**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ РАДИОФИЗИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**

**Кафедра информатики и компьютерных систем**

ВАСИЛЕНКО

Мария Николаевна

**РАЗРАБОТКА WEB-ПРИЛОЖЕНИЯ «ЛИЧНЫЙ КАБИНЕТ ДЛЯ СИСТЕМЫ МОНИТОРИНГА ТРАНСПОРТА С УЧЁТОМ ТРЕБОВАНИЙ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ**

Магистерская диссертация

Научный руководитель:

кандидат технических наук, доцент И. П. Стецко

Допущена к защите

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Зав. кафедрой информатики и компьютерных систем,

доктор технических наук, профессор С. Н. Семенович

Минск, 2024

* 1. **Описание предметной области**

На сегодняшний день в разных областях экономики становятся востребованными системы мониторинга транспорта. Основными пользователями система мониторинга становятся разного рода автотранспортные предприятия (АТП), использующие системы GPS мониторинга для широкого круга задач:

* осуществлять контроль за подвижными системами, что улучшает работу всего предприятия.
* Прокладывать оптимальный маршрут и обеспечивать оптимальное его прохождение.
* Минимизировать расход топлива.
* Повысить дисциплину в коллективе.

Система спутникового мониторинга транспорта – это программно-аппаратный комплекс, состоящий из трех частей:

* устройств мониторинга транспортных средств;
* сервера геоинформационной системы (ГИС);
* клиентских диспетчерских мест.

Система спутникового мониторинга транспорта основана на использовании глобальных навигационных спутниковых систем таких как GPS, GLONASS, Galileo. Работа системы мониторинга транспорта осуществляется по следующей схеме:

* На подвижный состав устанавливается терминал или трекер, который использует спутниковые системы для определения текущего местоположения подвижного состава.
* Полученная телеметрическая информация проходит предварительную обработку, сохраняется в память устройства.
* Посредством сотовых сетей GSM, при наличии связи данные о местоположении и другие данные, например, расход топлива, наличие заправок и сливов, температурный режим и значение других встроенных датчиков.
* Пользователь, имея доступ к серверу, например, используя Интернет соединение, посредством специализированного программного обеспечения, может получить данные с устройства. Схема работы системы мониторинга транспорта показана на рисунке 1.1

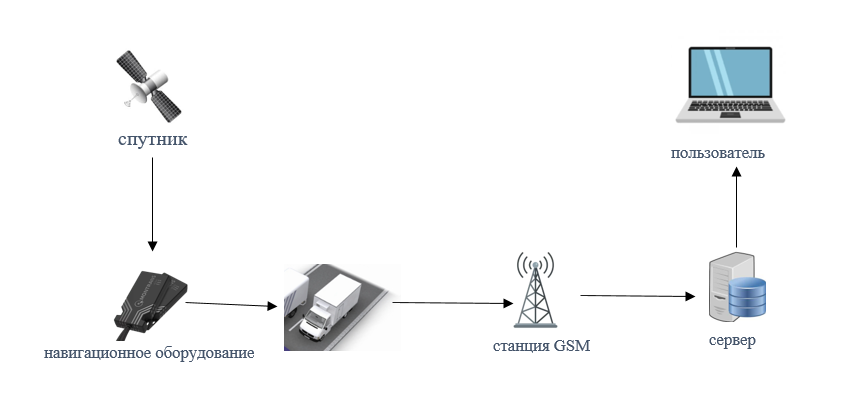
Преимуществами использования систем мониторинга транспорта для предприятий, имеющих автопарк следующие:

Рисунок 1.1 Схема работы системы мониторинга транспорта

* Автоматизация процесса управления автопарком, ускорение процесса транспортной логистики.
* Мониторинг в реальном времени местоположения транспортных средств.
* Дистанционный мониторинг технических показателей транспортных средств.
* Повышение дисциплины в трудовом коллективе.

Компании, осуществляющие мониторинг транспорта, предоставляют своим клиентам различные программные обеспечения, которые позволяют осуществлять следующие функции:

* Определять местоположение подвижного состава в режиме реального времени.
* Предоставлять детальные отчёты о затраченном времени в пути и стоянках.
* Управлять автопарком, осуществлять бронирование транспортного средства, вести контроль водителей.
* Контролировать расход топлива.
* Контролировать документооборот.

Для упрощения работы клиентов и автоматизации ряда процессов разрабатывается специальное приложение – «Личный кабинет».

«Личный кабинет» – персональная страница пользователя сайта, где он может выполнять различные действия и отслеживать связанную с ним и скрытую от посторонних информацию.

«Личный кабинет для системы мониторинга транспорта» - это WEB-приложение, использующиеся для автоматизации процесса управления автопарком, учёта различных рабочих процессов на предприятии, внесении разнообразной информации в базу данных клиентами.

Приложение «Личный кабинет» обладает следующими особенностями:

* Отсутствие необходимости установки приложения на устройстве клиента, за счёт применения WEB-интерфейса.
* Доступ в приложение в любое время, с любого устройства.
* Оперативное осуществление всевозможных операций.
* Снизить нагрузку в работе диспетчеров – многие процессы осуществляются автоматически.
* Удобный интерфейс приложения, позволяет пользователю сократить время на выполнение сложных операций, предоставляя возможность выполнять сложные операции в пару шагов.
* Пользователь может осуществлять быстрый поиск необходимой информации, машин, датчиков, и их привязок.
* Установка связей между различными базами данными и автоматизации процесса их заполнения.
* Уменьшает количество ошибок, так как многие процессы автоматизированы, а данные которые вносит пользователь перед записью в БД проверяются.
* Информация, необходимая пользователю для организации рабочего процесса расположена в одном окне.

При регистрации в системе каждому пользователю назначается роль. Роль – совокупность прав, на основе которых определяются возможные действия пользователя в информационной системе. Роль определяет полномочия пользователя, которыми он обладает при использовании приложения «Личный кабинет». Роль – круг пользователей или сотрудников, которым может быть поручена определенная функция в системе. В приложении пользователю может быть назначена одна из трех ролей:

* Администратор.
* Пользователь.
* Сотрудник.

Роль администратора назначается диспетчерам компании, предоставляющих клиентам услуги мониторинга транспорта. Администратор тесно взаимодействует с клиентами, координирует их действия в процессе взаимодействия с системой. Администратор приложения – привилегированный пользователь, наделённый расширенными полномочиями. Администраторы получают доступ к настройкам клиентских аккаунтов, добавление и изменение данных.

Роль пользователь назначается руководителям компании, пользующимися услугами мониторинга транспорта. Аккаунт пользователя создается администратором. Каждый пользователь может создавать аккаунты для сотрудников, например, для водителей или сотрудников отдела кадров, назначая им роль сотрудника. Пользователи могут контролировать действия сотрудников, добавлять, редактировать информацию, создавать привязки.

