Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина» (УрФУ)

Институт радиоэлектроники и информационных технологий – РтФ

Школа профессионального и академического образования

Учебно-научный центр «Информационная безопасность»

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Лабораторная работа №4  
по дисциплине «Разработка безопасных веб-приложений»

Преподаватель: С. Г. Мирвода \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО) (Подпись)

Студент: М. А. Коробова \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО) (Подпись)

Студент: Д. Д. Тимганова \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО) (Подпись)

Группа: РИ-491223

Екатеринбург

2023

Оглавление

[Поиск и устранение XSS уязвимостей. 3](#_Toc125372776)

[Создание базы данных и запуск сайта 3](#_Toc125372777)

[Reflected XSS в поиске книг 3](#_Toc125372778)

[Persisted (Stored) XSS при создании книги и отображении списка книг 4](#_Toc125372779)

[Потенциальную уязвимость через Cookie Injection 4](#_Toc125372780)

[Некорректное создание сессионной cookie, которое приводит к захвату сессии (Session hijacking) 5](#_Toc125372781)

[Исправление уязвимости 5](#_Toc125372782)

# **Поиск и устранение XSS уязвимостей.**

## Создание базы данных и запуск сайта

Создадим и заполним БД **lib** данными из папки **/db**.

Изображение выглядит как текст

Автоматически созданное описание

Скрипт для создания таблицы **users**:

create table if not exists users

(

name text,

pass text

);

insert into users (name, pass) values ('user1', MD5('12345'));

insert into users (name, pass) values ('user2', MD5('56921'));

insert into users (name, pass) values ('user3', MD5('48026'));

insert into users (name, pass) values ('user4', MD5('58923'));

insert into users (name, pass) values ('user5', MD5('70457'));

insert into users (name, pass) values ('user6', MD5('12789'));

## Reflected XSS в поиске книг

**Reflected XSS**-уязвимость возникает, когда пользовательский ввод с URL-адреса или данных POST отражается на странице без сохранения, что позволяет злоумышленнику внедрить вредоносный контент. Это означает, что злоумышленник должен отправить созданный вредоносный URL-адрес или почтовую форму жертве, чтобы вставить полезную нагрузку, и жертва должна щелкнуть ссылку.

При помощи строки поиска или url, сформируем запрос:

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

Это позволяет получить cookie пользователя, кликнувшего по ссылке.

Изображение выглядит как текст

Автоматически созданное описание

## Persisted (Stored) XSS при создании книги и отображении списка книг

Уязвимости Stored Cross-site возникают, когда полезная нагрузка (payload) сохраняется, например, в базе данных, а затем выполняется, когда пользователь открывает страницу в веб-приложении.

С помощью раздела «Добавить книгу» запишем код:

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

Этот код будет выполняться, когда пользователь откроет страницу с книгами.

Изображение выглядит как текст

Автоматически созданное описание

## Cookie Injection уязвимость

**Cross-site Scripting**, также известный как XSS, является способом обхода концепции SOP в уязвимом веб-приложении. Всякий раз, когда HTML-код генерируется динамически, а пользовательский ввод не очищается (то есть не проверяется) и отображается на странице как есть, злоумышленник может вставить свой собственный HTML-код.

В таком случае злоумышленник может легко вставить код JavaScript, который будет работать в контексте сайта. Таким образом, злоумышленник может получить доступ к другим страницам в том же домене и может читать данные, такие как CSRF-токены или установленные файлы cookie.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание



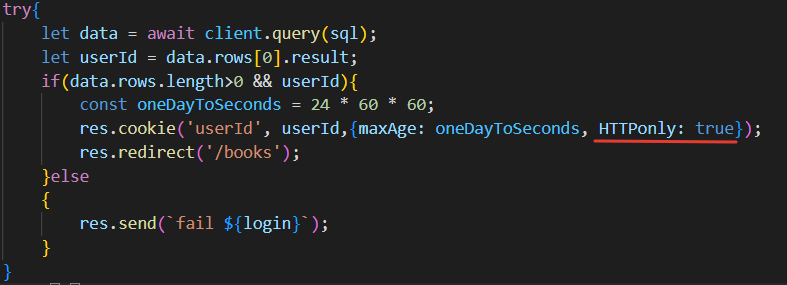
## Некорректное создание сессионной cookie, которое приводит к захвату сессии (Session hijacking)

Функция захвата сеанса — получить несанкционированный доступ к информации или службам в системе. Обычно это происходит, когда злоумышленник крадет или перехватывает файлы cookie сеанса, когда пользователь просматривает Интернет.

Если файлы cookie сеанса захвачены, они могут выдать себя за пользователя и войти в систему от его имени. Примером является Файлы cookie HTTP, которые могут быть украдены и служат для продолжения сеанса.

## Исправление уязвимости

Куки HTTPonly не доступны из JavaScript через свойства document.cookie API, что помогает избежать межсайтового скриптинга (XSS (en-US)).

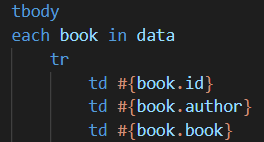


Для исправления уязвимости **Reflected XSS** исправим функцию обработки ошибок, заменив **res.send()** на **res.render()**. Разница заключается в параметре ContentType: в первом случае это application/json, во втором – text/html.

Изображение выглядит как текст

Автоматически созданное описание

Для исправления **Persisted XSS** заменим знак «!» на «#» в файле шаблона **pug**.



Для исправления **Cookie Injection** в файле **server.js** исправим строку **res.cookie()**, добавив параметр **httpOnly: true**.

