

Task 5: Capture and Analyze Network Trac Using Wireshark

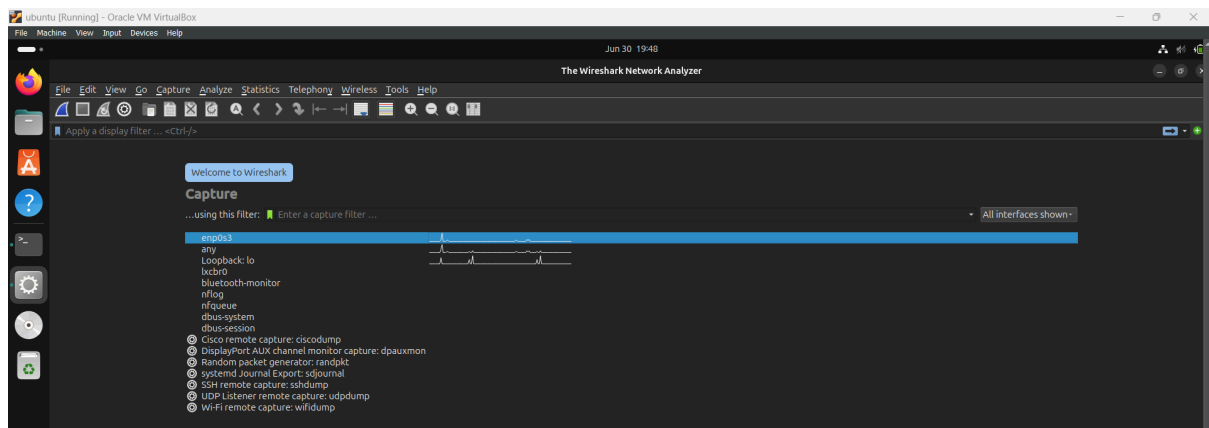
1) Install Wireshark

Here, i installed the wireshark.

```
vamsi@vamsi-VirtualBox:~$ sudo apt update
[sudo] password for vamsi:
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://in.archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://in.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://in.archive.ubuntu.com/ubuntu noble-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
9 packages can be upgraded. Run 'apt list --upgradable' to see them.
vamsi@vamsi-VirtualBox:~$ sudo apt install wireshark -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
wireshark is already the newest version (4.2.2-1.1build3).
The following package was automatically installed and is no longer required:
  python3-netifaces
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 9 not upgraded.
vamsi@vamsi-VirtualBox:~$
```

2) Start capturing on your active network interface.

This is the starting interface of the wireshark.



3) Browse a website or ping a server to generate traffic.

After starting the capture, a new terminal window was opened and the following command was run to ping Google, which generates ICMP packets.