**1.2 Анализ угроз безопасности web-приложений**

В процессе использования приложения «личный кабинет» актуальным является процесс сбора и хранения данных клиентов компании, например, информации о составе автопарк, закрепленных датчиков, личные данные клиента. Все большее значение приобретает обеспечение информационной безопасности приложения.

Угрозы информационной безопасности для приложения «личный кабинет» могут быть внешними и внутренними. Источником внешних угроз является человек, осуществляющий несанкционированный доступ к приложению. Проблема внешних угроз безопасности вытекает из особенностей приложения, например, невозможности изолировать приложение от несанкционированного доступа извне так как необходимо обеспечить доступ в сети Интернет. Источником внутренних угроз являются люди, осуществляющий санкционированный доступ к системе (администраторы, пользователи).

Согласно данным исследования проекта OWASP (Open Web Application Security Project) на 2023 существует 10 наиболее опасных угроз безопасности web-приложений:

1. Небезопасный контроль доступа.

2. Криптографические сбои.

3. Инъекции.

4. Небезопасный дизайн.

5. Небезопасная конфигурация.

6. Использование компонентов с известными уязвимостями.

7. Уязвимости идентификации и аутентификации.

8. Нарушение целостности программного обеспечения и данных.

9. Недостаточное журналирование и мониторинг.

10. Подделка запросов со стороны сервера.

Небезопасный контроль доступа – Broken Access Control – нарушение контроль доступа связанно с несанкционированным доступом, в результате чего злоумышленник получает привилегии в системе, которыми он не обладает, что приводит к несанкционированному доступу к информации, модификации, уничтожение данных или другим последствиям.

Криптографические сбои - Сryptographic Failures – неправильное использование криптографии и алгоритмов приводит к криптографическим сбоям, что может привести к раскрытие конфиденциальной информации и личных данных пользователей, например, паролей, адрес электронной почты, номера кредитных кард и др.

Инъекции – Injection – внедрение в запросы к базе данных кода, дополняющего данный запрос. Данный тип уязвимости может привести к краже информации, потере данных, отказу в обслуживании. Использование безопасных API, положительной валидации ввода на стороне сервера может предотвратить данный тип уязвимости.

Небезопасный дизайн - Insecure Design – данная уязвимость связанна с использованием небезопасных паттернов проектирования, незащищенных библиотек, компонентов и методологий. Для предотвращения данной уязвимости необходимо определять уровень безопасности приложения на этапе разработки.

Небезопасная конфигурация – Security Misconfiguration – Данная уязвимость связанна с неправильной настройкой элементов. Причиной неправильной конфигурации системы может стать установка устаревшего программного обеспечения или ненужных функций. Данная уязвимость может быть предотвращена внедрением безопасного процесса установки, использование автоматизированных процессов проверки установок и среды.

Использование компонентов с известными уязвимостями – Using Components with Known Vulnerabilities – уязвимость, связанна с использованием библиотек, Фреймворков и компонентов, содержащих различные уязвимости, в связи с этим необходимо использовать новые версии компонентов.

Уязвимости идентификации и аутентификации – Broken Identification and Authentication – уязвимость, связанна с получением неавторизированного доступа к приложению. Причиной данной уязвимости является недостаточно проработанная система идентификации и валидации пользователя в приложении.

Нарушение целостности программного обеспечения и данных - Software and Data Integrity Failures – нарушение целостности кода приводит к сбою в работе программного обеспечения и целостности данных. Внедрение средств защиты программного обеспечения, процесса проверки изменений кода и конфигурации – способы предотвращения данной уязвимости.

Недостаточное журналирование и мониторинг – Insufficient Logging and Monitoring – использование средств мониторинга и журналирования позволяют предотвратить утечку информации и своевременно обнаружить несанкционированный доступ.

Подделка запросов со стороны сервера - Server-Side Request Forgery – уязвимость связанна с получением поддельных запросов со стороны сервера. Данная уязвимость возникает, когда отсутствует проверка URL удаленного ресурса на стороне веб-приложения. Использование проверки вводимых данных, каждой схемы URL со списком разрешенных, позволяет предотвратить данный тип уязвимости [1].

Классификация угроз проекта WASC указывает на наиболее распространенные проблемы безопасности прикладного уровня. Согласно классификации, WASC существует 6 основных атак на web-приложения:

1. Аутентификация – класс атак, направленный на проверку идентификации пользователя. Выделяют следующие типы атак:
2. Подбор пароля. Решение:

– Отображение минимум информации об ошибке при неправильно введенном логине или пароле.

– Введение CAPTCHA после нескольких неудачных попыток входа подряд у одного пользователя.

– Проверка сложности пароля пользователя с целью предотвращения использования простых паролей [6].

1. Атака через уязвимости в недостаточной аутентификации. Решение

– Использовать аутентификацию на уровне протокола TLS, что потребует от злоумышленника доступ к компьютеру с сертификатом.

– Предоставление доступа к ресурсам административной части после аутентификации через логин и пароль [6].

1. Атака через уязвимость в небезопасном восстановлении пароля. Решение:

– При восстановлении пароля требовать старый пароль.

–Реализовать функционал восстановления пароля в автоматизированном режиме с проверкой запроса восстановления пароля администратором [6].

1. Авторизация – класс атак, направленный на определение полномочий пользователя в системе. Выделяют следующие типы атак:
2. Атаки через уязвимость в недостаточной авторизации. Решение

– На уровне сервера БД реализовать таблицы: ролей пользовательских действий, истории действий.

– При каждом запросе к системе осуществлять проверку прав доступа к системе [6].

1. Атака по отсутствию таймаута сессии. Решение [6]

– Установить срок действия сессии.

1. Фиксация сессии. Решение

– После успешного входа создавать новый идентификатор сессии.

– При закрытии вкладки браузера закрывать сессию.

– При смене IP адреса закрывать сессию.

1. Атаки на стороне клиента – класс атак, связанных с злоупотреблением пользователем полномочий на веб-сайте. Выделяют следующие атаки:
2. Подделка межсайтовых запросов. Решение

– Для запросов, которые каким-либо образом изменяют данные в БД, передавать вместе с запросом токен, находящейся на HTML странице, с которой совершается запрос.

1. XSS инъекции. Решение

– Фильтрация пользовательского ввода, в том числе с применением библиотек, распознающих вредоносный код в различных кодировках.

– Экранирование специальных символов в приложении, которые могли бы попасть в него через отправку пользовательских данных.

1. Выполнение кода – класс атак, направленных на выполнение команд путем внедрения злоумышленником вредоносного кода на сервер.
2. Выполнение команд ОС. Решение

– Использование методов, которые передают параметры командной строки безопасно.

1. SQL инъекции. Решение

– Безопасная передача параметров в массиве при любом использовании SQL.

