PROYECTO FINAL DE CIBERSEGURIDAD

Renato Lizza

Introduccion

Este informe documenta el análisis de un entorno comprometido en una máquina Debian proporcionada por 4Geeks. El objetivo ha sido detectar el punto de entrada del atacante, analizar vulnerabilidades adicionales y aplicar un plan estructurado de recuperación y prevención conforme a ISO 27001.

Alcance del Proyecto

Este proyecto forma parte de un ejercicio práctico de análisis forense y evaluación de seguridad sobre una máquina virtual comprometida. La máquina objetivo es un servidor Debian .ova previamente hackeado, proporcionado por el equipo académico como simulación de un entorno real afectado por incidentes de ciberseguridad.

El objetivo de este ejercicio fue:

- Identificar evidencias de compromiso.
- Evaluar configuraciones débiles y vulnerabilidades existentes.
- Acceder al sistema mediante técnicas de análisis pasivo y explotación controlada.
- Documentar hallazgos de seguridad con sus respectivas mitigaciones.
- Recomendar acciones correctivas que refuercen la postura de seguridad del sistema.

Herramientas Utilizadas

Durante el análisis y explotación del entorno comprometido, se emplearon las siguientes herramientas:

- Nmap
- Hydra
- Firefox
- Terminal Linux
- WordPress
- Comandos Unix (ls, chmod, chown, etc.)

Entorno y Máquinas Analizadas

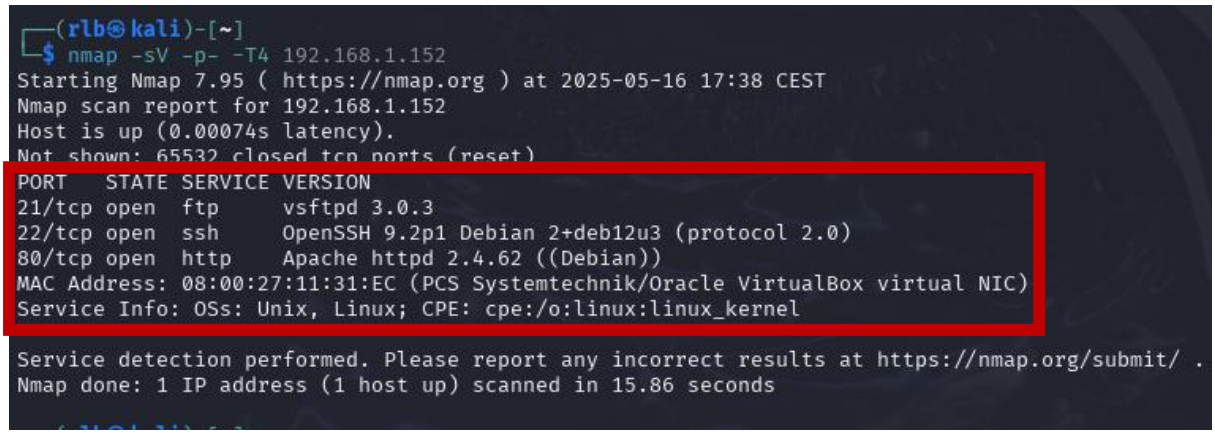
El análisis se realizó sobre el siguiente entorno simulado:

Maquina virtual	IP asignada	Rol
Debian.ova (objetivo)	192.168.1.152	Servidor comprometido con Apache, WordPress, MySQL, FTP y SSH.
Kali Linux (analista)	Local	Sistema de análisis, ataque y exploración forense

La máquina comprometida fue desplegada en VirtualBox con el objetivo de simular un escenario realista de intrusión y mala configuración, permitiendo al analista realizar un diagnóstico técnico completo y simular escenarios de explotación controlada.

Primera Fase:Reconocimiento

Análisis de Puertos Abiertos – Escaneo Nmap



Interpretación y Riesgos Potenciales

- FTP (puerto 21):
  - El servicio vsftpd está activo. Debería verificarse si permite acceso anónimo o si está mal configurado, ya que es un vector común de entrada.
- SSH (puerto 22):
  - El servidor SSH responde con OpenSSH 9.2p1. Es importante comprobar si permite acceso con contraseña débil o si permite login directo como root.
- HTTP (puerto 80):
  - Apache HTTP Server está operativo. Requiere un análisis más profundo (por ejemplo, fuzzing o escaneo de directorios) para detectar vulnerabilidades web como XSS, LFI o inyecciones.

