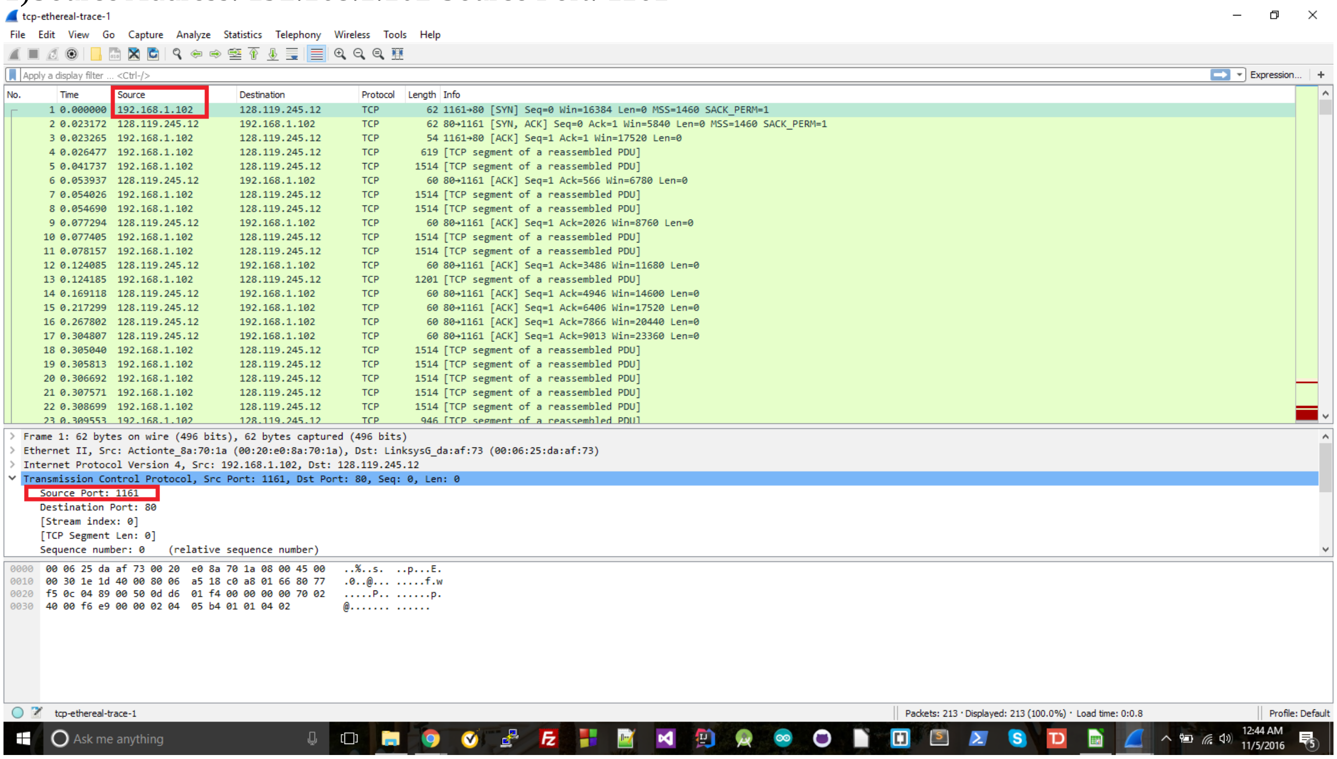
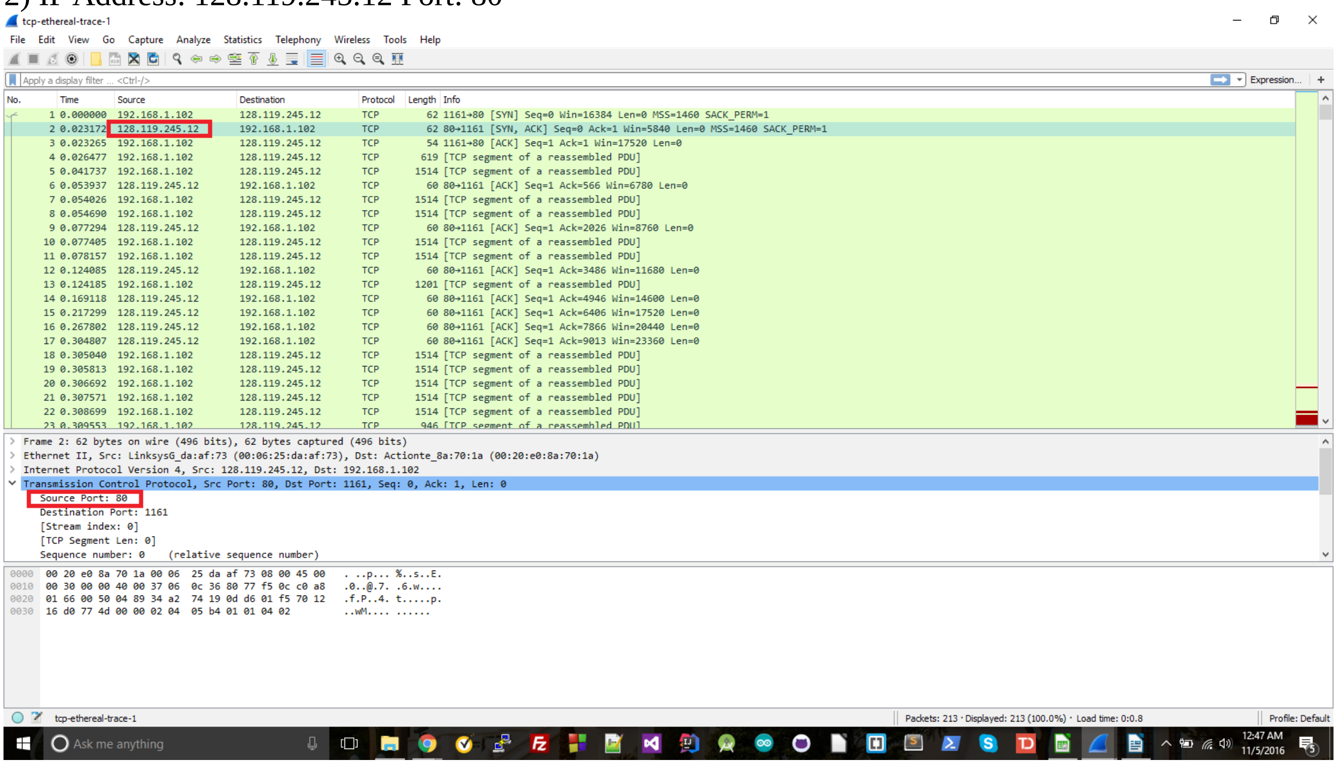
Travis Robinson
CS372
Lab 3
Fall 2016
Oregon State University

