



Tecnicatura Universitaria en Programación

Arquitectura y Sistemas Operativos

Profesor: Osvaldo Falabella

“Seguridad en los Sistemas”

Alumnos:

Agustín Emiliano Sotelo Carmelich (agustinemiliano22@gmail.com)

Bruno Giuliano Vapore (brunogvapore@gmail.com)

Fecha de Entrega: 05/06/2025

Índice

1. Introducción.....	1
2. Marco Teórico.....	3
3. Caso Práctico.....	7
4. Metodología Utilizada.....	12
5. Resultados Obtenidos.....	15
6. Conclusiones.....	17
7. Bibliografía.....	19
8. Anexos.....	21

1. Introducción

En un mundo globalizado en constante evolución, cuyos avances tecnológicos avanzan a pasos agigantados, las sociedades tienden a apoyarse en las nuevas tecnologías y en la digitalización de la información. Por ello, resulta necesario que los sistemas informáticos cuenten con medidas de seguridad suficientes para garantizar la protección de los datos personales, información sensible y los derechos de las personas que pueden verse afectados ante amenazas e intrusos externos.

La seguridad en los sistemas resulta ser un pilar fundamental en el estudio de las ciencias de la computación y, especialmente, en la Tecnicatura Universitaria en Programación. Los programadores deben comprender la seguridad en los sistemas operativos en los entornos que trabajen, en tanto los riesgos que conlleva subestimar las vulnerabilidades del sistema, puede acarrear consecuencias irreparables.

Sentado ello, a fin de aplicar los conocimientos teóricos de la materia en un escenario práctico y relevante, en miras a profundizar el tema elegido, el proyecto se centra en la investigación del proceso denominado como “hardening” (endurecimiento) de un servidor web Apache HTTP Server en un entorno Linux (Ubuntu Server).

Se definen como objetivos a alcanzar con este trabajo, los siguientes: establecer un entorno de laboratorio controlado para la práctica de hardening; implementar y validar diversas medidas de seguridad en un servidor web Apache; comprender la interconexión de conceptos de seguridad, redes, sistemas operativos y scripting; documentar el proceso, los resultados y las dificultades encontradas, fomentando el análisis crítico y la resolución de problemas.

Este proyecto busca no solo configurar un servidor de manera segura, sino también validar cada paso y entender las implicancias de cada medida de seguridad, abordando las vulnerabilidades críticas que, de no ser mitigadas, pueden comprometer seriamente la integridad, confidencialidad y disponibilidad de la información y los servicios en línea.

2. Marco Teórico

El hardening de sistemas es un conjunto de prácticas y configuraciones destinadas a reducir la superficie de ataque de un sistema informático, minimizando las vulnerabilidades y fortaleciendo sus defensas. Este proceso se basa en principios como el mínimo privilegio, la eliminación de componentes innecesarios y la configuración segura por defecto.

La revista UNIR, en una publicación titulada “*¿Qué es el hardening de sistemas en informática?*” refiere que “*Los expertos indican que el hardening en informática se refiere al proceso de asegurar un sistema computarizado al aplicar medidas de seguridad adicionales para reducir su vulnerabilidad ante posibles ataques. Estas acciones incluyen varias acciones diferentes que pueden comprender configuraciones, ajustes y políticas diseñadas para proteger los componentes del sistema (servidores, redes, aplicaciones y sistemas operativos), contra amenazas externas o internas. El objetivo principal del hardening es minimizar los riesgos y fortalecer la seguridad general del sistema.*

” (UNIR, 2023).

Podemos profundizar entonces en que las medidas de hardening abordan amenazas comunes como la exposición de información del servidor, el acceso no autorizado a archivos, la interceptación de datos en tránsito, la explotación de vulnerabilidades conocidas y la falta de monitoreo, entre otros.

Para un correcto desarrollo de lo que es el proceso de hardening y la seguridad en los sistemas, es importante traer a colación algunos conceptos claves que se han estudiado a lo largo de los módulos de la materia “Arquitectura y Sistemas Operativos” de la Tecnicatura Universitaria en Programación. Se destacan los siguientes conceptos:

- Vulnerabilidad: “*Debilidad en un sistema que puede ser explotada por atacantes para comprometer su seguridad*” (Módulo 8: documento “Introducción a la seguridad en sistemas operativos”).
- Firewall: “*Sistema de seguridad que controla y filtra el tráfico de red para prevenir accesos no autorizados*” (Idem anterior).

- Cifrado: “*Técnica utilizada para proteger la información mediante algoritmos de codificación*” (Idem anterior).
- Gestión de permisos: “...*El uso adecuado de permisos en archivos y directorios permite restringir accesos no autorizados y minimizar la posibilidad de alteraciones maliciosas. Además, una correcta gestión de usuarios y grupos, junto con la implementación de políticas de privilegios mínimos, reduce el riesgo de escalamiento de privilegios por parte de actores malintencionados. La aplicación de herramientas como chmod, chown y umask permite establecer controles precisos sobre quién puede leer, modificar o ejecutar ciertos archivos, fortaleciendo así la postura de seguridad del sistema*” (Módulo 8: documento “Seguridad en sistemas operativos”).
- Gestión de Parches: “*Las actualizaciones y parches son fundamentales para corregir vulnerabilidades conocidas y mejorar la estabilidad del sistema*” (Módulo 8: documento “Seguridad Avanzada y Mantenimiento Preventivo”).

Estos conceptos deben ser incorporados para un estudio adecuado del proceso de hardening y la seguridad de los sistemas. De esta manera, se aborda la instalación y configuración de servicios web esenciales como Apache HTTP Server (Módulo 6: Gestión de Servicios). Se utilizan fundamentos de redes, como protocolos TCP/IP, puertos y HTTP/HTTPS para configurar la conectividad y la seguridad en la comunicación (Módulo 4: Redes). Se emplean comandos de Bash para la administración del sistema y se explora la automatización de tareas de seguridad (Módulo 3: Introducción a Scripting (Bash) y Comandos básicos). Y, finalmente, se establece un entorno de laboratorio seguro y reproducible utilizando máquinas virtuales (Módulo 7: Virtualización).

Queda asentado que la interconexión de estos módulos resulta fundamental. Esta integración demuestra que la seguridad es un campo multidisciplinario que requiere una visión integral de la arquitectura y los sistemas operativos.

Ahondando en algunos riesgos en los que se encuentran expuestos los sistemas, a continuación se presenta una tabla que reúne algunas de las vulnerabilidades identificadas y las estrategias de mitigación.

Tipo de Vulnerabilidad	Descripción	Estrategia de Mitigación
Configuración por Defecto Insegura	Uso de configuraciones predeterminadas que exponen servicios o información.	Uso del Firewall para restringir acceso. Configuración de permisos de archivos.
Permisos de Archivos/Directorios Excesivos	Archivos o directorios con permisos de escritura o lectura para usuarios no autorizados.	Configuración de permisos y roles de usuarios (principio de menor privilegio).
Servicios/Puertos Innecesarios Expuestos	Servicios no utilizados o puertos abiertos que no son esenciales para la operación del servidor web.	Uso del Firewall para cerrar puertos innecesarios.
Software Desactualizado	Versiones antiguas del sistema operativo, servidor web o librerías con vulnerabilidades conocidas.	Gestión de parches y actualizaciones.
Credenciales Débiles	Uso de contraseñas fáciles de adivinar o por defecto para acceso administrativo.	Implementación de autenticación multifactor (MFA). Políticas de contraseñas robustas.
Falta de Cifrado de Tráfico	Comunicación HTTP sin cifrar, exponiendo datos sensibles.	Implementación de cifrado (HTTPS/TLS).

En base a ello, y considerando también las principales vulnerabilidades expuestas en la materia, la arquitectura de seguridad que se pretende implementar establece una defensa en capas para el servidor web: el Firewall es el punto de entrada controlado; el acceso administrativo es el canal seguro de gestión; y la gestión de permisos es la capa interna que protege los recursos del servidor web. Esta estructura asegura una defensa integral contra diversas amenazas.

En un principio, Internet representa el origen de todas las conexiones y el Firewall actúa como la primera barrera defensiva. Este componente filtra el tráfico de Internet, permitiendo únicamente las conexiones necesarias para el servidor web (HTTP en puerto 80, HTTPS en puerto 443) y el acceso administrativo (SSH en puerto 22). Todo lo demás es denegado por defecto.

Las solicitudes web permitidas por el firewall son dirigidas al servidor web Apache para el procesamiento del contenido web (HTTP/HTTPS). Luego, el tráfico SSH permitido por el firewall se utiliza para el acceso de administración al servidor. Este acceso está protegido por medidas de autenticación fuerte, como la autenticación multifactor (MFA), para asegurar que solo usuarios autorizados puedan gestionar el sistema.

Directamente vinculada al acceso administrativo, la gestión de permisos controla el acceso a los archivos y directorios del sistema operativo y del servidor web. Los administradores configuran permisos estrictos para asegurar que los procesos del servidor web solo tengan los permisos de lectura necesarios y que ningún usuario no autorizado pueda modificar archivos críticos o acceder a información sensible.

Finalmente, el servidor web accede al contenido web (HTML, CSS, imágenes) y a sus propios archivos de configuración, siempre bajo las restricciones impuestas por la gestión de permisos. La configuración se refiere a los archivos de configuración de Apache que definen cómo el servidor opera, qué información expone y cómo maneja las solicitudes. Estos archivos están protegidos por la gestión de permisos.

