

# Computer Networks Laboratory

NAME: PREM SAGAR J S

SRN: PES1UG20CS825

SEC: 'H'

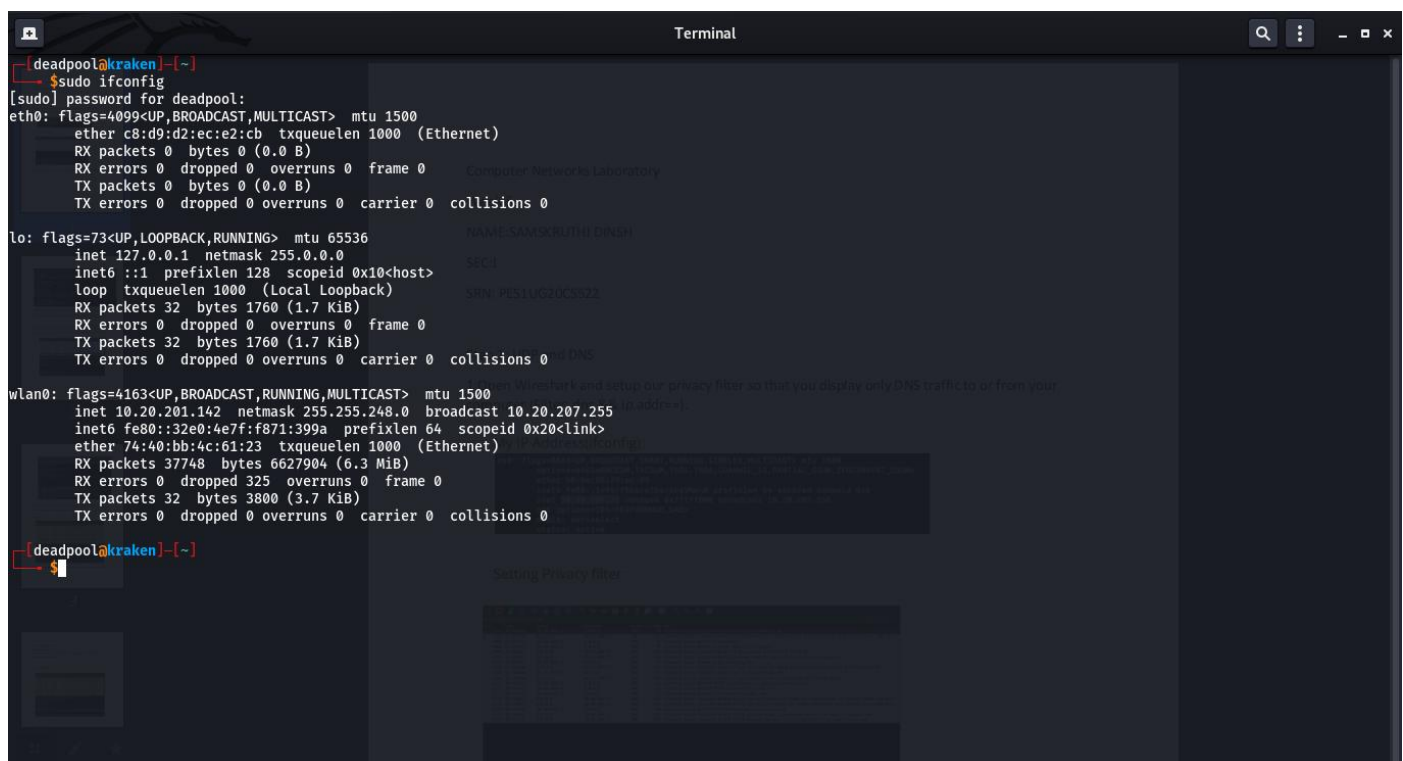
## Lab #5

### Understanding Transport and Network Layer using Wireshark

#### Step 1: UDP and DNS

1) Open Wireshark and set up our privacy filter so that you display only DNS traffic to or from your computer (Filter: **dns && ip.addr==<your IP address>**).

Ifconfig :



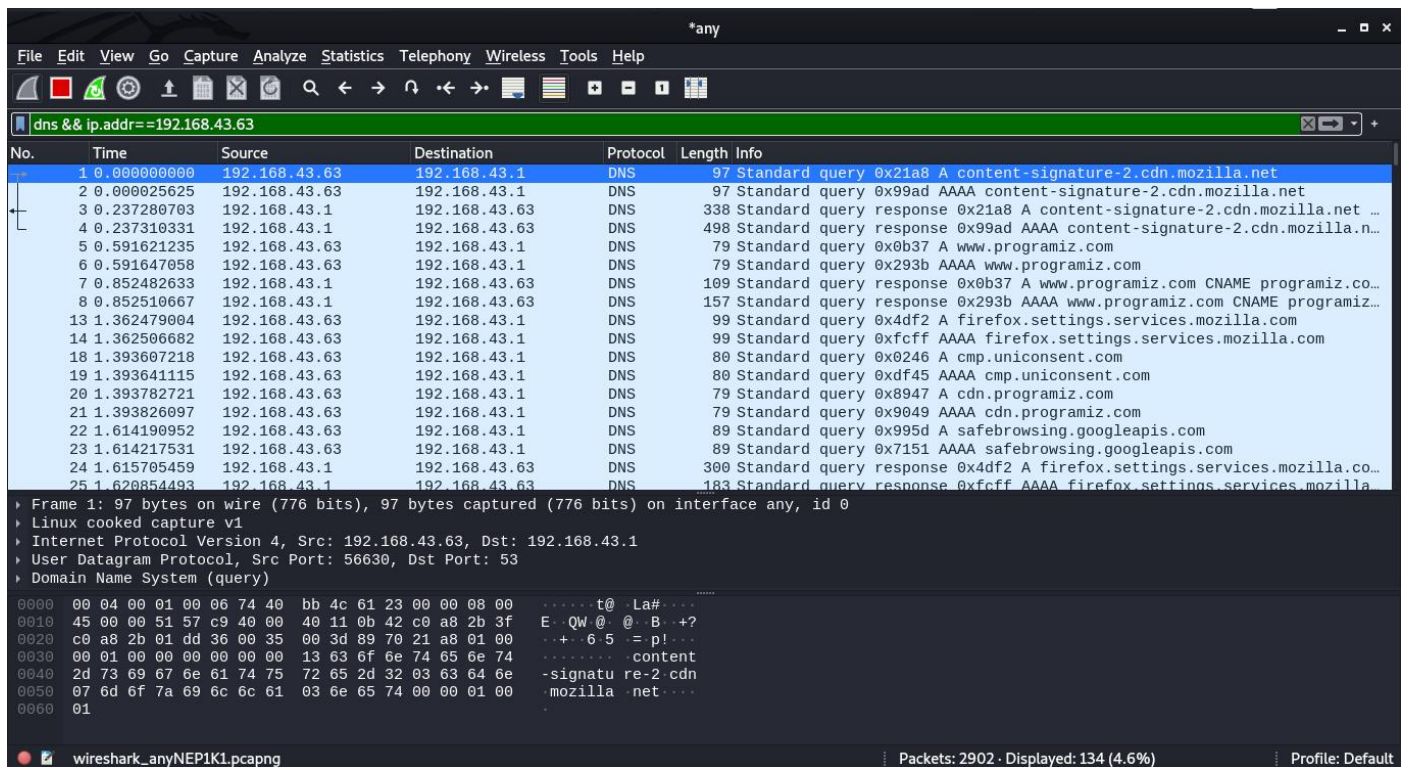
```
Terminal
[deadpool@kraken]~$ sudo ifconfig
[sudo] password for deadpool:
eth0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether c8:d9:d2:ec:e2:cb txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 32 bytes 1760 (1.7 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 32 bytes 1760 (1.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.20.201.142 netmask 255.255.248.0 broadcast 10.20.207.255
    inet6 fe80::32e0:4e7f:f871:399a prefixlen 64 scopeid 0x20<link>
    ether 74:40:bb:4c:61:23 txqueuelen 1000 (Ethernet)
    RX packets 37748 bytes 6627904 (6.3 MiB)
    RX errors 0 dropped 325 overruns 0 frame 0
    TX packets 32 bytes 3800 (3.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[deadpool@kraken]~$
```

filter (dns && ip.addr==<your IP address>):

