REPORTE PROYECTO FINAL:

Análisis Forense y Hardering del Servidor de 4GeeksAcademy

Autor: Adrià Coll Ortega
Fecha: 16/07/2025
Curso: spain-cs-pt-6
Profesores: Raúl Moncada y Javier Álvarez

Índice

Índice 3
Introducción 4
Metodología 4
Fase_1: Reconocimiento y recolección de evidencias 5
1) Identificación de archivos comprometidos 6
2) Identificación de archivos sospechosos 7
3) Escaneo del servidor en busca de malware 9
4) Bloqueo de exploit y detención de servicios comprometidos 10
5) Conclusión 13
Fase_2: Detectar y corregir una vulnerabilidad distinta 14
Fase_3: Plan de respuesta de incidentes y certificación 15
Introducción 15
Plan de Respuesta a Incidentes 15
1. Preparación 15
Herramientas implementadas: 15
2. Detección y Análisis 16
3. Contención 16
Acciones inmediatas: 16
Acciones a largo plazo: 16
4. Erradicación 16
5. Recuperación 16
6. Lecciones Aprendidas 17
Sistema de Gestión de Seguridad de la Información (ISO 27001) 17
1. Análisis de Riesgos 17
2. Políticas de Seguridad 17
Protección de Datos (DLP): 17
Respaldos: 17
3. Controles Técnicos 18
4. Conformidad y Auditoría 18
Conclusión 19

Introducción

Este informe redacta la simulación de un servidor comprometido de ***4GeeksAcademy***. Se trata de una máquina virtual Debian que se puede descargar haciendo clic aquí.

A nosotros se nos encarga el rol de **analista en ciberseguridad**, dónde nuestra tarea principal es **restaurar y proteger el servidor**. Para ello, este informe está dividido en 3 partes. Cada una se centra en una fase en particular y explica cada paso/proceso seguido para realizar nuestra tarea. Estas fases son:

- **Fase_1: Reconocimiento y recolección de evidencias**
- **Fase_2: Detectar y corregir una vulnerabilidad distinta**
- **Fase_3: Plan de respuesta a incidentes y certificación**

Metodología

Al hacer este proyecto, utilizamos el estándar ISO27001 para aplicar cambios y/o sugerir acciones o recomendaciones.

A lo largo del proyecto también veremos que el corazón del informe se centra en los 3 puntos anteriormente mencionados: **análisis**, **mitigación** y **hardering** de las vulnerabilidades encontradas en la máquina víctima.

Es importante remarcar que, como la ***VM Debian*** cuenta con registro de ***logs***, se me recomendó e hice caso de realizar ***"snapshoots"*** en cada punto crítico. De esta forma siempre he sido capaz de volver atrás y recuperar el estado previo a varios imprevistos o acciones que quería rehacer de forma rápida.

Para completar este trabajo, he usado un total de 4 máquinas (ordenadores) con distintos Sistemas Operativos. Sus roles son:

- **Máquina Host y ejecución de Autopsy:** Windows11 Pro
- **Máquina Atacante:** Kali Linux
- **Máquina Víctima/Servidor 4Geeks:** Servidor Debian

Fase_1: Reconocimiento y recolección de evidencias

En este punto, nuestro objetivo es claro, llevar a cabo un análisis forense para bloquear el exploit, corregir la vulnerabilidad y evitar que el atacante escale.

Para ello, nos guiaremos a través de los siguientes puntos:

1) Identificación de qué servicios han sido comprometidos y cómo el atacante ha accedido al servidor
2) Identificación de archivos sospechosos, procesos en ejecución y cualquier modificación inusual en el sistema
3) Realización de un escaneo del servidor para detectar rootkits o malware
4) Actualizar y corregir configuraciones de seguridad

El primer paso que hemos realizado, ha sido la creación de la imagen del servidor Debian para poder analizarla con la herramienta de software **Autopsy**.
Para ello hacemos uso de una carpeta compartida que tengo junto con la máquina virtual Kali, y ahí ejecutamos el siguiente comando:

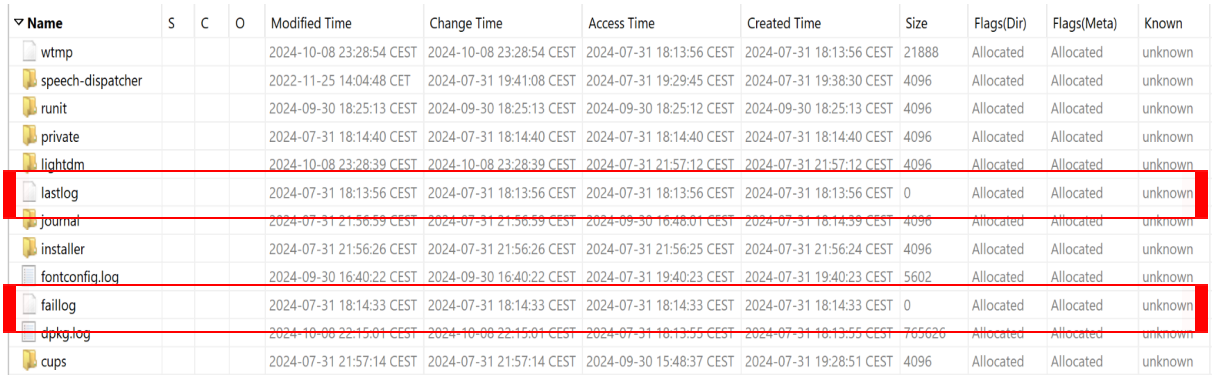
NOTA: Comandos como la creación de imágenes y otros, son mucho más sencillos de realizar desde linux, por eso me ayudo de una carpeta compartida entre este y mi máquina host.

