

Анализ и демонстрация атак межсайтового скриптинга (XSS)

Болтовский Гавриил Александрович

Приамурский государственный университет им. Шолом-Алейхема

Студент

Фатеенков Данила Витальевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В статье рассматриваются различные типы атак межсайтового скриптинга (XSS) и их влияние на безопасность веб-приложений. Исследование проводится на примере учебного приложения OWASP Juice Shop, позволяющего выявить и проанализировать уязвимости. В результате представлены эффективные методы обнаружения и предотвращения XSS-атак.

Ключевые слова: информационная безопасность, JavaScript, XSS

Demonstration of XSS and CSRF attacks using the OWASP Juice Shop web training application as an example

Boltovsky Gavriil Alexandrovich

Sholom-Aleichem Priamursky State University

Student

Fateenkov Danila Vitalievich

Sholom-Aleichem Priamursky State University

Student

Abstract

The article discusses different types of cross-site scripting (XSS) attacks and their impact on the security of web applications. The study is conducted on the example of OWASP Juice Shop training application, which allows to identify and analyze vulnerabilities. As a result, effective methods for detecting and preventing XSS attacks are presented.

Keywords: information security, JavaScript, XSS

1 Введение

1.1 Актуальность исследования

Атаки межсайтового скриптинга (XSS) представляют серьезную угрозу информационной безопасности, так как позволяют злоумышленникам выполнять произвольный код в контексте веб-приложения, что может

привести к компрометации пользовательских данных, изменению содержимого веб-страниц и выполнению несанкционированных действий. Учитывая широкую распространенность данной уязвимости, её изучение имеет важное практическое значение для повышения уровня защищённости веб-приложений. Понимание механизмов атак XSS и эффективных методов защиты от них является неотъемлемой частью подготовки специалистов в области кибербезопасности и разработки программного обеспечения. Учебное веб-приложение OWASP Juice Shop предоставляет безопасную и функциональную среду для практического освоения аспектов выявления и предотвращения XSS-атак, что делает его полезным инструментом для обучения и профессиональной подготовки.

1.2 Обзор исследований

Вопросы обеспечения информационной безопасности веб-приложений занимают ключевое место в современных исследованиях, уделяющих особое внимание угрозам межсайтового скриптинга (XSS). В исследовании «Detection of cross-site scripting (XSS) attacks using machine learning techniques: a review» [1] рассматриваются актуальные подходы к автоматизированному выявлению XSS-атак с использованием методов машинного обучения, что подчеркивает важность разработки интеллектуальных систем защиты. Применение подобных методов позволяет значительно повысить уровень обнаружения уязвимостей за счет анализа больших объемов данных и выявления сложных шаблонов атак.

Комплексный подход к информационной безопасности веб-приложений представлен в работе А.Д. Чеснокова [2], где исследуются основные аспекты обеспечения защиты, включая механизмы предотвращения атак XSS. Автор подчеркивает необходимость учета множества факторов, влияющих на уязвимость веб-ресурсов, и важность внедрения эффективных практик безопасного программирования.

Практические аспекты создания и функционирования безопасных веб-приложений, соответствующих современным требованиям информационной безопасности, рассматриваются в исследовании «Информационная безопасность как элемент экономической безопасности» [3]. В данной работе акцентируется внимание на важности соблюдения принципов защиты данных, обеспечивающих устойчивость информационных систем перед угрозами, включая XSS-атаки, которые могут привести к серьезным экономическим потерям.

Использование современных веб-фреймворков считается важным шагом в повышении защищенности веб-приложений. Однако вопрос их устойчивости к XSS-атакам остается актуальным. В исследовании «Where we stand (or fall): An analysis of CSRF defenses in web frameworks» [4] анализируются механизмы защиты, реализованные в популярных инструментах разработки, и оценивается их эффективность. Изучение данных решений позволяет выявить потенциальные недостатки и разработать рекомендации по усилению защиты от атак XSS.

1.3 Цель исследования

Целью данного исследования является изучение и практическая демонстрация атак межсайтового скриптинга (XSS) и подделки межсайтовых запросов (CSRF) на примере учебного веб-приложения OWASP Juice Shop.