ping google.com -c 5

Then, to generate HTTP, HTTPS, and DNS traffic, a website was accessed using

curl https://example.com

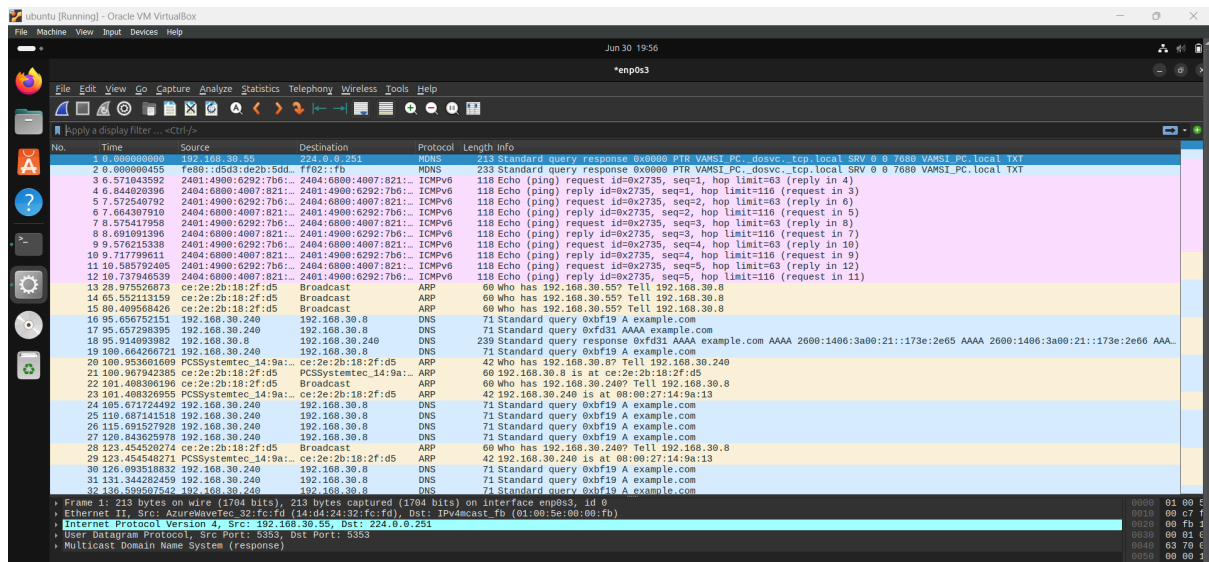
This simulates real-world browsing activity and helps capture various protocol packets like DNS (for name resolution), TCP (for reliable transport), and HTTP/HTTPS (for web content).

Here, we can see that packets are sent .

```
vamsi@vamsi-VirtualBox:~$ ping google.com -c 5
PING google.com (2404:6800:4007:821::200e) 56 data bytes
64 bytes from maa05s26-in-x0e.1e100.net (2404:6800:4007:821::200e): icmp_seq=1 ttl=116 time=273 ms
64 bytes from maa05s26-in-x0e.1e100.net (2404:6800:4007:821::200e): icmp_seq=2 ttl=116 time=91.8 ms
64 bytes from maa05s26-in-x0e.1e100.net (2404:6800:4007:821::200e): icmp_seq=3 ttl=116 time=116 ms
64 bytes from maa05s26-in-x0e.1e100.net (2404:6800:4007:821::200e): icmp_seq=4 ttl=116 time=142 ms
64 bytes from maa05s26-in-x0e.1e100.net (2404:6800:4007:821::200e): icmp_seq=5 ttl=116 time=152 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4013ms
rtt min/avg/max/ndev = 91.818/154.951/273.018/62.652 ms
vamsi@vamsi-VirtualBox:~$ curl https://example.com
```

After pressing the command in the terminal , these are the packets generated.



4).Stop capture after a minute.

I stopped capturing the packets after a minute by clicking on the button.

5).Filter captured packets by protocol (e.g., HTTP, DNS, TCP).

These are dns packets.

The screenshot shows a Kali Linux virtual machine with Wireshark open. The network interface 'enp0s3' is selected for capturing. The packet list shows 38 packets, all DNS queries from 192.168.30.240 to 192.168.30.8. The packet details pane shows the structure of a DNS query for 'example.com'. The packet bytes pane shows the raw data of the query.