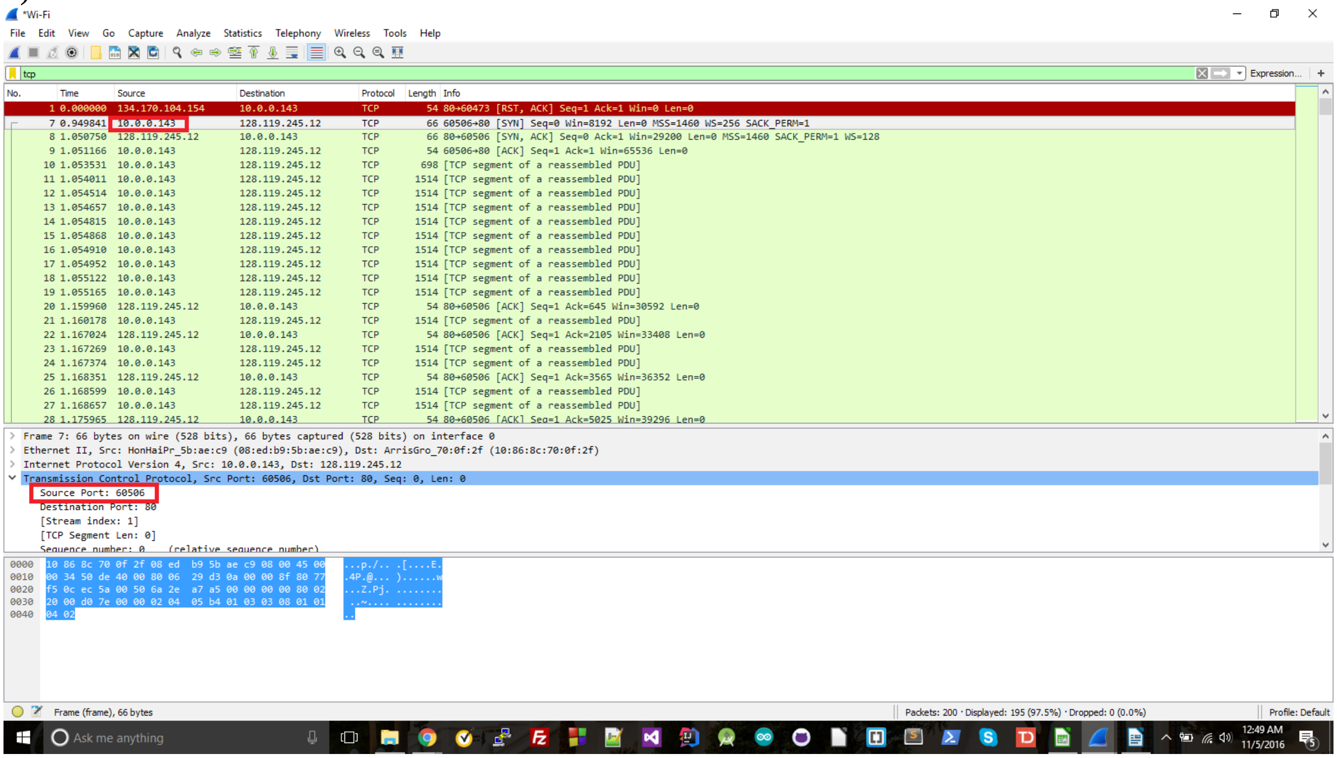
1)Source Address: 192.168.1.102 Source Port: 1161



