

Enumeración de servicios

FTP

FTP es un protocolo ampliamente utilizado para la **transferencia de archivos** en redes. La enumeración del servicio FTP implica recopilar información relevante, como la versión del servidor FTP, la configuración de permisos de archivos, los usuarios y las contraseñas (mediante ataques de fuerza bruta o guessing), entre otros.

Bien procedamos a instalar nuestro laboratorio para la clase

[Laboratorio de Docker ftp server](#)

Corramos este comando para crear un servidor ftp en docker

```
docker run \
  --detach \
  --env FTP_PASS=iloveyou \
  --env FTP_USER=admin \
  --name my-ftp-server \
  --publish 20-21:20-21/tcp \
  --publish 40000-40009:40000-40009/tcp \
  --volume /data:/home/user \
  garethflowers/ftp-server
```

Si todo salio bien procedemos a iniciar sesion en ftp

```
ftp localhost
```

Procedemos a poner nuestro usuario y contraseña, pero intentemos poner una contraseña incorrecta para ver si funciona

Bueno ahora es donde veremos como podemos intentar sacarnos la contraseña del ftp suponiendo que sabemos el usuario en este caso es admin

Lo que a mi se me viene a la mente es primero con nmap ver la version o algun script de ver si es vulnerable este ftp, para ellos lo vemos de la siguiente forma

```
nmap -sCV -p21 127.0.0.1
```

Si todo lo hicimos bien nos debería mostrar algo como esto:

```
(root@josh)-[/home/josh]
# nmap -sCV -p21 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-06 19:51 CST
Stats: 0:00:11 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 0.00% done
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000070s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.0.8 or later

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.31 seconds
```

Fíjense que no esta sacando que se puede ingresar como anónimo porque cuando en nmap lanza el para para probar scripts lo que hace es que prueba para ver si se puede iniciar sesion anónimamente normalmente.

Muy bin ahora tratemos de averiguar su contraseña del usuario usando la herramienta hydar

NOTA: les recomiendo siempre leer el manual de cada herramienta que vayan a usar y no lo conocen para ello se usa el paramentro

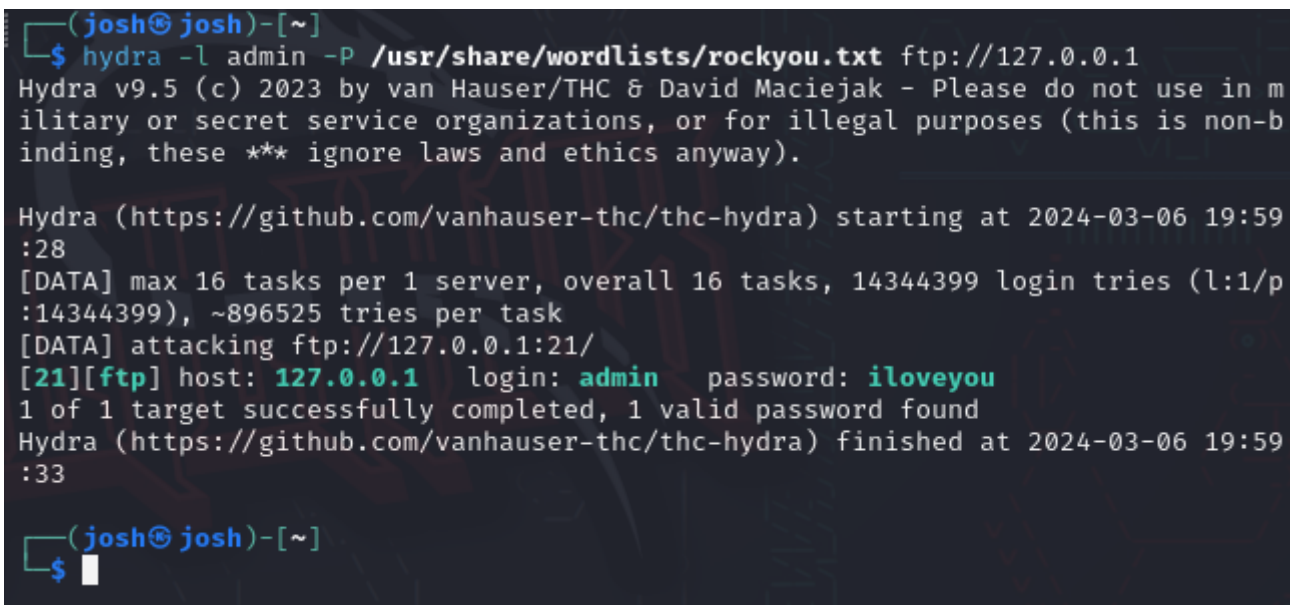
man nmap

Muy bien para poder sacar la contraseña de ftp sabiendo el usuario hacemos lo siguiente:

```
hydra -l admin -P /usr/share/wordlists/rockyou.txt ftp://127.0.0.1
```

