Lecture 13: Transport Layer Flow and Congestion Control

COMP 411, Fall 2022

Victoria Manfredi

Acknowledgements: materials adapted from Computer Networking: A Top Down Approach 7th edition: ©1996-2016, J.F Kurose and K.W. Ross, All Rights Reserved as well as from slides by Abraham Matta at Boston University, and some material from Computer Networks by Tannenbaum and Wetherall.

TCP

RELIABLE DATA TRANSFER

Duplicate ACKs

Time-out period often relatively long

- long delay before resending lost packet

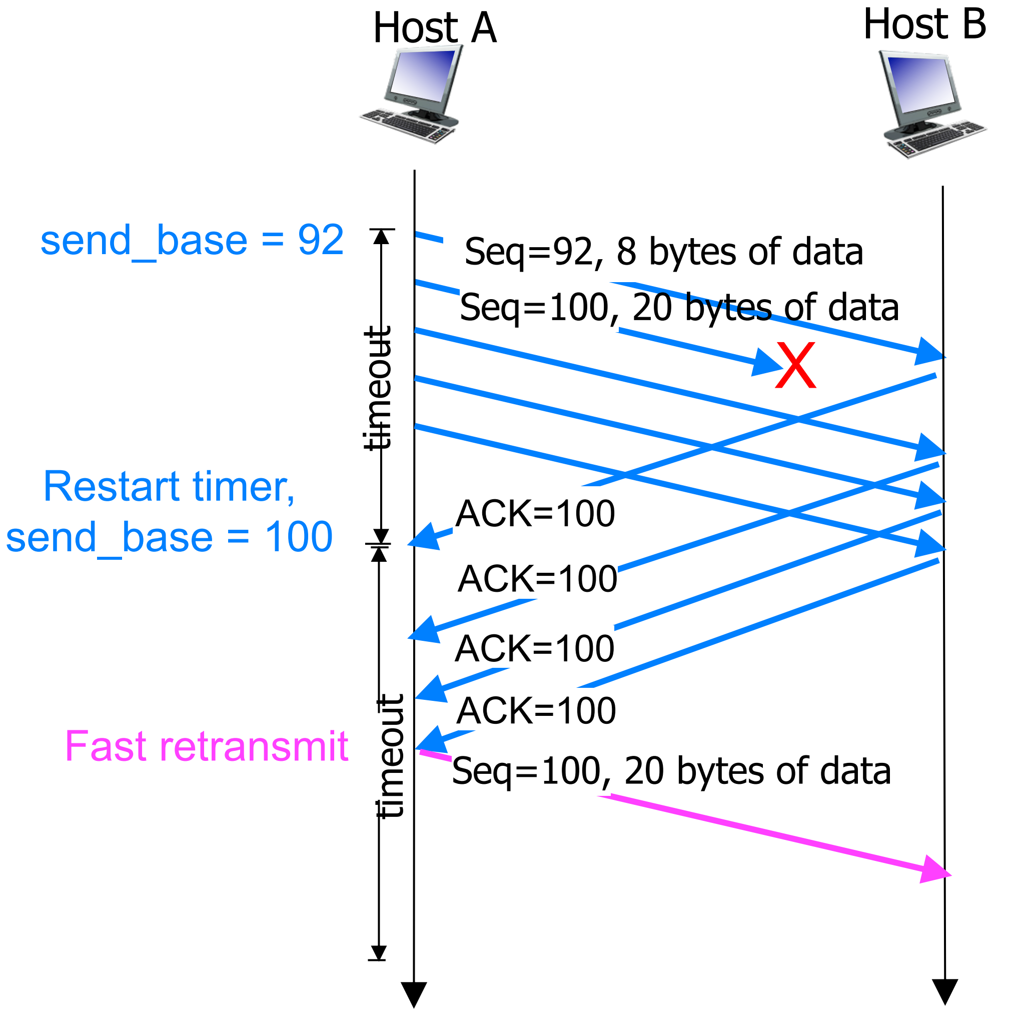
Duplicate ACKs indicate isolated loss

- rather than congestion causing many losses
  - sender often sends many segments back-to-back
  - if segment is lost, likely many duplicate ACKs
  - ACKs being received indicates some packets received at destination since ACK sent for every packet: so not congestion

TCP fast retransmit

- if sender receives 3 ACKs for same data (triple duplicate ACKs)
  - resend unacked segment with smallest seq #
- Q: why 3?
  - pkts may just have been reordered otherwise
  - likely that unacked segment lost, so don’t wait for timeout

