UNIVERSIDAD DEL ISTMO

FACULTAD DE INGENIERÍA

SEGURIDAD

SQL INJECTION

JAVIER ALEJANDRO LÓPEZ GUZMÁN

Guatemala, 12 de agosto de 2022

Contenido

Introducción

El robo de datos se ha convertido en un negocio redondo en los últimos tiempos ya que la información es realmente valiosa para todos. Existen miles de bases de datos alrededor del mundo las cuales pueden tener información realmente crítica sobre nosotros u otros individuos, por lo que el saber mantener nuestros datos protegidos es algo esencial en el siglo XXI.

Nuestra misión como desarrolladores es la creación de sistemas seguros en los que nuestros clientes puedan confiar e ingresar sus datos sin miedo a que estos lleguen a ser expuestos con el tiempo.

Uno de los métodos más fáciles y utilizados para la manipulación de datos en una base de datos es la práctica de *SQL Injection,* la cual se basa en la inyección de código en las consultas realizadas a las bases de datos del sistema por medio de inputs vulnerables los cuales no tienen métodos de seguridad para que estos querys maliciosos no tengan acceso a nuestra base de datos.

La implementación de buenas practicas y otros factores que se mencionaran en este documento es de suma importancia para mantener nuestros sistemas protegidos de este tipo de ataques, por lo que desde el inicio de la creación de estos sistemas, debemos de tomar como prioridad la seguridad de los mismos.

SQL Injection

La inyección de código es un problema el cual todos los desarrolladores combaten diariamente para que este no dañe los sistemas que tanto les costó desarrollar, la implementación de una buena seguridad contra inyección SQL en los sistemas es una forma de mantener todo nuestro sistema seguro, una buena forma de optimizar tiempo y perfecto para evitar problemas de confidencialidad e integridad.

SQL Injection es una vulnerabilidad en nuestros sistemas que permite a los atacantes ingresar código SQL de manera malintencionada para poder manipular la base de datos del sistema, por lo que todos los datos de nuestro sistema se encuentran comprometidos por su baja seguridad. La intención de esta practica es modificar el comportamiento de las consultas que hace nuestro sistema por medio de parámetros no deseados, con esta inyección de código se pueden realizar edición, visualización, inserción y eliminación de datos en nuestra base de datos.

La inyección ocurre normalmente por el mal uso de practicas al momento de desarrollar sistemas, normalmente al solicitar a los usuarios cualquier tipo de entrada que no esta validada, permite al usuario o atacante el ingreso de nuevas peticiones en la base de datos.

Ataque SQL Injection

Normalmente, la forma mas efectiva de hacer una inyección de código es mediante la validación de la consulta del sistema como verdadera. A continuación, se explica a detalle el procedimiento.

Esta query se utiliza en un sistema para validar que exista un usuario (Login) con el “username” y “password“ ingresadas:

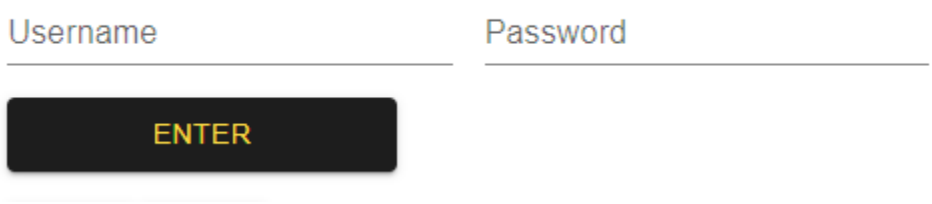
```
SELECT * FROM users WHERE username = ‘$username’ AND password = ‘$password’;
```

Por lo que el texto ingresado en el input de “username” será el texto que sustituye la variable `‘$username’` en el query.

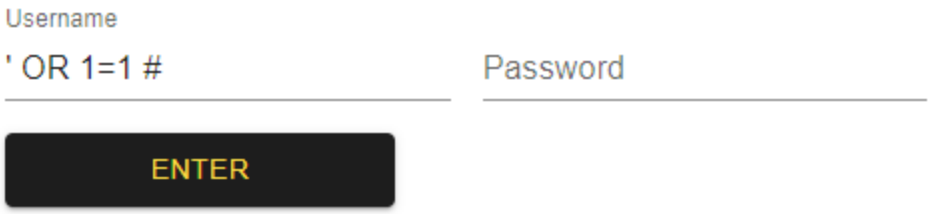
En este lenguaje de consultas (SQL) es importante saber que el símbolo “#” comenta cualquier línea de código que pongamos después de colocar el signo y también debemos de saber que el “;” es el signo que delimita un query de otro, por lo que después de colocar un punto y coma, podemos agregar cuantas consultas se nos ocurran.