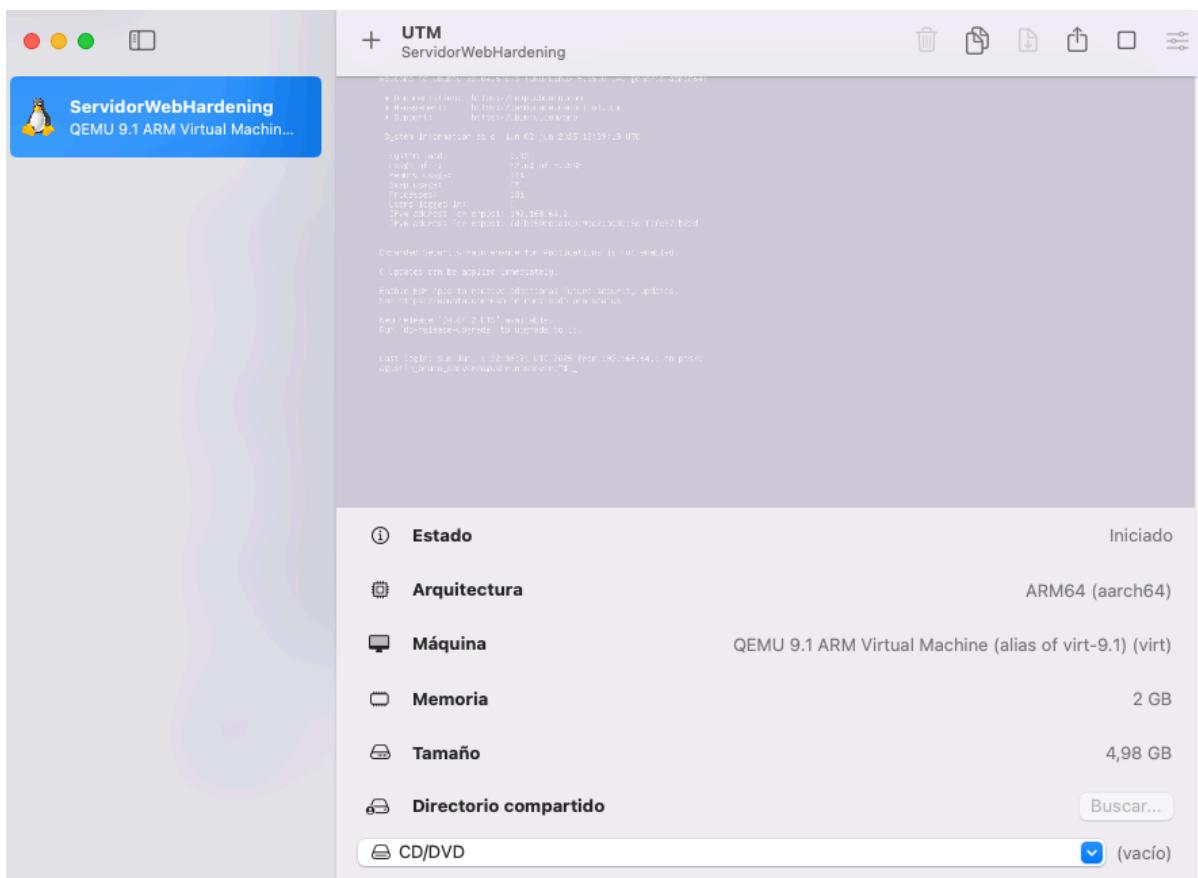
Podemos resumir el significado de un servidor web conforme se expone en la publicación “*¿Cuáles son los servidores web más utilizados?*” y que a continuación se detalla: “*El servidor web es responsable de garantizar que la comunicación entre el servidor y el cliente sea segura y sin fallos. El software funciona como un enlace entre dos máquinas (el servidor físico y el dispositivo de un usuario). Cuando un usuario realiza una petición, el servidor web coge los archivos del servidor físico y los entrega al usuario. De modo que los servidores web tienen que servir diferentes páginas a diferentes usuarios al mismo tiempo.*” (Stackscale, 2022).

Esa responsabilidad recae en un adecuado manejo de la seguridad en el servidor, para proteger en forma integral su contenido (data y código), recursos físicos de accesos no autorizados, destrucción o alteración maliciosa o accidentales (Stallings, 1998, prefacio, x); y, asimismo, a los usuarios involucrados. El caso práctico pondrá en evidencia los conocimientos establecidos en este marco teórico.

3. Caso Práctico

El caso práctico aborda una situación común y crítica: un servidor web Apache/Nginx en un entorno Linux expuesto a internet sin configuraciones de seguridad adecuadas. Esta negligencia puede resultar en accesos no autorizados, la manipulación de contenido web, o incluso ataques de denegación de servicio que interrumpan la disponibilidad del servicio. El problema concreto a resolver es la implementación de un conjunto de medidas de hardening para proteger este servidor, específicamente a través de la configuración de un firewall y la gestión de permisos de archivos y directorios críticos.

Para este caso práctico, se estableció un entorno de laboratorio aislado utilizando una máquina virtual con UTM. Se instaló una distribución de Linux (Ubuntu Server) como sistema operativo base. Dentro de esta máquina virtual, se configuró un servidor web Apache2 básico, accesible inicialmente sin restricciones de seguridad significativas.



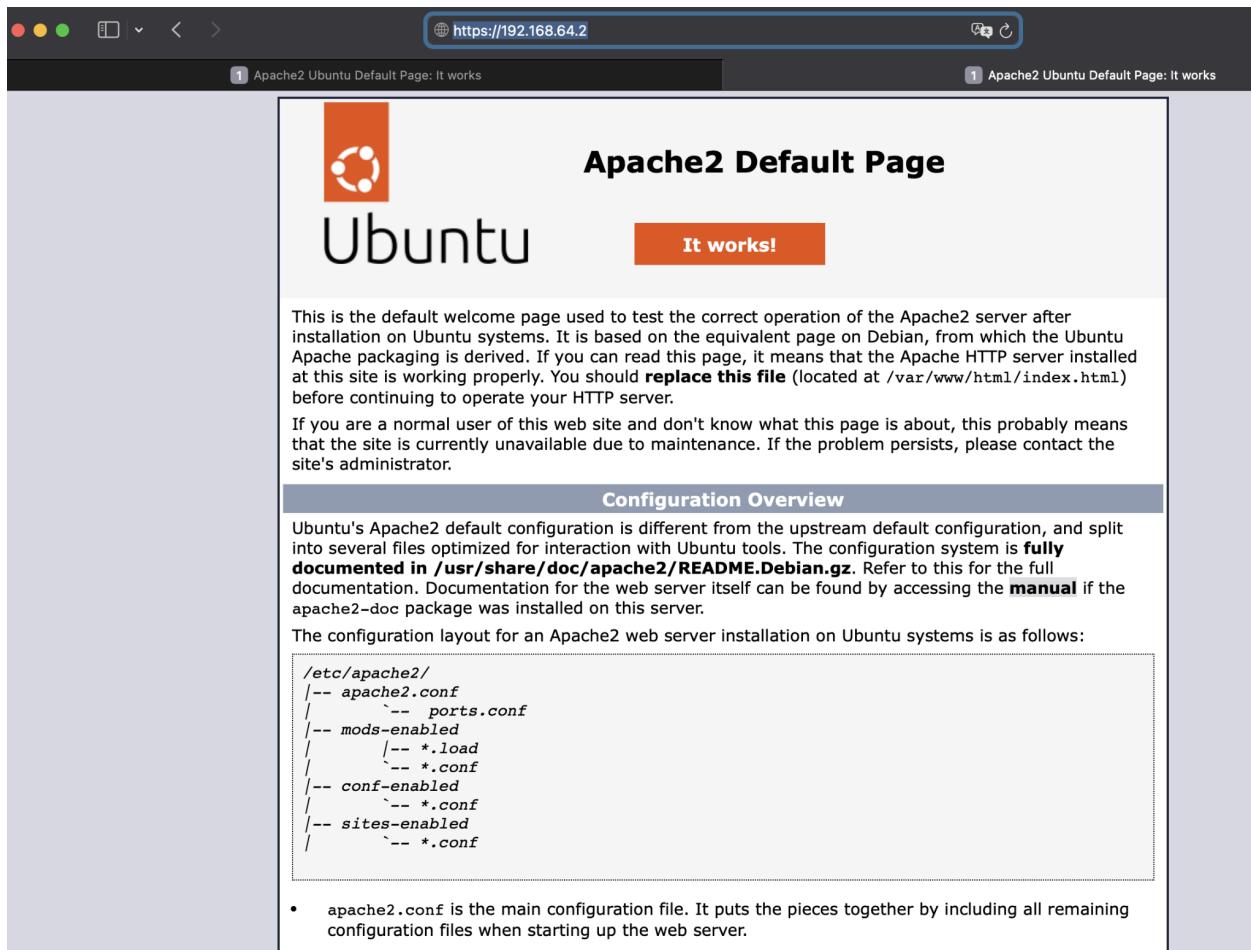
Los pasos detallados para la implementación de las medidas de seguridad fueron los siguientes:

- I. **Configuración Inicial del Servidor Web:** Se instaló Apache2 y se verificó su funcionamiento, confirmando que era accesible desde la red de la máquina virtual. Se creó una página HTML simple para servir como contenido de prueba.

Comandos:

```
sudo apt update
sudo apt install apache2 -y
sudo systemctl start apache2
sudo systemctl enable apache2
sudo systemctl status apache2
```

La salida de sudo systemctl status apache2 mostró Active: active (running), y al acceder desde el navegador, <http://192.168.64.2/> mostró la página de bienvenida "It works!".



