Министерство образования Калининградской области

государственное автономное учреждение

Калининградской области

профессиональная образовательная организация

«Колледж предпринимательства»

**КУРСОВАЯ РАБОТА**

**Тема: Анализ современных угроз информационной безопасности на веб-сайтах и способы их предотвращения**

Выполнил:

обучающийся группы ОБС 21-2

специальность 10.02.05 Обеспечение информационной безопасности автоматизированных систем

\_\_\_\_\_\_\_\_\_\_\_ Кудин А.Ю.

Руководитель:

\_\_\_\_\_\_\_\_\_\_\_ Трофимова Т.Д.

Калининград

2024

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc162829192)

[**1.** **ВИДЫ УЯЗВИМОСТЕЙ НА ВЕБ-САЙТАХ И СПОСОБЫ ИХ ПРЕДОТВРАЩЕНИЯ** 5](#_Toc162829193)

[**1.1 Используемые технологии** 5](#_Toc162829194)

[**1.2 Безопасность веб-ресурсов** 5](#_Toc162829195)

[**1.3 Протоколы HTTP и HTTPS** 6](#_Toc162829196)

[**1.4 Межсайтовый скриптинг (XSS)** 7](#_Toc162829197)

[**1.5 SQL инъекции** 10](#_Toc162829198)

[**2.** **ИМИТАЦИЯ XSS АТАК И ТЕСТИРОВАНИЕ СПОСОБОВ ИХ ПРЕДОТВРАЩЕНИЯ** 11](#_Toc162829199)

[**2.1** **Подготовка тестовой среды** 11](#_Toc162829200)

# **ВВЕДЕНИЕ**

В современном информационном обществе, где веб-сайты стали не только платформой для предоставления информации, но и основным каналом коммуникации, совершения покупок, проведения финансовых операций и многих других деятельностей, обеспечение информационной безопасности (ИБ) становится жизненно важным аспектом. Все больше организаций и частных лиц привлекаются к виртуальному пространству, что увеличивает риск возникновения различных угроз безопасности данных.

Сложившаяся ситуация поднимает ряд важных вопросов о защите данных, сохранности конфиденциальной информации и обеспечении непрерывной работоспособности веб-ресурсов. Угрозы информационной безопасности веб-сайтов могут варьироваться от традиционных атак, таких как SQL инъекции, к краже личных данных и распространению вредоносного программного обеспечения, такого как вредоносные скрипты и вирусы.

Для эффективной защиты веб-сайтов необходимо не только анализировать существующие угрозы, но и разрабатывать и применять соответствующие методы и средства их предотвращения. Это может включать в себя использование современных методов шифрования данных, регулярное обновление программного обеспечения и патчей безопасности, реализацию многоуровневой архитектуры защиты, обучение сотрудников по вопросам информационной безопасности, а также установку и настройку средств обнаружения и предотвращения атак.

Целью работы является проанализировать современные угрозы безопасности на веб-сайтах и выявить способы их предотвращения.

Для решение поставленной цели необходимо реализовать задачи:

* Изучить теоретическую составляющую современных угроз;
* Проанализировать методы реализации угроз на веб-сайты, а также их предотвращение;
* Проанализировать какие методы атак и защиты используются злоумышленниками;
* Создать среду для тестирования методов защиты от атак;
* Провести атаки на тестовую среду;
* Обеспечить защиту тестовой среды от атак.

# **ВИДЫ УЯЗВИМОСТЕЙ НА ВЕБ-САЙТАХ И СПОСОБЫ ИХ ПРЕДОТВРАЩЕНИЯ**

## **1.1 Используемые технологии**

Чтобы разобраться с уязвимостями на веб-сайтах, необходимо сначала понять, что из себя представляет веб-сайт, и с помощью каких технологий, языков программирования или иных инструментов злоумышленники могут взаимодействовать с веб-сайтами.

**HTML:** HTML (Hypertext Markup Language) является языком разметки для создания структуры веб-страниц. Он использует теги для определения различных элементов на странице, таких как заголовки, параграфы, изображения и ссылки. На базе HTML строится дерево элементов DOM (Document Object Model) которое представляет структуру документа. Он предоставляет программам (например, JavaScript) интерфейс для манипуляции содержимым и структурой веб-страницы.

