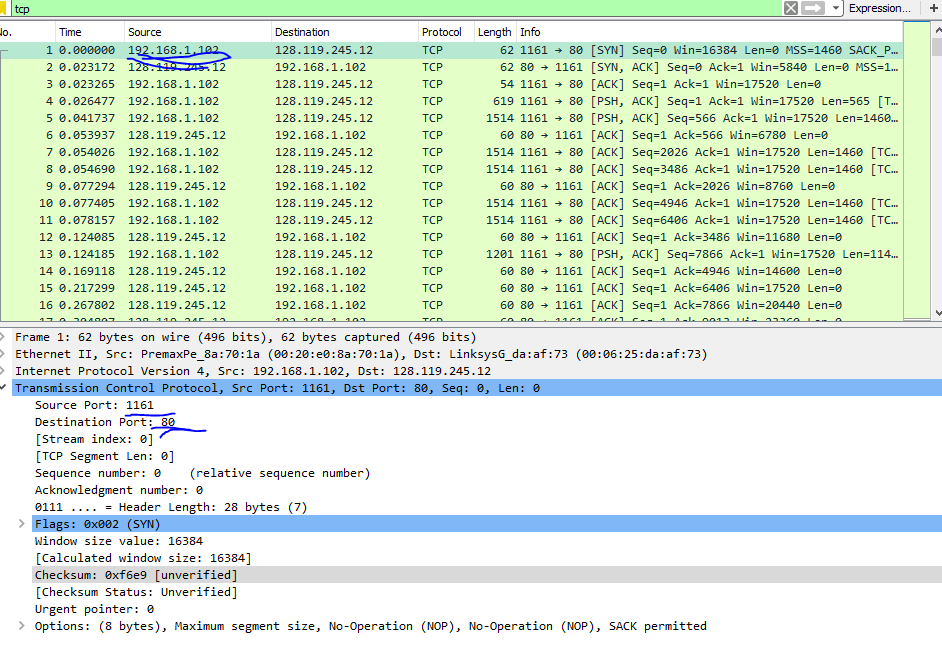
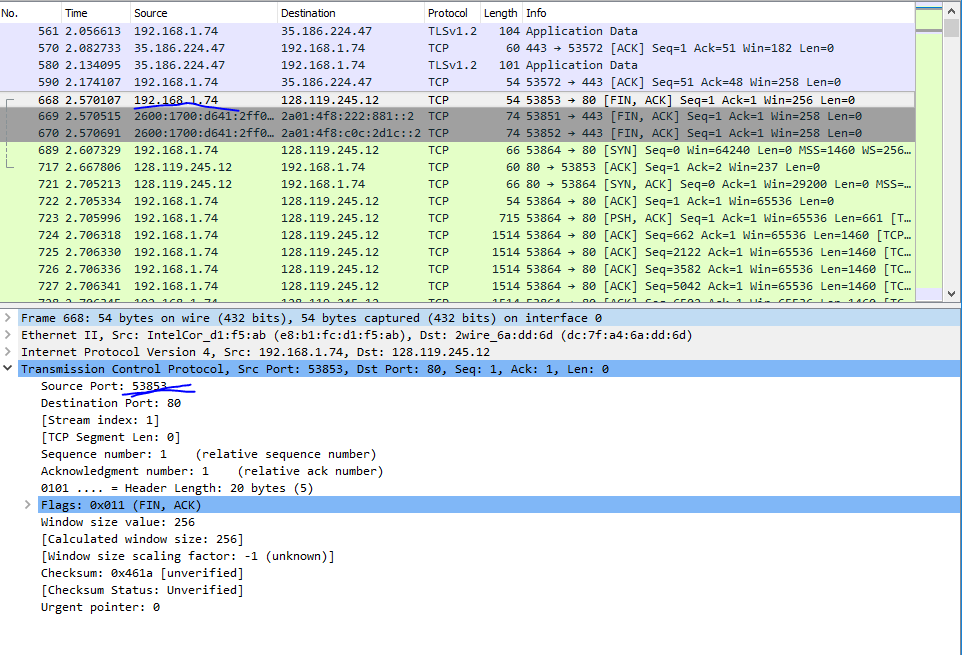
Valeriy Kutsar Wirehark Lab 3