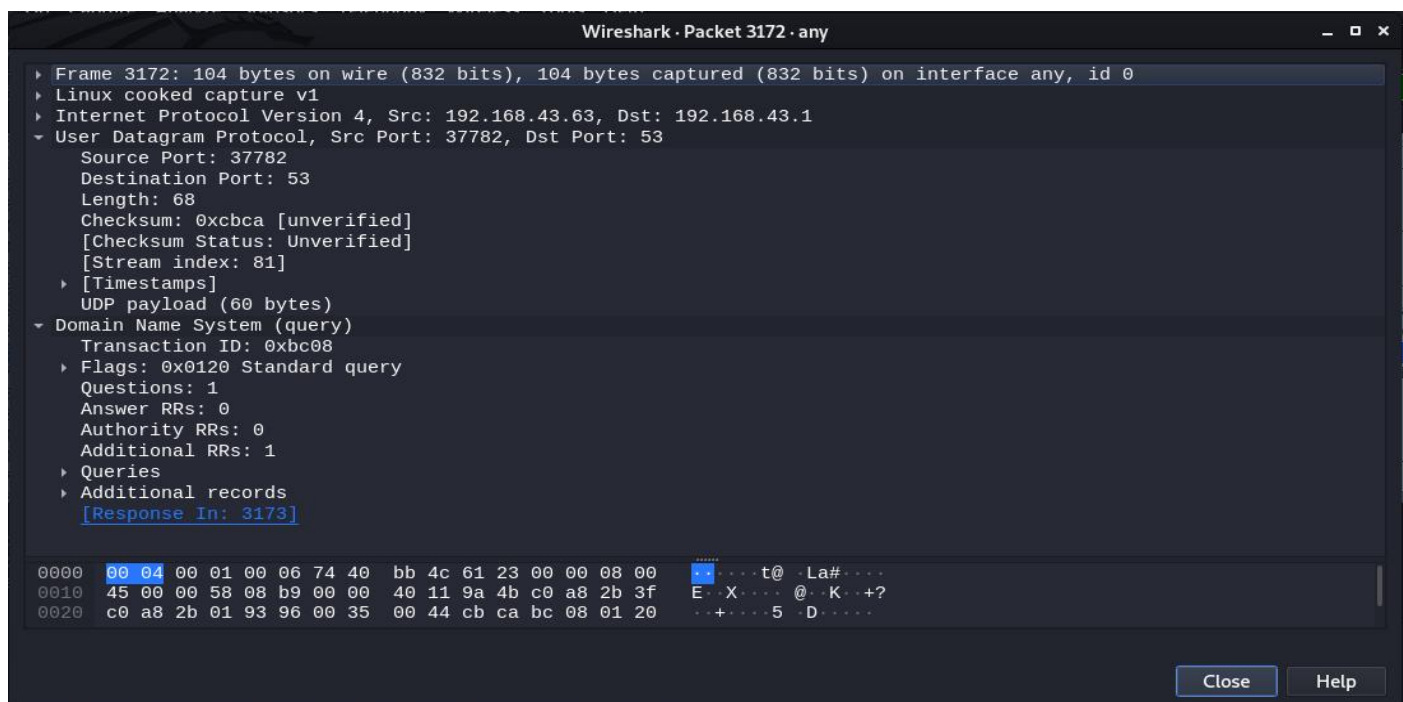
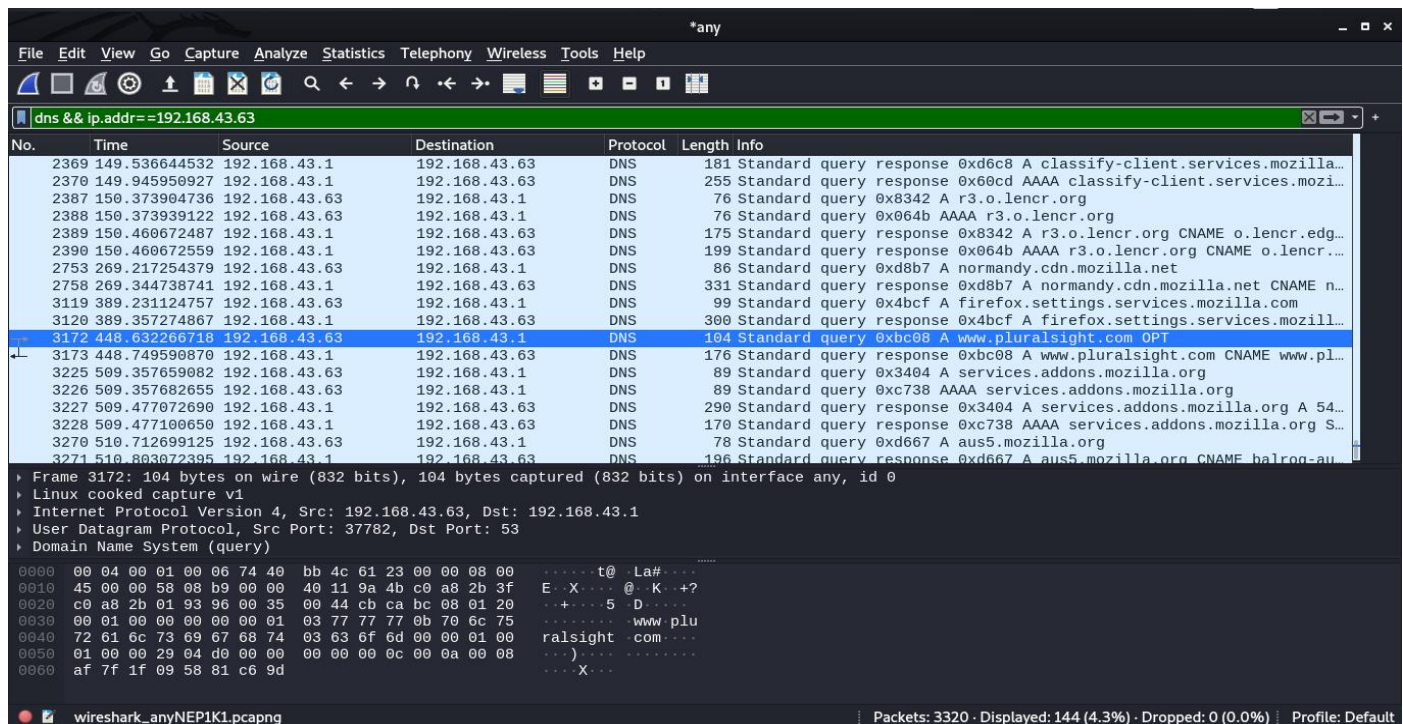


2) Use dig to generate a DNS query to lookup the domain name “www.pluralsight.com”. Then, stop the capture.



3) Before you look at the packets in Wireshark, think for a minute about what you expect to see as the UDP segment headers. What can you reasonably predict, and what could you figure out if you had some time and a calculator handy? Use your knowledge of UDP to inform your predictions. I expect to see Source-port and Destination-port numbers ,Length ,Check-sum and Application data.

4) Take a look at the query packet on Wireshark. You'll see a bunch of bytes (70-75 bytes) listed as the actual packet contents in the bottom Wireshark window. The bytes at offsets up to number 33-34 are generated by the lower-level protocols. You will also see Wireshark interpret the header contents. Matchup the bytes in the packet contents window with each field of the UDP header. Were your predictions correct?



5) Continue to examine the DNS request packet. Which fields does the UDP checksum cover? Wireshark probably shows the UDP checksum as "Validation Disabled". Why is that? Due to the prevalence of offloading in modern hardware and operating systems.



## Step 2: TCP

tcp && ip.addr== 192.168.43.63

No.	Time	Source	Destination	Protocol	Length	Info
4	1.501316702	192.168.43.63	128.2.131.88	TCP	76	53580 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=...
5	1.751892300	192.168.43.63	128.2.131.88	TCP	76	53582 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=...
6	1.882680502	128.2.131.88	192.168.43.63	TCP	76	80 → 53580 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1318 TSval=...
7	1.882736958	192.168.43.63	128.2.131.88	TCP	68	53580 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1653304886 TSe=...
8	1.883072375	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=1306 TSval=1653304887 ...
9	1.883075539	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=1307 Ack=1 Win=64256 Len=1306 TSval=165...
10	1.883087799	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=2613 Ack=1 Win=64256 Len=1306 TSval=16533048...
11	1.883089188	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=3919 Ack=1 Win=64256 Len=1306 TSval=165...
12	1.883133467	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=5225 Ack=1 Win=64256 Len=1306 TSval=16533048...
13	1.883135584	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=6531 Ack=1 Win=64256 Len=1306 TSval=165...
14	1.883153849	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=7837 Ack=1 Win=64256 Len=1306 TSval=16533048...
15	1.883155072	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=9143 Ack=1 Win=64256 Len=1306 TSval=165...
16	1.883188388	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=10449 Ack=1 Win=64256 Len=1306 TSval=1653304...
17	1.883190771	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=11755 Ack=1 Win=64256 Len=1306 TSval=16...
18	2.087431420	128.2.131.88	192.168.43.63	TCP	76	80 → 53582 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1318 TSval=...
19	2.087482724	192.168.43.63	128.2.131.88	TCP	68	53582 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1653305091 TSe=...
20	2.292138528	128.2.131.88	192.168.43.63	TCP	68	80 → 53580 [ACK] Seq=1 Ack=1307 Win=31872 Len=0 TSval=1523594936 ...
21	2.292184373	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=13061 Ack=1 Win=64256 Len=1306 TSval=1653305...
22	2.292190652	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=14367 Ack=1 Win=64256 Len=1306 TSval=16...
23	2.292138639	128.2.131.88	192.168.43.63	TCP	68	80 → 53580 [ACK] Seq=1 Ack=2613 Win=34816 Len=0 TSval=1523594936 ...
24	2.292203646	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=15673 Ack=1 Win=64256 Len=1306 TSval=1653305...
25	2.292205027	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=16979 Ack=1 Win=64256 Len=1306 TSval=16...
26	2.292210573	128.2.131.88	192.168.43.63	TCP	68	80 → 53580 [ACK] Seq=1 Ack=3919 Win=37760 Len=0 TSval=1523594936 ...
27	2.292215514	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=40576 Len...
28	2.292219379	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=43520 Len...
29	2.292219427	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=46336 Len...
30	2.292224282	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=49280 Len...
31	2.292227700	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=52224 Len...
32	2.292227736	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=55040 Len...

Frame 4: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0  
0000 00 04 00 01 00 06 74 40 bb 4c 61 23 00 00 08 00 .....t@ .La#.....

