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)  
Школа академического и профессионального образования

**Отчет по лабораторной работе № 4  
по дисциплине «Разработка безопасных web приложений»**

Группа РИ-581220

Студенты Е. А. Бородашкина

дата подпись

К. В. Крупин

дата подпись

Преподаватель С. Г. Мирвода

дата подпись

Екатеринбург

2022

# Reflected XSS

В строке поиска книг напишем следующую конструкцию: <script>alert(‘1’)</script>. При нажатии на кнопку «Применить» увидим, что скрипт выполнится:

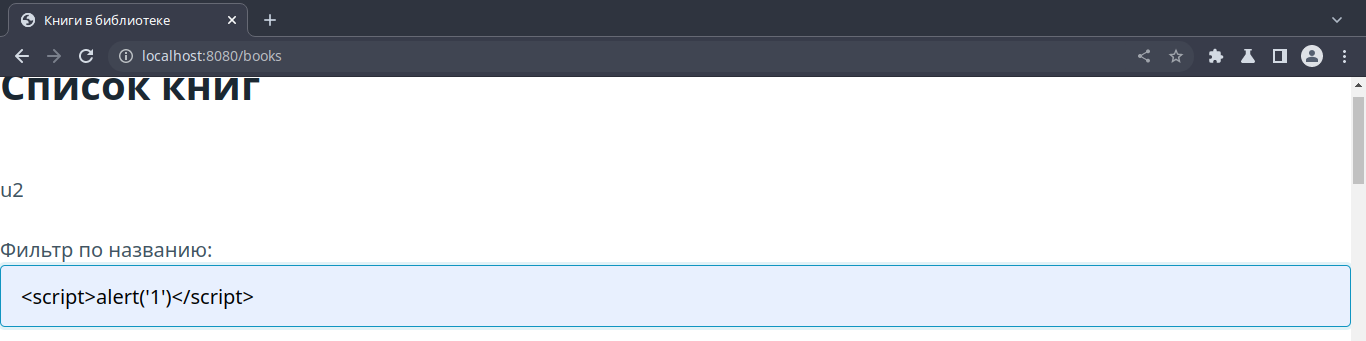


Рисунок 1 – Эксплуатация Reflected XSS

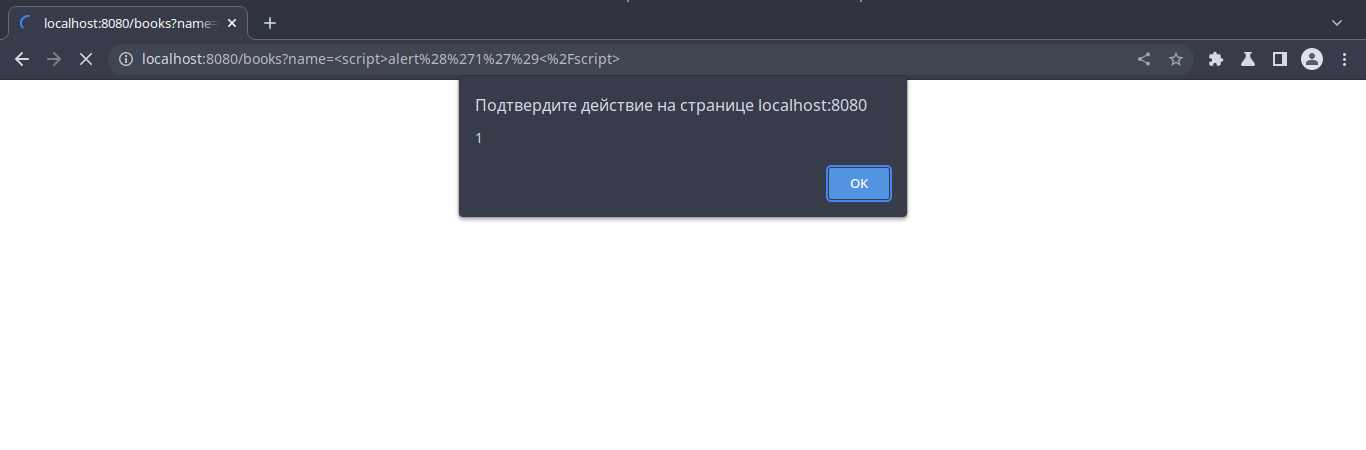


Рисунок 2 – Результат Reflected XSS

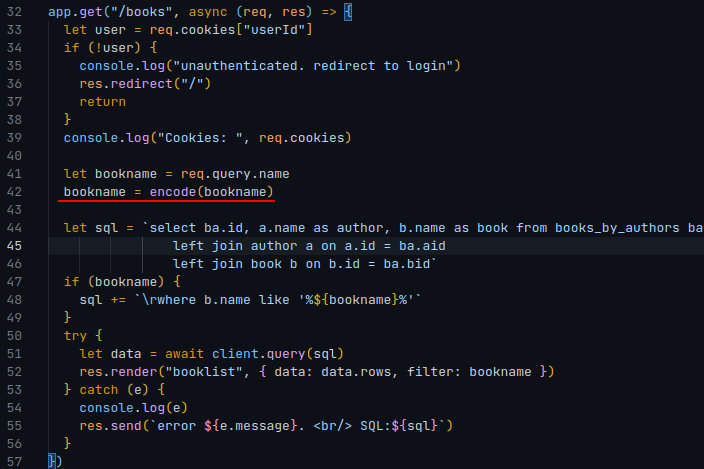


Рисунок 3 - Защита от Reflected XSS путем замены служебных символов HTML (<, >, и т.д.) на &lt; &gt и т.д. Функция encode из пакета html-entities.

Теперь при попытке вставить скрипт в поле поиска, будет следующий результат:

