

## **Assignment 2**

### **Wireshark packet trace analysis**

Mayank Sharma 194101030

Wireshark version 2.6.10 is used for capturing the internet traffic and analyse the client server at various layers of network. The wireshark captures a total of 35205 packets for about 350 seconds. There are a lot of network addresses, ports and protocols involved in the process and different formats and sizes of packets have been observed. Most of the packets transferred are for TCP, UDP, DNS and HTTP.

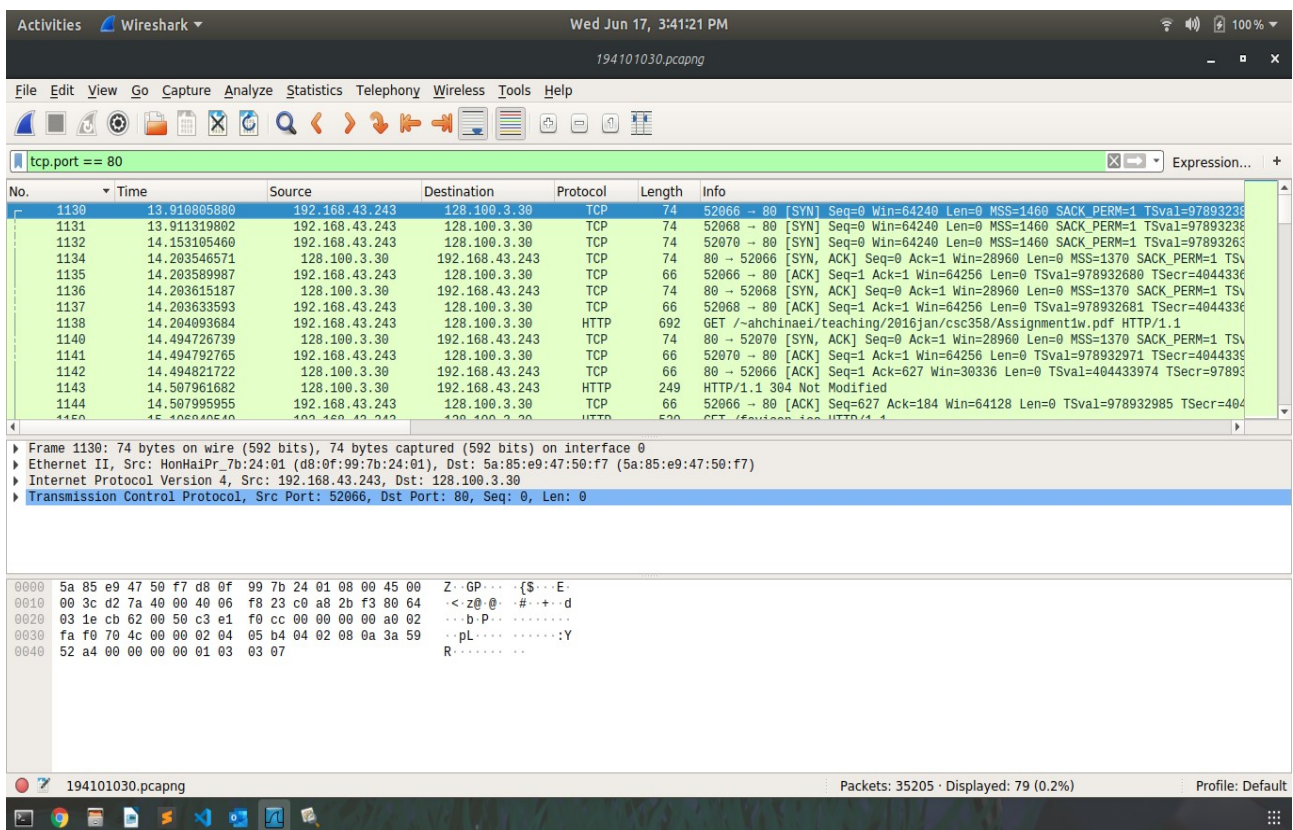
There are various packet formats observed in the packet trace which are given below:

- HTTP
- TCP
- UDP
- ARP
- DNS
- ICMPv6
- MDNS
- NTP
- SSDP
- TLSv1.2
- TLSv1.3

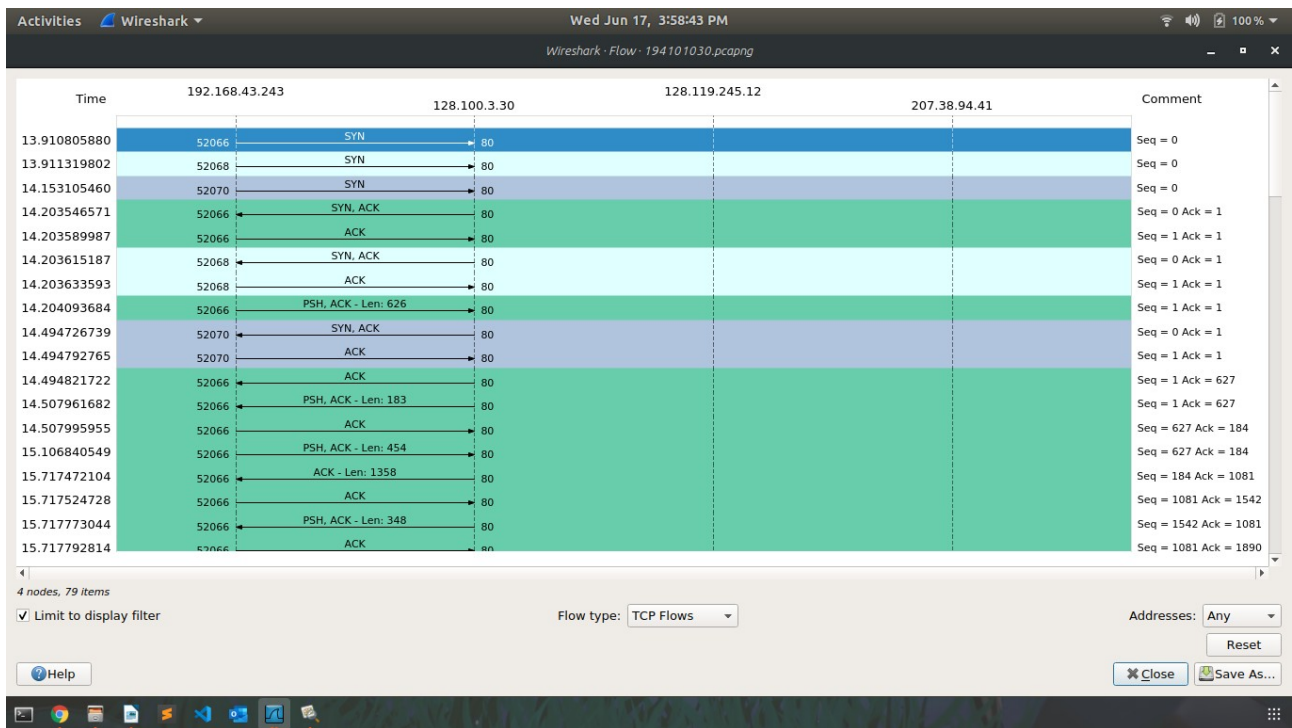
There are a lot of protocols observed in the trace including common protocols like HTTP, TCP, UDP, ARP and DNS, and some rare ones like ICMPv6, MDNS, NTP, SSDP, TLSv1.2 and TLSv1.3. Now, we will analyse each protocol by considering some parameters like number of packets, types of messages in packets, header fields, etc.