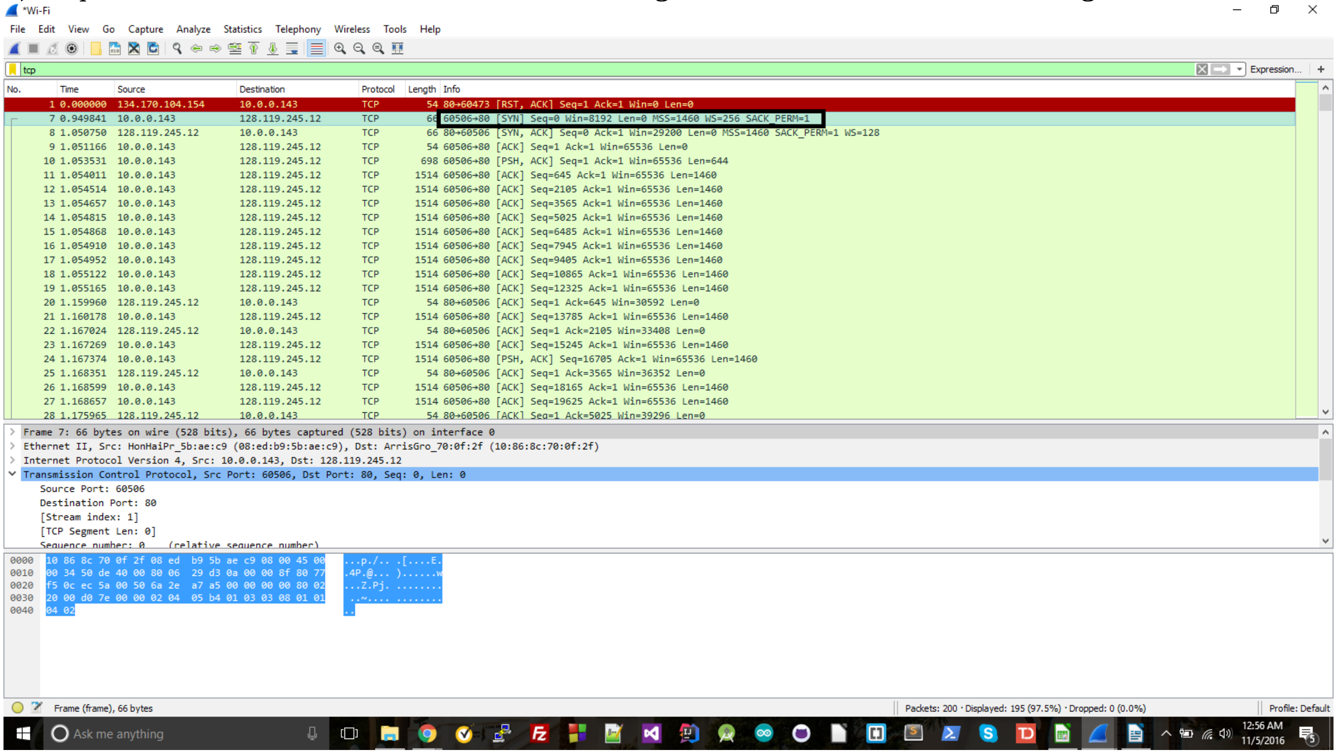
2) IP Address: 128.119.245.12 Port: 80



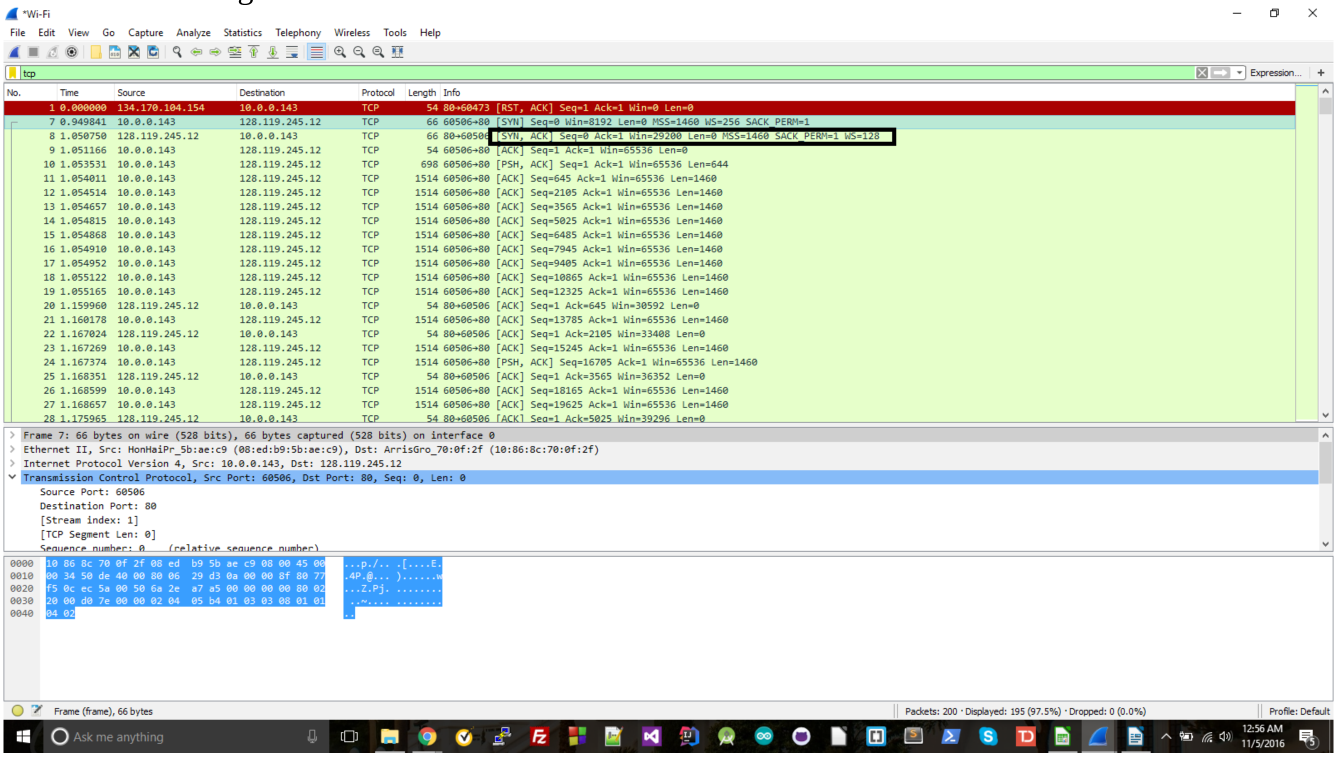
3) IP Address: 10.0.0.143 Port: 60506



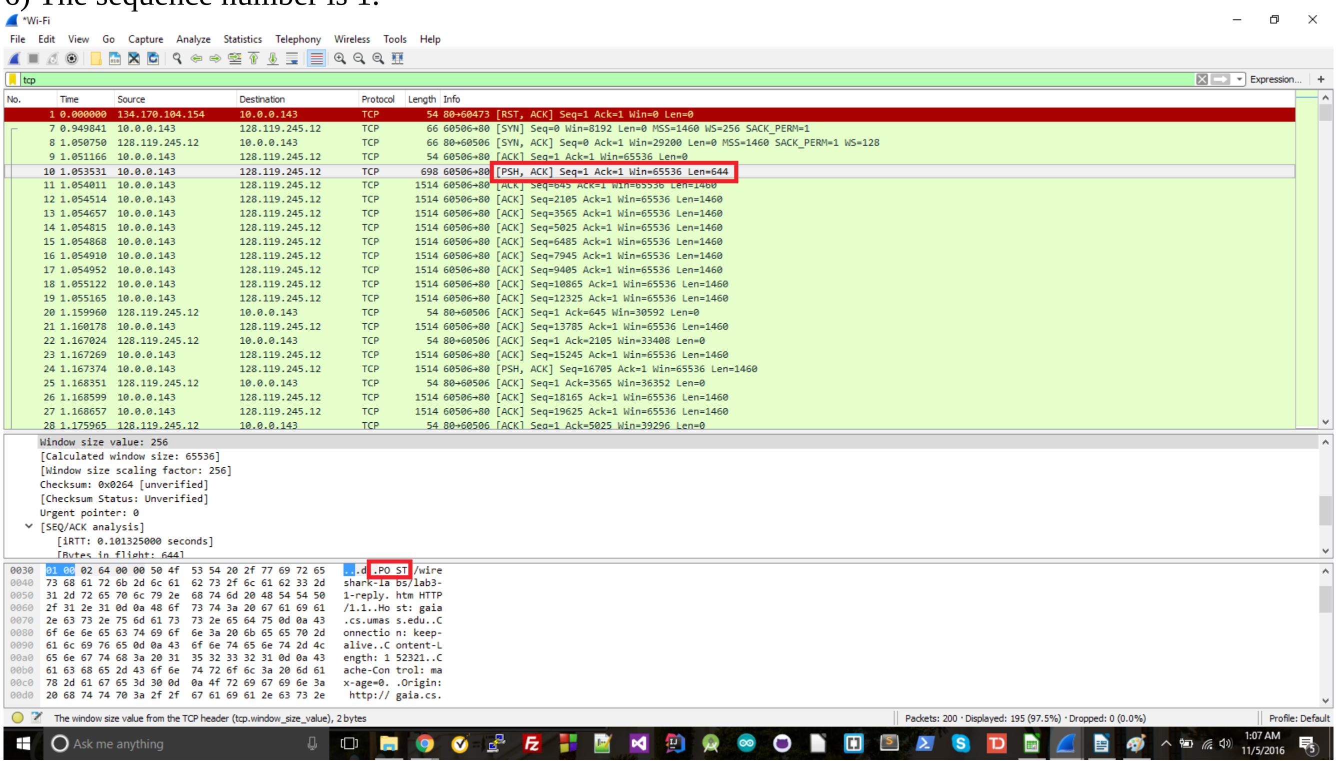
4) Sequence number is 0. We know it's the SYN segment because it has the SYN flag



