Armaan Shamsaasef

SQL Injection

SQL Injection - SY0-701 - 2.3

Definition: A web application vulnerability that allows an attacker to interfere with the database queries that an application makes. It occurs when user input is not properly validated and is included directly in a SQL query.

How SQL Injection Works

1. The Normal Process: A web application takes user input (e.g., a username from a login form) and inserts it into a SQL query template to communicate with the database.
    - Example Query: `SELECT * FROM users WHERE username = '[user_input]' AND password = '[hashed_input]'`
2. The Injection: An attacker submits maliciously crafted input that changes the structure and meaning of the intended SQL query.
    - Malicious Input: `' OR 1=1 --`
    - Resulting Query: `SELECT * FROM users WHERE username = '' OR 1=1 --' AND password = '...'`

Breaking Down the Malicious Input

- The Single Quote (`'`): This closes the string literal for the `username` field in the original query.
- `OR 1=1`: This adds a condition that is always true. Because of the `OR`, the entire `WHERE` clause now returns every row in the `users` table, regardless of the username or password.
- The Double-Dash (`--`): This is the SQL comment symbol. It comments out the rest of the original query, effectively removing the password check.

Result: The database returns the first user in the `users` table (often an administrator), and the application logs the attacker in as that user.

The Broader Impact

SQL injection is not limited to bypassing logins. A successful attack can allow an attacker to:

- Read sensitive data from any table in the database (e.g., credit cards, personal information).
- Modify or delete data in the database (e.g., change prices, delete user accounts).
- Execute administrative operations on the database (e.g., shut down the DBMS).

Video Example: The Login Bypass

The video demonstrates a simple login form. By entering the payload `' OR 1=1 --` into the username field and any random text for the password, the attacker bypasses authentication because the manipulated query no longer checks for a valid password and returns a user.

Summary

- SQL Injection exploits a lack of input validation and sanitization in a web application.
- The attacker "breaks out" of the data field and injects their own SQL commands into the query.
- The classic example for bypassing authentication is using the payload `' OR 1=1 --`.
- The consequences are severe, leading to data theft, data manipulation, and full system compromise.