Engineer

inyección SQL

Ataque que convierte la entrada en parte de una consulta SQL

Qué es

La inyección SQL, a menudo abreviada SQLi, es un ataque en el que una aplicación construye una orden SQL con texto aportado por otra persona y la base de datos acaba interpretando parte de ese texto como SQL en lugar de como un valor. Common Weakness Enumeration la recoge como CWE-89 y describe la debilidad como una orden construida total o parcialmente a partir de entradas externas sin neutralizar antes los caracteres que tienen un significado especial en SQL. La guía de prevención de OWASP expresa la misma idea de forma más sencilla: el ataque funciona cuando datos no fiables pasan a formar parte de la sintaxis de una consulta.

El problema nace de que SQL no establece una separación firme entre las instrucciones de una orden y los datos que contiene. Una comilla simple cierra un literal de cadena, y lo que el atacante escriba a continuación se analiza como más SQL. Dos guiones inician un comentario, lo que permite descartar el resto de la sentencia escrita por quien desarrolla la aplicación. Un punto y coma, cuando la base de datos y el controlador lo permiten, termina una sentencia e inicia otra. Ninguno de esos signos es extraño: también aparecen en apellidos y en texto libre.

La guía de pruebas de seguridad web de OWASP divide los ataques en tres clases según cómo recibe el atacante los resultados. Los ataques dentro de banda obtienen la respuesta por el mismo canal que utilizaron para enviar la solicitud, normalmente en la propia página de la aplicación. Los ataques fuera de banda hacen que la base de datos entregue los resultados en otro lugar, por ejemplo mediante una conexión HTTP independiente. Los ataques inferenciales o ciegos no reciben datos en absoluto y los reconstruyen pregunta a pregunta a partir del comportamiento de la aplicación. La guía enumera cinco técnicas habituales

entre esas clases: un operador `UNION` que añade un segundo `SELECT` al primero, condiciones booleanas, errores provocados deliberadamente, solicitudes fuera de banda y retrasos intencionados.

La inyección SQL pertenece a una familia más amplia. En CWE está incluida en la debilidad general de inyección, CWE-74, junto con el cross-site scripting, y OWASP Top Ten de 2025 sitúa ambas en la categoría A05 Injection. El rasgo común es que los datos cruzan una frontera y llegan a un lugar donde se interpretan como instrucciones. Aquí el intérprete es el analizador SQL de la base de datos, y la solución consiste en mantener todos los valores fuera del texto SQL.

Utilice el concepto en una decisión

Aplique «¿Qué es la inyección SQL?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Cuentas, contraseñas y permisos organizados para que cada persona tenga exactamente el acceso que necesita y lo pierda el mismo día que deja la empresa.

Explore nuestros servicios Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. 1.000 reglas RBAC para servidores Linux — 1,000 reglas de acceso escritas para cerrar una brecha de seguridad

1.000 reglas RBAC para servidores Linux Póngase en contacto →

Por qué importa

El daño depende de los permisos de la cuenta de base de datos de la aplicación, que a menudo son muy amplios. CWE-89 enumera la lectura de datos que el atacante nunca debería ver, eludir la autenticación sin conocer una contraseña, cambiar los datos que deciden quién puede hacer qué, modificar o borrar registros y, en algunas configuraciones, ejecutar órdenes del sistema operativo. La página de OWASP sobre el ataque añade operaciones administrativas como apagar la base de datos, leer archivos del servidor de base de datos y hacerse con su control como administrador.

Ambas fuentes consideran grave el problema. CWE califica de alta la probabilidad de explotación, y la página de OWASP describe el fallo como fácil de encontrar y de explotar. La debilidad aparece en todas las ediciones de la lista CWE Top 25 de las debilidades de software más peligrosas entre 2019 y 2025. La entrada de CWE también documenta una oleada de 2008 en la que una sola cadena de inyección SQL, utilizada contra muchos programas distintos, comprometió numerosos servidores web y los hizo servir código malicioso.

La defensa, en cambio, se conoce bien y resulta poco costosa. La guía de OWASP presenta las sentencias preparadas con parámetros vinculados como su primera defensa, y CWE también las recomienda: el SQL y los valores viajan por separado, de modo que la base de datos no puede confundir unos con otros. Todo lo demás, desde el principio de mínimo privilegio hasta un tratamiento cuidadoso de los errores, es una segunda capa que limita el daño cuando se cuela un error.