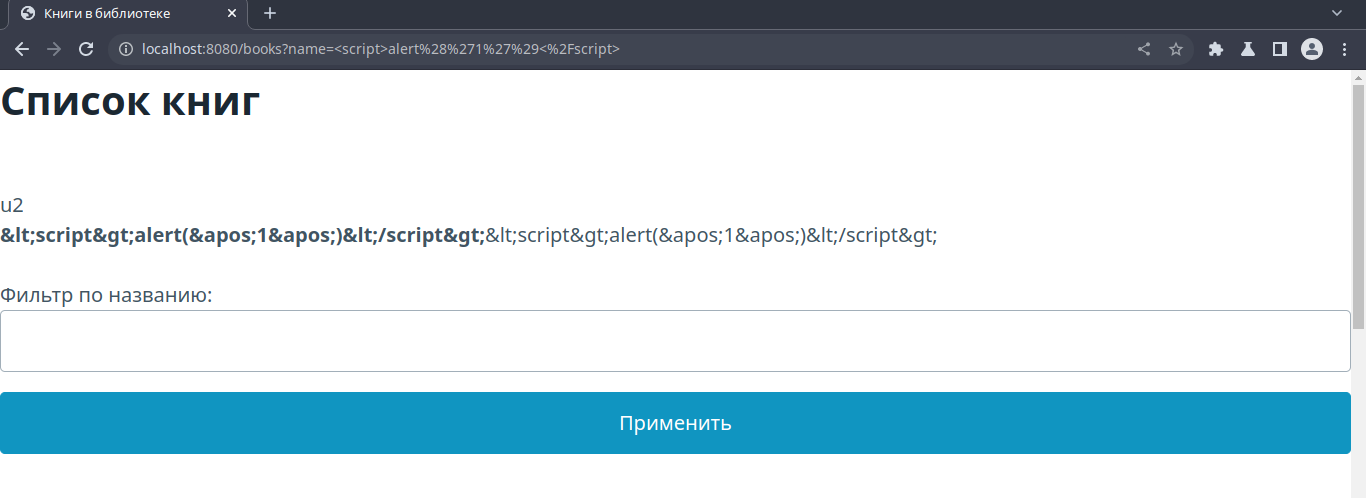


Рисунок 4 – Попытка эксплуатации Reflected XSS

# Stored XSS

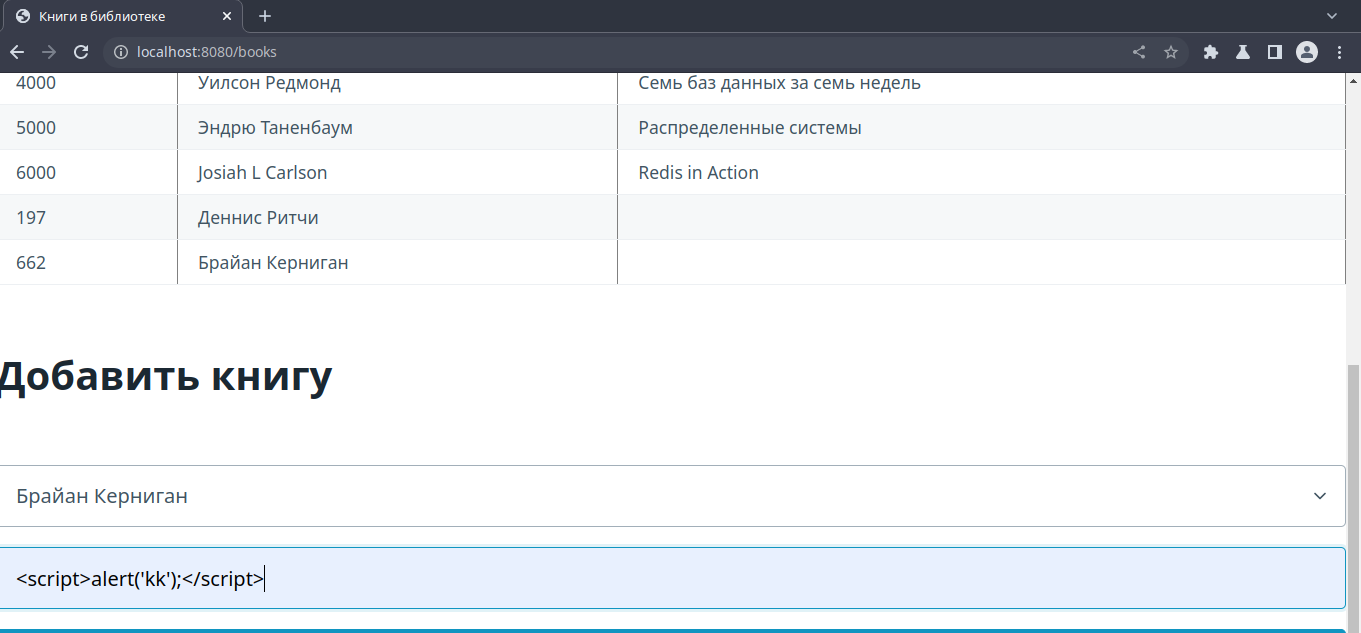


Рисунок 5 – Эксплуатация Stored XSS

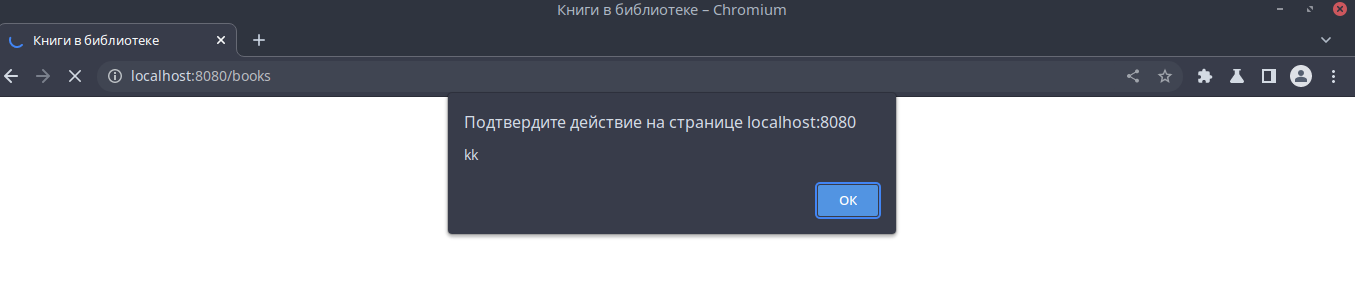