1.4 Постановка задачи

Для достижения поставленной цели предполагается изучить теоретические основы данных атак, их классификацию и механизмы эксплуатации, а также определить потенциальные точки входа в целевом веб-приложении. Далее проводится тестирование OWASP Juice Shop на уязвимости с последующей разработкой пошаговых сценариев атак, демонстрирующих эксплуатацию выявленных проблем безопасности.

2 Методы исследования

Межсайтовый скриптинг (XSS) представляет собой одну из наиболее распространенных уязвимостей веб-приложений, возникающую в результате внедрения вредоносного кода на веб-страницы, отображаемые пользователям. Данная атака позволяет злоумышленнику выполнять произвольные сценарии на стороне клиента, что может привести к краже конфиденциальных данных, компрометации пользовательских сессий и другим негативным последствиям. Уязвимость XSS возникает вследствие недостаточной фильтрации или экранирования пользовательских входных данных, которые в дальнейшем выводятся в веб-интерфейсе приложения.

Существует три основных типа атак межсайтового скриптинга: отражённый (Reflected XSS), хранимый (Stored XSS) и DOM-based XSS. Отражённый XSS-атакующий использует специально сформированный запрос, содержащий вредоносный код, который немедленно обрабатывается сервером и возвращается пользователю в ответе. Такая атака может быть реализована через URL-ссылки, GET- или POST-запросы и требует активного участия жертвы, например, перехода по скомпрометированной ссылке. Хранимый XSS представляет собой более опасную форму атаки, при которой вредоносный код сохраняется на сервере, а затем многократно исполняется на страницах, доступных другим пользователям. Это может происходить, например, при публикации комментариев или сообщений в веб-приложении. DOM-based XSS отличается тем, что атака осуществляется непосредственно в браузере пользователя, когда вредоносные данные обрабатываются на стороне клиента с помощью JavaScript без участия сервера.

OWASP Juice Shop является специализированным тренировочным веб-приложением, предназначенное для изучения и практической отработки навыков обеспечения информационной безопасности. Данный проект разработан в рамках инициативы Open Web Application Security Project (OWASP) и охватывает широкий спектр уязвимостей, включенных в список OWASP Top 10. Приложение имитирует функционирование типичного интернет-магазина, предоставляя пользователям возможность исследовать его

с точки зрения атакующего, выявлять и эксплуатировать различные уязвимости, а также разрабатывать методы их устранения.

Основной целью OWASP Juice Shop [5] является создание безопасной учебной среды, позволяющей специалистам на практике изучать механизмы атак межсайтового скриптинга (XSS) и отрабатывать методы их обнаружения и предотвращения без риска для реальных систем. Данный инструмент моделирует различные сценарии эксплуатации XSS-уязвимостей, демонстрируя возможности внедрения и исполнения вредоносного кода в веб-приложениях.

3 Результаты и обсуждения

OWASP Juice Shop предоставляет несколько способов развертывания, одним из которых является установка через пакетный менеджер npm. Этот метод позволяет быстро и удобно запустить приложение на локальной машине без необходимости использования дополнительных инструментов контейнеризации. Для установки OWASP Juice Shop через npm необходимо предварительно установить Node.js, git и пакетный менеджер npm. После установки необходимых компонентов процесс развертывания заключается в выполнении команд:

```
git clone https://github.com/juice-shop/juice-shop.git
cd juice-shop
npm install
npm start
```

Альтернативным методом развертывания является использование контейнеризации с помощью Docker.

Установленное приложение доступно в браузере по адресу <http://localhost:3000> (рис. 1).

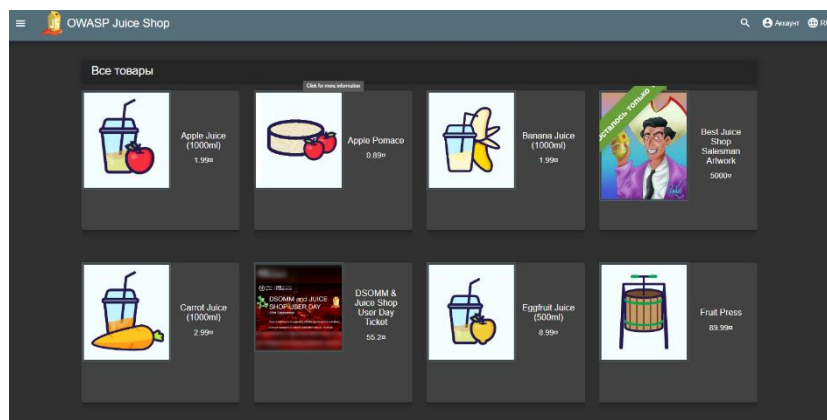


Рисунок 1 – Главная страница веб-приложения

Данный сайт уязвим для XSS атак. Этим видом атак обычно атакуются различные поля для ввода. Это могут быть поисковые строки, формы для комментариев.