Lo que hemos hecho ha sido, primero, descomprimir el archivo con extensión ***.ova*** con la herramienta tar, para conseguir el documento ***debian-disck001.vmdk***, que será el archivo que usaremos para convertirlo a extensión ***.dd*** (Extensión compatible con Autopsy).

1) Identificación de archivos comprometidos

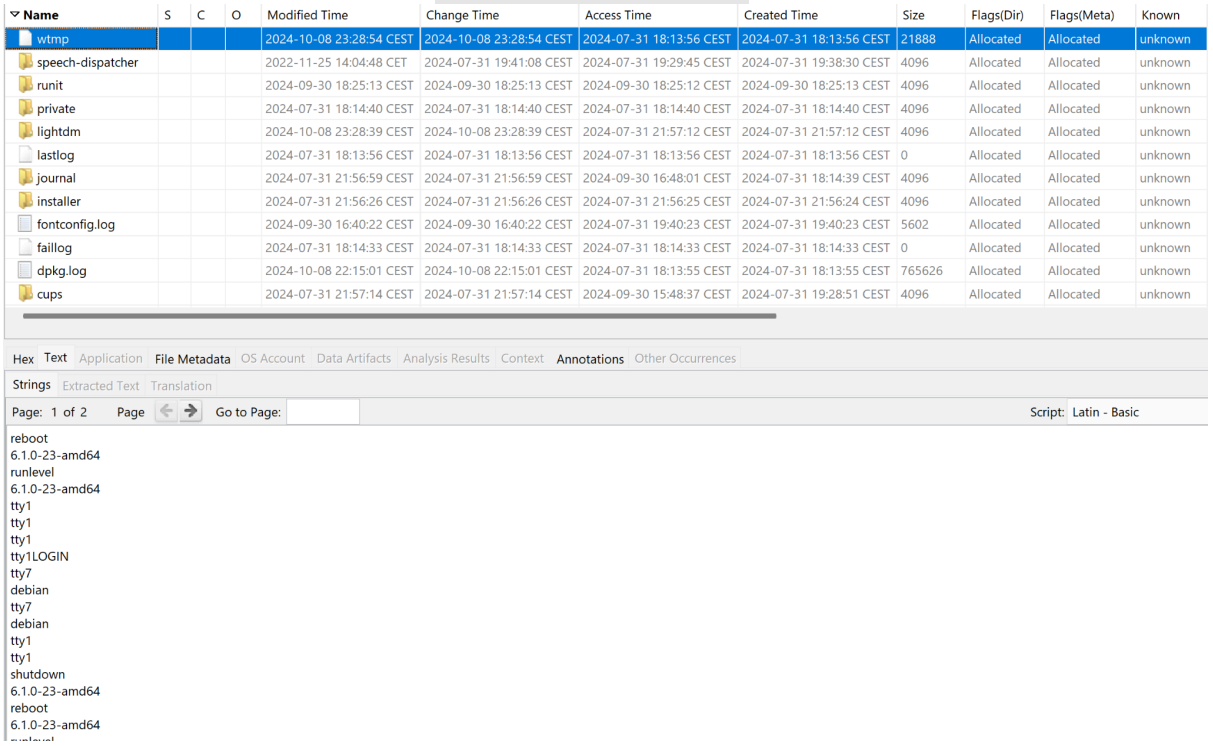
Ahora que tenemos ya todo listo para el Autopsy, es hora de empezar con el análisis forénsico.

Un consejo inicial que se nos da, es comprobar los logs del servidor. A lo que para nuestra sorpresa nos encontramos con lo siguiente:



Name	S	C	O	Modified Time	Change Time	Access Time	Created Time	Size	Flags(Dir)	Flags(Meta)	Known
wtmp				2024-10-08 23:28:54 CEST	2024-10-08 23:28:54 CEST	2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	21888	Allocated	Allocated	unknown
speech-dispatcher				2022-11-25 14:04:48 CET	2024-07-31 19:41:08 CEST	2024-07-31 19:29:45 CEST	2024-07-31 19:38:30 CEST	4096	Allocated	Allocated	unknown
runit				2024-09-30 18:25:13 CEST	2024-09-30 18:25:13 CEST	2024-09-30 18:25:12 CEST	2024-09-30 18:25:13 CEST	4096	Allocated	Allocated	unknown
private				2024-07-31 18:14:40 CEST	2024-07-31 18:14:40 CEST	2024-07-31 18:14:40 CEST	2024-07-31 18:14:40 CEST	4096	Allocated	Allocated	unknown
lightdm				2024-10-08 23:28:39 CEST	2024-10-08 23:28:39 CEST	2024-07-31 21:57:12 CEST	2024-07-31 21:57:12 CEST	4096	Allocated	Allocated	unknown
lastlog				2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	0	Allocated	Allocated	unknown
journal				2024-07-31 21:56:59 CEST	2024-07-31 21:56:59 CEST	2024-09-30 16:48:01 CEST	2024-07-31 18:14:39 CEST	4096	Allocated	Allocated	unknown
installer				2024-07-31 21:56:26 CEST	2024-07-31 21:56:26 CEST	2024-07-31 21:56:25 CEST	2024-07-31 21:56:24 CEST	4096	Allocated	Allocated	unknown
fontconfig.log				2024-09-30 16:40:22 CEST	2024-09-30 16:40:22 CEST	2024-07-31 19:40:23 CEST	2024-07-31 19:40:23 CEST	5602	Allocated	Allocated	unknown
faillog				2024-07-31 18:14:33 CEST	2024-07-31 18:14:33 CEST	2024-07-31 18:14:33 CEST	2024-07-31 18:14:33 CEST	0	Allocated	Allocated	unknown
dpkg.log				2024-10-08 22:15:01 CEST	2024-10-08 22:15:01 CEST	2024-07-31 18:13:55 CEST	2024-07-31 18:13:55 CEST	765626	Allocated	Allocated	unknown
cups				2024-07-31 21:57:14 CEST	2024-07-31 21:57:14 CEST	2024-09-30 15:48:37 CEST	2024-07-31 19:28:51 CEST	4096	Allocated	Allocated	unknown

Los 2 archivos generales encargados de guardar toda la actividad de inicio de sesión en la máquina (**lastlog** y **faillog**), están vacíos. Esto es una señal directa de que han sido truncados por el/los atacantes.
Si seguimos buscando, nos damos cuenta de que el archivo **wtmp** (documento encargado de registrar los eventos de inicio, cierre y sesiones del sistema) contiene la siguiente información:



Name	S	C	O	Modified Time	Change Time	Access Time	Created Time	Size	Flags(Dir)	Flags(Meta)	Known
wtmp				2024-10-08 23:28:54 CEST	2024-10-08 23:28:54 CEST	2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	21888	Allocated	Allocated	unknown
speech-dispatcher				2022-11-25 14:04:48 CET	2024-07-31 19:41:08 CEST	2024-07-31 19:29:45 CEST	2024-07-31 19:38:30 CEST	4096	Allocated	Allocated	unknown
runit				2024-09-30 18:25:13 CEST	2024-09-30 18:25:13 CEST	2024-09-30 18:25:12 CEST	2024-09-30 18:25:13 CEST	4096	Allocated	Allocated	unknown
private				2024-07-31 18:14:40 CEST	2024-07-31 18:14:40 CEST	2024-07-31 18:14:40 CEST	2024-07-31 18:14:40 CEST	4096	Allocated	Allocated	unknown
lightdm				2024-10-08 23:28:39 CEST	2024-10-08 23:28:39 CEST	2024-07-31 21:57:12 CEST	2024-07-31 21:57:12 CEST	4096	Allocated	Allocated	unknown
lastlog				2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	2024-07-31 18:13:56 CEST	0	Allocated	Allocated	unknown
journal				2024-07-31 21:56:59 CEST	2024-07-31 21:56:59 CEST	2024-09-30 16:48:01 CEST	2024-07-31 18:14:39 CEST	4096	Allocated	Allocated	unknown
installer				2024-07-31 21:56:26 CEST	2024-07-31 21:56:26 CEST	2024-07-31 21:56:25 CEST	2024-07-31 21:56:24 CEST	4096	Allocated	Allocated	unknown
fontconfig.log				2024-09-30 16:40:22 CEST	2024-09-30 16:40:22 CEST	2024-07-31 19:40:23 CEST	2024-07-31 19:40:23 CEST	5602	Allocated	Allocated	unknown
faillog				2024-07-31 18:14:33 CEST	2024-07-31 18:14:33 CEST	2024-07-31 18:14:33 CEST	2024-07-31 18:14:33 CEST	0	Allocated	Allocated	unknown
dpkg.log				2024-10-08 22:15:01 CEST	2024-10-08 22:15:01 CEST	2024-07-31 18:13:55 CEST	2024-07-31 18:13:55 CEST	765626	Allocated	Allocated	unknown
cups				2024-07-31 21:57:14 CEST	2024-07-31 21:57:14 CEST	2024-09-30 15:48:37 CEST	2024-07-31 19:28:51 CEST	4096	Allocated	Allocated	unknown

Hex Text Application File Metadata OS Account Data Artifacts Analysis Results Context Annotations Other Occurrences

