A logo for a company

Description automatically generated

**Ciberseguridad**

**Descripción de la situación:** Una maquina ha sido vulnerada (atacada), como analista de ciberseguridad encargado de restaurar y proteger un servidor crítico que ha sido comprometido en 4Geeks Academy. Se te analizará un servidor clave de la empresa, y tu tarea será restablecer su seguridad, corregir las vulnerabilidades explotadas y garantizar su funcionamiento óptimo. El ejercicio se divide en tres fases que pondrán a prueba tus habilidades en análisis forense, detección y corrección de vulnerabilidades, y respuesta ante incidentes.

**Índice**

* **Fase 1 - Corrección de un hackeo** Durante la primera fase, deberas realizar un análisis forense del incidente, identificar las vulnerabilidades explotadas por el atacante, y bloquear el exploit para evitar que el ataque escale.
* **Fase 2 - Detección y corrección de una nueva vulnerabilidad** En la segunda fase, se escanea el sistema en busca de una vulnerabilidad adicional, diferente a la explotada anteriormente. Tras detectarla, explota la vulnerabilidad de manera controlada para entender su impacto, escalar sus privilegios, corregirla, y crear un informe que explique todo el proceso.
* **Fase 3 - Corrección y Observaciones a las Vulnerabilidades** Esta fase implica la identificación, análisis y remediación de vulnerabilidades descubiertas en el sistema. Además, se documentan observaciones y lecciones aprendidas para mejorar la seguridad futura y evitar la reaparición de las mismas vulnerabilidades.
* **Fase 4 - Plan de respuesta a incidentes y certificación** La fase final consiste en diseñar un plan de respuesta a incidentes basado en las mejores prácticas de la industria, como las recomendaciones del NIST. Como parte de este ejercicio, debes desarrollar un Sistema de Gestión de Seguridad de la Información (SGSI) conforme a la norma ISO 27001, que incluya medidas para prevenir la fuga de datos mediante políticas de prevención de pérdida de datos (DLP).

**Fase 1: Análisis Forense**

**Objetivo:** Lleva a cabo un análisis forense para bloquear el exploit, corregir la vulnerabilidad y evitar que el atacante escale.

Para comenzar nuestro análisis forense necesitaremos realizar un mapeo, este análisis no realizaremos con nmap:

A screen shot of a computer

Description automatically generated

El escaneo revela algunos puertos abiertos en esta máquina que podrían tener riesgos o vulnerabilidades específicas. Aquí están los peligros más comunes asociados con cada uno de ellos:

**1. Puerto 21/tcp (FTP):**

Riesgos: El protocolo FTP (File Transfer Protocol) transmite información en texto plano, incluidas las credenciales de autenticación. Esto puede exponer contraseñas y datos a ataques de interceptación (sniffing).

Vulnerabilidades comunes: Algunos servidores FTP antiguos permiten ataques de fuerza bruta y carecen de configuraciones de seguridad como el uso de FTP seguro (FTPS). La configuración incorrecta o permisos de acceso inseguros también pueden permitir que atacantes accedan y modifiquen archivos en el servidor.

**2. Puerto 22/tcp (SSH):**

Riesgos: El protocolo SSH (Secure Shell) es generalmente seguro, pero si utiliza autenticación débil o contraseñas predecibles, puede ser susceptible a ataques de fuerza bruta.

Vulnerabilidades comunes: Un ataque exitoso en el puerto SSH podría proporcionar acceso directo al sistema, permitiendo al atacante ejecutar comandos y controlar la máquina. Esto es especialmente riesgoso si se permite el acceso como usuario root o si no hay restricciones de IP en el servicio SSH.

**3. Puerto 25/tcp (SMTP):**

Riesgos: Un servidor SMTP (Simple Mail Transfer Protocol) mal configurado podría ser explotado para realizar ataques de spam o phishing si permite el relay sin autenticación.

Vulnerabilidades comunes: Las configuraciones de SMTP inseguras pueden permitir a los atacantes enviar correos electrónicos maliciosos o realizar spoofing, lo que afecta la reputación del servidor. Además, los ataques de fuerza bruta en servicios SMTP pueden usarse para obtener credenciales válidas de correos.

**4. Puerto 80/tcp (HTTP):**

Riesgos: El puerto 80, que sirve contenido web sin cifrado, puede estar expuesto a ataques de interceptación, como ataques de "Man-in-the-Middle" (MitM).

Vulnerabilidades comunes: Las vulnerabilidades de HTTP incluyen inyecciones SQL, cross-site scripting (XSS), y problemas de configuración en la aplicación web. Sin HTTPS, el tráfico es más fácil de interceptar.

**5. Puerto 631/tcp (IPP):**

Riesgos: El protocolo IPP (Internet Printing Protocol) puede exponer configuraciones de impresoras y otros dispositivos de impresión, proporcionando información sensible sobre la red.

Vulnerabilidades comunes: Los servicios de impresión pueden ser explotados para capturar trabajos de impresión o explotar configuraciones de acceso no restringido. Además, algunos ataques pueden permitir el acceso a la red interna mediante exploits conocidos en servicios de impresión.

**6. Puerto 3306/tcp (MySQL/MariaDB):**

Riesgos: Un servicio MySQL expuesto sin configuración adecuada de firewall puede ser blanco de ataques de fuerza bruta para obtener acceso a la base de datos.

Vulnerabilidades comunes: Sin autenticación fuerte o configuración de usuario adecuada, un atacante puede acceder y manipular datos de la base de datos, o incluso ejecutar comandos maliciosos mediante inyecciones SQL.

