

# Mitigating Vulnerabilities in an Educational Website

# 12th Project – Milestone 2

Security in Software Engineering

Dr. Nada Hany Sherief Eng: Salma Elkady

***Team Members :***

Ahmed Elbialy\_211008712 Habiba Ahmed\_211003103

Ahmed Yasser\_211010273 Fares Fawaz\_19101617

Basel Mohamed\_211001137

**Part 1 - E-Learning System with Secure Logging and Secure Factory Pattern.**

**<https://github.com/ahmedelbialy1/Security-in-SE-Milestone2/blob/main/E-Learning-System-Java-GUI-Application-main.rar>**

**Part 2**

**\_\_\_\_\_\_**

**Penetration Testing Report**

**Target: Mutillidae on Metasploitable 2**  
**Tools Used:** Burp Suite, OWASP ZAP Custom Wordlists, Browser Developer Tools,VMWare

**Executive Summary**

This report presents a comprehensive penetration test on **Mutillidae**, a deliberately vulnerable web application hosted on **Metasploitable 2**. The testing focused on identifying and exploiting common web vulnerabilities including **SQL Injection**, **Cross-Site Scripting (XSS)**, and **Brute Force Attacks**, as well as **spidering and target scoping** to identify potential attack vectors. The goal was to assess risks, exploit vulnerabilities, and provide mitigation strategies.

**1. Scope of Work**

| **Area** | **Goal** |
| --- | --- |
| Spidering & Target Scoping | Identify all reachable pages, parameters, forms |
| SQL Injection | Detect injection points in inputs, login fields |
| Cross-Site Scripting | Detect reflected & stored XSS vulnerabilities |
| Brute Force Attacks | Test login pages using automated attack vectors |

**2. Methodology**

**Information Gathering**

* **Spidering:**
  + Tool: Burp Suite Spider
  + Result: 60+ unique endpoints and parameters discovered
* **Target Scoping:**
  + High-Value Targets:
    - login.php
    - search.php
    - add-to-your-blog.php
    - user-info.php
    - Comment Forms

**3. Vulnerability Assessment**

**SQL Injection**

* **Targets Tested:** login.php, user-info.php, search.php
* **Technique:**
  + Manual tampering and automated testing using **Burp Suite Intruder**
* **Findings:**
  + search.php?q=' OR '1'='1 returns full dataset → **Vulnerable**
  + login.php bypassed with ' OR '1'='1 → **Authentication Bypass**
* **Proof of Concept (PoC):**

sql

CopyEdit

Username: ' OR '1'='1 --

Password: [any]

**Cross-Site Scripting (XSS)**

* **Reflected XSS:**
  + Target: search.php
  + Payload: <script>alert('XSS')</script>
  + Result: JavaScript executes on result page → **Confirmed**
* **Stored XSS:**
  + Target: add-to-your-blog.php
  + Payload: <script>alert('Stored XSS')</script>
  + Result: Code stored and triggers on blog view page

**4. Brute Force Attacks**

* **Target:** login.php
* **Tool:** Burp Suite Intruder
* **Technique:**
  + Wordlist-based attack using top 100 passwords (rockyou subset)
  + Username: admin
* **Findings:**
  + Password cracked: admin using admin:admin credentials
  + No account lockout or CAPTCHA → **Vulnerable**

**5. Exploitation & Impact Assessment**

| **Vulnerability** | **Exploitation Outcome** | **Impact** |
| --- | --- | --- |
| SQL Injection | Extracted users table, accessed passwords | Unauthorized data access, potential RCE |
| Reflected XSS | Alert box shown | Could lead to phishing, session hijacking |
| Stored XSS | Persistent JS on blogs | Hijacks user sessions, steals cookies |
| Brute Force | Gained access to admin account | Full application access |

**6. Recommendations**

**SQL Injection**

* Use parameterized queries (prepared statements)
* Sanitize all user inputs
* Apply Web Application Firewall (WAF)

**XSS**

* Encode output (HTML, JavaScript context)
* Sanitize and validate user inputs
* Use Content Security Policy (CSP)

**Brute Force**

* Implement account lockout mechanisms
* Use CAPTCHA and multi-factor authentication
* Monitor login attempts and IP throttling

**General**

* Conduct regular vulnerability scans
* Educate developers on secure coding
* Update and patch all components

**7. Appendix**

**Tools Used**

* Burp Suite Professional
* Firefox Developer Tools
* Wordlists (rockyou, custom)

**Screenshots and Logs**