(Ver Anexos para más ilustraciones)

II. Implementación del Firewall (UFW): Se eligió UFW (Uncomplicated Firewall) por su facilidad de uso en entornos Linux. Se configuraron reglas para permitir únicamente el tráfico SSH (puerto 22), HTTP (puerto 80) y HTTPS (puerto 443), bloqueando todo lo demás.

Comandos:

`sudo apt install ufw`

`sudo ufw default deny incoming`

`sudo ufw default allow outgoing`

`sudo ufw allow ssh # Puerto 22 para acceso remoto`

```
sudo ufw allow http      # Puerto 80 para tráfico web no cifrado
sudo ufw allow https     # Puerto 443 para tráfico web cifrado
sudo ufw enable
sudo ufw status verbose
```

```
[agustin_bruno_server@agusbrunoserver:~$ sudo apt install ufw -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ufw is already the newest version (0.36.1-4ubuntu0.1).
ufw set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[agustin_bruno_server@agusbrunoserver:~$ sudo ufw status verbose
Status: inactive
[agustin_bruno_server@agusbrunoserver:~$ sudo ufw default deny incoming
[sudo ufw default allow outgoing
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
[agustin_bruno_server@agusbrunoserver:~$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
[agustin_bruno_server@agusbrunoserver:~$ sudo ufw allow http
[sudo ufw allow https
Rules updated
Rules updated (v6)
Rules updated
Rules updated (v6)
[agustin_bruno_server@agusbrunoserver:~$ sudo ufw enable
[Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
[agustin_bruno_server@agusbrunoserver:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
To          Action    From
__          ____     ____
22/tcp      ALLOW IN  Anywhere
80/tcp      ALLOW IN  Anywhere
443         ALLOW IN  Anywhere
22/tcp (v6) ALLOW IN  Anywhere (v6)
80/tcp (v6) ALLOW IN  Anywhere (v6)
443 (v6)   ALLOW IN  Anywhere (v6)
```

(Ver Anexos para más ilustraciones)

III. Gestión de Permisos de Archivos y Directorios: Se enfocó en el directorio /var/www/html/ (ubicación por defecto de los archivos web de Apache) y los archivos de configuración de Apache (/etc/apache2/).

Se aseguró que el propietario del directorio /var/www/html/ fuera root y el grupo www-data, con permisos que permitieran al servidor web leer los archivos, pero no escribir en ellos directamente desde el proceso del servidor. También se

verificó que los archivos de configuración de Apache tuvieran permisos restrictivos, accesibles solo por root.

Comandos:

```
sudo chown -R root:www-data /var/www/html/
```

```
sudo chmod -R 755 /var/www/html/ # Directorios: rwx para propietario, r-x para grupo y otros; Archivos: rw- para propietario, r-- para grupo y otros
```

```
sudo chmod 644 /etc/apache2/apache2.conf
```

```
agustin_bruno_server@agusbrunoserver:~$ ps aux | grep apache2 | grep -v grep
root      19245  0.0  0.2  7584  4532 ?        Ss   22:40  0:00 /usr/sbin/apache2 -k start
www-data  19247  0.0  0.2 1935832 5416 ?        Sl   22:40  0:00 /usr/sbin/apache2 -k start
www-data  19248  0.0  0.2 1935832 5432 ?        Sl   22:40  0:00 /usr/sbin/apache2 -k start
www-data  19378  0.0  0.0   3400   156 ?        Ss   22:40  0:00 /usr/bin/htcacheClean -d 120 -p /var/cache/apache2/m
od_cache_disk -l 300M -n
agustin_bruno_server@agusbrunoserver:~$ ls -l /var/www/html/
total 12
-rw-r--r-- 1 root root 10671 Jun  1 22:40 index.html
agustin_bruno_server@agusbrunoserver:~$ sudo chown -R root:root /var/www/html/
[sudo] password for agustin_bruno_server:
agustin_bruno_server@agusbrunoserver:~$ sudo chmod -R root:root /var/www/html/
agustin_bruno_server@agusbrunoserver:~$ sudo find /var/www/html/ -type d -exec chmod 755 {} \; # Permisos para directorios (lectura, escritura, ejecución para propietario; lectura y ejecución para grupo y otros)
    sudo find /var/www/html/ -type f -exec chmod 644 {} \; # Permisos para archivos (lectura y escritura para propietario; lectura para grupo y otros)
agustin_bruno_server@agusbrunoserver:~$ |
```

(Ver Anexos para más ilustraciones)

IV. Validación del funcionamiento: La validación de las medidas de seguridad implementadas se realizaron mediante pruebas específicas para cada configuración.

En cuanto al Firewall, se intentó acceder al servidor web desde la máquina host a través de los puertos 80 y 443, confirmando que el acceso era exitoso. (Comando: curl <http://192.168.64.2/>). Por otro lado, se intentó acceder a un puerto no permitido (ej. puerto 21 para FTP) desde la máquina host, verificando que la conexión fuera rechazada o se agotara el tiempo de espera. (Comando: nmap -p 21 192.168.64.2).

Con relación a los permisos de archivos, se intentó modificar un archivo en /var/www/html/ como un usuario no privilegiado (ej. testuser), confirmando que la operación fue denegada. (Comando: sudo -u testuser echo "Contenido malicioso" > /var/www/html/index.html). Asimismo, se verificó que el servidor web (proceso www-data) aún podía leer y servir los archivos correctamente.

Este caso práctico demostró la aplicación concreta de los conceptos teóricos de seguridad, transformando un servidor web vulnerable en un entorno más robusto y controlado. La especificidad del problema permitió una implementación tangible y una validación verificable de los resultados.

4. Metodología Utilizada

El desarrollo de este trabajo siguió una metodología estructurada, dividida en fases de investigación, diseño, implementación y pruebas, finalizando con la elaboración del informe, con tareas asignadas a cada integrante del grupo y la revisión en común, realizando así un trabajo más eficiente del tiempo empleado y de forma colaborativa.

01. Fase de Investigación Previa:

- a. Se inició con una revisión exhaustiva de la documentación oficial de Apache2, UFW y las mejores prácticas de seguridad para servidores Linux. Esto incluyó la consulta de guías de *hardening* de organismos de seguridad y artículos especializados.
- b. Se priorizó la documentación oficial y los recursos recomendados por la cátedra para asegurar la fiabilidad de la información.
- c. Se consultaron las fuentes de la bibliografía utilizada para adquirir mayores conocimientos, realizar el caso práctico y elaborar el informe.
- d. Se analizaron las vulnerabilidades comunes en servidores web, como configuraciones por defecto inseguras y permisos de archivos laxos, basándose en la información de seguridad de los sistemas operativos.

02. Fase de Diseño de la Solución:

- a. Se planificó la arquitectura del entorno de laboratorio, optando por una máquina virtual con UTM para asegurar un entorno aislado y controlable.

- b. Se definieron las reglas específicas del firewall necesarias para el servidor web, considerando los puertos HTTP, HTTPS y SSH. Se decidió utilizar UFW por su simplicidad y eficacia para este tipo de configuración.
- c. Se diseñaron las políticas de permisos para los directorios y archivos críticos del servidor web, aplicando el principio de menor privilegio. Esto implicó determinar los usuarios y grupos adecuados para cada recurso.

03. Fase de Implementación:

- a. Se procedió a la instalación y configuración del sistema operativo Linux dentro de la máquina virtual.
- b. Se instaló y configuró el servidor web Apache2, asegurando su funcionamiento básico antes de aplicar las medidas de seguridad.
- c. Se implementaron las reglas de firewall definidas en la fase de diseño, utilizando comandos de UFW.
- d. Se ajustaron los permisos de archivos y directorios críticos del servidor web, empleando comandos como chmod y chown, habilidades aprendidas en el módulo de scripting y comandos básicos.

04. Fase de Pruebas y Validación:

- a. Se realizaron pruebas funcionales para asegurar que el servidor web seguía siendo accesible a través de los puertos permitidos y que la redirección HTTPS funcionaba.
- b. Se llevaron a cabo pruebas de seguridad, intentando acceder a puertos bloqueados y modificar archivos protegidos con usuarios no autorizados. Esto permitió verificar la efectividad de las medidas implementadas.

-
- c. Se documentaron los resultados de estas pruebas, incluyendo capturas de pantalla y salidas de comandos.

05. Fase de Elaboración del Informe:

- a. Se redactó el presente informe final con los recursos disponibles y obtenidos durante las demás fases del proyecto.
- b. Se acompañó al trabajo integrador un video explicativo de lo desarrollado.

A su vez, se emplearon diversas herramientas y recursos, que facilitaron la implementación y validación de las medidas de seguridad, entre ellas:

- Virtualización: UTM como plataforma para crear el entorno de máquina virtual aislado, lo que permitió realizar configuraciones y pruebas sin afectar el sistema operativo principal.
- Sistema Operativo: Ubuntu Server 22.04 LTS (una distribución de Linux) como sistema operativo.
- Servidor Web: Apache HTTP Server (Apache2).
- Firewall: UFW (Uncomplicated Firewall), una interfaz simplificada para iptables.
- Scripting y Comandos: La terminal Bash fue la interfaz principal para ejecutar comandos de configuración del sistema, gestión de permisos (chmod, chown, useradd, groupadd) y control del firewall.
- Herramientas de Red: curl para pruebas de acceso HTTP/HTTPS y nmap para el escaneo de puertos y verificación de la configuración del firewall.
- Generación de Certificados: openssl para la creación de certificados SSL/TLS autofirmados.
- Gestión de Servicios: systemctl para controlar el estado de los servicios (Apache, UFW).