# TCP

Transmission Control Protocol is a Transport layer protocol which deals with transmission of segments. The trace of the TCP segments received and sent from our computer is obtained using Wireshark. There are 34326 TCP packets transferred in a span of around 350 seconds, which is more than 90% of the total packets captured. From the snapshot given below, it has been observed that the initial three way handshake is used for the connection establishment.

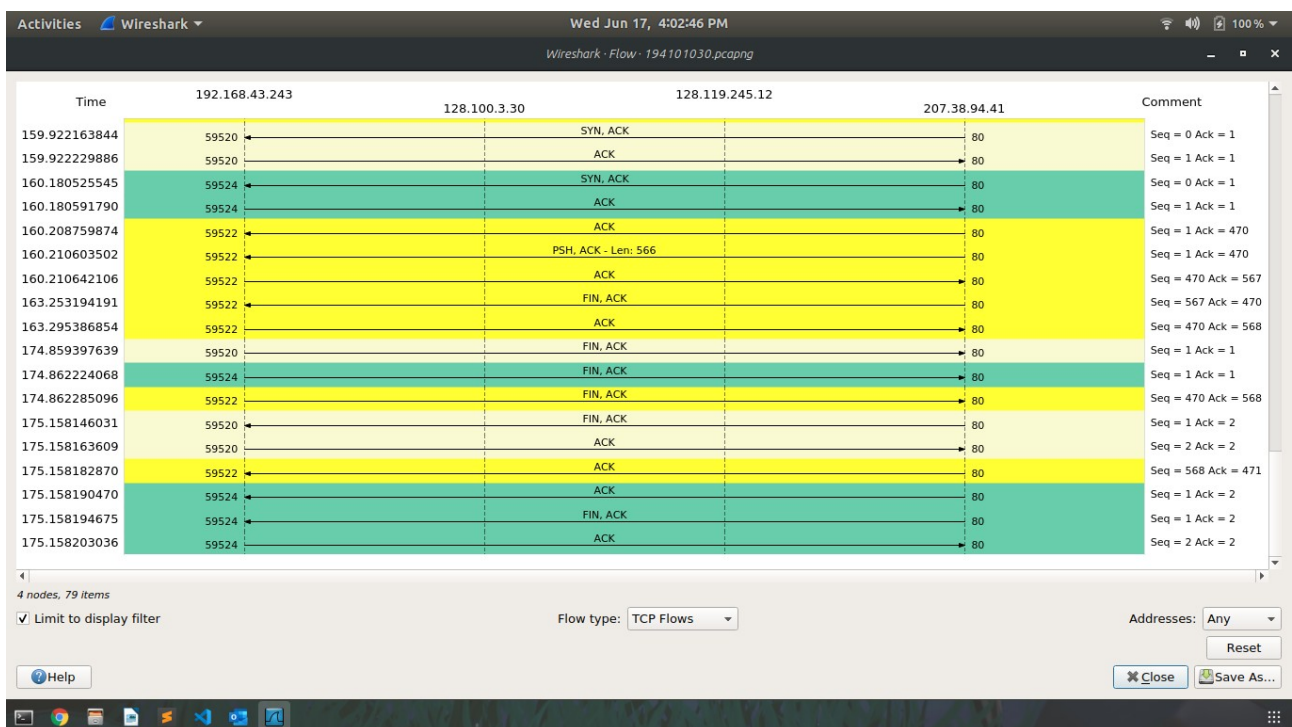


There is an option available in Wireshark to observe the flow of packets of different protocols, we can also choose TCP flow here. From this picture, the analysis of TCP packets can be done easily, especially for the SYN packets at the connection establishment phase and FIN packets at the connection termination phase.