No.	Time	Source	Destination	Protocol	Length	Info
16	95.656752151	192.168.30.240	192.168.30.8	DNS	71	Standard query 0xbf19 A example.com
17	95.657203395	192.168.30.240	192.168.30.8	DNS	71	Standard query 0xf631 AAAA example.com
18	95.914893982	192.168.30.8	192.168.30.240	DNS	239	Standard query response 0xf631 AAAA example.com AAAA 2600:1406:3a00:21::173e:2e65 AAAA 2600:1406:3a00:21::173e:2e66 AAAA 2600:1406:3a00:21::173e:2e67 AAAA 2600:1406:3a00:21::173e:2e68 AAAA 2600:1406:3a00:21::173e:2e69 AAAA 2600:1406:3a00:21::173e:2e6a AAAA 2600:1406:3a00:21::173e:2e6b AAAA 2600:1406:3a00:21::173e:2e6c AAAA 2600:1406:3a00:21::173e:2e6d AAAA 2600:1406:3a00:21::173e:2e6e AAAA 2600:1406:3a00:21::173e:2e6f AAAA 2600:1406:3a00:21::173e:2e70 AAAA 2600:1406:3a00:21::173e:2e71 AAAA 2600:1406:3a00:21::173e:2e72 AAAA 2600:1406:3a00:21::173e:2e73 AAAA 2600:1406:3a00:21::173e:2e74 AAAA 2600:1406:3a00:21::173e:2e75 AAAA 2600:1406:3a00:21::173e:2e76 AAAA 2600:1406:3a00:21::173e:2e77 AAAA 2600:1406:3a00:21::173e:2e78 AAAA 2600:1406:3a00:21::173e:2e79 AAAA 2600:1406:3a00:21::173e:2e7a AAAA 2600:1406:3a00:21::173e:2e7b AAAA 2600:1406:3a00:21::173e:2e7c AAAA 2600:1406:3a00:21::173e:2e7d AAAA 2600:1406:3a00:21::173e:2e7e AAAA 2600:1406:3a00:21::173e:2e7f AAAA 2600:1406:3a00:21::173e:2e80 AAAA 2600:1406:3a00:21::173e:2e81 AAAA 2600:1406:3a00:21::173e:2e82 AAAA 2600:1406:3a00:21::173e:2e83 AAAA 2600:1406:3a00:21::173e:2e84 AAAA 2600:1406:3a00:21::173e:2e85 AAAA 2600:1406:3a00:21::173e:2e86 AAAA 2600:1406:3a00:21::173e:2e87 AAAA 2600:1406:3a00:21::173e:2e88 AAAA 2600:1406:3a00:21::173e:2e89 AAAA 2600:1406:3a00:21::173e:2e8a AAAA 2600:1406:3a00:21::173e:2e8b AAAA 2600:1406:3a00:21::173e:2e8c AAAA 2600:1406:3a00:21::173e:2e8d AAAA 2600:1406:3a00:21::173e:2e8e AAAA 2600:1406:3a00:21::173e:2e8f AAAA 2600:1406:3a00:21::173e:2e90 AAAA 2600:1406:3a00:21::173e:2e91 AAAA 2600:1406:3a00:21::173e:2e92 AAAA 2600:1406:3a00:21::173e:2e93 AAAA 2600:1406:3a00:21::173e:2e94 AAAA 2600:1406:3a00:21::173e:2e95 AAAA 2600:1406:3a00:21::173e:2e96 AAAA 2600:1406:3a00:21::173e:2e97 AAAA 2600:1406:3a00:21::173e:2e98 AAAA 2600:1406:3a00:21::173e:2e99 AAAA 2600:1406:3a00:21::173e:2e9a AAAA 2600:1406:3a00:21::173e:2e9b