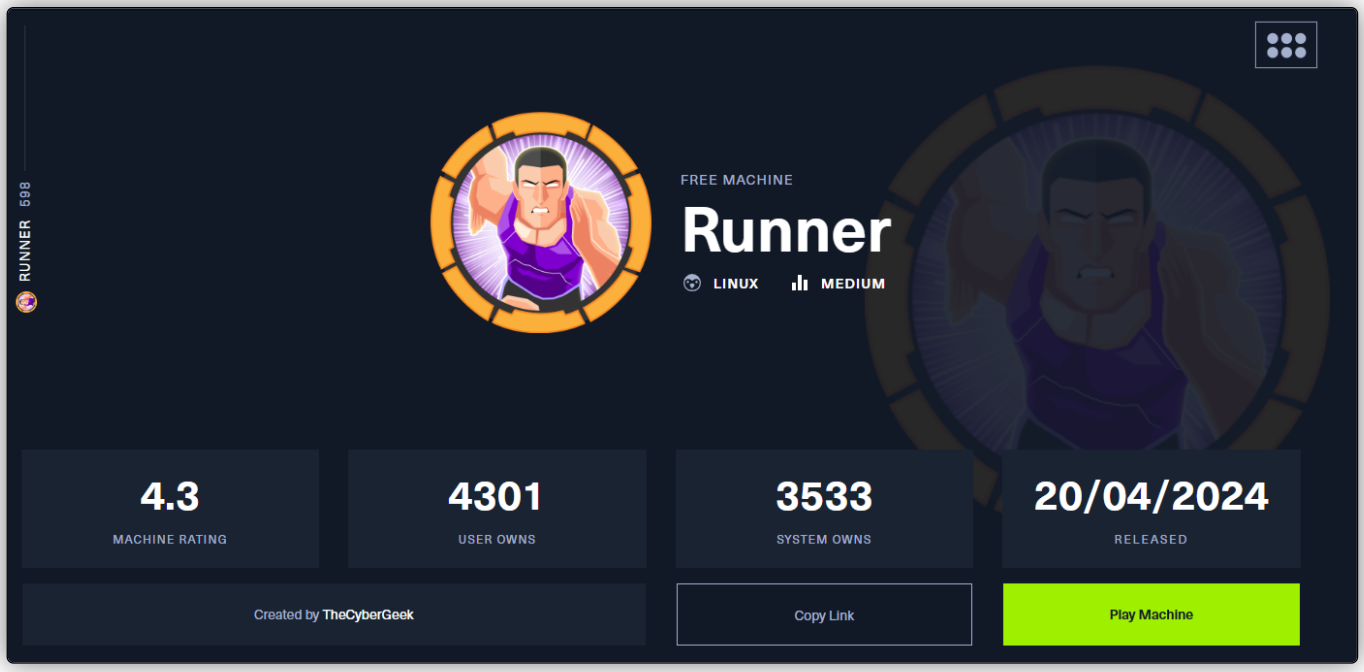
279- RUNNER

1. RUNNER

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

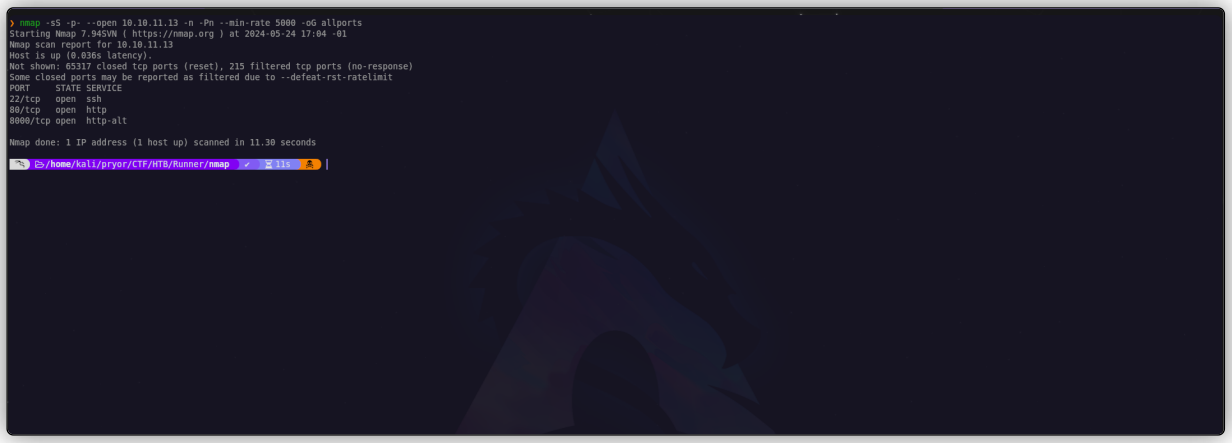


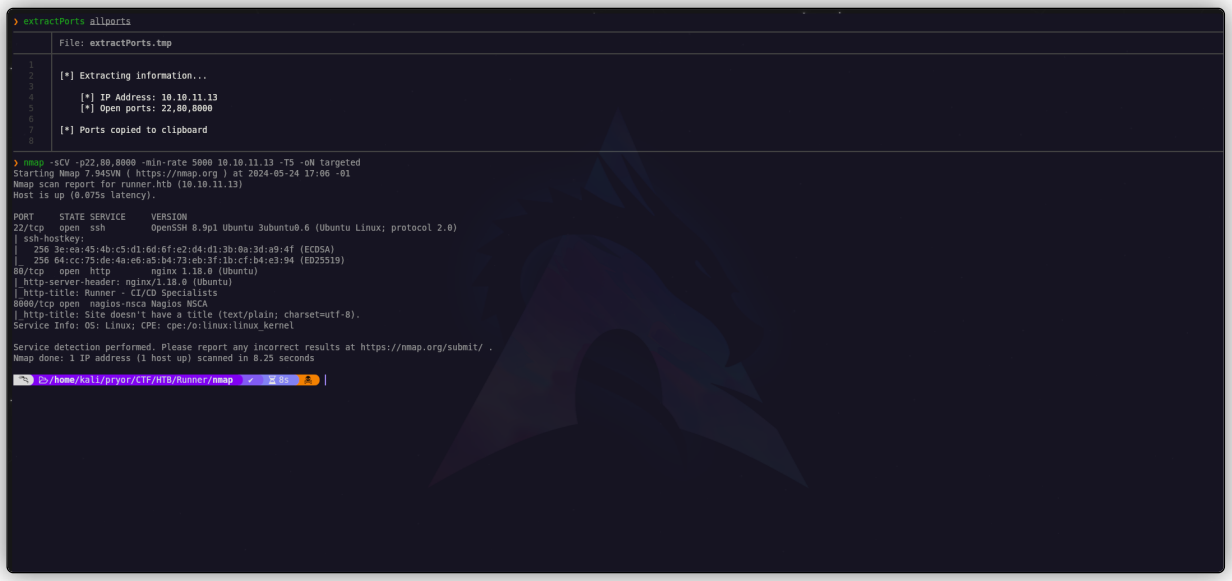
1.1. Preliminar

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

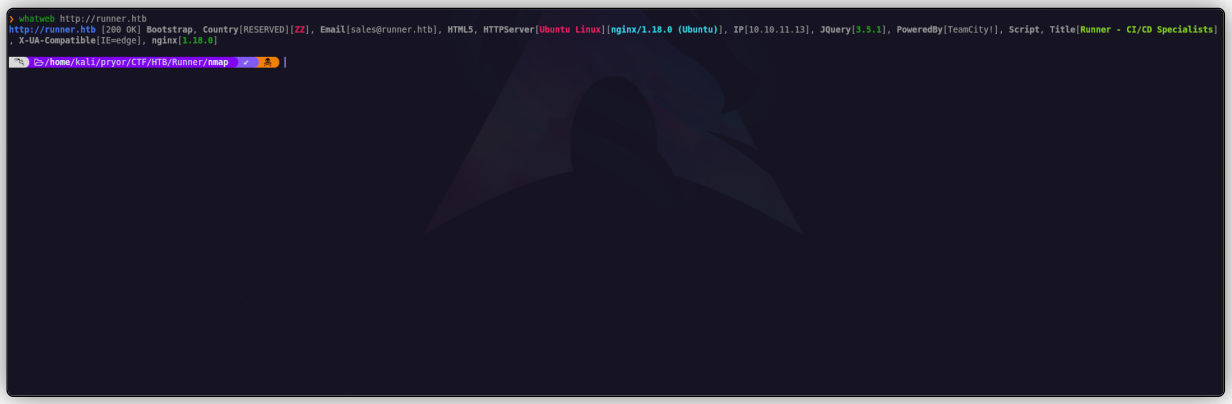
- 

1.2. Nmap

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

