

# Student Information

Full Name : Mehmet Rüçhan Yavuzdemir  
Id Number: 2522159

## 1 A first look at the captured trace

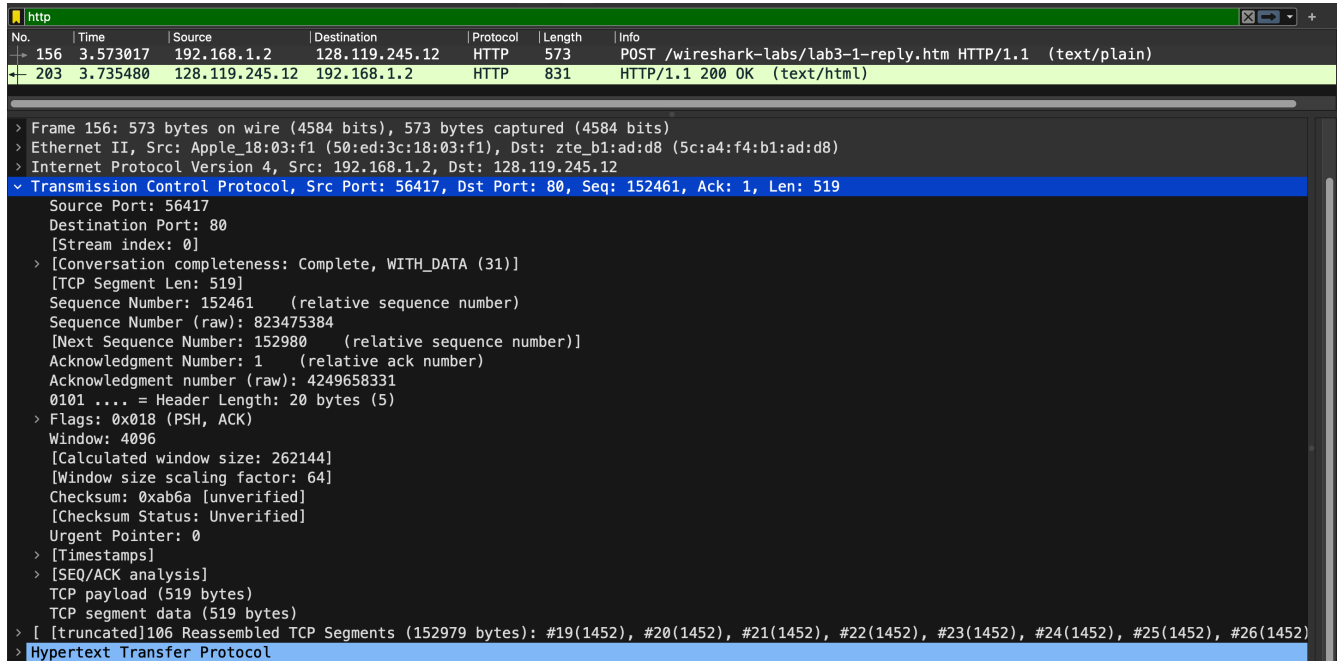


Figure 1: Answer 1-2

### Answer 1.1

The source IP address is 192.168.1.2, as shown in the figure, under the **Source** column in the packet listing window.

### Answer 1.2

The source port number is 56417, as shown in the figure under the **TCP Source Port** section.

### Answer 2.1

The destination IP address is 128.119.245.12, as shown in the figure, under the **Destination** column in the packet listing window.

### Answer 2.2

The destination port number is 80, as shown in the figure under the **TCP Destination Port** section. This is the port number from where the server is sending and receiving TCP segments for this connection.