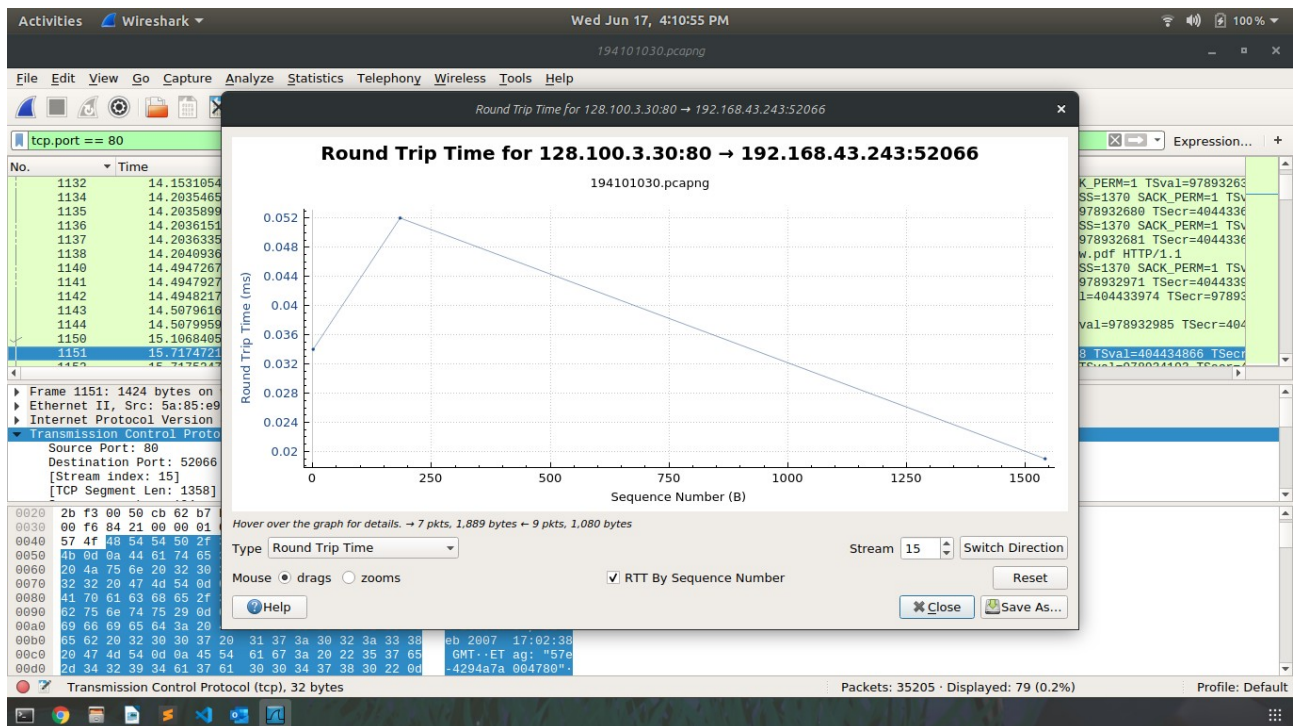
It can be seen that there are multiple network addresses involved in the TCP flow of packets.

We will observe only the flow from 192.168.43.243 to 128.100.3.30. Initially, few SYN packets are transferred to establish the connection and thenafter some data packets are sent.

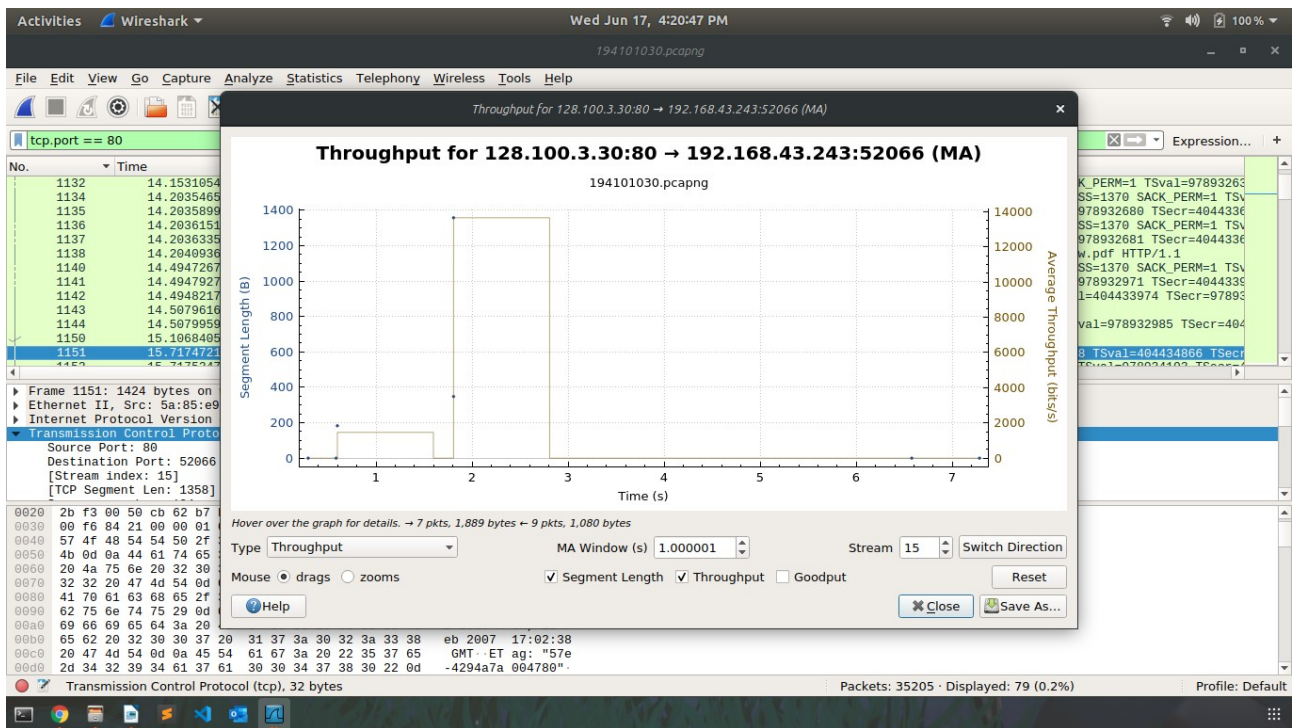


In the second picture, we can see some FIN packets, that are being used to terminate the connection.

One more feature of wireshark is that we can have the Round Trip Time for the packets sent over a particular connection. The picture given below is about RTT of the packets sent for the connection from 128.100.3.30: port 80 to 192.168.43.243: port 52066 (sorted based on the sequence number).



Another salient feature of wireshark TCP throughput for the packets sent over a particular connection. The picture given below is about throughput of the total packets sent for the connection from 128.100.3.30: port 80 to 192.168.43.243: port 52066 (sorted based on the sequence number).



