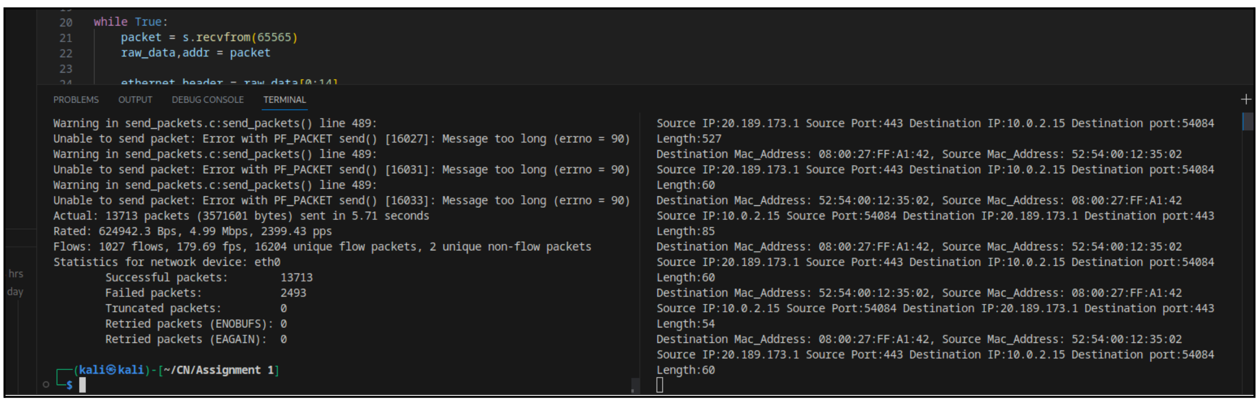
CS:433 Computer Networks

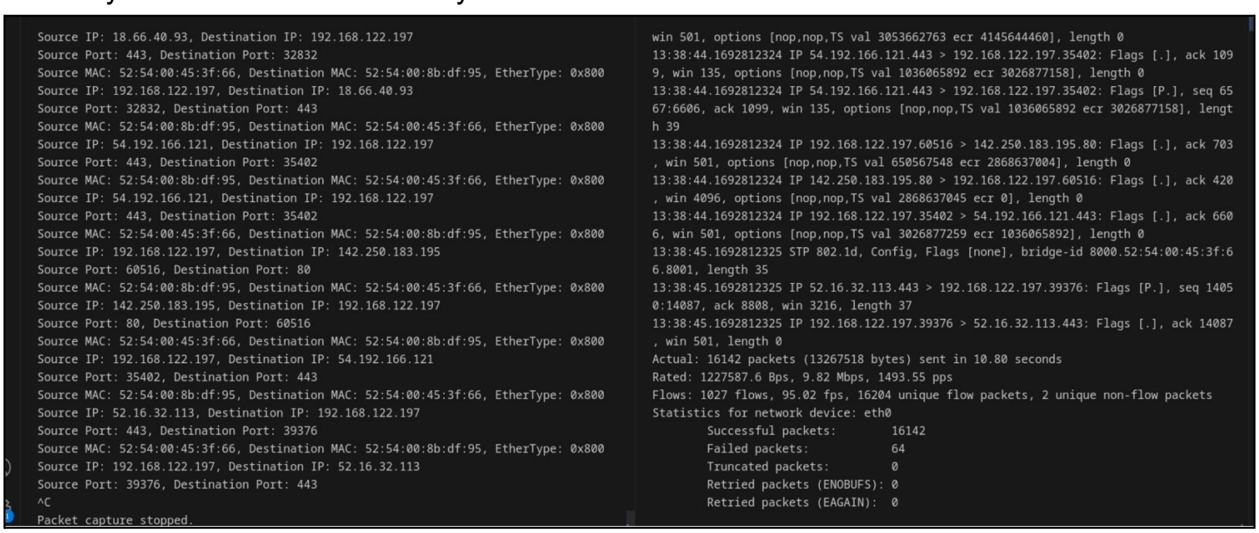
Assignment 1

Output & Observations

Q1. The raw socket is created and bound to the eth0 interface. The while loop parses the Ethernet and IP headers while capturing packets. The source IP, source port, destination IP, destination port, and protocol are extracted from an IPv4 and TCP packet, and a unique flow key is formed. If the flow is distinct, its details are written to a CSV file and included in the dictionary.



