Packet Sniffer Analysis Report

Sneha Gautam

22110255

Tools Used

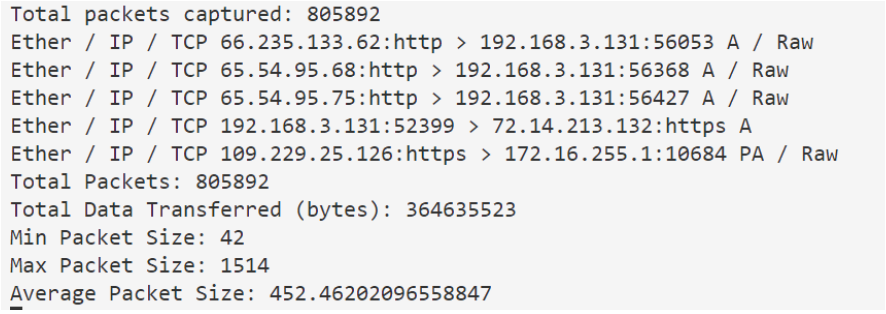
- **tcpreplay**: To replay captured packets.
- **Wireshark**: For detailed network analysis.
- **Matplotlib/Pandas**: For data visualization.

3. Part 1: Metrics and Plots

1. **Data Transfer Analysis -** Find the total amount of data transferred (in bytes), the total number of packets transferred, and the minimum, maximum, and average packet sizes. Also, show the distribution of packet sizes (e.g., by plotting a histogram of packet sizes).

```
from scapy.all import rdpcap, wrpcap

# Load the PCAP file
pcap_file = r"C:\Users\Sneha Gautam\Downloads\0.pcap"  # Replace with your .pcap file
packets = rdpcap(pcap_file)

wrpcap('packets_part1.pcap', packets)

# Print packet summary
print(f"Total packets captured: {len(packets)}")
for packet in packets[:5]:  # Print first 5 packets
    print(packet.summary())
```

```
import matplotlib.pyplot as plt

packet_sizes = [len(packet) for packet in packets]

# Metrics
print("Total Packets:", len(packets))
print("Total Data Transferred (bytes):", sum(packet_sizes))
print("Min Packet Size:", min(packet_sizes))
print("Max Packet Size:", max(packet_sizes))
print("Average Packet Size:", sum(packet_sizes) / len(packet_sizes))

# Plot histogram of packet sizes
plt.hist(packet_sizes, bins=20, color='blue', edgecolor='black')
plt.title("Packet Size Distribution")
plt.xlabel("Packet Size (bytes)")
plt.ylabel("Frequency")
plt.show()
```



```
Total packets captured: 805892
Ether / IP / TCP 66.235.133.62:http > 192.168.3.131:56053 A / Raw
Ether / IP / TCP 65.54.95.68:http > 192.168.3.131:56368 A / Raw
Ether / IP / TCP 65.54.95.75:http > 192.168.3.131:56427 A / Raw
Ether / IP / TCP 192.168.3.131:52399 > 72.14.213.132:https A
Ether / IP / TCP 109.229.25.126:https > 172.16.255.1:10684 PA / Raw
Total Packets: 805892
Total Data Transferred (bytes): 364635523
Min Packet Size: 42
Max Packet Size: 1514
Average Packet Size: 452.46202096558847
```

2. **Source-Destination Pair Analysis -** Find unique source-destination pairs (source IP:port and destination IP:port) in the captured data.

```
unique_pairs = set()
for packet in packets:
    if packet.haslayer("IP"):
        src_ip = packet["IP"].src
        dst_ip = packet["IP"].dst
        src_port = packet["IP"].sport if packet.haslayer("TCP") or
packet.haslayer("UDP") else None
        dst_port = packet["IP"].dport if packet.haslayer("TCP") or
packet.haslayer("UDP") else None
```

```
        unique_pairs.add((src_ip, src_port, dst_ip, dst_port))

print("Unique Source-Destination Pairs:")
for pair in unique_pairs:
    print(pair)
```

3. **IP Flow Analysis -** Display a dictionary where the key is the IP address and the value is the total flows for that IP address as the source. Similarly display a dictionary where the key is the IP address and the value is the total flows for that IP address as the destination. Find out which source-destination (source IP:port and destination IP:port) have transferred the most data.

```
from collections import defaultdict

src_flows = defaultdict(int)
dst_flows = defaultdict(int)

for packet in packets:
    if packet.haslayer("IP"):
        src_ip = packet["IP"].src
        dst_ip = packet["IP"].dst
        src_flows[src_ip] += 1
        dst_flows[dst_ip] += 1

print("Flows per Source IP:", dict(src_flows))
print("Flows per Destination IP:", dict(dst_flows))


pair_data = defaultdict(int)

for packet in packets:
    if packet.haslayer("IP"):
        src_ip = packet["IP"].src
        dst_ip = packet["IP"].dst
        size = len(packet)
        pair_data[(src_ip, dst_ip)] += size

max_pair = max(pair_data, key=pair_data.get)
```

```
print(f"Source-Destination Pair with Most Data: {max_pair} - {pair_data[max_pair]} bytes")
```

4. **Speed -** List the top speed in terms of `pps` and `mbps` that your program is able to capture the content without any loss of data when i) running both tcpreplay and your program on the same machine (VM), and ii) when running on different machines: Two student group should run the program on two different machines eg. tcpreplay on physical-machine of student1 and sniffer program physical-machine of student2. Single students should run between two VMs.

4. Part 2: Catch Me If You Can

(1) **IMS Server Connections -** How many unique connections were made to the IMS server ?

```
from scapy.all import rdpcap

packets = rdpcap('packets_part1.pcap')

ims_ip = "14.139.98.79" # Replace with IMS server IP "14.139.98.79" , ip.dst == IMS_IP or ip.src == IMS_IP, 10.0.137.79

ims_connections = set()

for packet in packets:
    if packet.haslayer("IP") and packet["IP"].dst == ims_ip:
        src_ip = packet["IP"].src
        ims_connections.add(src_ip)

print("Unique Connections to IMS Server:", len(ims_connections))
```

(2) **Course Registration Tracking -** I have registered for a course in IMS. What course did I register for?

```
for packet in packets:
    if packet.haslayer("Raw"):  # Look at application-layer payload
        if b"course" in bytes(packet["Raw"].load):
            print("Course Registration Packet:", packet["Raw"].load)
```

(3) **Total amount of data transferred over a port 4321 -** What is the total amount of data (in bytes) transferred over a port 4321 ?

```
port_data = sum(len(packet) for packet in packets if
packet.haslayer("UDP") and (packet["UDP"].sport == 4321 or
packet["UDP"].dport == 4321))
print(f"Total Data Transferred on Port 4321: {port_data} bytes")
```

(4) **SuperUsers -** There are many Superuser. Find how many SuperUsers are there?

```
superuser_count = 0

for packet in packets:
    if packet.haslayer("Raw") and b"superuser" in
bytes(packet["Raw"].load):
        superuser_count += 1

print("Number of SuperUsers:", superuser_count)
```

5. Part 3: Capturing the Packets

(1) **Wireshark Analysis and Protocols -** Run the Wireshark tool and capture the trace of the network packets on your host device. We expect you would be connected to the Internet and perform regular network activities.

