School of Hacking

Taller 3:

Herramientas básicas del hacker: Metasploit y Armitage

Antonio Díaz & José Antonio Gómez, 2015

UGRCyS
UGR Cyber Security Group

Índice

¿Qué es Metasploit?

- **Metasploit Framework**: herramienta open source para el desarrollo y ejecución de exploits contra una máquina remota.
- Versiones (www.metasploit.com - todas tienen como base MSF):
  - **Metasploit Community** – interfaz web para pruebas de instrusión.
  - **Metasploit Express** – edición profesional open-core para verificar vulnerabilidades (GUI, integra nmap, recolección de evidencias automática y de fuerza bruta).
  - **Metasploit Pro** – edición comercial open-core para pentesters (Express + escaneo y explotación de aplicaciones web, ingeniería social y VPN pivoting)

Terminología

- **Exploit** – Medio por el cual el atacante aprovecha una debilidad/vulnerabilidad de la red, aplicación o servicio.
- **Payload** – Programa o código que se traspasa a la víctima. Metasploit tiene payloads pre-diseñadas y permite construir otras propias.
- **Shellcode** – conjunto de instrucciones utilizadas como payload cuando se produce la explotación.
- **Módulo** – pieza de software intercambiable utilizada por Metasploit. Pueden ser módulos exploit o auxiliares.
- **Listener** – Componente que escucha las conexiones del sistema atacante al sistema objetivo.
- **Show** – orden que permite grabar un listados de todos los módulos, opciones, objetivos, etc. en el framework.

