Advance Programming Techniques (APT)

Lecture # 42

Ehtisham Rasheed

Department of Computer Science
University of Gurjat, Gujrat

What is SQL Injection?

- SQL Injection is a hacking technique where an attacker inserts malicious SQL code into your query to:
  - Steal data
  - Modify or delete records
  - Login without a password
  - Drop entire tables
  - Take control of the database
- Why it happens
  - Because **developer concatenates user input into SQL queries**, like:

```
string query = "SELECT * FROM Users WHERE Username = '" + txtUsername.Text + "'";
```

How SQL Injection Works

```
string query = "SELECT * FROM Users WHERE Username='" + txtUsername.Text +
               "' AND Password='" + txtPassword.Text + "'";
```

- Suppose attacker enters:
  - Username: admin' –
  - Password: (anything)
  - This becomes:

```
SELECT * FROM Users
WHERE Username='admin' --' AND Password=''
```

- **→ Attacker logs in as admin without password**

Dangerous SQL Injection Payloads

- Login bypass
  - ' OR '1'='1
  - This becomes true for every record
- Extract all data
  - ' UNION SELECT * FROM Users –
- Delete all rows
  - '; DELETE FROM Students –
- Drop table
  - '; DROP TABLE Students --

How C# Application Becomes Vulnerable

- Example:

```
string sql = "DELETE FROM Students WHERE ID = " + txtID.Text;
```

- If user enters:

```
1 OR 1=1
```

- The query becomes:

```
DELETE FROM Students WHERE ID = 1 OR 1=1
```

- → This deletes **ALL** students

How to Prevent SQL Injection

- There are three main techniques to prevent SQL Injection:
- **Parameterized Queries (ADO.NET)** — BEST METHOD

```
string query = "SELECT * FROM Users WHERE Username = @username AND Password = @password";

SqlCommand cmd = new SqlCommand(query, con);
cmd.Parameters.AddWithValue("@username", txtUsername.Text);
cmd.Parameters.AddWithValue("@password", txtPassword.Text);

SqlDataReader dr = cmd.ExecuteReader();
```

- Why this is safe?
  - Parameters are sent separately from SQL code
  - SQL Server treats them as **data**, not **commands**

How to Prevent SQL Injection

- Using SqlParameter (Safer Alternative to AddWithValue)

```
cmd.Parameters.Add("@username", SqlDbType.VarChar, 50).Value = txtUsername.Text;
cmd.Parameters.Add("@password", SqlDbType.VarChar, 50).Value = txtPassword.Text;
```

- Stored Procedures

```
CREATE PROCEDURE LoginUser
  @username VARCHAR(50),
  @password VARCHAR(50)
AS
BEGIN
  SELECT * FROM Users
  WHERE Username = @username AND Password = @password
END
```