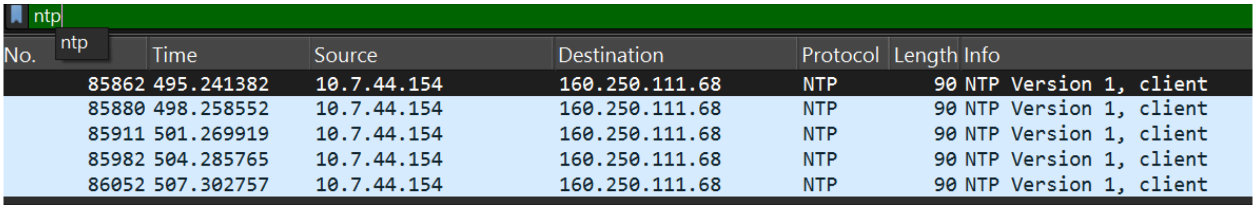
a. List at-least 5 different application layer protocols that we have not discussed so far in the classroom and describe in 1-2 sentences the operation/usage of protocol and its layer of operation and indicate the associated RFC number if any.

1. **NTP (Network Time Protocol)**

NTP is used to synchronize the clocks of computers over a network. It allows devices to obtain accurate time from a time server, ensuring that all devices in the network are synchronized to the same time source. It's commonly used in applications where accurate time is crucial, like logging, file systems, and communication protocols.

Application (Layer 7)

RFC: RFC 5905



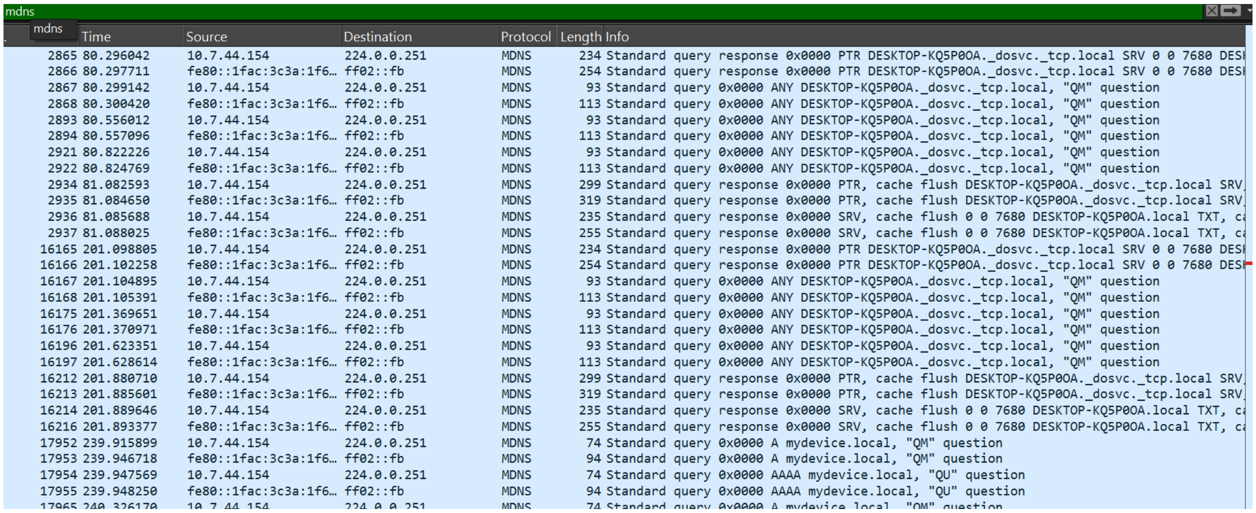
```
ntp
No.          Time        Source         Destination     Protocol  Length Info
       85862 495.241382  10.7.44.154    160.250.111.68  NTP           90 NTP Version 1, client
       85880 498.258552  10.7.44.154    160.250.111.68  NTP           90 NTP Version 1, client
       85911 501.269919  10.7.44.154    160.250.111.68  NTP           90 NTP Version 1, client
       85982 504.285765  10.7.44.154    160.250.111.68  NTP           90 NTP Version 1, client
       86052 507.302757  10.7.44.154    160.250.111.68  NTP           90 NTP Version 1, client
```

2. **MDNS (Multicast DNS)**

mDNS allows devices on a local network to resolve hostnames to IP addresses without requiring a DNS server. It uses multicast IP addressing to send queries for names and respond to those queries within the local network. It’s commonly used for service discovery on home networks, such as discovering printers, cameras, and other devices in environments like home automation systems.

Application (Layer 7)

