SQL Injection

O SQL Injection é um tipo de ataque que se aproveita de uma falha de segurança para acessar e manipular um determinado banco de dados. Para isso, no SQL Injection, o usuário insere instruções SQL nos campos de entrada do formulário. Por exemplo, se o sistema estiver desprotegido e a instrução **DELETE * FROM Users** for inserida no input, todos os dados da tabela Users (caso exista uma tabela nomeada assim) serão apagados.

Há outras instruções SQL que podem ser usadas no SQL Injection, como:

- ✓ ***Select*** (selecionar algo do banco);
- ✓ ***Creat database*** (criar um banco de dados do zero);
- ✓ ***Show databases*** (visualizar uma base de dados);
- ✓ ***Insert*** (inserir um dado);
- ✓ ***Update*** (atualizar um dado);
- ✓ ***Create table*** (criar uma tabela).

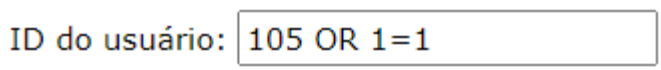
Todos os comandos listados podem ser inseridos no Front-End e afetar seu Back-End. Vale ressaltar, porém, que seu banco de dados só será atingido se existir uma falha de segurança, fazendo com que seu site fique vulnerável a esse tipo de ataque.

Analise o seguinte exemplo retirado do site **W3SCHOOLS** (https://www.w3schools.com/sql/sql_injection.asp).

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " + txtUserId;
```

No código acima, estamos definindo que uma variável (**txtSQL**) receberá como valor um **SELECT**. O **SELECT** acompanhado do asterisco seleciona todos os dados de **Users** (tabela), **Where** (onde) o **UserId** (trecho exemplo do código do SQL) seja " + **txtUserId** (**getRequestString**, em que "*get*" significa "obter", ou seja, a variável (**txtUserId**) obterá como valor a entrada do usuário).

Para ataques na web, imagine que o usuário passe como entrada um comando de:



No SQL, teremos:

```
SELECT * FROM Users WHERE UserId = 105 OR 1=1;
```

Assim, um hacker pode obter acesso a todos os nomes de usuários e senhas de um banco de dados inserindo, apenas, **105 OR 1 = 1** no campo de entrada. Para se proteger, é necessário implementar uma lógica na linguagem de programação que esteja integrada ao Front-End.

Outro erro de SQL pode ser:

```
uName = getRequestString("username");
uPass = getRequestString("userpassword");

sql = 'SELECT * FROM Users WHERE Name ="' + uName + '" AND Pass ="' + uPass + '"'
```

Como resultado, temos:

```
SELECT * FROM Users WHERE Name ="John Doe" AND Pass ="myPass"
```

Ou seja, um hacker pode obter acesso a nomes de usuários e senhas de um banco de dados simplesmente inserindo "**OU**" "**=**" na caixa de texto de nome de usuário ou senha.