**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №5**

**по дисциплине «Базы данных»**

Тема: Тестирование БД на безопасность

|  |  |  |
| --- | --- | --- |
| Студент гр. 1303 |  | Депрейс А.С. |
| Преподаватель |  | Заславский М.М. |

Санкт-Петербург

2023

**Цель работы.**

Сделать простой web-сервер и проверить реализованное API на безопасность.

**Текст задания**

Вариант 7

1. Сделать простой web-сервер для выполнения запросов из ЛР3, например с (express.js). Не обязательно делать авторизацию и т.п., хватит одного эндпоинта на каждый запрос, с параметрами запроса как query parameters.
2. Намеренно сделайте несколько (2-3) запроса, подверженных SQL-инъекциям
3. Проверьте Ваше API с помощью sqlmap (или чего-то аналогичного), передав эндпоинты в качестве целей атаки. Посмотрите, какие уязвимости он нашёл (и не нашёл), опишите пути к исправлению.

**Выполнение работы**

Для выполнения работы были использованы модели из лабораторной работы №3. Реализовано API для доступа к БД.



Рисунок 1. – Реализованный web-сервер для выполнения запросов.

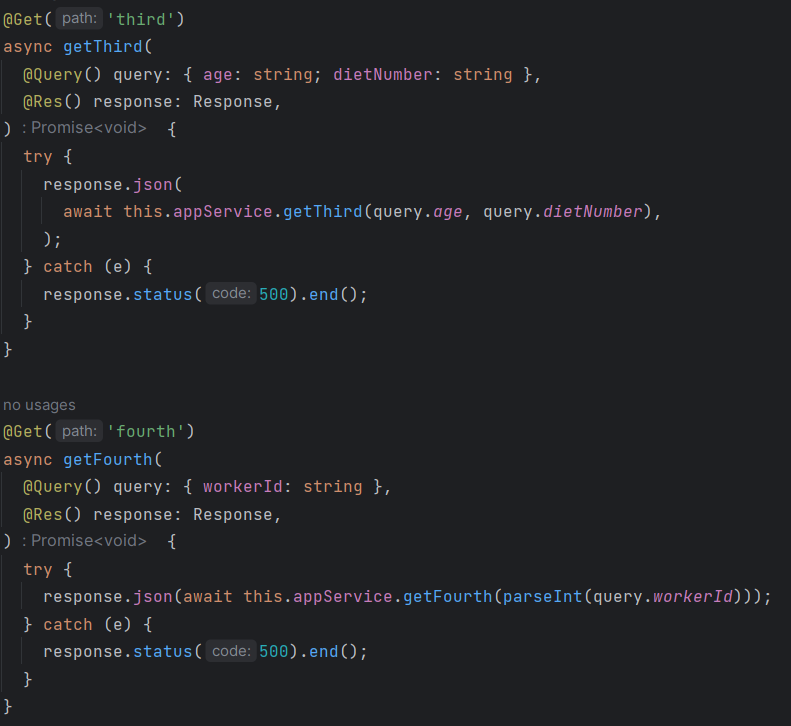


Рисунок 2 – Реализованный web-сервер для выполнения запросов.

