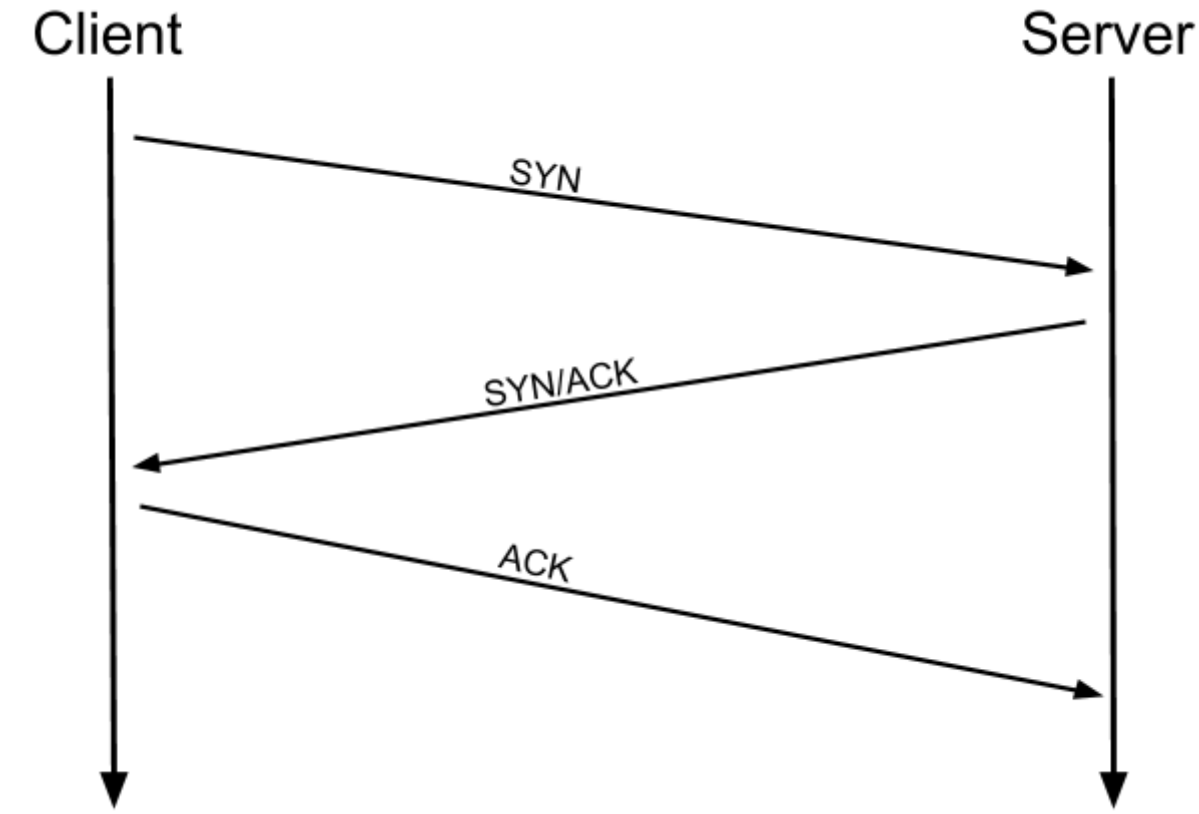
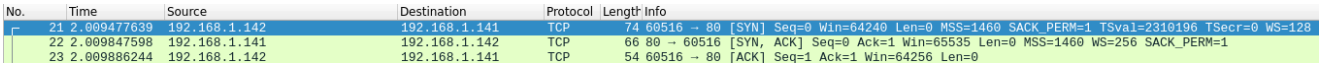
TCP Connect Scans

Como breve recapitulación, el *three-way handshake* consiste en tres etapas. En primer lugar, el terminal de conexión (nuestra máquina atacante, en este caso) envía una petición TCP al servidor de destino con la bandera SYN activada. A continuación, el servidor acusa recibo de este paquete con una respuesta TCP que contiene la bandera SYN, así como la bandera ACK. Por último, nuestro terminal completa el intercambio de información enviando una petición TCP con la bandera ACK activada.

