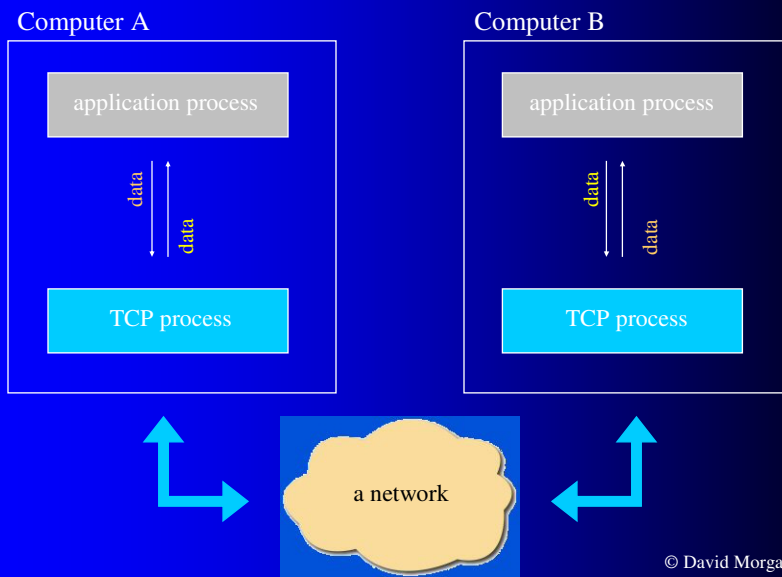


Linux Networking: tcp

David Morgan

© David Morgan 2003-12

TCP context and interfaces



TCP purposes and features

- Basic data transfer
- Process-to-process multiplexing
- Reliability
- Flow control
- Connections

© David Morgan 2003-12

Transport purposes and features

	<u>TCP</u>	<u>UDP</u>
● process-to-process data transfer	✓	✓
● reliability	✓	*
● flow control	✓	
● connections	✓	* discard, no recovery

© David Morgan 2003-12

Basic data transfer method

- Sending TCP
 - “blocks” (segments) the data stream
 - gives each block its own packet (“segment”)
- Receiving TCP
 - reassembles the blocks into original stream

© David Morgan 2003-12

Multiplexed “process-to-process” transfer

- processes given 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

© David Morgan 2003-12

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

© David Morgan 2003-12

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

© David Morgan 2003-12

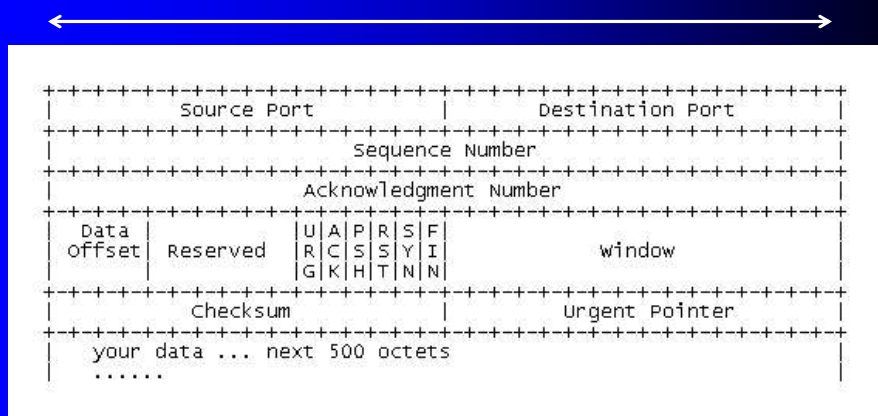
TCP connections

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

© David Morgan 2003-12

TCP packet (segment) header

32 bits



© David Morgan 2003-12

“Flag” bits

TCP flags field

U	A	P	R	S	F
R	C	S	S	Y	I
G	K	H	T	N	N

TCP Header

Source Port		Destination Port						
Sequence Number								
Acknowledgment Number								
Data offset	Reserved	U	A	P	R	S	F	Window
		R	C	S	S	Y	I	
		G	K	H	T	N	N	
Checksum			Urgent Pointer					
your data ... next 500 octets								
.....								