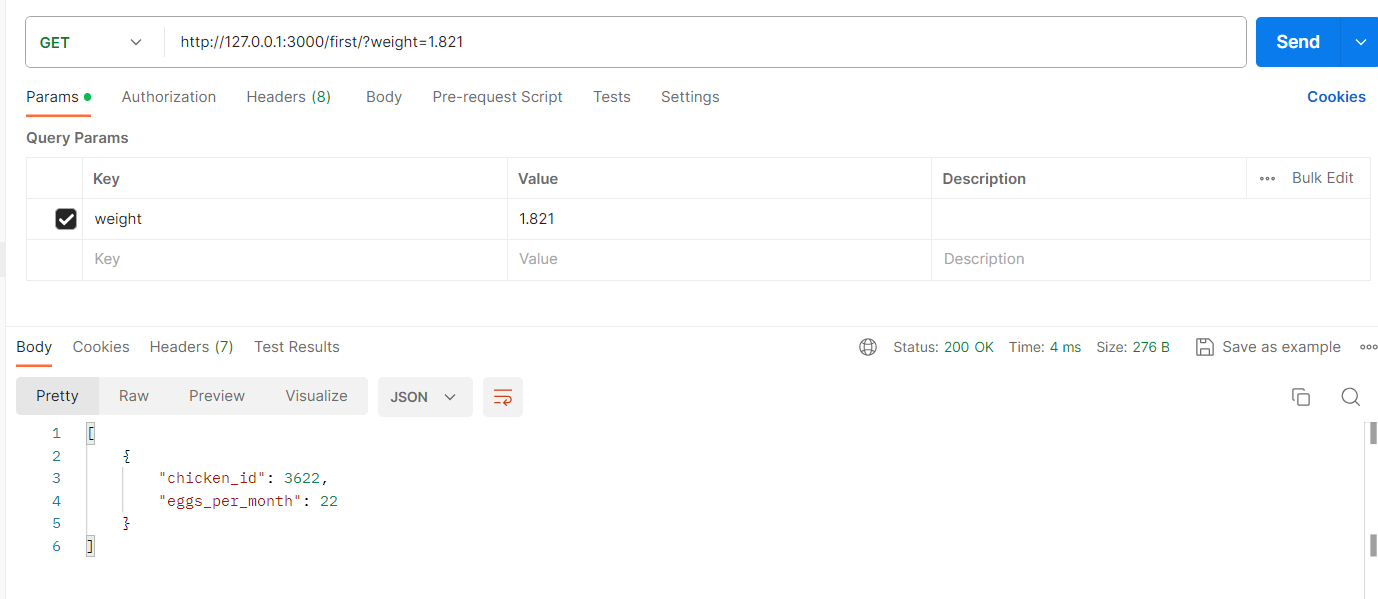


Рисунок 3. – Пример ответа web-сервера.

Второй и третий эндпоинты подвержены SQL-инъекциям:

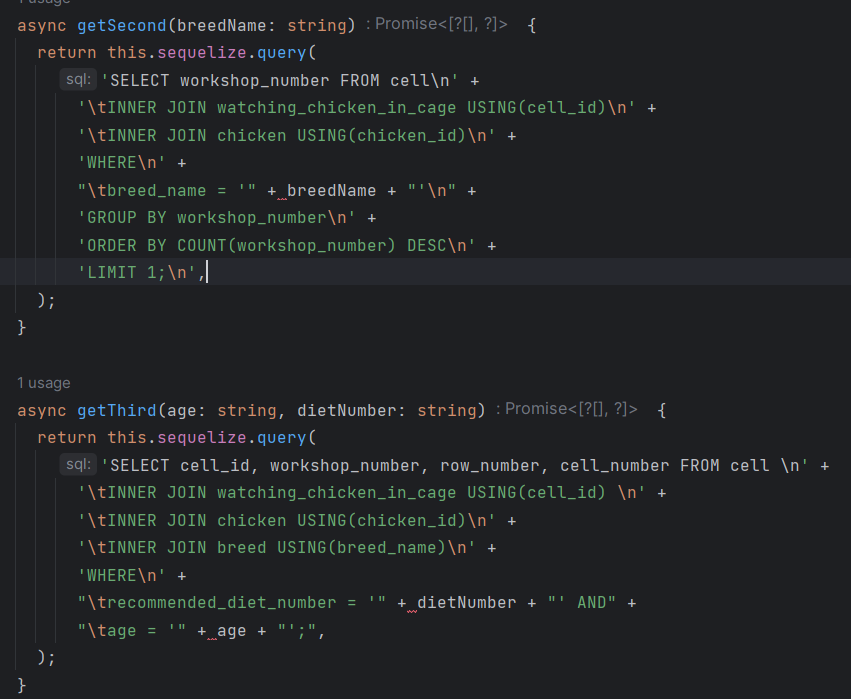


Рисунок 4. – Эндпоинты 2 и 3.

