



Tecnológico Nacional de México

Instituto Tecnológico de Tlaxiaco

Carrera: Ingeniería en Sistemas Computacionales

Materia: Seguridad Y Virtualización

Actividad: Reporte de Practica 4

Alumnos:	Feria Ortiz Eduardo Tomas	21620095
	Reyes Peña Isaí	21620053
	Zárate Reyes Irving	20620166

Grupo: 6US

Catedrático: Ing. Osorio Salinas Edward

Heroica Ciudad de Tlaxiaco.
Miércoles, 02 de septiembre de 2024.





Índice

Práctica. Inyección SQL	3
1. Configuración del Entorno	3
2. Creación de la Tabla para Pruebas	3
3. Creación de la Aplicación Web Vulnerable	5
4. Pruebas de Inyección SQL	5
5. Resultados Obtenidos	7
Investigación	8
Inyección de SQL	8
Inyección SQL Ciega	8
Inyección SQL Basada en Errores	9
Inyección SQL Basada en Tiempo	9
Inyección SQL en Procedimientos Almacenados	9
Inyección SQL en ORM	10
Herramientas para Detectar y Prevenir SQL Injection	10
Bibliografía:	11





Práctica. Inyección SQL

1. Configuración del Entorno

Para comenzar, iniciamos el Panel de Control de XAMPP y activamos los servicios Apache y MySQL.

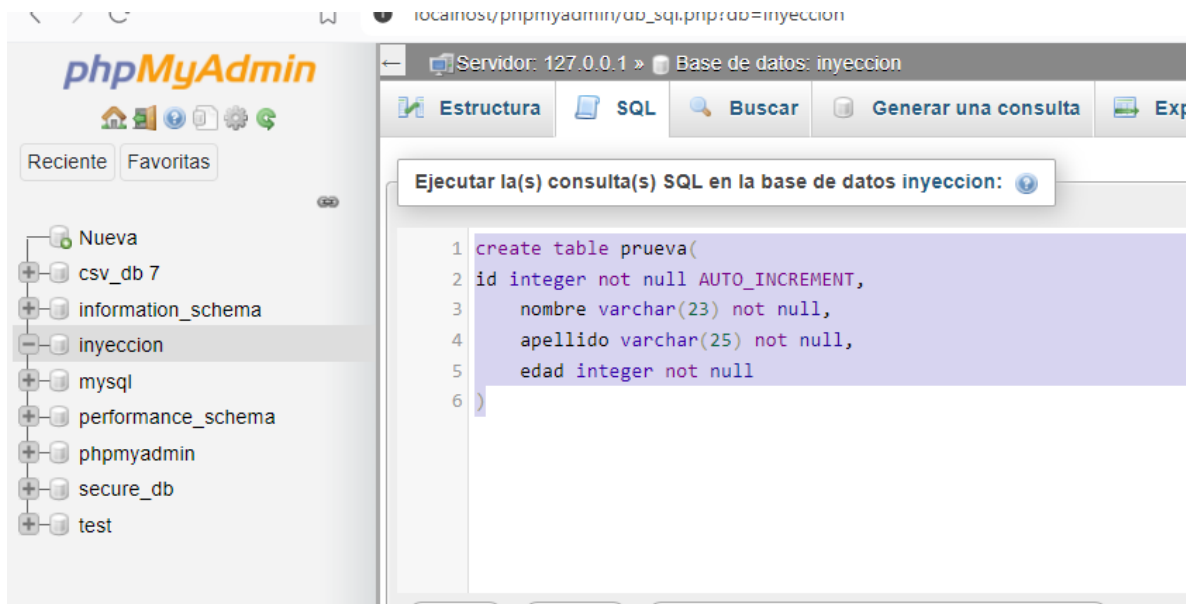
Abrimos PHPMyAdmin desde tu navegador mediante la URL:
<http://localhost/phpmyadmin>.

2. Creación de la Tabla para Pruebas

Creación de la Tabla: Creamos una tabla llamada prueba con los siguientes atributos:

- id integer not null Auto_Increment,
- nombre varchar (23) not null,
- apellido varchar (25) not null,
- edad integre not null,



