

HERRAMIENTAS DE AUDITORIA

1) Netcat — chat y transferencia de fichero (TCP simple)

Objetivo: entender la comunicación TCP básica cliente/servidor y verificar integridad de fichero.

Preparación (lab):

- VM-A (receptor): 192.168.56.10
- VM-B (emisor): 192.168.56.11
- prueba.txt en VM-B (pequeño texto de prueba).

Comandos (copiar/pegar en el laboratorio):

En VM-A (receptor):

```
# escuchar y guardar en recibido.txt
```

```
nc -l -p 4444 > recibido.txt
```

```
msfadmin@metasploitable:/tmp$ nc -lvpn 4444
listening on [any] 4444 ...
connect to [192.168.0.120] from (UNKNOWN) [192.168.0.28] 59698
hola
hi
msfadmin@metasploitable:/tmp$ nc -lvpn 4444
listening on [any] 4444 ...
```

```
~/Documents/CENECC main > nc 192.168.0.120 4444
hola
adioooooosssss

como estas
|
```

```
msfadmin@metasploitable:/tmp$ nc -l -p 4444 > lorena.txt
msfadmin@metasploitable:/tmp$ ls
4534.jsvc_up      lorena.elf      lornevirus.txt    shell32.elf      shellold.elf3
gconfd-msfadmin  lorena.txt      orbit-msfadmin   shellold.elf2
msfadmin@metasploitable:/tmp$ cat lorena.txt
hola
caracola
esto es
nuevoooo

in
,.-'0=_---' i.,,-;:_\n\m

msfadmin@metasploitable:/tmp$
```

```
~/Documents/box > nc 192.168.0.120 4444
hola
caracola
esto es
nuevoooo
```

```
|n
,.-'0=_—' i.,-;:_\m^[[D^[[D
^C
```

En VM-B (emisor):

```
# enviar fichero
```

```
nc 192.168.56.10 4444 < prueba.txt
```

```
nc -l -p 4444 > prueba.txt
ls
lornevirus.txt  prueba.txt
orbit-msfadmin  shell32.elf
```

```
~/Documents/box > nc 192.168.0.120 4444 < prueba.txt
```

Solución / verificación (en VM-A o VM-B):

```
sha256sum prueba.txt recibido.txt
```

Salida esperada (ejemplo):

```
e3b0c44298fc1c149afbf4c8996fb924... prueba.txt
```

```
e3b0c44298fc1c149afbf4c8996fb924... recibido.txt
```

```
c
~/Documents/box > sha256sum prueba.txt
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855  prueba.txt
~/Documents/box > |
```

Si los hashes coinciden, el fichero se transfirió correctamente.

Nota pedagógica:

- Explica que nc crea una conexión TCP simple en el puerto indicado.
- Señala que si la red tiene firewalls/iptables, el puerto debe estar permitido en el laboratorio.
- Si quieras practicar recepción múltiple, usa nc -l -p 4444 > recibido_(date +%s).txt para no sobrescribir.

2) nbtscan — descubrimiento NetBIOS

Objetivo: listar equipos con NetBIOS/SMB y entender la información visible.

Preparación: en la red de laboratorio tener al menos un equipo con Samba o Windows, por ejemplo 192.168.56.20.

Comando:

```
sudo nbtscan 192.168.56.0/24
```

Salida esperada (ejemplo):

```
192.168.56.20  WIN-VM20      WORKGROUP192.168.56.21  WINDOWS-PC    WORKGROUP
```

Interpretación (solución):

- La primera columna es la IP.
- La segunda columna es el nombre NetBIOS del equipo (identificador local).
- La tercera columna es el grupo de trabajo o dominio NetBIOS.
- Con esta info puedes identificar equipos que publican servicios Samba/Windows en la LAN.

```
c  
~/Documents/box > sudo nbtscan 192.168.0.0/24  
[sudo] password for kali:  
Doing NBT name scan for addresses from 192.168.0.0/24  


| IP address    | NetBIOS Name    | Server   | User           | MAC address       |
|---------------|-----------------|----------|----------------|-------------------|
| 192.168.0.5   | RNP00267399578C | <server> | <unknown>      | 00:26:73:99:57:8c |
| 192.168.0.21  | DESKTOP-1FR64MT | <server> | <unknown>      | 6c:62:6d:8d:fd:36 |
| 192.168.0.24  | PC-A03          | <server> | <unknown>      | 34:17:eb:c4:6c:81 |
| 192.168.0.42  | PC-SERGIO       | <server> | <unknown>      | 2c:41:38:ac:7d:1c |
| 192.168.0.46  | PC-A02          | <server> | <unknown>      | 98:ee:cb:46:94:f7 |
| 192.168.0.49  | DESKTOP-M7CM9BE | <server> | <unknown>      | dc:4a:3e:7c:25:3e |
| 192.168.0.38  | METASPLOITABLE  | <server> | METASPLOITABLE | 00:00:00:00:00:00 |
| 192.168.0.88  | PC-A04          | <server> | <unknown>      | 78:ac:c0:bd:a1:59 |
| 192.168.0.41  | DESKTOP-MAQCNB1 | <server> | <unknown>      | b8:8a:60:9f:c7:11 |
| 192.168.0.89  | PA1-LINKSYS     |          | <unknown>      | c0:56:27:eb:b6:7f |
| 192.168.0.120 | METASPLOITABLE  | <server> | METASPLOITABLE | 00:00:00:00:00:00 |

  
~/Documents/box > |
```

Alternativa si nbtscan no está disponible:

```
smbclient -L 192.168.56.20 -N  
nmap -p 137 --script nbstat.nse 192.168.56.0/24
```

```
~/Documents/box > smbclient -L 192.168.0.5 -N  
  
Sharename      Type      Comment  
_____         _____  
MPC2003       Printer  
IPC$          IPC  
Reconnecting with SMB1 for workgroup listing.  
  
Server        Comment  
_____         _____  
  
Workgroup     Master  
_____         _____  
  
~/Documents/box > smbclient -L 192.168.0.21 -N  
do_connect: Connection to 192.168.0.21 failed (Error NT_STATUS_IO_TIMEOUT)  
~/Documents/box > smbclient -L 192.168.0.24 -N  
session setup failed: NT_STATUS_ACCESS_DENIED  
~/Documents/box > smbclient -L 192.168.0.38 -N  
Anonymous login successful  
  
Sharename      Type      Comment  
_____         _____  
print$         Disk      Printer Drivers  
tmp            Disk      oh noes!  
opt             Disk  
IPC$           IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))  
ADMIN$         IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))  
Reconnecting with SMB1 for workgroup listing.  
Anonymous login successful  
  
Server        Comment  
_____         _____  
  
Workgroup     Master  
_____         _____  
WORKGROUP
```

Puntos de discusión para clase:

- NetBIOS puede filtrar metadatos que ayudan a un atacante a mapear la red.
- Medidas: deshabilitar servicios innecesarios, uso de firewalls, segmentación.

```
~/Documents/box > nmap --script smb-enum-shares -p445 192.168.0.38
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-22 14:19 CEST
Nmap scan report for 192.168.0.38
Host is up (0.0021s latency).

PORT      STATE SERVICE
445/tcp    open  microsoft-ds
MAC Address: 08:00:27:E3:6A:E9 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Host script results:
| smb-enum-shares:
|   account_used: <blank>
|   \\192.168.0.38\ADMIN$:
|     Type: STYPE_IPC
|     Comment: IPC Service (metasploitable server (Samba 3.0.20-Debian))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: <none>
|   \\192.168.0.38\IPC$:
|     Type: STYPE_IPC
|     Comment: IPC Service (metasploitable server (Samba 3.0.20-Debian))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
|   \\192.168.0.38\opt:
|     Type: STYPE_DISKTREE
|     Comment:
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: <none>
|   \\192.168.0.38\print$:
|     Type: STYPE_DISKTREE
|     Comment: Printer Drivers
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\var\lib\samba\printers
|     Anonymous access: <none>
|   \\192.168.0.38\tmp:
|     Type: STYPE_DISKTREE
|     Comment: oh noes!
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
```

INSTALACIÓN DE OPENVAS EN KALI LINUX

1) Preparar el sistema

```
sudo apt update && sudo apt upgrade -y
```

```
sudo apt install -y gvm
```

Si el paquete gvm no existe en tu repositorio, prueba sudo apt install -y openvas (aunque lo recomendable en Kali moderno es gvm).