Arquitectura de Metasploit

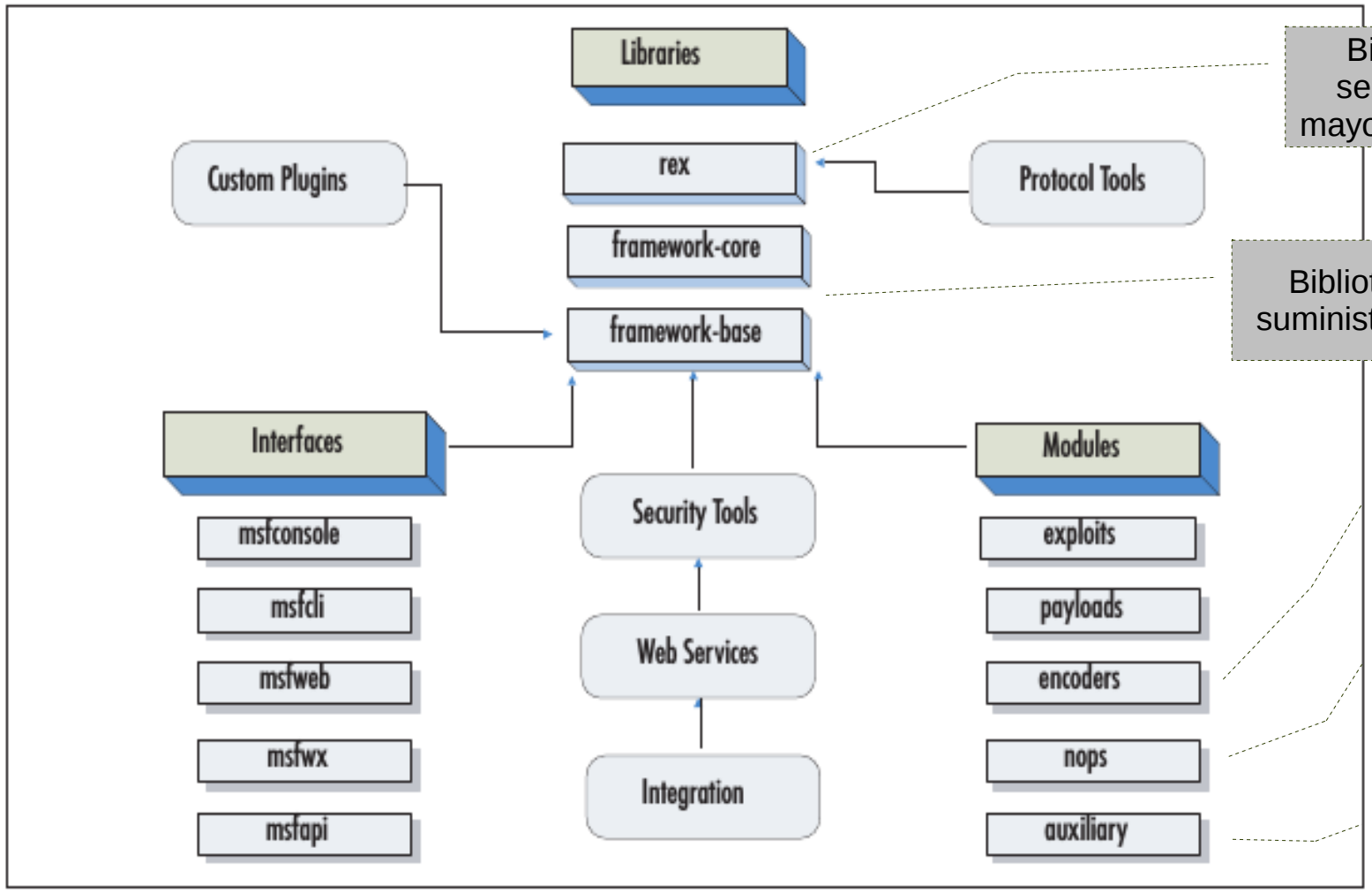


Figura extraída de D. Mayor et al.

MSF: interfaces

- Metasloit dispone de varias interfaces con las que interaccionar con el framework:
  - **Msfconsole** – consola desde la cual podemos utilizar todas las opciones.
  - **Armitage** – Interfaz gráfica para el framework.
  - **Msfweb** – web UI que permite gestionar el framework de manera remota.
  - **Msfcli** – pensada para automatizar la explotación no permite una interacción directa (sin lanzar msfconsole).
  - **Msfopcode** – base de datos de opcodes
  - **Msfpayload** – podemos cambiar los payloads mediante parámetros en la línea de órdenes y obtener la salida en C, Perl o Raw.
  - **Msencoder** – da acceso directo al encoder de payload dentro del framwork.
  - **Msfd** – abre una interfaz de red para msfconsole

MSF: exploits

- Los exploits puede ser:
  - **Activos** – explotan un host específico, se ejecutan hasta terminar.
  - **Pasivos** – esperan por un host entrante y lo explotan al conectarse (enfocados a clientes como navegadores web, ftp, ...)

MSF: payload

- Los tipos de payloads:
  - **Shell** – permite ejecutar scripts y órdenes contra el objetivo.
  - **Meterpreter** -permite controlar la pantalla de un dispositivo utilizando VNC y navegar, subir y descargar archivos.
  - **Payload dinámicas** – permite evadir defensas anti-virus generando payload singulares.

msfconsole

- Lanzamos la orden `msfconsole` desde el terminal que nos devolverá el identificador `msf >` para introducir las órdenes.
- Previamente será bueno actualizarlo: `root@kali.~# msfupdate`
- La consola MSF es como un mini-sistema de archivos donde las carpetas que cuelgan de él se encuentran físicamente en la ruta dónde se ha instalado el framework. Ejemplos:
  - Los exploits de Windows esta en la ruta `exploit/windows/<..>`
  - Los módulos auxiliares en `auxiliary/<...>`
  - Los encoders en `encoders/<tecnologia>`

MSF: órdenes básicas

- *Órdenes de ayuda*:
  - `msf > help` – lista las órdenes separadas en dos listados:
    - Órdenes del núcleo de msf
    - Órdenes de interacción con bases de datos.
  - `-h` permite obtener información de órdenes concretas.
- *Orden de búsqueda* – útil para la búsqueda de módulos por alguna característica o determinar si el framework esta actualizado.
    - `msf > search -h`

MSF: órdenes básicas (ii)

- `info` – aporta información sobre el módulo seleccionado bien con la orden `use` (que permite seleccionar un módulo), bien especificando la ruta:
  - `msf > use exploit/multi/handler`
  - `msf > info`
  - `msf > info <ruta>`
- `show` – muestra las diferentes opciones para los módulos del framework, exploits, payload, encoders, nops, etc.

