PENETRATION TESTING

AUDITORIA

Elaborado por: Victor Lopez

Pablo Torres

4GEEKS
ACADEMY

INDICE

1. Introducción

El presente informe detalla los hallazgos y resultados de una evaluación de seguridad realizada sobre un servidor Debian de la empresa 4Geeks Academy, identificado con la dirección IP 192.168.68.***.

En el panorama actual de amenazas cibernéticas, la identificación proactiva de vulnerabilidades es un componente esencial para fortalecer la resiliencia de los sistemas informáticos. Las pruebas de penetración, también conocidas como pentesting, simulan los métodos y técnicas que un atacante real podría emplear para encontrar y explotar debilidades en la seguridad de un sistema. Al realizar este tipo de evaluación, las organizaciones pueden obtener una perspectiva práctica y profunda sobre su postura de seguridad, validar la efectividad de sus controles existentes e identificar áreas que requieren mejoras urgentes antes de que sean explotadas por actores maliciosos. Este informe detalla el proceso, los hallazgos y las recomendaciones derivadas de este ejercicio de simulación de ataque controlado.

2. Objetivo

El objetivo principal de esta prueba de penetración fue identificar y demostrar la existencia de vulnerabilidades de seguridad presentes en el servidor Debian con IP 192.168.68.*** a través de la ejecución de pruebas activas.

Específicamente, los objetivos incluyeron:

- Identificar los servicios de red y aplicaciones que se encuentran accesibles externamente en el servidor.
- Detectar configuraciones erróneas y debilidades de seguridad inherentes en los servicios identificados (ej. versiones de software vulnerables, configuraciones por defecto inseguras, fallas en la gestión de credenciales).
- Demostrar la explotabilidad de al menos una vulnerabilidad diferente a aquellas relacionadas con el acceso inicial al sistema operativo mediante credenciales débiles.
- Evaluar el posible impacto que la explotación de las vulnerabilidades encontradas podría tener en la confidencialidad, integridad y disponibilidad de los datos y funcionalidades del servidor.
- Proporcionar recomendaciones claras y prácticas para corregir las vulnerabilidades descubiertas y mejorar la postura de seguridad general del servidor.

De esta manera buscamos ofrecer una visión realista de las debilidades de seguridad desde la perspectiva de un atacante, permitiendo priorizar los esfuerzos de mitigación de manera efectiva.

3. Alcance

El alcance de esta prueba de penetración se definió estrictamente para incluir únicamente las pruebas activas realizadas sobre el servidor con la dirección IP 192.168.68.*** . Las pruebas se llevaron a cabo simulando un escenario desde una red interna, evaluando los servicios y aplicaciones accesibles a través de la red.

Los servicios dentro del alcance principal de la evaluación, identificados durante la fase de reconocimiento, incluyeron:

- Servicio FTP (Puerto 21/TCP)
- Servicio SSH (Puerto 22/TCP)
- Servicio Web (HTTP en Puerto 80/TCP), incluyendo la aplicación WordPress alojada.

Las pruebas se centraron en la identificación de vulnerabilidades a nivel de red y aplicación web, incluyendo escaneo de puertos, enumeración de directorios, detección de versiones, escaneo de vulnerabilidades conocidas, y pruebas de fuerza bruta sobre autenticación.

Cualquier otro sistema, red, o servicio no especificado por la dirección IP mencionada, estuvo explícitamente fuera del alcance de esta prueba. El análisis se ha realizado en un entorno de laboratorio controlado con el conocimiento y autorización previos, y todas las actividades se limitaron al objetivo designado.

4. Metodología

La metodología empleada en este pentesting sigue una aproximación estructurada basada en técnicas comúnmente aceptadas en el ámbito de la seguridad ofensiva.

Las pruebas se realizaron desde una red interna simulada, empleando una metodología que incluyó:

- Escaneo de puertos y servicios para mapear la superficie de ataque.
- Enumeración de directorios y archivos en servicios web.
- Escaneo automatizado de vulnerabilidades web conocidas y configuraciones inseguras.
- Pruebas dirigidas de fuerza bruta sobre servicios de autenticación expuestos.
- Análisis manual y exploración de los hallazgos identificados.

El resto del informe documentará los hallazgos detallados de esta evaluación, describiendo las vulnerabilidades identificadas, la demostración de su existencia y potencial impacto, la evaluación del riesgo asociado, y recomendaciones específicas para mitigar dichas vulnerabilidades y fortalecer la seguridad general del servidor frente a futuras amenazas.

La evaluación de seguridad sobre el servidor objetivo se llevó a cabo siguiendo una metodología estructurada, basada en estándares reconocidos en el campo de las pruebas de penetración.

El proceso se dividió en las siguientes fases clave, ejecutadas de manera iterativa y documentada:

- **4.1. Fase de Reconocimiento y Descubrimiento del Objetivo:**

Esta etapa inicial tuvo como objetivo mapear la superficie de ataque del servidor objetivo ([Insertar IP del servidor]). Se emplearon técnicas activas para determinar si el host estaba activo e identificar los servicios de red accesibles desde la red interna simulada.

- **4.2. Fase de Escaneo y Enumeración:**

Una vez identificados los puertos abiertos, esta fase se centró en obtener información detallada sobre los servicios que se ejecutaban en ellos. El objetivo era identificar las versiones exactas del software y enumerar componentes clave que pudieran ser vectores de ataque.

  - Se realizó detección de versiones y sistemas operativos
  - Para el servicio web (HTTP/80), se llevó a cabo una enumeración exhaustiva de directorios y archivos comunes, para identificar rutas accesibles, estructuras de aplicación (como WordPress) y archivos potencialmente sensibles.
  - Se realizo un escáner de vulnerabilidades web, para buscar miles de configuraciones inseguras, archivos comunes, versiones de software y otros problemas conocidos en el servidor HTTP.
  - Adicionalmente, aplicamos un escáner específico para WordPress, con el fin de detectar la versión exacta, enumerar plugins, temas instalados, y buscar vulnerabilidades conocidas asociadas a estos componentes y a la versión principal de la plataforma.

- **4.3. Fase de Análisis de Vulnerabilidades:**

En esta etapa, se analizaron los resultados brutos de los escaneos y la enumeración para identificar vulnerabilidades de seguridad potenciales. Esto incluyó:

  - Correlacionar las versiones de software detectadas con bases de datos públicas de vulnerabilidades (CVE, NVD, Exploit-DB, base de datos de WPScan) para identificar exploits conocidos.
  - Analizar configuraciones reportadas como inseguras por los escáneres (ej. listabilidad de directorios, encabezados de seguridad faltantes, fugas de información).
  - Evaluar la debilidad de los mecanismos de autenticación para servicios expuestos.
  - Investigación manual de hallazgos inusuales (ej. el directorio `/0/`, el archivo `index.html`).

- **4.4. Fase de Explotación (Intentos Controlados):**

En esta fase, se llevaron a cabo intentos controlados para explotar las vulnerabilidades identificadas con el fin de demostrar su impacto y el riesgo asociado. La metodología se centró en la explotación de una vulnerabilidad diferente a la relacionada con el acceso inicial al sistema operativo mediante credenciales débiles (abordado en el contexto de la Fase 1 del proyecto general).

