A logo with a person in a hoodie using a computer

AI-generated content may be incorrect.

**MISSION HACKERS**

**BANGLADESH**

**Assignment No-09**

**Assignment Title: Web Application Hacking**

**Course Title: Cybersecurity & Ethical Hacking**

**Submitted by:**

**Name: Istiak Alam**

**Phone: 01765376101**

**Submission Date: 06-08-25**

**Lab Task Topic: SQL Injection, Burp Suit overview**

**Submitted to:**

**MD Sha Jalal**

**Founder of Mission Hackers Bangladesh**

**Part-01 SQL Injection and Commands**

**SQL injection :**

SQL Injection (SQLi) is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. [read more...](https://www.w3schools.com/sql/sql_injection.asp)

Attackers can use it to:

* **Bypass login pages**
* **Access hidden or private data**
* **Modify or delete data**
* **Execute administrative operations on the database**

## 🧠 Basic Concept

SQL Injection happens when **user input** is **incorrectly filtered** or **directly included in SQL queries**.

### Example of vulnerable SQL code:

**SELECT \* FROM users WHERE username = 'admin' AND password = '12345';**

Now imagine this is inside a PHP or JavaScript backend, like:

**$query = "SELECT \* FROM users WHERE username = '$username' AND password = '$password'";**

If the input is:

**username = admin' --**

**password = anything**

The query becomes:

**SELECT \* FROM users WHERE username = 'admin' -- ' AND password = 'anything';**

-- starts a comment in SQL. So the rest is ignored, making the query:

**SELECT \* FROM users WHERE username = 'admin';**

✅ This bypasses the password check!

## 🧪 Common SQL Injection Examples

### 1. **Login Bypass**

**' OR 1=1 --**

This turns the query into:

**SELECT \* FROM users WHERE username = '' OR 1=1 -- ' AND password = '';**

This always returns true.

### 2. **Retrieving All Data**

**' UNION SELECT \* FROM users --**

Used when the result is shown on a page.

### 3. **Detecting SQL Injection**

Use ' or " in fields to see if there's an error:

**'**

If we get a SQL error like:

**You have an error in your SQL syntax...**

That means it might be vulnerable!

## 🔍 Basic Types of SQL Injection

| Type | Description |
| --- | --- |
| **In-band** | Data is extracted using the same communication channel |
| - Error-based | Uses SQL errors to get info |
| - Union-based | Uses UNION to get data from other tables |
| **Inferential (Blind)** | No data is shown, but you can observe app behavior |
| - Boolean-based | Sends true/false payloads and checks the response |
| - Time-based | Uses delays (e.g., SLEEP(5)) to detect execution |
| **Out-of-band** | Data is sent through different channels (e.g., DNS or HTTP) |

## 🛠️ Sample SQL Injection Lab

Try this practice URL (if available):

**http://testphp.vulnweb.com**

**Login bypass example**:  
Try:

**username: admin' --**

**password: anything**

## 🧰 Tools for SQL Injection

| Tool | Description |
| --- | --- |
| **sqlmap** | Fully automated SQLi tool |
| **Burp Suite** | Manual & automated testing of web apps |
| **Havij** | GUI SQLi tool (old but useful for practice) |

## 🔐 How to Prevent SQL Injection

* ✅ Use **prepared statements** (e.g., in PHP: mysqli\_prepare())
* ✅ **Validate user input** (whitelisting, regex)
* ✅ Use **ORM frameworks** like SQLAlchemy, Hibernate
* ✅ Limit **database user privileges**
* ✅ Employ **WAFs** (Web Application Firewalls)

## 🎓 Practice Lab

* [PortSwigger Labs (Web Security Academy)](https://portswigger.net/web-security/sql-injection)
* [HackTheBox & TryHackMe](https://tryhackme.com/)
* [bWAPP, DVWA, WebGoat](https://owasp.org/) - Safe test environments