Рисунок 6 – При авторизации на сайте каждый раз выполняется скрипт

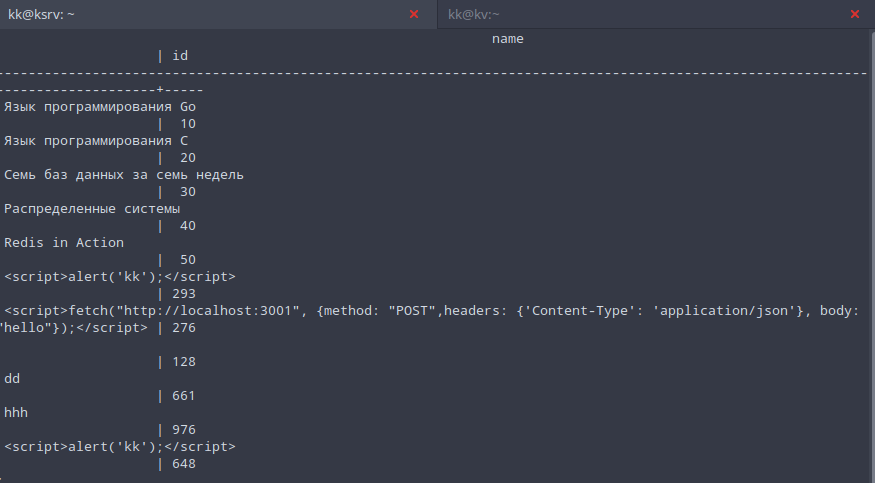


Рисунок 7 - Добавленные книги. Таким образом, при обращении пользователя к базе данных книг, он будет получать скрипты вместо книг.

