

Proyecto de ciberseguridad

Auditoria de seguridad informatica - grupo 6

Autores:

Mateo Valderrama caliz

Juan Pablo Salazar rueda

Isabel Tobon Florez

Estefania Zapata Suarez

Mariana puerta Gutierrez

Talento tech

Medellín - 2025

Tabla de contenido

1. Introducción

2. Diseño, Implementación y Aseguramiento de un Entorno de Red Virtualizado con Pruebas de Ciberseguridad para una Empresa Simulada

3. VirtualBox

4. Importancia y Uso de VirtualBox, Kali Linux y Metasploit

5. Uso de Ubuntu 24.04.3 en el Proyecto

6. Importancia del Aprendizaje de Herramientas y Uso de FTP/FileZilla

7. Instalación y Configuración de MariaDB en Ubuntu

7.1 Actualizar los paquetes del sistema

7.2 Instalar MariaDB

7.3 Iniciar y habilitar el servicio

7.4 Entrar a la consola de MariaDB

7.5 Creación del usuario y permisos

7.6 Dar permisos al usuario para usar la base de datos

7.7 Actualizar los privilegios

7.8 Salir de MariaDB

7.9 Comando para verificar el estado de MariaDB

8. Introducción a la Instalación de Apache y PHP en Ubuntu

9. Configuración de la Red Corporativa

10. Instalar Apache y PHP

10.1 Reiniciar Apache

10.2 Probar PHP

10.3 Configurar la página PHP

11. Conectar PHP con MariaDB

12. Descripción del Comando PING

13. Descripción del Comando Nmap

14 Análisis de Reconocimiento con Nmap

14.1 Escaneo Avanzado

14.2 Comando utilizado

14.3 Puertos abiertos identificados

14.4 Identificación del sistema operativo

14.5 Distancia en la red

14.6 Traceroute

14.7 Vulnerabilidades identificadas

14.8 Conclusión del reconocimiento

15. Enumeración Web con Scripts NSE de Nmap

15.1 Objetivo

15.2 Comandos utilizados

15.3 Resultados relevantes obtenidos

15.4 Conclusión

16. Directriz de Políticas de Seguridad de la Información

16.1 Disposiciones Generales

- Finalidad

- Ámbito técnico

- Autoridad

16.2 Obligaciones Mandatorias por Rol (RBAC)

16.3 Medidas Técnicas Obligatorias

- SQL Injection Mitigation

- SSH Hardening

- Cookies seguras

- Backups MariaDB

16.4 Procesos Obligatorios (Workflows)

16.5 KPIs de Cumplimiento

16.6 Canales de Comunicación

16.7 Checklist de Implementación

16.8 Auditoría y Verificación

16.9 Sanciones

16.10 Vigencia

17. Práctica: Reconocimiento Web con Nmap + Análisis de Base de Datos

17.1 Escaneo inicial

17.2 Escaneo de puertos web

17.3 Escaneo agresivo

17.4 Escaneo con scripts NSE

17.5 Análisis del archivo db.php

18. ¿Qué es un Ataque de Fuerza Bruta con Hydra?

19. Ataque de Fuerza Bruta con Hydra sobre Servicio SSH

20. SQL Injection en LuzEnergia

20.1 Identificación inicial de servicios

20.2 Análisis del formulario de inicio de sesión

20.3 Prueba de SQL Injection

20.4 Conclusión

Mitigación de ataques

21. Manual de Políticas de Seguridad de la Información – LUZENERGIA

21.1 Introducción

21.2 Alcance

21.3 Políticas Generales de Seguridad

21.4 Política de Tratamiento de Datos Personales

21.5 Política de Habeas Data

21.6 Política de Autenticación y Control de Acceso

21.7 Política de Gestión de Incidentes

21.8 Política de Uso Aceptable

21.9 Política de Seguridad de Backups

21.10 Revisión y Actualización de Políticas

22. Conclusión

1. Introducción

En un entorno empresarial cada vez más digitalizado, la seguridad de la información se ha convertido en un componente crítico para la continuidad operativa y la protección de los activos tecnológicos. Las organizaciones dependen de infraestructuras complejas que integran servidores, aplicaciones, redes internas, servicios web y sistemas de almacenamiento, los cuales deben ser evaluados y fortalecidos continuamente para mitigar riesgos de ciberataques.

El presente proyecto tiene como objetivo diseñar, implementar y auditar un entorno corporativo virtualizado mediante herramientas como VirtualBox, Ubuntu Server, Kali Linux y Metasploitable, simulando las condiciones reales de una empresa moderna. A través de esta infraestructura se realizan pruebas de ciberseguridad, incluyendo análisis de puertos, reconocimiento de servicios, escaneo de vulnerabilidades, pruebas de fuerza bruta, y explotación de fallas como inyección SQL.

El propósito final es comprender de manera práctica cómo se construye, asegura y evalúa una infraestructura tecnológica siguiendo lineamientos profesionales basados en las buenas prácticas de la industria, normas como ISO/IEC 27001 y la legislación de protección de datos vigente.

2. Diseño, Implementación y Aseguramiento de un Entorno de Red Virtualizado con Pruebas de Ciberseguridad para una Empresa Simulada

En la actualidad, las organizaciones dependen cada vez más de infraestructuras tecnológicas para el almacenamiento, procesamiento y protección de su información. Esta creciente digitalización ha incrementado la necesidad de implementar entornos seguros que permitan garantizar la integridad, disponibilidad y confidencialidad de los datos empresariales. En este contexto, el presente proyecto tiene como propósito diseñar, configurar y asegurar un entorno de red corporativa virtualizado, aplicando principios fundamentales de ciberseguridad y pruebas de penetración ética.

Para la simulación del entorno empresarial, se utiliza la plataforma VirtualBox, dentro de la cual se despliega un conjunto de máquinas virtuales que representan los servicios y roles típicos de una organización. Entre estas máquinas se incluyen:

Ubuntu Server, destinado a funciones como servidor web y servidor de bases de datos

Kali Linux, empleado como equipo de auditoría y pruebas de penetración.

Metasploitable, una máquina vulnerable diseñada para prácticas de explotación y detección de vulnerabilidades.

Además de los clientes simulados y otros servidores necesarios para la arquitectura propuesta.

El proyecto contempla el diseño de una red corporativa segmentada en áreas como Administración, IT y Operaciones, utilizando VLAN para aislar y proteger el tráfico entre departamentos. También se implementan políticas de acceso, controles de seguridad y configuraciones de red orientadas a reducir riesgos y fortalecer la infraestructura.

Posteriormente, se realizan pruebas de seguridad utilizando herramientas especializadas como Nmap, Burp Suite y Metasploit Framework, con el fin de identificar vulnerabilidades presentes en el entorno. Los resultados obtenidos

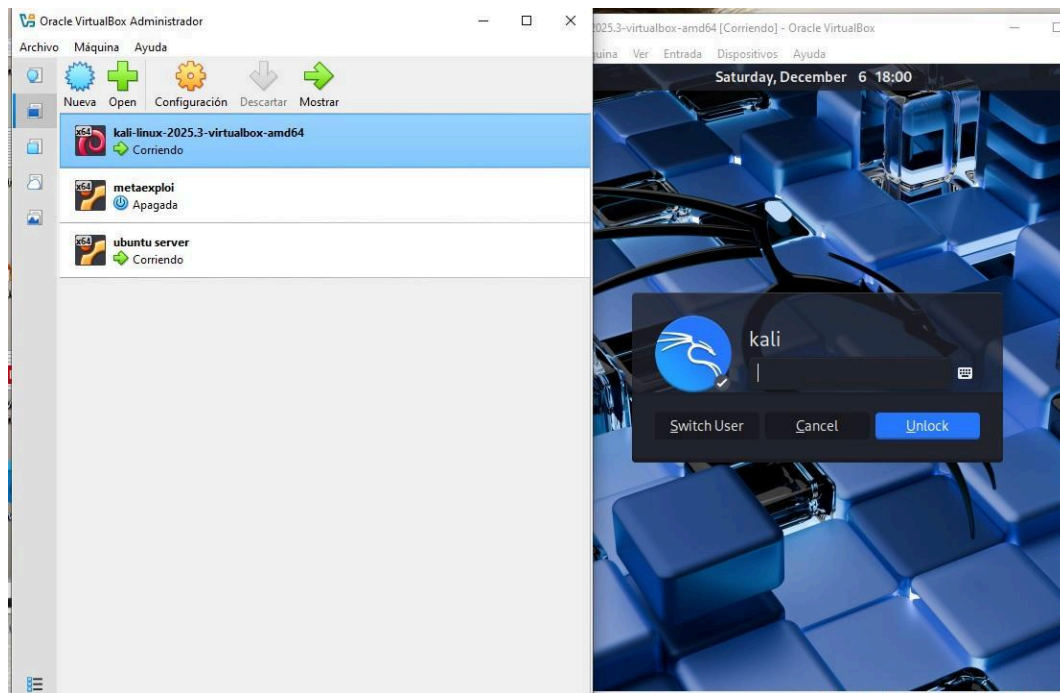
permiten evaluar el nivel de exposición, aplicar medidas de mitigación y fortalecer la postura de seguridad general del sistema.

Finalmente, se desarrollan políticas de seguridad, mecanismos de monitoreo y estrategias de respuesta a incidentes que refuerzan la protección de la información y la continuidad operativa. Este proyecto integra conocimientos teóricos y prácticos, ofreciendo una visión integral sobre cómo se diseña, implementa y asegura un entorno tecnológico real en el contexto empresarial moderno.

3. VirtualBox

VirtualBox es una herramienta de virtualización de código abierto que permite ejecutar múltiples sistemas operativos de manera simultánea en un solo equipo físico. Esta plataforma es ampliamente utilizada para crear entornos de laboratorio y pruebas, ya que permite aislar sistemas operativos invitados dentro de máquinas virtuales, sin afectar al sistema operativo principal.

En el contexto de este proyecto, VirtualBox se utiliza para desplegar diversas máquinas virtuales que simulan los distintos roles dentro de una red corporativa, incluyendo servidores de aplicaciones, bases de datos y clientes. Además, se integra con herramientas de ciberseguridad como Kali Linux y Metasploit para realizar pruebas de penetración y evaluar la seguridad del entorno virtual. Gracias a VirtualBox, se puede experimentar con la configuración de redes, segmentación mediante VLANs, políticas de acceso y protección de datos, garantizando un laboratorio seguro y controlado para el desarrollo del proyecto.

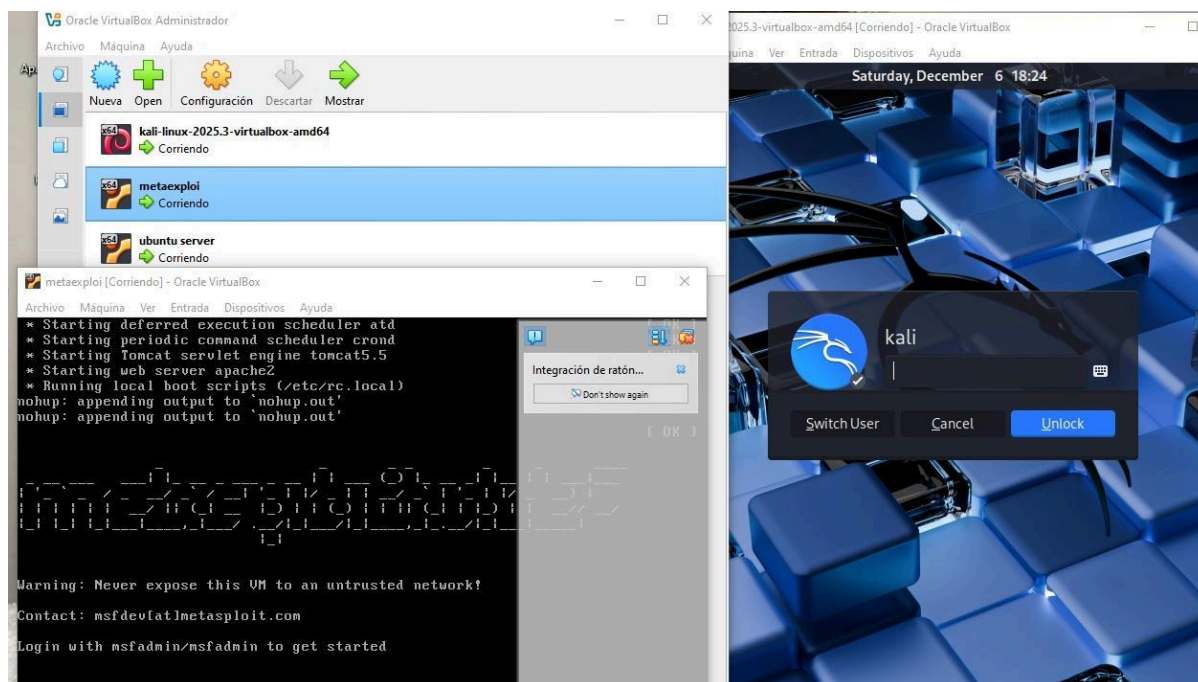


4. Importancia y Uso de VirtualBox, Kali Linux y Metasploit

Para la realización del proyecto, se empleó VirtualBox, una herramienta de virtualización que permite crear múltiples máquinas virtuales en un solo equipo físico. Esto facilitó simular un entorno de red corporativo completo, aislando los distintos sistemas y servicios sin afectar el sistema principal.