2) Configuración inicial (sincronizar feeds y crear certificados)

Hay un script que automatiza la puesta en marcha:

```
sudo gvm-setup
```

Ese script:

- crea certificados,
- descarga las firmas (NVTs),
- crea el usuario admin,
- arranca/activa servicios.

```
Upgrading. 0, Installing. 0, Removing. 0, Not Upgrading. 0
~ > sudo gvm-setup
This script is provided and maintained by Debian and Kali.
If you find any issue in this script, please report it directly to Debian or Kali

[>] Starting PostgreSQL service
[>] Creating GVM's certificate files
[>] Creating PostgreSQL database
[i] User _gvm already exists in PostgreSQL
[i] Database gvmd already exists in PostgreSQL
[i] Role DBA already exists in PostgreSQL

[*] Applying permissions
NOTICE: role "_gvm" has already been granted membership in role "dba" by role "postgres"
GRANT ROLE
[i] Extension uuid-ossp already exists for gvmd database
[i] Extension pgcrypto already exists for gvmd database
[i] Extension pg-gvm already exists for gvmd database
[>] Migrating database
[>] Checking for GVM admin user
[*] Configure Feed Import Owner
```

4) Crear usuario administrador (si no lo creó gvm-setup)

```
sudo gvmd --create-user=admin --password='password'
```

(O para cambiar la contraseña existente)

```
sudo gvmd --user=admin --new-password='TuNuevaPass'
```

5) Iniciar y habilitar servicios (systemd)

Servicios principales: ospd-openvas (daemon), gvmd (manager), gsad (web UI).

Arranca y habilita:

```
sudo systemctl enable --now ospd-openvas
```

```
sudo systemctl enable --now gvmd
```

```
sudo systemctl enable --now gsad
```

```
sudo: a password is required
~/Documents/IFCT0109 > sudo systemctl enable --now ospd-openvas
[sudo] password for kali:
sudo systemctl enable --now gvmd
[sudo] password for kali:
Created symlink '/etc/systemd/system/multi-user.target.wants/ospd-openvas.service' → '/usr/lib/
systemd/system/ospd-openvas.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/gvmd.service' → '/usr/lib/systemd/
system/gvmd.service'.
Created symlink '/etc/systemd/system/greenbone-security-assistant.service' → '/usr/lib/systemd/
system/gsad.service'.
Created symlink '/etc/systemd/system/multi-user.target.wants/gsad.service' → '/usr/lib/systemd/
system/gsad.service'.
```

Comprobar estado:

```
sudo systemctl status ospd-openvas gvmd gsad
```

```
~/Documents/IFCT0109 > sudo systemctl status ospd-openvas gvmd gsad
● ospd-openvas.service - OSPD Wrapper for the OpenVAS Scanner (ospd-openvas)
   Loaded: loaded (/usr/lib/systemd/system/ospd-openvas.service; enabled; preset: disabled)
   Active: active (running) since Mon 2025-10-27 12:52:03 CET; 16s ago
     Invocation: f41c196815ab48d099f61dc6eef668c9
       Docs: man:ospd-openvas(8)
              man:openvas(8)
    Main PID: 30912 (ospd-openvas)
      Tasks: 5 (limit: 9286)
     Memory: 182.9M (peak: 225.8M)
        CPU: 4.635s
      CGroup: /system.slice/ospd-openvas.service
              └─30912 /usr/bin/python3 /usr/bin/ospd-openvas --config /etc/gvm/ospd-openvas.con
                  ├─30914 /usr/bin/python3 /usr/bin/ospd-openvas --config /etc/gvm/ospd-openvas.con

Oct 27 12:52:01 kali systemd[1]: Starting ospd-openvas.service - OSPD Wrapper for the OpenVAS
Oct 27 12:52:03 kali systemd[1]: Started ospd-openvas.service - OSPD Wrapper for the OpenVAS S

● gvmd.service - Greenbone Vulnerability Manager daemon (gvmd)
   Loaded: loaded (/usr/lib/systemd/system/gvmd.service; enabled; preset: disabled)
   Active: active (running) since Mon 2025-10-27 12:52:07 CET; 12s ago
     Invocation: a8a34255fc3a434981cc9979d7c5ecd8
       Docs: man:gvmd(8)
```

6) Comprobar instalación

Hay una utilidad de comprobación:

```
sudo gvm-check-setup
```

```

~/Documents/IFCT0109 > sudo gvm-check-setup
gvm-check-setup 25.04.0
  This script is provided and maintained by Debian and Kali.
  Test completeness and readiness of GVM-25.04.0
Step 1: Checking OpenVAS (Scanner)...
  OK: OpenVAS Scanner is present in version 23.23.1.
  OK: Notus Scanner is present in version 22.7.2.
  OK: Server CA Certificate is present as /var/lib/gvm/CA/servercert.pem.
Checking permissions of /var/lib/openvas/gnupg/*
  OK: _gvm owns all files in /var/lib/openvas/gnupg
  OK: redis-server is present.
  OK: scanner (db_address setting) is configured properly using the redis-server socket
/var/run/redis-openvas/redis-server.sock
  OK: the mqtt_server_uri is defined in /etc/openvas/openvas.conf
  OK: _gvm owns all files in /var/lib/openvas/plugins
  OK: NVT collection in /var/lib/openvas/plugins contains 94791 NVTs.
  OK: The notus directory /var/lib/notus/products contains 506 NVTs.
Checking that the obsolete redis database has been removed
  OK: No old Redis DB
  OK: ospd-openvas service is active.
  OK: ospd-OpenVAS is present in version 22.9.0.

```

Lee la salida: te indicará cosas faltantes (por ejemplo feeds no sincronizados, permisos, certificados). Si la salida sugiere ejecutar comandos concretos, hazlo.

7) Acceder a la interfaz web

Abre en el navegador (local):

<https://127.0.0.1:9392>

- Usuario: admin
- Contraseña: la que creaste con gvmd --create-user o la que indicó gvm-setup.

The screenshot shows the Greenbone Network Scanner web interface. The URL in the address bar is <https://127.0.0.1:9392/dashboards>. The top navigation bar includes the Greenbone logo, user info (admin), and a timestamp (14:47). On the left, a sidebar menu lists: Dashboards, Scans, Assets, Resilience, Security Information, Configuration, Administration, and Help. The main content area features a 'Dashboards' section with tabs for 'Overview' (selected) and 'Tasks'. A prominent message box states: 'Feed is currently syncing. Please wait while the feed is syncing. Scans are not available during this time. For more information, visit the [Documentation](#)'. Below this, there are two cards: 'Tasks by Severity Class (Total: 0)' and 'Tasks by Status (Total: 0)'.

Se tiene que cambiar la clave con la siguiente instrucción.

```
(kali㉿kali)-[~]
$ sudo runuser -u _gvm -- gvmd --user=admin --new-password=tukey1
```

El sitio usa HTTPS con certificado auto-firmado. Acepta la excepción de seguridad en el navegador.

