Assignment 3

TCP Basics

1.*What are the first and last packets for the POST request?*

200-202

199	5.297341	192.168.1.102	128.119.245.12	HTTP	104	POST /ethereal-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
200	5.389471	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=162309 Win=62780 Len=0
201	5.447887	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=164041 Win=62780 Len=0
202	5.455830	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=164091 Win=62780 Len=0
203	5.461175	128.119.245.12	192.168.1.102	HTTP	784	HTTP/1.1 200 OK (text/html)

2.*What is the IP address and the TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?*

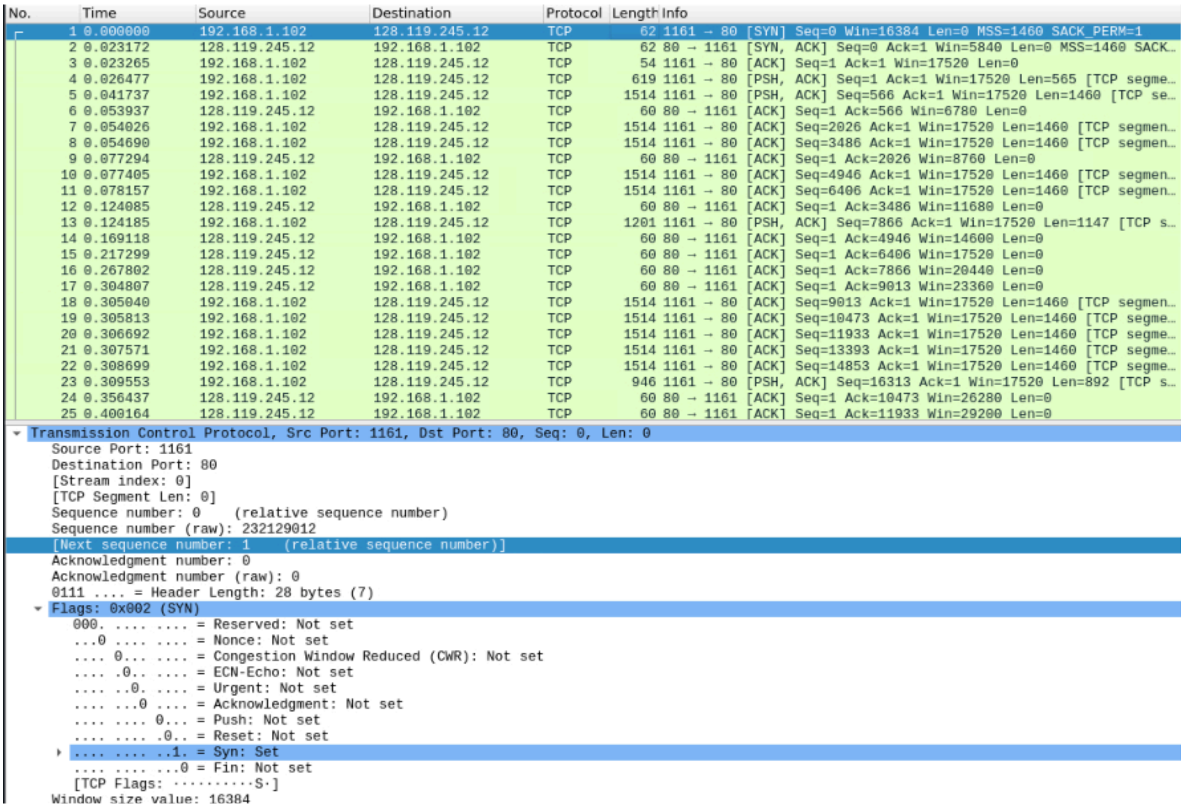
192.168.1.102, port 1161

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

128.119.245.12, port 80

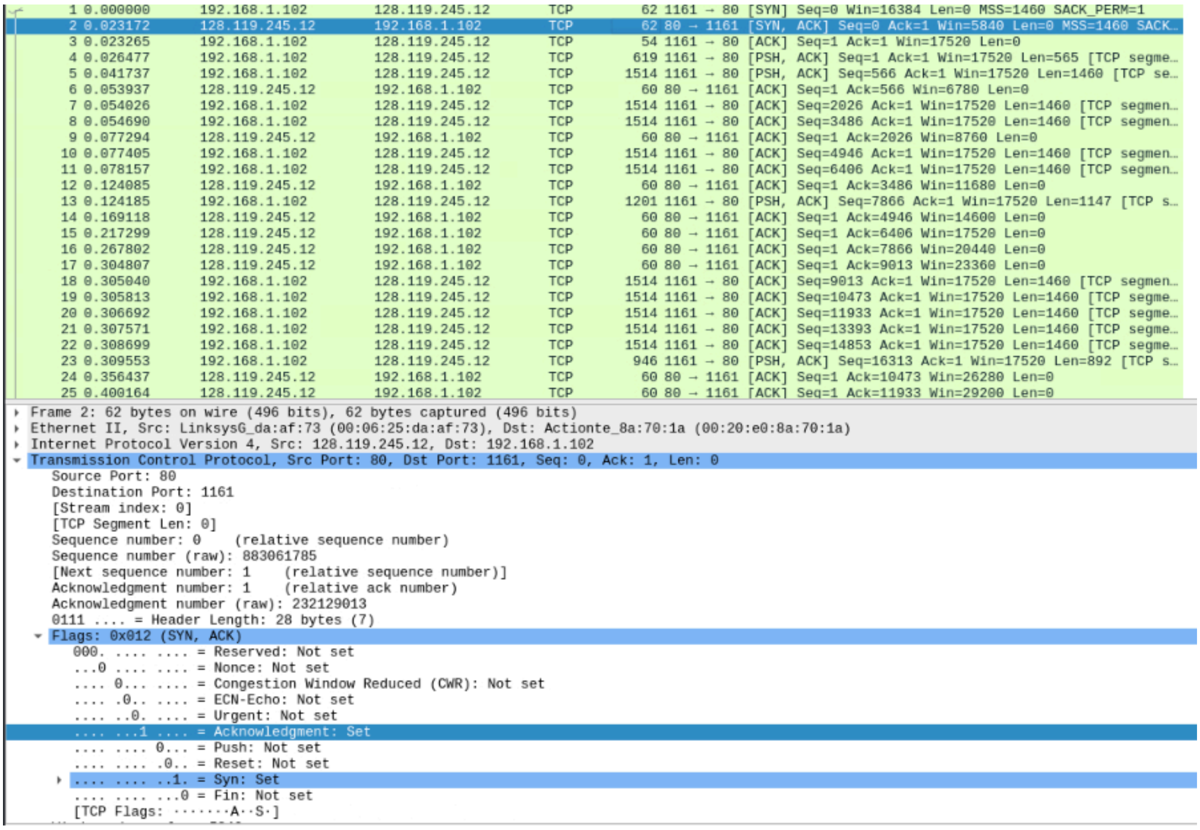
4.*What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?*

The sequence number is 0, we can see in the flags section under TCP that syn is set and the others are not which indicates that it is a SYN segment