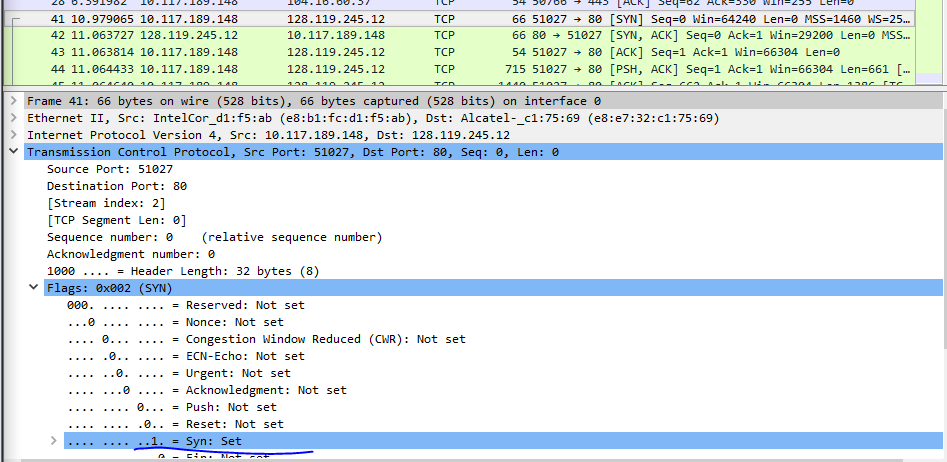
1. IP address is 192.168.1.102 the TCP port number is 1161.
2. IP address of gaia.cs.umass.edu is 128.119.245.12 and it is sending and receiving packets from source number 80.



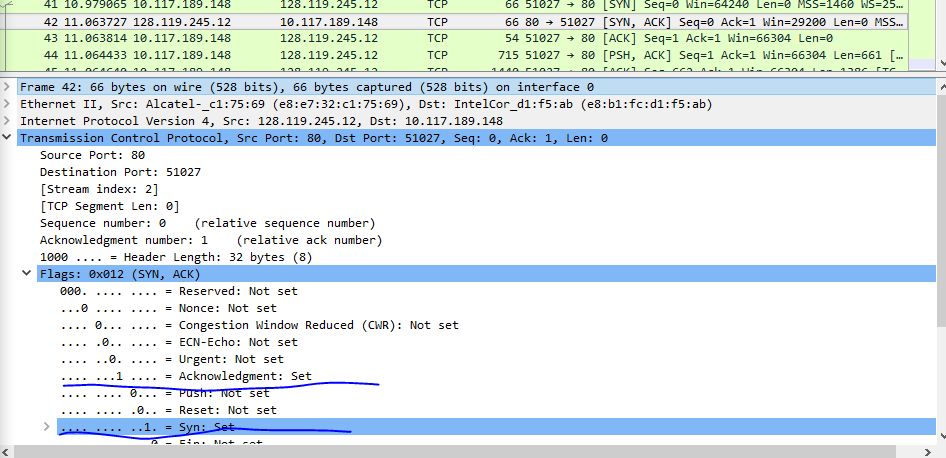
1. From my own trace, my IP address is 192.168.1.74 and the source port number is 53853.