**CSS**: CSS (Cascading Style Sheets) отвечает за стилизацию и внешний вид веб-страниц. С его помощью можно определить цвета, шрифты, расположение элементов и другие аспекты дизайна. Таблица стилей помогает расширять возможности языка HTML, добавляя различные методы манипуляции с текстовой, графической или видео информацией.

**JavaScript:** JavaScript является языком программирования, который выполняется в браузере пользователя. Он добавляет интерактивность и динамическое поведение на веб-страницах, позволяя создавать анимации, обрабатывать события и взаимодействовать с пользователем.

**MySQL:** MySQL – это свободная реляционная система управления базами данных.

**PHP:** PHP – язык программирования для написания серверной части веб-приложения.

## **1.2 Безопасность веб-ресурсов**

Временами можно услышать о ситуациях, когда веб-сайты становятся недоступными из-за атак, таких как DDoS (атака на отказ в обслуживании), или из-за изменения отображаемой информации на страницах, часто с повреждением контента. В других случаях, миллионы учетных записей, включая пароли, адреса электронной почты и данные кредитных карт, могут становиться общедоступными, подвергая пользователей веб-сайта личному недовольству, чаще к финансовым рискам.

Цель веб-безопасности заключается в предотвращении любых видов атак. Более формальным определением веб-безопасности является – способ защиты веб-сайтов от несанкционированного доступа, использования, изменения, уничтожения или нарушения работы.

Для эффективной безопасности веб-сайта необходимо уделять особое внимание к разработке: веб-приложения, конфигурации веб-сервера, при написании политик создания и обновления паролей, а также кода на стороне клиента. Хотя все это звучит очень зловеще, хорошая новость заключается в том, что если используется веб-фреймворк для серверной части, то он обеспечит «по умолчанию» надёжные и продуманные механизмы защиты от ряда наиболее распространённых атак. Другие атаки можно смягчить с помощью конфигурации персонального веб-сервера, например, включив HTTPS. Наконец, есть общедоступные инструменты для сканирования уязвимостей, которые могут помочь определить, если были допущены какие-либо очевидные ошибки.

## **1.3 Протоколы HTTP и HTTPS**

HTTP – это протокол прикладного уровня модели сетевого взаимодействия Open Systems Interconnection (OSI). Она определяет несколько типов запросов и ответов. Например, если необходимо просмотреть данные с веб-сайта, отправляется запрос HTTP GET. Если необходимо отправить информацию, например, заполнить контактную форму, отправляется запрос HTTP POST.

HTTP передает незашифрованные данные, а это означает, что информация, отправленная из браузера, может быть перехвачена и прочитана третьими лицами. Этот процесс не идеален, поэтому он был расширен до HTTPS, чтобы повысить уровень безопасности взаимодействия. HTTPS объединяет HTTP-запросы и ответы с технологиями SSL и TLS.

Веб-сайты HTTPS должны получить сертификат SSL/TLS от независимого центра сертификации (CA). Веб-ресурсы передают сертификат браузеру, а затем обмениваются данными для установления доверия. Также SSL-сертификат содержит криптографическую информацию, поэтому сервер и веб-браузеры могут обмениваться зашифрованными данными. Процесс работает следующим образом:

1. Пользователь открывает веб-сайт HTTPS, введя формат URL-адреса «https://» в адресной строке браузера;
2. Браузер пытается проверить подлинность сайта, запросив SSL-сертификат сервера;
3. В ответ сервер отправляет сертификат SSL, содержащий открытый ключ;
4. SSL-сертификат веб-сайта подтверждает личность сервера. Как только браузер удовлетворен, используется открытый ключ для шифрования и отправки сообщения, содержащего секретный ключ сеанса;
5. Веб-сервер использует персональный закрытый ключ для расшифровки сообщения и получения ключа сеанса. Затем шифрует сеансовый ключ и отправляет подтверждающее сообщение в браузер;
6. После завершения и браузер, и веб-сервер переходят на использование одинакового сеансового ключа для безопасного обмена сообщениями.

