SQL Injection

Jared Ricks, Jack Lin, Kolbe Williams-Wimmer

SQL injection is a vulnerability that can occur in SQL databases. SQL injection is entirely preventable with safe coding practices, but an inexperienced programmer could create a database subject to this attack. An attacker exploiting a SQL injection vulnerability will typically attempt to input some type of SQL code where user input is required, such as when querying the database. For example, a query that is vulnerable to SQL injection is:

```
SELECT * FROM users WHERE id = 'user_id';
```

This query relies on user input for user_id, meaning that a user could input anything they want into this field. If a user entered: `1' OR '1'='1`

It would modify the query to:

```
SELECT * FROM users WHERE id = '1' OR '1'='1';
```

This would always evaluate to true, meaning that all records for all user ID numbers would be displayed. This can have a major impact if the information in the database is very sensitive. Fortunately, many mitigation techniques exist that rely on good coding practices.

Some mitigation techniques include:

1. Parameterized queries - queries that use a ? in place of user input helps separate data from code, which prevents users from modifying the queries. The ? is used as a placeholder for data that will be gathered from the user. There are also other ways to use parameterized queries without using ? as a placeholder, but it is a common technique.
2. Escaping User Input - This method escapes special characters in user input so that it cannot be evaluated as executable code.
3. Input Validation - This method ensures that users supply the correct type of input. For example, if a user attempts to enter text into an ID field, it may ask the user to try again and ensure they are entering an ID number.
4. Using ORM or Stored Procedures- This method relies on using Object-Relational Mapping libraries to help guide programmers in writing safe code or to provide safe pre-defined SQL statements.