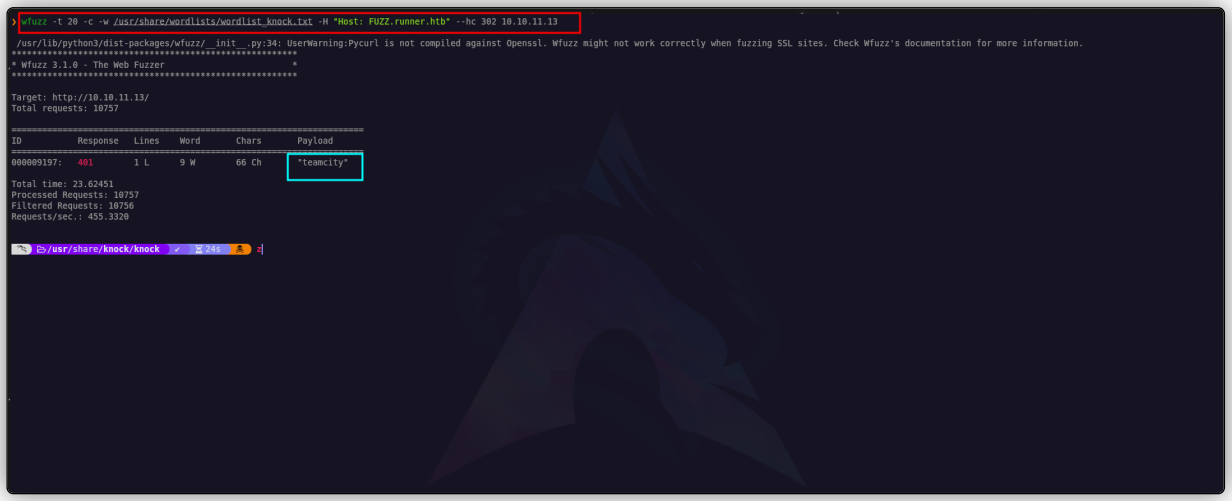
- Escaneo de scripts por defecto y versiones sobre los puertos abiertos, tomando como input los puertos de *allports* mediante `extractPorts`. Añadimos *runner.htb* a nuestro /etc/hosts.
  - 

1.3. Tecnologías web

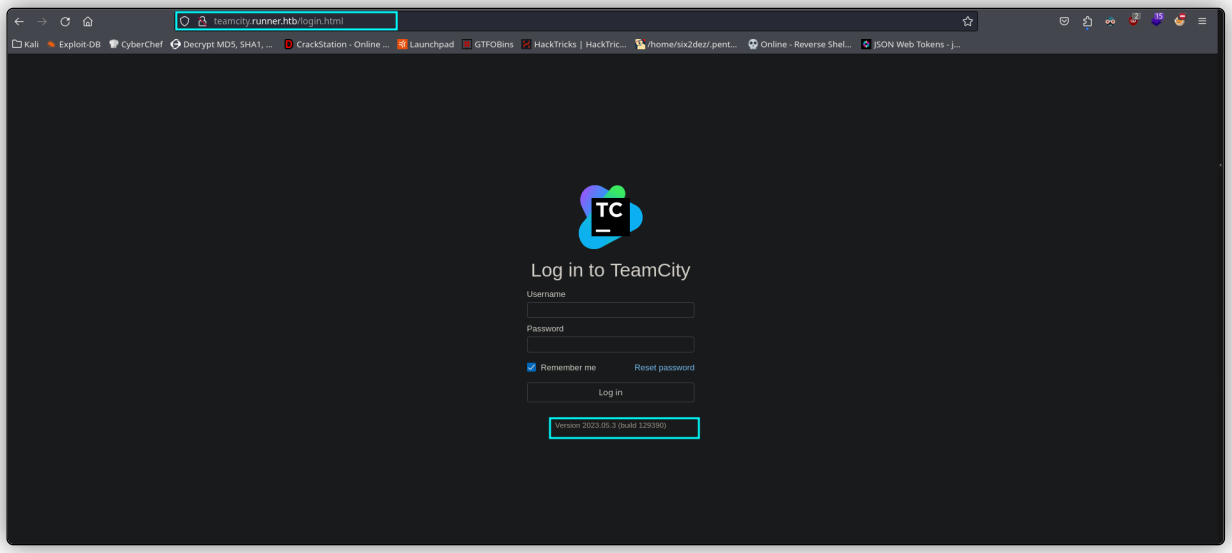
- **Whatweb**: nos reporta lo siguiente. Algo interesante que vemos es que el sitio web está impulsado por *TeamCity*.
    - 