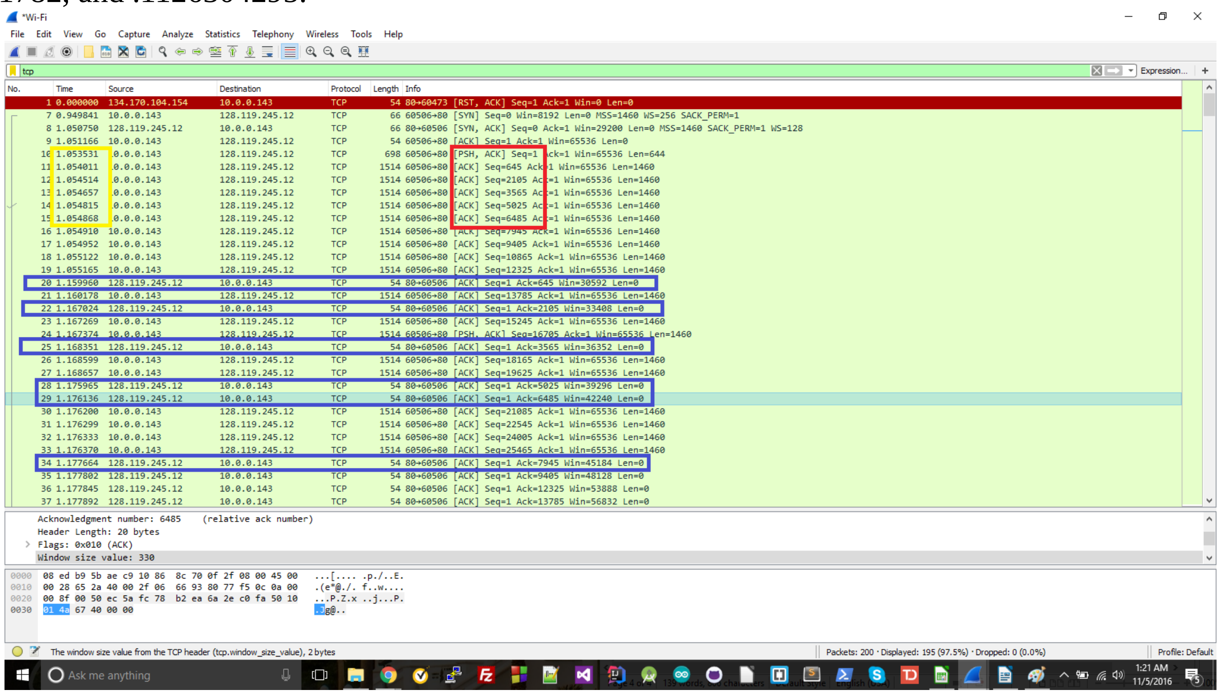
5) Sequence number is 0. ACK is 1. This is determined by what the next byte needed is (which since this is the SYN the first byte is the one that's needed.) It's identified as SYNACK by the presence of SYN and ACK flags.

