Task 3

Firewall & Network Security📡

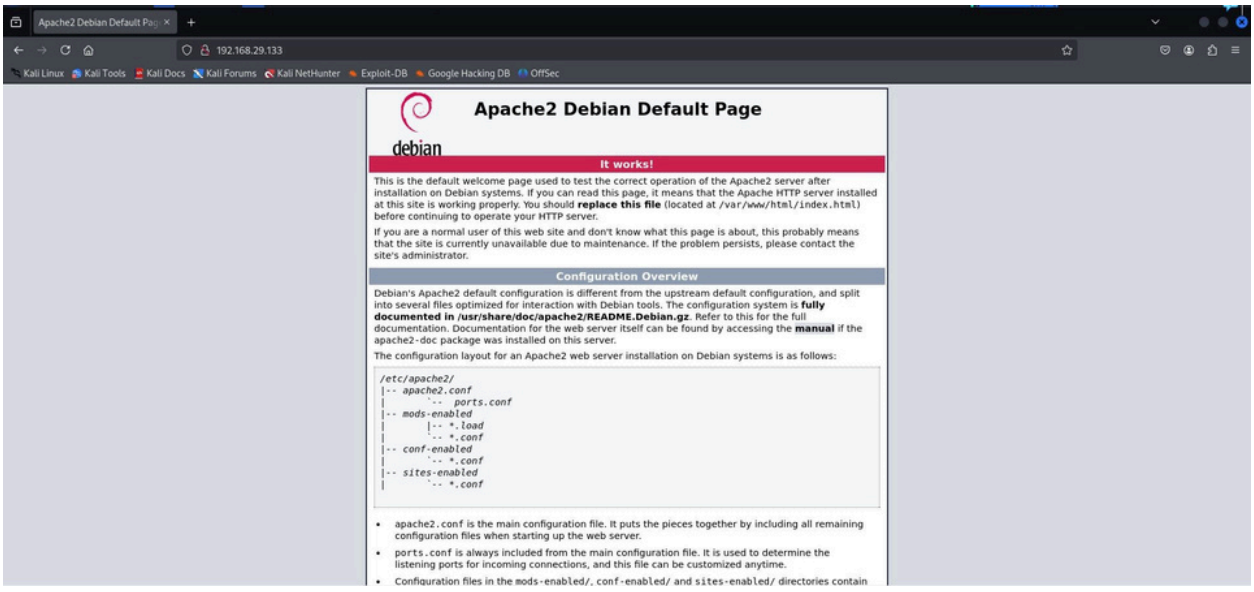
Setting up apache web server :

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl start apache2
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ sudo systemctl enable  apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
Created symlink '/etc/systemd/system/multi-user.target.wants/apache2.service' → '/usr/lib/systemd/system/apache2.service'.
```

We begin by starting and enabling the Apache2 server to ensure it is active and available for use.

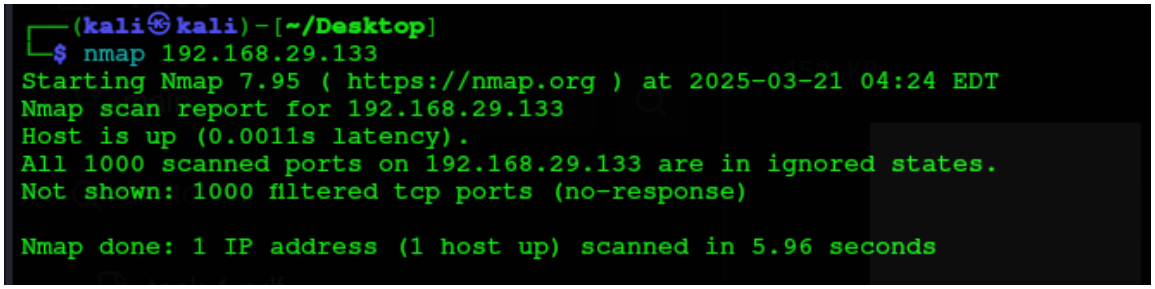
```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo ufw disable

[sudo] password for kali:
Firewall stopped and disabled on system startup
```

Then we disable firewall for testing.

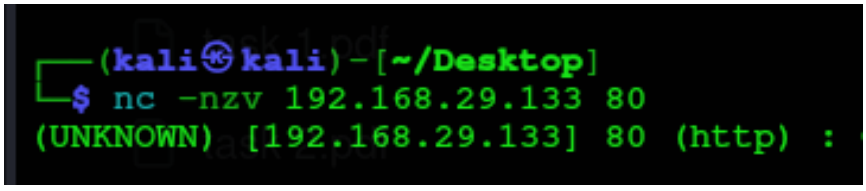


- We initiate the Apache2 server by starting and enabling it to ensure continuous operation and availability.

Exploit : port scanning with Nmap and Netcat



- First we perform a basic Nmap scan to check for open ports.



- Then we use Netcat to check if a specific port is open.
- These make them vulnerable to exploit.

Mitigation: Firewall & Network Hardening

