Network Architecture

Component	IP Address	Role
Attacker (Arch Linux)	192.168.100.3	Scans/Exploits Server 1
Server 1 (Linux Mint)	192.168.100.5	Real service host + redirection/proxy logic
Decoy System (Linux Mint)	192.168.100.6	Clone of Server 1 with extra open ports
Server 2 (Linux Mint)	192.168.100.7	Standby server for failover response
Bots (Linux Mint)	192.168.100.8	Simulate user activity inside Decoy
Legit Users	192.168.100.9	Accesses service via Server 1 or 2

Step-by-Step Deployment Instructions

On Server 1 (192.168.56.103)

```
sudo apt update && sudo apt install apache2 openssh-server
python3 python3-pip python3-venv tcpdump -y
sudo pip install scapy
```

Create web service:

```
sudo nano /var/www/html/index.html
# Paste: Secure Login Portal - Server 1
sudo systemctl enable apache2
sudo systemctl start apache2
```

Allow necessary ports:

```
sudo ufw allow 21
sudo ufw allow 22
sudo ufw allow 80
```

Create and activate Python virtual environment:

```
python3 -m venv myenv
source myenv/bin/activate
pip install scapy paramiko
```

Give passwordless sudo for Apache on Server 2: (At New Tab)

```
ssh server2@192.168.100.7
sudo visudo
# Add this line:
server2 ALL=(ALL) NOPASSWD: /bin/systemctl start apache2
```

Copy SSH key to Server 2: (At New Tab)

```
ssh-keygen -t rsa -b 4096
ls ~/.ssh/
ssh-copy-id server2@192.168.100.7
```

Move Apache to port 8080 (so Python can own port 80)

Edit `/etc/apache2/ports.conf`:

```
sudo nano /etc/apache2/ports.conf
```

Change:

```
Listen 80
```

To:

```
Listen 8080
```

Edit default site:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Change:

```
<VirtualHost *:80>
```

To:

```
<VirtualHost *:8080>
```

Save, then:

```
sudo systemctl restart apache2
```

Test:

```
curl http://127.0.0.1:8080
```

You should see **Server1 portal HTML**.

C. Create Python venv & copy deception_proxy.py

```
cd ~/deception

python3 -m venv myenv

source myenv/bin/activate

pip install scapy
```

Save the Deception script:

```
nano deception_proxy.py #Copy past the deception script
```

Run Deception Proxy:

```
sudo myenv/bin/python3 deception_proxy.py
```

On Decoy (192.168.56.106)

Open the decoy fake ports

Run Real/Vulnerable Services on Linux Mint (Decoy)

Update System First

```
sudo apt update && sudo apt upgrade -y
```

Open Port 80: Apache Web Server (HTTP)

```
sudo apt install apache2 -y
sudo systemctl enable apache2
sudo systemctl start apache2
```

Test:

```
curl <http://localhost>
```

Open Port 22: SSH

```
sudo apt install openssh-server -y
sudo systemctl enable ssh
sudo systemctl start ssh
```

To confirm:

```
sudo systemctl status ssh
```

Open Port 21: FTP Server (vsftpd)

```
sudo apt install vsftpd -y
sudo systemctl enable vsftpd
sudo systemctl start vsftpd
```

Disable Firewall or Allow Ports

If `ufw` is enabled:

```
sudo ufw allow 21
sudo ufw allow 22
sudo ufw allow 80
sudo ufw enable
```

Confirm Open Ports

On the decoy:

```
sudo ss -tuln
```

From attacker machine:

```
nmap -sS <decoy-ip>
```

- **Host a Fake Web Service:**

```
sudo systemctl enable apache2
sudo systemctl start apache2
sudo nano /var/www/html/index.html
# Use: "Secure Login Portal - Decoy"
```

- **Exploit trigger simulation:**

Create a file that attacker will trigger:

```
sudo touch /tmp/rooted
```

You can delete it to simulate non-exploited state.

Step-by-Step: Inject Spoof Aliases Inline in `.bashrc`

Step 1: Open the `.bashrc` file

```
nano ~/.bashrc
```

Step 2: Scroll to the bottom of the file, and paste these lines:

```
# Deception aliases (victim identity masking)
alias ip='echo "inet 192.168.56.103/24 brd 192.168.56.255 scope global eth0"'
alias ifconfig='echo -e "eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>
mtu 1500\\n    inet 192.168.56.103  netmask 255.255.255.0  broadcast
192.168.56.255\\n    ether 08:00:27:de:ad:be  txqueuelen 1000 (Ethernet)"'
alias hostname='echo victim-linux'
alias uname='echo "Linux victim-linux 5.15.0-kali #1 SMP"'
alias whoami='echo root'
alias id='echo uid=0(root) gid=0(root) groups=0(root)'
```

These aliases override common fingerprinting commands.

Step 3: Apply Changes Immediately

```
source ~/.bashrc
```

Now the spoofed environment is live.