1. Внедрение серверных расширений. Решение

– Запрет на использовании функции выполнения пользовательского кода на сервере [6].

1. Разглашение данных – класс атак, направленный на получение информации об архитектуре приложения, для дальнейшего использования уязвимостей для реализации атак.
2. Логические атаки – класс атак, направленный на злоупотребление функциональными возможностями приложения [2].

Использование рекомендаций, разработанных в рамках проекта OWASP, позволяют повысить уровень безопасности приложения и снизить риски угроз. Проектом OWASP были разработаны следующие рекомендации:

1. Следить за обновлениями, используемых компонентов, библиотек, Фреймворков.
2. Для передачи данных между клиентом и сервером использовать протоколы с шифрованием.
3. Регулярно проводить обновления.
4. Производить тестирование приложения, направленное на поиск уязвимостей.
5. Производить мониторинг и журналирование системы.

Атаки на веб-приложения злоумышленниками организуются с целью получения несанкционированного доступа к системе, модификации данных и утечки информации. Анализируя список атак на веб-приложение, представленных, открытым проектом OWASP, большинство атак возможно предотвратить, продумав логику веб-приложения и среду выполнения.

**1.3 Методы и средства обеспечения безопасности веб-приложений**

Для обеспечения информационной безопасности веб-приложений необходимо провести анализ основных уязвимостей приложения и обеспечить комплексную защиту, позволяющую учитывать каждую уязвимость. Методы применяемые для обеспечения безопасности веб-приложений должны гарантировать конфиденциальность и целостность данных пользователей этих приложений при передаче, обработке и хранении данных. Это подразумевает определение рисков, установление приемлемых уровней рисков, определение моделей угроз.

На этапе разработке следует в первую очередь устранить наиболее важные риски. Когда речь идет о критически важных веб-приложениях безопасность на уровне приложения является ключевым моментом. При моделировании возможных угроз становится очевидным более точное представление о недостатках системы безопасности. Моделирование угроз начинается с умением правильно определить угрозу. После этого необходимо понять угрозу и построить несколько возможных случаев, которые могли привести к возникновению угрозы. Разрабатываются решения направленные на устранение причины обнаруженной угрозы [5].

Для веб-приложений выделяют следующие категории угроз:

1. Случайные ошибки.
2. Автоматизированное вредоносное ПО, которое может нанести вред приложению.
3. Любопытный злоумышленник, который пытается выполнить действия, которые приложение выполнять не должно.
4. Мотивированный злоумышленник.

Немаловажным инструментом обеспечения безопасности приложения является проведение регулярной проверки на устойчивость существующим угрозам. Рекомендуется периодически проводить тесты, при которых моделируются атаки с использованием различных инструментов. Сбои, произошедшие при выполнении тестов, позволяют выявить потенциальные уязвимости в работе приложения [3].

Решение задачи по обеспечению безопасности web-приложения подразумевает учёт специфики функционирование приложения. Среди методов по обеспечению информационной безопасности веб-приложения могут быть выделены следующие:

1. Тестирование и аудит приложения на уязвимости.
2. Использование криптографических методов и защищенных протоколов для передачи данных.
3. Использование организационных мер.
4. Производить обновление библиотек и программных компонентов.
5. Применять программное обеспечение, обеспечивающего безопасность.

Тестирование и аудит приложения на уязвимости

На этапе разработки web-приложения производится определение возможных угроз и моделирование их реализации. При эксплуатации приложения производится проверка на уязвимости с использованием ручных или автоматизированных методов проверки. Проблемы безопасности должны быть обнаружены на ранней стадии внутри организации, а не извне от внешних злоумышленников.

Для осуществления автоматизированной проверки приложения могут быть использованы следующие инструменты [4]:

1. Approof by Positive Technologies – изучает конфигурацию приложения и находит лишний или вредоносный код.
2. WebCruiser Web Vulnerability Scanner – веб-сканер широкого спектра уязвимостей с графическим интерфейсом.
3. Snyk – сканирует JavaScripr, Ruby, Java-код на наличие уязвимостей.
4. OWASP Xenotics XSS Exploit Framework – проверяет веб-приложение на наличие XSS-уязвимости.
5. SQLmap – сканер, который тестирует приложение на возможность реализации SQL-инъекций.
6. Metasploit – сканирует приложение на SQL-инъекции, XSS-уязвимости.
7. SSL Server Test – сканирует на наличие SSL-уязвимостей.

Использование криптографических методов и HTTPS протоколов для передачи данных

Защита данных при передаче между клиентом и сервером является одной из важнейших задач для обеспечения информационной безопасности приложения. Использование недостаточных средств защиты становится причиной нарушения конфиденциальности, потери персональных данных и коммерческой информации.

Конфиденциальность обеспечивает недоступность информации для неавторизированных пользователей или пользователей не имеющих соответствующих привилегий доступа. Использование криптографических методов шифрования перед передаче конфиденциальной информации, например, паролей, позволяет повысить безопасность.

Использование защищенного протокола при передачи данных HTTPS защищает персональные данные пользователя путём шифрования трафика. При передачи персональных данных, информацию, содержащую коммерческую тайну, номера кредитных карт, необходимо использовать HTTPS протокол.

Обновление программного обеспечения, библиотек, компонентов.

Использование устаревшего программного обеспечения существенно снижает безопасность приложения. Уязвимости, существующие в старых версиях ПО, становятся уловками для хакеров.

За обновлениями библиотек, Фреймворков, компонентов и модулей также необходимо следить во избежание проблем безопасности приложения.

Организационные меры

Обучение клиентов приложения является важнейшим пунктом в обеспечении информационной безопасности. Пользователи приложения должны обладать информацией о требованиях к безопасности приложения, которое они собираются использовать. Организация ложна определять политику конфиденциальности и защиты данных. Иногда, полезным является принимать в команду консультантов по информационной безопасности, чтобы можно было быстро решить конкретные вопросы и проблемы.

Для веб-приложения должны быть определены требования к безопасности. Определение требований к безопасности может быть непросто, и это необходимо сделать правильно, иначе они будут бессмысленны. Требования могут начинаться с общего набора, например, следующего:

1. Включить все механизмы защиты.
2. Устранить все распространенные уязвимости.
3. Учесть все требования (нормативные акты, стандарты, передовой опыт).
4. Определить работу аутентификации.
5. Детализировать матрицу управления доступом (роли, активы, функции, разрешения).
6. Определить правила проверки ввода.
7. Определить подход, который будет использоваться при обработке ошибок и способ протоколирования.

Правильно разработанная архитектура веб-приложения является важной частью в обеспечении безопасности приложения. Цель состоит в гарантировании в том, что определённые части веб-приложения не могут получить доступ к конфиденциальной информации. Это часть связано с информацией баз данных и объектами, к которым осуществляется доступ и манипуляции, когда они не должны этого делать [5].

