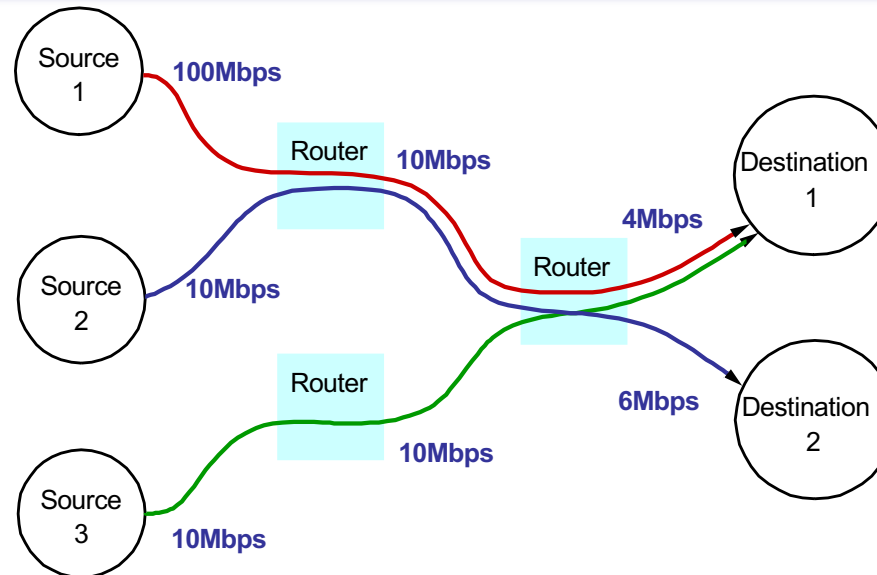
- We could abbreviate this setup, but it was chosen to be robust, especially against delayed duplicates
 - ◆ Three-way handshake from Tomlinson 1975
- Choice of changing initial sequence numbers (ISNs) minimizes the chance of hosts that crash getting confused by a previous incarnation of a connection
- How to choose ISNs?
 - ◆ Maximize period between reuse
 - ◆ Minimize ability to guess (why?)

CC Overview



- How fast should a TCP sender transmit data?
 - ◆ Not too fast, not too slow, just right...
- Should not be faster than the sender's share
 - ◆ Bandwidth allocation
- Should not be faster than the network can process
 - ◆ Congestion control
- Congestion control & bandwidth allocation are separate ideas, but frequently combined

Bandwidth Allocation



- How much bandwidth should each **flow** from a source to a destination receive when they compete for resources?
 - ♦ What is a “fair” allocation?



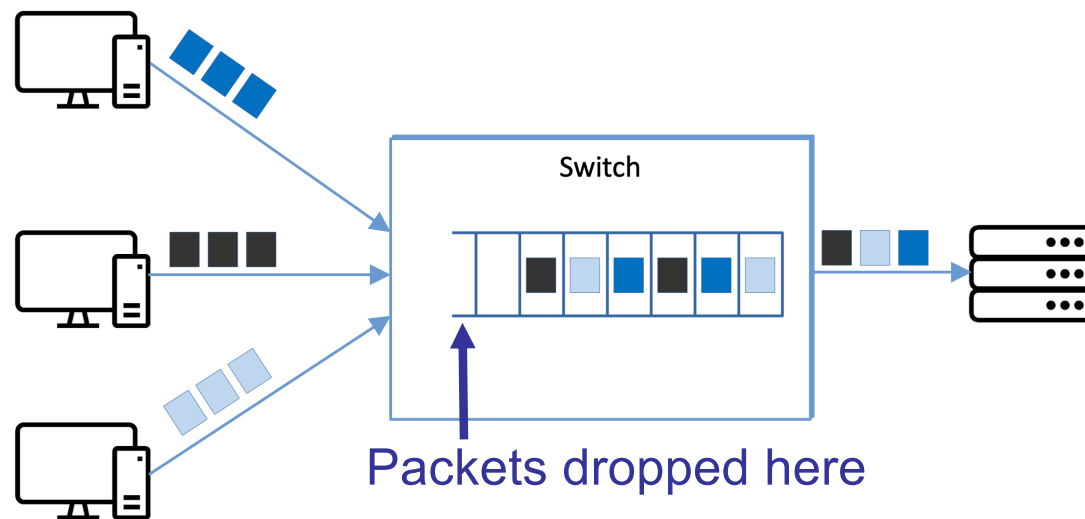
Jain's Fairness

- A reasonable definition of fair is all flows should share equally
- Given a set of flow throughputs $(x_1, x_2, \dots, x_n)$, the **fairness index** is

$$f(x_1, x_2, \dots, x_n) = \frac{(\sum_{i=1}^n x_i)^2}{n \sum_{i=1}^n x_i^2}$$

- Value ranges over $(0, 1]$
 - ♦ The closer each flow is to the average, the closer the index is to one
 - ♦ If all flows have the same throughput, index is exactly one

Congestion



- Buffer intended to absorb bursts when input rate $>$ output
- But if sending rate is persistently $>$ drain rate, queue builds
- Dropped packets represent wasted work; **goodput** $<$ throughput

For Next Time



- Read 13-13.2 in P&D
- HW solutions available on course website
- (Keep) going on the project...