COMPUTER NETWORKS

LAB REPORT

ASSIGNMENT 5

DEBJIT DHAR

BCSE UG 3

ROLL:002210501106

GROUP: A3

SUBMISSION: 18/11/2024

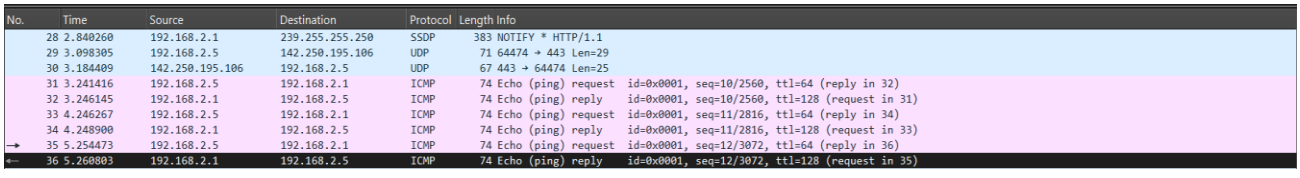
Problem Statement: Packet tracer and traffic analysis with Wireshark

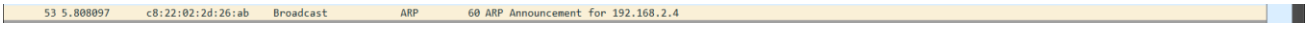
Wireshark Files at: https://github.com/Debjit-Dhar/Networks

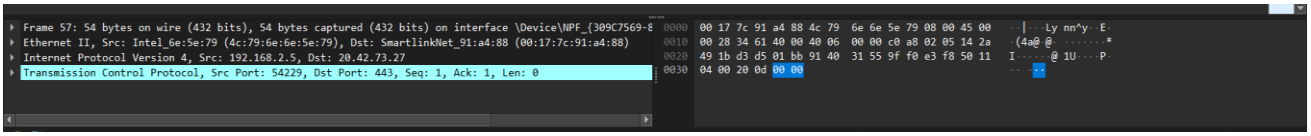
PROBLEM 1:

Generate some ICMP traffic by using the Ping command line tool to check the connectivity of a neighboring machine (or router). Note the results in Wireshark. The initial ARP request broadcast from your PC determines the physical MAC address of the network IP Address, and the ARP reply from the neighboring system. After the ARP request, the pings (ICMP echo request and replies) can be seen.

RESULTS:







EXPLANATION:

Packet Breakdown:

1. Packet Numbers 30-36:
    - Packets 30 and 31 are from an external IP (142.250.195.106),

unrelated to my ping command. They are UDP packets, not ICMP, so they might be background traffic or part of another application's activity.
    - Packets 32-36 are related to the ping (ICMP Echo) requests and replies between my computer (192.168.2.5) and the router (192.168.2.1).
2. Ping (ICMP) Request and Reply Packets:
    - Each ICMP packet has an identifier id=0x0001, showing that they are part of the same ping session.
    - The sequence numbers (seq) in each packet increment, which indicates different ping attempts:
        - Packet 32: seq=10
        - Packet 33: seq=10 (reply to packet 32)
        - Packet 34: seq=11
        - Packet 35: seq=11 (reply to packet 34)
        - Packet 36: seq=12 (reply to packet 35)
3. TTL (Time to Live) Field:
    - The TTL value in ICMP requests from my computer is 64, while replies from the router have a TTL of 128. The TTL value indicates the maximum number of hops a packet can take before being discarded. The initial TTL value is often set by the operating system:
        - Linux systems typically set it to 64.
        - Windows systems often set it to 128.

Packet Analysis (Details Section):

- Ethernet Frame:
    - Shows the MAC addresses of the source (00:17:7c:91:a4:88) and destination (4c:79:6e:6e:5e:79) devices on the network.
- IP Header:
    - Displays the source IP (192.168.2.1) and destination IP
    - (192.168.2.5), as well as the protocol in use (ICMP).
- ICMP Protocol Details:
    - The ICMP request and reply packets contain basic ping data:
        - Sequence number, ID, and TTL fields are visible.

- The TTL difference (64 vs. 128) between the source and destination suggests my system is initiating the ping, and the router is replying.

PROBLEM 2:

Generate some web traffic and

a. find the list the different protocols that appear in the protocol column in the unfiltered packet-listing window of Wireshark.

b. How long did it take from when the HTTP GET message was sent until the HTTPOK reply was received? (By default, the value of the Time column in the packet listing window is the amount of time, in seconds, since Wireshark tracing began. To display the Time field in time-of-day format, select the Wireshark View pull down menu, then select Time Display Format, then select Time-of-day.)

