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

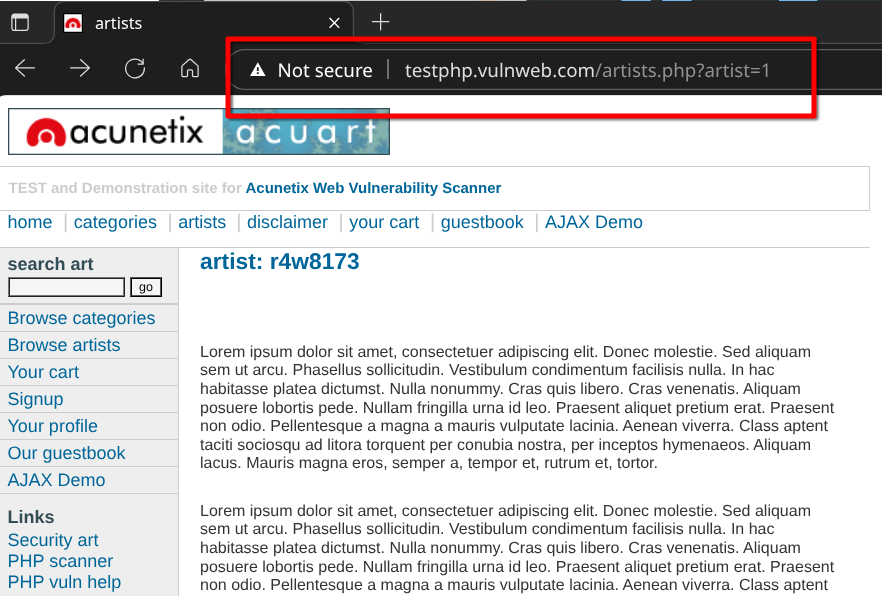
**Testing If the website is SQL injection able :**

First We have to test the page and insure if SQL injection is possible or not in this site.

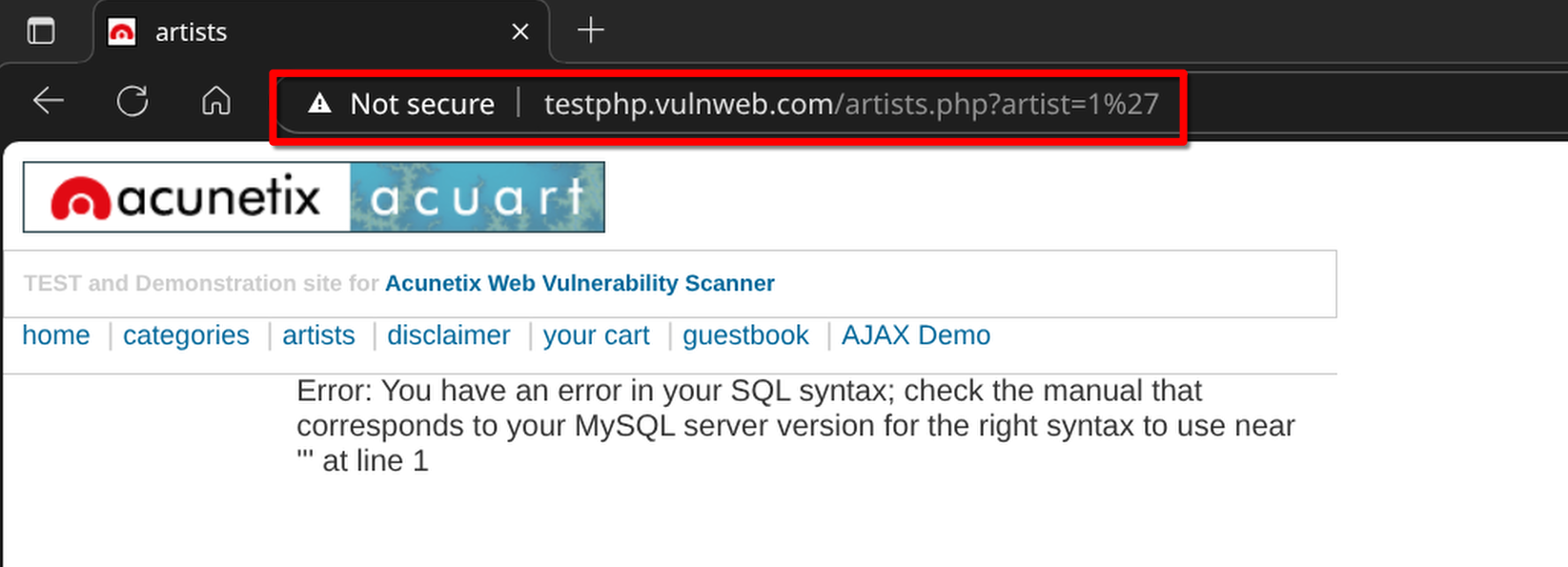
Step 1 : Go to <http://testphp.vulnweb.com/login.php>



Step 2 : Go to your cart or else use this link <http://testphp.vulnweb.com/artists.php?artist=1>



Step 3 : On the end of the link add a strick(‘) or %27 and see if it shows any error



Here is the Vulnerable Error which is called Error Based SQL injection.