Cómo se aplica en la práctica

Considere una tienda cuya página de historial de pedidos busca los pedidos mediante la dirección de correo introducida en un formulario. Quien desarrolla la aplicación construye el SQL con una f-string de Python:

```
cur.execute(f"SELECT id, total FROM orders WHERE email = '{email}'")
```

Una persona introduce `x' OR '1'='1` en el campo. El programa inserta ese texto y la base de datos recibe lo siguiente:

```
SELECT id, total FROM orders WHERE email = 'x' OR '1'='1'
```

La comilla situada después de `x` cierra la cadena que abrió quien escribió el programa, y la cláusula `OR` que sigue siempre es verdadera, por lo que la base de datos devuelve todos los pedidos de la tabla. CWE-89 y la guía de pruebas usan el mismo truco de una condición siempre verdadera, que también puede hacer que un formulario de acceso acepte a alguien que no conoce ni el nombre de usuario ni la contraseña.

Una entrada más peligrosa termina la sentencia e inicia otra. El ejemplo de CWE introduce `name'; DELETE FROM items; --` como nombre de un artículo, de modo que una segunda sentencia sigue a la primera y los guiones convierten el resto en un comentario. Que llegue a ejecutarse depende de la base de datos y del controlador, y varias interfaces actuales lo rechazan. `PQexecParams` de PostgreSQL admite como máximo una orden SQL por cadena, algo que su documentación considera de cierta utilidad frente a la inyección SQL. El módulo `sqlite3` de Python produce un error cuando `execute()` recibe más de una

sentencia, y psycopg rechaza varias sentencias en una sola llamada cuando se pasan parámetros. Nada de eso impide la condición siempre verdadera del ejemplo anterior, que solo necesita una sentencia.

La solución consiste en pasar el valor por separado. PEP 249, el estándar de interfaces de bases de datos de Python, exige que cada controlador declare su estilo de marcadores de posición en una constante del módulo llamada `paramstyle`, y enumera cinco estilos: `qmark` (`?`), `numeric` (`:1`), `named` (`:name`), `format` (`%s`) y `pyformat` (`%(name)s`). El módulo `sqlite3` tiene fijado `qmark` y también acepta `named`. Psycopg, el controlador de PostgreSQL, admite `%s` y `%(name)s`:

```
cur.execute(
    "SELECT id, total FROM orders WHERE email = %s",
    (email,),
)
```

El `%s` parece un formato de cadenas de Python, pero no se aplica ningún formato: el SQL y la tupla de valores llegan al controlador como dos argumentos separados. Psycopg 3 los mantiene separados durante todo el recorrido y envía la orden y sus parámetros al servidor por separado, en lugar de unirlos en el cliente. La guía de OWASP lo ilustra con su propia entrada de ejemplo, `tom' or '1'='1`: vinculada como parámetro, se busca literalmente como un nombre de usuario extraño y no coincide con nadie.

PostgreSQL ofrece la misma separación en SQL puro. `PREPARE` fija la sentencia con parámetros numerados, y `EXECUTE` aporta los valores más adelante:

```
PREPARE order_lookup (text) AS
    SELECT id, total FROM orders WHERE email = $1;
EXECUTE order_lookup('x'' OR ''1''=''1');
```

La entrada hostil, escrita aquí como un literal SQL, llega a `$1` como una única dirección de correo de aspecto extraño y no encuentra pedidos.

En una plataforma que almacena datos profesionales y de organizaciones, las entradas se validan antes de acceder a la base de datos y, a continuación, la API se conecta con un rol de mínimo privilegio que solo puede ejecutar las funciones

de la aplicación, con todo parametrizado. La vinculación impide que los datos se conviertan en SQL, y el rol restringido limita el alcance de un error.

Dónde surgen los problemas

Construir la cadena de todos modos. Una f-string, el operador `%`, `+` o `.format()` incorpora el valor al SQL antes de que lo vea el controlador. Lo mismo ocurre con un marcador escrito entre comillas, como `'%s'`. OWASP señala que concatenar valores no fiables antes de preparar una sentencia anula toda la protección, y la documentación de psycopg indica que nunca se deben usar `%` o `+` para unir valores a las consultas. Pase cada valor como segundo argumento de `execute()`, con el marcador sin comillas.

Escapar a mano. Duplicar las comillas o añadir barras inversas parece equivalente, pero las reglas dependen del dialecto SQL y de su configuración. OWASP pone como ejemplo el modo `NO_BACKSLASH_ESCAPES` de MySQL, que desactiva el escape mediante barras inversas, y su guía desaconseja firmemente esta defensa. Sustituya las cadenas escapadas a mano por parámetros vinculados.