## 2 TCP Basics

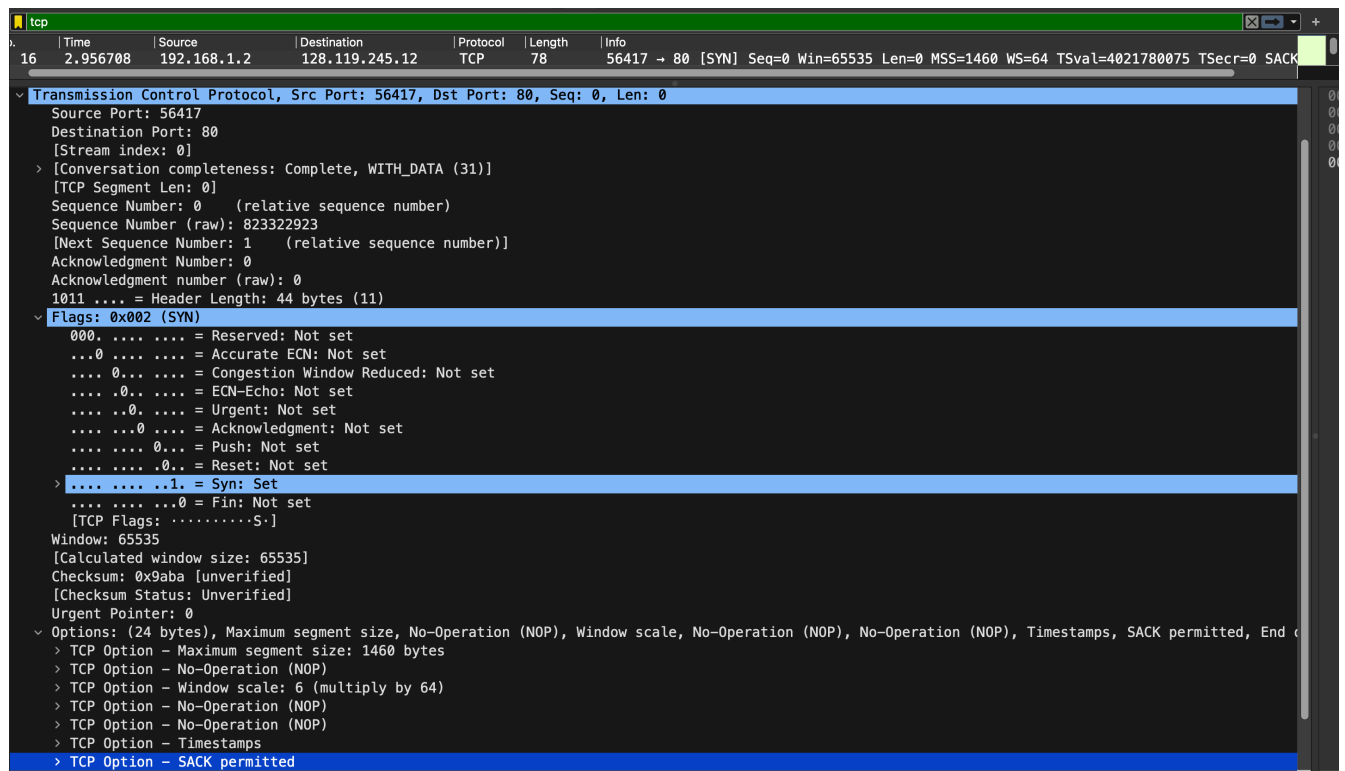


Figure 2: Answer 3

### Answer 3.1

Under the TCP section, we have a field called **Sequence Number (raw)**. This is exactly what we are looking for, not a relative sequence number, not a packet number. The sequence number of the TCP SYN segment initiating the TCP connection between client and server is 823322923.

### Answer 3.2

For this purpose, the TCP **Flags** section is used. As highlighted in the figure, **Syn** bit is set to 1, indicating that this segment is a **SYN** segment.

### Answer 3.3

In the TCP **Options** section, there is an option called **SACK permitted**. Hence, the TCP receiver will be able to use Selective Acknowledgments, so-called **SACKs**.