In the above image, you can see that many packets are failing as the default "**mtu**" is set to 1500, so it drops the packet with a size greater than 1500 bytes. We then increased the MTU to 16055 bytes as it was the limit of my device.

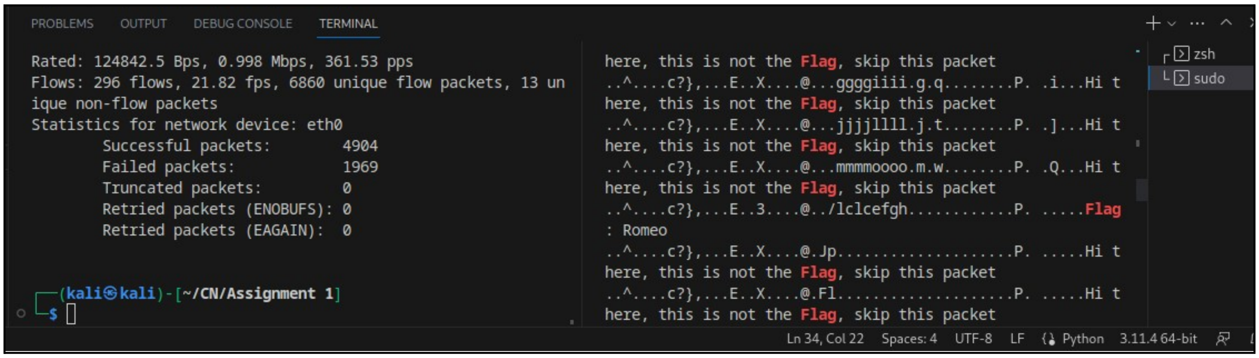


For part b, the “PartI_reversedns.csv” contains all the hostnames and IP addresses in the “0.pcap” file. From the CSV file, five hostnames and their IP addresses are provided in the table.

S.No	Hostname	IP Address
1	bom07s30-in-f14.1e100.net	142.250.183.14
2	a23-37-240-101.deploy.static.akamaitechnologies.com	23.37.240.101
3	cloudproxy10013.sucuri.net	192.124.249.13
4	ec2-54-246-192-162.eu-west-1.compute.amazonaws.com	54.246.192.162
5	sa.outbrain.com	66.225.223.95

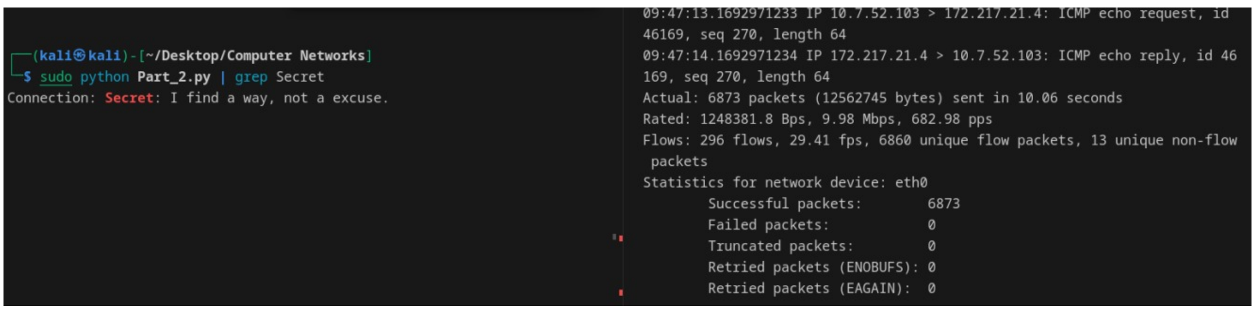
Q2. The code provided by us is sufficient to get all the payloads in the form of a CSV file, and then you can easily search for each of the questions with their hints and references in that CSV file.