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

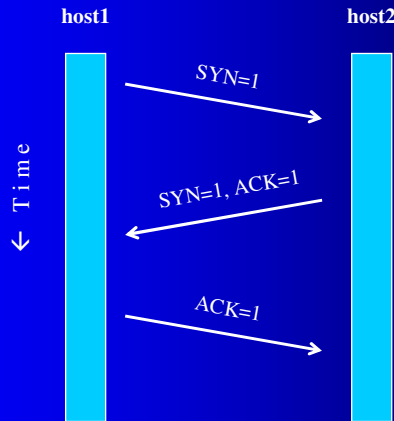
© David Morgan 2003-12

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

© David Morgan 2003-12

3-way handshake



© David Morgan 2003-12

TCP - SYN

google - Ethereal

No.	Time	Source	Destination	Protocol	Info
13	9.000000	thermador	www.google.com	TCP	32825 > http [SYN] Seq=1592481968 Ack=0 Win=6940 Len=0 MSS=1460 TSV=18763944 TSEB=0
14	9.100000	www.google.com	thermador	TCP	http > 32825 [SYN, ACK] Seq=1694990987 Ack=1592481969 Win=32200 Len=0 MSS=1400 TSV=75

Frame 13 (74 on wire, 74 captured)

- Ethernet II
- Internet Protocol, Src Addr: thermador (10.100.13.138), Dest Addr: www.google.com (216.239.39.100)
- Transmission Control Protocol, Src Port: 32825 (32825), Dest Port: http (80), Seq: 1592481968, Ack: 0
 - Source port: 32825 (32825)
 - Destination port: http (80)
 - Sequence number: 1592481968
 - Header length: 40 bytes
 - Flags: 0x0002 (SYN)
 - 0... .. = Congestion Window Reduced (CWR): Not set
 - ..0. = ECN-Echo: Not set
 - ...0. = Urgent: Not set
 - ...0 = Acknowledgment: Not set
 -0... = Push: Not set
 -0 = Reset: Not set
 -1. = Syn: Set
 -0.. = Fin: Not set
 - Window size: 6940
 - Checksum: 0xba02 (correct)
 - Options: (20 bytes)
 - Maximum segment size: 1460 bytes

Server

SYN flag set indicates new connection request

0000 00 a0 c5 e2 ee 16 00 d0 59 16 6d c0 08 00 45 00Y.n...E.
0010 00 36 7d 00 40 00 40 06 a5 7a 0a e4 0d 8a d8 ef ...3.0.2..z.d....
0070 27 f4 80 53 00 5b ea eb 58 10 00 00 00 ad 02 ..f.S.P..M.....

Filter: [ip.addr eq 10.100.13.138 and ip.addr eq 216.239.39.100] and (tcp.port eq 32825) / Reset File: google

TCP - SYN/ACK

The image shows a Wireshark packet capture of a SYN/ACK packet. The packet list pane shows a packet from 13.0.0.0000 to 15.0.100000. The packet details pane shows the following information:

- Internet Protocol, Src Addr: www.google.com (216.239.39.100), Dst Addr: thernador (10.100.13.138)
- Version: 4
- Header length: 20 bytes
- Differentiated Services Field: 0x00 (DSCP 0x00; Default; ECN: 0x00)
- Total Length: 60
- Identification: 0x41ad
- Flags: 0x04
- Fragment offset: 0
- Time to live: 49
- Protocol: TCP (0x06)
- Header checksum: 0xefdb (correct)
- Source: www.google.com (216.239.39.100)
- Destination: thernador (10.100.13.138)
- Transmission Control Protocol, Src Port: http (80), Dst Port: 32825 (32825), Seq: 163499567, Ack: 1592481969
- Source port: http (80)
- Destination port: 32825 (32825)
- Sequence number: 163499567
- Acknowledgment number: 1592481969
- Header length: 40 bytes
- Flags: 0x0012 (SYN, ACK)
- 0... .. = Congestion Window Reduced (CWR): Not set
- 0... .. = ECN-Echo: Not set
- 0... .. = Urgent: Not set
- ...1... .. = Acknowledgment: Set
- = Push: Not set
- = Reset: Not set
- = SYN: Set
- = FIN: Not set
- Window size: 32900

Red circles highlight the 'SYN' and 'ACK' flags in the 'Flags' field. Red text next to the circles reads 'SYN and ACK Flags set'.

