Task A:

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

   Nummer 4 is first and 199 last

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

   IP: 192.168.1.102 src: 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?

   IP: 128.119.245.12 dst 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?

   Sequence number: 232129012. The SYN flag is set

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?

   Sequence number: 883061785. The SYN and ACK flags are set, and the Acknowledgment number corresponds to adding 1 to the sequence number of the initial SYN segment.

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

   Sequence number: 232129013

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.

Segment 1: Seq# 232129013, sent: 0.026477, ack: 0.053937, RTT: 0.02746 s, eRTT: 0.02746 s

Segment 2: Seq# 232129578, sent: 0.041737, ack: 0.077294, RTT: 0.035557 s, eRTT: 0.028472125 s

Segment 3: Seq# 232131038, sent: 0.054026, ack: 0.124085, RTT: 0.070059 s, eRTT: 0.0336704844 s

Segment 4: Seq# 232132498, sent: 0.054690, ack: 0.169118, RTT: 0.114428 s, eRTT: 0.0437651738 s

Segment 5: Seq# 232133958, sent: 0.077405, ack: 0.217299, RTT: 0.139894 s, eRTT: 0.0557812771 s

Segment 6: Seq# 232135418, sent: 0.078157, ack: 0.267802, RTT: 0.189645 s, eRTT: 0.0725142425 s

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

Seq# 1 (rel), length 0

Seq# 2 (rel), length 0

Seq# 3 (rel), length 0

Seq# 4 (rel), length 565

Seq# 5 (rel), length 1460

Seq#6 (rel), length 0

Seq#7 (rel), length 1460

Seq#8 (rel), length 1460

Seq#9 (rel), length 0

Seq#10 (rel), length 1460

Seq#11 (rel), length 1460

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?

   The minimum available buffer space is 5840 bytes, as can be seen in the first ACK from the server (relative sequence number 2, real sequence number 883061785) under Window size value. The receiver window grows steadily until it reaches 62780 bytes, as seen in the last ACK from the server (relative sequence number 198, real sequence number 883061786).

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

    There are no retransmitted segments. As can be seen in the Time Sequence Graph (Stevens)

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).

    The receiver typically acknowledges 1460 bytes per ACK, which is the maximum TCP segment size. The ACKs are identifiable by their length of 0, and that they transmit from the server's port to the client's port.

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

    By looking at the TCP Throughput graph in Wireshark we can determine that the average throughput is 223000 bits/s, or 27875 bytes/s.

Conclusion Task A:

At a high level, a standard TCP handshake (SYN,SYN/ACK,ACK) is performed. The data is sent in smaller packets (1460 bytes, size defined in the first SYN of the handshake), and every so often (every 6 packets in this case) the data sets the PSH flag which sends the partial data up to the application layer. The last packet in each cycle is a rest packet and might contain less than the maximum allowed data. All packets sent before the PSH need to be ACK:ed before sending a new wave of packets. When all tcp-packets have been ACK:ed the transmission is complete.

Packet losses are handled by simply resending the lost packet. This is done in the way that we use the ACKs sent on a package to notice lost packages. If we send 5 packages and one is missing (by receiving one less ACK than we should have) we can discover this and send it back again to recoup the loss. The RTT is based on how many ACKs are sent. From the start they

are sent individually and when the connection is considered stable enough, the server will begin to send multiple ACKs at the same time.

Task B:

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 server (Figure 2a and Figure 2b). For each of the two traces, can you identify where TCP's *slow start* phase begins and ends, and where *congestion avoidance* takes over? If you can, explain how. If not, explain why not. To better identify these phases, you may need to find the number of unacknowledged packets (or bytes). You can look at different graphs, such as Time Sequence (tcptrace). Note that the number of unacknowledged packets at different times can be found by comparing the number of packets that have been sent with the number of packets that have been acknowledged. Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.