Dentro de VirtualBox, se utilizaron Ubuntu como servidor de aplicaciones y base de datos, y Kali Linux como plataforma de pruebas de seguridad. En Kali Linux se implementó Metasploit, una herramienta de pentesting que permitió identificar vulnerabilidades y evaluar la seguridad del entorno virtual.

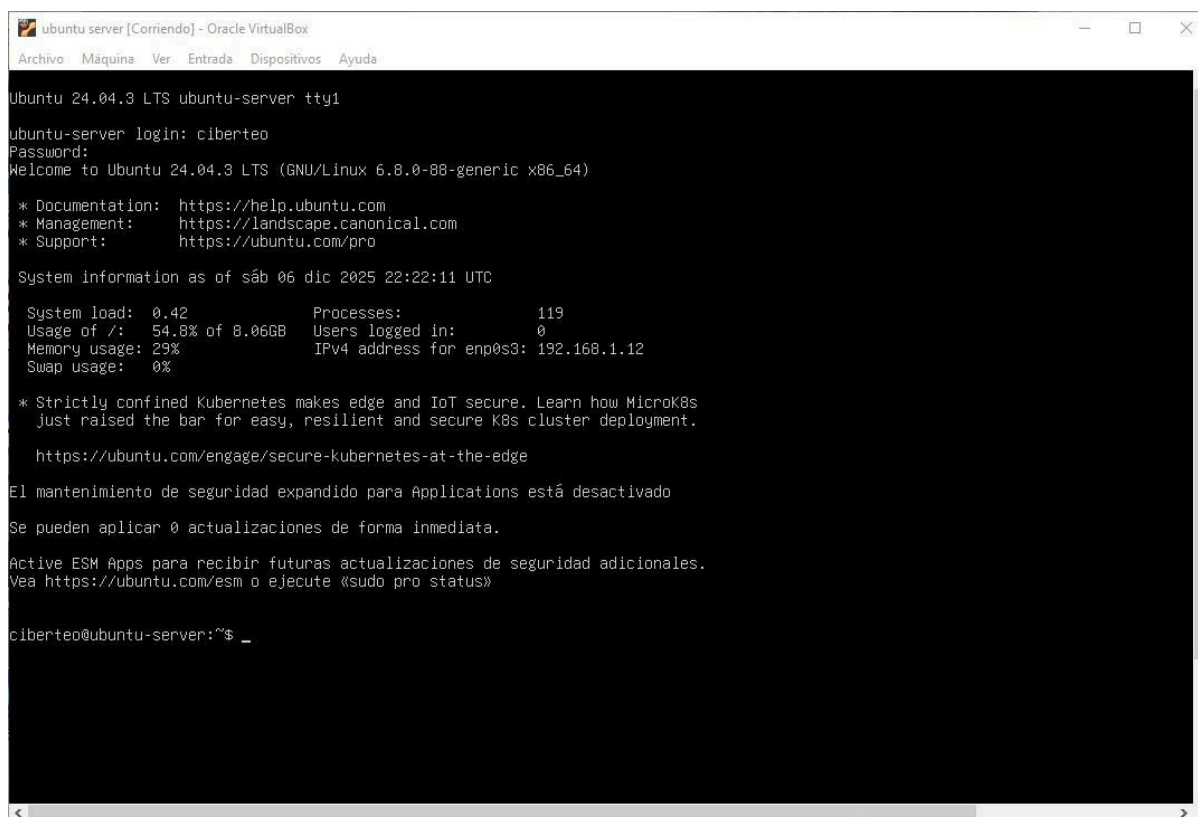
El uso conjunto de estas herramientas proporcionó un entorno seguro y controlado para experimentar con configuraciones de red, pruebas de seguridad y administración de sistemas, fortaleciendo la comprensión práctica de conceptos clave en ciberseguridad y virtualización.



5. Uso de Ubuntu 24.04.3 en el Proyecto

Para el desarrollo del proyecto, se utilizó Ubuntu 24.04.3, una distribución de Linux estable y confiable, ideal para servidores y entornos de pruebas. Ubuntu permitió configurar el servidor de aplicaciones y la base de datos de manera segura, ejecutando servicios como Apache, MySQL/MariaDB y PHP, necesarios para el funcionamiento de la página web desarrollada.

Su estabilidad y soporte a largo plazo garantizan un entorno confiable, permitiendo implementar medidas de seguridad, administrar usuarios y controlar el acceso a los recursos de manera eficiente. Además, su compatibilidad con herramientas de administración y monitoreo facilita la práctica de técnicas de ciberseguridad y administración de sistemas dentro del proyecto.



```
ubuntu server [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda

Ubuntu 24.04.3 LTS ubuntu-server tty1
ubuntu-server login: ciberteo
Password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-88-generic x86_64)

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

System information as of sáb 06 dic 2025 22:22:11 UTC

System load:  0.42          Processes:           119
Usage of /:    54.8% of 8.06GB   Users logged in:    0
Memory usage: 29%          IPv4 address for enp0s3: 192.168.1.12
Swap usage:   0%

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster deployment.
   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

El mantenimiento de seguridad expandido para Applications está desactivado
Se pueden aplicar 0 actualizaciones de forma inmediata.

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

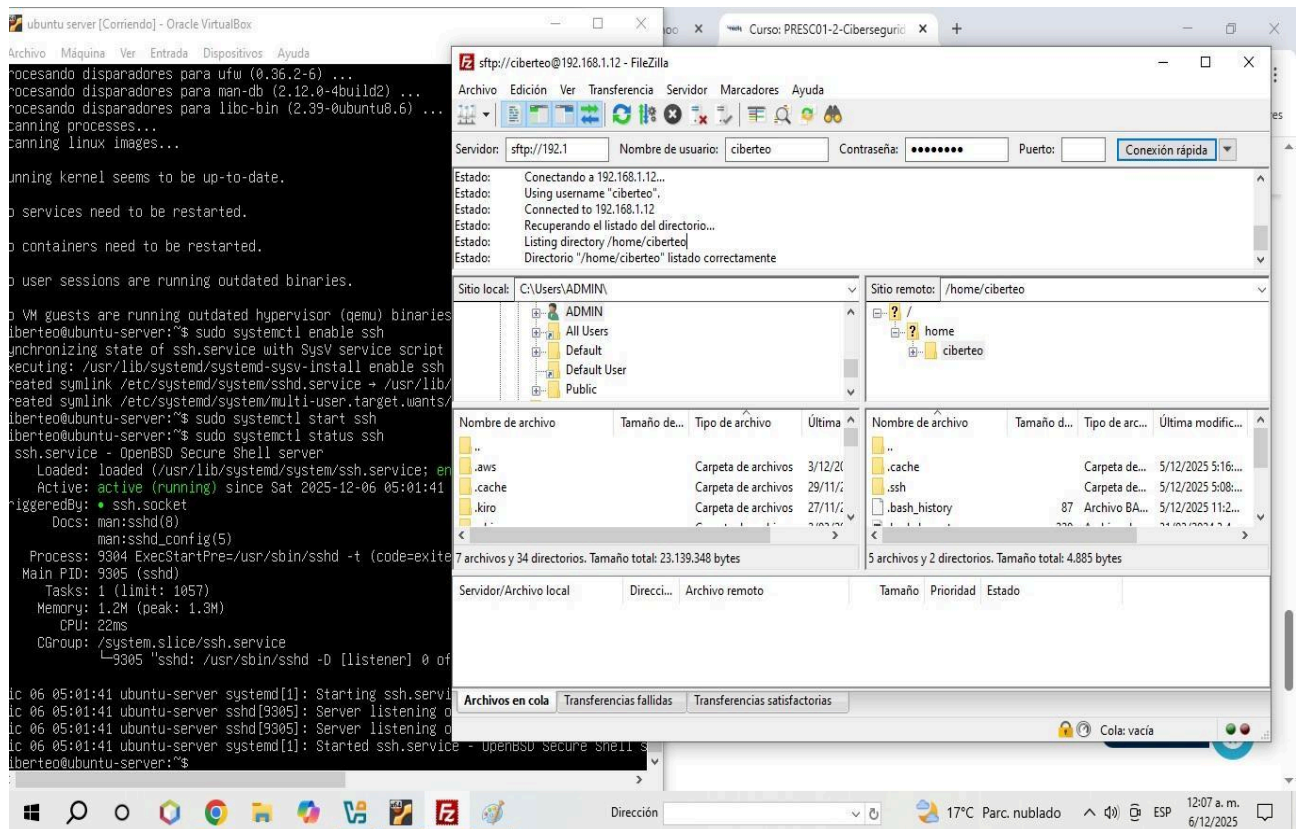
ciberteo@ubuntu-server:~$ _
```

6. Importancia del Aprendizaje de Herramientas y Uso de FTP/FileZilla

Durante el proyecto se adquirieron conocimientos prácticos sobre herramientas de ciberseguridad y administración de sistemas, fundamentales para comprender la gestión de redes y la protección de datos.

El protocolo FTP (**protocolo de transporte de archivos**) permitió transferir archivos desde Windows con XAMPP hacia el servidor Ubuntu, facilitando la puesta en marcha de la página web. Para ello se utilizó FileZilla, que ofrece una interfaz sencilla para subir y descargar archivos de manera eficiente y segura.

El uso de estas herramientas reforzó habilidades prácticas en administración de servidores, transferencia de datos y manejo de protocolos de red, competencias esenciales para la gestión de entornos corporativos y proyectos de ciberseguridad.



7. Instalación y configuración de MariaDB en Ubuntu

```
ubuntu server [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
just raised the bar for easy, resilient and secure K8s cluster deployment.
https://ubuntu.com/engage/secure-kubernetes-at-the-edge
El mantenimiento de seguridad expandido para Applications está desactivado
Se pueden aplicar 0 actualizaciones de forma inmediata.
Active ESM Apps para recibir futuras actualizaciones de seguridad adicionales.
Vea https://ubuntu.com/esm o ejecute «sudo pro status»

ciberteo@ubuntu-server:~$ sudo mariabd
[sudo] password for ciberteo:
sudo: mariabd: command not found
ciberteo@ubuntu-server:~$ sudo mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 40
Server version: 10.11.13-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> ^C
MariaDB [(none)]> show database
-> proyecto_ciber
-> ^C
MariaDB [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| proyecto_ciber |
| sys |
+-----+
5 rows in set (0.003 sec)

MariaDB [(none)]> _
```

- **Actualizar los paquetes del sistema**

```
sudo apt update
```

Descripción: Actualiza la lista de paquetes disponibles en Ubuntu para asegurar que todo lo que se instale esté en su última versión.

- **Instalar MariaDB**

```
sudo apt install mariadb-server -y
```

Descripción: Descarga e instala el servidor de MariaDB en el sistema.

- **Iniciar y habilitar el servicio de MariaDB**

```
sudo systemctl start mariadb
sudo systemctl enable mariadb
```

Descripción:

start: inicia el servicio MariaDB.

enable: hace que MariaDB se inicie automáticamente cada vez que arranque Ubuntu.

- **Entrar a la consola de MariaDB**

```
sudo mariadb
```

Descripción: Abre la consola del motor MariaDB como usuario administrador del sistema.

- **Creación del usuario y permisos**

```
CREATE USER "luzuser"@"localhost" IDENTIFIED BY "12345";
```

Descripción: Crea un usuario dentro de MariaDB (no del sistema) que se llamará luzuser y solo podrá conectarse desde localhost usando la contraseña 12345.

- **Dar permisos al usuario para usar la base de datos**

```
GRANT ALL PRIVILEGES ON proyecto_ciber.* TO "luzuser"@"localhost";
```

Descripción: Otorga todos los permisos (leer, escribir, modificar) sobre la base de datos proyecto ciber al usuario recién creado.

- **Actualizar los privilegios**

```
FLUSH PRIVILEGES;
```

Descripción: Recarga los privilegios para aplicar los cambios de inmediato.

- **Salir de MariaDB**

```
EXIT;
```

Descripción: Cierra la consola de MariaDB.

- **Comando para verificar si MariaDB está bien instalado**

```
systemctl status mariadb
```

Descripción: Muestra el estado del servicio para confirmar que está corriendo correctamente.

8. Introducción a la instalación de Apache y PHP en Ubuntu

Para poner en funcionamiento un servidor web en Ubuntu, es fundamental instalar Apache, que se encarga de recibir y gestionar las solicitudes HTTP de los usuarios, y PHP, que permite ejecutar scripts del lado del servidor para generar contenido web dinámico.

El proceso de instalación en Ubuntu 24.04 consiste en utilizar comandos de terminal para actualizar el sistema, instalar Apache y PHP, y verificar que ambos servicios estén activos y funcionando correctamente. Esta configuración permite crear un entorno de desarrollo local en el que se pueden alojar páginas web, gestionar bases de datos y probar aplicaciones dinámicas de manera segura y controlada.

La instalación de Apache y PHP en Ubuntu es un paso esencial para cualquier proyecto web, ya que proporciona la infraestructura necesaria para el funcionamiento de sitios web y aplicaciones que interactúan con bases de datos, como la página de LuzEnergia.

9. Configuración de la red corporativa

En este proyecto , usamos un total de 5 máquinas virtuales , cada una con un objetivo diferente dentro de nuestro entorno corporativo.

Tenemos 4 máquinas que simulan un entorno en base linux y una con base windows.

La red corporativa está segmentada mediante VLANs para garantizar seguridad, control de acceso y un flujo de información ordenado entre las diferentes áreas de la organización. Toda la infraestructura se encuentra protegida por un **firewall perimetral**, que filtra el tráfico proveniente de Internet antes de ingresarlo a la red interna.

