

ASSIGNMENT 5: TCP

HRISHIKESH RAVINDRA KARANDE

Part1

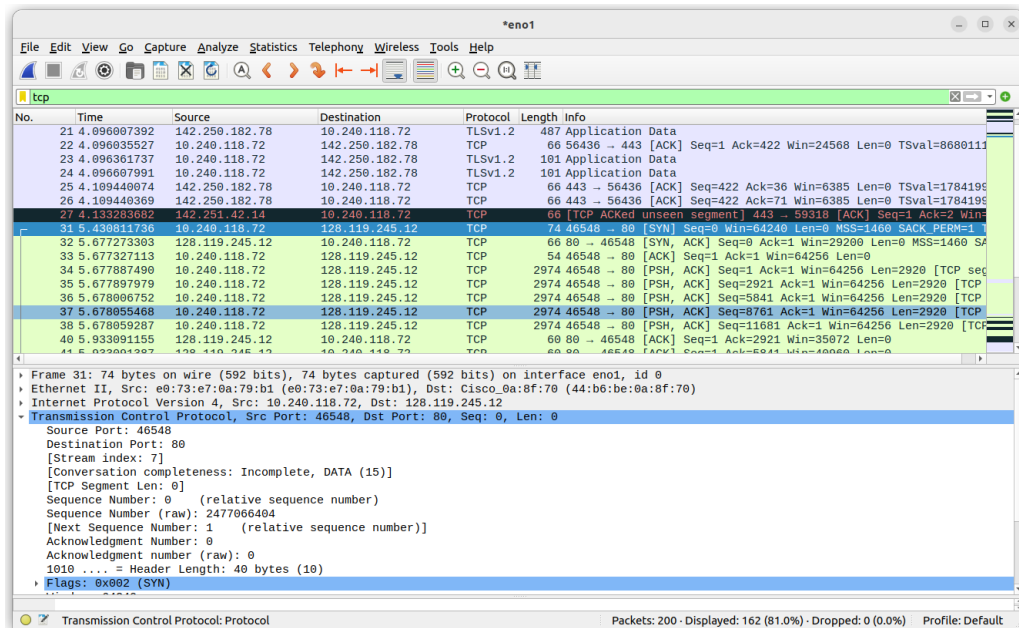
No.	Time	Source	Destination	Protocol	Length	Info
11	1.372035430	54.148.110.228	10.240.118.72	TCP	66	[TCP ACKed unseen segment] 443 → 40996 [ACK] Seq=1 Ack=2 Win=...
20	4.095770729	10.240.118.72	142.251.42.14	TCP	66	59318 → 443 [ACK] Seq=1 Ack=1 Win=501 Len=0 TSval=2121829754
21	4.096007392	142.250.182.78	10.240.118.72	TLSv1.2	487	Application Data
22	4.096035527	10.240.118.72	142.250.182.78	TCP	66	56436 → 443 [ACK] Seq=1 Ack=422 Win=24568 Len=0 TSval=8680111
23	4.096361737	10.240.118.72	142.250.182.78	TLSv1.2	101	Application Data
24	4.096607991	10.240.118.72	142.250.182.78	TLSv1.2	101	Application Data
25	4.109440074	142.250.182.78	10.240.118.72	TCP	66	443 → 56436 [ACK] Seq=422 Ack=36 Win=6385 Len=0 TSval=1784199
26	4.109440369	142.250.182.78	10.240.118.72	TCP	66	443 → 56436 [ACK] Seq=422 Ack=71 Win=6385 Len=0 TSval=1784199
27	4.133283682	142.251.42.14	10.240.118.72	TCP	66	[TCP ACKed unseen segment] 443 → 59318 [ACK] Seq=1 Ack=2 Win=...
31	5.430811736	10.240.118.72	128.119.245.12	TCP	74	46548 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1
32	5.677273303	128.119.245.12	10.240.118.72	TCP	66	80 → 46548 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SA...
33	5.677327113	10.240.118.72	128.119.245.12	TCP	54	46548 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0
34	5.677887490	10.240.118.72	128.119.245.12	TCP	2974	46548 → 80 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=2920 [TCP seq...
35	5.677897979	10.240.118.72	128.119.245.12	TCP	2974	46548 → 80 [PSH, ACK] Seq=2921 Ack=1 Win=64256 Len=2920 [TCP...
36	5.678006752	10.240.118.72	128.119.245.12	TCP	2974	46548 → 80 [PSH, ACK] Seq=5841 Ack=1 Win=64256 Len=2920 [TCP...
37	5.678055468	10.240.118.72	128.119.245.12	TCP	2974	46548 → 80 [PSH, ACK] Seq=8761 Ack=1 Win=64256 Len=2920 [TCP...
38	5.678059387	10.240.118.72	128.119.245.12	TCP	2974	46548 → 80 [PSH, ACK] Seq=11681 Ack=1 Win=64256 Len=2920 [TCP...

