МЕТОДЫ ОБНАРУЖЕНИЯ И ПРЕДОТВРАЩЕНИЯ УЯЗВИМОСТЕЙ В ФРОНТЕНД РАЗРАБОТКЕ

Логинова Н.В.

Логинова Наталья Владиславовна - старший инженер-программист,
компания Epam systems Montenegro DOO Podgorica,
г. Подгорица

Аннотация: *статья посвящена проблеме обеспечения безопасности во фронтенд разработке. Рассмотрены основные типы уязвимостей, которые могут возникнуть в процессе работы, и представлены эффективные методы их обнаружения и предотвращения. Статья может быть полезной как для начинающих разработчиков, которые только начинают знакомиться с проблематикой безопасности при разработке веб-приложений, так и для опытных программистов, которые хотят углубить свои знания в этой области. Приведенные в статье меры и рекомендации будут способствовать улучшению качества кода и повышению уровня защиты, разрабатываемых веб-риешенииий.*
Ключевые слова: *фронтенд разработка, безопасность веб-приложений, обнаружение уязвимостей, предотвращение уязвимостей, безопасное программирование, санитизация ввода, анализ кода, пентестинг, контейнер-сервисы, аудит зависимостей, кодирование данных.*

METHODS OF DETECTION AND PREVENTION OF VULNERABILITIES IN FRONTEND DEVELOPMENT

Loginova N.V.

Loginova Natalia Vladislavovna – Senior Software Engineer,
EPAM SYSTEMS MONTENEGRO DOO PODGORICA,
PODGORICA

Abstract: *This article is devoted to the issue of ensuring security in frontend development. The main types of vulnerabilities that may arise in the process of work are examined, and effective methods of their detection and prevention are presented. The article may be useful for beginner developers who are just starting to get acquainted with the issue of security in web application development, as well as for experienced programmers who want to deepen their knowledge in this area. The measures and recommendations outlined in the article will contribute to the improvement of code quality and an increased level of protection for the developed web solutions.*
Keywords: *frontend development, web application security, vulnerability detection, vulnerability prevention, secure programming, input sanitization, code analysis, pentesting, container services, dependency auditing, data encoding.*

УДК 004.056.53

1. Введение

В современном мире, где цифровые технологии стали неотъемлемой частью нашей повседневной жизни, вопрос безопасности веб-приложений стоит весьма остро. Фронтенд разработка, которая обеспечивает взаимодействие пользователя с веб-сайтами, становится одной из ключевых областей, в которой необходимо обеспечить надежную защиту от уязвимостей. Уязвимости во фронтенде могут привести к несанкционированному доступу к персональной информации пользователей, взлому системы с последующим распространением вредоносного программного обеспечения и другим проблемам. Поэтому использование эффективных методов обнаружения и предотвращения подобных уязвимостей является не только рекомендацией, но и частью потребности в гарантированной безопасности. В этой статье мы рассмотрим, какие уязвимости могут возникнуть во фронтенд разработке, как их обнаружить и, что самое главное, как предотвратить их появление.

2. Разновидности уязвимостей во фронтенде

Перед тем как подробно разобрать методы обнаружения и устранения уязвимостей, необходимо понимать, с какими конкретно угрозами мы имеем дело [3, 8]. В области фронтенд разработки выделяют несколько наиболее распространенных и опасных типов уязвимостей, каждая из которых требует специфического подхода в решении проблемы безопасности. Давайте рассмотрим каждый из них более подробно.

Разновидности уязвимостей во фронтенде:

1. Cross-Site Scripting (XSS): Это одна из самых распространенных уязвимостей, которая позволяет атакующим внедрять произвольный JavaScript-код в страницы, которые просматривают другие пользователи. Если XSS уязвимость существует в веб-приложении, атакующие могут перехватывать сессии пользователей, вредить контенту сайта и выполнять другие действия от имени жертвы [11, 12].

2. Cross-Site Request Forgery (CSRF): Эта уязвимость позволяет атакующим выдавать заявки от имени жертвы без ее знания или согласия. Это может привести к несанкционированным действиям в веб-приложениях, которые пользователь считает безопасными.

3. Clickjacking: Техника, при которой атакующий "маскирует" вредоносные ссылки или кнопки под невинные элементы веб-страницы. Пользователь, кликая по ним, несет риск стать жертвой фишинга или других вредоносных действий.