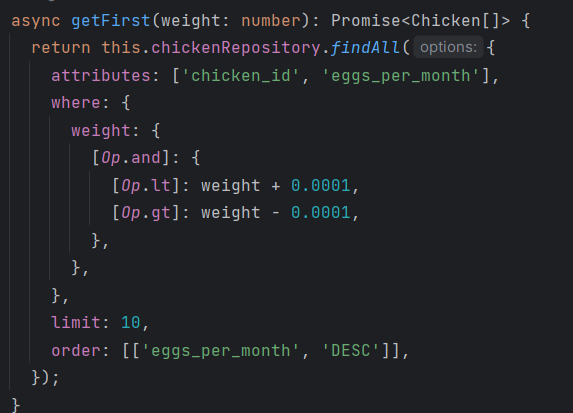
Первый и четвертый эндпоинты не подвержены SQL-инъекциям: 

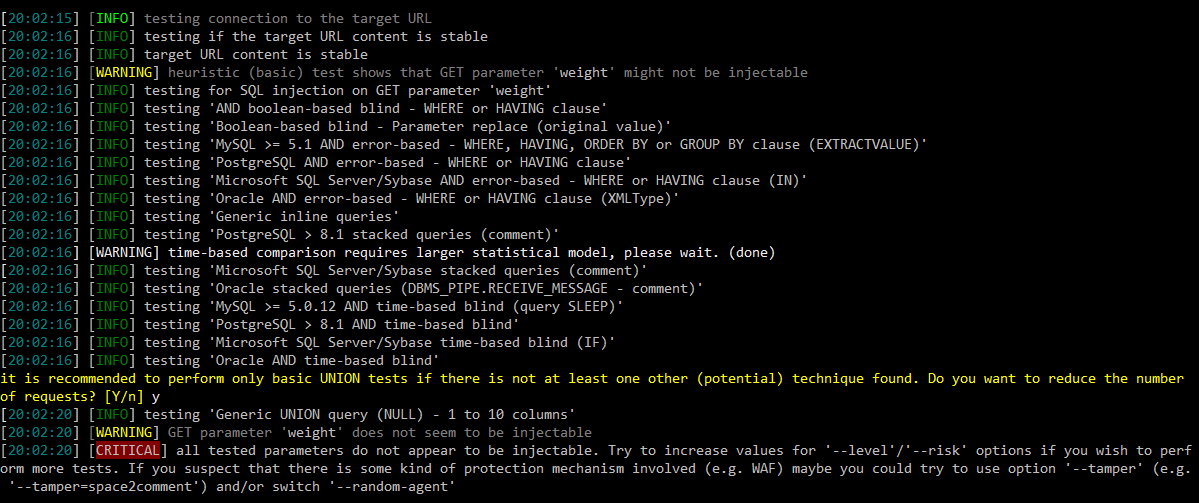
Рисунок 5. – Эндпоинт 1.



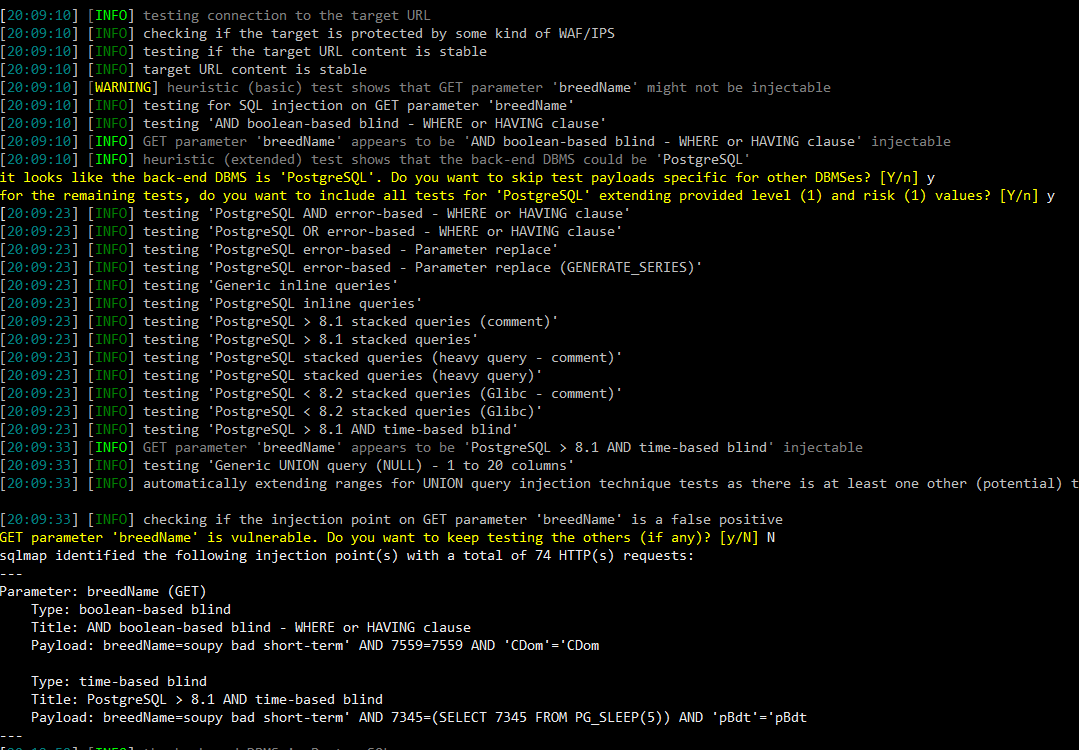
Рисунок 6. – Эндпоинт 4.

