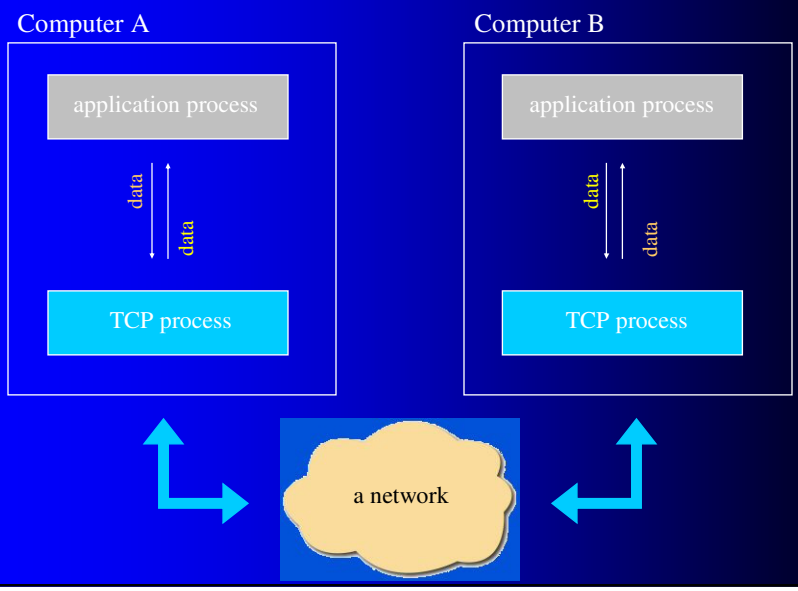
Linux Networking: tcp

David Morgan

TCP context and interfaces



TCP purposes and features

- basic data transfer
- process-to-process multiplexing
- reliability
- flow control
- connections

Transport purposes and features

	TCP	UDP
process-to-process data transfer	✓	✓
reliability	✓	*
flow control	✓	
connections	✓	

* discard, no recovery

Basic data transfer method

- sending TCP
  - "blocks out" (segments) the data stream
  - gives each block its own packet ("segment")
- receiving TCP
  - reassembles the blocks into original stream

Multiplexed "process-to-process" transfer

- processes get identifying numbers ("ports")
- IP address/TCP port pair is a local "socket"
- pair of sockets, one on each of 2 machines, associated with a unique bilateral "connection"
- packets between machines belong to a particular one of the machines' connections
- overall packet flow contains separate flow for each connection

Reliability

- problems with data
  - damaged
  - lost
  - duplicated
  - delivered out-of-order
- solution
  - Sending TCP
    - number the data
    - require acknowledgement
    - resend unacknowledged

Receiving TCP

acknowledge good data

discard bad data

reassemble by the numbers

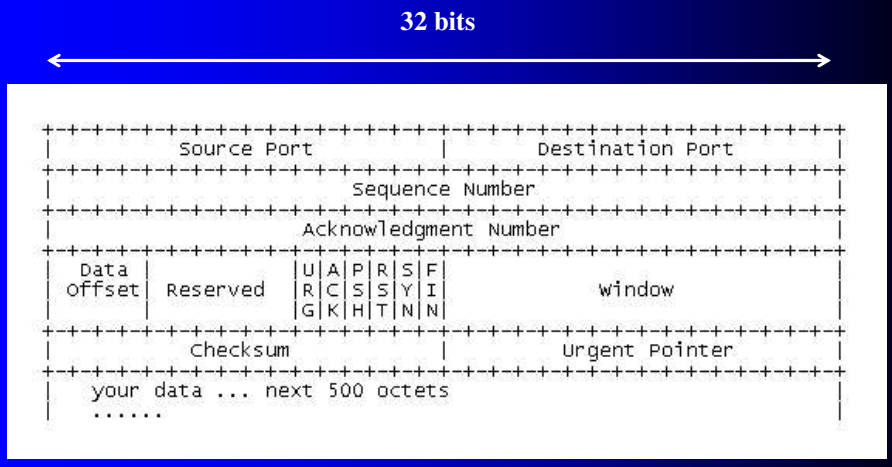
Flow control

- problem
  - sending TCP might overwhelm receiving TCP
- solution
  - constrain sender by requiring receiver's permission which data, by number range, may be transmitted

TCP connections

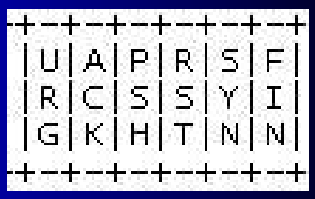
- reliability/flow control require state info
- each TCP initializes/maintains it for each data stream
- connection ends, state info data structures freed