**Medidas de mitigación:**

Control de acceso y autenticación: Usar contraseñas fuertes, autenticación de clave pública en SSH, y restringir acceso basado en IP para servicios sensibles.

Configuración segura: Deshabilitar servicios innecesarios, cifrar el tráfico (por ejemplo, usar HTTPS en lugar de HTTP), y configurar firewalls para limitar el acceso a puertos críticos como el de MySQL.

Monitoreo y actualizaciones: Monitorear los accesos y mantener actualizados todos los servicios para protegerse contra vulnerabilidades conocidas.

Mantener una política de seguridad sólida es esencial para reducir los riesgos y proteger los servicios en estos puertos.

Continuando en nuestro análisis forense y detección de vulnerabilidades realizaremos lo siguiente:

**Revisión de Conexiones SSH en /var/log/auth.log**

Comando para buscar intentos de acceso SSH fallidos:

**/var/log/auth.log**

El archivo ***auth.log*** no está disponible o ha sido eliminado (lo cual podría ser un indicio de actividad maliciosa), aún hay otros métodos y estrategias que puedes utilizar para realizar un análisis forense y tratar de identificar qué servicios fueron comprometidos y cómo el atacante pudo haber accedido al servidor. A continuación, te detallo varios enfoques:

Otro comando que se puede utilizar para buscar intentos fallidos es: ***/var/log/syslog,*** sin embargo como podemos ver en la siguiente imagen :

El archivo syslog no se encuentra.

**Logs específicos de aplicaciones**: Si estás ejecutando servicios como servidores web, bases de datos o servidores de correo, revisa sus registros:

* **Apache/Nginx**:

/var/log/apache2/access.log,

/var/log/nginx/access.log,

/var/log/apache2/error.log

A close up of words

Description automatically generated

En caso de no tener acceso a *apache2* intentar el siguiente paso, acceder a MariaDB.

**Identifica archivos sospechosos, procesos en ejecución y cualquier modificación inusual en el sistema.**

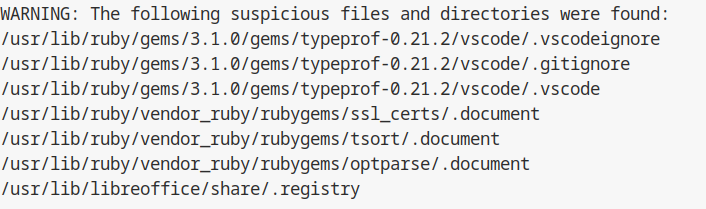
Se instala y ejecuta el programa chkrootkit para escanear el sistema en búsqueda de malwares y warnigs.



Resultados:



La aparición de "aliens" en el escaneo de **chkrootkit** no significa necesariamente que tu sistema esté infectado con malware, pero sí indica que hay archivos o procesos que podrían ser sospechosos.



Se analizará cada archivo y en caso de que se encuentre algún tipo de archivo malicioso se reportará en este documento.

***rkhunter*** es otra herramienta útil que revisa los rootkits, backdoors y exploits locales.





Después de ejecutar el escaneo los archivos saliron :

Hubo una excepción en el siguiente archivo:



Archivo no estándar o modificado

rkhunter verifica la integridad de ciertos archivos críticos del sistema. Si el archivo lwp-request ha sido modificado o no se encuentra en su estado original, esto puede generar un warning.

Nota: Ver en la sección de mitigación y correcciones lo suguerido para la organización para este archivo.

**Detener los servicios comprometidos temporalmente**

****

Este comando nos ayudara a detener un servicio que este posiblemente comprometido mientras continuamos con nuestro análisis de detección y corrección de vulnerabilidades

**Fase 2: Detecta y corrige una vulnerabilidad diferente**

**Análisis de usuarios y contraseñas en la Base de Datos MariaDB**

* **MySQL/MariaDB**: /var/log/mysql/error.log

A screenshot of a computer program

Description automatically generated

**Identificación de Usuarios con Contraseñas Débiles**

Revisar las contraseñas utilizadas: Por cuestiones de privacidad, MariaDB no guarda las contraseñas en texto plano, pero se puede verificar si hay contraseñas en blanco o sospechosamente cortas, que suelen ser indicadores de debilidad.

El siguiente comando busca usuarios que tienen contraseñas en blanco o de longitud inferior a 8 caracteres.



A white background with text

Description automatically generated

Como podemos ver en los resultados anterior, arrojan 2 usuarios con 8 o menos caracteres en su contraseña, esto no necesariamente es una vulnerabilidad **ALTA,** sin embargo, es una bandera amarilla u observación a la organización para fortalecer las reglas para al crear un usuario y contraseña (que sea mayor a 8 caracteres).

**Revisión de Usuarios sin Contraseña**: Verifica si hay cuentas que permiten inicio de sesión sin autenticación.

El siguiente comando muestra usuarios que tienen configurado mysql\_no\_login, que es el plugin que permite inicios de sesión sin contraseña.



A close up of a computer screen

Description automatically generated

Y como podemos visualizar, no arrojo algún usuario que permita iniciar sesión sin contraseña.

**Analizar Configuraciones de Seguridad Relacionadas**

Verificar Configuración de Autenticación: Es importante que el método de autenticación no esté configurado como *native\_password* sin encriptación fuerte.

Si el resultado es mysql\_native\_password sin soporte para hashing seguro, es recomendable cambiar a caching\_sha2\_password, que ofrece una encriptación más fuerte.

A white background with black text

Description automatically generated

Sin resultados.