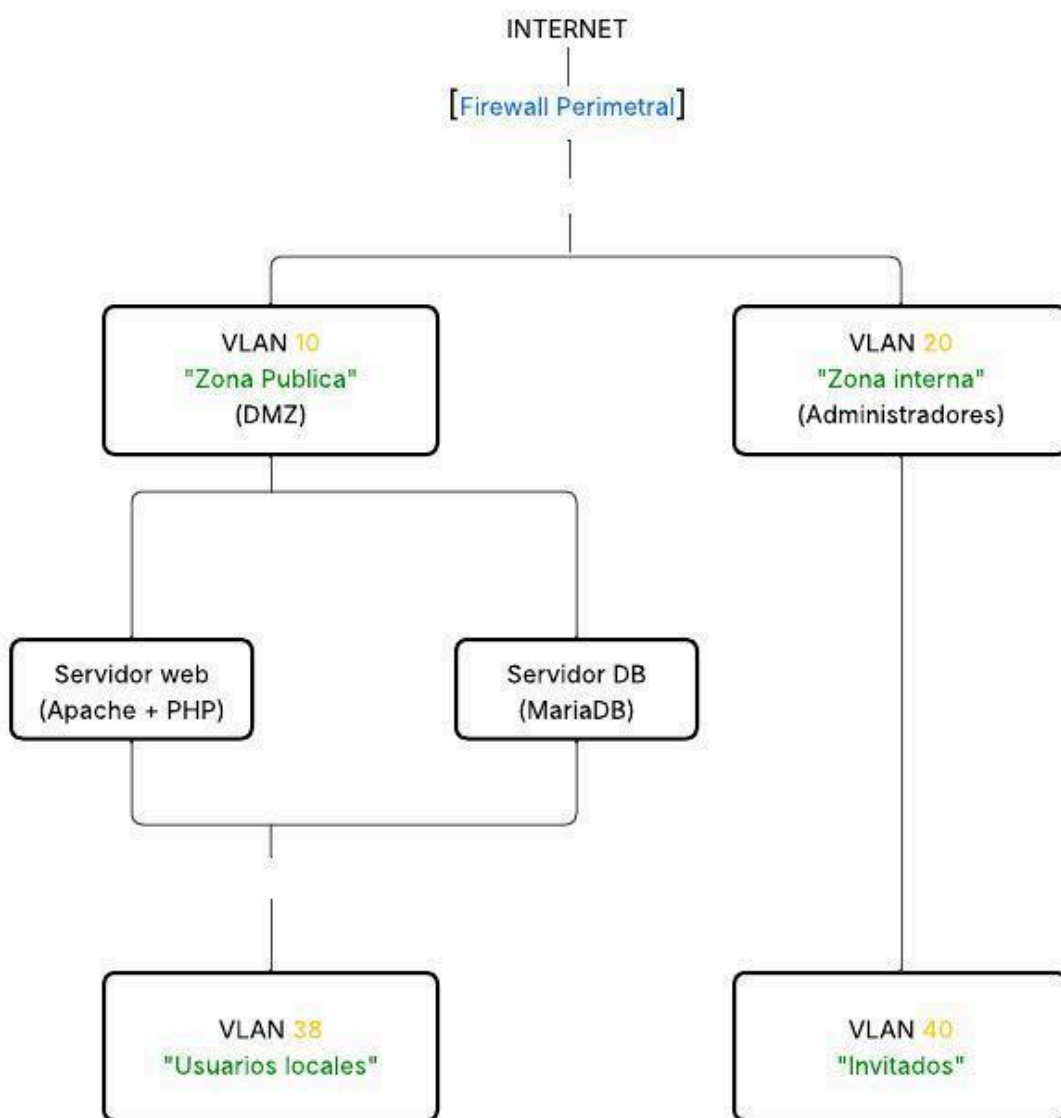
La **VLAN 10 (Zona Pública – DMZ)** aloja el servidor web (Apache + PHP), el cual es accesible desde Internet. Este servidor está aislado del resto de la red para mitigar riesgos en caso de ataques externos.

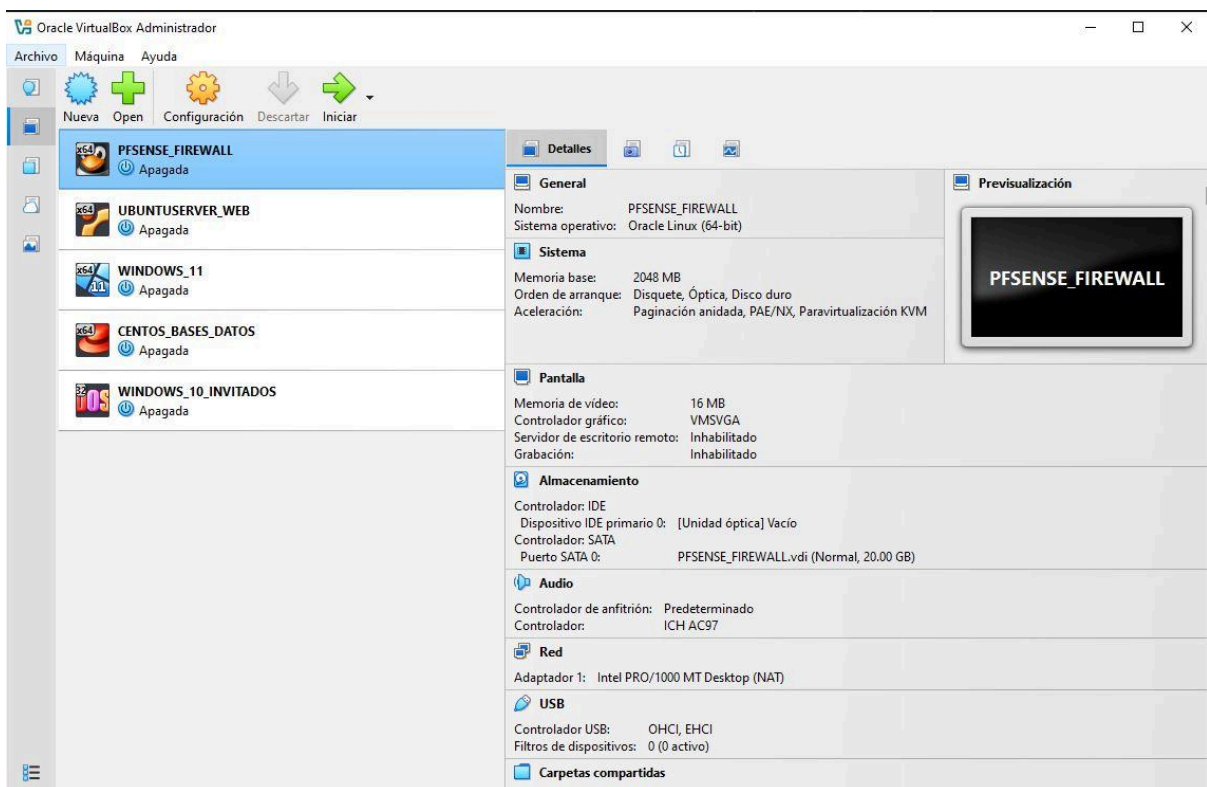
La **VLAN 20 (Zona Interna – Administradores)** es exclusiva para el personal autorizado encargado de la administración de sistemas y bases de datos. Desde esta red se gestiona tanto la infraestructura como los servidores críticos.

El servidor de base de datos (MariaDB) se encuentra protegido en un segmento interno y únicamente puede ser accedido desde la DMZ y desde la zona de administradores, reduciendo la exposición y reforzando el principio de mínimo privilegio.

Por su parte, la **VLAN 38 (Usuarios Locales)** está destinada a estaciones de trabajo internas que requieren acceso a servicios corporativos, mientras que la **VLAN 40 (Invitados)** ofrece conectividad limitada y aislada para dispositivos externos, manteniéndolos separados de los recursos sensibles.

Gracias a esta segmentación, la red mantiene un tráfico controlado, minimiza superficies de ataque y asegura que cada grupo de usuarios solo acceda a los recursos estrictamente necesarios.





¿Por qué elegirnos?

10. Instalar Apache y PHP

```
sudo apt update  
sudo apt install apache2 php libapache2-mod-php php-mysql
```

Descripción:

apache2 → servidor web.

php → lenguaje de programación para tus páginas dinámicas.

libapache2-mod-php → módulo que permite que Apache interprete PHP.

php-mysql → extensión que permite que PHP se conecte a MySQL/MariaDB.

- **Reiniciar Apache para que reconozca PHP**

```
sudo systemctl restart apache2
```

Descripción: Aplica los cambios y activa PHP con Apache.

- **Probar PHP**

Crear un archivo de prueba:

```
sudo nano /var/www/html/test.php
```

Pegar dentro:

```
<?php  
phpinfo();  
?>
```

Guardar y cerrar (Ctrl + O, Enter, Ctrl + X).

Abrir en el navegador:

```
http://localhost/test.php
```

Descripción: Muestra información de PHP instalada y configurada. Si ves esta página, PHP funciona correctamente.

- **Configurar tu página PHP**

Copia todos los archivos de tu proyecto a:

```
/var/www/html/
```

Asegúrate de que los permisos sean correctos:

```
sudo chown -R www-data:www-data /var/www/html/  
sudo chmod -R 755 /var/www/html/
```

Descripción:

www-data → usuario y grupo de Apache.

Esto permite que Apache lea y ejecute los archivos PHP.

11. Conectar PHP con MariaDB

Edita tu archivo db.php para usar la base de datos correcta:

```
<?php
$host = "localhost";
$user = "root"; // o el usuario que creaste
$pass = "";     // la contraseña que definiste
$dbname = "proyecto_ciber";

$conn = mysqli_connect($host, $user, $pass, $dbname);

if (!$conn) {
    die("Error de conexión: " . mysqli_connect_error());
}
?>
```

Prueba la conexión creando un archivo test db.php y usa:

```
<?php
include 'db.php';
echo "Conexión exitosa!";
?>
```

12.Descripción del comando PING

El comando ping es una herramienta básica de diagnóstico de red que permite verificar si un dispositivo está activo y accesible dentro de una red. Su funcionamiento se basa en el envío de paquetes ICMP (Internet Control Message Protocol) al equipo de destino. Si el equipo responde, significa que la comunicación entre ambos dispositivos es exitosa.

```
Session  Actions  Edit  View  Help
zsh: corrupt history file /home/kali/.zsh_history
(kali@kali)-[~]
$ ping 192.168.1.12
PING 192.168.1.12 (192.168.1.12) 56(84) bytes of data.
64 bytes from 192.168.1.12: icmp_seq=1 ttl=255 time=0.913 ms
64 bytes from 192.168.1.12: icmp_seq=2 ttl=255 time=0.812 ms
64 bytes from 192.168.1.12: icmp_seq=3 ttl=255 time=0.850 ms
64 bytes from 192.168.1.12: icmp_seq=4 ttl=255 time=1.29 ms
64 bytes from 192.168.1.12: icmp_seq=5 ttl=255 time=0.867 ms
64 bytes from 192.168.1.12: icmp_seq=6 ttl=255 time=0.828 ms
64 bytes from 192.168.1.12: icmp_seq=7 ttl=255 time=0.862 ms
64 bytes from 192.168.1.12: icmp_seq=8 ttl=255 time=0.776 ms
64 bytes from 192.168.1.12: icmp_seq=9 ttl=255 time=0.816 ms
64 bytes from 192.168.1.12: icmp_seq=10 ttl=255 time=0.768 ms
^C
— 192.168.1.12 ping statistics —
10 packets transmitted, 10 received, 0% packet loss, time 9114ms
rtt min/avg/max/mdev = 0.768/0.878/1.293/0.144 ms
(kali@kali)-[~]
$
```

Cuando ejecutamos un ping simple, como:

ping 192.168.1.12 este ping puede ser eterno se pausa con control c

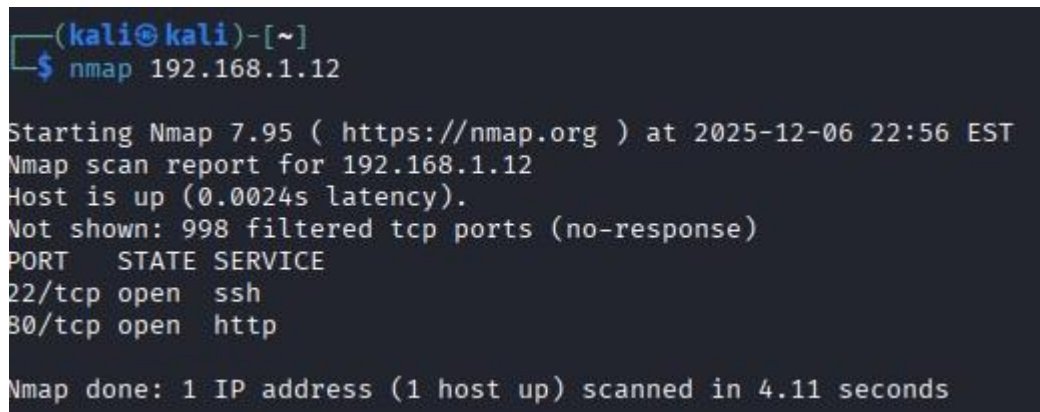
o se puede usar un ping -c 4 192.168.1.12

4 pings y se detiene automáticamente.