Frame 31: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface eno1, id 0
Ethernet II, Src: e0:73:e7:0a:79:b1 (e0:73:e7:0a:79:b1), Dst: Cisco_0a:8f:70 (44:b6:be:0a:8f:70)
Internet Protocol Version 4, Src: 10.240.118.72, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 46548, Dst Port: 80, Seq: 0, Len: 0

0000 44 b6 be 0a 8f 70 e0 73 e7 0a 79 b1 08 00 45 00 D...p.s...y...E
0010 00 3c d3 49 40 00 40 06 70 b6 0a f0 76 48 80 77 -<I@.p...vH.w
0020 f5 0c b5 d4 00 50 93 a5 08 a4 00 00 00 00 a0 02P.....
0030 fa f0 f6 ea 00 00 02 04 05 b4 04 02 08 0a 8e 83
0040 23 7a 00 00 00 00 01 03 03 07 #z.....

Part2

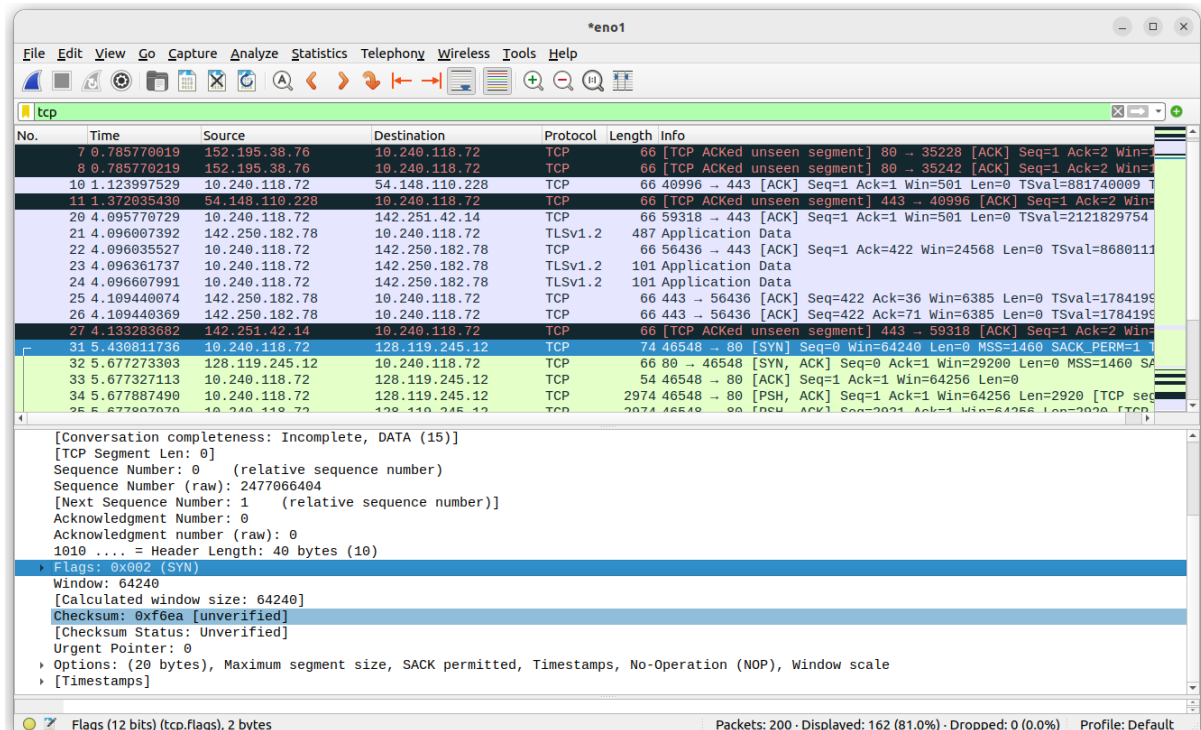
- 1) Client IP Address:10.240.118.72 Port: 46548