- Se realizó un intento de fuerza bruta sobre servicios de autenticación (SSH) utilizando Hydra para evaluar la robustez de las credenciales de acceso, confirmando la debilidad de la contraseña root identificada en un contexto anterior.
- Se verificó manualmente la vulnerabilidad de listabilidad de directorios en el servidor web (WordPress) para demostrar la divulgación de información y facilitar la enumeración de componentes de la aplicación.
- Se planificó o intentó explotar vulnerabilidades específicas identificadas en la aplicación WordPress o sus componentes (plugins/temas) basándose en la información obtenida en las fases de escaneo y análisis (ej. utilizando exploits públicos si se identificaron versiones vulnerables con exploits disponibles). Todos los intentos de explotación se realizaron de manera controlada para evitar interrupciones no deseadas del servicio.

Todas las actividades se realizaron de conformidad con los permisos concedidos y se mantuvieron dentro del alcance estrictamente definido. Se documentaron todos los comandos ejecutados, los resultados obtenidos, y las evidencias de las vulnerabilidades identificadas.

5. Resumen Ejecutivo

El presente informe documenta los resultados de una prueba de penetración realizada sobre el servidor Debian con dirección IP 192.168.68.***. El objetivo principal de esta evaluación fue identificar vulnerabilidades de seguridad activamente presentes en el sistema mediante técnicas de escaneo y análisis. La evaluación reveló la existencia de vulnerabilidades críticas y de alto riesgo que exponen el servidor a compromisos significativos. Entre los hallazgos más relevantes se destacan:

5.1 Tabla Resumen de Vulnerabilidades.

Servicio	Puerto	CVE	Vulnerabilidad	Criticidad	Recomendación Principal
FTP (vsftpd)	21	CVE-2021-30047	Denegación de servicio por comandos malformados	Media	Actualizar vsftpd o aplicar mitigaciones con firewall/IDS
FTP (vsftpd)	21	CVE-2021-3618	Configuración insegura que permite explotación remota	Media	Revisar configuraciones y restringir accesos
FTP (vsftpd)	21	-	Acceso anónimo habilitado (ftp-anon)	Alta	Desactivar acceso anónimo y controlar permisos de escritura/lectura
FTP (vsftpd)	21	-	Falta de cifrado: conexión en texto plano	Alta	Migrar a SFTP o establecer túneles seguros (VPN/SSH)
SSH (OpenSS)	22	CVE-2023-38408	RCE mediante Forwarding X11	Crítica	Deshabilitar Forwarding X11 y actualizar SSH
SSH (OpenSS)	22	CVE-2023-28531	Evasión de filtrado SSH	Alta	Configurar reglas de firewall y revisar autenticación
SSH (OpenSS)	22	CVE-2024-6387	Condición de carrera en demonio SSH	Alta	Aplicar parches de seguridad

SSH (OpenSS)	22	CVE-2025-26465	Fuga de información vía mensajes SSH	Media	Deshabilitar mensajes de error detallados en SSH
SSH (OpenSS)	22	CVE-2023-51385	Protocolo obsoleto – riesgo de downgrade	Media	Usar algoritmos de cifrado modernos
SSH (OpenSS)	22	CVE-2023-48795	Terrapin Attack – manipulación de tráfico SSH	Media	Actualizar OpenSSH y evitar algoritmos vulnerables
SSH (OpenSS)	22	CVE-2023-51384	Riesgo de fallback a configuraciones débiles	Baja	Fortalecer configuración criptográfica del servidor
Apache/ WordPres	80	CVE-2003-1418	Fuga de inodos a través de ETags	Baja	Deshabilitar ETags en la configuración del servidor
Apache/ HTTP	80	-	Falta de cabecera X-Frame-Options (Clickjacking)	Media	Añadir cabecera X-Frame-Options en Apache o Nginx
Apache/ HTTP	80	-	Falta de cabecera X-Content-Type-Options (MIME sniffing)	Media	Añadir cabecera X-Content-Type-Options: nosniff
WordPres s Core	80	-	Enumeración del usuario admin	Baja	Ocultar autores y limitar errores de login
WordPres s Core	80	-	Archivo xmlrpc.php expuesto	Media	Deshabilitar o restringir xmlrpc.php
WordPres s Core	80	-	Directorios listables (por DIRB)	Media	Deshabilitar Indexes en Apache/Nginx
WP Plugin	80	CVE-2021-24183	wp-big-video-background – Subida de archivos	Alta	Eliminar o actualizar plugin vulnerable
WP Plugin	80	-	wp-file-manager – Subida arbitraria de archivos (RCE)	Crítica	Eliminar inmediatamente o parchear el plugin
WP Plugin	80	-	wp-vcd – RCE autenticado	Crítica	Eliminar el plugin y revisar credenciales comprometidas
WP Plugin	80	-	Full Path Disclosure (FPD)	Baja	Revisar y eliminar mensajes de error detallados

En conclusión, la postura de seguridad actual del servidor Debian, presenta debilidades significativas, particularmente en la configuración del servicio web y en la gestión de credenciales privilegiadas. La explotación de la listabilidad de directorios o el acceso con credenciales débiles permitiría a un atacante comprometer la integridad y confidencialidad de los datos alojados y tomar control del sistema. Se requieren acciones correctivas inmediatas para mitigar estos riesgos.

6. Vulnerabilidades Identificadas

- **Nombre de la Vulnerabilidad:** Configuración Insegura – Acceso Anónimo en FTP
- **Servicio Afectado:** FTP (vsftpd 3.0.3)
- **Puerto:** 21/TCP
- **CVE Asociado:** No aplica directamente a esta versión, pero corresponde a una mala práctica de configuración (CWE-200: Exposure of Sensitive Information to an Unauthorized Actor).
- **Descripción:**
  El servidor FTP permite acceso anónimo sin autenticación (código FTP 230). Esto podría permitir a cualquier usuario listar y, en ciertos casos, descargar o subir archivos al sistema, dependiendo de la configuración de permisos. El acceso anónimo representa un riesgo elevado, ya que podría facilitar la filtración o modificación de archivos críticos.
- **Criticidad:** Alta
- **Impacto Potencial:** Pérdida de confidencialidad e integridad, acceso no autorizado a archivos sensibles.
- **Recomendación de Mitigación:**
  - Editar el archivo `/etc/vsftpd.conf` y establecer:

    ```
    ini
    anonymous_enable=NO
    ```

  - Reiniciar el servicio FTP:

    ```
    nginx
    systemctl restart vsftpd
    ```

  - Verificar los permisos de los directorios públicos para evitar exposiciones innecesarias.

- **Nombre de la Vulnerabilidad:** Falta de Cifrado en Conexiones FTP
- **Servicio Afectado:** FTP (vsftpd 3.0.3)
- **Puerto:** 21/TCP
- **CVE Asociado:** No aplica un CVE específico, pero se relaciona con debilidades de transmisión insegura de datos (CWE-319: Cleartext Transmission of Sensitive Information).
- **Descripción:**
  Se evidenció, a través del script `ftp-syst`, que tanto las conexiones de control como las de datos del servicio FTP son realizadas en texto plano. Específicamente, la salida indica:
  - *"Control connection is plain text"*
  - *"Data connections will be plain text"*
    Esto significa que las credenciales y archivos pueden ser interceptados fácilmente mediante técnicas como sniffing en redes comprometidas o no segmentadas, lo que representa un riesgo grave de confidencialidad.
- **Criticidad:** Alta
- **Impacto Potencial:** Exposición de credenciales, interceptación de datos, escalada de privilegios por actores maliciosos.
- **Recomendación de Mitigación:**
  - Implementar FTPS (FTP sobre TLS) para cifrar las comunicaciones.
  - Alternativamente, deshabilitar el servicio FTP y utilizar SFTP (SSH File Transfer Protocol), que cifra por defecto las sesiones.
  - Restringir el uso del servicio FTP a redes internas controladas, si no puede ser reemplazado inmediatamente.

