# Week #5

**Week 5 - Understanding Transport layer and Network Layer**

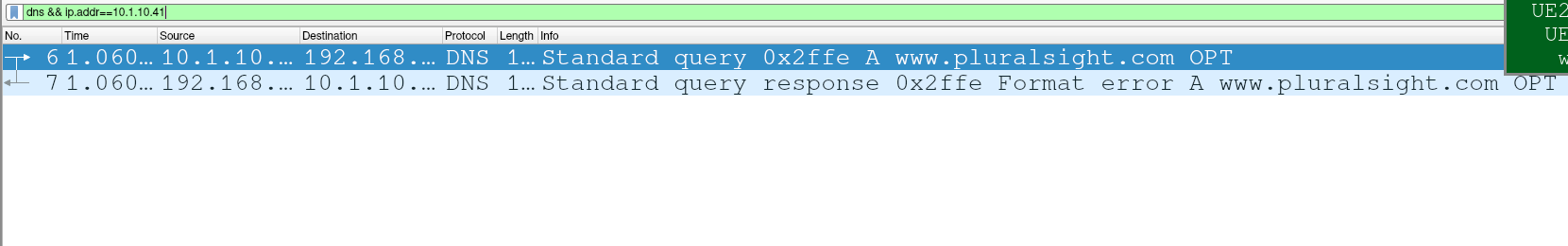
**SRN : PES2UG20CS237**

**Name : P K Navin Shrinivas**

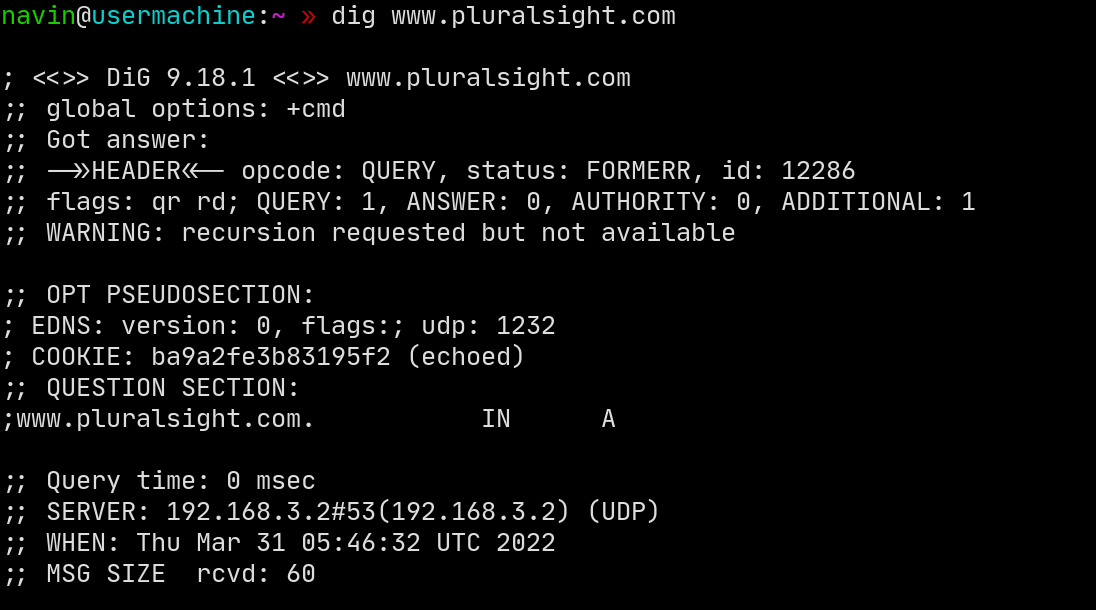
**Section : D**

## Task 1 :UDP and DNS

* Opening wireshark and filtering for DNS and UDP signals specific to my ip address.
* Pinging [www.pluralsight.com](http://www.pluralsight.com) to generate DNS packets.

****

Filtered packets

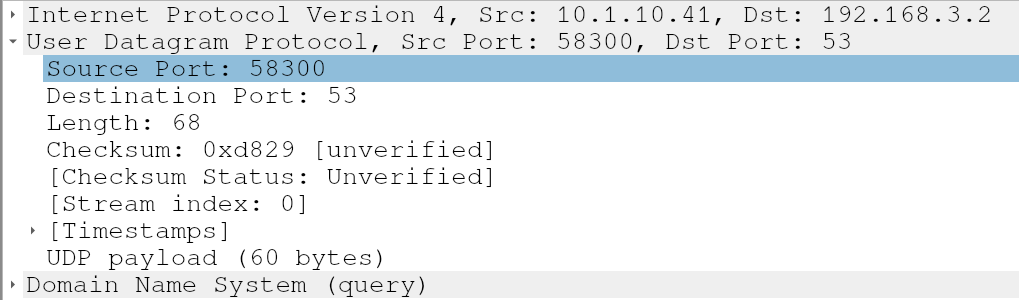
****

dig command to generate UDP packets

## (Q) My predictions for UDP structure :

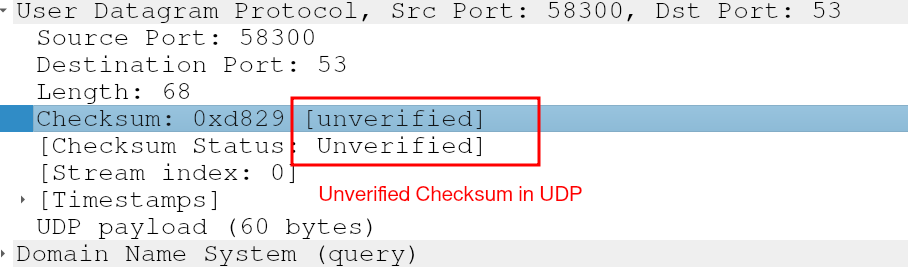
|  |  |
| --- | --- |
| 2 bytes for source port | 2 bytes for dest port |
| 2 bytes for length | 2 bytes for UDP checksum |
| X bytes for payload | X bytes of payload |