**Part-02 Introduction to Burp Suit**

**What is Burp Suit…?**

**Burp Suite** is an integrated platform for performing **security testing of web applications**. It works by acting as a **proxy** between your browser and the target website, allowing you to intercept, modify, and replay requests and responses.

## 💻 Burp Suite Editions

| Edition | Description |
| --- | --- |
| **Community** | Free, with limited features (great for learning) |
| **Professional** | Paid, full-featured version |
| **Enterprise** | For automated large-scale scans |

## 🧰 Tools in Burp Suite (Main Tabs)

| Tool Name | Use |
| --- | --- |
| **Proxy** | Intercept & modify HTTP/S requests from the browser |
| **Target** | View website structure and vulnerabilities |
| **Intruder** | Automate attacks like brute-force, fuzzing, etc. |
| **Repeater** | Manually modify & resend requests |
| **Scanner** (Pro) | Automatic vulnerability scanner |
| **Decoder** | Encode/decode Base64, URL, HTML, etc. |
| **Comparer** | Compare two HTTP responses or requests |
| **Extender** | Install community extensions or your own scripts |

## **🛠️ Setup Burp Suite with Browser (Manual or Extension)**

### Step 1: Run Burp Suite (on Kali Linux or Windows)

**burpsuite**

### Step 2: Configure Browser Proxy

Set proxy to:

* **IP:** 127.0.0.1
* **Port:** 8080

Or install the **Burp Suite CA Certificate**:

1. Open browser → visit [http://burpsuite](http://burpsuite/)
2. Download **CA Certificate**
3. Import it into your browser’s certificate store (for HTTPS interception)

### Step 3: Intercept Traffic

* Go to **Proxy → Intercept → ON**
* Visit any website in your browser → Request is captured by Burp
* Modify or forward the request

## 🔥 Burp Suite Practical Use Cases

### 1. **Intercept Login Requests**

* Login to a site (e.g., DVWA)
* Burp will capture the POST request
* We’ll see:

**POST /login.php HTTP/1.1**

**Host: dvwa.local**

**username=admin&password=admin&Login=Login**

* Send it to **Repeater** → Try injecting payloads manually

### 2. **Using Repeater for Manual Testing**

* Click **Send to Repeater** (Right-click the request)
* Modify parameters, headers, cookies
* Click **Send** and review the response

Useful for testing:

* SQLi
* XSS
* File Inclusion
* Authentication bypass

### 3. **Using Intruder for Brute Force or Fuzzing**

1. Right-click → Send to **Intruder**
2. Select positions to attack (e.g., username or password field)
3. Add a wordlist (e.g., rockyou.txt)
4. Start attack

**Use Cases**:

* Brute-forcing login credentials
* Finding hidden parameters
* Fuzzing headers, cookies, inputs

### 4. **Decoder**

* Decode things like:
  + Base64 → **aGVsbG8=**
  + URL encoded → **%3Cscript%3Ealert(1)%3C/script%3E**
* Encode or hash data for testing

### 5. **Comparer**

* Useful for **differential analysis** — comparing responses from different payloads
* Can help identify blind vulnerabilities

## 🧪 Practice Labs for Burp Suite

You can practice Burp Suite with these:

| Platform | Description |
| --- | --- |
| **DVWA** | Damn Vulnerable Web App (install locally) |
| **bWAPP** | Buggy Web App for testing vulnerabilities |
| **PortSwigger Labs** | [Burp’s own Web Security Academy](https://portswigger.net/web-security) |
| **TryHackMe** | Web Hacking Labs with Burp integrated |
| **HackTheBox** | Realistic machines and labs |

## 📚 Common Burp Suite Attack Types

| Vulnerability | Use Burp to... |
| --- | --- |
| **SQL Injection** | Intercept & inject SQL payloads |
| **XSS** | Test parameters with script tags |
| **CSRF** | Create and send fake requests |
| **Auth Bypass** | Modify cookies, tokens, or parameters |
| **File Upload Bypass** | Change file extension/content |
| **Open Redirect** | Modify redirect parameters |

## 🧠 Tips for Beginners

* Always turn **Intercept OFF** when done, or browser won't load
* Use **Repeater** for manual testing (less noise)
* Install useful **extensions** from BApp Store
  + **Logger++**
  + **Autorize**
  + **ActiveScan++**

## ✅ Want a Real-Life Practice?

I can walk you through:

1. **Login Bypass** with Burp Suite (DVWA)
2. **Fuzzing with Intruder**
3. **Capturing Session Hijacking**
4. **Finding Hidden Directories**

Just tell me your current setup:

* Do you have **Burp Suite + DVWA** installed?
* Or do you want to use **PortSwigger's Online Labs**?

Let me know, and I’ll guide you in **hands-on hacking with Burp Suite.**

**Part-03 Burp suit Setup and Execution**

Reference links :

1. [PayloadAllThings](https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/SQL%20Injection)

2. [W3School](https://www.w3schools.com/sql/sql_injection.asp)

3. [PortSwigger Labs (Web Security Academy)](https://portswigger.net/web-security/sql-injection)

4. [TryHackMe](https://tryhackme.com/)

5. B[WAPP, DVWA, WebGoat](https://owasp.org/) - Safe test environments