Verificación de Usuario MySQL

Durante el análisis forense, se examinó el historial de comandos ejecutados por el usuario root en el sistema. Se identificó la creación del usuario MySQL wordpressuser con una contraseña extremadamente débil (123456) en texto plano.

```
debian@debian:~$ sudo cat /root/.mysql_history
[sudo] password for debian:
_HiStOrY_V2_
CREATE\040DATABASE\040wordpress\040DEFAULT\040CHARACTER\040SET\040utf8\040COLLATE\040utf8_unicode_ci;
CREATE\040USER\040'wordpressuser'@'localhost'\040IDENTIFIED\040BY\040'123456';
GRANT\040ALL\040PRIVILEGES\040ON\040wordpress.*\040TO\040'wordpress'@'localhost';\040
GRANT\040ALL\040PRIVILEGES\040ON\040wordpress.*\040TO\040'wordpressuser'@'localhost';
FLUSH\040PRIVILIGES;
FLUSH\040PRIVILEGES;
EXIT;
CREATE\040USER\040'user'@'localhost'\040IDENTIFIED\040BY\040'password';
GRANT\040ALL\040PRIVILEGES\040ON\040*.*\040TO\040'user'@'localhost'\040WITH\040GRANT\040OPTION;
FLUSH\040PRIVILEGES;
EXIT;
```

Por resaltar el usuario wordpressuser tiene privilegios completos sobre la base de datos y el Usuario user tambien y la opcion de otorgar privilegios a otros usuarios.

Riesgo identificado

- Uso de una contraseña débil (123456), fácilmente explotable por diccionario o fuerza bruta.
- Almacenamiento de la contraseña en texto plano en un archivo accesible por el usuario root.
- Permisos excesivos al usuario (ALL PRIVILEGES) sobre la base de datos.

Detección de archivos críticos e inseguros

Durante el análisis de la instalación de WordPress en el servidor, se revisaron los permisos del directorio raíz de la aplicación web: /var/www/html/

El comando ejecutado fue Ls –la /var/www/html/ y se encontro permiso y contraseña en toda la base de datos expuestas en el archivo wp-confing.php

```
debian@debian:~$ ls -la /var/www/html/
total 260
drwxrwxrwx  5 www-data www-data  4096 May 16 11:39 .
drwxr-xr-x  3 root     root      4096 Sep 30  2024 ..
-rwxrwxrwx  1 www-data www-data   523 Sep 30  2024 .htaccess
-rwxrwxrwx  1 www-data www-data 10701 Sep 30  2024 index.html
-rwxrwxrwx  1 www-data www-data   405 Feb  6  2020 index.php
-rwxrwxrwx  1 www-data www-data 19903 May 16 11:39 license.txt
-rwxrwxrwx  1 www-data www-data  7425 May 16 11:39 readme.html
-rwxrwxrwx  1 www-data www-data  7387 Feb 13  2024 wp-activate.php
drwxrwxrwx  9 www-data www-data  4096 Sep 10  2024 wp-admin
-rwxrwxrwx  1 www-data www-data   351 Feb  6  2020 wp-blog-header.php
-rwxrwxrwx  1 www-data www-data  2323 Jun 14  2023 wp-comments-post.php
-rwxrwxrwx  1 www-data www-data  3017 Sep 30  2024 wp-config.php
-rwxrwxrwx  1 www-data www-data  3336 May 16 11:39 wp-config-sample.php
drwxrwxrwx  6 www-data www-data  4096 May 16 11:39 wp-content
-rwxrwxrwx  1 www-data www-data  5617 May 16 11:39 wp-cron.php
drwxrwxrwx 30 www-data www-data 12288 May 16 11:39 wp-includes
-rwxrwxrwx  1 www-data www-data  2502 Nov 26  2022 wp-links-opml.php
-rwxrwxrwx  1 www-data www-data  3937 Mar 11  2024 wp-load.php
-rwxrwxrwx  1 www-data www-data 51414 May 16 11:39 wp-login.php
-rwxrwxrwx  1 www-data www-data  8727 May 16 11:39 wp-mail.php
-rwxrwxrwx  1 www-data www-data 30081 May 16 11:39 wp-settings.php
-rwxrwxrwx  1 www-data www-data 34516 May 16 11:39 wp-signup.php
-rwxrwxrwx  1 www-data www-data  5102 May 16 11:39 wp-trackback.php
-rwxrwxrwx  1 www-data www-data  3205 May 16 11:39 xmlrpc.php
```

Escaneo de Rootkits y Malware

Observaciones

Rootkits: Durante la ejecución del escaneo de rootkits con la herramienta chkrootkit, se detectó un posible proceso relacionado con captura de paquetes en la interfaz de red principal (enp0s3). Esto fue reportado por el módulo sniffer de la herramienta como un resultado con advertencia.

```
Checking `sniffer'...                                    WARNING

WARNING: Output from ifpromisc:
lo: not promisc and no packet sniffer sockets
enp0s3: PACKET SNIFFER(/usr/sbin/NetworkManager[431], /usr/sbin/NetworkManager[4
31])
```

Este resultado indica que el proceso NetworkManager estaba vinculado a la interfaz en modo promiscuo, una técnica comúnmente utilizada por herramientas de análisis de red y sniffers.