TCP fast retransmit



fast retransmit after sender receipt of triple duplicate ACK

TCP

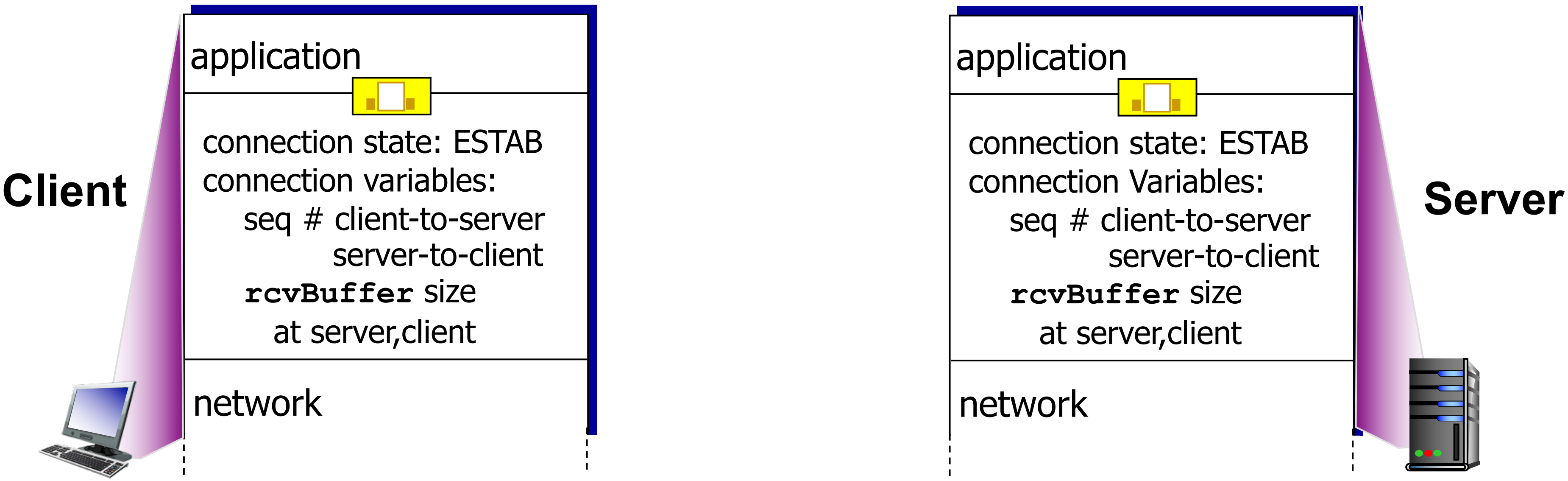
CONNECTION MANAGEMENT

Connection Management

Before exchanging data, sender/receiver handshake

– establish connection and connection parameters

– tear down connection when done



Client

application

connection state: ESTAB
connection variables:
seq # client-to-server
server-to-client
rcvBuffer size
at server,client

network

Server

application

connection state: ESTAB
connection Variables:
seq # client-to-server
server-to-client
rcvBuffer size
at server,client

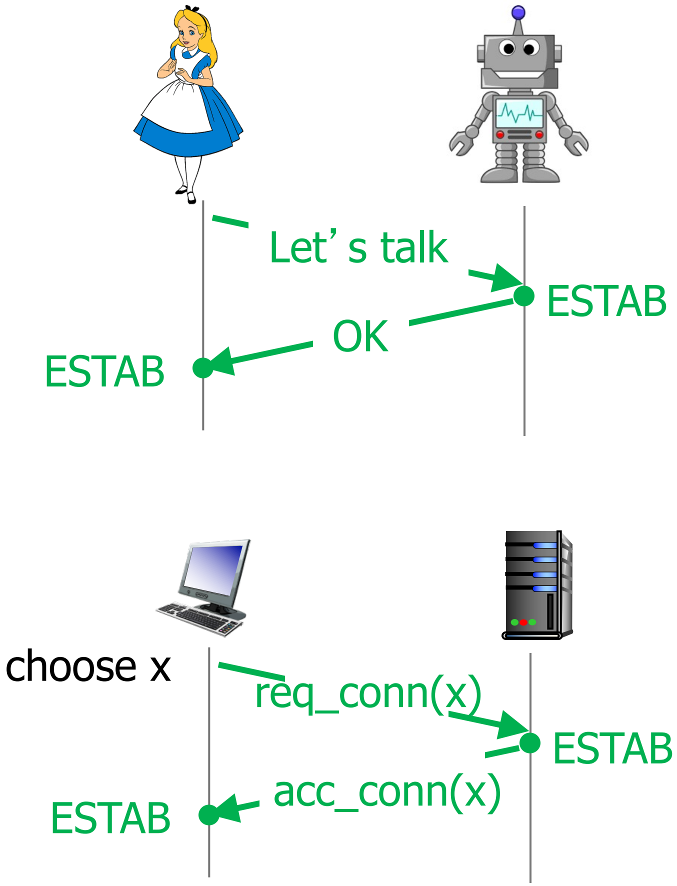
network

```
sock = sock.connect((host, port))
```

```
conn, addr = server_sock.accept()
```