- Gestión de Logs: tail para monitorear logs en tiempo real, y logrotate para su gestión.
- Edición de Archivos: Nano en la terminal para ediciones rápidas y Visual Studio Code con la extensión "Remote - SSH" para una edición más cómoda de los archivos de configuración remotos.

La selección de estas herramientas se basó en su disponibilidad, su relevancia con los contenidos del curso y su capacidad para simular escenarios reales de seguridad, como se puede observar en la sección de Anexos correspondiente.

5. Resultados Obtenidos

El caso práctico de *hardening* del servidor web Apache en un entorno Linux arrojó resultados positivos, demostrando la efectividad de las medidas de seguridad implementadas. Se logró una reducción de la superficie de ataque del servidor, limitando el acceso a los servicios esenciales y protegiendo los recursos del sistema.

Las principales medidas de seguridad implementadas funcionaron correctamente según lo previsto. La configuración del firewall fue exitosa y se bloquearon los intentos de conexión no autorizados a puertos distintos de los configurados (SSH, HTTP, HTTPS). Las pruebas con nmap confirmaron que solo los puertos 22, 80 y 443 estaban accesibles desde el exterior, mientras que otros puertos comunes estaban correctamente filtrados. Esto es crucial para prevenir escaneos de puertos y ataques dirigidos a servicios innecesariamente expuestos.

Por otra parte, las políticas de permisos aplicadas al directorio /var/www/html/ y a los archivos de configuración de Apache (/etc/apache2/) se validaron con éxito. Los usuarios no privilegiados no pudieron modificar los archivos del sitio web, lo que previene la inyección de código malicioso o la alteración no autorizada del contenido. El proceso del servidor web mantuvo la capacidad de leer los archivos

necesarios para servir la página, asegurando la funcionalidad sin otorgar privilegios excesivos de escritura.

El servidor web Apache continuó sirviendo contenido HTTP/HTTPS de manera normal a través de los puertos permitidos.

Los aspectos que funcionaron correctamente fueron la implementación precisa de las reglas de UFW y la configuración de permisos del sistema de archivos. La capacidad de UFW para aplicar reglas de "denegar por defecto" y luego "permitir explícitamente" los servicios necesarios demostró ser una estrategia robusta para la seguridad de red. De manera similar, la aplicación del principio de menor privilegio a través de chmod y chown para los archivos del servidor web garantizó que solo los procesos autorizados tuvieran el acceso necesario, evitando posibles escaladas de privilegios en caso de compromiso de una cuenta de usuario.

Durante el proceso de implementación, se encontraron algunas dificultades que fueron resueltas, lo que proporcionó valiosas lecciones aprendidas. Por ejemplo, **el bloqueo de tráfico legítimo por el Firewall**. Inicialmente, al configurar UFW, se olvidó permitir el tráfico SSH (puerto 22) antes de habilitar el firewall. Esto resultó en la pérdida de la conexión SSH a la máquina virtual, requiriendo el acceso a través de la consola de UTM para corregir la regla y permitir el acceso remoto. Este incidente subrayó la importancia de verificar la conectividad esencial antes de activar reglas restrictivas y la secuencia correcta de comandos.

También hubo **problemas de permisos con archivos nuevos**. Al agregar nuevos archivos o directorios al sitio web, se observó que no siempre heredan los permisos correctos, lo que causaba errores de "Forbidden" al intentar acceder a ellos desde el navegador. Este problema se resolvió ejecutando chmod -R 755 /var/www/html/ y chown -R root:www-data /var/www/html/ nuevamente después de añadir contenido, asegurando que todos los archivos y subdirectorios tuvieran los permisos adecuados para el servidor web. Se destaca la necesidad de una gestión continua de permisos, especialmente cuando se modifica el contenido del servidor.

6. Conclusiones

Este trabajo integrador sobre el *hardening* de servidores web en entornos Linux ha sido una experiencia de aprendizaje fundamental que ha permitido consolidar y aplicar directamente los conocimientos adquiridos en la materia de Arquitectura y Sistemas Operativos. Se han cumplido los objetivos de establecer un entorno de laboratorio, implementar y validar medidas de seguridad clave, y comprender la interconexión de conceptos fundamentales. La aplicación práctica de firewall, gestión de permisos, cifrado y automatización ha demostrado de manera concreta cómo se protegen los sistemas, reforzando las habilidades esenciales para un técnico universitario en programación.

Se logró analizar las principales vulnerabilidades que afectan a los servidores web, comprendiendo que muchas de ellas derivan de configuraciones por defecto o de la falta de gestión de permisos. Los objetivos de diseñar e implementar un firewall robusto y de configurar permisos de archivos y directorios críticos se cumplieron satisfactoriamente. La validación empírica de estas medidas, a través de pruebas específicas, confirmó su efectividad en la reducción de la superficie de ataque del servidor.

Las dificultades que surgieron durante el proceso, como el bloqueo accidental del acceso SSH o los problemas con los permisos de archivos recién creados, fueron oportunidades valiosas para el aprendizaje. La necesidad de solucionar estos problemas en tiempo real no solo fortaleció las habilidades de depuración y resolución de problemas, sino que también profundizó la comprensión de la interacción entre las distintas capas de seguridad. Estas experiencias subrayan que la seguridad de sistemas es un proceso iterativo que requiere atención al detalle y una capacidad constante de adaptación y corrección.

En cuanto a posibles mejoras o extensiones futuras, el trabajo podría ampliarse significativamente. Se podría integrar un sistema de detección de intrusiones (IDS) o un sistema de prevención de intrusiones (IPS) para monitorear y reaccionar a actividades sospechosas en tiempo real. Otra extensión valiosa sería la exploración de la automatización de la gestión de parches y actualizaciones

utilizando herramientas como Ansible o Puppet, lo que garantizaría que el servidor se mantenga actualizado de forma continua y eficiente. Además, se podría profundizar la investigación de la implementación de certificados SSL/TLS para HTTPS de manera automatizada (por ejemplo Let's Encrypt) y la configuración de seguridad a nivel de aplicación (por ejemplo Web Application Firewall) para una protección más completa. Finalmente, si el servidor web se desplegara en contenedores Docker, se abrirían nuevas vías para explorar la seguridad de contenedores, incluyendo el escaneo de imágenes en busca de vulnerabilidades y la implementación de políticas de seguridad a nivel de orquestador.

En síntesis, este proyecto ha permitido no solo aplicar conocimientos técnicos específicos de seguridad en sistemas operativos, sino también desarrollar una visión más crítica y estratégica sobre la importancia de la seguridad en el ciclo de vida del desarrollo y la administración de sistemas. La capacidad de identificar, mitigar y validar medidas de seguridad es una habilidad indispensable para cualquier técnico universitario en programación que aspire a construir soluciones informáticas robustas y confiables.

7. Bibliografía

Apache HTTP Server. (s.f.). *Apache HTTP Server Documentation*. Recuperado el

26 de mayo de 2025, de <https://httpd.apache.org/docs/>

NinjaOne. (s.f.). *Guía completa para el hardening de sistemas*. NinjaOne.

Recuperado el 27 de Mayo de 2025, de

<https://www.ninjaone.com/es/blog/complete-guide-to-systems-hardening/>

Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating System Concepts*.

Wiley.

Stackscale. (05 de Septiembre de 2022). *¿Cuáles son los servidores web más*

utilizados? Stackscale. Recuperado el 31 de Junio de 2025, de

<https://www.stackscale.com/es/blog/top-servidores-web/>

Stallings, W. (1998). *Operating systems: internals and design principles*. Prentice

Hall.

Tecnicatura Universitaria en Programación a Distancia. Arquitectura y Sistemas

Operativos. Unidad 1: *Introducción a la Arquitectura de Computadoras*.

Unidad 2: *Introducción a los Sistemas Operativos*. Unidad 3: *Introducción al*

Scripting (BASH) y comandos básicos. Unidad 4: *Redes*. Unidad 5:

Arquitectura de Aplicaciones. Unidad 6: *Gestión de Archivos*. Unidad 7:

Virtualización. Unidad 8: *Seguridad en los Sistemas Operativos*. (2025).

Universidad Tecnológica Nacional.

UFW. (s.f.). Official Ubuntu Documentation. Recuperado el 27 de Mayo de 2025, de

<https://help.ubuntu.com/community/UFW>

UNIR. (s.f.). *¿Qué es el hardening de sistemas en informática?*. Recuperado el 1 de Junio de 2025, de <https://www.unir.net/revista/ingenieria/hardening-que-es/>

8. Anexos

Se adjuntan algunas capturas de pantallas que muestran la preparación del entorno, herramientas y algunos comandos utilizados durante la ejecución del caso práctico.

