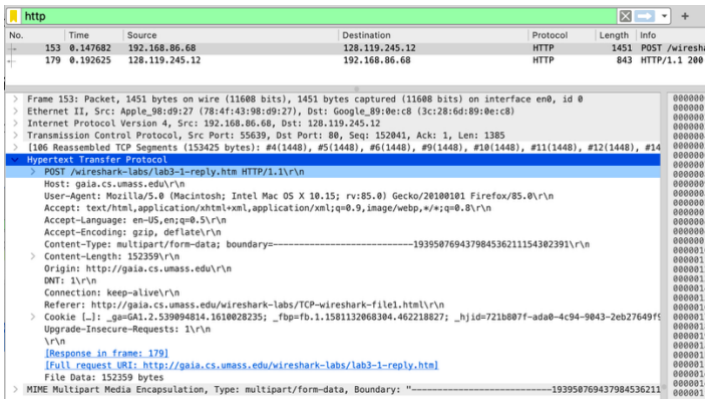
Catalina Cisneros

Wireshark Lab: TCP v9



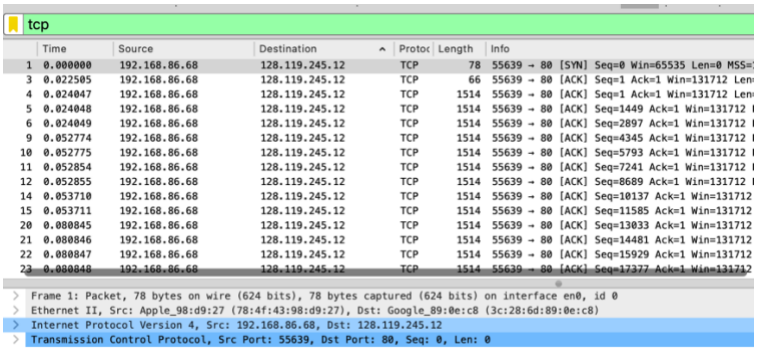
1. Capturing a bulk TCP transfer from your computer to a remote server

→ Using the trace pcap file *tcp-wireshark-trace1-1*

2. A first look at the captured trace

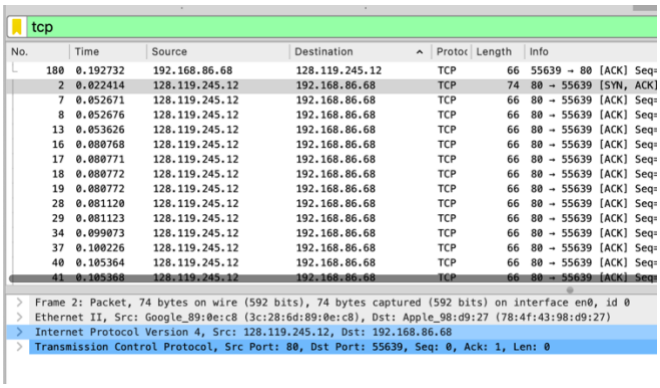
1. What is the IP address and TCP port number used by the client computer (source) that is transferring the alice.txt file to gaia.cs.umass.edu? To answer this question, it's probably easiest to select an HTTP message and explore the details of the TCP packet used to carry this HTTP message, using the "details of the selected packet header window" (refer to Figure 2 in the "Getting Started with Wireshark" Lab if you're uncertain about the Wireshark windows).

   → Client IP Address: 192.168.86.44
   → Client Port: 55639
   → Server IP Address: 128.119.245.12
   → Server Port: 80 (HTTP)