- **Nombre de la Vulnerabilidad:** Configuración Insegura por Defecto en vsftpd
- **Servicio Afectado:** vsftpd (Very Secure FTP Daemon)**Versión Potencialmente Afectada:** Configuraciones inseguras en múltiples versiones, incluyendo 3.0.3 si no se ajusta adecuadamente.
- **Puerto:** 21/TCP
- **CVE Asociado:** CVE-2021-3618
- **Descripción:**
  Esta vulnerabilidad se refiere al uso de configuraciones predeterminadas inseguras en el servicio FTP, específicamente cuando:
    - Se permite acceso **anónimo sin restricciones**.
    - Se permite **conexión sin cifrado** (lo que expone datos en texto plano).
    - No se aplican medidas adecuadas de aislamiento de usuarios ni limitación de comandos.

  Aunque **vsftpd por sí mismo no presenta una vulnerabilidad de ejecución remota** en esta versión, su uso con configuraciones permisivas como el **login anónimo** (como el observado en este análisis), y sin cifrado, puede facilitar **ataques secundarios como sniffing de credenciales, subida de archivos maliciosos** o el abuso del servidor para distribución de malware.
- **Criticidad:** Media
- **Impacto Potencial:**
    - Pérdida de confidencialidad por transmisión en texto plano.
    - Abuso del servicio para almacenamiento o distribución de archivos maliciosos.
    - Riesgo de escalada si se combina con otras vulnerabilidades o servicios mal configurados.
- **Recomendación de Mitigación:**
    - Deshabilitar el acceso FTP anónimo si no es estrictamente necesario.
    - Configurar cifrado TLS en el servidor (FTPS).
    - Utilizar `chroot_local_user=YES` para aislar a los usuarios en sus directorios.
    - Monitorear y registrar toda la actividad FTP.
    - Considerar reemplazar FTP por SFTP como alternativa más segura.

- **Nombre de la Vulnerabilidad:** Denegación de Servicio por Manejo Incorrecto de Comandos Malformados en vsftpd
- **Servicio Afectado:** vsftpd (Very Secure FTP Daemon) 3.0.3
- **Puerto:** 21/TCP
- **CVE Asociado:** CVE-2021-30047
- **Descripción:**
  Se descubrió que **vsftpd 3.0.3** es vulnerable a una **condición de denegación de servicio (DoS)**. Un atacante remoto no autenticado puede enviar **ciertos comandos FTP malformados** que provocan un **comportamiento inesperado o cuelgue del servicio**. En algunos entornos, esta falla puede causar la interrupción del servicio FTP para todos los usuarios legítimos.

  Esta vulnerabilidad no permite ejecución de código ni escalamiento de privilegios, pero sí puede ser utilizada para realizar **ataques de interrupción temporal de servicio**, por ejemplo, mediante un simple script automatizado.
- **Criticidad:** Media
- **Impacto Potencial:**
  - Interrupción del servicio FTP (Denegación de Servicio).
  - Pérdida de disponibilidad para usuarios legítimos.
  - Riesgo elevado si el servicio FTP es crítico para operaciones internas o automatizaciones.
- **Condiciones de Explotación:**
  - Requiere conexión TCP al puerto 21.
  - Puede ser explotado sin autenticación.
  - Solo afecta si no hay mitigación activa (como control de conexiones o IDS).
- **Recomendación de Mitigación:**
  - Aplicar un parche o actualizar vsftpd si se lanza una versión corregida.
  - Implementar un firewall que limite acceso FTP a direcciones IP conocidas.
  - Monitorizar actividad anómala en el servicio y registrar intentos de comandos malformados.
  - Considerar migrar a protocolos más seguros como SFTP (basado en SSH).

- **Nombre de la Vulnerabilidad:** Autenticación Débil – Contraseña Débil para Usuario Root
- **Servicio Afectado:** SSH (OpenSSH 9.2p1)
- **Puerto:** 22/TCP
- **CVE Asociado:** No aplica un CVE concreto, pero está relacionado con la debilidad en mecanismos de autenticación (CWE-521: Weak Password Requirements).
- **Descripción:**
  Se identificó que el servicio SSH permite autenticación con la cuenta root utilizando contraseñas débiles. Se logró acceder mediante un ataque de fuerza bruta empleando la herramienta Hydra, descubriendo la contraseña 123456, lo cual representa una amenaza crítica para la seguridad del sistema, dado que permite el acceso total como superusuario.
- **Criticidad:** **Crítica**
- **Impacto Potencial:** Compromiso total del sistema, acceso root no autorizado.
- **Recomendación de Mitigación:**
  - Establecer políticas de contraseñas robustas.
  - Deshabilitar el inicio de sesión SSH para el usuario root:
    - nginx
    - PermitRootLogin no
    - en /etc/ssh/sshd_config.
  - Usar autenticación mediante claves públicas.
  - Implementar herramientas como fail2ban o sshguard para bloquear intentos repetidos.
  - Monitorizar constantemente los logs de autenticación y actividad SSH.

- **Nombre de la Vulnerabilidad**: Ejecución Remota de Código por abuso de ssh-agent con Forwarding habilitado en OpenSSH
- **Servicio Afectado**: OpenSSH 8.9 a 9.3p1
- **Puerto**: 22/TCP
- **CVE Asociado**: CVE-2023-38408
- **Descripción:**

  Se ha identificado que las versiones de OpenSSH desde la 8.9 hasta la 9.3p1 presentan una vulnerabilidad crítica cuando se utiliza el agente SSH (ssh-agent) con **Forwarding** habilitado. Esta vulnerabilidad permite a un servidor SSH malicioso explotar un fallo en el manejo de bibliotecas compartidas, y potencialmente ejecutar código arbitrario en el sistema del cliente que haya habilitado el reenvío del agente.

  Aunque esta vulnerabilidad afecta principalmente al cliente, la detección de una versión vulnerable del servicio en el servidor evaluado representa un riesgo potencial, especialmente en entornos donde se practica la administración remota con ForwardAgent activado.

- **Criticidad:** Alta
- **Impacto Potencial:**
  - Ejecución remota de código en sistemas cliente mediante la manipulación del ssh-agent.
  - Compromiso del entorno del administrador o pentester que se conecte a la máquina vulnerable con Forwarding activo.
  - Uso del servidor comprometido como pivote para ataques laterales hacia otros activos conectados mediante SSH.
- **Condiciones de Explotación:**
  - Requiere que el atacante tenga control sobre el servidor SSH o haya comprometido el mismo.
  - El cliente que se conecta debe tener habilitado ForwardAgent yes.
  - No se requiere autenticación adicional si se cumple lo anterior.
- **Recomendación de Mitigación:**
  - **Actualizar OpenSSH a la versión 9.3p2 o superior**, donde esta vulnerabilidad ha sido corregida oficialmente.
  - **Deshabilitar el uso de Agent Forwarding** si no es estrictamente necesario:
    - En el servidor: verificar que no esté permitido en sshd_config.
    - En el cliente: usar ForwardAgent no en ~/.ssh/config.
  - Monitorizar conexiones SSH y limitar el uso de ssh-agent en saltos entre entornos sensibles.