4. Межсайтовое скриптовое включение (JSON/JavaScript Hijacking): Подобно XSS, но атакующий использует кросс-доменные запросы для извлечения JSON или JavaScript данных, которые не должны быть доступны.

Несмотря на разнообразие уязвимостей, которые могут встречаться во фронтенде разработке, существуют специализированные инструменты и методы, способные эффективно обнаруживать и предотвращать их [1 ,6, 7]. Понимание специфики каждого типа уязвимости позволяет разработчикам строить правильную стратегию обеспечения безопасности и вовремя принимать надлежащие меры для защиты веб-приложений от потенциальных атак. В следующих разделах мы подробно рассмотрим, какие инструменты и подходы существуют для обнаружения и предотвращения данных угроз, для того чтобы вы могли применить знания на практике.

3. Методы обнаружения уязвимостей в фронтенд разработке

После того как мы узнали о основных типах уязвимостей во фронтенд разработке [4], необходимо изучить какие существуют методы и подходы для их обнаружения. Точное и своевременное определение уязвимостей является одним из ключевых этапов в укреплении безопасности веб-приложения [9]. Рассмотрим несколько широко используемых методов обнаружения уязвимостей.

Методы обнаружения уязвимостей:

1. Мануальный анализ кода. Данный метод обычно используется программистами для проверки кода вручную на предмет ошибок и уязвимостей [5]. Этот метод полезен для обнаружения специфических для проекта уязвимостей, которые могут быть пропущены автоматическими инструментами [13].

2. Автоматическое сканирование кода. Этот метод включает в себя использование программного обеспечения или онлайн-сервисов для сканирования кода на предмет распространенных уязвимостей. Эти инструменты обычно базируются на наборе правил, позволяющих идентифицировать потенциальные угрозы [14].

3. Пентестинг. Другой вариант - использование профессиональных "инструментов хакера" для тестирования безопасности веб-приложений. Пентестеры, или специалисты по проникновению, обычно используют атакующие стратегии для выявления уязвимостей, которые могут быть затем исправлены.

4. Контейнер-сервисы для проверки безопасности. Специальные сервисы, такие как Docker Bench, помогают обнаруживать уязвимости на уровне контейнеров, что жизненно важно для приложений, использующих микросервисную архитектуру.

5. Аудит зависимостей. Многие современные проекты активно используют сторонние библиотеки и пакеты, в которых также могут присутствовать уязвимости. Для их выявления следует регулярно проводить аудит зависимостей.

Обнаружение уязвимостей во фронтенд-разработке - это сложный и непрерывный процесс. Технологии могут меняться, и новые уязвимости могут возникнуть со временем [2]. Поэтому методы обнаружения должны быть так же динамичными и адаптивными. Эффективное использование представленных методов обнаружения вместе с регулярным обновлением знаний о лучших отраслевых практиках обеспечит заметное сокращение риска кибератак [15]. В то же время, просто обнаруживание уязвимостей не достаточно, также важно предпринимать шаги для их устранения и предотвращения, о которых мы поговорим в следующем разделе.

4. Методы предотвращения уязвимостей в фронтенд разработке

После обнаружения уязвимостей критически важно предпринять эффективные меры для их устранения и предотвращения в будущем. Фронтенд разработчики должны быть знакомы с основными техниками и механизмами обеспечения безопасности в коде. Давайте рассмотрим наиболее распространенные методы предотвращения уязвимостей, которые должен использовать каждый разработчик [14].

Методы предотвращения уязвимостей:

1. Применение Web Content Security Policy (CSP). Этот инструмент позволяет задать принципы, согласно которым будет загружаться весь контент на сайте. Благодаря ему можно весьма серьёзно ограничить возможности для внедрения вредоносного кода.

2. Санитизация ввода. Является ключевым механизмом для предовращения межсайтового скриптинга (XSS) и инъекций. Санитизация включает в себя контроль, обработку и проверку данных, вводимых пользователем [2].

3. Использование безопасных HTTP-заголовков. Некоторые HTTP-заголовки были разработаны для предотвращения определенных типов атак. Например, заголовки, такие как X-Frame-Options и X-XSS-Protection, могут помочь в борьбе с clickjacking и XSS соответственно.

4. Использование принципа наименьших привилегий. Каждый процесс должен иметь минимальный набор привилегий, которые необходимы ему для выполнения своих функций. Это помогает ограничить возможные действия в случае компрометации процесса.

