By Jacinta Izundu

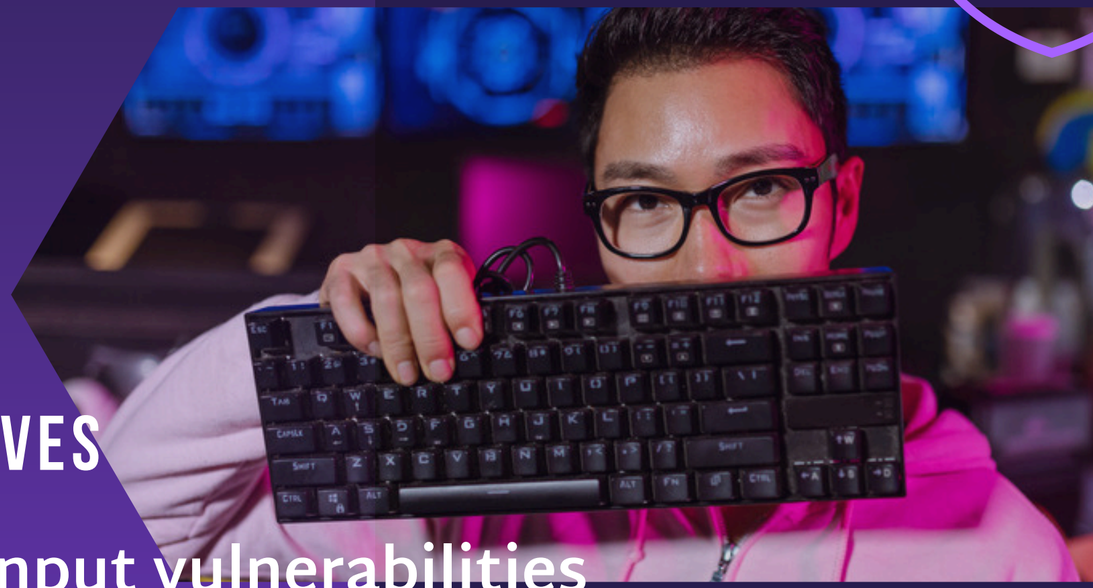
EXPLOITATION OF WEB INPUT VULNERABILITIES: SQL INJECTION & CROSS-SITE SCRIPTING DEMONSTRATION

Executing Real-World Attack Scenarios to Reveal Input Validation Weaknesses

OVERVIEW OF THIS EXERCISE

In this lab, we performed three common web attacks:

- SQL Injection (2 labs)
- Reflected Cross-Site Scripting (XSS)
- Stored Cross-Site Scripting (XSS)



LEARNING OBJECTIVES

- Understand web input vulnerabilities
- Learn how SQL queries can be manipulated
- Demonstrate how malicious scripts execute in browsers
- Recognize the importance of input validation & sanitization

WHAT IS SQL INJECTION

SQL Injection occurs when an attacker manipulates input fields or URL parameters to modify backend SQL queries.