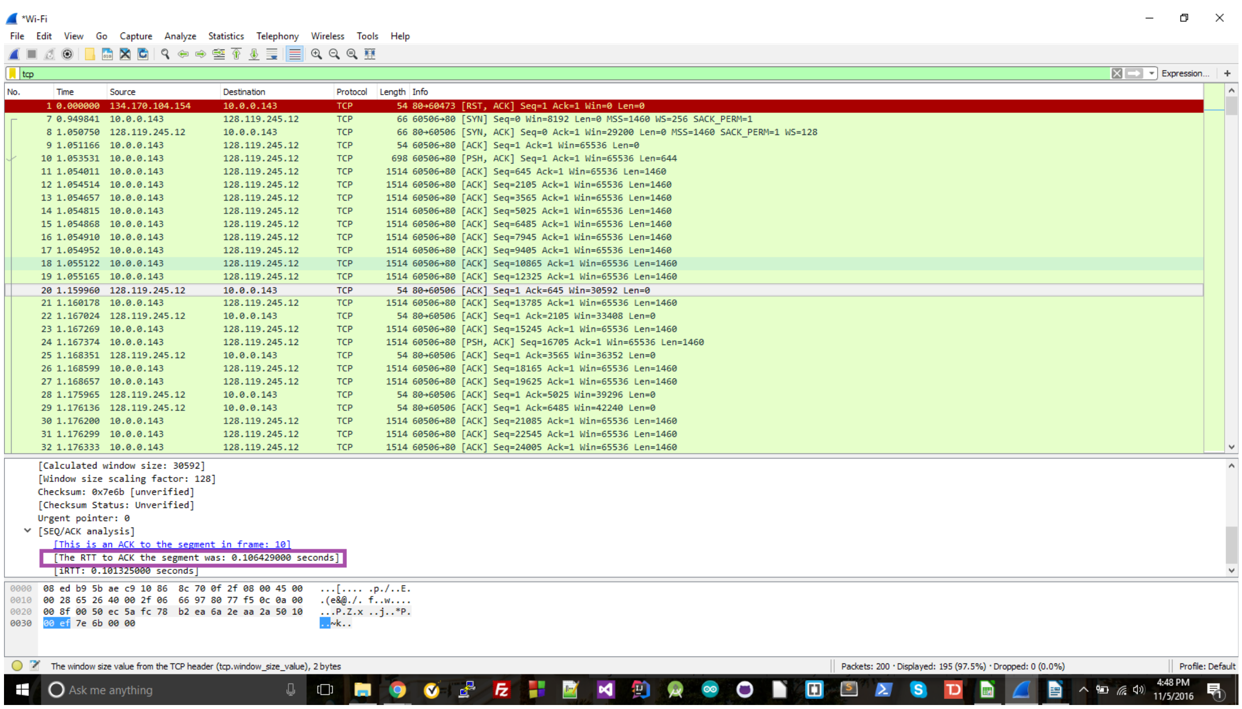


6) The sequence number is 1.

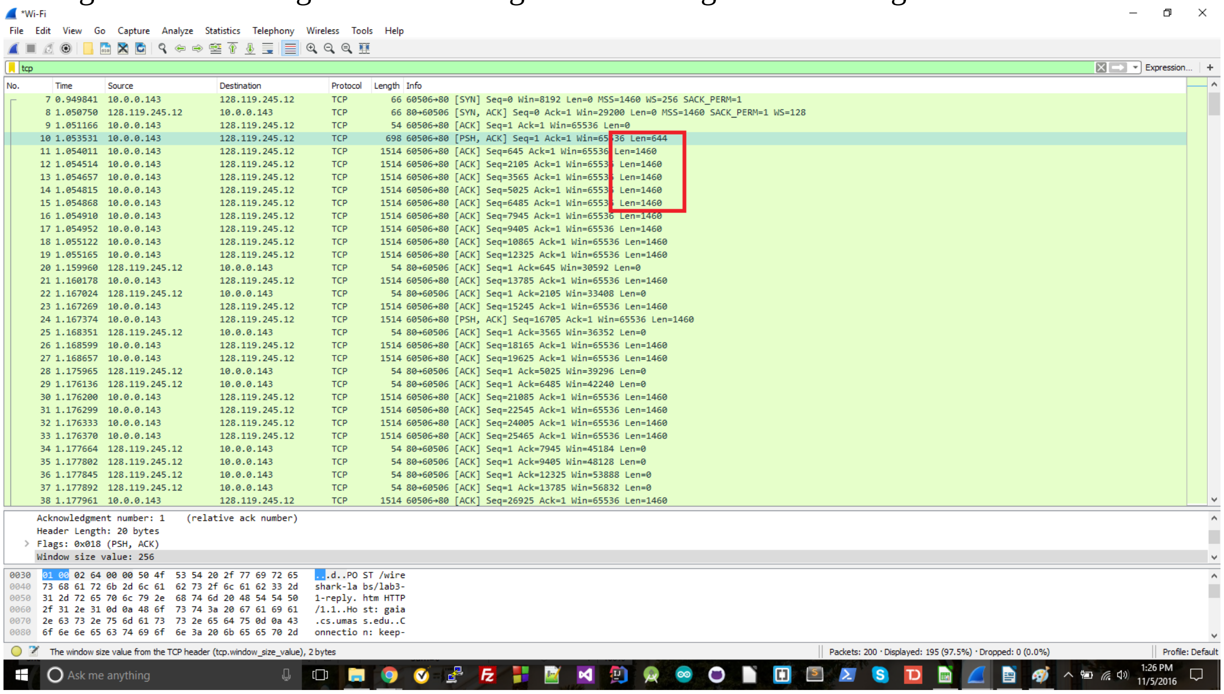


7) The first six sequence numbers are 1, 645, 2105, 3565, 5025 and 6485. (Red)
They were sent at times 1.053531, 1.054011, 1.054514, 1.054657, 1.054815, and 1.054868 respectively. (Yellow)
They were acknowledged at time 1.159960, 1.167024, 1.168351, 1.175965, 1.176136, and 1.177664. (Blue)
The RTT's are .106429, .113013, .113837, .121308, .121321, and .122796 (calculated by subtracting the sent time from the ACK received time. This can be verified in the SEQ/ACK analysis section{purple})
Using an α of .125, the estimated RTT following each ACK is .106429, .107252, .108075, .10972923, .1111782, and .1126304295.