Agreeing to establish a connection

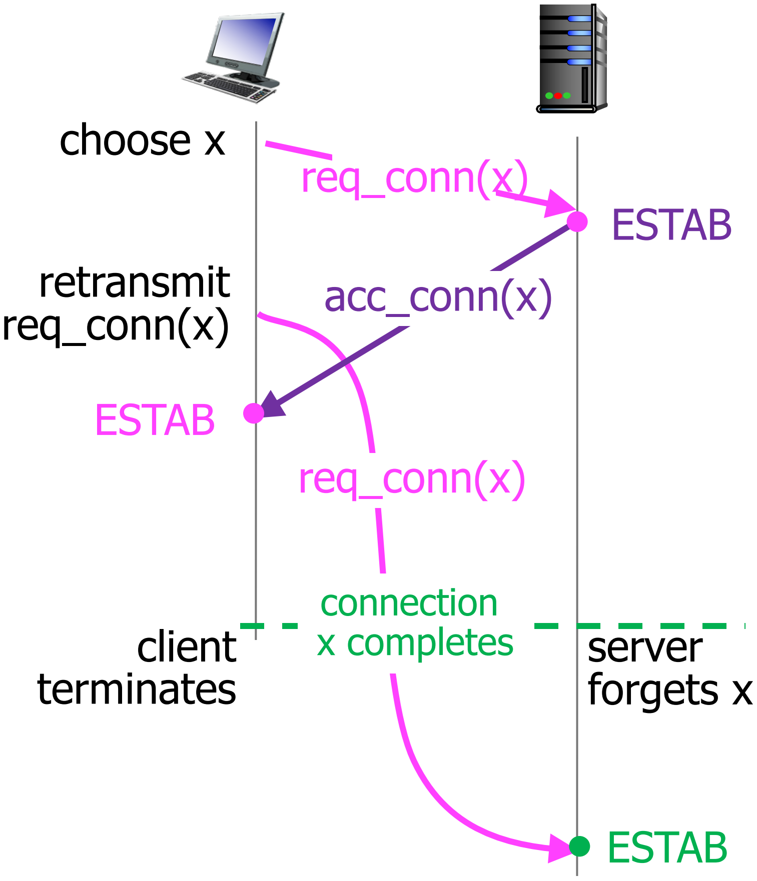
2-way handshake:



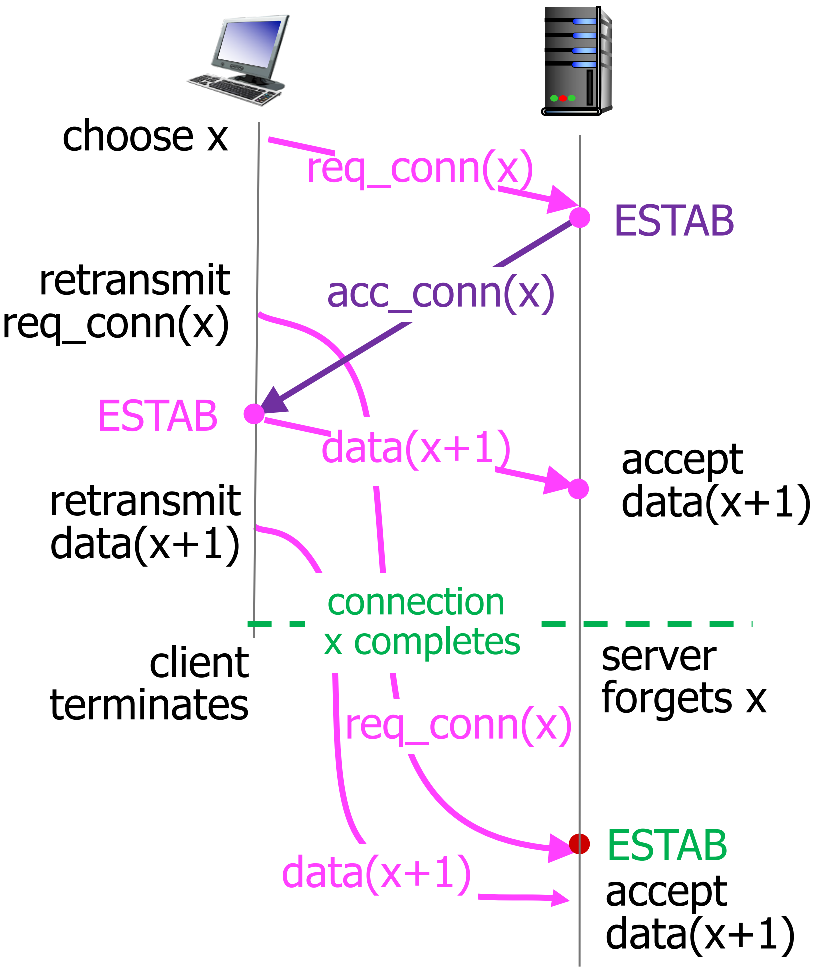
Q: will 2-way handshake always work in network?