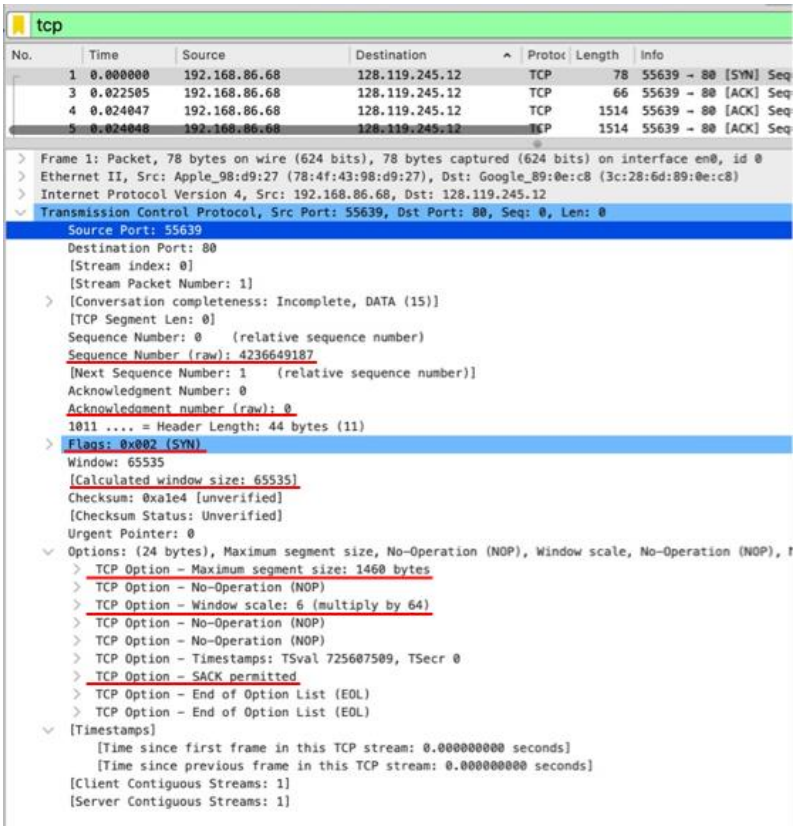
2. What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

   → Server (gaia.cs.umass.edu) IP: 128.119.245.12
   → Server Port: 80
   → Client IP: 192.168.86.44
   → Client Port: 56939



3. TCP Basics

3. What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? (Note: this is the "raw" sequence number carried in the TCP segment itself; it is NOT the packet # in the "No." column in the Wireshark window. Remember there is no such thing as a "packet number" in TCP or UDP; as you know, there are sequence numbers in TCP and that's what we're after here. Also note that this is not the relative sequence number with respect to the starting sequence number of this TCP session.). What is it in this TCP segment that identifies the segment as a SYN segment? Will the TCP receiver in this session be able to use Selective Acknowledgments (allowing TCP to function a bit more like a "selective repeat" receiver, see section 3.4.5 in the text)?
   → Sequence Number: 0 (raw = 4236649187)
   → Acknowledgment Number: 0
   → Flags: SYN
   → Window Size: 65535
   → MSS: 1460
   → Window Scale: present
   → SACK Permitted: Yes
   → Timestamps: present



4. Questions

    a. What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN?

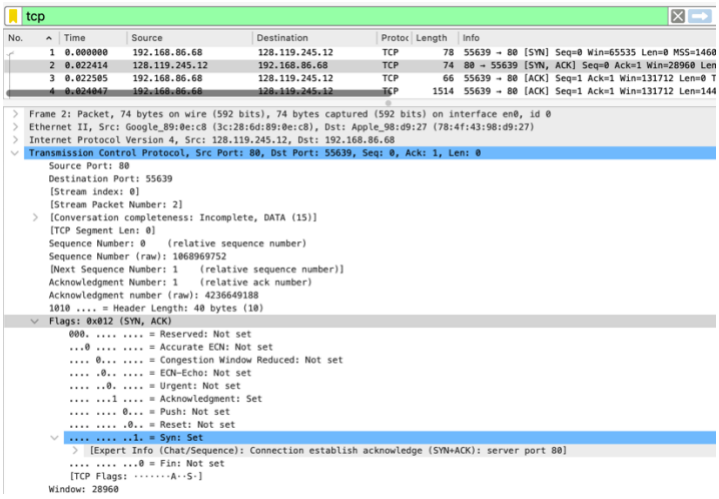
    → Relative sequence number: 0
    → Raw sequence number: 1068969752

    b. What is it in the segment that identifies the segment as a SYNACK segment?