A screenshot of the Greenbone Security Assistant web interface. The URL in the address bar is 127.0.0.1:9392/tasks. The sidebar on the left has 'Tasks' selected. The main area shows a summary: 'Tasks 0 of 0'. Below this are three tabs: 'Tasks by Severity Class (Total: 0)', 'Tasks with most High Results per Host', and 'Tasks by Status (Total: 0)'. The top right corner shows the user 'admin'.

8) Actualizar manualmente después (si quieres refrescar feeds)

```
sudo greenbone-nvt-sync
```

```
sudo greenbone-scapdata-sync
```

```
sudo greenbone-certdata-sync
```

Y luego reinicia servicios si es necesario:

```
sudo systemctl restart ospd-openvas gvmd gsad
```

```
~/Documents/IFCT0109 > sudo greenbone-nvt-sync
sudo greenbone-scapdata-sync
sudo greenbone-certdata-sync
Running as root. Switching to user '_gvm' and group '_gvm'.
Trying to acquire lock on /var/lib/openvas/feed-update.lock
Acquired lock on /var/lib/openvas/feed-update.lock
: Downloading Notus files from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/vt-data/notus/ to
/var/lib/notus
: Downloading NASL files from
rsync://feed.community.greenbone.net/community/vulnerability-feed/24.10/vt-data/nasl/ to
/var/lib/openvas/plugins
```

9) Logs y resolución de problemas

- Ver logs systemd:

```
sudo journalctl -u ospd-openvas -f
```

```
~/Documents/CENEC main > sudo journalctl -u ospd-openvas -f
[sudo] password for kali:
Oct 27 12:52:01 kali systemd[1]: Starting ospd-openvas.service - OSPD Wrapper for the OpenVAS S
canner (ospd-openvas)...
Oct 27 12:52:03 kali systemd[1]: Started ospd-openvas.service - OSPD Wrapper for the OpenVAS Sc
anner (ospd-openvas).
```

```
sudo journalctl -u gvmd -f
```

```
sudo journalctl -u gsad -f
```

```
~/Documents/CENEC main > sudo journalctl -u gsad -f
* INT
Oct 27 12:52:08 kali systemd[1]: Starting gsad.service - Greenbone Security Assistant daemon (g
sad)...
Oct 27 12:52:08 kali systemd[1]: Started gsad.service - Greenbone Security Assistant daemon (gs
ad).
Oct 27 12:52:08 kali gsad[31165]: gsad main:MESSAGE:2025-10-27 11h52.08 utc:31165: Starting GSA
D version 24.5.4~git
Oct 27 12:52:08 kali gsad[31170]: gsad main:WARNING:2025-10-27 11h52.08 utc:31170: main: start_
http daemon redirect failed !
Oct 27 12:54:37 kali gsad[31165]: gsad gmp:MESSAGE:2025-10-27 11h54.37 utc:31165: Authenticatio
n success for 'admin' from 127.0.0.1
```

- Logs adicionales (según instalación):

- /var/log/gvm/
- /var/log/openvas/

Revisa errores de permisos, fallos de sincronización o problemas con feeds.

```
~/Documents/CENEC main > sudo cat /var/log/gvm/openvas.log
libgvm util:MESSAGE:2025-10-27 11h57.21 utc:31205: Updated NVT cache from version 0 to 20251027
0701
```

Ejercicio 2: Primer escaneo local

Objetivo: analizar vulnerabilidades del propio Kali.

Pasos:

- Accede a la interfaz gráfica.
- Crea un *Target* con la IP 127.0.0.1.
- Crea un *Task* (tarea) tipo “Full and Fast”.
- Lanza el escaneo y espera resultados.

Entregable:

- Captura del informe generado.
 - Indica cuántas vulnerabilidades detectó (bajas, medias, altas, críticas).
-

Ejercicio 3: Escanear otra máquina en laboratorio

Objetivo: identificar vulnerabilidades en una máquina virtual vulnerable (p. ej., Metasploitable2 o DVWA).

Pasos:

- Añade la IP del objetivo (por ejemplo, 192.168.56.101).
- Crea una tarea con política “Full and Fast Ultimate”.
- Espera al informe y analízalo.

Entregable:

1. Captura de la lista de vulnerabilidades.
 2. Describe una vulnerabilidad crítica detectada y qué servicio la causa.
-

Ejercicio 4: Escaneo selectivo por puerto

Objetivo: usar OpenVAS para centrarse en un servicio específico.

Pasos:

- En el target, selecciona *Port List* → crea una personalizada (por ejemplo, solo puerto 22 o 80).
- Lanza el escaneo de ese target.

Entregable:

- Captura del resultado.
 - Explica qué vulnerabilidades están relacionadas con ese servicio.
-

Ejercicio 5: Interpretar un CVE

Objetivo: analizar un CVE reportado.

Pasos:

- En el informe, elige una vulnerabilidad con CVE (por ejemplo, CVE-2017-0144).
- Busca en <https://nvd.nist.gov> o cve.mitre.org.
- Explica:
 - Qué vulnerabilidad es.

- o Qué impacto tiene.
 - o Cómo se puede mitigar.
-

Ejercicio 6: Comparar políticas de escaneo

Objetivo: comprobar diferencias entre distintos niveles de análisis.

Pasos:

- Lanza dos tareas sobre el mismo objetivo:
 - o “Full and Fast”
 - o “Full and Very Deep”
- Compara el número de vulnerabilidades encontradas.

Entregable: tabla comparativa y conclusiones.

Ejercicio 7: Exportar y documentar informes

Objetivo: generar informes técnicos.

Pasos:

- Abre cualquier informe en la interfaz web.
- Exporta el resultado en PDF o HTML.
- Añade al documento tus observaciones.

Entregable: informe exportado + resumen de hallazgos

4) Nmap / NetworkMiner

Instalación y primer uso de Nmap (NetworkMiner) en Kali Linux

Nota: estas instrucciones están pensadas para la Community / Trial installer de Rapid7 (archivo .bin). Los nombres/URLs pueden variar según versión; si el .bin no se puede descargar, el profesor debe proporcionar el archivo.

1) Preparar Kali (actualizar + utilidades)

Abre una terminal y ejecuta:

```
sudo apt update && sudo apt upgrade -y
```

```
sudo apt install wget net-tools unzip -y
```

```
~/Documents/IFCT0109 > sudo apt install wget net-tools unzip -y
[sudo] password for kali:
wget is already the newest version (1.25.0-2).
net-tools is already the newest version (2.10-2).
net-tools set to manually installed.
unzip is already the newest version (6.0-29).
unzip set to manually installed.
Summary:
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
~/Documents/IFCT0109 >
```

Comprueba la IP de la VM (anótala; la usarás para acceder al GUI):

```
ip a | grep -E "inet .*brd" -n
```

```
~/Documents/IFCT0109 > ip a | grep -E "inet .*brd" -n
9:      inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
17:      inet 172.18.0.1/16 brd 172.18.255.255 scope global br-37885286c3f9
21:      inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
~/Documents/IFCT0109 >
```

2) Descargar el instalador (ejemplo genérico)

Descarga el instalador (sustituye el nombre si el fichero es distinto)

```
wget -O Rapid7Setup-Linux64.bin
```

```
"https://download2.rapid7.com/download/InsightVM/Rapid7Setup-Linux64.bin"
```

```
~/Documents/IFCT0109 > wget -O Rapid7Setup-Linux64.bin "https://download2.rapid7.com/download/InsightVM/Rapid7Setup-Linux64.bin"
--2025-10-28 09:33:26-- https://download2.rapid7.com/download/InsightVM/Rapid7Setup-Linux64.bin
Resolving download2.rapid7.com (download2.rapid7.com)... 99.84.9.73, 99.84.9.20, 99.84.9.11, ...
Connecting to download2.rapid7.com (download2.rapid7.com)|99.84.9.73|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2015232303 (1.9G) [application/octet-stream]
```