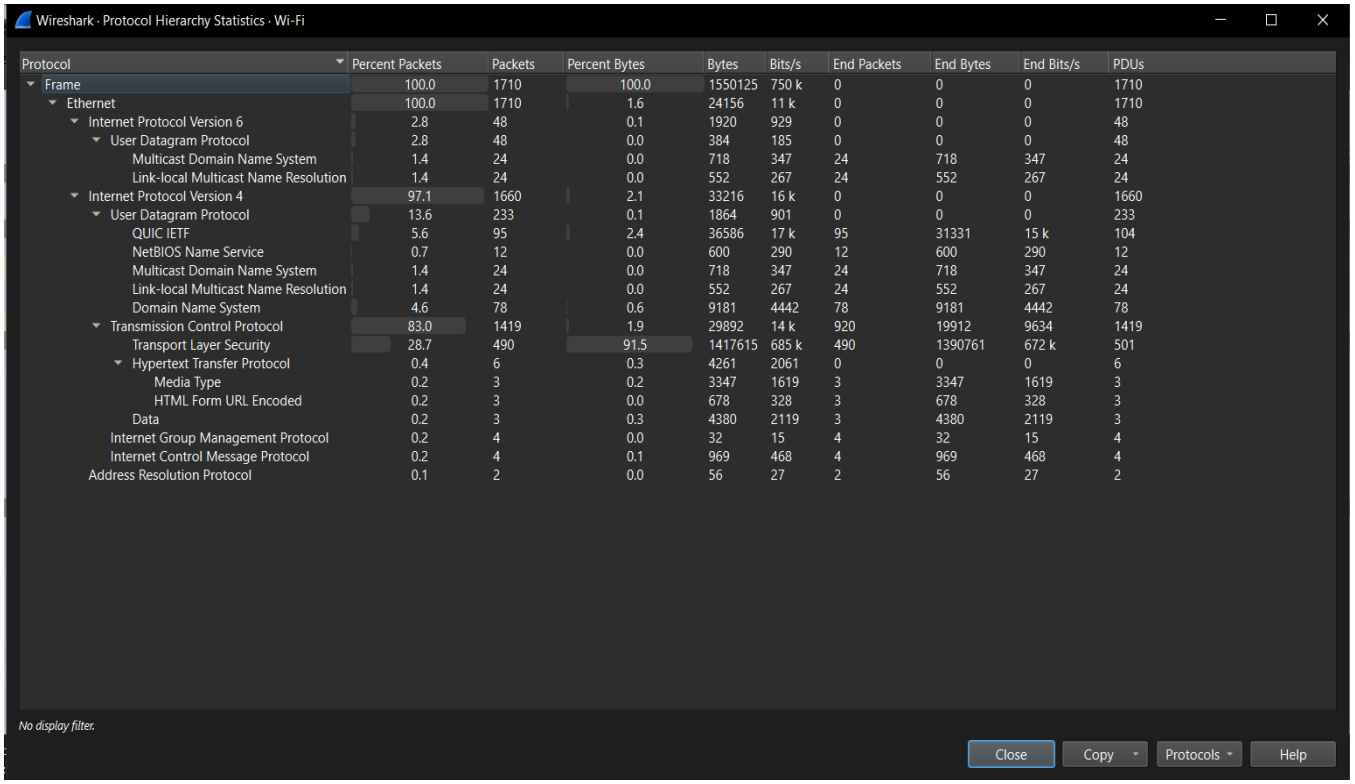
c. What is the Internet address of the website? What is the Internet address of your computer?

d. Search back through your capture, and find an HTTP packet containing a GET command. Click on the packet in the Packet List Panel. Then expand the HTTP layer in the Packet Details Panel, from the packet.

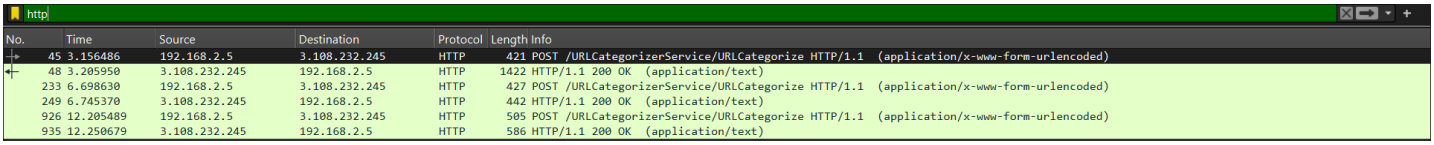
e. Find out the value of the Host from the Packet Details Panel, within the GET command.

RESULTS AND EXPLANATIONS:

a) All the protocols that were captured are listed below.

