OWASP Security Shepherd Project

Mobile/Web Security Awareness and Education
What is Security Shepherd?

Security Shepherd

OWASP
The Open Web Application Security Project
What is SQL Injection?

Injection flaws, such as SQL injection occur when hostile data is sent to an interpreter as part of a command or query. The hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data. Injections attacks are of a high severity. Injection flaws can be exploited to access any information held on the system and removing a system's confidentiality. These security risks can then be extended to execute updates to existing data affecting the systems integrity and availability. These attacks are easily exploitable as they can be initiated by anyone who can interact with the system through any data they pass to the application.

In the following form's parameters are concatenated to a string that will be passed to a SQL server. This means that the data can be interpreted as part of the code.

The objective here is to modify the result of the query with SQL injection so that all of the table's rows are returned. This means you want to change the boolean result of the query's WHERE clause. The easiest way to ensure the boolean result is always true is to inject a boolean 'OR' operator followed by a true statement like 1 = 1.

If the parameter is been interpreted as a string, you can escape the string with an apostrophe. That means that everything after the apostrophe will be interpreted as SQL code.

Use SQL injection in the following example to retrieve all of the tables rows. The lesson's solution key will be found in one of these rows! The results will be posted beneath the search form.
Admin

Cheat Sheet Management
Module Management

Open Floor Modules
CTF Mode
Enable Module Block
Disable Module Block
Set Module Status
View Feedback
View Progress
Scoreboard

User Management
Configuration

CTF Mode

If you enable the CTF floor plan, players will have to complete lessons to unlock links to the next module.

Enable CTF Mode
# Scoreboard

Select the class you would like to show the scoreboard for

<table>
<thead>
<tr>
<th>Username</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>This_Is_Bananas!</td>
<td>1028</td>
</tr>
<tr>
<td>caomh</td>
<td>1007</td>
</tr>
<tr>
<td>AbePetrillo</td>
<td>1005</td>
</tr>
<tr>
<td>davo69</td>
<td>1004</td>
</tr>
<tr>
<td>gotomanners</td>
<td>1002</td>
</tr>
<tr>
<td>JustinCredible</td>
<td>954</td>
</tr>
<tr>
<td>amsqr</td>
<td>952</td>
</tr>
<tr>
<td>Shane</td>
<td>952</td>
</tr>
<tr>
<td>paulwhelan</td>
<td>950</td>
</tr>
<tr>
<td>Conor</td>
<td>950</td>
</tr>
<tr>
<td>pendo19</td>
<td>905</td>
</tr>
<tr>
<td>Enda1</td>
<td>887</td>
</tr>
</tbody>
</table>
Lesson & Challenge Demonstration
Lesson & Challenge Demonstration
private boolean login(String username, String password) {
    try {
        String dbPath = this.getDatabasePath("Members.db").getPath();

        SQLiteDatabase db = SQLiteDatabase.openOrCreateDatabase(dbPath, dbPass, null);

        String query = "SELECT * FROM MEMBERS WHERE memName = '" + username + ' AND memPass = '" + password + '"");
        Cursor cursor = db.rawQuery(query, null);

        if (cursor.getCount() <= 0) {
            return false;
        }
    }
}
if (CheckName.contentEquals("Root")
    && CheckPass.contentEquals("rootPassword")) {

    Toast loggedIn = Toast.makeText(BadApp.this,
           "Logged in!", Toast.LENGTH_SHORT);

    loggedIn.show();
Mark Denihan
@markdenihan
mark.denihan@owasp.org

Sean Duggan
@duggan4sean
sean.duggan@owasp.org