Hazlo ejecutable:

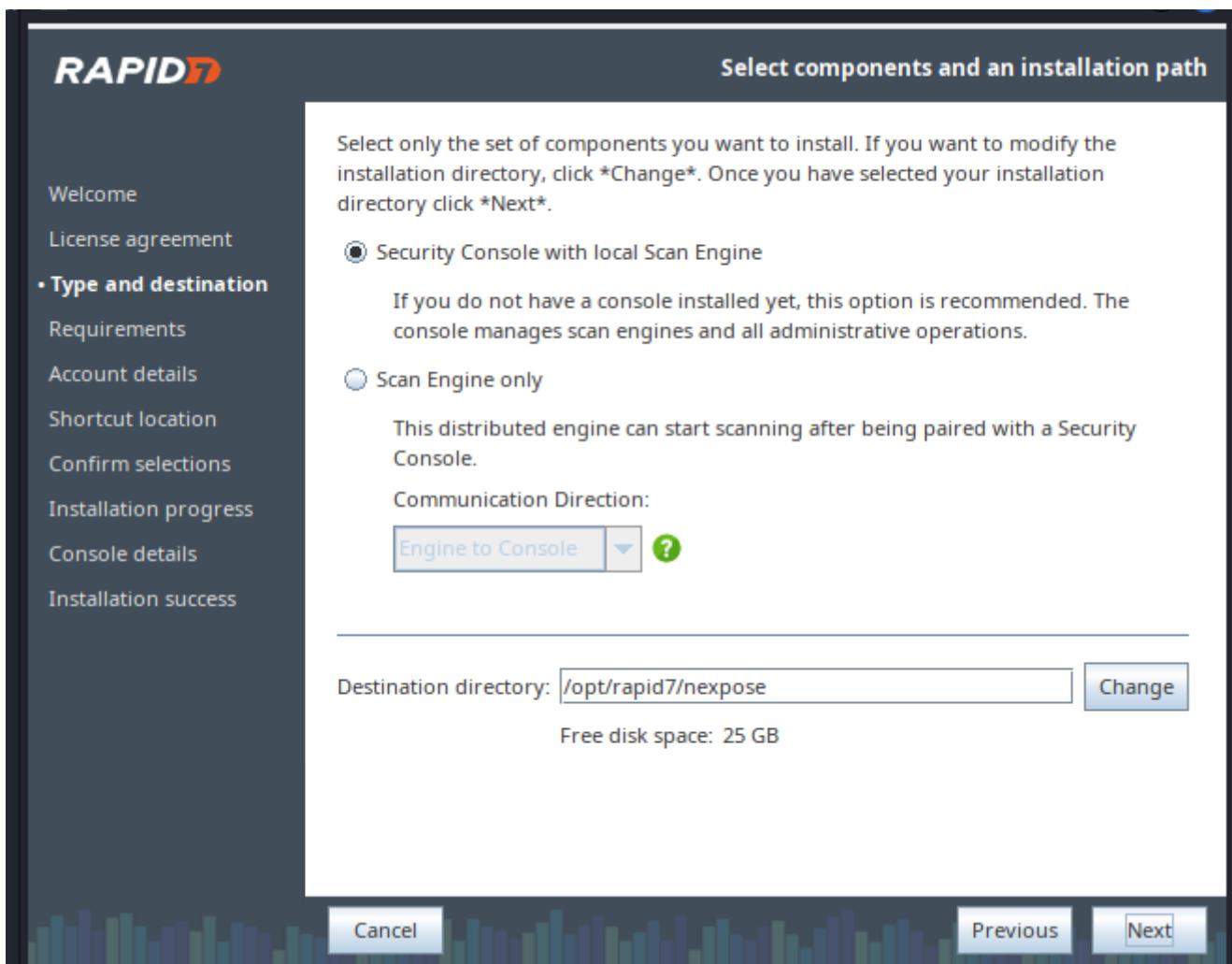
```
chmod +x Rapid7Setup-Linux64.bin
```

3) Ejecutar el instalador (modo texto / consola)

Lánzalo con sudo:

```
sudo ./Rapid7Setup-Linux64.bin
```

- El instalador puede abrir un instalador gráfico o modo texto en terminal.
- Acepta EULA/Next.
- Instálalo en la ruta por defecto (p. ej. /opt/rapid7/nexpose).
- Si te pregunta por puertos, deja por defecto (UI: 3780).

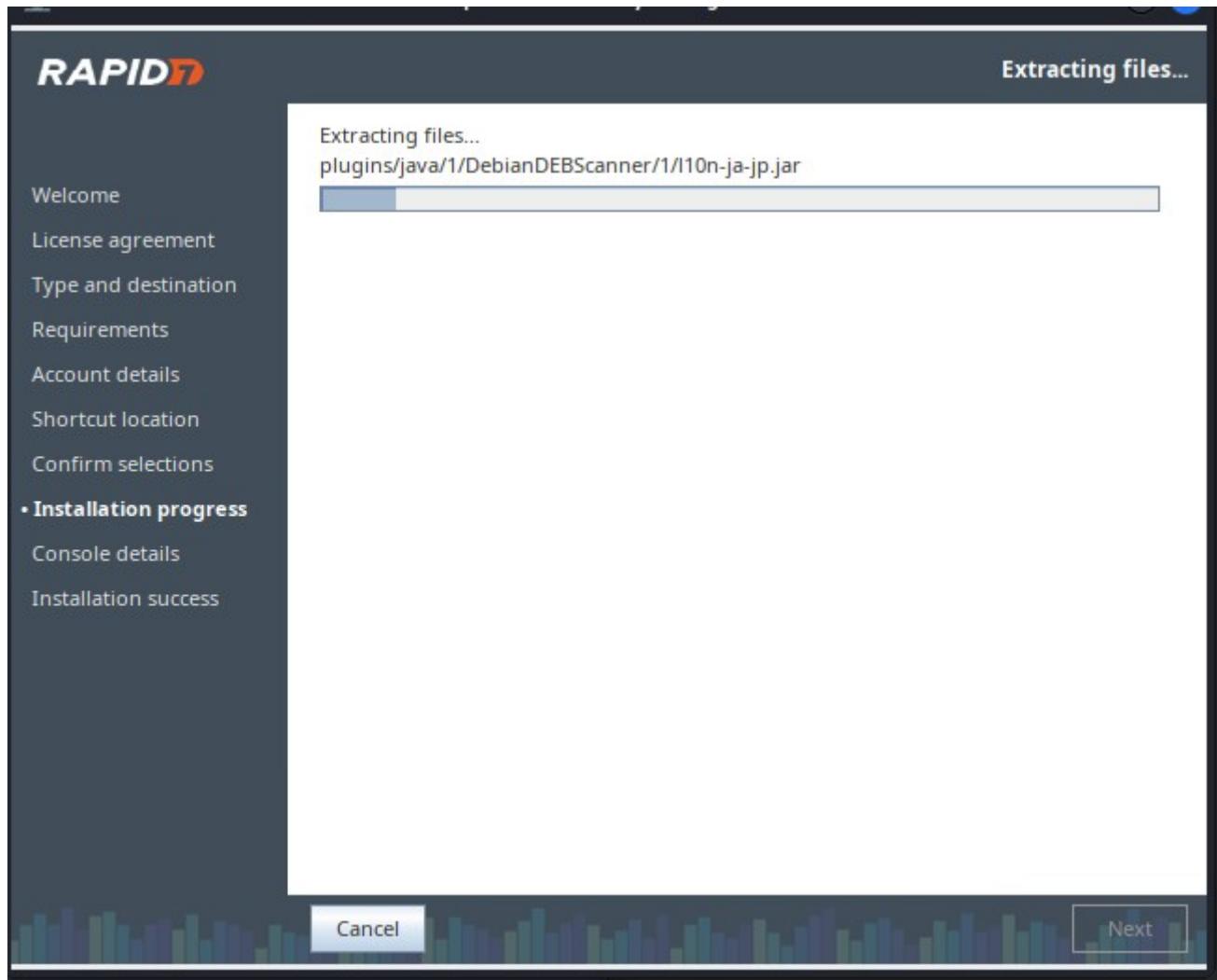


Si no ves interfaz gráfica durante la instalación (instalador gráfico necesita entorno X), el instalador suele funcionar también en modo texto. Sigue las instrucciones en pantalla.