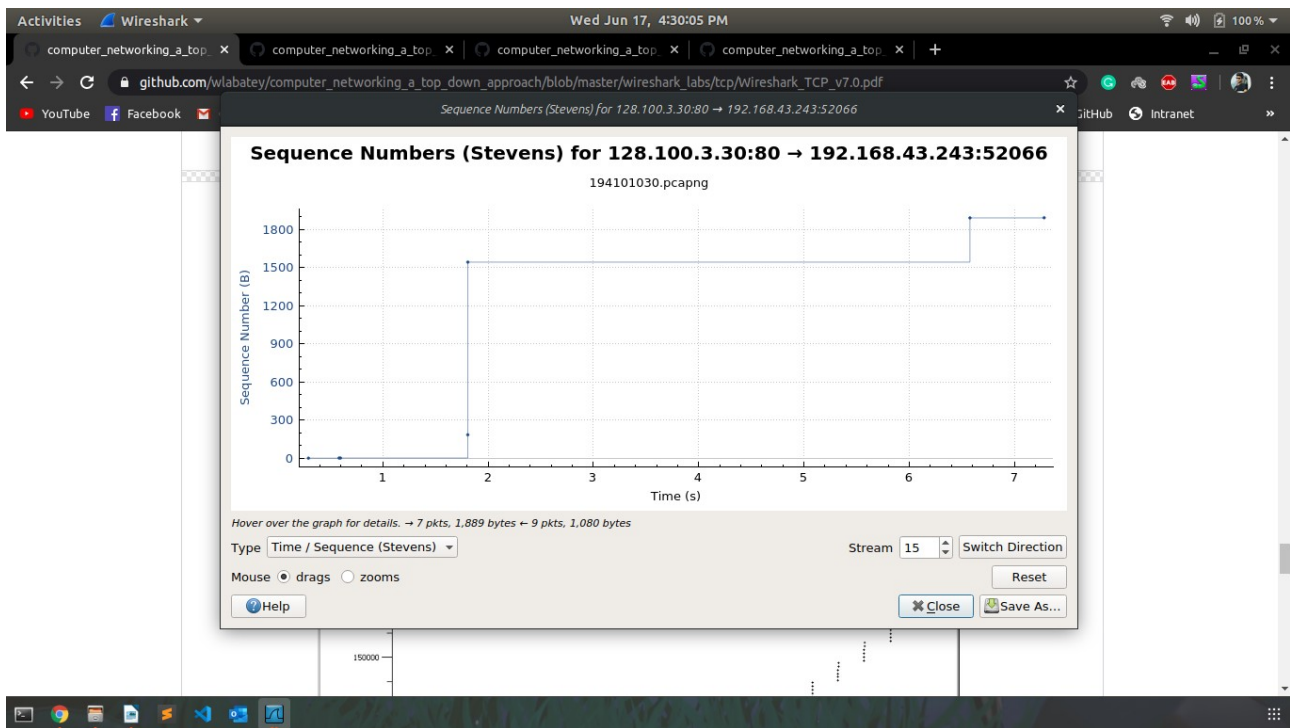
For this connection, throughput has risen upto 13500 bits/sec.

Let's now observe the amount of data packets sent per unit time between client and server.

It is again for the same connection from 128.100.3.30: port 80 to 192.168.43.243: port 52066 (sorted based on the sequence number).

In the picture given below, a dot represents that a packet is sent at that time with that sequence number.





## UDP

The web browser uses HTTP protocol for accessing web documents and HTTP is an application layer protocol. The HTTP payload is always encapsulated in TCP segments (not UDP segments). Thus, in order to generate UDP packets, we have to make a connection to File Transfer Protocol (TFTP) or write a program using UDP sockets or use UDP based protocols like Domain Name System (DNS), Network Time Protocol (NTP) or Dynamic Host Configuration Protocol (DHCP).

The screenshot given below shows some UDP packets and port numbers of corresponding UDP based protocols.

Wireshark - UDP Multicast Streams · 194101030.pcapng

Source Address	Source Port	Destination Address	Destination Port	Packets	Packets/s	Avg BW (bps)	Max BW (bps)	Max Burst	Bur
2409:4043:2d17:f644:91a6:c27d:8c48:e94	5353	ff02::fb	5353	3	0.06	50	0	1 / 100ms	
192.168.43.243	5353	224.0.0.251	5353	3	0.06	40	0	1 / 100ms	
192.168.43.243	45065	239.255.255.250	1900	4	1.33	2.268	0	1 / 100ms	
192.168.43.243	33115	239.255.255.250	1900	4	1.33	2.270	0	1 / 100ms	
192.168.43.243	33569	239.255.255.250	1900	4	1.33	2.270	0	1 / 100ms	

5 streams, avg bw: 78bps, max bw: 0bps, max burst: 1 / 100ms, max buffer: 2048

Burst measurement interval (ms): 100      Burst alarm threshold (packets): 50      Buffer alarm threshold (B): 10000

Stream empty speed (Kb/s): 5000      Total empty speed (Kb/s): 100000

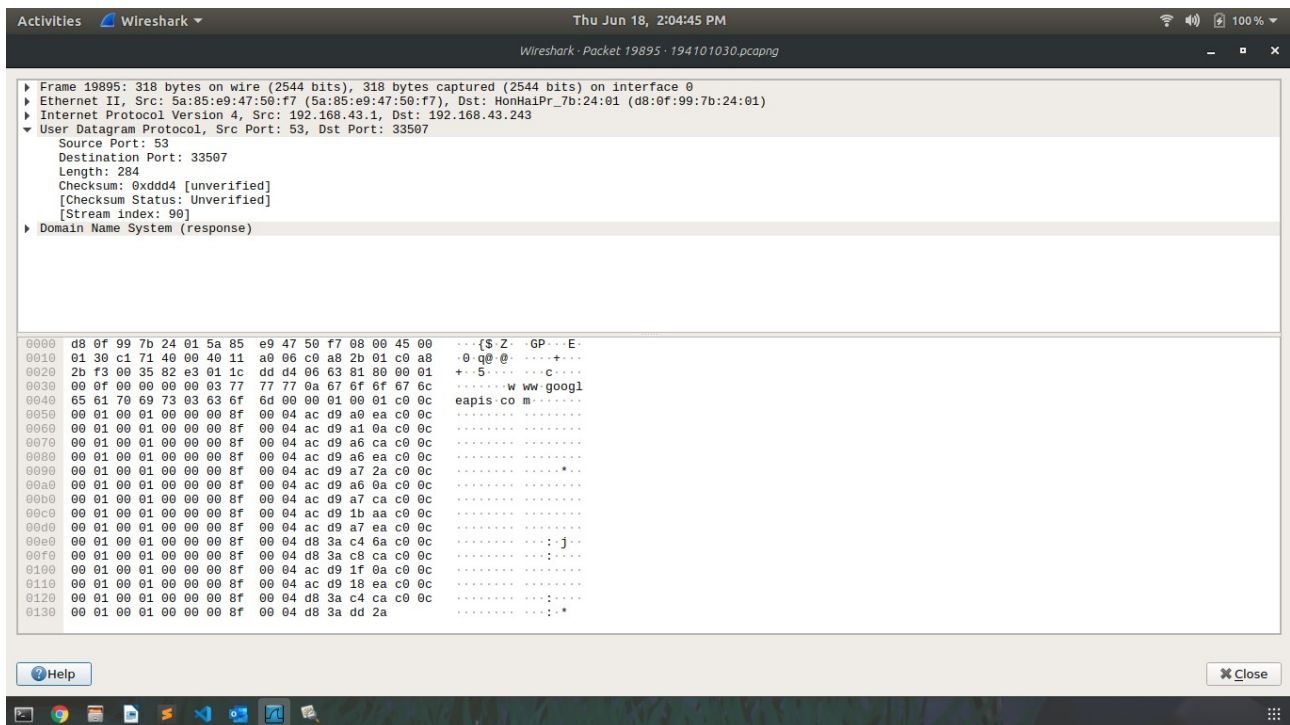
Display filter: Enter a display filter ...      Apply      Copy      Save as...      Close