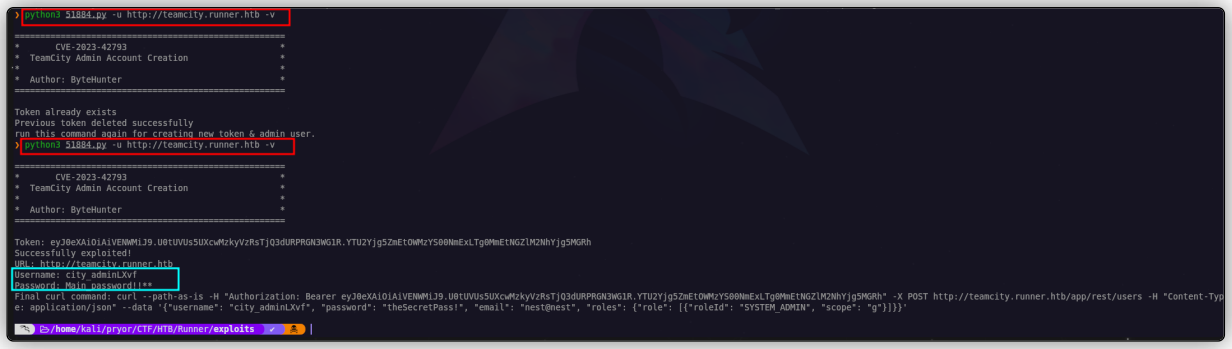
> - **TeamCity** es un servidor de integración continua y entrega continua (CI/CD) desarrollado por JetBrains. Es una herramienta utilizada por equipos de desarrollo de software para automatizar la compilación, prueba y despliegue de aplicaciones.

1.4. Fuzzing subdomains

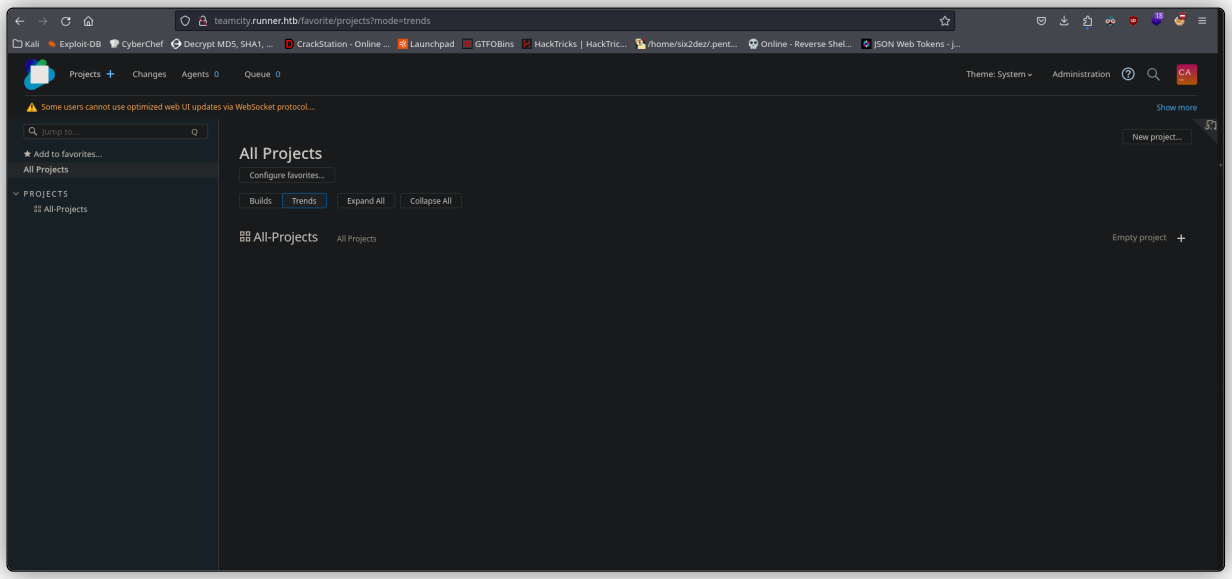
- Entramos a la web, pero tras examinarla, no encontramos nada. Hacemos fuzzing de directorios y subdominios, pero tampoco encontramos nada. No obstante, con **Wfuzz**, al cambiar de diccionario y usar el que trae por defecto **Knockpy**, encontramos otro dominio: *teamcity.runner.htb* el cual añadimos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

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*.
  - 

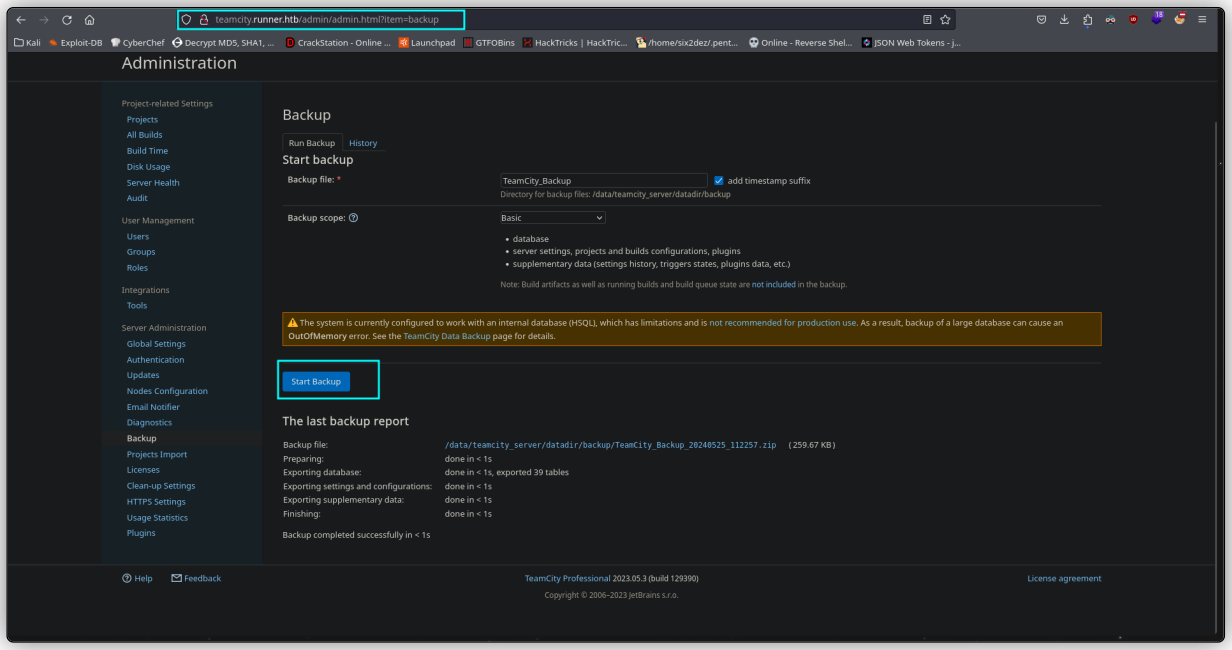
- Buscamos exploits para esta versión. Encontramos uno que compartimos a continuación. Este exploit nos permite bypasear el login de la página y derivar en una ejecución remota de comandos. Lo descargamos. Tendremos que ejecutarlo dos veces: una para eliminar el token de usuario ya existente, y otra para crear un nuevo usuario administrador, para el cual obtendremos las credenciales de acceso.
  - https://www.exploit-db.com/exploits/51884
  - 

