# Web Application Penetration Testing Report

**Project Title:** Web Application Penetration Testing Report
**Target:** PortSwigger Academy Lab – SQLi Authentication Bypass
**Analyst:** G Narendran
**Date:** 28-June-2025
**Tools Used:** Burp Suite (Community Edition), Web Browser

## 1. Executive Summary

This report documents the penetration testing performed on a simulated vulnerable web application provided by PortSwigger Academy. The main objective was to identify and exploit SQL Injection (SQLi) vulnerabilities in the login form to bypass authentication controls. The test successfully uncovered a critical SQL Injection flaw, allowing unauthorized access to the administrator account without valid credentials. The engagement was conducted in a controlled lab environment for educational and ethical purposes.

---

## 2. Scope of Testing

- **Target URL:** PortSwigger SQLi Lab
- **Functionality Tested:** Login form (`/login`)
- **Tested For:**
    1. SQL Injection (SQLi)
    2. Authentication Bypass
    3. Basic Error-Based SQL Responses

The scope included testing input handling at the login endpoint to identify any lack of input sanitization or improper SQL query construction.

## 3. Tools Used

✓ **Burp Suite (Community Edition):** Used to intercept, analyze, and manipulate HTTP requests during the login process.
✓ **Browser (Chrome/Firefox):** Used to interact with the lab interface and monitor redirection and lab completion.

# 4. Methodology

### 1. Reconnaissance

- Navigated to the lab-provided login page.
- Identified `/login` as the primary POST endpoint for authentication.
- Captured and analyzed HTTP request using Burp Suite's Proxy feature.

### 2. Payload Injection

- Modified the `username` field in the intercepted request to:
- `administrator'--`
- Kept the CSRF token intact from the original request to avoid CSRF protection errors.
- Used any random string as the password since password validation was bypassed due to the SQLi.

### 3. Authentication Bypass Confirmation

- The server responded with an **HTTP 302 redirect**, indicating a successful login.
- The browser was redirected to the admin dashboard.
- PortSwigger marked the lab as **"Solved"**, confirming administrator access without valid credentials.

# 5. Vulnerability Details

### ● SQL Injection in Login (Authentication Bypass)

- **Vulnerability:** SQL Injection
- **Endpoint:** `POST /login`
- **Parameter Affected:** `username`
- **Payload Used:** `administrator'--`

### Proof of Concept

- **Username:** `administrator'--`
- **Password:** `anything`
- **Result:** Logged in as administrator, successfully bypassed authentication

### Impact

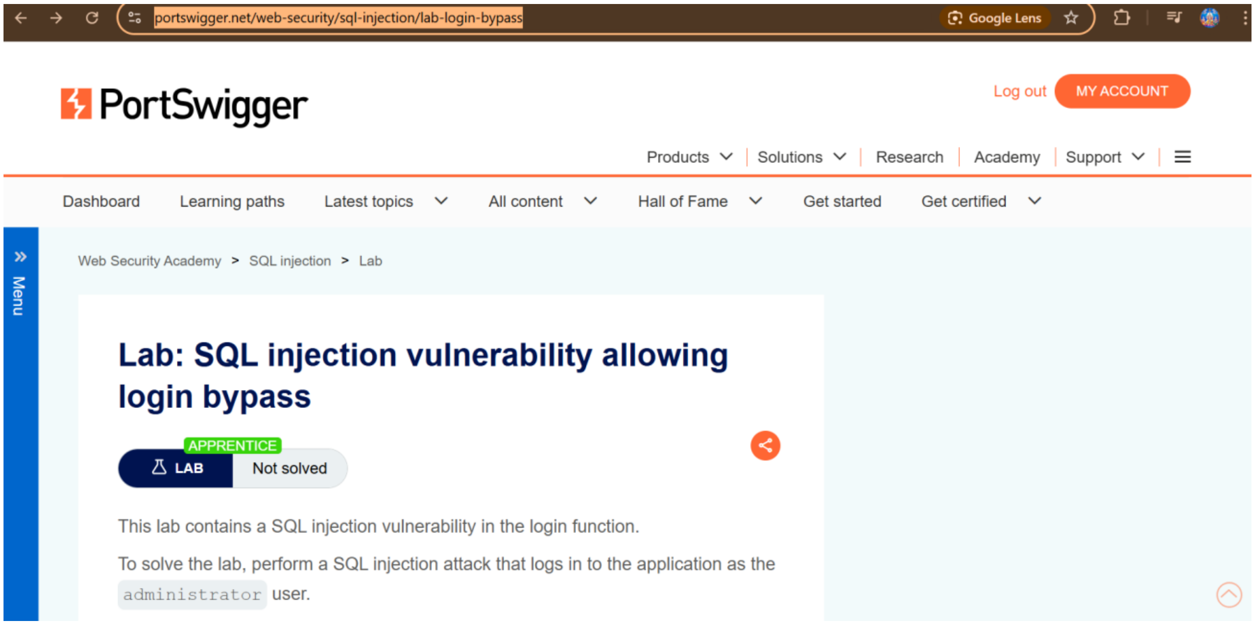
This vulnerability allows an attacker to:

- Bypass login mechanisms
- Gain unauthorized access to sensitive accounts
- Escalate privileges in vulnerable applications

## 6. Recommendations

- Implement **prepared statements (parameterized queries)** to avoid direct SQL query concatenation.
- Use a **robust input validation/sanitization layer** for all user input.
- Avoid displaying database error messages directly to users.
- Regularly conduct security audits and penetration testing to detect similar issues.

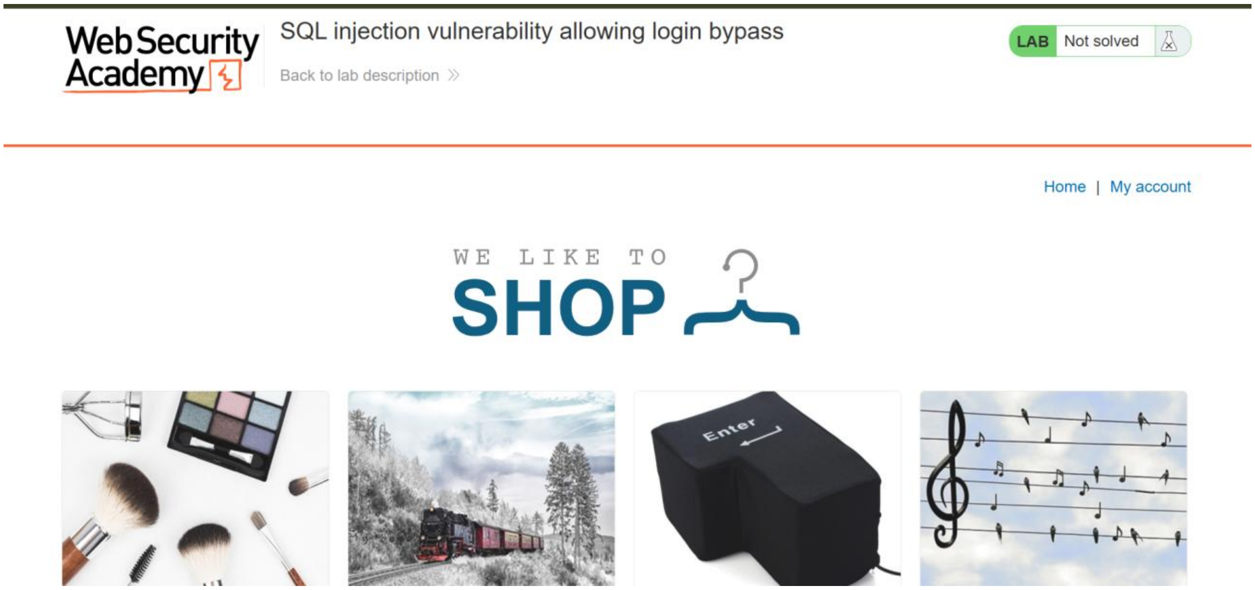
