Transmission Control Protocol
TCP Overview

• Specification: RFC 793 (1981)
  – Originally IP and TCP were one protocol (!)
  – Congestion Control added ca. 1998

• Connection-oriented service
  – Endpoints must **establish a connection** prior to data transfer
  – Negotiation among initiator (client), network layer, and responder (server)

• Reliable byte-stream data transfer service
  – **Same model as files** in UNIX (= sequence of bytes)
  – No message boundaries preserved
    • A single receive(...) may return:
      – Partial data from a send(...), or
      – Data from one send(...) or
      – Data from multiple sends(...)  -- See Practical Guide book
TCP Header
(Same format for all segments)

Every byte is numbered; so are some control flags

Applies to payload of this packet
Applies to data traveling in the opposite direction
Flow control: Sender may not have more than this many bytes outstanding (unacked)
TCP Header

- Source IP Address
- Destination IP Address
- zero
- protocol
- IP Length
- Source Port
- Destination Port
- Sequence Number
- Acknowledgment Number
- Data Offset
- Rsvd
- Flags
- Window Size
- Checksum
- Urgent Pointer
- Options
- Pad
- Payload

Checksum covers payload plus a 12-byte "pseudo-header"
TCP Connection Management

- Problem: initialize state variables for data transfer between "sockets"
  - Next expected sequence number
  - Flow-control window (= how much can be transmitted)
  - Next sequence number to send

- Note: data transfer in **two directions simultaneously** (full-duplex connection)
  - Most application protocols do not use this capability
    - One side transmits, then the other
  - TCP might as well have been a half-duplex protocol!
  - Exercise: find a protocol that runs over TCP that uses simultaneous transmission
TCP Connection Management

• Problem: the network may deliver old messages
  − How to distinguish between old and new messages with the same sequence number?!
    • Note: the protocol must not look at the payload! (why not?)

• Note: we need help from the IP layer to solve this problem!
  − Assume a bound on the time messages stay in the network
  − IP will drop messages rather than deliver them more than MSL seconds after they were sent
    • MSL = Maximum Segment Lifetime
  − It is theoretically possible, but very inefficient, to solve this problem if messages can be delayed arbitrarily long!
TCP Connection Management

• TCP has a large sequence number space
  (Though not as large as it used to be, due to increases in channel speeds)

• Idea: use a different part of the sequence number space for each connection between the same two endpoints

• But what does "between the same two endpoints" mean?
  - Same programs?
  - Same two "sockets"?
  - Same two machines?
TCP Connection Definition

• Connection = 4-tuple:
  (local IP address, local port, remote IP address, remote port)

• At most one connection can exist for any 4-tuple at a time

• Note: these are all different connections:
  (204.198.76.63, 35791, 128.163.140.219, 80)
  (204.198.76.63, 35793, 128.163.140.219, 80)
  (204.198.76.63, 35793, 128.163.140.219, 513)
  (204.198.76.63, 35791, 130.207.6.44, 80)