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

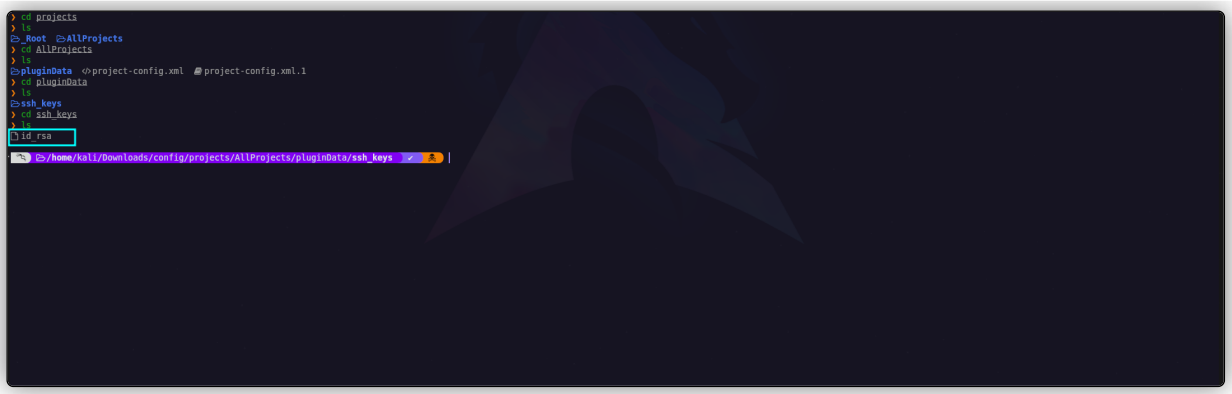


1.6. Database dump

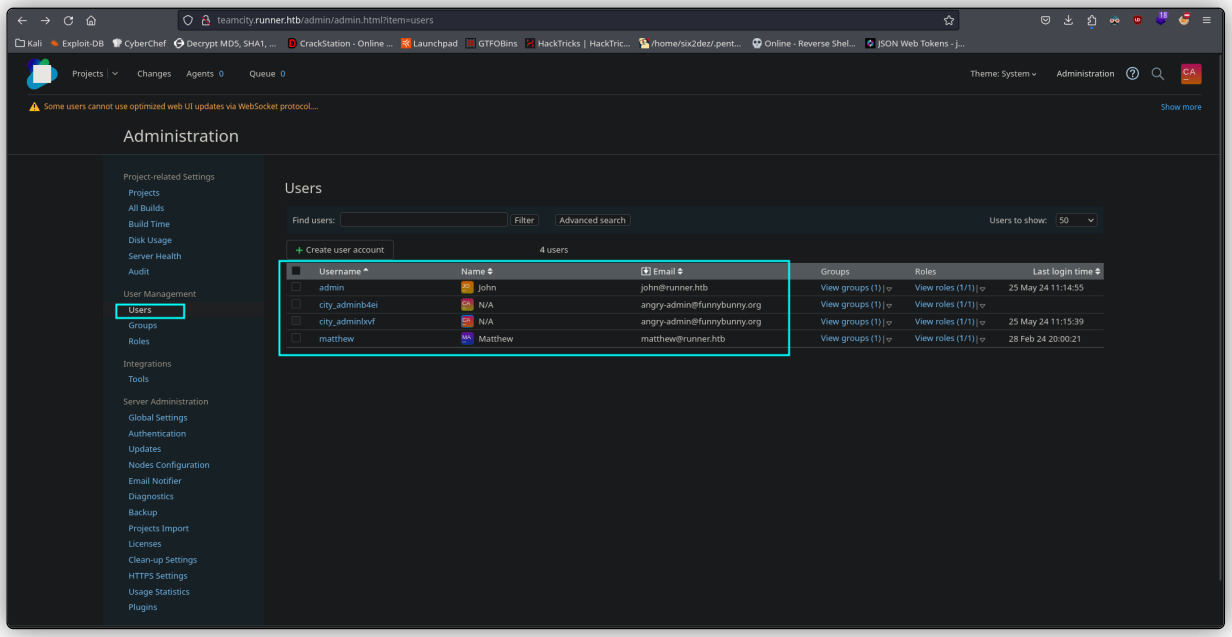
- Explorando este sitio web, vemos que podemos hacer un backup de los datos del servidor. Iniciamos este backup y descargamos el comprimido resultante a nuestro sistema.