**ГЛАВА 2 АРХИТЕКТУРА WEB-ПРИЛОЖЕНИЯ И ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ**

**2.1 Архитектурные подходы при построении веб-приложений**

Архитектура веб-приложений – совокупность различных решений, определяющих взаимодействие между частями приложения, серверами, промежуточными программными обеспечениями и базами данных. Правильно выбранная и разработанная архитектура определяет быстроту реализации проекта, возможность масштабирования, сложность в поддержки приложения, эффективность работы и длительность загрузки страницы. Большое внимание необходимо уделить разработки архитектуры. Выбор наилучшего способа разбиения приложения на части, выбора способа взаимодействия частей друг с другом и передачи информации между частями на этапе проектирования разрабатываемого веб-приложения позволяет снизить сбои приложения и повысить эффективность работы. При выборе архитектурного подхода разработчику необходимо учитывать ряд факторов: распределение нагрузки между клиентской и серверной частью, функционал приложения, бизнес-логику и прочее [9].

Веб-приложения строится в соответствии с классической трёхуровневой модели и состоят из следующих частей:

1. Пользовательский интерфейс, выполняемы в браузере на странице пользователя.

2. Сервер, обрабатывает запросы со стороны клиента

3. база данных.

Приложение на стороне сервера представляет собой монолит, выполняемый как одно целое.

Архитектура одностраничных приложений –SPA – приложения, в которых все функции хранятся на одной странице. SPA не содержат загрузку страницы с сервера, страница генерируется внутри HTML-файла.

**сервер**

**клиент**

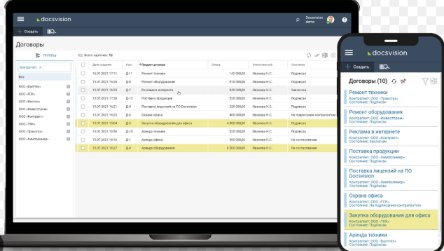


Рисунок 2.1Архитектура одностраничного приложения

ответ

запрос

На сервере хранится вся логика приложения. При взаимодействии пользователя со страницей на сервер отправляется запрос при этом страница не перегружается. Взаимодействие клиента и сервера происходит в фоновом режиме. Данные приложения хранятся на сервере. Интерфейс приложения разрабатывается на клиентской части.

Для распределения нагрузки сервера и масштабирования приложения одним из подходов является использование микросервисной архитектуры. В разработке веб-приложений доминируют два подхода – монолитная и микросервисная архитектура. Архитектура микросервисов декомпозирует сервер на небольшие, автономные, слабосвязанные и независимо развертываемые сервера. Каждый микросервис содержит свои собственные функции, бизнес-логику, функции взаимодействия с клиентом. Каждый микросервис может включать собственную базу данных или взаимодействовать с общей базой данной. В подходе микросервисной архитектуры при организации взаимодействия предпочтение отдается таким протоколом как HTTP, REST, JMS, AMQP. Приложения, основанные на микро сервисной архитектуре, хорошо масштабируются по горизонтали, как в технической структуре, так и в организационном плане. Преимуществом микросервисной архитектуры является повышение отказоустойчивости всего приложения, так как отказ одного компонента необязательно приводит к отказу всего приложения.

Однако, использование микросервисной архитектуры имеет ряд недостатков:

* определение оптимальных границ микросервисов
* сложность интеграции компонентов
* согласование данных и управление транзакциями между серверами
* сложность целостного понимания системы
* увеличение вычислительных ресурсов [10]

Таким образом использование микросервисного архитектурного подхода имеет как преимущества, так и недостатки. В зависимости от разрабатываемого приложения, его масштаба и функционала, команда разработчиков выбирает подходящий архитектурный подход. Так, например, концепция микросервисного подхода предполагает изоляцию данных, что создает необходимость внедрения нескольких независимых баз данных, что увеличивает сложность в распределении приложение. Правильное масштабирование приложения на основе микросервисов может покрыть расходы, затраченные на разработку и поддержания функционала автономных компонентов.

Увеличение функционала приложения, добавление новых информации, отображающейся пользователю, приводит к росту размера и сложности фронтенд части веб-приложения. Расширением микросервисного подхода, используемого в бэкенд части (сервер приложения), является концепция микрофронтендов. Основная идея использования подхода микрофронтендов состоит в разделении одного большого монолитного приложения на несколько автономных компонентов, что в свою очередь позволяет повысить производительность приложения. Компоненты могут быть написаны с использованием различных технологий, использованием разных библиотек, что позволяет не нагружать различные страницы и тем самым повысить скорость загрузки страниц и отказоустойчивость всего приложения.

**2.1 Способы предоставления доступа**

Для входа пользователя в приложение и получения доступа к выполнению определённых действий, ему необходимо выполнить ряд взаимосвязанных процедур: идентификацию, аутентификацию и авторизацию.

**Идентификация** – процедура распознавания субъекта. При входе пользователя на первую страницу приложение – страницу регистрации, пользователь предъявляет системе свой идентификатор (логин, имя, почта). Система проверяет его по своей базе данных. Если такой субъект существует, то для системы он считается легальным, все остальные нелегальными.

**Аутентификация** – это процедура проверки подлинности субъекта, которая позволяет достоверно убедиться в том, что субъект, предъявивший свой идентификатор, на самом деле является именно тем субъектом, идентификатор которого он использует. Для этого пользователю необходимо предоставить система информацию, которая может быть известно только ему одному (пароль, секретный вопрос, отпечаток пальцев и т. д.).

**Авторизация** – процедура предоставления субъекту определённых прав доступа к ресурсам системы после прохождения их процедуры аутентификации. Для каждого пользователя определяется ряд прав, которые он может осуществлять при обращении к ресурсам системы.

При обеспечении доступа пользователя к системе аутентификация является основой для обеспечения безопасности в информационных системах. В процессе развития технологий и выявления новых угроз становится острая необходимость в более тщательной защите информационной системе. Для аутентификации пользователя изначально использовалась однофакторная аутентификация, которая имеет стандартное описание. Пользователь отправляет идентификатор в систему, на основе которого в системе вычисляется ключ. При условии совпадения ключа пользователя и системы, пользователь получает доступ к информационным ресурсам.

Существует большое количество способов аутентификации пользователя в информационной системе: аутентификация на основе паролей, аутентификация на основе биометрических данных, использование сим-карт.

Однофакторная аутентификация не является безопасным способом, так как недостаточно сложный пароль может быть легко подобран. При непродуманной разработке, приложение может иметь множество уязвимых мест, упрощающих несанкционированный доступ к личным данным пользователя.