Se puede consultar los hashes de las contraseñas con el siguiente comando, sin embargo, estos valores no son las contraseñas en texto plano, sino los hashes.



A screenshot of a computer screen

Description automatically generated

Como podemos ver en los resultados, hay 5 usuarios, uno de ellos sin contraseña y uno con contraseña invalida; la cual es una señal de alerta **ALTA.**

**Estas contraseñas se pueden traducir a texto plano y los resultados serían los siguientes:**

|  |  |  |
| --- | --- | --- |
| **Usuario** | **Password/Hash** | **Password/**  **Texto Plano** |
| **root** |  | **123456** |
| **wordpressuser** |  | **123456** |
| **user** |  | **password** |

Con base a los resultados, podemos concluir que el usuario ***“user”*** fue creado con una contraseña débil, ya que es predecible, no cumple con los estadandares básicos de seguridad como: Contraseña con al menos una mayúscula, algún número y si son varios números que no sean consecutivos y algún signo.

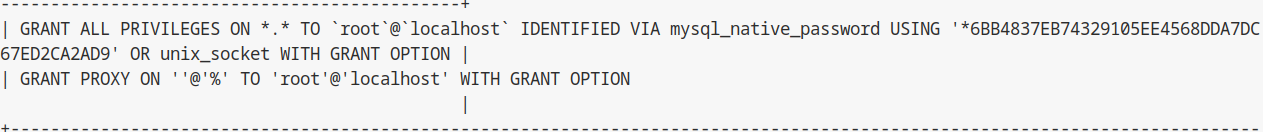
Esto plantea serios riesgos de ciberseguridad que pueden tener consecuencias graves para la integridad y confidencialidad de los datos. A continuación, algunos de los peligros principales:

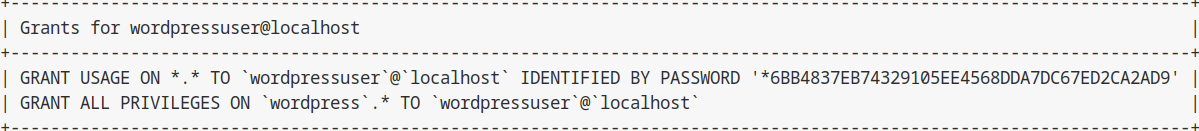
1. **Acceso no autorizado**: Las contraseñas débiles facilitan los ataques de fuerza bruta, donde los atacantes prueban múltiples combinaciones hasta encontrar la correcta. Una vez dentro, pueden realizar consultas, modificar datos o incluso eliminarlos, comprometiendo la seguridad y operación de la base de datos.
2. **Escalada de privilegios**: Si un usuario con contraseña débil tiene permisos elevados, el atacante podría tener control total de la base de datos. Esto puede permitirles realizar configuraciones maliciosas, cambiar permisos o crear usuarios adicionales con acceso privilegiado, facilitando ataques internos.
3. **Exposición a ataques de botnets**: Los ataques automatizados de botnets escanean constantemente en busca de cuentas con contraseñas débiles. Un atacante que obtenga acceso puede instalar malware, extraer datos sensibles o usar la infraestructura para otros ataques.
4. **Filtración de datos sensibles**: Una contraseña débil puede exponer información confidencial almacenada en la base de datos, lo que puede derivar en violaciones de privacidad y pérdida de reputación para la organización. Información como datos personales, financieros o propiedad intelectual podría filtrarse o ser vendida en el mercado negro.
5. **Puerta de entrada para ataques mayores**: Una base de datos comprometida con una contraseña débil podría permitir a los atacantes realizar movimientos laterales dentro de la red interna, accediendo a otros sistemas y comprometiendo así la infraestructura de TI completa.

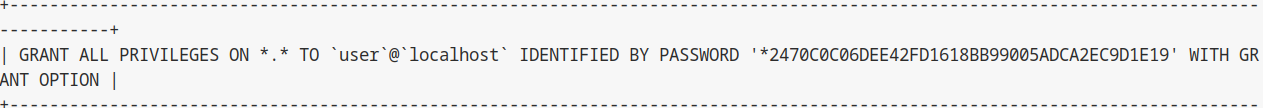
Para evitar estos peligros, es esencial implementar políticas de contraseñas robustas, usar autenticación multifactor (MFA), y realizar auditorías regulares en los sistemas de acceso a la base de datos. Estas medidas pueden reducir considerablemente el riesgo de compromisos de seguridad en la base de datos.

Adicionalmente podemos ver los privilegios de cada usuario, con el siguiente comando

Usuario ***”root”***



Usuario ***“wordpressuser”***

Usuario***“user”*** ******

Analizando los resultados sobre el usuario ***“user”*** podemos indentificar que cuenta con todos los privilegios, esto, sumando a la contraseña debil, es una alarta **ALTA,** que se debe reportar ya que vulnera Confidencialidad, Integridad y Disponibilidad de los datos de la organización.

Una vez que ya hayamos identificado las vulnerabilidades antes mencionadas, como, usuarios que comparten la misma contraseña, usuarios con todos los privilegios sin justificación y contraseñas débiles, podemos continuar en nuestra búsqueda vulnerabilidades o configuraciones atacadas, en este caso, verificaremos la configuración de FTP.

**Análisis Configuración Servidor FTP**

Este análisis es fundamental ya que en caso de haber sido vulnerado o estar mal configurado se puede, presentar accesos inseguros y el atacante puede tener disponibilidad a archivos y servicios dentro de la organización.

**Verificar Permisos de Archivos y Directorios**

* Comprobar que el FTP solo permita acceso de lectura y no de escritura en los directorios públicos, a menos que sea necesario. Si los usuarios pueden escribir en directorios públicos, podrían cargar archivos maliciosos.