En este caso hemos hecho una fuerza bruta para adivinar la contraseña del usuario admin del servicio ftp

si todo les salió bien se debería ver algo así:



```
(josh@josh)-[~]  
$ hydra -l admin -P /usr/share/wordlists/rockyou.txt ftp://127.0.0.1  
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in m  
ilitary or secret service organizations, or for illegal purposes (this is non-b  
inding, these *** ignore laws and ethics anyway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-03-06 19:59  
:28  
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p  
:14344399), ~896525 tries per task  
[DATA] attacking ftp://127.0.0.1:21/  
[21][ftp] host: 127.0.0.1 login: admin password: iloveyou  
1 of 1 target successfully completed, 1 valid password found  
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-03-06 19:59  
:33  
  
(josh@josh)-[~]  
$
```

En la imagen se muestra el usuario y la contraseña encontrada para el dicho usuario

Ahora para probar el usuario de ftp anonimo usaremos este repositorio:

[Docker AnonFtp](#)

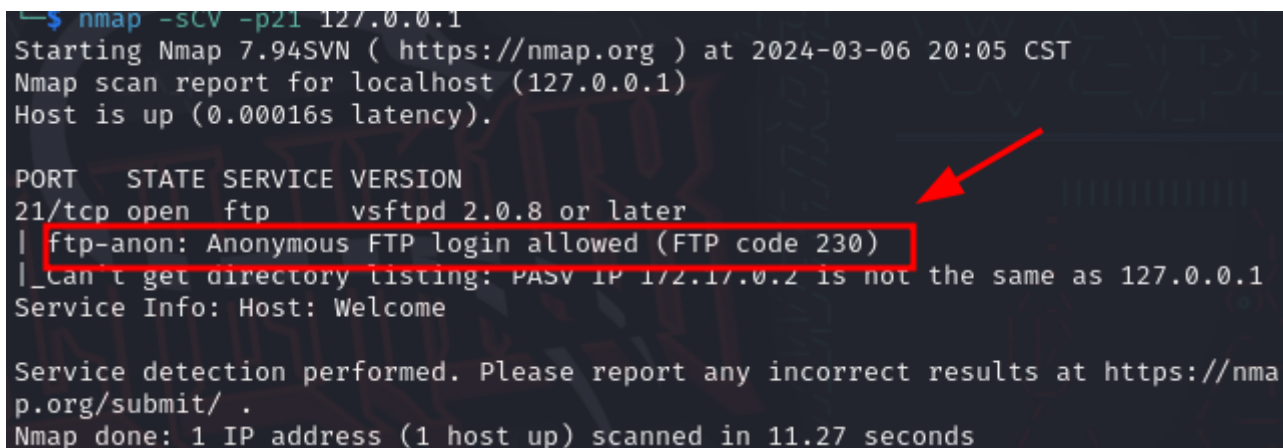
Copiamos el comando que nos dice el repositorio y empezara a desplegarse el servidor en docker

```
docker run -d -p 20-21:20-21 -p 65500-65515:65500-65515 -v /tmp:/var/ftp:ro metabrainz/docker-anon-ftp
```

Muy bien hacemos lo mismo que hicimos en el anterior

```
nmap -sCV -p21 127.0.0.1
```

y nos mostrar algo como esto:



```
$ nmap -sCV -p21 127.0.0.1
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-03-06 20:05 CST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00016s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.0.8 or later
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_ Can't get directory listing: PASV IP 172.17.0.2 is not the same as 127.0.0.1
Service Info: Host: Welcome

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.27 seconds
```

A red box highlights the line "ftp-anon: Anonymous FTP login allowed (FTP code 230)" and a red arrow points to it from the right.

Como les comentaba en este caso, nos esta mostrando que el servidor se puede ingresar como anónimo y para hacerlo hacemos:

```
ftp localhost
```

Nos pedirá el usuario y pondremos anonymous, nos pedirá contraseña le daremos enter ya que por defecto este usuario no necesita contraseña y nos dejara acceder

```
(josh@josh)-[~]  
$ ftp localhost  
Trying [::1]:21 ...  
Connected to localhost.  
220 Welcome to an awesome public FTP Server  
Name (localhost:josh): anonymous  
331 Please specify the password.  
Password:  
230 Login successful.  
Remote system type is UNIX.  
Using binary mode to transfer files.  
ftp> █
```

SMB

SMB significa **Server Message Block**, es un **protocolo** de comunicación de red utilizado para compartir archivos, impresoras y otros recursos entre dispositivos de red. Es un protocolo propietario de **Microsoft** que se utiliza en sistemas operativos **Windows**.

