

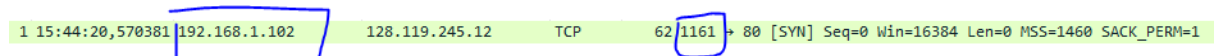
1. What are the first and last packets for the POST request?

First and last packet is a (PSH,ACK) message. First one indicates that the client has no further data to add and the last one is an indication that no further data is going to be transmitted.

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?

IP : 192.168.1.102

Port : 1161



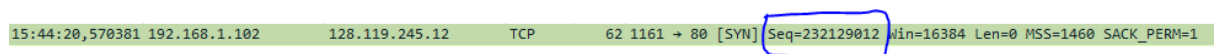
1 15:44:20,570381 192.168.1.102 128.119.245.12 TCP 62 1161 → 80 [SYN] Seq=0 Win=16384 Len=0 MSS=1460 SACK_PERM=1

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

Port: 80 (Look at picture in question 2)

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?



15:44:20,570381 192.168.1.102 128.119.245.12 TCP 62 1161 → 80 [SYN] Seq=232129012 Win=16384 Len=0 MSS=1460 SACK_PERM=1

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?

```
62 1161 → 80 [SYN] Seq=232129012 Win=16384 Len=0 MSS=1460 SACK_PERM=1
62 80 → 1161 [SYN, ACK] Seq=883061785 Ack=232129013 Win=5840 Len=0 MSS=1460 SACK_PERM=1
```

The value is the same as the SYN request's seq-number but incremented by 1.

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

```
Sequence Number: 883061786
```

```
.....P... ..4..t..P..
Dp.....PO ST /ethe
real-lab s/lab3-1
```

- 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.**

	Seq-number	Sent time	ACK received time	RTT
Segment 1 (4)	232129013	0.026477	0.053937	0.02746
Segment 2 (5)	232129578	0.041737	0.077294	0.035557
Segment 3 (7)	232131038	0.054026	0.124085	0.070059
Segment 4 (8)	232132498	0.054690	0.169118	0.11443
Segment5(10)	232133958	0.077405	0.217299	0.13989
Segment6(11)	232135418	0.078157	0.267802	0.18964

EstimatedRTT = $0.875 * \text{EstimatedRTT} + 0.125 * \text{SampleRTT}$ EstimatedRTT
after the receipt of the

ACK of segment 1: EstimatedRTT = RTT for Segment 1 = 0.02746

ACK of segment 2: EstimatedRTT = $0.875 * 0.02746 + 0.125 * 0.035557 = 0.0285$

ACK of segment 3: EstimatedRTT = $0.875 * 0.0285 + 0.125 * 0.070059 = 0.0337$

ACK of segment 4: EstimatedRTT = $0.875 * 0.0337 + 0.125 * 0.11443 = 0.0438$

segment 5: EstimatedRTT = $0.875 * 0.0438 + 0.125 * 0.13989 = 0.0558$

segment 6: EstimatedRTT = $0.875 * 0.0558 + 0.125 * 0.18964 = 0.0725$

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

First : 565

Second - sixth : 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?

Window: 5840

The sender is never throttled because we never reach the full capacity of the window.

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

No, no segments were ever retransmitted. This is shown by the fact that an old Sequence/Acknowledgement number was never resent in order to re-request former packets.

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 is typically acking 1460. There are cases where the receiver acks every other segment. This is shown when more than one ack occurs in a row.

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

$164090 / 5.429353 = 30.222 \text{ KByte/sec.}$

The throughput can be calculated by using the value of the last ack (164091) - (1) the first sequence number, divided by the time since the first frame ($5.455830 - 0.026477$) = 30.222 KByte/sec.

Task A:

High-level:

The first step when uploading a file is to open a connection. This is done by an initial 3-way handshake between the client (sending to port 80) and the server (sending to port 1161 in this case). Afterwards, the client starts sending the file. This is done by separating the file in TCP segments. If a segment has been received by the server, the server sends an "ACK"-response acknowledging the client that the segment has been successfully received. If the client doesn't get an ACK-response, the client sends the same segment again so that the whole file is guaranteed to arrive.