AAAA 2600:1406:3a00:21::173e:2e9c AAAA 2600:1406:3a00:21::173e:2e9d AAAA 2600:1406:3a00:21::173e:2e9e AAAA 2600:1406:3a00:21::173e:2e9f AAAA 2600:1406:3a00:21::173e:2ea0 AAAA 2600:1406:3a00:21::173e:2ea1 AAAA 2600:1406:3a00:21::173e:2ea2 AAAA 2600:1406:3a00:21::173e:2ea3 AAAA 2600:1406:3a00:21::173e:2ea4 AAAA 2600:1406:3a00:21::173e:2ea5 AAAA 2600:1406:3a00:21::173e:2ea6 AAAA 2600:1406:3a00:21::173e:2ea7 AAAA 2600:1406:3a00:21::173e:2ea8 AAAA 2600:1406:3a00:21::173e:2ea9 AAAA 2600:1406:3a00:21::173e:2eaa AAAA 2600:1406:3a00:21::173e:2eab AAAA 2600:1406:3a00:21::173e:2eac AAAA 2600:1406:3a00:21::173e:2ead AAAA 2600:1406:3a00:21::173e:2eae AAAA 2600:1406:3a00:21::173e:2eaf AAAA 2600:1406:3a00:21::173e:2eb0 AAAA 2600:1406:3a00:21::173e:2eb1 AAAA 2600:1406:3a00:21::173e:2eb2 AAAA 2600:1406:3a00:21::173e:2eb3 AAAA 2600:1406:3a00:21::173e:2eb4 AAAA 2600:1406:3a00:21::173e:2eb5 AAAA 2600:1406:3a00:21::173e:2eb6 AAAA 2600:1406:3a00:21::173e:2eb7 AAAA 2600:1406:3a00:21::173e:2eb8 AAAA 2600:1406:3a00:21::173e:2eb9 AAAA 2600:1406:3a00:21::173e:2eba AAAA 2600:1406:3a00:21::173e:2ebb AAAA 2600:1406:3a00:21::173e:2ebc AAAA 2600:1406:3a00:21::173e:2ebd AAAA 2600:1406:3a00:21::173e:2ebe AAAA 2600:1406:3a00:21::173e:2ebf AAAA 2600:1406:3a00:21::173e:2ec0 AAAA 2600:1406:3a00:21::173e:2ec1 AAAA 2600:1406:3a00:21::173e:2ec2 AAAA 2600:1406:3a00:21::173e:2ec3 AAAA 2600:1406:3a00:21::173e:2ec4 AAAA 2600:1406:3a00:21::173e:2ec5 AAAA 2600:1406:3a00:21::173e:2ec6 AAAA 2600:1406:3a00:21::173e:2ec7 AAAA 2600:1406:3a00:21::173e:2ec8 AAAA 2600:1406:3a00:21::173e:2ec9 AAAA 2600:1406:3a00:21::173e:2eca AAAA 2600:1406:3a00:21::173e:2ecb AAAA 2600:1406:3a00:21::173e:2ecc AAAA 2600:1406:3a00:21::173e:2ecd AAAA 2600:1406:3a00:21::173e:2ece AAAA 2600:1406:3a00:21::173e:2ecf AAAA 2600:1406:3a00:21::173e:2ed0 AAAA 2600:1406:3a00:21::173e:2ed1 AAAA 2600:1406:3a00:21::173e:2ed2 AAAA 2600:1406:3a00:21::173e:2ed3 AAAA 2600:1406:3a00:21::173e:2ed4 AAAA 2600:1406:3a00:21::173e:2ed5 AAAA 2600:1406:3a00:21::173e:2ed6 AAAA 2600:1406:3a00:21::173e:2ed7 AAAA 2600:1406:3a00:21::173e:2ed8 AAAA 2600:1406:3a00:21::173e:2ed9 AAAA 2600:1406:3a00:21::173e:2eda AAAA 2600:1406:3a00:21::173e:2edb AAAA 260

