Seguridad e inyección en SQL

Administración y Diseño de Bases de Datos

Cheuk Kelly Ng Pante, Javier González de la Barreda Arimany y Samuel Toledo Hernández

15 de noviembre de 2023

Índice general

1. Introducción a la seguridad de bases de datos **2**
1.1. Introducción a las bases de datos 2
1.2. Seguridad de bases de datos 2

2. Riesgos y consecuencias de la inyección SQL **5**
2.1. Introducción a SQL 5
2.2. Inyección SQL 5
2.2.1. Riesgos y consecuencias que conlleva una inyección SQL 5
2.2.2. Como evitar ataques e inyección SQL 6

3. Tipos de ataques de inyección SQL y ejemplos de casos reales **7**

4. Buenas prácticas en el diseño de bases de datos para prevenir la inyección SQL **8**
4.1. Introducción a las buenas prácticas 8
4.2. Consultas concatenadas 8
4.3. Otras prácticas recomendadas 9

5. Marco legal y cumplimiento normativo **10**
5.1. Introducción al marco legal 10
5.2. Cumplimiento normativo 10

1. Introducción a la seguridad de bases de datos

1.1. Introducción a las bases de datos

Una base de datos consiste en una colección de datos interrelacionados que representan información sobre una organización o área en particular. Estas bases de datos están organizadas según modelos de datos que definen la estructura, almacenamiento y manipulación de la información. El modelo principal es un modelo relacional que representa datos a través de tablas y crea relaciones entre ellos.

Los sistemas de gestión de bases de datos (DBMS) son programas informáticos que gestionan, modifican, consultan y crean bases de datos. Entre sus ventajas importantes se incluyen la independencia de los datos con respecto a la aplicación respectiva, la garantía de integridad mediante reglas de validación; seguridad de la información mediante control de acceso; y optimización del rendimiento a través de índices/vistas.

En el diseño de bases de datos intervienen una variedad de etapas, incluido el análisis de requisitos, el diseño conceptual, el diseño lógico y el diseño físico. Los modelos de entidad, atributos y relaciones se utilizan en el modelo de relación de entidad para representar datos en el diseño conceptual. El modelo relacional es la base del diseño lógico, ya que presenta datos a través de tablas, claves y restricciones.

El lenguaje utilizado para consultar la base de datos es un conjunto de instrucciones que permiten a los usuarios seleccionar, insertar, actualizar y eliminar datos. La mayoría de los DBMS pueden ser compatibles con SQL, qué es el lenguaje de consulta más utilizado.

En el ámbito de la programación de bases de datos, se emplean subrutinas, que son bloques de código encargados de llevar a cabo tareas específicas relacionadas con los datos. Estas subrutinas pueden manifestarse como procedimientos almacenados o funciones, diferenciándose en que los procedimientos almacenados pueden retornar varios valores o ninguno, mientras que las funciones solo pueden devolver un valor. La ejecución de estas subrutinas dentro del Sistema de Gestión de Bases de Datos (SGBD) contribuye a mejorar tanto la eficiencia como la seguridad de las bases de datos

1.2. Seguridad de bases de datos

La seguridad de la base de datos abarca un conjunto de herramientas, medidas y controles diseñados para proteger la integridad, confidencialidad y disponibilidad de los datos almacenados en una base de datos. Estas “copias” están implementadas para evitar el acceso no autorizado, modificaciones inapropiadas o pérdidas accidentales. En este contexto, se cubrirán aspectos técnicos y organizativos, incluidas actividades como autenticación, auditoría, cumplimiento normativo, gestión de riesgos, educación, etc.

La importancia de la seguridad de las bases de datos reside en su función esencial para asegurar el funcionamiento adecuado de las organizaciones y la salvaguardia de la privacidad de las personas. Para preservar la integridad del sistema, resulta crucial hacer frente a amenazas como ataques internos, errores humanos, vulnerabilidades de software, ataques de inyección SQL, ataques de denegación de servicio y la presencia de malware.

Ante las amenazas mencionadas, se sugiere adoptar las mejores prácticas de ciberseguridad, que incluyen medidas como salvaguardar la integridad física, aplicar controles administrativos y de acceso a la red, asegurar dispositivos y cuentas de usuarios finales, proteger el software de bases de datos, garantizar la seguridad de servidores de aplicaciones y web, así como implementar precauciones en las copias de respaldo y llevar a cabo auditorías periódicas.