Tratar la validación o un WAF como la solución. La validación detecta errores, pero el texto libre tiene que admitir los caracteres que necesita un atacante: el ejemplo de CWE es el apellido O'Reilly, que probablemente superaría la validación y aun así contiene una comilla. Las listas de bloqueo de palabras peligrosas tienen resquicios, y CWE considera que la eficacia de un cortafuegos de aplicaciones es solo moderada. Como lo expresa OWASP: «Validar un valor no hace que sea seguro concatenarlo al SQL». Siga validando para comprobar la corrección y vincule los parámetros para mantener la seguridad.

Tomar de la entrada nombres de tablas, nombres de columnas o el sentido de ordenación. Los marcadores solo representan valores, por lo que quien necesita permitir al usuario elegir una columna o la dirección de `ORDER BY` puede caer en la tentación de insertar el texto. La respuesta de OWASP es asociar cada elección a un conjunto reducido de nombres controlados por la aplicación y rechazar el resto, por ejemplo convirtiendo la dirección solicitada en el literal `ASC` o `DESC`. Cuando un nombre realmente deba ser dinámico, el módulo `psycopg.sql` de psycopg compone identificadores, aunque OWASP advierte que entrecomillar un nombre no limita hacia qué tablas se puede dirigir la aplicación.

Dar por hecho que los procedimientos almacenados son seguros. Llamar a un procedimiento con argumentos vinculados protege la llamada, pero no el SQL que el propio procedimiento construye internamente. CWE señala que usar un procedimiento almacenado solo descarta algunas vías de inyección y deja otras abiertas. Dentro de la base de datos, use SQL estático o vincule también los valores en el SQL dinámico. En PostgreSQL eso significa usar `EXECUTE ... USING` para valores y `format('%I', ...)` para identificadores.

Usar un ORM con cadenas insertadas. Un mapeador objeto-relacional vincula los valores cuando se usa como está previsto, y CWE atribuye una protección considerable a capas de persistencia como Hibernate cuando se utilizan correctamente. OWASP añade que las consultas de Hibernate siguen siendo vulnerables a la inyección cuando se concatena la entrada al texto de la consulta. Use los parámetros del mapeador en lugar de construir cadenas.

Una cuenta de aplicación con derechos de administrador. Si la cuenta puede hacerlo todo, el atacante también. OWASP pide cuentas de solo lectura cuando únicamente hacen falta lecturas, un usuario de base de datos distinto por aplicación web, vistas que expongan solo las columnas que necesita la cuenta y ningún derecho de administrador para las cuentas de aplicación. Cuando una aplicación solo trabaja mediante procedimientos almacenados, sus permisos deberían limitarse a ejecutar esos procedimientos y nada más. El mínimo privilegio limita el daño; no evita la inyección.

Mostrar a los usuarios errores detallados de la base de datos. Un mensaje de error que revela la estructura de la sentencia ayuda al atacante a preparar el siguiente intento, y la técnica basada en errores de la guía de pruebas se apoya precisamente en eso. CWE recomienda mensajes de error mínimos. Muestre un fallo genérico al usuario y conserve los detalles en el servidor para el equipo.

Confiar en los datos que ya están en la base de datos. Un valor almacenado de forma segura puede seguir siendo peligroso cuando otro código lo inserta más tarde en SQL. OWASP indica que sus controles se aplican sea cual sea el origen de los datos: una solicitud HTTP, otro servicio o una escritura anterior en el almacenamiento. CWE advierte que los procesos posteriores pueden olvidar escapar lo que se guardó. Vincule los valores leídos de sus propias tablas igual que vincula las entradas de un formulario.

Lecturas adicionales

- CWE-89 en Common Weakness Enumeration — La entrada de la debilidad: consecuencias, ejemplos y medidas de mitigación con sus límites.
- Guía de prevención de inyección SQL de OWASP — Consultas parametrizadas, procedimientos almacenados, identificadores dinámicos y mínimo privilegio.
- Guía de pruebas de seguridad web de OWASP: pruebas de inyección SQL — Las clases y técnicas de ataque, tal como se presentan al realizar pruebas.
- PEP 249: especificación de la API de bases de datos de Python v2.0 — La interfaz común de los controladores de Python, incluidos los cinco estilos de marcadores de posición.
- Psycopg 3: pasar parámetros a consultas SQL — Las reglas de los marcadores de posición y la forma correcta de construir SQL dinámico.
- PostgreSQL: funciones de ejecución de órdenes — Cómo envía libpq los parámetros separados del texto de la orden.

Glosario

Elija según sus restricciones reales

Describa la ruta de despliegue, los riesgos operativos y quiénes se encargan de las guardias. Podemos definir el alcance de un cambio con comprobaciones, una vía de recuperación y responsabilidades claras.

Vea cuánto cuesta Póngase en contacto →

Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es la inyección SQL?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/sql-injection/