These are icmpv6 packets.

[illegible]

These are arp packets.

The screenshot displays the Wireshark network protocol analyzer interface. The top menu bar includes File, Machine, View, Input, Devices, and Help. Below the menu is a toolbar with various icons for file operations, packet navigation, and analysis. The main window is divided into four panes: Packet List, Packet Details, Packet Bytes, and Packet Info. The Packet List pane shows a list of 14 captured packets, with the selected packet being an ARP request from 192.168.30.8 to 192.168.30.8. The Packet Details pane shows the structure of the selected packet, including Ethernet II, Internet Protocol Version 4, and ARP. The Packet Bytes pane shows the raw data of the selected packet in hexadecimal and ASCII. The Packet Info pane shows the summary of the selected packet.

No.	Time	Source	Destination	Protocol	Length	Info
14	0.975526873	ce:2e:2b:18:2f:d5	Broadcast	ARP	60	Who has 192.168.30.8? Tell 192.168.30.8
14.03.552113159	ce:2e:2b:18:2f:d5	Broadcast	ARP	60	Who has 192.168.30.8? Tell 192.168.30.8	
15	0.99568426	ce:2e:2b:18:2f:d5	Broadcast	ARP	60	Who has 192.168.30.8? Tell 192.168.30.8
20	100.953601609	PCSSystemtec-14:9a...	ce:2e:2b:18:2f:d5	ARP	42	Who has 192.168.30.8? Tell 192.168.30.240
21	100.967942385	PCSSystemtec-14:9a...	Broadcast	ARP	60	192.168.30.8 is at ce:2e:2b:18:2f:d5
22	101.408306196	ce:2e:2b:18:2f:d5	Broadcast	ARP	60	Who has 192.168.30.240? Tell 192.168.30.8
23	101.408326955	PCSSystemtec-14:9a...	ce:2e:2b:18:2f:d5	ARP	42	192.168.30.240 is at 08:00:27:14:9a:13
24	123.45452674	ce:2e:2b:18:2f:d5	Broadcast	ARP	60	Who has 192.168.30.240? Tell 192.168.30.8
29	123.454548271	PCSSystemtec-14:9a...	ce:2e:2b:18:2f:d5	ARP	42	192.168.30.240 is at 08:00:27:14:9a:13
34	142.191579654	ce:2e:2b:18:2f:d5	Broadcast	ARP	60	Who has 192.168.30.240? Tell 192.168.30.8
35	142.191602288	PCSSystemtec-14:9a...	ce:2e:2b:18:2f:d5	ARP	42	192.168.30.240 is at 08:00:27:14:9a:13
36	147.032283477	PCSSystemtec-14:9a...	ce:2e:2b:18:2f:d5	ARP	42	Who has 192.168.30.8? Tell 192.168.30.240
37	147.043881149	ce:2e:2b:18:2f:d5	PCSSystemtec-14:9a...	ARP	60	192.168.30.8 is at ce:2e:2b:18:2f:d5

6). Identify at least 3 different protocols in the capture.

1. ICMP – Internet Control Message Protocol

- Purpose:
Used primarily for network diagnostics and error reporting.
- Example:
Executing `ping google.com` sends ICMP Echo Request packets and receives Echo Reply packets in return.
- Use Case:
 - Verifies if a host or device is reachable on the network.
 - Measures round-trip time (RTT) to diagnose latency or packet loss issues.

2. TCP – Transmission Control Protocol

- Purpose:
Provides reliable, ordered, and error-checked delivery of data between applications.
- Example:
Protocols such as HTTP, HTTPS, FTP, and SSH use TCP to ensure that data is transmitted correctly.
- Use Case:
 - Initiates a 3-way handshake (SYN → SYN-ACK → ACK) to establish a connection.
 - Guarantees that all data is delivered without loss or duplication, and in the correct order.

3. DNS – Domain Name System

Purpose:
Translates human-readable domain names into machine-readable IP addresses .

Example:

When accessing , your device sends a DNS query to resolve the domain name to its corresponding IP address.

Use Case:

Acts as the first step in any web browsing session.

Without DNS resolution, the browser cannot locate the server hosting the website.

7)Export the capture as a .pcap file.

I have completely exported the .pcap files.

8)Summarize your findings and packet details.

1. ICMP – Internet Control Message Protocol

- Purpose:
Used for network diagnostics and error reporting.
- Typical Usage:
Commonly utilized in tools like **ping** to check the reachability of a host and to measure latency (round-trip time).
- Key Functions:
 - Sends Echo Requests and receives Echo Replies.
 - Helps troubleshoot network issues like unreachable hosts or packet loss.

2. TCP – Transmission Control Protocol

- Purpose:
A connection-oriented protocol designed to ensure reliable, ordered, and error-checked delivery of data.
- Typical Usage:
Forms the backbone of major internet applications such as:
 - Web browsing: HTTP/HTTPS

- File transfer: FTP
 - Email: SMTP/IMAP/POP3
 - Key Features:
 - Establishes a reliable connection via the 3-way handshake process (SYN → SYN-ACK → ACK).
 - Guarantees that data arrives complete and in the correct sequence.
-

3. DNS – Domain Name System

- Purpose:
Resolves human-readable domain names into machine-usable IP addresses.
- Typical Usage:
DNS queries are essential before any website, application, or cloud service can be accessed.
- Key Details:
 - Operates over UDP (default) or TCP (for larger queries), using port 53.
 - Enables seamless web browsing and service accessibility by mapping domain names to IPs.