Low-level:

After the client has sent a TCP-segment and is waiting for the ACK-response, the client starts a timer called "Retransmission Timeout"(RTO). If the ACK-response hasn't arrived until the end of the timer it retransmits the segment. This RTO can impact the RTT. Each time the same segment is sent, it doubles the RTO. This creates an exponential increase in the SampleRTT (essentially the time it takes to receive the ACK-response after sending). This then increases the EstimatedRTT which explains why our EstimatedRTT values are not only increasing, but also exponentially.

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.

Slow start starts at 1 at 0.023265s and ends at 7866 seq-number at 0.124185s.
Congestion avoidance takes over at 7866. Jagged and more uneven.

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).

i) Congestion window starts out small and increases with every packet sent. When a loss is detected this value is cut in half and set as `ssthresh`. This value is for the sender to set the threshold for the slow-start.

ii) The receiver advertised window is the maximum size able to be received by the receiver before waiting for ACK's is needed.

iii) The number of unacknowledged bytes is the number of bytes the sender can send without needing an ACK. This value is limited by the congestion window and the receiver window, since both are a kind of threshold for 'reliable' sending.

iv) The effective window at the sender is a value specifying the amount of bytes that the sender can send. It depends on the number of unacknowledged bytes.

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.

No, but if there are packet losses the value could be estimated by using the threshold value. Otherwise, this is a value kept local by the sender.

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.

$$1 \ (165095720/521) * 8 = 2\ 535\ 059 \text{ bps}$$

$$2 \ (165842766/521) * 8 = 2\ 546\ 530 \text{ bps}$$

$$3 \ (165458792/514) * 8 = 2\ 575\ 234 \text{ bps}$$

$$4 \ (163235772/512) * 8 = 2\ 550\ 559 \text{ bps}$$

The total bandwidth of the host in this case, is how many bits the host sent over the connections in this segment of time.

$$2535059 + 2546530 + 2575234 + 2550559 = 10\ 207\ 382 \text{ bps}$$

There is a slight difference in the results for each user. This could be because of dropped packages or differences in how their different clients work. The parameters are already as fair as they can get.

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.

$$1. \ (261319130/90) * 8 = 23\ 228\ 367 \text{ bps}$$

2. $(175995832/90)*8 = 15\,644\,074$ bps

3. $(151894552/90)*8 = 13\,501\,738$ bps

4. $(140388568/90)*8 = 12\,478\,984$ bps

5. $(108610702/90)*8 = 9\,654\,285$ bps

6. $(70644690/90)*8 = 6\,279\,528$ bps

7. $(65744938/90)*8 = 5\,843\,995$ bps

8. $(43212876/90)*8 = 3\,841\,145$ bps

9. $(39222524/90)*8 = 3\,486\,447$ bps

Host's total bandwidth: 93 958 563 bps.

The connection with the smallest RTT gets the most data during the allotted time. And the two connections with the highest RTT get the least amount of data. There are outliers though, 6 get less data than 2, even though it has a smaller RTT. and 8 gets more than 9 despite having a higher RTT. On the other hand, the difference between 6 and 7 is not as big as the difference in RTT would indicate it should be. All this suggests that the host is deciding that some clients are more equal than others.

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?

bps

1. $(108851134/58)*8 = 15\,013\,950$

2. $(90435681/58)*8 = 12\,473\,887$

3. $(57971584/53)*8 = 8\,750\,428$

4. $(32000012/29)*8 = 8\,827\,590$

5. $(32557334/35)*8 = 7\,441\,676$

6. $(27199361/31)*8 = 7\,019\,190$

7. $(26329578/31)*8 = 6\,794\,730$

8. $(38834490/56)*8 = 5\,547\,784$

9. $(23571761/35)*8 = 5\,387\,831$

10. $(36252962/55)*8 = 5\,273\,158$

Total bandwidth: 82 530 224 bps.

TCP is kinda fair, but certain connections on the unfair end. Connections 4, 9 and 10 with around 65-75 RTT are different from each other throughout, which is unfair.