Ряд уязвимостей, которые могут привести к взлому приложения:

1. при хранении паролей в базе данных приложение не использует хэш-функции и пароли хранятся в открытом виде;
2. между клиентом и сервером пароли пересылаются в незашифрованном виде по незащищённому соединению;
3. при регистрации пользователя в системе приложение самостоятельно генерирует пароль и отправляет пользователю на почту, а при первом входе пользователя в систему приложение не требует от пользователя сменить пароль;
4. приложение не проверяет создаваемые пользователем пароли на сложность, в результате чего они могут быть подобранны злоумышленниками;
5. при совершении важных операций приложение не требует от пользователя пройти повторную аутентификацию;
6. Web-приложение создаёт долгосрочные токены (день, неделю) и не уничтожает их при долгой неактивности пользователя;
7. приложение создаёт токены, которые легко могут быть воспроизведены.

Таким образом, однофакторная аутентификация с использованием паролей при правильной организации является относительно безопасным способом для тех приложений, где не требуется обеспечить повышенную безопасность в хранениях данных пользователя.

Для повышения безопасности при предоставлении доступа к информационному ресурсу используется многофакторная аутентификация. Многофакторная аутентификация – аутентификация, при выполнении которой используется не менее двух различных факторов аутентификации [7]. Целью использования многофакторной аутентификации является создание многоуровневой зашиты, которая препятствует несанкционированный доступ к системе. Многофакторная аутентификация используется в системах повышенной защищенности, содержащих информацию о клиентах или банковских транзакциях.

Факторы аутентификации – категории данных, используемых для проверки подлинности. Каждый новый фактор повышает уверенность в подлинности идентификатора объекта, запрашиваемого доступ к системе. Выделяют три основных фактора для многофакторной аутентификации: фактор знания, фактор владения, фактор свойства [8].

Фактор знания – информация, которой владеет пользователь и которая храниться в базе данных информационной системы. Примером такой информации могут быть пароли, кодовое слово на секретный вопрос, коды [8].

Фактор владения – предметы, принадлежащие пользователю системы. К таким предметам можно отнести смс, получаемые по мобильному телефону, смарт-карты для генерации одноразовых паролей, токены [8].

Фактор свойства – биометрические данные, например, отпечатки пальцев, рисунки радужной оболочки глаз, очертание лица, тембр голоса), модель поведения [8].

Помимо основных факторов могут использоваться дополнительные. Примером дополнительных факторов являются факторы, основанные на местоположении и времени с использованием метаданных, параметров сети, GPS-данных.

Фактор на основе рисков – сочетание адаптивной аутентификации и алгоритмов вычисления рисков. Это аутентификация предназначена для уменьшения количества избыточных входов в систему [8].

По способу построения могут быть выделены следующие схемы аутентификации: последовательные, параллельные и комбинированные.

Последовательная схема – схема, в которой предоставление и проверка данных происходит последовательно. Например, пользователь предоставляет системе фактор знания(пароль), после успешной проверки, система запрашивает у пользователя следующий фактор. Порядок предоставление факторов сохраняется системой.

Примером, параллельной схемы аутентификации являются система, в которых пользователю необходимо одновременно предоставить два фактора. При этом на сервер для проверки данные отправляются одним сообщением.

В комбинированных схемах аутентификации одновременно используются параллельный и последовательный метод.

Рассмотрим пример организации двухфакторной аутентификации с использованием фактора знания и свойства. Система запрашивает у пользователя пароль. После проверки, пользователю необходимо предоставить фактор свойства – отпечаток пальцев. После успешной проверки пользователю выдается сессионный токен и пользователь считается авторизированным.

Для пользователей, устройства которых не имеет датчика для считывания отпечатков пальцев, вместо фактора свойства, может использоваться фактор владения – одноразовый пароль, генерируемый на сервере и отправляемы пользователю по смс или на электронную почту.

При выборе факторов, используемых для реализации многофакторной аутентификации необходимо учитывать не только уровни риска, но и расходы, затрачиваемые на реализацию данного фактора. В таблице представлены типы аутентификации, сложность реализации и уровень защиты.

|  |  |  |
| --- | --- | --- |
| Типа аутентификации | Сложность реализации | Уровень защиты |
| Долговременный пароль | легкая | низкий |
| Долговременный и одноразовый пароль | средний | средний |
| Долговременный пароль и биометрия | сложный | высокий |

**Глава 3. Политика безопасности для приложения «Личный кабинет» для системы мониторинга транспорта**

1. Краткое описание разрабатываемого продукта (информационная или иная система, программное обеспечение и др.), пользователи, основные функции, информационные ресурсы и информационные потоки.

Разрабатываемый продукт представляет программное обеспечение «Личный кабинет» для системы мониторинга транспорта. Приложение предназначено для удаленного управления рабочими процессами, обеспечения сбора данных, онлайн-мониторинга, визуализации рабочих характеристик.

Разрабатываемая система имеет клиент-серверную архитектуру. Клиентская часть – пользовательский веб-интерфейс в браузере. На сервере размещается основная логика и реализуется функции для управления данными. На сервере формируются запросы к базам данных, в которых хранится информация о пользователях системы, данных автомобильных устройств, принадлежащих пользователям, данные мобильных устройств.

Для работы в системе пользователю необходимо авторизоваться в системе. На странице входа пользователь вводит логин и пароль. На сервере происходит проверка корректности введённых данных и если они верны, то для пользователя генерируется одноразовый пароль, который отправляется пользователю на электронную почту, после введения одноразового пароля пользователю предоставляется доступ к системе. При входе в систему на сервере для каждого пользователя формируется session token. При следующих запросах на сервер в заголовке запроса отправляется token, который позволяет определить авторизирован ли пользователь в системе.

В процессе регистрации для каждого пользователя в приложении устанавливается роль. В зависимости от роли, которую имеет конкретный пользователь для него будут доступны определённые функции. В системе определены следующие роли:

* Администратор;
* Клиент;
* Рабочий.

В системе для администратора доступны следующие функции:

* Добавлять новых пользователей в систему, редактировать личные данные пользователя, изменять роль пользователя, просматривать списки существующих пользователей.
* Просматривать список всех транспортных средств, принадлежащих клиентам, удалять, редактировать параметры.
* Просматривать список всех датчиков, закреплять датчики за определенными транспортными средствами, редактировать параметры и удалять.
* Просматривать заявки клиентов на установки и обслуживание датчиков.
* Просматривать отчёты.

Для клиента в системе доступны функции:

* Просмотр списка закреплённых рабочих, редактирование данных рабочих. Добавлять и удалять рабочих. Не имеет право просматривать личную информацию других клиентов, добавлять и удалять клиентов;
* Просматривать списки транспортных средств, принадлежащих данному клиенту, списки датчиков.
* Добавлять, редактировать и удалять транспортные средства.
* Оформлять заявки на обслуживание и прекращения обслуживания датчиков.
* Просматривать отчёты за определённый период времени.