Packets: 2307 · Displayed: 2251 (97.6%) · Dropped: 0 (0.0%) · Profile: Default

12) What is the IP address and TCP port number used by your computer (client) to transfer the file?

- TCP port number : 53589

What is the IP address of the server?

- 128.2.131.88

On what port number is it sending and receiving TCP segments for this transfer of the file?

- 80

After Disabling HTTP:

tcp && ip.addr== 192.168.43.63

No.	Time	Source	Destination	Protocol	Length	Info
4	1.501316702	192.168.43.63	128.2.131.88	TCP	76	53580 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=...
5	1.751892300	192.168.43.63	128.2.131.88	TCP	76	53582 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=...
6	1.882680502	128.2.131.88	192.168.43.63	TCP	76	80 → 53580 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1318 TSval=...
7	1.882736958	192.168.43.63	128.2.131.88	TCP	68	53580 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1653304886 TSe=...
8	1.883072375	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=1306 TSval=1653304887 ...
9	1.883075539	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=1307 Ack=1 Win=64256 Len=1306 TSval=165...
10	1.883087799	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=2613 Ack=1 Win=64256 Len=1306 TSval=16533048...
11	1.883089188	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=3919 Ack=1 Win=64256 Len=1306 TSval=165...
12	1.883133467	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=5225 Ack=1 Win=64256 Len=1306 TSval=16533048...
13	1.883135584	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=6531 Ack=1 Win=64256 Len=1306 TSval=165...
14	1.883153849	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=7837 Ack=1 Win=64256 Len=1306 TSval=16533048...
15	1.883155072	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=9143 Ack=1 Win=64256 Len=1306 TSval=165...
16	1.883188388	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=10449 Ack=1 Win=64256 Len=1306 TSval=1653304...
17	1.883190771	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=11755 Ack=1 Win=64256 Len=1306 TSval=16...
18	2.087431420	128.2.131.88	192.168.43.63	TCP	76	80 → 53582 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1318 TSval=...
19	2.087482724	192.168.43.63	128.2.131.88	TCP	68	53582 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1653305091 TSe=...
20	2.292138528	128.2.131.88	192.168.43.63	TCP	68	80 → 53580 [ACK] Seq=1 Ack=1307 Win=31872 Len=0 TSval=1523594936 ...
21	2.292184373	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=13061 Ack=1 Win=64256 Len=1306 TSval=1653305...
22	2.292190652	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=14367 Ack=1 Win=64256 Len=1306 TSval=16...
23	2.292138639	128.2.131.88	192.168.43.63	TCP	68	80 → 53580 [ACK] Seq=1 Ack=2613 Win=34816 Len=0 TSval=1523594936 ...
24	2.292203646	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [ACK] Seq=15673 Ack=1 Win=64256 Len=1306 TSval=1653305...
25	2.292205027	192.168.43.63	128.2.131.88	TCP	1374	53580 → 80 [PSH, ACK] Seq=16979 Ack=1 Win=64256 Len=1306 TSval=16...
26	2.292210573	128.2.131.88	192.168.43.63	TCP	68	80 → 53580 [ACK] Seq=1 Ack=3919 Win=37760 Len=0 TSval=1523594936 ...
27	2.292215514	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=40576 Len...
28	2.292219379	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=43520 Len...
29	2.292219427	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=46336 Len...
30	2.292224282	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=49280 Len...
31	2.292227700	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=52224 Len...
32	2.292227736	128.2.131.88	192.168.43.63	TCP	68	[TCP Window Update] 80 → 53580 [ACK] Seq=1 Ack=3919 Win=55040 Len...

Frame 4: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface any, id 0  
0000 00 04 00 01 00 06 74 40 bb 4c 61 23 00 00 08 00 .....t@ .La#.....

Packets: 2307 · Displayed: 2251 (97.6%) · Dropped: 0 (0.0%) · Profile: Default

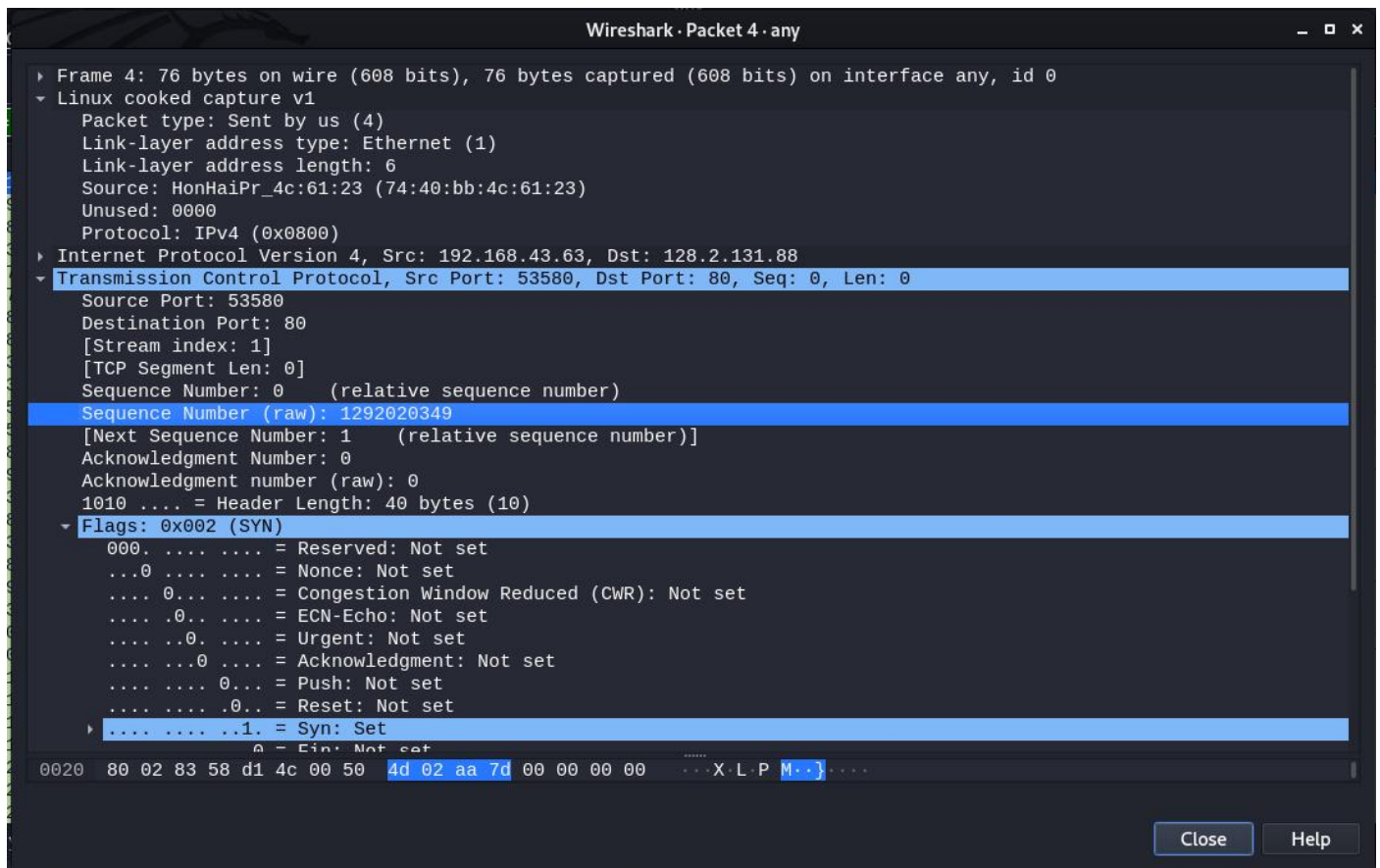
## Step 2b: TCP Basics

What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection?

- o Relative Sequence Number:0
- o Raw Sequence number:1292020349

What element of the segment identifies it as a SYN segment?

- o Here the SYN bit is set to 1 in the flags section in below figure. So, It can be identified as a SYN segment.