RFC: RFC 6762



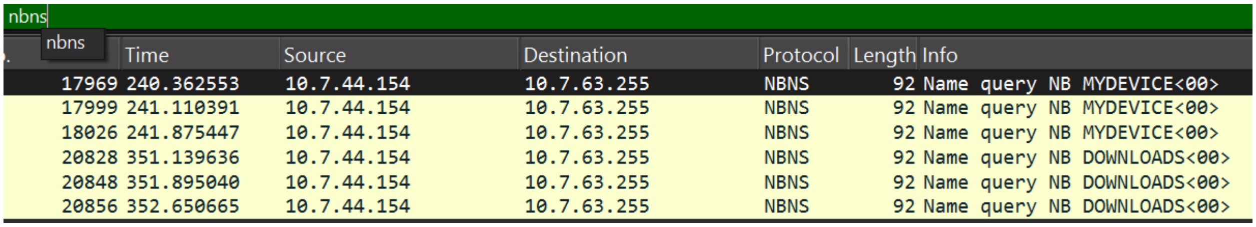
```
mdns
.          Time        Source                 Destination  Protocol Length Info
      2865 80.296042   10.7.44.154            224.0.0.251  MDNS      234 Standard query response 0x0000 PTR DESKTOP-KQ5P0OA._dosvc._tcp.local SRV 0 0 7680 DESK
      2866 80.297711   fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      254 Standard query response 0x0000 PTR DESKTOP-KQ5P0OA._dosvc._tcp.local SRV 0 0 7680 DESK
      2867 80.299142   10.7.44.154            224.0.0.251  MDNS       93 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
      2868 80.300420   fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      113 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
      2893 80.556012   10.7.44.154            224.0.0.251  MDNS       93 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
      2894 80.557096   fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      113 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
      2921 80.822226   10.7.44.154            224.0.0.251  MDNS       93 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
      2922 80.824769   fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      113 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
      2934 81.082593   10.7.44.154            224.0.0.251  MDNS      299 Standard query response 0x0000 PTR, cache flush DESKTOP-KQ5P0OA._dosvc._tcp.local SRV,
      2935 81.084650   fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      319 Standard query response 0x0000 PTR, cache flush DESKTOP-KQ5P0OA._dosvc._tcp.local SRV,
      2936 81.085688   10.7.44.154            224.0.0.251  MDNS      235 Standard query response 0x0000 SRV, cache flush 0 0 7680 DESKTOP-KQ5P0OA.local TXT, ca
      2937 81.088025   fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      255 Standard query response 0x0000 SRV, cache flush 0 0 7680 DESKTOP-KQ5P0OA.local TXT, ca
     16165 201.098805  10.7.44.154            224.0.0.251  MDNS      234 Standard query response 0x0000 PTR DESKTOP-KQ5P0OA._dosvc._tcp.local SRV 0 0 7680 DESK
     16166 201.102258  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      254 Standard query response 0x0000 PTR DESKTOP-KQ5P0OA._dosvc._tcp.local SRV 0 0 7680 DESK
     16167 201.104895  10.7.44.154            224.0.0.251  MDNS       93 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
     16168 201.105391  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      113 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
     16175 201.369651  10.7.44.154            224.0.0.251  MDNS       93 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
     16176 201.370971  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      113 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
     16196 201.623351  10.7.44.154            224.0.0.251  MDNS       93 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
     16197 201.628614  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      113 Standard query 0x0000 ANY DESKTOP-KQ5P0OA._dosvc._tcp.local, "QM" question
     16212 201.880710  10.7.44.154            224.0.0.251  MDNS      299 Standard query response 0x0000 PTR, cache flush DESKTOP-KQ5P0OA._dosvc._tcp.local SRV,
     16213 201.885601  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      319 Standard query response 0x0000 PTR, cache flush DESKTOP-KQ5P0OA._dosvc._tcp.local SRV,
     16214 201.889646  10.7.44.154            224.0.0.251  MDNS      235 Standard query response 0x0000 SRV, cache flush 0 0 7680 DESKTOP-KQ5P0OA.local TXT, ca
     16216 201.893377  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS      255 Standard query response 0x0000 SRV, cache flush 0 0 7680 DESKTOP-KQ5P0OA.local TXT, ca
     17952 239.915899  10.7.44.154            224.0.0.251  MDNS       74 Standard query 0x0000 A mydevice.local, "QM" question
     17953 239.946718  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS       94 Standard query 0x0000 A mydevice.local, "QM" question
     17954 239.947569  10.7.44.154            224.0.0.251  MDNS       74 Standard query 0x0000 AAAA mydevice.local, "QU" question
     17955 239.948250  fe80::1fac:3c3a:1f6…   ff02::fb     MDNS       94 Standard query 0x0000 AAAA mydevice.local, "QU" question
     17965 240.326170  10.7.44.154            224.0.0.251  MDNS       74 Standard query 0x0000 A mydevice.local, "QM" question
```

3. **NBNS (NetBIOS Name Service)**

NBNS is used for name resolution in local area networks (LANs), primarily in **Windows** environments. It allows computers to register and resolve **NetBIOS** names (hostnames) to **IP addresses**. NBNS is typically used to resolve names for network services such as file sharing and printer access in Windows networks. It functions similarly to DNS, but it is designed for local network use.

Application (Layer 7)

RFC: RFC 1001, RFC 1002



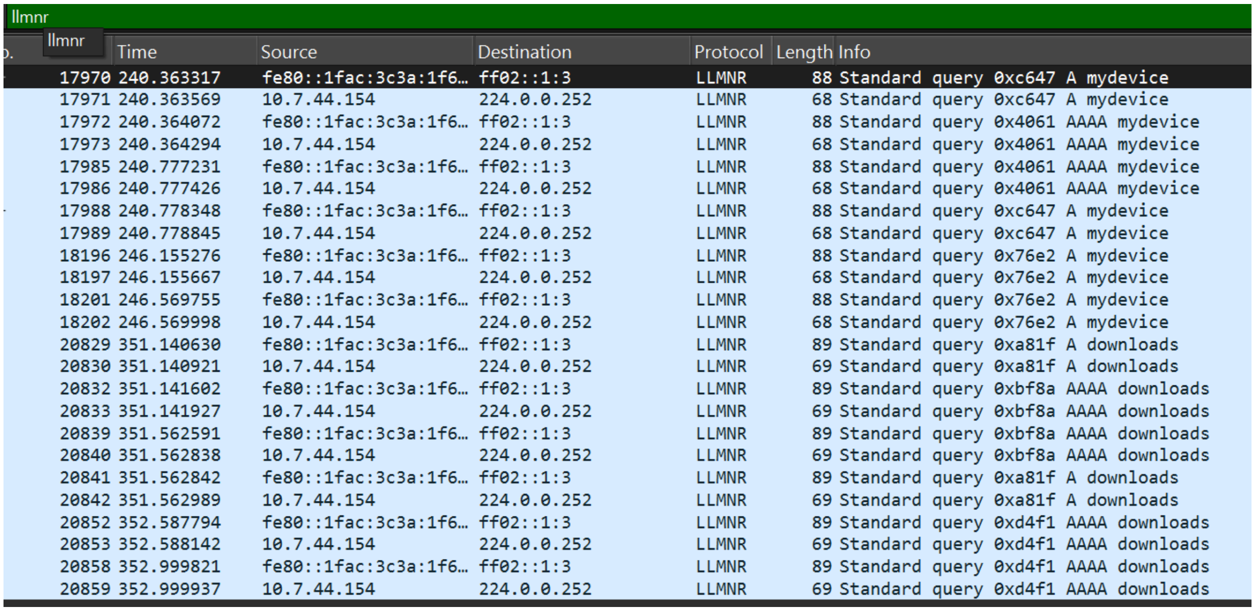
	Time	Source	Destination	Protocol	Length	Info
17969	240.362553	10.7.44.154	10.7.63.255	NBNS	92	Name query NB MYDEVICE<00>
17999	241.110391	10.7.44.154	10.7.63.255	NBNS	92	Name query NB MYDEVICE<00>
18026	241.875447	10.7.44.154	10.7.63.255	NBNS	92	Name query NB MYDEVICE<00>
20828	351.139636	10.7.44.154	10.7.63.255	NBNS	92	Name query NB DOWNLOADS<00>
20848	351.895040	10.7.44.154	10.7.63.255	NBNS	92	Name query NB DOWNLOADS<00>
20856	352.650665	10.7.44.154	10.7.63.255	NBNS	92	Name query NB DOWNLOADS<00>

4. LLMNR **(Link-Local Multicast Name Resolution)**

LLMNR allows devices on the same local network to resolve hostnames to IP addresses without the need for a DNS server. It is used primarily in IPv6 networks but can also operate in IPv4 networks. LLMNR is used when a DNS server is unavailable, enabling name resolution for services on the local network. It typically works over **UDP** and uses multicast to query other devices in the local network.