Entendemos

que el three-way handshake consite en tres etapas, la maquina atacante envia una petición TCP al serpidor de destino con la bandera SYN activada. y el servidor responde con SYN y ACK activadas y para finalizar nuestra maquina completa el intercambio de información enviando una petición TCP con la bandera ACK activada.





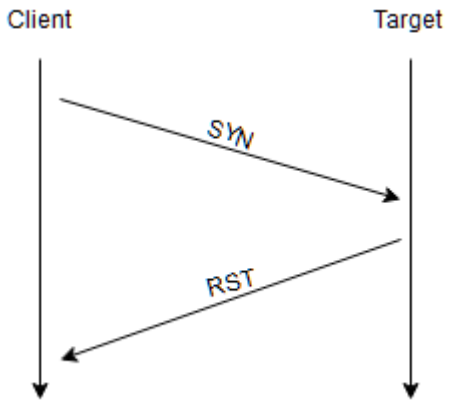
Este es uno de los principios fundamentales de las redes TCP/IP, pero ¿cómo se relaciona con Nmap?

Bueno, como su nombre indica, un escaneo de conexión TCP funciona realizando el apretón de manos de tres vías con cada puerto de destino sucesivamente. En otras palabras, Nmap intenta conectarse a cada puerto TCP especificado y determina si el servicio está abierto por la respuesta que recibe.

Por ejemplo, si un puerto está cerrado, el RFC 793 establece que:

"... Si la conexión no existe (**CLOSED**) entonces se envía un reset en respuesta a cualquier segmento entrante, excepto otro reset. En particular, los SYN dirigidos a una conexión inexistente son rechazados por este medio".

En otras palabras, si Nmap envía una petición TCP con la bandera SYN establecida a un puerto cerrado, el servidor objetivo responderá con un paquete TCP con la bandera RST (Reset) establecida. Mediante esta respuesta, Nmap puede establecer que el puerto está cerrado.



Sin embargo, si la petición se envía a un puerto abierto, el objetivo responderá con un paquete TCP con las banderas SYN/ACK activadas. Nmap marca entonces este puerto como abierto (y completa el handshake enviando de vuelta un paquete TCP con ACK activado).

Todo esto está muy bien, pero hay una tercera posibilidad.

¿Y si el puerto está abierto, pero oculto tras un cortafuegos?

Muchos cortafuegos están configurados para simplemente descartar los paquetes entrantes. Nmap envía una petición TCP SYN y no recibe nada de vuelta. Esto indica que el puerto está protegido por un cortafuegos y, por tanto, se considera que el puerto está filtrado.

Dicho esto, es muy fácil configurar un cortafuegos para que responda con un paquete TCP RST. Por ejemplo, en IPtables para Linux, una versión simple del comando sería la siguiente:

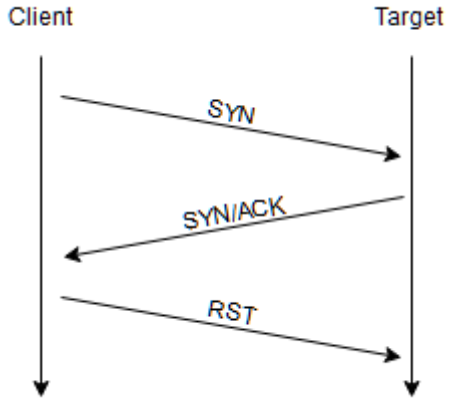
```
`iptables -I INPUT -p tcp --dport <port> -j REJECT --reject-with tcp-reset`
```

Esto puede hacer que sea extremadamente difícil (si no imposible) obtener una lectura precisa del objetivo(s).

SYN Scans

Al igual que los escaneos TCP, los escaneos SYN (-sS) se utilizan para escanear el rango de puertos TCP de uno o varios objetivos; sin embargo, los dos tipos de escaneo funcionan de forma ligeramente diferente. Los escaneos SYN se denominan a veces escaneos "semiabiertos" o escaneos "sigilosos".