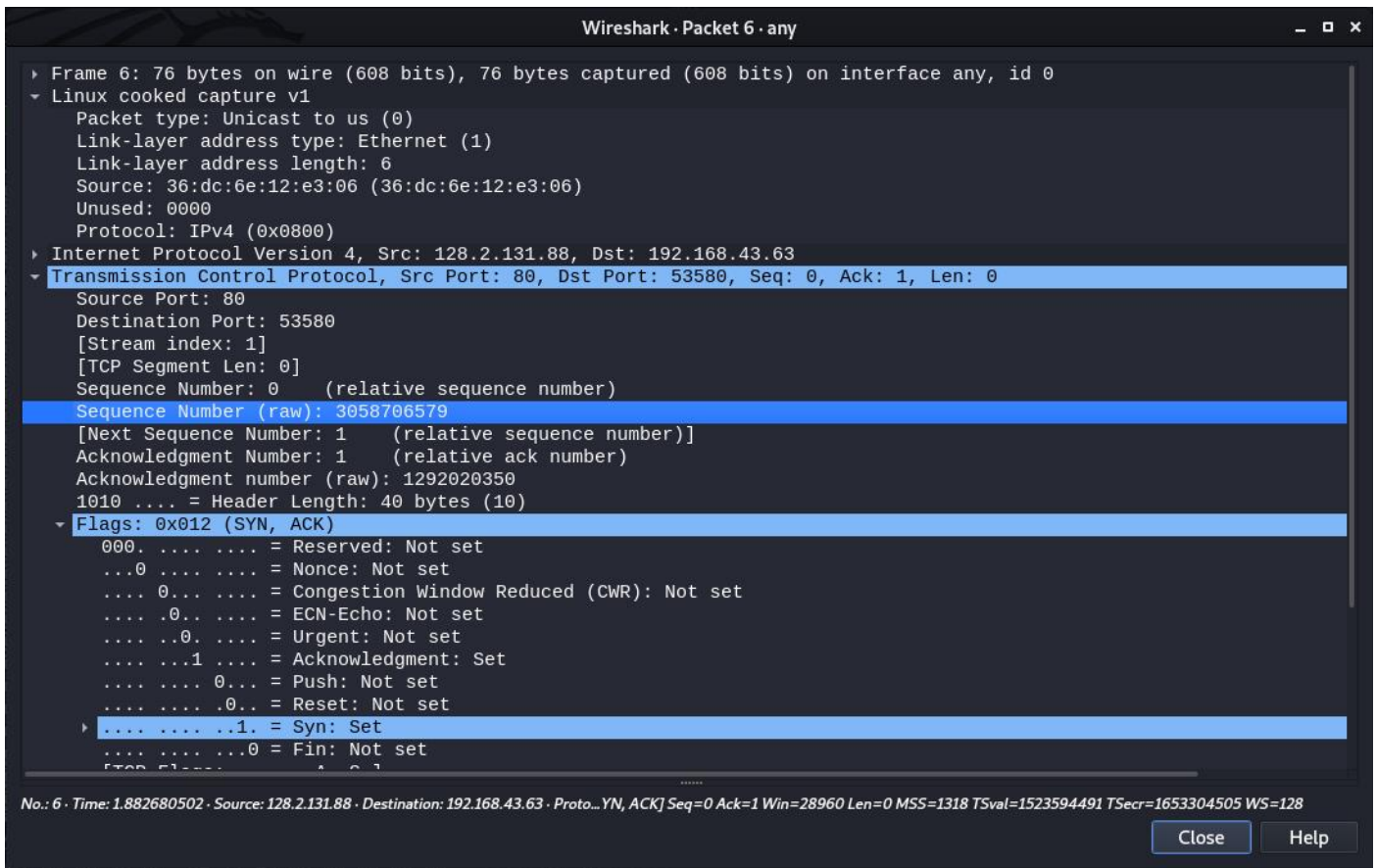


Wireshark uses relative sequence numbers by default. Can you obtain absolute sequence numbers instead? How?

- o Absolute Sequence number=Relative Sequence Number+offset

What is the sequence number of the SYNACK segment sent by the server in reply to the SYN?





o the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN is 0.

What is the value of the Acknowledgement field in the SYNACK segment?

o The value of the acknowledgement field in the SYNACK segment is 1 How did the server determine that value?

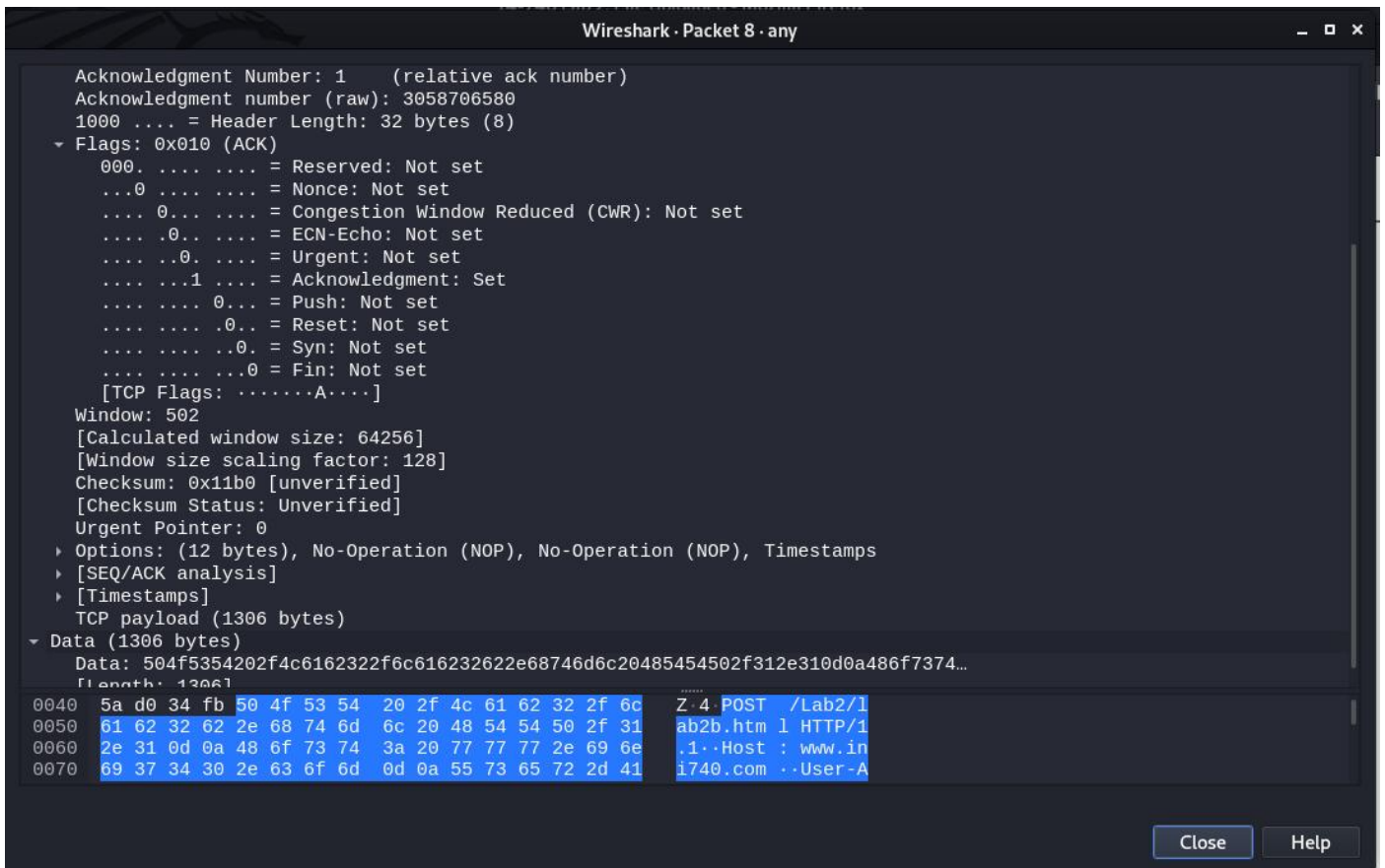
o The server adds 1 to the initial sequence number of SYN segment from the client computer. For this case, the initial sequence number of SYN segment from the client computer is 0, thus the value

of the ACKnowledgement field in the SYNACK segment is

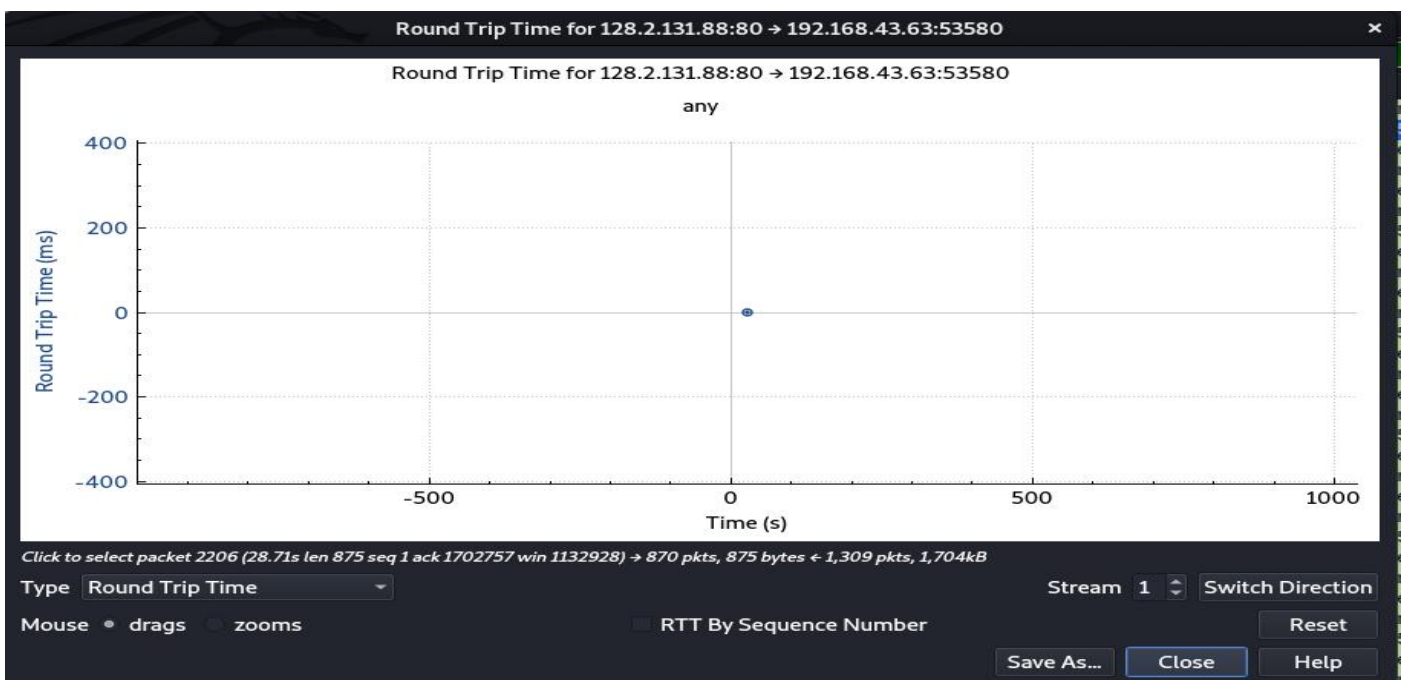
1. What element in the segment identifies it as a SYNACK segment?

o A segment will be identified as a SYNACK segment if both SYN flag and Acknowledgement in the segment are set to 1.