Application (Layer 7)

RFC: RFC 4795



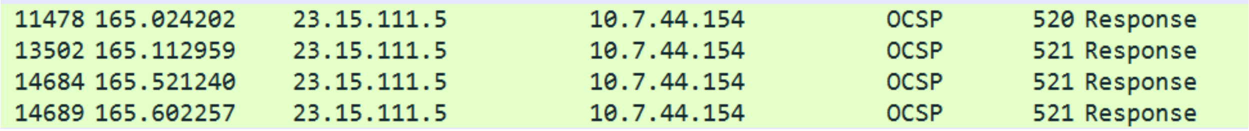
```
llmnr
        llmnr
o.           Time          Source                 Destination          Protocol Length Info
       17970 240.363317 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        88 Standard query 0xc647 A mydevice
       17971 240.363569 10.7.44.154          224.0.0.252          LLMNR        68 Standard query 0xc647 A mydevice
       17972 240.364072 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        88 Standard query 0x4061 AAAA mydevice
       17973 240.364294 10.7.44.154          224.0.0.252          LLMNR        68 Standard query 0x4061 AAAA mydevice
       17985 240.777231 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        88 Standard query 0x4061 AAAA mydevice
       17986 240.777426 10.7.44.154          224.0.0.252          LLMNR        68 Standard query 0x4061 AAAA mydevice
       17988 240.778348 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        88 Standard query 0xc647 A mydevice
       17989 240.778845 10.7.44.154          224.0.0.252          LLMNR        68 Standard query 0xc647 A mydevice
       18196 246.155276 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        88 Standard query 0x76e2 A mydevice
       18197 246.155667 10.7.44.154          224.0.0.252          LLMNR        68 Standard query 0x76e2 A mydevice
       18201 246.569755 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        88 Standard query 0x76e2 A mydevice
       18202 246.569998 10.7.44.154          224.0.0.252          LLMNR        68 Standard query 0x76e2 A mydevice
       20829 351.140630 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        89 Standard query 0xa81f A downloads
       20830 351.140921 10.7.44.154          224.0.0.252          LLMNR        69 Standard query 0xa81f A downloads
       20832 351.141602 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        89 Standard query 0xbf8a AAAA downloads
       20833 351.141927 10.7.44.154          224.0.0.252          LLMNR        69 Standard query 0xbf8a AAAA downloads
       20839 351.562591 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        89 Standard query 0xbf8a AAAA downloads
       20840 351.562838 10.7.44.154          224.0.0.252          LLMNR        69 Standard query 0xbf8a AAAA downloads
       20841 351.562842 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        89 Standard query 0xa81f A downloads
       20842 351.562989 10.7.44.154          224.0.0.252          LLMNR        69 Standard query 0xa81f A downloads
       20852 352.587794 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        89 Standard query 0xd4f1 AAAA downloads
       20853 352.588142 10.7.44.154          224.0.0.252          LLMNR        69 Standard query 0xd4f1 AAAA downloads
       20858 352.999821 fe80::1fac:3c3a:1f6… ff02::1:3            LLMNR        89 Standard query 0xd4f1 AAAA downloads
       20859 352.999937 10.7.44.154          224.0.0.252          LLMNR        69 Standard query 0xd4f1 AAAA downloads
```

5. **OCSP (Online Certificate Status Protocol)**

OCSP is used to check the revocation status of digital certificates in real time. It is an alternative to Certificate Revocation Lists (CRLs). When a client (such as a browser) connects to a server using SSL/TLS, it may query the OCSP responder to determine whether the server's certificate is still valid or if it has been revoked. OCSP enhances security by providing up-to-date certificate status information during secure communications, ensuring that the certificate is not compromised or revoked.

Application (Layer 7)

RFC: RFC 6960



```
11478 165.024202    23.15.111.5          10.7.44.154          OCSP       520 Response
13502 165.112959    23.15.111.5          10.7.44.154          OCSP       521 Response
14684 165.521240    23.15.111.5          10.7.44.154          OCSP       521 Response
14689 165.602257    23.15.111.5          10.7.44.154          OCSP       521 Response
```

(2) Website Request Line and Headers - Analyze the following details by visiting the following websites in your favourite browser.

i) canarabank.in

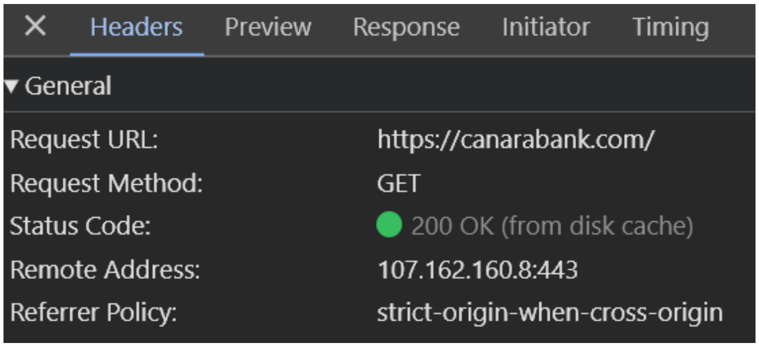
ii) github.com

iii) netflix.com

A. Identify `request line` with the version of the application layer protocol and the IP address. Also, identify whether the connection(s) is/are persistent or not.

1.Canarabank.com

Request line:GET / HTTP/1.1
IP address:107.162.160.8:443
Connection: close (non-persistent)

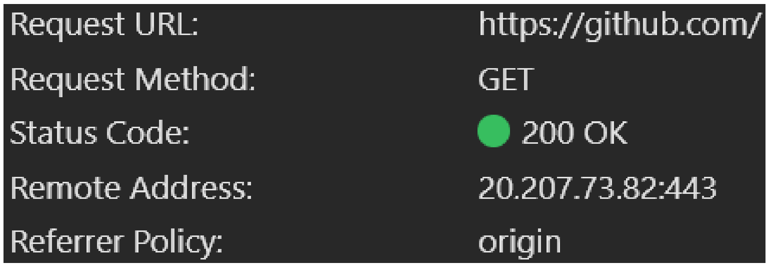


Request URL:	https://canarabank.com/
Request Method:	GET
Status Code:	● 200 OK (from disk cache)
Remote Address:	107.162.160.8:443
Referrer Policy:	strict-origin-when-cross-origin