A screenshot of a computer

Description automatically generated

**d**: Indica que es un directorio. Si fuera un archivo, sería un guion -.

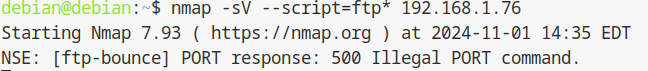
**rww**: El propietario (dueño del archivo) tiene permisos de lectura (r) y escritura (w). Hay un tercer w en lugar de una **x, lo cual es inusual en sistemas regulares y podría indicar un problema de configuración, ya que el permiso de ejecución normalmente sería necesario para que el propietario acceda al directorio.**

**r-x**: El grupo tiene permisos de lectura (r) y ejecución (x), pero no de escritura.

**r-x**: Los demás (otros usuarios) tienen permisos de lectura (r) y ejecución (x), pero no de escritura.

**2**: Indica el número de enlaces (links) al directorio.

**Escaneo de Vulnerabilidades del Servidor FTP**

****

Esto ejecutó un script de Nmap específico para FTP, el cual puede identificar versiones inseguras, configuraciones peligrosas y vulnerabilidades.

En este caso podemos observar que el FTP rechazan explícitamente el comando PORT para evitar ataques del tipo **FTP Bounce**. Este ataque aprovecha el comando PORT para realizar escaneos de puertos de otras redes o incluso atacar otros sistemas mediante el servidor FTP como intermediario.

Se puede continuar con la siguiente revisión.

**Análisis de Configuración de SSH**

Este análisis es importante ya que, en caso de estar mal configurado podrá permitir autentificación con passwords débiles o que no cumplen las normas basiscas de ciberseguridad.

A screenshot of a computer

Description automatically generated**Revisar los Métodos de Autenticación Permitidos**

Como podemos observar en la ejecución del comando /etc/ssh/ssh\_config, no cumple con una configuración segura ya que debería estar configurado con los siguientes puntos:

* **Clave pública (public key)** como el método preferido: PubkeyAuthentication yes.
* **Autenticación por contraseña**: PasswordAuthentication no, si es posible, para evitar contraseñas débiles.
* **Autenticación basada en desafío-respuesta**: ChallengeResponseAuthentication no

**Puertos Innecesarios Abiertos**

Como pudimos observar en el inicio de nuestro análisis forense realizando nuestro escaneo de puertos con ***nmap,*** nuestra maquina vulnerada cuenta con varios puertos abiertos incecesarios :

A screen shot of a computer

Description automatically generated

1. **Puerto 21 (FTP)**: FTP se usa para transferencias de archivos y puede ser inseguro porque transmite credenciales en texto plano. Si no necesitas FTP en la máquina, se recomienda deshabilitar este puerto, especialmente si tienes otros métodos de transferencia seguros, como SFTP o SCP.
2. **Puerto 22 (SSH)**: Este puerto se utiliza para conexiones seguras de administración remota. Es esencial si necesitas acceso remoto seguro al sistema, por lo que generalmente se mantiene abierto.
3. **Puerto 25 (SMTP)**: SMTP es el protocolo para el envío de correo electrónico. Si la máquina no es un servidor de correo o no necesita enviar correos, este puerto puede considerarse innecesario y debe cerrarse para reducir el riesgo de abuso.
4. **Puerto 80 (HTTP)**: Este puerto se usa para tráfico web HTTP no cifrado y debe estar abierto solo si la máquina aloja un servidor web accesible. Si el servidor no necesita recibir solicitudes HTTP, puedes cerrarlo y solo usar HTTPS en el puerto 443.
5. **Puerto 631 (IPP)**: IPP es para impresoras en red. Si tu máquina no está configurada para compartir impresoras o no está en un entorno que lo requiera, este puerto puede ser innecesario y se recomienda cerrarlo para evitar posibles ataques.
6. **Puerto 3306 (MySQL/MariaDB)**: Es el puerto estándar para bases de datos MySQL/MariaDB. Si la base de datos solo se usa internamente (no accesible externamente), puedes restringir este puerto para que solo acepte conexiones desde localhost o redes específicas, o incluso cerrarlo si no es esencial.

**Recomendación Resumida:**

* **Cerrar los puertos 21 (FTP), 25 (SMTP) y 631 (IPP)** si no son esenciales para las funciones de la máquina.
* **Restringir el puerto 3306 (MySQL/MariaDB)** solo a IPs necesarias si no debe ser accesible de forma amplia.

Reducir los puertos abiertos a solo los necesarios mejora significativamente la seguridad del sistema.

**Permisos Incorrectos en wp-config.php**

El análisis a este archivo es muy importante ya que en el alberga las credenciales críticas de la BD y posiblemente este expuesto a usuarios no autorizados dependiendo a exceso de permisos que no necesariamente necesita el usuario debido a sus funciones.

**Verificar Permisos del Archivo**

Los permisos seguros suelen ser 600 o 640 para evitar el acceso no autorizado.



Este comando muestra los permisos actuales del archivo en formato -rw-r--r--. Asegúrate de que solo el propietario tiene permisos de lectura y escritura.

* **r**: Permiso de lectura (read).
* **w**: Permiso de escritura (write).
* **x**: Permiso de ejecución (execute).

Esto significa que **cualquiera** (el propietario, los miembros del grupo y otros) puede leer, escribir y ejecutar el archivo o directorio. Estos permisos son considerados muy permisivos y pueden representar un riesgo de seguridad, ya que permiten que cualquier usuario realice cualquier acción en el archivo o directorio.