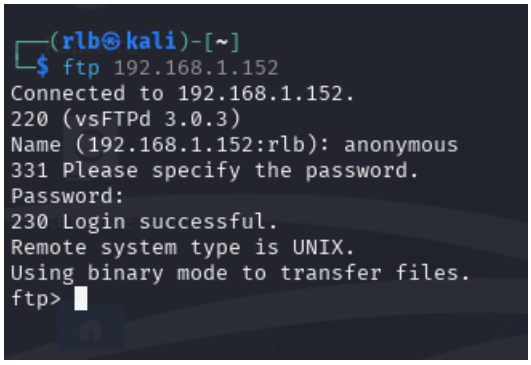
Rkhunter: Durante el escaneo del sistema con la herramienta rkhunter, se detectó una configuración crítica en el servicio SSH que permite el acceso directo al sistema como usuario root.

```
Checking for an SSH configuration file                  [ Found ]
Checking if SSH root access is allowed                  [ Warning ]
Checking if SSH protocol v1 is allowed                  [ Not set ]
```

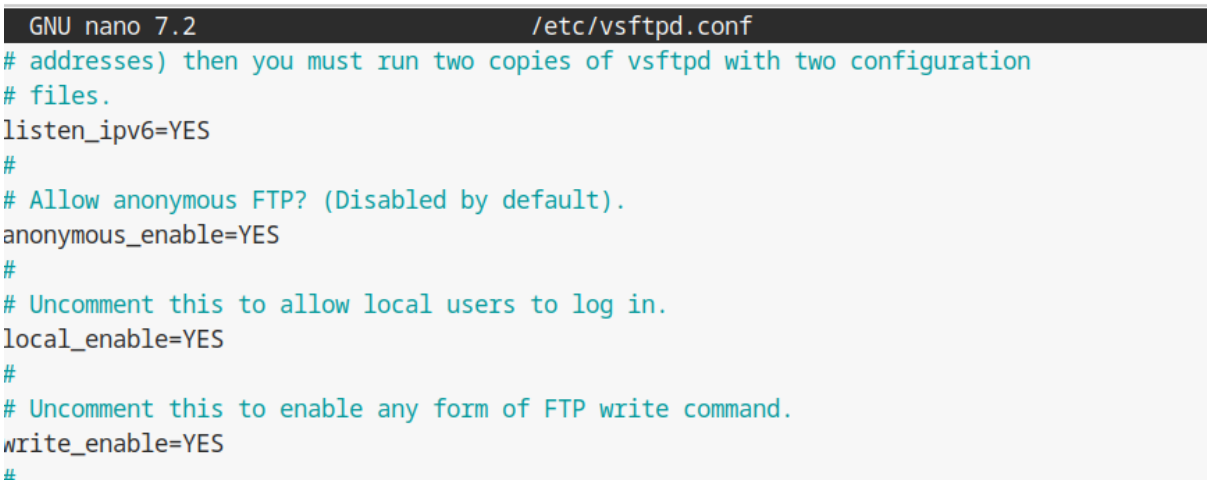
El acceso SSH como root está permitido por configuración. Esto significa que cualquier usuario con las credenciales correctas puede iniciar sesión directamente con privilegios de administrador, sin necesidad de escalar desde un usuario limitado.

Vulnerabilidad en Ftp

Durante el análisis de seguridad del servidor, se identificó que el servicio FTP (vsFTPD 3.0.3) está configurado con acceso anónimo habilitado y permite comandos de escritura, lo cual representa una vulnerabilidad crítica.



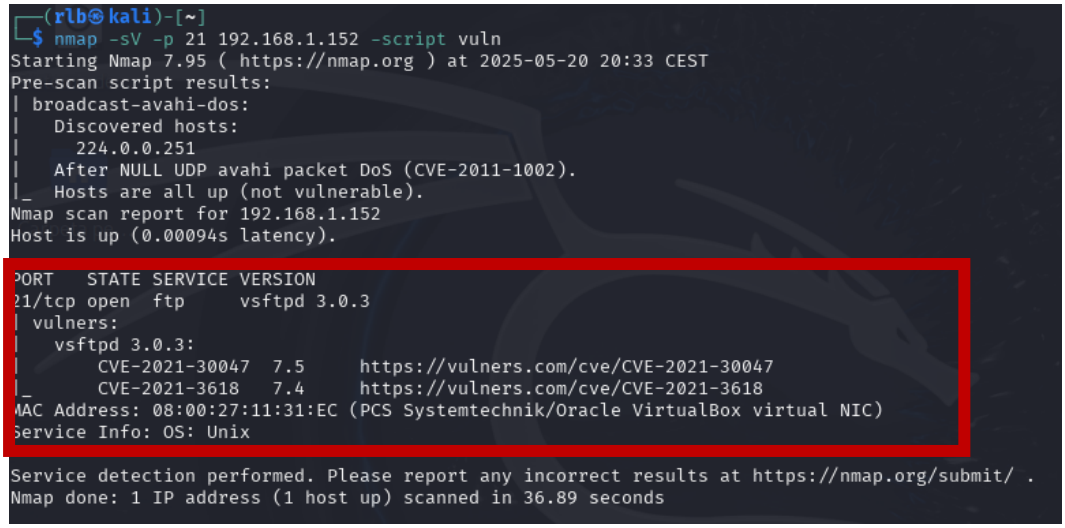
Se estableció conexión remota al servidor (192.168.1.152) a través del cliente FTP. El sistema permitió autenticación sin credenciales válidas mediante el usuario anonymous, confirmando el acceso sin autenticación.