The following UDP based protocols have been observed in the wireshark packet trace using web browser;

- DNS (UDP port number 53)
- SSTP (UDP port number 1900)
- NTP (UDP port number 123)
- MDNS (UDP port number 5353)

Let's analyze one UDP (DNS with port number 53) packet. We have taken frame number 19895 that has total of 318 bytes on wire. This packet has been sent from IP address 192.168.43.1 to 192.168.43.243.

A typical header of a UDP packet consists of source port number, destination port number, length and checksum (2 bytes each makes total header size to 8 bytes). The selected segments has 53 as source port number, 33507 as destination port number, 284 bytes as length and some checksum bits shown in hexadecimal format.



## DNS

The Domain Name System (DNS) converts domain names into network addresses of the webpages. The user or client sends the request for IP address to a local DNS server and gets response in the form of an address. The DNS protocol is an application layer protocol and is based on TCP i.e. the DNS packets are encapsulated into TCP segments.

There may be some DNS packets in the packet trace but if no packets are found, we can use terminal command called “nslookup”, this command returns the IP address corresponding to the domain name entered after it. We can also use another command “ifconfig” that displays some network related information on the terminal.

The figure given below lists all the relevant records only for DNS packets including requests from the client and replies from the server.



Activities Wireshark Mon Jun 15, 9:09:05 PM Wireshark - DNS - 194.10.10.30.pcapng

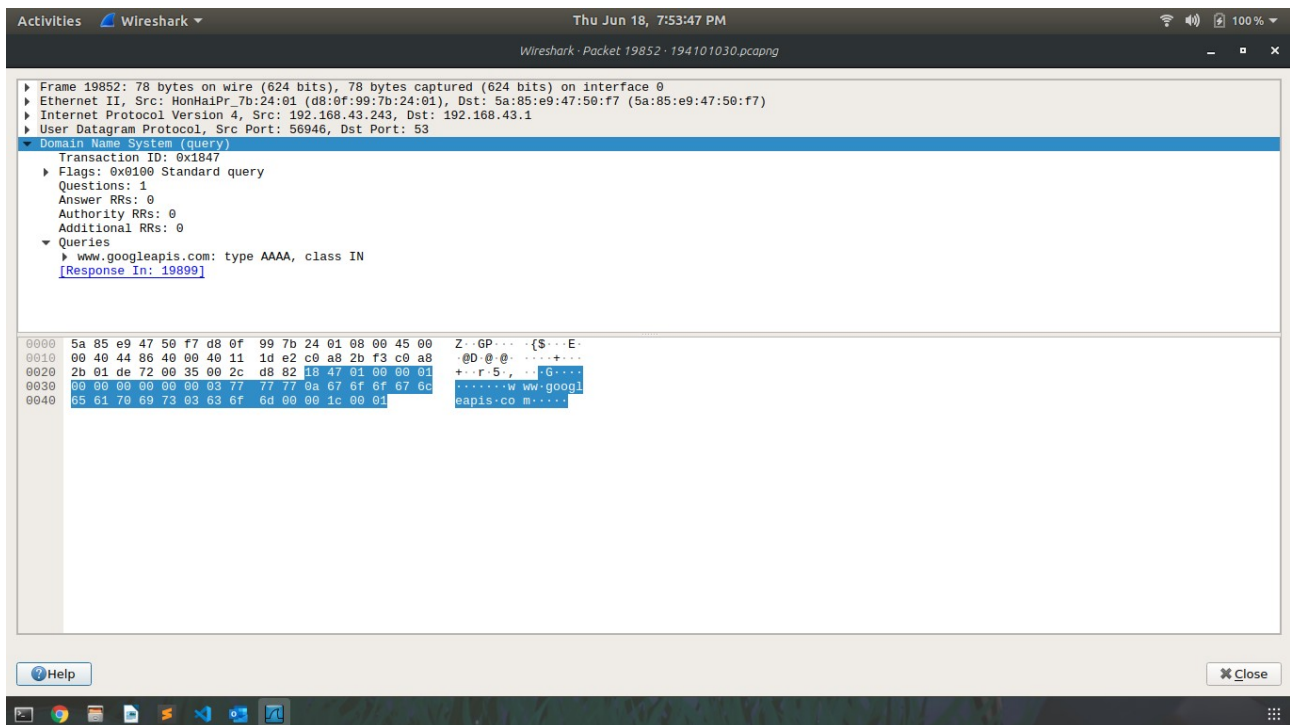
Topic / Item	Count	Average	Min val	Max val	Rate (ms)	Percent	Burst rate	Burst start
▼ Total Packets	241				0.0007	100%	0.1200	41.888
▼ rcode	241				0.0007	100.00%	0.1200	41.888
No such name	6				0.0000	2.49%	0.0400	322.173
No error	235				0.0007	97.51%	0.1200	41.888
▼ opcodes	241				0.0007	100.00%	0.1200	41.888
Standard query	241				0.0007	100.00%	0.1200	41.888
▼ Query/Response	241				0.0007	100.00%	0.1200	41.888
Response	120				0.0004	49.79%	0.0600	41.928
Query	121				0.0004	50.21%	0.0600	17.226
▼ Query Type	241				0.0007	100.00%	0.1200	41.888
AAAA (IPv6 Address)	117				0.0004	48.55%	0.0600	41.888
A (Host Address)	124				0.0004	51.45%	0.0700	322.167
▼ Class	241				0.0007	100.00%	0.1200	41.888
IN	241				0.0007	100.00%	0.1200	41.888
▼ Service Stats	0				0.0000	100%	-	-
request-response time (µs)	119	144142.55	2511	2697327	0.0004		0.0600	41.928
no. of unsolicited responses	0				0.0000		-	-
no. of retransmissions	1				0.0000		0.0100	246.029
▼ Response Stats	0				0.0000	100%	-	-
no. of questions	240	1.00	1	1	0.0007		0.1200	41.928
no. of authorities	240	0.05	0	1	0.0007		0.1200	41.928
no. of answers	240	1.54	0	15	0.0007		0.1200	41.928
no. of additionals	240	0.00	0	0	0.0007		0.1200	41.928
▼ Query Stats	0				0.0000	100%	-	-
Qname Len	121	18.93	5	36	0.0004		0.0600	17.226
▼ Label Stats	0				0.0000		-	-
4th Level or more	21				0.0001		0.0400	283.274
3rd Level	90				0.0003		0.0600	17.226
2nd Level	10				0.0000		0.0200	136.809
1st Level	0				0.0000		-	-
Payload size	241	55.93	23	305	0.0007	100%	0.1200	41.888