Strings Extracted Text Translation

Page: 1 of 2 Page Go to Page: Script: Latin - Basic

```
reboot
6.1.0-23-amd64
runlevel
6.1.0-23-amd64
tty1
tty1
tty1
tty1LOGIN
tty7
debian
tty7
debian
tty1
tty1
shutdown
6.1.0-23-amd64
reboot
6.1.0-23-amd64
```

Esto nos da a entender 2 puntos muy importantes. El primero es que ha habido varios inicio de sesión exitosos con el **usuario debian**. El

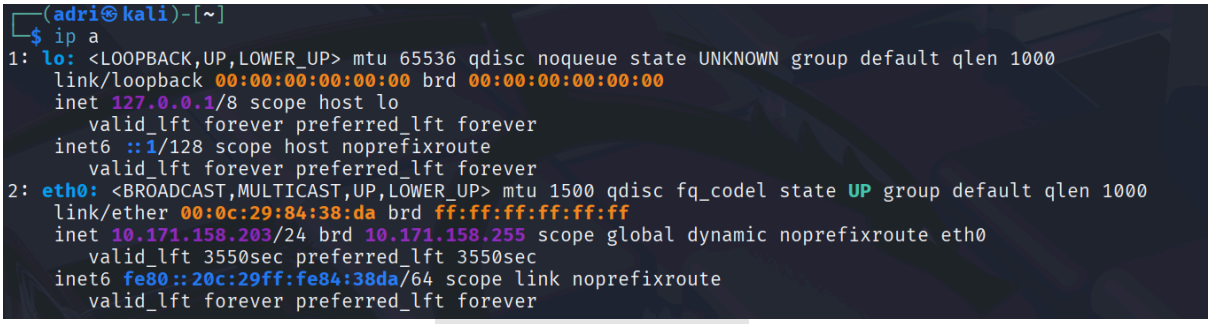
segundo punto es que durante el ataque, se reinició la máquina varias veces, esto es una posible señal de que el/los atacantes estuvieran borrando pruebas.
Para este punto, no encuentro ningún otro registro de logs, por lo que da a entender que seguramente el/los atacantes accedieron a la máquina de forma remota (SSH y/o FTP) y borraron todo indicio de pruebas.

2) Identificación de archivos sospechosos

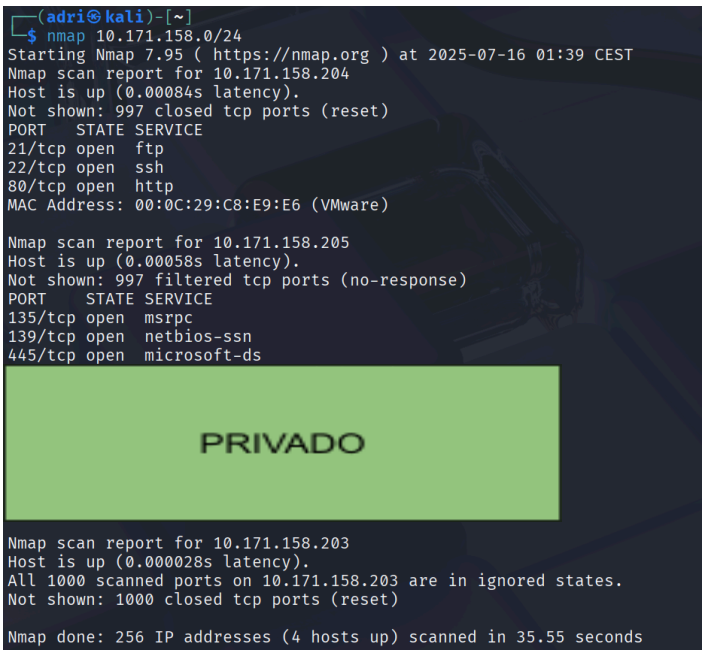
A partir de aquí, dejo Autopsy y sigo con el análisis usando:

- Máquina Kali
- Servidor Debian afectado.

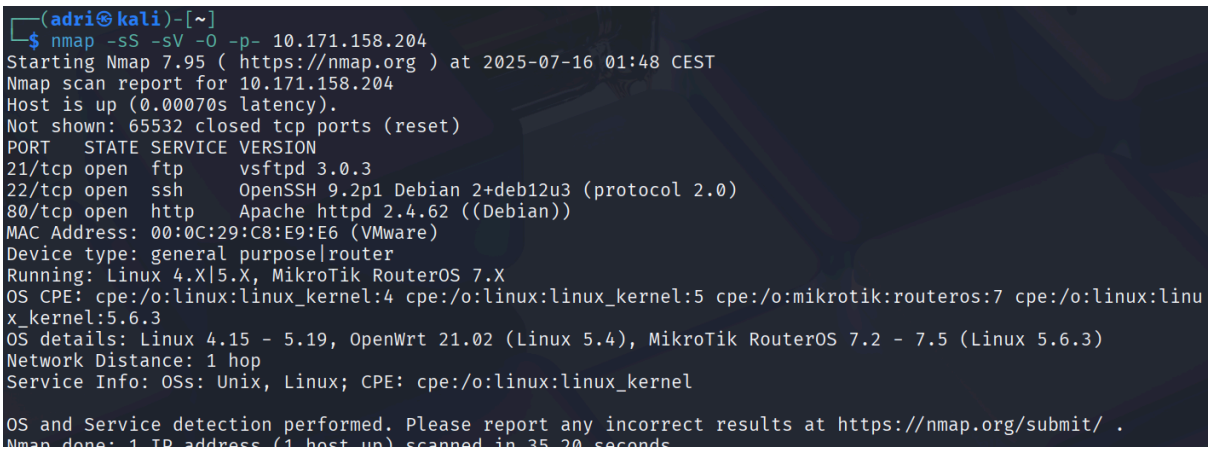
Primeramente, identificamos la IP de nuestra Kali, siendo esta: **10.171.158.203/24**