Рассмотрим атаку на поисковую строку (рис. 2). Код поисковой выдачи можно увидеть через инструменты разработчика в браузере.

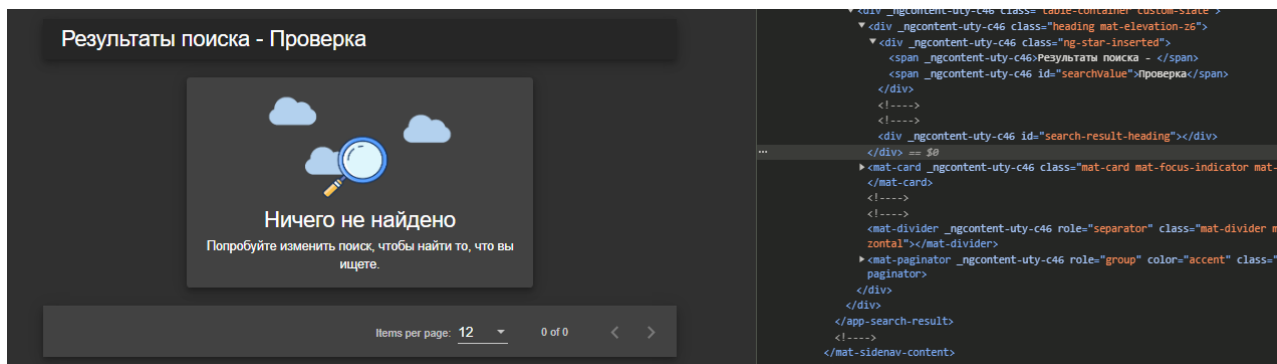


Рисунок 2 – Главная страница веб-приложения.

Результат поиска выводится в теге `span`. Можно попробовать интегрировать в поисковую строку какой-либо сторонний код.

```
"><script>alert('XSS')</script>
```

В данном примере происходит ложное закрытие тега `span`, добавляется сторонний JS скрипт (рис. 3).

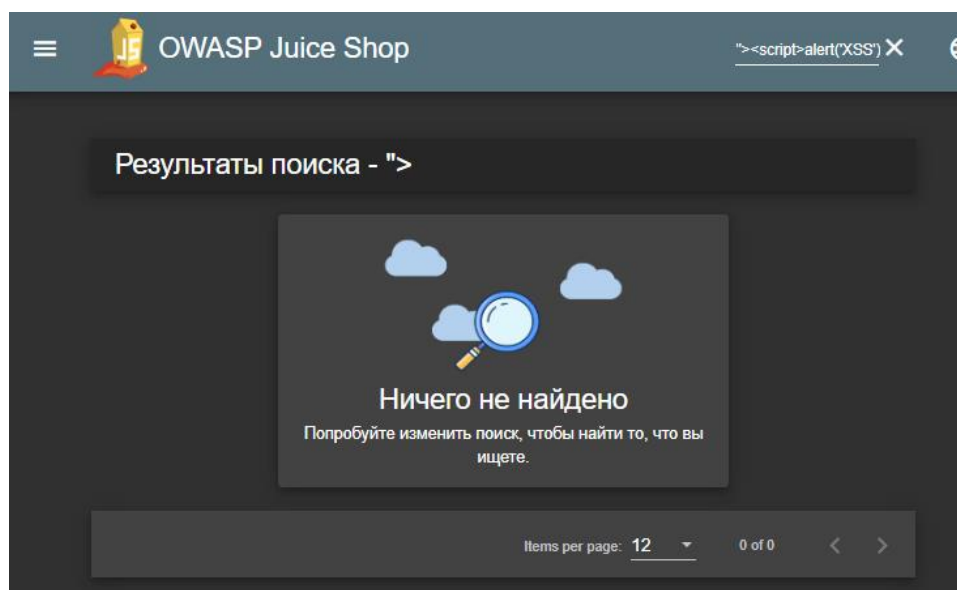


Рисунок 3 – Атака результата не дала

Видно, что на сайте предусмотрена защита. Однако, её можно обойти, если интегрировать сторонний код через атрибут другого тега.

```
"><img src=x onerror=alert('XSS')>
```