TCP packet (segment) header

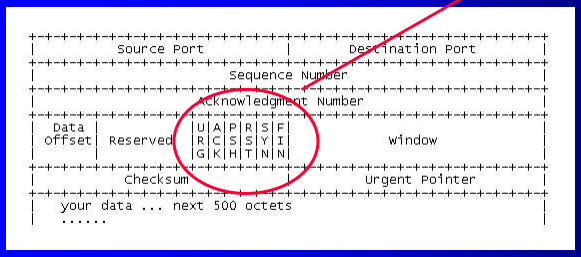


“Flag” bits

TCP flags field



TCP Header

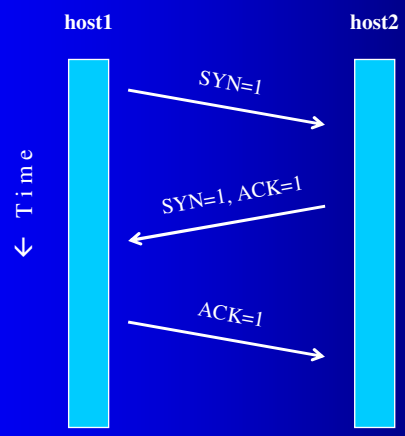


URG = urgent
ACK= acknowledgement
PSH = push
RST = reset
SYN = synchronize
FIN = finish

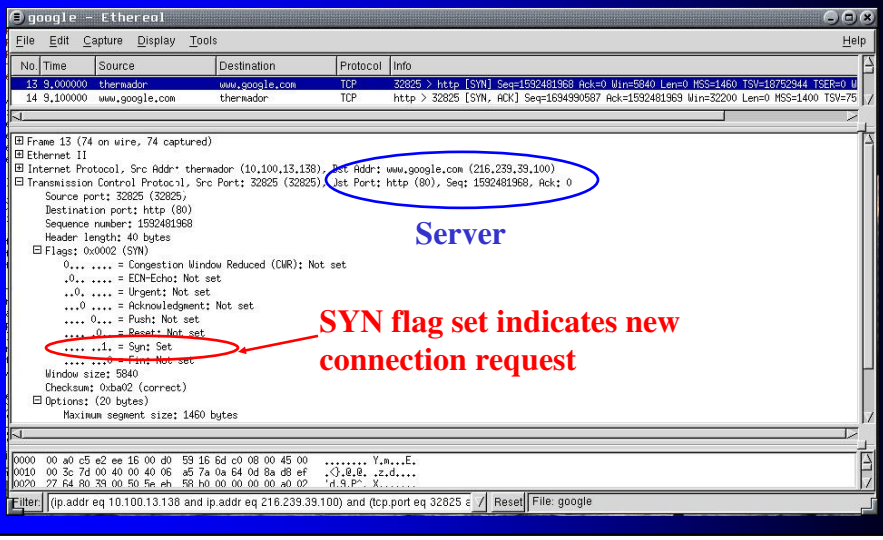
Establishing a “connection”

- client sends packet with SYN bit set
- server returns packet with SYN & ACK set
- client sends packet with ACK set
- called “3-way handshake”
- connection establishment’s signature sequence