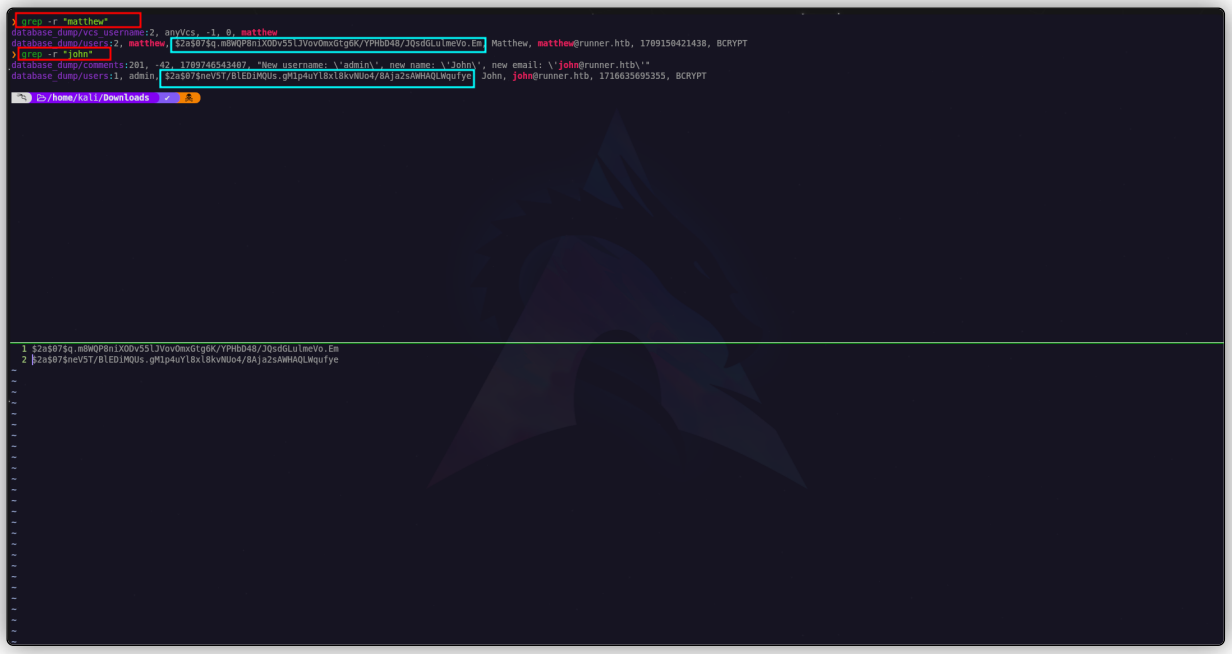
- Explorando ahora estos archivos, encontramos una clave *id_rsa*. Tratamos de conectarnos al sistema usando esta clave SSH y usando los nombres que encontramos en el primer dominio web, pero no obtenemos acceso.



- 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.

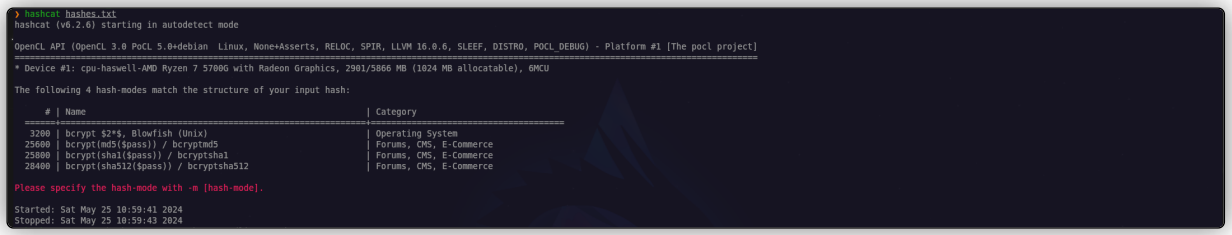


- Probamos ahora a buscar información sobre estos usuarios de nuevo en los directorios del backup. Para ello, hacemos: `grep -r "matthew"` y `grep -r "john"` (usuarios que, como hemos dicho, encontramos en el primer dominio web). Encontramos contraseñas hasheadas para estos usuarios, las cuales guardamos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