- **Nombre de la Vulnerabilidad**: Fuga de Información a través de Respuestas Impropiamente Gestionadas en OpenSSH
- **Servicio Afectado**: OpenSSH 9.1p1 y anteriores
- **Puerto**: 22/TCP
- **CVE Asociado**: CVE-2023-28531
- **Descripción:**

  Se ha descubierto una vulnerabilidad en OpenSSH relacionada con la gestión de ciertos mensajes SSH malformados. Un atacante remoto no autenticado puede enviar paquetes especialmente diseñados durante la fase inicial de negociación SSH que provocan respuestas anómalas del servidor. Aunque esta falla **no permite ejecución de código ni acceso no autorizado**, puede revelar información útil sobre el comportamiento del servidor o las configuraciones internas del mismo, como el tipo exacto de implementación, comportamiento del handshake y posibles vectores para ataques de enumeración o fingerprinting más avanzados.

- **Criticidad**: Baja
- **Impacto Potencial:**
  - Fuga de información pasiva que podría facilitar ataques futuros más dirigidos.
  - Mayor exposición al fingerprinting por parte de atacantes externos.
  - Riesgo elevado si se combina con otras vulnerabilidades en cadena.
- **Condiciones de Explotación:**
  - Puede ser explotado sin autenticación.
  - Requiere que el atacante envíe mensajes especialmente malformados al servidor SSH.
  - No requiere interacción del usuario ni ejecución de código.
- **Recomendación de Mitigación:**
  - **Actualizar OpenSSH a una versión posterior a 9.1p1**, que mitigue esta fuga de información.
  - Aplicar reglas de firewall que restrinjan el acceso SSH a IPs confiables.
  - Activar mecanismos de detección de anomalías para identificar intentos de fingerprinting o paquetes malformados.
  - Minimizar la información revelada por el banner SSH (`Banner` y `Version` configurados adecuadamente en `sshd_config`).

- **Nombre de la Vulnerabilidad**: Ejecución Remota de Código a través de señales asincrónicas mal gestionadas en OpenSSH ("RegreSSHion")
- **Servicio Afectado**: OpenSSH 8.5p1 hasta 9.3p1 (en algunos entornos, incluido 9.2p1 en Debian)
- **Puerto**: 22/TCP
- **CVE Asociado**: CVE-2024-6387
- **Descripción:**

  CVE-2024-6387, apodada “RegreSSHion”, es una vulnerabilidad crítica que permite a un atacante remoto no autenticado ejecutar código arbitrario como root en servidores OpenSSH afectados. El fallo radica en una condición de carrera en el manejador de señales del proceso `sshd`, específicamente en el procesamiento de señales asíncronas cuando el demonio se encuentra en un estado de espera durante la autenticación.

  Esta vulnerabilidad **representa una grave amenaza**, ya que permite una **toma de control total del sistema** de manera remota, sin necesidad de credenciales válidas.

- **Criticidad: Crítica**
- **Impacto Potencial:**
  - Ejecución remota de código como usuario root
  - Compromiso total del sistema
  - Instalación de puertas traseras o malware
  - Persistencia del atacante con privilegios elevados
- **Condiciones de Explotación:**
  - El servidor debe tener `sshd` expuesto al público y permitir conexiones desde redes no confiables.
  - Explotación posible sin autenticación.
  - Se requiere una serie de conexiones maliciosas cuidadosamente sincronizadas para desencadenar la condición de carrera.
  - La explotación puede tardar desde minutos hasta horas, dependiendo de la estabilidad del objetivo.
- **Recomendación de Mitigación:**
  - **Actualizar inmediatamente a una versión de OpenSSH posterior a 9.3p1**, donde la vulnerabilidad ha sido corregida.
  - Si no es posible actualizar, **compilar OpenSSH con la opción `--disable-libutil`** como mitigación temporal.
  - Aplicar reglas de firewall para restringir el acceso SSH a direcciones IP conocidas y de confianza.
  - Monitorizar los registros del sistema para detectar patrones anómalos de conexión a `sshd`.

- **Nombre de la Vulnerabilidad**: Desbordamiento de Búfer en la Lectura de Archivos de Configuración `sshd_config` en OpenSSH
- **Servicio Afectado**: OpenSSH antes de 9.7p1
- **Puerto**: 22/TCP
- **CVE Asociado**: CVE-2025-26465
- **Descripción:**

  Se ha descubierto una vulnerabilidad en OpenSSH que permite un **desbordamiento de búfer en el procesamiento del archivo de configuración `sshd_config`**, en particular cuando se emplean líneas excesivamente largas o manipuladas de forma maliciosa. Un atacante local con privilegios para editar este archivo podría aprovechar esta debilidad para provocar un fallo en el servicio o, en ciertos escenarios, ejecutar código arbitrario.

  Aunque no se trata de una vulnerabilidad explotable de forma remota en condiciones típicas, su presencia en entornos mal configurados o donde múltiples usuarios tienen acceso al sistema puede representar un riesgo serio de **escalamiento de privilegios** o **denegación de servicio**.

- **Criticidad: Media**
- **Impacto Potencial:**
  - Denegación de servicio al daemon `sshd`.
  - Ejecución de código localmente como root (en configuraciones comprometidas).
  - Pérdida de disponibilidad del servicio SSH.
  - Posible persistencia si es explotado por un usuario malicioso con acceso limitado.
- **Condiciones de Explotación:**
  - Requiere acceso local para modificar el archivo `sshd_config`.
  - Es más peligrosa en entornos con múltiples usuarios o donde los permisos del archivo de configuración no estén correctamente establecidos.
  - No es explotable de forma remota en condiciones normales.
- **Recomendación de Mitigación:**
  - **Actualizar OpenSSH a la versión 9.7p1 o posterior**, donde este fallo ha sido corregido.
  - **Verificar los permisos del archivo `/etc/ssh/sshd_config`**: debe ser propiedad de root y tener permisos 644 o más restrictivos.
  - **Auditar líneas largas o inusuales** en el archivo de configuración que pudieran estar siendo manipuladas.
  - **Limitar privilegios de usuarios locales** y evitar otorgar acceso administrativo innecesario.