## (Q) And in wireshark : Hence concluding predictions were right!



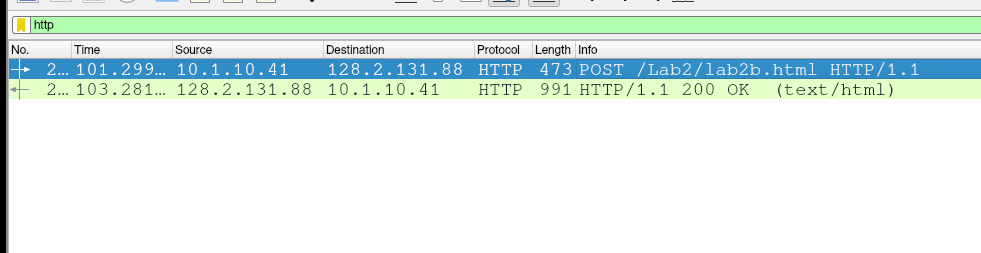
UDP in Wireshark

* Why is the checksum “unverified” in wireshark UDP packets? It is because these UDP packets were genrated by dig, iptrace or tcpdump will calculate these checksums and elad to “verified” status.

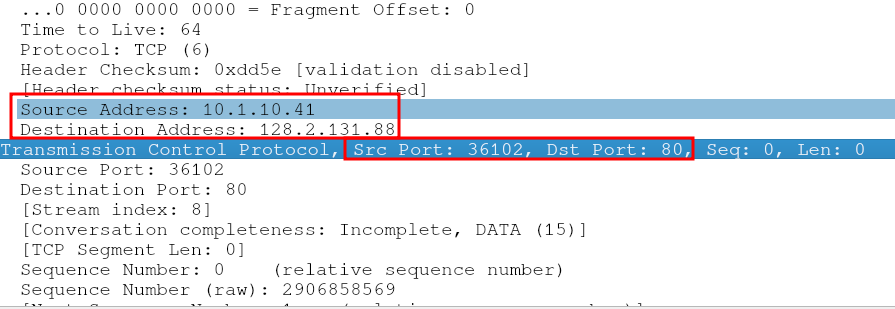


## Task 2 :TCP

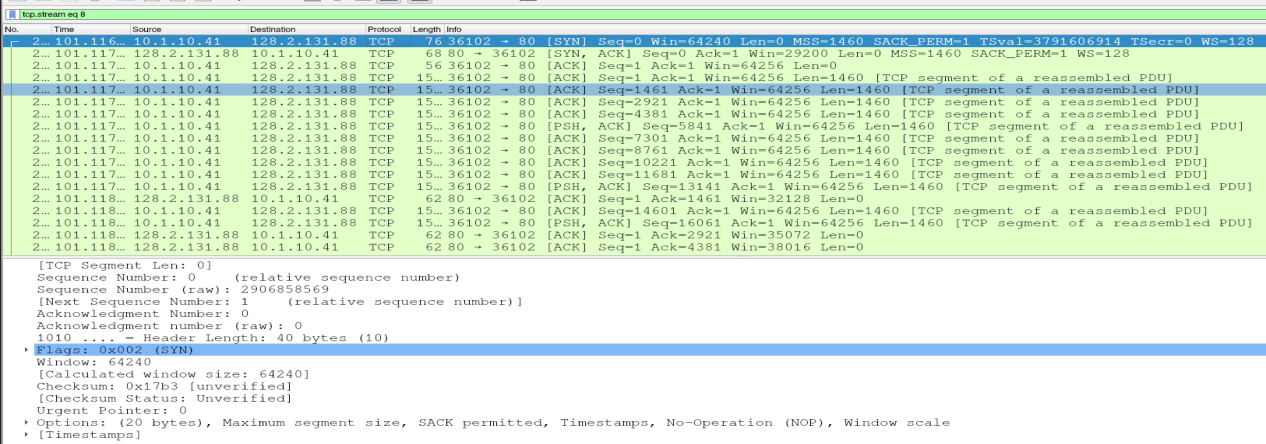
* Download the file from <http://www.gutenberg.org/ebooks/2383.txt.utf-8>
* Open up wireshark and keep it prime to filter TCP packets from and to <http://www.ini740.com/Lab2/lab2a.html>
* Upload the files and stop wireshark, wireshark output :



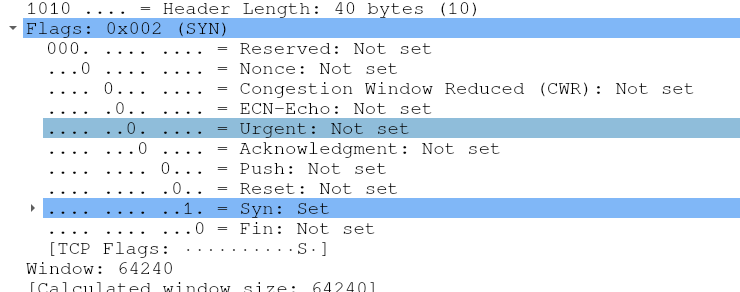
* (Q) What is the source/dest ip address and source port? IP :



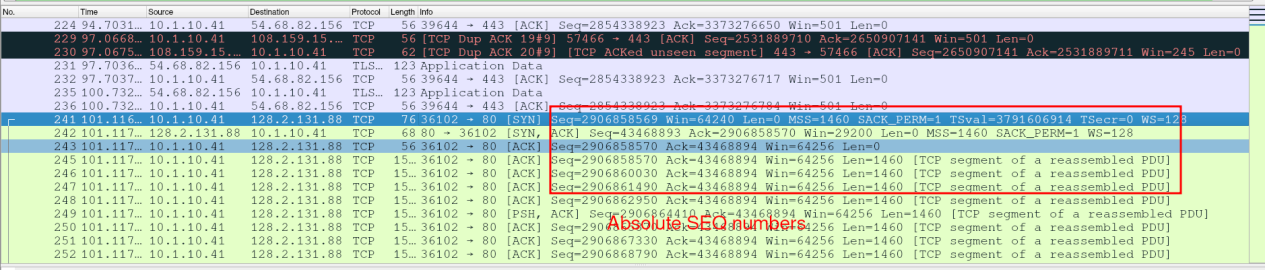
* Point of this task is to observe TCP packets, hence filter for them :