1. In this question, we filtered all the payload ASCII values of all the TCP packets and then screened the keyword “Flag” among them using the “grep” command.



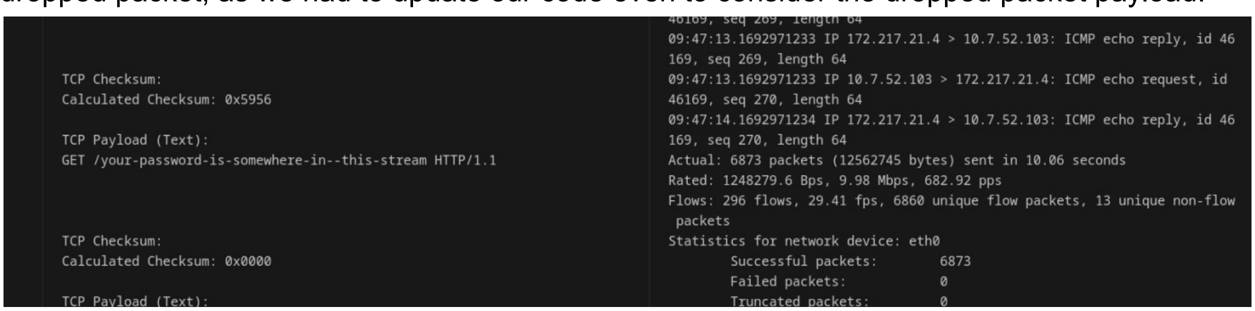
Ans: Flag: **Romeo**

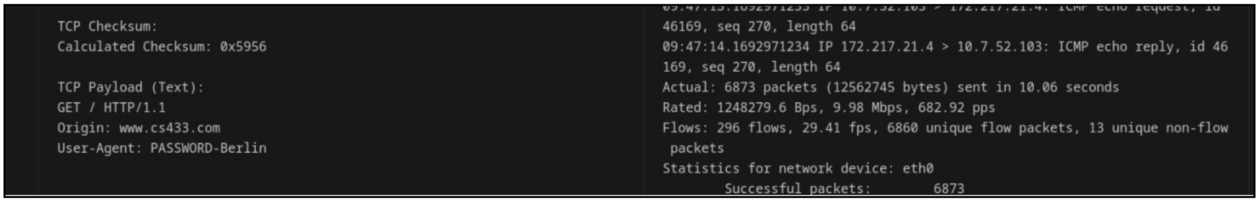
2. In this question, we filtered all the payload ASCII values of all the TCP packets and then screened the keyword “secret” among them using the “grep” command.



Ans: Secret: **I find a way, not a excuse.**

3. In this question, we printed all the payload ASCII values of the TCP packets whose checksum was equal to “0xf436”; we got the message that your password is somewhere in this stream, meaning the password packet must have the same IP and port then we searched for the same IP in our CSV file. We got the packet with the payload text as the password: “Berlin”. It was a dropped packet, as we had to update our code even to consider the dropped packet payload.

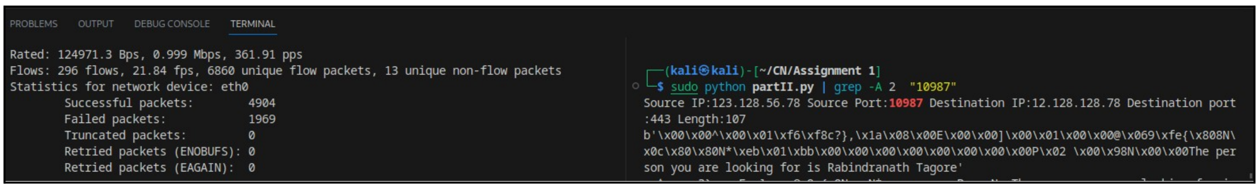




Ans: Instructions:

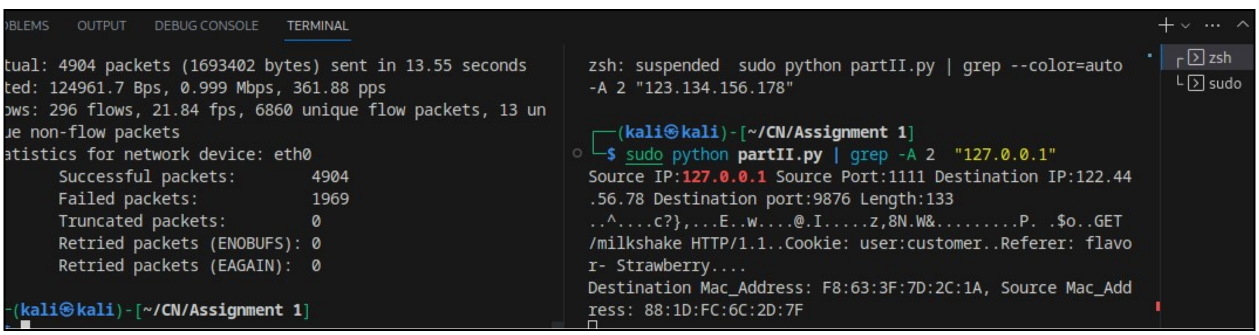
1. **your -password-is-somewhere-in--this-stream HTTP/1.1**
2. **User-Agent: PASSWORD-Berlin**

4. In this question, we filtered all the payload ASCII values of all the TCP packets and then screened the IP “123.134.156.178” among them using the “grep” command. Then further, we added the value of both source and destination ports and used the grep command again to reach our resultant packet.



Ans: **The person you are looking for is Rabindranath Tagore**

5. In this question, we filtered all the payload ASCII values of all the TCP packets and then filtered the keyword “milkshake” among them using the “grep” command.



Ans: Flavor-**Strawberry**

Q3. In this part, we simply ran the same code present in partIII but only for 30-second intervals. After which, a prompt is created which takes in the port number and, in return, provides the PID. In case there is no activity on the port, the prompt returns “No process found using port __”

```
Destination Mac_Address: 08:00:27:FF:A1:42, Source Mac_Address: 52:54:00:12:35:02
Source IP:54.192.153.239 Source Port:443 Destination IP:10.0.2.15 Destination port:54044 Length:1917
Destination Mac_Address: 52:54:00:12:35:02, Source Mac_Address: 08:00:27:FF:A1:42
Source IP:10.0.2.15 Source Port:54044 Destination IP:54.192.153.239 Destination port:443 Length:54
Destination Mac_Address: 52:54:00:12:35:02, Source Mac_Address: 08:00:27:FF:A1:42
Source IP:10.0.2.15 Source Port:49085 Destination IP:23.200.154.78 Destination port:443 Length:75
Destination Mac_Address: 08:00:27:FF:A1:42, Source Mac_Address: 52:54:00:12:35:02
Source IP:23.200.154.78 Source Port:443 Destination IP:10.0.2.15 Destination port:49085 Length:70
Destination Mac_Address: 52:54:00:12:35:02, Source Mac_Address: 08:00:27:FF:A1:42
Source IP:10.0.2.15 Source Port:54044 Destination IP:54.192.153.239 Destination port:443 Length:85
Destination Mac_Address: 08:00:27:FF:A1:42, Source Mac_Address: 52:54:00:12:35:02
Source IP:54.192.153.239 Source Port:443 Destination IP:10.0.2.15 Destination port:54044 Length:60
Destination Mac_Address: 52:54:00:12:35:02, Source Mac_Address: 08:00:27:FF:A1:42
Source IP:10.0.2.15 Source Port:57510 Destination IP:142.250.192.110 Destination port:443 Length:93


Finished sniffing packets for 30 seconds.

Enter the port number: 443
Process ID for port 443: 29769
Enter the port number: 57510
No process found using port 57510.
Enter the port number: 54044
Process ID for port 54044: 29769
Enter the port number:
```

Q.4>

Q.1> Part-a>

SSDP allows devices such as printers, modems, and surveillance cameras to be discovered on a network quickly and easily. It does this by broadcasting a message to the network, which other devices can respond to.
The RFC for SSDP is RFC 2660. It is titled "Simple Service Discovery Protocol."

