# Лабораторная работа №2 Исследование уязвимостей ПО и методов их устранения

1. Цель работы
   1. Изучить основные угрозы безопасности веб-приложений, такие как SQL-инъекции, XSS-инъекции, а также методы защиты от них;
   2. Научиться применять защитные меры в разработке безопасных веб-приложений.
2. Литература
   1. Зверева В. П., Сопровождение и обслуживание программного обеспечения компьютерных систем : учебник для студ. учреждений сред. проф. Образования / В. П. Зверева, А. В. Назаров. – М. : Издательский центр «Академия», 2018. – 256 с.
3. Подготовка к работе
   1. Повторить теоретический материал (см. п.2).
   2. Изучить описание лабораторной работы.
4. Основное оборудование
   1. Персональный компьютер.
5. Задание
   1. Используя приложение п. 9, реализовать и запустить приложение, подверженное уязвимостям;
   2. Протестировать SQL-инъекции, выполнить авторизацию без учетных данных, а также при помощи объединения таблиц комментариев и пользователей вывести на странице комментариев учетные данные пользователей.
   3. Протестировать XSS-инъекции, выполнить инъекцию кода заменяющего гиперссылку пункта меню «Главная» на другую произвольную страницу.
   4. Протестировать уязвимости авторизации, изменить почту другого пользователя, опубликовать комментарий от имени несуществующего пользователя.
   5. Исправить код с уязвимостями используя рекомендации из приложения п.9.2
6. Порядок выполнения работы
   1. Повторить теоретический материал п. 3.1;
   2. Исследовать уязвимости веб-приложений ПО п. 5.1-5.4;
   3. Ответить на контрольные вопросы п. 8;
   4. Заполнить отчет п. 7.
7. Содержание отчета
   1. Титульный лист;
   2. Цель работы;
   3. Код исправленных модулей п. 5.3;
   4. Ответы на контрольные вопросы п. 6.3;
   5. Вывод по проделанной работе.
8. Контрольные вопросы
   1. Какие методы используются для защиты от sql-инъекций?
   2. Какие методы используются для защиты от xss-инъекций?
9. Приложение
   1. Настройка базы данных

Для начала создадим базу данных и таблицы, необходимые для работы приложения.

CREATE DATABASE lab\_security;

USE lab\_security;

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL

);

CREATE TABLE comments (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(255) NOT NULL,

comment TEXT NOT NULL

);

-- Добавляем пользователя для тестирования

INSERT INTO users (username, password, email) VALUES ('admin', 'password', 'example@mail.ru');

INSERT INTO users (username, password, email) VALUES ('user', 'password', 'other@mail.ru');

CREATE USER 'lab\_security'@'localhost' identified with mysql\_native\_password BY 'lab\_security';

GRANT ALL PRIVILEGES ON lab\_security.\* TO 'lab\_security'@'localhost';

* 1. Реализация уязвимого приложения

Этот код демонстрирует уязвимости к SQL-инъекциям при отправке запросов к БД, позволяя злоумышленнику выполнять произвольные SQL-запросы через поля ввода c помощью '; SQL-команда #. Для борьбы с SQL-инъекциями используется экранирование или подготовленные параметризованные запросы.

Этот код демонстрирует уязвимость к XSS, позволяя пользователям вставлять произвольный HTML или JavaScript код в поле комментария и имени пользователя, например <script>alert()</script>. Для борьбы с XSS-инъекциями используется экранирование HTML-символов.

Этот код демонстрирует уязвимость авторизации, позволяя пользователям использовать чужие учетные данные при отправке комментариев и смене учетных данных. Для борьбы с данной уязвимостью можно использовать получение данных для данных запросов из сессии вместо прямого указания данных в поле ввода.