No.	Time	Source	Destination	Protocol	Length	Info
16	2.956788	192.168.1.2	128.119.245.12	TCP	78	56417 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=4021780075 TSecr=0 SA
17	3.112469	128.119.245.12	192.168.1.2	TCP	66	80 → 56417 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1452 SACK_PERM WS=128
18	3.112664	192.168.1.2	128.119.245.12	TCP	54	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
19	3.114329	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled PD
20	3.114337	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1453 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled

Transmission Control Protocol, Src Port: 80, Dst Port: 56417, Seq: 0, Ack: 1, Len: 0

Source Port: 80

Destination Port: 56417

[Stream index: 0]

[Conversation completeness: Complete, WITH\_DATA (31)]

[TCP Segment Len: 0]

Sequence Number: 0 (relative sequence number)

Sequence Number (raw): 4249658330

[Next Sequence Number: 1 (relative sequence number)]

Acknowledgment Number: 1 (relative ack number)

Acknowledgment number (raw): 823322924

1000 .... = Header Length: 32 bytes (8)

Flags: 0x012 (SYN, ACK)

000. .... = Reserved: Not set

...0 .... = Accurate ECN: Not set

.... 0... = Congestion Window Reduced: Not set

.... .0.. = ECN-Echo: Not set

.... ..0. = Urgent: Not set

.... ...1 = Acknowledgment: Set

.... .... 0... = Push: Not set

.... .... .0.. = Reset: Not set

.... .... ..1. = Syn: Set

.... .... ...0 = Fin: Not set

[TCP Flags: .....A..S.]

Figure 3: Answer 4

### Answer 4.1

Again, under the TCP section, we have a field called **Sequence Number (raw)**. This is exactly what we are looking for, not a relative sequence number, not a packet number. The sequence number of the TCP SYN sent by the server to the client is 4249658330.

### Answer 4.2

For this purpose, the TCP Flags section is used. In the figure above, Acknowledgment and Syn bit is set to 1, indicating that this segment is a SYNACK segment.

### Answer 4.3

The value of Acknowledgment field in the SYNACK segment is 823322924.

### Answer 4.4

The value of the Acknowledgment field in the SYNACK segment is the sequence number of the next expected data to be received at the server from the client, and in this case, one higher than the sequence number in the initial SYN segment sent from the client to server.

3

```

Transmission Control Protocol, Src Port: 56417, Dst Port: 80, Seq: 1, Ack: 1, Len: 1452
  Source Port: 56417
  Destination Port: 80
  [Stream index: 0]
  > [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 1452]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 823322924
  [Next Sequence Number: 1453 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 4249658331
  0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
  Window: 4096
  [Calculated window size: 262144]
  [Window size scaling factor: 64]
  Checksum: 0x5036 [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
  TCP payload (1452 bytes)
  [Reassembled PDU in frame: 156]
  TCP segment data (1452 bytes)
0030 10 00 50 36 00 00 50 4f 53 54 20 2f 77 69 72 65 ..P6..P0 ST /wire
0040 73 68 61 72 6b 2d 6c 61 62 73 2f 6c 61 62 33 2d shark-lab/lab3-
0050 31 2d 72 65 70 6c 79 2e 68 74 6d 20 48 54 54 50 1-reply.htm HTTP
0060 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 67 61 69 61 /1.1..Host: gaia
0070 2e 63 73 2e 75 6d 61 73 73 2e 65 64 75 0d 0a 55 .cs.umass.edu..U
0080 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c ser-Agent: Mozil
0090 6c 61 2f 35 2e 30 20 28 4d 61 63 69 6e 74 6f 73 la/5.0 ( Macintos
00a0 68 3b 20 49 6e 74 65 6c 20 4d 61 63 20 4f 53 20 h; Intel Mac OS
00b0 58 20 31 30 2e 31 35 3b 20 72 76 3a 31 30 39 2e X 10.15; rv:109.
00c0 30 29 20 47 65 63 6b 6f 2f 32 30 31 30 30 31 30 0) Gecko /2010010
00d0 31 20 46 69 72 65 66 6f 78 2f 31 31 39 2e 30 0d 1 Firefo x/119.0
00e0 0a 41 63 63 65 70 74 3a 20 74 65 78 74 2f 68 74 .Accept: text/ht
00f0 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 ml,appli cation/x
0100 68 74 6d 6c 2b 78 6d 6c 2c 61 70 70 6c 69 63 61 html+xml ,applica
0110 74 69 6f 6e 2f 78 6d 6c 3b 71 3d 30 2e 39 2c 69 tion/xml ;q=0.9,i

```

Figure 4: Answer 5

## Answer 5.1

I found the TCP segment containing the header of the HTTP POST request, and the beginning of the `alice.txt` file (it did not fit into the screen though). In the packet content window, you can see the real HTTP POST header information in ASCII format. The sequence number can be retrieved from the highlighted TCP Sequence Number (raw) field, so the sequence number is 823322924.