Seguimos directamente con un escaneo de la red para identificar también el servidor Debian comprometido:



Aquí nos damos cuenta de que la **IP de Servidor Debian es la 10.171.158.204** (Las demás IP pertenecen a otros dispositivos de la red).
Ahora que sabemos la IP del servidor, podemos hacer un análisis más exhaustivo a través de un escaneo completo:



Los parámetros de este comando tienen las siguientes funciones:

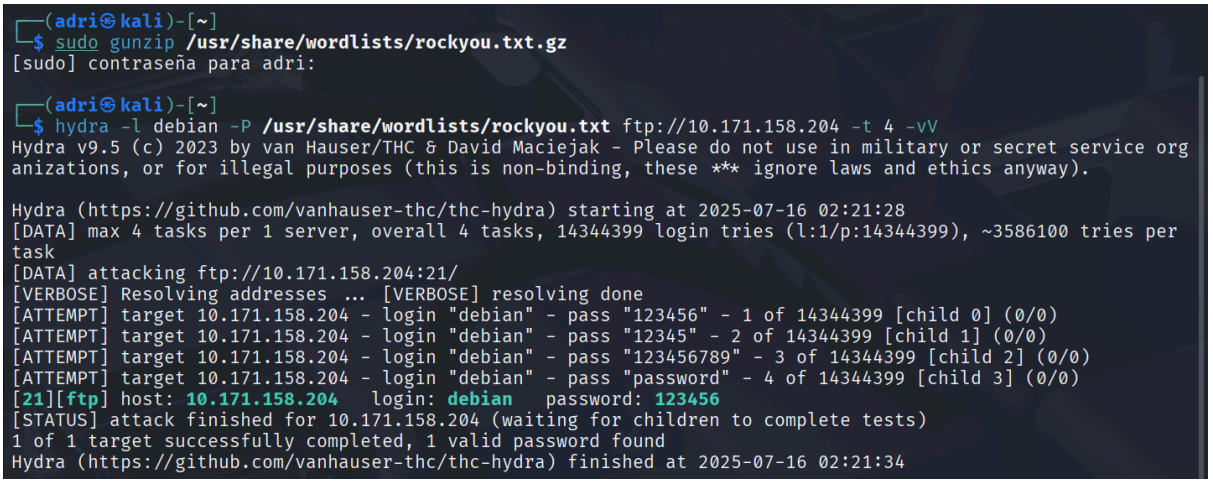
- **-sS** → Escaneo TCP SYN en modo sigiloso (No hace tanto ruido en la red)
- **-sV** → Detección de versiones de servicios
- **-O** → Detecta el Sistema Operativo de la máquina
- **-p-** → Escanea TODOS los puertos de la máquina

Y los resultados indican que tiene un total de 3 puertos expuestos:

Estos resultados respaldan la idea inicial de que el/los atacantes hayan accedido de forma remota, o bien por FTP o por SSH.

Cabe destacar que de los 3 servicios expuestos, FTP es el que tiene la versión más desactualizada y con más vulnerabilidades conocidas, entre ellas que se permite el inicio de sesión FTP anónimo (**Prinicipal Sospecha**).

Ahora que sabemos tanto la IP del Servidor Debian, su usuario y las vulnerabilidades que tiene, realizamos un ataque de fuerza bruta con la ayuda de la herramienta ***hydra*** y la wordlist ***rockyou.txt***.



Gracias a estas herramientas, obtenemos la contraseña de acceso al servidor Debian.
Un avance crucial ahora que tenemos acceso al servidor, es **realizar de forma inmediata una actualización general** de todo el sistema y de sus dependencias.
Realizamos el comando: ***sudo apt update && sudo apt upgrade -y***

3) Escaneo del servidor en busca de malware

Ahora mismo, tenemos el sistema actualizado a día de hoy, y eso ya ha solucionado una gran parte de las vulnerabilidades que tenía el servidor.
Sin embargo, aún debemos buscar cualquier tipo de rootkits malware que tenga instalado.
Para ello, usaremos un total de 2 herramientas:
Primero, haremos uso de la herramienta **chkrootkit**. Encargada de encontrar rootkits conocidos ***(t0rn, Volc, slapper)*** y los resultados son:

- **/usr/lib/libreoffice/share/.registry:** WARNING
- **Output from ifpromsic:** WARNING

```
WARNING: Output from ifpromisc:
lo: not promisc and no packet sniffer sockets
ens33: PACKET SNIFFER(/usr/sbin/NetworkManager[41231], /usr/sbin/NetworkManager[
41231])
```

Segundo, he instalado el antivirus **ClamAV**. Se encarga de detectar y eliminar malware, virus, troyanos, ransomware, etc. He hecho un escaneo completo del sistema, y estos han sido los resultados:

```
----------- SCAN SUMMARY -----------
Known viruses: 8707603
Engine version: 1.0.7
Scanned directories: 36279
Scanned files: 176061
Infected files: 0
Total errors: 1038
Data scanned: 8853.20 MB
Data read: 7197.16 MB (ratio 1.23:1)
Time: 2613.228 sec (43 m 33 s)
Start Date: 2025:07:16 00:20:39
End Date:   2025:07:16 01:04:12
```

Debido a estos resultados, doy a entender que a pesar de que la máquina ha sido comprometida, no cuenta con ningún malware, ni backdoor físico, ni fichero corrupto que pueda comprometer nuestra máquina y que no pueda ser arreglado por las medidas de seguridad que están más adelante.

4) Bloqueo de exploit y detención de servicios comprometidos

Ahora que hemos encontrado las vulnerabilidades y peligros del servidor, el siguiente paso es protegerlo mediante el cierre de puertos y configurando correctamente cada servicio.

Empezamos con la instalación del firewall ***ufw*** (usado anteriormente en el curso), y añadimos las siguientes reglas:

- sudo ufw default deny incoming
- sudo ufw default allow outgoing
- sudo ufw allow from <IP_KALI_LINUX> to any port 22
- sudo ufw enable

```
debian@debian:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
debian@debian:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
debian@debian:~$ sudo ufw allow from 10.171.158.203 to any port 22
Rules updated
debian@debian:~$ sudo ufw enable
Firewall is active and enabled on system startup
debian@debian:~$
```

Una vez tenemos un firewall bien configurado, pasamos a configurar/eliminar los servicios comprometidos

- **FTP:** Empezamos directamente eliminando el servicio ***vsftpd***, ya que no le encuentro un motivo fundamental para mantenerlo. Importante también cerrar el puerto en el firewall mediante el comando ***sudo ufw deny 21/tcp*** (si no tenemos ningún servicio expuesto, entonces es mejor cerrar el puerto).

```
debian@debian:~$ sudo apt purge vsftpd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libdaxctl1 libndctl6 libpmem1 linux-image-6.1.0-22-amd64 linux-image-6.1.0-23-amd64
Use 'sudo apt autoremove' to remove them.
The following packages will be REMOVED:
  vsftpd*
0 upgraded, 0 newly installed, 1 to remove and 0 not upgraded.
After this operation, 351 kB disk space will be freed.
Do you want to continue? [Y/n] Y
(Reading database ... 180648 files and directories currently installed.)
Removing vsftpd (3.0.3-13+b2) ...
Processing triggers for man-db (2.11.2-2) ...
(Reading database ... 180594 files and directories currently installed.)
Purging configuration files for vsftpd (3.0.3-13+b2) ...
```

- **SSH:** En este caso sí voy a dejar activo el servicio SSH, porque al fin y al cabo se trata de un servidor Debian que, en algún momento, puede requerir de conexión remota para ser gestionado en el futuro.
  Para ello, requiere de una modificación en la configuración del servicio ssh. Lo haremos a partir del fichero ***/etc/ssh/ssh_config***. La configuración es la siguiente:

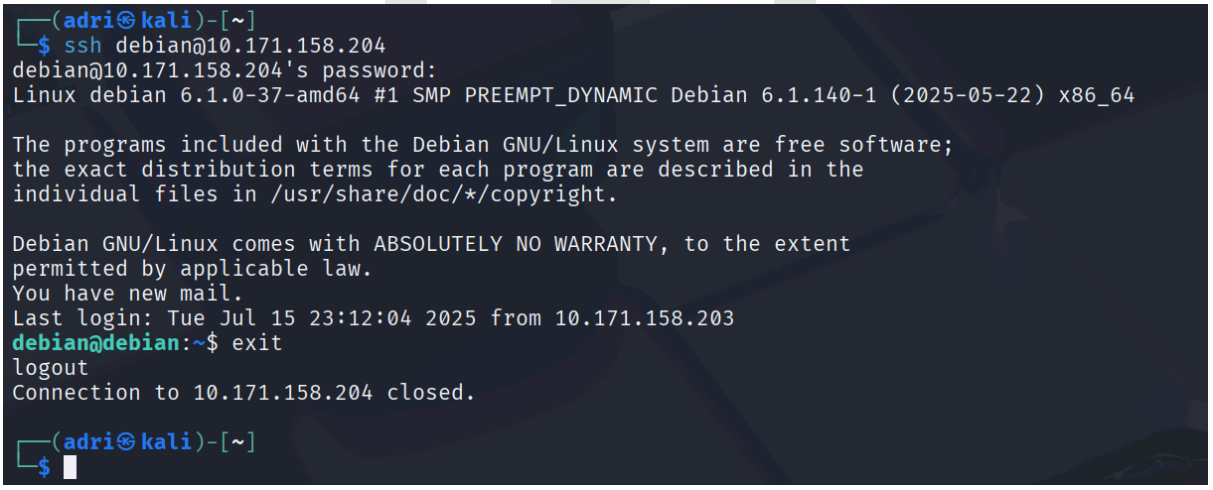
```
#1 Deshabilitar caractristicas inseguras
HostbasedAuthentication no
GSSAPIAuthentication no
PasswordAuthentication no
PermitEmptyPasswords no
ChallengeResponseAuthentication no
PermitRootLogin no

#2 Configuracion de Privacidad
StrictHostKeyChecking yes
HashKnownHosts yes
AddressFamily inet
ConnectTimeout 30
TCPKeepAlive no
```