NOTA: NO ES LO MISMO SMB QUE SAMBA

Samba, por otro lado, es una implementación libre y de código abierto del **protocolo SMB**, que se utiliza principalmente en sistemas operativos basados en **Unix** y **Linux**. Samba proporciona una manera de compartir archivos y recursos entre dispositivos de red que ejecutan sistemas operativos diferentes, como Windows y Linux.

El laboratorio que usaremos será el siguiente:

[Samba Authenticated RCE](#)

Para este laboratorio usaremos una de mis servicios preferidos el cual es docker-compose, si no la tienes instalado lo podemos hacer de la siguiente manera:

```
sudo apt install docker-compose
```

Una vez instalado, procedemos a clonar la repo

```
curl https://codeload.github.com/vulhub/vulhub/tar.gz/master | tar -xz --strip=2 vulhub-master/samba/CVE-2017-7494
```

Una vez descargado el yml procedemos a levantarlo, se hace de la siguiente manera:

```
docker-compose up -d
```

Bien ahora una de las herramientas que utilizamos para la fase de reconocimiento es '**smbmap**'. Smbmap es una herramienta de línea de comandos utilizada para enumerar recursos compartidos y permisos en un servidor SMB (Server Message Block) o Samba. Es una herramienta muy útil para la enumeración de redes y para la identificación de posibles vulnerabilidades de seguridad.

Con smbmap, puedes enumerar los recursos compartidos en un servidor SMB y obtener información detallada sobre cada recurso, como los permisos de acceso, los usuarios y grupos autorizados, y los archivos y carpetas compartidos. También puedes utilizar smbmap para identificar recursos compartidos que no requieren autenticación, lo que puede ser un problema de seguridad.

Además, smbmap permite a los administradores de sistemas y a los auditores de seguridad verificar rápidamente la configuración de permisos en los recursos compartidos en un servidor SMB, lo que puede ayudar a identificar posibles vulnerabilidades de seguridad y a tomar medidas para remediarlas.

A continuación, se proporciona una breve descripción de algunos de los parámetros comunes de smbmap:

- **-H**: Este parámetro se utiliza para especificar la dirección IP o el nombre de host del servidor SMB al que se quiere conectarse.
- **-P**: Este parámetro se utiliza para especificar el puerto TCP utilizado para la conexión SMB. El puerto predeterminado para SMB es el 445, pero si el servidor SMB está configurado para utilizar un puerto diferente, este parámetro debe ser utilizado para especificar el puerto correcto.
- **-u**: Este parámetro se utiliza para especificar el nombre de usuario para la conexión SMB.
- **-p**: Este parámetro se utiliza para especificar la contraseña para la conexión SMB.
- **-d**: Este parámetro se utiliza para especificar el dominio al que pertenece el usuario que se está utilizando para la conexión SMB.

- **-s:** Este parámetro se utiliza para especificar el recurso compartido específico que se quiere enumerar. Si no se especifica, smbmap intentará enumerar todos los recursos compartidos en el servidor SMB.

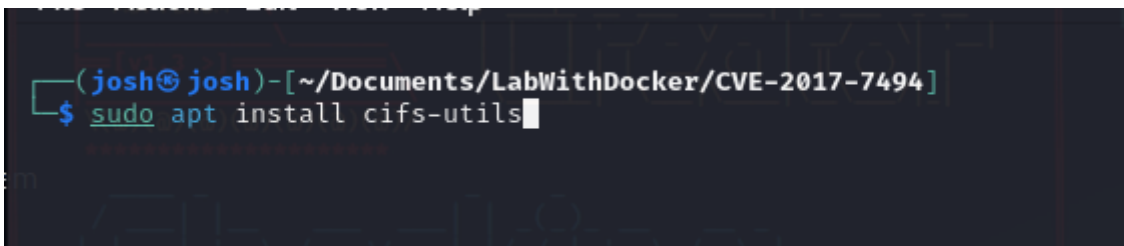
Asimismo, otra de las herramientas que se ven en esta clase es '**smbclient**'. Smbclient es otra herramienta de línea de comandos utilizada para interactuar con servidores SMB y Samba, pero a diferencia de smbmap que se utiliza principalmente para enumeración, smbclient proporciona una interfaz de línea de comandos para interactuar con los recursos compartidos SMB y Samba, lo que permite la descarga y subida de archivos, la ejecución de comandos remotos, la navegación por el sistema de archivos remoto, entre otras funcionalidades.