What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command, you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" with in its DATA field.



According to the above figure, segment 379 contains the POST command. It's sequence number is 1. Consider the TCP segment containing the HTTP POST as the first segment in the non-overhead part of the TCP connection. For the segments which follow, put together a table with one row per segment (and columns for whatever data you think is useful) until you have enough segments to calculate four SampleRTT values according to the RTT estimation techniques discussed in class. Calculate what those SampleRTT values are, as well as the EstimatedRTT after each Sample is collected. Discuss this calculation, including what your initial EstimatedRTT was, your choice of parameters, and any segments that weren't used in the calculation. Note: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the "listing of captured packets" window that is being sent from the client to the server. Then select: Statistics → TCP Stream Graph → Round Trip Time Graph.



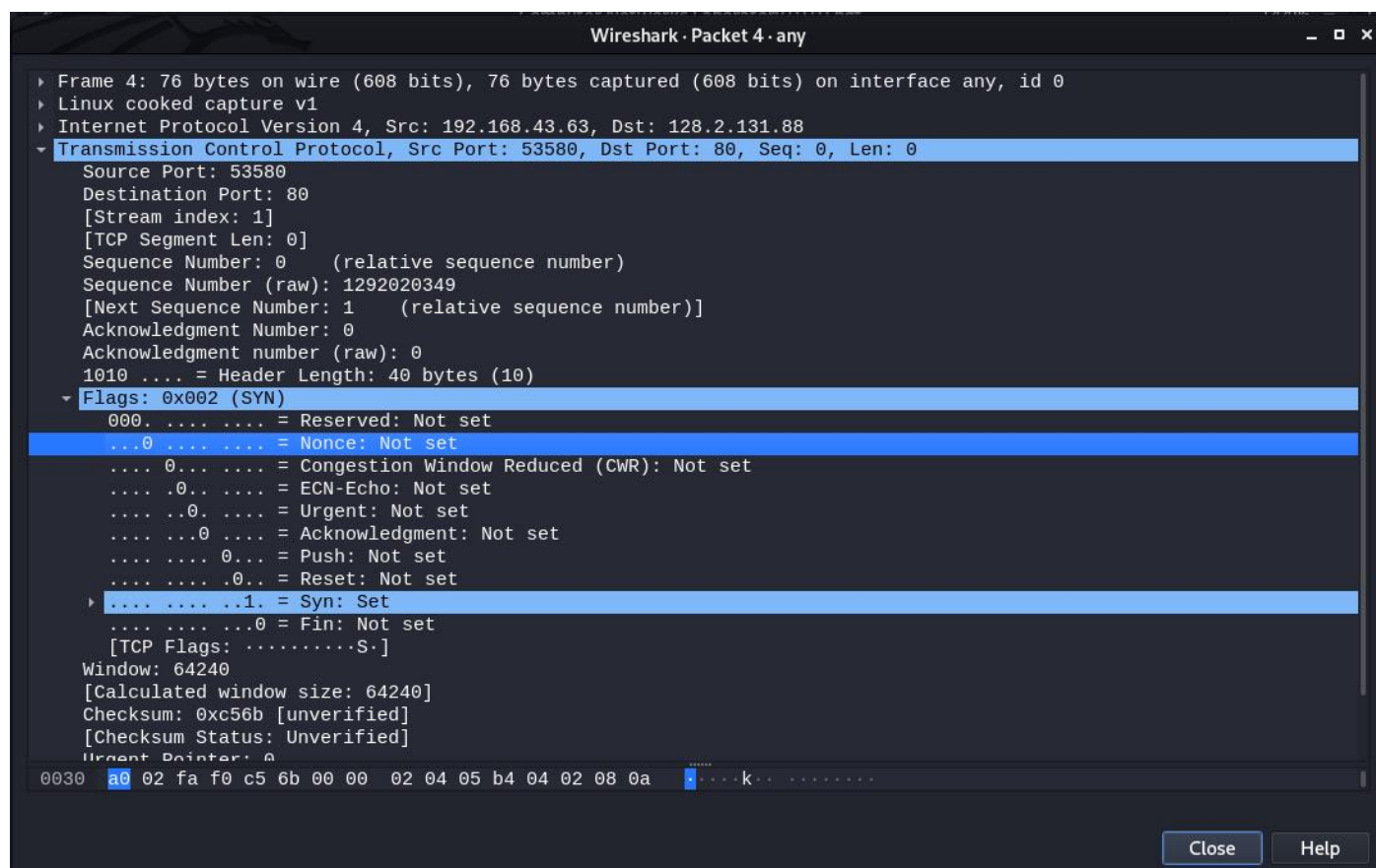
- As we can see in the above figure, the first 6 segments are numbered as 379,380,381,382,383,384
- Sequence number of segment 1:1
- Sequence number of segment 2:580
- Sequence number of segment 3:718
- Sequence number of segment 4:2158
- Sequence number of segment 5:3598
- Sequence number of segment 6:5038

Segment	Sent Time	ACK received time	RTT
1	48.441203	48.844942	6.458
2	48.442227	48.844942	6.457
3	48.444505	48.844943	6.456
4	48.444512	48.844943	6.455
5	48.444516	48.844943	6.455
6	48.444520	48.844944	6.455

EstimatedRTT = 0.875 \* EstimatedRTT + 0.125 \*sampleRTT

S1:  $0.875 \times 0.403739 + 0.125 \times 48.844942 = 6.458889$   
S2:  $0.875 \times 0.402715 + 0.125 \times 48.844942 = 6.457993$   
S3:  $0.875 \times 0.400438 + 0.125 \times 48.844943 = 6.456001$   
S4:  $0.875 \times 0.400431 + 0.125 \times 48.844943 = 6.455995$   
S5:  $0.875 \times 0.400427 + 0.125 \times 48.844943 = 6.455991$   
S6:  $0.875 \times 0.400424 + 0.125 \times 48.844944 = 6.455989$

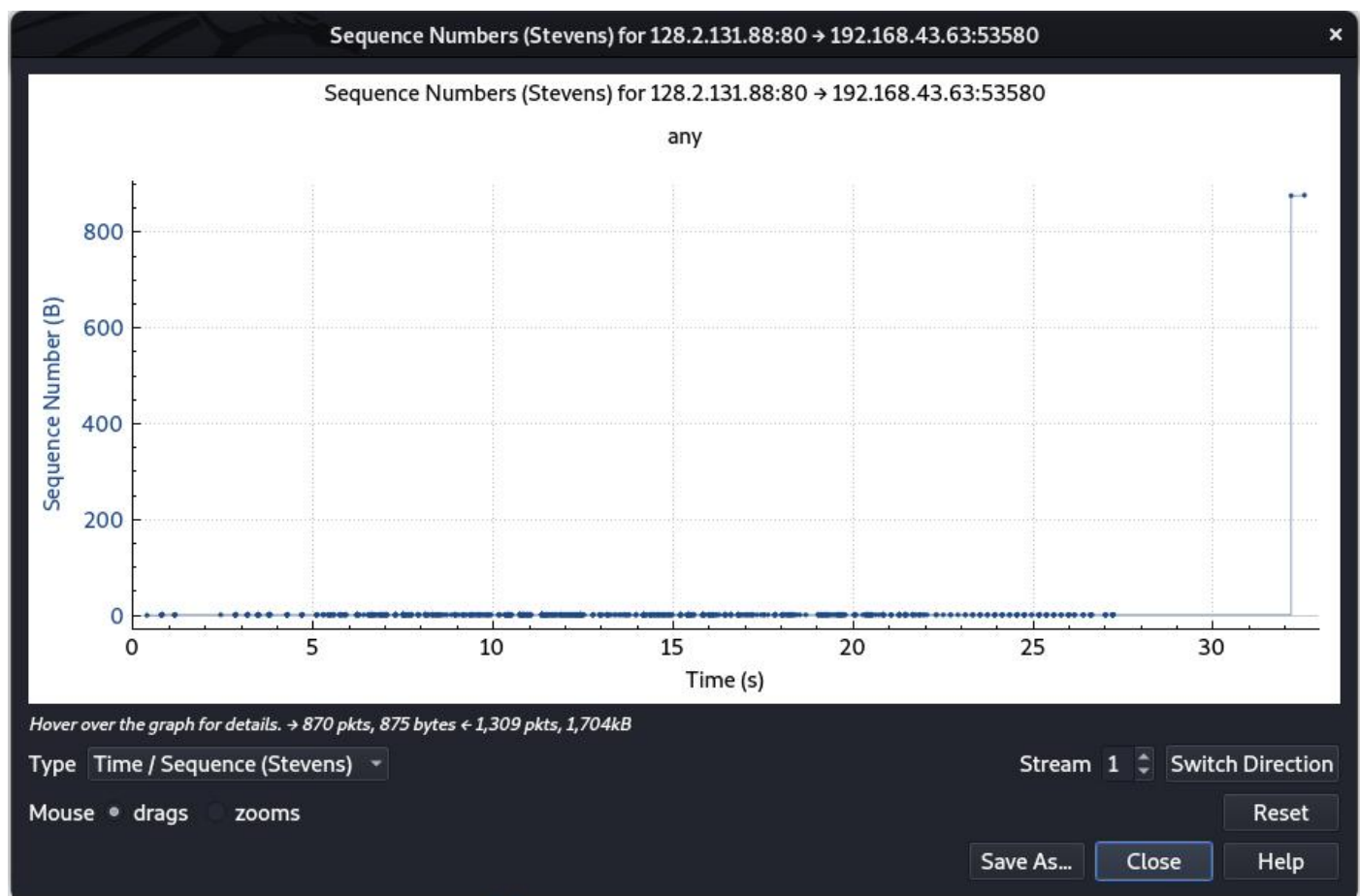
What is the minimum amount of available buffer space advertised at the receiver for the entire trace? Does the lack of receiver buffer space ever throttle the sender?





o The minimum amount of available buffer space advertised at the received for the entire trace is indicated first ACK from the server, its value is 64240 bytes (shown in above figure).