Результат выполнения такого кода представлен на рисунке (рис. 4).

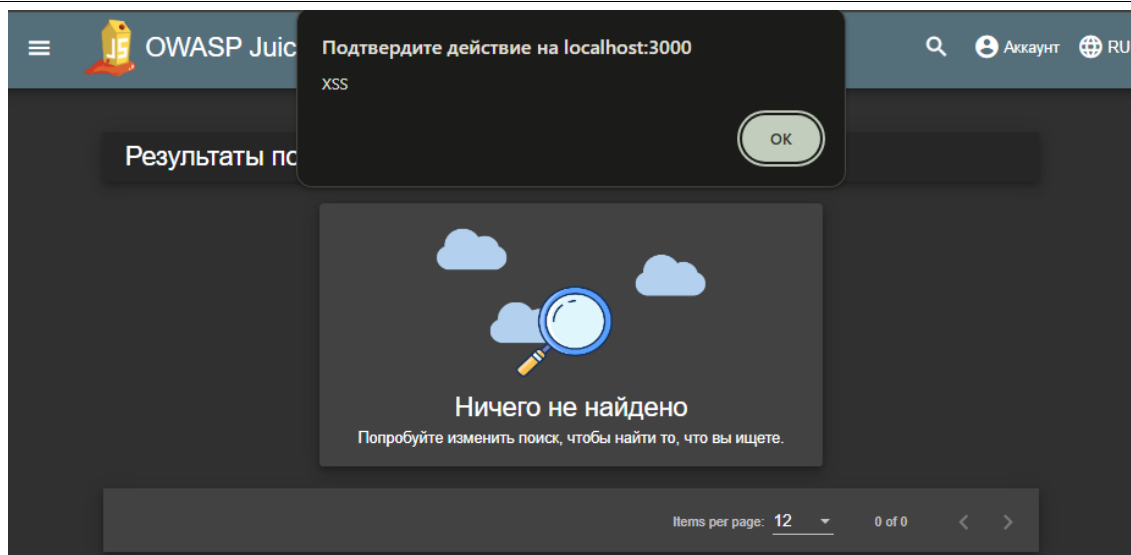


Рисунок 4 – Поиск приводит к исполнениям чужого кода

Показанные примеры демонстрируют отражённую XSS атаку. Она приводит к исполнению чужого кода только на странице конечного пользователя и не сохраняется на сервере. Возможна так же через адресную строку

Хранимая XSS атака может быть доступна в разделе комментариев на сайте, когда чужой комментарий сохраняется и может содержать сторонний код (рис. 5).