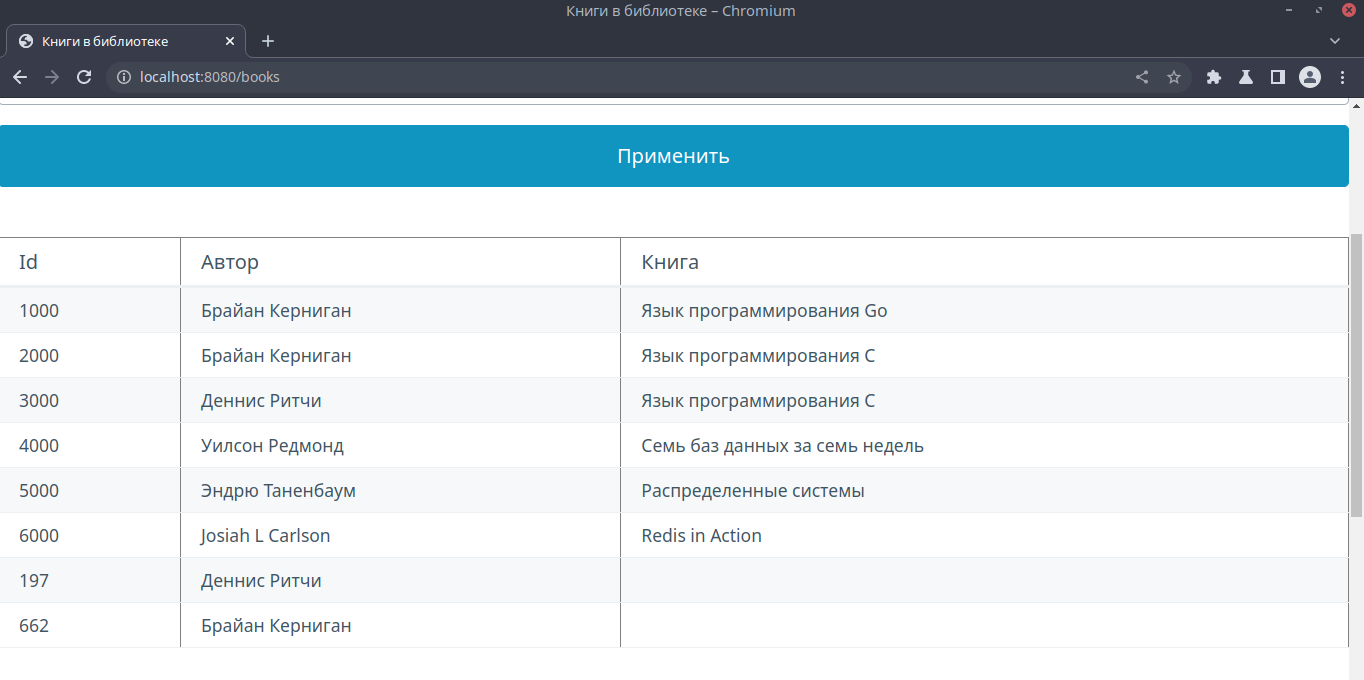


Рисунок 8 – Добавилась «книга»