# WORKING SCREENSHOTS



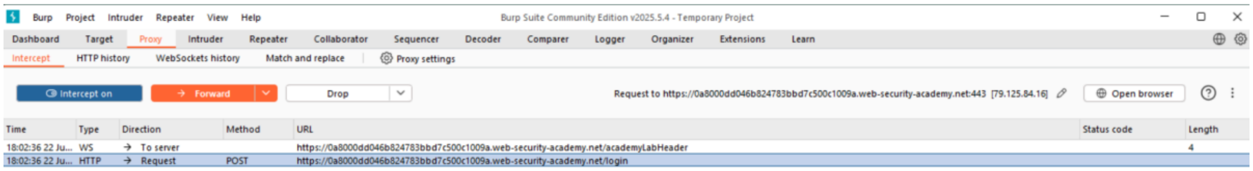
Lab: SQL injection vulnerability allowing login bypass

This lab contains a SQL injection vulnerability in the login function.

To solve the lab, perform a SQL injection attack that logs in to the application as the `administrator` user.



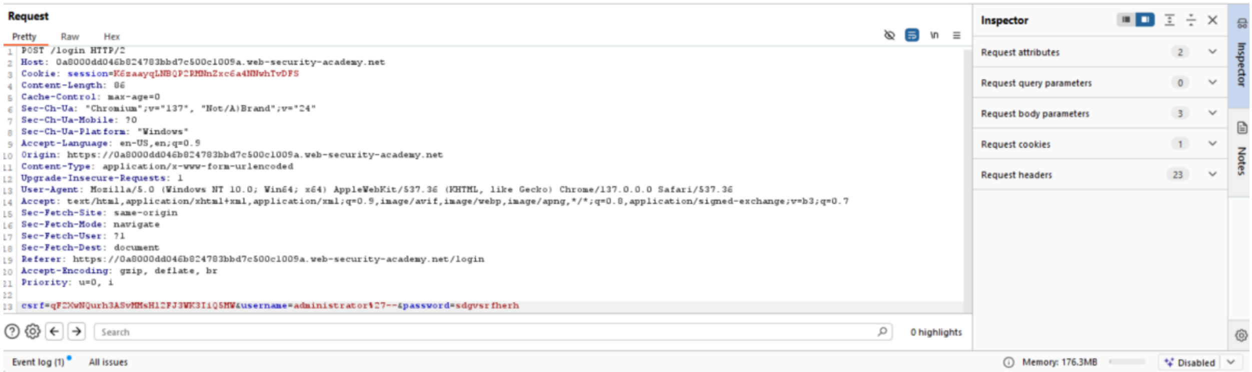
SQL injection vulnerability allowing login bypass



Burp Suite Community Edition v2025.5.4 - Temporary Project

Request to https://0a8000dd046b824783bbd7c500c1009a.web-security-academy.net:443 [79.125.84.16]