- variable delays
- retransmitted messages
  - e.g. req_conn(x)) due to message loss
- message reordering
- can’t see other side

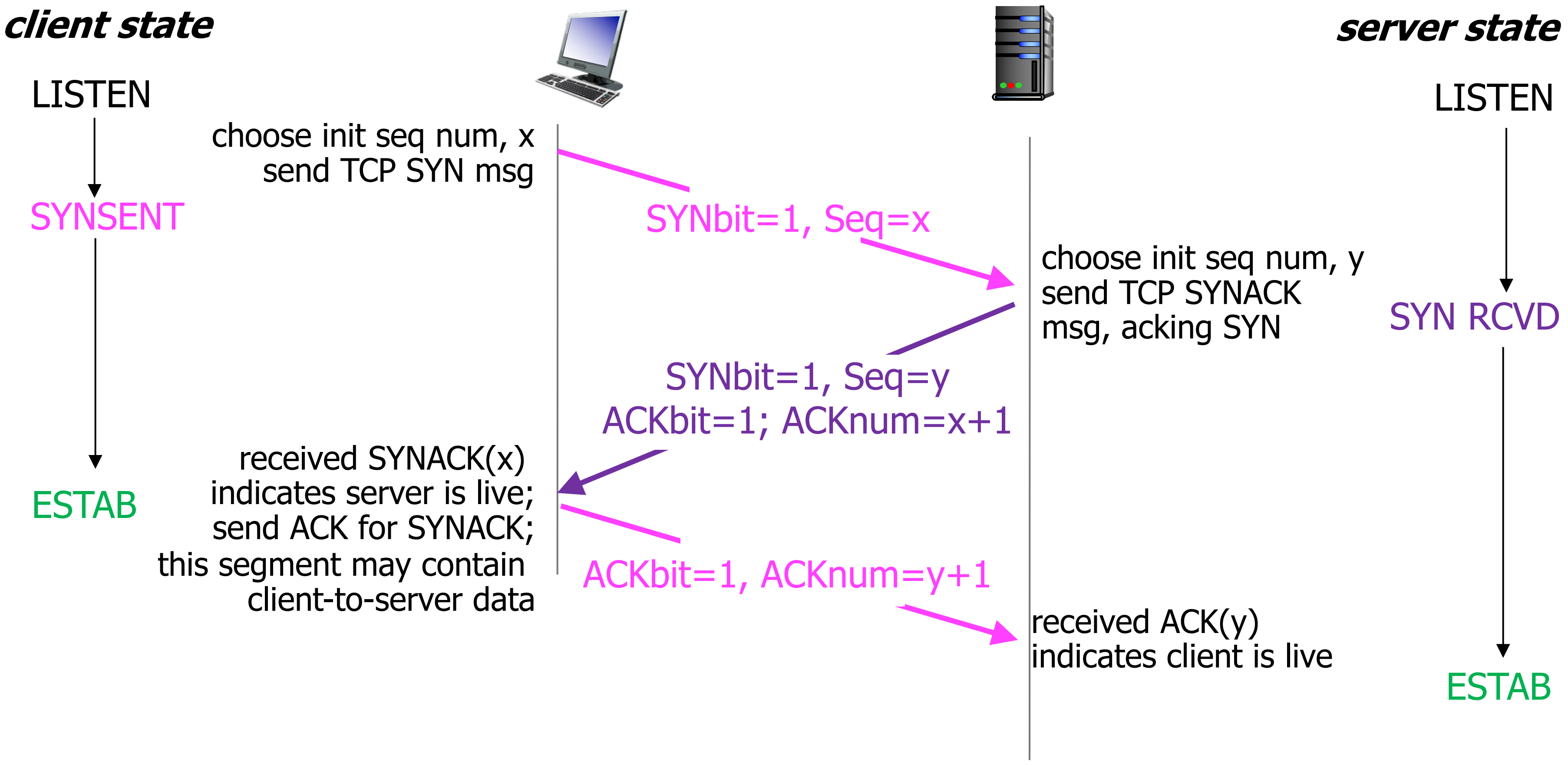
2-way handshake failure scenarios



half open connection!
(no client!)



TCP 3-way handshake



client state

LISTEN

choose init seq num, x
send TCP SYN msg

SYNSENT

SYNbit=1, Seq=x

server state

LISTEN

choose init seq num, y
send TCP SYNACK
msg, acking SYN

SYN RCVD

SYNbit=1, Seq=y
ACKbit=1; ACKnum=x+1

received SYNACK(x)
indicates server is live;
send ACK for SYNACK;
this segment may contain
client-to-server data