4) Iniciar/gestionar servicios Nexpose

Tras la instalación, inicia el servicio (comandos típicos):

```
sudo /etc/init.d/nexposeconsole start
```



o (systemd)

```
sudo systemctl start nexposeconsole.service
```

```
sudo systemctl status nexposeconsole.service
```

Salida esperada: Active: active (running) o similar. Si ves errores, copia la salida para diagnóstico.

```
root@kali: ~
~ > sudo systemctl start nexposeconsole.service
~ > sudo systemctl status nexposeconsole.service
● nexposeconsole.service - Security Console Service
   Loaded: loaded (/etc/systemd/system/nexposeconsole.service; enabled; preset: disabled)
     Active: active (running) since Tue 2025-10-28 09:56:40 CET; 6s ago
   Invocation: 6f72d2facbb24f04b64d3eac60eb1acd
     Process: 3513 ExecStart=/run/media/kali/2fb599ef-7a78-4ea2-891f-17571de0f2df/mierda/nsc/n
      Main PID: 3530 (screen)
        Tasks: 30 (limit: 26214)
       Memory: 440.2M (peak: 440.2M)
         CPU: 15.420s
        CGroup: /system.slice/nexposeconsole.service
                  └─3530 SCREEN -d -m -S nexposeconsole /run/media/kali/2fb599ef-7a78-4ea2-891f-17
                     ├─3533 /bin/sh /run/media/kali/2fb599ef-7a78-4ea2-891f-17571de0f2df/mierda/nsc/n
                     ├─3704 ./..DLLCACHE/nexserv -className=com/rapid7/nexpose/nsc/NSC
Oct 28 09:56:30 kali systemd[1]: Starting nexposeconsole.service - Security Console Service..
Oct 28 09:56:40 kali nexposeconsole.rc[3513]: Starting NeXpose security console: nexposeconso
Oct 28 09:56:40 kali systemd[1]: Started nexposeconsole.service - Security Console Service.
Lines: 1 17/17 [END]
```

```
2025-10-28T10:12:30 [INFO] [Thread: main] com.rapid7.nexpose.NSC Starting NSC
using Java 17.0.15 with PID 6619 (/run/media/kali/2fb599ef-7a78-4ea2-891f-17571de0f2df/mierda/
nsc/lib/nsc.jar started by root in /run/media/kali/2fb599ef-7a78-4ea2-891f-17571de0f2df/mierda/
nsc)
2025-10-28T10:12:30 [INFO] [Thread: main] com.rapid7.nexpose.NSC No active pr
ofile set, falling back to 1 default profile: "default"
2025-10-28T10:12:36 [INFO] [Thread: main] default Generating c
ertificate for CN=Rapid7 Security Console, O=asdasd.
2025-10-28T10:12:36 [INFO] [Thread: main] c.r.c.cert.CertificateGenerator Generating 4
096-bit SHA512withRSA certificate.
2025-10-28T10:12:39 [INFO] [Thread: main] c.r.c.cert.CertificateGenerator 4096-bit RSA
certificate generated.
2025-10-28T10:12:39 [INFO] [Thread: main] c.r.n.tomcat.TomcatCertificateConfig nscweb.ks fi
le not found: /run/media/kali/2fb599ef-7a78-4ea2-891f-17571de0f2df/mierda/nsc keystores/nscweb.
ks
2025-10-28T10:12:39 [INFO] [Thread: main] default Logging init
ialized. [Name = default] [Level = INFO, WARN, ERROR] [Timezone = Europe/Madrid (Central Europe
an Standard Time, GMT01:00)]
2025-10-28T10:12:39 [INFO] [Thread: main] update Logging init
ialized. [Name = update] [Level = INFO, WARN, ERROR] [Timezone = Europe/Madrid (Central Europea
n Standard Time, GMT01:00)]
2025-10-28T10:12:39 [INFO] [Thread: main] default Initializing
scheduler...
2025-10-28T10:12:40 [INFO] [Thread: main] default Custom Envir
onment Property Override: "com.rapid7.nexpose.nsc.web.cache", with value: "1".
2025-10-28T10:12:40 [INFO] [Thread: main] default Custom Envir
onment Property Override: "com.rapid7.nexpose.jessValidation", with value: "1".
2025-10-28T10:12:40 [INFO] [Thread: main] default Custom Envir
onment Property Override: "com.rapid7.jdbc", with value: "../shared/conf/jdbcDrivers.xml".
2025-10-28T10:12:40 [ERROR] [Thread: main] default The nsc.xml
configuration file was not found, but the database is available. Contact Technical Support.
2025-10-28T10:12:40 [WARN] [Thread: main] ConfigServletWebServerApplicationContext Exception en
countered during context initialization - cancelling refresh attempt: org.springframework.conte
xt.ApplicationContextException: Unable to start web server
2025-10-28T10:12:40 [INFO] [Thread: main] default Extension ma
```

Comprueba que el puerto web está escuchando:

```
sudo ss -tulpn | grep 3780 || sudo ss -tulpn | grep nexpose
```

5) Abrir la interfaz web (desde Kali o cualquier máquina con acceso)

Abre un navegador en Kali (o en otra máquina de la red) e introduce:

```
https://<IP-de-la-Kali>:3780
```

- Acepta el certificado autofirmado si aparece.
- Usuario por defecto suele ser nxadmin (o te pedirá crear uno durante la instalación). Usa la contraseña que hayas definido.

Si no accede desde otra máquina, revisa firewall local:

```
sudo ufw status
```

si está activo y bloquea, permitir puerto:

```
sudo ufw allow 3780/tcp
```

6) Activar licencia Community / crear usuario

- En el primer acceso te pedirá crear/activar la cuenta.
- Para Community Edition puede solicitar un activation key (gratis tras registro en Rapid7). El flujo GUI te guía: registra un correo y pega la clave en la consola.
- Si no puedes obtener clave en clase, pide al profesor que la proporcione.

7) Crear primer Site y lanzar un escaneo (pasos GUI simples)

- Login → menú principal.
- Assets → Sites → New Site
 - Name: Lab_192.168.56.30
 - Add asset: 192.168.56.30 → Add → Save.
- Abrir el Site → botón Scan → elegir política por defecto (ej.: *Full audit without Web Spider* o similar) → Start.
- Espera a que termine. Captura pantalla de Site con status Finished.

8) Validaciones / comandos útiles (en terminal del equipo del grupo)

Usa estos comandos sencillos para comprobar hallazgos que veáis en el informe.

