
# Módulo 2: Finger printing and grabbing.

Mauro Céspedes Araya

Hacking y Pentesting con Python

En primer lugar, se levantan dos máquinas virtuales: una con **Kali** y otra con **Metasploit**. Se podrán ver ambas maquinas a través de una conexión **Bridge**. Para comprobar que existe comunicación se realiza un simple ping. Para realizar el ping, se debe antes obtener la dirección IP por medio del comando **ifconfig** (ipconfig en windows).

Esta la ip de mi maquina objetivo(metasploit):

```
root@metasploitable:/# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:3f:5b:1e
          inet addr:192.168.1.150  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe3f:5b1e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5022 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3480 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:400810 (391.4 KB)  TX bytes:268877 (262.5 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:644 errors:0 dropped:0 overruns:0 frame:0
          TX packets:644 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:289733 (282.9 KB)  TX bytes:289733 (282.9 KB)

root@metasploitable:/#
```

Ahora se continúa trabajando con Metasploit desde la maquina **Kali**. Metasploit es un marco de trabajo de código abierto basado en RUBY. Se utiliza para validar las vulnerabilidades de un sistema.

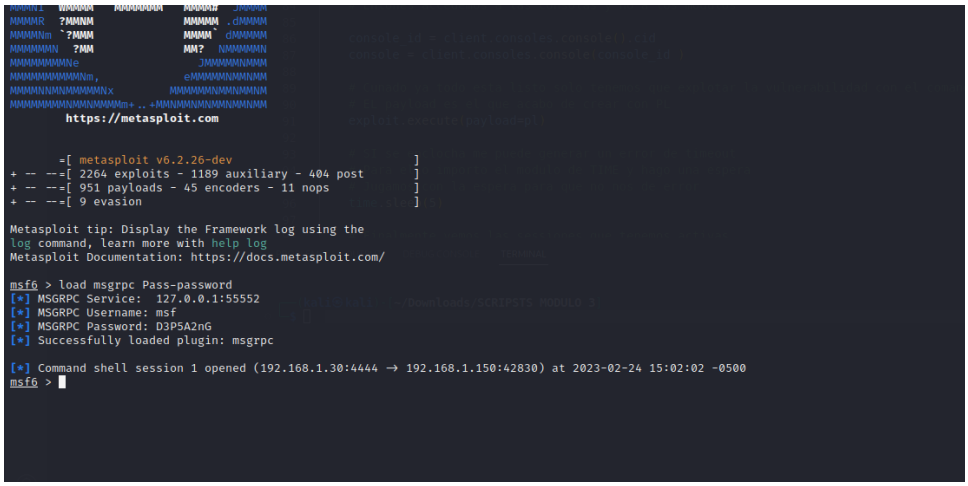
Aquí es importante diferenciar entre **explotación** y **payload**. El payload es lo que se ejecuta en la máquina objetivo al explotar las vulnerabilidades. Es decir, las instrucciones que se desean una vez dentro del sistema.

Ahora se escanean los puertos con **NMAP** para saber las diferentes vulnerabilidades que se pueden explotar. De esta manera se tiene una idea de cuales se pueden incluir dentro del script de Python más adelante.

*nmap -sV -Pn 192.168.1.150*

```
FreeBSD 8.0/7.3/7.2 i386, ProFTPD 1.3.2a/e/c Server (binary)
Debian GNU/Linux 5.0, ProFTPD 1.3.2e Server (Plesk binary)
220 (vsFTPd 2.3.4)
Mandrake Linux 7.1 (apache-1.3.14-2)
Samba smbd 3.X (workgroup:MYGROUP)
Cyrus pop3d 2.3.7-Invoca-RPM-2.3.7-7.el5_6.4
MiniServ 0.01 (Webmin httpd)
OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
Linux telnetd
Postfix smtpd
ISC BIND 9.4.2
Apache httpd 2.2.8 ((Ubuntu) DAV/2)
Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
exec?
OpenBSD or Solaris rlogind
tcpwrapped
GNU Classpath grmiregistry
Metasploitable root shell
2-4 (RPC #100003)
ProFTPD 1.3.1
MySQL 5.0.51a-3ubuntu5
PostgreSQL DB 8.3.0 - 8.3.7
VNC (protocol 3.3)
(access denied)
UnrealIRCd
Apache Jserv (Protocol v1.3)
Apache Tomcat/Coyote JSP engine 1.1
```

Ahora se necesita levantar el servicio de **Metasploit** y activar el **plugin** para utilizarlo en el sript de Python. Co esto se obtiene el puerto del servicio y la contraseña para poder utilizarlo. Se levanta el servicio y se le indica a Metasploit como trabajar.