Configuración insegura confirmada en el archivo /etc/vsftpd.conf

Desde la propia máquina Debian se verificó que los parámetros de configuración están habilitados para acceso anónimo y escritura.

Detección de versión vulnerable de vsFTPd



CVE	Severidad	Descripción breve
CVE-2021-30047	7.5	Vulnerabilidad que puede permitir ejecución remota de comandos.
CVE-2021-3618	7.4	Fallo de seguridad que puede comprometer la integridad del servicio FTP.

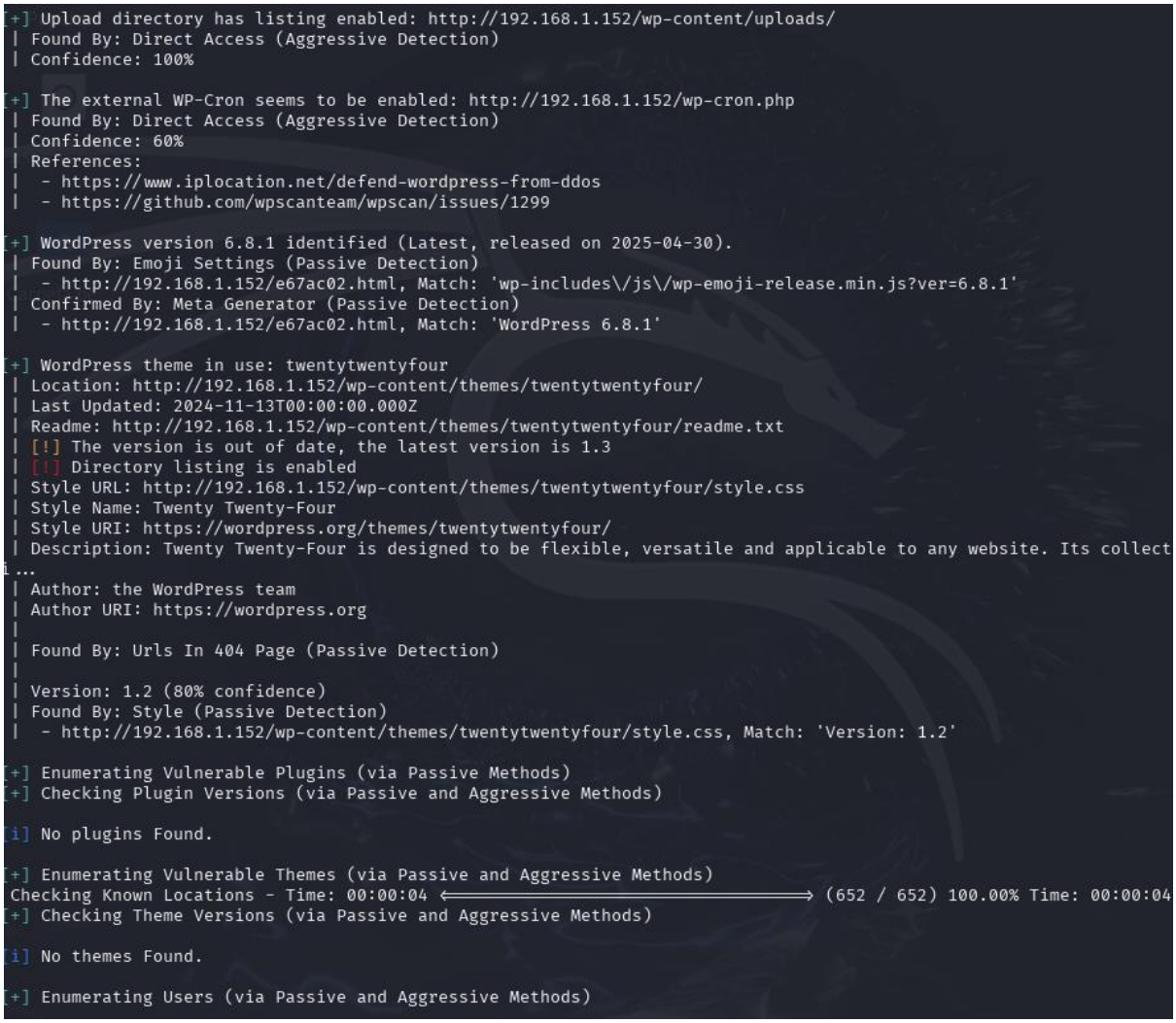
- La presencia de estas CVEs indica que el software FTP utilizado en el servidor no ha sido actualizado ni parcheado.
- Combinado con el acceso anónimo y permisos de escritura, representa una superficie de ataque extremadamente peligrosa.
- Un atacante con acceso a la red podría explotar estas vulnerabilidades para obtener acceso no autorizado o ejecución remota de código (RCE).