22…	49.2085…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
22…	49.2089…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
22…	50.2207…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
22…	50.2215…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
23…	51.2239…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
23…	51.2239…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
25…	52.2338…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
25…	52.2338…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
13…	110.011…	10.7.0.238	239.255.255.250	SSDP	167	M-SEARCH * HTTP/1.1
17…	150.679…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
17…	150.698…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
17…	151.690…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
17…	151.710…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
17…	152.695…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
17…	152.710…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1

ICMP stands for Internet Control Message Protocol. It is a network protocol IP hosts and routers use to send error messages and status information.
The RFC for ICMP is RFC 792. It is titled "Internet Control Message Protocol."

12…	icmpv6	7.0.238	142.250.67.238	ICMP	74	Echo (ping) request id=0x0001, seq=4897/8467, ttl=128 (reply in 12346)
12…	87.6060…	142.250.67.238	10.7.0.238	ICMP	74	Echo (ping) reply id=0x0001, seq=4897/8467, ttl=115 (request in 12345)
12…	88.5952…	10.7.0.238	142.250.67.238	ICMP	74	Echo (ping) request id=0x0001, seq=4898/8723, ttl=128 (reply in 12360)
12…	88.6117…	142.250.67.238	10.7.0.238	ICMP	74	Echo (ping) reply id=0x0001, seq=4898/8723, ttl=115 (request in 12358)
13…	89.6082…	10.7.0.238	142.250.67.238	ICMP	74	Echo (ping) request id=0x0001, seq=4899/8979, ttl=128 (reply in 13143)
13…	89.6231…	142.250.67.238	10.7.0.238	ICMP	74	Echo (ping) reply id=0x0001, seq=4899/8979, ttl=115 (request in 13141)
13…	90.6227…	10.7.0.238	142.250.67.238	ICMP	74	Echo (ping) request id=0x0001, seq=4900/9235, ttl=128 (reply in 13161)
13…	90.6359…	142.250.67.238	10.7.0.238	ICMP	74	Echo (ping) reply id=0x0001, seq=4900/9235, ttl=115 (request in 13160)

The Address Resolution Protocol (**ARP**) is a network protocol that maps IP addresses to MAC addresses. It is used in local area networks (LANs) to determine the physical address of a device based on its IP address.
RFC 826

316	10.5902…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
319	12.6354…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
52…	63.2242…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
66…	65.9866…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
11…	82.7807…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
14…	116.062…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 10.7.0.1 (Request)
14…	116.164…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 10.7.0.1 (Request)
14…	116.471…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 10.7.0.1 (Request)
15…	127.837…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
16…	128.454…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
16…	128.861…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
16…	129.373…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 0.0.0.0 (Request)
16…	129.783…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 10.7.0.1 (Request)
16…	129.885…	Cisco_bb:7c:c0	Broadcast	ARP	60	Gratuitous ARP for 10.7.0.1 (Request)

NBNS is a broadcast protocol, which means that it sends messages to all devices on the network.
RFC 1001