    → Relative acknowledgment number: 1
    → Raw acknowledgment number: 4236649188

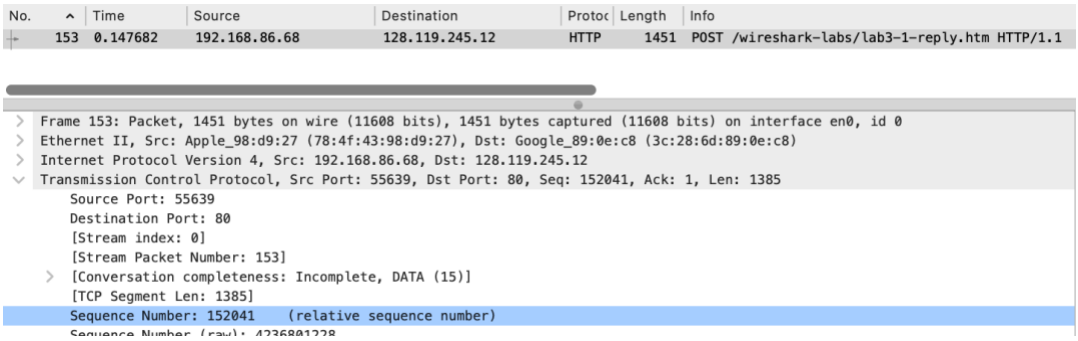
    c. What is the value of the Acknowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value?

    → Because the SYN flag and the ACK flag are both set in the TCP header. This indicates that the server is acknowledging the client's SYN and simultaneously sending its own SYN to continue the TCP three way handshake.



5. What is the sequence number of the TCP segment containing the header of the HTTP POST command? Note that in order to find the POST message header, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with the ASCII text "POST" within its DATA field4,5. How many bytes of data are contained in the payload (data) field of this TCP segment? Did all of the data in the transferred file alice.txt fit into this single segment?

   → Sequence number of segment with POST header: 152041
   → Payload size of this segment: 1385 bytes
   → No. Only part of the file is included in this segment.



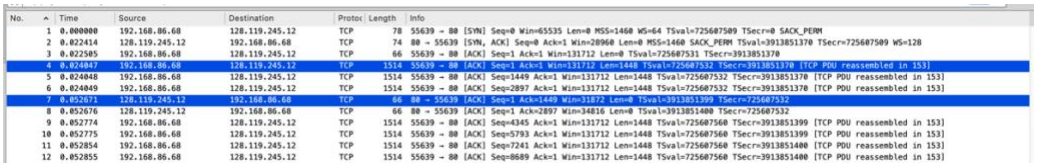
6. Consider the TCP segment containing the HTTP "POST" as the first segment in the data transfer part of the TCP connection.

   a. At what time was the first segment (the one containing the HTTP POST) in the data-transfer part of the TCP connection sent?

      → 0.147682 s

b. At what time was the ACK for this first data-containing segment received?

