**Informe de Pentesting**

**Reportado por:**

**Germán Alberto Parra Araque**

**4Geeks Academy**

**Colombia**

[**1. Resumen Ejecutivo 3**](#_1j4kpf6zixqf)

[**2. Alcance 3**](#_mmqet83l1s3h)

[**3. Metodología 4**](#_b32y8cnt9n5q)

[**4. Resumen de hallazgos técnicos 5**](#_784giacbi4uk)

[**5. Resultados 6**](#_fag0c8vid22k)

[**5.1. Directory Listing Enabled[Medio] 6**](#_k2ysu751fpve)

[**5.2. Sensitive Information Disclosure via Publicly Accessible Files[Medio] 8**](#_7xto0ppdtmuc)

[**6. Referencias 12**](#_kv5675yg8u10)

[**7. Anexos 12**](#_vvle7sw044bx)

## **Resumen Ejecutivo**

El equipo de Pentesting del Banco Teusaquillo fue contactado tras una irrupción en el servidor web con sistema operativo Debian 12, ocurrida el día 6 de junio. Como resultado del incidente, se vio comprometido el servidor que aloja el sitio WordPress de la organización. Se solicitó la realización de pruebas de penetración con el objetivo de identificar posibles vulnerabilidades generadas por esta intrusión.

Durante la evaluación, se detectó una vulnerabilidad crítica derivada de una configuración insegura en el archivo de configuración de WordPress. Esta mala práctica permite que un atacante potencial acceda a los directorios del servidor, exponiendo información sensible y aumentando el riesgo de movimientos laterales dentro de la red.

Dado que el servidor web cumple una función crítica en la operación de la entidad y proporciona acceso a otros servicios empresariales, se recomienda implementar medidas correctivas de forma inmediata. La mitigación de estas vulnerabilidades es esencial para garantizar la continuidad operativa y proteger la integridad de los sistemas del Banco Teusaquillo.

## **Alcance**

El alcance definido para la prueba pentesting se limita al servidor Debian 12, con la ip 192.168.1.124

## **Metodología**

Para el desarrollo de esta fase del informe de pentesting, se trabajó con dos sistemas principales: el servidor Debian 12 perteneciente al Banco de Teusaquillo (objetivo de las pruebas) y una máquina Kali Linux utilizada por el analista de seguridad (equipo atacante).

Las pruebas se realizaron sobre el servicio HTTP alojado en el puerto 80 del servidor objetivo. Como marco metodológico, se aplicó el estándar **PTES (Penetration Testing Execution Standard)**, el cual proporciona una estructura sistemática para la ejecución de pruebas de penetración, permitiendo un análisis ordenado, repetible y completo.

Este enfoque incluye fases como:

* **Reconocimiento**
* **Enumeración**
* **Análisis de vulnerabilidades**
* **Explotación**
* **Post-explotación**
* **Reporte**

La aplicación de PTES asegura que la evaluación del servidor Debian 12 se realice de manera rigurosa, documentada y alineada con buenas prácticas de la industria de la ciberseguridad.

## **Resumen de hallazgos técnicos**

Para clasificar el nivel de severidad de cada vulnerabilidad identificada durante el análisis, se utilizó el **Sistema de Puntuación de Vulnerabilidades Común (CVSS) versión 3.1**. Este estándar permite asignar una puntuación numérica a cada hallazgo, determinando así su impacto y urgencia de corrección.

A continuación, se presentan las vulnerabilidades detectadas clasificadas según su severidad:

| Severidad | Baja | Media | Alta | Crítica |
| --- | --- | --- | --- | --- |
| Número de vulnerabilidades | 0 | 2 | 0 | 0 |

Los hallazgos catalogados como **"Media"** corresponden a exposiciones de información sensible que, aunque no representan una amenaza inmediata de explotación directa, pueden facilitar ataques más avanzados si no se mitigan adecuadamente.