Проверим API с помощью sqlmap:

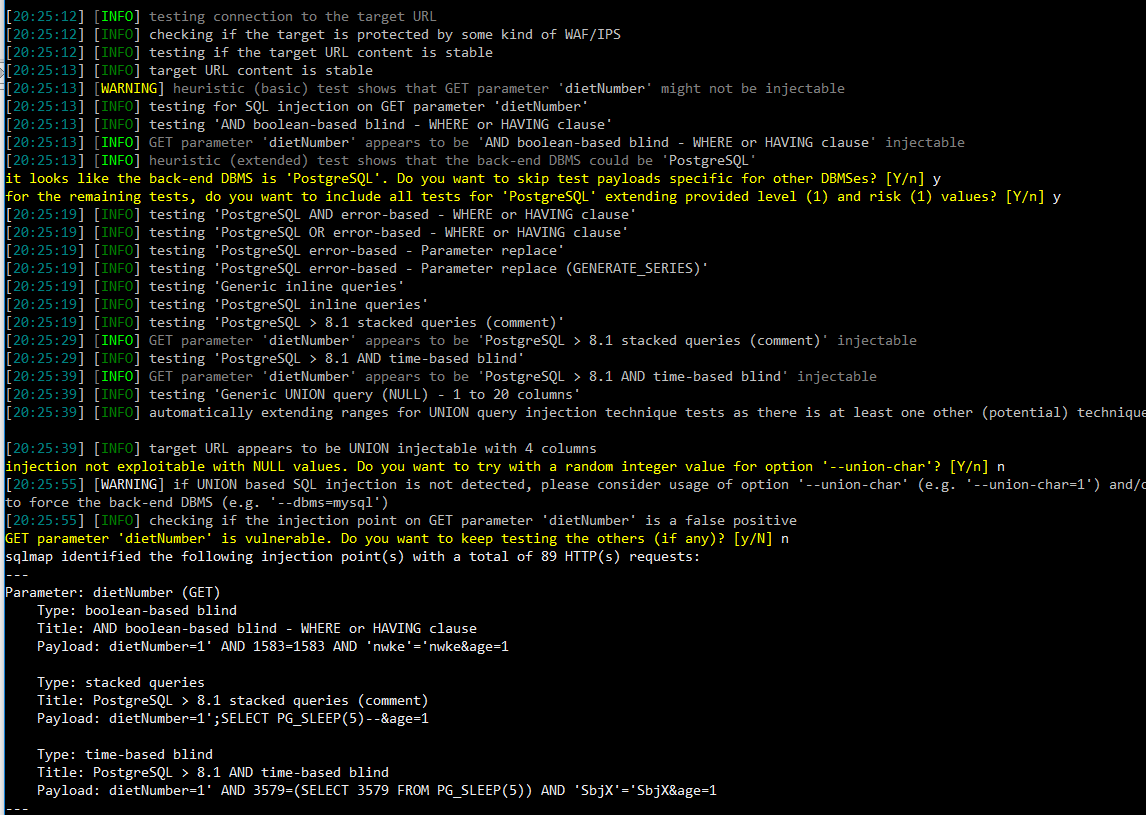
1. python sqlmap.py -u "http://localhost:3000/first?weight=1.8" -p "weight"



