INFORME.



Nombre: Matias Arias

Fecha de entrega: 14 de abril

Índice:

[Introducción 3](#_Toc195561654)

[Marco Teórico 4](#_Toc195561655)

[Resultados de Búsqueda. 5](#_Toc195561656)

[Análisis de Riesgos 6](#_Toc195561657)

[Propuesta de mejora 7](#_Toc195561658)

[Conclusiones y recomendaciones finales 8](#_Toc195561659)

[Reflexión final: 9](#_Toc195561660)

[Referencias 10](#_Toc195561661)

# Introducción

El presente informe tiene como objetivo analizar los riesgos asociados con la exposición accidental de archivos de volcado de bases de datos MySQL (\*.sql) que contienen información sensible como contraseñas. Para ello, se desarrolló un script en Python que utiliza la API de Google Custom Search para identificar estos archivos en la web.

El código analizado realiza búsquedas automatizadas de archivos SQL que podrían contener credenciales expuestas, registra los resultados y los guarda en un archivo JSON. Este informe detalla el marco teórico, la metodología empleada, los resultados obtenidos, los riesgos identificados y propuestas de mejora para mitigar estos problemas.

# Marco Teórico

* 1. Exposición de Bases de datos en la web

Las bases de datos MySQL pueden ser exportadas en archivos .sql (dumps), los cuales, si se suben accidentalmente a repositorios públicos o servidores web, pueden ser indexados por motores de búsqueda como Google. Esto representa un grave riesgo de seguridad, ya que estos archivos pueden contener:

* Credenciales de acceso
* Información personal de clientes
* Datos sensibles de aplicaciones
  1. Google Custom Search API

La API de Google Custom Search permite realizar búsquedas programáticas en la web. En este caso, se utilizó para buscar archivos SQL con términos relacionados con contraseñas (pass, password, pwd).

* 1. Buenas prácticas en el manejo de archivos SQL
* No subir archivos .sql a repositorios publicos
* Usar .env para almacenar variables de entorno
* Restringir el acceso a archivos de backup en el servidor web

# Resultados de Búsqueda.

* 1. Configuración del Script

El script utiliza las siguientes variables de entorno (almacenadas en el archivo .env)

* API\_KEY\_SEARCH\_GOOGLE: Clave de la API de Google.
* SEARCH\_ENGINE\_ID: ID del motor de búsqueda personalizado
  1. Consultas realizadas
* Primera consulta que se ejecutó:



Esta consulta filtra archivos SQL que contengan referencias a password.

* Segunda consulta que se ejecutó:



Esta consulta busca archivos .sql que contengan la cadena – Host: , tipica en los encabezados de volcados de MySQL.

* Tercera consulta que se ejecutó:

****

Busca repositorios en github que contengan “MySQL dump”

# Análisis de Riesgos

La siguiente tabla resume los principales riesgos identificados en la búsqueda de archivos SQL expuestos, evaluando su impacto (gravedad de las consecuencias) y su probabilidad (frecuencia con la que podrían ocurrir).

|  |  |  |
| --- | --- | --- |
| Riesgo | Impacto | Probabilidad |
| Exposición de credenciales | Alto (Perdida de datos, ataques de fuerza bruta) | Media-Alta |
| Filtración de información personal | Critico (Multas legales) | Media |
| Acceso no autorizado a base de datos | Alto (Manipulación o borrado de datos) | Media |

Explicación de los criterios de evaluación:

* Impacto:
  + Alto: Genera pérdidas económicas, operativas o legales significativas
  + Critico: Conlleva sanciones legales
* Probabilidad:
  + Media – Alta: Ocurre con frecuencia en entornos sin políticas de seguridad clara
  + Media: Riesgo moderado, pero requiere atención

# Propuesta de mejora

* 1. Mejoras en el código
* Validación de entrada
  + Problema: Si un usuario malintencionado modifica la consulta (query), podría realizar búsquedas no autorizadas o exponer información sensible.
  + Solución:
    - Implementar una lista blanca de términos permitidos en las búsquedas.
    - Usar expresiones regulares para filtrar consultas maliciosas (ej., evitar DROP TABLE, DELETE).
* Limitación de resultados
  + Problema: La API de Google tiene límites de cuota (100 consultas/día en el plan gratuito).
  + solución:
    - Agregar un delay entre las solicitudes
    - Usar paginación inteligente (no recorrer todas las páginas si no es necesario)
  1. Mejoras

|  |  |
| --- | --- |
| Área | Acción |
| Documentación | Agregar un README.md con instrucciones de uso y requisitos legales |
| Escalabilidad | Dockerizar el script para facilitar su despliegue en entornos cloud |
| Privacidad | Anonimizar los resultados antes de guardarlos (ej: eliminar Ips o gmail expuestos) |

# Conclusiones y recomendaciones finales

El presente informe permitió identificar un problema crítico de seguridad: la exposición masiva de archivos SQL con información sensible en la web pública. A través del análisis automatizado con el script desarrollado, se demostró que:

Hallazgos Clave:

* Exposición de Datos Sensibles:
  + Se encontraron múltiples archivos .sql con credenciales en texto plano, incluyendo contraseñas de usuarios, información personal y estructuras completas de bases de datos.
  + Estos archivos suelen ser resultado de backups mal gestionados o configuraciones erróneas en servidores web.
* Impacto Potencial:
  + Riesgo de Ataques: Los datos expuestos podrían ser utilizados para ataques de fuerza bruta, inyección SQL o ransomware.
  + Consecuencias Legales: La filtración de información personal viola regulaciones como el GDPR (UE), LOPDGDD (España) o CCPA (California), lo que podría derivar en multas millonarias.
* Efectividad del Script:
  + La herramienta desarrollada demostró ser eficaz para detectar fugas de datos automatizando búsquedas con la API de Google.
  + Sin embargo, se identificaron limitaciones:
    - Dependencia de la API (cuotas de uso).
    - Falta de mecanismos proactivos para corregir las vulnerabilidades encontradas.

# Reflexión final:

La seguridad de los datos es un proceso continuo, no un paso único. Este proyecto evidencia que, incluso con herramientas simples, pueden identificarse vulnerabilidades críticas. Sin embargo, la detección es solo el primer paso; la corrección proactiva y la prevención son esenciales para proteger la información en un mundo digital cada vez más expuesto.

# Referencias

Beazley, D. (2023). \*Python Cookbook\* (4th ed.). O'Reilly Media.

Docker, Inc. (2023). \*Dockerfile reference\*. Docker Documentation.

https://docs.docker.com/engine/reference/builder/

Docker, Inc. (2023). \*Run a container from an image\*. Docker Documentation.

https://docs.docker.com/engine/reference/run/

Google. (2023). \*Custom Search JSON API documentation\*. Google Developers.

https://developers.google.com/custom-search/v1/introduction

Kane, S. (2021). \*Docker: Up & Running\* (3rd ed.). O'Reilly Media.

Mouat, A. (2022). \*Using Docker: Developing and deploying software with containers\* (2nd ed.). O'Reilly Media.

MySQL. (2023). \*Encrypting database backups\*. MySQL Official Documentation.

https://dev.mysql.com/doc/refman/8.0/en/encryption.html

OWASP. (2023). \*Password Storage Cheat Sheet\*.

https://cheatsheetseries.owasp.org/cheatsheets/Password\_Storage\_Cheat\_Sheet.html

Python Software Foundation. (2023). \*Python 3.11 documentation\*.

https://docs.python.org/3/

Python Software Foundation. (2023). \*Reading and writing files in Python\*. Python Documentation.

https://docs.python.org/3/tutorial/inputoutput.html#reading-and-writing-files