## **Resultados**

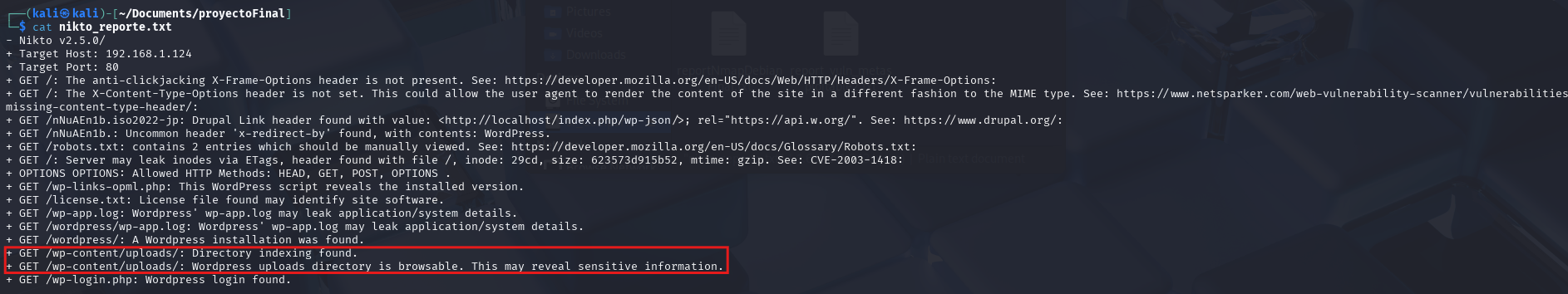
Con el fin de identificar posibles vulnerabilidades en el servidor web, se empleó la herramienta Nikto, la cual forma parte del conjunto de utilidades de seguridad incluidas en Kali Linux. Este escaneo permite detectar configuraciones inseguras, archivos expuestos y posibles fallos en la implementación del servidor HTTP (ver Figura 1).

Figura 1. Uso de Nikto para escanear el servidor http.

## **Directory Listing Enabled[Medio]**

* **CVSS v3.1 Base Score: 5.3 (Medium)**
* **Vector String:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Se identificó que el directorio **192.168.1.124/wp-content/uploads/** es listable, lo que permite la navegación por sus subdirectorios. Esta configuración indebida expone archivos que no deberían estar accesibles al usuario final, representando un riesgo de fuga de información. El acceso se logra ingresando directamente la URL con la ruta completa desde un navegador o herramienta de análisis.