5. Кодирование данных. В местах, где санитизация ввода является недостаточной, кодирование обеспечивает безопасность, преобразуя ввод пользователя в безопасный формат [10].

6. Статический анализ кода. Использование инструментов статического анализа помогает выявить уязвимости на ранних стадиях разработки.

Резюмируя, можно сказать, что использование вышеуказанных методов предотвращения уязвимостей обеспечивает существенное улучшение безопасности веб-приложений. Эти подходы позволяют уменьшить риск несанкционированного доступа и возможного ущерба, который может возникнуть в случае возникновения этих уязвимостей. Следует помнить, что каждый разработчик играет ключевую роль в обеспечении безопасности приложения. Поэтому реализация эффективных методов предотвращения - это не только вопрос профессионализма, но и ответственности перед пользователями.

5. Анализ безопасности на примере веб-приложения “MyBookstore”

В процессе работы над улучшением безопасности веб-приложения "MyBookstore", команда разработчиков нашла и устранила ряд уязвимостей различных типов. Работа над улучшением безопасности - неоднократный и постоянный процесс, требующий использования различных методов обнаружения уязвимостей.

Для обнаружения угроз была применена комбинация методов, включающих ручной анализ кода, автоматическое сканирование, использование сервисов контейнеров и аудит зависимостей.

Понимание того, какие методы обнаружения оказались наиболее эффективными в конкретной ситуации, является ключевым для оптимизации процессов по улучшению безопасности. Взглянув на объем уязвимостей, обнаруженных каждым методом, и учитывая степень их критичности, мы можем сделать выводы об эффективности этих методов.

В следующей таблице подробно представлены данные об эффективности различных методов обнаружения уязвимостей, применяемых на "MyBookstore". Данные приведены в Таблице 1.

Таблица 1. Методы обнаружения уязвимостей и количество выявленных уязвимостей по степени критичности.

Метод обнаружения	**Всего обнаружено уязвимостей**	**Уязвимостей с высокой критичностью**	**Уязвимостей с средней критичностью**	**Уязвимостей с низкой критичностью**
Мануальный анализ кода	20	5	8	7
Автоматическое сканирование	25	7	9	9
Пентестинг	15	4	6	5
Контейнер-сервисы	7	2	3	2
Аудит зависимостей	10	3	4	3

Полученные данные позволяют сделать несколько ключевых выводов относительно эффективности различных методов обнаружения уязвимостей. Автоматическое сканирование оказалось самым эффективным методом обнаружения по количеству найденных уязвимостей. Это подчеркивает важность автоматических инструментов сканирования в современной разработке ПО.

Несмотря на то, что ручной анализ кода и пентестинг обнаружили меньшее количество уязвимостей по сравнению с автоматическим сканированием, они оказались особенно эффективными в выявлении уязвимостей с высоким и средним уровнем критичности. Это подтверждает, что экспертный взгляд и человеческий фактор остаются важными составляющими в обеспечении безопасности ПО.

В то же время, использование сервисов контейнеров и аудит зависимостей, несмотря на обнаружение меньшего количества уязвимостей, внесли важный вклад в усиление безопасности веб-приложения "MyBookstore". Это подчеркивает, что для общей стратегии обеспечения безопасности важно использовать различные методы обнаружения уязвимостей.

В общем и целом, команда разработки "MyBookstore" должна продолжать использовать комбинированный подход к обнаружению уязвимостей, чтобы обеспечивать всеобъемлющий и эффективный контроль за безопасностью веб-приложения.

Важно не только обнаруживать уязвимости, но и понимать, где именно они возникают. Проэтому был проведен анализ, целью которого было определить, в каких областях приложения обнаружено наибольшее количество уязвимостей. Ниже на рисунке 1 приведена столбчатая диаграмма, которая наглядно показывает распределение обнаруженных уязвимостей по различным областям приложения "MyBookstore". Это поможет понять, в каких областях приложения наиболее часто могут встречатся уязвимости, и соответствующим образом скорректировать командой разработки стратегию безопасности.