## Answer 5.2

As highlighted in the figure above, the size of the payload(data) of this TCP segment is 1452 bytes. Of course, the whole `alice.txt` file did not fit into one segment, not even close! Remember from Figure 1 that there were 106 TCP segments, so there is a lot to transmit!

No.	Time	Source	Destination	Protocol	Length	Info
16	2.956708	192.168.1.2	128.119.245.12	TCP	78	56417 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=4021780075 TSecr=0 SA
17	3.112469	128.119.245.12	192.168.1.2	TCP	66	80 → 56417 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1452 SACK_PERM WS=128
18	3.112664	192.168.1.2	128.119.245.12	TCP	54	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
19	3.114329	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled PD
20	3.114337	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1453 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
21	3.114340	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=2905 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
22	3.114343	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=4357 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
23	3.114345	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=5809 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
24	3.114349	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=7261 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
25	3.114352	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=8713 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
26	3.114354	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=10165 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble
27	3.114355	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=11617 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble
28	3.114356	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=13069 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble
29	3.269410	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=7261 Win=43776 Len=0
30	3.269411	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=14521 Win=58240 Len=0

Figure 5: Answer 6

### Answer 6.1

As highlighted in the figure above, the first TCP segment containing the HTTP POST request in the data-transfer part of the TCP connection was sent 3.114329 seconds after the packet sniffing started.

### Answer 6.2

Again, as highlighted above, the first ACK from server to client for the first data-containing segment arrived at 3.269410 seconds after the packet sniffing started.

### Answer 6.3

Since we solved the two questions above, finding RTT for the first data-containing segment is easy. RTT is basically how much time has passed to wait for the ACK for the data-containing segment. Using the values from the previous two questions, the RTT for the first data-containing segment is  $3.269410 - 3.114329 = 0.155081$  seconds.

### Answer 6.4

For the second data-carrying TCP segment, there is no ACK segment for the second segment specifically sent by the server. Instead, the server sent cumulative ACK for this purpose, which can be understood by examining the window sizes. Therefore, we can approximate the RTT from the values we currently have. The client sent the second data-carrying TCP segment at the time 3.114337, and its corresponding ACK segment was received at the time 3.269410. Hence, the RTT for the second data-carrying TCP segment is  $3.269410 - 3.114337 = 0.155073$  seconds.

### Answer 6.5

In this question, we are asked to find the `EstimatedRTT` after the ACK for the second data-carrying segment is received by the client. We assume that the initial value of the `EstimatedRTT` is equal to the measured RTT for the first segment, and its new value is calculated by the formula below, with  $\alpha = 0.125$ .

$$\text{EstimatedRTT} = (1-\alpha) \cdot \text{EstimatedRTT} + \alpha \cdot \text{SampleRTT}$$

If we plug the values we found from the answers 6.3 and 6.4 into the formula, where `EstimatedRTT` is the measured RTT for the first data-carrying segment, and `SampleRTT` is the measured RTT for the second data-carrying segment, we will find the `EstimatedRTT` after the client receives the ACK for the second data-carrying segment.

$$\text{EstimatedRTT} = (0.875) \cdot 0.155081 + (0.125) \cdot 0.155073 = 0.15508$$