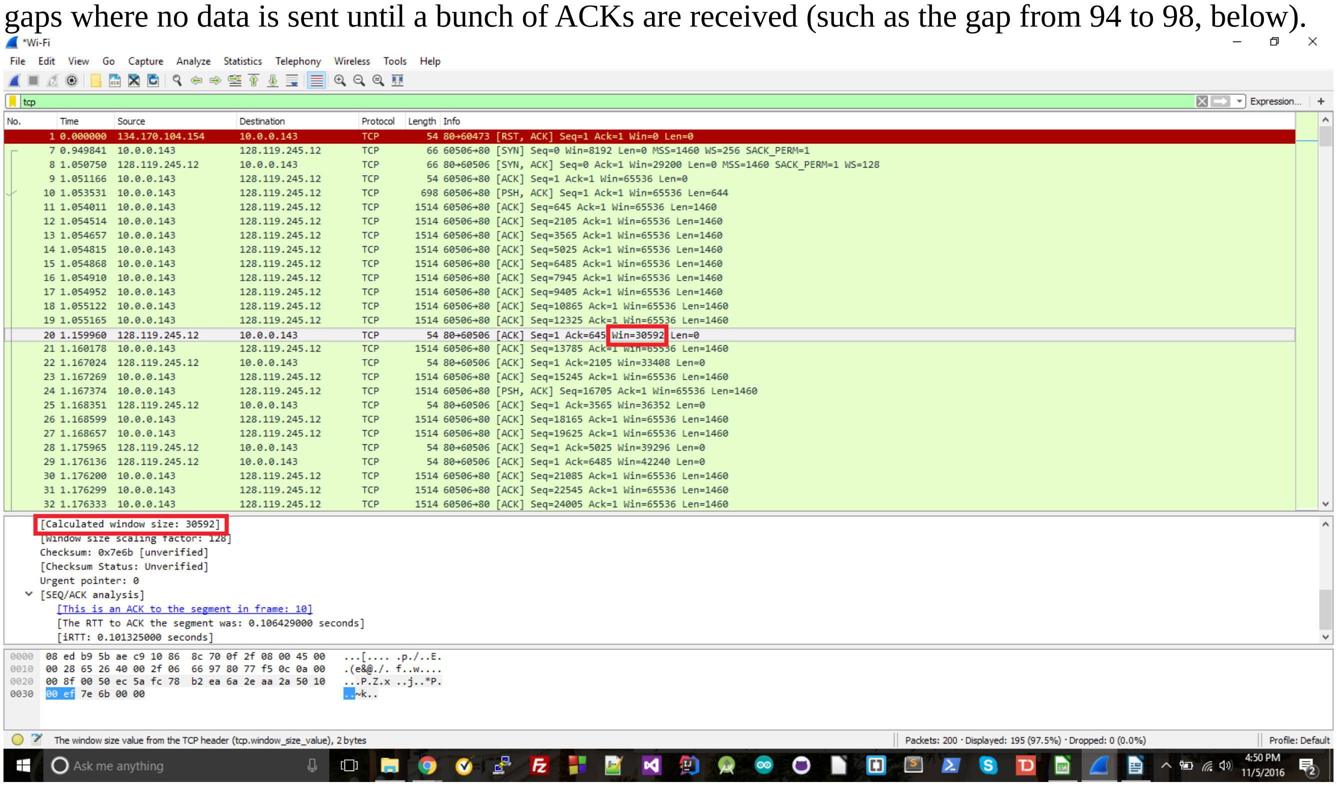


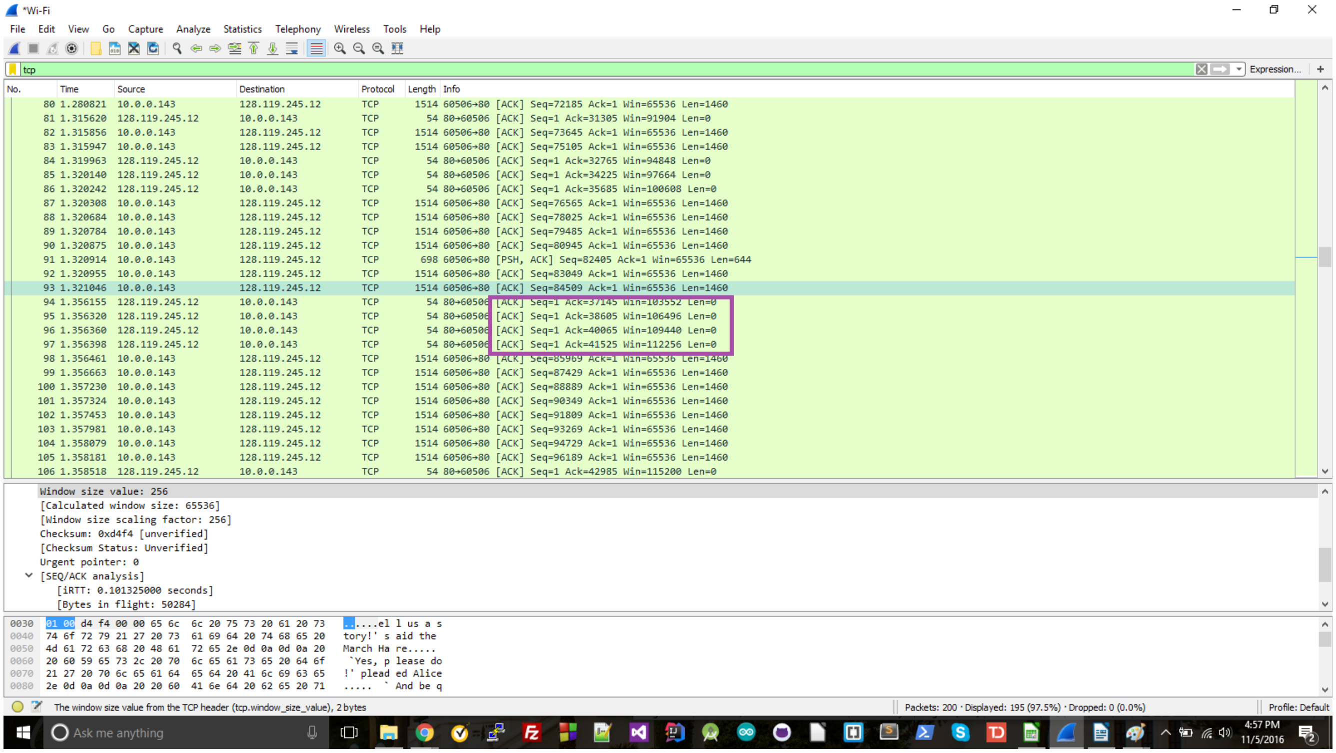


8) The length of the first segment is 644. Segments 2 through 6 are all length 1460