    According to the Time Sequence (Stevens) Graph, the TCP slow start phase begins with packet 1, and ends at packet 13. After this the congestion control phase takes over, as can be seen by the long plateaus in the graph. More definitively, the congestion control phase takes over when the congestion window exceeds the slow-start threshold (*sstresh* changes depending on implementation and connection).

14. Explain the relationship between (i) the congestion window, (ii) the receiver advertised window, (iii) the number of unacknowledged bytes, and (iv) the effective window at the sender.

    (i) The Congestion Window (cwnd): the cwnd starts out small, and increases incrementally as long as there are no packet losses. The cwnd will be reduced in size if a packet loss is detected, how much the size is reduced is based on the specific implementation.

    (ii)The receiver advertised window is the max amount of bytes the client can buffer. This increases gradually, when the limit is capped an ACK must be sent to inform.

(iii) The number of unacknowledged bytes: Any packet that doesn't get acknowledged by the receiver must be retransmitted by the sender

(iv) The effective window is the least amount of data we can receive from each ACK. Based on the advertised window and the congestion window.

The relationship between the four is that they are all used to make the central TCP Congestion Control algorithm.

15. Is it **generally** possible to find the congestion window size (i.e. *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. Your answer may also benefit from trying to describe and discuss your answer in the context of the two prior questions, for example.

    No, when we study the graph (Stevens) we can clearly see that the congestion window always has 5 packages before the ACK. This follows through the whole graph until the fin.

Task C: Please carefully answer and discuss questions 16-18 as outlined in this section.

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.

Another case to consider is downloading the same file from different mirror servers around the world. The following table lists the details of each of the connections:

Connection 1: 2 475.64 Kb/s

Connection 2: 2 486.84 Kb/s

Connection 3: 2 514.87 Kb/s

Connection 4: 2 490.78 Kb/s

TCP fairness, looking at the RTT of each connection, we can draw the conclusion that it is distributed fairly on 12 mil sec each.

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.

The final case to consider is a BitTorrent download from multiple peers. Similar to the previous cases, the details of each of the connections is presented in the following table. This time only ten of the connections are presented.

(TTB*8)/Duration/1024 = Convert bytes to bits, divide by duration, convert to kibibits

Connection 1: 22 683.95 Kb/s

Connection 2: 15 277.41 Kb/s

Connection 3: 13 185.29 Kb/s

Connection 4: 12 186.50 Kb/s

Connection 5: 9 428,01 Kb/s

Connection 6: 6132.35 Kb/s

Connection 7: 5707.02 Kb/s

Connection 8: 3 751.11 Kb/s

Connection 9: 3404.73 Kb/s

TCP fairness, looking at the RTT of each connection, we can draw the conclusion that they are not fairly distributed, and therefore cannot be said to have TCP fairness.

18. Discuss the TCP fairness for this case. For all of these questions you must take a closer look at the relationships between the characteristics of the different connections and discuss your findings in the context of the different experiments. You are expected to show that you understand the concept of TCP fairness and how the different scenarios may impact the throughput relationships that you observe and those that you may expect in general. To help the discussion you may for example want to create a scatter plot that show the estimated round trip time (RTT) and throughput against each other (for the different connections). You also want to carefully examine and discuss the above throughput equation and how it may apply to each scenario.

Connection 1: 14 662.06 Kb/s

Connection 2: 12 181.53 Kb/s

Connection 3: 8 545.33 Kb/s

Connection 4: 8 620.69 Kb/s

Connection 5: 7 267.26 Kb/s

Connection 6: 6854.67 Kb/s

Connection 7: 6635.47 Kb/s

Connection 8: 5417.75 Kb/s

Connection 9: 5261.55 Kb/s

Connection 10: 5149.56 Kb/s

As we analyse the TCP fairness in the three cases we clearly can see that it is not always fair. A lot of external factors can change the end result. How big is the RTT for example? If it is low we might not get enough throughput to grant fairness. Hardware limitations can be a big factor and even how many people are on the link can change the measure.