La seguridad de las bases de datos emerge como un tema de gran importancia y complejidad. En este contexto, se busca resguardar la información almacenada en dichas bases contra accesos no autorizados, alteraciones indebidas y pérdidas accidentales o maliciosas. La protección de la base de datos involucra diversos elementos críticos como:

- **Datos de la base de datos:** Esta información, que abarca desde nombres y contraseñas hasta números de tarjetas de crédito y debe mantenerse confidencial, íntegra y accesible únicamente para usuarios autorizados.
- **Sistema de gestión de bases de datos (DBMS):** Desempeña un papel fundamental al crear, administrar y manipular las bases de datos. Es esencial que el DBMS esté actualizado, configurado y protegido de manera adecuada para prevenir vulnerabilidades y ataques.
- **Aplicaciones asociadas:** Los programas que interactúan con la base de datos para realizar diversas operaciones deben validar y filtrar las entradas de los usuarios. Además, es crucial que utilicen consultas parametrizadas o procedimientos almacenados, y limiten los permisos y privilegios de los usuarios y las bases de datos.
- **Servidor de base de datos físico y/o virtual y hardware subyacente:** Estos dispositivos, ya sean físicos o virtuales, donde residen la base de datos y el DBMS, deben contar con protección física, sistemas de copia de seguridad y recuperación, así como medidas de seguridad de red, como firewalls, antivirus y cifrado.
- **Infraestructura informática y/o de red para acceder a la base de datos:** Los medios a través de los cuales usuarios y aplicaciones se comunican con la base de datos, como ordenadores, dispositivos móviles e internet, deben garantizar la seguridad y privacidad de las comunicaciones. Esto implica el uso de protocolos seguros, contraseñas robustas, certificados de seguridad, entre otras medidas.

La importancia de asegurar las bases de datos radica en la necesidad de prevenir o reducir al mínimo los riesgos y las consecuencias asociadas a los ciberataques. Estos eventos pueden resultar en daños irreparables tanto para los datos almacenados como para las organizaciones y los usuarios involucrados. Algunos de los ciberataques más comunes y peligrosos pueden ser:

- **La inyección SQL:** Implica la introducción de código SQL malicioso en las entradas de los usuarios para alterar o acceder a la información de la base de datos. Este tipo de ataque puede ocasionar la pérdida o el robo de datos sensibles, la modificación o eliminación de información crucial, la ejecución de comandos arbitrarios en el servidor, o la revelación de datos internos o confidenciales de una organización.
- **El robo de credenciales:** Se refiere a la obtención de contraseñas o nombres de usuario de usuarios o aplicaciones que acceden a la base de datos. El propósito de este ataque es suplantar identidades o llevar a cabo acciones no autorizadas, lo que puede resultar en acceso no autorizado a los datos, alteración o borrado de información, o la propagación de malware o virus.
- **El ransomware:** Implica cifrar los datos de la base de datos o bloquear su acceso, con el objetivo de exigir un rescate a cambio de su liberación. Este tipo de ataque puede conducir a la inaccesibilidad de los datos, interrupciones en el funcionamiento del negocio o el pago de grandes sumas de dinero

La seguridad de las bases de datos exige la implementación de un conjunto integral de medidas técnicas, organizativas y legales destinadas a resguardar los datos de posibles amenazas, tanto internas como

externas, como hackers, empleados deshonestos, errores humanos, fallas de hardware o software, desastres naturales y violaciones normativas. Diversas prácticas, políticas y tecnologías pueden ser adoptadas para fortalecer la seguridad de las bases de datos, entre las que se encuentran:

- Diseñar una arquitectura de base de datos segura que separe los datos sensibles de los no sensibles, que minimice los puntos de acceso y que aplique el principio de mínimo privilegio.
- Implementar un sistema de gestión de bases de datos (DBMS) actualizado y configurado adecuadamente, que ofrezca funciones integradas de seguridad, tales como cifrado, control de acceso, registro de eventos y detección de anomalías.
- Desarrollar una aplicación segura que utilice métodos de conexión seguros, que evite la inyección de código malicioso, valide los datos de entrada y cifre la información en tránsito y en reposo.
- Llevar a cabo pruebas de seguridad periódicas que evalúen la vulnerabilidad de la base de datos e identifiquen y corrijan posibles debilidades o brechas.
- Impartir formación y concienciación a los usuarios y administradores de la base de datos acerca de las mejores prácticas de seguridad, como la utilización de contraseñas sólidas, el cambio regular de credenciales, el bloqueo de sesiones inactivas y la notificación de cualquier incidente sospechoso