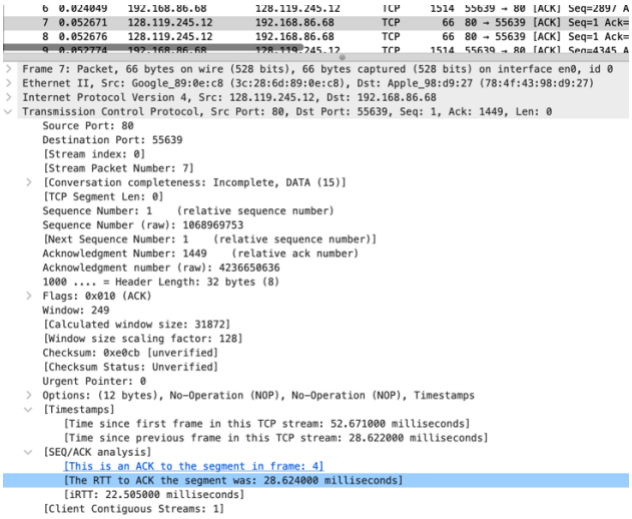
→ 0. 024047 s



No.	Time	Source	Destination	Protoc	Length	Info
1	0.000000	192.168.86.68	128.119.245.12	TCP	78	55639 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=725607509 TSecr=0 SACK_PERM
2	0.022414	128.119.245.12	192.168.86.68	TCP	74	80 → 55639 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM TSval=3913851370 TSecr=725607509 WS=128
3	0.022585	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP PDU reassembled in 153]
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP PDU reassembled in 153]
6	0.024049	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP PDU reassembled in 153]
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP PDU reassembled in 153]
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP PDU reassembled in 153]
11	0.052854	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=7241 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP PDU reassembled in 153]
12	0.052855	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=8689 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851400 [TCP PDU reassembled in 153]

c. What is the RTT for this first data-containing segment?

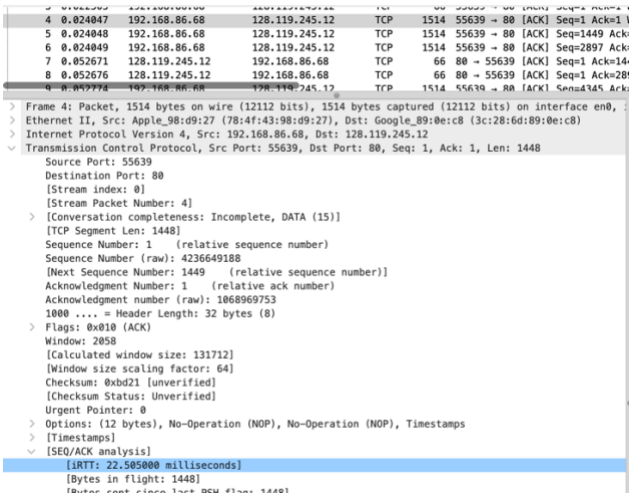
→ 28.624 milliseconds



```
6  0.024049   192.168.86.68        128.119.245.12       TCP      1514  55639 → 80 [ACK] Seq=2897 A
7  0.052671   128.119.245.12       192.168.86.68        TCP        66  80 → 55639 [ACK] Seq=1 Ack=
8  0.052676   128.119.245.12       192.168.86.68        TCP        66  80 → 55639 [ACK] Seq=1 Ack=
9  0.052774   192.168.86.68        128.119.245.12       TCP      1514  55639 → 80 [ACK] Seq=4345 A
> Frame 7: Packet, 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface en0, id 0
> Ethernet II, Src: Google_89:0e:c8 (3c:28:6d:89:0e:c8), Dst: Apple_98:d9:27 (78:4f:43:98:d9:27)
> Internet Protocol Version 4, Src: 128.119.245.12, Dst: 192.168.86.68
v Transmission Control Protocol, Src Port: 80, Dst Port: 55639, Seq: 1, Ack: 1449, Len: 0
    Source Port: 80
    Destination Port: 55639
    [Stream index: 0]
    [Stream Packet Number: 7]
  > [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 0]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 1068969753
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 1449    (relative ack number)
    Acknowledgment number (raw): 4236650636
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x010 (ACK)
    Window: 249
    [Calculated window size: 31872]
    [Window size scaling factor: 128]
    Checksum: 0xe0cb [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  v [Timestamps]
      [Time since first frame in this TCP stream: 52.671000 milliseconds]
      [Time since previous frame in this TCP stream: 28.622000 milliseconds]
  v [SEQ/ACK analysis]
      [This is an ACK to the segment in frame: 4]
      [The RTT to ACK the segment was: 28.624000 milliseconds]
      [iRTT: 22.505000 milliseconds]
    [Client Contiguous Streams: 1]
```

d. What is the RTT value the second data-carrying TCP segment and its ACK?