13. Descripción del comando Nmap

Nmap (Network Mapper) es una herramienta avanzada utilizada para analizar y mapear redes, permitiendo identificar hosts activos, servicios disponibles y puertos abiertos en un dispositivo. Es fundamental en pruebas de seguridad porque muestra la superficie de ataque que un atacante podría aprovechar.

Cuando ejecutamos:



```
(kali@kali)-[~]  
$ nmap 192.168.1.12  
  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-06 22:56 EST  
Nmap scan report for 192.168.1.12  
Host is up (0.0024s latency).  
Not shown: 998 filtered tcp ports (no-response)  
PORT      STATE SERVICE  
22/tcp    open  ssh  
80/tcp    open  http  
  
Nmap done: 1 IP address (1 host up) scanned in 4.11 seconds
```

En este nmap se identificó que el host está activo

- La IP 192.168.1.12 responde correctamente (latencia baja de 0.0024s).
 - ✓ Tiene 2 puertos abiertos
 - 22/tcp – Servicio SSH
Este puerto permite acceso remoto seguro al servidor.
Significa que el servidor Ubuntu tiene OpenSSH habilitado.
Si no tiene un firewall configurado, podría ser un riesgo de ataque por fuerza bruta.
 - 80/tcp – Servicio HTTP
Es el puerto donde corre Apache.
Confirma que la página web está disponible en este puerto.
- ✓ 998 puertos filtrados, esto significa que la mayoría de los puertos NO están accesibles.

Es bueno para la seguridad: solo 22 y 80 están expuestos.

Ahora con un comando nmap mas avanzado

sudo nmap -p- -sS -sV -A -O 192.168.1.12

¿Qué hace ese comando?

Opción	Explicación
--------	-------------

- | | |
|-----|--------------------------------------------------------------|
| -p- | Escanea los 65.535 puertos completos. |
| -sS | Escaneo stealth SYN (rápido y discreto). |
| -sV | Detecta la versión de cada servicio. |
| -A | Habilita: traceroute + scripts NSE + detección de servicios. |
| -O | Detecta el sistema operativo. |
- 192.168.1.12 = IP objetivo.

```

Nmap scan report for 192.168.1.12
Host is up (0.00035s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.14 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   256 bd:97:25:ac:ed:c7:53:6d:af:65:17:0f:08:cd:14:88 (ECDSA)
|_  256 9a:22:eb:64:0a:14:50:75:8c:5b:4f:e7:66:7f:64:ae (ED25519)
80/tcp    open  http      Apache httpd 2.4.58 ((Ubuntu))
|_ http-title: LuzEnergia - Empresa de Energ\xC3\xADa El\xC3\xA9ctrica
|_ http-server-header: Apache/2.4.58 (Ubuntu)
|_ http-cookie-flags:
|   /:
|   PHPSESSID:
|_      httponly flag not set
Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
Device type: bridge|VoIP adapter|general purpose
Running (JUST GUESSING): Oracle Virtualbox (98%), Slirp (98%), AT&T embedded
(95%), QEMU (94%)
OS CPE: cpe:/o:oracle:virtualbox cpe:/a:danny_gasparovski:slirp cpe:/a:qemu:
qemu
Aggressive OS guesses: Oracle Virtualbox Slirp NAT bridge (98%), AT&T BGW210
voice gateway (95%), QEMU user mode network gateway (94%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 80/tcp)
HOP RTT      ADDRESS
1   0.32 ms  192.168.1.12

```

14. Análisis de Reconocimiento con Nmap

Escaneo Avanzado

Con el objetivo de identificar servicios, puertos abiertos y posibles vectores de ataque en el servidor Ubuntu 24.04.3, se realizó un análisis avanzado utilizando la herramienta Nmap desde la máquina Kali Linux.

Este procedimiento corresponde a la fase de reconocimiento activo dentro de un proceso de auditoría o pentesting.

2. Comando Utilizado

El escaneo se ejecutó con el siguiente comando:

```
sudo nmap -p- -sS -sV -A -O 192.168.1.12
```

Descripción de cada parámetro

sudo: ejecuta Nmap con privilegios para utilizar técnicas avanzadas de escaneo.

-p-: escanea todos los puertos TCP (1–65535).

-sS: realiza un escaneo SYN Stealth, sigiloso y más rápido.

-sV: identifica versiones de los servicios que están corriendo en los puertos abiertos.

- A: habilita detección agresiva → sistema operativo, servicios, scripts NSE.
 - O: intenta identificar el sistema operativo del host objetivo.
- 192.168.1.12: dirección IP del servidor Ubuntu dentro de VirtualBox.

3. Resultados del Escaneo

3.1 Puertos Abiertos Identificados

El escaneo reveló solo dos puertos abiertos, lo cual muestra un entorno relativamente seguro, pero con información relevante para análisis posterior.

- ♦ Puerto 22 — Servicio SSH

22/tcp open ssh OpenSSH 9.6p1 Ubuntu

Interpretación:

El servidor permite conexiones remotas mediante SSH.

La versión detectada (OpenSSH 9.6p1) indica un entorno actualizado.

Si el usuario o contraseña es débil, este puerto podría ser blanco de ataques de fuerza bruta (Hydra, Medusa, etc.).

Vulnerabilidad de acceso remoto por exposición del puerto SSH.

- ♦ Puerto 80 — Servidor Web Apache

80/tcp open http Apache httpd 2.4.58 ((Ubuntu))

Interpretación:

El servidor web está en funcionamiento mediante Apache 2.4.58.

El sitio detectado es “LuzEnergia”, correspondiente al proyecto web instalado.

Se encontró una cookie PHP sin la bandera de seguridad httponly.

PHPSESSID → httponly flag not set

Vulnerabilidad de robo de sesión (session hijacking) en caso de existir fallos XSS.

3.2 Identificación del Sistema Operativo

El análisis agresivo logró identificar que el host está ejecutándose dentro de un entorno virtual: Running: Oracle VirtualBox (98%)

Interpretación:

Nmap reconoció que el servidor Ubuntu está virtualizado en VirtualBox, lo cual coincide con la configuración del proyecto.

3.3 Distancia en la Red

Network Distance: 1 hoP

Interpretación:

La máquina Kali está conectada directamente al servidor Ubuntu.

Confirma que ambas máquinas están en la misma red NAT/Bridge de VirtualBox.

3.4 Trazado de Ruta (Traceroute)

HOP 1 → 192.168.1.12

Interpretación:

El servidor está accesible sin intermediarios, lo cual es ideal para pruebas de penetración.

4. Vulnerabilidades Identificadas

1. Exposición del puerto SSH (22)

El servicio SSH está accesible desde toda la red.

Posible ataque de fuerza bruta si las credenciales son débiles.

2. Cookie PHP insegura

Falta la bandera httponly, permitiendo potencial robo de sesión.

Puede ser explotado junto con un ataque XSS.

3. Divulgación de versión de Apache

El servidor expone su versión (Apache 2.4.58).

Permite a un atacante buscar exploits específicos.

5. Conclusión del Reconocimiento

El escaneo inicial mediante Nmap permitió identificar los servicios principales del servidor Ubuntu, validar que la máquina se encuentra correctamente expuesta en la red y encontrar vulnerabilidades útiles para las siguientes fases del proyecto, especialmente en análisis web, enumeración y pruebas de intrusión.

Este reconocimiento constituye una base fundamental para continuar con ataques más avanzados como:

Enumeración HTTP

Fuerza bruta a SSH

Análisis de directorios ocultos

Pruebas XSS y SQL Injection

15. Enumeración Web con Scripts NSE de Nmap

La enumeración web es una fase fundamental dentro del análisis de vulnerabilidades, ya que permite identificar servicios, directorios, configuraciones y posibles fallos en una aplicación expuesta a la red. Para este proceso se utilizaron los Nmap Scripting Engine (NSE), un conjunto de scripts avanzados que automatizan tareas de reconocimiento y análisis en

servidores web. Estos scripts permiten obtener información detallada sobre la estructura del sitio, cabeceras HTTP, tecnologías utilizadas y posibles puntos débiles del sistema.

Objetivo

Realizar un reconocimiento avanzado del servidor web alojado en Ubuntu, utilizando scripts NSE de Nmap con el fin de detectar:

Directorios expuestos.

Cabeceras HTTP inseguras.

Configuraciones incorrectas del servidor.

Posibles vectores de ataque web.

Este análisis apoya la documentación del proyecto y evidencia prácticas reales de pentesting.

```
msf auxiliary(scanner/http/http_header) > set RHOSTS 192.168.1.12
RHOSTS => 192.168.1.12
msf auxiliary(scanner/http/http_header) > run
[+] 192.168.1.12:80      : CACHE-CONTROL: no-store, no-cache, must-revalidate
[+] 192.168.1.12:80      : CONTENT-TYPE: text/html; charset=UTF-8
[+] 192.168.1.12:80      : SERVER: Apache/2.4.58 (Ubuntu)
[+] 192.168.1.12:80      : SET-COOKIE: PHPSESSID=7hb64nq7ejjdkg535bp
todcdgv; path=/
[+] 192.168.1.12:80      : detected 4 headers
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(scanner/http/http_header) > Interrupt: use the 'exit'
command to quit
msf auxiliary(scanner/http/http_header) > use auxiliary/scanner/http
/dir_scanner
msf auxiliary(scanner/http/dir_scanner) > set RHOSTS 192.168.1.12
RHOSTS => 192.168.1.12
msf auxiliary(scanner/http/dir_scanner) > set PATH /
PATH => /
msf auxiliary(scanner/http/dir_scanner) > run
[*] Detecting error code
[*] Using code '404' as not found for 192.168.1.12
[+] Found http://192.168.1.12:80/config/ 200 (192.168.1.12)
[+] Found http://192.168.1.12:80/css/ 200 (192.168.1.12)
[+] Found http://192.168.1.12:80/database/ 200 (192.168.1.12)
[+] Found http://192.168.1.12:80/icons/ 403 (192.168.1.12)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Comandos Utilizados

1. Escaneo de cabeceras HTTP

Este script recupera todas las cabeceras retornadas por el servidor web.

Sirve para identificar configuraciones inseguras como faltante de HttpOnly, falta de CSP, cookies sin protección, etc.

nmap --script http-headers -p80 192.168.1.12

2. Escaneo para detectar archivos y directorios web

Este script intenta descubrir carpetas o archivos que existen en el servidor.

Permite detectar directorios expuestos como /config, /css, /database, etc.

nmap --script http-enum -p80 192.168.1.12

3. Escaneo de configuración HTTP (método OPTIONS)

Permite identificar qué métodos HTTP están habilitados:

(GET, POST, PUT, DELETE, OPTIONS...).

Si están activos métodos peligrosos, puede existir riesgo de ataque.

nmap --script http-methods -p80 192.168.1.12

4. Escaneo NSE completo contra el sitio web

Realiza un reconocimiento combinado usando scripts de la categoría "default", "safe", "vuln".

nmap -p80 --script "default,safe,vuln" 192.168.1.12

Resultados Relevantes Obtenidos

- Se descubrieron directorios expuestos: /config/, /database/, /css/.
- El servidor responde con Apache/2.4.58 (Ubuntu).
- La cookie PHPSESSID no tiene la bandera HttpOnly.
- No existe protección de métodos HTTP adicionales.
- La exposición de /database/ permitió visualizar el archivo schema.sql.

Conclusión

La ejecución de scripts NSE permitió obtener una visión clara de la configuración del servidor web y detectar fallas importantes en el entorno, como directorios expuestos y cabeceras inseguras. Esta fase de reconocimiento es esencial dentro del proceso de ciberseguridad, ya que facilita la identificación temprana de vulnerabilidades críticas antes de realizar ataques más avanzados.

16. DIRECTRIZ DE POLÍTICAS DE SEGURIDAD DE LA INFORMACIÓN

. DISPOSICIONES GENERALES

Finalidad

MANDATORIO: Cumplir políticas del Manual LUZENERGIA protegiendo CID de datos en stack Apache/PHP/MariaDB contra vulnerabilidades identificadas (SQLi, SSH débil, cookies inseguras).

Ámbito técnico

VirtualBox: Ubuntu Server (192.168.1.12) + Kali Linux
auditoría

Servicios: Apache:80, SSH:22, MariaDB puerto 3306

BD: proyecto_ciber (usuarios/facturas) - Usuario:
luzuser@localhost

Web: /var/www/html/luzenergia/ - login.php vulnerable
SQLi[file:36][file:18]

1.3 Autoridad

DBA LUZENERGIA → Reporte SIC mensual + Auditorías Nmap/Hydra simuladas.

OBLIGACIONES MANDATORIAS POR ROL (RBAC)

ROL	RESPONSABILIDADES CRÍTICAS	FRECUENCIA	EVIDENCIA	SANCIONES

DBA (luzuser)	<code>mysqldump</code> diario, <code>general_log=ON</code> , rotación backups	Diaria 02:00	<code>/backups/luzener</code> <code>gia/</code>	Suspensión + Ley 1273
Admin Apache	HTTPS/TLS1.3, <code>HttpOnly/Secure/SameSite</code> cookies PHPSESSID	Inmediata	<code>http-headers</code> NSE	Judicial
Desarrolla dor PHP	Prepared statements vs SQLi login.php, validar <code>\$_POST</code>	Cada commit	<code>sqlmap -u</code> <code>login.php</code>	Civil
Usuario CLI	Reportar bloqueos (5 intentos), MFA admins	Inmediata	Logs <code>/var/log/apache2</code> <code>/</code>	Cierre cuenta
Auditor Kali	Nmap mensual <code>-p- -sS -sV -A</code> , Hydra tests controlados	Mensual	Reporte PDF	N/A

. MEDIDAS TÉCNICAS OBLIGATORIAS (Post-Auditoría)

3.1 SQL Injection Mitigation (login.php)

MANDATORIO: Reemplazar `mysql_query` por PDO prepared statements

PROHIBIDO: `$_POST['usuario']` directo en query

// VULNERABLE (Eliminar)

`$query = "SELECT * FROM usuarios WHERE usuario='$_POST[usuario]'";`

// SEGURO (Implementar)