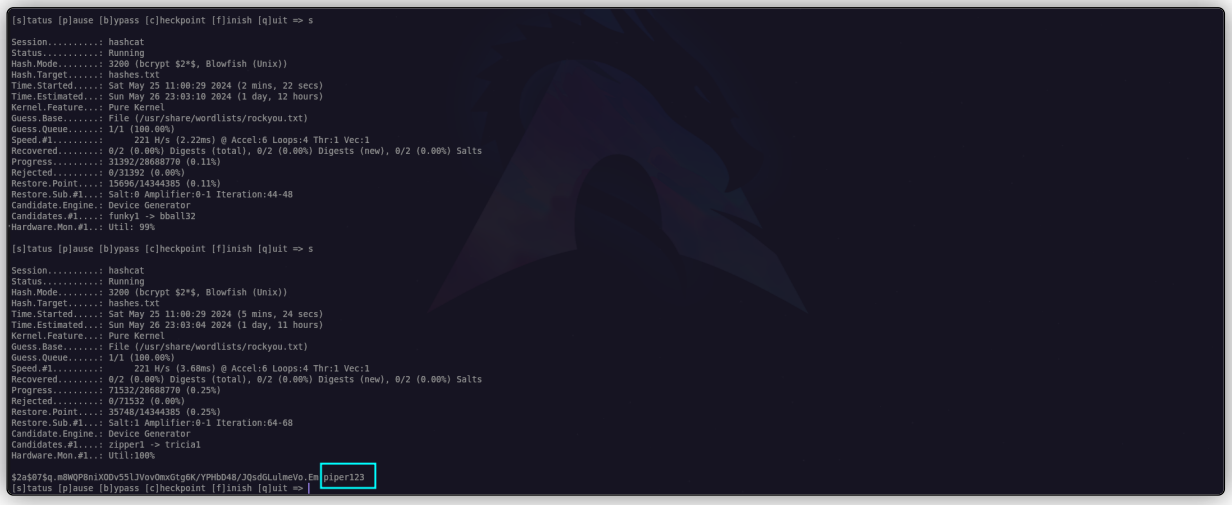


1.7. Cracking hashes with Hashcat

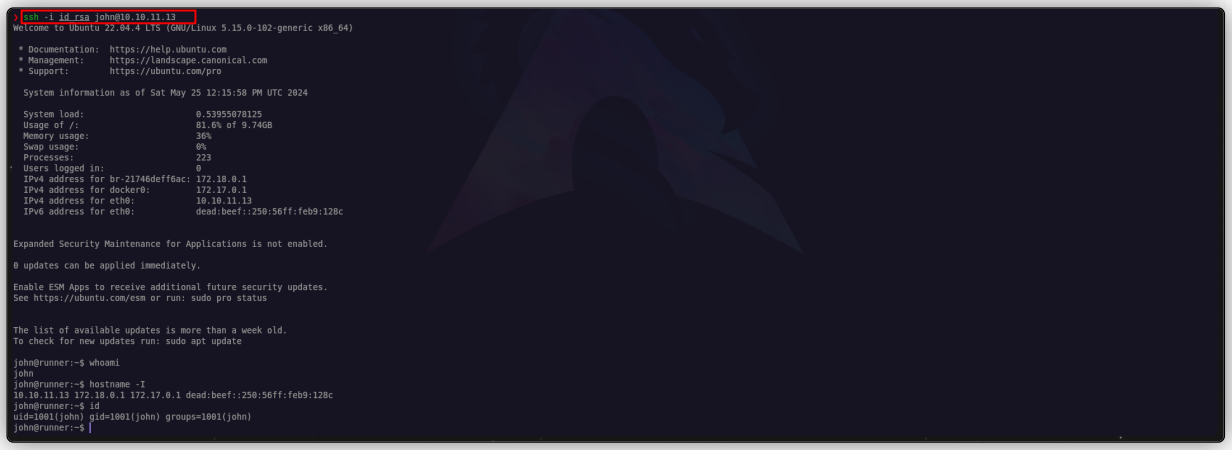
- Vamos a crackear ahora estos hashes, pero primero comprobaremos qué algoritmo de hash están usando: `hashcat hashes.txt`. Vemos que el formato más probable es *bcrypt*, un algoritmo que incorpora de manera automática *salt*.



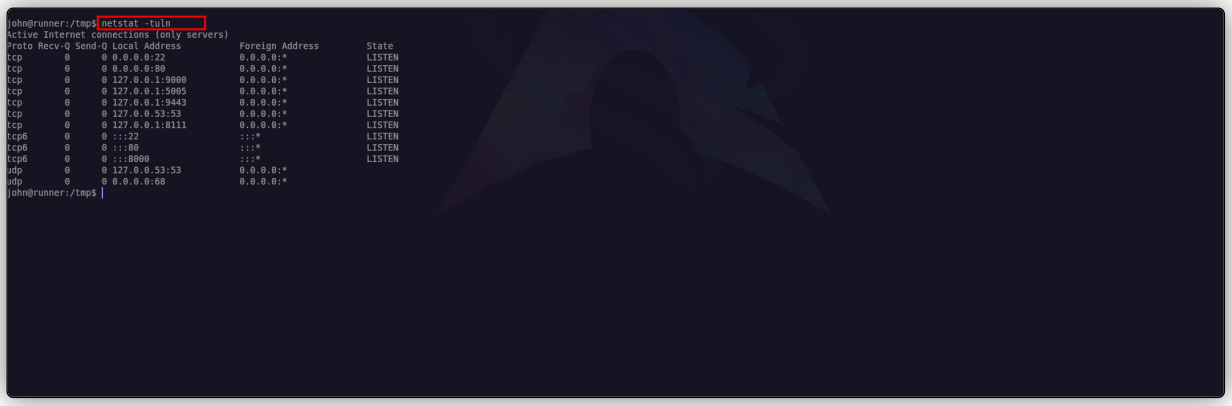
- Usamos `hashcat -m 3200 hashes.txt /usr/share/wordlists/rockyou.txt` para crackear las contraseñas. Al cabo de unos minutos, obtenemos una en texto claro: *piper123*.