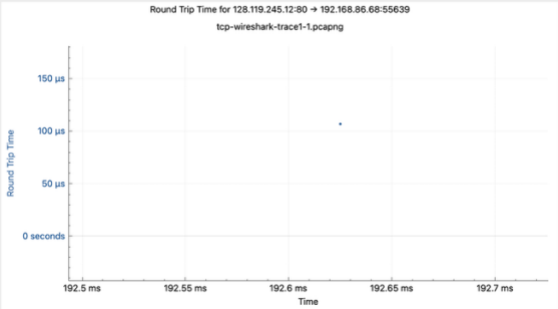
→22. 505 milliseconds



```
4  0.024047   192.168.86.68        128.119.245.12       TCP      1514  55639 → 80 [ACK] Seq=1 Ack=1
5  0.024048   192.168.86.68        128.119.245.12       TCP      1514  55639 → 80 [ACK] Seq=1449 Ack
6  0.024049   192.168.86.68        128.119.245.12       TCP      1514  55639 → 80 [ACK] Seq=2897 Ack
7  0.052671   128.119.245.12       192.168.86.68        TCP        66  80 → 55639 [ACK] Seq=1 Ack=14
8  0.052676   128.119.245.12       192.168.86.68        TCP        66  80 → 55639 [ACK] Seq=1 Ack=28
9  0.052774   192.168.86.68        128.119.245.12       TCP      1514  55639 → 80 [ACK] Seq=4345 Ack
> Frame 4: Packet, 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface en0, 
> Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Google_89:0e:c8 (3c:28:6d:89:0e:c8)
> Internet Protocol Version 4, Src: 192.168.86.68, Dst: 128.119.245.12
v Transmission Control Protocol, Src Port: 55639, Dst Port: 80, Seq: 1, Ack: 1, Len: 1448
    Source Port: 55639
    Destination Port: 80
    [Stream index: 0]
    [Stream Packet Number: 4]
  > [Conversation completeness: Incomplete, DATA (15)]
    [TCP Segment Len: 1448]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 4236649188
    [Next Sequence Number: 1449    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 1068969753
    1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x010 (ACK)
    Window: 2058
    [Calculated window size: 131712]
    [Window size scaling factor: 64]
    Checksum: 0xbd21 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  > [Timestamps]
  v [SEQ/ACK analysis]
      [iRTT: 22.505000 milliseconds]
      [Bytes in flight: 1448]
      [Bytes sent since last PSH flag: 1448]
```

e. What is the EstimatedRTT value (see Section 3.5.3, in the text) after the ACK for the second data-carrying segment is received? Assume that in making this calculation after the received of the ACK for the second segment, that the initial value of EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 242, and a value of a = 0.125. *Note: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the "listing of captured packets" window that is being sent from the client to the gaia.cs.umass.edu server. Then select: Statistics->TCP Stream Graph->Round Trip Time Graph*.

→ EstimatedRTT = 0.875 × 0.029 + 0.125 × 0.029 ≈ 0.029 seconds (29 ms)



7. What is the length (header plus payload) of each of the first four data-carrying TCP segments?

→ Frame 4: Total = 1514 bytes, IP packet = 1500 bytes, TCP payload = 1448 bytes

→ Frame 5: Total = 1514 bytes, IP packet = 1500 bytes, TCP payload = 1448 bytes

→ Frame 6: Total = 1514 bytes, IP packet = 1500 bytes, TCP payload = 1448 bytes

→ Frame 9: Total = 1514 bytes, IP packet = 1500 bytes, TCP payload = 1448 bytes

3	0.022505	192.168.86.68	128.119.245.12	TCP	66	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=725607531 TSecr=3913851370
4	0.024047	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP PDU reassembled in 153]
5	0.024048	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=1449 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP PDU reassembled in 153]
6	0.024049	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=2897 Ack=1 Win=131712 Len=1448 TSval=725607532 TSecr=3913851370 [TCP PDU reassembled in 153]
7	0.052671	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=1449 Win=31872 Len=0 TSval=3913851399 TSecr=725607532
8	0.052676	128.119.245.12	192.168.86.68	TCP	66	80 → 55639 [ACK] Seq=1 Ack=2897 Win=34816 Len=0 TSval=3913851400 TSecr=725607532
9	0.052774	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=4345 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP PDU reassembled in 153]
10	0.052775	192.168.86.68	128.119.245.12	TCP	1514	55639 → 80 [ACK] Seq=5793 Ack=1 Win=131712 Len=1448 TSval=725607560 TSecr=3913851399 [TCP PDU reassembled in 153]

8. What is the minimum amount of available buffer space advertised to the client by gaia.cs.umass.edu among these first four data-carrying TCP segments7? Does the lack of receiver buffer space ever throttle the sender for these first four data-carrying segments?