```
$stmt = $pdo->prepare("SELECT * FROM usuarios WHERE usuario=?");
```

```
$stmt->execute([$ _POST['usuario']]);
```

SSH Hardening (Puerto 22)

Fail2ban: Bloqueo IP tras 3 intentos Hydra

Cambiar puerto: Port 2222 en /etc/ssh/sshd_config

Prohibir root: PermitRootLogin no

Claves SSH obligatorias: PasswordAuthentication no

Cookies Seguras (PHPSESSID)

// En php.ini o .htaccess

```
session.cookie_httponly = 1
```

```
session.cookie_secure = 1
```

```
session.cookie_samesite = Strict
```

Backups MariaDB (RPO<1h RTO<4h)

Cron 02:00: mysqldump --single-transaction proyecto_ciber | gpg

Retención: 30 días local + AWS S3 SSE-KMS

Test restore: Semanal staging VirtualBox[file:18]

PROCESOS OBLIGATORIOS (Workflows)

PROCESO	RESPONSABLE	PLAZO	COMANDO/EVIDENCIA
Consentimiento RGPD	Frontend	Registro	Checkbox /registro.php log
Habeas Data ARCO	DBA	≤15 días	habeas@luzenergia.com + habeas_log tabla
Nmap Reconocimiento	Auditor	Mensual	nmap -p- -sS -sV -A 192.168.1.12
Hydra Test Controlado	Auditor	Trimestral	hydra -l ciberteo -P passlist.txt ssh://192.168.1.12
Brecha SIC	Legal	≤72h	Constancia PDF numerada
Restore Backup	DBA	<4h	mysql < luzenergia-YYYYMMDD.sql.gpg

KPIs CUMPLIMIENTO (Monitoreo Automático)

KPI	META	HERRAMIENTA	ALERTA
Backups exitosos	100%	Cron logs	Email DBA 03:00
SQLi detectadas	0	<code>sqlmap --batch</code>	WAF + Slack
SSH brute force	0	Fail2ban logs	IP ban + SMS
Uptime Apache	99.9%	<code>systemctl status apache2</code>	Nagios/Prometheus
Habeas response	≤15 días	Google Sheets	Recordatorio auto

SCRIPT VERIFICACIÓN DIARIA (cron 03:00):

```
if [ ! -f "/backups/$(date +%Y%m%d)*.gpg" ]; then
    mail -s "BACKUP FALLÓ LUZENERGIA" dba@talentotech.edu.co
fi
```

CANALES COMUNICACIÓN (24/7)

EMERGENCIAS: incidentes@luzenergia.com → ≤4h respuesta

HABEAS DATA: habeas@luzenergia.com → ≤15 días

AUDITORÍAS: auditoria@talentotech.edu.co → Reportes Nmap/PDF

SIC BRECHAS: sic@luzenergia.com → ≤72h Superintendencia

REPORTES BUGS: github.com/grupo6/luzenergia/issues

CHECKLIST IMPLEMENTACIÓN (FASES)

FASE 1 - CRÍTICA (48h)

- PDO prepared statements login.php
- Fail2ban SSH + Apache
- Cookies HttpOnly/Secure/SameSite
- Cron backups + test restore

FASE 2 - ALTA (Semana 1)

- HTTPS Let's Encrypt
- Directorios protegidos (.htaccess)
- Logs MariaDB general_log=ON

FASE 3 - MEDIA (Mensual)

- Nmap auditoría
- KPIs dashboard
- Capacitación grupo 6

AUDITORÍA Y VERIFICACIÓN

Mensual (Día 1):

1. `nmap -p- --script vuln 192.168.1.12`
2. `grep "SQLi\|Hydra" /var/log/apache2/`
3. `ls -la /backups/ | tail -30`
4. `SELECT COUNT(*) FROM habeas_log;`

Trimestral: OWASP ZAP + sqlmap completo

Anual: Certificación ISO 27001 simulada

SANCIONES (Escalada Automática)

GRAVEDAD	MEDIDA	BASE LEGAL
Leve (KPI 90%)	Advertencia + capacitación	Interna
Moderada (Incidente SQLi)	Suspensión 30 días	Disciplinaria
Grave (Brecha >100 registros)	Despido/cierre	Ley 1273/2009
Crítica (Ransomware)	Judicial + SIC	Código Penal

VIGENCIA

Entrada en vigor: INMEDIATA

Revisión: Diciembre 2026 o post-incidente OWASP Top 10

17. Práctica: Reconocimiento web con Nmap + Análisis de Base de Datos

1. Escaneo inicial del host

- Este escaneo es para saber si el servidor está activo y qué puertos están abiertos.

Comando:

nmap 192.168.1.12

2. Escaneo de puertos comunes web

- Para identificar servicios web en el servidor.

Comando:

nmap -p 80,443 192.168.1.12

3. Escaneo agresivo de servicios web

- Sirve para obtener información del servidor, versión de Apache, PHP, etc.

Comando:

nmap -A 192.168.1.12

4. Escaneo usando scripts NSE para web

Scripts importantes:

- ✓ Detección de vulnerabilidades web

nmap --script http-vuln* 192.168.1.12

- ✓ Enumeración de archivos y rutas

nmap --script http-enum 192.168.1.12

- ✓ Información del servidor

nmap --script http-server-header 192.168.1.12

nmap --script http-headers 192.168.1.12

5. Análisis de Base de Datos (archivo db.php)

El objetivo era verificar si el archivo db.php era accesible desde el navegador.

- Ver archivo desde curl

curl -i http://192.168.1.12/config/db.php

- Ver como texto plano

curl -i -H "Accept: text/plain" http://192.168.1.12/config/db.php

- Descargar el archivo db.php

curl -i --output db.php http://192.168.1.12/config/db.php

- Ver contenido descargado

cat schema.sql

```
(kali@kali)-[~]
$ cat schema.sql

-- Base de datos para LuzEnergia
CREATE DATABASE IF NOT EXISTS luzenergia;
USE luzenergia;

-- Tabla de usuarios
CREATE TABLE IF NOT EXISTS usuarios (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre VARCHAR(100) NOT NULL,
  usuario VARCHAR(50) NOT NULL UNIQUE,
  email VARCHAR(100) NOT NULL,
  password VARCHAR(255) NOT NULL,
  numero_cliente VARCHAR(20) NOT NULL UNIQUE,
  fecha_registro TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Tabla de facturas
CREATE TABLE IF NOT EXISTS facturas (
  id INT AUTO_INCREMENT PRIMARY KEY,
  numero_cliente VARCHAR(20) NOT NULL,
  mes VARCHAR(20) NOT NULL,
  consumo DECIMAL(10,2) NOT NULL,
  monto DECIMAL(10,2) NOT NULL,
  estado VARCHAR(20) DEFAULT 'Pendiente',
```

18. ¿Qué es un ataque de Fuerza Bruta con Hydra?

Un ataque de fuerza bruta consiste en probar automáticamente miles o millones de combinaciones de contraseña hasta encontrar la correcta.

Hydra es una herramienta de pentesting muy potente incluida en Kali Linux y permite atacar servicios como:

- SSH
- FTP
- HTTP
- RDP
- Telnet
- MySQL
- SMB

...entre muchos otros.

Comando:

hydra -l USERNAME -P /usr/share/wordlists/rockyou.txt ssh://IP -t 4

Descripción del comando:

- -l USERNAME → nombre del usuario objetivo
- -P /usr/share/wordlists/rockyou.txt → diccionario masivo (14 millones de contraseñas)

ssh://IP → servicio a atacar

- -t 4 → número de hilos para acelerar



Tiempo real estimado

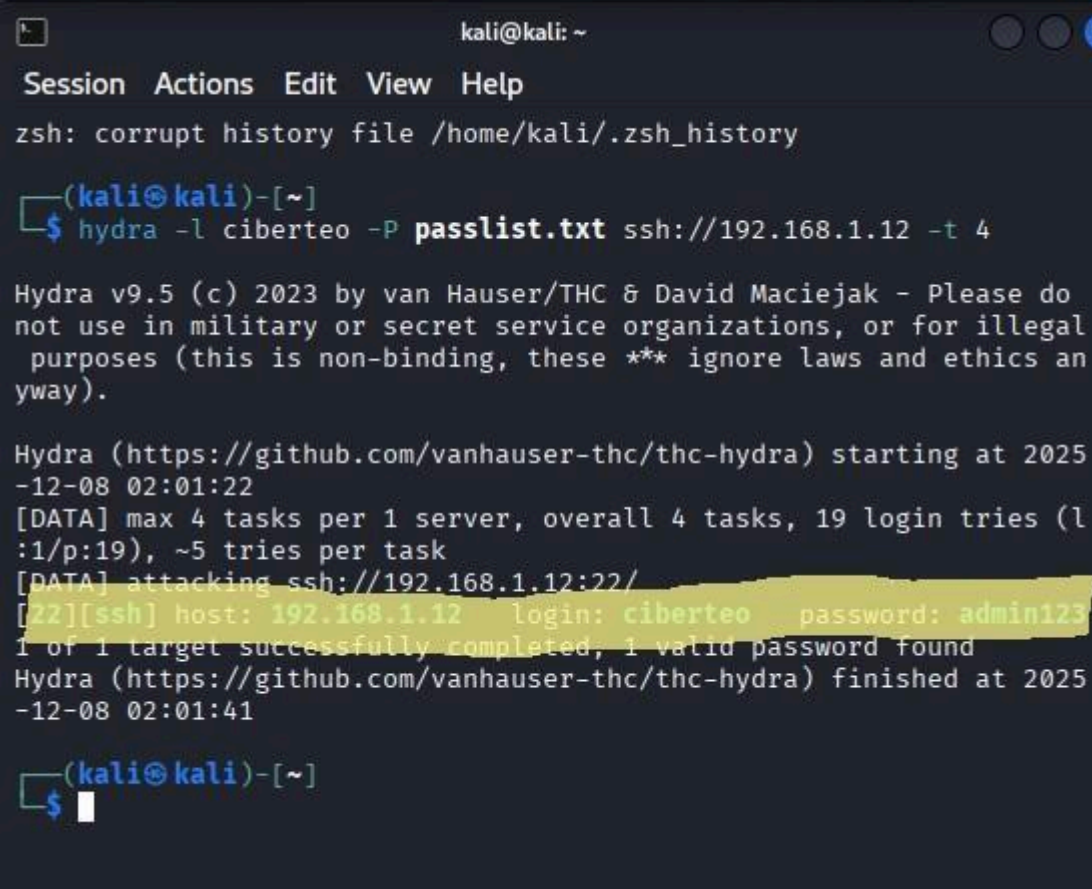
Rockyou.txt tiene 14,344,399 contraseñas.

En un equipo normal Hydra puede probar:

~100 a 500 contraseñas por segundo

→ entre 8 y 24 horas de ataque

Por eso se considera lento y muy ruidoso (fácil de detectar).



```
kali@kali: ~  
Session Actions Edit View Help  
zsh: corrupt history file /home/kali/.zsh_history  
(kali@kali)-[~]  
$ hydra -l ciberteo -P passlist.txt ssh://192.168.1.12 -t 4  
  
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do  
not use in military or secret service organizations, or for illegal  
purposes (this is non-binding, these *** ignore laws and ethics an  
yway).  
  
