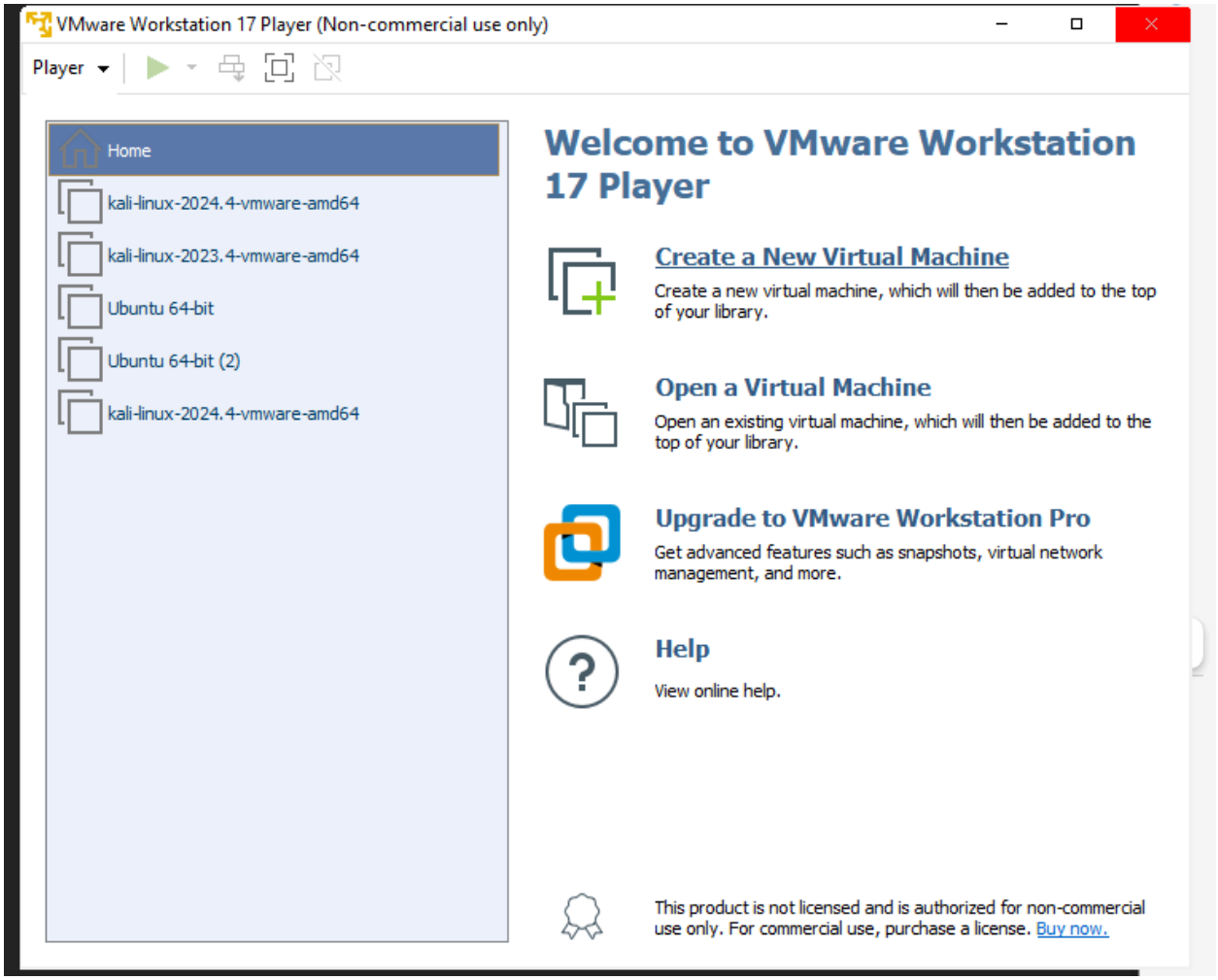
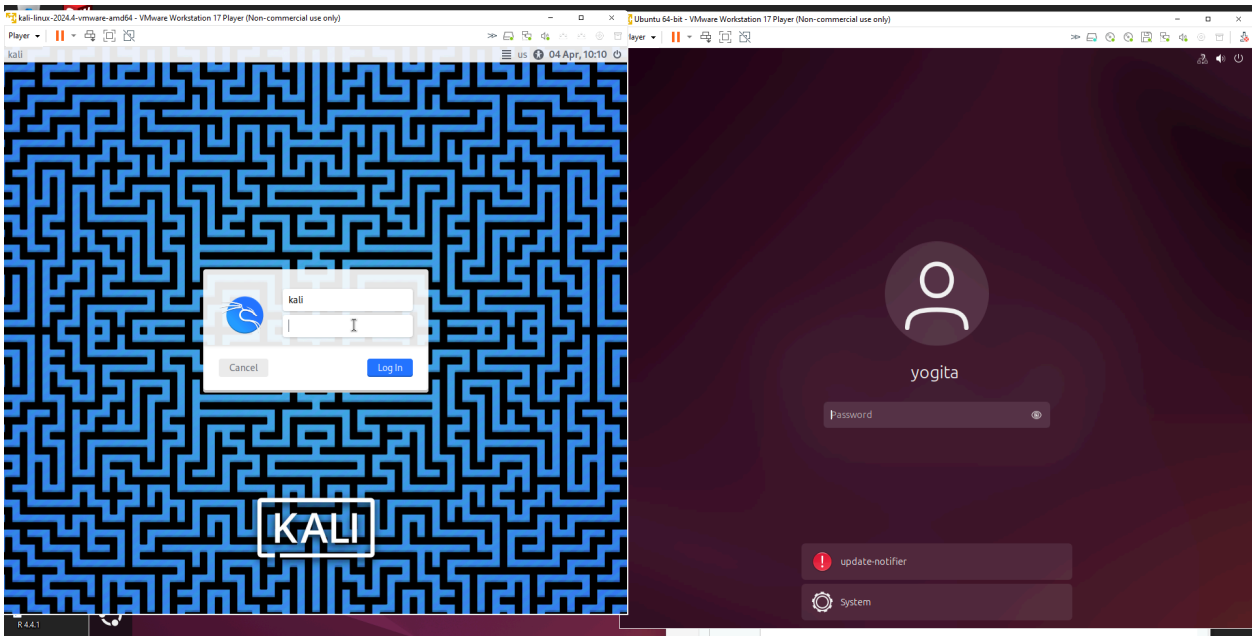
# Assignment 9