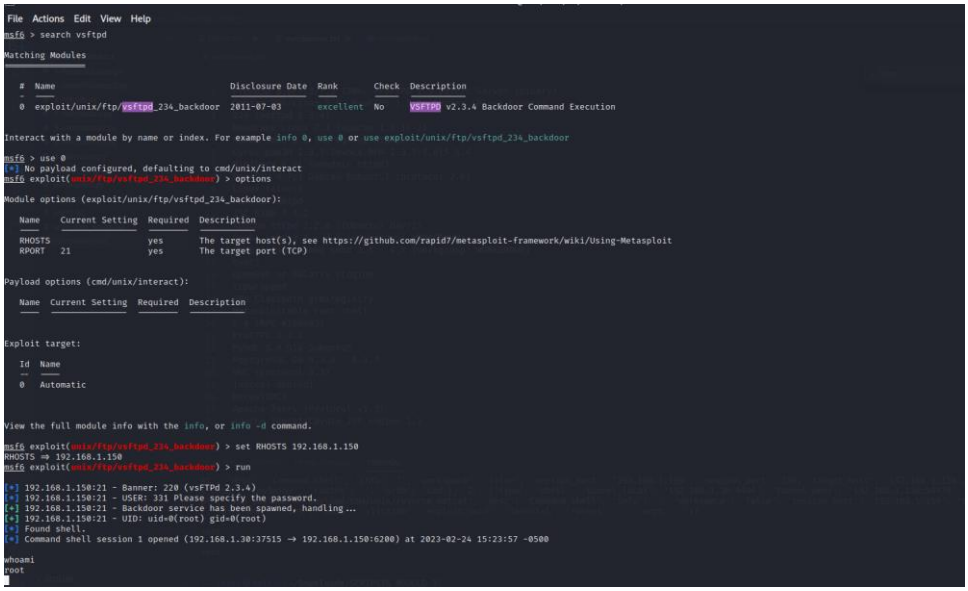



Ahora se necesita escoger el tipo de vulnerabilidad a explotar de la lista que se obtuvo con **nmap**.

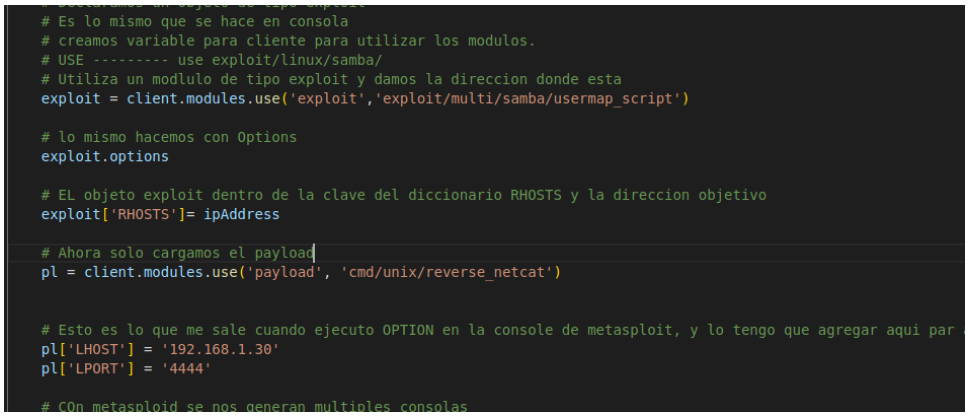
Cuando se quiere explotar una vulnerabilidad se necesita la ruta para ejecutarla dentro del sistema objetivo. Es decir, la ruta exacta en donde se encuentra la vulnerabilidad. Se puede hacer una búsqueda en Metasploit para averiguar esta información y sacar que se necesita para ejecutarla.

Este proceso en Metasploit se quiere realizar directamente con un script en Python utilizando la librería **pymetasploit3**. La comunicación de GSG RPC se localizará dentro del módulo **mfs.rpc.**

En este caso se utilizaron los ejemplos expuestos en clase por su sencillez, ya que solo es requerido el RHOST y el puerto. Las vulnerabilidades que se explotan son **SAMBA** y **vsftp_234_backdoor**.




Una vez identificada la ruta en la máquina virtual de **Metasploitable2** se procede a incluirla en el script de Python.




El script funciona con un método que sirve para establecer la conexión con el servicio de Metasploit. Como parámetros recibe el **puerto** donde se levanta el servicio y la **contraseña**. Con esta información el método regresa un **cliente**.

Se recibe la información que se ingresa por pantalla para ejecutar el **cliente** recién creado. Pueden pasar muchas cosas, entonces todo se ejecutará dentro de un bucle **if**. En caso de que si exista un cliente activo se declara un objeto de tipo **exploit.** Todo este procedimiento es lo mismo que hacerlo directo desde la consola de Metasplot.