## Task 2b : TCP Basics

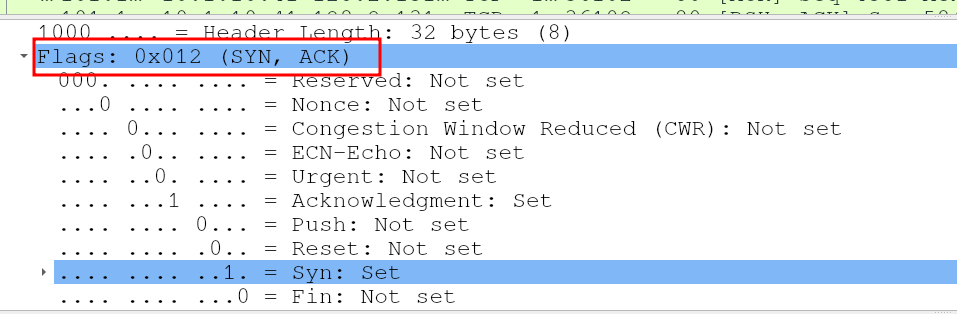
* (Q14) Sequence number of SYN packets are 0, SYN segments can be identified using the TCP flag segments. Yes wireshark can display absolute seq numbers by : edit->prefrence->protocols->tcp->untick relative numbering.

SYN Identification

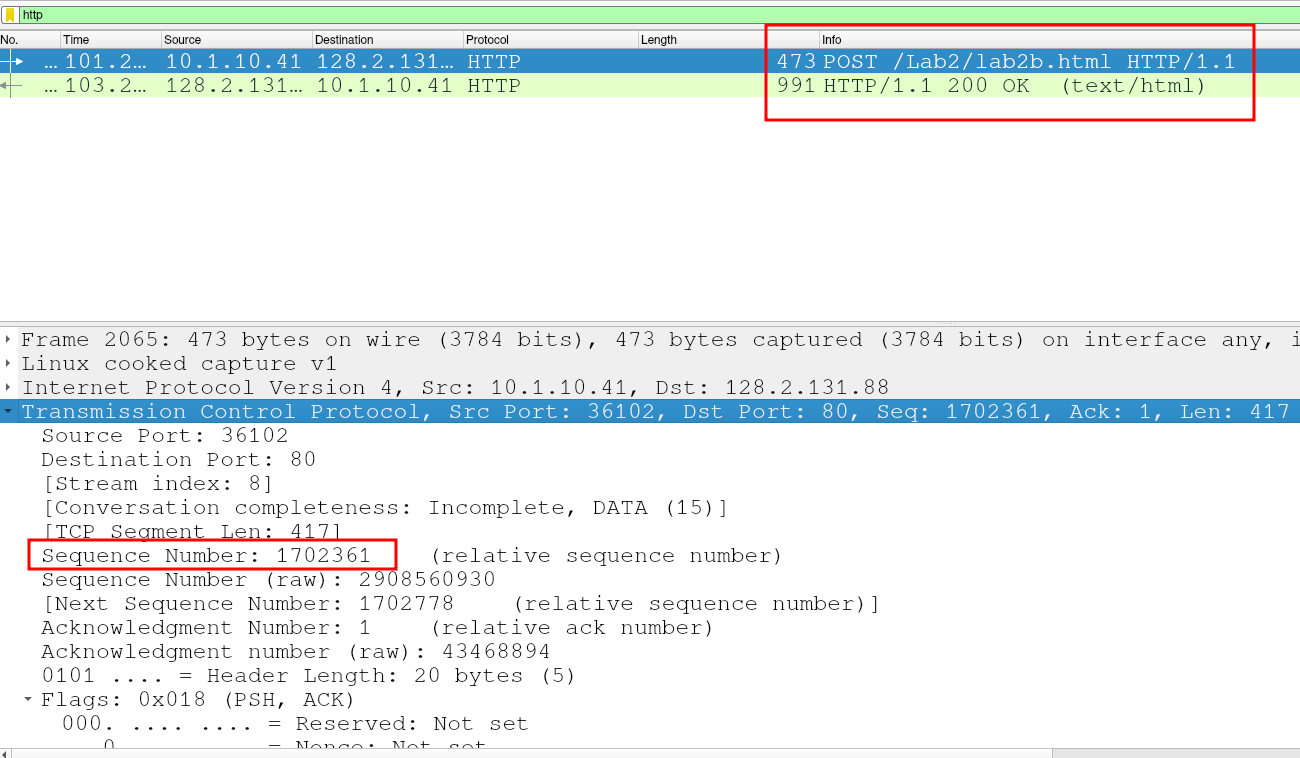


Relative TCP seq numbering

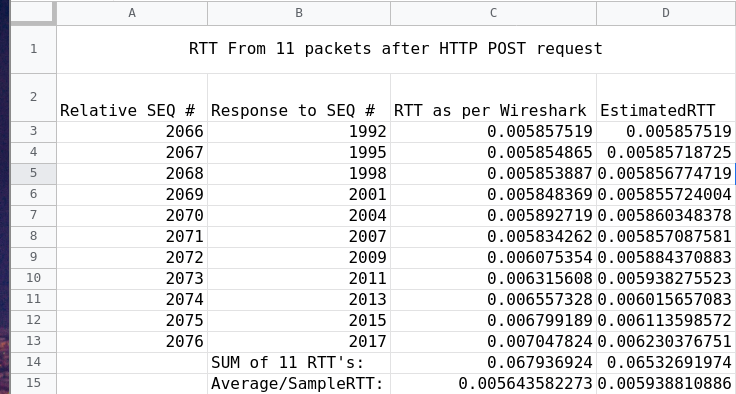
* (Q15) SYNACK has a relative sequence number of 1, ACK number of SYNACK packets is 1, this is because the server is asking for the best bit waiting for the three way handshake.The Flag field in TCP packets shows it is a SYNACK packet:



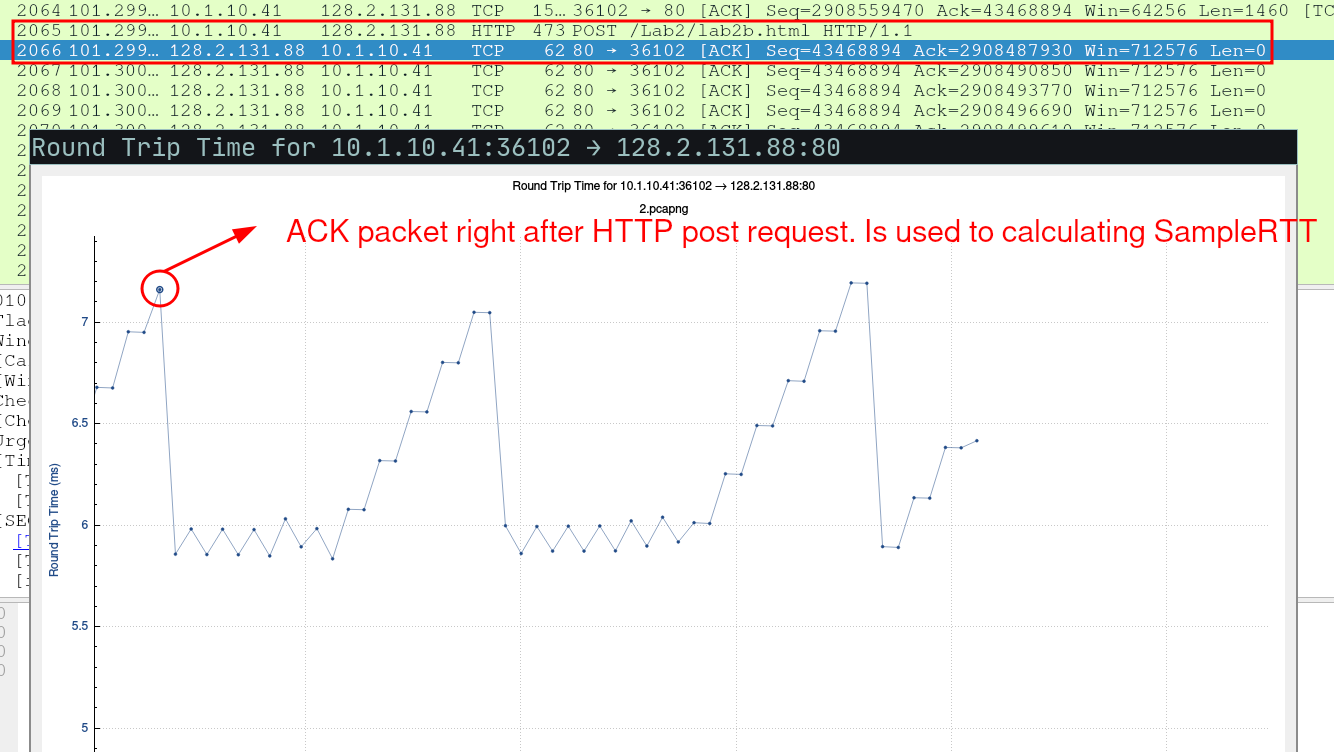
* (Q16) TCP SEQ number (relative) of HTTP post is : 1702361, this I feel is the entirty of uploaded content and remaining payload (1702278) are for headers.



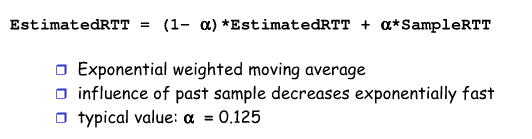
* (Q17)Sample RTT is simply the time taken to ACK a given segment, but to get smoother EstimatedRTT we take many samples from recent packets :



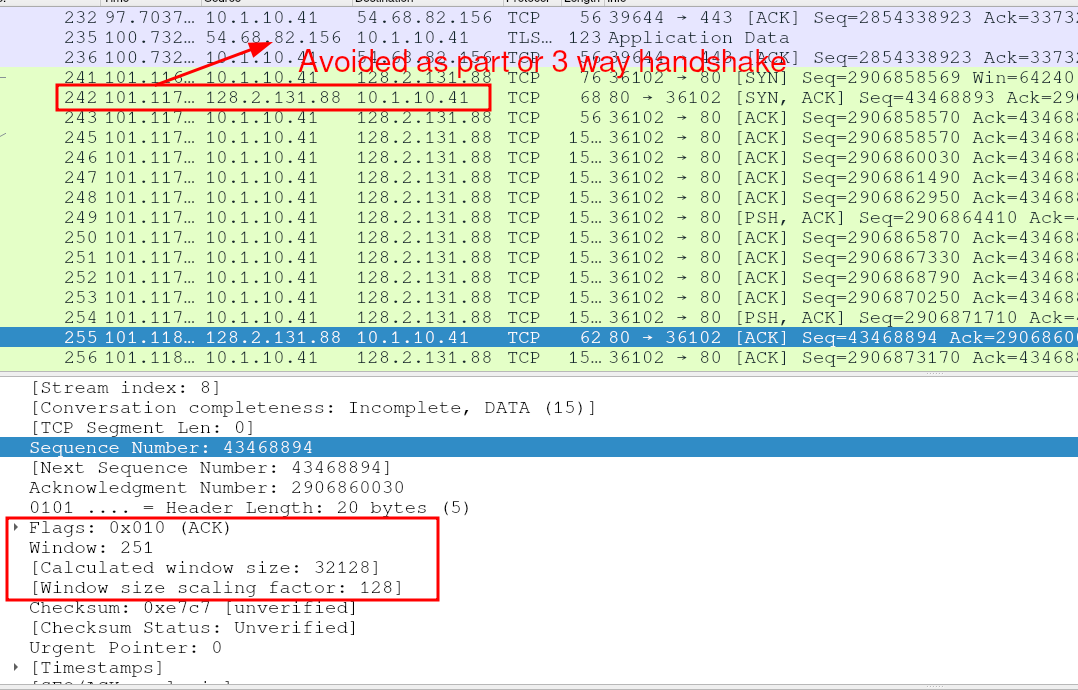
[Source](https://docs.google.com/spreadsheets/d/1hQjS8L1xW5pE3Ya-bxX8tmEiIFyVGTkClU3lQkpCMIc/edit?usp=sharing)