**Directorio Web listable**

El análisis a esta configuración nos permitirá si los directorios web han sido configurados correctamente, en caso contrario será una vulnerabilidad y el atacante puede aprovechar esta, para, explorar y acceder a archivos que deberían estar ocultos o resguardados.

**Revisar la configuración del servidor web**

Directorio de configuración: Generalmente, se encuentra en /etc/apache2/. Los archivos relevantes son apache2.conf, 000-default.conf, y otros archivos dentro de sites-enabled/.

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer

Description automatically generated



A screenshot of a computer screen

Description automatically generated

Como podemos observar en el archivo ***000-default.conf,*** esta mal configurado ya que en los usuarios se encuentras permisos mas allá de sus tareas.

**Fase 3- Corrección y Observaciones a las Vulnerabilidades**

**Observación y sugerencia a la organización**

El warning que aparece en el escaneo de rkhunter sobre /usr/bin/lwp-request generalmente indica que hay algo inusual o potencialmente sospechoso relacionado con ese archivo. Aquí hay algunas razones comunes por las que podrías ver un warning:

**1. Archivo no estándar o modificado**

rkhunter verifica la integridad de ciertos archivos críticos del sistema. Si el archivo lwp-request ha sido modificado o no se encuentra en su estado original, esto puede generar un warning.

**2. Permisos inusuales**

Si los permisos del archivo no son los esperados (por ejemplo, permisos excesivos), rkhunter podría señalarlo. Puedes verificar los permisos usando:

***ls -l /usr/bin/lwp-request***

**3. Versión de archivo**

rkhunter puede estar alertando sobre la versión de lwp-request. Si no coincide con la versión esperada de tu distribución o si parece que es una versión no oficial, eso puede causar un warning.

**4. Falsos positivos**

Es importante tener en cuenta que rkhunter puede dar falsos positivos. En algunos casos, los archivos legítimos pueden ser marcados como sospechosos debido a su comportamiento o configuración.

Pasos a seguir

1. Verifica la integridad del archivo:
   * + Compara el hash del archivo con una fuente confiable, si es posible. Puedes hacerlo con sha256sum:

***sha256sum /usr/bin/lwp-request***

1. Reinstala el paquete:
   * + Si crees que el archivo podría estar corrupto o modificado, considera reinstalar el paquete que proporciona lwp-request. En Debian, generalmente pertenece al paquete libwww-perl:

***sudo apt reinstall libwww-perl***

1. Verifica el archivo:
   * + Asegúrate de que el archivo lwp-request provenga de un repositorio oficial y no haya sido agregado manualmente.
2. Investiga si es un falso positivo:
   * + Puedes buscar el warning específico en línea para ver si ha sido reportado como un falso positivo.

**Corrección a la vulnerabilad en la Base de Datos MariDB**

Para corregir las vulnerabilidades relacionadas con usuarios y contraseñas en la base de datos MariaDB, seguir esta lista de acciones:

**1. Revisar y Actualizar Contraseñas de Usuarios:**

Cambia la contraseña del usuario con ID "user" que actualmente contiene "password". Usa una contraseña segura y única.

Cambia la contraseña del usuario root, reemplazando "123456" con una contraseña segura, única y compleja.

**En sql**

*ALTER USER 'user'@'localhost' IDENTIFIED BY 'NuevaContraseñaSegura!';*

*ALTER USER 'root'@'localhost' IDENTIFIED BY 'NuevaContraseñaSeguraParaRoot!';*

**2. Asignar Contraseñas Únicas a Todos los Usuarios:**

Asegúrarse de que cada usuario en el sistema tenga una contraseña única para evitar el uso compartido de credenciales.

**3. Eliminar o Restringir el Usuario con Privilegios Amplios:**

Revisa si el usuario "user" realmente necesita privilegios extensivos. De ser posible, revoca los permisos y asigna solo los mínimos necesarios.

**En sql**

*REVOKE ALL PRIVILEGES, GRANT OPTION FROM 'user'@'localhost';*

*GRANT SELECT, INSERT, UPDATE ON database\_name.\* TO 'user'@'localhost';*

**4. Eliminar Usuarios Innecesarios o de Prueba:**

Si el usuario con ID "user" es de prueba o no es necesario, considera eliminarlo completamente para reducir el riesgo.

**En sql**

*DROP USER 'user'@'localhost';*

**5. Auditar Usuarios y Privilegios:**

Revisa todos los usuarios de la base de datos y asegúrate de que solo los usuarios autorizados tengan acceso. Audita periódicamente.

**En sql**

SELECT user, host FROM mysql.user;

**6. Establecer Políticas de Contraseña Segura:**

Configura políticas de seguridad para las contraseñas, como longitud mínima, uso de caracteres especiales, y caducidad periódica.

**7. Registrar Actividad y Monitorear el Acceso:**

Implementa registros de auditoría para monitorear el acceso y las acciones de los usuarios, especialmente los que tienen altos privilegios.

**Bloquear posibles exploit y prevénir la escalación.**

**1.Mantener el sistema actualizado**

* + **Actualiza regularmente el sistema y los paquetes** para asegurar que estés protegido contra vulnerabilidades conocidas:

*sudo apt update && sudo apt upgrade*

**2. Configuración de un firewall**

* + **Utiliza ufw (Uncomplicated Firewall)** o iptables para restringir el tráfico de red. Configura reglas que permitan solo el tráfico necesario.