2.Github.com

Protocol: HTTP/2
IP Address: 20.207.73.82:443

Connection: Persistent

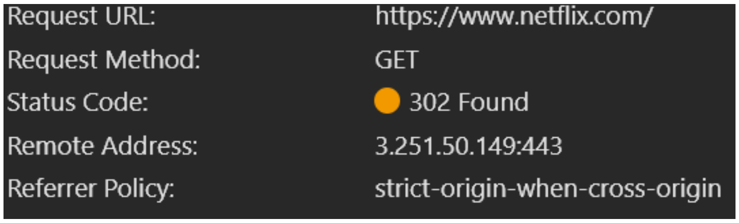


Request URL:	https://github.com/
Request Method:	GET
Status Code:	● 200 OK
Remote Address:	20.207.73.82:443
Referrer Policy:	origin

3.Netflix.com

Protocol: HTTP/2
IP Address: 3.251.50.149:443

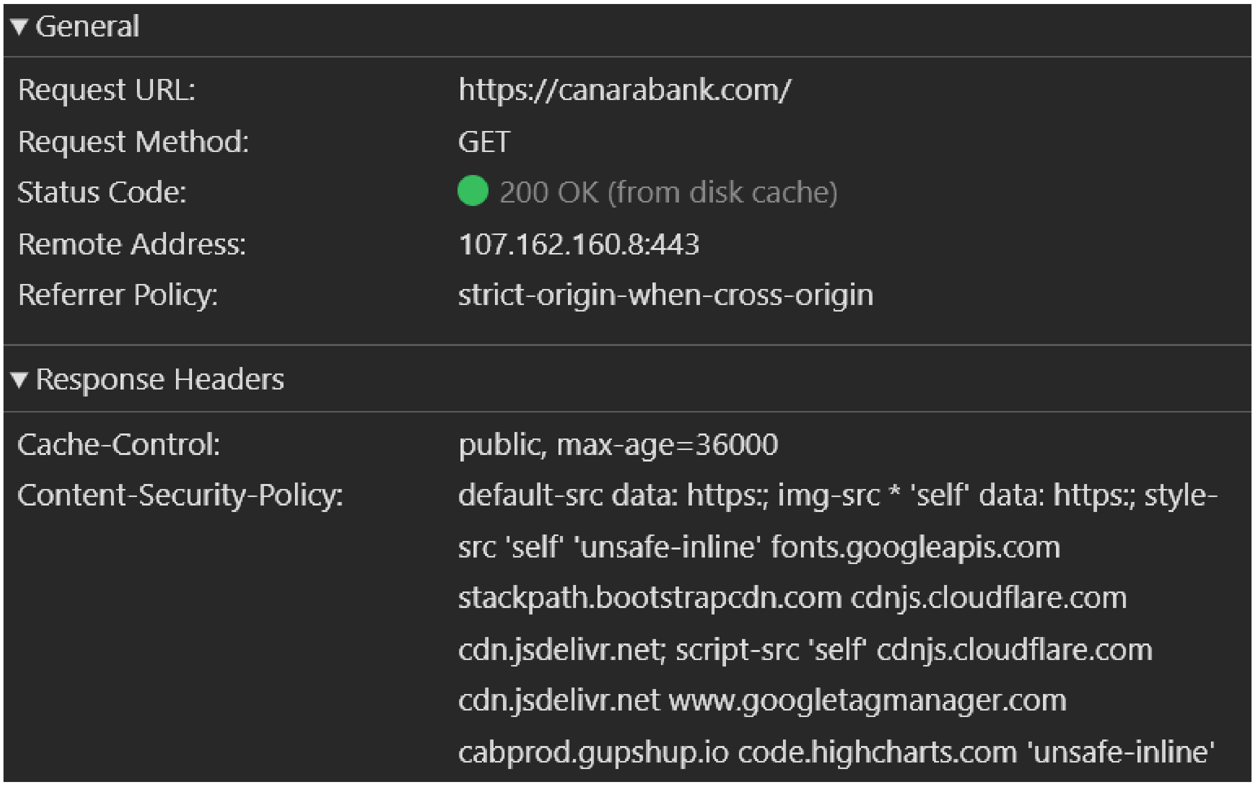
Connection : **persistent**



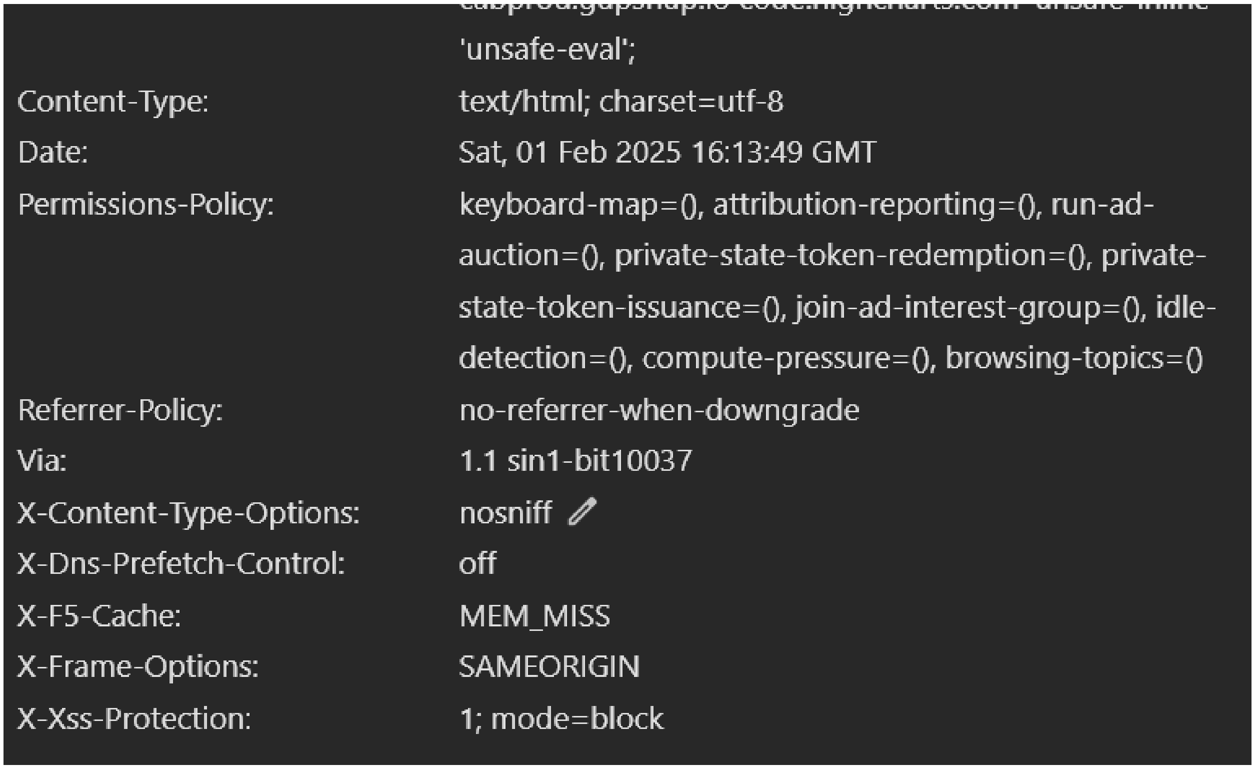
Request URL:	https://www.netflix.com/
Request Method:	GET
Status Code:	● 302 Found
Remote Address:	3.251.50.149:443
Referrer Policy:	strict-origin-when-cross-origin