## **1.4 Межсайтовый скриптинг (XSS)**

Межсайтовый скриптинг (XSS или *Cross-Site Scripting* ) это термин, используемый для описания типа атак, которые позволяют злоумышленнику внедрять вредоносный код через веб-сайт в браузеры других пользователей. Поскольку внедрённый код поступает в браузер с сайта, он является доверенным и может выполнять действие отправки авторизационного файла *cookie* пользователя злоумышленнику. Когда у злоумышленника есть файл *cookie*, он имеет возможность войти на сайт, как обычный пользователь, и проделать необходимые операции, например, получить доступ к данным кредитной карты, просмотреть контактные данные или изменить пароли.

Существует множество различных видов XSS атак, рассмотрим некоторые из них:

1. **Отражённые XSS-уязвимости.** Злоумышленник внедряет вредоносный код, обычно в виде скрипта в параметры URL, формы или другие места, где ввод пользователя отображается на странице. Вредоносный код отражается обратно на пользователя при выполнении запроса к серверу. Например, злоумышленник может создать вредоносную ссылку, и если пользователь перейдет по ней, то вредоносный код выполнится в браузере. Атаки этого типа часто используют социальную инженерию для убеждения пользователя в выполнении определенных действий;
2. **Хранимые XSS-уязвимости.** Внедряется вредоносный код, например, в комментарии, формы, или другие места, где данные сохраняются на сервере и впоследствии отображаются для других пользователей. Когда иные пользователи просматривают эти данные, вредоносный код выполняется в браузерах владельцев. Примером может быть атака, при которой злоумышленник добавляет вредоносный скрипт в комментарии на форуме. Когда другие пользователи просматривают комментарий, скрипт выполняется в браузерах, что может привести к утечке конфиденциальной информации;
3. **DOM-based XSS.** Это форма атаки скриптинга, которая происходит на стороне клиента (в браузере пользователя). Вместо того чтобы направленно внедрять вредоносный код на сервере, как это происходит в других типах XSS, злоумышленник эксплуатирует уязвимости в дереве объектной модели документа (DOM) браузера. Простыми словами, когда веб-сайт не безопасно обрабатывает пользовательский ввод и позволяет злоумышленнику вставить вредоносный код в адресную строку или другие элементы страницы, браузер может неправильно интерпретировать этот код. Злоумышленник имеет возможность создать специальные ссылки или ввод, которые при выполнении приводят к изменению содержимого страницы или совершению вредоносных действий в контексте пользователя.

Пример DOM-based XSS может быть реализован, когда сайт использует JavaScript для динамического обновления содержимого страницы без должной фильтрации ввода. Если пользователь вставляет в текст ссылки вредоносный код, который впоследствии выполнится в браузере других пользователей при переходе по этой ссылке, то операция может привести к небезопасному выполнению скрипта на стороне клиента.

Также, рассмотрим некоторые способы предотвращения XSS атак:

* **Экранирование и кодирование вывода данных**: Все данные, вводимые пользователем и отображаемые на веб-страницах, должны быть экранированы или закодированы перед тем, как они будут отображены в HTML, JavaScript или других контекстах. Это поможет предотвратить внедрение вредоносного кода на страницу;
* **Использование HTTP заголовков Content Security Policy (CSP):** CSP позволяет вам указать браузеру, какие источники содержимого разрешены для конкретных ресурсов на странице. Это помогает предотвратить выполнение вредоносного JavaScript-кода, внедряемого через XSS;
* **Валидация и фильтрация ввода данных:** на серверной стороне следует осуществлять строгую валидацию и фильтрацию ввода данных, чтобы исключить возможность внедрения вредоносного кода. Это особенно важно для данных, которые могут быть отображены на веб-странице;
* **Использование HTTPOnly и Secure флагов для куки:** Установка флага HTTPOnly для куки помогает предотвратить доступ к куки через JavaScript, тем самым снижая риск XSS атак. Флаг Secure обеспечивает передачу куки только через защищенное соединение HTTPS;
* **Регулярное обновление и защита браузеров и фреймворков:** Регулярное обновление используемых браузеров и фреймворков важно для получения исправлений уязвимостей, в том числе уязвимостей, связанных с XSS;
* **Мониторинг и журналирование:** Регулярный мониторинг веб-приложений и журналирование активности могут помочь выявить и предотвратить XSS атаки на ранних стадиях.

