Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение высшего образования  
«Уральский федеральный университет имени первого Президента России Б. Н. Ельцина»  
Институт радиоэлектроники и информационных технологий (ИРИТ-РТФ)

**ЛАБОРАТОРНАЯ РАБОТА №4  
по дисциплине «разработка безопасных веб-приложений»**

Студент Е.А. Беньковский

дата подпись

Группа РИ-491223

Преподаватель С. Г. Мирвода

дата подпись

Екатеринбург 2022

Оглавление

[1. Reflected XSS в поиске книг 3](#_Toc122881104)

[2. Persisted XSS при создании книги и отображении списка книг 3](#_Toc122881105)

[3. Потенциальная уязвимость через Cookie Injection 4](#_Toc122881106)

[4. Session hijacking 5](#_Toc122881107)

[5. Защита 6](#_Toc122881108)

[Reflected XSS 6](#_Toc122881109)

[Persisted XSS 7](#_Toc122881110)

[Cookie injection 7](#_Toc122881111)

[Проверка защищённости 8](#_Toc122881112)

# Reflected XSS в поиске книг

Данная уязвимость возникает при включении приложением в ответ данных, полученных им из запроса.

При помощи строки поиска или url, сформируем запрос:

**<img src=1 onerror='javascript:alert(document.cookie)'/>**

Тем самым, появилась возможность получить cookie пользователя, кликнувшего по ссылке.

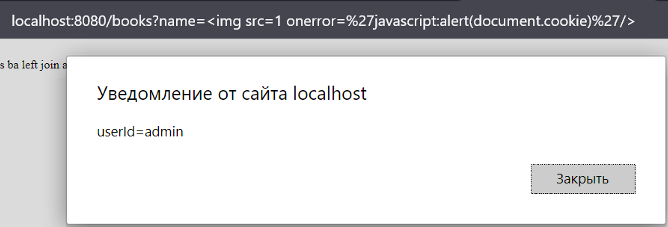


Рис. 1 - Cookie пользователя, кликнувшего по ссылке

# Persisted XSS при создании книги и отображении списка книг

При получении кодом данных и небезопасном включении их в последующие ответы.

Запишем в поле названия добавляемой книги:

**<script>alert(document.cookie);</script>**

В результате получаем куки очередного пользователя.

# Потенциальная уязвимость через Cookie Injection

Другими словами, это возможность отправить запрос с куки другого пользователя и дополнительными параметрами.

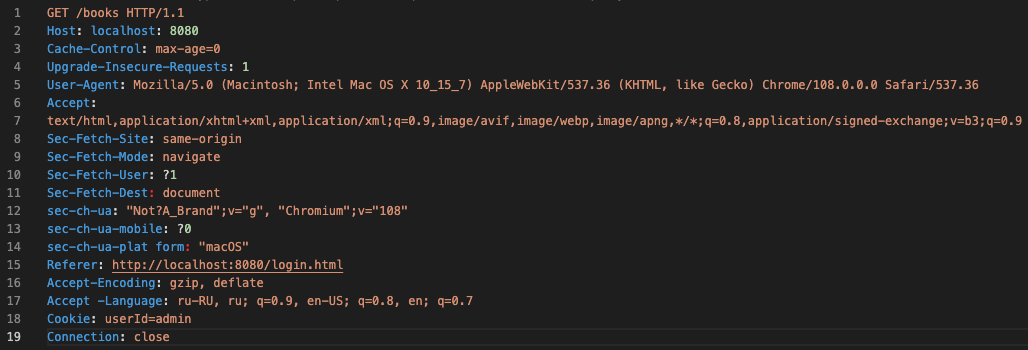


Рис. 2 - куки другого пользователя и дополнительными параметрами

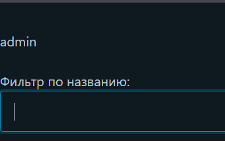


Рис. 3 – Войдём как админ

# Session hijacking

Некорректное создание сессионной cookie, которое приводит к захвату сессии и даёт возможность злоумышленнику украсть данные пользователя.