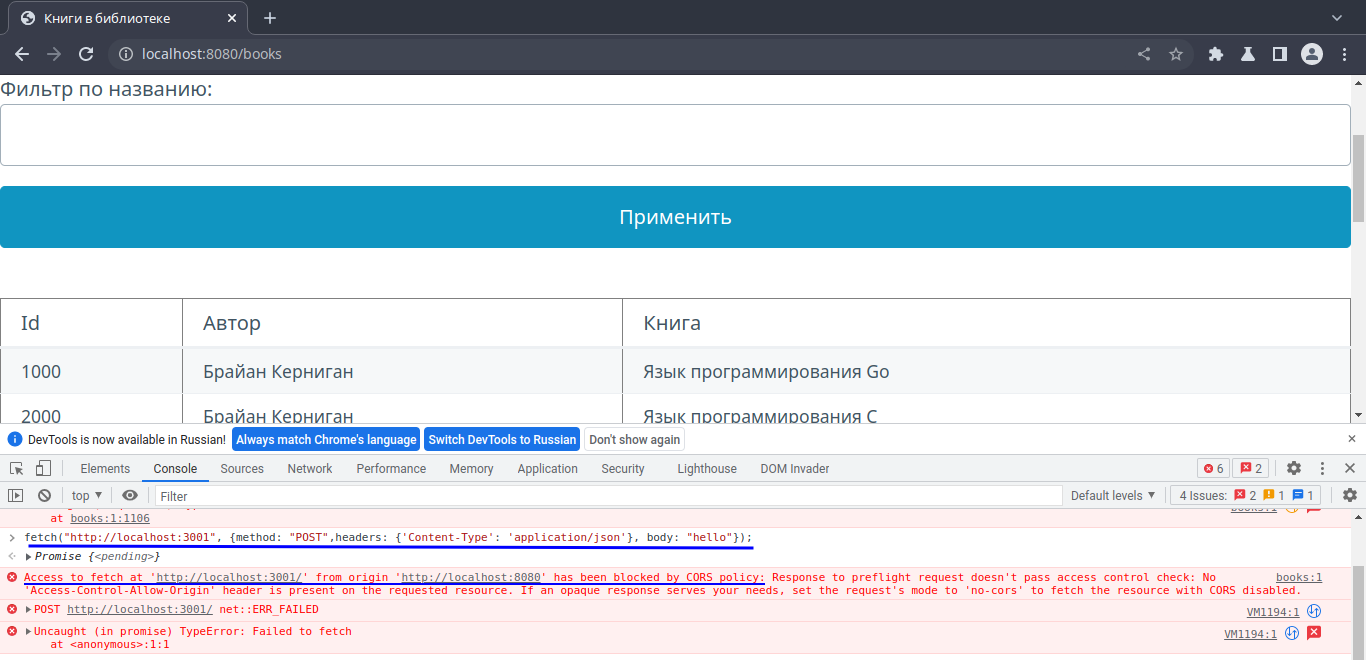


Рисунок 9 - Попытка выполнить запрос из браузера пользователя. Таким образом, даже если и получится получить значение cookie из кода js (что также запрещено флагом httpOnly) его не получится отправить на другой сервер.

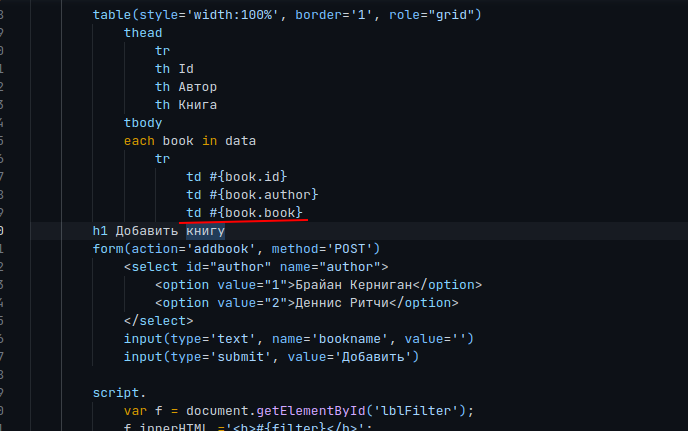


Рисунок 10 - Устранение уязвимости Stored XSS. В данном случае приложение экранирует все небезопасные символы.

# Session hijacking

Будучи неавторизованным, но имея чужой cookie выполним запрос к http://localhost:8080/books