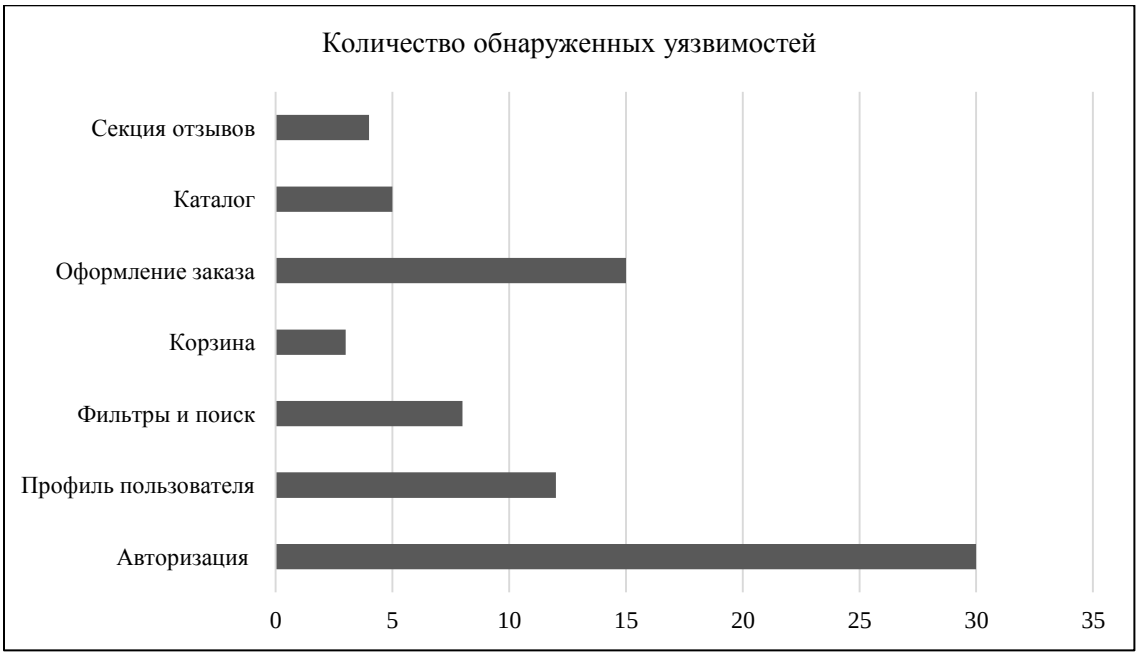

Рис. 1. Распределение обнаруженных уязвимостей по различным областям веб-приложения MyBookstore."

Изучив распределение уязвимостей по различным областям веб-приложения "MyBookstore", мы можем сделать несколько ключевых выводов.

Наибольшее количество уязвимостей было обнаружено в авторизации, что делает эту область приложения наиболее уязвимой. Это подчеркивает необходимость улучшения системы авторизации и подтверждает, что безопасность данных пользователя должна быть приоритетом в процессе разработки веб-приложения.

Также стоит обратить внимание на область "Оформление заказа". В этой части веб-приложения было выявлено значительное количество уязвимостей. Кроме того, область "Оформление заказа" также является критической для безопасности, поскольку здесь пользователи оставляют свои личные данные и данные банковских карт.

С другой стороны, области "Корзина", "Каталог" и "Секция отзывов" имеют относительно меньшее количество уязвимостей, что говорит о хорошей начальной защищенности этих составляющих веб-приложения "MyBookstore".

Эти данные помогают определить действия по поддержанию безопасности приложения и наметить, в каких областях необходимы дополнительные усилия. Работа над безопасностью - это процесс непрерывного улучшения, и эти данные служат важным инструментом для постоянного мониторинга и отслеживания прогресса.

При обнаружении уязвимостей команда предпринимала необходимые действия для их устранения и усилила защитные меры для предотвращения подобных угроз в будущем. На рисунке 2 представлена

диаграмма с различными типами уязвимостей, которые были обнаружены, и время, затраченное на их устранение без учёта времени, потраченного на внедрение методов предотвращения уязвимостей, командой разработки "MyBookstore".