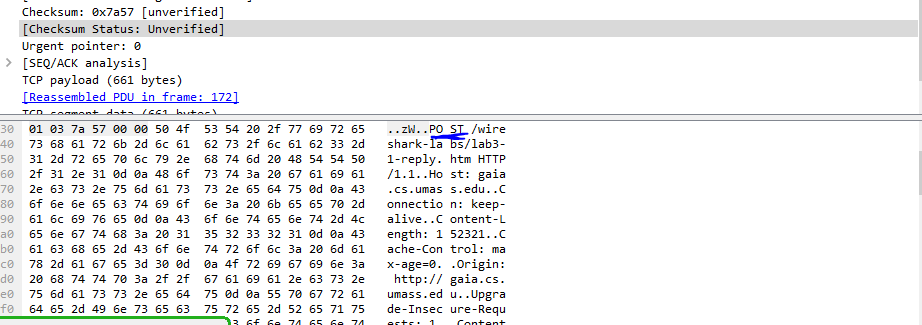
1. The sequence number is 0. The ack number acknowledges the number as part of the SYN segment. The SYN flag is set to 1 and it indicates that this segment is a SYN segment.



1. The sequence number sent from gaia is 0. The acknowledgement value is 1. It determined it by adding 1 to the initial sequence number of SYN segment from the client computer. The SYN flag and ACK flag in the segment are set to 1 and they indicate that this segment is a SYNACK segment.