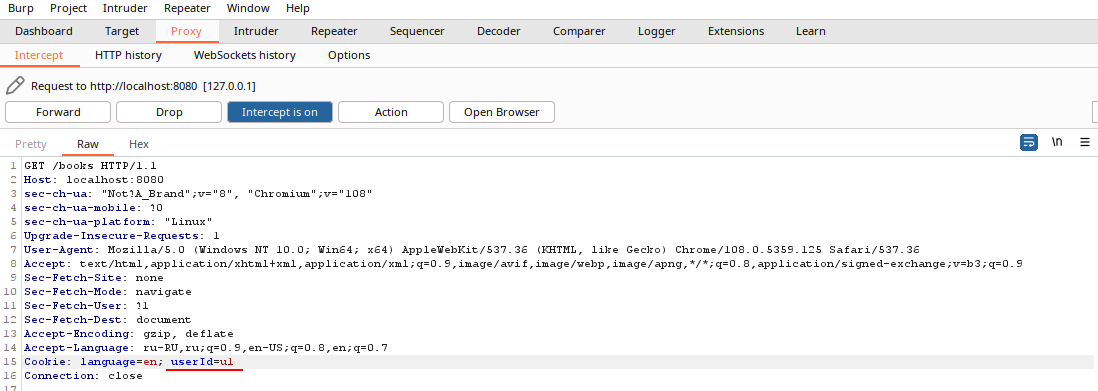


Рисунок 11 - Скажем серверу, что у нас есть cookie. Таким образом, запрос выполняется от имени пользователя u1.

В качестве защиты можно предложить запоминать на сервере какие-либо параметры пользователя, например IP, User-agent, и т.д. и сравнивать их с полученными при попытке зайти значениями cookie. Если они не совпадают, то обнулить cookie, и попросить авторизоваться.