Mientras que los escaneos TCP realizan un handshake completo de tres vías con el objetivo, los escaneos SYN devuelven un paquete TCP RST después de recibir un SYN/ACK del servidor (esto evita que el servidor intente hacer la petición repetidamente). En otras palabras, la secuencia para escanear un puerto abierto es la siguiente



No.	Time	Source	Destination	Protocol	Length	Info
39	8.389443540	192.168.1.142	192.168.1.238	TCP	58	53425 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
40	8.389900067	192.168.1.238	192.168.1.142	TCP	60	80 → 53425 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460
41	8.389992786	192.168.1.142	192.168.1.238	TCP	54	53425 → 80 [RST] Seq=1 Win=0 Len=0

Entendemos

los escaneos SYN (-sS) se utilizan para escanear el rango de puertos TCP de uno o varios objetivos.

Sin embargo, los dos tipos de escaneos funcionan de forma ligeramente diferente.

Los escaneos de SYN se denominan a veces escaneos "Semiabiertos" o "Sigilosos".

Mientras que los TCP realizan un handshake completo de tres visa, los escaneos SYn devuelven un paquete TCP RST despues ed recibir un SYN/ACK del servidor.

Esto tiene una serie de ventajas para nosotros como hackers:

- Puede utilizarse para eludir los sistemas de detección de intrusos más antiguos, ya que buscan un apretón de manos completo de tres vías. Este ya no es el caso de las soluciones IDS

modernas; es por esta razón que los escaneos SYN todavía se conocen como escaneos "sigilosos".

- Los escaneos SYN a menudo no son registrados por las aplicaciones que escuchan en puertos abiertos, ya que la práctica estándar es registrar una conexión una vez que se ha establecido completamente. De nuevo, esto juega con la idea de que los escaneos SYN son sigilosos.
- Sin tener que molestarse en completar (y desconectarse de) un apretón de manos de tres vías para cada puerto, los escaneos SYN son significativamente más rápidos que un escaneo TCP Connect estándar.

> **Entendemos**
>
> Tiene ventajas ya que puede utilizarse para eludir los sistemas de detección más antiguos.
>
> Los escaneos SYN a menudo no son registrados por la aplicacion que escuchan en puertos abiertos.
>
> Los escaneos SYN son significativamente más rapidos.

Sin embargo, los escaneos SYN tienen un par de desventajas, a saber:

- Requieren permisos sudo[1] para funcionar correctamente en Linux. Esto se debe a que los escaneos SYN requieren la capacidad de crear paquetes sin procesar (en contraposición al handshake TCP completo), que es un privilegio que sólo el usuario root tiene por defecto.
- Los servicios inestables a veces son derribados por escaneos SYN, lo que podría resultar problemático si un cliente ha proporcionado un entorno de producción para la prueba.

> **Entendemos:**
>
> Se necesitan permisos sudo para funcionar correctamente en Linux y los servicios inestables a veces son derribados por escaneos SYN.

En definitiva, los pros superan a los contras.

Por esta razón, los análisis SYN son los análisis por defecto que utiliza Nmap si se ejecuta con permisos sudo. Si se ejecuta sin permisos sudo, Nmap utiliza por defecto el sondeo TCP Connect que vimos en la tarea anterior.

Cuando se utiliza un escaneo SYN para identificar puertos cerrados y filtrados, se aplican exactamente las mismas reglas que con un escaneo TCP Connect.

Si un puerto está cerrado, el servidor responde con un paquete TCP RST. Si el puerto está filtrado por un cortafuegos, el paquete TCP SYN se descarta o se falsifica con un reinicio TCP.

En este sentido, los dos escaneos son idénticos: la gran diferencia está en cómo manejan los puertos abiertos.