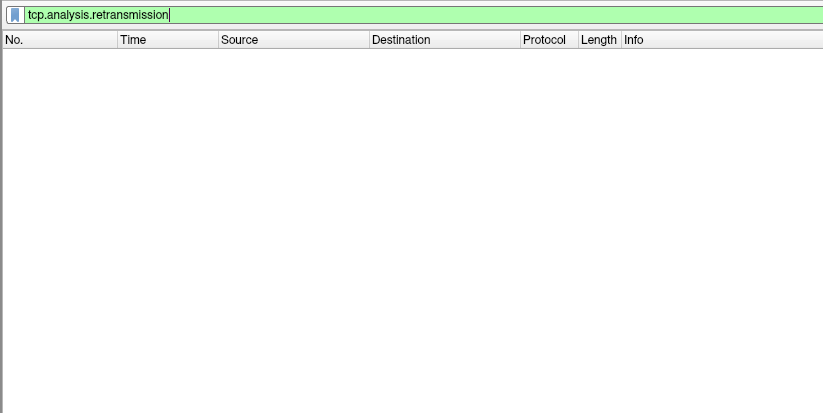
Now calculating the EstimatedRTT (In Table):



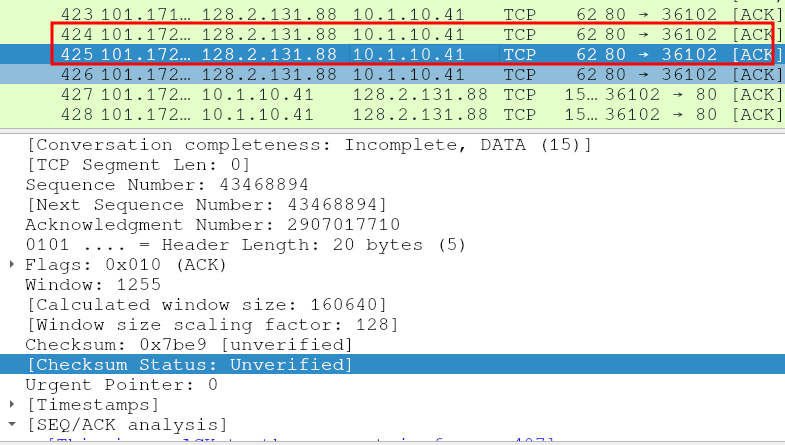
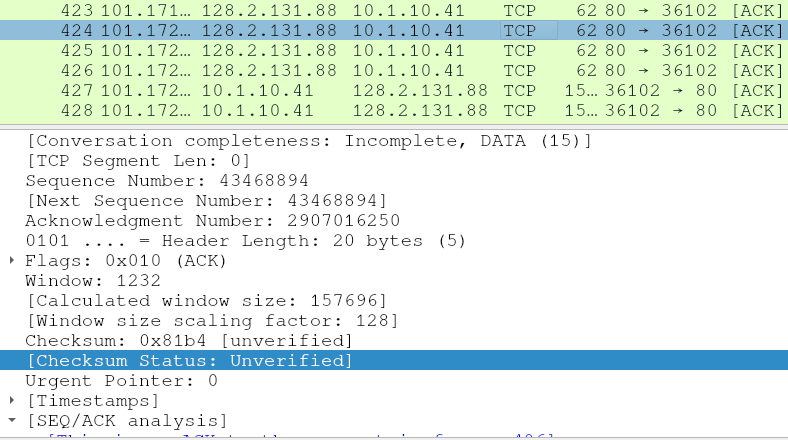
* (Q18)The window size on packets from A to B indicate how much buffer space is available on A for receiving packets. So when B receives a packet with window size 1, it would tell B how many bytes it is allowed to send to A. Hence looking at the window sizes here on reponses (ACK). We also see the windows are scaled with a factor of 128. Lowest observed window size : 32128



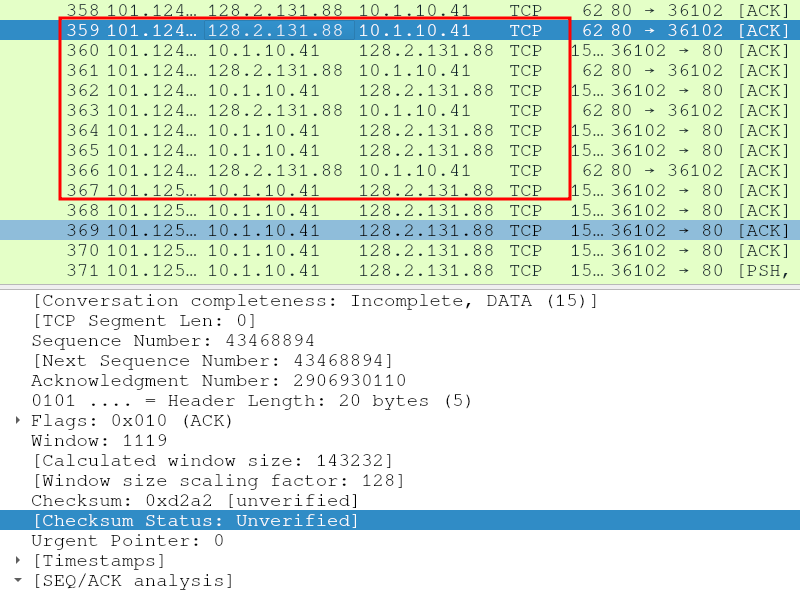
* (Q19)No, there are no retransmitted packets, I checked for these by filtering them using : tcp.analysis.retrasmission :



* (Q20)Comapring Sequence number of 2 consecutive ACK’s will give us the answer we need, I am comparing these two :



