Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Информационные сети. Основы безопасности

ОТЧЁТ
к лабораторной работе №6
на тему

ЗАЩИТА ОТ АТАКИ МЕТОДОМ ВНЕДРЕНИЯ SQL-КОДА

Выполнил: студент гр. 253503
Тимошевич К.С.

Проверил: ассистент кафедры информатики Герчик А.В.

Минск 2025

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

В данной лабораторной работе рассматривается проблема безопасности, связанная с атакой методом внедрения *SQL*-кода (*SQL*-инъекция), которая является одной из наиболее распространённых уязвимостей в веб-приложениях и системах, работающих с базами данных. *SQL*-инъекция позволяет злоумышленнику выполнять произвольные *SQL*-запросы к базе данных, что может привести к несанкционированному доступу к конфиденциальной информации, изменению или удалению данных, а также к нарушению работы системы в целом [1].

Цель работы – изучить механизмы возникновения *SQL*-инъекций, их последствия, а также разработать и протестировать методы защиты от таких атак.

Основные задачи работы включают создание тестовой базы данных с таблицей пользователей и заполнение её данными для последующего анализа. Далее необходимо разработать уязвимую функцию, которая использует конкатенацию строк для формирования *SQL*-запросов, что делает её подверженной атакам *SQL*-инъекции. Это позволит наглядно продемонстрировать, как злоумышленник может использовать недостатки в обработке пользовательского ввода для выполнения произвольных *SQL*-запросов. После этого требуется разработать защищённую функцию, использующую параметризованные запросы для предотвращения *SQL*-инъекций. Это обеспечит безопасную обработку данных и исключит возможность внедрения вредоносного кода. Следующим этапом является демонстрация работы уязвимой и защищённой функций на примере попытки входа в систему с использованием корректных данных и зловредного пароля, который может быть использован для *SQL*-инъекции. Это позволит показать, как уязвимая функция может быть взломана, а защищённая – противостоит атаке.

2 ХОД ВЫПОЛНЕНИЯ РАБОТЫ

Для выполнения лабораторной работы было разработано программное средство на языке *Python*, которое демонстрирует уязвимость к атакам методом внедрения *SQL*-кода (*SQL*-инъекция) и методы защиты от них. Программа состоит из двух частей: уязвимой и защищённой реализации аутентификации пользователя.

Для тестирования была создана база данных с использованием модуля *sqlite3* [2]. В базе данных была создана таблица *users* с полями *id*, *username* и *password*. Таблица была заполнена тестовыми данными, включая пользователя с логином *admin* и паролем *secure_password*, а также другими пользователями для более реалистичного тестирования.

Для демонстрации уязвимости была реализована функция *unsafe_login*, которая формирует *SQL*-запрос с использованием конкатенации строк. Такой подход делает приложение уязвимым к *SQL*-инъекциям, так как пользовательский ввод напрямую подставляется в запрос без какой-либо обработки. Например, при вводе зловредного пароля OR '1'='1' злоумышленник может обойти проверку аутентификации, так как условие '1'='1' всегда истинно.

Для защиты от *SQL*-инъекций была реализована функция *safe_login*, которая использует параметризованные запросы. В этом случае пользовательский ввод передается в запрос отдельно от *SQL*-кода, что автоматически экранирует специальные символы и предотвращает возможность внедрения вредоносного *SQL*-кода.

Результаты тестирования программы показаны на рисунке 2.1. В уязвимой версии при вводе зловредного пароля система пропускает злоумышленника, что подтверждает наличие уязвимости. В защищенной версии атака предотвращается, и доступ отклоняется, что демонстрирует эффективность предложенного метода защиты.

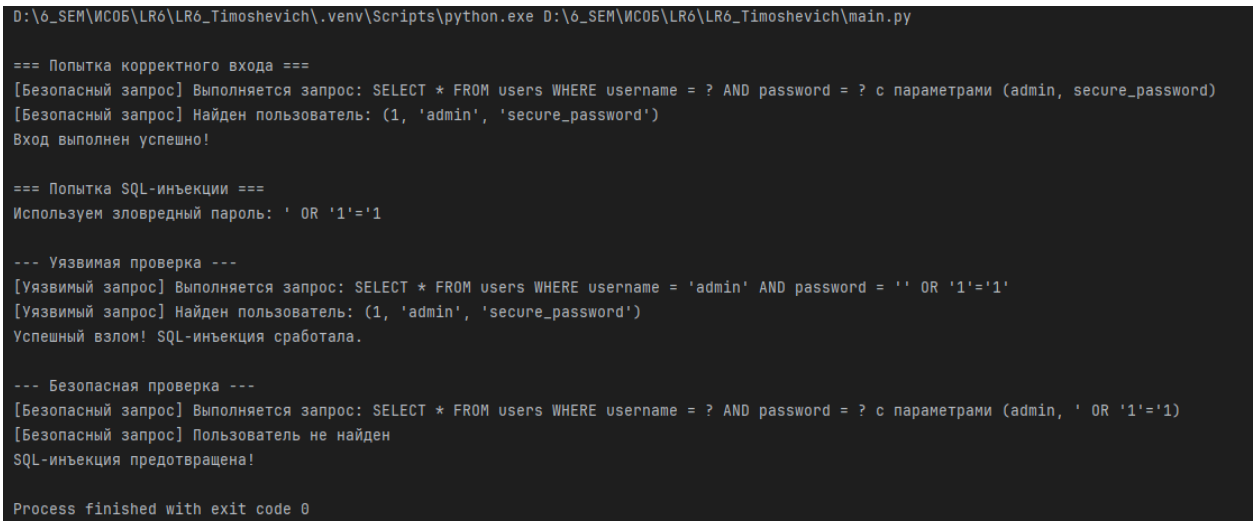

Рисунок 2.1 – Результат выполнения программы

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы было разработано программное средство, демонстрирующее уязвимость к атакам методом внедрения *SQL*-кода (*SQL*-инъекция) и методы защиты от них. Была создана тестовая база данных с таблицей пользователей, которая была заполнена тестовыми данными для проверки функциональности программы. Разработана уязвимая версия аутентификации, использующая конкатенацию строк для формирования *SQL*-запросов, что позволило наглядно продемонстрировать успешную *SQL*-инъекцию при вводе злоумышленником специально сформированных данных, таких как OR '1'='1' .

Для защиты от подобных атак была реализована безопасная версия, основанная на параметризованных запросах. Этот подход автоматически экранирует пользовательский ввод, предотвращая возможность внедрения вредоносного *SQL*-кода. Тестирование программы подтвердило эффективность параметризованных запросов: в уязвимой версии при вводе зловредного пароля аутентификация проходила успешно, что наглядно показало наличие уязвимости. В защищенной версии та же атака была предотвращена, и доступ к системе отклонялся, что подтвердило надежность данного метода защиты.

Результаты работы подчеркнули важность соблюдения принципов безопасности при разработке приложений, работающих с базами данных. Неправильная обработка пользовательского ввода может привести к серьезным последствиям, включая утечку конфиденциальной информации, несанкционированный доступ и повреждение данных. Использование параметризованных запросов, *ORM*-систем и других современных методов защиты позволяет минимизировать риски и обеспечить безопасность приложения.

Выполнение данной работы позволило не только изучить теоретические основы *SQL*-инъекций, но и получить практический опыт реализации защищенных приложений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Что такое SQL-инъекция? [Электронный ресурс]. – Режим доступа: https://www.kaspersky.ru/resource-center/definitions/sql-injection. – Дата доступа: 11.03.2025

[2]Работа с SQLite в Python [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/754400/. – Дата доступа: 11.03.2025