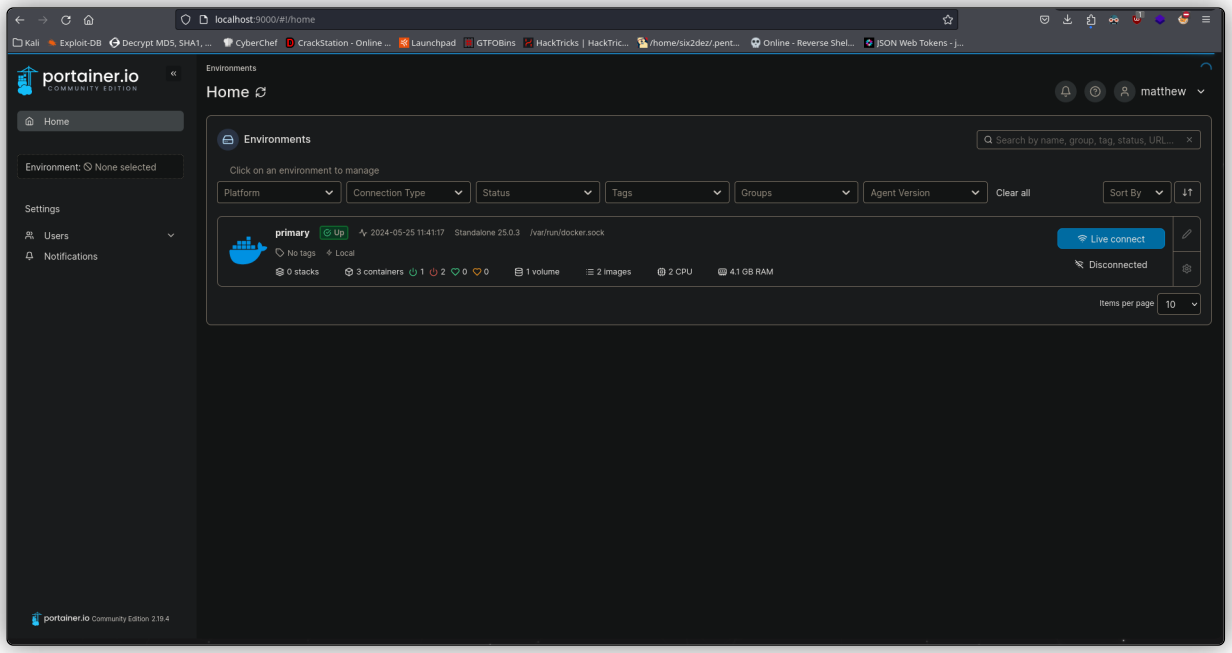
1.8. Access via SSH

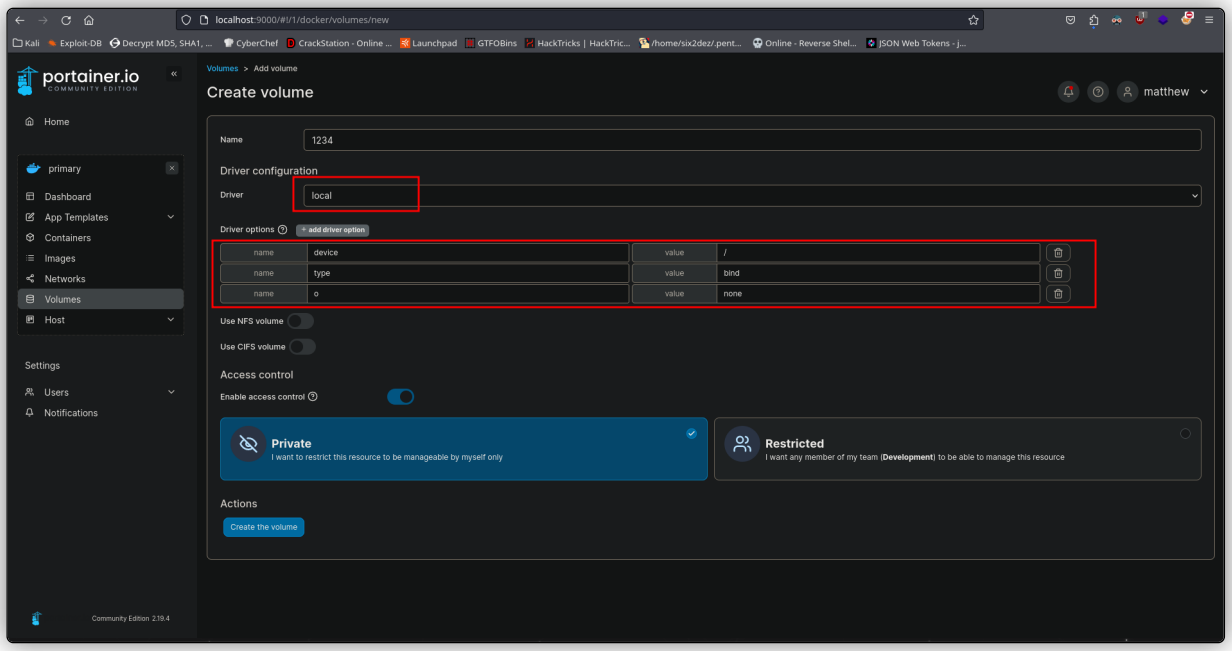
- Usamos estas credenciales y la clave *id_rsa* para conectarnos como *john* por SSH. Obtenemos acceso.
  - A parte de usar en este caso tanto la contraseña como la clave privada SSH, es importante que tengamos en cuenta los permisos de las claves SSH, ya que en un principio no podíamos acceder al sistema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

1.9. Remote port forwarding with Chisel

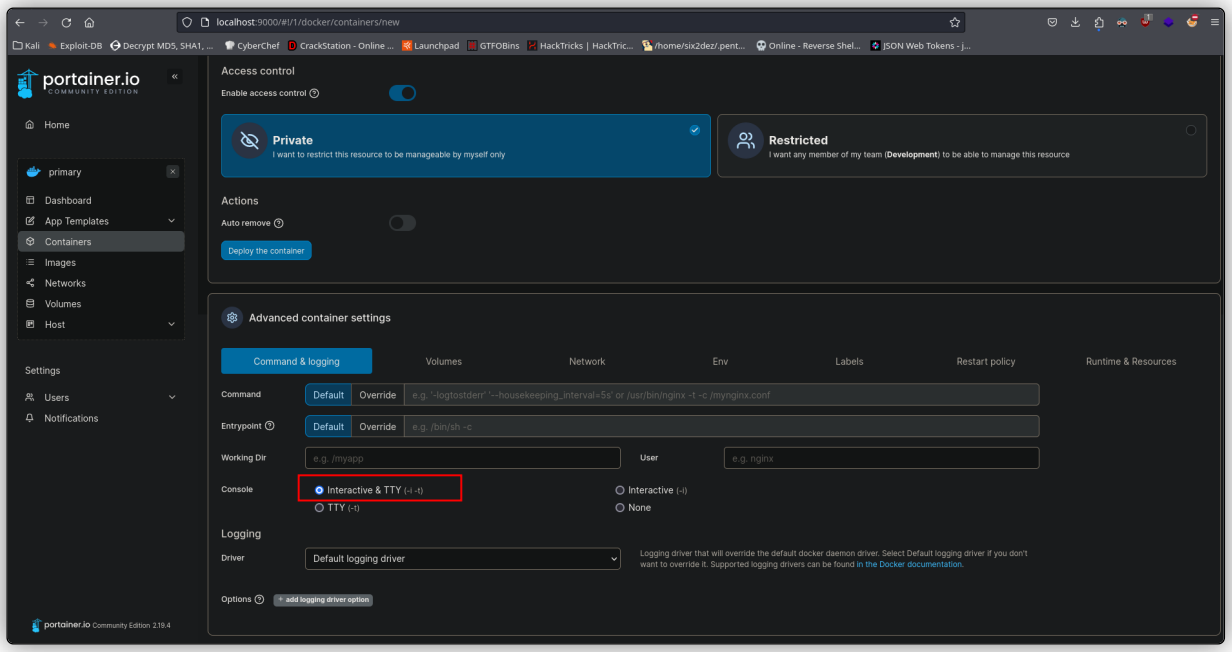
- Vemos ahora los puertos abiertos en el sistema víctima con `netstat -tuln`. De entre todos estos puertos, consideramos interesante traernos el *puerto 9000* (puerto que no vimos abierto en nuestro escaneo de Nmap) a nuestro sistema mediante un *remote port forwarding* con **Chisel**. Transferimos Chisel 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`.
  - 