2) IP Address of gaia.cs.umass.edu: 128.119.245.12, port number is 80. This can also be seen in the screenshot of above question.

Part3:

Q1) Sequence number = 0. The Flags in this segment help to identify it is SYN packet.



Q2) Sequence Number of SYN ACK segment = 0

The value of Ack field is determined by gaia.cs.umass.edu by adding 1 to the previous SYN Segment from client computer. The packet is identified as SYN-ACK by the value of SYN and ACK flags which is

set to 1.

Q3) The sequence number of the TCP segment containing the header of the HTTP POST command is 1. The Payload field of request contains 1250 bytes of data. All the data from Alice.txt is not transferred in a single request.

No.	Time	Source	Destination	Protocol	Length	Info
14	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1 Win=0 Len=0 (TCP segment of a reassembled PDU)
15	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
16	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
17	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
18	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
19	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
20	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
21	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
22	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
23	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
24	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
25	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
26	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
27	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
28	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
29	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
30	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
31	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
32	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
33	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
34	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
35	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
36	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
37	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
38	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
39	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
40	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
41	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
42	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
43	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)
44	0.000000	10.0.0.0	10.0.0.0	TCP	60	65535 → 80 [ACK] Seq=1251 Win=0 Len=0 (TCP segment of a reassembled PDU)

Q4) i) The first segment containing the HTTP Post was sent at this time: 02:00:18.708842.

ii) The first ack was received at Time : 02:00:19.077793

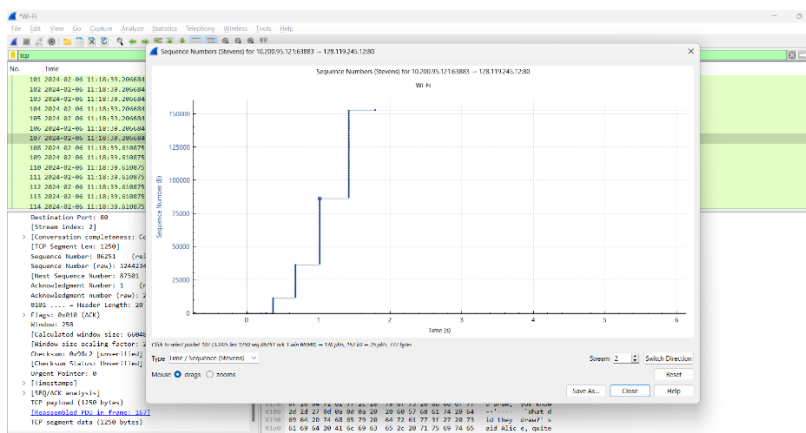
iii) The RTT for the first Data containing segment is: 0.368951000s

iv) The RTT for the second Data containing segment is: 0.368951000s

Q5) First 5 TCP segments have payload = 1250bytes + headers = 20bytes so total length of 1270bytes.