*sudo apt install ufw*

*sudo ufw enable*

*sudo ufw allow ssh # Permitir acceso SSH*

*sudo ufw allow http # Permitir acceso HTTP*

*sudo ufw allow https # Permitir acceso HTTPS*

**3. Desactivar servicios innecesarios**

* Revisa los servicios en ejecución y desactiva los que no necesites. Usa systemctl para administrar servicios.

*systemctl list-units --type=service*

*sudo systemctl stop nombre\_del\_servicio*

*sudo systemctl disable nombre\_del\_servicio*

**4. Usar SELinux o AppArmor**

* Considera habilitar **SELinux** o **AppArmor**, que son sistemas de control de acceso obligatorio que pueden ayudar a restringir las acciones de los procesos y los usuarios.

Para AppArmor:

*sudo apt install apparmor*

*sudo systemctl enable apparmor*

*sudo systemctl start apparmor*

**5. Configurar SSH de forma segura**

* **Deshabilitar el acceso de root** y permitir solo el acceso mediante claves SSH.
* Edita /etc/ssh/sshd\_config:

*PermitRootLogin no*

*PasswordAuthentication no* **# Asegúrate de usar autenticación con clave**

* **Cambia el puerto SSH** a un número diferente del 22 para dificultar ataques de fuerza bruta:

*Port 2222 # Cambia 2222 por un puerto no utilizado*

**6. Implementar límites de recursos**

* Usa ulimit o systemd para limitar los recursos que los procesos pueden usar (como CPU y memoria) y prevenir la escalación de privilegios a través de abusos de recursos.

**7. Monitorear registros del sistema**

* Revisa regularmente los registros del sistema en /var/log/, como auth.log y syslog, para detectar accesos no autorizados o actividades sospechosas.

**8. Deshabilitar SUID/SGID innecesarios**

* Revisa y elimina los permisos SUID/SGID de archivos que no lo necesiten. Usa el siguiente comando para encontrar archivos con estos permisos:

*find / -perm /6000 -type f 2>/dev/null*

**9. Instalar herramientas de seguridad**

* Utiliza herramientas como fail2ban para proteger el servidor contra ataques de fuerza bruta:

*sudo apt install fail2ban*

**10. Auditoría y escaneo de seguridad**

* Realiza auditorías regulares con herramientas como, **rkhunter**, y **chkrootkit** para identificar vulnerabilidades y asegurarte de que no haya rootkits.

**11. Aplicar el principio de privilegio mínimo**

* Asegúrate de que los usuarios y procesos solo tengan los permisos necesarios para realizar su trabajo. Esto ayuda a limitar el daño en caso de que una cuenta sea comprometida.

**12. Seguridad de aplicaciones web**

* Si ejecutas aplicaciones web, asegúrate de que estén actualizadas y configuradas correctamente. Implementa medidas de seguridad como WAF (Web Application Firewall) y sanitización de entradas.

**Mitigación a la Vulnerabilidad SSH**

Con este comando se implementa la biblioteca **libpam-pwquality** para exigir contraseñas fuertes. Puedes instalarla y configurarla con:

A screenshot of a computer

Description automatically generatedConfiguraremos políticas de contraseña y ajustaremos las opciones como minlen (longitud mínima) y retry (intentos fallidos permitidos).

Para mitigar futuros riesgos realizaremos lo siguiente:

**Deshabilitar el Acceso SSH para el Usuario Root**

* En el mismo archivo /etc/ssh/sshd\_config, asegúrate de que PermitRootLogin esté configurado como no. Esto reduce el riesgo de ataques de fuerza bruta dirigidos al usuario root: Con el comando ***PermitRootLogin no***

**Restricción de Permisos por Dirección IP**

* Configura restricciones para que solo direcciones IP específicas puedan acceder por SSH. Puedes usar AllowUsers o AllowGroups en el archivo ***/etc/ssh/sshd\_config*** para limitar accesos.

**Cerrar Puertos Innecesarios**

* **Cerrar los puertos 21 (FTP), 25 (SMTP) y 631 (IPP)** si no son esenciales para las funciones de la máquina.

**Bloquear el Puerto en el Firewall**

Se bloqueo el puerto 21 con los siguientes comandos





*Nota: Puede ser tan solo con solo uno de estos comandos*

* **Restringir el puerto 3306 (MySQL/MariaDB)** solo a IPs necesarias si no debe ser accesible de forma amplia.

**Configurar my.cnf para Permitir Solo Conexiones desde IPs Específicas**

Editar el archivo de configuración de MySQL/MariaDB (my.cnf o mariadb.conf.d/50-server.cnf):

***sudo nano /etc/mysql/my.cnf***

En la sección [mysqld], la organización debe asegurar que que la opción bind-address esté configurada solo para aceptar conexiones locales o de una IP específica:

***bind-address = 127.0.0.1***

Si la organizacipon desea permitir el acceso desde una IP externa específica (por ejemplo, 192.168.1.10), reemplaza 127.0.0.1 con la dirección IP deseada. Luego, reinicia el servicio:

***sudo systemctl restart mysql***

**2. Usar el Firewall para Limitar el Acceso**

Configura el firewall para permitir conexiones solo desde IPs específicas. Puedes usar ufw o iptables:

* **ufw**:

*sudo ufw allow from 192.168.1.10 to any port 3306*

* **iptables**:

*sudo iptables -A INPUT -p tcp -s 192.168.1.10 --dport 3306 -j ACCEPT*

*sudo iptables -A INPUT -p tcp --dport 3306 -j DROP*

**Mitigación de la Vulnerabilidad en el archivo wp-config.php**

1. Revisar y ajustar permisos:

* Cambia los permisos a un nivel más seguro usando el comando chmod. Por ejemplo, para un archivo o directorio que solo necesita ser accesible por el propietario, puedes establecer los permisos a 755 (para directorios) o 644 (para archivos):

***chmod 755 nombre\_del\_directorio # Para un directorio***

***chmod 644 nombre\_del\_archivo # Para un archivo***

* Esto otorgará al propietario permisos de lectura, escritura y ejecución, y al grupo y a otros solo permisos de lectura y ejecución (en el caso de directorios).