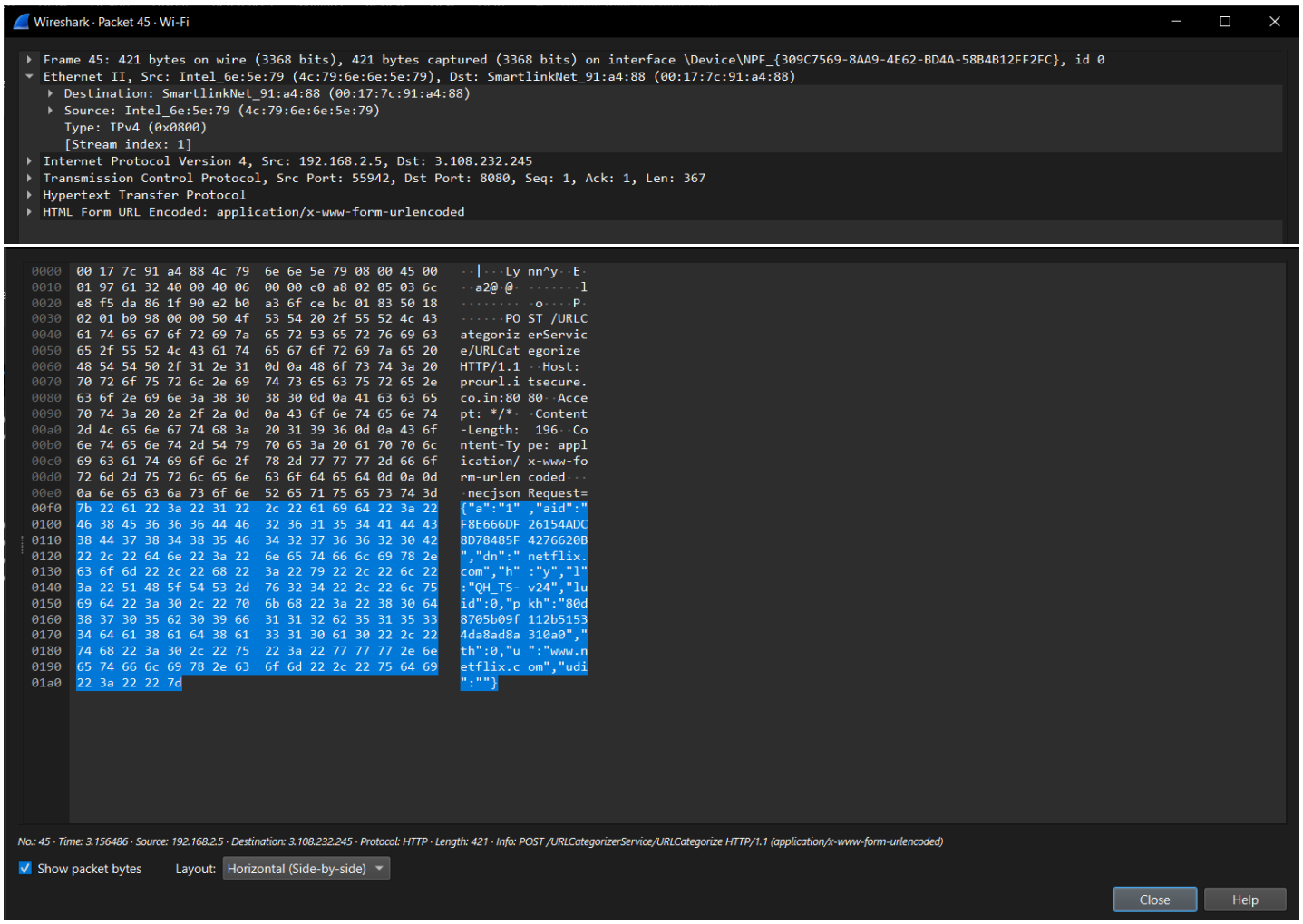


b)According to the delta time taken(between post and ok), it took 0.49464 seconds approximately to get the HTTP response.



c) The local IP is 192.168.2.5 and the server IP is 3.108.232.245(as shown in the above figure)

d) Shown below is the details of a HTTP (post) packet



e) The value of Host as shown above is: www.netflix.com (as shown in line 0180 to 01a0) in the above figure.

PROBLEM 3:

Highlight the Hex and ASCII representations of the packet in the Packet Bytes Panel.

RESULTS and EXPLANATIONS:

The picture in part 2e clearly shows the hex and ASCII representation of the packet in Packet Bytes panel.

PROBLEM 4:

Find out the first 4 bytes of the Hex value of the Host parameter from the Packet Bytes Panel.

RESULTS and EXPLANATIONS:

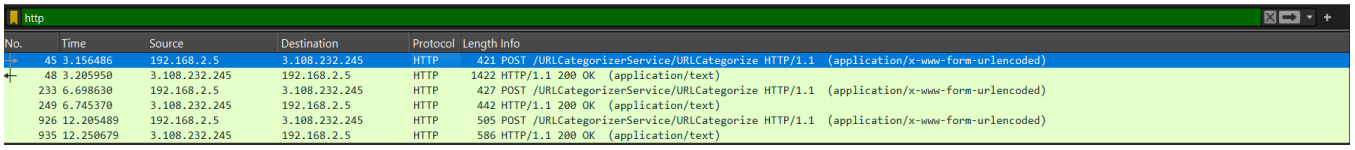
Referring to the image in 2e->

The first 4 bytes of the Hex value of the Host parameter from the Packet Bytes Panel are: 48 6f 73 74

PROBLEM 5:

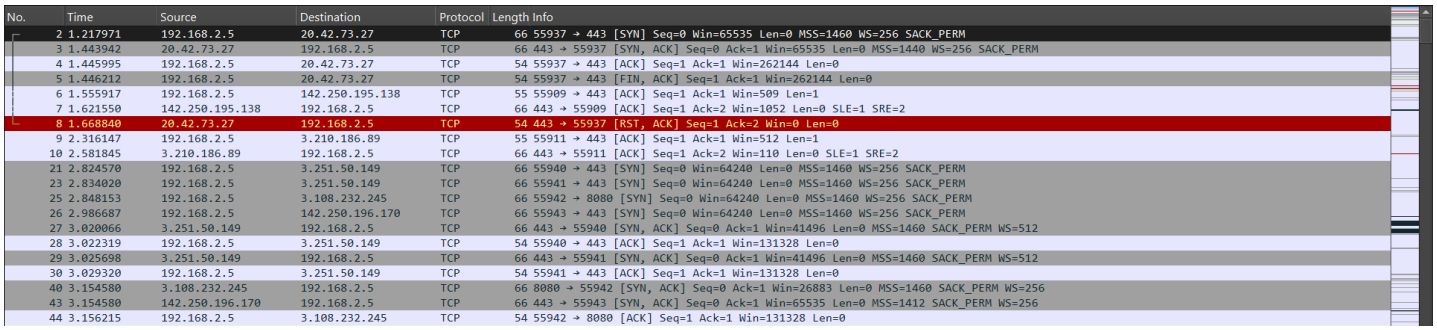
Filter packets with http, TCP, DNS and other protocols.

RESULTS:



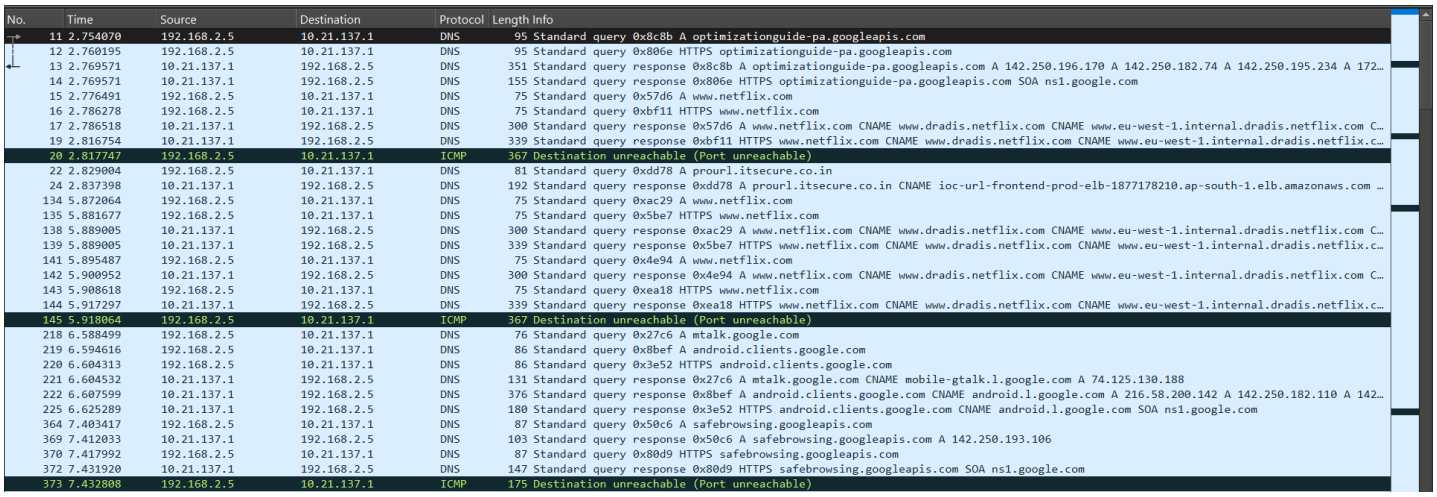
No.	Time	Source	Destination	Protocol	Length	Info
45	3.156486	192.168.2.5	3.108.232.245	HTTP	421	POST /URLCategorizerService/URLCategorize HTTP/1.1 (application/x-www-form-urlencoded)
48	3.205950	3.108.232.245	192.168.2.5	HTTP	1422	HTTP/1.1 200 OK (application/text)
233	6.698630	192.168.2.5	3.108.232.245	HTTP	427	POST /URLCategorizerService/URLCategorize HTTP/1.1 (application/x-www-form-urlencoded)
249	6.745370	3.108.232.245	192.168.2.5	HTTP	442	HTTP/1.1 200 OK (application/text)
926	12.205489	192.168.2.5	3.108.232.245	HTTP	505	POST /URLCategorizerService/URLCategorize HTTP/1.1 (application/x-www-form-urlencoded)
935	12.250679	3.108.232.245	192.168.2.5	HTTP	586	HTTP/1.1 200 OK (application/text)

HTTP Packets



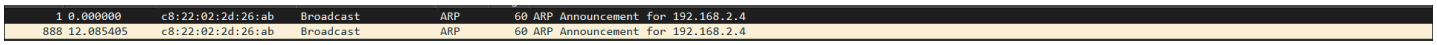
No.	Time	Source	Destination	Protocol	Length	Info
2	1.217971	192.168.2.5	20.42.73.27	TCP	66	55937 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM
3	1.443942	20.42.73.27	192.168.2.5	TCP	66	443 → 55937 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM
4	1.445995	192.168.2.5	20.42.73.27	TCP	54	55937 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
5	1.446212	192.168.2.5	20.42.73.27	TCP	54	55937 → 443 [FIN, ACK] Seq=1 Ack=1 Win=262144 Len=0
6	1.555917	192.168.2.5	142.250.195.138	TCP	55	55909 → 443 [ACK] Seq=1 Ack=1 Win=509 Len=1
7	1.621550	142.250.195.138	192.168.2.5	TCP	66	443 → 55909 [ACK] Seq=1 Ack=2 Win=1052 Len=0 SLE=1 SRE=2
8	1.668840	20.42.73.27	192.168.2.5	TCP	54	443 → 55937 [RST, ACK] Seq=1 Ack=2 Win=0 Len=0
9	2.316147	192.168.2.5	3.210.186.89	TCP	55	55911 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1
10	2.581845	3.210.186.89	192.168.2.5	TCP	66	443 → 55911 [ACK] Seq=1 Ack=2 Win=110 Len=0 SLE=1 SRE=2
21	2.824570	192.168.2.5	3.251.50.149	TCP	66	55940 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
23	2.834020	192.168.2.5	3.251.50.149	TCP	66	55941 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
25	2.848153	192.168.2.5	3.108.232.245	TCP	66	55942 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
26	2.986687	192.168.2.5	142.250.196.170	TCP	66	55943 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
27	3.020066	3.251.50.149	192.168.2.5	TCP	66	443 → 55940 [SYN, ACK] Seq=0 Ack=1 Win=41496 Len=0 MSS=1460 SACK_PERM WS=512
28	3.022319	192.168.2.5	3.251.50.149	TCP	54	55940 → 443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
29	3.025698	3.251.50.149	192.168.2.5	TCP	66	443 → 55941 [SYN, ACK] Seq=0 Ack=1 Win=41496 Len=0 MSS=1460 SACK_PERM WS=512
30	3.029320	192.168.2.5	3.251.50.149	TCP	54	55941 → 443 [ACK] Seq=1 Ack=1 Win=131328 Len=0
40	3.154580	3.108.232.245	192.168.2.5	TCP	66	8080 → 55942 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0 MSS=1460 SACK_PERM WS=256
43	3.154580	142.250.196.170	192.168.2.5	TCP	66	443 → 55943 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1412 SACK_PERM WS=256
44	3.156215	192.168.2.5	3.108.232.245	TCP	54	55942 → 8080 [ACK] Seq=1 Ack=1 Win=131328 Len=0