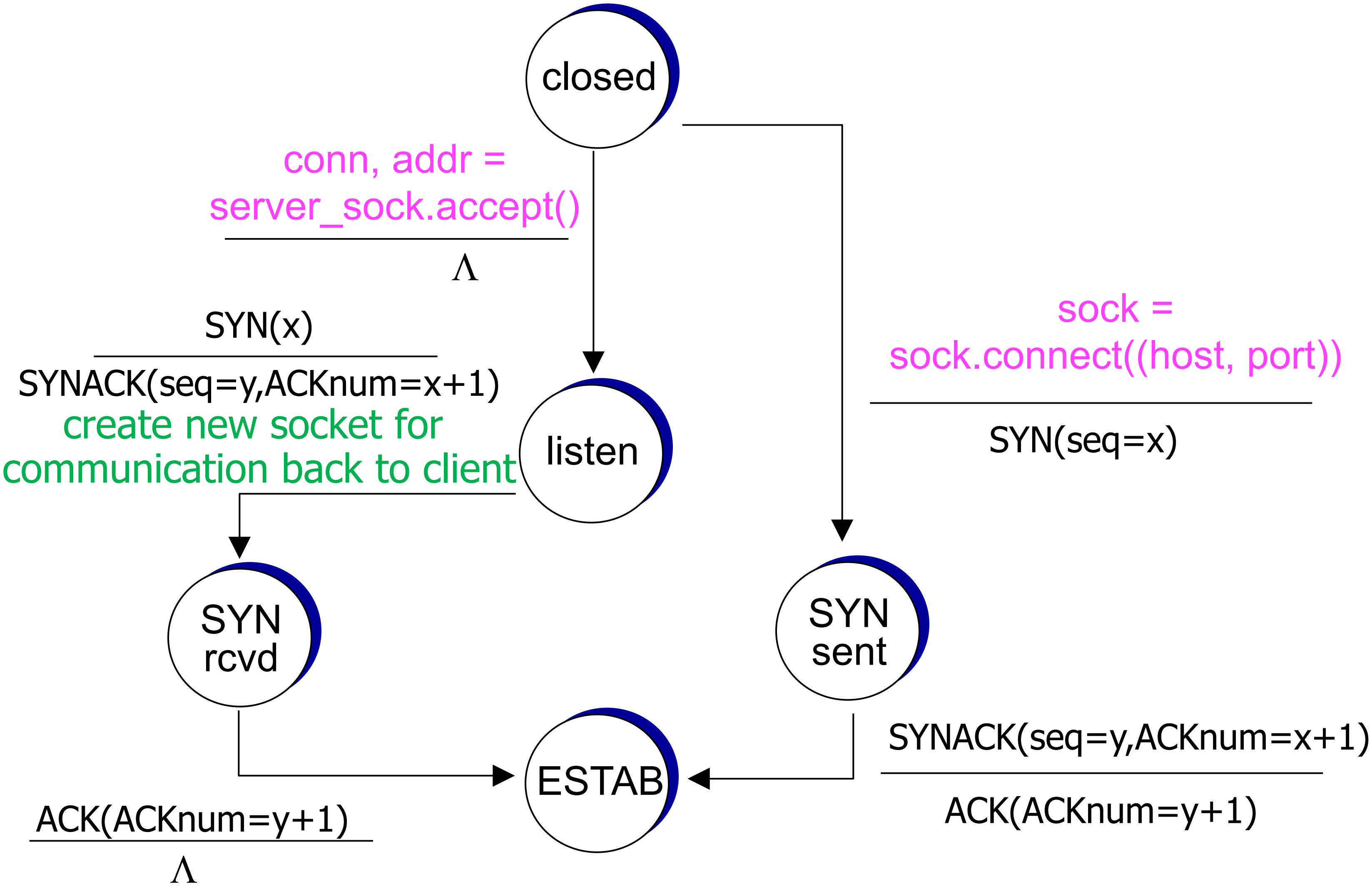
ESTAB

ACKbit=1, ACKnum=y+1

received ACK(y)
indicates client is live

ESTAB

TCP 3-way handshake: FSM



closed

conn, addr = server_sock.accept()

Λ

SYN(x)

SYNACK(seq=y,ACKnum=x+1)
create new socket for communication back to client

listen

sock = sock.connect((host, port))

SYN(seq=x)