Ubuntu Server (Linux) e información del sistema:

```
● ● ● ⌂ ⟲ ServidorWebHardening
Ubuntu 22.04.4 LTS agusbrunoserver tty1
agusbrunoserver login: agustin_bruno_server
Password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-140-generic aarch64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

 System information as of dom 01 jun 2025 22:26:15 UTC

 System load: 0.134765625
 Usage of /: 50.3% of 9.75GB
 Memory usage: 10%
 Swap usage: 0%
 Processes: 127
 Users logged in: 0
 IPv4 address for enp0s1: 192.168.64.2
 IPv6 address for enp0s1: fdfb:50be:a100:9dc2:a03b:5eff:fe57:b22d

El mantenimiento de seguridad expandido para Applications está desactivado
Se pueden aplicar 82 actualizaciones de forma inmediata.
Para ver estas actualizaciones adicionales, ejecute: apt list --upgradable
Active ESM Apps para recibir futuras actualizaciones de seguridad adicionales.
Vea https://ubuntu.com/esm o ejecute «sudo pro status»


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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

agustin_bruno_server@agusbrunoserver:~$
```

Ingreso al servidor a través de SSH:

```
> ssh agustin_bruno_server@192.168.64.2
The authenticity of host '192.168.64.2' (192.168.64.2) can't be established.
ED25519 key fingerprint is SHA256:itaayqA4d0dTpxFnTsDeqa1ah8t6VaJIytu2J3ole5g.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.64.2' (ED25519) to the list of known hosts.
agustin_bruno_server@192.168.64.2's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-140-generic aarch64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of dom 01 jun 2025 22:36:21 UTC

System load:          0.0
Usage of /:           50.4% of 9.75GB
Memory usage:         11%
Swap usage:           0%
Processes:            120
Users logged in:     1
IPv4 address for enp0s1: 192.168.64.2
IPv6 address for enp0s1: fdfb:50be:a100:9dc2:a03b:5eff:fe57:b22d

El mantenimiento de seguridad expandido para Applications está desactivado

Se pueden aplicar 82 actualizaciones de forma inmediata.
Para ver estas actualizaciones adicionales, ejecute: apt list --upgradable

Active ESM Apps para recibir futuras actualizaciones de seguridad adicionales.
Vea https://ubuntu.com/esm o ejecute «sudo pro status»

New release '24.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun Jun 1 22:26:15 2025
-bash: warning: setlocale: LC_ALL: cannot change locale (en_US.UTF-8)
agustin_bruno_server@agusbrunoserver:~$
```

Instalación de actualización de paquetes y Apache2:

```
[agustin_bruno_server@agusbrunoserver:~$ sudo apt update
Hit:1 http://ports.ubuntu.com/ubuntu-ports jammy InRelease
Get:2 http://ports.ubuntu.com/ubuntu-ports jammy-updates InRelease [128 kB]
Hit:3 http://ports.ubuntu.com/ubuntu-ports jammy-backports InRelease
Get:4 http://ports.ubuntu.com/ubuntu-ports jammy-security InRelease [129 kB]
Fetched 257 kB in 3s (99.7 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
[agustin_bruno_server@agusbrunoserver:~$ sudo apt 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.
[agustin_bruno_server@agusbrunoserver:~$ sudo apt autoremove -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
[agustin_bruno_server@agusbrunoserver:~$ sudo apt install unattended-upgrades -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unattended-upgrades is already the newest version (2.8ubuntu1).
unattended-upgrades set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
agustin_bruno_server@agusbrunoserver:~$ |
```

```
agu - agustin_bruno_server@agusbrunoserver: ~ -- ssh agustin_bruno_server@192.168.64.2 - 139x47
Setting up ubuntu-standard (1.481.4) ...
Setting up ubuntu-advantage-tools (35.1ubuntu0~22.04) ...
Setting up ubuntu-minimal (1.481.4) ...
Setting up liblvm2cmd2.03:arm64 (2.03.11-2.1ubuntu5) ...
Setting up dmeventd (2:1.02.175-2.1ubuntu5) ...
dm-event.service is a disabled or a static unit not running, not starting it.
Setting up python3-distupgrade (1:22.04.20) ...
Setting up lvm2 (2.03.11-2.1ubuntu5) ...
update-initramfs: deferring update (trigger activated)
Setting up ubuntu-release-upgrader-core (1:22.04.20) ...
Setting up python3-update-manager (1:22.04.21) ...
Setting up ubuntu-server-minimal (1.481.4) ...
Setting up update-manager-core (1:22.04.21) ...
Setting up update-notifier-common (3.192.54.8) ...
update-notifier-download.service is a disabled or a static unit not running, not starting it.
update-notifier-motd.service is a disabled or a static unit not running, not starting it.
Setting up ubuntu-server (1.481.4) ...
Processing triggers for install-info (6.8-4build1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.10) ...
Processing triggers for syslog (8.2112.0-2ubuntu2.2) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for plymouth-theme-ubuntu-text (0.9.5+git20211018-1ubuntu3) ...
update-initramfs: deferring update (trigger activated)
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
Processing triggers for initramfs-tools (0.140ubuntu13.5) ...
update-initramfs: Generating /boot/initrd.img-5.15.0-140-generic
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
systemctl restart cron.service irqbalance.service multipathd.service polkit.service ssh.service udisks2.service
Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart networkd-dispatcher.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service
systemctl restart user@1000.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
agustin_bruno_server@agusbrunoserver:~$ sudo apt install apache2 -y
```

```
agu - agustin_bruno_server@agusbrunoserver: ~ -- ssh agustin_bruno_server@192.168.64.2 - 139x47
agustin_bruno_server@agusbrunoserver:~$ sudo nano /etc/apt/apt.conf.d/50unattended-upgrades
agustin_bruno_server@agusbrunoserver:~$ sudo unattended-upgrades --dry-run --debug
Running on the development release
Starting unattended upgrades script
Allowed origins are: o=Ubuntu,a=jammy, o=Ubuntu,a=jammy-security, o=UbuntuESMApps,a=jammy-apps-security, o=UbuntuESM,a=jammy-infra-security
, o=Ubuntu,a=jammy, o=Ubuntu,a=jammy-security, o=UbuntuESMApps,a=jammy-apps-security, o=UbuntuESM,a=jammy-infra-security
Initial blacklist:
Initial whitelist (not strict):
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-backports_universe_i18n_Translation-en' a=jammy-backports,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=56030 ID:23> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-backports_universe_binary-arm64_Packages' a=jammy-backports,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=136689 ID:22> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-backports_main_i18n_Translation-en' a=jammy-backports,c=main,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=128722 ID:21> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-backports_main_binary-arm64_Packages' a=jammy-backports,c=main,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=442735 ID:20> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_multiverse_i18n_Translation-en' a=jammy-updates,c=multiverse,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=118836 ID:19> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_multiverse_binary-arm64_Packages' a=jammy-updates,c=multiverse,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=138632 ID:18> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_universe_i18n_Translation-en' a=jammy-updates,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=2667564 ID:17> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_universe_binary-arm64_Packages' a=jammy-updates,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=6741787 ID:16> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_restricted_i18n_Translation-en' a=jammy-updates,c=restricted,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=12578532 ID:15> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_restricted_binary-arm64_Packages' a=jammy-updates,c=restricted,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=18520377 ID:14> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_main_i18n_Translation-en' a=jammy-updates,c=main,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=10146845 ID:13> with -32768 pin
Marking not allowed <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_main_binary-arm64_Packages' a=jammy-updates,c=main,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=13247845 ID:12> with -32768 pin
Applying pinning: PkgFilePin(id=23, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-backports_universe_i18n_Translation-en' a=jammy-backports,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch='' site='ports.ubuntu.com' IndexType='Debian Translation Index'
```

```
● ● ● agu - agustin_bruno_server@agusbrunoserver: ~ ssh agustin_bruno_server@192.168.64.2 - 139x47
dates_multiverse_i18n_Translation-en' a=jammy-updates,c=multiverse,v=22.04,o=Ubuntu,l=Ubuntu arch=' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=118836 ID:19>
Applying pinning: PkgFilePin(id=18, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_multiverse_binary-arm64_Packages' a=jammy-updates,c=multiverse,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=138632 ID:18>
Applying pinning: PkgFilePin(id=17, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_universe_i18n_Translation-en' a=jammy-updates,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch=' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=2667564 ID:17>
Applying pinning: PkgFilePin(id=16, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_universe_binary-arm64_Packages' a=jammy-updates,c=universe,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=6741787 ID:16>
Applying pinning: PkgFilePin(id=15, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_restricted_i18n_Translation-en' a=jammy-updates,c=restricted,v=22.04,o=Ubuntu,l=Ubuntu arch=' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=12578532 ID:15>
Applying pinning: PkgFilePin(id=14, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_restricted_binary-arm64_Packages' a=jammy-updates,c=restricted,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=18520377 ID:14>
Applying pinning: PkgFilePin(id=13, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_main_i18n_Translation-en' a=jammy-updates,c=main,v=22.04,o=Ubuntu,l=Ubuntu arch=' site='ports.ubuntu.com' IndexType='Debian Translation Index' Size=10146845 ID:13>
Applying pinning: PkgFilePin(id=12, priority=-32768)
Applying pin -32768 to package_file: <apt_pkg.PackageFile object: filename:'/var/lib/apt/lists/ports.ubuntu.com_ubuntu-ports_dists_jammy-updates_main_binary-arm64_Packages' a=jammy-updates,c=main,v=22.04,o=Ubuntu,l=Ubuntu arch='arm64' site='ports.ubuntu.com' IndexType='Debian Package Index' Size=13247845 ID:12>
Using (^linux-.*[1-9][0-9]*.[0-9]+.[0-9]+-[0-9]+(-.+)?$|^kfreebsd-.*[1-9][0-9]*.[0-9]+.[0-9]+-[0-9]+(-.+)?$|^gnumach-.*[1-9][0-9]*.[0-9]+[0-9]+(-.+)?$|^kernel-[0-9][0-9]*.[0-9]+[0-9]+(-.+)?$|^li
nux-.*[1-9][0-9]*.[0-9]+[0-9]+(-.+)?$|^kfreebsd-.*[1-9][0-9]*.[0-9]+[0-9]+(-.+)?$|^gnumach-.*[1-9][0-9]*.[0-9]+[0-9]+[0-9]+(-.+)?$|^kernel-[1-9][0-9]*.[0-9]+[0-9]+(-.+)?$|^gnumach-.*[1-9][0-9]*.[0-9]+[0-9]+(-.+)?$|^kernel-[1-9][0-9]*.[0-9]+[0-9]+(-.+)?$) regexp to fi
nd kernel packages
Using (^linux-.*5\.15\.0\.-140\.-generic$|^linux-.*5\.15\.0\.-140$|^kfreebsd-.*5\.15\.0\.-140\.-generic$|^kfreebsd-.*5\.15\.0\.-140$|^gnumach-.*5\.15\.0\.-140\.-generic$|^gnumach-.*5\.15\.0\.-140$|^modules-5\.15\.0\.-140$|^modules-5\.15\.0\.-140$|^kernel-5\.15\.0\.-140$|^kernel-5\.15\.0\.-140$|^gnumach-.*5\.15\.0\.-140$|^gnumach-.*5\.15\.0\.-140$|^modules-5\.15\.0\.-140$|^modules-5\.15\.0\.-140$|^kernel-5\.15\.0\.-140$|^kernel-5\.15\.0\.-140$) regexp to find running kernel packages
pkgs that look like they should be upgraded:
Fetched 0 B in 0s (0 B/s)
fetchch.run() result: 0
Packages blacklist due to configfile prompts: []
No packages found that can be upgraded unattended and no pending auto-removals
The list of kept packages can't be calculated in dry-run mode.
agustin_bruno_server@agusbrunoserver:~$ |
```