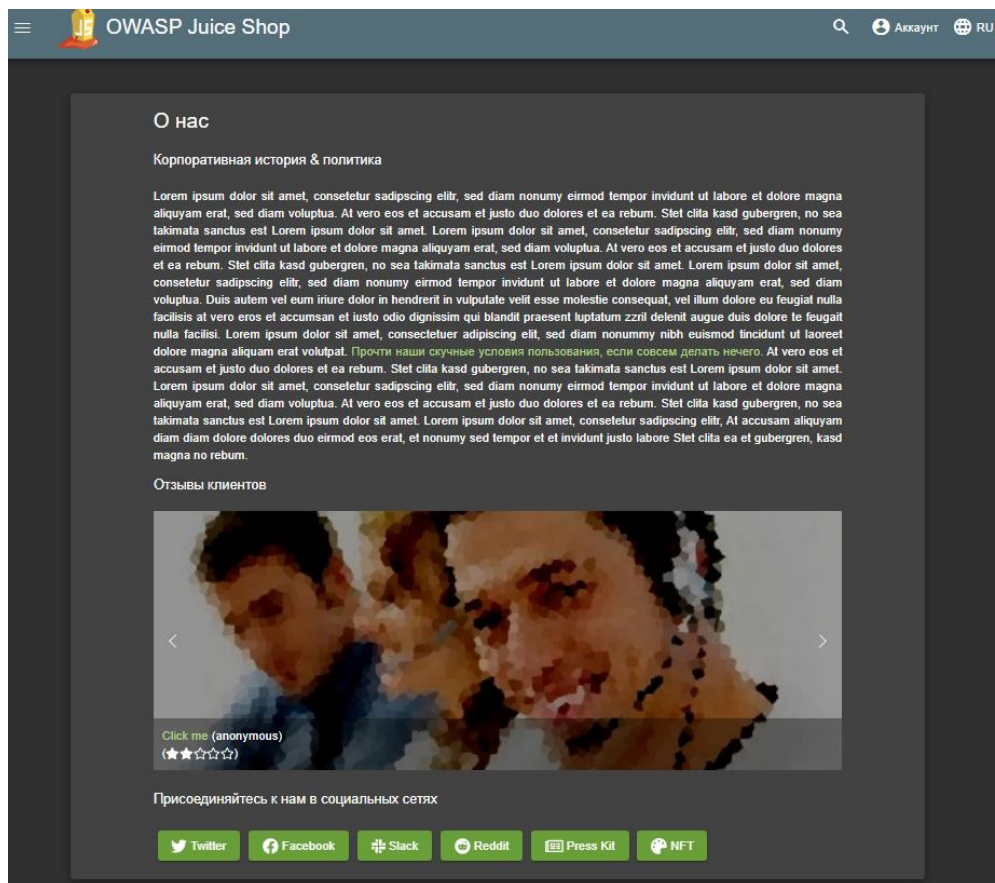


Рисунок 5 – Пример хранимой атаки

Переход по гиперссылке в комментарии приводит к исполнению произвольного кода.

Защита веб-приложений от атак межсайтового скриптинга (XSS) требует комплексного подхода, включающего как корректную обработку пользовательского ввода, так и применение надежных механизмов безопасности. Одним из ключевых аспектов является фильтрация и экранирование входных данных перед их использованием в динамически генерируемом контенте. Это помогает предотвратить внедрение вредоносного кода в HTML-страницы. Также важно применять современные заголовки безопасности, такие как Content Security Policy (CSP), которые позволяют ограничить источники исполняемого кода и минимизировать риски успешной эксплуатации уязвимостей. В процессе разработки необходимо придерживаться принципов безопасного программирования, используя готовые решения для работы с пользовательским вводом, например, встроенные средства фреймворков и библиотек, которые обеспечивают автоматическую защиту от XSS. Регулярное тестирование приложения с применением инструментов статического и динамического анализа помогает своевременно выявлять потенциальные угрозы и устранять их до того, как они станут серьезной проблемой.

4 Выводы

Таким образом были продемонстрированы различные XSS атаки на веб-приложение. В качестве веб-приложения выступила тестовая информационная система OWASP Juice Shop, в которой преднамеренно были допущены ошибки при проектировании, чтобы разработчики могли практиковаться в умении работы с различными рода атаками на сервера (в том числе и с XSS-атаками).

Библиографический список

1. Kaur J., Garg U., Bathla G. Detection of cross-site scripting (XSS) attacks using machine learning techniques: a review //Artificial Intelligence Review. 2023. Т. 56. №. 11. С. 12725-12769.
2. Чесноков А. Д. Информационная безопасность // StudNet. 2022. Т. 5. №. 1. – С. 478-489.
3. Ибрагимова З. М., Батчаева З. Б., Ткаченко А. Л. Информационная безопасность как элемент экономической безопасности // Инженерный вестник Дона. 2022. №. 11 (95). С. 26-33.
4. Likaj X., Khodayari S., Pellegrino G. Where we stand (or fall): An analysis of CSRF defenses in web frameworks //Proceedings of the 24th International Symposium on Research in Attacks, Intrusions and Defenses. 2021. С. 370-385.
5. OWASP Juice Shop. URL: <https://owasp.org/www-project-juice-shop/> (дата обращения: 25.01.2025)