© David Morgan 2003-12

TCP ACK

The image shows a Wireshark packet capture of an ACK packet. The packet list pane shows a packet from 14.0.100000 to 15.0.100000. The packet details pane shows the following information:

- Internet Protocol, Src Addr: thernador (10.100.13.138), Dst Addr: www.google.com (216.239.39.100)
- Version: 4
- Header length: 20 bytes
- Differentiated Services Field: 0x00 (DSCP 0x00; Default; ECN: 0x00)
- Total Length: 52
- Identification: 0x7404
- Flags: 0x04
- Fragment offset: 0
- Time to live: 64
- Protocol: TCP (0x06)
- Header checksum: 0xa281 (correct)
- Source: thernador (10.100.13.138)
- Destination: www.google.com (216.239.39.100)
- Transmission Control Protocol, Src Port: http (80), Dst Port: 32825 (32825), Seq: 1592481969, Ack: 163499568
- Source port: http (80)
- Destination port: 32825 (32825)
- Sequence number: 1592481969
- Acknowledgment number: 163499568
- Header length: 32 bytes
- Flags: 0x0010 (ACK)
- 0... .. = Congestion Window Reduced (CWR): Not set
- 0... .. = ECN-Echo: Not set
- 0... .. = Urgent: Not set
- ...1... .. = Acknowledgment: Set
- = Push: Not set
- = Reset: Not set
- = SYN: Not set
- = FIN: Not set
- Window size: 5840

Red circles highlight the 'ACK' flag in the 'Flags' field. Red text next to the circles reads 'ACK Flag'.

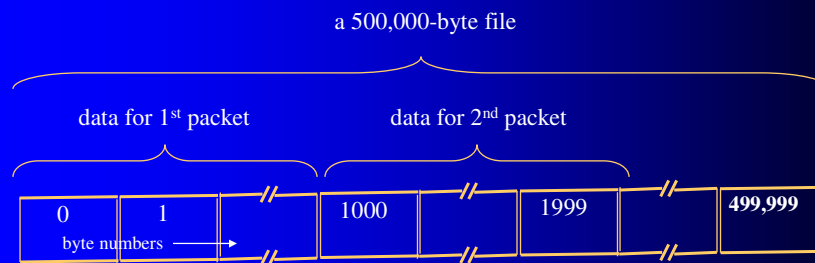
© David Morgan 2003-12

TCP is “stream oriented”

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

© David Morgan 2003-12

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

© David Morgan 2003-12

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

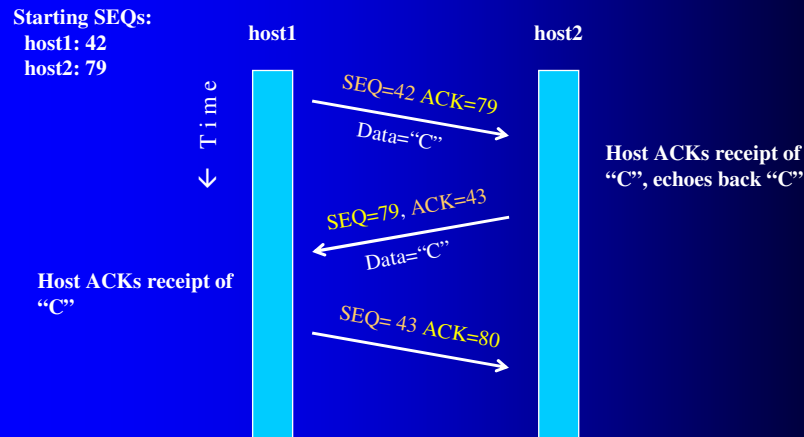
© David Morgan 2003-12

Acknowledgement 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

© David Morgan 2003-12

Numbering example*: ("C" keystroke in telnet)