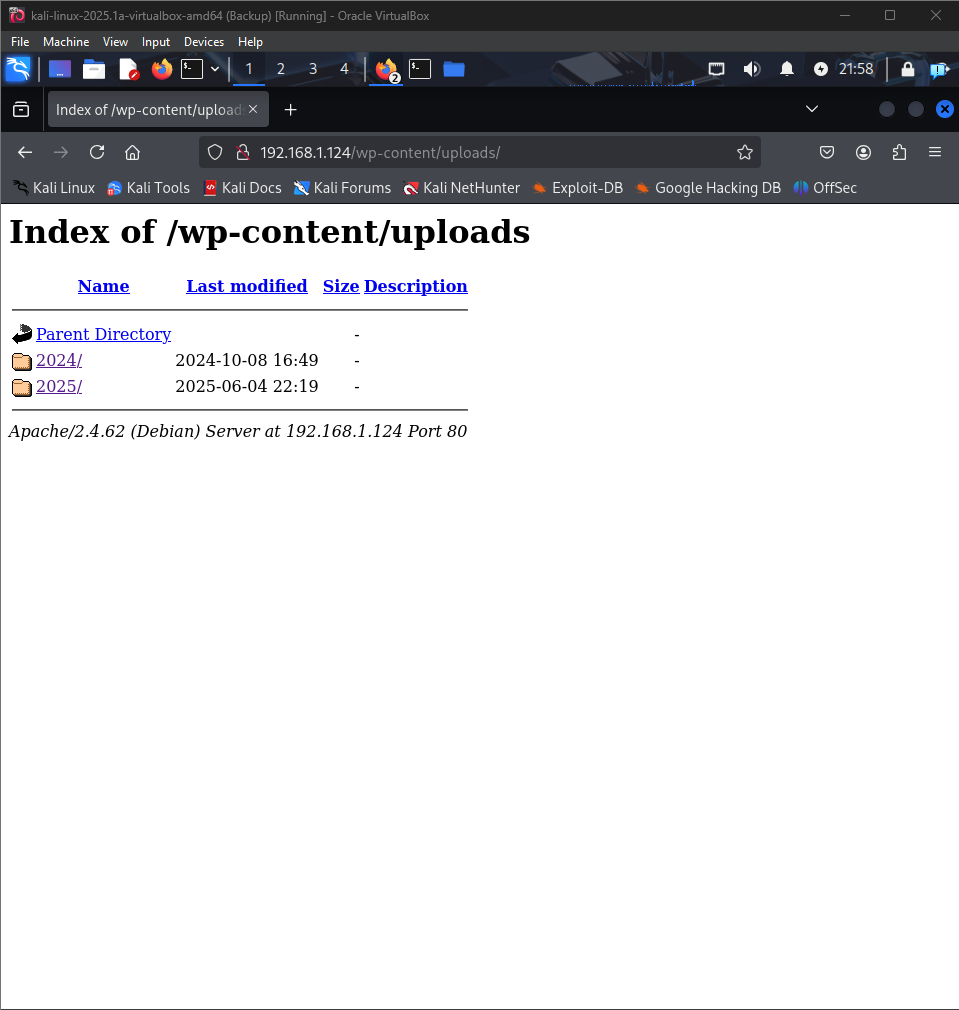


Figura 2. Demostración de acceso a subdirectorios a sitio web.

* **Remediación recomendada**

Para evitar que el servidor web Apache muestre el contenido de directorios sin un archivo índice (como index.html o index.php), es necesario modificar la configuración global del servidor Debian 12.

Se debe configurar el archivo **apache2.conf**, ubicado en el directorio **/etc/apache2** y se bloquea que se muestran directorios sin archivo índice, como se ve en la imagen.

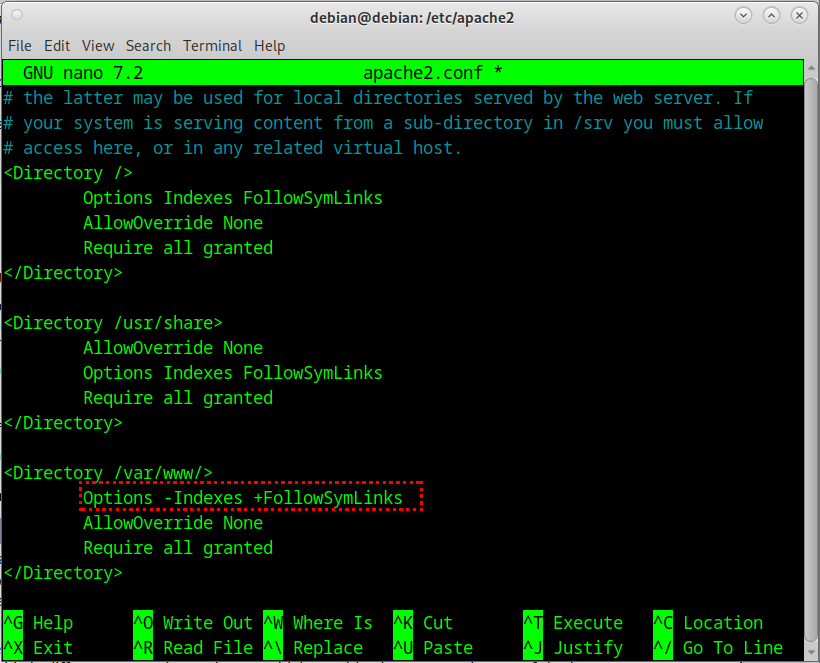


Figura 3. Configuración **apache2.conf**.