1. Ajustar la propiedad:

* Asegúrate de que el archivo o directorio pertenezca al usuario correcto y, si es necesario, al grupo correcto. Puedes cambiar la propiedad usando chown:

***sudo chown usuario:grupo nombre\_del\_archivo***

1. Utilizar un archivo .htaccess (si aplica):

* Si estás utilizando un servidor web como Apache, puedes usar un archivo .htaccess para restringir el acceso a ciertos archivos o directorios.

1. Deshabilitar la ejecución de scripts innecesarios:

* En algunos casos, puede ser útil deshabilitar la ejecución de scripts en directorios que no lo requieren. Esto se puede hacer a través de la configuración del servidor web.

1. Monitorear los permisos:

* Realiza auditorías periódicas de los permisos de archivos y directorios en tu servidor. Puedes crear un script que verifique los permisos y te alerte sobre configuraciones inseguras.

1. Aplicar el principio de privilegio mínimo:

* Asegúrate de que cada usuario, aplicación o proceso tenga solo los permisos necesarios para realizar su trabajo. Esto limita la exposición en caso de un ataque.

1. Usar herramientas de seguridad:

* Implementa herramientas de seguridad y monitoreo que te ayuden a detectar cambios no autorizados en los permisos o archivos críticos.

Siguiendo estos pasos, puedes mejorar la seguridad de tu sistema y mitigar las vulnerabilidades asociadas con permisos excesivos.

**Corregir la configuración web listables**

* **Configuraciones clave**:

**Deshabilitar el listado de directorios**: Asegúrate de que no esté habilitado el listado de directorios. Busca la línea Options en la configuración de los directorios:

***<Directory /var/www/html>***

***Options -Indexes***

***</Directory>***

* **Restringir acceso a archivos sensibles**: Puedes usar reglas como estas para proteger archivos como wp-config.php o .htaccess:

***<Files wp-config.php>***

***Order Allow,Deny***

***Deny from all***

***</Files>***

**Nginx**

Si utilizas Nginx, revisa la configuración en:

* **Directorio de configuración**: Generalmente en /etc/nginx/. Los archivos relevantes son nginx.conf y archivos dentro de sites-enabled/.
* **Configuraciones clave**:

Deshabilitar el listado de directorios:

apache

***location / {***

***autoindex off;***

***}***

Proteger archivos sensibles:

nginx

***location ~\* wp-config.php {***

***deny all;***

***}***

**2. Revisar permisos de archivos y directorios**

Verifica que los permisos de los archivos y directorios sean apropiados, como se mencionó anteriormente. Usa ls -l para revisar los permisos en el directorio raíz de la instalación de tu aplicación web.

**3. Utilizar un firewall**

Configura un firewall (como ufw o iptables) para restringir el acceso a puertos no esenciales y proteger tu servidor de ataques externos.

* Para habilitar ufw:

***sudo ufw allow 'Apache Full' # o 'Nginx Full'***

***sudo ufw enable***

**4. Auditar el uso de módulos**

Revisa los módulos del servidor web habilitados para asegurarte de que no haya módulos innecesarios que puedan presentar vulnerabilidades. Por ejemplo, en Apache, puedes listar módulos habilitados con:

***apache2ctl -M***

Deshabilita cualquier módulo innecesario.

**5. Configuración de PHP**

Si tu aplicación web utiliza PHP, asegúrate de que esté configurado de manera segura:

* **Deshabilitar funciones peligrosas**: Edita el archivo php.ini y desactiva funciones potencialmente peligrosas como exec, shell\_exec, system, etc.:

***ini***

***disable\_functions = exec,shell\_exec,system***

* **Evitar la divulgación de información**: Configura display\_errors a Off en el php.ini para evitar que se muestren errores a los usuarios.

**6. Verificar registros de acceso y error**

Revisa los registros de acceso y error del servidor web:

* **Apache**: Los registros suelen encontrarse en /var/log/apache2/access.log y /var/log/apache2/error.log.
* **Nginx**: Los registros suelen encontrarse en /var/log/nginx/access.log y /var/log/nginx/error.log.

Busca patrones inusuales o intentos de acceso no autorizados.

**7. Escaneo de seguridad**

Utiliza herramientas de escaneo de seguridad como **Nikto**, **OpenVAS** o **Nmap** para identificar vulnerabilidades potenciales en tu configuración web.

**8. Mantener el software actualizado**

Asegúrate de que tu servidor web, sistema operativo y cualquier software utilizado (como CMS) estén actualizados para protegerte contra vulnerabilidades conocidas.

Siguiendo estos pasos, podrás investigar y asegurar la configuración de tu servidor web en Debian para reducir el riesgo de accesos no autorizados a tus archivos.

**Fase 4: Plan de respuesta de incidentes y certificación**

**Plan de Respuesta a Incidentes Basado en la Guía del NIST SP 800-61**

Este plan define el ciclo completo de respuesta a incidentes en una organización, basado en el marco del \*\*NIST SP 800-61\*\* y orientado a incidentes como el hackeo analizado en este documento. Abarca las fases de identificación, contención, erradicación y recuperación para mitigar vulnerabilidades y prevenir ataques futuros.