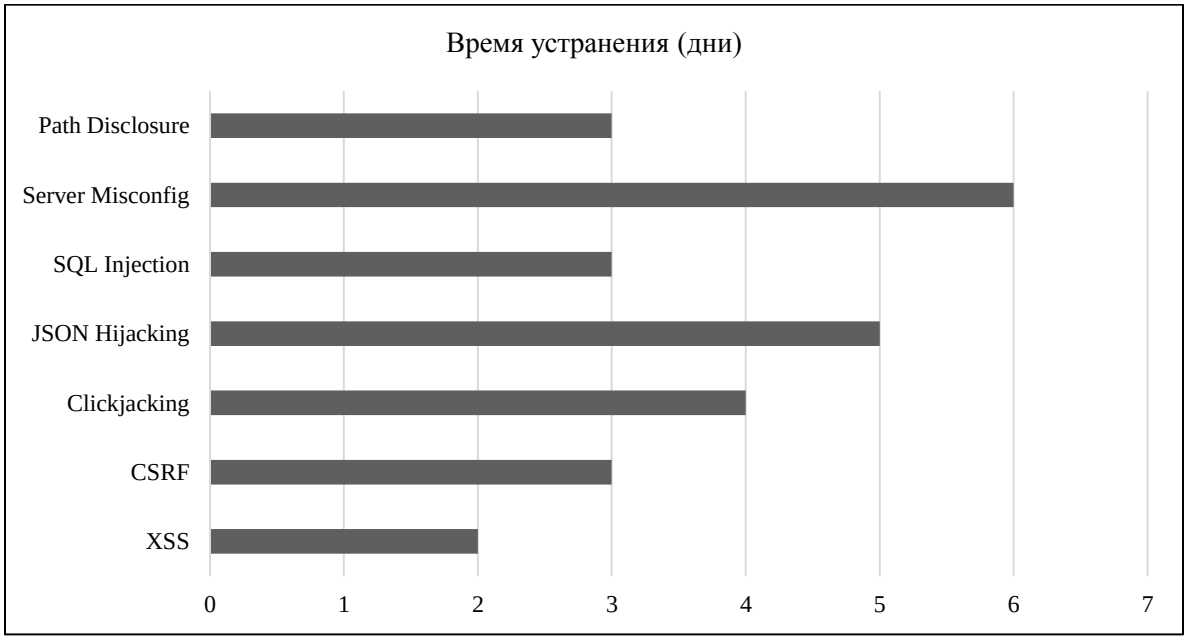

Рис. 2. Время устранения различных типов уязвимостей в веб-приложении "MyBookstore."

Из представленных данных видно, что время, за которое удаётся устранить различные типы уязвимостей, варьируется. Так, для устранения уязвимостей типа JSON Hijacking и Server Misconfig потребовалось наибольшее время, что свидетельствует о высокой сложности данных угроз.

С другой стороны, уязвимости типа XSS, CSRF, SQL Injection и Path Disclosure были исправлены быстрее, что говорит о хорошей подготовке команды разработчиков и эффективности методов, используемых для устранения проблем.

Оценка эффективности и сложности внедрения различных методов предотвращения уязвимостей является ключевым аспектом при планировании стратегии безопасности веб-приложения. Особенно это актуально при разработке таких сложных проектов, как веб-приложение "MyBookstore".

Далее представлена таблица 2, в которой приведены методы предотвращения уязвимостей, которые были применены командой разработки веб-приложения "MyBookstore". В таблице методы предотвращения уязвимостей сраниваются в контексте их эффективности, сложности внедрения, требований к квалификации IT-специалистов и количества устранённых уязвимостей.

Таблица 2. Сравнительный анализ методов предотвращения уязвимостей в веб-приложении 'MyBookstore'.

Метод предотвращения	**Эффективность**	**Уровень сложности внедрения**	**Требования к квалификации IT-специалиста**	**Уязвимостей устранено**
Шифрование данных	Высокая	Сложное	Высокие	7
Санитизация ввода	Средняя	Простое	Средние	23
Двухфакторная аутентификация	Высокая	Среднее	Высокие	8
Использование Content Security Policy (CSP)	Средняя	Сложное	Высокие	29
Использование Web Application Firewalls (WAF)	Высокая	Среднее	Средние	10

На основе представленной таблицы можно сделать ряд выводов о различных методах предотвращения уязвимостей, применённых командой разработки в веб-приложении "MyBookstore".

Во-первых, видно, что методы шифрование данных и двухфакторная аутентификация были выбраны командой "MyBookstore" для обеспечения высокого уровня защиты. Эти методы сложны в внедрении и требуют высокой квалификации IT-специалиста, но в то же время они обладают высокой степенью эффективности в предотвращении уязвимостей.

Во-вторых, команда "MyBookstore" выбрала в качестве одного из методов защиты санитизацию ввода. Несмотря на то, что он обладает средней эффективностью, этот метод представляет собой простое в применении решение, требующее относительно небольшого уровня квалификации специалистов.