TCP Packets



No.	Time	Source	Destination	Protocol	Length	Info
11	2.754070	192.168.2.5	10.21.137.1	DNS	95	Standard query 0x8c8b A optimizationguide-pa.googleapis.com
12	2.760195	192.168.2.5	10.21.137.1	DNS	95	Standard query 0x806e HTTPS optimizationguide-pa.googleapis.com
13	2.769571	10.21.137.1	192.168.2.5	DNS	351	Standard query response 0x8c8b A optimizationguide-pa.googleapis.com A 142.250.196.170 A 142.250.182.74 A 142.250.195.234 A 172…
14	2.769571	10.21.137.1	192.168.2.5	DNS	155	Standard query response 0x806e HTTPS optimizationguide-pa.googleapis.com SOA ns1.google.com
15	2.776491	192.168.2.5	10.21.137.1	DNS	75	Standard query 0x57d6 A www.netflix.com
16	2.786278	192.168.2.5	10.21.137.1	DNS	75	Standard query 0xbf11 HTTPS www.netflix.com
17	2.786518	10.21.137.1	192.168.2.5	DNS	300	Standard query response 0x57d6 A www.netflix.com CNAME www.dradis.netflix.com CNAME www.eu-west-1.internal.dradis.netflix.com C…
19	2.816754	10.21.137.1	192.168.2.5	DNS	339	Standard query response 0xbf11 HTTPS www.netflix.com CNAME www.dradis.netflix.com CNAME www.eu-west-1.internal.dradis.netflix.c…
20	2.817747	192.168.2.5	10.21.137.1	ICMP	367	Destination unreachable (Port unreachable)
22	2.829004	192.168.2.5	10.21.137.1	DNS	81	Standard query 0xdd78 A prourl.itsecure.co.in
24	2.837398	10.21.137.1	192.168.2.5	DNS	192	Standard query response 0xdd78 A prourl.itsecure.co.in CNAME ioc-url-frontend-prod-elb-1877178210.ap-south-1.elb.amazonaws.com …
134	5.872064	192.168.2.5	10.21.137.1	DNS	75	Standard query 0xac29 A www.netflix.com
135	5.881677	192.168.2.5	10.21.137.1	DNS	75	Standard query 0x5be7 HTTPS www.netflix.com
138	5.889005	10.21.137.1	192.168.2.5	DNS	300	Standard query response 0xac29 A www.netflix.com CNAME www.dradis.netflix.com CNAME www.eu-west-1.internal.dradis.netflix.com C…
139	5.889005	10.21.137.1	192.168.2.5	DNS	339	Standard query response 0x5be7 HTTPS www.netflix.com CNAME www.dradis.netflix.com CNAME www.eu-west-1.internal.dradis.netflix.c…
141	5.895487	192.168.2.5	10.21.137.1	DNS	75	Standard query 0x4e94 A www.netflix.com
142	5.900952	10.21.137.1	192.168.2.5	DNS	300	Standard query response 0x4e94 A www.netflix.com CNAME www.dradis.netflix.com CNAME www.eu-west-1.internal.dradis.netflix.com C…
143	5.908618	192.168.2.5	10.21.137.1	DNS	75	Standard query 0xea18 HTTPS www.netflix.com
144	5.917297	10.21.137.1	192.168.2.5	DNS	339	Standard query response 0xea18 HTTPS www.netflix.com CNAME www.dradis.netflix.com CNAME www.eu-west-1.internal.dradis.netflix.c…
145	5.918064	192.168.2.5	10.21.137.1	ICMP	367	Destination unreachable (Port unreachable)
218	6.588499	192.168.2.5	10.21.137.1	DNS	76	Standard query 0x27c6 A mtalk.google.com
219	6.594616	192.168.2.5	10.21.137.1	DNS	86	Standard query 0x8bef A android.clients.google.com
220	6.604313	192.168.2.5	10.21.137.1	DNS	86	Standard query 0x3e52 HTTPS android.clients.google.com
221	6.604532	10.21.137.1	192.168.2.5	DNS	131	Standard query response 0x27c6 A mtalk.google.com CNAME mobile-gtalk.l.google.com A 74.125.130.188
222	6.607599	10.21.137.1	192.168.2.5	DNS	376	Standard query response 0x8bef A android.clients.google.com CNAME android.l.google.com A 216.58.200.142 A 142.250.182.110 A 142…
225	6.625289	10.21.137.1	192.168.2.5	DNS	180	Standard query response 0x3e52 HTTPS android.clients.google.com CNAME android.l.google.com SOA ns1.google.com
364	7.403417	192.168.2.5	10.21.137.1	DNS	87	Standard query 0x50c6 A safebrowsing.googleapis.com
369	7.412033	10.21.137.1	192.168.2.5	DNS	103	Standard query response 0x50c6 A safebrowsing.googleapis.com A 142.250.193.106
370	7.417992	192.168.2.5	10.21.137.1	DNS	87	Standard query 0x80d9 HTTPS safebrowsing.googleapis.com
372	7.431920	10.21.137.1	192.168.2.5	DNS	147	Standard query response 0x80d9 HTTPS safebrowsing.googleapis.com SOA ns1.google.com
373	7.432808	192.168.2.5	10.21.137.1	ICMP	175	Destination unreachable (Port unreachable)

DNS Packets



1	0.000000	c8:22:02:2d:26:ab	Broadcast	ARP	60	ARP Announcement for 192.168.2.4
888	12.085405	c8:22:02:2d:26:ab	Broadcast	ARP	60	ARP Announcement for 192.168.2.4

ARP Packets

EXPLANATIONS:

1. HTTP Traffic

- This section shows HTTP traffic between two IP addresses: 192.168.2.5 (source) and 3.108.232.245 (destination).
- The HTTP requests are POST requests made to a service endpoint /URLCategorizerService/URLCategorize.
- Each POST request contains various response statuses with 200 OK, indicating successful responses.

- The requests seem to be sending data encoded as application/x-www-form-urlencoded and receiving responses in application/text.
- The HTTP POST requests indicate communication with a URL categorization or filtering service, possibly a security or web-filtering application.

2. TCP Traffic

- This section shows TCP traffic with various source and destination IPs.
- It includes typical TCP flags like SYN, ACK, RST, and FIN, used to manage connections between hosts.
- Notably:
    - Packet #8 from 20.42.73.27 to 192.168.2.5 contains an RST (reset) flag, indicating a forced connection termination.
    - The TCP packets indicate several connection attempts to port 443 (commonly HTTPS) and 8080.
- These interactions may represent normal web traffic or attempted connections to web servers, with some connections being reset or terminated.