1. The post has a sequence number of 1.



1. Segment 1 seq #: 1

Segment 2 seq#:662

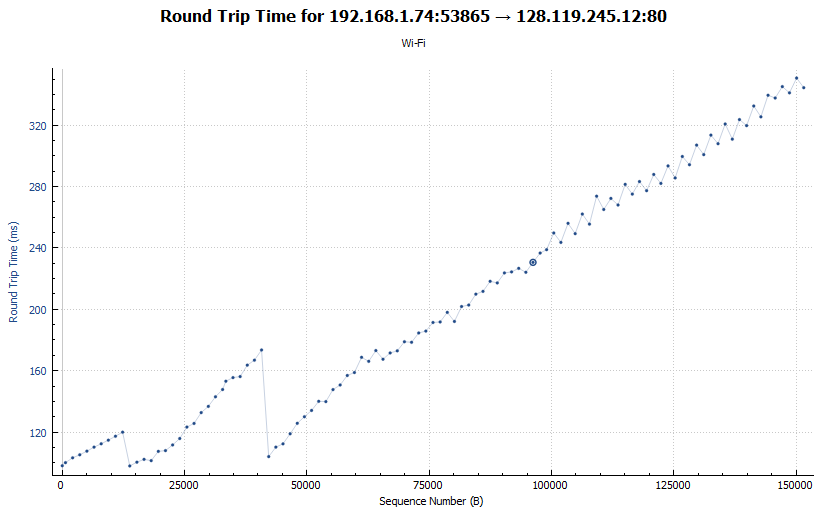
Segment 3 seq#:2122

Segment 4 seq#:3582

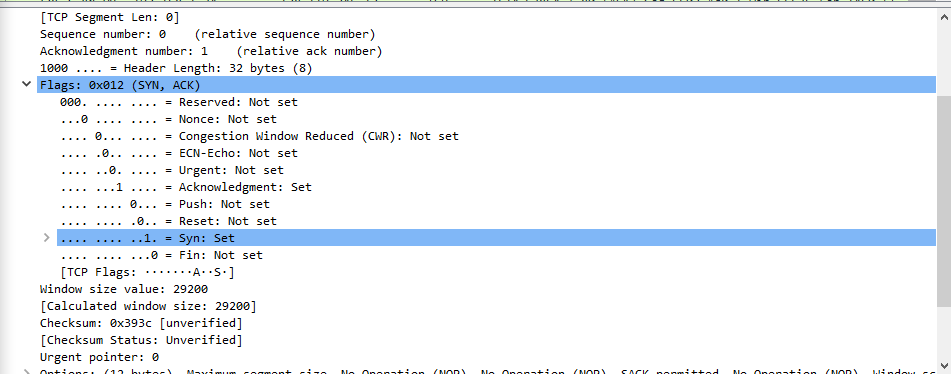
Segment 5 seq#:5042

Segment 6 seq#:6502

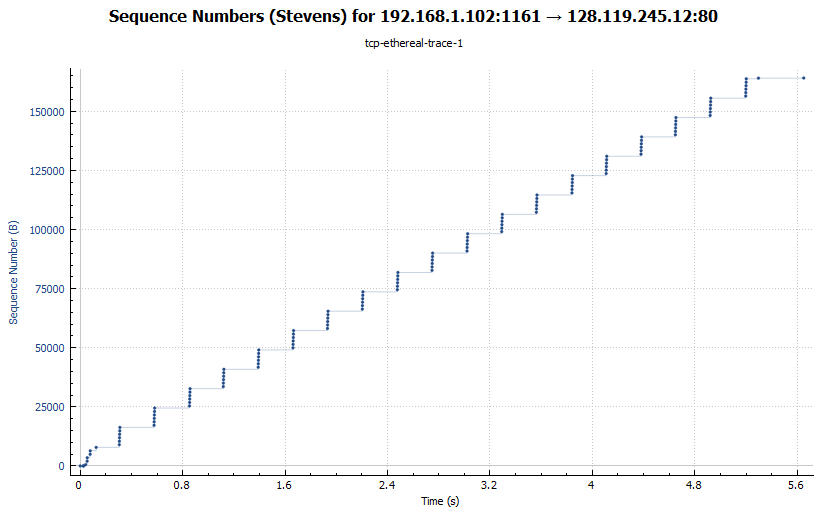
|  |  |  |  |
| --- | --- | --- | --- |
| Segment # | Sent Time | ACK received time | RTT |
| 1 | 2.705996 | 2.759786 | .09528 |
| 2 | 2.706318 | 2.804927 | .098931 |
| 3 | 2.706330 | 2.806412 | .100094 |
| 4 | 2.706336 | 2.809844 | .103514 |
| 5 | 2.706341 | 2.81141 | .105074 |
| 6 | 2.706345 | 2.813538 | .107197 |



1. Length of first TCP segment with POST: 661 bytes. Length of each of the other 5 TCP segments: 1460 bytes.
2. The minimum amount of buffer space(receiver window) advertised at gaia for the entire trace is 29200 bytes, which shows in the first ACK from the server. This receiver window grows steadily until a maximum buffer size of 64768 bytes. The sender is never throttled due to lacking of receiver buffer space by inspecting this trace.



1. No packets we retransmitted because there were not any duplicate ACK’s.
2. The difference between the acknowledged sequence numbers of to consecutive ACK’s indicates the data received by the server between these two ACK’s. By looking at the segments, we can see that some segments acknowledged data with 2 times the regular data.
3. The throughput can be computed as the ratio between the total amount of data and the total transmission time. The total amount data transmitted can be computed by the difference between the sequence number of the first TCP and the acknowledged sequence number of the last ACK(152983-1 = 152982). The whole transmission time is the difference of the time instant of the first TCP segment(3.472449 – 2.705996 = .766456). Therefore the computed throughput for my trace was 152982/.766453) which gives 199597.366 Bytes/sec.
4. From the chart we can see that the amount of data being sent increased in the beginning, but did not exceed a certain limit. We cannot exactly know where the slow start phase and the start of the congestion avoidance phase for this trace because the sender is not sending data aggressively enough for the need to use congestion avoidance. Before it receives the acknowledgement for the whole block of data, the application will not send more data and that shows how before the end of the slow start phase, the application stops sending data for a short time.



Congestion control did not need take over. The TCP sender is not sending data aggressively enough to push to the congestion state. Before the sender receives acknowledgement of the sent data, it stops sending data. The rate of the transfer grows exponentially, and if there was a need for congestion control, there would be drops in the graph, of which there are none in this trace. Same as the wireshark sample graph, the application stopped sending data for a short time before it received the whole block of data, and then continued sending data