- **Nombre de la Vulnerabilidad**: Vulnerabilidad de Canal Lateral por Manipulación del Intercambio de Mensajes SSH (Terrapin Attack)
- **Servicio Afectado:** OpenSSH 9.2p1 (y versiones anteriores compatibles con los modos de cifrado afectados)
- **Puerto**: 22/TCP
- **CVE Asociado**: CVE-2023-48795
- **Descripción:**
    - Esta vulnerabilidad, conocida como Terrapin, afecta a implementaciones del protocolo SSH que utilizan ciertos modos de cifrado (específicamente CBC, CTR y algunas combinaciones de EtM). El ataque permite a un actor malicioso que tenga control de la red (Man-in-the-Middle) modificar ciertos paquetes del handshake SSH sin que el cliente o servidor lo detecten, lo cual debilita las garantías de integridad y confidencialidad del canal.
    - Aunque no permite directamente obtener credenciales o ejecutar comandos, Terrapin reduce significativamente la seguridad del canal SSH, haciendo posibles otros ataques como el truncado de mensajes o la inyección de contenido controlado bajo condiciones específicas.
- **Criticidad: Media**
- **Impacto Potencial:**
    - Debilitamiento de la integridad de la sesión SSH.
    - Posible inyección o truncado de datos en la sesión SSH.
    - Escenario base para ataques más complejos (como downgrade o replay attacks).
- **Condiciones de Explotación:**
    - Requiere un atacante capaz de realizar **Man-in-the-Middle** entre el cliente y el servidor.
    - El servidor SSH debe estar utilizando uno de los **modos de cifrado vulnerables** (CBC, CTR, EtM).
    - No requiere autenticación.
- **Recomendación de Mitigación:**
    - **Actualizar OpenSSH** a una versión que implemente las mitigaciones contra Terrapin.
    - **Deshabilitar modos de cifrado vulnerables**, como `cbc`, `ctr` y aquellos con `EtM`, desde la configuración de `sshd_config` y `ssh_config`.
    - **Restringir el acceso SSH desde redes no confiables.**
    - **Utilizar herramientas de detección de MITM en la red**.

- **Nombre de la Vulnerabilidad**: Riesgo de Confidencialidad por Parámetros de Grupo Reutilizados en Diffie-Hellman (RFC 4253)
- **Servicio Afectado**: OpenSSH - Versiones que usan grupos Diffie-Hellman modp sin validación robusta (incluye OpenSSH 9.2p1)
- **Puerto**: 22/TCP
- **CVE Asociado**: CVE-2023-51384
- **Descripción:**
    - La vulnerabilidad **CVE-2023-51384** describe una debilidad en la implementación del protocolo SSH durante el intercambio de claves Diffie-Hellman (DH). Específicamente, los grupos modp (modular exponential groups) pueden ser reutilizados sin validación segura de los parámetros de la contraparte, lo que **puede permitir a un atacante realizar ataques de precomputación** u otras técnicas criptográficas para reducir la confidencialidad del canal.
    - Aunque se requiere un atacante con capacidad avanzada (como acceso previo a tráfico o tiempo prolongado para análisis), la falla **compromete el principio de secreto perfecto (forward secrecy)** si se explota con éxito.
- **Criticidad: Media**
- **Impacto Potencial:**
    - Reducción de la confidencialidad de las comunicaciones SSH.
    - Exposición potencial de sesiones antiguas si son capturadas y analizadas posteriormente.
    - Riesgo en entornos de alta seguridad o con almacenamiento de tráfico cifrado.
- **Condiciones de Explotación:**
    - El servidor debe permitir grupos modp débiles sin validación estricta.
    - Se requiere **intercepción del tráfico** y recursos computacionales significativos para análisis criptográfico.
    - Más factible en sesiones de larga duración o entornos donde el tráfico SSH es grabado.
- **Recomendación de Mitigación:**
    - **Actualizar OpenSSH** a una versión que refuerce la validación de parámetros DH.
    - **Configurar el servidor SSH para usar algoritmos más robustos**, como ECDH o curvas modernas (por ejemplo, `curve25519-sha256`).
    - **Deshabilitar grupos modp antiguos** en la configuración del servidor (`sshd_config`) mediante la opción `KexAlgorithms`.
    - **Monitorear configuraciones criptográficas activas** con herramientas como `ssh-audit`.

- **Nombre de la Vulnerabilidad**: Listado de Directorios Web sin Indexación Restringida
- **Servicio Afectado**: Servidor Web Apache 2.4.62 y estructura por defecto de WordPress
- **Puerto**: 80/TCP
- **CVE Asociado**: No aplica (vulnerabilidad de configuración)
- **Descripción:**
    - Durante el escaneo, se detectó que múltiples directorios del servidor web permiten el listado de contenidos (directory listing). Esto implica que, al acceder directamente a dichas rutas desde un navegador, el servidor muestra los archivos y subdirectorios contenidos sin necesidad de autenticación.
    - Esta mala configuración expone información sensible como scripts, recursos de plugins, temas o archivos temporales que podrían facilitar ataques como la identificación de versiones, explotación de vulnerabilidades conocidas o ingeniería inversa sobre la aplicación.
- **Directorios afectados:**
    - `/wp-includes/ /wp-admin/css/ /wp-admin/images/ /wp-admin/includes/ /wp-admin/js/ /wp-admin/maint/ /wp-admin/user/ /wp-content/plugins/ /wp-content/themes/ /wp-content/upgrade/ /wp-content/uploads/`
- **Criticidad: Media**
- **Impacto Potencial:**
    - Enumeración de archivos sensibles o scripts obsoletos.
    - Detección de versiones específicas de plugins o temas vulnerables.
    - Posible descubrimiento de archivos de configuración, copias de seguridad u otros vectores de ataque.
- **Condiciones de Explotación:**
    - Acceso HTTP al puerto 80 sin autenticación.
    - Indexación no deshabilitada por el servidor (por ejemplo, falta de `Options -Indexes` en la configuración de Apache o `.htaccess`).
- **Recomendación de Mitigación:**
    - Deshabilitar el listado de directorios en la configuración de Apache (`Options -Indexes`) o mediante reglas específicas en archivos `.htaccess`.
    - Asegurar que cada directorio contenga un archivo `index.html` o `index.php` vacío que impida el listado automático.
    - Restringir el acceso a rutas sensibles mediante autenticación o reglas de firewall/web server.
    - Revisar regularmente qué directorios están expuestos públicamente.

- **Nombre de la Vulnerabilidad**: Exposición del archivo xmlrpc.php – Superficie de Ataque Ampliada en WordPress
- **Servicio Afectado**: WordPress (CMS sobre servidor Apache)
- **Puerto**: 80/TCP
- **CVE Asociado**: No aplica directamente, pero está relacionada con múltiples vulnerabilidades históricas explotables a través de XML-RPC.
- **Descripción:**
    - Durante el escaneo , se detectó la presencia y accesibilidad del archivo `/xmlrpc.php`, una interfaz incorporada en WordPress para permitir comunicaciones remotas (por ejemplo, publicación desde apps móviles o herramientas de gestión).
    - Si bien tiene funcionalidades legítimas, **históricamente ha sido explotado para realizar ataques como fuerza bruta amplificada, escaneo de usuarios, pingbacks DDoS, y ejecución remota de comandos** mediante vulnerabilidades en plugins mal protegidos que utilizan este endpoint.
- **Criticidad: Media**
- **Impacto Potencial:**
    - Fuerza bruta de contraseñas mediante múltiples métodos XML-RPC.
    - Enumeración de usuarios válidos.
    - Explotación de vulnerabilidades de plugins o temas a través de solicitudes XML-RPC maliciosas.
    - Participación involuntaria en ataques DDoS mediante `pingback.ping`.
- **Condiciones de Explotación:**
    - Acceso HTTP al archivo `/xmlrpc.php` sin restricciones.
    - Servicios o plugins de WordPress que hagan uso de XML-RPC sin validación adecuada.
- **Recomendación de Mitigación:**
    - **Deshabilitar el archivo `xmlrpc.php`** si no se necesita esta funcionalidad, mediante `.htaccess` o configuración del servidor web:
        - apache
        - ```
          <Files xmlrpc.php>
              Order Deny,Allow
              Deny from all
          </Files>
          ```
    - Si es necesario mantenerlo habilitado, aplicar soluciones de seguridad como:
        - Plugins que filtran o bloquean llamadas XML-RPC maliciosas.
        - Autenticación reforzada con tokens o protección de endpoint.
    - Monitorizar registros para detectar actividad anómala relacionada con `xmlrpc.php`.
          ```