Name: Yogita Mundankar
Rollno. B22CS068

For both the TASKS (Common steps):

1) Created two VMs and configured them to NAT :






I have used NAT as:

- In **NAT mode**, VirtualBox/VMware creates an internal network where the two VMs can communicate.
- **NAT allows the VMs to access the internet** through the host machine. This is important for installing tools (Ettercap, Bettercap, Arpspoof) and testing network traffic interception.
- This allows you to **test ARP spoofing attacks inside a safe environment** without external interference.

Ubuntu(Victim):
Kali Linux (Attacker):

2) Installed the required tools on the attacker VM i.e Kali LINUX: Ettercap, Bettercap, Arpspoof

```
┌──(kali㉿kali)-[/home/kali]
└─PS> ettercap --version

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

ettercap 0.8.3.1

┌──(kali㉿kali)-[/home/kali]
└─PS> bettercap --version
bettercap v2.33.0 (built for linux amd64 with go1.22.6)
```

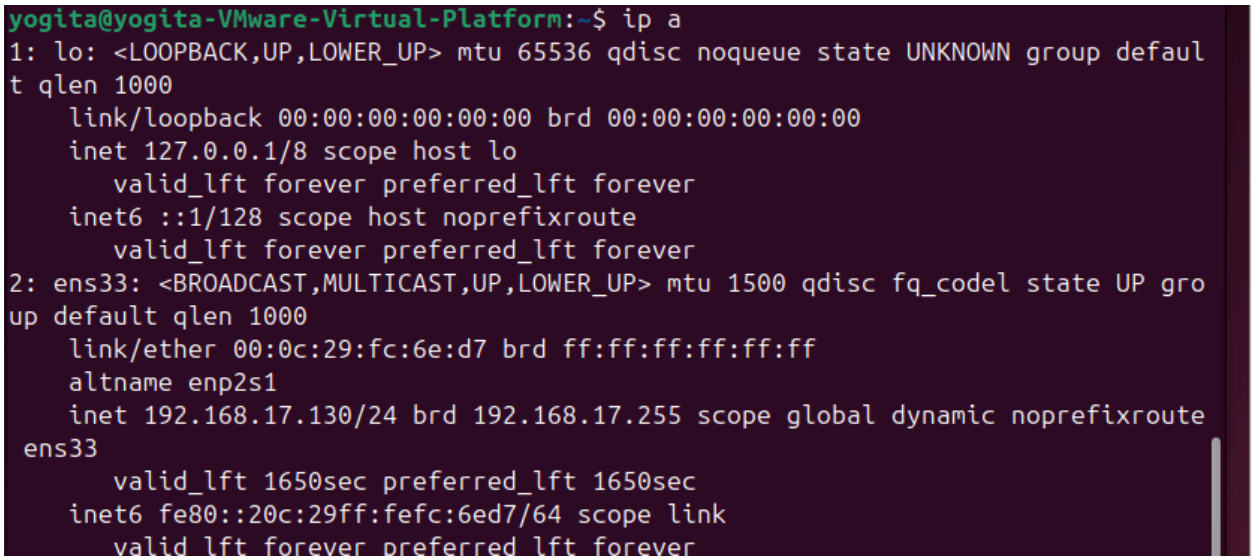
```
┌──(kali㉿kali)-[/home/kali]
└─PS> sudo arpspoof --version
[sudo] password for kali:
arpspoof: invalid option -- '-'
Version: 2.4
Usage: arpspoof [-i interface] [-c own|host|both] [-t target] [-r] host

┌──(kali㉿kali)-[/home/kali]
└─PS> wireshark --version
Wireshark 4.4.5.
```

3) Got the ip addresses of the two VM's and the ip address of the gateway :