B. **For any one of the websites, list any three header field names and corresponding values in the request and response message. Any three HTTP error codes obtained while loading one of the pages with a brief description.**

Response Header



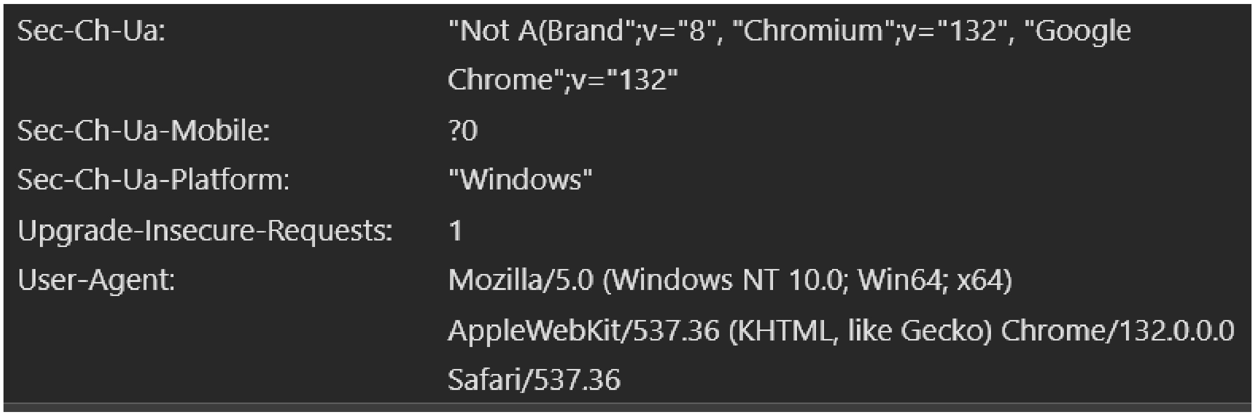
```
▼ General
Request URL:                 https://canarabank.com/
Request Method:              GET
Status Code:                 ● 200 OK (from disk cache)
Remote Address:              107.162.160.8:443
Referrer Policy:             strict-origin-when-cross-origin

▼ Response Headers
Cache-Control:               public, max-age=36000
Content-Security-Policy:     default-src data: https:; img-src * 'self' data: https:; style-
                             src 'self' 'unsafe-inline' fonts.googleapis.com
                             stackpath.bootstrapcdn.com cdnjs.cloudflare.com
                             cdn.jsdelivr.net; script-src 'self' cdnjs.cloudflare.com
                             cdn.jsdelivr.net www.googletagmanager.com
                             cabprod.gupshup.io code.highcharts.com 'unsafe-inline'
```



cabprod.gupshup.io code.highcharts.com 'unsafe-inline'

```
                             'unsafe-eval';
Content-Type:                text/html; charset=utf-8
Date:                        Sat, 01 Feb 2025 16:13:49 GMT
Permissions-Policy:          keyboard-map=(), attribution-reporting=(), run-ad-
                             auction=(), private-state-token-redemption=(), private-
                             state-token-issuance=(), join-ad-interest-group=(), idle-
                             detection=(), compute-pressure=(), browsing-topics=()
Referrer-Policy:             no-referrer-when-downgrade
Via:                         1.1 sin1-bit10037
X-Content-Type-Options:      nosniff
X-Dns-Prefetch-Control:      off
X-F5-Cache:                  MEM_MISS
X-Frame-Options:             SAMEORIGIN
X-Xss-Protection:            1; mode=block
```

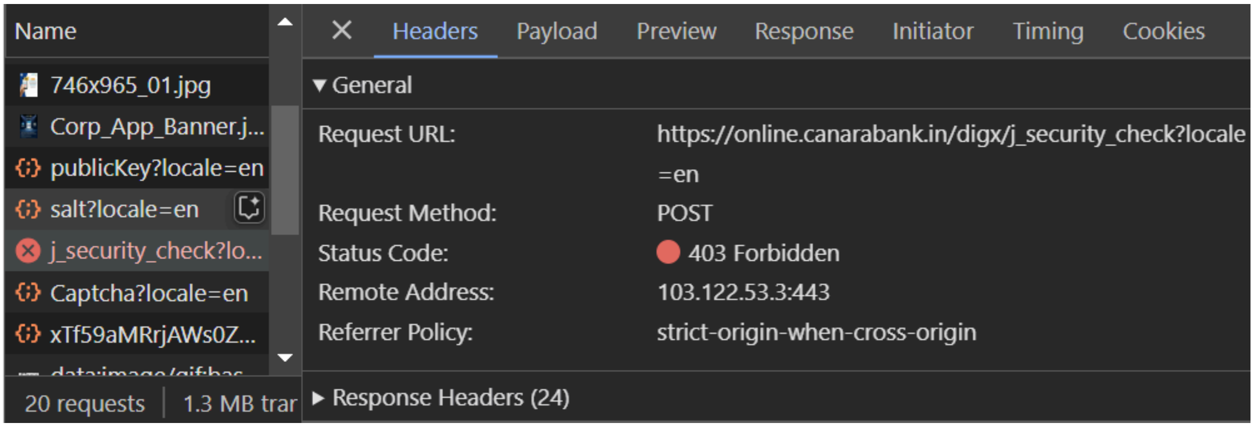
Request Header



```
Sec-Ch-Ua:                    "Not A(Brand";v="8", "Chromium";v="132", "Google
                              Chrome";v="132"
Sec-Ch-Ua-Mobile:             ?0
Sec-Ch-Ua-Platform:           "Windows"
Upgrade-Insecure-Requests:    1
User-Agent:                   Mozilla/5.0 (Windows NT 10.0; Win64; x64)
                              AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0
                              Safari/537.36
```

