UNIVERSIDAD DEL ISTMO

FACULTAD DE INGENIERÍA

PARCIAL I: SQL INJECTION

Seguridad

ADALÍ GARRÁN JIMÉNEZ

Carné #20191137

Guatemala, 18 de agosto de 2022

ÍNDICE

INTRODUCCIÓN

De acuerdo con UNIR Ecuador, la seguridad informática o ciberseguridad es: “la protección de la información y, especialmente, al procesamiento que se hace de la misma, con el objetivo de evitar la manipulación de datos y procesos por personas no autorizadas”. Esta tiene como objetivo que personas, equipos y datos informáticos estén protegidos ante amenazas y eventualidades que puedan ocurrir en estos. La seguridad informática se ha vuelto de gran importancia en los últimos años debido al crecimiento en el uso de la tecnología y uso de internet en el que se manejan grandes cantidades de datos, entre los cuales hay información sensible.

Las bases de datos son mecanismos que permiten la definición, almacenamiento y manipulación de datos. Tal como lo explica su definición es en ellas que la información manejada por sistemas informáticos es almacenada, por lo que son el objetivo de muchos ataques cibernéticos. Una clase de ataques de este tipo es la inyección SQL.

El objetivo de este trabajo es explicar qué es un ataque de inyección SQL, qué implicaciones puede tener y algunas formas de prevenirlo; así como un ejemplo de realización de este ataque y de prevención de este utilizando diferentes librerías como apoyo.

SQL INJECTION

La inyección SQL es un tipo de ataque de inyección de código en una base de datos, en el que se mandan comandos a la base de datos que no deberían poder ser enviados por el usuario con el objetivo de obtener datos a los que no se debería tener acceso, alterar datos o borrar información. Este tipo de ataque podría incluso llevar a un borrado completo de la base de datos de una aplicación y es uno de los ataques web más comunes que existen.

La inyección SQL ocurre cuando se tienen "queries" o consultas dinámicas, en las que el cliente debe ingresar información un input o caja de texto, como un usuario o una contraseña, pero en lugar de ingresar dicha información, agrega más texto de forma que el "query" se extienda, corriendo una consulta que no debería en la base de datos.

Algunos ejemplos de inyección SQL pueden ser:

- Obtener data oculta: modificar una consulta para que muestre más datos de los que debería.
- Cambio de la lógica de la aplicación: modificar una consulta, cambiando la lógica de la aplicación.
- Ataques de unión: obtener datos de diferentes tablas, uniendo el query original con otro nuevo, permitiendo ver tablas que no estaban indicadas en el query.
- Examinación de la base de datos: obtener información de la estructura y especificaciones de la base de datos.
- Alteración de datos: en los que se modifican o eliminan datos de la base de datos.

IMPLICACIONES

La inyección SQL puede tener diversas consecuencias negativas para una base de datos; por ejemplo (de acuerdo con la OWASP):

- Confidencialidad: cuando se permite la inyección SQL, la data sensible queda al descubierto para el atacante, afectando la confidencialidad de la información.
- Autenticación: de la mano de la confidencialidad, el atacante podría tener acceso a usuarios y contraseñas guardadas en la base de datos. De esta forma, el acceso a una cuenta puede quedar al descubierto.

- Autorización: si se consiguen datos de roles y permisos de usuario en una base de datos, es posible escalar en la información, permitiendo al atacante obtener toda clase de datos y la autorización para acceder a ellos.
- Integridad: al igual que es posible que el atacante pueda leer y extraer información, también es posible alterar la data en la base, e incluso eliminar información de esta, afectando la integridad de los datos que se encuentran almacenados.

Así, la inyección SQL, puede permitir a un atacante falsificar su identidad, alterar datos, anular o alterar transacciones, divulgar datos, cambiar permisos, destruir data importante, robar data sensible, e incluso destruir una base de datos por completo. Además, en algunos servidores de base de datos, es posible acceder al sistema operativo por medio de estos. En estos casos, la inyección SQL puede funcionar como un escalón inicial en un ataque mayor a una red o servidor.