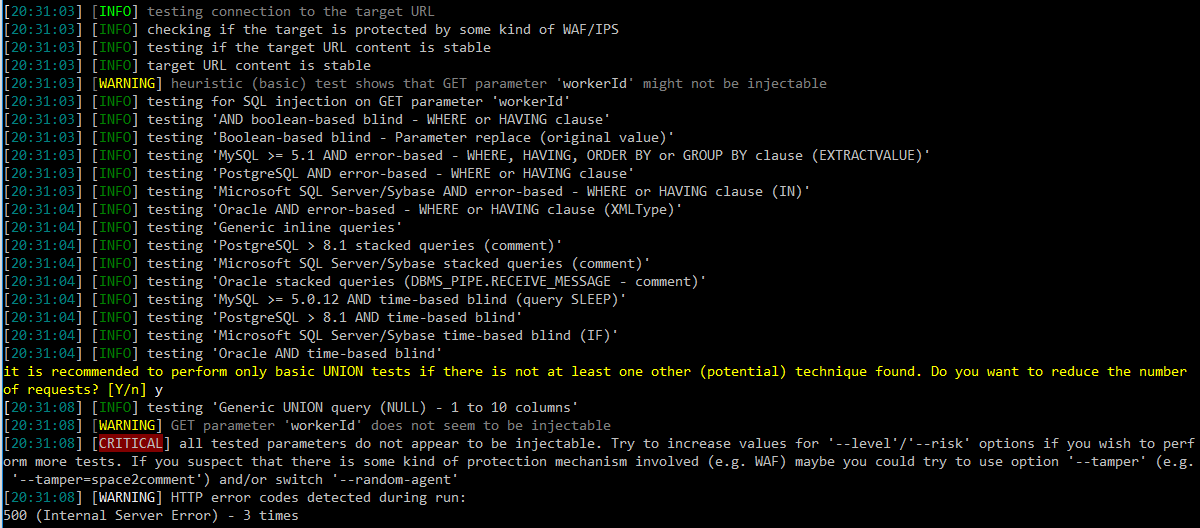
Параметр weight не подвержено инъекции, так как если передать не число, то parseInt отдаст NaN и тогда запрос будет “корректным”.

1. python sqlmap.py -u "http://127.0.0.1:3000/second/?breedName=soupy bad short-term" -p "breedName" 

Параметр breedName подвержено SQL инъекциям boolean-based blind и time-based blind типа. Первое позволяет получить информацию из БД с помощью реакций на логические выражения в инъекции, а вторая позволяет получить информацию из БД в случае если нету видимого вывода, при успешной инъекции БД отвечает с задержкой.

1. python sqlmap.py -u "http://127.0.0.1:3000/third/?dietNumber=1&age=1" -p "dietNumber,age" 

Параметр dietNumber подвержено SQL инъекциям. Кроме уже встречающихся инъекций появилась stacked queries инъекция, которая позволяет совершить несколько запросов из-за того, что параметр вставляется в конце и с помощью точки с запятой можно указать конец первого запроса и сделать второй.

1. python sqlmap.py -u "http://127.0.0.1:3000/fourth/?workerId=1" -p "workerId" 

Параметр workerId не подвержен инъекциям.

**Выводы.**

Протестировав доступ к БД с помощью ORM Sequelize на безопасность, выяснили, что два из четырех эндпоинта подвержены SQL инъекциям. Для противодействия таким инъекциям необходимо перед выполнением запросов форматировать ввод, если вводимая информация имеет определенную форму, то проверять на соблюдение формы, не предоставлять информацию об ошибках БД.

**ПРИЛОЖЕНИЕ А**

Pull request: <https://github.com/moevm/sql-2023-1303/pull/62>