17…	151.174…	10.7.0.238	224.0.0.252	LLMNR	64	Standard query 0x434e A wpad
17…	151.580…	10.7.0.238	10.7.63.255	NBNS	92	Name query NB WPAD<00>
17…	151.581…	10.7.0.238	10.7.63.255	NBNS	92	Name query NB WPAD<00>
17…	151.594…	fe80::5f93:9c38:d483:c29d	ff02::1:3	LLMNR	84	Standard query 0xcea8 A wpad
17…	151.594…	fe80::5f93:9c38:d483:c29d	ff02::1:3	LLMNR	84	Standard query 0x434e A wpad
17…	151.594…	10.7.0.238	224.0.0.252	LLMNR	64	Standard query 0x434e A wpad
17…	151.594…	10.7.0.238	224.0.0.252	LLMNR	64	Standard query 0xcea8 A wpad
17…	151.690…	10.7.0.238	239.255.255.250	SSDP	217	M-SEARCH * HTTP/1.1
17…	151.710…	10.7.0.238	239.255.255.250	SSDP	212	M-SEARCH * HTTP/1.1
17…	152.171…	10.7.0.238	224.0.0.251	MDNS	70	Standard query 0x0000 A wpad.local, "QM" question
17…	152.172…	10.7.0.238	224.0.0.251	MDNS	70	Standard query 0x0000 A wpad.local, "QM" question
17…	152.172…	fe80::5f93:9c38:d483:c29d	ff02::fb	MDNS	90	Standard query 0x0000 A wpad.local, "QM" question
17…	152.173…	fe80::5f93:9c38:d483:c29d	ff02::fb	MDNS	90	Standard query 0x0000 A wpad.local, "QM" question
17…	152.337…	10.7.0.238	10.7.63.255	NBNS	92	Name query NB WPAD<00>
17…	152.337…	10.7.0.238	10.7.63.255	NBNS	92	Name query NB WPAD<00>

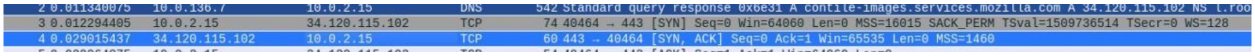
QUIC stands for Quick UDP Internet Connections. It is a new transport layer protocol designed to improve the performance of web browsing and other internet applications.
RFC 9000

98…	77.6220…	10.7.0.238	35.186.224.25	QUIC	1292	Initial, DCID=200a86f5887bdc98, PKN: 1, CRYPTO, PING, PING, CRYPTO, CRY
98…	77.6655…	35.186.224.25	10.7.0.238	QUIC	1292	Initial, SCID=e00a86f5887bdc98, PKN: 1, ACK, PADDING
98…	77.7018…	35.186.224.25	10.7.0.238	QUIC	1292	Protected Payload (KP0)
98…	77.7085…	10.7.0.238	35.186.224.25	QUIC	1292	Handshake, DCID=e00a86f5887bdc98
98…	77.7091…	10.7.0.238	35.186.224.25	QUIC	200	Protected Payload (KP0), DCID=e00a86f5887bdc98
98…	77.7100…	10.7.0.238	35.186.224.25	QUIC	1288	Protected Payload (KP0), DCID=e00a86f5887bdc98
98…	77.7101…	10.7.0.238	35.186.224.25	QUIC	706	Protected Payload (KP0), DCID=e00a86f5887bdc98
98…	77.7247…	35.186.224.25	10.7.0.238	QUIC	1292	Protected Payload (KP0)
98…	77.7247…	35.186.224.25	10.7.0.238	QUIC	162	Protected Payload (KP0)
98…	77.7247…	35.186.224.25	10.7.0.238	QUIC	69	Protected Payload (KP0)
98…	77.7250…	10.7.0.238	35.186.224.25	QUIC	74	Protected Payload (KP0), DCID=e00a86f5887bdc98
99…	77.7498…	35.186.224.25	10.7.0.238	QUIC	66	Protected Payload (KP0)
99…	77.7659…	10.7.0.238	35.186.224.25	QUIC	74	Protected Payload (KP0), DCID=e00a86f5887bdc98
99…	77.9299…	35.186.224.25	10.7.0.238	QUIC	309	Protected Payload (KP0)

Part-b>

Connection = TCP
Time of sending packet = 0.012294405

Time of response = 0.029015437
RTT = 0.029015437 - 0.012294405
=> 0.016721032 sec = 16.721032 ms