[1] Los escaneos SYN también pueden funcionar dándole a Nmap las capacidades CAP_NET_RAW, CAP_NET_ADMIN y CAP_NET_BIND_SERVICE; sin embargo, esto puede no permitir que muchos de los scripts de NSE se ejecuten correctamente.

UDP Scans

A diferencia de TCP, las conexiones UDP no tienen estado. Esto significa que, en lugar de iniciar una conexión con un "apretón de manos" de ida y vuelta, las conexiones UDP se basan en el envío de paquetes a un puerto de destino y, esencialmente, en la esperanza de que lo consigan. Esto hace que UDP sea excelente para las conexiones que se basan en la velocidad por encima de la calidad (por ejemplo, el intercambio de vídeo), pero la falta de reconocimiento hace que UDP sea significativamente más difícil (y mucho más lento) de escanear. El interruptor para un sondeo UDP de Nmap es (-sU)

Cuando se envía un paquete a un puerto UDP abierto, no debería haber respuesta. Cuando esto ocurre, Nmap se refiere al puerto como abierto|filtrado. En otras palabras, sospecha que el puerto está abierto, pero podría estar protegido contra el fuego. Si obtiene una respuesta UDP (lo cual es muy inusual), entonces el puerto se marca como abierto. Lo más habitual es que no haya respuesta, en cuyo caso la petición se envía una segunda vez como doble comprobación. Si

sigue sin haber respuesta, el puerto se marca como abierto|filtrado y Nmap sigue adelante.

Cuando se envía un paquete a un puerto UDP cerrado, el objetivo debería responder con un paquete ICMP (ping) que contiene un mensaje de que el puerto es inalcanzable. Esto identifica claramente los puertos cerrados, que Nmap marca como tales y sigue adelante.

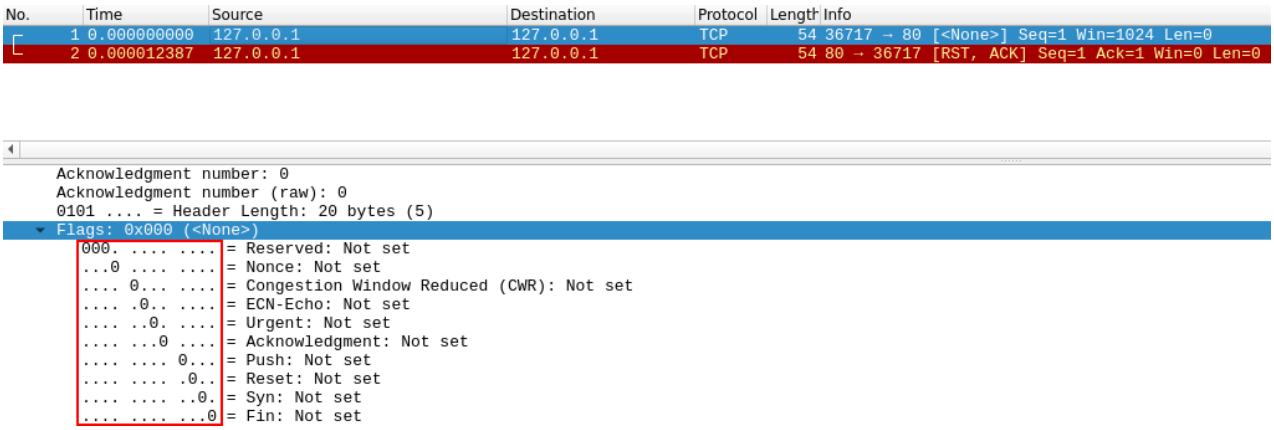
> **Entendemos:**
>
> Las conexiones UDP en lugar de iniciar una conexiòn con un handshake de ida y vuelta, se basan en el envio de paquetes a un puerto de destino y la Esperanza de que loo consigan.
>
> El interruptor para un sonde UDP de Nmp es (-sU).
>
> Cuando e envìa un paquete a un puerto UDP abierto, no deberia haber respuesta. Cuando esto ocurre, Nmap se refiere al puerto como open|filtered (abierto|filtrado), en otras palabras sospecha que el puerto esta abierto, pero podria estar protegido con un firewall.
>
> Cuando se envia un paquete a un puerto UDP cerrado, el objetivo deberia responder con un paquete ICMP (ping).