MSF: órdenes básicas (iii)

- Órdenes de interacción y configuración:
  - `back` – permite salir del módulo (contrario a `use`)
  - `set` y `setg` – asignan valores a variables: set para un módulo, setg para en contexto del framework.
  - `unset` y `unsetg` – desasignan valores a parámetros o variables
  - `connect` – permite conectarnos a otra máquina para su gestión o administración dado la dirección IP y el puerto.
  - `irb` – permite ejecutar un interprete de Ruby para el framework para ejercutar órdenes y scripts.
  - `load`, `unload` y `loadpath` – load/unload especifica el plugins a cargar/descargar, o directorio donde se almacenan (loadpath)

MSF: órdenes básicas (iv)

- `Check` – permite verificar si un sistema es vulnerable a cierta vulnerabilidad antes de lanzar el script.
- `Exploit` – lanza el código malicioso, una vez seleccionado y configurado el módulo, sobre la máquina, o prepara el entorno para vulnerar la máquina. Devuelve el control mediante un shell o un *Meterpreter*[1]. Opciones:
  - `-j` ejecutar exploit en segundo plano
  - `-z` no se interactúa con la sesión tras explotación exitosa
  - `-e` lanza el paylod con al codificación establecida

[1] *Meta-interprete*: payload que permite cargar e inyectar en un programa del sistema atacado las extensiones que hemos desarrollado en formato .dll

MSF: órdenes básicas (v)

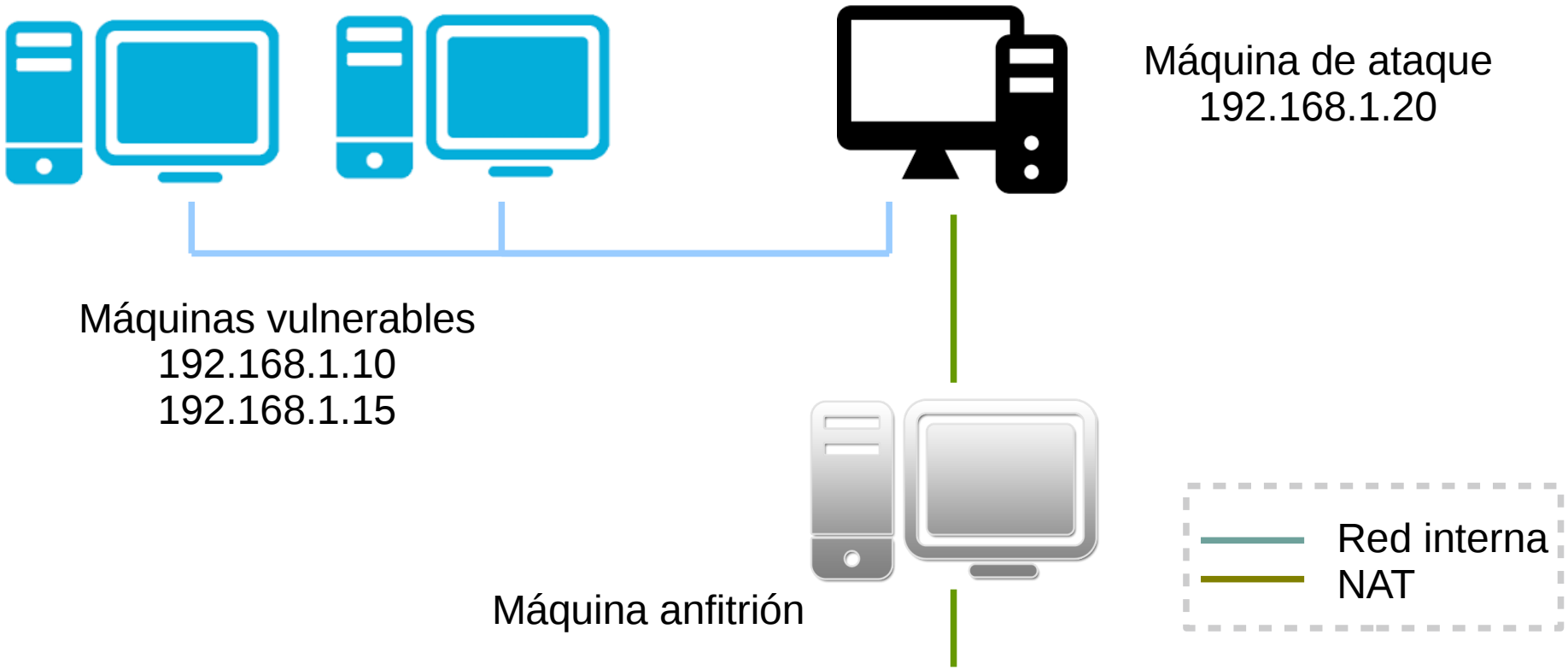
- `Sessions` – las shells obtenidas en sistemas vulnerados se organizan por sesiones. Esta orden permite ver las sesiones que tenemos abiertas:
  - `-l` lista sesiones disponibles
  - `-v` muestra información extra
  - `-s <script>` ejecuta script sobre todas las sesiones del Meterpreter
  - `-K` finaliza todas las sesiones abiertas
  - `-c <orden>` ejecutar órdenes sobre sesiones abiertas del meterpreter
  - `-u` permite actualizar la shell remota tipo Win32 a un meterpreter especificando la sesión.
  - `-i` especifican sesión con la que interaccionar.