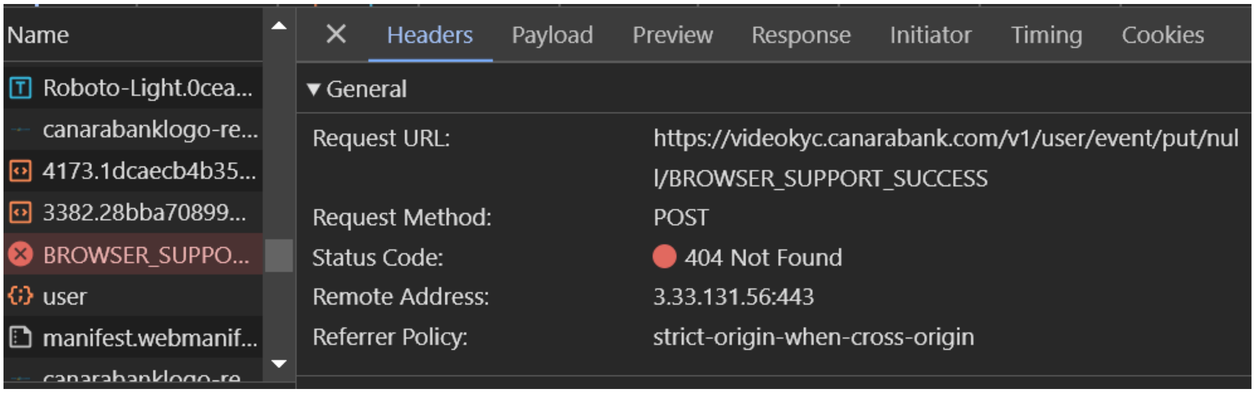
Errors for canarabank.in

1. **403** Forbidden error occurs when you attempt to access a resource that you don't have permission to view.



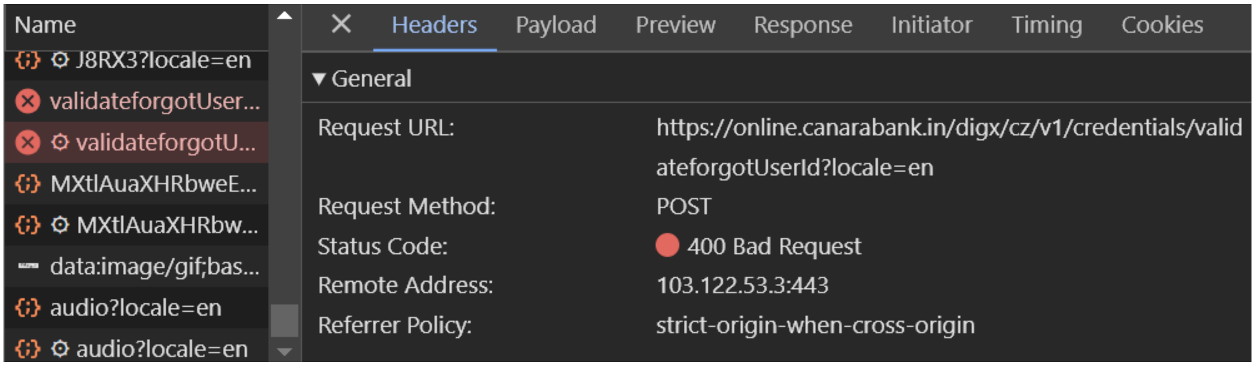
```
Request URL:       https://online.canarabank.in/digx/j_security_check?locale
                   =en
Request Method:    POST
Status Code:       ● 403 Forbidden
Remote Address:    103.122.53.3:443
Referrer Policy:   strict-origin-when-cross-origin
```

2. **404 Not Found:** happen if you try to access a URL or page that doesn't exist on the server.



```
Request URL:       https://videokyc.canarabank.com/v1/user/event/put/nul
                   l/BROWSER_SUPPORT_SUCCESS
Request Method:    POST
Status Code:       ● 404 Not Found
Remote Address:    3.33.131.56:443
Referrer Policy:   strict-origin-when-cross-origin
```

3. **400 Bad Request:** Happens when the request format is invalid (e.g., incorrect URL structure or missing parameters).



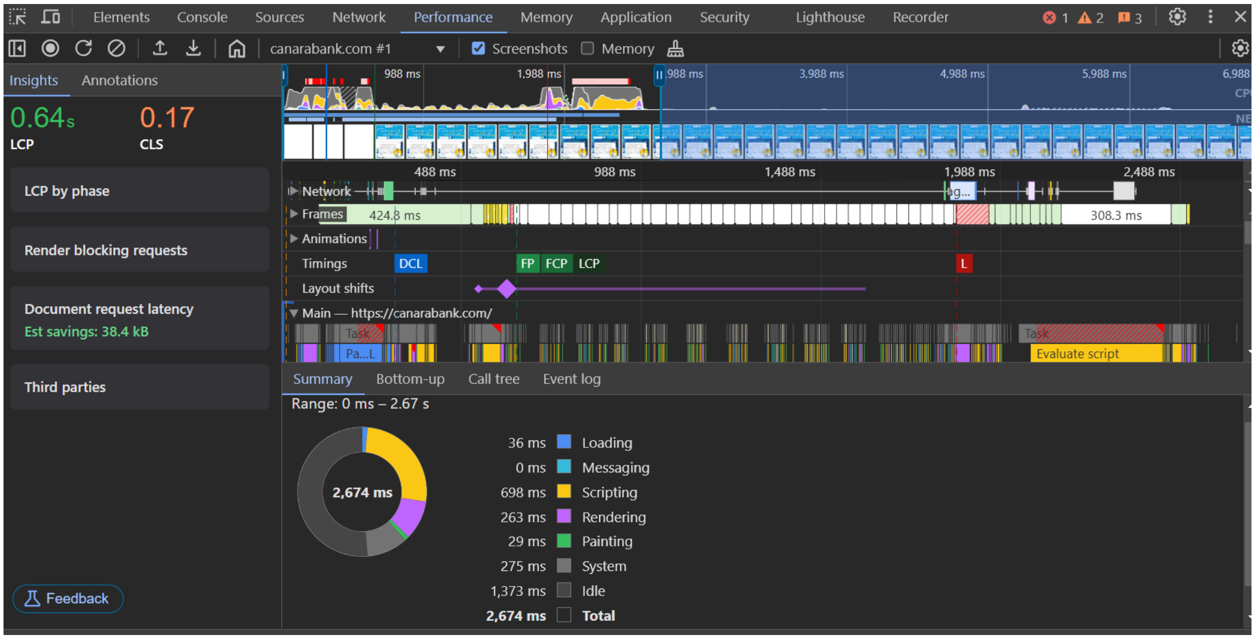
J8RX3?locale=en
validateforgotUser...
validateforgotU...
MXtlAuaXHRbweE...
MXtlAuaXHRbw...
data:image/gif;bas...
audio?locale=en
audio?locale=en

▼General

Request URL:	https://online.canarabank.in/digx/cz/v1/credentials/validateforgotUserId?locale=en
Request Method:	POST
Status Code:	● 400 Bad Request
Remote Address:	103.122.53.3:443
Referrer Policy:	strict-origin-when-cross-origin

C. Capture the Performance metrics that your browser records when a page is loaded and also report the list the cookies used and the associated flags in the request and response headers. Please report the browser name and screenshot of the performance metrics reported for any one of the page loads.