  - **HostbasedAuthentication no:** Confía en la IP del equipo conectado (más práctico).
  - **GSSAPIAuthentication no:** Deshabilita autenticación por Kerberos, eliminando así complejidad y posibles vectores de ataque.

- **PasswordAuthentication no:** Obliga ausar claves SSH mucho más seguras
- **PermitEmptyPasswords no:** Bloquea cuentas con contraseña vacía
- **ChallengeResponseAuthentication no:** Simplifica autenticación y evita posibles exploits en métodos obsoletos.
- **PermitRootLogin no:** Niega el inicio de sesión como root.

- **StrictHostKeyChecking yes:** Previene ataques Man-in-the-Middle (MITM)
- **HashKnownHosts yes:** Protege la privacidad de los servidores a los que te conectas.
- **AddressFamily inet:** Si no usamos IPv6, reduce la superfície de ataque.
- **ConnectTimeout 30:** Evita que SSH quede colgado en redes lentas o hosts inalcanzables.
- **TCPKeepAlive no:** Mitiga riesgos como TCP hijacking.



Habiendo actualizado y mejorado los servicios, ahora es turno de cambiar tanto las contraseñas del usuario debian, como la del usuario root.
En el servidor, el mismo usuario debian es también el usuario root. Así que al cambiar la contraseña de uno, también cambiará la otra.

```
root@debian:/run/user/1000# passwd debian
New password:
Retype new password:
passwd: password updated successfully
root@debian:/run/user/1000# exit
exit
```

La nueva contraseña segura es: **XqbmKfpt.45!** Una contraseña así es lo bastante segura como para no ser crackeada tan fácilmente como la anterior, ganando así mucha seguridad.

5) Conclusión

Después de obtener el servidor Debian comprometido, hemos sido capaces de no solo realizar un análisis forense con la ayuda de Autopsy, sino que con herramientas cómo nmap, hydra, chkrrotkit, ufw, ClamAV,… Sinó también de mitigar y asegurar un servidor comprometido.

Para ello hemos hecho uso de escáner de rootkits y malware, modificado la configuración de distintos servícios y la instalación extra de firewall y antivirus.

Además, hemos realizado un mini examen de auditoría, el cual nos ha salido con una seguridad de aproximadamente 7/10. Las opciones a mejorar tocan temas poco conocidos por mi, así que por esta razón entiendo que dentro de mis conocimientos, he elaborado una mitigación correcta y decente.

```
Hardening index : 63 [###########       ]
Tests performed : 262
Plugins enabled : 1

Components:
- Firewall              [V]
- Malware scanner       [V]
```

Fase_2: Detectar y corregir una vulnerabilidad distinta

Llegados a este punto, sucede algo que no me esperaba.
Dado que en la fase 1 ya neutralizamos la gran mayoría de errores y vulnerabilidades, al pasar el examen de auditoría con la herramienta **Lynis**, nos salió un total de una protección de 7 sobre 10.

Los demás puntos que aconseja a mejorar la herramienta, trata temas los cuales desconozco y que se escapan de mi área de conocimiento.
Es por ello que no he corregido una vulnerabilidad más.

De igual forma, confío en que las acciones varias que he realizado en la fase 1 son las suficientes para asegurar una buena mitigación y corrección del servidor comprometido, además de las múltiples capas de seguridad añadidas.

Fase_3: Plan de respuesta de incidentes y certificación

Introducción

Tras el incidente de seguridad en el servidor crítico de 4Geeks Academy, he desarrollado un plan integral de respuesta a incidentes basado en el estándar NIST SP 800-61, junto con un Sistema de Gestión de Seguridad de la Información (SGSI) alineado con la norma ISO 27001.

Este documento detalla las acciones para mitigar el ataque actual, prevenir futuros incidentes y garantizar la protección de los datos sensibles de la organización.