Kali Linux (Attacker):

```
┌──(kali㉿kali)-[/home/kali]
└─PS> ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:0e:cc:c2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.17.129/24 brd 192.168.17.255 scope global dynamic noprefixroute eth0
       valid_lft 1566sec preferred_lft 1566sec
    inet6 fe80::34e8:c24:4fd2:94e3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Ubuntu (Victim):

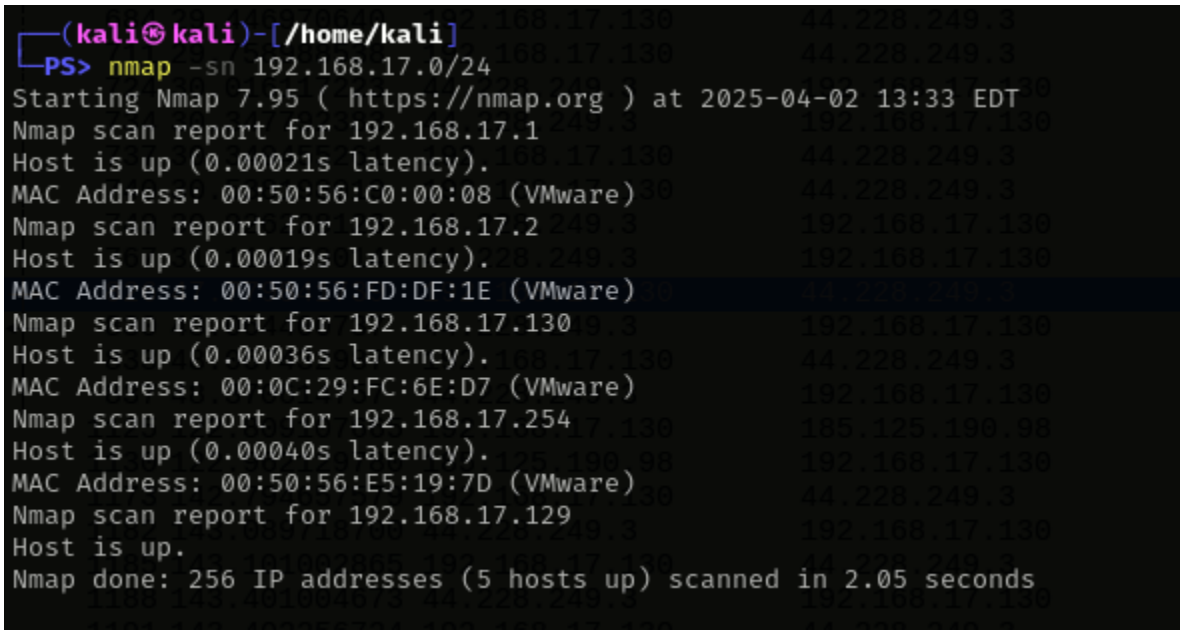



IP Address of gateway on Attacker:


IP Address of:
Attacker: 192.168.17.129
Victim: 192.168.17.130
Gateway: 192.168.17.2
The attacker and victim belong to the same subnet: 192.168.17.0/24

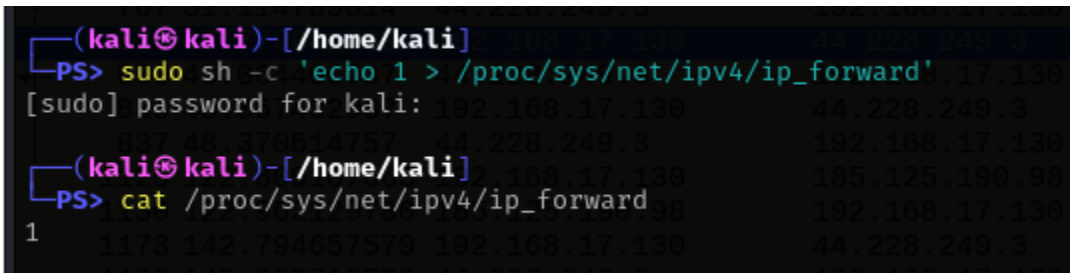
# TASK 1:

Performed nmap Scan on Attacker:




- This gives us only the devices that are online but not the services running and the open ports.
- All hosts have **VMware MAC addresses**, indicating they are virtual machines running in a VMware environment.

4) Enable IP Forwarding (Packet Routing):



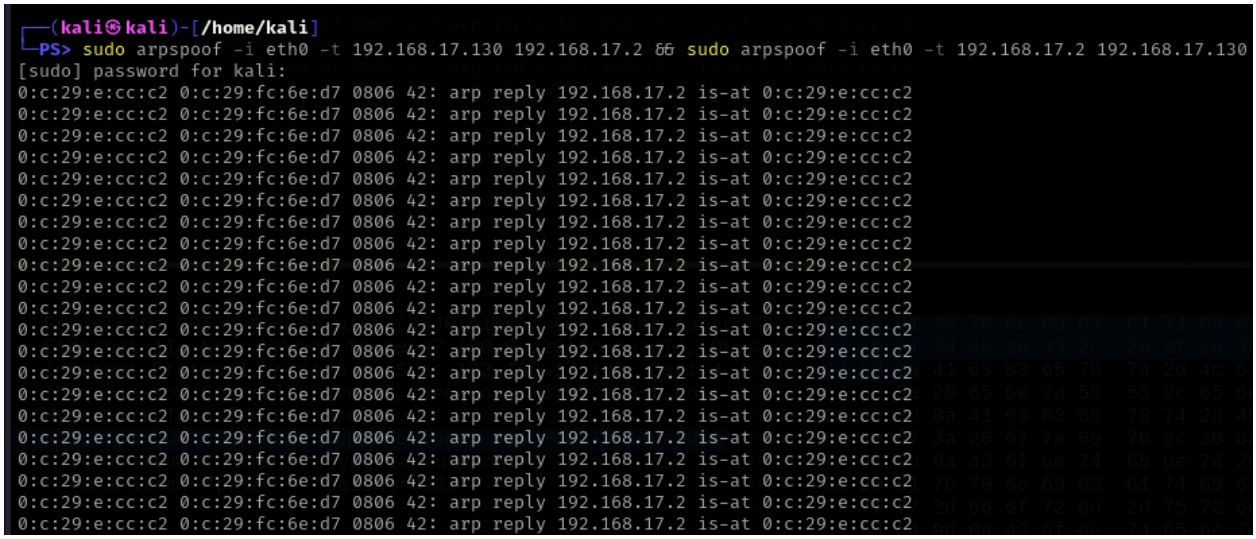

echo 1 > /proc/sys/net/ipv4/ip_forward →

- Sets the value to 1, enabling IP forwarding.
- sudo sh -c → Runs the command with root privileges.

cat /proc/sys/net/ipv4/ip_forward

- This checks if **IP forwarding is enabled**.
- If it prints 1, then **packets will be forwarded between devices**.

5) Sending spoofed ARP messages:




I have written two commands here:

```
sudo arpspoof -i eth0 -t 192.168.17.130 192.168.17.2
```

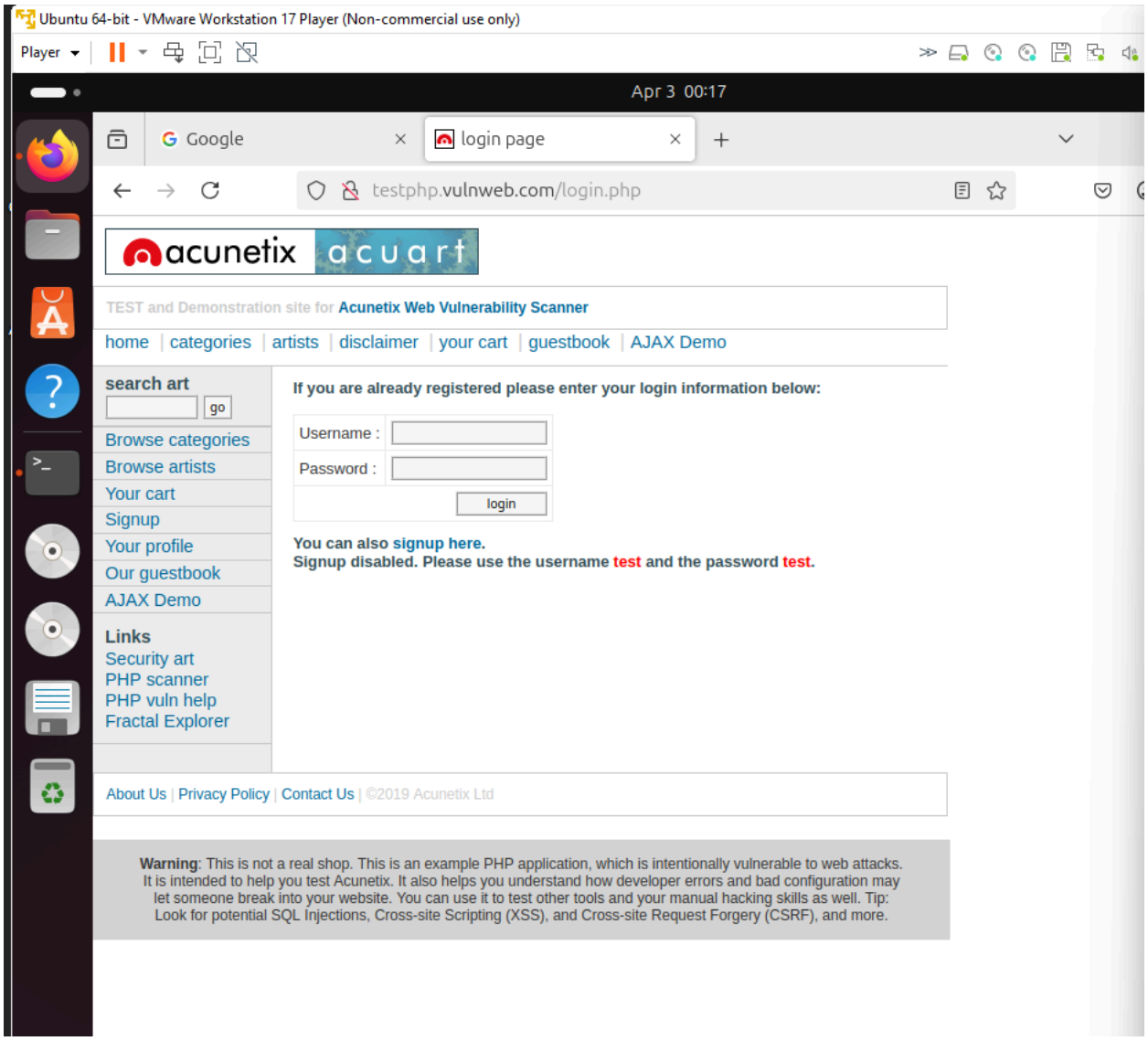
- i eth0 → Uses network interface eth0 for the attack.
- -t 192.168.17.130 → Targets victim 192.168.17.130.
- 192.168.17.2 → Fakes being 192.168.17.2 (gateway).
- This targets the victim and makes it believe that the mac ID of the attacker has the ip address of the gateway.