FORMAS DE PEREVENCIÓN

La única forma de asegurar una aplicación ante una inyección SQL, es mediante una validación de inputs, parametrización de consultas y limpieza de caracteres en el texto. No se debe permitir que el texto ingresado por un cliente en un input de una aplicación vaya directamente a la base de datos sin ser revisado primero. De acuerdo con el sitio web acutenix, prevenir una inyección SQL no es sencillo porque puede haber vulnerabilidades en diferentes puntos; sin embargo, sí existen algunas prácticas que ayudan a evitarla; por ejemplo:

- Limpiar caracteres especiales. Por ejemplo, eliminar símbolos como ‘,’, ‘/’, ‘;’, ‘””… que puedan utilizarse para agregar información al query, o en el caso de “;”, añadir un nuevo *SQL Statement*, que pueda alterar la base de datos.
- Validar tipo de datos en un input. Esto ayuda a que el usuario no pueda ingresar caracteres o texto que no corresponda con el tipo de dato que se requiere en el query dinámico.
- Tratar todos los inputs como si fueran inseguros. Cualquier input que se utilice en un query dinámico puede resultar un riesgo, no importa el usuario que sea. Todo input debe ser revisado para identificar vulnerabilidades y poder eliminarlas.

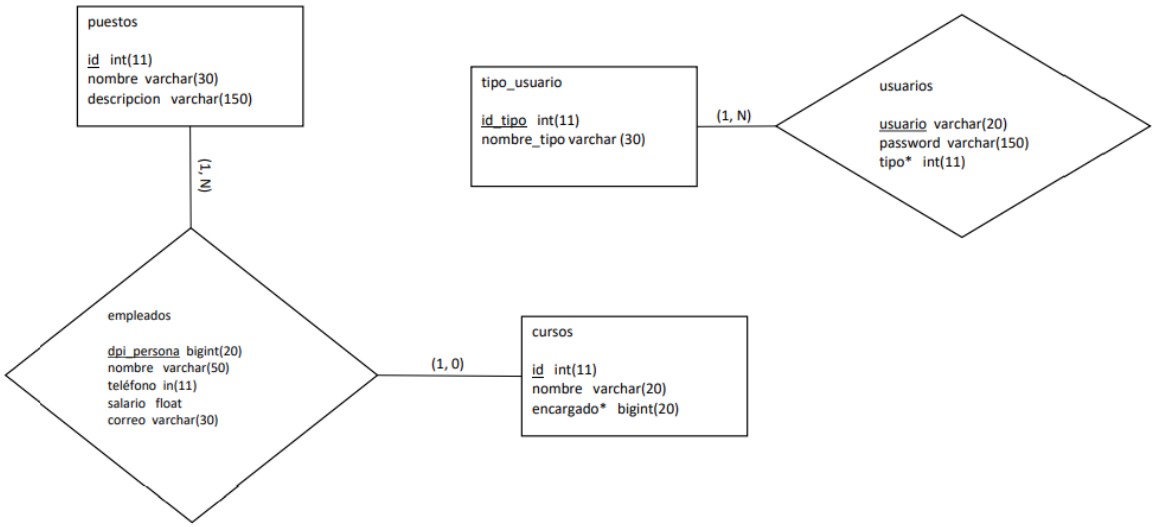
- Utilizar librerías especializadas. Estas pueden ayudar a limpiar inputs o identificar cuando el usuario intenta modificar un query de una forma que no debería.
- Escanear las aplicaciones web constantemente en búsqueda de vulnerabilidades nuevas.

PROYECTO

Para el proyecto se realizó una base de datos de una aplicación de empleados de un centro educativo. La base de datos fue realizada en MySQL. Se tienen 5 tablas tal como se muestra en la Figura 1.

FIGURA 1

Esquema de la base de datos



Fuente: propia

La aplicación web consistía en un login de usuarios que daba paso a un home page. El BackEnd se realizó en PHP, mientras que el FrontEnd se realizó con ReactJS.