1. Canara bank



0.64s LCP

0.17 CLS

LCP by phase

Render blocking requests

Document request latency
Est savings: 38.4 kB

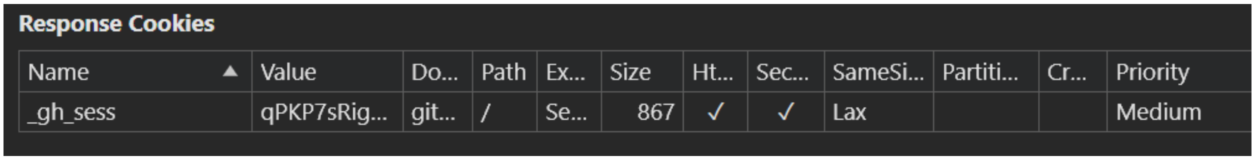
Third parties

Range: 0 ms – 2.67 s

36 ms	Loading
0 ms	Messaging
698 ms	Scripting
263 ms	Rendering
29 ms	Painting
275 ms	System
1,373 ms	Idle
2,674 ms	**Total**

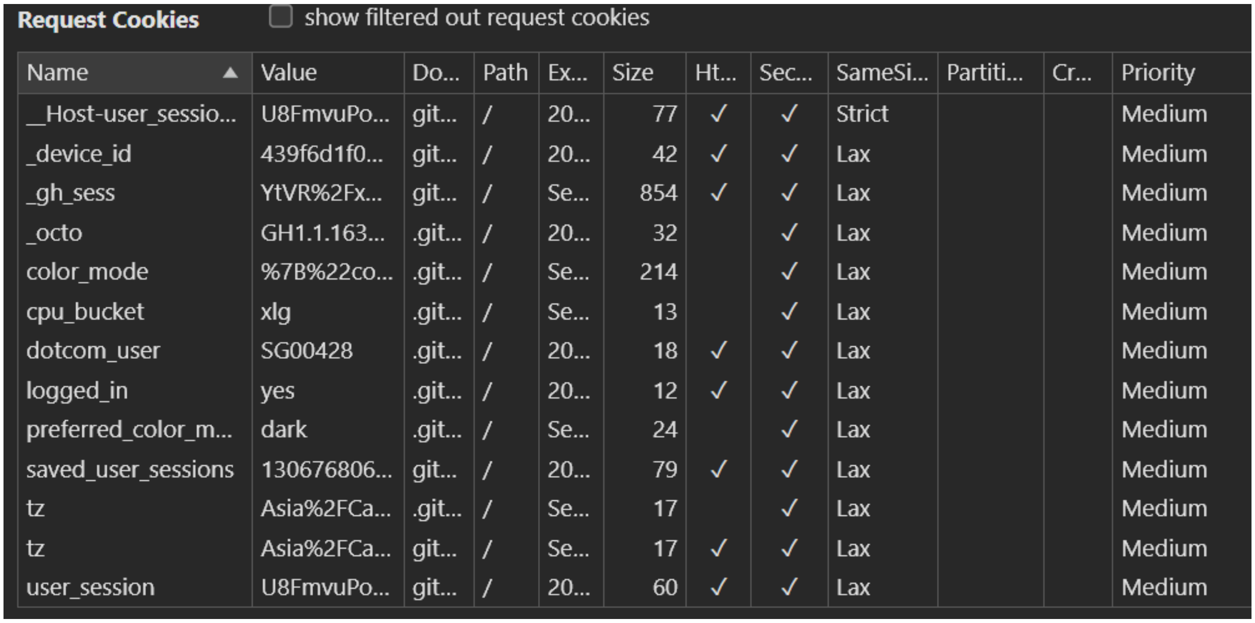
INP : 59ms
DCL : 301.11 ms
FP : 643.20 ms
FCP : 643.20 ms
L: 1.87 s

Cookies:



Response Cookies

Name ▲	Value	Do...	Path	Ex...	Size	Ht...	Sec...	SameSi...	Partiti...	Cr...	Priority
_gh_sess	qPKP7sRig...	git...	/	Se...	867	✓	✓	Lax			Medium



Request Cookies □ show filtered out request cookies

Name ▲	Value	Do...	Path	Ex...	Size	Ht...	Sec...	SameSi...	Partiti...	Cr...	Priority
__Host-user_sessio...	U8FmvuPo...	git...	/	20...	77	✓	✓	Strict			Medium
_device_id	439f6d1f0...	git...	/	20...	42	✓	✓	Lax			Medium
_gh_sess	YtVR%2Fx...	git...	/	Se...	854	✓	✓	Lax			Medium
_octo	GH1.1.163...	.git...	/	20...	32		✓	Lax			Medium
color_mode	%7B%22co...	.git...	/	Se...	214		✓	Lax			Medium
cpu_bucket	xlg	.git...	/	Se...	13		✓	Lax			Medium
dotcom_user	SG00428	.git...	/	20...	18	✓	✓	Lax			Medium
logged_in	yes	.git...	/	20...	12	✓	✓	Lax			Medium
preferred_color_m...	dark	.git...	/	Se...	24		✓	Lax			Medium
saved_user_sessions	130676806...	git...	/	20...	79	✓	✓	Lax			Medium
tz	Asia%2FCa...	.git...	/	Se...	17		✓	Lax			Medium
tz	Asia%2FCa...	git...	/	Se...	17	✓	✓	Lax			Medium
user_session	U8FmvuPo...	git...	/	20...	60	✓	✓	Lax			Medium

1. **_Host-user_session_same_site - Flags: HttpOnly,Secure, Samesite(strict)**
2. **_device_id - Flags:HttpOnly,Secure, SameSite(Lax)**
3. **_gh_sess - Flags:HttpOnly,Secure, SameSite(Lax)**
4. **_octo - Flags:Secure, SameSite(Lax)**
5. **color_mode - Flags:Secure, SameSite(Lax)**
6. **cpu_bucket - Flags:,Secure, SameSite(Lax)**
7. **dotcom_user - Flags:HttpOnly,Secure, SameSite(Lax)**
8. **logged_in - Flags:HttpOnly,Secure, SameSite(Lax)**
9. **preferred_color_mode - Flags:Secure, SameSite(Lax)**
10. **saved_user_sessions - Flags:HttpOnly,Secure, SameSite(Lax)**
11. **tz - Flags:Secure, SameSite(Lax)**
12. **user_session - Flags:HttpOnly,Secure, SameSite(Lax)**

6. Conclusion

Summarize the insights gained from network traffic analysis, unique packet queries, and live packet capture.

7. References

[List sources, RFCs, and documentation]