# Сookie Injection

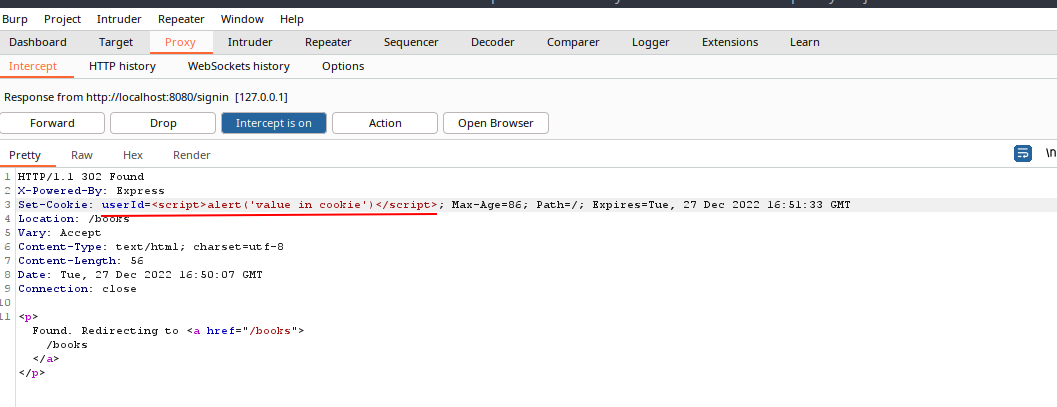


Рисунок 13 - После авторизации, сервер выдает cookie пользователю. Перехватим ответ, и запишем свое значение.

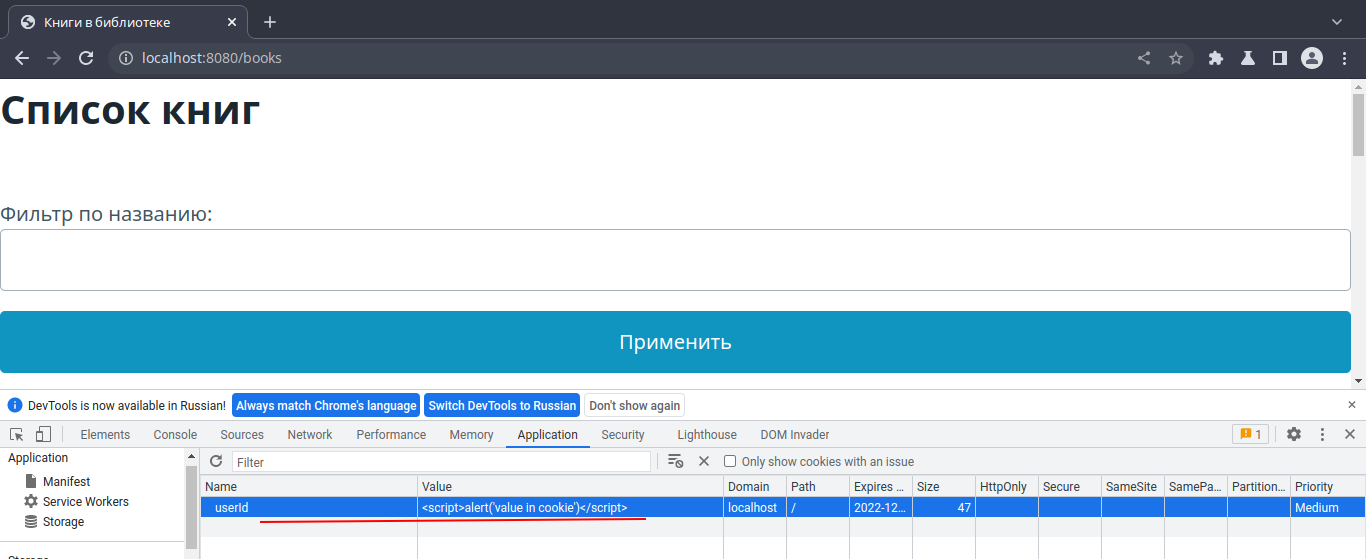


Рисунок 14 - Установлен cookie злоумышленника. В данном случае код не исполнится, так как клиентский код не вызывает функцию eval