## **1.5 SQL инъекции**

SQL Injection (SQLI) - это тип XSS атаки на веб-приложения, при которой злоумышленник внедряет вредоносные SQL-запросы в поля ввода или другие механизмы общения с базой данных. Целью SQL Injection является выполнение несанкционированных операций в базе данных или получение конфиденциальной информации. Когда веб-приложение недостаточно проверяет и фильтрует ввод пользователя, злоумышленник может внедрить SQL-код в строку запроса, предназначенную для взаимодействия с базой данных. Если удается совершить операцию, то злоумышленник получает доступ к данным, может изменять содержимое базы данных, а в некоторых случаях даже выполнять удаленные команды на сервере.

Так как SQLI – это один из вариантов XSS атаки, способы предотвращения данной атаки на веб приложение, включают в себя те же самые меры, что и для XSS атак, (сделанных) с помощью скриптов написанных на языке JavaScript. Однако, данный тип атаки имеет свою специфику, поэтому для него стоит выделить отдельные меры защиты, такие как:

* **Применение принципа минимальных привилегий:** Назначение минимально необходимых прав доступа к базе данных для приложения или пользователя также помогает снизить риск SQL инъекций. Если злоумышленнику удается выполнить инъекцию, он будет ограничен в своих возможностях манипуляции данными;
* **Санитизация ввода данных:** Перед выполнением SQL запроса следует проводить проверку и очистку (санитизацию) входных данных. Удаление специальных символов, экранирование кавычек и других метасимволов может предотвратить внедрение зловредного SQL кода;
* **Использование белого списка:** Вместо того, чтобы фильтровать нежелательные символы, лучше создать белый список разрешенных символов и проверять, соответствуют ли входные данные этому списку;
* **Использование средств защиты от SQL инъекций**: Некоторые современные фреймворки и ORM (Object-Relational Mapping) библиотеки предоставляют встроенные механизмы защиты от SQL инъекций. Использование таких средств может существенно снизить риск успешной атаки.

# **ИМИТАЦИЯ XSS АТАК И ТЕСТИРОВАНИЕ СПОСОБОВ ИХ ПРЕДОТВРАЩЕНИЯ**

## **Подготовка тестовой среды**

Для практической работы с уязвимостями было подготовлено все необходимое для реализации веб-приложения. Был выбран хостинг-провайдер под названием “SpaceWeb”, который предоставляет бесплатный сервер для хостинга одного сайта. Данный хостинг также предоставляет базы данных такие как “MySQL” И “PostgreSQL”, которые будут участвовать в практической работе.

С помощью FTP-клиента “FileZilla” было установлено подключение к удаленному FTP серверу, который содержит файлы тестируемого веб-приложения. Все файлы проекта содержатся на сервере.

После вышеописанных действий, сайт готов к работе под доменом выделенном этим же хостингом: “<http://artem39rma.temp.swtest.ru>/”

Для разработки проекта был развернут локальный сервер с помощью программы “Open Server”. Open Server предоставляет доступ к СУБД “MySQL” и “PostgreSQL”, а также к веб-интерфейсам данных СУБД. В данном проекте было принято решение использовать СУБД “MySQL”, а используемый серверный язык программирования “PHP”.

## **Применение XSS и защита от атаки**

Для работы с XSS была разработана веб-страница, которая имитирует привычный нам блок комментариев в интернете. Также была создана база данных под названием “main” в которой, в свою очередь, была создана таблица под названием “comments”. Структура данной таблицы представляет собой:

* **id:** Уникальный идентификатор записи в таблице
* **textData:** Текст комментария введенный пользователем.

На странице присутствует поле ввода, через которое пользователь может сделать запись в базе, а также список комментариев каждый из которых представляет одну запись в базе данных.