Estado del servicio de Apache2:

```
● ● ● agu - agustin_bruno_server@agusbrunoserver: ~ ssh agustin_bruno_server@192.168.64.2 - 139x47
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service → /lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache-htcacheclean.service → /lib/systemd/system/apache-htcacheclean.service.
Processing triggers for ufw (0.36.1-4ubuntu0.1) ...
Processing triggers for man-db (2.18.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.10) ...
Scanning processes...
Scanning candidates...
Scanning linux images...
Running kernel seems to be up-to-date.

Restarting services...
Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart networkd-dispatcher.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service
systemctl restart user@1000.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
agusbruno_server@agusbrunoserver:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
     Active: active (running) since Sun 2025-06-01 22:40:34 UTC; 21s ago
       Docs: https://httpd.apache.org/docs/2.4/
 Main PID: 19245 (apache2)
    Tasks: 55 (limit: 2197)
   Memory: 5.1M
      CPU: 15ms
     CGroup: /system.slice/apache2.service
             └─19245 /usr/sbin/apache2 -k start
                 ├─19247 /usr/sbin/apache2 -k start
                 ├─19248 /usr/sbin/apache2 -k start
                 └─19249 /usr/sbin/apache2 -k start

Jun 01 22:40:34 agusbrunoserver systemd[1]: Starting The Apache HTTP Server...
Jun 01 22:40:34 agusbrunoserver apache2[19244]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, i
Jun 01 22:40:34 agusbrunoserver systemd[1]: Started The Apache HTTP Server.
lines 1-16 (END)
```

Dirección de IP:

```
[root@192.168.64.2 agusbrunouser] ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether a2:3b:5e:57:b2:2d brd ff:ff:ff:ff:ff:ff
    inet 192.168.64.2/24 metric 100 brd 192.168.64.255 scope global dynamic enp0s1
        valid_lft 3210sec preferred_lft 3210sec
    inet6 fdfb:50be:a100:9dc2:a03b:5eff:fe57:b22d/64 scope global dynamic mngtmpaddr noprefixroute
        valid_lft 2591983sec preferred_lft 604783sec
    inet6 fe80::a03b:5eff:fe57:b22d/64 scope link
        valid_lft forever preferred_lft forever
agusbrunouser@agusbrunouser:~$
```

Ping para verificar la conexión de IP:

```
agu — agu@Laptop-de-Agustín — ~ — -zsh — 96x25
64 bytes from 192.168.64.2: icmp_seq=49 ttl=64 time=0.994 ms
64 bytes from 192.168.64.2: icmp_seq=50 ttl=64 time=0.448 ms
64 bytes from 192.168.64.2: icmp_seq=51 ttl=64 time=1.485 ms
64 bytes from 192.168.64.2: icmp_seq=52 ttl=64 time=1.079 ms
64 bytes from 192.168.64.2: icmp_seq=53 ttl=64 time=0.994 ms
64 bytes from 192.168.64.2: icmp_seq=54 ttl=64 time=1.032 ms
64 bytes from 192.168.64.2: icmp_seq=55 ttl=64 time=0.715 ms
64 bytes from 192.168.64.2: icmp_seq=56 ttl=64 time=1.262 ms
64 bytes from 192.168.64.2: icmp_seq=57 ttl=64 time=0.698 ms
64 bytes from 192.168.64.2: icmp_seq=58 ttl=64 time=0.728 ms
64 bytes from 192.168.64.2: icmp_seq=59 ttl=64 time=0.497 ms
64 bytes from 192.168.64.2: icmp_seq=60 ttl=64 time=0.716 ms
64 bytes from 192.168.64.2: icmp_seq=61 ttl=64 time=1.232 ms
^C
--- 192.168.64.2 ping statistics ---
62 packets transmitted, 62 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.448/1.101/4.048/0.455 ms
```

Nmap para escanear puerto 21 (FTP):

```
> nmap -p 21 192.168.64.2
Starting Nmap 7.97 ( https://nmap.org ) at 2025-06-01 20:08 -0300
Nmap scan report for 192.168.64.2
Host is up (0.0017s latency).

PORT      STATE      SERVICE
21/tcp     filtered  ftp

Nmap done: 1 IP address (1 host up) scanned in 0.74 seconds
```

Prueba de acceso a archivo:

```
[agustín_bruno_server@agusbrunouser:~$ whoami
agustín_bruno_server
[agustín_bruno_server@agusbrunouser:~$ id
uid=1000(agustín_bruno_server) gid=1000(agustín_bruno_server) groups=1000(agustín_bruno_server),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd)
[agustín_bruno_server@agusbrunouser:~$ echo "Esto es una prueba" > /var/www/html/mi_prueba.txt
-bash: /var/www/html/mi_prueba.txt: Permission denied
```

Certificación SSL:

```
● ● ● agu — agustin_bruno_server@agusbrunoserver: ~ — ssh agustin_bruno_server@192.168.64.2 — 97x53
GNU nano 6.2          /etc/apache2/sites-available/default-ssl.conf
<IfModule mod_ssl.c>

<VirtualHost _default_:443>
    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

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    #   SSL Engine Switch:
    #   Enable/Disable SSL for this virtual host.
    SSLEngine on

    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    SSLCertificateFile      /etc/apache2/ssl/apache-selfsigned.crt
    SSLCertificateKeyFile  /etc/apache2/ssl/apache-selfsigned.key

    # Server Certificate Chain:
    # Point SSLCertificateChainFile at a file containing the
    # concatenation of PEM encoded CA certificates which form the
    # certificate chain for the server certificate. Alternatively
    # the referenced file can be the same as SSLCertificateFile
    # when the CA certificates are directly appended to the server
    # certificate for convinience.
    #SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt

    # Certificate Authority (CA):
    # Set the CA certificate verification path where to find CA
    # certificates for client authentication or alternatively one
    # huge file containing all of them (file must be PEM encoded)
    # Note: Inside SSLCACertificatePath you need hash symlinks
                                [ Read 135 lines ]
^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location
^X Exit          ^R Read File      ^P Replace       ^U Paste         ^J Justify       ^/ Go To Line
```

```
● ● ● agu — agustin_bruno_server@agusbrunoserver: ~ — ssh agustin_bruno_server@192.168.64.2 — 97x53
GNU nano 6.2          /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
    # The ServerName directive sets the request scheme, hostname and port that
    # 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

    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

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    Redirect permanent / https://192.168.64.2/

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

[ Read 32 lines ]
^G Help          ^O Write Out      ^W Where Is       ^K Cut           ^T Execute        ^C Location
^X Exit          ^R Read File      ^Y Replace        ^U Paste          ^J Justify        ^/ Go To Line
```

