SQL инжекция и ползване на параметризирани команди

Учителски екип

Обучение за ИТ кариера

https://it-kariera.mon.bg/e-learning/

Съдържание

1. SQL инжекция
2. Параметризирани команди

SQL инжекция

Какво е SQL инжекцията? Как да се защитим?

Какво е SQL инжекцията? (1)

```
bool IsPasswordValid(string username, string password)
{
  string sql =
    $"SELECT COUNT(*) FROM Users " +
    $"WHERE UserName = '{username}' AND" +
    $"PasswordHash = '{CalcSHA1(password)}'";
  SqlCommand cmd = new SqlCommand(sql, dbConnection);

  int matchedUsersCount = (int) cmd.ExecuteScalar();
  return matchedUsersCount > 0;
}
```

Какво е SQL инжекцията? (2)

```
bool normalLogin =
  IsPasswordValid("peter", "qwerty123"); // true

bool sqlInjectedLogin =
  IsPasswordValid(" ' or 1=1 --", "qwerty123"); // true

bool evilHackerCreatesNewUser =
  IsPasswordValid("' INSERT INTO Users VALUES('hacker','') --",
  "qwerty123");
```

Как работи SQL инжекцията?

- Следните SQL команди се изпълняват:
  - Обичайна проверка на паролата (без SQL инжекция):

```
SELECT COUNT(*) FROM Users WHERE UserName = 'peter'
and PasswordHash = 'XOwXWxZePV5iyeE86Ejvb+rIG/8='
```

  - SQL-инжектирана проверка за парола:

```
SELECT COUNT(*) FROM Users WHERE UserName = ' ' or 1=1
-- ' and PasswordHash = 'XOwXWxZePV5iyeE86Ejvb+rIG/8='
```

  - SQL-инжектирана INSERT команда:

```
SELECT COUNT(*) FROM Users WHERE UserName = ''
INSERT INTO Users VALUES('hacker','')
--' and PasswordHash = 'XOwXWxZePV5iyeE86Ejvb+rIG/8='
```

Защита от SQL инжекция

- Начини за защита от SQL инжекции:
  - SQL-избягване на всички данни идващи „отвън“ приложението:

```
string escapedUsername = username.Replace("'", "''");
string sql =
    "SELECT COUNT(*) FROM Users " +
    "WHERE UserName = '" + escapedUsername + "' and " +
    "PasswordHash = '" + CalcSHA1(password) + "'";
```

    - Не се препоръчва: Само в краен случай!
  - Предпочитан подход:
    - Чрез параметризирани заявки /parameterized queries/
    - Отделя се SQL командата от нейните аргументи

SqlParameter

- Какво са **SqlParameter**?
  - SQL заявките и съхранените процедури могат да имат входни и изходни параметри
  - Достъпват се чрез **Parameters** свойството на класа **SqlCommand**
- Свойства на **SqlParameter**:
  - **ParameterName** – име на параметъра
  - **DbType** – SQL тип (**NVarChar**, **Timestamp**, …)
  - **Size** – размер на типа (ако е приложимо)
  - **Direction** – вход/изход

Пример: Параметризирани команди

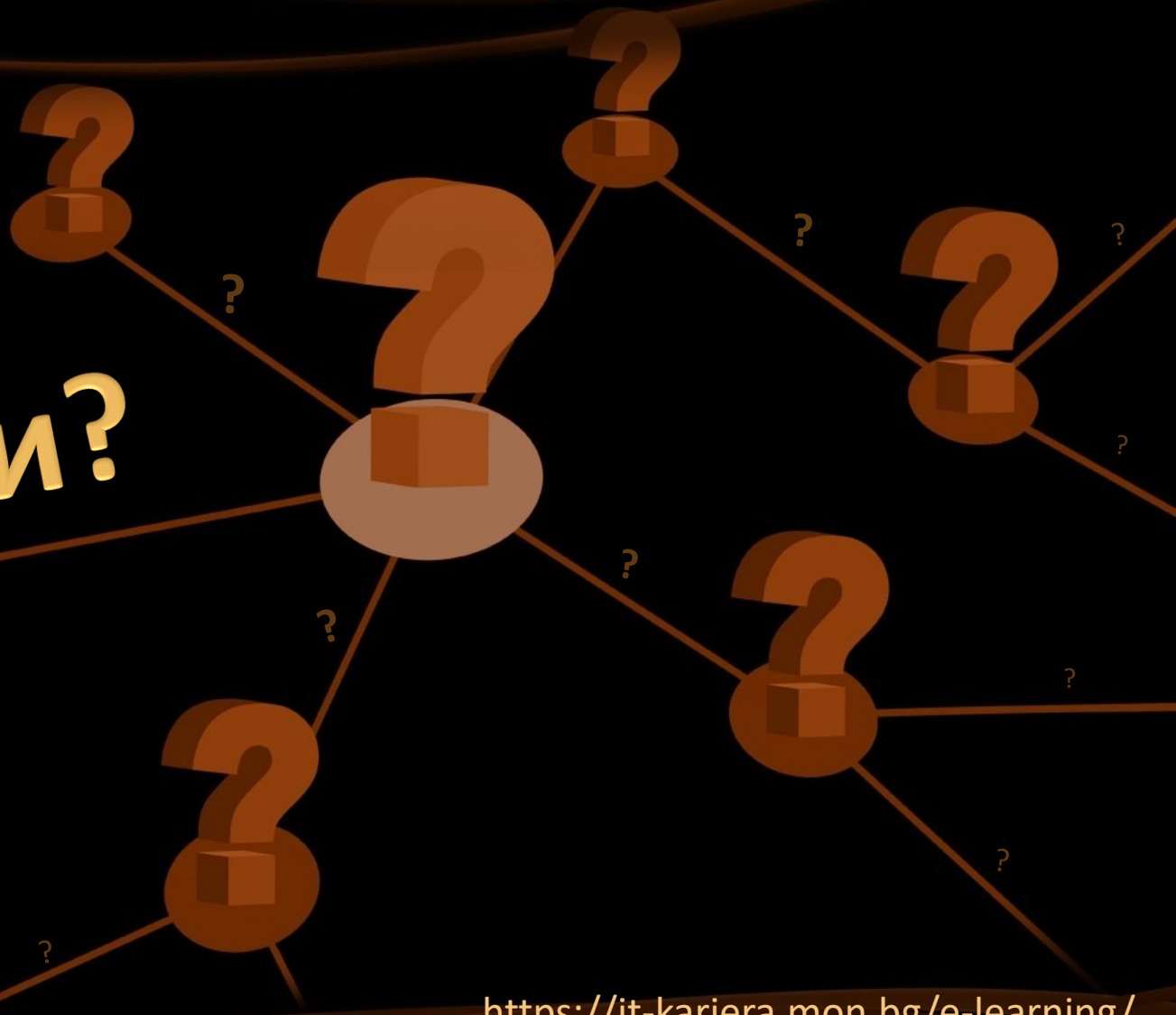
```
void InsertProject(string name, string description, DateTime
startDate)
{
  SqlCommand cmd = new SqlCommand("INSERT INTO Projects " +
    "(Name, Description, StartDate, EndDate) VALUES " +
    "(@name, @desc, @start, @end)", dbCon);

  cmd.Parameters.AddWithValue("@name", name);
  cmd.Parameters.AddWithValue("@desc", description);
  cmd.Parameters.AddWithValue("@start", startDate);

  cmd.ExecuteNonQuery();
}
```

SQL инжекция и ползване на параметризирани команди

Въпроси?



https://it-kariera.mon.bg/e-learning/

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**