WHY IT'S DANGEROUS

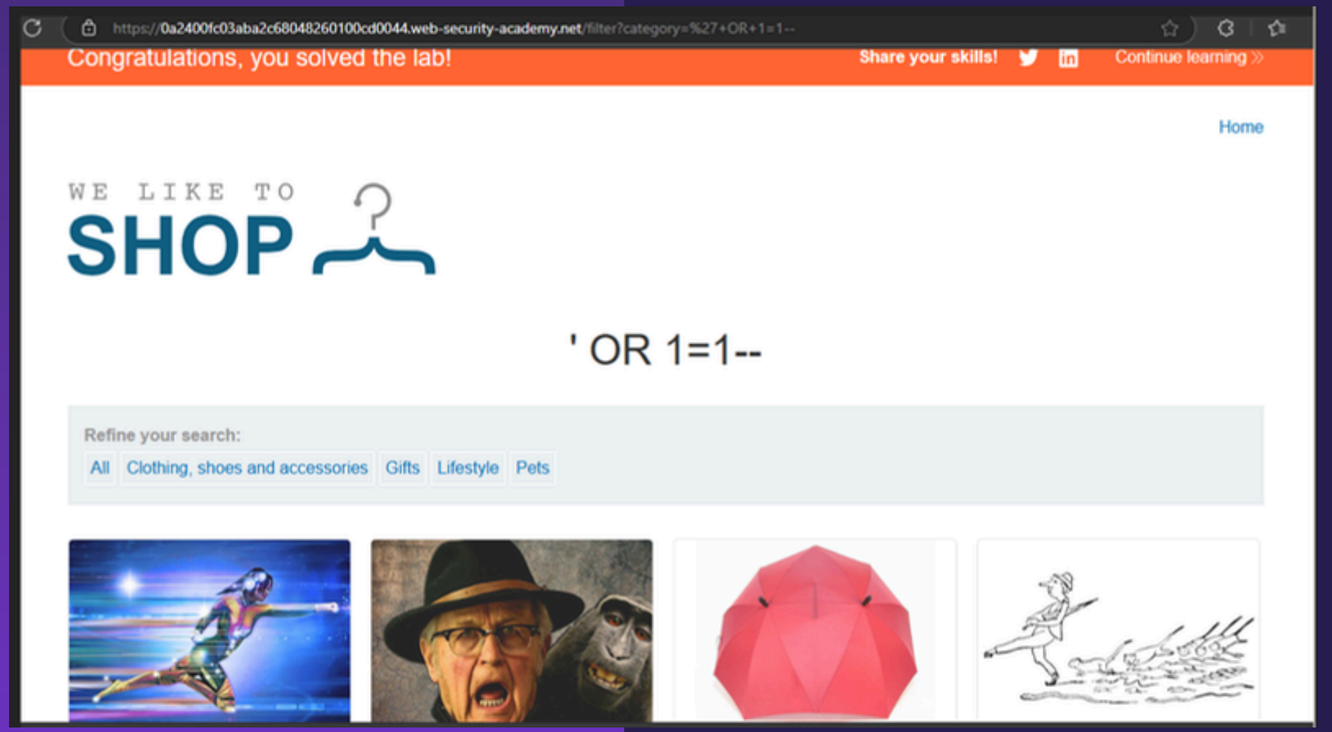
- Bypass authentication
- Retrieve or modify data
- Dump entire database tables
- Take full control of applications

SQL INJECTION LAB 1: CATEGORY INJECTION

We replaced the category parameter with a malicious SQL payload to return all products

HOW IT WORKS

- 1=1 always evaluates TRUE
- -- comments out the rest of the SQL query
- The backend returns every product



EXPLANATION: SQL INJECTION LAB 1

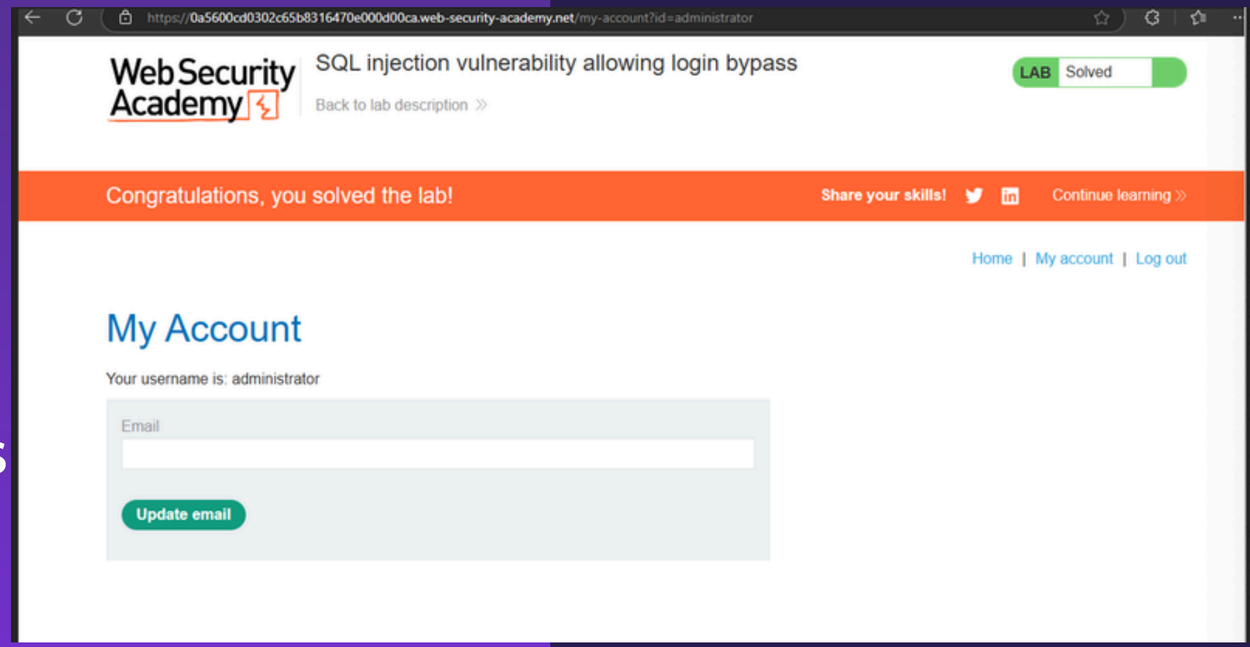
I carried out a SQL Injection by modifying the URL parameter for the category. Instead of a valid category, I entered '+OR+1=1--, which forced the backend SQL query to return all products by making the condition always true (1=1). The double dash (--) commented out the rest of the original query. This showed that the application did not properly sanitize input before using it in a SQL query.