Также в веб-приложении "MyBookstore" было использовано Content Security Policy (CSP). Этот метод обеспечивает средний уровень защиты, однако считаются относительно сложными в реализации и требуют высокой квалификации IT-специалистов.

Другим важным методом, применённым для предотвращения уязвимостей веб-приложения, было использование Web Application Firewalls (WAF). Метод обладает средней эффективностью и является относительно несложным в внедрении.

Таким образом, команда "MyBookstore" приняла решение о внедрении множества различных методов защиты, каждый из которых имеет свои преимущества и недостатки. Это позволило обеспечить глубокую защиту веб-приложения на разных уровнях и защитить его от различных типов уязвимостей.

Внедрение методов безопасности в веб-приложение - это основополагающий шаг в его разработке, призванный защитить его от потенциальных уязвимостей. Однако, каждый метод требует разного количества времени для корректного внедрения. Чтобы более наглядно представить эту разницу в затрачиваемых на внедрение ресурсах, была построена диаграмма, которая представлена на рисунке 3.

Рис. 3. Сравнение времени внедрения методов предотвращения уязвимостей в веб-приложении 'MyBookstore.

Согласно построенной диаграмме можно увидеть, что наибольший промежуток времени уходит на внедрение метода шифрования данных. Это говорит о сложности данного метода и о необходимости его тщательной реализации для обеспечения безопасности данных пользователей.

Методы использование Content Security Policy (CSP) и использование Web Application Firewalls (WAF) занимают сравнимое количество времени - 4 дня, что отражает их статус как важных и действенных инструментов защиты веб-приложения от различных видов атак.

Двухфакторная аутентификация и санитизация ввода, несмотря на эффективность в предотвращении уязвимостей, требуют меньше времени на внедрение - 3 и 2 дня соответственно. Это говорит об относительной легкости внедрения данных методов и простоте их использования в контексте проекта.

В общем и целом, данные на диаграмме предоставляют ценную информацию о требуемых при разработке веб-приложений ресурсах времени для внедрения различных методов защиты от уязвимостей.

6. Практические рекомендации по обнаружению и предотвращению уязвимостей

На основании проведенного анализа безопасности для веб-приложения "MyBookstore" можно сформулировать следующие практические рекомендации по обнаружению и предотвращению уязвимостей.

Прежде всего, следует продолжать использовать комбинированный подход в обнаружении уязвимостей. Анализ показывает, что комбинация различных методов (анализ кода, автоматическое сканирование, пентестинг, сервисы контейнеров и аудит зависимостей) позволяет обнаружить наибольшее количество угроз. Необходимо особое внимание уделить областям авторизации и оформления заказа. Эти зоны оказались наиболее уязвимыми в веб-приложении "MyBookstore", поэтому повышенное внимание к безопасности в этих областях поможет предотвратить большую часть возможных уязвимостей.

Важно также прилагать усилия к устранению уязвимостей, используя различные методы. Исследование показало, что методы, такие как шифрование данных, использование двухфакторной аутентификации, санитизация ввода, использование Content Security Policy (CSP) и Web Application Firewalls (WAF), хоть и требуют разного количества ресурсов на внедрение, оказываются достаточно эффективными.

Важность учета сложности и времени, необходимого для внедрения различных методов безопасности, не может быть недооценена. Этот аспект играет ключевую роль на этапе планирования, когда речь идет о разработке и реализации мер безопасности веб-приложения. В контексте "MyBookstore" мы столкнулись с тем, что внедрение метода шифрования данных было наиболее трудоемким и длительным процессом. Это подчеркивает необходимость проведения тщательного анализа и планирования на ранних стадиях разработки, для того чтобы наиболее эффективно интегрировать различные методы устранения уязвимостей.

Кроме того, необходимо постоянно обновлять и анализировать данные о последних тенденциях и изменениях в сфере безопасности. Мир кибербезопасности постоянно меняется и новые угрозы, и уязвимости могут появляться в любое время. Поэтому постоянный мониторинг является ключевым в сохранении безопасности ПО.

Обучение сотрудников также является важной частью процесса улучшения безопасности веб-приложений. Повышение их уровня знаний и навыков в области безопасности только усилит защиту вашего веб-приложения.