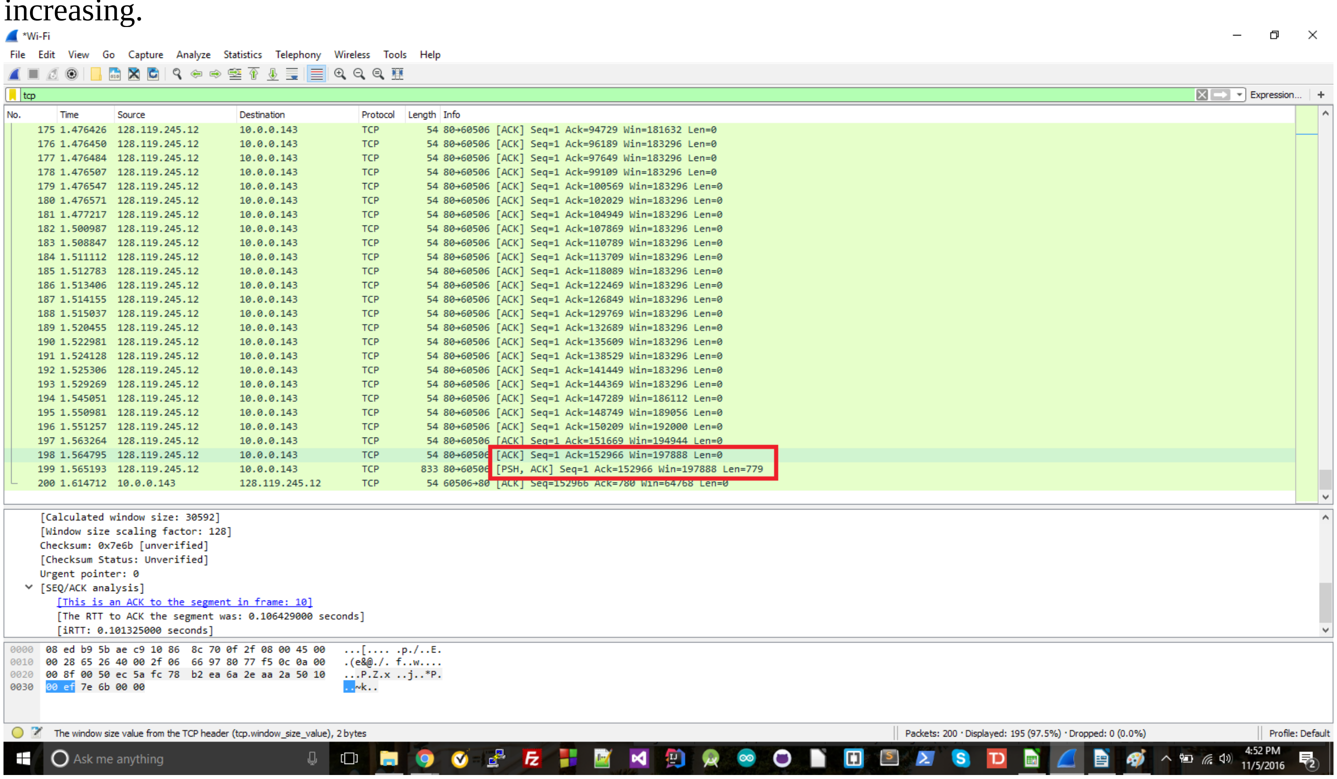


9) The minimum buffer space from the receiver is 30592. After this the window keeps growing. There doesn't appear to be much throttling by the sender; it looks like the receiver is able to keep up with the sender by growing the window and sending ACK's. There are some instances though where there are gaps where no data is sent until a bunch of ACKs are received (such as the gap from 94 to 98, below).