```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo ufw enable

Firewall is active and enabled on system startup
```

First we enable firewall for active defense.

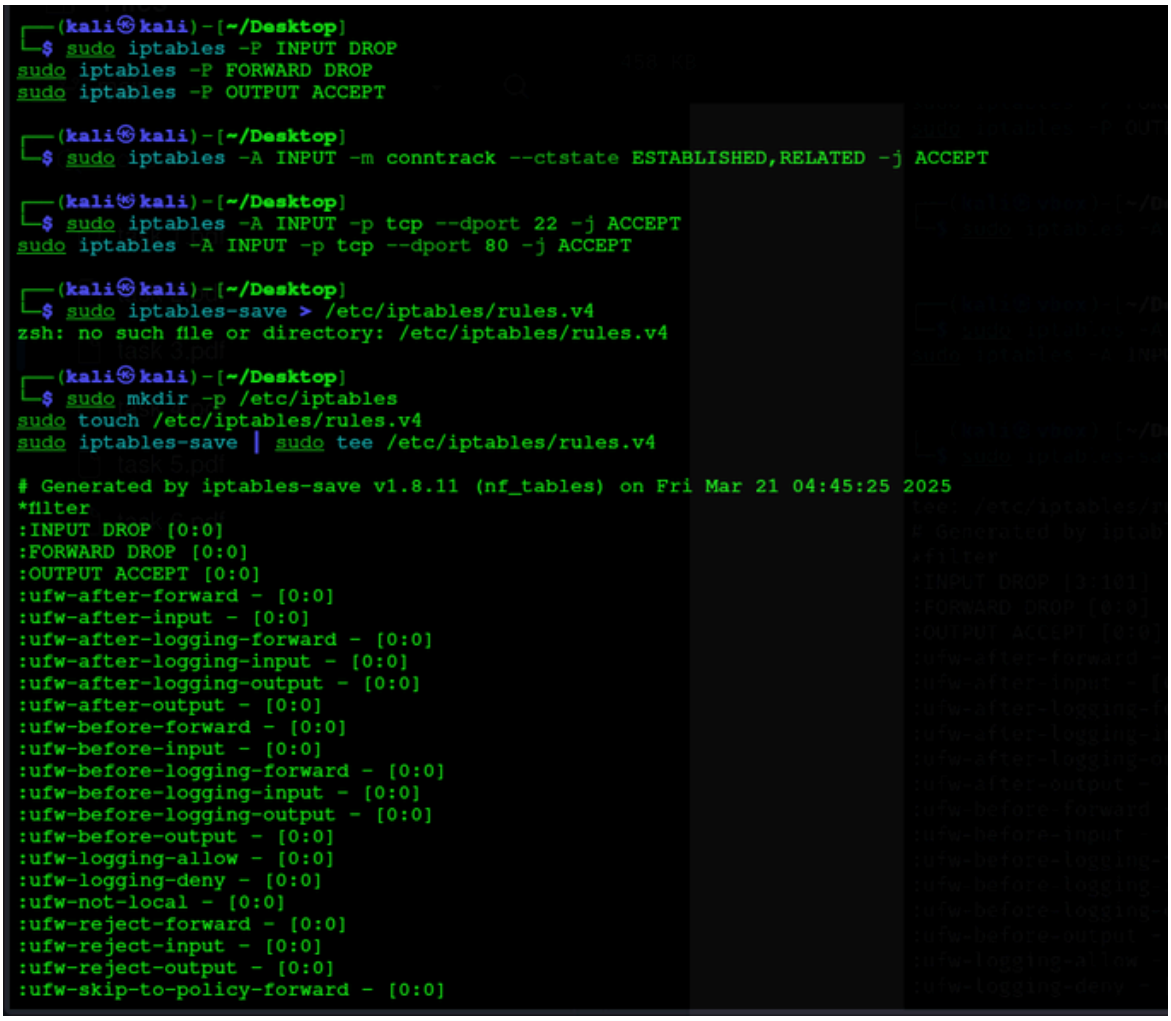
```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo ufw default deny incoming
sudo ufw default allow outgoing
sudo ufw allow ssh
sudo ufw allow http

Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
Rule added
Rule added (v6)
Rule added
Rule added (v6)
```

Then we set rules for allowing necessary ports.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ sudo ufw reload

Firewall reloaded
```

Finally we reload the firewall to save changes.

Additional Protection with Iptables🔒



To configure **iptables** for enhanced security, set the default policies to **DROP** incoming and forwarded traffic while **ACCEPTING** outgoing connections, allow **established** and **related** connections, permit incoming traffic for **SSH (port 22)** and **HTTP (port 80)**, and finally save the rules using `sudo iptables-save | sudo tee /etc/iptables/rules.v4` to ensure they persist across reboots.

🎯 Summary

✅ Installed Apache with UFW disabled (vulnerable state)

✅ Simulated an attack using `nmap` and `netcat`

✅ Hardened the server by restricting access via UFW and Iptables