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



As the sequence number-time graph is monotonically increasing from the above graph,we can conclude that no packet is retransmitted How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is delayed ACKing segments?Explain how or why not.

The difference between the acknowledged sequence numbers of two consecutive ACKs indicates the data received by the server between these two ACKs. The receiver is ACKing every other segment. For example, segment of Number. 390 acknowledged data with 718-580=38 bytes.

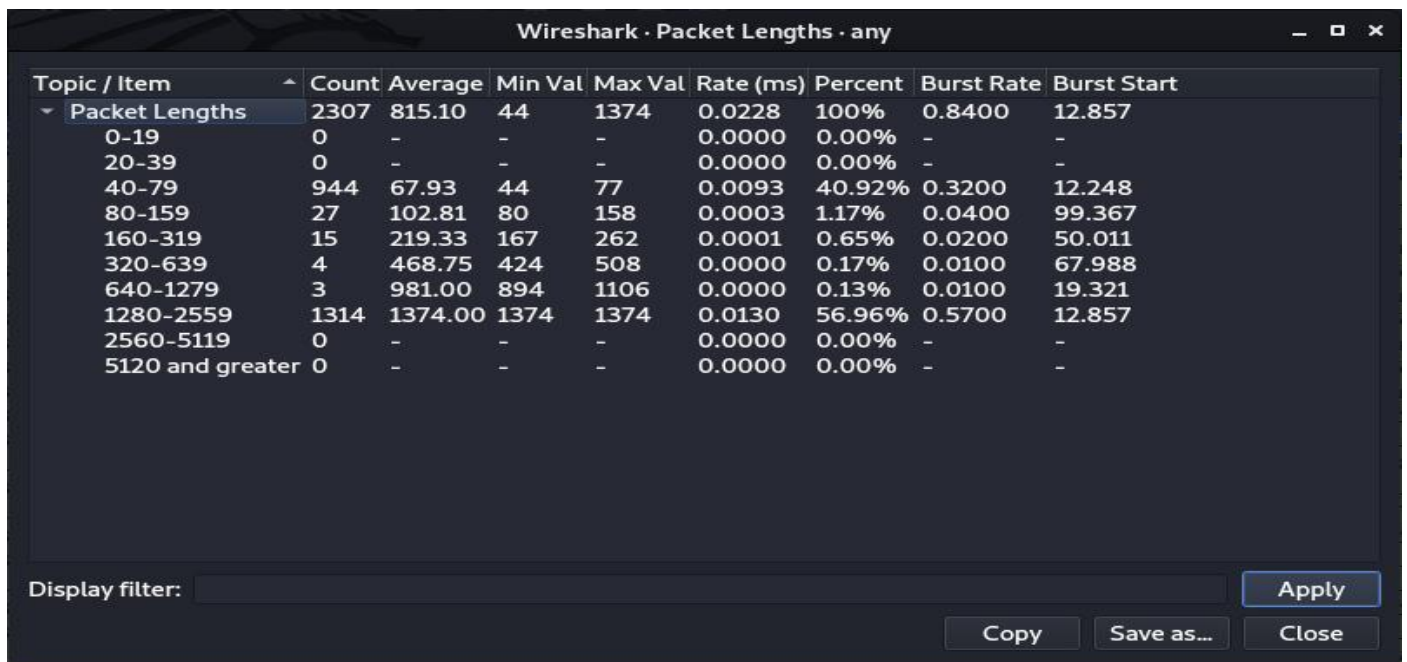
1947	18.706618610	128.2.131.88	192.168.43.63	TCP	68 80 → 53580	[ACK]	Seq=1 Ack=1218499 Win=865920 Len=0 TSval=1523611...
1948	18.706674203	192.168.43.63	128.2.131.88	TCP	1374 53580 → 80	[ACK]	Seq=1687353 Ack=1 Win=64256 Len=1306 TSval=16533...
1949	18.719666790	128.2.131.88	192.168.43.63	TCP	68 80 → 53580	[ACK]	Seq=1 Ack=1219805 Win=868864 Len=0 TSval=1523611...
1950	18.719715704	192.168.43.63	128.2.131.88	TCP	1374 53580 → 80	[ACK]	Seq=1688659 Ack=1 Win=64256 Len=1306 TSval=16533...
1951	18.719721832	192.168.43.63	128.2.131.88	TCP	1374 53580 → 80	[PSH, ACK]	Seq=1689965 Ack=1 Win=64256 Len=1306 TSval=...
1952	18.828023556	128.2.131.88	192.168.43.63	TCP	68 80 → 53580	[ACK]	Seq=1 Ack=1221111 Win=871680 Len=0 TSval=1523611...
1953	18.828100264	192.168.43.63	128.2.131.88	TCP	1374 53580 → 80	[ACK]	Seq=1691271 Ack=1 Win=64256 Len=1306 TSval=16533...
1954	18.942859027	128.2.131.88	192.168.43.63	TCP	68 80 → 53580	[ACK]	Seq=1 Ack=1222417 Win=874624 Len=0 TSval=1523611...
1955	18.942931415	192.168.43.63	128.2.131.88	TCP	1374 53580 → 80	[ACK]	Seq=1692577 Ack=1 Win=64256 Len=1306 TSval=16533...
1956	18.942940703	192.168.43.63	128.2.131.88	TCP	1374 53580 → 80	[PSH, ACK]	Seq=1693883 Ack=1 Win=64256 Len=1306 TSval=...
1957	19.024579225	128.2.131.88	192.168.43.63	TCP	68 80 → 53580	[ACK]	Seq=1 Ack=1223723 Win=877568 Len=0 TSval=1523611...

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

The file on the hard drive is 17,01,981 bytes, and the download time is 51.057952 (last TCP segment) - 48.441203 (last ACK) = 2.616749 second. Therefore, the throughput for the TCP connection is computed as 17,01,981/2.616749=650418.133340 bytes/second.

## Step 2c: Statistics

What is the most common TCP packet length range? What is the second most common TCP packet length range? Why is the ratio of TCP packets of length < 40 bytes equal to zero? Describe what actions you took to get answers to these questions from Wireshark.



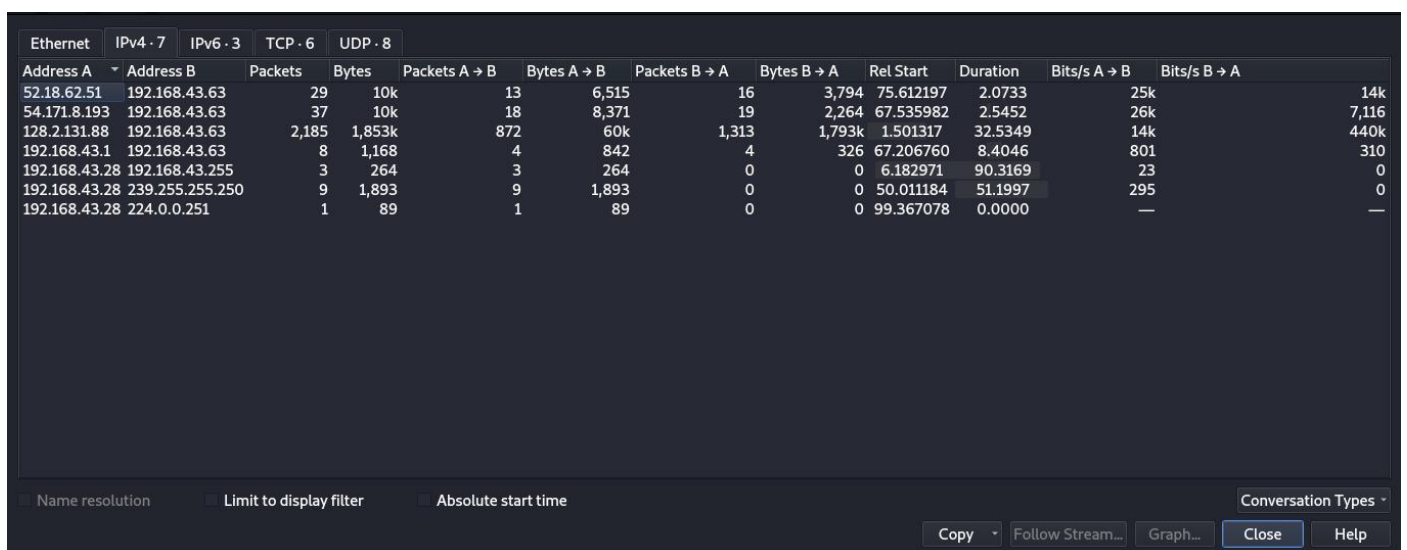
The screenshot shows the 'Wireshark - Packet Lengths - any' window. It displays a table of statistics for packet lengths. The 'Packet Lengths' category is expanded, showing a list of ranges from 0-19 to 5120 and greater. The 'Count' column shows the number of packets in each range, and the 'Percent' column shows the percentage of total packets. The 'Rate (ms)' column shows the time taken for each range, and the 'Burst Rate' and 'Burst Start' columns show the burst rate and start time for each range.