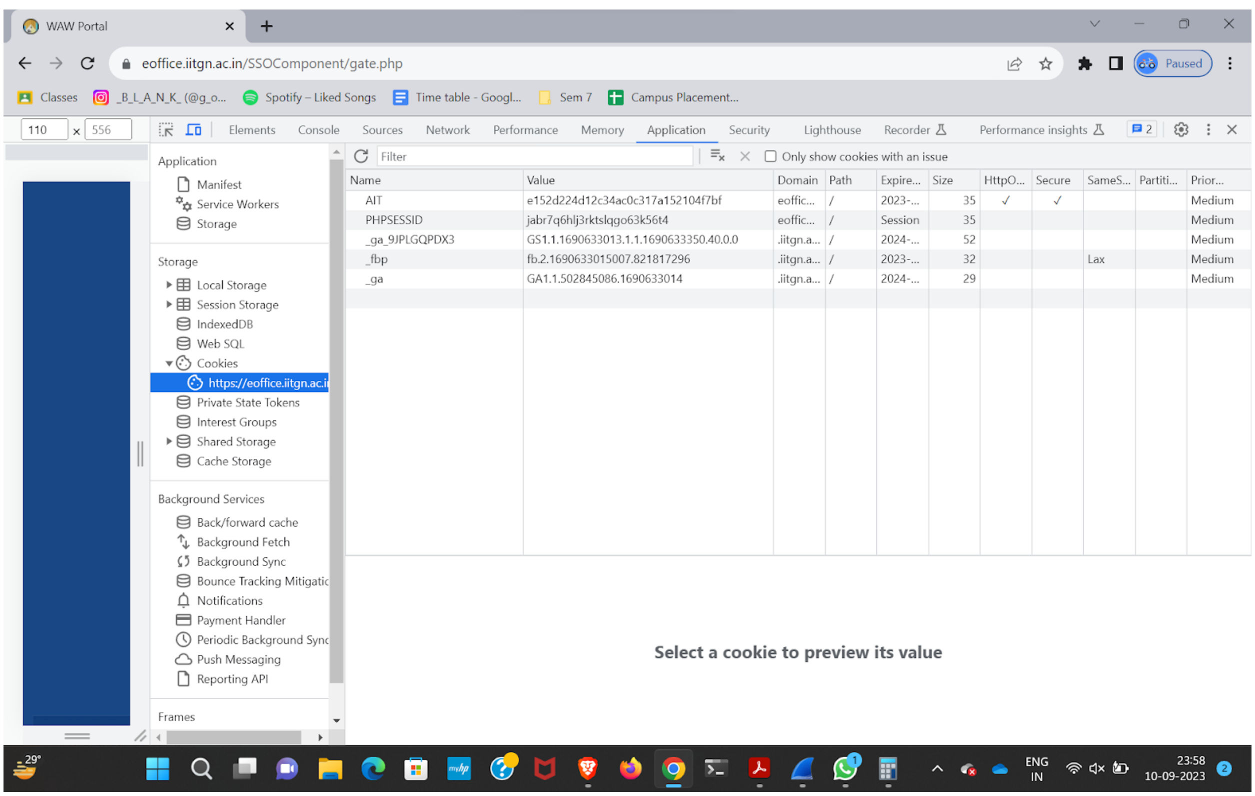
Q.2>

GitHub: Hypertext Transfer Protocol (HTTP) version 2 (HTTP/2)
Netflix: Hypertext Transfer Protocol (HTTP) version 1.1
Google: Hypertext Transfer Protocol (HTTP) version 2 (HTTP/2)

Difference between HTTP/2 and HTTP/1.1

- HTTP/2 uses a binary framing format for messages, while HTTP/1.1 uses a text-based framing format. This makes HTTP/2 more efficient and reliable.
- HTTP/2 supports multiplexing, which allows multiple requests to be sent over the same connection.
- HTTP/2 uses encryption by default, while HTTP/1.1 does not. This makes HTTP/2 more secure.

Q.3>



- **_ga**: This cookie is used by Google Analytics to track your visits to the website. It is a persistent cookie, meaning it expires after two years.
- **PHPSESSID**: The PHPSESSID cookie is a first-party cookie, which means it is set by the website you visit. It is a session cookie that expires when you close your browser.
- **_fbp**: It tracks users across different websites and serves them with targeted advertising. The _fbp cookie is persistent, meaning it expires after three months.

References:-

- https://www.uv.mx/personal/angelperez/files/2018/10/sniffers_texto.pdf
- https://github.com/jshreyans/sniffer