Display filter: Enter a display filter ... Apply Copy Save as... Close

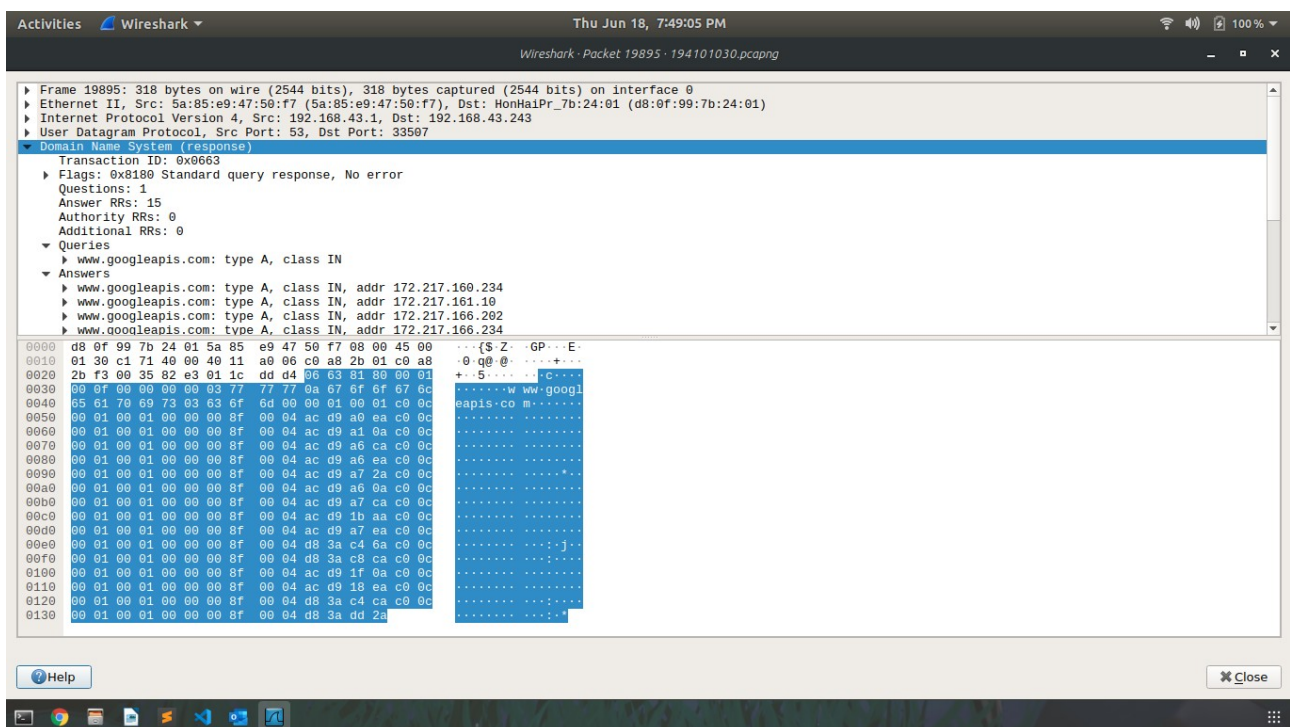
Packets: 35205 - Displayed: 35205 (100.0%) Profile: Default

There are total of 241 DNS packets sent and recieved between client and server, out of which 121 queries (120 queries and 1 retransmission) are transmitted by the client and 120 responses are received successfully from server.

Let's take two DNS packets (request and its response) and try to observe its contents and analyze it.