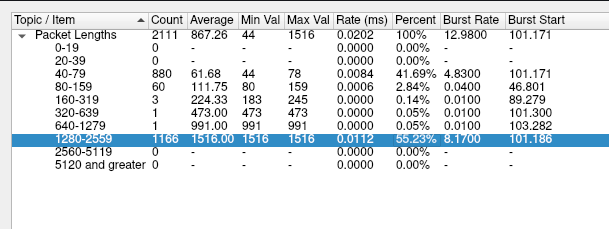
Subracting the two sequence number gives us : 1460, this the amount of bytes ACK’ed in each ACK response.Also, here we can see where the server is ACK’ing every other packet :



* (Q21)The HTTP reponse has a “time since request” value of 1.98 seconds, the HTTP POST has a content size of 1702219 bytes. This gives us a throughput of : ~840KB/second

## Task 2c : Statistics

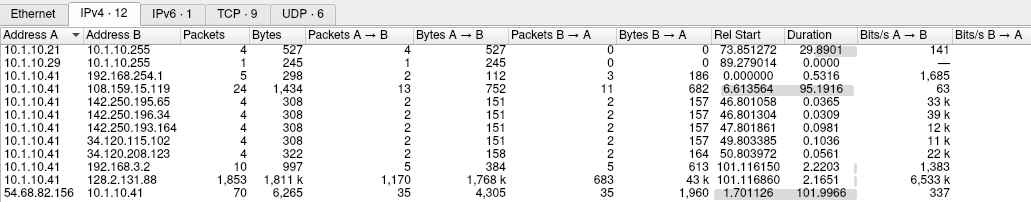
* (Q22) The most common packet length is 1280-2559 :



* (Q23) Throughput and other statistics :

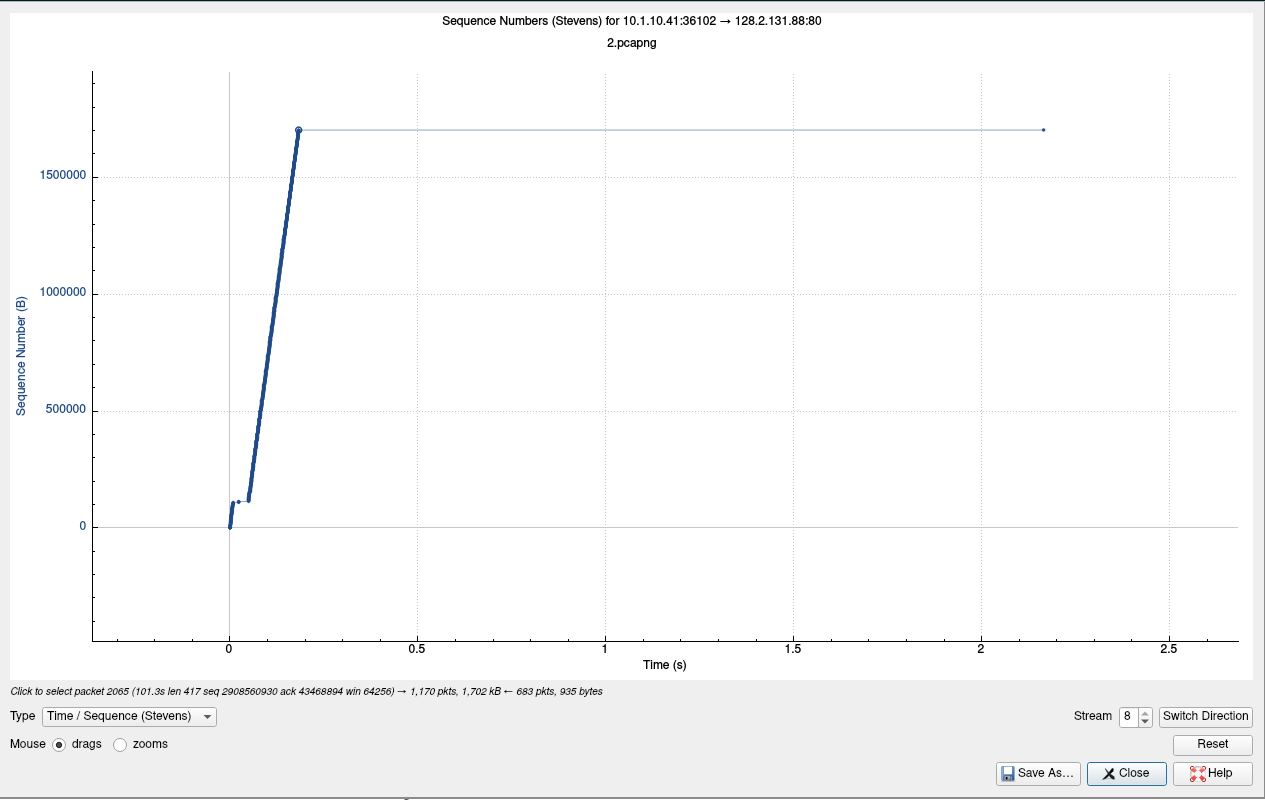


* (Q24)Details about converstation between hosts is as follows :