Vulnerabilidad en HTTP

Para comprobar las vulnerabilidades en HTTP utilizamos wpscan

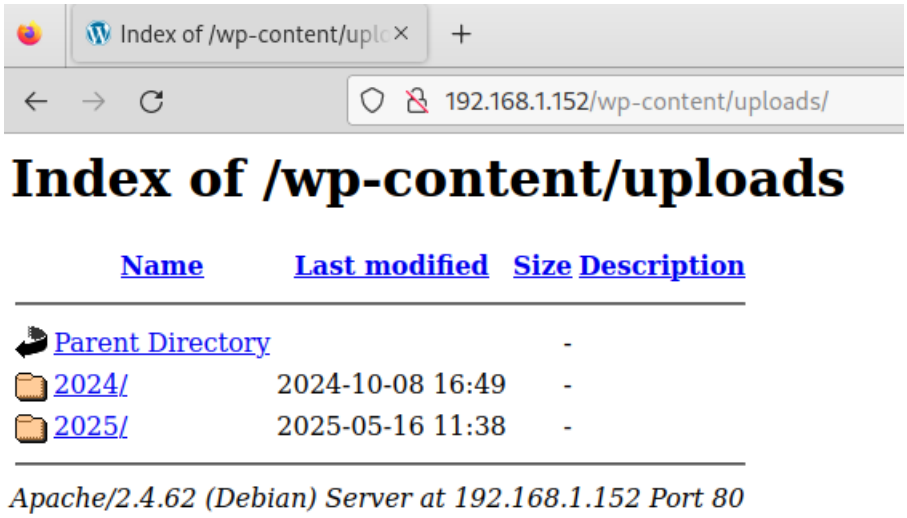
- Servidor web Apache/2.4.62 (Debian) detectado en las cabeceras.
- Presencia de robots.txt revelando rutas internas: /wp-admin/ y /admin-ajax.php.
- Archivo readme.html expuesto, confirmando instalación de WordPress.
- Directorio /wp-content/uploads/ con listado habilitado, lo que permite navegar archivos subidos.
- Interfaz XML-RPC habilitada, lo que puede ser explotado para ataques de:
    - Denegación de servicio (DoS)
    - Enumeración de usuarios
    - Fuerza bruta
    - Pingback attack

Estos elementos son signos de una instalación mal endurecida, fácilmente explotable en fases de reconocimiento y explotación web.



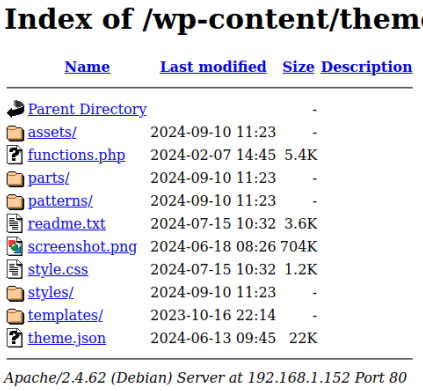
- **Tema en uso: twentytwentyfour versión 1.2**, pero la última versión disponible es la 1.3.
- El directorio del tema está **listable públicamente**, permitiendo a un atacante descargar el código fuente del tema y buscar vulnerabilidades conocidas o implementar ataques como LFI o RCE.

Listado de Directorio Público en el Navegador



- Se observan las carpetas 2024/ y 2025/, creadas automáticamente por WordPress para agrupar archivos subidos por año.
- El servidor revela estructura de carpetas y fechas de modificación, lo que puede facilitar ataques dirigidos como:
- **Acceso a archivos sensibles o confidenciales.**
- **Carga y ejecución de shells web si el entorno permite escritura.**