```
sudo arpspoof -i eth0 -t 192.168.17.2 192.168.17.130
```

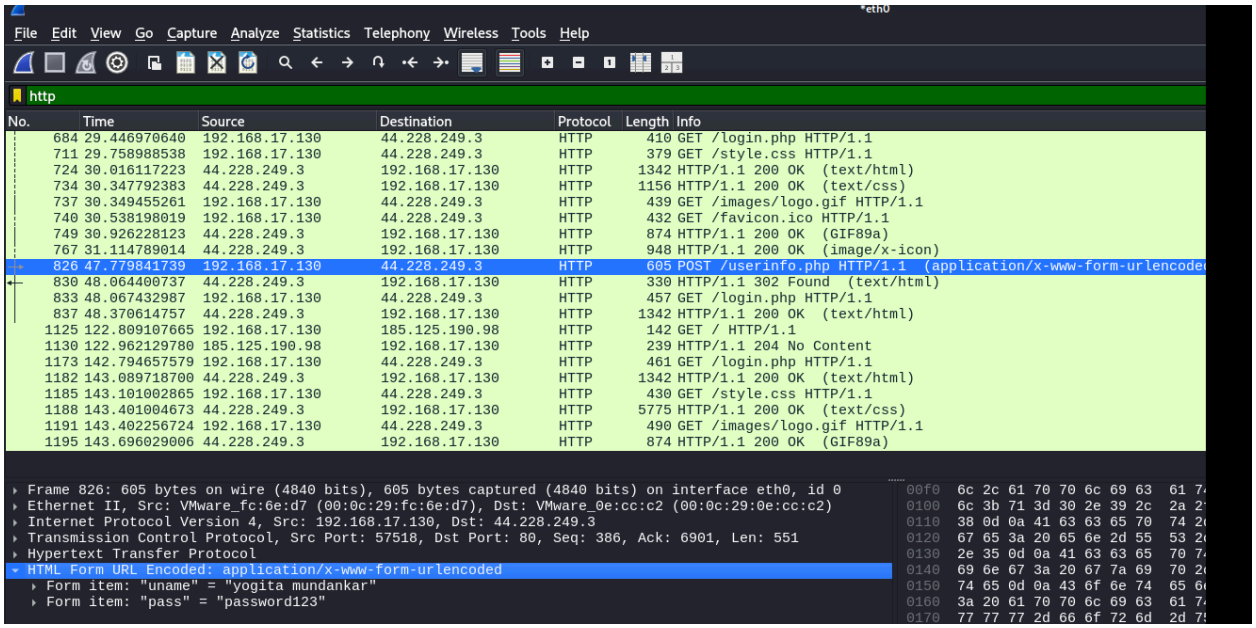
- Targets **the router (192.168.17.2)** and makes it believe **your Kali machine** is **192.168.17.130**.

## 6) Using the Victim VM for sending doing tasks involving http , ftp traffic:

Filled the form on: http://testphp.vulnweb.com/login.php



## 7) Capturing packets on the Attacker VM using wireshark:




- **Observations:**
  The form fields can been seen in plain text:
  Uname: “yogita mundankar”
  Pass = “password123”
  **Thus HTTP is not secure as intercepted network packets reveal the sensitive credentials.**

- **Analysis:**
  The attacker used `arpspoof` to launch an ARP poisoning attack, misleading both the victim and the gateway into routing their packets through the attacker's machine.

- After enabling IP forwarding, a transparent man-in-the-middle (MITM) setup was established.

- From the victim machine (Ubuntu), a simple HTTP form was submitted to a web server.

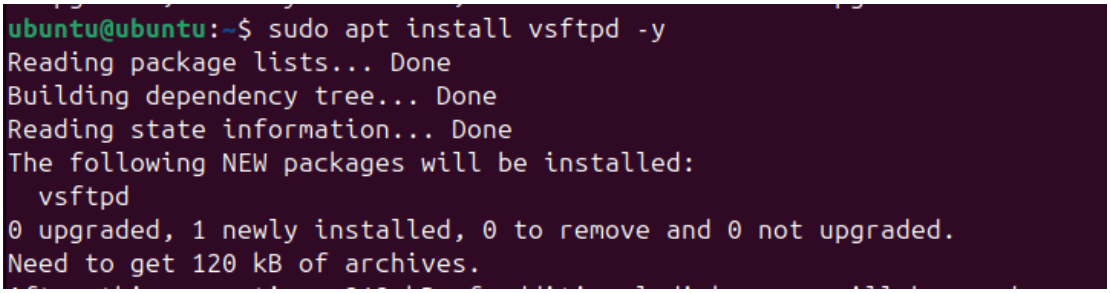
- The attacker, monitoring traffic using Wireshark, successfully intercepted the HTTP request.

- Form data sent over the network was captured in plaintext, including the contents of input fields from the victim's submission.

- This confirmed that any unencrypted web traffic (HTTP) is highly vulnerable to interception when a MITM attacker is present on the network.

**FTP Traffic spying:**

- First, I have created another VM machine(VM3-Ubuntu) to host the locally created ftp server.
  Installing ftp:



Created a user:

```
ubuntu@ubuntu:~$ sudo nano /etc/vsftpd.conf
ubuntu@ubuntu:~$ sudo systemctl restart vsftpd
ubuntu@ubuntu:~$ sudo systemctl enable vsftpd
Synchronizing state of vsftpd.service with SysV service script with /usr/lib/systemd/systemd-s
ysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable vsftpd
ubuntu@ubuntu:~$ sudo useradd -m ftpuser
ubuntu@ubuntu:~$ sudo passwd ftuser
passwd: user 'ftuser' does not exist
ubuntu@ubuntu:~$ sudo passwd ftpuser
New password:
Retype new password:
passwd: password updated successfully
ubuntu@ubuntu:~$ sudo ufw allow 21/tcp
Rules updated
Rules updated (v6)
ubuntu@ubuntu:~$ sudo systemctl restart vsftpd
ubuntu@ubuntu:~$ ip a
```

IP Address of VM3-Ubuntu: 192.168.17.131

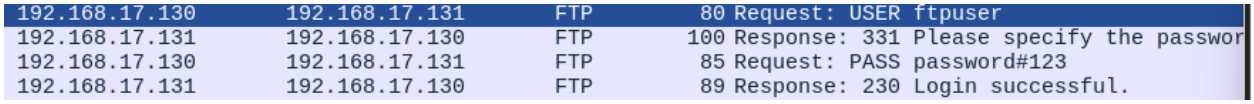
- Now I am access the ftp server and logging in from the Victim VM:

```
yogita@yogita-VMware-Virtual-Platform:~$ ftp 192.168.17.131
Connected to 192.168.17.131.
220 (vsFTPd 3.0.5)
Name (192.168.17.131:yogita): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

- Captured  ftp traffic using wireshark on Attacker VM:

ftp

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 19 | 20.599077889 | 192.168.17.131 | 192.168.17.130 | FTP | 86 | Response: 220 (vsFTPd 3.0.5) |
| 68 | 87.737376717 | 192.168.17.130 | 192.168.17.131 | FTP | 80 | Request: USER ftpuser |
| 70 | 87.743333364 | 192.168.17.131 | 192.168.17.130 | FTP | 100 | Response: 331 Please specify the passwor |
| 80 | 94.313761256 | 192.168.17.130 | 192.168.17.131 | FTP | 85 | Request: PASS password#123 |
| 82 | 94.776056835 | 192.168.17.131 | 192.168.17.130 | FTP | 89 | Response: 230 Login successful. |
| 84 | 94.776335325 | 192.168.17.130 | 192.168.17.131 | FTP | 72 | Request: SYST |
| 86 | 94.776435875 | 192.168.17.131 | 192.168.17.130 | FTP | 85 | Response: 215 UNIX Type: L8 |
| 87 | 94.777527035 | 192.168.17.130 | 192.168.17.131 | FTP | 72 | Request: FEAT |
| 88 | 94.777604366 | 192.168.17.131 | 192.168.17.130 | FTP | 81 | Response: 211-Features: |
| 89 | 94.777604445 | 192.168.17.131 | 192.168.17.130 | FTP | 73 | Response:  EPRT |
| 90 | 94.777604468 | 192.168.17.131 | 192.168.17.130 | FTP | 73 | Response:  EPSV |
| 91 | 94.777692455 | 192.168.17.131 | 192.168.17.130 | FTP | 73 | Response:  MDTM |
| 93 | 94.777692563 | 192.168.17.131 | 192.168.17.130 | FTP | 73 | Response:  PASV |
| 95 | 94.778168868 | 192.168.17.131 | 192.168.17.130 | FTP | 80 | Response:  REST STREAM |
| 96 | 94.778168964 | 192.168.17.131 | 192.168.17.130 | FTP | 73 | Response:  SIZE |
| 97 | 94.778168989 | 192.168.17.131 | 192.168.17.130 | FTP | 73 | Response:  TVFS |
| 99 | 94.778326652 | 192.168.17.131 | 192.168.17.130 | FTP | 75 | Response: 211 End |




**Observations from the Wireshark Capture:**

- The communication is between two IP addresses:
  - **192.168.17.130** (Client)
  - **192.168.17.131** (FTP Server)
- The FTP authentication process is visible in plaintext, including:
  - **Username**: `ftpuser`
  - **Password**: `password#123`
- The server responds with:
  - `331 Please specify the password` → The FTP server requests a password after the username.
  - `230 Login successful.` → The authentication is successful.

## Conclusion:

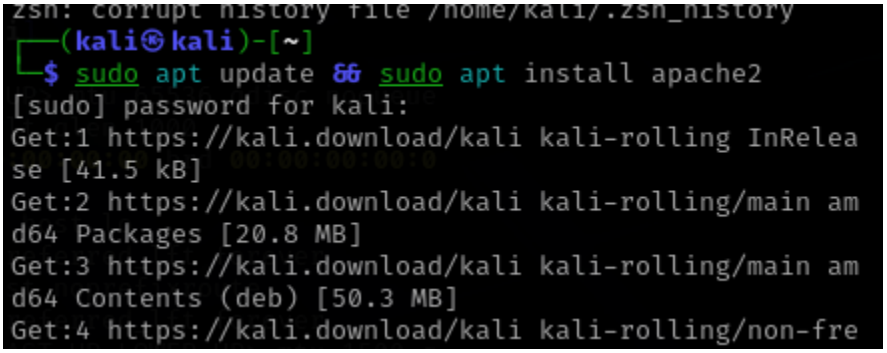
**Plaintext Credentials Exposure**: FTP does not encrypt login details, making it highly vulnerable to **Man-in-the-Middle (MITM) attacks**.