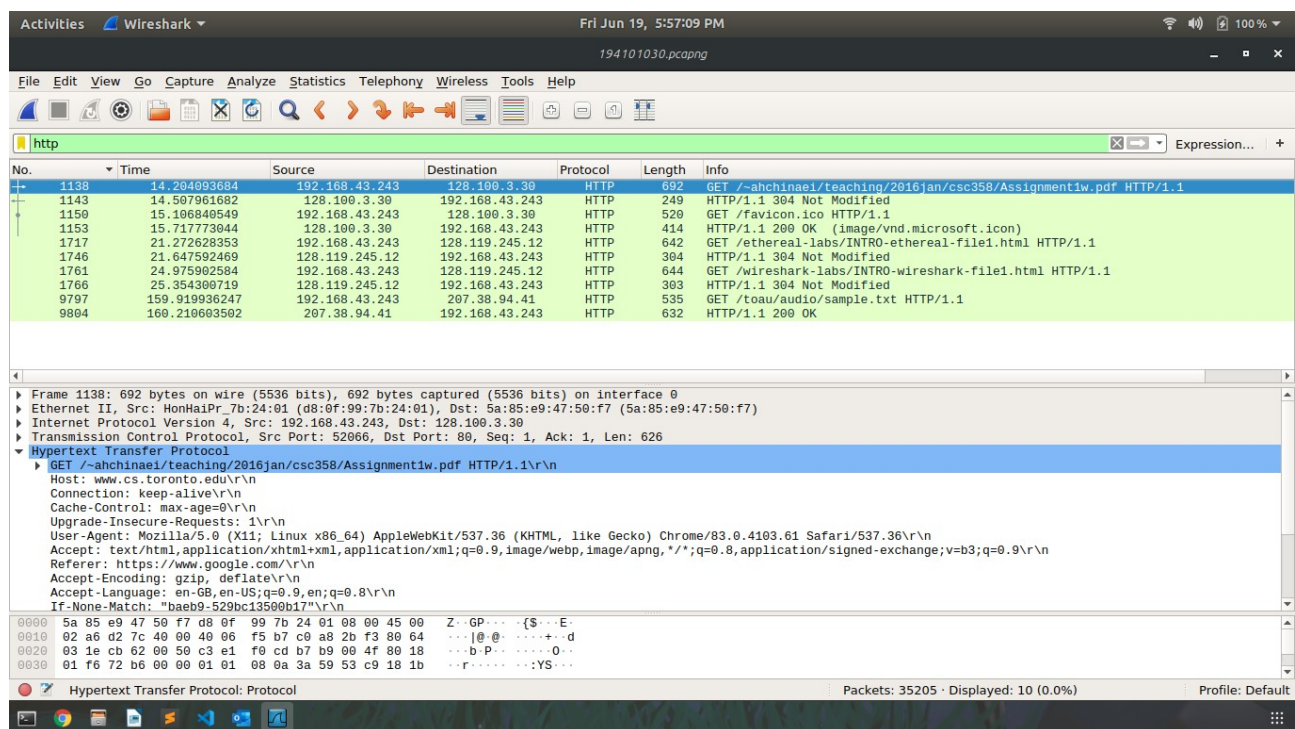
This packet is the standard DNS query packet which means the client has requested the server for IP address corresponding to webpage “[www.googleapis.com](http://www.googleapis.com)” and now the server will send one response packet.



It can be seen in the figure that it is a “standard query response” packet, which means that first the client sent the request for IP address of domain name ”[www.googleapis.com](http://www.googleapis.com)”. And, this packet is the response to that query and it contains total of 15 answers. One thing to notice here is that, DNS is based on both TCP/UDP but here both of these packets are encapsulated into UDP segment.

## HTTP

It is a stateless application layer protocol which is used to fetch the webpages from the webserver to the webclient. It is based on TCP and makes use of port number 80. There are 10 HTTP packets in the current packet capture of which 5 are GET packets and other 5 are responses to them.



Activities Wireshark Fri Jun 19, 5:57:09 PM 194101030.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
1138	14.204083684	192.168.43.243	128.100.3.30	HTTP	692	GET /~ahchinaei/teaching/2016jan/csc358/Assignment1w.pdf HTTP/1.1
1143	14.507901682	128.100.3.30	192.168.43.243	HTTP	249	HTTP/1.1 304 Not Modified
1150	15.106840549	192.168.43.243	128.100.3.30	HTTP	520	GET /favicon.ico HTTP/1.1
1153	15.717773044	128.100.3.30	192.168.43.243	HTTP	414	HTTP/1.1 200 OK (image/vnd.microsoft.icon)
1717	21.272628353	192.168.43.243	128.119.245.12	HTTP	642	GET /ethereal-labs/INTRO-ethereal-file1.html HTTP/1.1
1746	21.647592469	128.119.245.12	192.168.43.243	HTTP	304	HTTP/1.1 304 Not Modified
1761	24.975902584	192.168.43.243	128.119.245.12	HTTP	644	GET /wireshark-labs/INTRO-wireshark-file1.html HTTP/1.1
1766	25.354300719	128.119.245.12	192.168.43.243	HTTP	303	HTTP/1.1 304 Not Modified
9797	159.919936247	192.168.43.243	207.38.94.41	HTTP	535	GET /toau/audio/sample.txt HTTP/1.1
9804	160.210603502	207.38.94.41	192.168.43.243	HTTP	632	HTTP/1.1 200 OK

Frame 1138: 692 bytes on wire (5536 bits), 692 bytes captured (5536 bits) on interface 0

Ethernet II, Src: HonkaiPr.7b:24:91 (d8:0f:99:7b:24:91), Dst: 5a:85:e9:47:50:f7 (5a:85:e9:47:50:f7)

Internet Protocol Version 4, Src: 192.168.43.243, Dst: 128.100.3.30

Transmission Control Protocol, Src Port: 52066, Dst Port: 80, Seq: 1, Ack: 1, Len: 626

Hypertext Transfer Protocol

GET /~ahchinaei/teaching/2016jan/csc358/Assignment1w.pdf HTTP/1.1\r\n

Host: www.cs.toronto.edu\r\n

Connection: keep-alive\r\n

Cache-Control: max-age=0\r\n

Upgrade-Insecure-Requests: 1\r\n

User-Agent: Mozilla/5.0 (X11; Linux x86\_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36\r\n