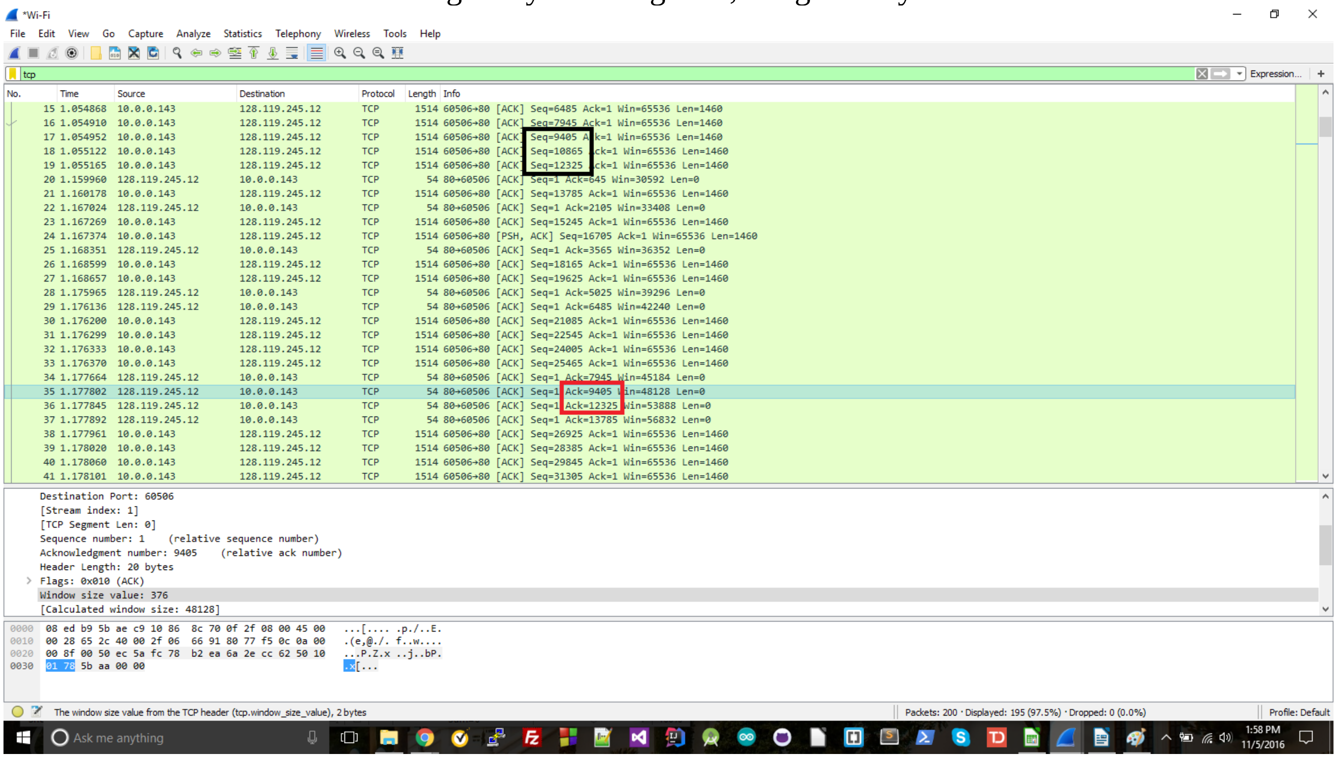




10) There were no retransmitted segments, though there was an ACK that was received multiple times, suggesting that it was sent multiple times. To check for this I looked to see if there were any sequence numbers that showed up more than once. All sequence numbers were unique and were steadily increasing.

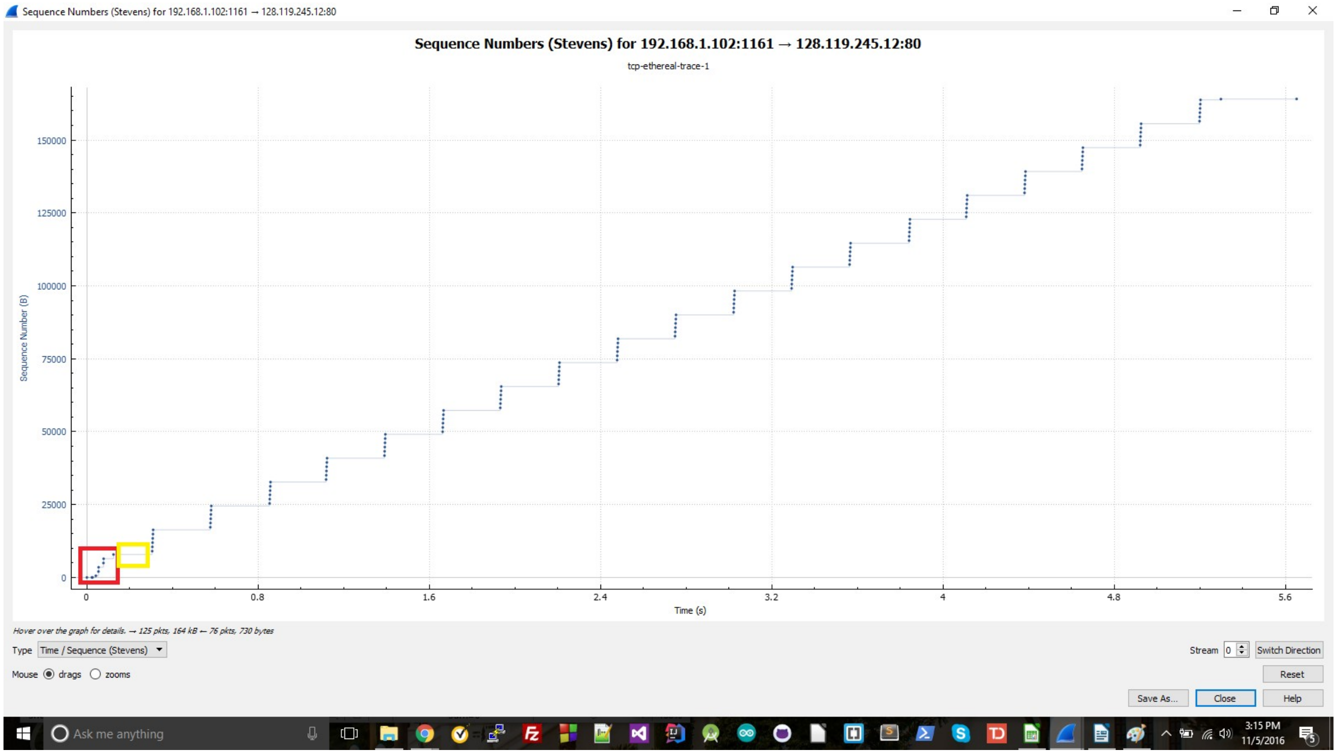


11) The receiver typically acknowledges between 1000 and 1500 bytes in an ACK, or one segment. There was an instance of it ACKing every other segment, but generally it ACKed one at a time.



12) The throughput is calculated by dividing the amount of data by the amount of time it takes to send the data. Using the total length of 152966 bytes, and a total time of .560171 seconds (the time from the first packet of the post until the last), the throughput is 273070.187 bytes/second.

13) The slow-start begins on the left side in the red box, while it ends one the right side. The yellow box is where congestion control takes over and the sender waits to receive ACKs to avoid sending too many packets at once. The measured data is different from ideal TCP behavior because there are a bunch of back-to-back sequences that are being sent out, then nothing until the next batch of back-to-back signals. While this does show pipelining, in ideal TCP I think there'd be at least some signals going out between batches due to responding to every other ACK. As it stands, it seems to be waiting for an entire batch of sequences to be ACKed before sending out the next set. In addition to that, if it were following ideal TCP, the size of the pipelines would be continuing to increase until a segment was lost or there was a timeout. The number of segments being sent out at a time stays constant at six segments per bundle.



14) The beginning of the slow-start phase starts in the left side of the red box, with its ending on the right side of that same box. The yellow box shows where congestion control takes over and begins throttling back on the number of segments that are sent out at one time. The data differs from ideal TCP in that like in question 13, it seems to wait for an entire cluster of segments to be received before sending out the next cluster (though it doesn't do this as much), as opposed to sending out a segment with every other ACK.