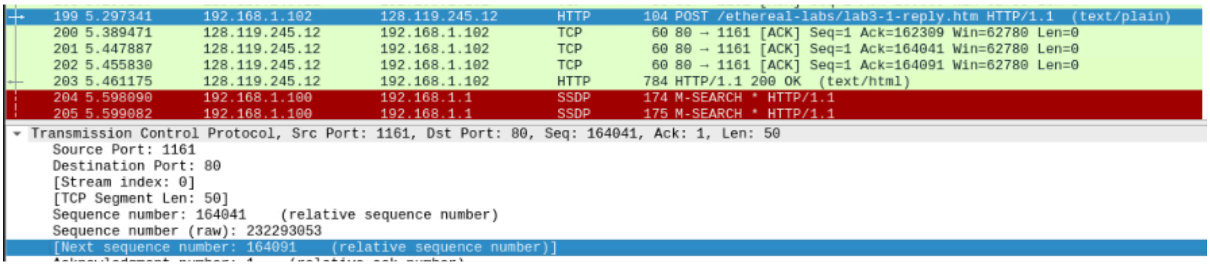
5.*What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN? What is the value of the ACKnowledgement field in the SYNACK segment? How did gaia.cs.umass.edu determine that value? What is it in the segment that identifies the segment as a SYNACK segment?*

The sequence number here is also 0, ACKnowledgement is set to 1. gaia.cs.umass.edu determined the value by incrementing the sequence number of the initial SYN segment. We can determine by looking at the flags, there we can see that both ACKnowledgement and SYN are set.



6.*What is the sequence number of the TCP segment containing the HTTP POST command?*

The sequence number is 164041.



7.Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments? What is the EstimatedRTT value (see Section 3.5.3, page 269 in text) after the receipt of each ACK? Assume that the value of the EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation on page 270 for all subsequent segments.

Between packet 4 and 11, (4,5,7,8,10,11).

Segment nr	Sent Time	Ack	RTT
1	0.026477	0.053937	0.02746
2	0.041737	0.077294	0.035557
3	0.054026	0.124085	0.070059
4	0.054690	0.169118	0.11443
5	0.077405	0.217299	0.13989
6	0.078157	0.267802	0.18964

Estimated-RTT:

*Estimated RTT = 0.875 * Estimated RTT + 0.125 * Sample RTT*

EstimatedRTT after the receipt of the ACK of segment 1:

EstimatedRTT = RTT for Segment 1 = 0.02746 second

Estimated RTT after the receipt of the ACK of segment 2:

Estimated RTT = 0.875 * 0.02746 + 0.125 * 0.035557 = 0.0285

Estimated RTT after the receipt of the ACK of segment 3:

Estimated RTT = 0.875 * 0.0285 + 0.125 * 0.070059 = 0.0337

Estimated RTT after the receipt of the ACK of segment 4:

Estimated RTT = 0.875 * 0.0337+ 0.125 * 0.11443 = 0.0438

Estimated RTT after the receipt of the ACK of segment 5:

Estimated RTT = 0.875 * 0.0438 + 0.125 * 0.13989 = 0.0558

EstimatedRTT after the receipt of the ACK of segment 6:

Estimated RTT = 0.875 * 0.0558 + 0.125 * 0.18964 = 0.0725 second

8.*What is the length of each of the first six TCP segments?*

565 for the first, 1460 for the rest.

9.*What is the minimum amount of available buffer space advertised at the receiver for the entire trace? Does the lack of receiver buffer space ever throttle the sender?*

We can look at the SYN, ACK message and there we can see that the window is set to the minimum of 5840 bytes, the receiver window grows steadily until a maximum receiver buffer size of 62780 bytes. The sender is never throttled due to the lack of receiver buffer space by inspection of the trace.

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

We checked in the sequence number graph, there we could see that the sequence number steadily increased. If there were any retransmitted segments in the trace file, the sequence number would decrease or be lower than the previous sequence number.

11.*How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment (see Table 3.2 on page 278 in the text).*

Typically 1460. We subtracted the previous acknowledgement number from the current and found that there were some times where the receiver was ACKing every other received segment. Segment 87 and 88 for example (64005-61085=2920 | 2920/2=1460). We could also see that segment 52 is aching two packets by viewing the ACK analysis for the segment

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

The throughput can be calculated by getting the ratio between the total amount of data and the total transmission time.

Total amount of data transmitted is 164091 - 1 = 164090 bytes. We can see this by viewing packet number 4 and number 202, the first TCP segment and the last ACK.