- Comprobar SSH (versión/banners):

```
nmap -sV -p 22 192.168.56.30
```

```
# salida ejemplo: 22/tcp open ssh OpenSSH 7.2p2
```

- Comprobar HTTP / server-status (mod_status):

```
curl -I http://192.168.56.30/server-status
```

```
curl http://192.168.56.30/server-status | head -n 30
```

- Comprobar certificado TLS (fechas):

```
echo | openssl s_client -connect 192.168.56.30:443 -servername 192.168.56.30
```

```
2>/dev/null | openssl x509 -noout -dates
```

```
# devuelve notBefore / notAfter
```

1. Comprobar puertos y servicios:

```
nmap -sV 192.168.56.30
```

2. Comprobar conectividad al servidor Nmapse (desde un cliente):

```
ping -c 3 <IP-Nmapse>
```

```
curl -k https://<IP-Nmapse>:3780
```

9) Exportar informe desde la GUI

- Tras terminar el escaneo: Reports → Generate Report → elegir formato PDF o HTML → Download.
 - Guardad el PDF como nmapse_report_<grupo>.pdf.
-

10) Problemas comunes y soluciones rápidas

- Instalador no se ejecuta / falta permisos → asegúrate chmod +x y ejecuta con sudo.
- Instalación gráfica falla en entorno headless → ejecuta el instalador en modo texto (seguir prompts en terminal).
- Servicio no arranca → ver logs en /opt/rapid7/nmapse/nsc/logs/ o /var/log/ (ruta puede variar).
- sudo journalctl -u nmapseconsole.service -n 200
- No podéis activar la licencia → pedir al profesor que active o proporcione la key

5) dsniff / tcpdump — captura pasiva (solo lectura)

Objetivo: capturar tráfico HTTP no cifrado y analizar headers visibles.

Preparación/seuridad: usar solo la red de laboratorio, páginas HTTP locales. No capturar tráfico de terceros.

Comandos (captura y análisis):

```
# Capturar 20 segundos en interfaz eth0
```

```
sudo tcpdump -i eth0 -w captura_lab.pcap -c 500
```

```
~/Documents/box > sudo tcpdump -i eth0 -w captura_lab.pcap -c 500
tcpdump: listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
^C495 packets captured
497 packets received by filter
0 packets dropped by kernel
~/Documents/box > ls
captura_lab.pcap probando2.txt Rapid7Setup-Linux64.bin
```

```
# Analizar HTTP en pcap con tshark
```

```
tshark -r captura_lab.pcap -Y http.request -T fields -e ip.src -e http.request.method -e
http.host -e http.request.uri
```

```
~/Documents/box > tshark -r captura_lab.pcap -Y http.request -T fields -e ip.src -e http.request.method -e http.host -e http.request.uri
```

Salida de ejemplo:

```
192.168.56.11 GET interno.lab /pagina1.html
192.168.56.11 GET interno.lab /assets/logo.png
```

Interpretación (solución):

- Se ven las URLs solicitadas (host + path) y la IP origen.
- Explicar que en HTTP los headers van en texto claro; con TLS (HTTPS) estos datos están cifrados y solo se vería SNI (o encriptado si TLS1.3 sin ESNI).

Conclusión para el alumno: TLS protege el contenido y los headers; obligar HTTPS evita exposición de navegación.

6) ARP poisoning / MitM — versión simulada / análisis forense (NO ejecutar en redes reales)

Aviso: No proporciones ni ejecutes técnicas MitM en redes que no sean laboratorio controlado y autorizadas. Para enseñar, usa pcap de ejemplo que muestre el ataque o una red aislada preparada por el profesorado.

Actividad segura (pasos):

- Usar el fichero un mitm_capture.pcap donde en las tablas ARP aparecen respuestas ARP maliciosas.
- Los alumnos analizarán el pcap para detectar señales de ARP poisoning: múltiples ARP replies con la misma IP pero distinta MAC, o cambio súbito de la MAC asociada a la gateway.

Comandos para análisis (en la VM del alumno con el pcap):

```
# listar paquetes ARP
```

```
tshark -r mitm_capture.pcap -Y arp -T fields -e frame.time -e arp.src.proto_ipv4 -e arp.src.hw_mac -e arp.dst.proto_ipv4 -e arp.dst.hw_mac
```

```
~/Documents/box > tshark -r captura_lab.pcap -Y arp -T fields -e frame.time -e arp.src.proto_ip v4 -e arp.src.hw_mac -e arp.dst.proto_ipv4 -e arp.dst.hw_mac
```

```
# mostrar conversaciones IP para ver si el atacante aparece en medio (tcpdump/tshark)
```

```
tshark -r mitm_capture.pcap -q -z conv,ip
```

```
~/Documents/box > tshark -r captura_lab.pcap -q -z conv,ip
=====
IPv4 Conversations
Filter:<No Filter>
| Relative | Duration | | | | | | | | Total
| Start | | | | | | | | | |
10.0.2.15 1.749079000 <-> 162.159.61.4 106 22 kB 93 11 kB 199 33 kB
10.0.2.15 1.409591000 <-> 104.18.27.90 21 35 kB 21 1,907 bytes 42 37 kB
10.0.2.15 1.742942000 <-> 23.90.190.145 21 13 kB 21 2,571 bytes 42 16 kB
10.0.2.15 2.424569000 <-> 98.96.213.145 16 7,476 bytes 15 5,234 bytes 31 1 kB
10.0.2.15 2.584089000 <-> 3.224.3.210 15 7,764 bytes 13 5,713 bytes 28 1 kB
10.0.2.15 2.347877000 <-> 18.97.36.13 13 6,666 bytes 12 4,384 bytes 25 1 kB
10.0.2.15 3.493853000 <-> 13.224.83.57 12 7,400 bytes 11 3,016 bytes 23 1 kB
10.0.2.15 2.102297000 <-> 104.18.95.41 12 4,966 bytes 10 2,947 bytes 22 7 kB
10.0.2.15 2.518469000 <-> 163.181.49.183 9 6,568 bytes 9 2,942 bytes 18 9 kB
10.0.2.15 2.286867000 <-> 18.97.36.52 9 808 bytes 8 961 bytes 17 1,769 bytes
10.0.2.15 1.977113000 <-> 47.246.45.185 9 2,894 bytes 8 4,115 bytes 17 7 kB
10.0.2.15 2.266 bytes <-> 155.102.46.10 8 1,762 bytes 7 3,504 bytes 15 5 kB
194.124.205.65 <-> 10.0.2.15 3 202 bytes 3 252 bytes 6 454 b bytes
188.114.97.5 5.034381000 <-> 10.0.2.15 1 54 bytes 1 91 bytes 2 145 b bytes
188.114.97.5 6.423552000 <-> 10.0.2.15 0.0003 =====
```

Salida/ejemplo esperado (fragmento):

```
2026-04-01 10:02:12 192.168.56.1 aa:bb:cc:11:22:33 192.168.56.11 00:11:22:33:44:55  
2026-04-01 10:02:13 192.168.56.1 66:77:88:99:aa:bb 192.168.56.11 00:11:22:33:44:55
```

Observa que la dirección MAC asociada a 192.168.56.1 cambia de aa:bb:cc... a 66:77:88... → indicio de poisoning.

Qué pedir al alumno (solución):

- Identificar el timestamp donde cambió la MAC de la gateway.
- Explicar por qué ip_forward=1 es necesario en un atacante real (permite que el atacante reenvíe tráfico para no provocar caída).
- Proponer mitigaciones: ARP estático en hosts críticos, DHCP snooping, port-security en switches.

7) dnsspoof — demo de falsificación DNS (versión segura)

Objetivo: ver el efecto de una resolución DNS falsificada sin realizar spoofing en red real.