SYN rcvd

SYN sent

ESTAB

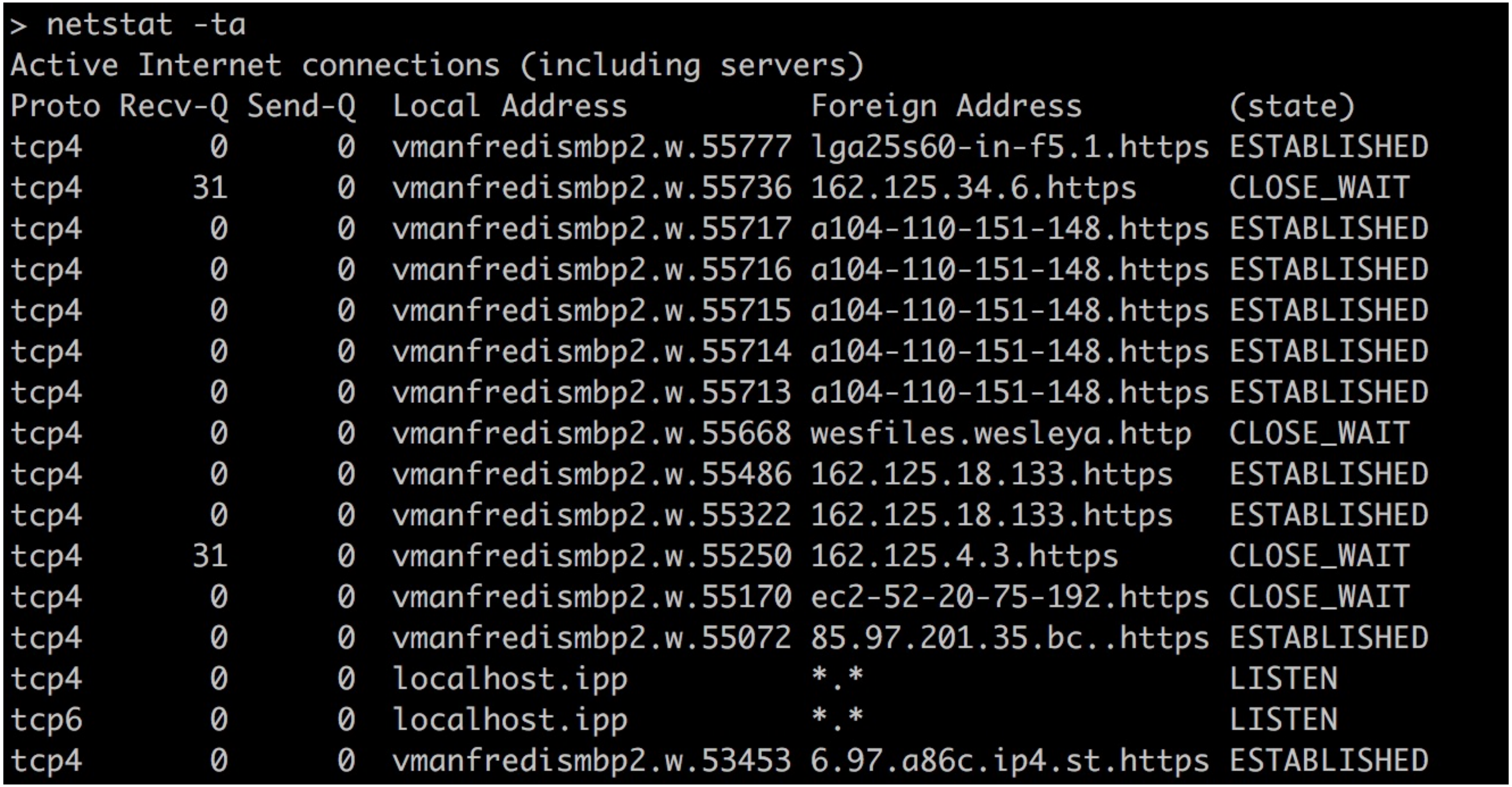
ACK(ACKnum=y+1)

Λ

SYNACK(seq=y,ACKnum=x+1)

ACK(ACKnum=y+1)