Для рабочего в системе доступны функции:

* Редактирование своих данных;
* Просматривать транспортные средства, закрепленные за данным рабочим, редактировать параметры;
* Просматривать список датчиков, изменять зависимость датчик-транспортное средство;
* Просматривать отчёты.

В системе содержатся данные о пользователях системы такие как логин, имя и фамилия пользователя, e-mail, роль. Технические характеристики транспортных средств, принадлежащих клиентам и закрепленными за рабочими. Технические характеристики датчиков, выдаваемых клиентом в использование и закрепленных за определёнными рабочими. Сохраняются отчеты о использовании датчиков за определённые промежутки времени. Пользователь системы имеет право загружать данные транспортных средств и датчиков, закреплять датчики за определёнными транспортными средствами, подавать заявки на техническое обслуживание, формировать отчёты.

2. Классификация информации, хранящейся и обрабатываемой в системе, отнесение информационной системы (подсистемы) к классу типовых информационных систем в порядке, установленном СТБ 34.101.30-2017 и Приказом ОАЦ при Президенте Республики Беларусь № 195.

Информация, обрабатываемая в системе, описывает технические характеристики продукта, данные клиентов и их имущество. Данная информация является коммерческой тайной. Информация в системе передаётся по открытым каналам передачи данных, так как пользователи должны получать доступ к системе из любой точки посредством Интернет соединения. Соответственно описываемая система относится к классу 3-юл.

3. Анализ угроз (источник угрозы, предполагаемый способ реализации угрозы, уязвимости, которые являются предпосылкой для реализации угрозы, активы, которые являются целью нападения, нарушаемые свойства безопасности активов, возможные последствия реализации угрозы), модели нарушителей.

Источниками угроз системы могут быть:

-люди (внешние нарушители и пользователи системы);

-несанкционированное программное-обеспечение;

-природные явления и аварийные ситуации;

Предполагаемым способом реализации угроз является: несанкционированный доступ к системе путём использования уязвимостей аутентификации и программного обеспечения, перехват информации, передаваемой по незащищенным каналам связи, перехват информации с использованием электромагнитных устройств, физический доступ к рабочему месту пользователя системы, нарушение служебных полномочий пользователей системы.

Уязвимости, которые являются предпосылкой для реализации угроз:

-недостатки механизмов мониторинга и резервного копирования событий информационной безопасности;

-недостатки механизмов идентификации и аутентификации системы;

-ошибки при разграничении прав доступа;

-уязвимости программного обеспечения и программного кода.

Активами, которые являются целью нападения является коммерческая тайна, данные транспортных средств и установленных на транспортных средствах.

Последствиями реализации угроз является материальный ущерб.

Угрозы информационной безопасности классифицируются по следующим критериям:

-по воздействия на базовые свойства защищаемой информации;

-по источнику возникновения;

-по типу источника;

-по способу воздействия на систему.

По воздействиям на базовые свойства защищаемой информации выделяют три вида угроз:

-угроза целостности;

-угроза конфиденциальности;

-угроза доступности.

По типу воздействия угрозы могут быть разделены на случайные и преднамеренные. Примерами случайных воздействий могут быть: природные стихии, аварийные ситуации, сбои или поломка аппаратуры, действия пользователей системы. Преднамеренные угрозы имеют целенаправленный характер и могут осуществляться как внешними нарушителями, так и пользователями системы. Примерами таких угроз могут быть, несанкционированный доступ к системе, установка несанкционированного программного обеспечения, нарушение полномочий пользователей системы.

Угроза нарушения целостности направлена на нарушение качества информации или её полного уничтожения путём искажения и незаконной модификации информации. Источником реализации данной угрозы могут стать: пользователи системы, сбои в работе программного обеспечения, вредоносные программы. Уязвимости системы, которые могут являются предпосылками для реализации данной угрозы: слабая физическая и программная защищенность рабочих мест сотрудников, ошибки в конфигурации системы (неправильное разделение прав доступа), передача информации в незашифрованном виде по незащищенным каналам связи, отсутствие механизмов мониторинга активностей, возможность осуществления непосредственного доступа к пакетам, передающимся по сети.

Угроза нарушения конфиденциальности системы направлена на получения несанкционированного доступа к объектам лицам, которые не имеют на это непосредственных прав. Источниками реализации данной угрозы являются и пользователи системы. Уязвимости системы, которые могут являться предпосылками для реализации данной угрозы: слабые механизмы аутентификации, уязвимости в разделении полномочий пользователей системы, отсутствие регистрации и мониторинга информации о событиях информационной безопасности, отсутствие защиты от физического доступа к рабочим местам пользователям и программного обеспечения системы.

Угроза нарушения доступности реализуется путём создание таких ситуации, когда запрещается легальный доступ к ресурсам системы. Источниками реализации данной угрозы могут быть: несанкционированные устройства и программные обеспечения, аварийные ситуации, природные явления, пользователи системы. Уязвимости системы, которые являются предпосылками для реализации данной угрозы: невнимательность пользователей, отсутствие мониторинга активностей, недостатки программного кода.

Угроза – внешний нарушитель может предпринять попытки несанкционированного доступа к системе.

Источник реализации угрозы – внешние нарушители.

Способ реализации угрозы – получение доступа к системе путем перебора вводимых данных, при наличии уязвимостей механизмов аутентификации системы, перехват передачи незашифрованных данных.

Виды активов потенциально подверженных угрозе – информация о клиентах, транспортных средствах и датчиках.

Нарушаемые свойства безопасности – конфиденциальность, целостность.

Угроза – нарушение работы системы при неправильном использовании.

Источники реализации угрозы – пользователи системы.

Способы реализации угрозы – запрещённая деятельность пользователей системы.

Уязвимости - несовершенство программного кода.

Нарушаемое свойство безопасности – доступность.

Угроза – пользователи могут получить несанкционированный доступ к функциям и данным путём превышения полномочий.

Источник реализации угрозы – пользователи системы.

Способы реализации угроза – превышение полномочий пользователей.

Уязвимости – неправильное разграничение полномочий пользователей.

Нарушаемые свойства безопасности – конфиденциальность информации.

Угроза – нарушитель может воспользоваться открытой сессией с целью модификации информации.

Способы реализации – доступ к рабочему месту пользователя системы.

Уязвимости - долгие сессии, отсутствие автоматической прекращении сессии пользователя при долгих не отправках запроса на сервер.

Нарушаемые свойства безопасности – конфиденциальность, целостность.

**Модели нарушителей системы**.

В зависимости от наличия прав доступа к системе нарушители могут быть внешними и внутренними. Примеры внешних нарушителей: неустановленное физическое лицо, бывший пользователь системы, конкуренты, преступные группы (хакеры). К внутренним нарушителям системы относятся пользователи системы, осуществляющие действия, выходящие за рамки их полномочий.