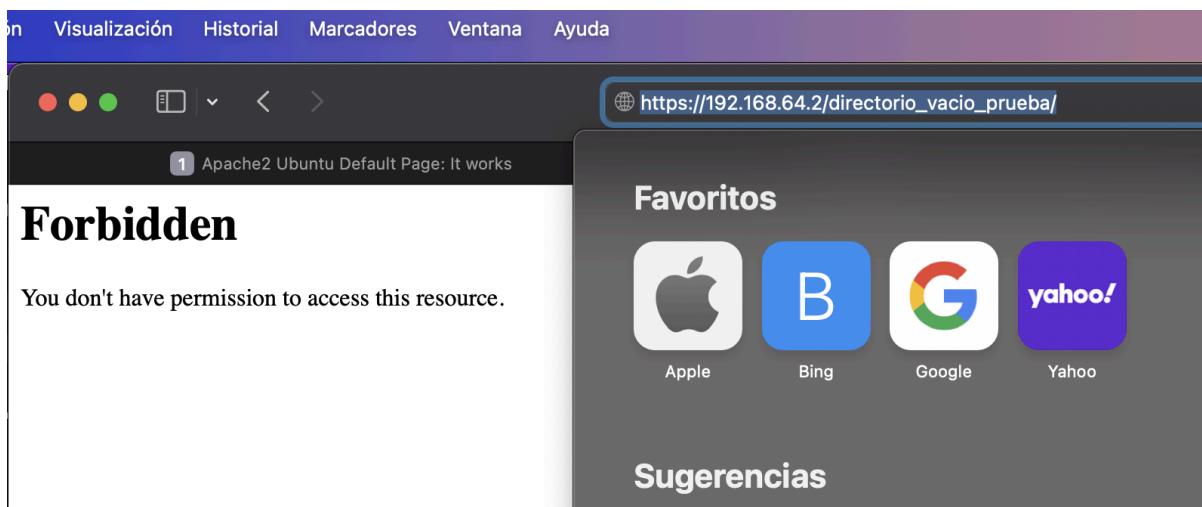
Edición de archivos con nano y prueba de acceso con curl:

```
[agustin_bruno_server@agusbrunoserver:~$ sudo nano /etc/apache2/conf-available/security.conf ]  
[agustin_bruno_server@agusbrunoserver:~$ sudo apache2ctl configtest  
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using fd  
fb:50be:a100:9dc2:a03b:5eff:fe57:b22d. Set the 'ServerName' directive globally to suppress this m  
essage  
Syntax OK  
[agustin_bruno_server@agusbrunoserver:~$ sudo systemctl restart apache2  
[agustin_bruno_server@agusbrunoserver:~$ curl -I -k https://192.168.64.2/  
HTTP/1.1 200 OK  
Date: Mon, 02 Jun 2025 01:14:28 GMT  
Server: Apache  
Last-Modified: Sun, 01 Jun 2025 22:40:33 GMT  
ETag: "29af-6368a55d5e57a"  
Accept-Ranges: bytes  
Content-Length: 10671  
Vary: Accept-Encoding  
Content-Type: text/html
```

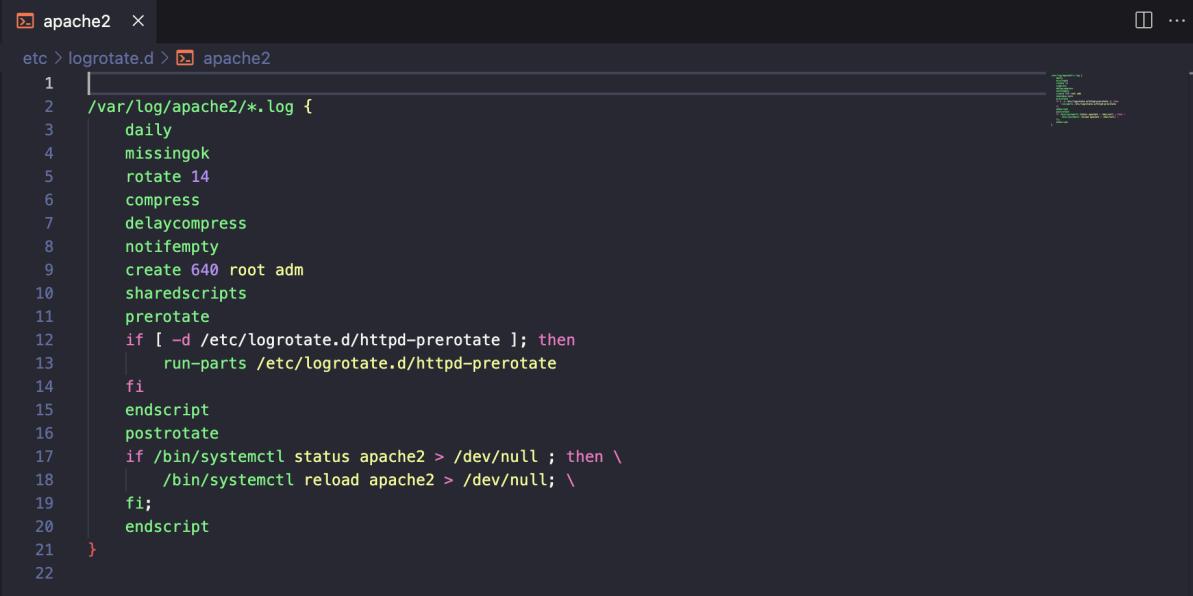
Creación de directorio con mkdir:

```
[agustin_bruno_server@agusbrunoserver:~$ sudo mkdir /var/www/html/directorio_vacio_prueba ]  
[agustin_bruno_server@agusbrunoserver:~$ sudo apt update  
Hit:1 http://ports.ubuntu.com/ubuntu-ports jammy InRelease  
Get:2 http://ports.ubuntu.com/ubuntu-ports jammy-updates InRelease [128 kB]  
Hit:3 http://ports.ubuntu.com/ubuntu-ports jammy-backports InRelease  
Get:4 http://ports.ubuntu.com/ubuntu-ports jammy-security InRelease [129 kB]  
Fetched 257 kB in 3s (99.7 kB/s)  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
All packages are up to date.  
[agustin_bruno_server@agusbrunoserver:~$ sudo apt 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.  
[agustin_bruno_server@agusbrunoserver:~$
```

Página en funcionamiento pero sin permisos para acceder:



Visual Studio para ver archivo de registros de Apache2:

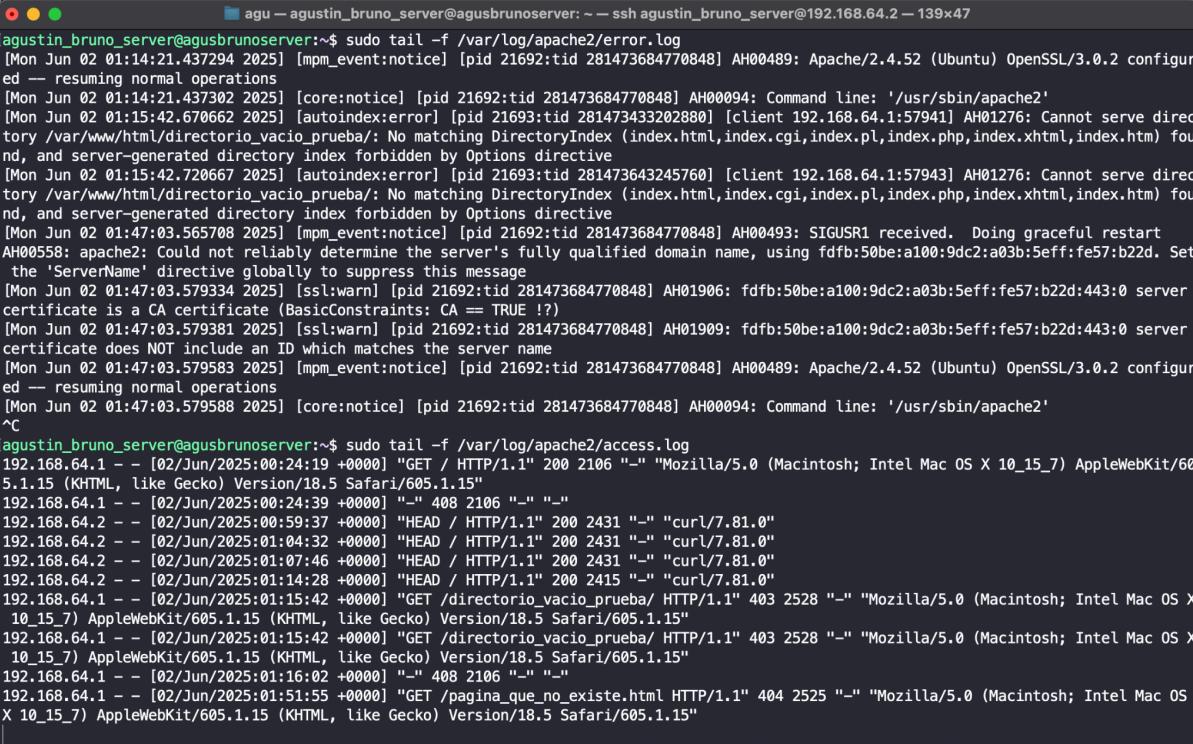


```

apache2
etc > logrotate.d > apache2
1
2  /var/log/apache2/*.log {
3    daily
4    missingok
5    rotate 14
6    compress
7    delaycompress
8    notifempty
9    create 640 root adm
10   sharedscripts
11   prerotate
12   if [ -d /etc/logrotate.d/httpd-prerotate ]; then
13     run-parts /etc/logrotate.d/httpd-prerotate
14   fi
15   endscript
16   postrotate
17   if /bin/systemctl status apache2 > /dev/null ; then \
18     /bin/systemctl reload apache2 > /dev/null; \
19   fi;
20   endscript
21 }
22