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025  
-12-08 02:01:22  
[DATA] max 4 tasks per 1 server, overall 4 tasks, 19 login tries (1  
:1/p:19), ~5 tries per task  
[DATA] attacking ssh://192.168.1.12:22/  
[22][ssh] host: 192.168.1.12 login: ciberteo password: admin123  
1 of 1 target successfully completed, 1 valid password found  
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025  
-12-08 02:01:41  
(kali@kali)-[~]  
$
```

19. Ataque de Fuerza Bruta con Hydra sobre servicio SSH

Para evaluar la fortaleza del servicio SSH del servidor (192.168.1.12), se realizó una prueba de fuerza bruta utilizando la herramienta Hydra incluida en Kali Linux.

Primero se ejecutó el comando estándar utilizado en pentesting profesional:

```
hydra -l ciberteo -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.12 -t 4
```

Sin embargo, debido al tamaño del diccionario (14 millones de contraseñas) y el tiempo estimado de ejecución (entre 8 y 24 horas), se decidió crear un diccionario más pequeño para fines educativos. El diccionario personalizado (passlist.txt)

incluyó contraseñas comunes, débiles y relacionadas con la temática del proyecto. Con este diccionario reducido, se ejecutó el ataque:

```
hydra -l ciberteo -P passlist.txt ssh://192.168.1.12 -t 4
```

Después de unos segundos, Hydra encontró la contraseña correcta del usuario:
password found: admin123

Resultado: El servicio SSH fue comprometido exitosamente. Esto demuestra que el servidor utilizaba una contraseña débil y fácilmente vulnerable a ataques automatizados.

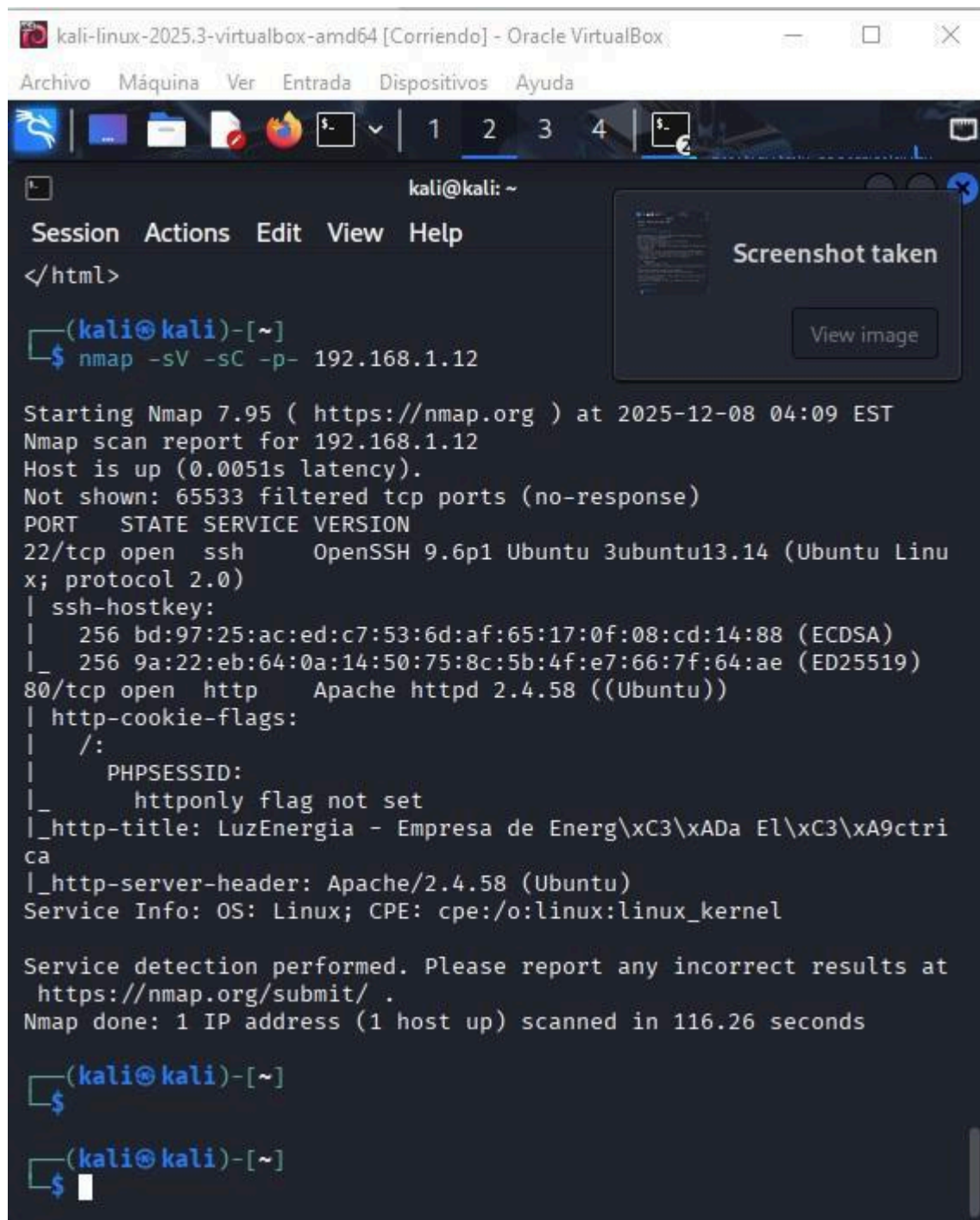
20. SQL Injection en LuzEnergia

Como parte del proceso de pruebas de seguridad, se realizó primero un reconocimiento del servidor utilizando Nmap, con el objetivo de identificar los servicios activos y posibles debilidades.

1. Identificación inicial de servicios

Mediante el comando:

```
nmap -sV -sC -p- 192.168.1.12
```



```
kali@kali: ~  
Session Actions Edit View Help  
</html>  
(kali@kali)-[~]  
$ nmap -sV -sC -p- 192.168.1.12  
  
Starting Nmap 7.95 ( https://nmap.org ) at 2025-12-08 04:09 EST  
Nmap scan report for 192.168.1.12  
Host is up (0.0051s latency).  
Not shown: 65533 filtered tcp ports (no-response)  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.14 (Ubuntu Linux; protocol 2.0)  
| ssh-hostkey:  
|   256 bd:97:25:ac:ed:c7:53:6d:af:65:17:0f:08:cd:14:88 (ECDSA)  
|_  256 9a:22:eb:64:0a:14:50:75:8c:5b:4f:e7:66:7f:64:ae (ED25519)  
80/tcp    open  http      Apache httpd 2.4.58 ((Ubuntu))  
| http-cookie-flags:  
|   /:  
|   PHPSESSID:  
|_   httponly flag not set  
|_ http-title: LuzEnergia - Empresa de Energ\xC3\xADa El\xC3\xA9ctrica  
|_ http-server-header: Apache/2.4.58 (Ubuntu)  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel  
  
Service detection performed. Please report any incorrect results at  
https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 116.26 seconds  
  
(kali@kali)-[~]  
$  
  
(kali@kali)-[~]  
$
```

se detectó:

Un servidor Apache ejecutando la aplicación web (puerto 80).

Un servicio SSH expuesto (puerto 22).

La existencia de un formulario de autenticación en:

`http://192.168.1.12/login.php`

Además, Nmap reportó configuraciones inseguras relacionadas con cookies de sesión, lo que indicaba una baja madurez en la protección del lado del servidor.

- análisis del formulario de inicio de sesión

`curl -X POST -d "usuario=test&password=123"`

<http://192.168.1.12/login.php>

Este comando simula un intento de inicio de sesión contra el formulario login.php de tu aplicación web, pero sin usar un navegador, sino directamente desde la terminal de Kali Linux.

- curl: herramienta para enviar peticiones HTTP desde terminal.
- -X POST: indica que la petición es de tipo POST (como un formulario).
- -d "usuario=...&password=..." : datos que se envían (campos usuario y password).
- URL: destino de la petición (el script que procesa el login).
- Al inspeccionar el comportamiento del login, se observó que:

```

<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Iniciar Sesión - LuzEnergia</title>
  <link rel="stylesheet" href="css/style.css">
</head>
<body>
  <header>
    <nav class="navbar">
      <div class="logo">
        <h1> LuzEnergia</h1>
      </div>
      <ul class="nav-links">
        <li><a href="index.php">Inicio</a></li>
        <li><a href="login.php">Iniciar Sesión</a></li>
      </ul>
    </nav>
  </header>

  <main>
    <div class="login-container">
      <h2>Iniciar Sesión</h2>

      <div class="error">Usuario o contraseña
      incorrectos</div>

      <form method="POST" action="login.php">
        <div class="form-group">
          <label>Usuario:</label>
          <input type="text" name="usuario" required>
        </div>

```

El sistema mostraba mensajes diferentes según el resultado (“Usuario o contraseña incorrectos”).

No existía un mecanismo visible de protección como CAPTCHA, bloqueo de intentos o tokens CSRF.

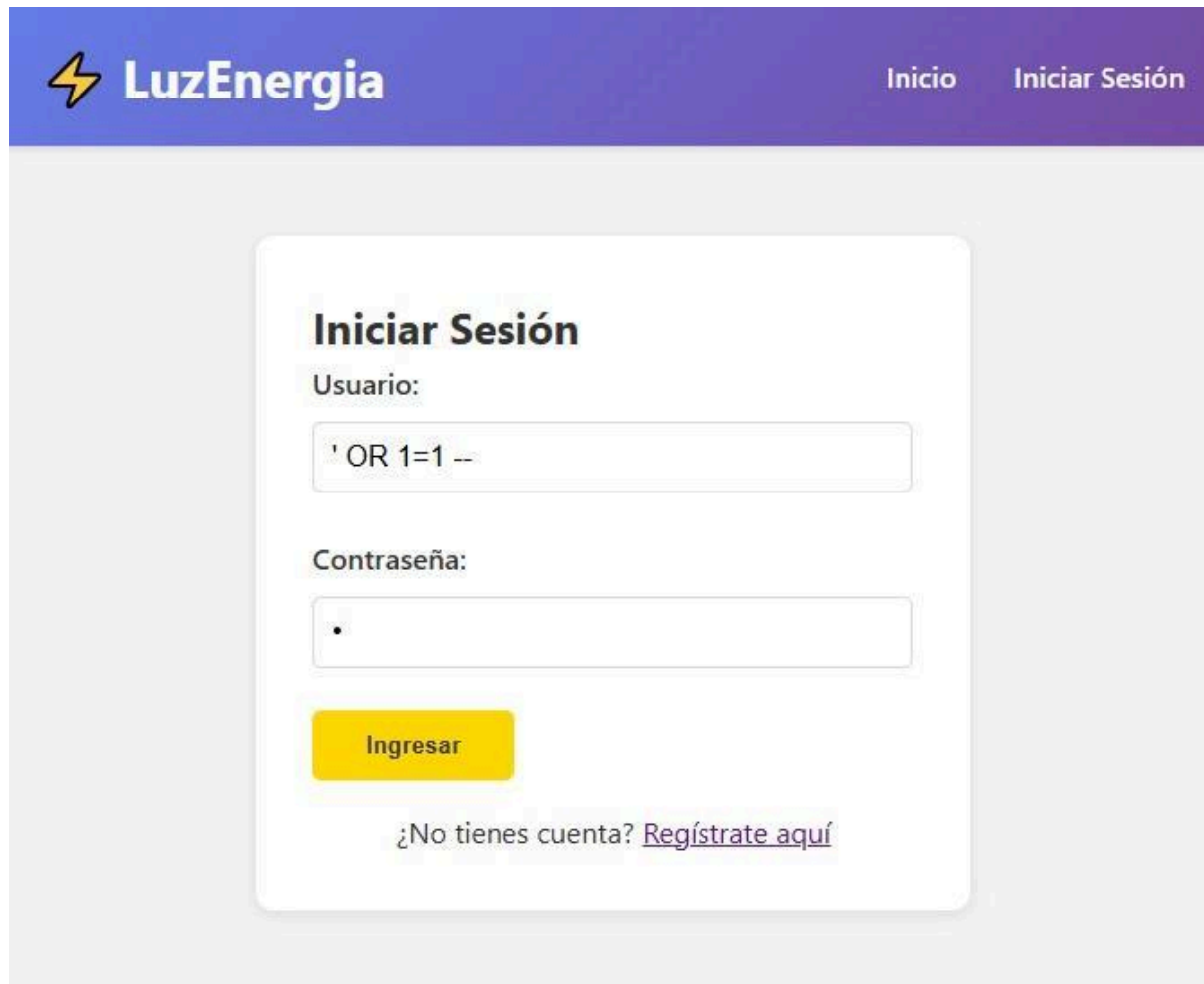
El parámetro usuario parecía ser enviado directamente al servidor sin validaciones fuertes.

Esto sugirió la posibilidad de una inyección SQL en el punto de autenticación.

- Prueba de SQL Injection

Se realizó una prueba básica enviando una consulta maliciosa clásica: usuario: ' OR '1'='1

password: cualquier_cosa



The screenshot shows the 'Iniciar Sesión' (Login) page of the LuzEnergia website. The page has a purple header with the logo and navigation links 'Inicio' and 'Iniciar Sesión'. The login form is centered and contains the following elements:

- Iniciar Sesión** (Login)
- Usuario:** (User) field containing the malicious query: `' OR 1=1 --`
- Contraseña:** (Password) field containing a single dot: `.`
- Ingresar** (Login) button
- Link: [¿No tienes cuenta? Regístrate aquí](#)

Esto simuló una condición siempre verdadera en la consulta SQL del servidor.

Al enviar la carga al formulario, el sistema permitió el acceso, confirmando que el formulario era vulnerable a SQL Injection.

Esto indicó que el servidor ejecutaba directamente los valores ingresados por el usuario dentro de la consulta SQL, sin sanitización ni preparación de parámetros.

Bienvenido, Juan Pérez

Información de tu cuenta

Usuario: juan

Email: juan@email.com

Número de Cliente: CLI10001

Tus Facturas Recientes

Mes	Consumo (kWh)	Monto	Estado
Marzo 2024	165.75	\$49.73	Pendiente
Febrero 2024	180.25	\$54.08	Pagada
Enero 2024	150.50	\$45.15	Pagada

Introducción: Implementación de Seguridad y Mitigación de Vulnerabilidades

En el ámbito de la ciberseguridad, uno de los pilares fundamentales es la identificación y corrección de vulnerabilidades presentes en los sistemas y aplicaciones. Todo proyecto tecnológico debe contemplar no solo la creación de un servicio funcional, sino también la protección de la infraestructura frente a posibles ataques que puedan comprometer la integridad, disponibilidad o confidencialidad de la información.

En este proyecto, después de realizar un análisis de la infraestructura utilizando herramientas de reconocimiento como Nmap, se identificaron varios servicios expuestos y potencialmente vulnerables, entre ellos:

- SSH (puerto 22)
- Servidor web Apache (puerto 80)

Aplicación web con formulario de autenticación

A partir de esta información, se realizaron pruebas controladas de penetración (pentesting) para evaluar la seguridad de la plataforma, entre ellas:

- Ataques de fuerza bruta contra SSH y el login web
- Intentos de inyección SQL en el sistema de autenticación

Revisiones de encabezados HTTP y configuraciones del servidor

El propósito de este apartado es documentar las medidas implementadas para mitigar dichas vulnerabilidades, endurecer la seguridad del entorno (hardening) y demostrar cómo cada ajuste mejora significativamente la protección del sistema.

Se incluyen las soluciones prácticas aplicadas, como:

Implementación de **Fail2Ban** para frenar ataques de fuerza bruta sobre SSH.

Añadido de controles de bloqueo de intentos en el login web.

Corrección del código vulnerable a inyección SQL mediante consultas preparadas.

Aplicación de medidas de hardening en Apache para evitar filtración de información sensible del servidor.

Mitigación de Fuerza Bruta en SSH (Ubuntu Server)

Durante la fase de pruebas de seguridad del proyecto “LuzEnergia”, se detectó que varios servicios del sistema eran susceptibles a ataques de fuerza bruta, especialmente el acceso SSH del servidor Ubuntu y el formulario de inicio de sesión de la aplicación web.

Los ataques de fuerza bruta consisten en intentar muchas contraseñas o combinaciones posibles hasta encontrar la correcta. Aunque los ataques fueron realizados con fines educativos usando herramientas como Hydra, es esencial implementar medidas de mitigación que reduzcan o eliminen este tipo de riesgo.

A continuación, se presentan los controles aplicados y su respectiva configuración para endurecer el servidor y la aplicación web frente a ataques de fuerza bruta.

- **instalar Fail2Ban**

sudo apt install fail2ban -y