On Server 2 (192.168.56.105)

- **Host Web Service:**

```
sudo apt install apache2 -y
sudo systemctl enable apache2
sudo systemctl start apache2
sudo nano /var/www/html/index.html
# Use: "Secure Login Portal - Server 2"
```

Apache will be OFF initially. Server 1 will trigger its launch after detecting an attacker.

On Bot Machine (or Decoy itself)

```
sudo apt update
sudo apt install python3 python3-pip -y
python3 -m venv myenv
source myenv/bin/activate
pip install requests
```

Run the bot script:

```
python3 bot_decoy_simulator.py
```

On Attacker Machine (192.168.56.102)

Before Scan:

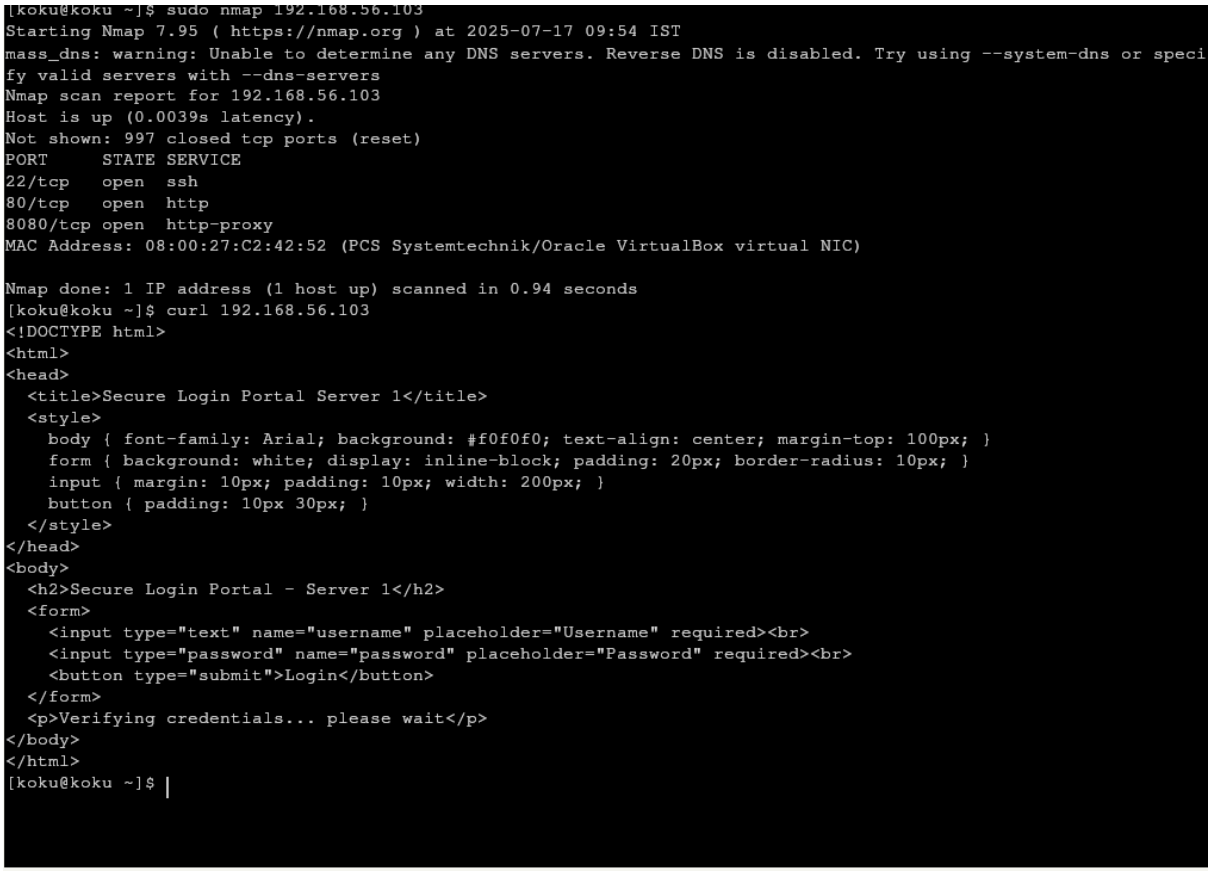
Open a web browser and visit:

http://192.168.100.5

You can access the real web service.

Or run the following command in the terminal:

curl 192.168.100.5



To simulate a SYN scan, run:

nmap -sS 192.168.100.5

After Scan:

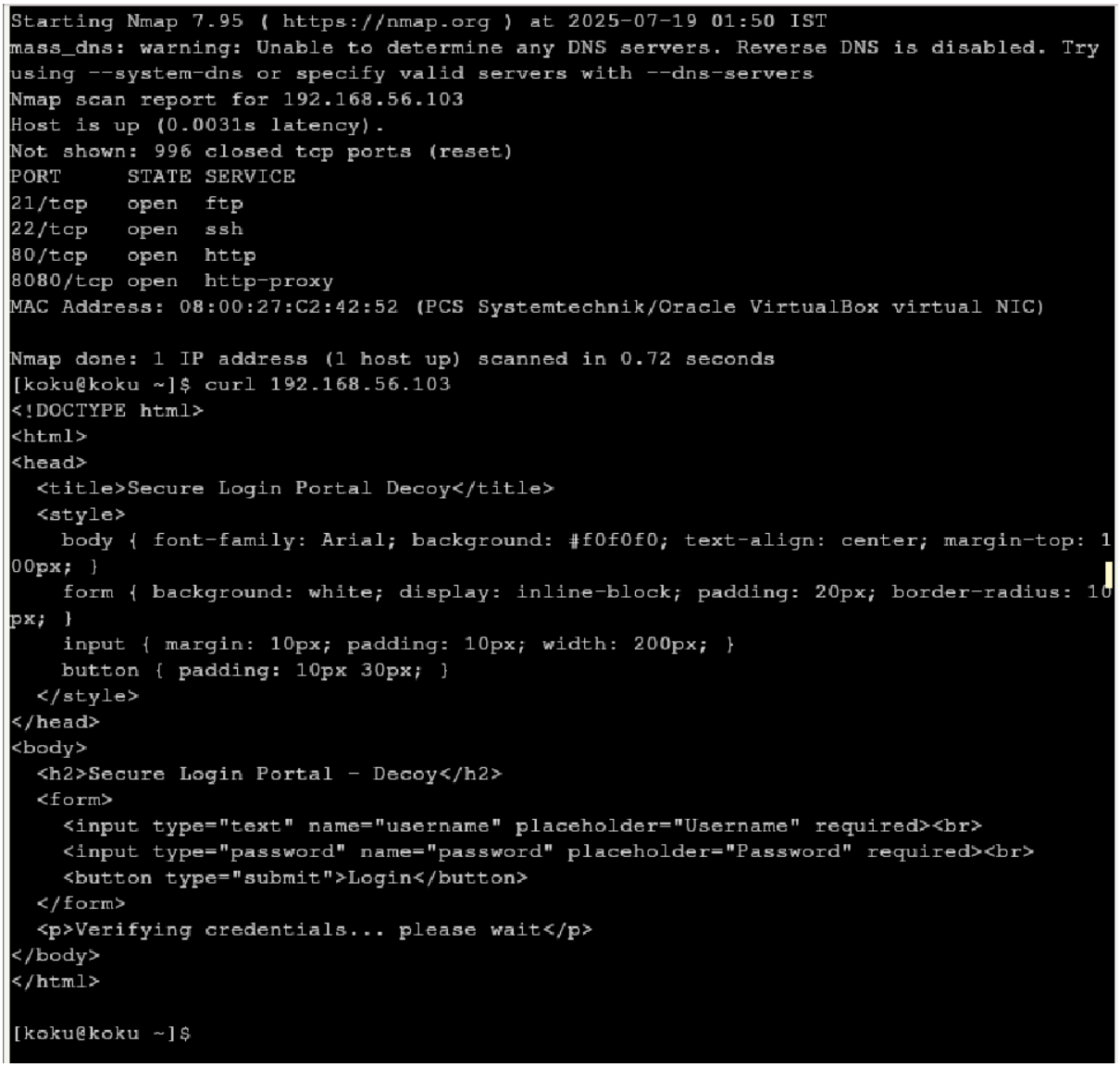
Open a web browser and visit:

http://192.168.100.5

You will now access the fake (decoy) web service.

Or run the following command in the terminal:

```
curl 192.168.100.5
```



```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-19 01:50 IST
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try
using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.56.103
Host is up (0.0031s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
8080/tcp open  http-proxy
MAC Address: 08:00:27:C2:42:52 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.72 seconds
[koku@koku ~]$ curl 192.168.56.103
<!DOCTYPE html>
<html>
<head>
  <title>Secure Login Portal Decoy</title>
  <style>
    body { font-family: Arial; background: #f0f0f0; text-align: center; margin-top: 1
00px; }
    form { background: white; display: inline-block; padding: 20px; border-radius: 10
px; }
    input { margin: 10px; padding: 10px; width: 200px; }
    button { padding: 10px 30px; }
  </style>
</head>
<body>
  <h2>Secure Login Portal - Decoy</h2>
  <form>
    <input type="text" name="username" placeholder="Username" required><br>
    <input type="password" name="password" placeholder="Password" required><br>
    <button type="submit">Login</button>
  </form>
  <p>Verifying credentials... please wait</p>
</body>
</html>

[koku@koku ~]$
```

After all implementations:

- Run the Python Deception Proxy:

```
sudo myenv/bin/python3 deception_proxy.py
```

Execute this script on Server 1.

- After starting the proxy, initiate the scan from the attacker machine.

🔸 Once the scan begins, go to Server 1. It will prompt for the password of Server 2. Enter the password to enable redirecting all real services to Server 2.

Testing / Validation Checklist

Action	Expected Result
User opens Server 1	Sees "Secure Login Portal - Server 1"
Attacker scans Server 1	Gets ports: 80 (real) + 21, 23 (fake)
Attacker flagged	Redirected to Decoy
User reopens Server 1	Gets redirected to Server 2 portal