1.10. Privesc via Portainer root directory mount

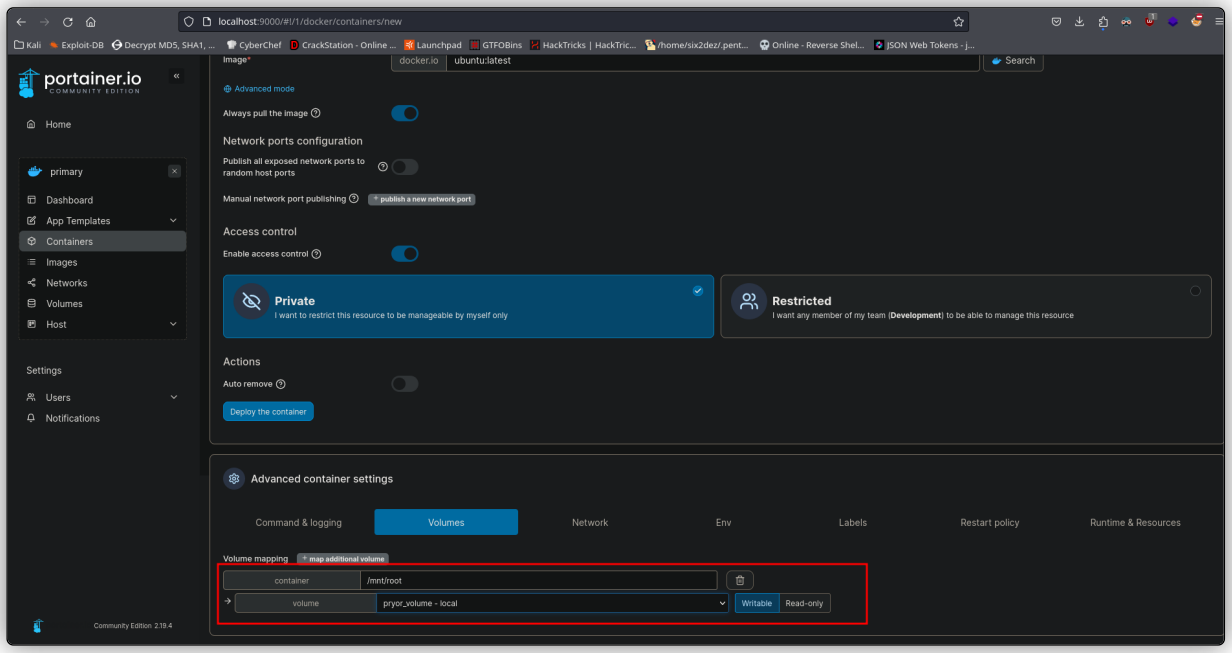
- Accedemos ahora desde nuestro navegador al *puerto 9000*. Aquí está corriendo **Portainer**, que es una plataforma de administración para Docker. Tenemos un panel de login, pero conseguimos acceso usando las credenciales de *matthew*. La idea es realizar una montura en un contenedor con el directorio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

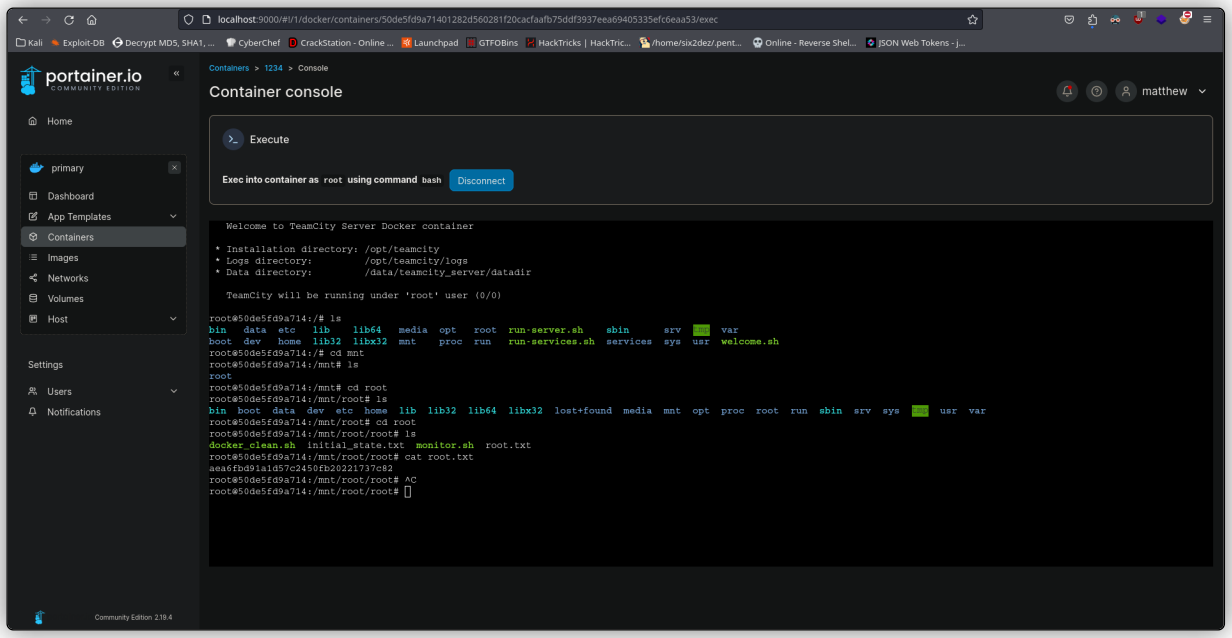
- Configuramos esto tal y como aparece aquí. Esto servirá para obtener una shell interactiva que nos permita interactuar con el contenedor.



- 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.


localhost:9000/#!/1/docker/containers/50de5fd9a7140282d560281f20cacfaafb75ddf3937eea69405335efc6eaa53/exec
Containers > 1234 > Console
Container console
matthew
Execute
Exec into container as root using command bash
Disconnect
Welcome to TeamCity Server Docker container
* Installation directory: /opt/teamcity
* Logs directory: /opt/teamcity/logs
* Data directory: /data/teamcity_server/datadir
TeamCity will be running under 'root' user (0/0)
root@50de5fd9a714:/# ls
bin data etc lib lib64 media opt root run-server.sh sbin srv tmp var
boot dev home lib32 libx32 mnt proc run run-services.sh services sys usr welcome.sh
root@50de5fd9a714:/# cd mnt
root@50de5fd9a714:/mnt# ls
root
root@50de5fd9a714:/mnt# cd root
root@50de5fd9a714:/mnt/root# ls
bin boot data dev etc home lib lib32 lib64 libx32 lost+found media mnt opt proc root run sbin srv sys tmp usr var
root@50de5fd9a714:/mnt/root# cd root
root@50de5fd9a714:/mnt/root/root# ls
docker_clean.sh initial_state.txt monitor.sh root.txt
root@50de5fd9a714:/mnt/root/root# cat root.txt
aea6fbd91a1d57c2450fb20221737e82
root@50de5fd9a714:/mnt/root/root# ^C
root@50de5fd9a714:/mnt/root/root#
Home
primary
Dashboard
App Templates
Containers
Images
Networks
Volumes
Host
Settings
Users
Notifications
Community Edition 2.19.4