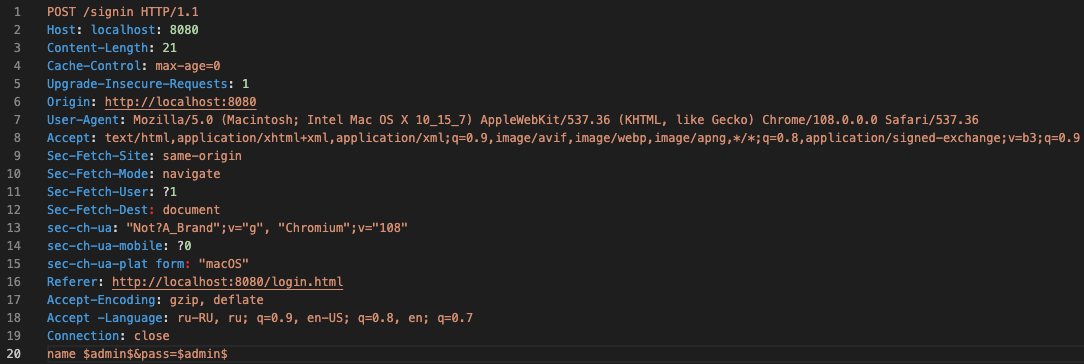


Рис. 4 - Некорректная куки

# Защита

## Reflected XSS

Для защиты от Reflected XSS необходимо изменить функцию обработки ошибок. А именно, заменить **res.send()** на **res.render()** в коде файла server.js.

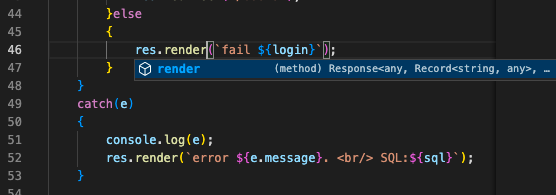


Рис. 5 - заменить res.send() на res.render() в коде файла server.js

Полный текст ошибки:

Error: Failed to lookup view "error ошибка синтаксиса (примерное положение: "javascript"). <br/> SQL:select ba.id, a.name as author, b.name as book from books\_by\_authors ba

left join author a on a.id = ba.aid

left join book b on b.id • ba.bid

where b.name like '&<img src=1 onerror='javascript:alert (document.cookie)'/>%\*" in views directory "Users/krest/Downloads/lab4/views"

at Function.render (Users/krest/Downloads/lab4/node\_modules/express/lib/application.js:597:17)

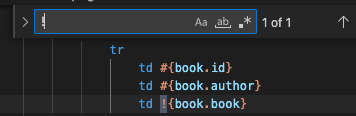
at ServerResponse.render (Users/krest/Downloads/lab4/node\_modules/express/lib/response.js: 1039:7)

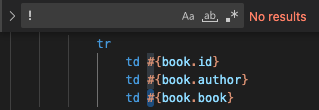
at Users/krest/Downloads/lab4/server.js: 56:13

at process.processTicksAndRejections (node: internal/process/task\_queues:95:5)

## Persisted XSS

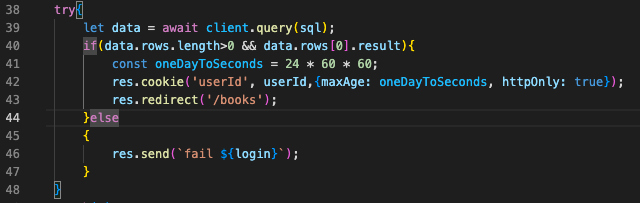
Для защиты от Persisted XSS необходимо и заменить знак «!» на «#» шаблоне Pug.





## Cookie injection

Необходимо запретить доступ JavaScript к файлу cookie. Для этого следует установить атрибут httpOnly в положение true в коде файла server.js.



## Проверка защищённости

Запустим document.cookie в панели разработчика:

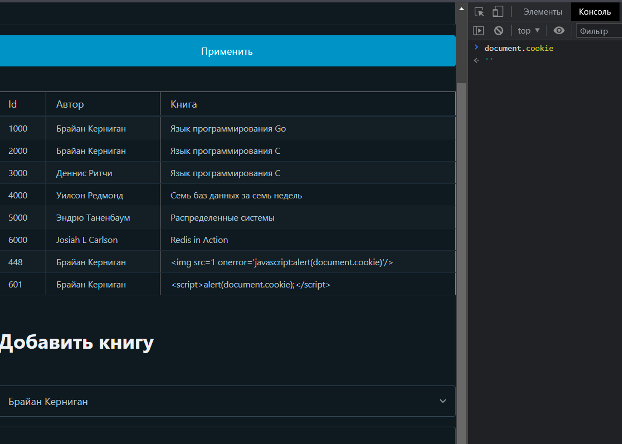


Рис. 6 - Проверка защищённости