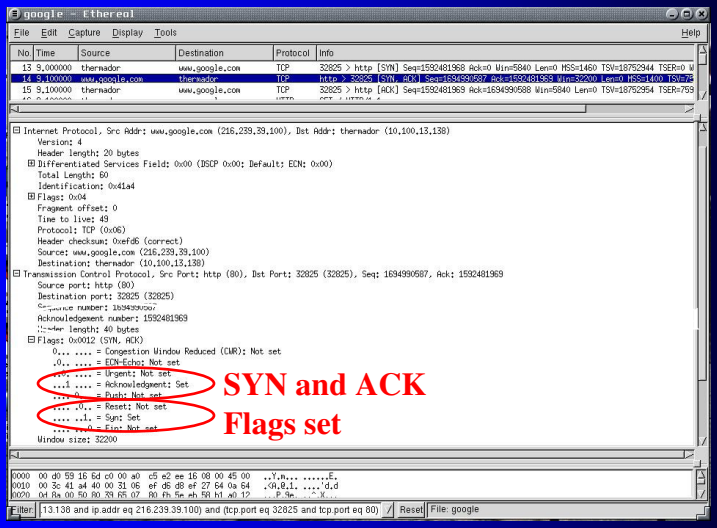
3-way handshake



TCP - SYN

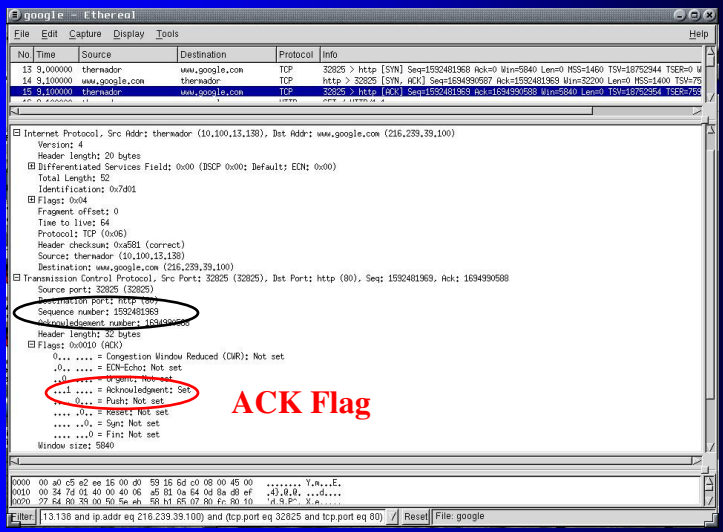


TCP - SYN/ACK



SYN and ACK Flags set

TCP ACK

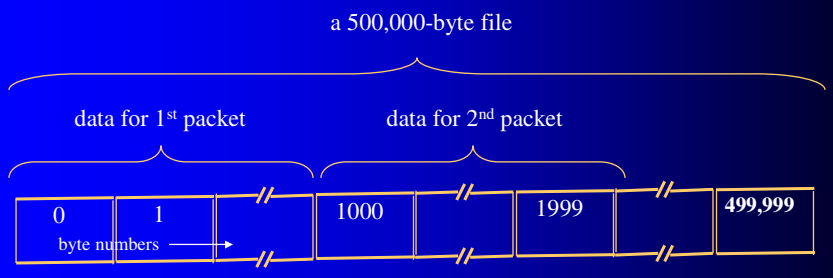


ACK Flag

TCP is “stream oriented”

- data transmitted during connection viewed as one continuous stream
- bytes are consecutively numbered
- stream segmented into packets for transmittal

File deconstruction into sequenced packets



Packet’s sequence number is the byte-stream number of the 1st data byte in the packet.

sequence number assignments:

1st packet – 0
2nd packet – 1000
3rd packet – 2000
etc

Sequence numbers

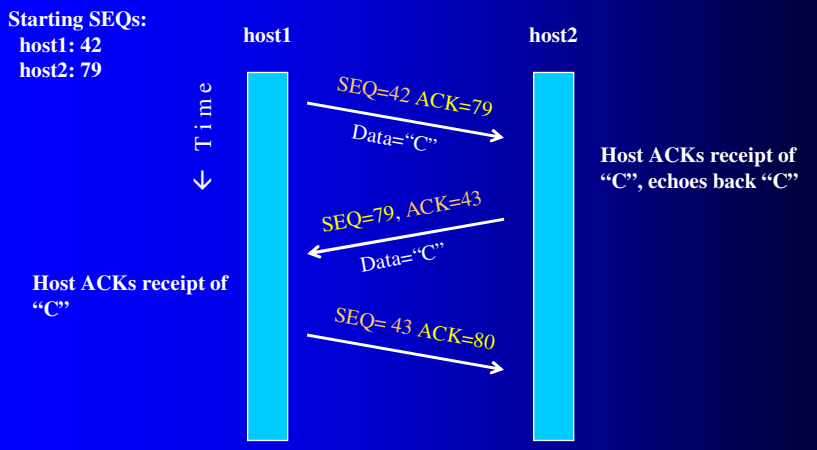
- relative to byte stream, not packet series
- initial sequence number randomly chosen
  - during connection setup handshake
  - actual byte count does not start from zero
- two number sequences
  - TCP carries 2 flows (full-duplex)
  - a separate sequence for each flow/direction

Acknowlegement number

- also byte-stream relative
- is sequence number next-expected from partner
- acknowledges receipt of all prior bytes
- therefore called “cumulative” acknowledgement
- acknowledgements are piggybacked
  - client-to-server acks ride with server-to-client data
  - server-to-client acks ride with client-to-server data

Numbering example*:

("C" keystroke in telnet)