```
desempaquetando python3-pyasyncore (1.0.2-2) ...
seleccionando el paquete fail2ban previamente no seleccionado.
preparando para desempaquetar .../fail2ban_1.0.2-3ubuntu0.1_all.deb ...
desempaquetando fail2ban (1.0.2-3ubuntu0.1) ...
seleccionando el paquete python3-pyinotify previamente no seleccionado.
preparando para desempaquetar .../python3-pyinotify_0.9.6-2ubuntu1_all.deb ...
desempaquetando python3-pyinotify (0.9.6-2ubuntu1) ...
seleccionando el paquete whois previamente no seleccionado.
preparando para desempaquetar .../whois_5.5.22_amd64.deb ...
desempaquetando whois (5.5.22) ...
configurando whois (5.5.22) ...
configurando python3-pyasyncore (1.0.2-2) ...
configurando fail2ban (1.0.2-3ubuntu0.1) ...
usr/lib/python3/dist-packages/fail2ban/tests/fail2banregextestcase.py:224: SyntaxWarning: "1490349000 test failed.dns.ch", "\s*test <F-ID>\S+</F-ID>"
usr/lib/python3/dist-packages/fail2ban/tests/fail2banregextestcase.py:435: SyntaxWarning: '^'+prefix+'<F-ID>User <F-USER>\S+</F-USER></F-ID> not allowed\n'
usr/lib/python3/dist-packages/fail2ban/tests/fail2banregextestcase.py:443: SyntaxWarning: '^'+prefix+'User <F-USER>\S+</F-USER> not allowed\n'
usr/lib/python3/dist-packages/fail2ban/tests/fail2banregextestcase.py:444: SyntaxWarning: '^'+prefix+'Received disconnect from <F-ID><ADDR> port \d+</F-ID>'
usr/lib/python3/dist-packages/fail2ban/tests/fail2banregextestcase.py:451: SyntaxWarning: _test_variants('common', prefix="\s*\S+ sshd\[<F-MLFID>\d+</F-MLFID>\]:\s+")
usr/lib/python3/dist-packages/fail2ban/tests/fail2banregextestcase.py:537: SyntaxWarning: 'common[prefregex="^svc\[<F-MLFID>\d+</F-MLFID>\] connect <F-CONTENT>.+</F-CONTENT>'
usr/lib/python3/dist-packages/fail2ban/tests/servertestcase.py:1375: SyntaxWarning: "{ nft -a list chain inet f2b-table f2b-chain | grep -oP '@addr-set-j-w-nft-"
usr/lib/python3/dist-packages/fail2ban/tests/servertestcase.py:1378: SyntaxWarning: "{ nft -a list chain inet f2b-table f2b-chain | grep -oP '@addr6-set-j-w-nft-"
usr/lib/python3/dist-packages/fail2ban/tests/servertestcase.py:1421: SyntaxWarning: "{ nft -a list chain inet f2b-table f2b-chain | grep -oP '@addr-set-j-w-nft-"
usr/lib/python3/dist-packages/fail2ban/tests/servertestcase.py:1424: SyntaxWarning: "{ nft -a list chain inet f2b-table f2b-chain | grep -oP '@addr6-set-j-w-nft-"
```

Descripción:

Instala la herramienta fail2ban que monitorea los logs y bloquea IPs que fallen muchas veces al iniciar sesión por SSH.

- **Activar la protección para SSH**

Comando:

sudo nano /etc/fail2ban/jail.local

- Pegar dentro del archivo:

[sshd]

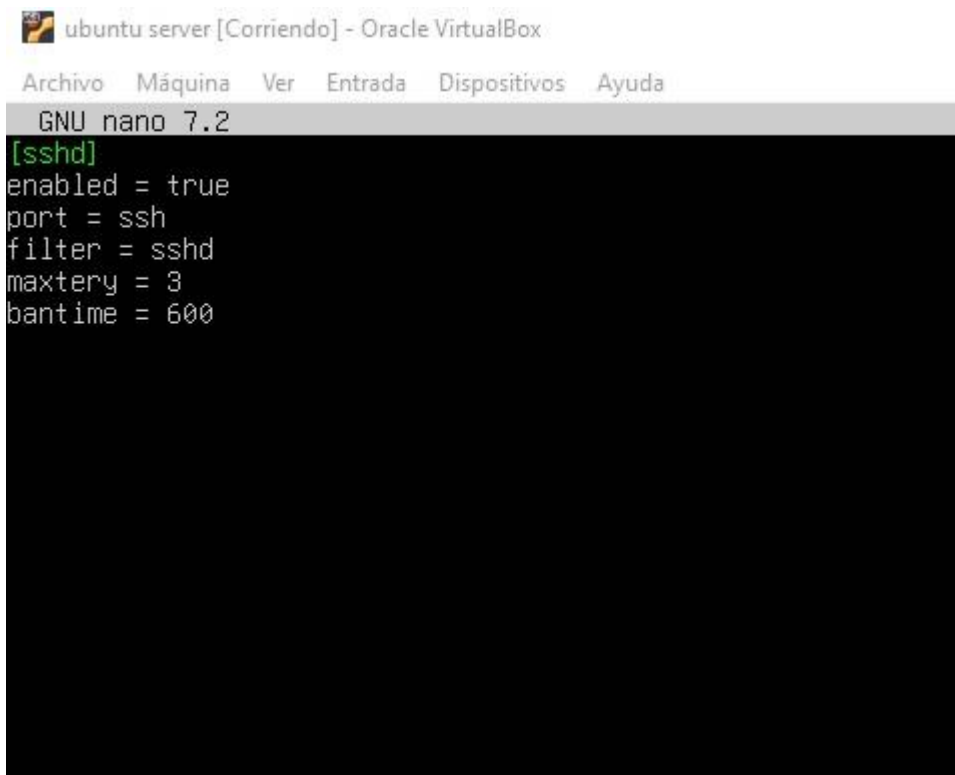
enabled = true

port = ssh

filter = sshd

maxretry = 3

bantime = 600



The screenshot shows a terminal window titled "ubuntu server [Corriendo] - Oracle VirtualBox". The terminal is running the GNU nano 7.2 text editor. The editor's content shows the configuration for the [sshd] service, with the following lines: enabled = true, port = ssh, filter = sshd, maxretry = 3, and bantime = 600. The terminal has a standard menu bar with options: Archivo, Máquina, Ver, Entrada, Dispositivos, and Ayuda.

```
ubuntu server [Corriendo] - Oracle VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 7.2
[sshd]
enabled = true
port = ssh
filter = sshd
maxretry = 3
bantime = 600
```

[sshd]

Indica que estamos configurando la protección específicamente para

el servicio SSH.

- **enabled = true**

Activa la protección

- **port = ssh**

Indica que Fail2ban debe vigilar el puerto SSH (por defecto, 22).

- **filter = sshd**

Usa el filtro por defecto para detectar intentos fallidos en SSH.

- **maxretry = 3**

Después de 3 intentos fallidos, Fail2ban bloqueará la IP atacante.

- **bantime = 600**

El bloqueo durará 600 segundos (10 minutos).

- **Reiniciar Fail2ban para aplicar cambios**

En Ubuntu ejecuta:

sudo systemctl restart fail2ban

- **Confirmar que Fail2ban está funcionando**

sudo fail2ban-client status sshd

```
ciberteo@ubuntu-server:~$ sudo systemctl restart fail2ban
[sudo] password for ciberteo:
ciberteo@ubuntu-server:~$ sudo fail2ban-client status sshd
Status for the jail: sshd
|- Filter
|   |- Currently failed: 0
|   |- Total failed:    0
|   \- Journal matches: _SYSTEMD_UNIT=sshd.service + _COMM=sshd
- Actions
    |- Currently banned: 0
    |- Total banned:    0
    \- Banned IP list:
ciberteo@ubuntu-server:~$
```

2. Mitigación de Fuerza Bruta en el Login Web (PHP + Apache)

El ataque de fuerza bruta consiste en probar miles de combinaciones de usuario y contraseña hasta encontrar una válida.

Durante el análisis, se detectó que el formulario de login del sistema aceptaba intentos ilimitados, lo cual permite a herramientas como Hydra automatizar intentos masivos.

Por esta razón, se implementaron controles para mitigar este riesgo.

Código Vulnerable (Versión Antigua del Login)

```

<?php
session_start();
include 'config/db.php';

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $usuario = $_POST['usuario'];
    $password = $_POST['password'];

    // ❌ CONSULTA VULNERABLE A SQLi
    $query = "SELECT * FROM usuarios WHERE usuario='$usuario' AND password='$password'";
    $resultado = mysqli_query($conn, $query);

    if ($resultado && mysqli_num_rows($resultado) > 0) {
        $user = mysqli_fetch_assoc($resultado);

        $_SESSION['usuario'] = $user['usuario'];
        $_SESSION['id'] = $user['id'];
        $_SESSION['numero_cliente'] = $user['numero_cliente'];

        header("Location: panel.php");
        exit();
    } else {
        $error = "Usuario o contraseña incorrectos";
    }
}
?>

```

Por qué este código es vulnerable

1. Vulnerable a SQL Injection

El código inserta directamente los valores enviados por el usuario dentro de la consulta SQL:

' OR '1'='1

Esto permitiría iniciar sesión sin conocer la contraseña.

2. Permite Fuerza Bruta

No existen medidas como:

Límite de intentos

bloqueo temporal

delay entre intentos

registro de intentos fallidos

Un atacante puede automatizar miles de intentos por minuto.

● 3. Contraseñas sin hash

Las contraseñas se comparan en texto plano (`password='12345'`), lo cual es inseguro si la base de datos es comprometida.

Código Seguro (Versión Mejorada del Login)

```

<?php
session_start();
include 'config/db.php';

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $usuario = $_POST['usuario'];
    $password = $_POST['password'];

    // ☒ Consulta segura usando prepared statements
    $stmt = $conn->prepare("SELECT * FROM usuarios WHERE usuario=? AND password=?");
    $stmt->bind_param("ss", $usuario, $password);
    $stmt->execute();
    $resultado = $stmt->get_result();

    if ($resultado->num_rows > 0) {
        $user = $resultado->fetch_assoc();
        $_SESSION['usuario'] = $user['usuario'];
        header("Location: panel.php");
        exit();
    } else {
        $error = "Usuario o contraseña incorrectos";
    }
}
?>

```

Por qué este código es seguro:

1. Uso de Prepared Statements

Evita SQL Injection porque:

los valores del usuario no se mezclan con el código SQL

MySQL los interpreta como texto, no como comandos

2. Evita bypass de autenticación

Una inyección como:

' OR '1'='1

ya no funciona.

● 3. Facilita añadir medidas anti-fuerza bruta

Con este código se puede implementar:

conteo de intentos fallidos

bloqueo de IP por tiempo

retrasos entre intentos

Hardening del Servidor Apache (Ubuntu Server)

El servidor web Apache, por defecto, expone información sensible como su versión exacta, módulos cargados y detalles del sistema operativo subyacente. Esta información puede ser utilizada por un atacante para relacionar la versión del servidor con vulnerabilidades ya conocidas.

Durante las pruebas de reconocimiento realizadas mediante Nmap y análisis manual, se identificó que el encabezado HTTP del servidor revelaba lo siguiente:

Apache/2.4.58 (Ubuntu)

La divulgación de esta información facilita la identificación de vulnerabilidades específicas presentes en esa versión.

Para reducir este riesgo, se aplicaron medidas de *hardening* enfocadas en disminuir la superficie de ataque y evitar la filtración de datos innecesarios sobre el servidor.

- **Ocultar la versión del servidor Apache**

Comando para abrir el archivo de configuración:

sudo nano /etc/apache2/conf-available/security.conf

```
ServerTokens
This directive configures what you return as the Server HTTP response
Header. The default is 'Full' which sends information about the OS-Type
and compiled in modules.
Set to one of: Full | OS | Minimal | Minor | Major | Prod
where Full conveys the most information, and Prod the least.
ServerTokens Minimal
ServerTokens OS
ServerTokens Full

Optionally add a line containing the server version and virtual host
name to server-generated pages (internal error documents, FTP directory
listings, mod_status and mod_info output etc., but not CGI generated
documents or custom error documents).
Set to "EMail" to also include a mailto: link to the ServerAdmin.
Set to one of: On | Off | EMail
ServerSignature Off
ServerSignature On

Allow TRACE method

Set to "extended" to also reflect the request body (only for testing and
diagnostic purposes).