<?php

session\_start();

$host = 'localhost';

$user = 'lab\_security';

$pass = 'lab\_security';

$dbname = 'lab\_security';

$conn = new mysqli($host, $user, $pass, $dbname);

if ($conn->connect\_errno) {

die("Не удалось подключиться к БД: " . $conn->connect\_error);

}

// Маршрутизация: ?page=login | comments | update-email | logout | home

$page = isset($\_GET['page']) ? $\_GET['page'] : 'home';

// Обработка действий

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

if ($page === 'login') {

$username = $\_POST['username'];

$password = $\_POST['password'];

$query = "SELECT \* FROM users WHERE username = '$username' AND password = '$password'";

$result = $conn->query($query);

if ($result && $result->num\_rows > 0) {

$\_SESSION['username'] = $username;

$login\_message = "Добро пожаловать, $username!";

} else {

$login\_message = "Неверное имя пользователя или пароль.";

}

}

if ($page === 'comments') {

$username = $\_POST['username'];

$comment = $\_POST['comment'];

$query = "INSERT INTO comments (username, comment) VALUES ('$username', '$comment')";

$conn->query($query);

header('Location: ?page=comments');

exit;

}

if ($page === 'update-email') {

$username = $\_POST['username'];

$new\_email = $\_POST['new\_email'];

$query = "UPDATE users SET email = '$new\_email' WHERE username = '$username'";

$result = $conn->query($query);

if ($result) {

$update\_message = "Email for user $username updated to $new\_email"; // XSS possible

}

}

}

$comments\_result = $conn->query("SELECT \* FROM comments");

?>

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Lab Security — Unified App</title>

<style>

body { font-family: Arial, sans-serif; padding: 20px; }

nav a { margin-right: 10px; }

form { margin-bottom: 20px; }

.message { padding: 8px; background: #efefef; border-radius: 4px; margin-bottom: 12px; }

</style>

</head>

<body>

<h1>Lab Security</h1>

<nav>

<a href="?page=home">Главная</a>

<a href="?page=login">Вход</a>

<?php if (isset($\_SESSION['username'])): ?>

<a href="?page=comments">Комментарии</a>

<a href="?page=update-email">Обновить email</a>

<a href="?page=logout">Выйти (<?php echo $\_SESSION['username']; ?>)</a>

<?php endif; ?>

</nav>

<hr>

<?php if ($page === 'home'): ?>

<h2>Главная</h2>

<p>Данное приложение демонстрирует применение SQL-инъекций, XSS-уязвимостей и уязвимостей авторизации</p>

<?php if (isset($login\_message)): ?><div class="message"><?php echo $login\_message; ?></div><?php endif; ?>

<?php if (isset($update\_message)): ?><div class="message"><?php echo $update\_message; ?></div><?php endif; ?>

<h3>Текущий пользователь</h3>

<p><?php echo isset($\_SESSION['username']) ? $\_SESSION['username'] : 'Гость'; ?></p>

<?php elseif ($page === 'login'): ?>

<h2>Вход</h2>

<?php if (isset($login\_message)): ?><div class="message"><?php echo $login\_message; ?></div><?php endif; ?>

<form method="POST" action="?page=login">

Логин: <input type="text" name="username"><br>

Пароль: <input type="password" name="password"><br>

<input type="submit" value="Login">

</form>

<?php elseif ($page === 'comments'): ?>

<h2>Комментарии</h2>

<form method="POST" action="?page=comments">

Логин: <input type="text" name="username" value="<?php echo isset($\_SESSION['username']) ? $\_SESSION['username'] : ''; ?>"><br>

Комментарий: <textarea name="comment"></textarea><br>

<input type="submit" value="Отправить">

</form>

<h3>Все комментарии:</h3>

<ul>

<?php while ($row = $comments\_result->fetch\_assoc()): ?>

<li><strong><?php echo $row['username']; ?>:</strong> <?php echo $row['comment']; ?></li>

<?php endwhile; ?>

</ul>

<?php elseif ($page === 'update-email'): ?>

<h2>Сменить email</h2>

<form method="POST" action="?page=update-email">

<label for="username">Логин</label>

<input type="text" id="username" name="username" required value="<?php echo isset($\_SESSION['username']) ? $\_SESSION['username'] : ''; ?>"><br>

<label for="new\_email">Новый Email:</label>

<input type="email" id="new\_email" name="new\_email" required><br>

<input type="submit" value="Update Email">

</form>

<?php if (isset($update\_message)): ?><div class="message"><?php echo $update\_message; ?></div><?php endif; ?>

<?php elseif ($page === 'logout'): ?>

<?php session\_destroy(); header('Location: ?page=home'); exit; ?>

<?php else: ?>

<h2>Not Found</h2>

<p>Unknown page.</p>

<?php endif; ?>

</body>

</html>