Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ***ИУК «Информатика и управление***

КАФЕДРА ***ИУК4 «Программное обеспечение ЭВМ, информационные технологии»***

ЛАБОРАТОРНАЯ РАБОТА №6

«Оценка безопасности web-страниц и приложений с использованием ручного и автоматизированного анализа наличия уязвимостей типа «SQL Injection»»

ДИСЦИПЛИНА: «Защита информации»

Выполнил: студент гр. ИУК4 -72Б _____(Калашников А.С.____)
(Подпись) (Ф.И.О.)

Проверил: _____(____Ерохин И.И.___)
(Подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

Калуга, 2023

Цель работы: Освоение и систематизация знаний об уязвимостях и инструментах их выявления.

Задачи: ознакомиться с понятием уязвимостей типа «SQL Injection», принципами проверки на наличие уязвимостей и действиями в случае их обнаружения. Освоить принципы инструментального аудита безопасности информационной системы. Понять важность встраивания механизмов защиты от некорректных входных данных на этапе разработки программного обеспечения. Произвести поиск или создать собственный сайт уязвимый для SQL-инъекций. Осуществить проверку на уязвимость, предпринять действия по её устранению.

Задание:

- Написать собственный web-сайт или приложение, которое будет уязвимо для SQL-инъекций.
- Проверить его уязвимость.
- Исправить уязвимость и убедиться, что уязвимости больше нет.

Результат проверки на sql-инъекции:

- hi' or 1=1-- работает
- ' or 1=1- работает
- " or 1=1– не работает
- or 1=1- не работает
- ' or 'a'='a работает
- " or "a"="a не работает
- ') or ('a'='a sqlite3.OperationalError

Исправление уязвимостей

Ниже представлен один из возможных вариантов исправления уязвимостей

```
import sqlite3
from flask import Flask, redirect, render_template, session, url_for, request

def delete_dangerous(s):
    l = ["\\", '"', "'", "\0", ]
    for i in l:
        if i in s:
            s = s.replace(i, '')
    return s
```

```
app = Flask(__name__)
# Секретный ключ, необходимый для сессии
app.secret_key = 'A0Zr98j/3yX R~XHH!jmN]LWX/,?RT'

# Этот код выполняется если URL-путь пустой
@app.route('/')
def index():
    # Если пользователь есть в сессии загружаем шаблон index.html
    if 'username' in session:
        return render_template('index.html')
    # Перенаправление на страницу login
    return redirect(url_for('login'))

@app.route('/login')
def login():
    if 'username' in session:
        return redirect(url_for('index'))
    return render_template('login.html')

@app.route('/login/authentication', methods=['POST', 'GET'])
def authentication():
    login = request.form['login']
    password = request.form['pass']
    login = delete_dangerous(str(login))
    password = delete_dangerous(str(password))
    conn = sqlite3.connect('test_db.sqlite')
    cursor = conn.cursor()
    # Пытаемся получить число совпадений с пользователем и паролем
    cursor.execute("SELECT COUNT(*) FROM users WHERE login = '%s' AND pass =
'%s'" % (login, password,))
    res = cursor.fetchone()
    conn.close()
    # Если есть совпадения то добавляем имя в сессию
    if res[0] != 0:
        session['username'] = request.form['login']
        return redirect(url_for('index'))
    return redirect(url_for('login'))

@app.route('/logout')
def logout():
# Удаляем сессию
    session.pop('username', None)
    return redirect(url_for('login'))

if __name__ == '__main__':
    # host, port, debug_mode
    app.run('127.0.0.1', 80, True)
```

Вывод: В процессе выполнения лабораторной работы были Освоены и систематизированы знания об уязвимостях и инструментах их выявления.