Se reinicia el servicio apache como se muestra a continuación:



Figura 4. Reinicio servicio apache2.

Finalmente se vuelve a acceder desde el navegador de la máquina Kali y se puede observar que el acceso está prohibido.

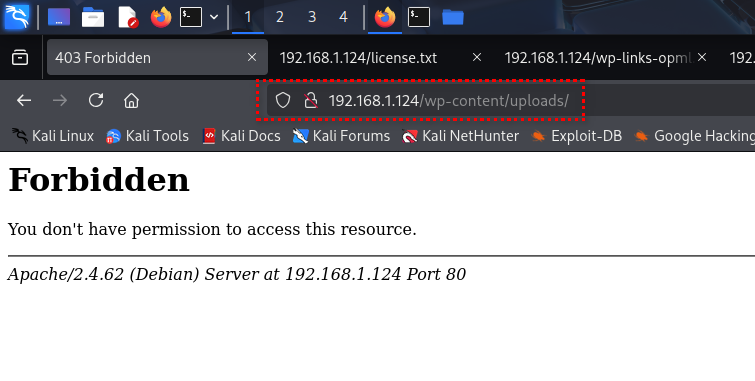


Figura 5. Acceso prohibido al directorio y subdirectorios.

## **Sensitive Information Disclosure via Publicly Accessible Files[Medio]**

* **Base Score: 5.3 (Medium)**
* **Vector:** CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

Se identificaron archivos sensibles, expuestos desde la web **192.168.1.124/**, lo que permite la exposición de datos del servidor. Esta configuración indebida expone archivos que no deberían estar accesibles al usuario final, representando un riesgo de fuga de información. El acceso se logra ingresando directamente a las siguientes URL´S desde un navegador:

* **192.168.1.124/license.txt**

Se muestra la licencia de software de wordpress.