Null, FIN and Xmas

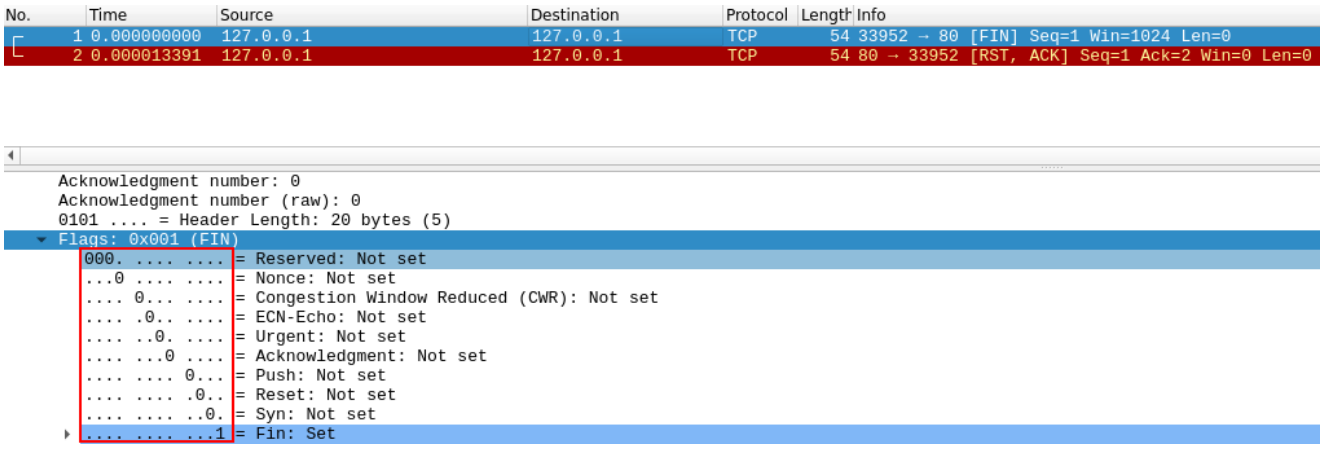
Los escaneos de puertos TCP NULL, FIN y Xmas son menos utilizados que cualquiera de los otros que ya hemos cubierto, por lo que no entraremos en una gran cantidad de profundidad aquí. Los tres están interrelacionados y se usan principalmente porque tienden a ser más sigilosos, relativamente hablando, que un escaneo "sigiloso" SYN. Empezando por los escaneos NULL:

- Como su nombre indica, los escaneos NULL (-sN) son cuando la petición TCP se envía sin **ningún tipo de bandera**. Según el RFC, el host de destino **debe responder con un RST** si el puerto está

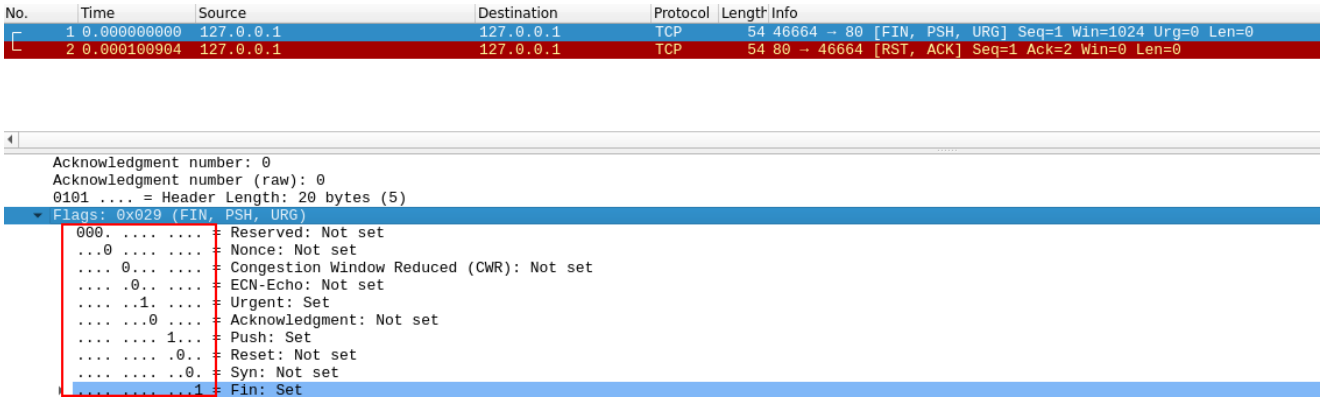
cerrado.