No.	Time	Source	Destination	Protocol	Length	Info
16	2.956708	192.168.1.2	128.119.245.12	TCP	78	56417 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=4021780075 TSecr=0 SA
17	3.112469	128.119.245.12	192.168.1.2	TCP	66	80 → 56417 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1452 SACK_PERM WS=128
18	3.112664	192.168.1.2	128.119.245.12	TCP	54	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
19	3.114329	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled PD
20	3.114337	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1453 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
21	3.114340	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=2905 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
22	3.114343	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=4357 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled

```

> Frame 19: 1506 bytes on wire (12048 bits), 1506 bytes captured (12048 bits)
> Ethernet II, Src: Apple_18:03:f1 (50:ed:3c:18:03:f1), Dst: zte_b1:ad:d8 (5c:a4:f4:b1:ad:d8)
> Internet Protocol Version 4, Src: 192.168.1.2, Dst: 128.119.245.12
< Transmission Control Protocol, Src Port: 56417, Dst Port: 80, Seq: 1, Ack: 1, Len: 1452
  Source Port: 56417
  Destination Port: 80
  [Stream index: 0]
  > [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 1452]
    Sequence Number: 1 (relative sequence number)
    Sequence Number (raw): 823322924
    [Next Sequence Number: 1453 (relative sequence number)]
    Acknowledgment Number: 1 (relative ack number)
    Acknowledgment number (raw): 4249658331

```

Figure 6: Answer 7

## Answer 7

As highlighted in the figure above, the length of each of the first four data-carrying TCP segments is 1452 bytes.

tcp && ip.src == 128.119.245.12							
No.	Time	Source	Destination	Protocol	Length	Info	
17	3.112469	128.119.245.12	192.168.1.2	TCP	66	80 → 56417	[SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1452 SACK_PERM WS=128
29	3.269410	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=7261 Win=43776 Len=0
30	3.269411	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=14521 Win=58240 Len=0
51	3.417436	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=17425 Win=64128 Len=0
52	3.417440	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=20329 Win=69888 Len=0
53	3.417488	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=21781 Win=72832 Len=0
64	3.418704	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=23233 Win=75776 Len=0
67	3.420014	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=24685 Win=78720 Len=0
68	3.420015	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=26137 Win=81536 Len=0
69	3.420016	128.119.245.12	192.168.1.2	TCP	60	80 → 56417	[ACK] Seq=1 Ack=29041 Win=87424 Len=0

Figure 7: Answer 8

## Answer 8.1

I filtered the TCP packets having the IP address of the server so that we can see how window size, which can also be used interchangeably with buffer space in this case, changes over time. As highlighted in the figure above, the initial and also minimum available buffer space provided by the server to the client is 29200 bytes. It increases over time. The window size indicated in the SYNACK segment is for the first data-carrying TCP segment, hence the value we are looking for is again 29200.

## Answer 8.2

No, actually, the reality is the exact opposite. The receiver buffer space is large, hence it does not throttle the sender for the first four data-carrying segments. It can fit almost 20 TCP segments at a time, no overflow occurred, and everything is fine. Furthermore, remember that the window size is growing over time. Therefore, no throttling occurred.

tcp && ip.src == 192.168.1.2 && ip.dst == 128.119.245.12							
No.	Time	Source	Destination	Protocol	Length	Info	
16	2.956708	192.168.1.2	128.119.245.12	TCP	78	56417 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=4021780075 TSecr=0 SA	
18	3.112664	192.168.1.2	128.119.245.12	TCP	54	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0	
19	3.114329	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled PD	
20	3.114337	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1453 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled	
21	3.114340	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=2905 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled	
22	3.114343	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=4357 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled	
23	3.114345	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=5809 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled	
24	3.114349	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=7261 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled	
25	3.114352	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=8713 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled	
26	3.114354	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=10165 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
27	3.114355	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=11617 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
28	3.114356	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=13069 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
31	3.269579	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=14521 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
32	3.269583	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=15973 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
33	3.269586	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=17425 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
34	3.269587	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=18877 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
35	3.269589	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=20329 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
36	3.269591	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=21781 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
37	3.269593	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=23233 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
38	3.269595	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=24685 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
39	3.269596	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=26137 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
40	3.269598	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=27589 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
41	3.269675	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=29041 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
42	3.269676	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=30493 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
43	3.269677	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=31945 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
44	3.269678	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=33397 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
45	3.269679	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=34849 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
46	3.269681	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=36301 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
47	3.269682	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=37753 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
48	3.269684	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=39205 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
49	3.269685	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=40657 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
50	3.269687	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=42109 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
54	3.417679	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=43561 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
55	3.417683	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=45013 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
56	3.417687	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=46465 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
57	3.417692	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=47917 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	
58	3.417806	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=49369 Ack=1 Win=262144 Len=1452 [TCP segment of a reassemble	

Figure 8: Answer 9

## Answer 9.1

No, there are not any retransmitted segments in my trace file.