# TASK 2:

I have used **ARPSPOOF+ETTERCAP** for DNS Spoofing.

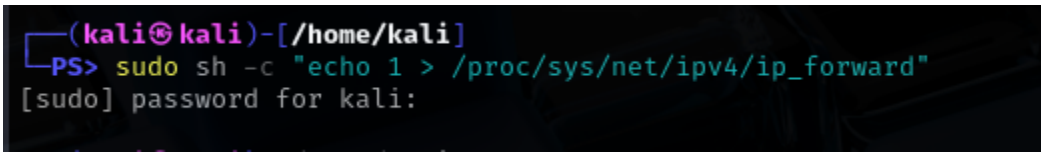
1) Created a fake website that the victim should be directed to when the victim searches for google.com

   - Install Apache:

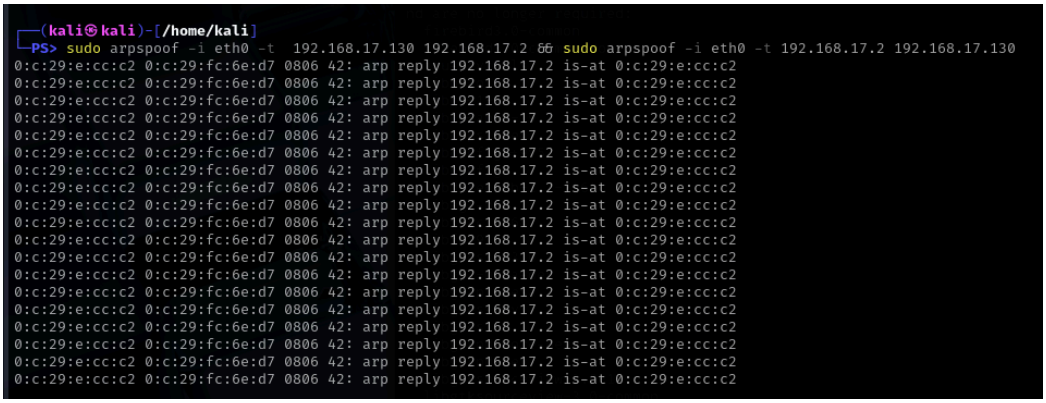



   - Create fake website:


2) Enable IP Forwarding on Attacker VM (Kali):

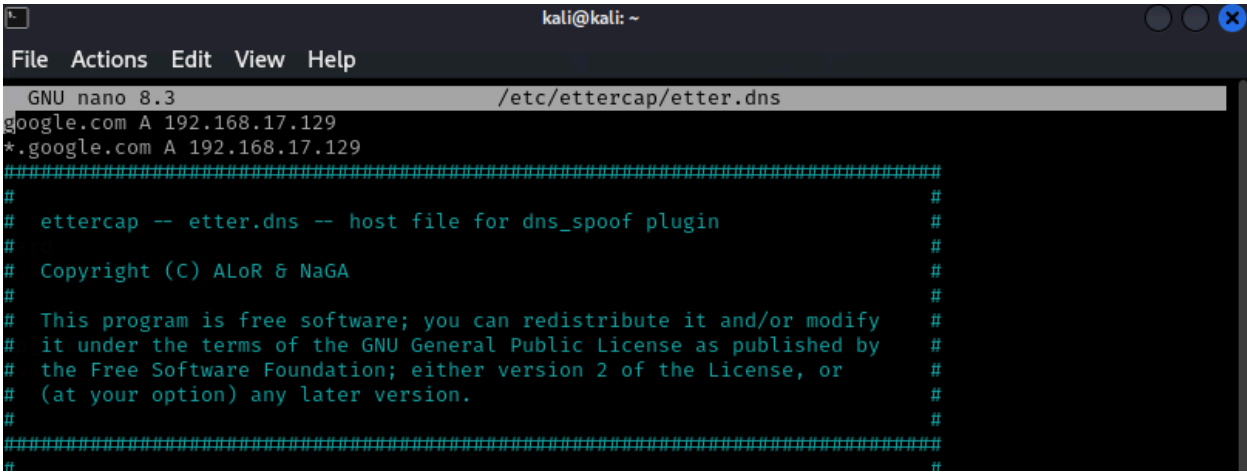



3) Launch ARP Spoofing:



4) Configure ettercap for DNS Spoofing:

```
┌──(kali㉿kali)-[~]
└─$ sudo nano /etc/ettercap/etter.dns
```