MSF: órdenes básicas (vi)

- `Resource` – permite la carga de un archivo (*.rc*) con acciones especificas sobre el framework para automatizar tareas.
- `Makerc` – almacena en un archivo el historial de órdenes y acciones que se han realiza en la sesión en curso (*nombre-usuario* en el directorio *.msf3*)
- `Save` – aporta persistencia a la configuración del entorno, especialmente es test complicados y largos (archivo config en .msf3).
- `Jobs` – muestra/finaliza los módulos en ejecución en segundo plano
- `Run` – permite ejecutar un módulo auxiliar cargado en el contexto de la consola.
- `Route` – enruta sockets a sesiones (similar al `route` de Linux). Útil en *pivoting*.

Entorno de trabajo

- Para los ejemplos que vamos a ver hemos creado el siguiente entorno de trabajo en Virtualbox:



Configuración VMs

- Maquina vulnerable: http://sourceforge.net/projects/metasploitable/
  - Conexion de red: eth0 – interna

```
ifconfig eth0 192.168.1.10 up
```

  - Login/passwd: `msfadmin / msfadmin`
- Maquina atacante – kali
  - Conexión de red: eth0 – NAT y eth1 – interna

```
ifconfig eth1 192.168.1.20 up
```

- Otras distribuciones Linux vulnerables: https://www.vulnhub.com/
- Distribuciones de Windows: https://www.modern.ie/en-us/virtualization-tools#downloads

Pasos

- Tras arrancar la consola MSF Framework: msfconsole.
- Seleccionamos el exploit: `use <exploit>`
  - Metasploit no es especialmente aconsejable para indicar las vulnerabilidades de un host, debemos utilizar otra herramienta:
    - Un análisis de vulnerabilidades que nos permitirá usar un exploit específico (OpenVas).
    - Un escaneador de puertos que nos permitirá explotar puertos abiertos probando todos los exploits.
  - Seleccionar el payload: `set PAYLOAD <payload>`
  - Seleccionar las opciones: `show options`
  - Explotar: `exploit`

Bibliografía

- Pablo González Perez, *Metasploit para Pentesters, 3ª Ed.*, 0xWord, 2014.
- M. Agarwal and A. Singh, *Metasploit Penetration Testing Cookbook, 2nd Ed.*, PACKT Open Source, 2013.
- D. Kennedy, et al, *Metasploit. The Penetration Tester's Guide*, No Starch Press, 2011.
- D. Mayor et al, *Metasploit Toolkit for Penetration Testing, Exploit Development, and Vulnerability Research*, Syngress, 2007.
- N. Jaswal, *Mastering Metasploit*, Packt Open Source, 2014.
- A. Balapure, *Learning Metasploit Exploitation and Development*, Packt Open Source, 2013.