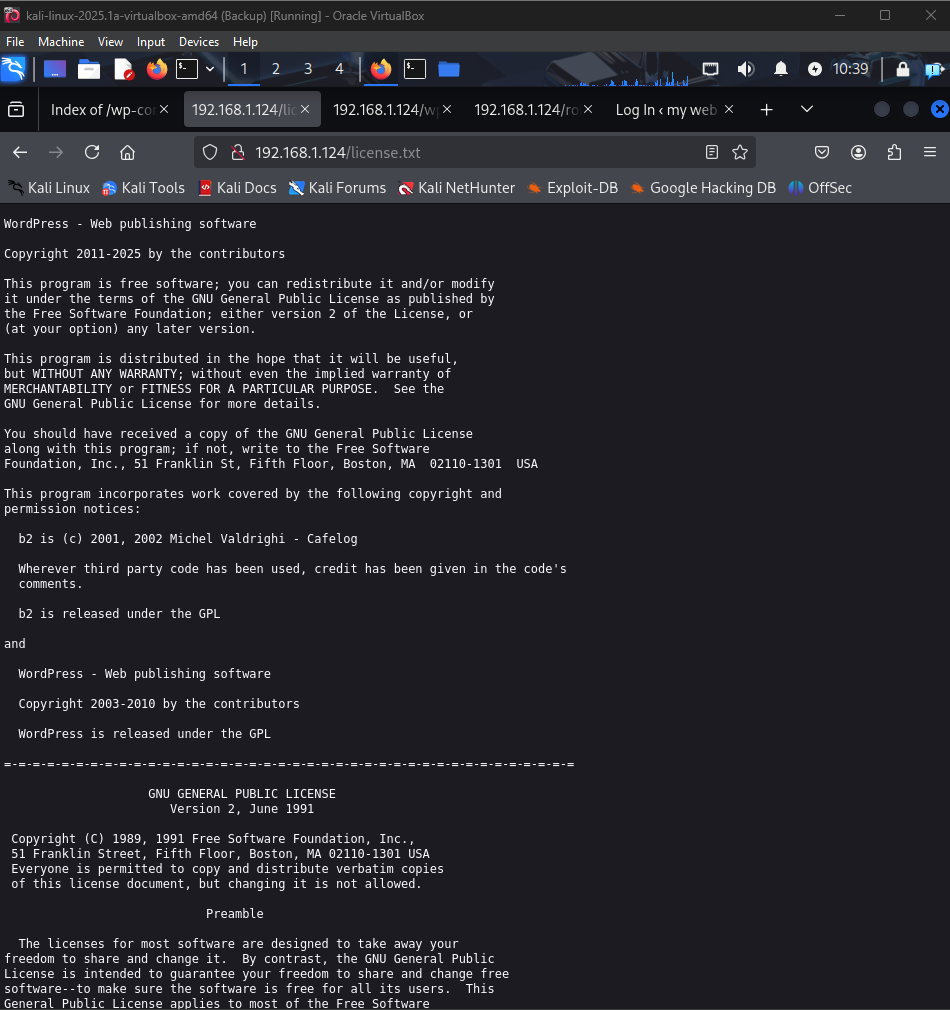
****

Figura 5. Archivo de licencia de Wordpress **license.txt**.

* **192.168.1.124/wp-links-opml.php**

Se otorga información sobre la versión del servidor wordpress.