**1. Preparación**

Capacitación: Capacitar al personal de TI y usuarios clave sobre procedimientos de respuesta y buenas prácticas de seguridad.

Herramientas de Seguridad: Implementar y configurar herramientas de detección de intrusos (IDS/IPS), antivirus, firewalls, y sistemas de monitoreo para detectar actividades anómalas en tiempo real.

Inventario de Activos: Documentar y mantener actualizado un inventario de sistemas, aplicaciones y usuarios. Asegurarse de que los permisos y roles estén bien definidos.

Copia de Seguridad y Cifrado: Realizar respaldos de información crítica y cifrar datos sensibles. Esto garantiza la recuperación de datos en caso de un incidente.

**2. Identificación del Incidente**

Detección y Monitoreo: Implementar monitoreo continuo con alertas para detectar accesos no autorizados o cambios en la configuración del sistema.

Registro de Actividades: Establecer un sistema de auditoría de logs para los servicios críticos (por ejemplo, acceso a bases de datos) que permita detectar actividades anómalas.

Análisis de Amenazas: Clasificar amenazas y realizar análisis de comportamiento para determinar el origen y alcance de los incidentes.

**3. Contención del Incidente**

Contención Temporal: Detener el servicio afectado temporalmente para limitar la propagación (por ejemplo, detener servicios comprometidos usando `systemctl stop`).

Aislamiento de Sistemas: Aislar las máquinas afectadas de la red para evitar que el atacante acceda a otros sistemas.

Refuerzo de Contraseñas: Cambiar contraseñas de usuarios con privilegios elevados y auditar cuentas que compartan contraseñas.

Aplicación de Parcheo Rápido: Si hay actualizaciones de seguridad o parches disponibles para los sistemas comprometidos, aplicarlos de inmediato.

**4. Erradicación**

Remover Malware: Escanear los sistemas comprometidos y eliminar cualquier software malicioso o archivos sospechosos encontrados.

Corrección de Configuraciones\*\*: Corregir configuraciones inseguras en los sistemas afectados, como deshabilitar usuarios innecesarios, ajustar permisos y revisar contraseñas.

Revisión de Cuentas: Auditar cuentas de usuario en bases de datos y sistemas críticos, eliminando usuarios que ya no son necesarios y aplicando controles de acceso adecuados.

Monitoreo de Seguridad: Establecer un monitoreo avanzado de seguridad en los sistemas previamente comprometidos.

**5. Recuperación**

Restablecimiento de Sistemas: Una vez asegurado el sistema, reiniciar servicios y restaurar sistemas a un estado seguro, utilizando respaldos cuando sea necesario.

Revisión de Configuraciones: Verificar que los cambios realizados durante la erradicación sean permanentes y que se apliquen políticas de acceso mínimo.

Revisión de Registros de Auditoría: Monitorear el sistema tras el incidente para detectar cualquier intento de acceso adicional o comportamiento sospechoso.

Pruebas de Integridad: Realizar pruebas de funcionalidad y seguridad para asegurar que los servicios operen sin comprometer la seguridad.

**Mecanismos de Protección de Datos**

1. Respaldos Periódicos: Realizar respaldos completos y programados de la información crítica, almacenándolos en ubicaciones seguras (tanto en la nube como en sitios físicos).

2. Cifrado de Datos Sensibles: Implementar cifrado en reposo y en tránsito para proteger datos críticos y sensibles, garantizando que solo usuarios autorizados puedan acceder a la información.

3. Controles de Acceso: Establecer un sistema de control de acceso estricto, aplicando autenticación multifactor (MFA) para usuarios privilegiados y siguiendo el principio de acceso mínimo (least privilege).

4. Políticas de Contraseñas Seguras: Implementar políticas que requieran contraseñas complejas y únicas para cada usuario, así como la actualización periódica de estas.

**Implementación del Sistema de Gestión de Seguridad de la Información (SGSI) – ISO 27001**

**1. Análisis de Riesgos:**

Realizar un análisis de riesgos para identificar, evaluar y priorizar amenazas que puedan afectar la seguridad de la información.

Definir las probabilidades de ocurrencia y el impacto de cada amenaza para establecer controles y medidas de mitigación.

**2. Definición de Políticas de Seguridad:**

Crear políticas de seguridad de la información que abarquen control de acceso, gestión de contraseñas, respuesta a incidentes, protección de datos y privacidad.

Comunicar estas políticas a todo el personal para asegurar el cumplimiento y fomentar la cultura de seguridad.

**3. Planes de Acción y Medidas de Protección:**

Desarrollar planes de acción específicos que incluyan la implementación de controles técnicos y administrativos para proteger la información crítica.

Incluir acciones preventivas como la realización de auditorías internas, análisis de vulnerabilidades y evaluaciones de seguridad periódicas.

**4. Evaluación y Mejora Continua:**

Monitorear y evaluar regularmente el SGSI para identificar mejoras y ajustes necesarios. Realizar auditorías internas y auditorías de terceros según la norma ISO 27001.

Implementar una política de mejora continua basada en los resultados de los análisis de riesgos y auditorías, para mantener la seguridad y cumplir con los estándares.

**Conclusión**

Este plan de respuesta a incidentes basado en NIST SP 800-61, junto con la implementación de un SGSI bajo ISO 27001, garantiza que la organización esté preparada para identificar, contener, erradicar y recuperarse de incidentes de seguridad, además de aplicar mecanismos de protección de datos y políticas de seguridad continuas para prevenir futuros ataques.