3. DNS and ICMP Traffic

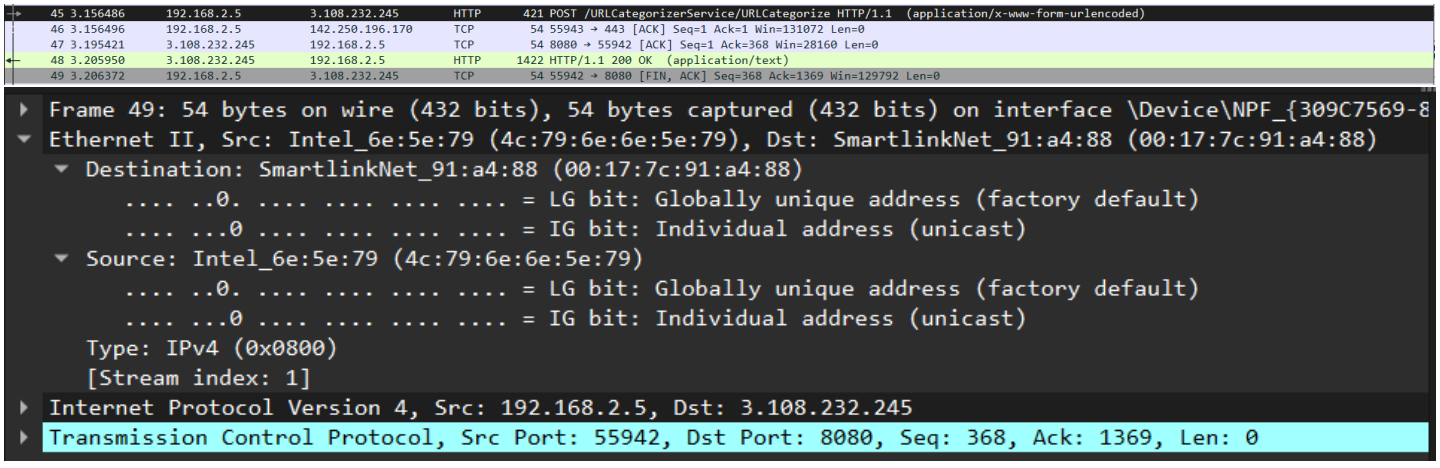
- This screenshot shows DNS query and response traffic between 192.168.2.5 and external servers.
- Queries include lookups for services like googleapis.com, netflix.com, and clients.google.com, possibly for service access or monitoring.
- The DNS response includes CNAME records, pointing to other domains (e.g., netflix.com CNAME resolution to dradis.netflix.com).
- **ICMP Messages**:

- ICMP packets, labeled as "Destination unreachable (Port unreachable)," appear interspersed with DNS requests.
- These ICMP messages are responses from the DNS servers back to the client 192.168.2.5.
- An ICMP Destination Unreachable message with a "Port unreachable" code usually indicates that the DNS server attempted to reach a particular service on a port that was not available or was blocked by the firewall.
- This could suggest the client tried querying a service that isn't accessible, or the server did not support the requested DNS service type.
- The repeated ICMP messages could indicate network configuration issues or firewall settings that block specific DNS responses.

PROBLEM 6:

Search through your capture, and find an HTTP packet coming back from the server (TCP Source Port == 80). Expand the Ethernet layer in the Packet Details Panel.

RESULTS:



EXPLANATIONS:

1. Packet 45:
    - This is an HTTP POST request from 192.168.2.5 to 3.108.232.245.
    - It's being sent to port 8080, as indicated in the HTTP/1.1 request line.
    - The HTTP POST request is targeting the endpoint /URLCategorizerService/URLCategorize.
    - The content type is application/x-www-form-urlencoded, which suggests that the data is being submitted in a form format.
2. Packets 46 and 47:
    - These packets are TCP acknowledgment (ACK) packets between 192.168.2.5 and another IP, 3.108.232.245.
    - The capture shows the TCP connection details (like sequence and acknowledgment numbers) but not the HTTP content, as they're just part of the handshake and data acknowledgment.
3. Packet 48:
    - This is an HTTP response from 3.108.232.245 back to 192.168.2.5.
    - It indicates a successful 200 OK response with the Content-Type as application/text.
4. Packet 49:
    - This packet shows a TCP FIN (Finish) flag from 192.168.2.5 to 3.108.232.245, indicating the closing of the connection.

Why Port 8080 Instead of Port 80?

Port 8080 is often used as an alternative HTTP port when the default port 80 is unavailable or deliberately not used. Here are some common reasons for using port 8080:

- Alternative HTTP Port: Port 8080 is commonly used for web traffic, particularly when port 80 is reserved for another application or service.
- Testing and Development: Many applications use port 8080 for testing purposes, especially in development environments.
- Web Proxies: In some setups, port 8080 is used by proxy servers to handle HTTP traffic.

In this case, 192.168.2.5 is communicating with 3.108.232.245 over port 8080, likely because the server is configured to handle HTTP requests on port 8080 rather than the default port 80.

PROBLEM 7:

What are the manufacturers of your PC's Network Interface Card (NIC), and the servers NIC?

RESULTS:

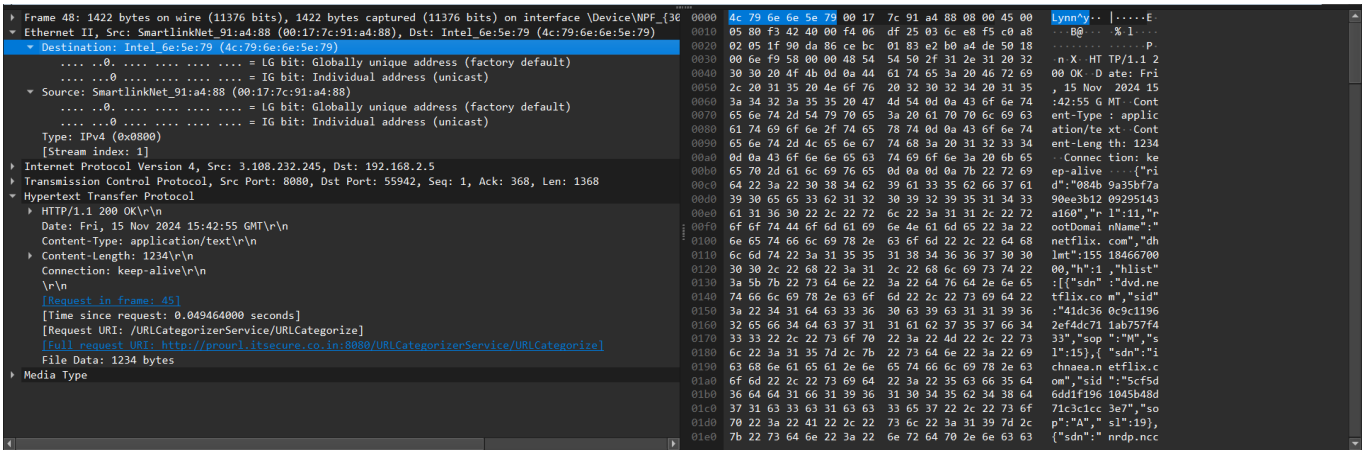
So according to the image in problem 6, my PC's Network Interface Card (NIC) has the manufacturer: Intel.

and the server's Network Interface Card (NIC) has the manufacturer: SmartlinkNet

PROBLEM 8:

What are the Hex values (shown the raw bytes panel) of the two NICS Manufacturers OUIs?