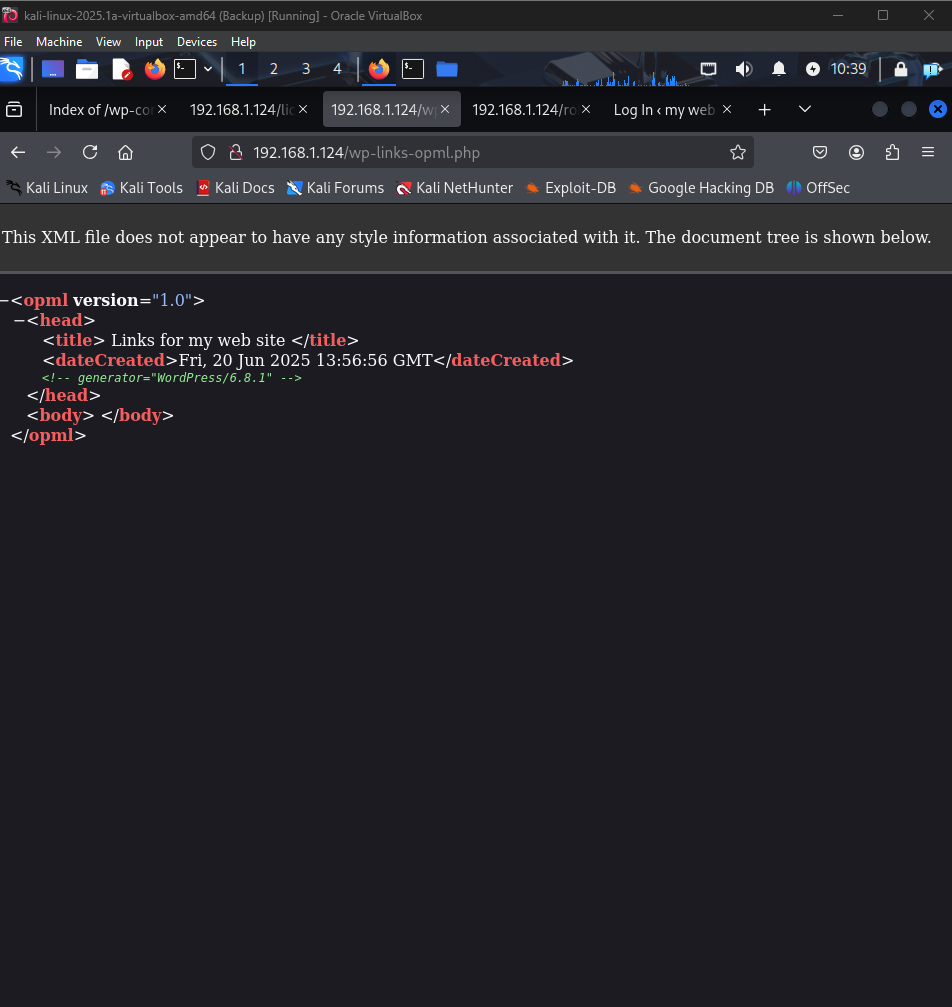
****

Figura 6. Archivo de **wp-links-opml.php**.

* **192.168.1.124/robots.txt**

Revela rutas sensibles que pueden ser usadas por atacantes.

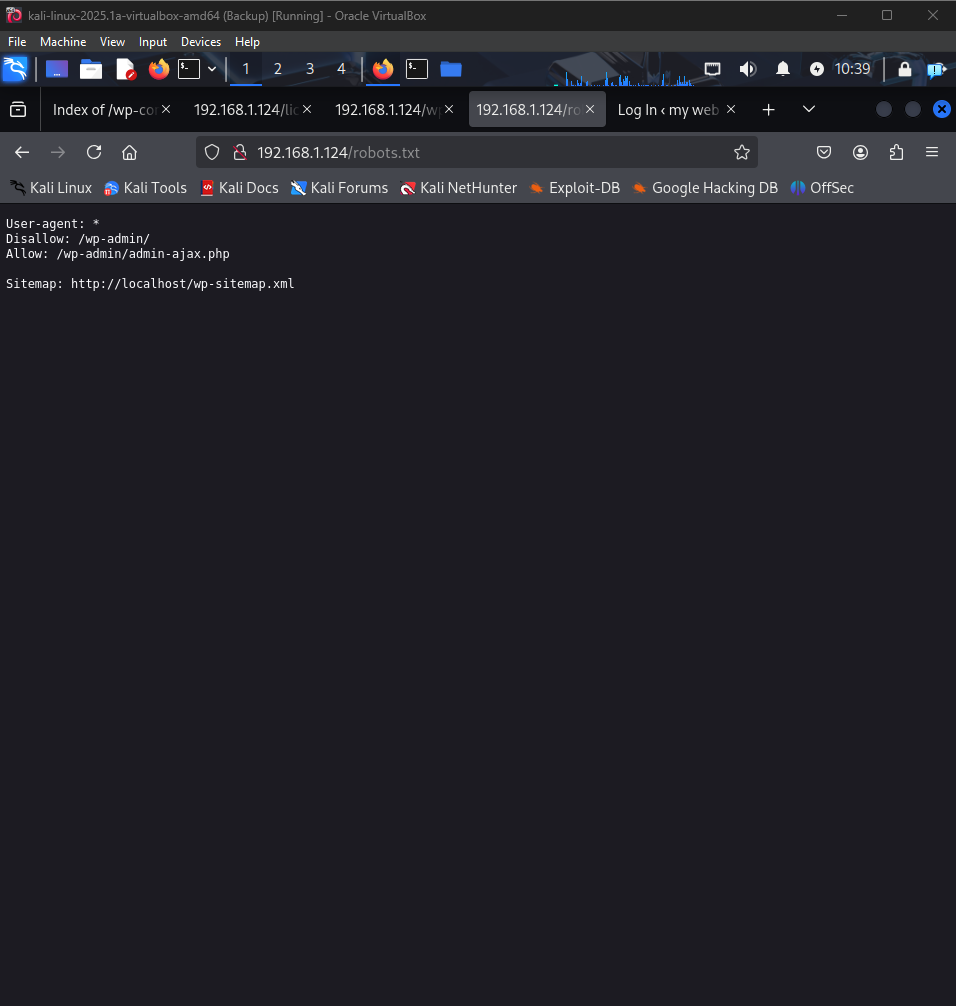
****

Figura 7. Archivo de **robots.txt**.