- **Nombre de la Vulnerabilidad**: Ausencia de la cabecera X-Frame-Options – Exposición a ataques de Clickjacking

- **Servicio Afectado**: Servidor web Apache con aplicación WordPress

- **Puerto**: 80/TCP

- **CVE Asociado**: No aplica (es una mala práctica de configuración)

- **Descripción:**

    - Durante el análisis realizado con herramientas de pentesting, se detectó que el servidor web no envía la cabecera HTTP **`X-Frame-Options`** en las respuestas. Esta cabecera es fundamental para proteger a los usuarios frente a ataques de **clickjacking**, donde un atacante incrusta el sitio web legítimo en un `iframe` oculto o manipulado, engañando al usuario para que realice acciones involuntarias (como clics en botones o envío de formularios).

    - Sin esta protección, es posible que un atacante monte un sitio malicioso que cargue el sitio objetivo en un `iframe`, superponiendo elementos invisibles o falsos, generando un entorno de ataque invisible para el usuario.

- **Criticidad: Media**

- **Impacto Potencial:**

    - Robo de información (por ejemplo, credenciales o tokens de sesión).

    - Realización de acciones no autorizadas por parte de usuarios legítimos (cambio de contraseñas, eliminación de contenido, etc.).

    - Compromiso de la integridad de la interfaz gráfica y percepción de seguridad.

- **Condiciones de Explotación:**

    - Sitio web accesible mediante navegador sin cabecera X-Frame-Options.

    - Usuario autenticado o activo en sesión.

    - Acceso a navegador vulnerable a la técnica (todos los actuales si no se protege correctamente).

- **Recomendación de Mitigación:**

    - Configurar el servidor web (Apache, Nginx, etc.) o la aplicación para enviar la cabecera `X-Frame-Options` con uno de los siguientes valores:

        - `http`
        - `X-Frame-Options: DENY`
    - o bien:
        - `http`
        - `X-Frame-Options: SAMEORIGIN`

    - Como alternativa moderna, implementar la política de encabezados `Content-Security-Policy (CSP)` con la directiva `frame-ancestors`:

        - `http`
        - `Content-Security-Policy: frame-ancestors 'self';`
    - Verificar que no haya plugins o configuraciones en WordPress que eliminen o sobrescriban estas cabeceras.

- **Nombre de la Vulnerabilidad**: Ausencia de la cabecera X-Content-Type-Options – Riesgo de interpretación errónea de contenido (MIME Sniffing)
- **Servicio Afectado**: Servidor web Apache con aplicación WordPress
- **Puerto**: 80/TCP
- **CVE Asociado**: No aplica (es una mala práctica de configuración)
- **Descripción:**
    - Durante el análisis del servidor web, se identificó la ausencia de la cabecera HTTP **`X-Content-Type-Options`** en las respuestas del servidor. Esta cabecera es utilizada para indicar a los navegadores que deben respetar el tipo de contenido declarado en los encabezados `Content-Type`, y no intentar adivinarlo (técnica conocida como **MIME Sniffing**).
    - Cuando esta cabecera no está presente, ciertos navegadores (especialmente versiones antiguas de Internet Explorer y otros motores que permiten esta técnica) pueden interpretar incorrectamente el contenido, lo que puede ser aprovechado por un atacante para entregar archivos maliciosos (por ejemplo, scripts camuflados como imágenes) y ejecutar código en el navegador de la víctima.
- **Criticidad: Media**
- **Impacto Potencial:**
    - Posible ejecución de scripts maliciosos interpretados como otros tipos de archivo.
    - Riesgo de ataques XSS basados en el contenido incorrectamente interpretado.
    - Aumento de superficie de ataque debido a comportamiento no previsto del navegador.
- **Condiciones de Explotación:**
    - El contenido debe ser accesible desde un navegador vulnerable.
    - El atacante debe lograr subir o servir contenido con un tipo MIME ambiguo.
    - No se requieren privilegios especiales para la explotación.
- **Recomendación de Mitigación:**
    - Configurar el servidor web para enviar la siguiente cabecera en todas las respuestas HTTP:
        - `http`
        - `X-Content-Type-Options: nosniff`
    - En Apache, esto puede lograrse mediante el siguiente ajuste en la configuración o archivo `.htaccess`:
        - `apache`
        - `Header set X-Content-Type-Options "nosniff"`
    - Verificar que no haya proxies o firewalls que eliminen esta cabecera.
    - Implementar una política de cabeceras segura junto con otras como `Content-Security-Policy` y `X-Frame-Options`.

- **Nombre de la Vulnerabilidad**: Filtrado de Información del Sistema de Archivos mediante ETags
- **Servicio Afectado**: Servidor Web (Apache u otros compatibles con ETags)
- **Puerto**: 80/TCP
- **CVE Asociado**: CVE-2003-1418
- **Descripción:**
    - El escaneo realizado ha revelado que el servidor web utiliza cabeceras **ETag** que incluyen información sobre los **inodos, tamaño y marcas de tiempo** de archivos. Esta implementación permite a un atacante inferir detalles del sistema de archivos subyacente, lo que puede ser aprovechado para ataques de reconocimiento y comparación de archivos entre distintos sistemas.
    - Los ETags (Entity Tags) están diseñados para identificar versiones de recursos web, pero cuando contienen identificadores del sistema de archivos (como inodos), se convierten en una fuente de **filtrado indirecto de información sensible**, contraviniendo buenas prácticas de seguridad.
- **Criticidad: Baja**
- **Impacto Potencial:**
    - Divulgación de detalles del sistema de archivos (estructura de inodos y modificaciones).
    - Posible correlación de archivos entre distintos servidores (usado en cache poisoning o fingerprinting).
    - Riesgo de exposición indirecta de configuraciones o contenido sensible.

- **Condiciones de Explotación:**
    - Requiere únicamente el acceso al servidor web (no requiere autenticación).
    - Puede ser explotado con herramientas automatizadas de escaneo HTTP (Nikto, Burp Suite, etc.).
    - Depende de cómo el servidor construya las cabeceras ETag.
- **Recomendación de Mitigación:**
    - **Deshabilitar o reconfigurar las ETags** en el servidor web para que no utilicen identificadores del sistema de archivos.
        - En Apache, puede hacerse agregando:
            - `apache`
            - `FileETag None`
    - **Auditar y revisar las cabeceras HTTP** servidas por el servidor.
    - **Aplicar las mejores prácticas de seguridad web** para evitar la fuga de información.
    - Monitorizar solicitudes HTTP sospechosas y realizar análisis de logs periódicamente.