En cuanto a los parámetros más comunes de smbclient, algunos de ellos son:

- **-L:** Este parámetro se utiliza para enumerar los recursos compartidos disponibles en el servidor SMB o Samba.
- **-U:** Este parámetro se utiliza para especificar el nombre de usuario y la contraseña utilizados para la autenticación con el servidor SMB o Samba.
- **-c:** Este parámetro se utiliza para especificar un comando que se ejecutará en el servidor SMB o Samba.

Estos son algunos de los parámetros más comunes utilizados en smbclient, aunque hay otros disponibles. La lista completa de parámetros y sus descripciones se pueden encontrar en la documentación oficial de la herramienta.

Antes de empezar a ver este laboratorio primero vamos a instalar cifs

A terminal window with a dark background. The prompt is '(josh@josh)-[~/Documents/LabWithDocker/CVE-2017-7494]'. The user has entered the command '\$ sudo apt install cifs-utils' and the cursor is at the end of the line.

```
(josh@josh)-[~/Documents/LabWithDocker/CVE-2017-7494]  
$ sudo apt install cifs-utils
```

Una vez ya instalado, procedamos a empezar, primero como vimos usaremos smbcliente para intentar conectarnos al servicio smb para ello hacemos lo siguiente

```
smbclient -L 127.0.0.1 -N
```

Lo que significa cada parámetro ya lo hemos visto, pero recordando el -L nos permite enumerar recursos compartidos del servidor como estamos jugando como docker y forward de puertos, por eso usamos la 127.0.0.1.

el -N es para ver si es posible acceder con una sesión nula, como un tipo Anonymous en ftp

Nos debería verse algo así:

```
(josh@josh)-[~/Documents/LabWithDocker/CVE-2017-7494]
$ smbclient -L 127.0.0.1 -N

      Sharename      Type      Comment
      ────
      myshare        Disk
      IPC$           IPC       IPC Service (Samba Server Version 4.6.3)
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 127.0.0.1 failed (Error NT_STATUS_CONNECTION_REFUSED)
Unable to connect with SMB1 -- no workgroup available

(josh@josh)-[~/Documents/LabWithDocker/CVE-2017-7494]
$
```

Como no sabemos que permisos tiene el directorio compartido, usamos otra herramienta llamada, smbmap, eso nos mostrar de una manera más bonita los permisos que tiene a donde nos podemos contar, para usarlo sería así;


```
(josh@josh)-[~/Documents/LabWithDocker/CVE-2017-7494]
$ smbmap -H 127.0.0.1

SMBMap - Samba Share Enumerator | Shawn Evans - ShawnDEvans@gmail.com
https://github.com/ShawnDEvans/smbmap

[*] Detected 1 hosts serving SMB
[*] Established 1 SMB session(s)

[+] IP: 127.0.0.1:445   Name: localhost   Status: Authenticated
    Disk                Permissions    Comment
    ---                -
    myshare              READ, WRITE
    IPC$                 NO ACCESS     IPC Service (Samba Server Version 4.6.3)
```

Como podemos ver, ahora vemos los permisos de escritura y lectura en la carpeta compartida, procedemos a conectarnos

```
smbclient //127.0.0.1/myshare -N
```

y se nos debería ver algo así

```
(josh@josh)-[~/Documents/LabWithDocker/CVE-2017-7494]
$ smbclient //127.0.0.1/myshare -N
Try "help" to get a list of possible commands.
smb: \>
```

Muy bien, a veces puede ser molesto poder navegar y no poder ver como el linux que por medio del comando tree se ve mucho mejor la estructura de las carpetas etc...

Para ello lo que podemos hacer es usar la herramienta que instalamos al principio, ¿Cómo?, de la siguiente forma:

primero nos crearemos una carpeta en el directorio mnt

```
sudo mkdir /mnt/mounted
```

Una vez ya creado hacemos que se monte todo el contenido que hay en el servidor para así poder verlo de manera más bonita y no tardar horas haciéndolo

```
mount -t cifs //127.0.0.1/myshare
```

Al darle enter nos pedirá contraseña, pero no se preocupen solo le damos enter ya que al no proporcionar una contraseña considerará que es una sesión nula, muy bien ahora en el directorio /mnt/mounted tendremos todo lo que hay en el servidor smb y podemos navegar de forma más rápida y cómoda

NOTA: Todo lo que le hagas, borres o modifiques se le aplicará al servidor también así que mucho cuidado, y para desmontar solo será poner

```
umount /mnt/mounted
```