SQL INJECTION LAB 2: LOGIN BYPASS

In the login form, a crafted username bypassed authentication

WHY IT WORKS

- ' closes the username field
- - starts an SQL comment
- Password check becomes irrelevant
- Login succeeds without knowing credentials



WHAT IS XSS (CROSS-SITE SCRIPTING)

Cross-Site Scripting (XSS) occurs when attackers inject JavaScript or HTML that the website displays without sanitizing

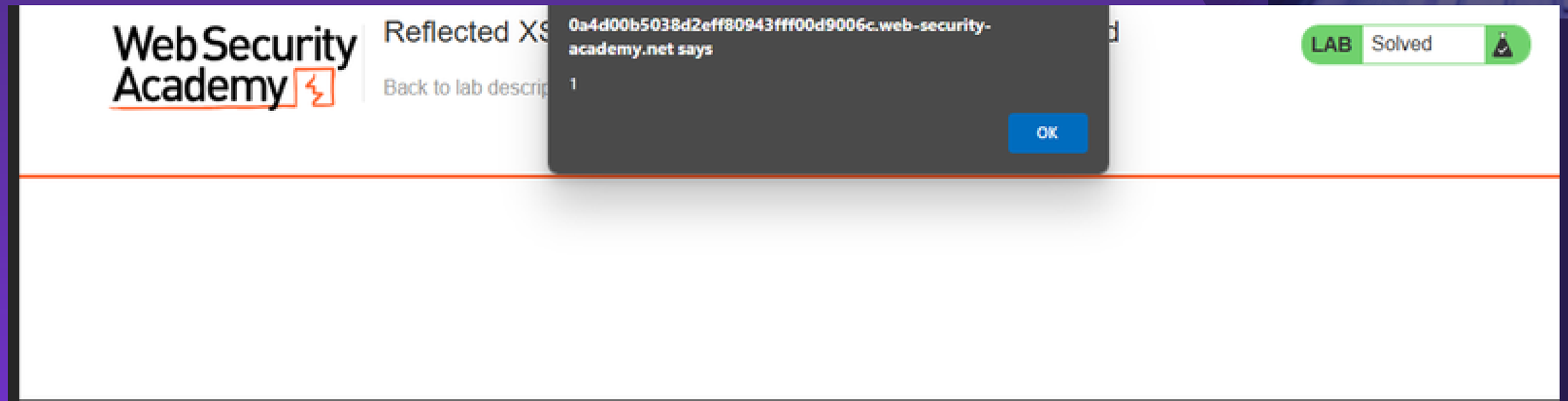
WHY IT'S DANGEROUS

- Steal cookies/session tokens
- Redirect vitims
- Deface websites
- Install malware
- Steal login information