Topic / Item	Count	Average	Min Val	Max Val	Rate (ms)	Percent	Burst Rate	Burst Start
Packet Lengths	2307	815.10	44	1374	0.0228	100%	0.8400	12.857
0-19	0	-	-	-	0.0000	0.00%	-	-
20-39	0	-	-	-	0.0000	0.00%	-	-
40-79	944	67.93	44	77	0.0093	40.92%	0.3200	12.248
80-159	27	102.81	80	158	0.0003	1.17%	0.0400	99.367
160-319	15	219.33	167	262	0.0001	0.65%	0.0200	50.011
320-639	4	468.75	424	508	0.0000	0.17%	0.0100	67.988
640-1279	3	981.00	894	1106	0.0000	0.13%	0.0100	19.321
1280-2559	1314	1374.00	1374	1374	0.0130	56.96%	0.5700	12.857
2560-5119	0	-	-	-	0.0000	0.00%	-	-
5120 and greater	0	-	-	-	0.0000	0.00%	-	-

Most common TCP Packet range is 40-79 (has 1428 packets) Second Most common TCP Packet range is 1280-2559 (has 1185 packets) The header length is 40 bytes in handshaking stage as it consists of 10 headers so as the minimum packet length is 40 bytes without any data in it, the ratio of tcp packets of length <40 is 0.

This information can be obtained by navigating to 'statistics <packet lenth's' in wireshark menu.

What average throughput did you use in Mbps? How many packets were captured in the packet capture session? How many bytes in total? Explain your methods. Average Throughput=1986545/154.938=12821.547974 Bps=0.102572383792 Mbps 3052 packets have been captured in the packet capture session. 1986545 bytes in total are captured.



The screenshot shows the 'Wireshark - Conversation' window. It displays a table of conversations between hosts. The 'Address A' and 'Address B' columns show the IP addresses of the hosts. The 'Packets' and 'Bytes' columns show the number of packets and bytes captured for each conversation. The 'Packets A → B' and 'Bytes A → B' columns show the number of packets and bytes sent from host A to host B. The 'Packets B → A' and 'Bytes B → A' columns show the number of packets and bytes sent from host B to host A. The 'Rel Start' and 'Duration' columns show the relative start time and duration of each conversation. The 'Bits/s A → B' and 'Bits/s B → A' columns show the bit rate for each direction of traffic.

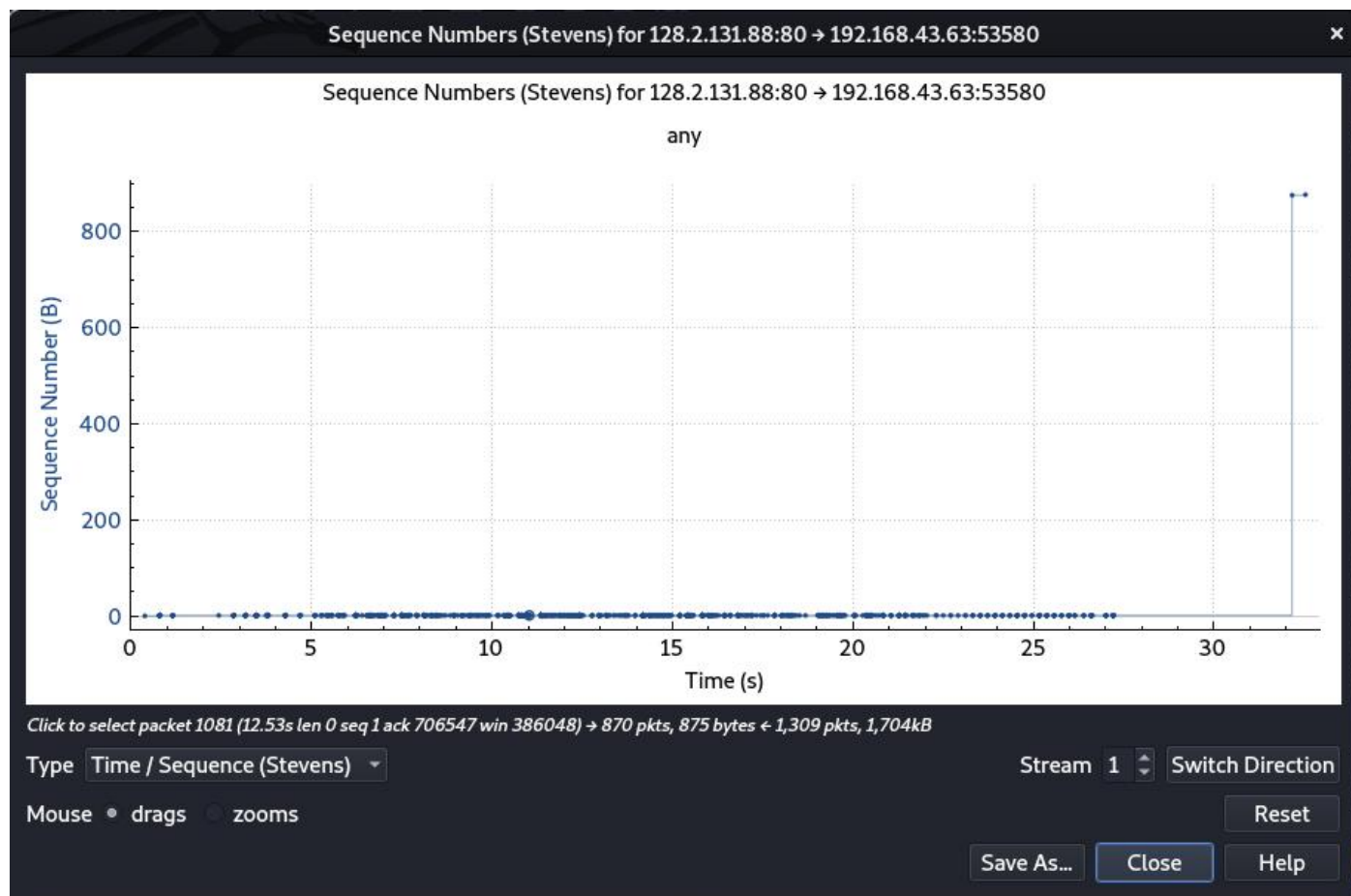
Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration	Bits/s A → B	Bits/s B → A
52.18.62.51	192.168.43.63	29	10k	13	6,515	16	3,794	75.612197	2.0733	25k	14k
54.171.8.193	192.168.43.63	37	10k	18	8,371	19	2,264	67.535982	2.5452	26k	7,116
128.2.131.88	192.168.43.63	2,185	1,853k	872	60k	1,313	1,793k	1.501317	32.5349	14k	440k
192.168.43.1	192.168.43.63	8	1,168	4	842	4	326	67.206760	8.4046	801	310
192.168.43.28	192.168.43.255	3	264	3	264	0	0	6.182971	90.3169	23	0
192.168.43.28	239.255.255.250	9	1,893	9	1,893	0	0	50.011184	51.1997	295	0
192.168.43.28	224.0.0.251	1	89	1	89	0	0	99.367078	0.0000	—	—

A conversation represents a traffic between two hosts. With which remote host did your localhost converse the most (in bytes)? How many packets were sent from your host? How many packets

were sent from the remote host? With 128.2.131.88 my local host conversed the most(1840k) 1200 packets were sent from my host  
872 packets received from remote host.

### Step 3: Congestion Control

Use the Time-Sequence-Graph(Stevens) plotting tool to view the sequence number versus time plot of segments being sent. Can you identify where TCP's slow start phase begins and ends, 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. Make sure to include a copy of the plot in your report.



### Step 4: The Network Layer

Take a look at the IP section of the DNS query (the packet that was generated when you used dig to request the address of www.pluralsight.com). Match up the header fields with the format we discussed in class (don't just look through Wireshark's display -- instead, match the raw bytes with the pictures we saw in lecture, which I've copied on the right).



```
Wireshark · Packet 2 · [no capture file]

Frame 2: 176 bytes on wire (1408 bits), 176 bytes captured (1408 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 192.168.177.35, Dst: 192.168.177.63
User Datagram Protocol, Src Port: 53, Dst Port: 60110
Domain Name System (response)
  Transaction ID: 0xaf71
  Flags: 0x8180 Standard query response, No error
    1... .. = Response: Message is a response
    .000 0... .. = Opcode: Standard query (0)
    .... 0... .. = Authoritative: Server is not an authority for domain
    .... ..0... .. = Truncated: Message is not truncated
    .... ..1... .. = Recursion desired: Do query recursively
    .... ..1... .. = Recursion available: Server can do recursive queries
    .... ..0... .. = Z: reserved (0)
    .... ..0... .. = Answer authenticated: Answer/authority portion was not authenticated by the server
    .... ..0... .. = Non-authenticated data: Unacceptable
    .... ..0000 = Reply code: No error (0)
  Questions: 1
  Answer RRs: 3
  Authority RRs: 0
  Additional RRs: 1
  Queries
    www.pluralsight.com: type A, class IN
  Answers
    www.pluralsight.com: type CNAME, class IN, cname www.pluralsight.com.cdn.cloudflare.net
    www.pluralsight.com.cdn.cloudflare.net: type A, class IN, addr 104.19.161.127
    www.pluralsight.com.cdn.cloudflare.net: type A, class IN, addr 104.19.162.127
  Additional records
    [Request In: 1]
    [Time: 0.248129702 seconds]
```

[ closed ] · No.: 2 · Time: 0.248129702 · Source: 192.168.177.35 · Destination: 192.168.177.6...sight.com CNAME www.pluralsight.com.cdn.cloudflare.net A 104.19.161.127 A 104.19.162.127 OPT

Close Help

Most of the fields should matchup and make perfect sense. Verify the Datagram Length, Upper-layer protocol and the IP address fields. Are there any interesting features of the data in the identifier/flags/offset fields?