Versión segura (recomendada): editar /etc/hosts del cliente víctima en el laboratorio (o usar un archivo hosts-spoof) para simular redirección y comparar resoluciones dig antes y después.

Pasos (simulación):

- En la VM cliente, ejecutar:

```
dig +short example.com @8.8.8.8
```

guarda IP_original

```
~/Documents/box > dig +short example.com @8.8.8.8  
23.220.75.245  
23.192.228.80  
23.192.228.84  
23.215.0.136  
23.220.75.232  
23.215.0.138
```

- Editar /etc/hosts (necesitas permiso) y añadir:

```
192.168.56.50 example.com
```

1. Volver a consultar:

```
dig +short example.com  
# ahora devuelve 192.168.56.50
```

```
~/Documents/box > cat /etc/hosts  
127.0.0.1      localhost  
127.0.1.1      kali  
::1            localhost ip6-localhost ip6-loopback  
ff02::1        ip6-allnodes  
ff02::2        ip6-allrouters  
  
10.0.2.15      example.com
```

```
~/Documents/box > dig +short example.com  
23.215.0.136  
23.215.0.138  
23.220.75.232  
23.220.75.245  
23.192.228.80  
23.192.228.84
```

Explicación (solución):

- Mostrar la diferencia en resolución: antes → IP original pública; después → IP del servidor controlado en lab.
- Concluir que la respuesta DNS del resolver fue suplantada por /etc/hosts local, simulando el efecto del spoofing sin riesgos.

Alternativa de análisis de pcap: proporcionar dnsspoof_capture.pcap y pedirles que extraigan la respuesta DNS falsa con tshark -r dnsspoof_capture.pcap -Y "dns.flags.rcode == 0 && dns.qry.name == \"example.com\"".

8) filesnarf — ver nombres de ficheros transferidos (metadatos)

Objetivo: mostrar que los nombres/tamaños de ficheros pueden ser visibles en transferencias no cifradas.

Preparación: servicio HTTP/FTP local en la red de laboratorio; subir/descargar prueba.txt.

Comando de análisis (usar pcap o filesnarf si está disponible):

```
# alternativa con tshark para extraer headers HTTP que contienen nombres  
tshark -r captura_http.pcap -Y http -T fields -e http.request.full_uri -e http.content_length
```

Salida ejemplo:

```
curl: (5, SSL: bad hostname  
~/Documents/box > curl -v -F "file=@prueba.txt" http://localhost/upload/  
* Host localhost:80 was resolved.  
* IPv6: ::1  
* IPv4: 127.0.0.1  
* Trying [::1]:80...  
* Connected to localhost (::1) port 80  
* using HTTP/1.x  
> POST /upload/ HTTP/1.1  
> Host: localhost  
> User-Agent: curl/8.15.0  
> Accept: */*  
> Content-Length: 205  
> Content-Type: multipart/form-data; boundary=-----x6SSVUrPrw0E644VKiNYrW  
>  
* upload completely sent off: 205 bytes  
< HTTP/1.1 404 Not Found  
< Date: Tue, 28 Oct 2025 09:37:18 GMT  
< Server: Apache/2.4.65 (Debian)  
< Content-Length: 271  
< Content-Type: text/html; charset=iso-8859-1  
<  
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">  
<html><head>  
<title>404 Not Found</title>  
</head><body>  
<h1>Not Found</h1>  
<p>The requested URL was not found on this server.</p>  
<hr>  
<address>Apache/2.4.65 (Debian) Server at localhost Port 80</address>  
</body></html>  
~/Documents/box > tshark -r captura_lab.pcap -Y http -T fields -e http.request.full_uri -e http.content_length
```

```
http://192.168.56.30/upload/prueba.txt 1024
```

Interpretación (solución):

- Se ve la URL exacta y la longitud de contenido; en protocolos que no cifran metadatos (p.ej. FTP sin TLS o HTTP), estos metadatos son visibles.
- Recomendación: usar HTTPS/FTPS/SFTP para proteger el contenido y metadatos cuando sea posible.

9) macof — inundación de MAC (simulado)

Objetivo: mostrar el **efecto observable** (aumento de tráfico broadcast / inundación) sin necesidad de acceder a una CAM table real. Fácil de montar con 2-3 VMs en VirtualBox.

Topología (VirtualBox)

- VM **Kali-attacker** (donde se genera la inundación)
- VM **Victim** (máquina objetivo, p. ej. Ubuntu)
- (Opcional) VM **Monitor** o usar la Victim para capturas

Todas las VMs en la **misma red interna** de VirtualBox (por ejemplo, red llamada red_lab — NIC configurada como *Internal Network*).

Pasos (preparación)

- Crear snapshots de todas las máquinas.
- Asegurarse de que las VMs pueden hacer ping entre sí (p. ej. ping 192.168.56.11).
- En la VM Monitor (o Victim), preparad tcpdump/tshark:

```
# en la VM Monitor (o Victim) como root
```

```
sudo apt update
```

```
sudo apt install tcpdump tshark -y
```

```
# Preparar fichero de captura
```

```
sudo tcpdump -i eth0 -w /tmp/mac_flood_before.pcap &
```

```
# (luego ctrl+c para detener cuando corresponda)
```

Generar la inundación (Kali-attacker)

Tienes dos opciones: usar macof (si está disponible) o usar un script Python/Scapy (más controlable). **Recomiendo Scapy** por ser didáctico y reproducible.

Opción 1 — Scapy (recomendado)

En Kali instala scapy si hace falta:

```
sudo apt update
```

```
sudo apt install python3-scapy -y
```

Script flood_mac.py (genera tramas Ethernet con **múltiples MACs fuente aleatorias** hacia la red, provocando que el “switch virtual” vea muchas MACs nuevas):

```
#!/usr/bin/env python3
```

```
from scapy.all import Ether, sendp, RandMAC
```

```

import argparse
import time

parser = argparse.ArgumentParser()
parser.add_argument("--iface", required=True, help="Interface to send on (e.g. eth0)")
parser.add_argument("--target", required=False, help="Destination MAC (broadcast by default)", default="ff:ff:ff:ff:ff:ff")
parser.add_argument("--count", type=int, default=5000, help="Number of frames to send")
parser.add_argument("--interval", type=float, default=0.0, help="Interval between frames")
args = parser.parse_args()

for i in range(args.count):
    src = str(RandMAC())
    pkt = Ether(src=src, dst=args.target)/b'\x00'*46
    sendp(pkt, iface=args.iface, verbose=False)
    if args.interval>0:
        time.sleep(args.interval)
print("Done")

```

```

~/Documents/box > vim run.sh
~/Documents/box > chmod +x run.sh
~/Documents/box > ./run.sh
usage: run.sh [-h] --iface IFACE [--target TARGET] [--count COUNT] [--interval INTERVAL]
run.sh: error: the following arguments are required: --iface
~/Documents/box > sudo ./run.sh --iface eth0 --count 3000

```

Ejecutar (como root):

```
sudo python3 flood_mac.py --iface eth0 --count 3000
```

Ajustad --count según lo que queráis demostrar (3000 es suficiente en lab).

```
10:19:52.483740 Loopback (invalid)
10:19:52.484136 Loopback (invalid)
10:19:52.484531 Loopback (invalid)
10:19:52.484987 Loopback (invalid)
10:19:52.485593 Loopback (invalid)
10:19:52.486051 Loopback (invalid)
10:19:52.486475 Loopback (invalid)
10:19:52.486893 Loopback (invalid)
10:19:52.487286 Loopback (invalid)
10:19:52.487592 Loopback (invalid)
10:19:52.487983 Loopback (invalid)
10:19:52.488288 Loopback (invalid)
10:19:52.488888 Loopback (invalid)

⌚ /run/me/k/2fb599ef-7a78-4ea2-891f-17571de0f2df/mierda/nsc > cdbox
~/Documents/box > ls
captura_lab.pcap  mensaje.txt  probando2.txt  prueba.txt  Rapid7Setup-Linux64.bin  run.sh  shell_old.elf
~/Documents/box > sudo ./run.sh --iface eth0 --count 3000
```