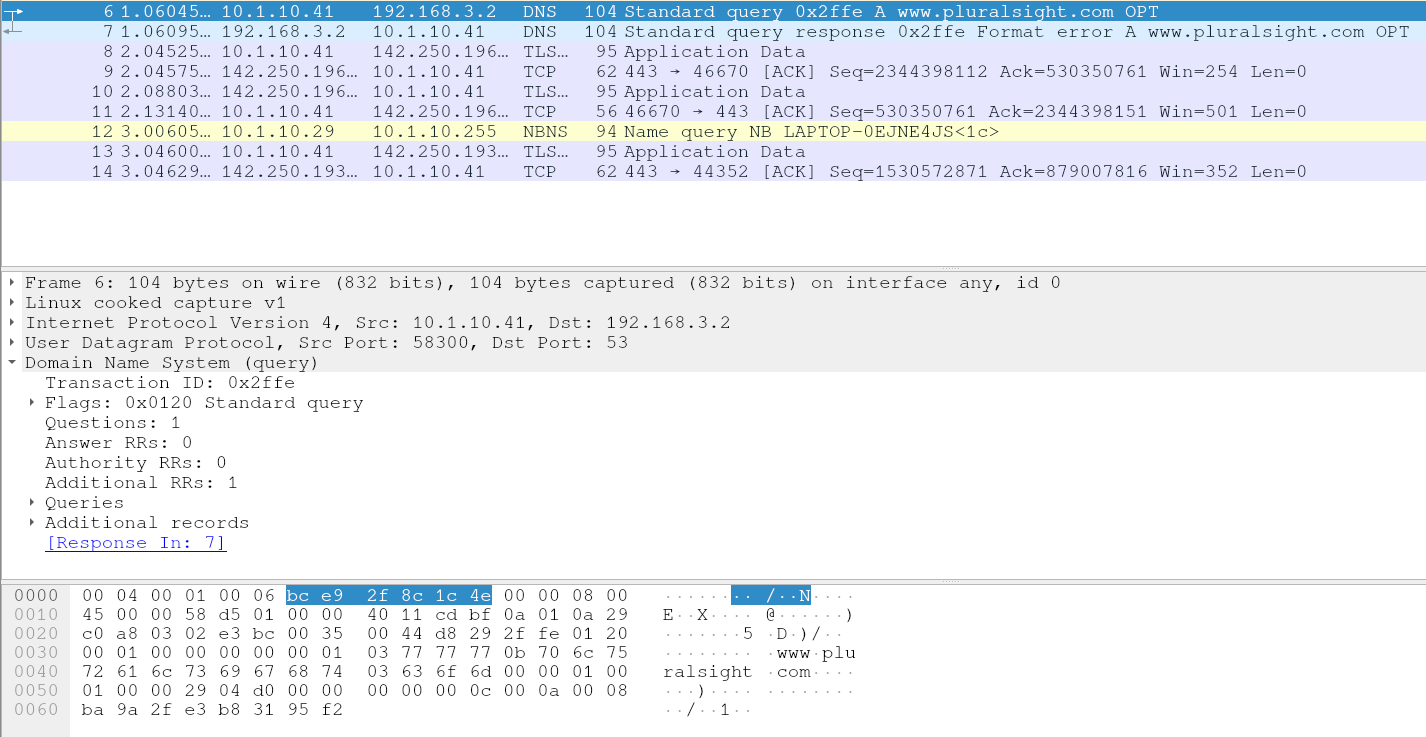
## Task 3 : Congestion Control

* (Q25 and Q26)The observed graph using wiresharks TCP graphs is :



## Task 4 : Network layer

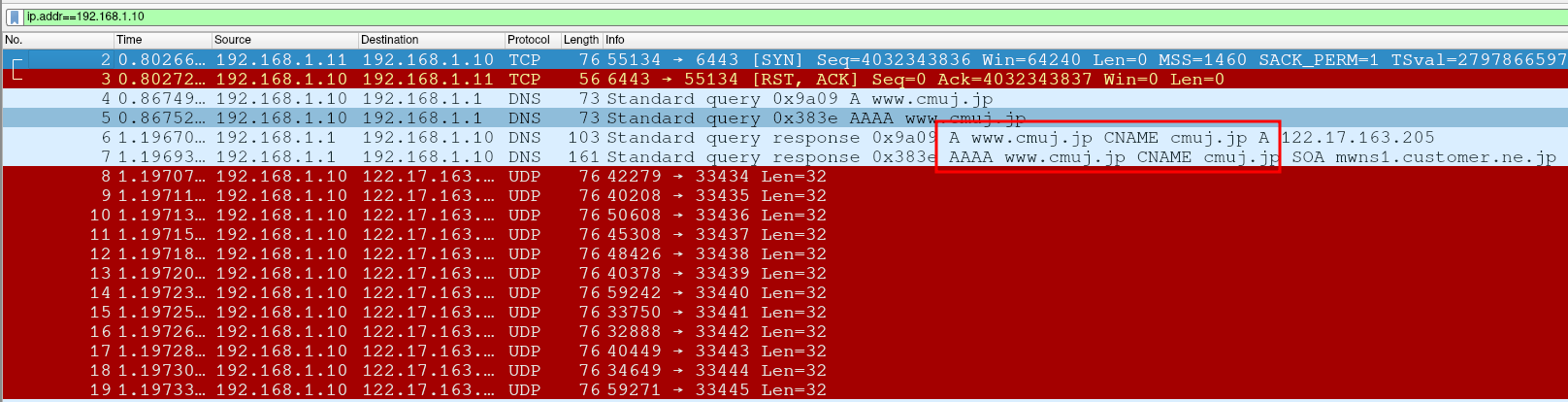
* The DNS query was made with [www.pluralsight.com](http://www.pluralsight.com) as the Domain name. These were the observed query packets in wireshark



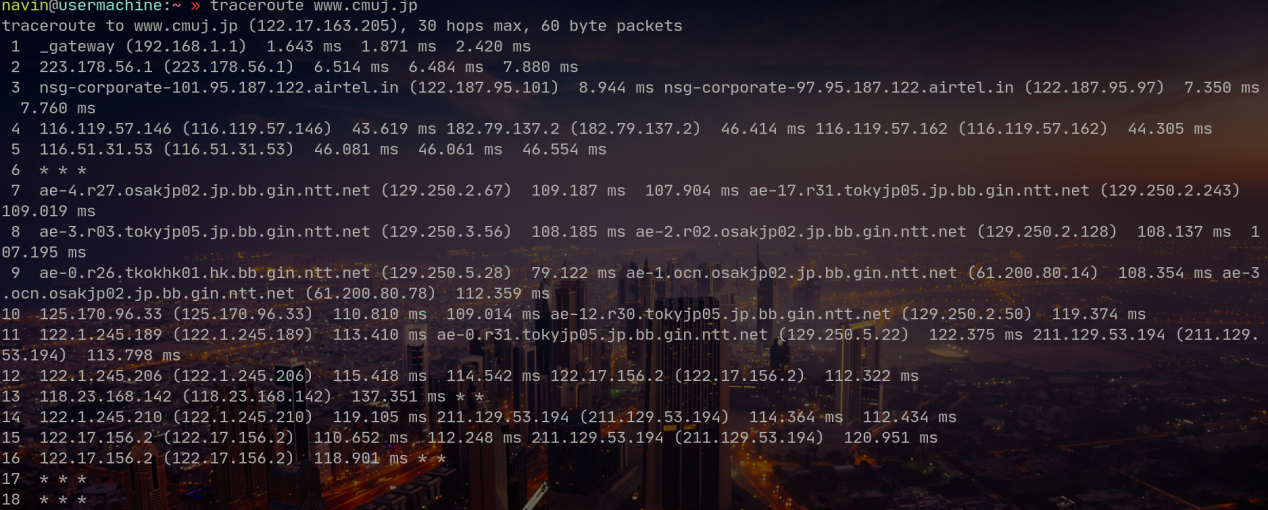
* (Q28 and Q29) All the needed fields do match up
* (Q31) TTL is set as 64, my OS being arch derivative of linux running on linux 5.16.