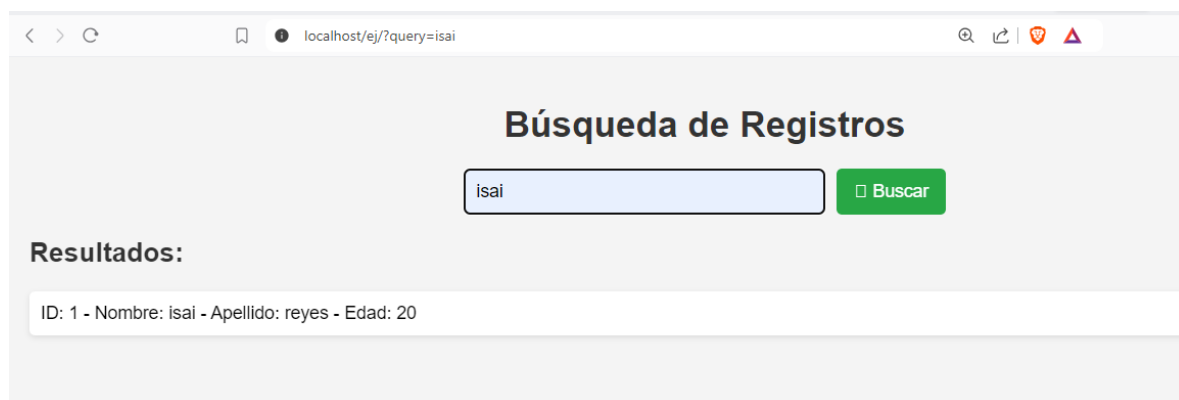


Insertión de Datos: Para realizar las pruebas de practica insertamos dos registros en la tabla.



3. Creación de la Aplicación Web Vulnerable

Se creó una aplicación PHP que permite realizar búsquedas en la tabla prueba de la base de datos. Esta aplicación es vulnerable a la inyección SQL, ya que no valida las entradas del usuario.



4. Pruebas de Inyección SQL

Después de implementar la aplicación, se realizaron pruebas de inyección SQL, mostrando en las capturas los siguientes resultados:

Consulta básica:

Al buscar 1 en el formulario de búsqueda, se devuelve el registro con id = 1, lo que es un comportamiento esperado.



localhost/ej/?query=isai

Búsqueda de Registros - localhost/ej/?query=isai

localhost/ej/?query=isai - Búsqueda de Brave

eduardo

Buscar

Resultados:

ID: 3 - Nombre: eduardo - Apellido: feria - Edad: 21

Inyección SQL básica:

Se ingresó la consulta `1 OR 1=1 --` en el campo de búsqueda, lo que devolvió todos los registros de la tabla. Esto se debe a que la condición `1=1` siempre es verdadera y anula la lógica de la consulta original.

localhost/ej/?query=isai

Búsqueda de Registros

Buscar por ID, nombre o apellido

Buscar

Resultados:

ID: 1 - Nombre: isai - Apellido: reyes - Edad: 20

Inyección SQL destructiva:

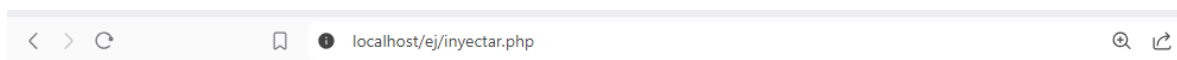
Para probar una inyección SQL que podría eliminar la tabla, se ingresó `' OR 1=1; DROP TABLE prueba; --`.



5. Resultados Obtenidos

En las pruebas de inyección SQL, los resultados muestran cómo se puede acceder a datos no autorizados e incluso manipular o eliminar tablas completas:

- **Inyección simple (1 OR 1=1 --):** Permite visualizar todos los registros de la base de datos, violando las restricciones de acceso.
- **Inyección destructiva (' OR 1=1; DROP TABLE productos; --):** Si se ejecutara con permisos suficientes, podría eliminar tablas y destruir datos.



☐ Mostrar todo | Número de filas: 25 ▼ | Filtrar filas: Sort

+ Opciones

				id	nombre	apellido	edad
<input type="checkbox"/>		Editar		Copiar		Borrar	1 isai reyes 20
<input type="checkbox"/>		Editar		Copiar		Borrar	2 irving zarate 24
<input type="checkbox"/>		Editar		Copiar		Borrar	3 eduardo feria 21
<input type="checkbox"/>		Editar		Copiar		Borrar	4 Juan Pérez 25



Investigación

Investiga y describe los siguientes conceptos:

Inyección de SQL

La inyección de SQL es una técnica de ataque donde un actor malintencionado inserta o "inyecta" código SQL malicioso en una consulta a la base de datos a través de entradas de usuario no validadas. Esto puede llevar a la exposición, alteración o eliminación de datos confidenciales. Este tipo de ataque ocurre cuando las consultas SQL no están adecuadamente protegidas, permitiendo que los atacantes manipulen la base de datos mediante comandos inesperados. Un ejemplo típico es cuando un campo de entrada acepta un ' OR '1'='1', que puede engañar al sistema y ejecutar consultas indeseadas.