Регулярное тестирование, применение принципа наименьших привилегий и создание резервных копий данных также являются важными шагами в стратегии эффективного обеспечения безопасности. Кроме того, стоит обратить внимание на установку и конфигурацию систем обнаружения уязвимостей и систем предотвращения уязвимостей.

Тщательное и внимательное применение этих практических рекомендаций поможет улучшить уровень безопасности веб-приложения "MyBookstore", а также поможет минимизировать риски, связанные с уязвимостями. Ключевой момент здесь - это непрерывный процесс обнаружения, анализа и устранения уязвимостей, а также применение передовых технологий для обеспечения максимальной защиты данных пользователей.

7. Заключение

Обеспечение безопасности веб-приложений, таких как "MyBookstore", является одним из ключевых аспектов в сфере фронтенд разработки. "MyBookstore" - это динамичное веб-приложение, которое предоставляет пользователям доступ к широкому спектру функций, включая поиск и покупку книг, различные формы взаимодействия с контентом и многие другие. Эти функции делают "MyBookstore" одновременно удобным в использовании, но и потенциально уязвимым для различных видов атак.

В процессе анализа мы обнаружили ряд возможных уязвимостей, которые могут быть эксплуатированы злоумышленниками. Реагирование на эти угрозы требует использования комплекса различных методов обнаружения и предотвращения, которые были описаны в данной работе. При использовании этих методов стоит помнить о непрерывности и многогранности процесса обеспечения безопасности - он затрагивает все этапы разработки веб-приложения, от проектирования архитектуры до написания конкретных блоков кода.

Безопасность - это постоянная работа. В свете продолжающегося роста и эволюции интернета, а также все большего использования веб-приложений в повседневной жизни, обеспечение безопасности "MyBookstore" и подобных веб-приложений становится все более важной и актуальной задачей. Пользуясь приведенными методами и следуя приведенным рекомендациям, мы, как разработчики, можем значительно повысить уровень защиты своих веб-приложений и обеспечить безопасность для миллионов пользователей в сети.

Список литературы / References

1. *Seacord R. C.* (2013). Secure coding in C and C++. Pearson Education.

2. *Zalewski M.* (2011). Silence on the wire: a field guide to passive reconnaissance and indirect attacks. No Starch Press.
3. OWASP Foundation. (2017). OWASP Top 10 - 2017: The Ten Most Critical Web Application Security Risks.
4. *Zakas N.C.* (2014). Principles of Object-Oriented JavaScript. No Starch Press.
5. *Zeller A., & Prechelt L.* (2001). Type inference of statically checked exceptions. ACM Transactions on Software Engineering and Methodology (TOSEM), 10(1), 20-57.
6. *Adams B., Jung H., & Han'A D.* (2011). How and why JavaScript developers use linters. IEEE Software, 34(2), 45-50.
7. *Barnum S.* (2020). Common Attack Pattern Enumeration and Classification (CAPEC). MITRE Corporation.
8. *Williams A., Wichers D., & Boberski J.* (2011). The OWASP testing guide. The OWASP Foundation.
9. *Silber J., Crosta M., & Ball T.* (2016). How we learned to build a semantic browser extension. IEEE Internet Computing, 20(1), 32-40.
10. *Bell D., & Parrish B.* (2016). Building a static site generator with React, Preact, and webpack. ACM SIGAPP Applied Computing Review, 16(4), 37-44.
11. *Van Acken E., Vanhoef M., Schils M., & Piessens F.* (2013). ScriptShield: preventing injection attacks on web applications by real-time detection and prevention of script injections. In Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security.
12. *Kostadinov D., & Vassilev R.* (2017). A new XSS detection and protection approach based on html entity encoding. In Proceedings of the 18th International Conference on Computer Systems and Technologies.
13. *Wang Z., Cheng M., Gu G., Ratliff B., & Zhang H.* (2016). Spider: stealthy binary program instrumentation and reverse engineering. In Proceedings of the 31st Annual Computer Security Applications Conference.
14. *Cimpan S., Gazagnaire T., & Madhavapeddy A.* (2017). Safe and secure compilers: validation and verification. In Proceedings of the 7th ACM SIGPLAN International Workshop on Mechanizing Metatheory.
15. *Schrittwieser S., Kieseberg P., Echizen I., Wohlgemuth S., Sonehara N., & Weippl E.* (2012). Internet Anonymity in the Age of Web 2.0: The Case of Tor and Facebook. In ECIS.