RESULTS:



EXPLANATIONS:

The hex values (shown the raw bytes panel) of the two NICS Manufacturers OUIs are: 4c 79 6e 6e 5c 79 (Intel (my)NIC raw bytes) and 00 17 7c 91 a4 88 (SmartlinkNet (Server) NIC)

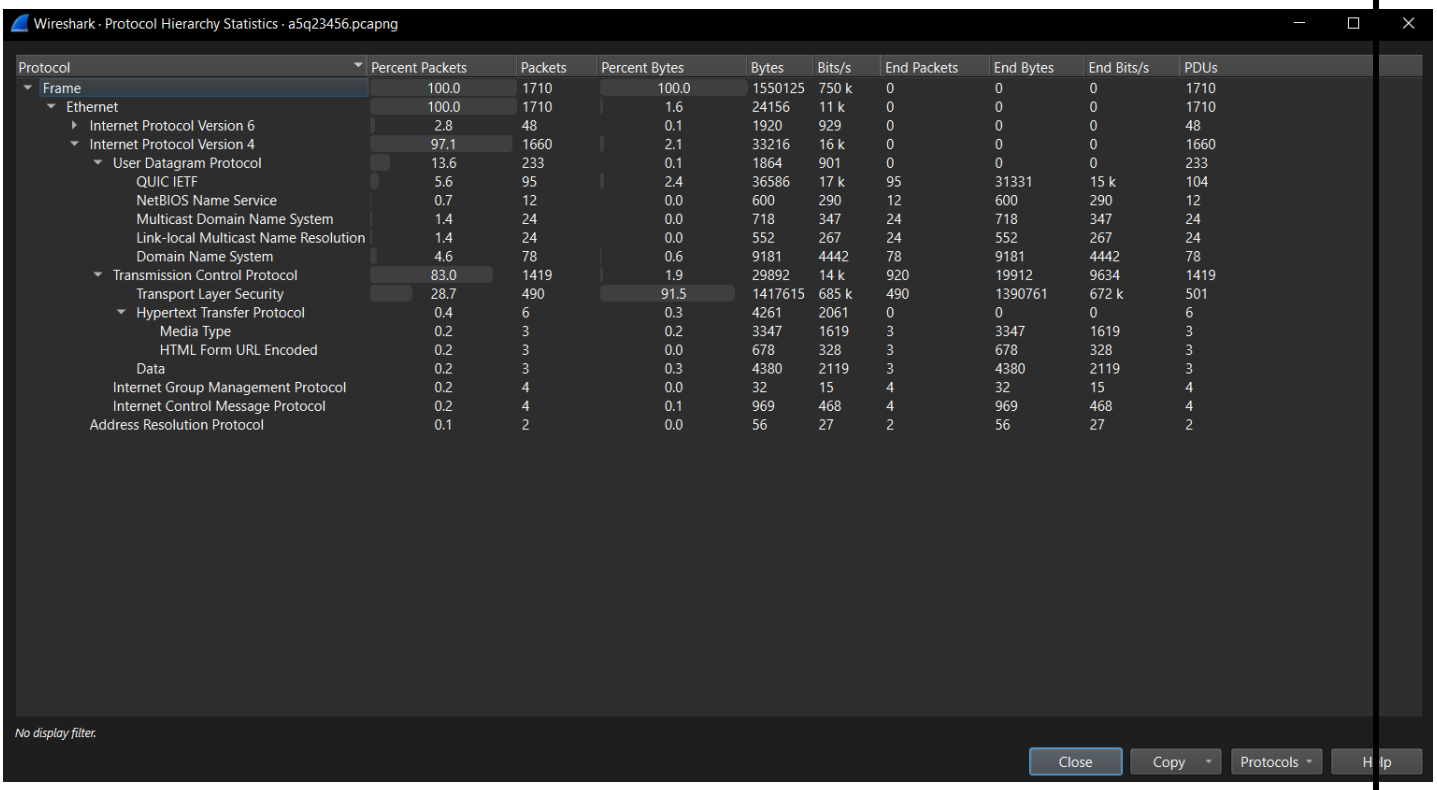
PROBLEM 9:

Find the following statistics:

a. What percentage of packets in your capture are TCP, and give an example of the higher level protocol which uses TCP?

b. What percentage of packets in your capture are UDP, and give an example of the higher level protocol which uses UDP?

RESULTS:



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	1710	100.0	1550125	750 k	0	0	0	1710
Ethernet	100.0	1710	1.6	24156	11 k	0	0	0	1710
Internet Protocol Version 6	2.8	48	0.1	1920	929	0	0	0	48
Internet Protocol Version 4	97.1	1660	2.1	33216	16 k	0	0	0	1660
User Datagram Protocol	13.6	233	0.1	1864	901	0	0	0	233
QUIC IETF	5.6	95	2.4	36586	17 k	95	31331	15 k	104
NetBIOS Name Service	0.7	12	0.0	600	290	12	600	290	12
Multicast Domain Name System	1.4	24	0.0	718	347	24	718	347	24
Link-local Multicast Name Resolution	1.4	24	0.0	552	267	24	552	267	24
Domain Name System	4.6	78	0.6	9181	4442	78	9181	4442	78
Transmission Control Protocol	83.0	1419	1.9	29892	14 k	920	19912	9634	1419
Transport Layer Security	28.7	490	91.5	1417615	685 k	490	1390761	672 k	501
Hypertext Transfer Protocol	0.4	6	0.3	4261	2061	0	0	0	6
Media Type	0.2	3	0.2	3347	1619	3	3347	1619	3
HTML Form URL Encoded	0.2	3	0.0	678	328	3	678	328	3
Data	0.2	3	0.3	4380	2119	3	4380	2119	3
Internet Group Management Protocol	0.2	4	0.0	32	15	4	32	15	4
Internet Control Message Protocol	0.2	4	0.1	969	468	4	969	468	4
Address Resolution Protocol	0.1	2	0.0	56	27	2	56	27	2