- o Datagram Length=56
- o Upper-Layer protocol-IP
- o IP ADDRESS fields: source-10.20.204.75 destination-8.8.8.8

In class, we discussed the TTL field and determined that we didn't know a good way to set this. What does your OS set this field to? o 54 o OS:MAC OS Monterey o OS VERSION:12.2(21D49)

## Step 5: ICMP

33) In a terminal window, execute the traceroute utility to trace from your computer to www.cmuj.jp or www.regjeringen.no or some other far-away destination (like we did in our class). If you are having trouble with the weird traceroutes, try this from an on-campus location (your home, a restaurant, etc).

Do whatever you can to get a traceroute consisting of about a dozen steps.

```
Terminal
[deadpool@kraken]~$ ping www.regjeringen.no
PING www.regjeringen.no(2606:4700:9c60:4adf:27c6:b9:fbba:812d (2606:4700:9c60:4adf:27c6:b9:fbba:812d)
) 56 data bytes
64 bytes from 2606:4700:9c60:4adf:27c6:b9:fbba:812d (2606:4700:9c60:4adf:27c6:b9:fbba:812d): icmp_seq
=1 ttl=57 time=116 ms
64 bytes from 2606:4700:9c60:4adf:27c6:b9:fbba:812d (2606:4700:9c60:4adf:27c6:b9:fbba:812d): icmp_seq
=2 ttl=57 time=343 ms
64 bytes from 2606:4700:9c60:4adf:27c6:b9:fbba:812d (2606:4700:9c60:4adf:27c6:b9:fbba:812d): icmp_seq
=3 ttl=57 time=988 ms
64 bytes from 2606:4700:9c60:4adf:27c6:b9:fbba:812d (2606:4700:9c60:4adf:27c6:b9:fbba:812d): icmp_seq
=4 ttl=57 time=1346 ms
64 bytes from 2606:4700:9c60:4adf:27c6:b9:fbba:812d (2606:4700:9c60:4adf:27c6:b9:fbba:812d): icmp_seq
=5 ttl=57 time=326 ms
^C
--- www.regjeringen.no ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4020ms
rtt min/avg/max/mdev = 116.480/624.084/1346.118/464.490 ms, pipe 2
[deadpool@kraken]~$
```

Capturing from any					
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help					
Apply a display filter ... <Ctrl-/>					
No.	Time	Source	Destination	Protocol	Length Info
1	0.000000000	192.168.177.63	192.168.177.35	DNS	80 Standard query 0x1dea A www.regjeringen.no
2	0.000123273	192.168.177.63	192.168.177.35	DNS	80 Standard query 0xf3ed AAAA www.regjeringen.no
3	0.874531628	192.168.177.35	192.168.177.63	DNS	112 Standard query response 0x1dea A www.regjeringen.no A 104.18.2.141 A ...
4	0.874584558	192.168.177.35	192.168.177.63	DNS	108 Standard query response 0xf3ed AAAA www.regjeringen.no AAAA 2606:4700...
5	0.876970702	192.168.177.63	192.168.177.35	DNS	134 Standard query 0x0ef7 PTR d.2.1.8.a.b.b.f.9.b.0.6.c.7.2.f.d.a.4.0.6...
6	0.879736793	192.168.177.35	192.168.177.63	DNS	134 Standard query response 0x0ef7 No such name PTR d.2.1.8.a.b.b.f.9.b.0...
7	0.883613179	2405:204:5413:5dc1::...	2606:4700:9c60:4adf::...	ICMPv6	120 Echo (ping) request id=0x934f, seq=1, hop limit=64 (reply in 8)
8	1.000011869	2606:4700:9c60:4adf::...	2405:204:5413:5dc1::...	ICMPv6	120 Echo (ping) reply id=0x934f, seq=1, hop limit=57 (request in 7)
9	1.000890474	192.168.177.63	192.168.177.35	DNS	134 Standard query 0x8a1a PTR d.2.1.8.a.b.b.f.9.b.0.6.c.7.2.f.d.a.4.0.6...
10	1.004498035	192.168.177.35	192.168.177.63	DNS	134 Standard query response 0x8a1a No such name PTR d.2.1.8.a.b.b.f.9.b.0...
11	1.885274874	2405:204:5413:5dc1::...	2606:4700:9c60:4adf::...	ICMPv6	120 Echo (ping) request id=0x934f, seq=2, hop limit=64 (reply in 12)
12	2.228351107	2606:4700:9c60:4adf::...	2405:204:5413:5dc1::...	ICMPv6	120 Echo (ping) reply id=0x934f, seq=2, hop limit=57 (request in 11)
13	2.229179738	192.168.177.63	192.168.177.35	DNS	134 Standard query 0x4581 PTR d.2.1.8.a.b.b.f.9.b.0.6.c.7.2.f.d.a.4.0.6...
14	2.231764643	192.168.177.35	192.168.177.63	DNS	134 Standard query response 0x4581 No such name PTR d.2.1.8.a.b.b.f.9.b.0...
15	2.885220961	2405:204:5413:5dc1::...	2606:4700:9c60:4adf::...	ICMPv6	120 Echo (ping) request id=0x934f, seq=3, hop limit=64 (reply in 16)
16	3.872858608	2606:4700:9c60:4adf::...	2405:204:5413:5dc1::...	ICMPv6	120 Echo (ping) reply id=0x934f, seq=3, hop limit=57 (request in 15)
17	3.873784845	192.168.177.63	192.168.177.35	DNS	134 Standard query 0x6b14 PTR d.2.1.8.a.b.b.f.9.b.0.6.c.7.2.f.d.a.4.0.6...
18	3.879088347	192.168.177.35	192.168.177.63	DNS	134 Standard query response 0x6b14 No such name PTR d.2.1.8.a.b.b.f.9.b.0...
19	3.883971438	2405:204:5413:5dc1::...	2606:4700:9c60:4adf::...	ICMPv6	120 Echo (ping) request id=0x934f, seq=4, hop limit=64 (reply in 25)
20	4.903767805	2405:204:5413:5dc1::...	2606:4700:9c60:4adf::...	ICMPv6	120 Echo (ping) request id=0x934f, seq=5, hop limit=64 (reply in 26)
21	5.223525093	HonHaiPr_4c:61:23		ARP	44 Who has 192.168.177.35? Tell 192.168.177.63
22	5.225082725	86:8e:3c:00:66:f5		ARP	44 Who has 192.168.177.63? Tell 192.168.177.35
23	5.225095642	HonHaiPr_4c:61:23		ARP	44 192.168.177.63 is at 74:40:bb:4c:61:23
24	5.226290435	86:8e:3c:00:66:f5		ARP	44 192.168.177.35 is at 86:8e:3c:00:66:f5
25	5.230003434	2606:4700:9c60:4adf::...	2405:204:5413:5dc1::...	ICMPv6	120 Echo (ping) reply id=0x934f, seq=4, hop limit=57 (request in 19)
26	5.230031724	2606:4700:9c60:4adf::...	2405:204:5413:5dc1::...	ICMPv6	120 Echo (ping) reply id=0x934f, seq=5, hop limit=57 (request in 20)
27	5.230800911	192.168.177.63	192.168.177.35	DNS	134 Standard query 0x7966 PTR d.2.1.8.a.b.b.f.9.b.0.6.c.7.2.f.d.a.4.0.6...
28	5.233587599	192.168.177.35	192.168.177.63	DNS	134 Standard query response 0x7966 No such name PTR d.2.1.8.a.b.b.f.9.b.0...
29	5.235120247	192.168.177.63	192.168.177.35	DNS	134 Standard query 0x6e85 PTR d.2.1.8.a.b.b.f.9.b.0.6.c.7.2.f.d.a.4.0.6...

Used ping as traceroute in my system is not working for some reason

35) What are the transmitted

segments like? Describe the important features of the segments you observe. In particular, examine the destination port field. What characteristics do you observe about this port number and why would it be chosen so?

o Ping doesn't use a port number as traceroute uses port number 33434

o For every hop port number increases by 136)

What about the return packets? What are the values of the various header fields?

- Code:0
- Type:0
- Identifier:44969 (BE)
- Identifier:43439 (LE)
- Sequence number:1 (BE)
- Sequence Number:256 (LE)

37) The ICMP packets carry some interesting data. What is it? Can you show the relationship to the sent packets?

- It contains timestamp

38) Lab1 asserted that ping operates in a similar fashion to traceroute. Use Wireshark to show the degree to which this is true. What differences and similarities are there between the network traffic of ping versus traceroute?

- Did for ping