2. Riesgos y consecuencias de la inyección SQL

2.1. Introducción a SQL

SQL (Structured Query Language) se trata de un lenguaje de programación especializado diseñado para la gestión y manipulación de bases de datos relacionales. Su función principal radica en permitir que los usuarios y las aplicaciones realicen consultas, inserten, actualicen y modifiquen datos almacenados en una base de datos. Este lenguaje incorpora diversos comandos y cláusulas que posibilitan la interacción de los usuarios con los sistemas de gestión de bases de datos (DBMS), tales como MySQL, PostgreSQL, Oracle, Microsoft SQL Server, entre otros. Los comandos fundamentales de SQL abarcan desde la recuperación de datos mediante la cláusula SELECT, la inserción de datos mediante la cláusula INSERT, hasta la actualización y eliminación de datos mediante las cláusulas UPDATE y DELETE, respectivamente.

Además de estos comandos básicos, SQL ofrece la capacidad de crear y modificar esquemas de bases de datos mediante comandos de definición de datos, como CREATE, ALTER y DROP. Estos permiten a los usuarios establecer la estructura de la base de datos y sus objetos, como tablas, vistas, índices y procedimientos almacenados.

En términos generales, SQL provee a los usuarios de una herramienta poderosa y eficiente para la administración y manipulación efectiva de grandes volúmenes de datos. Esto lo posiciona como una herramienta esencial en el ámbito de la gestión de bases de datos y el desarrollo de software. Sin embargo, es crucial destacar que su uso indebido o incorrecto puede dar lugar a problemas de seguridad, como la inyección SQL, que puede comprometer la integridad y confidencialidad de los datos almacenados en la base de datos.

2.2. Inyección SQL

La inyección SQL es un tipo de ciberataque que se aprovecha de los errores existentes en aplicaciones web para meter código malicioso y atacar bases de datos de SQL. Al introducir el código van con el fin de quebrantar las medidas de seguridad y privacidad y así acceder a datos protegidos o de carácter sensible con el objetivo de eliminar información o incluso editar las bases de datos. Existen tres clases de ataques principales de inyección SQL. Estos son:

- **Ataque por error:** Son los más comunes. Se puede obtener información de la base de datos.
- **Ataque por unión:** Una página web muestra más resultados de los que debería, entre los que se incluye la amenaza.
- **Ataque ciego:** Es el ataque más complicado de realizar. Se hacen preguntas cerradas a la base de datos

2.2.1. Riesgos y consecuencias que conlleva una inyección SQL

Los atacantes de inyección SQL plantean una variedad de amenazas de seguridad para la organización afectada, afectando así los usuarios como las entidades que sufren este tipo de agresiones. Una vez los ciberdelincuentes se aprovechan de una vulnerabilidad, pueden:

- **Robo de datos sensibles:** los agresores tienen la capacidad de adquirir información confidencial, como contraseñas, detalles de tarjetas de crédito, datos personales o información de clientes. Este tipo de datos obtenidos pueden ser utilizados con intenciones maliciosas, como llevar a cabo fraudes, robar identidades o realizar chantajes.

- **Modificación o eliminación de contenido de la base de datos:** los atacantes pueden manipular o suprimir información almacenada en la base de datos. Esta acción puede tener consecuencias perjudiciales para el funcionamiento de la aplicación web, generando pérdidas económicas o dañando la reputación de la organización.
- **Manipulación y/o ejecución de código:** los atacantes pueden manipular o suprimir información almacenada en la base de datos. Esta acción puede tener consecuencias perjudiciales para el funcionamiento de la aplicación web, generando pérdidas económicas o dañando la reputación de la organización.
- **Escalada de privilegios:** Los atacantes pueden obtener privilegios de administrador o de usuario en la base de datos o en el servidor. Esto les permite llevar a cabo acciones que normalmente estarían fuera de su alcance, como la creación, modificación o eliminación de usuarios, tablas o registros

2.2.2. Como evitar ataques e inyección SQL

Con el fin de prevenir la inyección SQL, es esencial implementar diversas medidas de seguridad, tanto por parte de los desarrolladores como de los usuarios. Algunas de estas medidas incluyen:

- Validar y filtrar las entradas de los usuarios para evitar la introducción de caracteres o comandos no deseados.
- Emplear consultas parametrizadas o procedimientos almacenados, los cuales impiden que el código SQL se modifique al ejecutarse.
- Restringir los permisos y privilegios de los usuarios y bases de datos para limitar el acceso y las acciones posibles.
- Mantener actualizados los sistemas, aplicaciones y bases de datos para corregir posibles vulnerabilidades o fallos de seguridad.
- Utilizar contraseñas seguras y gestores de contraseñas para salvaguardar las credenciales de acceso y prevenir posibles robos o adivinanzas.
- Proporcionar información personal únicamente en sitios web de confianza que cuenten con el protocolo HTTPS y un certificado de seguridad válido.