Listado Publico del tema activo



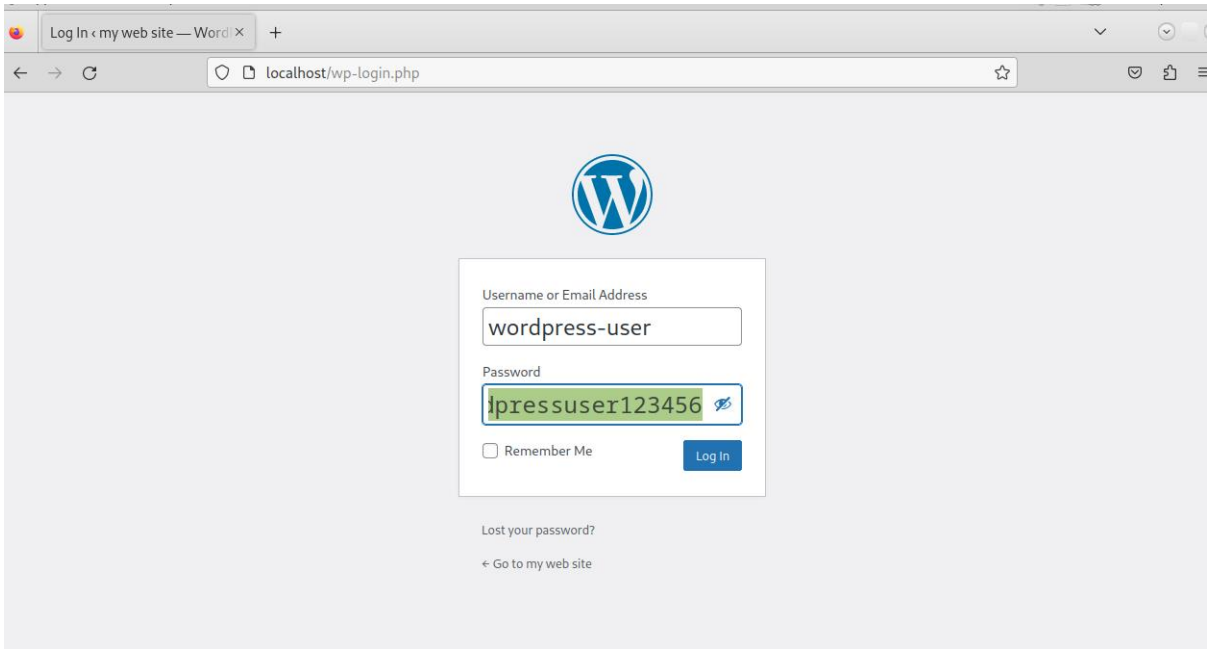
Cualquier atacante puede descargar el codigo fuente completo del tema y buscar vulnerabilidades personalizadas o errores de configuracion.

Si el archivo functions.php contiene funciones vulnerables como eval(), exec(), incluide () pueden ser explotadas si hay otro vector de entrada como una carga de archivos o XSS, esta exposicion tambien facilita la identificacion exacta de

la version del tema, lo que permite correlacionarla con CVEs publicos o exploits conocidos.

Credenciales almacenadas localmente en el navegador

Durante la fase de reconocimiento, se descubrió que el navegador tenía almacenadas las credenciales de acceso al panel de administración de WordPress (wp-login.php). Esto permitió el acceso sin necesidad de ejecutar un ataque por fuerza bruta.



- **Usuario:** wordpress-user
- **Contraseña:** wordpressuser123456
- Acceso completo al backend de WordPress.
- Posibilidad de subir código malicioso, crear cuentas, modificar contenido, instalar plugins y plugins maliciosos, etc.
- Riesgo de persistencia del atacante si no se revocan accesos.

Segunda Fase: Análisis de Vulnerabilidades

Puertos abiertos innecesarios:

El escaneo con Nmap reveló servicios abiertos que no están en uso, como:

- FTP (21)
- SSH (22)
- MySQL (3306)
- HTTP (80)

Impacto:

- Mayor superficie de ataque.
- Posible explotación de servicios olvidados.

Servidor FTP con acceso anónimo habilitado

El servidor FTP permite acceso sin autenticación (anonymous login), lo que expone archivos del sistema o del sitio.

Impacto:

- Filtración de archivos.
- Escalada de privilegios si permite escritura.

MySQL con usuario débil

Usuario root o wordpressuser con contraseñas simples (123456,root, etc.).

- **Impacto:**
- Acceso total a la base de datos.
- Exposición de usuarios, hashes, configuraciones internas.

Acceso SSH inseguro

El servicio SSH está expuesto y configurado con autenticación por contraseña. Posiblemente permite login con root.

Impacto:

- Riesgo de fuerza bruta.
- Si root está habilitado, escalada de privilegios inmediata.

Directorios web listables

Acceso a rutas como /wp-content/ o /uploads/ permite listar archivos si Options +Indexes está habilitado en Apache.

Impacto:

- Exposición de archivos sensibles o backup olvidados.
- Reconocimiento fácil de la estructura del sitio por parte de un atacante.

Permisos inseguros en wp-config.php

Descripción:
El archivo wp-config.php contiene credenciales de base de datos y configuración sensible. Si tiene permisos laxos (como 644), puede ser accedido por otros usuarios en el sistema.

Impacto:

- Exposición de la contraseña de la base de datos.
- Riesgo de acceso no autorizado al contenido del sitio.