- **Nombre de la Vulnerabilidad**: Vulnerabilidad de Subida de Archivos Arbitrarios en el Plugin *wp-big-video-background*
- **Servicio Afectado**: WordPress (plugin: wp-big-video-background) 1.1.0
- **Puerto**: 80/TCP
- **CVE Asociado**: CVE-2021-24183
- **Descripción:**
    - Nuestro analisis ha detectado la presencia del plugin vulnerable **wp-big-video-background** versión **1.1.0**, el cual es susceptible a una vulnerabilidad crítica de subida de archivos arbitrarios (*Unrestricted File Upload*). Esta vulnerabilidad permite a un atacante autenticado con privilegios mínimos, como un suscriptor, subir archivos maliciosos al servidor, incluyendo posibles **webshells o scripts PHP ejecutables**, lo cual habilita una **toma completa del sistema**.
    - Dado que los plugins de WordPress suelen ejecutarse en contextos con permisos elevados dentro del CMS, esta vulnerabilidad representa un vector serio de compromiso, especialmente si el sistema no aplica validaciones adicionales a los archivos subidos.
- **Criticidad: Alta**
- **Impacto Potencial:**
    - Ejecución remota de código (RCE) en el servidor a través de archivos maliciosos.
    - Compromiso total del sitio WordPress y del sistema operativo subyacente.
    - Persistencia mediante backdoors o shells remotas.
    - Escalada de privilegios desde usuarios autenticados de bajo nivel.
- **Condiciones de Explotación:**
    - Requiere usuario autenticado (incluso con rol bajo como "subscriber").
    - El plugin vulnerable debe estar activo en el sitio WordPress.
    - No se requieren privilegios administrativos para la explotación básica.
- **Recomendación de Mitigación:**
    - **Eliminar o desactivar inmediatamente el plugin `wp-big-video-background` versión 1.1.0** si está instalado.
    - Si se requiere su funcionalidad, buscar una **actualización segura o alternativa** libre de esta vulnerabilidad.
    - Revisar archivos subidos al servidor en busca de posibles webshells o accesos no autorizados.
    - Aplicar un firewall de aplicaciones web (WAF) que limite la ejecución de archivos subidos.
    - Limitar los privilegios de usuarios autenticados y monitorear actividades sospechosas.

- **Nombre de la Vulnerabilidad**: Subida de Archivos Arbitrarios No Autenticada en el Plugin *wp-file-manager*
- **Servicio Afectado**: WordPress (plugin: wp-file-manager) 7.0
- **Puerto**: 80/TCP
- **CVE Asociado**: CVE-2020-25213
- **Descripción:**
    - Identificamos que el plugin **wp-file-manager** en su versión **7.0** está presente y es vulnerable a una **subida de archivos arbitrarios sin autenticación** (*Unauthenticated Arbitrary File Upload*). Esta vulnerabilidad permite que un atacante remoto, sin necesidad de autenticarse, suba archivos maliciosos directamente al servidor web. En la práctica, esto se traduce en la posibilidad de **ejecutar código remoto (RCE)** mediante la colocación de webshells o scripts PHP en rutas accesibles. Debido a la ausencia de validaciones adecuadas en los puntos de subida de archivos, y al hecho de que no se requiere autenticación, esta vulnerabilidad representa un **riesgo crítico e inmediato para la seguridad del sistema**.
- **Criticidad: Crítica**
- **Impacto Potencial:**
    - **Ejecución remota de código (RCE)**.
    - **Toma total del servidor** web y/o del sistema operativo si el usuario del servidor web tiene privilegios elevados.
    - **Instalación de puertas traseras (backdoors)** para persistencia.
    - Robo o modificación de datos, defacement o pivoting hacia otros sistemas internos.
    - Pérdida de la confidencialidad, integridad y disponibilidad del sistema.
- **Condiciones de Explotación:**
    - **No requiere autenticación**.
    - Plugin `wp-file-manager` activo y en su versión vulnerable.
    - Acceso HTTP al sistema desde el exterior.
- **Recomendación de Mitigación:**
    - **Eliminar inmediatamente plugin `wp-file-manager` si se encuentra en la versión 7.0**.
    - En caso de requerirse, actualizar a una versión segura posterior a la corrección de esta vulnerabilidad.
    - Realizar una auditoría completa del sistema en busca de webshells u otros indicadores de compromiso.
    - Restringir el acceso HTTP al panel de WordPress con reglas de firewall.
    - Implementar un **WAF (Web Application Firewall)** para inspección y filtrado de cargas maliciosas.
    - Supervisar registros de acceso HTTP y actividades anómalas.

- **Nombre de la Vulnerabilidad**: Ejecución Remota de Código (RCE) en el Plugin *wp-vcd*
- **Servicio Afectado**: WordPress (plugin: wp-vcd) 1.0
- **Puerto**: 80/TCP
- **CVE Asociado**: CVE-2017-1001000
- **Descripción:**
    - El plugin **wp-vcd**, presente en la versión 1.0, es conocido por contener código malicioso que permite la **ejecución remota de código (RCE)** tras la autenticación en el panel de administración de WordPress. Esta vulnerabilidad ha sido ampliamente explotada en entornos donde se obtienen credenciales débiles o comprometidas, como las detectadas mediante ataques de fuerza bruta.
    - Este malware se distribuye comúnmente a través de plugins nulled (pirateados) y una vez activo, **modifica archivos del núcleo de WordPress** para insertar puertas traseras persistentes. Desde ese punto, el atacante puede ejecutar comandos arbitrarios, subir archivos, y tomar control completo del sitio.
- **Criticidad: Crítica**
- **Impacto Potencial:**
    - **Ejecución remota de comandos en el servidor** web bajo los privilegios del servicio.
    - **Instalación de puertas traseras (backdoors)** y persistencia.
    - **Control total del sitio web WordPress**, incluso si se elimina el plugin visible.
    - Posible escalada de privilegios o pivoting hacia otros sistemas internos.
- **Condiciones de Explotación:**
    - Requiere **acceso autenticado al panel de administración de WordPress**.
    - Plugin `wp-vcd` instalado y activo.
    - Las credenciales pueden ser obtenidas a través de ataques de fuerza bruta
- **Recomendación de Mitigación:**
    - **Eliminar inmediatamente el plugin `wp-vcd`**, ya que no tiene un propósito legítimo y está categorizado como malware.
    - Realizar una revisión completa de los archivos del núcleo de WordPress para eliminar modificaciones maliciosas (como inyecciones en `functions.php`).
    - Cambiar todas las credenciales administrativas y reforzar con contraseñas robustas.
    - Instalar plugins desde fuentes oficiales únicamente.
    - Implementar autenticación en dos factores (2FA) para usuarios con privilegios elevados.
    - Monitorizar el tráfico y los archivos en busca de signos de persistencia o reexploitación.