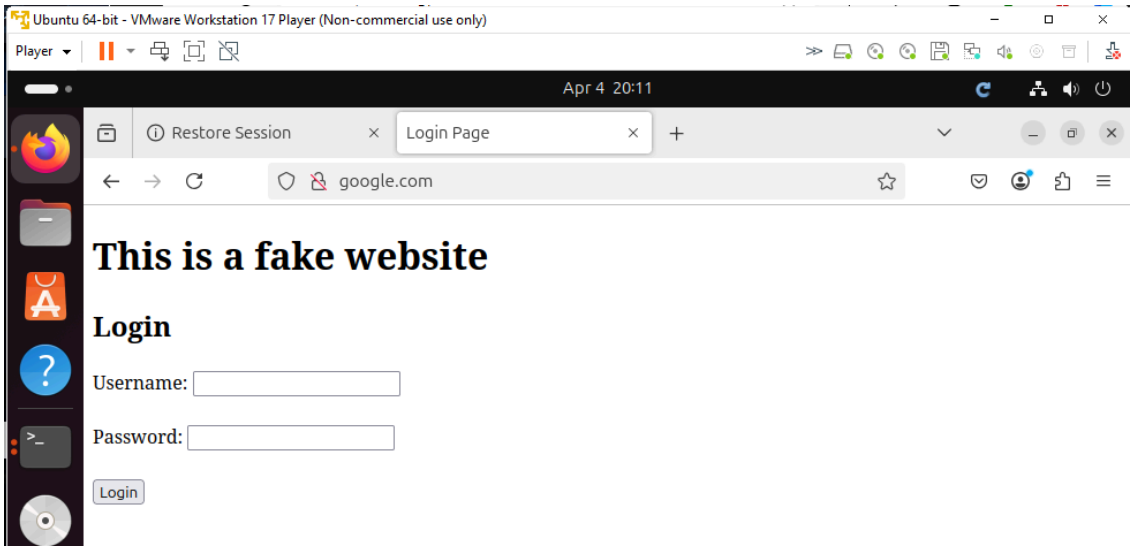


5) Run Ettercap in DNS Spoofing Mode:

6) Test the Attack:

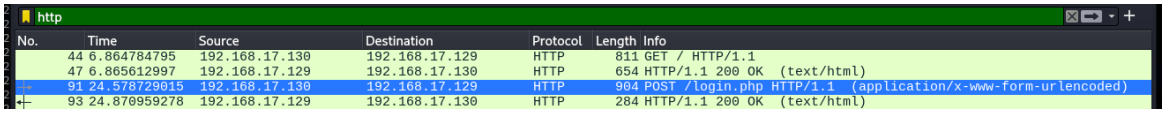
On the victim machine, open a browser and go to:google.com.

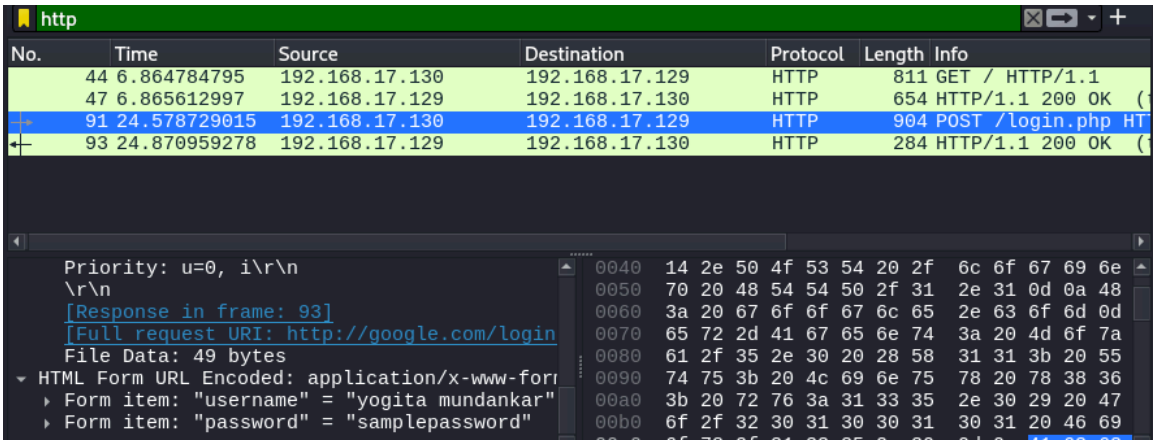
It is redirected to the fake website created on the attacker VM.




7) Capture Packets using Wireshark on Attacker VM:

HTTP Packets captured on attack via wireshark:




The form items can easily be seen : username : yogita munadnkar

Password: samplepassword

# Summary and Analysis for Task 2:

- A **fake website** was hosted on the attacker VM using Apache.
- DNS spoofing was configured by modifying `etter.dns` and running `ettercap`.
- When the victim typed a domain like `google.com`, the attacker responded with their own IP, redirecting the victim to the **fake site**.
- The DNS poisoning was successful and seamless; the victim did not notice the redirection.
- The attacker's Apache log showed the victim's browser requesting the fake page, proving the redirection worked as planned.

# Conclusion:

The lab successfully demonstrated multiple network-layer attacks across a **three-VM virtual environment**, simulating realistic scenarios of exploitation.

**Task 1** showed how ARP spoofing can enable full visibility into network traffic, allowing an attacker to:

- Capture **HTTP form data** submitted by a user
- Intercept **FTP login credentials** sent in plaintext

**Task 2** proved the danger of **DNS spoofing**, where a victim can be silently redirected to a malicious site without any alert or browser warning.

These attacks reinforce key cybersecurity lessons:

- Always use **HTTPS** and **SFTP/FTPS** instead of HTTP and FTP
- Implement **ARP and DNS spoofing detection tools** on networks
- Use **strong encryption**, **DNSSEC**, and **network segmentation** to reduce the attack surface

Overall, the experiment highlights how easily traffic can be manipulated or intercepted without physical access, underlining the **importance of proactive defensive strategies** in cybersecurity.