EXPLANATIONS:

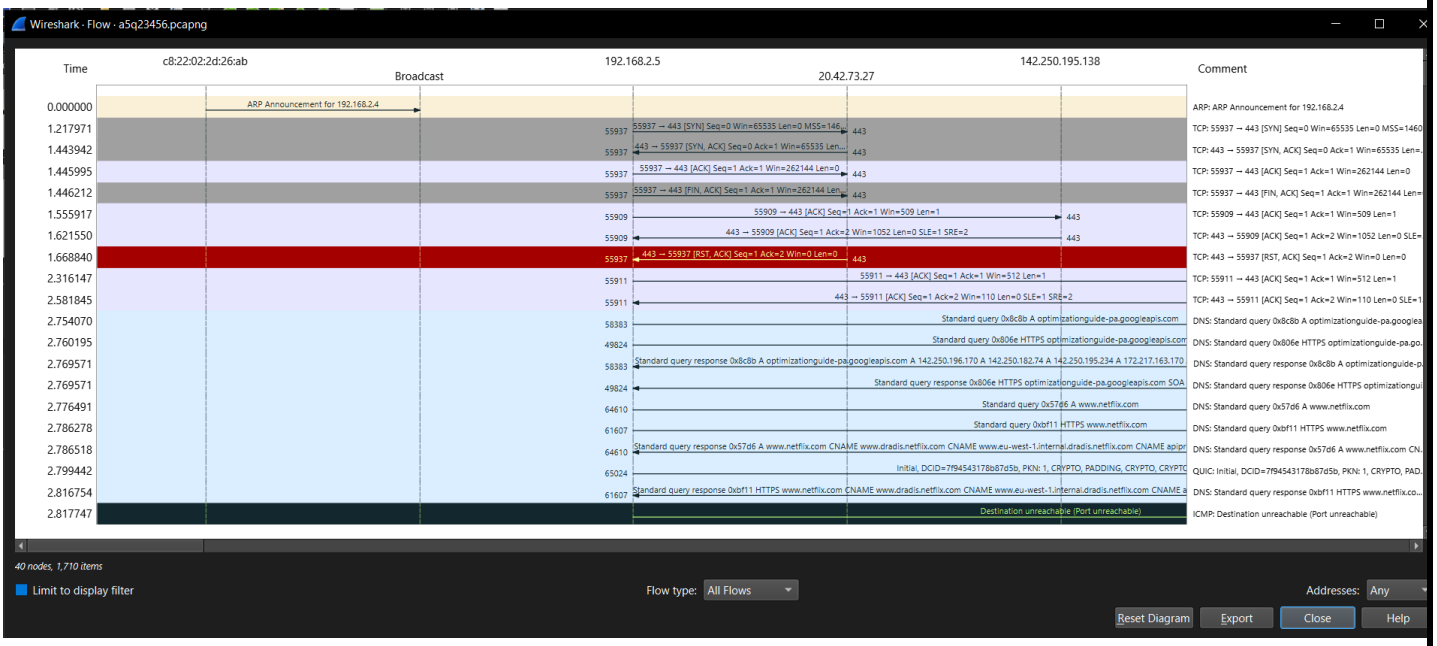
a) From the above statistics, the percentage of TCP packets is 83.00% A protocol that uses TCP is HTTP.

b) The percentage of UDP packets is 13.60%. A protocol that uses UDP is DNS.

PROBLEM 10:

Find the traffic flow Select the Statistics->Flow Graph menu option. Choose General Flow and Network Source options, and click the OK button.

RESULTS:



EXPLANATIONS:

This Wireshark flow view displays a network capture involving several IP addresses, protocols, and ports. Here's a breakdown of what's visible in the screenshot:

1. **ARP Broadcast**: The capture begins with an ARP announcement for IP 192.168.2.4, which is broadcasted to all devices. ARP (Address Resolution Protocol) is used to resolve IP addresses to MAC addresses on a local network.
2. **TCP Connections**:
    - IP 192.168.2.5 is establishing connections with IP 20.42.73.27 and IP 142.250.195.138 on several ports.
    - For the connection between 192.168.2.5 and 20.42.73.27, TCP port 443 (commonly used for HTTPS) and port 55937 are involved.
    - There's a standard three-way handshake observed with SYN, SYN-ACK, and ACK packets. However, this connection later shows a RST (reset) packet, which indicates that the connection was forcibly closed. This could happen if the server or client

abruptly terminated the session.

3. **DNS Queries**:
    - 192.168.2.5 is making DNS queries to resolve domain names related to optimizationguide-pa.googleapis.com and netflix.com.
    - DNS responses indicate successful resolution of IP addresses for these domains, and CNAME (canonical name) records are also returned. CNAME records are used to alias one domain name to another.
4. **QUIC Protocol**:
    - A QUIC connection attempt is seen with initial handshake data (DCID and CRYPTO fields) to www.netflix.com. QUIC is a newer protocol designed to improve the performance of HTTP/3 traffic.
    - QUIC may use UDP instead of TCP, which can improve speed but has its own reliability mechanisms.
5. **ICMP "Destination Unreachable"**:
    - Towards the end, there’s an ICMP (Internet Control Message Protocol) message indicating "Destination unreachable (Port unreachable)," which suggests that a specific service or port could not be reached. This may happen if the requested port is closed or unavailable on the remote server.