- Los escaneos FIN (-sF) funcionan de **forma casi idéntica**; sin embargo, en lugar de enviar un paquete completamente vacío, se **envía una petición con la bandera FIN** (normalmente utilizada para cerrar con gracia una conexión activa). Una vez más, **Nmap espera un RST si el puerto está cerrado**.



- Al igual que los otros dos escaneos de esta clase, los escaneos Xmas (-sX) envían un paquete **TCP malformado** y esperan una **respuesta RST** para los puertos cerrados. Se denomina escaneo de navidad porque las **banderas que establece (PSH, URG y FIN)** le dan la apariencia de un árbol de navidad parpadeante cuando se ve como una captura de paquetes en Wireshark.



La respuesta esperada para los puertos abiertos con estos escaneos también es idéntica, y es muy similar a la de un escaneo UDP. Si el puerto está abierto, no hay respuesta al paquete malformado. Desafortunadamente (al igual que con los puertos UDP abiertos), ese es también un comportamiento esperado **si el puerto está protegido por un firewall, por lo que los escaneos NULL, FIN y Xmas sólo identificarán los puertos como abiertos|filtrados, cerrados o filtrados**. Si un puerto se identifica como filtrado con uno de estos escaneos, normalmente es porque el objetivo ha respondido con un paquete ICMP inalcanzable.

También vale la pena señalar que **mientras el RFC 793 ordena que los hosts de la red respondan a los paquetes malformados con un paquete TCP RST para los puertos cerrados**, y no respondan en absoluto para los puertos abiertos; esto no siempre es el caso en la práctica. En particular, se sabe que **Microsoft Windows (y muchos dispositivos de red de Cisco) responden con un RST a cualquier paquete TCP malformado**, independientemente de si el puerto está abierto o no. Esto hace que todos los puertos aparezcan como cerrados.

Dicho esto, el objetivo aquí es, por supuesto, la evasión del cortafuegos. Muchos cortafuegos están configurados para descartar los paquetes TCP entrantes a los puertos bloqueados que tienen la bandera SYN activada (bloqueando así las solicitudes de inicio de nuevas conexiones). Enviando peticiones que no contengan la bandera SYN, se evita efectivamente este tipo de cortafuegos. Aunque esto es bueno en teoría, la mayoría de las soluciones de IDS modernas son inteligentes para estos tipos de escaneo, así que no confíes en que sean 100% efectivas cuando se trata de sistemas modernos.

ICMP Network Scanning

Al conectarnos por primera vez a una red de destino en una asignación de caja negra, nuestro primer **objetivo es obtener un**

"mapa" de la estructura de la red -- o, en otras palabras, queremos ver qué direcciones IP contienen hosts activos y cuáles no.

Una forma de hacerlo es utilizando Nmap para realizar el llamado "barrido de ping". Esto es exactamente como el nombre sugiere: Nmap envía un paquete ICMP a cada posible dirección IP de la red especificada. Cuando recibe una respuesta, marca la dirección IP que ha respondido como viva. Por razones que veremos en una tarea posterior, esto no siempre es exacto; sin embargo, puede proporcionar una línea de base y por lo tanto vale la pena cubrirlo.