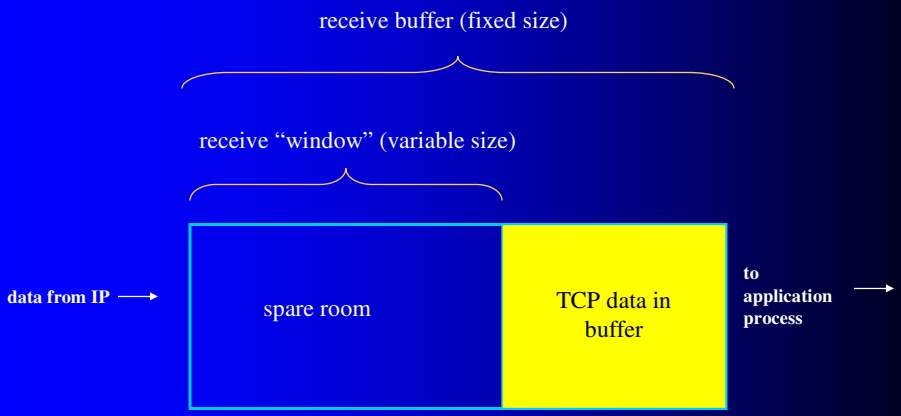


* Kurose & Ross, p. 234

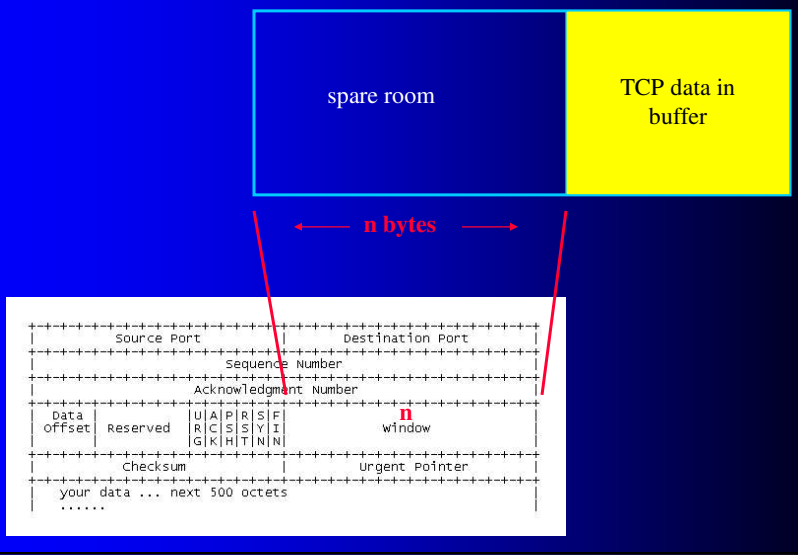
Traffic control

- flow control
  - adapt rate to partner's capacity
  - depends on spare room in partner's receive buffer
- congestion control
  - adapt rate to intervening path's capacity
  - depends on "just-about-anything"

Flow control: receive window



Partner given "willingness-to-accept"



Congestion control

- cap sent-but-unacknowledged data amount
- congestion limit can exceed flow limit
- vary the cap per perceived network congestion
  - cap more severely when packet loss rate rises
  - relax cap when it drops

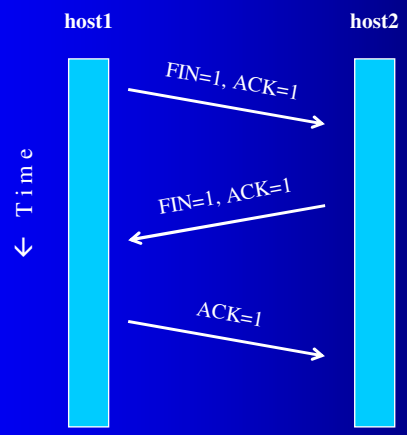
TCP Socket

- connection defined by socket pair
  - combination of IP address and port = socket
- client IP = 10.100.13.138
- client Port = 32825
  - client Socket = 10.100.13.138:32825
- server IP = 216.239.39.100
- server Port = 80 (http Default)
  - Server Socket = 216.239.39.100:80

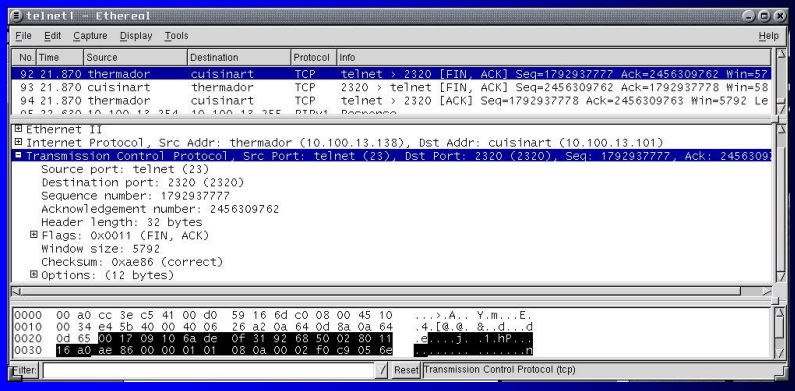
well-known TCP ports

- 21 - ftp
- 22 - ssh
- 23 - telnet
- 25 - smtp (Simple Mail Transport Protocol)
- 80 - http
- 110 - pop3 (Post Office Protocol)
- 123 - Network Time Protocol

TCP connection teardown



FIN/ACK



Biblio

- Computer Networking, Kurose & Ross, Addison-Wesley, 2003; Chapter 3 “Transport Layer”
- “Telnet Protocol Specification,” RFC 854, 1983