Look at the state of tcp connections

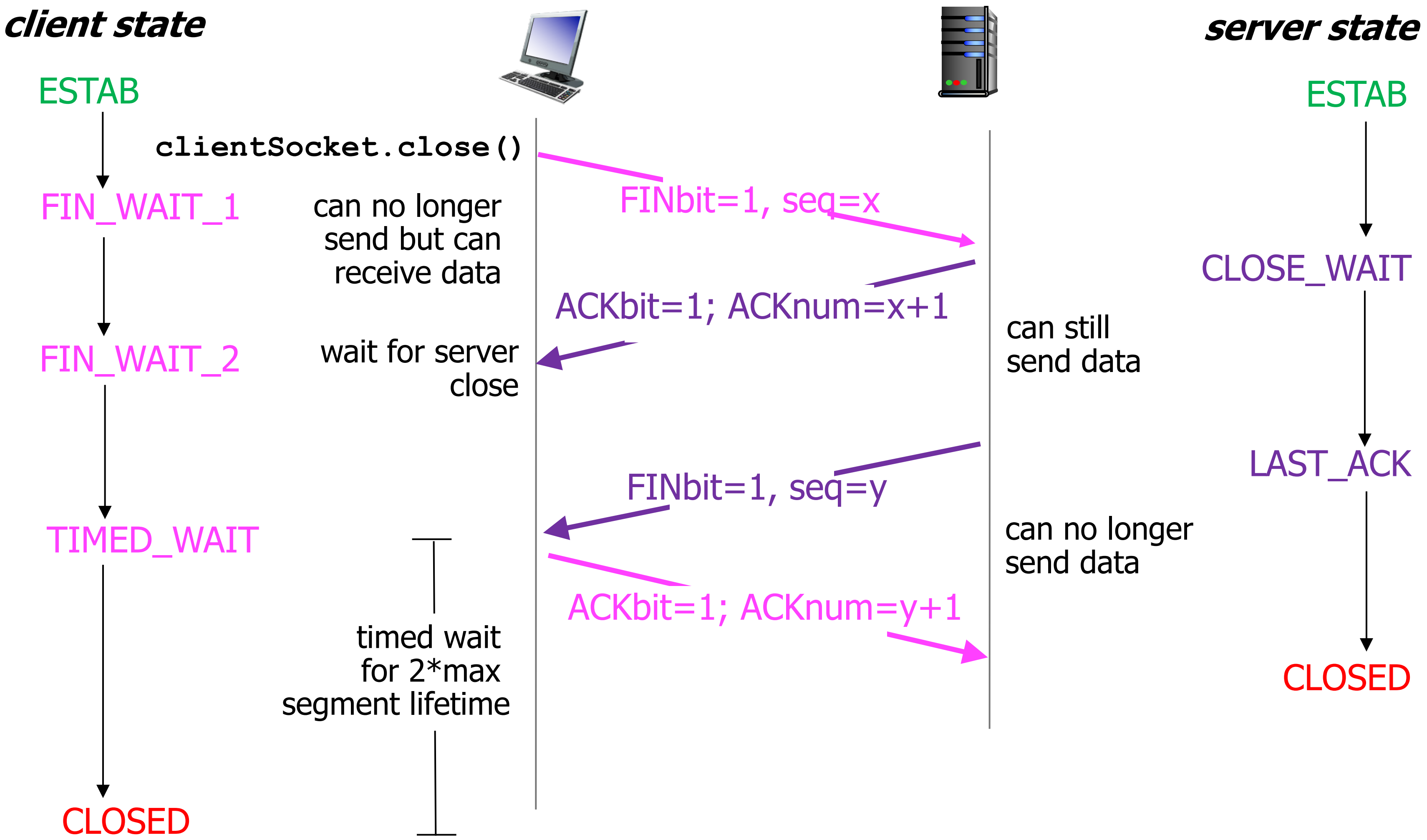


```
> netstat -ta
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp4       0      0  vmanfredismbp2.w.55777 lga25s60-in-f5.1.https ESTABLISHED
tcp4      31      0  vmanfredismbp2.w.55736 162.125.34.6.https     CLOSE_WAIT
tcp4       0      0  vmanfredismbp2.w.55717 a104-110-151-148.https ESTABLISHED
tcp4       0      0  vmanfredismbp2.w.55716 a104-110-151-148.https ESTABLISHED
tcp4       0      0  vmanfredismbp2.w.55715 a104-110-151-148.https ESTABLISHED
tcp4       0      0  vmanfredismbp2.w.55714 a104-110-151-148.https ESTABLISHED
tcp4       0      0  vmanfredismbp2.w.55713 a104-110-151-148.https ESTABLISHED
tcp4       0      0  vmanfredismbp2.w.55668 wesfiles.wesleya.http  CLOSE_WAIT
tcp4       0      0  vmanfredismbp2.w.55486 162.125.18.133.https   ESTABLISHED
tcp4       0      0  vmanfredismbp2.w.55322 162.125.18.133.https   ESTABLISHED
tcp4      31      0  vmanfredismbp2.w.55250 162.125.4.3.https      CLOSE_WAIT
tcp4       0      0  vmanfredismbp2.w.55170 ec2-52-20-75-192.https CLOSE_WAIT
tcp4       0      0  vmanfredismbp2.w.55072 85.97.201.35.bc..https ESTABLISHED
tcp4       0      0  localhost.ipp          *.*                    LISTEN
tcp6       0      0  localhost.ipp          *.*                    LISTEN
tcp4       0      0  vmanfredismbp2.w.53453 6.97.a86c.ip4.st.https ESTABLISHED
```