Total transmission time is the time difference of the first TCP segment, packet 4 and the last ACK, packet 202. 5.455830 - 0.026477 = 5.4294 seconds.

So the throughput is 164090/5.4294 = 30222,4924 byte per second.

Task A observations:

We learned by viewing in the source and destination the IP address that were sending and receiving, we could also see by inspection the info that the port numbers from the source to the destination port. We also inspected and learned that the three initial messages were the three way handshake to initiate the connection. Further we found the sequence numbers for every segment by viewing the sequence number field in the Transport layer section.

RTT(Round Trip Time) is a measure of how fast we can send packages, the bigger RTT is the longer it takes to get an ACK. Packet loss happens when the sender's window is bigger than the receiver's window. If the connection loses packets it will retransmit the lost packets and change the size of the window to match the receiver's.

TCP Congestion Control in Action

13.*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. Can you identify if and where TCP's slow start phase begins and ends, as well as if and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.*

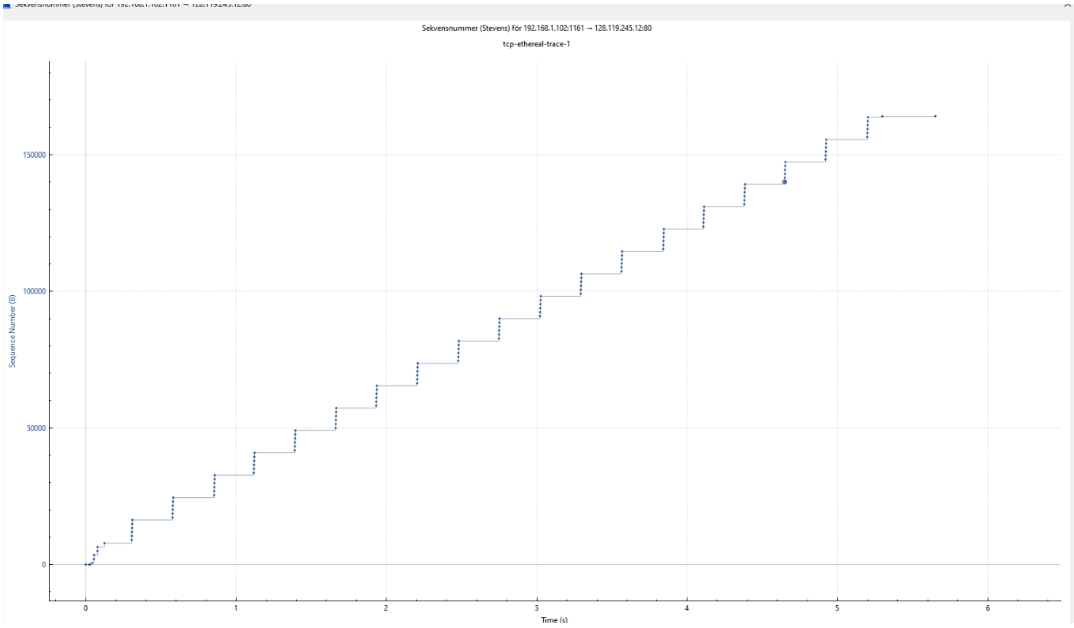


Table 1

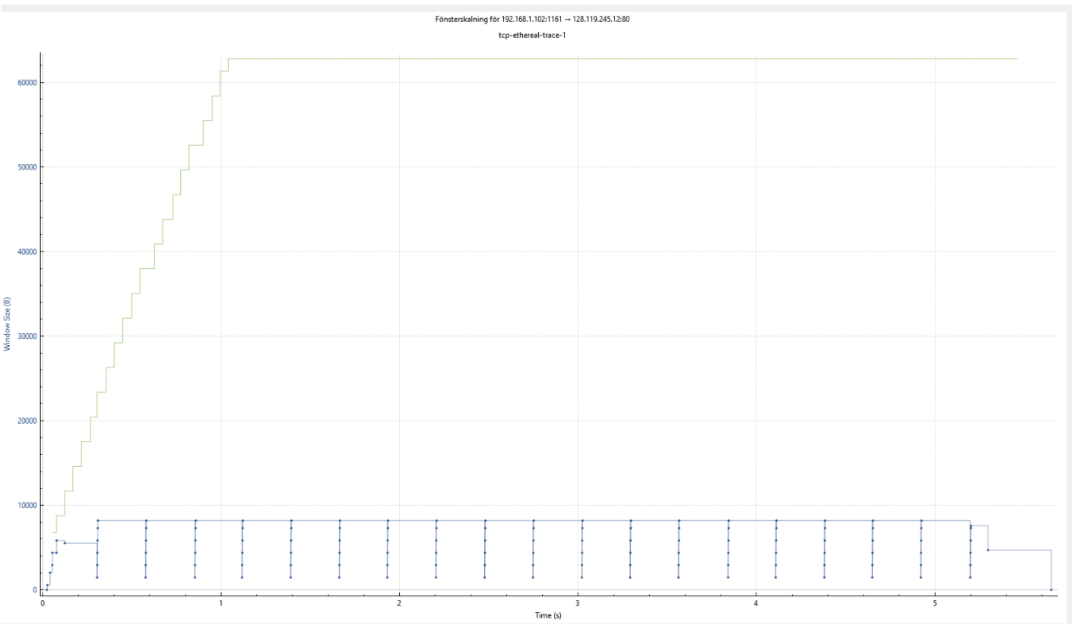


Table 2

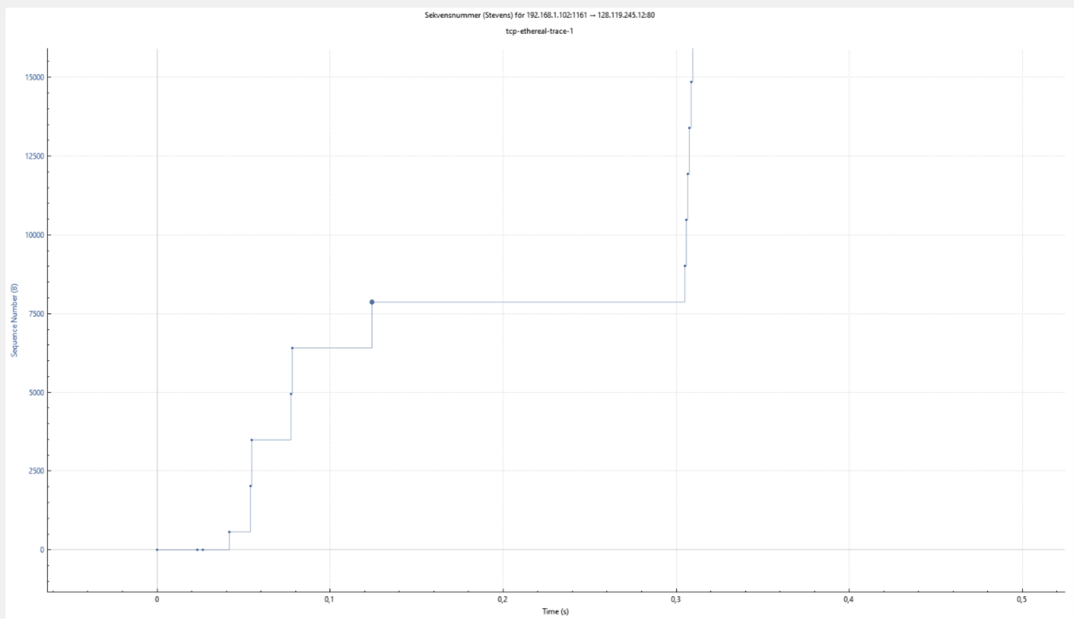


Table 3

We can initially see the TCP slow start as it begins with a slow stream of packets, it then goes over to a larger and larger bursts of packets being sent and it's there that TCP slow start ends, so it ends around packet 13 with sequence number 7866. We can also see that it's there that congestion control takes over.