Неустановленное физическое лицо – внешний нарушитель с низким потенциалом. Возможная мотивация: нанесения ущербы, любопытство, выявление уязвимости системы. Возможности: осуществления несанкционированного доступа, осуществления атаки с целью выведение системы из строя, попытка получения информации от пользователей системы.

Бывшие пользователи – внешний нарушитель с низким потенциалом. Мотивация – месть, нанесение ущерба. Возможности – осуществление атак, с целью выведения системы из строя, знания об уязвимостях системы.

Конкуренты – внешние нарушители со средним потенциалом. Мотивация – нанесения ущерба, выведывание коммерческих тайн. Возможности – осуществление атак на систему, получение информации об уязвимостях системы, возможность получить информацию о структурно-функциональных характеристиках системы.

Преступные группы – внешние нарушители со средним потенциалом. Мотивация – нанесение ущерба, выявление уязвимостей системы с целью получения выгоды. Возможности – самостоятельно осуществить атаку на систему и программное обеспечения, возможность получить информацию об уязвимостях, возможность получить информацию об архитектуре программного обеспечения, возможность перехвата информации по каналу связи.

Пользователи системы – внутренние нарушители с низким потенциалом. Мотивация – любопытство, неквалифицированные действия, месть. Возможности – получить информацию об уязвимостях системы, о методах реализации безопасности информации, получение доступа к закрытым компонентам системы.

4. Основные требования к системе защиты информации. Предполагаемые способы их реализации.

Основными требованиями к системе защите информации является необходимость обеспечения целостности, конфиденциальности и доступности информации.

Обеспечение целостности и конфиденциальности информации реализуется с помощью:

-обеспечение идентификации и аутентификации пользователей системы;

-разграничением прав доступа пользователей к информации;

-использовании методов шифрования при передаче информации по каналам;

-резервное копирование информации;

-осуществление мониторинга событий системы.

Обеспечение доступности пользователей к ресурсам системы осуществляется с помощью:

-резервирование программных и аппаратных средств системы;

-обеспечение защиты от несанкционированного доступа к физическим серверам системы;

-обеспечение устойчивости сервисов системы к атакам.

Для обеспечения безопасности системы необходимо предпринимать организационно-правовые и организационно-технические меры. Исключить несанкционированный доступ к серверным помещениям, работы в серверных помещениях должна контролироваться сотрудниками безопасности. Пользователи системы должны использовать антивирусную защиту на рабочих местах. Несанкционированная деятельность пользователей должна обнаруживаться и немедленно пресекаться.

5. Требования к разработке (доработке) политик безопасности, используемых в эксплуатирующей систему организации.

Предприятие является собственником информационных ресурсов системы.

Доступ к информационным ресурсам системы регламентируется исходя из должностных обязанностей, которые определяются руководителями организации.

Доступ внешнего пользователя к системе осуществляется по предварительной аутентификации по результатам которой определяется решение о разрешении или запрете доступа пользователя к информационным ресурсам системы.

Данными на основании которых пользователю предоставляется доступ к системе являются идентификатор и пароль.

Первоначальное значение пароля учётной записи пользователя устанавливается Администраторами системы при добавлении пользователя.

После первого входа и при дальнейшем использовании пароли выбираются пользователя самостоятельно.

Каждые 6 месяца пользователи должны изменять пароль. Старый и новый пароль не должны повторяться.

При создании собственных паролей пользователям необходимо соблюдать следующие требования:

* Пароль должен содержать минимум 8 символов.
* Пароль обязательно должен содержать следующие символы:

1. буквы нижнего регистра;
2. буквы верхнего регистра;
3. цифры.

* Пароли не должны содержать легко вычисляемые сочетания.

После 10 неудачных попыток введения пароля учётная запись пользователя должна блокироваться и восстанавливается только Администратором.

Пользователям системы запрещается говорить другим свой пароль, хранить пароли в записных книжках, сохранять пароли в браузере.

Пользователи системы должны проходить обучение по эксплуатации системы.

Пользователи несут ответственность за безопасность действий, выполняемых ими при эксплуатации системы.

Ведётся аудит информационного обмена системы и действий пользователей.

Нарушение правил эксплуатации системы должно контролироваться Администратором.

При нарушении правил безопасности эксплуатации системы в отношении сотрудников должны применяться дисциплинарные меры.

При работе с системой допускается использование только лицензионного программного обеспечения, утверждённого в перечне разрешённого программного обеспечения.

Для защиты от вредоносного программного обеспечения необходимо использовать соответствующие антивирусные программы.

6. Иные вопросы обеспечения и проверки информационной безопасности проектируемой системы.

**Глава 4. Практическая часть. Разработка web-приложения «Личный кабинет»**

**4.1 Способ предоставления доступа посредством авторизации пользователя в приложении «Личный кабинет»**

Для получения доступа к информации и функциям разработанного приложения пользователю необходимо пройти этап авторизации.