## Answer 9.2

To be able to analyze the trace file, I used this filter query:

```
tcp && ip.src == 192.168.1.2 && ip.dst == 128.119.245.12
```

Since the sequence numbers are strictly increasing, there is no retransmission happened.



No.	Time	Source	Destination	Protocol	Length	Info
16	2.956708	192.168.1.2	128.119.245.12	TCP	78	56417 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=4021780075 TSecr=0 SA
17	3.112469	128.119.245.12	192.168.1.2	TCP	66	80 → 56417 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1452 SACK_PERM WS=128
18	3.112664	192.168.1.2	128.119.245.12	TCP	54	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
19	3.114329	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled PD
20	3.114337	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=1453 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
21	3.114340	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=2905 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
22	3.114343	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=4357 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
23	3.114345	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=5809 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
24	3.114349	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=7261 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
25	3.114352	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=8713 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
26	3.114354	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=10165 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
27	3.114355	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=11617 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
28	3.114356	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=13069 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
29	3.269410	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=7261 Win=43776 Len=0
30	3.269411	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=14521 Win=58240 Len=0
31	3.269579	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=14521 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
32	3.269583	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=15973 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
33	3.269586	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=17425 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
34	3.269587	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=18877 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
35	3.269589	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=20329 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
36	3.269591	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=21781 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
37	3.269593	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=23233 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
38	3.269595	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=24685 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
39	3.269596	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=26137 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled
40	3.269598	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=27589 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembled

Figure 9: Answer 10

### Answer 10.1

The receiver typically acknowledges 5 TCP segments of data at a time. For each of the 5 data-containing segments, it accumulates them and sends a cumulative ACK segment. As a result, the receiver typically acknowledges 7260 bytes of data at a time.

### Answer 10.2

Yes! After the first ACK every remaining ACK acknowledges 5 segments' worth of payload data.

No.	Time	Source	Destination	Protocol	Length	Info
149	3.572949	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=142297 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
150	3.572952	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=143749 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
151	3.572956	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=145201 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
152	3.572958	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=146653 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
153	3.573009	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=148105 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
154	3.573013	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=149557 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
155	3.573015	192.168.1.2	128.119.245.12	TCP	1506	56417 → 80 [ACK] Seq=151009 Ack=1 Win=262144 Len=1452 [TCP segment of a reassembl
156	3.573017	192.168.1.2	128.119.245.12	HTTP	573	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)

Figure 10: Answer 11-1

No.	Time	Source	Destination	Protocol	Length	Info
187	3.714633	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=104545 Win=187520 Len=0
188	3.716467	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=108901 Win=187520 Len=0
189	3.716468	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=111805 Win=187520 Len=0
190	3.716469	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=114709 Win=187520 Len=0
191	3.719678	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=117613 Win=187520 Len=0
192	3.719680	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=121969 Win=187520 Len=0
193	3.721611	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=124873 Win=187520 Len=0
194	3.721613	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=127777 Win=187520 Len=0
195	3.725643	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=130681 Win=187520 Len=0
196	3.725645	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=135037 Win=187520 Len=0
197	3.726887	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=137941 Win=187520 Len=0
198	3.726888	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=140845 Win=187520 Len=0
199	3.729965	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=143749 Win=187520 Len=0
200	3.729966	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=148105 Win=187520 Len=0
201	3.729967	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=151009 Win=187520 Len=0
202	3.733738	128.119.245.12	192.168.1.2	TCP	60	80 → 56417 [ACK] Seq=1 Ack=152980 Win=187520 Len=0
203	3.735480	128.119.245.12	192.168.1.2	HTTP	831	HTTP/1.1 200 OK (text/html)
204	3.735599	192.168.1.2	128.119.245.12	TCP	54	56417 → 80 [ACK] Seq=152980 Ack=778 Win=261312 Len=0

Figure 11: Answer 11-2

## Answer 11.1

The throughput for the TCP connection is 333514 bytes per second.