* Kurose & Ross, p. 234

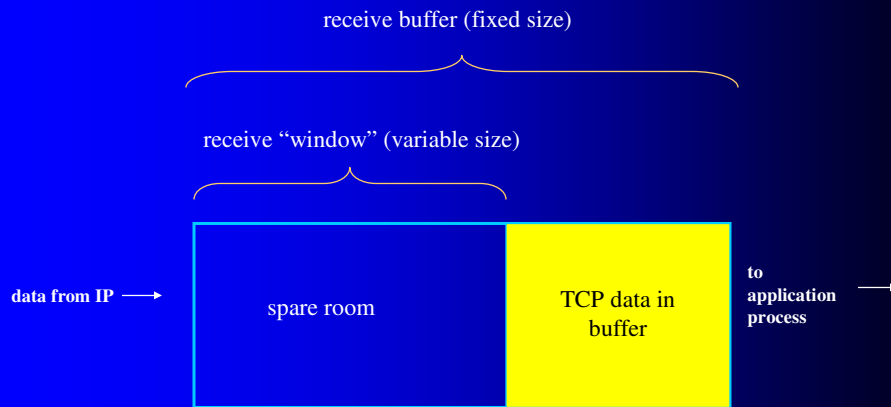
© David Morgan 2003-12

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"

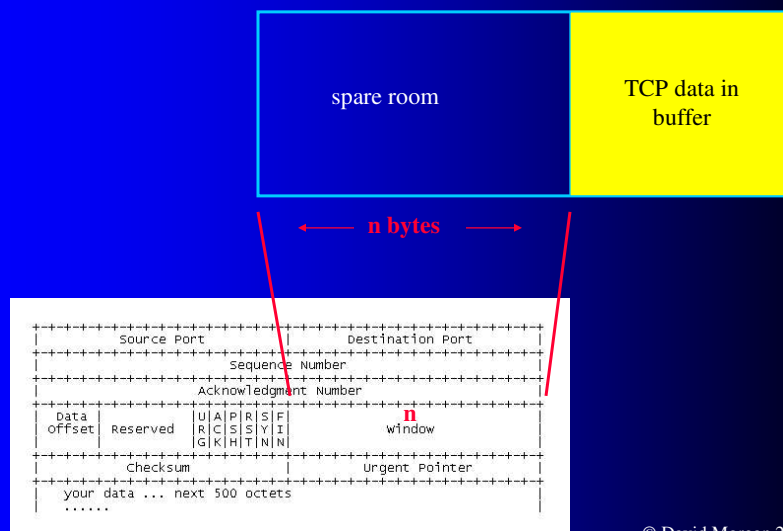
© David Morgan 2003-12

Flow control: receive window



© David Morgan 2003-12

Partner given "willingness-to-accept"



© David Morgan 2003-12

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

© David Morgan 2003-12

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

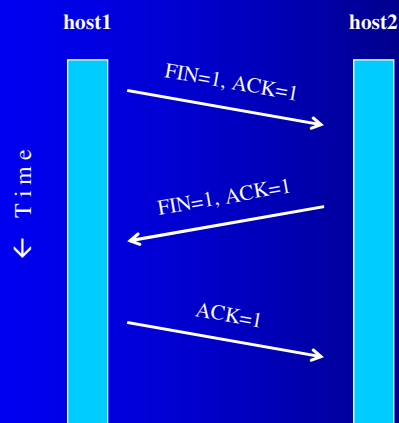
© David Morgan 2003-12

well-known TCP ports

- 21 - FTP Control
- 20 - FTP Data
- 23 - Telnet
- 25 - SMTP (Simple Mail Transport Protocol)
- 80 - HTTP
- 110 - POP3
- 119 - Network News Transfer Protocol

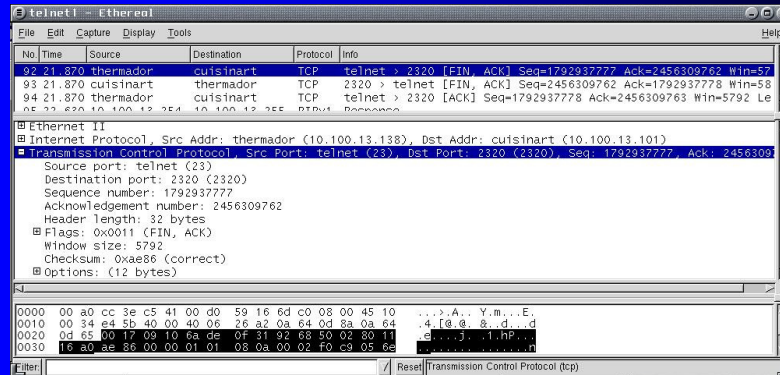
© David Morgan 2003-12

TCP connection teardown



© David Morgan 2003-12

FIN/ACK



© David Morgan 2003-12

Biblio

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

© David Morgan 2003-12