Para poder burlar este sistema solo es necesario ponernos un poco creativos y visualizar un nuevo query, por lo que como atacantes tenemos que validar la query original del sistema, cerrar la querry e ingresar las consultas que querremos que se realicen a la base de datos del sistema.

Imaginemos que nuestra inyección de código lo queremos realizar en el login de una web page, lo mas seguro es que el input de “username” use un query como el mostrado anteriormente, visualicemos el formulario de login así:



Para poder realizar nuestra inyección de SQL solo es necesario terminar el query del sistema con una validación como esta:



La consulta quedaría de la siguiente forma:

```
SELECT * FROM users WHERE username = ‘’ OR 1=1; # AND password = ‘$password’;
```

Por lo que el query se ejecuta hasta el primer punto y coma y el resto del código es comentado y no ejecutado después de la almohadilla “#”, por lo que como atacantes podemos ingresar todas las consultas que queramos después del primer punto y coma.

Select:

Insert:

```
SELECT * FROM users WHERE username = ‘’ OR 1=1; INSERT INTO `users`(`username`, `password`, `name`, `birth`) VALUES ('Hacker1','123','SecretName','0000-00-00'); #
```

Update:

```
SELECT * FROM users WHERE username = ‘’ OR 1=1; UPDATE `users` SET`username`='[value-1]',`password`='[value-2]',`name`='[value-3]',`birth`='[value-4]'WHERE username = "Hacker1"; #
```

Delete:

```
SELECT * FROM users WHERE username = ‘’ OR 1=1; DELETE FROM “USERS” WHERE username = "Hacker1"; #
```

Prevención de ataques SQL

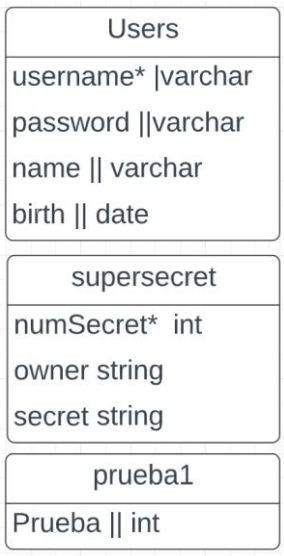
- **Limitación de ingreso de caracteres en los inputs en los cuales el usuario coloca sus credenciales.** Esta buena práctica de programación hace que el usuario no pueda ingresar líneas de código que tienen muchos caracteres, por lo que un query no podrá ser ingresado en el input. Esta buena práctica se puede burlar ya que es algo que se realiza desde el frontend, por lo tanto, un usuario con algo de experiencia podría editar la cantidad de caracteres a ingresar.
- **Filtración de caracteres ingresados.** Con esta acción podemos eliminar todos los caracteres que pueden ocasionar una inyección de código (/[$#’”;]/), con la eliminación de estos caracteres, nuestra base de datos no podrá realizar las peticiones maliciosas ya que no llevaran la sintaxis necesaria. Esta función también puede ser burlada si la función estuviera almacenada en el frontend.
- **Uso de librerías en el servidor.** Con estas librerías especializadas en combatir el SQL injection, podemos encontrar funcionalidades que eliminen cualquier tipo de carácter que lleve nuestro string para que a la base de datos llegue un string limpio. También hay librerías que hacen que solo se evalué solo la primera consulta que se le da al servidor.
- **Uso de vistas para las consultas.** El utilizar tablas volátiles es una buena práctica al desarrollar sistemas ya que así nuestros datos originales no están expuestos.
- **SQLMap.** Es una herramienta de código abierto que automatiza los procesos de detección de SQL Injection.

Librerias Usadas

- MySQLi: Librería que nos permite que el servidor lea una sola consulta a la base de datos, sin importar cuantas vayan, solo lee la primera.

Diagrama entidad relación

Por temas de practicidad, ninguna de las tablas en la base de datos tiene relación entre si esto con el fin de una mejor comprensión de sistema



Conclusiones

El cuidado de nuestros datos en el siglo XXI es de suma importancia ya que su valor es realmente alto, por lo que, tanto como usuarios y desarrolladores tenemos que procurar que esta información se encuentra protegida de la mejor forma posible.

Un ataque de SQL injection es fácil de mitigar, pero normalmente pasamos por alto el problema y solo por esa razón podemos obtener sistemas enteros inseguros en donde nuestros datos estén expuestos. La prevención de estos ataques es fundamental para cualquier sistema informático, nos ahorrara tiempo y dinero, ya que la filtración de datos de un sistema puede llevar a problemas legales.

Bibliografía

- Redondo, P. C. (2021, 25 octubre). *¿Qué es SQL Injection?* OpenWebinars.net. https://openwebinars.net/blog/que-es-sql-injection/