Credenciales guardadas en el navegador

Descripción:
El formulario de login de WordPress (wp-login.php) autocompleta las credenciales wordpress-user : wordpressuser123456, permitiendo acceso directo al backend.

Impacto:

- Compromiso total del CMS.
- Posibilidad de instalación de malware, creación de usuarios, modificación de contenido.
- Persistencia del atacante sin necesidad de explotación activa.

Tercera Fase: Mitigación

Puertos abiertos innecesarios:

- Cerrar puertos innecesarios.
- Aplicar firewall (UFW, iptables) o reglas en router/firewall perimetral.

Servidor FTP con acceso anónimo habilitado

- Desactivar acceso anónimo.
- Usar SFTP si es necesario transferir archivos.
- Limitar acceso con permisos chroot.

MySQL con usuario débil

- Usar contraseñas robustas y únicas.
- Limitar acceso remoto.
- Aplicar cifrado en conexiones y backups.

Acceso SSH inseguro

- Deshabilitar root login: PermitRootLogin no
- Usar autenticación por clave pública.
- Cambiar el puerto por defecto y limitar IPs permitidas.

Directorios web listables

- Restringir acceso a directorios con .htaccess
- Revisión y limpieza de archivos expuestos
- **Deshabilitar el listado de directorios**
- Plugins de seguridad en WordPress

Permisos inseguros en wp-config.php

- Restringir permisos con chmod
- Establece permisos **mínimos necesarios**: chmod 600 wp-config.php
- **Verificar propiedad del archivo: chown www-data:www-data wp-config.php**
- Mover wp-config.php fuera del root del sitio web
- Configurar reglas de denegación en Apache/Nginx
- **Copias de seguridad cifradas**
- Monitorización de integridad con agentes com Wazuh

Credenciales guardadas en el navegador

- Prohibir guardar credenciales en navegadores de entornos críticos.
- Configurar políticas de expiración de sesiones.

- Cambiar credenciales por contraseñas robustas.
- Aplicar 2FA.

Plan de Respuesta a Incidentes y SGSI

Este documento presenta un Plan de Respuesta a Incidentes (PRI) y un Sistema de Gestión de Seguridad de la Información (SGSI), centrado exclusivamente en la máquina Debian comprometida analizada durante el proyecto. El enfoque está basado en las vulnerabilidades detectadas: configuración insegura de MySQL, acceso anónimo por FTP, acceso SSH débil, permisos inseguros en el archivo wp-config.php y directorios web listables.

1. Plan de Respuesta a Incidentes (PRI)

1.1 Objetivo

Establecer un conjunto de procedimientos específicos para responder ante incidentes derivados de las vulnerabilidades reales detectadas en el entorno comprometido.

1.2 Fases del Plan

1.2.1 Identificación

- Monitoreo de logs de Apache y sistema.
- Detección de accesos anónimos en FTP y conexiones SSH no autorizadas.
- Uso de Wazuh para alertas en tiempo real.
- Verificación de integridad en wp-config.php y archivos críticos del sistema.

1.2.2 Contención

- Deshabilitar acceso anónimo al FTP.
- Aislamiento temporal de servicios expuestos.
- Cambios urgentes en las credenciales del sistema.
- Reglas de firewall ASA para limitar el acceso externo a servicios vulnerables.

1.2.3 Erradicación

- Eliminación de usuarios sospechosos y shells.
- Reinstalación de servicios FTP y revisión de configuraciones SSH.

- Restricción de listado de directorios desde configuración del servidor web.
- Corrección de permisos en archivos sensibles.

1.2.4 Recuperación

- Restauración de backups verificados.
- Verificación de logs post-restauración.
- Pruebas de seguridad tras reactivación de los servicios.
- Seguimiento con monitoreo intensivo por 72 horas.

1.2.5 Lecciones Aprendidas

- Fortalecimiento de controles de acceso.
- Aplicación de hardening al sistema.
- Integración de escaneo periódico con OpenVAS o Lynis.
- Revisión y mejora del PRI en base al incidente.

1.2.6 RPO – Recovery Point Objective (Objetivo de Punto de Recuperación)

- **Servicio web y base de datos:** No más de 2 horas de pérdida de datos gracias a backups frecuentes.
- **Configuraciones críticas (wp-config.php, SSH, FTP):** Revisión cada 12 horas mediante copias cifradas.
- **Logs de seguridad y auditoría:** Sincronización cada 15 minutos con el SIEM.

Justificación: El entorno requiere respaldos regulares y automatizados siguiendo la política 3-2-1 para garantizar un punto de restauración reciente y consistente.

1.2.7 RTO – Recovery Time Objective (Objetivo de Tiempo de Recuperación)

- **Servicio Web (Apache + WordPress):** 4 horas
- **Base de Datos MySQL:** 2 horas
- **Acceso SSH seguro:** 3 horas
- **Servicio de monitoreo (Wazuh):** 6 horas
- **FTP (en transición a SFTP):** 6 horas (limitado a entornos internos)

Justificación: El impacto crítico de estos servicios sobre la disponibilidad y recuperación de datos requiere tiempos breves de restauración, priorizando el backend y el control administrativo del sistema.