14.*Explain the relationship between (i) the congestion window (cwnd), (ii) the receiver advertised window (rwnd), (iii) the number of unacknowledged bytes, and (iv) the effective window at the sender (i.e., the window effectively limiting the data transmission).*

Cwnd are how many packets are sent simultaneously and the rwnd is the receiver explaining how many packets it can receive at once. The cwnd is increased exponentially until the rwnd is reached. The maximum number of unacknowledged bytes is two, either the buffer size of the sender or the rwnd. The effective window is instead the minimum value of cwnd and rwnd.

15.*Is it* ***generally*** *possible to find the congestion window size (cwnd) and how it changes with time, from the captured trace files? If so, please explain how. If not, please explain when and when not. Motivate your answer and give examples.*

```
[SEQ/ACK analysis]
    [iRTT: 0.023265000 seconds]
    [Bytes in flight: 2025]
    [Bytes sent since last PSH flag: 1460]
TCP payload (1460 bytes)
[Reassembled PDU in frame: 199]
```

You can generally find the cwnd by inspecting the amount of sent packages before a receive. But doing this correctly can be difficult because it can be hard keeping track of all the packages that were supposed to be sent before an ack was to be received. But we can view the cwnd window increase in the TCP slow start as shown clearly in table 2 and then stay the same size until the end. So by looking at the Sequence number graph (Stevens) we can see 6 packets being able to be sent in one cwnd.

A Short Study of TCP Fairness

16.*What is the throughput of each of the connections in bps (bits per second)? What is the total bandwidth of the host on which the clients are running? Discuss the TCP fairness for this case.*

$$Average\ throughput\ of\ a\ connection = \frac{1.22 \cdot MSS}{RTT\sqrt{L}}$$

Typical MSS for ethernet connections is 1460.

RTT is 12 milliseconds.

We assume an acceptable packet loss rate of 1%.

Connection	Real throughput (Transferred bytes/Duration * 8)	Ideal throughput
1	2 535 059	1 189 067
2	2 546 530	1 189 067
3	2 575 234	1 189 067
4	2 506 499	1 189 067

Total bandwidth is 10 163 322.

We think the TCP fairness is pretty fair because the RTT is the same for all the connections and the real throughputs are quite similar, the ratio between the real and ideal throughputs are very similar as well.

17.*What is the throughput of each of the connections in bps (bits per second)? What is the total bandwidth of the host on which the clients are running? Discuss the TCP fairness for this case.*

Connections	Real throughput (Transferred bytes/Duration * 8)	Ideal throughput
1	23 228 367	137 015
2	15 644 074	50 891
3	13 501 737	26 194
4	12 478 984	24 400
5	9 654 285	36 351
6	6 279 528	53 976
7	5 843 994	13 194
8	3 841 145	5 464
9	3 486 447	5 532

We think that this example isn’t fair, because the throughputs are not very similar. And the ratio between the real and ideal throughputs are not all very similar.

18.*Discuss the TCP fairness for this case. How does it differ from the previous cases, and how is it affected by the use of BitTorrent?*

Connections	Real throughput (Transferred bytes/Duration * 8)	Ideal throughput
1	15 013 949	3 546 800
2	1 559 235	3 976 080
3	1 093 803	1 424 960
4	1 103 449	2 103 524
5	930 209	4 621 936
6	877 398	4 321 937
7	849 341	1 167 511
8	692 473	980 000
9	673 479	1 922 184
10	659 145	2 163 393

By looking at the table above we can see that the TCP fairness isn't good in this case either, however because it is a bittorrent and you can both upload and download files it is difficult to do anything about it. Taken into account is also the RTT which varies heavily between the connections.