Lab 04: Vulnerable Web Application

1. Web Application Features

1.1. Purpose of the Application:

The primary purpose of this web application is to demonstrate SQL Injection (SQLi) and Cross-Site Scripting (XSS) vulnerabilities for educational purposes.

1.2. Features Offered:

The application offers:

- User Greeting (vulnerable to XSS)
- User Comment Lookup (vulnerable to SQLi)
- Simulated Admin Credentials Storage

Figure 1: Docker Desktop Showing Running Container

2. Identifying Vulnerabilities

2.1. SQL Injection (SQLi)

The application is vulnerable to SQL Injection in the User Comment Lookup. User input is directly used in the SQL query.
Location in Code ('src/index.php'):

```
$sql_id = $_GET['id'];
$query = "SELECT name, comment FROM users WHERE id = $sql_id";
$result = $db->query($query);
```

2.2. Cross-Site Scripting (XSS - Reflected)

The application is vulnerable to Reflected XSS in the User Greeting. User input is directly output to the HTML.
Location in Code ('src/index.php'):

```
$xss_name = $_GET['name'];
// ...
<p>Hello, <?php echo $xss_name; ?>!</p>
```

3. Black-Box Testing

3.1. Tools and Techniques:

- **Manual Input Probing:**
  - **SQLi:** $Testwith`'`, `1OR1 = 1`, `0UNIONSELECTusername, password_hashFROMadmin_credentials`.
  - **XSS:** $Testwith` < script > alert('XSS') < /script > `, ` < imgsrc = xonerror = alert(1) > `.
- **Browser Developer Tools:** Inspect requests/responses.

3.2. Signs or Responses:

- **SQLi:** Database errors, unexpected data.
- **XSS:** JavaScript execution (e.g., alert box).

4. How to Exploit

4.1. SQL Injection

Objective: Retrieve admin credentials.
Payload: '0 UNION SELECT username, password$_{h}ashFROMadmin_{c}redentials$'
Steps:

1. Go to http://localhost:8088/.
2. Enter payload in "Enter User ID to view comment".
3. Click "View Comment".
4. **Observation:** Admin credentials are displayed. (See Figure 2)

Proof: Figure 2 shows the retrieved data.

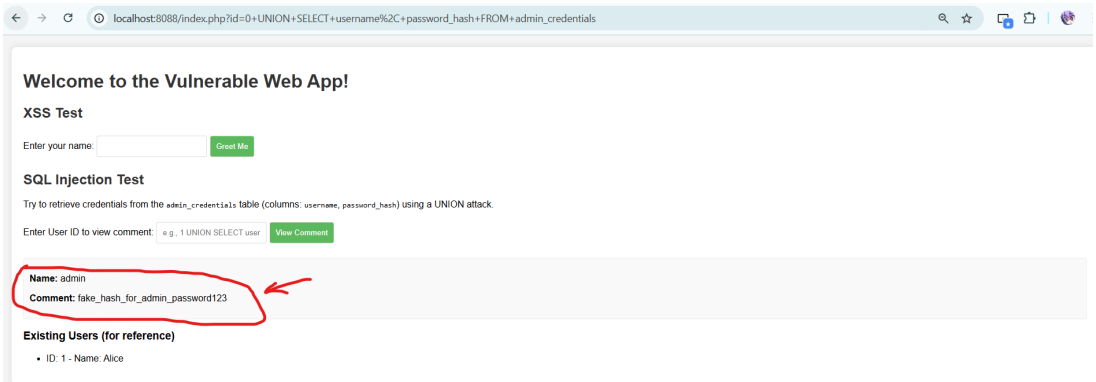

Figure 2: SQL Injection Exploit

4.2. XSS

Objective: Execute JavaScript.
Payload: "'$html < script > alert('XSSAttack!'); < /script >$ "
Steps:

1. Go to http://localhost:8088/.
2. Enter payload in "Enter your name".

3. Click "Greet Me".
4. **Observation:** An alert box appears. (See Figure 3)

Proof: Figure 3 shows the alert box.