REFLECTED XSS ATTACK

HOW IT WORKS

- Attacker inputs a malicious script
- The script is immediately reflected back in the HTTP response
- Browser executes the script

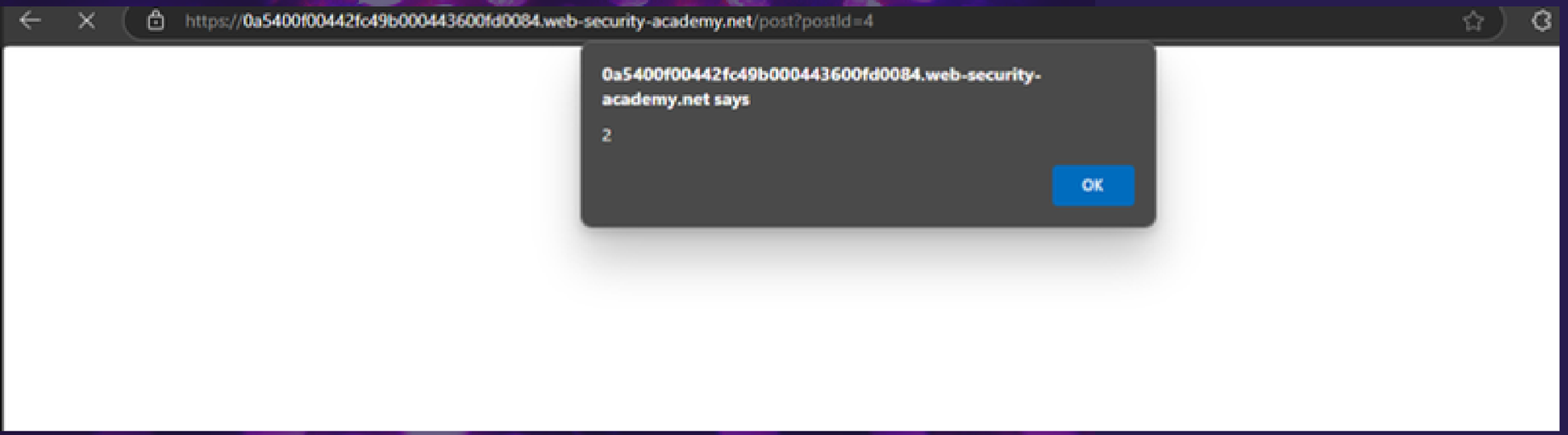


EXPLANATION: REFLECTED XSS

I performed a reflected XSS attack by entering <script>alert(1)</script> into the search box. The website reflected this input back onto the page without filtering it, so my JavaScript code was executed in the browser. This is dangerous because an attacker could inject scripts that steal cookies or redirect users.

STORED XSS ATTACK

Stored XSS is more dangerous, the script is saved in the database and runs for every vistor.



RESULT

Every time blog page loads, the alert box appers for all users

EXPLANATION: STORED XSS

I carried out a stored XSS attack by submitting a blog comment containing <script>alert(2)</script>. The comment was saved and displayed to future visitors without being sanitized. When the page was loaded, the browser ran the script and showed an alert box. This shows the site is vulnerable to stored XSS and can be exploited to run malicious code for every viewer.

KEY DIFERENCES: REFLECTED VS STORED XSS

Reflected XSS	**Stored XSS**
Temporary	Permanent
Triggered by attacker’s link	Triggered anytime page loads
Not stored in DB	Stored in DB
Less severe	More severe

HOW THESE ATTACKS COULD BE PREVENTED

SQL INJECTION PREVENTION

- Use prepared statements
- Use parameterized queries
- Escape special characters
- Avoid constructing SQL with string concatenation

XSS PREVENTION

- Sanitize user input
- Encode output
- Use Content Security Policy (CSP)
- Validate input strictly

SUMMARY & FINAL THOUGHTS

- SQL Injection exploits database queries
- Reflected XSS exploits immediate unsanitized output
- Stored XSS affects all future visitors
- Proper input sanitization & secure coding stop these attacks
- Hands-on practice demonstrates how real systems can be compromised

THANK YOU!

+1 217 670 4436

clemsjacy@gmail.com

LinkedIn http://linkedin.com/in/jacinta-izundu-b36a20233