→ The smallest advertised receive window in the first four data segments is 31,872 bytes, calculated from Frame 7 where the Window Size Value is 249 and the scale factor is 7 (249 × 128). This does not throttle the sender because the window is far larger than the 1,448 byte segment size and can hold roughly 22 segments, while the sender never transmits anywhere near that many without receiving ACKs.

```
    1000 .... = Header Length: 32 bytes
  > Flags: 0x010 (ACK)
    Window: 249
    [Calculated window size: 31872]
    [Window size scaling factor: 128]
    Checksum: 0xe0cb [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
```

9. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

   → No retransmitted segments found in the trace

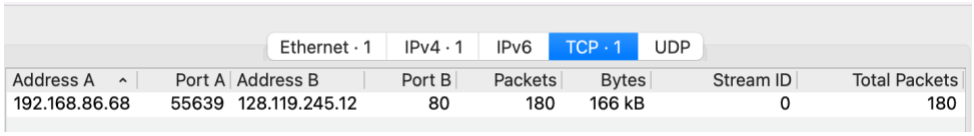
10. How much data does the receiver typically acknowledge in an ACK among the first ten data-carrying segments sent from the client to gaia.cs.umass.edu? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 in the text) among these first ten data-carrying segments?

    1. Frame 2: ACK = 1
    2. Frame 7: ACK = 1449 (increment: 1448)
    3. Frame 8: ACK = 2897 (increment: 1448)
    4. Frame 13: ACK = 4345 (increment: 1448)
    5. Frame 16: ACK = 5793 (increment: 1448)
    6. Frame 17: ACK = 7241 (increment: 1448)
    7. Frame 18: ACK = 8689 (increment: 1448)
    8. Frame 19: ACK = 10137 (increment: 1448)
    9. Frame 28: ACK = 11585 (increment: 1448)
    10. Frame 29: ACK = 13033 (increment: 1448)

    → Typical ACK increment: 1448 bytes

11. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

→ 166kb



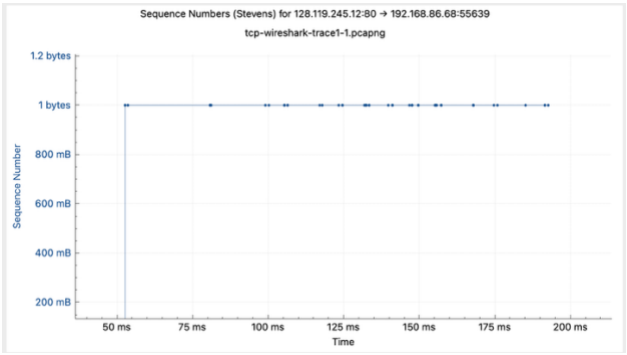
Ethernet · 1 | IPv4 · 1 | IPv6 | TCP · 1 | UDP

Address A	Port A	Address B	Port B	Packets	Bytes	Stream ID	Total Packets
192.168.86.68	55639	128.119.245.12	80	180	166 kB	0	180

4. TCP congestion control in action

12. Use the Time-Sequence-Graph(Stevens) plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server. Consider the "fleets" of packets sent around t = 0.025, t = 0.053, t = 0.082 and t = 0.1. Comment on whether this looks as if TCP is in its slow start phase, congestion avoidance phase or some other phase. Figure 6 shows a slightly different view of this data.

→



13. These "fleets" of segments appear to have some periodicity. What can you say about the period?

→

14. Answer each of two questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu

→