Para realizar un barrido de ping, utilizamos el parámetro -sn junto con rangos de IP que pueden especificarse con un guión (-) o con la notación CIDR. Por ejemplo, podríamos escanear la red 192.168.0.x utilizando:

```
nmap -sn 192.168.0.1-254
```

o:

```
nmap -sn 192.168.0.1-254
```

El parámetro -sn le dice a Nmap que no analice ningún puerto, lo que le obliga a depender principalmente de los paquetes de eco ICMP (o de las peticiones ARP en una red local, si se ejecuta con sudo o directamente como usuario root) para identificar los objetivos. Además de las peticiones de eco ICMP, la opción -sn también hará que nmap envíe un paquete TCP SYN al puerto 443 del objetivo, así como un paquete TCP ACK (o TCP SYN si no se ejecuta como root) al puerto 80 del objetivo.

Overview

El motor de scripts de Nmap (NSE) es una adición increíblemente potente a Nmap, que amplía su funcionalidad de forma considerable. Los scripts de NSE están escritos en el lenguaje de programación Lua, y pueden usarse para hacer una gran variedad de cosas: desde el escaneo de vulnerabilidades, hasta la automatización de exploits para las mismas. El NSE es particularmente útil para el reconocimiento, sin embargo, vale la pena tener en cuenta lo extensa que es la biblioteca de scripts.

Hay muchas categorías disponibles. Algunas categorías útiles son:

- seguro:- No afectará al objetivo
- `intrusivo`:- No es seguro: puede afectar al objetivo
- `vuln`:- Búsqueda de vulnerabilidades
- `exploit`:- Explota una vulnerabilidad
- `auth`:- Intentar saltarse la autenticación de los servicios en ejecución (por ejemplo, entrar en un servidor FTP de forma anónima)
- `brute`:- Intentar forzar las credenciales de los servicios en ejecución
- `descubrimiento`:- Intentar consultar los servicios en ejecución para obtener más información sobre la red (por ejemplo, consultar un servidor SNMP).

Puede encontrar una lista más exhaustiva aquí.

En la siguiente tarea veremos cómo interactuar con el NSE y hacer uso de los scripts de estas categorías.

Working with the NSE

En la Tarea 3 vimos muy brevemente el interruptor `--script` para activar los scripts de NSE de la categoría vuln utilizando `--script=vuln`. No debería sorprender que las otras categorías funcionen exactamente de la misma manera. Si se ejecuta el comando `--script=safe`, entonces cualquier script seguro aplicable se ejecutará contra el objetivo (Nota: sólo se activarán los scripts que tengan como objetivo un servicio activo).

Para ejecutar un script específico, usaríamos `--script=<nombre-del-script>` , por ejemplo `-script=http-fileupload-exploiter`.

Se pueden ejecutar varios scripts simultáneamente de esta manera, separándolos con una coma. Por ejemplo: `--script=smb-enum-users,smb-enum-shares`.

Algunos scripts requieren argumentos (por ejemplo, credenciales, si están explotando una vulnerabilidad autenticada). Estos se pueden dar con el parámetro `--script-args` de Nmap. Un ejemplo de esto sería el script http-put (utilizado para subir archivos utilizando el

método PUT). Este script toma dos argumentos: la URL a la que se va a subir el fichero, y la ubicación del fichero en el disco. Por ejemplo:

```
nmap -p 80 --script http-put --script-args http-put.url='/dav/shell.php',http-put.file='./shell.php'
```

Tenga en cuenta que los argumentos se separan con comas y se conectan al script correspondiente con puntos (es decir, `<nombre del script>.<argumento>`).

Puede encontrar una lista completa de scripts y sus correspondientes argumentos (junto con ejemplos de uso) aquí.

Los scripts de Nmap vienen con menús de ayuda incorporados, a los que se puede acceder usando `nmap --script-help <nombre-del-script>`. Esto tiende a no ser tan extenso como en el enlace dado anteriormente, sin embargo, todavía puede ser útil cuando se trabaja localmente.