3. Tipos de ataques de inyección SQL y ejemplos de casos reales

4. Buenas prácticas en el diseño de bases de datos para prevenir la inyección SQL

4.1. Introducción a las buenas prácticas

Con el objetivo de evitar la inyección SQL, se crearon unos estándares para poder aplicar unas "técnicas" adecuadas para evitar estos problemas. Algunos de ellos son:

1. Emplear parámetros en lugar de realizar consultas concatenadas. Estas últimas son conocidas como una de las técnicas más frecuentes para la inserción de código malicioso. La utilización de parámetros previene que el código malicioso sea inyectado en la consulta. Debido a que las consultas concatenadas incluyen directamente variables que pueden tener un contenido malicioso a la hora de hacer la consulta.

2. Asegurarse de validar todas las entradas de usuario. Este parámetro es crucial para garantizar que no se incluye código malicioso. La validación puede lograrse mediante diversas técnicas, incluyendo la comprobación del formato, la longitud y el contenido de las entradas.

3. Utiliza filtros de salida para proteger los datos sensibles de posibles exposiciones. Estos filtros son eficaces para eliminar caracteres especiales que puedan ser aprovechados para inyectar código malicioso.

4.2. Consultas concatenadas

Las consultas concatenadas como hemos dicho anteriormente, al permitir la inyección de código malicioso a través de la simple concatenación con las entradas de usuario, representan un riesgo evidente, como se demuestra en este ejemplo:

```
SELECT * FROM users WHERE username = 'admin' AND password = '1234';
```

Un atacante o un "craker" puede llegar a aprovecharlo para inyectar código malicioso y eliminar la tabla de usuarios de la base de datos. Para evitar la inyección SQL, resulta de crucial importancia sustituir las consultas concatenadas con el uso de parámetros. De esta manera, los valores se especifican en la consulta, dificultando la inyección de código malicioso. Por ejemplo:

```
SELECT * FROM users WHERE username = :username AND password = :password;
```

Aquí, el desarrollador especifica los valores de username y password como parámetros.

- La validación de todas las entradas de usuario es un paso esencial para asegurarse de las buenas prácticas. También es de debida importancia asegurarse de que las entradas cumplan con formatos específicos, limitar la longitud y eliminar caracteres especiales es clave para prevenir la inyección de código malicioso.

- Los filtros de salida son efectivos para eliminar caracteres especiales, como punto y coma (;), numeral (#), comilla simple ('), comilla doble ("), menor que (¡) y mayor que (¿), reduciendo el riesgo de inyección SQL.

4.3. Otras prácticas recomendadas

Existen otras medidas que se pueden tomar en el diseño de bases de datos para prevenir la inyección SQL y que están dentro de las buenas prácticas para poder evitar exploits o funcionamientos inadecuados. Estas medidas son clasificables como las siguientes:

1. **Utilizar tablas separadas para datos sensibles.** Los datos sensibles, como las contraseñas, las tarjetas de crédito y los números de seguridad social, deben almacenarse en tablas separadas. Esto dificulta que un atacante acceda a estos datos si logra inyectar código malicioso en la base de datos.

   Los datos sensibles, como las contraseñas, las tarjetas de crédito y los números de seguridad social, deben almacenarse en tablas separadas. Esto dificulta que un atacante acceda a estos datos si logra inyectar código malicioso en la base de datos.

2. **Utilizar funciones hash para almacenar contraseñas.** Las contraseñas no deben almacenarse en texto plano en la base de datos. En su lugar, deben almacenarse en forma de hash. Esto hace que sea más difícil para un atacante descifrar las contraseñas si logra robarlas. Esto se debe a que las tablas hashes están encriptadas por su función hash.

   Las contraseñas no deben almacenarse en texto plano en la base de datos. En su lugar, deben almacenarse en forma de hash. Esto hace que sea más difícil para un atacante descifrar las contraseñas si logra robarlas.

   Las funciones hash convierten una contraseña en un valor único que es difícil de descifrar. Cuando un usuario inicia sesión, la contraseña ingresada se compara con el hash almacenado en la base de datos. Si los hashes coinciden, el usuario se autentica.

3. **Utilizar procedimientos almacenados.** Los procedimientos almacenados son funciones que se ejecutan en el servidor de base de datos. Estos procedimientos pueden utilizarse para validar las entradas de usuario y ejecutar consultas SQL de forma segura.