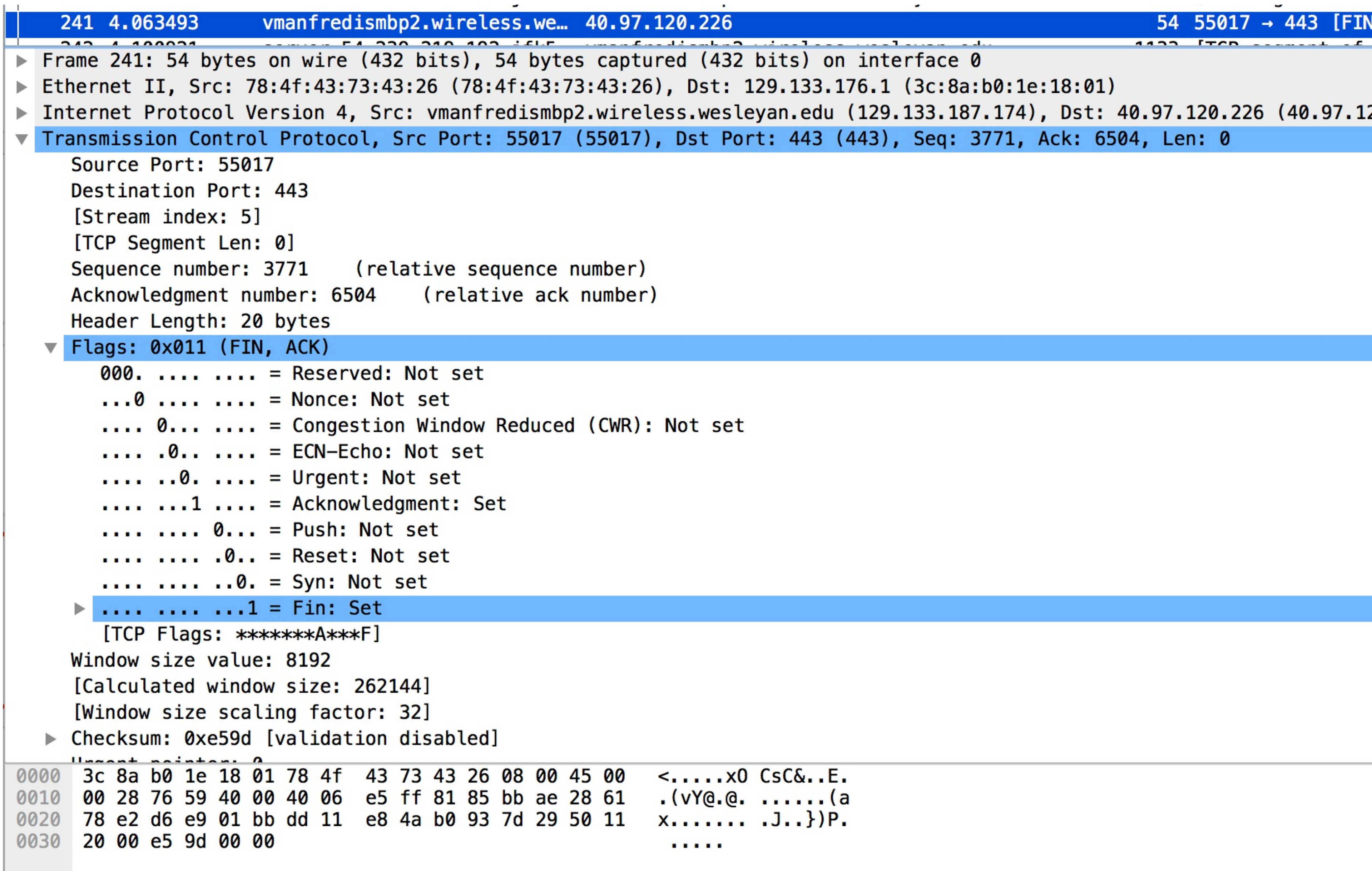
TCP: politely closing a connection

Client, server each sends TCP segment with FIN bit = 1

- respond to received FIN with ACK (ACK can be combined with own FIN)



FIN segment in Wireshark



```
241 4.063493     vmanfredismbp2.wireless.we… 40.97.120.226                                54 55017 → 443 [FIN
▶ Frame 241: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
▶ Ethernet II, Src: 78:4f:43:73:43:26 (78:4f:43:73:43:26), Dst: 129.133.176.1 (3c:8a:b0:1e:18:01)
▶ Internet Protocol Version 4, Src: vmanfredismbp2.wireless.wesleyan.edu (129.133.187.174), Dst: 40.97.120.226 (40.97.12
▼ Transmission Control Protocol, Src Port: 55017 (55017), Dst Port: 443 (443), Seq: 3771, Ack: 6504, Len: 0
    Source Port: 55017
    Destination Port: 443
    [Stream index: 5]
    [TCP Segment Len: 0]
    Sequence number: 3771    (relative sequence number)
    Acknowledgment number: 6504    (relative ack number)
    Header Length: 20 bytes
  ▼ Flags: 0x011 (FIN, ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
    ▶ .... .... ...1 = Fin: Set
      [TCP Flags: *******A***F]
    Window size value: 8192
    [Calculated window size: 262144]
    [Window size scaling factor: 32]
  ▶ Checksum: 0xe59d [validation disabled]
    Urgent pointer: 0
0000  3c 8a b0 1e 18 01 78 4f  43 73 43 26 08 00 45 00   <.....xO CsC&..E.
0010  00 28 76 59 40 00 40 06  e5 ff 81 85 bb ae 28 61   .(vY@.@. ......(a
0020  78 e2 d6 e9 01 bb dd 11  e8 4a b0 93 7d 29 50 11   x....... .J..})P.
0030  20 00 e5 9d 00 00                                   .....
```