## Task 5 : ICMP

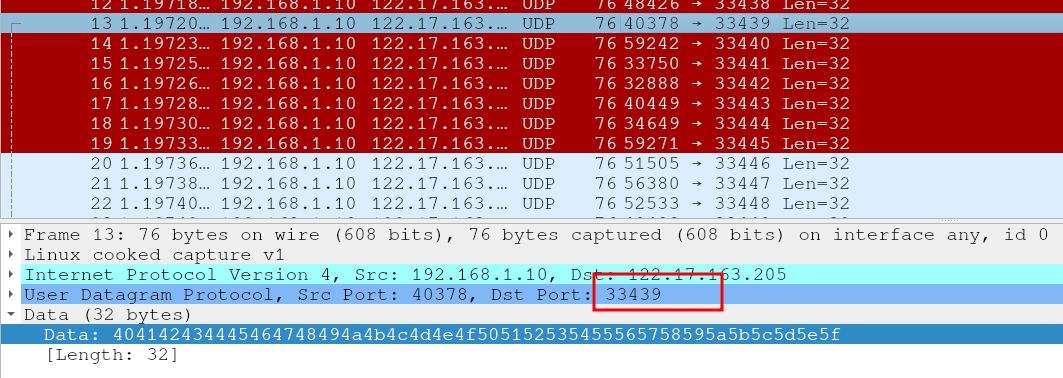
* Starting a new capture when simultaneously pinging : <http://www.cmuj.jp/> using the traceroute utility :

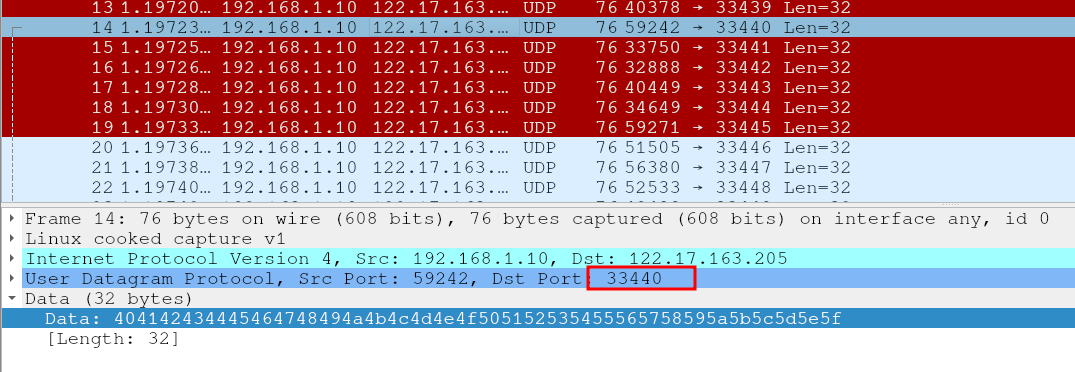


DNS queries of traceroute

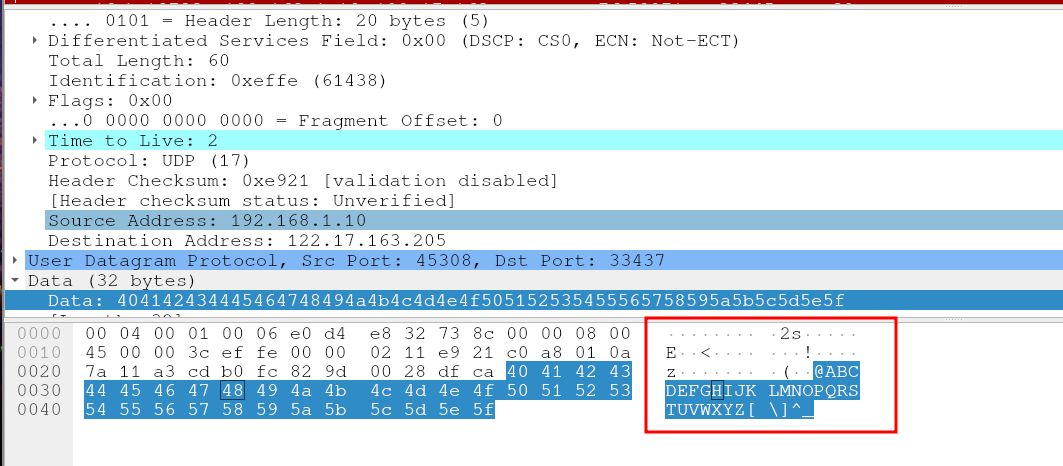


* (Q35) The dest port increases by one in every sent packet, indicating that traceroute tries to reach the server in multiple ports, as seen here:





* My chances at traceroute did not get a reponse back from the server, hence couldn’t observe those packet.
* (Q37) ICMP packet do contain very intresting data values, The alphabets as seen here :



* (Q38) We can observe the ping request and reponse followed by a bunch of UDP packets to a fixed port of 443 with random data.

