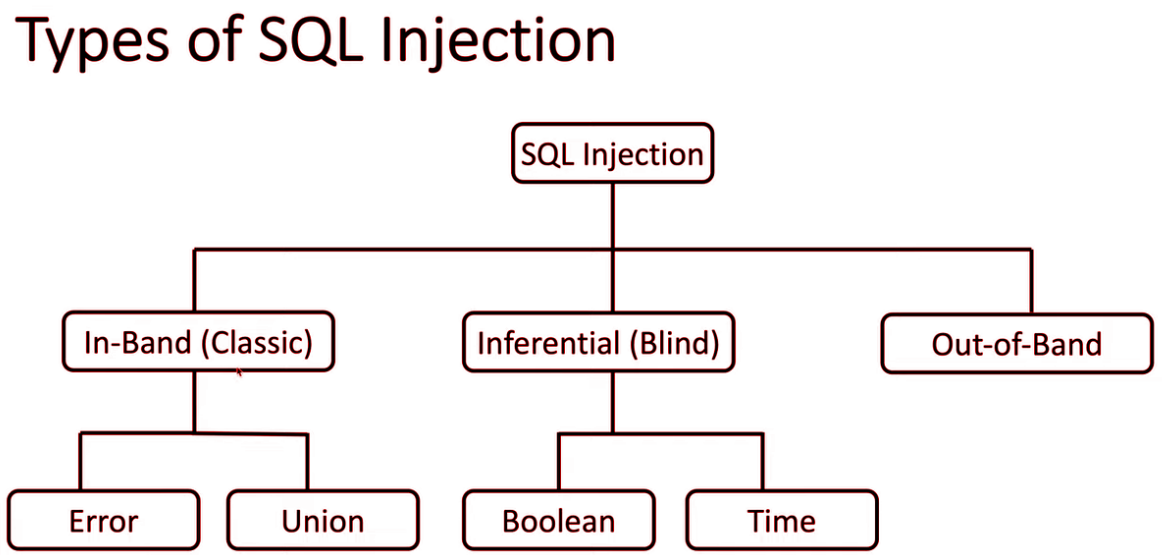
SQL Injection

SQL-Injection, auch SQLI genannt, ist ein häufiger Angriffsvektor, der bösartigen SQL-Code zur Manipulation von Backend-Datenbanken verwendet, um auf Informationen zuzugreifen, die nicht angezeigt werden sollten. Diese Informationen können eine beliebige Anzahl von Elementen umfassen, darunter vertrauliche Unternehmensdaten, Benutzerlisten oder private Kundendaten.



Videos:

SQL Injection ¦ Complete Guide

SQL Injection Hacking Tutorial (Beginner to Advanced)

Quellen:

https://www.w3schools.com/sql/sql_injection.asp

https://www.youtube.com/watch?v=wcaiKgQU6VE

Planung:

Wir wollen eine kleine Login- Page aufbauen, mit welcher wir dann eine SQL Injection darstellen und zeigen, wie der SQL Code dahinter dann aussieht. Nachdem wir geklärt haben, was eine SQL Injection ist, wollen wir Maßnahmen zeigen, wie man ein System vor einer SQL Injection schützt.

- Docker File mit Webserver, SQL Server und index.html index.css und login .php

Docker:

Die Config, über welche Version von php und sql genutzt wird, steht in der sample.env Datei.

Wichtig:

- im www folder ist die index .php
- in der .env sind am schluss die credentials für die database conecction:
  If you need to give the docker user access to more databases than the "docker" db you can grant the privileges with phpmyadmin to the user.
  MYSQL_USER=docker
  MYSQL_PASSWORD=docker
  MYSQL_DATABASE=docker
  #MySQL root user password
  MYSQL_ROOT_PASSWORD=tiger

Verbesserungen im Code der Index.php:

inder index.php folgendes probieren:

```
 // Überprüfe, ob der Benutzer bereits angemeldet ist
if(isset($_SESSION['username'])) {
    echo "Hey Yo Mister White, du bisch scho a gemälde du depp"; //
    echo '<form action="clear_cache.php" method="post">
            <input type="submit" value="Cache löschen">
          </form>';
    exit();
}
```

html css verbesserungen:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login Page SQL-Injection</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            margin: 0;
            padding: 0;
```

```
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
        }
        .container {
            background-color: #fff;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        h2 {
            text-align: center;
            color: #333;
        }
        form {
            text-align: center;
        }
        label {
            display: block;
            margin-bottom: 5px;
            color: #555;
        }
        input[type="text"],
        input[type="password"] {
            width: 100%;
            padding: 10px;
            margin-bottom: 10px;
            border: 1px solid #ccc;
            border-radius: 5px;
            box-sizing: border-box;
        }
        input[type="submit"] {
            width: 100%;
            padding: 10px;
            background-color: #007bff;
            color: #fff;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            transition: background-color 0.3s;
        }
        input[type="submit"]:hover {
            background-color: #0056b3;
        }
        .error-message {
            color: #ff0000;
            text-align: center;
```

```
            margin-top: 10px;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Login</h2>
        <?php
        // Fehlermeldung anzeigen, wenn Anmeldung fehlgeschlagen ist
        if (isset($login_error)) {
            echo "<p class='error-message'>$login_error</p>";
        }
        ?>
        <form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]); ?>">
            <label for="name">Benutzername:</label><br>
            <input type="text" id="name" name="name"><br>
            <label for="passwort">Passwort:</label><br>
            <input type="password" id="passwort" name="passwort"><br><br>
            <input type="submit" name="sub" value="Anmelden">
        </form>
    </div>
</body>
</html>
```

Lösungen: to prevent sql injection The code you provided is vulnerable to SQL injection attacks. It's recommended to use prepared statements to prevent this. Here's an example of how you can rewrite the code using prepared statements:

```
// Überprüfe, ob das Formular abgeschickt wurde
if ($_SERVER["REQUEST_METHOD"] == "POST") {
   // Benutzername und Passwort aus dem Formular erhalten
   $input_username = $_POST['name'];
   $input_password = $_POST['passwort'];

   // Vorhandene Anweisung vorbereiten
   $sql = "SELECT * FROM user WHERE name=? AND passwort=?";
   $stmt = mysqli_prepare($link, $sql);

   // Parameter binden
   mysqli_stmt_bind_param($stmt, "ss", $input_username, $input_password);
```

```php
// Anweisung ausführen
mysqli_stmt_execute($stmt);

// Ergebnis abrufen
$result = mysqli_stmt_get_result($stmt);
```

This way, the input values are safely escaped and can't be used to manipulate the SQL query.