## Answer 11.2

Recall from Answer 6.1 that the first segment containing the first bytes of data was sent at the time 3.114329. From Figure 10, we can observe when the last segment containing the last bytes of data was sent, which is the time 3.573017 in terms of seconds. From Figure 11, we can observe how many bytes were transferred from the client to the server, which is the last ACK value being sent from the server to the last data-containing TCP segment sent by the client,  $152980 - 2 = 152978$ . Throughput is how many bytes were transferred during the amount of time between when the client sent the first segment containing the first bytes of data in alice.txt and when the last segment in the connection containing the last bytes of data in alice.txt was sent.

The calculation of the throughput is as follows:

$$152978 / (3.573017 - 3.114329) = 333514 \text{ bytes per second}$$

### 3 TCP congestion control in action

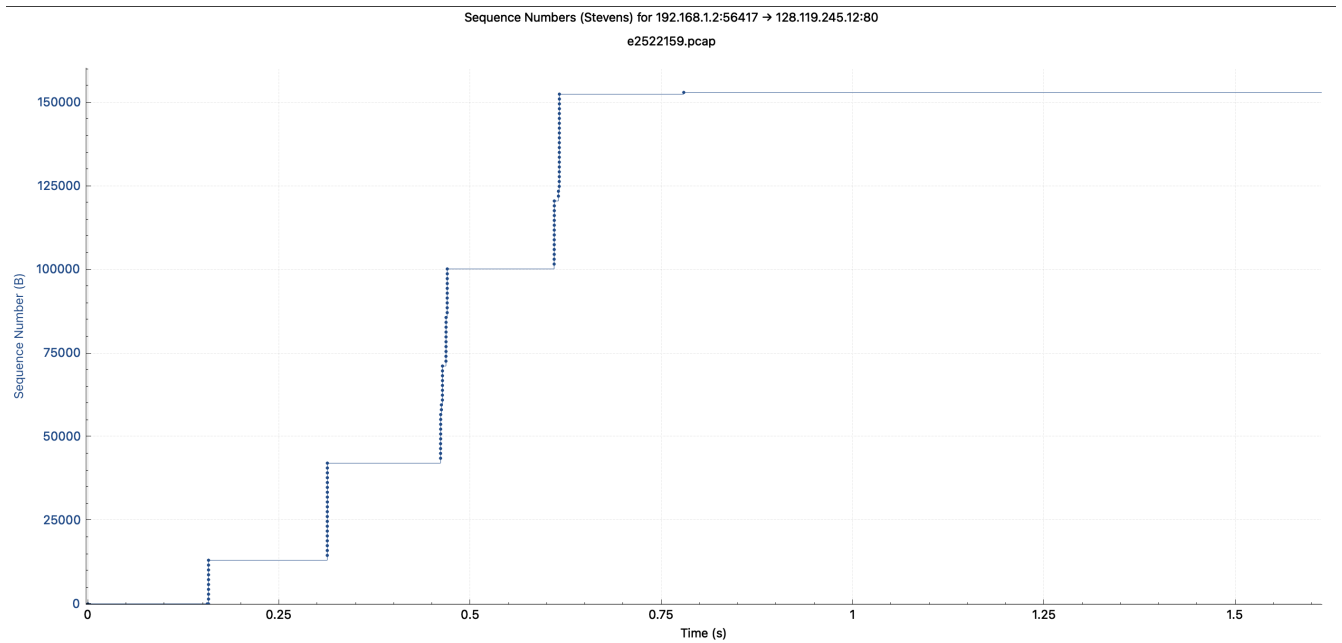


Figure 12: Answer 12

#### Answer 12.1

The number of packets being sent in each fleet of packets doubles, i.e. exponentially increases. Therefore, it is obvious that TCP is in its slow start phase.

#### Answer 12.2

It seems that the graph has a period of roughly 0.16 seconds.