1.2.8 MTTR – Mean Time to Recovery (Tiempo Medio de Recuperación)

- **Estimación general para Debian + Apache + MySQL:** 3 horas
- **Escenarios simulados con restauración de backups verificados:** 2h 45m en promedio
- **Recuperación tras detección de rootkits o shells maliciosos:** 4 horas

Justificación: Basado en pruebas controladas durante el ejercicio, incluyendo restauración y endurecimiento post-compromiso.

1.2.9 MTBF – Mean Time Between Failures (Tiempo Medio Entre Fallos)

- **Antes de mitigaciones:** Incidentes cada 15-30 días (por mala configuración, vulnerabilidades sin parches)
- **Después de mitigaciones (SGSI, hardening, monitorización activa): ≥ 90 días estimados**

Justificación: El fortalecimiento de políticas de seguridad, junto con Wazuh, hardening de servicios y eliminación de accesos inseguros, reduce significativamente la probabilidad de fallo recurrente.

2. Sistema de Gestión de Seguridad de la Información (SGSI)

2.1 Alcance

El SGSI se aplicará a la máquina Debian utilizada en el proyecto, incluyendo servicios WordPress, base de datos MySQL, FTP y SSH. Su objetivo es proteger los activos de información frente a accesos no autorizados, pérdida de datos y ataques.

2.2 Análisis de Riesgos

- Activos: servidor Debian, base de datos, archivos de configuración.
- Amenazas: explotación de contraseñas débiles, FTP abierto, acceso SSH inseguro.
- Vulnerabilidades: servicios mal configurados, permisos inadecuados.
- Riesgo: alto impacto por exposición a internet sin protección adecuada.

2.3 Políticas de Seguridad

- Política de contraseñas robustas y rotación periódica.
- Política de backups automáticos y cifrados.
- Prohibición de accesos anónimos a servicios.
- Control de acceso por usuarios y grupos.

2.4 Controles ISO Relevantes

- A.9 Control de accesos: SSH con autenticación por clave.
- A.10 Cifrado: aplicación de SSL en el servidor web.
- A.12 Seguridad operativa: revisión de logs y permisos.
- A.14 Seguridad de desarrollo: pruebas en entorno de staging antes de producción.

2.5 Plan de Acción

- Auditorías de configuración cada 3 meses.
- Simulacros de recuperación ante incidentes.
- Capacitación en ciberseguridad para el personal.
- Actualización semestral del SGSI.

Recomendaciones Generales

- Implementar **doble autenticación (2FA)** para todos los accesos administrativos.
- **Eliminar o sustituir** servicios inseguros como FTP por SFTP o SCP.
- Establecer una **política de actualizaciones periódicas** del sistema operativo y aplicaciones.

- Realizar **análisis de vulnerabilidades mensuales** con herramientas como OpenVAS o Lynis.
- **Documentar y versionar** configuraciones clave de servicios como Apache, MySQL y SSH.
- Automatizar la **revisión diaria de logs** de seguridad con herramientas SIEM como Wazuh.
- Crear un **entorno de pruebas (staging)** para validar configuraciones antes de producción.
- Capacitar periódicamente al personal en **ciberseguridad y manejo de incidentes**.
- Desarrollar procedimientos escritos y prácticos de respuesta ante incidentes para todo el equipo.
- Aplicar **principios de hardening** para reducir la superficie de ataque de la máquina.

Conclusión

La auditoría de seguridad sobre la máquina Debian comprometida permitió identificar una serie de vulnerabilidades críticas, entre ellas: acceso anónimo a FTP, credenciales débiles en MySQL, configuración insegura en SSH, permisos inadecuados en archivos sensibles como wp-config.php, y la exposición innecesaria de directorios web listables.

A partir de estas debilidades reales, se diseñó un **Plan de Respuesta a Incidentes (PRI)** basado en la guía del NIST SP 800-61, que proporciona un marco sólido para identificar, contener, erradicar y recuperar incidentes de seguridad. Paralelamente, se elaboró un **Sistema de Gestión de Seguridad de la Información (SGSI)** conforme a ISO 27001, incluyendo análisis de riesgos, políticas de seguridad y controles alineados con las mejores prácticas internacionales.

Este proyecto refleja la importancia de actuar **proactivamente ante amenazas informáticas**, no solo corrigiendo vulnerabilidades actuales, sino también **fortaleciendo las defensas a futuro** mediante políticas, procedimientos y cultura organizacional en ciberseguridad. La prevención, el monitoreo continuo y la formación del personal son claves para garantizar la integridad y continuidad operativa en entornos digitales expuestos.