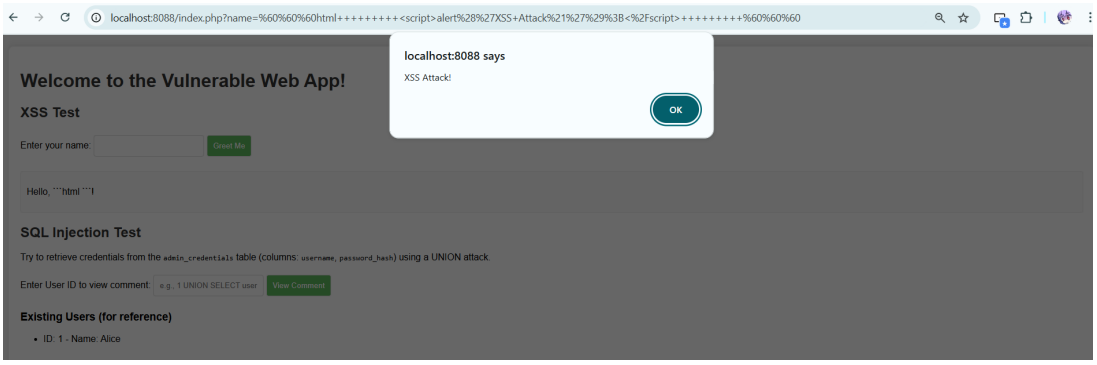

Figure 3: XSS Exploit

5. Root Cause

5.1. SQL Injection:

User input is directly used in the SQL query without proper sanitization.

5.2. XSS:

User input is directly output to the HTML without encoding.

6. Proposed Mitigations

6.1. SQL Injection:

The primary solution to prevent SQL Injection is to use **parameterized queries** (also known as prepared statements). This technique separates the SQL code from the user-provided data, preventing the database from interpreting the data as part of the SQL command. Input validation can be used as an additional layer of defense, but it is not sufficient on its own. The database user should also be granted the minimum necessary privileges to limit the potential damage from a successful attack.

6.2. XSS:

The primary solution to prevent Reflected XSS is to use **contextual output encoding**. This involves converting potentially dangerous characters in user-provided data into their safe HTML entities before displaying them on the page. Input validation can also be used as a secondary defense. Additionally, implementing a **Content Security Policy (CSP)** can further mitigate the risk by controlling which resources the browser is allowed to load. Setting the **HttpOnly flag on cookies** can also help reduce the impact of some XSS attacks by preventing JavaScript from accessing those cookies.

Deliverables

GitHub Repository Link: https://github.com/ThoSugoi/vulnerable-php-app
Docker Proof:
The application can be run using Docker. The following commands were used:

```
docker build -t my-vulnerable-app .
docker run -p 8088:80 my-vulnerable-app
```

The application is then accessible at `http://localhost:8088/`. (See Figure 4 for Docker Desktop screenshot).

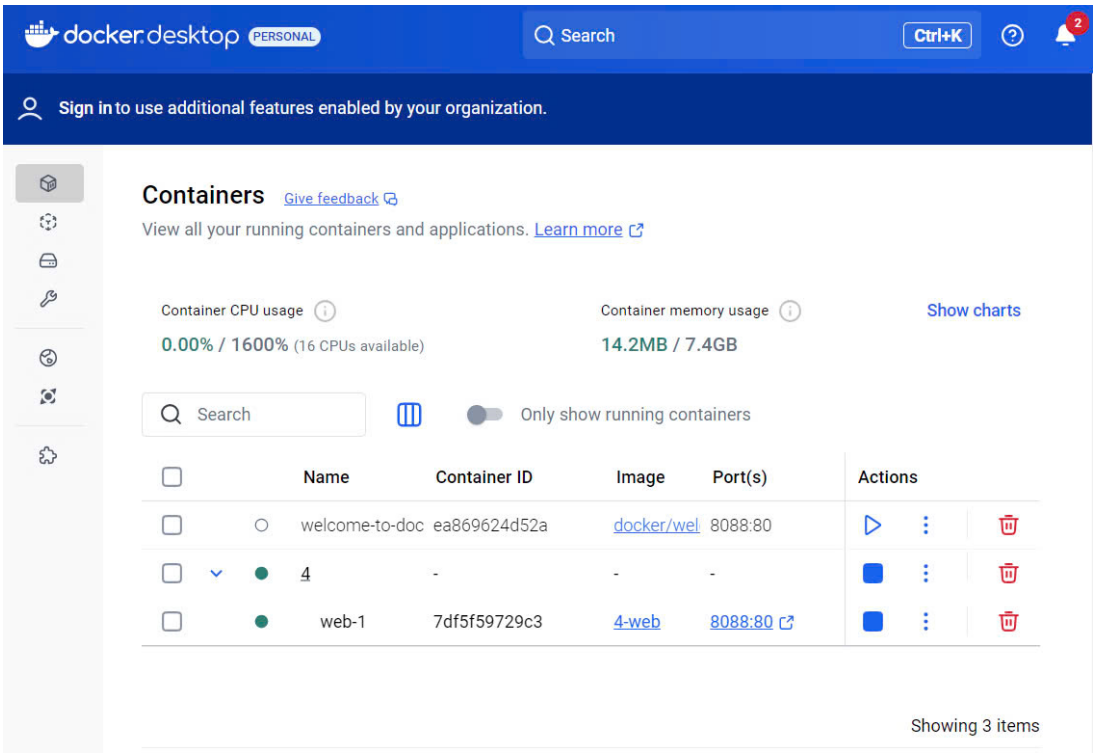

Figure 4: Docker Desktop Showing Running Container