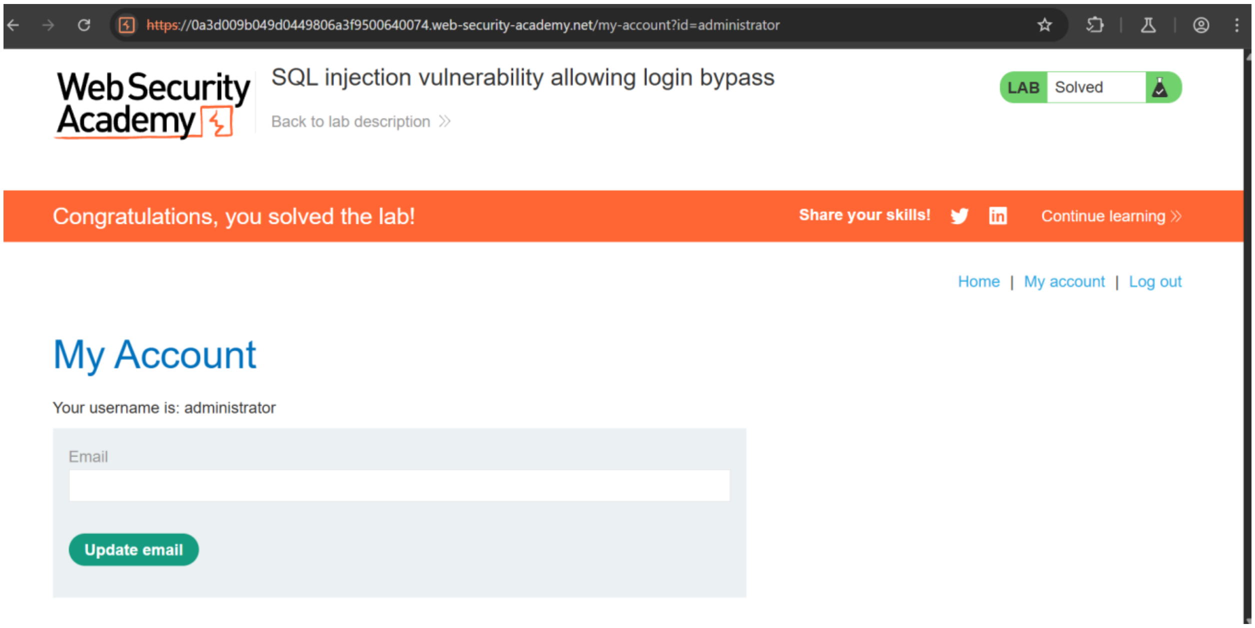
| Time | Type | Direction | Method | URL | Status code | Length |
|---|---|---|---|---|---|---|
| 18:02:36 22 Ju... | WS | → To server | | https://0a8000dd046b824783bbd7c500c1009a.web-security-academy.net/academyLabHeader | | 4 |
| 18:02:36 22 Ju... | HTTP | → Request | POST | https://0a8000dd046b824783bbd7c500c1009a.web-security-academy.net/login | | |



```
POST /login HTTP/2
Host: 0a8000dd046b824783bbd7c500c1009a.web-security-academy.net
Cookie: session=K6zaayqLNBQP2RMNnZxc6a4NNwhTvDFS
Content-Length: 86
Cache-Control: max-age=0
Sec-Ch-Ua: "Chromium";v="137", "Not/A)Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Accept-Language: en-US,en;q=0.9
Origin: https://0a8000dd046b824783bbd7c500c1009a.web-security-academy.net
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/137.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://0a8000dd046b824783bbd7c500c1009a.web-security-academy.net/login
Accept-Encoding: gzip, deflate, br
Priority: u=0, i

csrf=qFZXwNQurh3ASvMMsH12FJ3WK3IiQ5MW&username=administrator%27--&password=sdgvsrfherh
```

| | |
|---|---|
| Request attributes | 2 |
| Request query parameters | 0 |
| Request body parameters | 3 |
| Request cookies | 1 |
| Request headers | 23 |



https://0a3d009b049d0449806a3f9500640074.web-security-academy.net/my-account?id=administrator

SQL injection vulnerability allowing login bypass

Back to lab description »

LAB Solved

Congratulations, you solved the lab!

Share your skills! Continue learning »

Home | My account | Log out

My Account

Your username is: administrator

Email

Update email

# 7. Conclusion

The login mechanism in the PortSwigger lab demonstrated a critical SQL Injection vulnerability allowing authentication bypass. The lack of input sanitization on the `username` parameter enabled the bypass of password verification through query manipulation. This exercise emphasizes the importance of secure coding practices, including the use of parameterized queries or prepared statements in login logic.

As a penetration tester, this lab provided hands-on experience in identifying and exploiting SQLi vulnerabilities and highlighted the real-world impact of insecure login implementations.