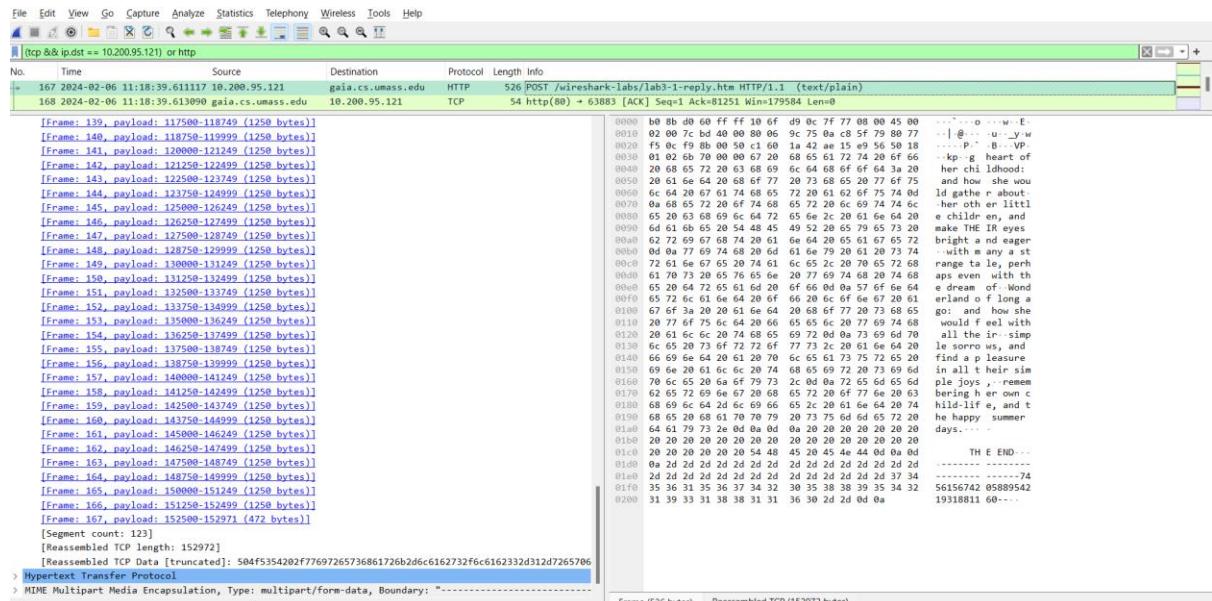
Q6) The minimum window size ranges from 5840 to 6870 bytes of data. The receiver window grows from min to max in buffer. The sender is never throttled due to lacking of receiver buffer space by inspecting this trace. If there is retransmission then it should display In info window and also plot the packet numbers. Also the packet number are monotonically increasing order.

Q7) There are no retransmitted segments in the trace file, this can be concluded after analyzing from the TimeSequence Graph plot using **Statistics > TCP StreamGraph > Time-Sequence Graph (tcptrace)**. We can observe that the packet number continuously increases, if there was any retransmission of packet trend could not be observed.



Q8) The data acknowledge can be obtained by difference in the Ack numbers.

I have received total of 123 segments from 26 to 167, therefore data transferred equal to 3750 bytes. In my case the receiver is acking every other segment. The difference in consecutives acks send by receiver is different.



Q9) Average throughput is the total payload over the entire session divided by the total time. Total time is calculated by taking the difference in timestamps between the first and last packet.

Apply http filter select follow TCP stream then find the first packet which includes the POST. To find the last packet search in trace for [ACK,FIN] after OK is received get the sequence number and time of last packet

First TCP Segment Ack number = 1

First TCP Segment Time = 11:18:38.460210

Last TCP Segment Ack number = 152974

Last TCP Segment Time = 11:18:44.932829

Throughput = (152974-1)bytes/(6.784199)seconds = 23.66 KBytes/sec

Part4

- 1) The slow start is in the beginning of the packet transfer until 2.69s then as the packets are received successfully at the other end the window size increases exponentially. It should increase until congestion but there is no congestion observed here. Therefore it end phase is when all the data is completely transmitted.