LIBRERÍA UTILIZADA

Para eliminar el riesgo del ataque de inyección SQL se utilizaron funciones de MySQLi, dado que la base de datos fue creada en MySQL, con una conexión a través de un Backend de

PHP. MySQLi es una extensión de PHP utilizada para establecer una comunicación entre PHP y el servidor de base de datos (MySQL en este caso). Esta librería contiene diversas funciones que pueden ayudar a prevenir un ataque de inyección SQL, limpiando los inputs desde el servidor de BackEnd.

Funciones utilizadas:

Para este proyecto se utilizaron dos funciones diferentes para evitar la inyección SQL:

- mysqli_query(): Esta función se utiliza para realizar una consulta a la base de datos. En este caso, esta permite que únicamente se ejecute una operación en la base de datos. De esta forma, si se intenta agregar una nueva consulta dentro de la variable a sustituir en el query dinámico, MySQLi no lo permitirá. Así, no se pueden alterar datos ni acceder a otras tablas.
- mysqli_real_escape_string(): se salta los caracteres especiales de un string para utilizarse en un query de SQL. Los caracteres saltados son NUL (ASCII 0), \n, \r, \, ', ", y Control-Z. En este caso específico, las comillas pueden afectar el código y permitir una inyección SQL.

Así, con el uso de estas dos funciones se evita tener más de un statement a ejecutar (mysqli_query) y alargar un query para que muestre más datos (mysqli_real_escape_string()).

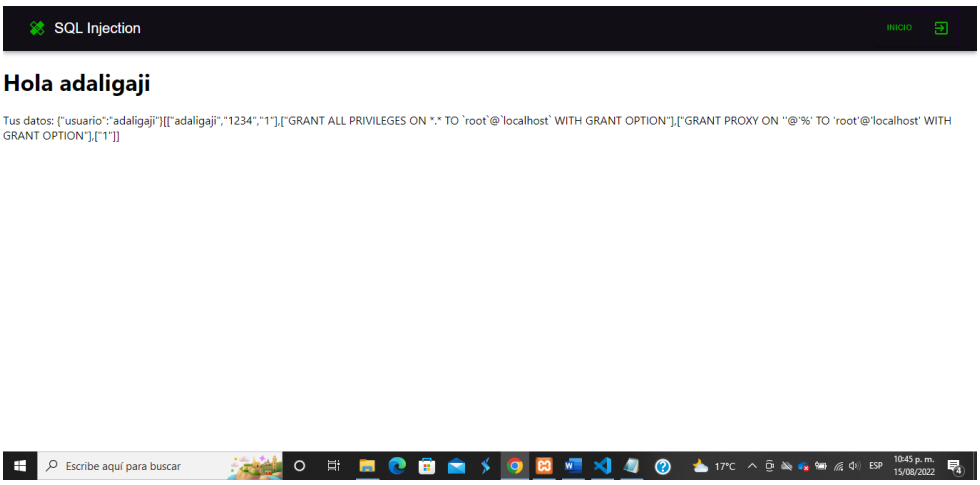
Además de estas funciones, se trae el resultado de la consulta únicamente si se tiene solamente un valor que coincida con clave y password; si se encuentran más tuplas no se traen los datos ni se autentica al usuario.

En las Figura 2 y Figura 3 se muestran los resultados mostrados por la aplicación web antes y después, respectivamente, de utilizar las funciones para prevenir la inyección SQL ingresando el siguiente string en la caja de texto de la contraseña:

1234'; show grants; select '1

FIGURA 2

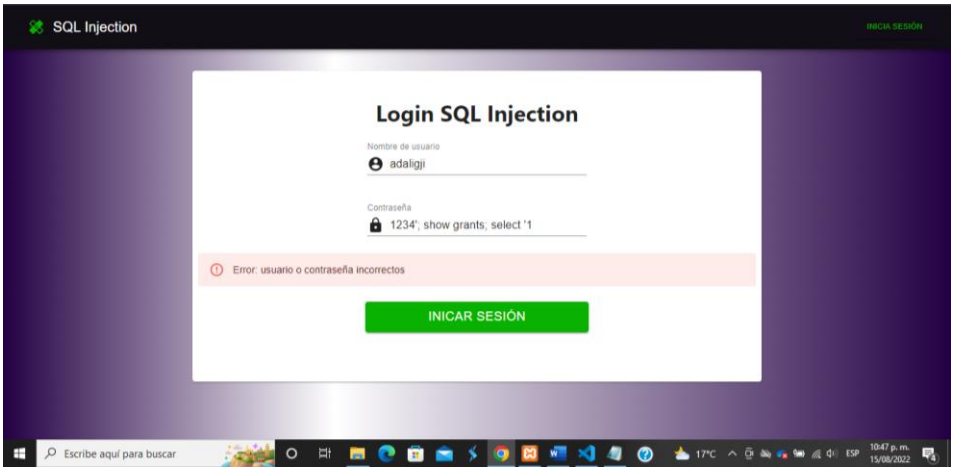
SQL Injection



Fuente: propia

FIGURA 3

SQL Injection prevenida



Fuente: propia

CONCLUSIONES

- La inyección SQL es uno de los tipos de ataques más comunes, en el que se mandan comandos a la base de datos que no deberían poder ser enviados por el usuario con el objetivo de obtener datos a los que no se debería tener acceso, alterar datos o borrar información.
- Las inyecciones SQL tienen implicaciones para la confidencialidad, autenticación, autorización, integridad… en una base de datos y compromete la información que se almacene en este.
- Para evitar un ataque de inyección SQL es muy importante limpiar el texto que ingresa un usuario en un input que se utilice en un query dinámico, de forma que no se le permita modificar la consulta que se tiene pensada según la lógica de la aplicación.
- Existen muchas librerías para diferentes lenguajes que ayudan a los desarrolladores con la prevención de las inyecciones SQL, protegiendo los datos de las aplicaciones.
- En el caso de PHP (en el que son muy comunes las inyecciones SQL), la extensión MySQLi (para las bases de datos en MySQL), posee diferentes funciones que permiten asegurar que solo se realice un query por ejecución, así como funciones para limpiar caracteres especiales de los inputs, fortaleciendo la aplicación ante ataques de inyección SQL.
- Es importante escanear constantemente nuestras aplicaciones en busca de vulnerabilidades que puedan comprometer datos de los usuarios de estas.

BIBLIOGRAFÍA

ACUNETIX. What is SQL Injection (SQLi) and How to Prevent It. *Acutenix by Invicti.* s.f. Disponible en: https://www.acunetix.com/websitesecurity/sql-injection/ [consulta: 15/08/2022].

IONOS. Novedad en PHP: MySQLi. *Digital Guide IONOS.* 2020. Disponible en: https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/introduccion-a-mysqli/ [consulta: 15/08/2022].

KINGTHORIN. SQL Injection. *OWASP*. s.f. Disponible en: https://owasp.org/www-community/attacks/SQL_Injection [consulta: 15/08/2022].

PHP. mysqli::real_escape_string. *PHP*. s.f. Disponible en: https://www.php.net/manual/es/mysqli.real-escape-string.php [consulta: 15/08/2022].

PORT SWIGGER. SQL Injection. *Web Security Academy.* s.f. Disponible en: https://portswigger.net/web-security/sql-injection [consulta: 15/08/2022].

UNIR. ¿Qué es la seguridad informática y cuáles son sus tipos? *Universidad en Internet.* 2021. Disponible en: https://ecuador.unir.net/actualidad-unir/que-es-seguridad-informatica/ [consulta: 15/08/2022].

W3. SQL Injection. *W3 Schools.* s.f. Disponible en: https://www.w3schools.com/sql/sql_injection.asp [consulta: 15/08/2022].