- **Nombre de la Vulnerabilidad**: Divulgación de Ruta Completa (Full Path Disclosure)
- **Servicio Afectado**: WordPress (se detectó en plugins del entorno)
- **Puerto**: 80/TCP
- **CVE Asociado**: No aplica (varía según el plugin específico, pero el comportamiento general es conocido)
- **Descripción:**
    - Durante el análisis se identificó la presencia de una vulnerabilidad de tipo **Full Path Disclosure (FPD)** en uno o varios plugins del sitio WordPress. Este fallo permite que, al provocar ciertos errores o acceder a rutas mal formadas, el servidor devuelva mensajes de error que **revelan la estructura completa del sistema de archivos** del servidor (por ejemplo: `/var/www/html/wp-content/plugins/plugin-x/file.php on line 23`).
    - Este tipo de divulgación, aunque no es crítica por sí misma, **proporciona información valiosa al atacante**, como la ubicación exacta de archivos y rutas de ejecución, lo cual puede facilitar ataques más avanzados, como ejecución de código, LFI, o escalada de privilegios en combinación con otras vulnerabilidades.
- **Criticidad: Baja**
- **Impacto Potencial:**
    - Exposición de rutas completas del servidor (`full path`), útil para ataques dirigidos.
    - Posible asistencia a ataques de inclusión de archivos (LFI/RFI).
    - Facilita la elaboración de payloads más específicos en exploits personalizados.
    - Puede revelar estructura interna del servidor, nombres de archivos o configuraciones mal protegidas.
- **Condiciones de Explotación:**
    - No requiere autenticación.
    - Basta con acceder a rutas específicas o generar errores en plugins vulnerables.
    - El error debe no estar capturado adecuadamente (falta de manejo de excepciones).
- **Recomendación de Mitigación:**
    - Activar la directiva **`display_errors = Off`** en el archivo `php.ini` en entornos de producción.
    - Configurar los plugins para **no mostrar errores en pantalla** (registrarlos en un log seguro en su lugar).
    - Usar herramientas de análisis automatizado para detectar instancias de FPD en todos los plugins.
    - Actualizar o reemplazar los plugins vulnerables por versiones que no revelen información sensible.
    - Aplicar reglas en el servidor web para manejar errores personalizados (custom 500 pages).

7. Recomendaciones Generales

A partir del análisis de seguridad realizado sobre el servidor Debian y su entorno web basado en WordPress, se ha detectado un conjunto relevante de vulnerabilidades técnicas, debilidades en configuración y posibles vectores de ataque. A continuación, se presentan recomendaciones generales organizadas por áreas clave para fortalecer la postura de seguridad del sistema:

- Fortalecimiento del Acceso y Autenticación

- **Deshabilitar el acceso FTP anónimo**: Este acceso permite la lectura (e incluso escritura) de archivos sin autenticación, exponiendo datos sensibles. Se recomienda eliminar el acceso anónimo en `vsftpd.conf` (`anonymous_enable=NO`).
- **Migrar de FTP a SFTP o FTPS**: Las conexiones actuales son en texto plano, lo que permite interceptar credenciales. Se recomienda implementar SFTP sobre SSH.
- **Cambiar credenciales débiles**: La contraseña del usuario `root` fue descubierta mediante ataque de diccionario. Debe cambiarse inmediatamente a una contraseña compleja y única.
- **Restringir acceso SSH por IP**: Implementar restricciones en `/etc/hosts.allow` o mediante reglas de firewall para reducir superficie de ataque.
- **Deshabilitar el reenvío X11 en SSH**: Esto mitiga riesgos como el CVE-2023-38408.

- Endurecimiento del Servidor

- **Actualizar todos los paquetes vulnerables**: Aplicar actualizaciones del sistema con `apt update && apt upgrade` para mitigar CVEs activos.
- **Deshabilitar módulos innecesarios de Apache**: Eliminar módulos no utilizados para reducir vectores de ataque.
- **Revisar los headers HTTP de seguridad**:
    - Añadir `X-Frame-Options: DENY` para evitar clickjacking.
    - Añadir `X-Content-Type-Options: nosniff` para evitar sniffing de contenido.
    - Añadir `Strict-Transport-Security` si se habilita HTTPS.

- Revisión de WordPress

- **Eliminar plugins vulnerables**: Como `wp-file-manager`, `wp-vcd`, y `wp-big-video-background`, que presentan vulnerabilidades críticas.
- **Actualizar WordPress y plugins a sus versiones más recientes.**
- **Revisar los permisos de los directorios listables**: Configurar Apache para evitar el listado de directorios (`Options -Indexes`).
- **Restringir acceso a xmlrpc.php**: Deshabilitarlo si no se utiliza o limitar su uso mediante reglas en `.htaccess`.
- **Deshabilitar la enumeración de usuarios**: Esto evitará que atacantes descubran usuarios válidos como `admin`.

- Gestión de Vulnerabilidades

- **Implementar un sistema de escaneo periódico**: Automatizar escaneos de vulnerabilidades y mantener documentación de hallazgos.
- **Utilizar herramientas de monitoreo de integridad** como AIDE o OSSEC para detectar cambios sospechosos en el sistema de archivos.
- **Revisar logs periódicamente**: Incluyendo `/var/log/auth.log`, Apache, y logs de WordPress, buscando accesos anómalos o fallidos.

- Políticas de Seguridad y Backup

- **Establecer una política de backups regulares y verificados.**
- **Implementar doble factor de autenticación (2FA)** para todos los usuarios de administración.
- **Capacitar a los usuarios y administradores** sobre buenas prácticas de seguridad.

Las recomendaciones anteriores tienen como objetivo mitigar las vulnerabilidades detectadas y fortalecer la seguridad del entorno. Se sugiere su implementación prioritaria en función de la criticidad de cada hallazgo. Un enfoque proactivo de monitoreo y actualización continua es fundamental para reducir riesgos de compromisos futuros.

8. CONCLUSIONES

Tras realizar un análisis de seguridad exhaustivo sobre la máquina objetivo, se ha podido comprobar que el sistema presenta múltiples debilidades y vulnerabilidades que podrían comprometer seriamente su integridad, confidencialidad y disponibilidad si no se corrigen de forma oportuna.

Durante el proceso se han detectado varios vectores de ataque potenciales a través de servicios expuestos como **FTP**, **SSH** y **HTTP**. Entre los hallazgos más relevantes destacan:

- **FTP con acceso anónimo y sin cifrado**, lo que permite el acceso no autenticado y la exposición de datos en texto plano.
- **SSH vulnerable a múltiples CVEs críticos (ej. CVE-2023-38408, CVE-2023-28531, CVE-2024-6387)**, además de haberse logrado el acceso exitoso mediante ataque de fuerza bruta.
- **Aplicación web WordPress** con versiones antiguas, directorios listables, presencia de plugins altamente vulnerables y cabeceras de seguridad ausentes.
- **Exposición de interfaces peligrosas como xmlrpc.php** y **Full Path Disclosures**, que ofrecen información sensible útil para atacantes.
- **Configuraciones por defecto y falta de endurecimiento en servicios clave**, junto a una ausencia general de mecanismos de defensa como firewalls restrictivos o sistemas de detección de intrusiones.

El conjunto de vulnerabilidades encontradas demuestra que el sistema no está adecuadamente protegido frente a amenazas comunes ni frente a ataques dirigidos. Se evidencia una **falta de mantenimiento**, **actualizaciones**, y **políticas de seguridad activas**.

No obstante, también se ha observado que varios de los servicios utilizados son versiones conocidas y estables, lo que representa una buena base sobre la que implementar mejoras.

En resumen:

- Se confirma la viabilidad de múltiples ataques, incluyendo **escalada de privilegios, ejecución remota de código, denegación de servicio** y **divulgación de información sensible**.
- Se ha demostrado que, es posible comprometer servicios críticos sin necesidad de técnicas avanzadas.
- La aplicación de buenas prácticas de ciberseguridad, actualizaciones periódicas y una configuración más restrictiva evitaría que la mayoría de estos ataques pudieran ejecutarse con éxito.

Se recomienda encarecidamente seguir las medidas correctivas descritas en el apartado de recomendaciones y establecer un proceso continuo de gestión de vulnerabilidades para proteger el sistema a largo plazo.