* **Remediación recomendada**

Para evitar que el servidor web apache permita el acceso a archivos sensibles (como **license.txt**, **wp-link-opml.php** y **robots.txt** ) desde un navegador, es necesario añadir una configuración dentro de **apache2.conf.** A continuación se muestra una configuración para bloquear el acceso de dichos archivos, este código se puede colocar en la parte final del archivo de configuración:

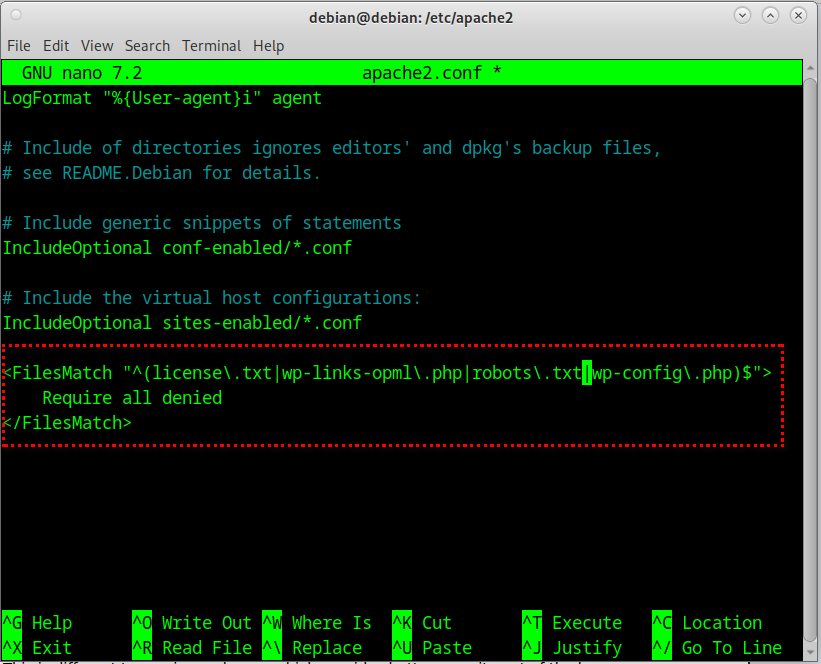


Figura 8. Archivo **apache2.conf**.

Volviendo a acceder desde el navegador, se muestra que se bloqueo el acceso a los archivos configurados en apache2.conf:

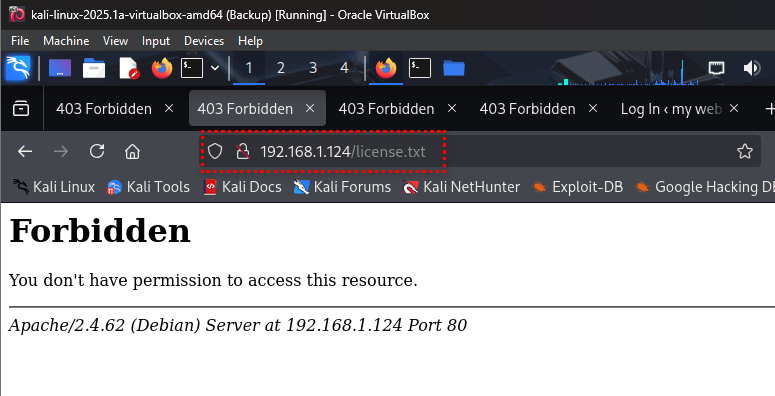


Figura 9. Acceso bloqueado **license.txt**.