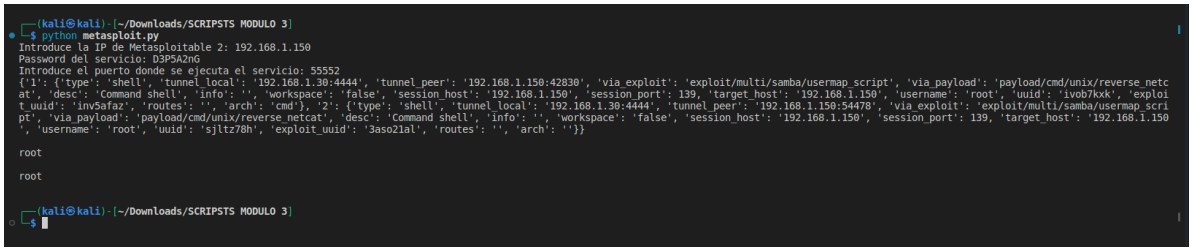
Se crea una variable para el cliente y que pueda utilizar los módulos. Con el método "**use**" se indica la ruta en donde está la vulnerabilidad. Ahora se utiliza "**options**" para indicarle el objeto **exploit** dentro de la clave del diccionario RHOSTS y la dirección objetivo.

Ya con esta información se puede indicar el "**payload**" que se desea cargar a la maquina explotada. En este caso, solo se hará un payload "dummy" a manera de prueba. Se agrega también la información sobre IP y puerto que sale al ejecutar **option** en la consola de Metasploit. Esta información es necesaria para poder realizar la conexión.

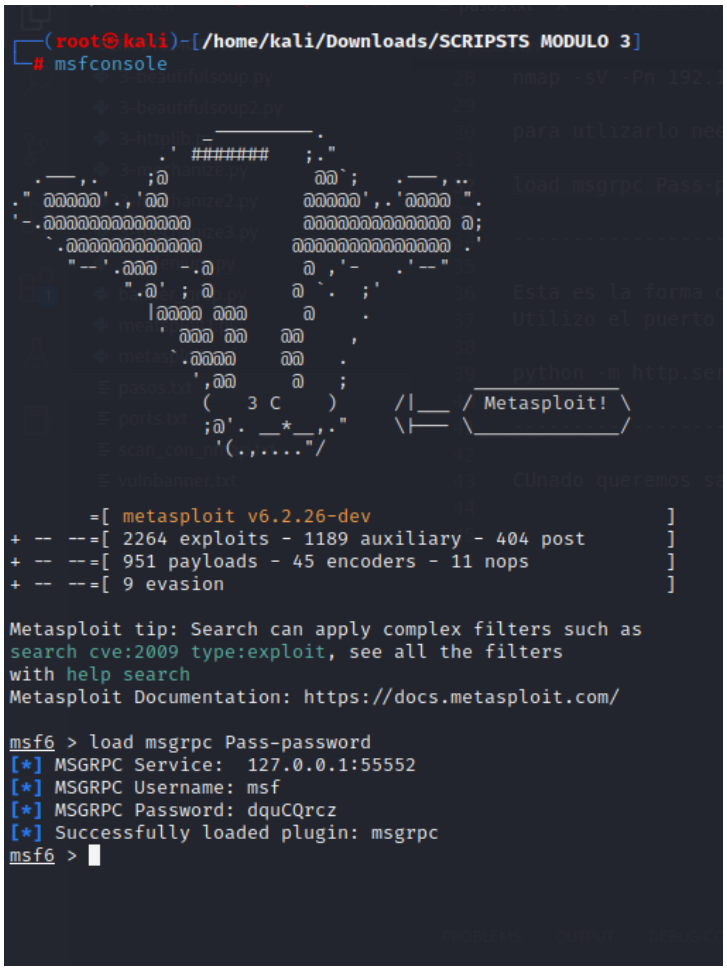
Con Metasploit se generan múltiples consolas y es necesario tener un ID para poder acceder a cada una. De esta forma se puede acceder las consolas y no a los módulos.

Cuando ya todo está listo solo se debe de explotar la vulnerabilidad con el comando **EXPLOIT**, y se le asigna el **payload** que se quiere ejecutar. Algunas veces puede tardar un poco, por eso es bueno agregar un timeout con la librería Time. Se juega con la espera para que evitar este timpo de errores.

Finalmente, dentro del script se imprime en pantalla la información sobre las sesiones activas. Es igual que en consola, cuando se lanza el ataque se tiene una sesión activa y el resultado de la ejecución del **shell** en la maquina objetivo. El siguiente es el resultado de explotar la vulnerabilidad **Samba**.




La siguiente es la informacion de la segunda explotación , la cual utilizo la vulnerabilidad de **vsftpd**.




Adjunto a la terea está el código del script de Python con la descripción del funcionamiento de cada línea de código.