   Los procedimientos almacenados pueden ayudar a prevenir la inyección SQL de las siguientes maneras:

   - **Validación de entradas:** Los procedimientos almacenados pueden utilizarse para validar las entradas de usuario antes de ejecutar una consulta SQL. Esto ayuda a garantizar que las entradas no contengan código malicioso.
   - **Consultas seguras:** Los procedimientos almacenados pueden utilizarse para ejecutar consultas SQL de forma segura. Esto ayuda a evitar que un atacante utilice una consulta SQL para inyectar código malicioso en la base de datos.

4. **Utilizar cifrado para proteger los datos.** Los datos sensibles pueden cifrarse para protegerlos de la exposición. Esto puede hacerse en el servidor de base de datos o en el cliente.

   Los datos sensibles pueden cifrarse para protegerlos de la exposición. Esto puede hacerse en el servidor de base de datos o en el cliente. El cifrado en el servidor de base de datos se realiza antes de que los datos se almacenen en la base de datos. El cifrado en el cliente se realiza antes de que los datos se envíen al servidor de base de datos. El cifrado puede ayudar a proteger los datos sensibles de los ataques de inyección SQL. Si un atacante logra inyectar código malicioso en la base de datos, los datos cifrados estarán protegidos.

5. Marco legal y cumplimiento normativo

5.1. Introducción al marco legal

La inyección SQL como hemos visto hasta ahora, es una gran vulnerabilidad de seguridad en organizaciones o ambientes de desarrollo de software. Los "crakers" o los atacantes pueden llegar a aprovecharse de estas vulnerabilidades, llegando a robar datos datos o información crítica. Esto nos da paso a nuestro siguiente punto el cual es la importancia de comprender y cumplir con el marco legal y las regulaciones relacionadas con la inyección SQL.

En el contexto Español, la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD), establece las bases legales para abordar la inyección SQL. Dicha ley establece las responsabilidades que deben de manejar las entidades que influyen o manejan datos personales, para que así estas contengan las medidas de seguridad correspondientes para proteger la información.

Además el artículo 32 de la LOPDGDD registra que aquellos que son responsables del tratamiento de datos deben adoptar medidas necesarias para garantizar un nivel de seguridad adecuado, incluyendo:

- La aplicación de medidas técnicas para proteger los datos personales contra el acceso no autorizado, la alteración, la divulgación, la destrucción o la pérdida accidental, ilícita o maliciosa.
- La implementación de medidas organizativas adecuadas para asegurar que solo personas autorizadas tengan acceso a los datos personales.
- La realización de copias de seguridad de los datos personales como precaución adicional.

5.2. Cumplimiento normativo

La inyección SQL es una vulnerabilidad de seguridad grave que puede dar lugar a consecuencias penales graves. En España, la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD), establece las siguientes consecuencias penales para la inyección SQL:

- Multa si la infracción se comete por imprudencia grave o negligencia grave.
- Prisión de uno a tres años.

Además, la inyección SQL también puede dar lugar a consecuencias penales en virtud de otras leyes, como la Ley Orgánica 10/1995, de 23 de noviembre, del Código Penal. Por ejemplo, la inyección SQL puede utilizarse para cometer los siguientes delitos:

- Robo de datos personales (artículo 197.1 del Código Penal).
- Daños informáticos (artículo 264 del Código Penal).
- Amenazas (artículo 169 del Código Penal).
- Coacción (artículo 172 del Código Penal).

Por lo tanto, es importante que las organizaciones que manejan datos personales tomen medidas para prevenir la inyección SQL. Estas medidas incluyen:

- Implementar controles de seguridad adecuados, como la validación de entrada, el uso de contraseñas seguras y la formación del personal.
- Tener un plan de respuesta a incidentes en caso de que se produzca una violación de datos.

El Real Decreto 43/2021, de 26 de febrero, por el que se modifica el Reglamento de desarrollo de la Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales, establece nuevas obligaciones para las organizaciones que manejan datos personales. Estas obligaciones incluyen:

- Implementar medidas de seguridad técnicas y organizativas adecuadas para garantizar la seguridad de los datos personales. Estas medidas deben ser proporcionales al riesgo al que están expuestos los datos.
- Realizar una evaluación de riesgos para identificar los riesgos a los que están expuestos los datos personales.
- Seleccionar y aplicar medidas de seguridad adecuadas para mitigar los riesgos identificados.
- Documentar las medidas de seguridad implementadas.