Plan de Respuesta a Incidentes

1. Preparación

Para responder eficazmente a futuros incidentes, se ha establecido un equipo CSIRT (Computer Security Incident Response Team) con roles definidos:

- Líder de Respuesta: Coordina las acciones y toma decisiones estratégicas.
- Analista Forense: Investiga los logs y artefactos para identificar el alcance del ataque.
- Administrador de Sistemas: Aplica parches y restaura los servicios afectados.
- Responsable de Comunicación: Informa a las partes interesadas, garantizando transparencia y cumplimiento legal.

Herramientas implementadas:

- Análisis forense: Autopsy para investigar archivos y procesos maliciosos.
- Contención: Reglas de firewall (iptables) para bloquear direcciones IP sospechosas.

2. Detección y Análisis

- Durante la investigación del incidente, se identificaron los siguientes indicadores de compromiso:
- Accesos SSH no autorizados: Detectados en /var/log/auth.log.
- Servicios vulnerables: Apache con una versión desactualizada, explotada para **ejecución remota de código** (RCE).
- Se priorizó la contención del backdoor y la escalada de privilegios como riesgos críticos.

3. Contención

Acciones inmediatas:

- Aislamiento del servidor de la red para evitar la propagación del ataque.
- Cambio de todas las contraseñas y revocación de claves SSH comprometidas.

Acciones a largo plazo:

- Actualización de los servicios vulnerables ***(sudo apt upgrade apache2)***.
- Cierre de puertos innecesarios con UFW (sudo ufw deny 1337).

4. Erradicación

Eliminación de archivos maliciosos:
bash
*sudo find / -name "*backdoor*" -delete*
Restauración de archivos críticos desde backups, verificando su integridad con sha256sum.
NOTA: No ha sido necesário en nuestro caso

5. Recuperación

- Validación de los servicios restaurados (systemctl status apache2).
- Implementación de ClamAV para segurizar los programas y ficheros post-inidente.

6. Lecciones Aprendidas

- El análisis post-incidente reveló que la falta de actualizaciones automáticas y la configuración insegura de SSH facilitaron el ataque. Como mejora, se implementarán:
- Parches automáticos con unattended-upgrades.
- Hardening de SSH (deshabilitar acceso root y autenticación por contraseña).

Sistema de Gestión de Seguridad de la Información (ISO 27001)

1. Análisis de Riesgos

- Se identificaron los siguientes activos críticos y amenazas:
- Servidor web (Apache): Vulnerable a RCE si no se parchea.
- Base de datos de estudiantes: Riesgo de fuga por inyección SQL.
- Claves SSH: Exposición a ataques de fuerza bruta.

2. Políticas de Seguridad

Protección de Datos (DLP):

- Cifrado de archivos sensibles con ecryptfs.
- Restricción de permisos (chmod 600 para archivos de configuración).

Respaldos:

- Copias diarias automatizadas con rsync a un servidor seguro.

3. Controles Técnicos

- Hardening de SSH:
  - bash

- ➢ /etc/ssh/sshd_config:
  PermitRootLogin no
  PasswordAuthentication no

❖ Monitoreo: Alertas del SIEM para patrones de ataque conocidos.

4. Conformidad y Auditoría

❖ Checklist ISO 27001:

❖ Anexo A.12 (protección contra malware).

❖ Anexo A.14 (seguridad en desarrollo).

❖ Auditorías trimestrales:
- ➢ bash
- ➢ lynis audit system

Conclusión

En este proyecto, se ha llevado a cabo un análisis forense exhaustivo y un proceso de ***hardening*** en el servidor debian comprometido de 4Geeks Academy.
Aplicando metodologías como ISO 27001 y NIST SP 800-61 para garantizar una respuesta efectiva ante incidentes.

Los logros principales son la **identificación y mitigación de vulnerabilidades:**

- Se detectó acceso no autorizado mediante FTP anónimo y SSH vulnerable, explotado con fuerza bruta.
- Se eliminaron servicios inseguros (vsftpd) y se reforzó SSH con autenticación por claves y configuración restrictiva.
- Se actualizó el sistema y se aplicó un firewall (UFW) para restringir accesos no autorizados.

En cuanto al malware respecta, este se ha tratado mediante el uso de **Autopsy**, **chkrootkit** y **ClamAV** para descartar backdoors o malware persistente.

Para el **plan de respuesta a incidentes** hemos definido los puntos:

- Contención del ataque, erradicación de archivos maliciosos y recuperación segura del servidor.
- Implementación de copias de seguridad, monitoreo con SIEM y parches automáticos para prevenir futuros ataques.

En resumen, aunque el servidor quedó asegurado con un nivel de protección del 7/10, se identificaron áreas de mejora en configuraciones avanzadas. Este proyecto demuestra la importancia del análisis proactivo, el hardening continuo y un plan de respuesta sólido para proteger infraestructuras críticas.

Una recomendación clave es **mantener actualizaciones automáticas**, **auditorías periódicas** y **formación en ciberseguridad** para el equipo técnico.

El servidor de 4Geeks Academy ahora está más seguro, resiliente y preparado contra futuros ciberataques.