Set to one of: On | Off | extended
TraceEnable Off
TraceEnable On
```

Modificaciones dentro del archivo

Buscar estas líneas y cambiarlas:

ServerTokens Prod

ServerSignature Off

Explicación

ServerTokens Prod

Solo muestra "Apache" sin versión ni sistema operativo.

Antes: Apache/2.4.58 (Ubuntu)

Ahora: Apache

ServerSignature Off

Quita la firma del servidor en páginas de error (403, 404, 500).

Esto evita que un atacante conozca detalles internos del sistema.

Aplicar cambios

```
where Full conveys the most information, and Prod the least.
ServerTokens prod
ServerTokens prod
ServerTokens prod

Optionally add a line containing the server version and virtual host
name to server-generated pages (internal error documents, FTP directo
r listings, mod_status and mod_info output etc., but not CGI generated
documents or custom error documents).
Set to "EMail" to also include a mailto: link to the ServerAdmin.
Set to one of: On | Off | EMail
ServerSignature Off
ServerSignature off

Allow TRACE method

Set to "extended" to also reflect the request body (only for testing
diagnostic purposes).

Set to one of: On | Off | extended
TraceEnable Off
TraceEnable off

Forbid access to version control directories

If you use version control systems in your document root, you should
probably deny access to their directories.

Examples:
```

sudo systemctl restart apache2

Deshabilitar módulos innecesarios

Ver módulos habilitados

apache2ctl -M

Deshabilitar un módulo inseguro (ejemplo AutoIndex)

sudo a2dismod autoindex

sudo systemctl restart apache2

Explicación

Reduce superficie de ataque.

Evita filtración de archivos internos si un directorio queda sin index.html.

Activar firewall UFW para proteger Apache

```
iberteo@ubuntu-server:~$ sudo systemctl restart apache2
iberteo@ubuntu-server:~$ sudo ufw allow "Apache"
Rules updated
Rules updated (v6)
iberteo@ubuntu-server:~$ sudo ufw enable
Firewall is active and enabled on system startup
iberteo@ubuntu-server:~$
```

Permitir solo tráfico web necesario

sudo ufw allow 'Apache'

sudo ufw enable

Explicación

Activo firewall para filtrar tráfico entrante.

Solo habilita los puertos esenciales de la web (80 y 443 si existiera SSL).

Configurar permisos seguros en el directorio web

Asignar permisos recomendados

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

sudo chmod -R 755 /var/www/html

```
iberteo@ubuntu-server:~$ sudo chown -R www-data:www-data /var/www/html
iberteo@ubuntu-server:~$ sudo chmod -R 755 /var/www/html
iberteo@ubuntu-server:~$
```

Explicación

Evita que otros usuarios del sistema modifiquen los archivos de la web.

Protege el código del sitio ante accesos no autorizados

Seguridad en la Base de Datos (MariaDB)

La base de datos es uno de los componentes más críticos del sistema. Durante las pruebas del proyecto LuzEnergia, se identificó que la base de datos permitía accesos con configuraciones por defecto y sin restricciones adecuadas, lo cual aumenta el riesgo ante ataques como:

Acceso no autorizado

Robo de datos

Inyección SQL

Elevación de privilegios

Para mitigar estos riesgos, se aplicaron diversas medidas de hardening en MariaDB.

Asegurar MariaDB con mysql_secure_installation

Comando:

sudo mysql_secure_installation

¿Qué hace este comando?

Este script oficial de MariaDB:

- Configura contraseña segura para el usuario root
- Deshabilita accesos remotos inseguros
- Elimina usuarios anónimos
- Borra bases de datos de prueba
- Refuerza la política de contraseñas

```

... Success!

You already have your root account protected, so you can safely answer 'n'
change the root password? [Y/n]
new password:
re-enter new password:
Sorry, passwords do not match.

new password:
re-enter new password:
password updated successfully!
Reloading privilege tables..
... Success!

By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them. This is intended only for testing, and to make the installation
a bit smoother. You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] n
... skipping.

Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
... skipping.

```

Se ejecutó el script `mysql_secure_installation` para eliminar configuraciones por defecto que representan un riesgo de seguridad. Esto asegura que no existan usuarios anónimos y que solo root pueda acceder al servidor localmente.

21, MANUAL DE POLÍTICAS DE SEGURIDAD DE LA INFORMACIÓN

21.1. INTRODUCCIÓN

LUZENERGIA es una plataforma web diseñada para la gestión de usuarios y facturación de servicios de energía. Este manual establece las políticas de seguridad obligatorias que protegen los datos personales (tabla usuarios: nombres, correos electrónicos, contraseñas hash) y facturas (tabla facturas: consumo, montos, estados) almacenadas en MariaDB.

21.2. ALCANCE

Aplicación a toda la infraestructura de LUZENERGIA: frontend, backend PHP, base de datos MariaDB, servidores y accesos de usuarios/administradores.

21.3. POLÍTICAS GENERALES DE SEGURIDAD

Objetivo

Establecer lineamientos generales para garantizar la protección de la información y reducir los riesgos de seguridad asociados a la operación de la página web.

Principios CID

- **Confidencialidad:** solo usuarios autorizados acceden a la información de las facturas.
- **Integridad:** los datos no se modifican de forma indebida o no autorizada.

- **Disponibilidad:** el sistema y los datos están accesibles cuando se necesitan. 99.9% optime con backups diarios.

Responsables

- **Administrador del sistema:** auditorías mensuales, gestión de usuarios, respuesta a incidentes.
- **Usuarios finales:** usar correctamente el sistema, proteger sus credenciales y reportar anomalías.
- **Equipo de desarrollo:** aplicar buenas prácticas de seguridad en el código.

Controles generales

- Uso obligatorio de HTTPS y certificados digitales válidos.
- Contraseñas almacenadas mediante algoritmos de hash seguros (por ejemplo, bcrypt o equivalente).
- Copias de seguridad automáticas de la base de datos con una retención mínima de 30 días.
- Revisión y actualización anual de las políticas de seguridad.

21.4. POLITICA DE TRATAMIENTO DE DATOS PERSONALES

Objetivo

Definir las condiciones bajo las cuales se recolectan, usan, almacenan y eliminan los datos personales de los usuarios de la plataforma.

Base jurídica

Ley 1581/2012 Colombia y Decreto 1377/2013

Datos tratados

CAMPO	TABLA	FINALIDAD
Nombre, email, password, numerocliente y notificaciones	Usuarios	Autenticación y notificaciones
Numerocliente, consumo, monto , estado	Facturas	Gestión facturación energía

Consentimiento

Checkbox obligatorio en /registro.php: "Acepto tratamiento datos para facturas LUZENERGIA".

Seguridad técnica

- Limitación de velocidad: 5 intentos de inicio de sesión → bloqueo 15 min.

- Registros auditables de accesos a /mis-facturas.php.

21.5. POLÍTICA DE HABEAS DATA

Objetivo

Regular el ejercicio de los derechos de los titulares de los datos personales (acceso, actualización, rectificación, supresión y oposición) sobre la información almacenada en la plataforma.

Derechos del titular: El usuario, como titular de los datos, tiene derecho a:

- Conocer qué datos personales se almacenan sobre él.
- Solicitar actualización o corrección de datos inexactos o incompletos.
- Solicitar la supresión de datos cuando considere que no se requieren para las finalidades autorizadas o se usen indebidamente.
- Oponerse a tratamientos específicos, salvo que exista obligación legal de mantenerlos.

Procedimiento para ejercer derechos

- El podrá presentar solicitudes a través de un formulario en la página del usuario, por correo electrónico oficial o por escrito físico si se define.

- La solicitud deberá incluir al menos: nombre completo, identificación, descripción clara de la petición y datos que desea consultar o modificar.
- El responsable confirmará la recepción de la solicitud y dará respuesta en los plazos legales (por ejemplo, hasta 15 días hábiles, dependiendo de la normativa aplicable).

Respuesta y decisiones

- Si la solicitud es procedente, se actualizarán, corregirán, suprimen o restringirán los datos según corresponda.
- Si no es posible acceder a lo solicitado, se explicarán de forma clara los motivos legales o técnicos.

21.6. POLÍTICA DE AUTENTIFICACIÓN Y CONTROL DE ACCESO

Objetivo

Asegurar que solo usuarios autorizados accedan al sistema ya la información acorde con su rol.

Reglas de contraseñas

- Longitud mínima (por ejemplo, 8 caracteres) y combinación de letras, números y caracteres especiales.
- Prohibido reutilizar contraseñas evidentes o relacionadas con el usuario (nombre, número de cliente, etc.).

- Renovación periódica de contraseña (por ejemplo, cada 90 días para administradores).

Control de sesiones

- Bloqueo automático de cuenta tras un número determinado de intentos fallidos de inicio de sesión.
- Cierre de sesión automático tras un periodo de inactividad.

Roles y privilegios

- Rol Usuario: solo puede acceder a su propia información y facturas.
- Rol Administrador: puede gestionar usuarios, revisar logs y administrar facturas, con registro de todas sus acciones críticas.

21.7. POLITICA DE GESTION DE INCIDENTES Y DE SEGURIDAD

Objetivo

Establecer un proceso para detectar, registrar, analizar y resolver incidentes de seguridad de la información.

Definición de incidente

Se considera incidente cualquier evento que afecte o pueda afectar la confidencialidad, integridad o disponibilidad de la información, como accesos no autorizados, pérdida de datos, malware o vulnerabilidades explotadas.

Proceso de gestión de incidentes

- **Detección:** identificación de comportamientos anómalos, alertas de monitoreo o informes de usuarios.
- **Registro:** documentación del incidente (fecha, hora, sistema afectado, descripción).
- **Contención:** medidas inmediatas para limitar el impacto (bloqueo de cuentas, aislamiento de servidores, etc.).
- **Erradicación y recuperación:** eliminación de la causa raíz, restauración desde copias de seguridad cuando sea necesario.
- **Lecciones aprendidas:** análisis posterior para mejorar controles y evitar recurrencias.

Comunicación

- Los usuarios dispondrán de un canal de reporte de incidentes (correo, formulario o teléfono).
- En caso de brecha de datos personales, se deberá notificar a los usuarios afectados y, si aplica, a la autoridad competente, dentro de los plazos legales.

21.8. POLÍTICA DE USO ACEPTABLE DE LA PLATAFORMA

Objetivo

Regular el uso adecuado de la página web y los recursos tecnológicos relacionados.

Conductas permitidas

- Usar la plataforma únicamente para la consulta y gestión legítima de facturas y datos asociados.
- Mantener actualizados los datos personales necesarios para la correcta prestación del servicio.

Conductas prohibidas

- Intentar acceder a cuentas, información o recursos de otros usuarios.
- Realizar ataques de fuerza bruta, inyección de código, exploración de vulnerabilidades o cualquier actividad de hacking no autorizada.
- Introducir malware o contenidos maliciosos en el sistema.
- Compartir credenciales de acceso con terceros.

Sanciones

- Advertencia escrita.
- Suspensión temporal de la cuenta.
- Cierre definitivo de la cuenta.
- Acciones legales cuando corresponda (Suspensión – Judicial “LEY 1273/2009 delitos informáticos”)

21.9. POLÍTICA DE SEGURIDAD DE BACKUPS

BASES DE DATOS LUZNERGIA – MariaDB

Objetivo

Garantizar la disponibilidad y recuperabilidad de la base de datos LUZENERGIA (tablas usuarios y facturas) mediante backups seguros, consistentes y auditables que permitan RPO <1 hora y RTO <4 horas.

Alcance

COMPONENTE	DESCRIPCIÓN	FRECUENCIA BACKUP
Base Principal	luzenergia (usuarios, facturas)	Diaria + Horaria
Logs MariaDB	general_log , slow_query_log	Semanal

Configuración	my.cnf , usuarios MySQL	Mensual
Aplicación Web	PHP files /var/www/luzenergia/	Diaria

Clasificación de Backups

TIPO	FRECUENCIA	RETENCIÓN	UBICACIÓN
Full	Diaria 02:00 AM	30 días	Local + Nube
Incremental	Cada 4 horas	7 días	Local
Transaccional	Contiguo (binlog)	72 horas	Local
Prueba Recuperación	Semanal	N/A	Entorno staging

Responsabilidades

ROL	RESPONSABILIDADES
DBA LUZENERGÍA	Configurar cron jobs, verificar integridad, rotación backups
Web Developer	Backup código PHP, verificar restauración app
Administrador Sistema	Monitoreo espacio disco, replicación nube
Auditor	Verificar cadena custodia backup trimestral

21.10. Revisión y Actualización de las Políticas

- Las políticas de este manual deberán revisarse al menos una vez al año o cuando se introduzcan cambios significativos en la plataforma, en la legislación aplicable o en los riesgos identificados.

- Toda modificación deberá quedar documentada indicando la fecha de actualización y la persona o área responsable.

Conclusión

La vulnerabilidad detectada permite:

Saltarse el login sin conocer credenciales reales.

Acceder al panel interno de la aplicación.

Potencialmente extraer o modificar datos almacenados.

Este tipo de falla corresponde a la categoría A03:2021 – Inyección, según el OWASP Top 10, y representa una de las vulnerabilidades más críticas dentro de aplicaciones web.