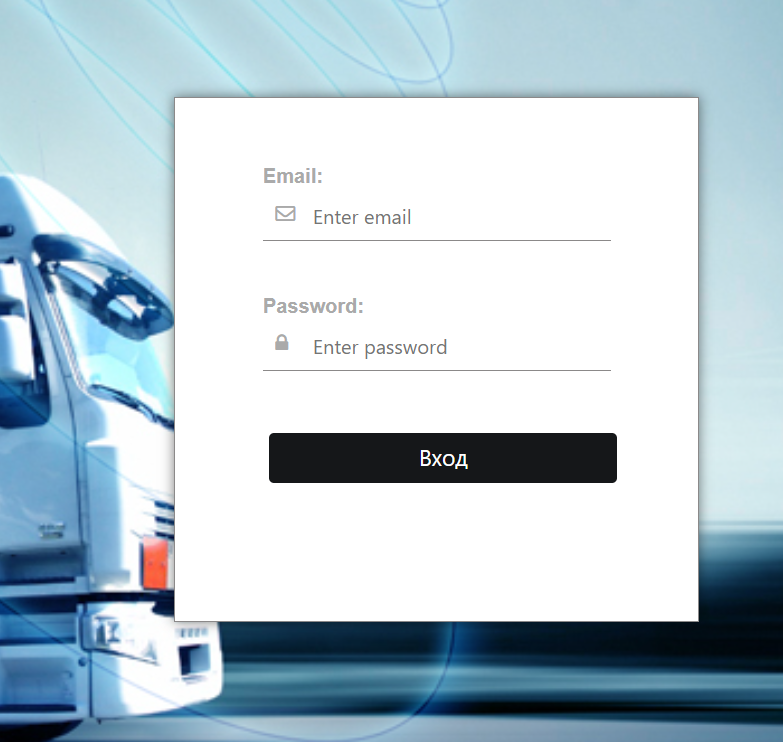
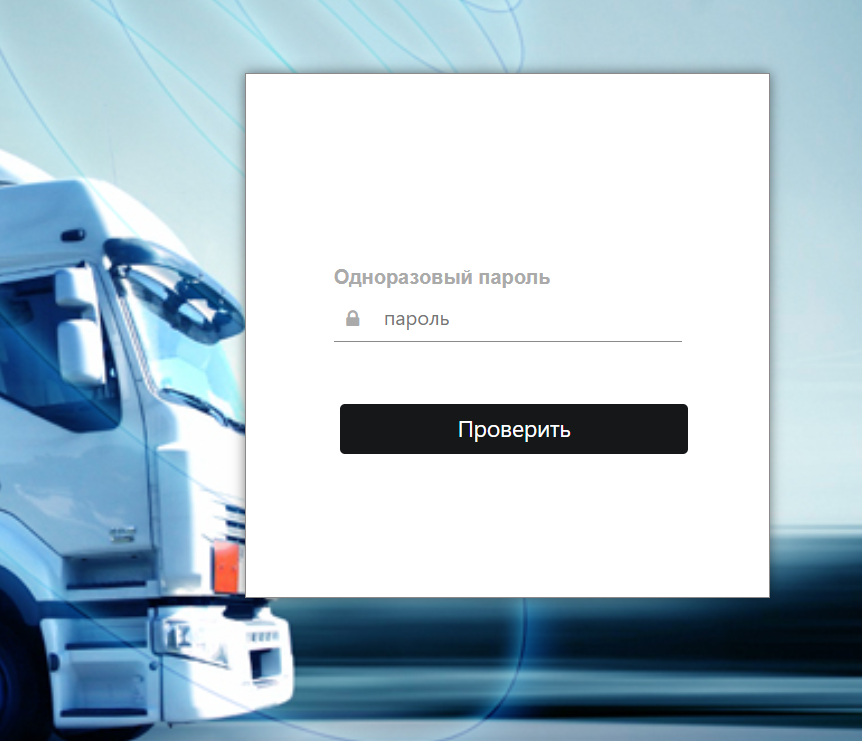
Информация, используемая в приложении «Личный кабинет» является конфиденциальной. Когда пользователи вводят личную информацию необходимо чтобы каналы связи были защищены от неавторизированных пользователей при передачи и хранении. Однофакторная аутентификация на основе пароля является наиболее распространенным методом аутентификации. При таком подходе пользователи предоставляют уникальную комбинацию имени пользователя и пароля для получения доступа к приложению. Однако этот метод уязвим, если пароль слабый или если он был украден. Для повышения уровня безопасности вместо однофакторной аутентификации предлагается использования многофакторной аутентификации.

При выборе факторов, используемых для реализации многофакторной аутентификации необходимо учитывать не только уровни риска, но и расходы, затрачиваемые на реализацию данного фактора. Учитывая эти факторы для реализации многофакторной аутентификации используется долговременный пароль и одноразовый пароль.

Одноразовый пароль – это пароль действительный только для одного сеанса входа в систему или транзакции. Пароль генерируется автоматически и отправляется пользователю на электронную почту. Одноразовый пароль представляет собой криптографический хэш. Генерируется с использованием математического алгоритма, который принимает на вход имя пользователяб секретный ключ и текущее время. Затем генерируется уникальный пароль, который действителен только для одной сессии. В приложении уникальный код действует короткий промежуток времени – 180 секунд. Коды при передачи защищаются, однако их можно украсть, если получить доступ к серверу, генерирующему коды.

При переходе по ссылке доступа к приложению «Личный кабинет» пользователи оказываются на странице авторизации, на которой располагается форма, где пользователю предлагается ввести логин и пароль (рисунок 4.1).

Рисунок 4.1 Страница входа в приложение "Личный кабинет"



После заполнения формы и нажатия на кнопку «вход», пароль шифруется вместе с логином отправляются на сервер. На сервере происходит проверка, полученных данных. Если пользователь с таким логином существует, проверяется совпадение паролей. Если пользователь ввел верный пароль, то на сервере генерируется одноразовый пароль, который отправляется пользователю на электронную почту. Пользователю предоставляется возможность ввести одноразовый пароль и после проверки его на сервере, для каждого пользователя формируется «session token», который содержит некоторую информацию о пользователях – идентификатор и роль пользователя. В будущем это информация используется при формировании запросов на сервер.

Наличие токена в запросе подтверждает, что пользователь авторизирован. Каждый токен имеет определённое время жизни. Когда время жизни токена истекает, он становится недействителен, пользователь автоматически выходит из системы. Если пользователь выходит из системы нажимая на кнопку «выход», то токен автоматически аннулируется и удаляется.

ввод логина и пароля

шифрование данных и отправка на сервер

пользователь с таким логином существует

отправка ошибки. Сообщение об ошибки: пользователя с указанным логином не существует

нет

пароли совпадают

да

отправка ошибки. Сообщения об ошибки: пароль указан неверно.

нет

генерирование одноразового пароля

да

отправка пароля на почту, ввод пароля

пароли совпадают

да

нет

генерация токена

отправка ответа с токеном и информацией о пользователе

Рисунок 4.2 Блок-схема - Авторизация пользователя

Список источников

1. OWASP Top – 10 – 2023 – The Ten Most Critical Web Application Security Risks. [https://owasp.org/ (дата](https://owasp.org/%20(дата) доступа: 04.03.2024)
2. Web Application Security Consortium. <http://projects.webappsec.org/f/WASC-TC-v2_0.pdf/> (дата доступа: 05.03.2024)
3. Волосенков В. О, Лупорев С. Н. Способы обеспечения информационной безопасности веб-приложения // Проблемы безопасности российского общества-2015 № 1.
4. Сердюк В. А. Организация и технология защиты информации. Обнаружение и предотвращение информационных атак в автоматизированных системах предприятий – Москва: ВШЭ, 2011.576с.
5. Teodoro, N., & Serrao, C. (2011). *Web application security: Improving critical web-based applications quality through in-depth security analysis. International Conference on Information Society (i-Society 2011).* doi:10.1109/i-society18435.2011.5978496
6. Вульфин А.М. Анализ защищенности веб-приложения для доступа к системе хранения критически важных данных// Моделирование, оптимизация и информационные технологии – 2021 № 9(4)
7. ГОСТ. Национальный стандарт российской федерации. Защита информации. Уровни идентификации и аутентификации.
8. Надейкина В.С., Лагуткина Т.В. Анализ способов реализации многофакторной аутентификации // Научный результат. Информационные технологии. – Т. 7 №4,2022
9. Гуляев М.Ю. Московский государственный технический университет им. Н.Э. Баумана ОСНОВНЫЕ АРХИТЕКТУРНЫЕ ПОДХОДЫ ПРИ ПОСТРОЕНИИ ВЕБПРИЛОЖЕНИЙ
10. G. Blinowski et al.: Monolithic vs. Microservice Architecture: Performance and Scalability Evaluation