```

Monitorear cambios con tail en los registros de errores y accesos:



```

agu -- agustin_bruno_server@agusbrunouser: ~$ sudo tail -f /var/log/apache2/error.log
[Mon Jun 02 01:14:21.437294 2025] [mpm_event:notice] [pid 21692:tid 281473684770848] AH00489: Apache/2.4.52 (Ubuntu) OpenSSL/3.0.2 configured — resuming normal operations
[Mon Jun 02 01:14:21.437302 2025] [core:notice] [pid 21692:tid 281473684770848] AH00094: Command line: '/usr/sbin/apache2'
[Mon Jun 02 01:15:42.670662 2025] [autoindex:error] [pid 21693:tid 281473433202880] [client 192.168.64.1:57941] AH01276: Cannot serve directory /var/www/html/directorio_vacio_prueba/: No matching DirectoryIndex (index.html,index.cgi,index.pl,index.php,index.xhtml,index.htm) found, and server-generated directory index forbidden by Options directive
[Mon Jun 02 01:15:42.720667 2025] [autoindex:error] [pid 21693:tid 281473643245760] [client 192.168.64.1:57943] AH01276: Cannot serve directory /var/www/html/directorio_vacio_prueba/: No matching DirectoryIndex (index.html,index.cgi,index.pl,index.php,index.xhtml,index.htm) found, and server-generated directory index forbidden by Options directive
[Mon Jun 02 01:47:03.565708 2025] [mpm_event:notice] [pid 21692:tid 281473684770848] AH00493: SIGUSR1 received. Doing graceful restart
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using fdfb:50be:a100:9dc2:a03b:5eff:fe57:b22d. Set the 'ServerName' directive globally to suppress this message
[Mon Jun 02 01:47:03.579334 2025] [ssl:warn] [pid 21692:tid 281473684770848] AH01906: fdfb:50be:a100:9dc2:a03b:5eff:fe57:b22d:443:0 server certificate is a CA certificate (BasicConstraints: CA == TRUE !?)
[Mon Jun 02 01:47:03.579381 2025] [ssl:warn] [pid 21692:tid 281473684770848] AH01909: fdfb:50be:a100:9dc2:a03b:5eff:fe57:b22d:443:0 server certificate does NOT include an ID which matches the server name
[Mon Jun 02 01:47:03.579583 2025] [mpm_event:notice] [pid 21692:tid 281473684770848] AH00489: Apache/2.4.52 (Ubuntu) OpenSSL/3.0.2 configured — resuming normal operations
[Mon Jun 02 01:47:03.579588 2025] [core:notice] [pid 21692:tid 281473684770848] AH00094: Command line: '/usr/sbin/apache2'
^C
agu -- agustin_bruno_server@agusbrunouser: ~$ sudo tail -f /var/log/apache2/access.log
192.168.64.1 - - [02/Jun/2025:00:24:19 +0000] "GET / HTTP/1.1" 200 2106 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
192.168.64.1 - - [02/Jun/2025:00:24:39 +0000] "-" 408 2106 "-" "-"
192.168.64.2 - - [02/Jun/2025:00:59:37 +0000] "HEAD / HTTP/1.1" 200 2431 "-" "curl/7.81.0"
192.168.64.2 - - [02/Jun/2025:01:04:32 +0000] "HEAD / HTTP/1.1" 200 2431 "-" "curl/7.81.0"
192.168.64.2 - - [02/Jun/2025:01:07:46 +0000] "HEAD / HTTP/1.1" 200 2431 "-" "curl/7.81.0"
192.168.64.2 - - [02/Jun/2025:01:14:28 +0000] "HEAD / HTTP/1.1" 200 2415 "-" "curl/7.81.0"
192.168.64.1 - - [02/Jun/2025:01:15:42 +0000] "GET /directorio_vacio_prueba/ HTTP/1.1" 403 2528 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
192.168.64.1 - - [02/Jun/2025:01:15:42 +0000] "GET /directorio_vacio_prueba/ HTTP/1.1" 403 2528 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
192.168.64.1 - - [02/Jun/2025:01:16:02 +0000] "-" 408 2106 "-" "-"
192.168.64.1 - - [02/Jun/2025:01:51:55 +0000] "GET /pagina_que_no_existe.html HTTP/1.1" 404 2525 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"

```

```
agustin_bruno_server@agusbrunoserver:~$ sudo tail /var/log/apache2/access.log
192.168.64.1 - - [02/Jun/2025:00:24:39 +0000] "—" 408 2106 "—" "-
192.168.64.2 - - [02/Jun/2025:00:59:37 +0000] "HEAD / HTTP/1.1" 200 2431 "—" "curl/7.81.0"
192.168.64.2 - - [02/Jun/2025:01:04:32 +0000] "HEAD / HTTP/1.1" 200 2431 "—" "curl/7.81.0"
192.168.64.2 - - [02/Jun/2025:01:07:46 +0000] "HEAD / HTTP/1.1" 200 2431 "—" "curl/7.81.0"
192.168.64.2 - - [02/Jun/2025:01:14:28 +0000] "HEAD / HTTP/1.1" 200 2415 "—" "curl/7.81.0"
192.168.64.1 - - [02/Jun/2025:01:15:42 +0000] "GET /directorio_vacio_prueba/ HTTP/1.1" 403 2528 "—" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
192.168.64.1 - - [02/Jun/2025:01:15:42 +0000] "GET /directorio_vacio_prueba/ HTTP/1.1" 403 2528 "—" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
192.168.64.1 - - [02/Jun/2025:01:16:02 +0000] "—" 408 2106 "—" "-
192.168.64.1 - - [02/Jun/2025:01:51:55 +0000] "GET /pagina_que_no_existe.html HTTP/1.1" 404 2525 "—" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/18.5 Safari/605.1.15"
192.168.64.2 - - [02/Jun/2025:01:55:00 +0000] "GET /?search=<script>alert('XSS');</script> HTTP/1.1" 200 13108 "—" "curl/7.81.0"
agustin_bruno_server@agusbrunoserver:~$
```

Script de bash para verificar el estado de hardening (automatización):

```
GNU nano 6.2
#!/bin/bash

# Script para verificar el estado de hardening del servidor web Apache
# Autor: [Tu Nombre]
# Fecha: [Fecha Actual]

# --- Configuración ---
SERVER_IP="192.168.64.2" # Reemplaza con la IP real de tu VM
WEB_ROOT_DIR="/var/www/html"

echo "---- Verificación de Hardening del Servidor Web Apache ----"
echo "Fecha de ejecución: $(date)"
echo "-----"

# --- 1. Estado del Servicio Apache ---
echo -e "\n[1/5] Verificando estado del servicio Apache..."
sudo systemctl status apache2 | grep "Active:"
if [ $? -eq 0 ]; then
    echo " Estado: OK – Apache está activo y corriendo."
else
    echo " Estado: ERROR – Apache no está corriendo. ¡Investigar!"
fi

# --- 2. Estado del Firewall (UFW) ---
echo -e "\n[2/5] Verificando estado del Firewall (UFW)..."
sudo ufw status verbose | grep "Status: active"
if [ $? -eq 0 ]; then
    echo " Estado: OK – UFW está activo."
    echo " Reglas UFW:"
    sudo ufw status | grep -E "80|443|22" # Muestra solo las reglas importantes
else
    echo " Estado: ADVERTENCIA – UFW no está activo. ¡Habilitar!"
fi

# --- 3. Ocultamiento de Banners del Servidor ---
echo -e "\n[3/5] Verificando ocultamiento de banners del servidor..."
SERVER_HEADER=$(curl -I -k https://$SERVER_IP/ 2>/dev/null | grep "Server:")
if [[ "$SERVER_HEADER" == *"Server: Apache"* && "$SERVER_HEADER" != */* ]]; then
    echo " Estado: OK – El banner del servidor está oculto (Server: Apache)."
else
    echo " Estado: ADVERTENCIA – El banner del servidor NO está oculto correctamente."
    echo " Encabezado Server actual: $SERVER_HEADER"
fi
```

Ejecución del script en el servidor:

```
agustin_bruno_server@agusbrunoserver:~$ mkdir ~/scripts
agustin_bruno_server@agusbrunoserver:~$ cd ~/scripts
agustin_bruno_server@agusbrunoserver:~/scripts$ nano check_hardening.sh
agustin_bruno_server@agusbrunoserver:~/scripts$ chmod +x check_hardening.sh
agustin_bruno_server@agusbrunoserver:~/scripts$ sudo ./check_hardening.sh
[sudo] password for agustin_bruno_server:
--- Verificación de Hardening del Servidor Web Apache ---
Fecha de ejecución: Mon Jun 2 02:12:25 UTC 2025

[1/5] Verificando estado del servicio Apache...
  Active: active (running) since Mon 2025-06-02 01:14:21 UTC; 58min ago
  Estado: OK - Apache está activo y corriendo.

[2/5] Verificando estado del Firewall (UFW)...
Status: active
  Estado: OK - UFW está activo.
  Reglas UFW:
22/tcp          ALLOW      Anywhere
80/tcp          ALLOW      Anywhere
443            ALLOW      Anywhere
22/tcp (v6)     ALLOW      Anywhere (v6)
80/tcp (v6)     ALLOW      Anywhere (v6)
443 (v6)       ALLOW      Anywhere (v6)

[3/5] Verificando ocultamiento de banners del servidor...
  Estado: OK - El banner del servidor está oculto (Server: Apache).

[4/5] Verificando deshabilitación de listado de directorios...
  Estado: OK - El listado de directorios está deshabilitado (código 403 Forbidden).

[5/5] Verificando permisos del directorio web (/var/www/html)...
Directorio /var/www/html:
  Permisos: drwxr-xr-x (Esperado: drwxr-xr-x o 755)
  Propietario: root (Esperado: root)
  Archivo /var/www/html/index.html:
    Permisos: -rw-r--r-- (Esperado: -rw-r--r-- o 644)
    Propietario: root (Esperado: root)
  Estado: OK - Permisos del directorio web y archivo index.html son correctos.

--- Verificación de Hardening Completada ---
```