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n

Referer: https://www.google.com\r\n

Accept-Encoding: gzip, deflate\r\n

Accept-Language: en-GB,en-US;q=0.9,en;q=0.8\r\n

If-None-Match: "baeb9-529bc13500b17"\r\n

0000 5a 85 e9 47 50 f7 d8 0f 99 7b 24 01 08 00 45 00 Z GP...-(\$...E

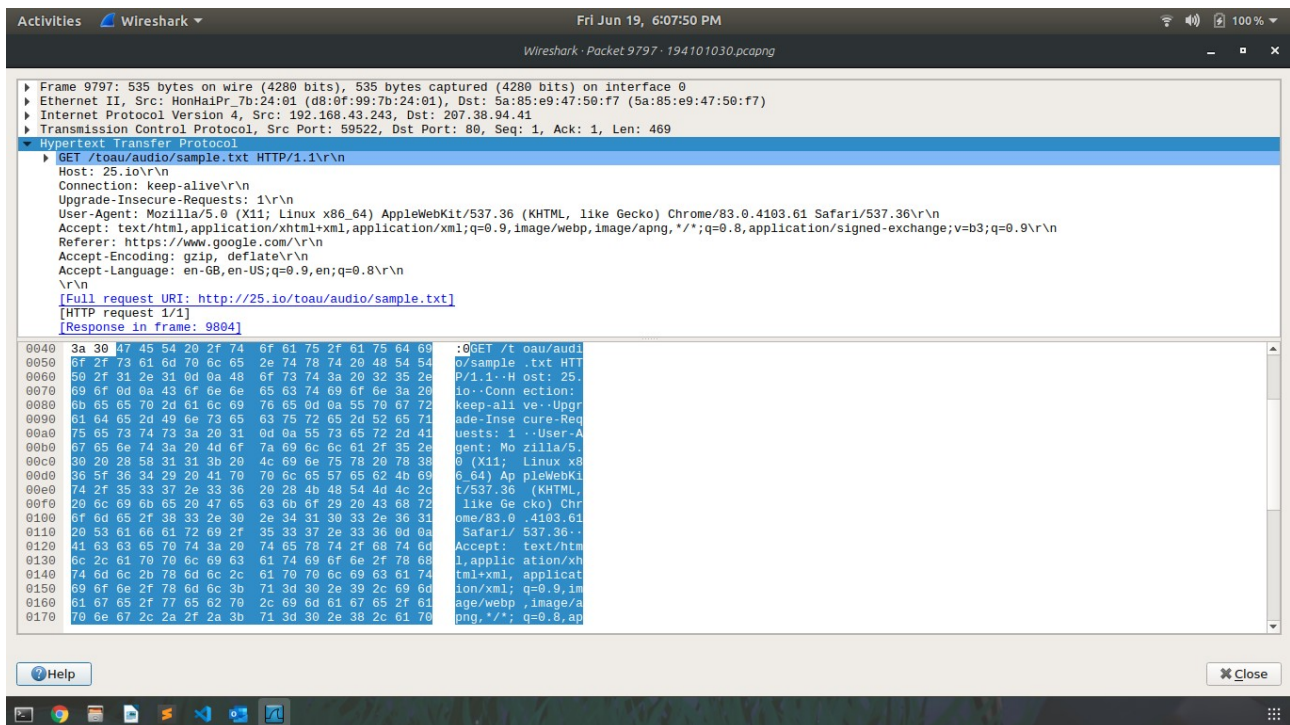
0010 02 a6 d2 7c 40 00 40 06 f5 b7 c0 a8 2b f3 80 64 ...|@@:....+..d

0020 03 1e cb 62 00 50 c3 e1 f9 cd b7 b9 00 4f 80 18 ...b.P...0..

0030 01 f6 72 b6 00 00 01 01 08 0a 3a 59 53 c9 18 1b ...r.....:YS...

Hypertext Transfer Protocol: Protocol Packets: 35205 · Displayed: 10 (0.0%) Profile: Default

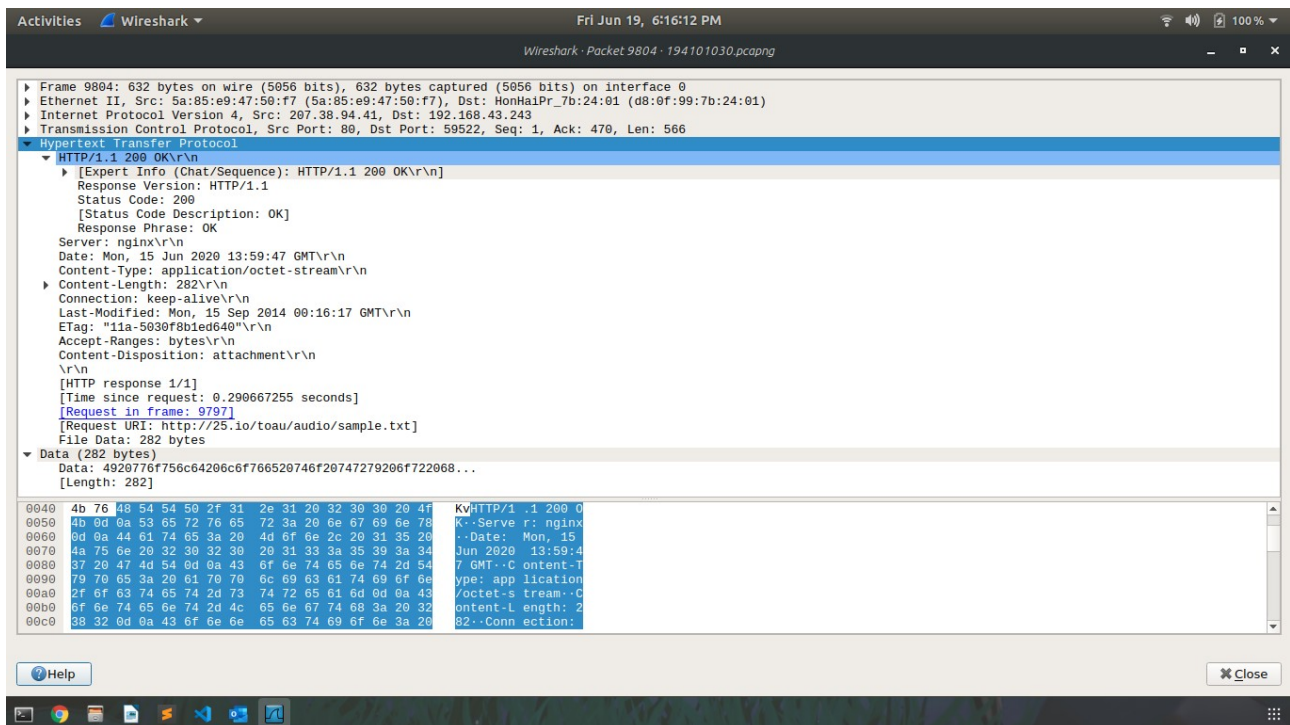
Now, let's take one HTTP GET packet. It is sent from 192.168.43.243 to 207.38.94.41.



This packet is sent by the web client to the server while downloading a sample text file from the internet using web browser. There are some details like connection type, user agent, host, languages accepted etc. which can be seen in the figure.

HTTP query contains “GET /toau/audio/sample.txt HTTP/1.1”, where GET method signifies that it’s a query message, “/toau/audio/sample.txt” is the Request URI, and HTTP/1.1 is the version of HTTP being used.

Now, let’s take response packet of this packet, which is sent by the web server to the client. It is sent from 207.38.94.41 to 192.168.43.243.



This HTTP packet is of response type and it contains some data (downloaded text file) of 282 bytes in total.

HTTP response contains “HTTP/1.1 200” which represents the HTTP version 1.1 and “200” is the status code. The response phrase for 200 is “OK”, that means the response is successfully transferred.

**NOTE :** The screenshots used in the report are also present in the folder along with this report.