Opción 2 — macof (si preferís usar herramienta existente)

Instalar:

```
sudo apt install dsniff -y
```

Ejecutar (como root — **solo en lab**):

```
sudo macof -i eth0
```

(macof enviará tramas con MACs aleatorias hasta ser detenido con Ctrl+C)

Capturar y observar

En Monitor/Victim, antes y durante la inundación:

```
# iniciar captura antes
sudo tcpdump -i eth0 -w /tmp/flood_before.pcap

# detener con Ctrl+C cuando empiezas la inundación y querás analizar "antes"

# iniciar captura durante la inundación
sudo tcpdump -i eth0 -w /tmp/flood_during.pcap

# detener con Ctrl+C tras unos segundos
```

Analizar rápido con tshark:

```
# Contar paquetes por tipo (ARP/IPv4/broadcast)
tshark -r /tmp/flood_during.pcap -q -z io,phs

# Contar tramas broadcast / multicast
tshark -r /tmp/flood_during.pcap -Y "eth.dst == ff:ff:ff:ff:ff:ff" -T fields -e frame.number | wc -l
```

```
# Ver primeras 20 tramas para comprobar src MACs distintas  
tshark -r /tmp/flood_during.pcap -T fields -e eth.src | sort | uniq -c | sort -nr | head -n 30
```

Qué observar (discusión / solución)

1. Aumento notable de **tramas broadcast/multicast** en flood_during.pcap frente a flood_before.pcap.
2. Lista larga de **MACs fuente distintas** en el análisis eth.src.
3. En un switch real, si la CAM table se llena, el switch **no tiene entradas para muchas MACs y comienza a floodear** (reenvía tramas por todos los puertos) — en nuestro lab observaréis más tráfico broadcast en la Victim/Monitor.

10) mailsnarf — capturar correos sin TLS (solo en lab)

Objetivo: mostrar exposición de cabeceras y cuerpos si SMTP/IMAP/POP no usan TLS.

Preparación: servidor SMTP/IMAP de laboratorio sin TLS y cliente que envía correo de prueba.

Comando para capturar:

```
sudo tcpdump -i eth0 -w mail_capture.pcap port 25 or port 110 or port 143  
# analizar con tshark  
tshark -r mail_capture.pcap -Y smtp -V | less
```

Salida ejemplo (fragmento):

```
From: alumno@lab.local  
To: profesor@lab.local  
Subject: Prueba  
Body: Este es un correo de ejemplo...
```

Explicación (solución):

- Se muestran cabeceras y cuerpo en texto claro.
- Contramedidas: habilitar STARTTLS/SMTPS/IMAPS/POP3S, autenticación segura, DANE/TLSA si procede.

11) urlsnarf / urlenarf — URLs visitadas

Objetivo: listar URLs visitadas desde una máquina cliente en HTTP no cifrado.

Preparación: navegar desde VM cliente a páginas HTTP en el servidor local.

Comando:

```
tshark -r captura_http.pcap -Y http.request -T fields -e ip.src -e http.host -e http.request.uri
```

o

```
sudo urlsnarf -i eth0
```

Salida ejemplo:

```
sudo urlsnarf -i eth0
```

```
c
~/Documents/box > sudo urlsnarf -i eth0
urlsnarf: listening on eth0 [tcp port 80 or port 8080 or port 3128]
^C
~/Documents/box > | x INT 48s
x INT 39s

c
~/Documents/box > printf "secret1" | md5sum
e52d98c459819a11775936d8dfbb7929 -
~/Documents/box > |
```

```
192.168.56.11 interno.lab /pagina1.html
```

```
192.168.56.11 interno.lab /assets/logo.png
```

Interpretación (solución):

- Se puede correlacionar qué URLs visita cada IP.
- Recomendación: forzar HTTPS y HSTS en servidores.

12) Brutus / hydra — fuerza bruta controlada (solo cuenta de laboratorio)

13) RainbowCrack — mini-demo de tabla arcoíris (conceptual y local)

Objetivo: entender el poder de tablas precomputadas y por qué el *salt* las inutiliza.

Actividad práctica (local, sin usar tablas públicas):

- Generar hash MD5 de la palabra secret1:

```
printf "secret1" | md5sum
```

```
# salida: 5ebe2294ecd0e0f08eab7690d2a6ee69
```

1. Crear una pequeña base local con hashes de una wordlist y buscar:

```
# crear local_hash_db.txt

while read -r pw; do
    printf "%s" "$pw" | md5sum | awk '{print $1 ":" "pw"}'
done < wordlist.txt > local_hash_db.txt

# buscar

grep "^\$5\$ebe2294ecd0e0f08eab7690d2a6ee69:" local_hash_db.txt

# salida esperada: \$5\$ebe2294...:secret1
```

Interpretación (solución):

- Si el hash está en la base local, se recupera la contraseña.
- Añade un *salt* (por ejemplo salt+password) y vuelve a calcular: entonces el hash resultante no aparece en la tabla precomputada.
- Conclusión: usar salt + algoritmos KDF (bcrypt, Argon2, PBKDF2) aumenta el coste de ataques.

```
~/Documents/box > cat wordlist.txt
hola
adios
password
1234
~/Documents/box > while read -r pw; do
    printf "%s" "$pw" | md5sum | awk '{print $1 ":" "pw"}'
done < wordlist.txt > local_hash_db.txt

~/Documents/box > cat local_hash_db.txt
4d186321c1a7f0f354b297e8914ab240:
6e6e2ddb6346ce143d19d79b3358c16a:
5f4dcc3b5aa765d61d8327deb882cf99:
81dc9bdb52d04dc20036dbd8313ed055:
~/Documents/box > |
```

14) CrackStation / consulta de hashes (uso didáctico, offline)

Objetivo: simular consulta de hashes contra una base local.

Pasos (local):

1. Crear local_hash_db.txt como en el ejercicio anterior con wordlist pequeña.
2. Buscar el hash de password123 (482c811da5d5b4bc6d497ffa98491e38) en la DB local con grep.

Comando:

```
grep "^482c811da5d5b4bc6d497ffa98491e38:" local_hash_db.txt
```

```
# salida: 482c811....:password123
```

Explicación (solución):

2. Si la contraseña es común, aparecerá.
3. Medidas: usar hashing fuerte con salt y KDF; evitar contraseñas comunes.

GLOSARIO

Qué significa NetBIOS

NetBIOS significa **Network Basic Input/Output System**.

Es una **API (interfaz de comunicación)** creada en los años 80 para que las aplicaciones pudieran comunicarse entre ordenadores dentro de una red local (LAN).

👉 No es un protocolo de red por sí mismo, sino una **capa de servicios** que permite a los equipos de una red local:

3. Reconocerse por **nombre NetBIOS** (por ejemplo, EQUIPO01 en lugar de una IP).
4. **Compartir recursos**, como carpetas o impresoras (a través de SMB).
5. Enviar mensajes o sesiones entre ordenadores (algo que hoy casi no se usa).

¿Qué es Samba?

Samba es un **software libre** que permite que un sistema **Linux o Unix** pueda **compartir archivos e impresoras** con equipos **Windows** dentro de una red local.