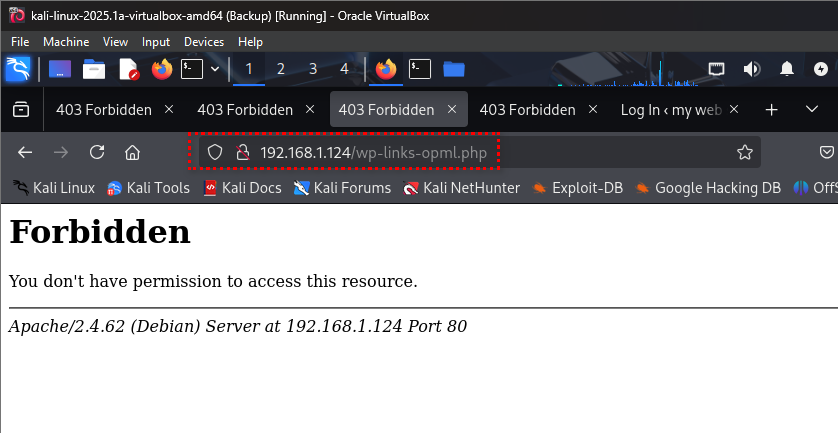


Figura 10. Acceso bloqueado **wp-links-opml.php**.

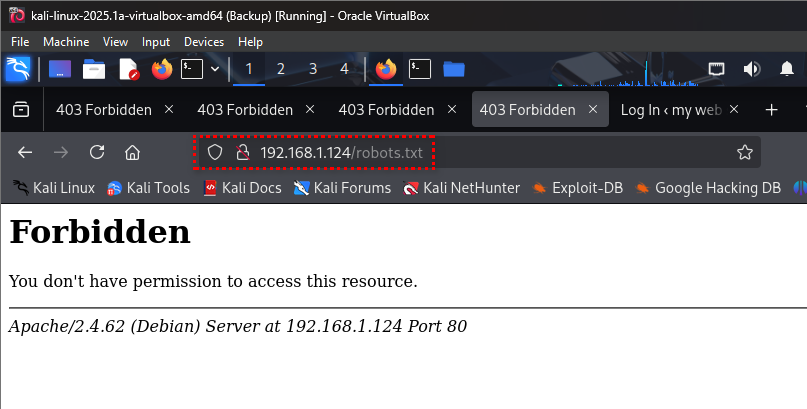


Figura 11. Acceso bloqueado **wp-links-opml.php**.

## 

## **Referencias**

<https://cwe.mitre.org/data/definitions/200.html>

<https://cwe.mitre.org/data/definitions/548.html>

## **Anexos**

Reporte Nikto

- Nikto v2.5.0/

+ Target Host: 192.168.1.124

+ Target Port: 80

+ GET /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options:

+ GET /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/:

+ GET /nNuAEn1b.iso2022-jp: Drupal Link header found with value: <http://localhost/index.php/wp-json/>; rel="https://api.w.org/". See: https://www.drupal.org/:

+ GET /nNuAEn1b.: Uncommon header 'x-redirect-by' found, with contents: WordPress.

+ GET /robots.txt: contains 2 entries which should be manually viewed. See: https://developer.mozilla.org/en-US/docs/Glossary/Robots.txt:

+ GET /: Server may leak inodes via ETags, header found with file /, inode: 29cd, size: 623573d915b52, mtime: gzip. See: CVE-2003-1418:

+ OPTIONS OPTIONS: Allowed HTTP Methods: HEAD, GET, POST, OPTIONS .

+ GET /wp-links-opml.php: This WordPress script reveals the installed version.

+ GET /license.txt: License file found may identify site software.

+ GET /wp-app.log: Wordpress' wp-app.log may leak application/system details.

+ GET /wordpress/wp-app.log: Wordpress' wp-app.log may leak application/system details.

+ GET /wordpress/: A Wordpress installation was found.

+ GET /wp-content/uploads/: Directory indexing found.

+ GET /wp-content/uploads/: Wordpress uploads directory is browsable. This may reveal sensitive information.

+ GET /wp-login.php: Wordpress login found.