Лабораторная работа: Эксплуатация SQL-инъекций

Описание стенда

Для демонстрации уязвимости SQL-инъекций использовались:
- **FastAPI** — фреймворк для создания веб-приложений
- **SQLite3** — встроенная реляционная база данных
- **Uvicorn** — ASGI-сервер для запуска приложения

Реализованы два маршрута:

1. `/` — уязвимая форма аутентификации
2. `/safety` — безопасная форма с защитой от SQLi

Реализация уязвимости

Небезопасный маршрут (/)

Использует прямую подстановку данных в SQL-запрос:

```
query = f"SELECT * FROM users WHERE username='{username}' AND
    password='{password}'"
cursor.execute(query)
```

Примеры эксплуатации:
- `Логин: ' OR 1=1-` — вывод всех пользователей
- `Логин: admin'-` — обход проверки пароля
- `Логин: ' UNION SELECT name, type, sql FROM sqlite_master -` — получение структуры БД

Безопасный маршрут (/safety)

Использует параметризованные запросы:

```
query = "SELECT * FROM users WHERE username=? AND password=?"
cursor.execute(query, (username, password))
```

Параметризация предотвращает внедрение SQL-кода за счет:
- Автоматического экранирования специальных символов
- Разделения SQL-кода и пользовательских данных

Демонстрация защиты

Попытка SQL-инъекции на безопасном маршруте:
- `Логин: ' OR 1=1-`
- Результат: Ошибка авторизации

Выводы

1. SQL-инъекции возможны при прямом использовании пользовательского ввода в запросах
2. Параметризованные запросы надежно защищают от подобных атак
3. Безопасная разработка требует тщательной обработки всех входных данных
4. **В промышленных проектах применяют:**
   (a) **ORM-системы** (SQLAlchemy, Django ORM) — автоматическое экранирование и абстракции над SQL
   (b) **Валидацию данных** через JSON Schema или библиотеки вроде Pydantic
   (c) **Санитизацию ввода** (удаление/экранирование опасных символов)

При реализации проекта по информационным технологиям в этом семестре мы реализовывали веб сервис по взаимодействию студенто и преподавателей. Использовали FastAPI + SQLAlchemy + Pydantic, что возволило обезопасить страницы авторизации и регистрации от SQL-инъекций.