Inyección SQL Ciega

La inyección SQL ciega ocurre cuando la base de datos no devuelve mensajes de error al atacante, pero las consultas siguen ejecutándose. En este caso, los atacantes utilizan técnicas para obtener información valiosa de la base de datos de manera indirecta, haciendo preguntas que solo pueden tener respuestas verdaderas o falsas. Aunque no reciben resultados visibles, los tiempos de respuesta o la falta de respuesta pueden revelar detalles importantes.





Inyección SQL Basada en Errores

Este tipo de inyección se aprovecha de los mensajes de error devueltos por la base de datos. Si un sistema no está configurado para ocultar estos mensajes, los atacantes pueden usarlos para obtener información sobre la estructura de la base de datos, como nombres de tablas y columnas, lo que facilita ataques más dirigidos.

Inyección SQL Basada en Tiempo

En este tipo de ataque, los atacantes manipulan consultas SQL de manera que el servidor tarde más tiempo en responder si una condición es verdadera o falsa. Esto les permite extraer información valiosa sin necesitar mensajes de error. Este enfoque es común cuando la aplicación no proporciona mensajes útiles o visibles en pantalla.

Inyección SQL en Procedimientos Almacenados

Los procedimientos almacenados pueden ser vulnerables a la inyección de SQL si no están bien diseñados. Aunque encapsulan consultas SQL, si reciben parámetros que no son validados o saneados adecuadamente, pueden ejecutar comandos maliciosos. Para prevenir esto, es recomendable utilizar sentencias preparadas dentro de los procedimientos almacenados.





Inyección SQL en ORM

Un ORM (Object-Relational Mapping) convierte el código de alto nivel en consultas SQL, facilitando la interacción entre la base de datos y el código de la aplicación. Sin embargo, si no se utilizan consultas preparadas o validaciones adecuadas, el ORM también puede ser vulnerable a la inyección de SQL. Aunque los ORMs ayudan a reducir los riesgos, no son inmunes si se permiten entradas de usuario no seguras.

Herramientas para Detectar y Prevenir SQL Injection

Existen varias herramientas para detectar y prevenir inyecciones SQL. Algunas de las más destacadas incluyen:

- **OWASP ZAP:** Una herramienta de código abierto que permite realizar pruebas de seguridad, incluida la detección de vulnerabilidades de inyección SQL.
- **Acunetix:** Un escáner comercial que automatiza el análisis de vulnerabilidades en aplicaciones web, incluida la inyección SQL.
- **Nikto:** Un escáner web que detecta configuraciones inseguras y vulnerabilidades conocidas, incluidas las relacionadas con SQL Injection.





Bibliografía:

- ataques_a_BB_DD_SQL_injection-libre.pdf. (s/f). Recuperado el 2 de octubre de 2024, de https://dlwqtxtslxzle7.cloudfront.net/43922194/ataques_a_BB_DD_SQL_injection-libre.pdf?1458487541=&response-content-disposition=inline%3B+filename%3DAtaques_a_BB_DD_SQL_Injection.pdf&Expires=1727929294&Signature=A-jZa08gcgqDBL9WeG09Ah7J9tvGU6n03NELT0pZ0HITcy7YK09PJ0NxmNkv2PrJiEllnUJtTcpgRC9s4tCrZD6~HBZzvG5WHJfgN5ZOIQU92yR9eQ5XrPm0ouhi6Euo2s2XrcCbBTaG2eap6tA53o43vP6EIIQI5nvoRslEP6xTSuknxzqfsNaL-1XnE5RsRp2TLrWpzZYb~CSyQ5qXKx6s7NN42GCfdi-7k05ka3td~eKWOFbcQ7UDQcVlzAK3fZPD5b80zxwwAGM-888awyjeObHdxqU9XJ4~MATNCu9~7joO2vzHDI3JWqFAWtFIEGTf7Dh9rLkoUobuHObg__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA
- Chicaiza, G., Ponce, L., & Campos, G. V. (s/f). INYECCIÓN DE SQL, CASO DE ESTUDIO OWASP.
- Gayol, P. F. (s/f). Estudio de los Principales Tipos de Ataques por Inyección de Código a Aplicaciones Web y Sistema para Determinar si un Código Fuente es Vulnerable a SQL Injection.
- Kulkarni, M. A. (s/f). President, Yashaswi Education Society.
- LEC ING SIST 0155 2020.pdf. (s/f). Recuperado el 2 de octubre de 2024, de <https://repositorio.usam.ac.cr/xmlui/bitstream/handle/11506/2417/LEC%20ING%20SIST%200155%202020.pdf?sequence=1&isAllowed=y>
- Rojas, E. (s/f). DETECCIÓN DE SQL INJECTION BASADA EN RANDOM FOREST.

