ВЕБ-ОРИЕНТИРОВАННЫЙ МЕНЕДЖЕР ПАРОЛЕЙ ДЛЯ МАЛОЙ ОРГАНИЗАЦИИ НА ПЛАТФОРМЕ ASP.NET CORE С МАСТЕР-ПАРОЛЕМ И ШИФРОВАНИЕМ ХРАНИЛИЩА

Молотков Даниил Юрьевич
Студент, кафедра «Вычислительные системы и информационная безопасность», Донской государственный технический университет

Газизов Андрей Равильевич
Доцент, кандидат педагогических наук, заведующий кафедрой «Вычислительные системы и информационная безопасность», Донской государственный технический университет

В малых организациях часто возникает необходимость безопасного совместного хранения корпоративных паролей (учётные записи сервисов, оборудование). Использование незащищённых таблиц или общих файлов создаёт высокие риски утечки данных. В статье представлена архитектура и реализация веб-приложения на ASP.NET Core, обеспечивающего централизованное хранение паролей в зашифрованном виде, доступ на основе ролей и мастер-пароля, а также полный аудит действий. Шифрование данных осуществляется прозрачно для пользователей на уровне приложения с использованием AES-256, ключ шифрования защищён производным от мастер-пароля ключом. Система позволяет разграничить доступ сотрудников к группам паролей и отслеживать историю их использования.

Ключевые слова: *ASP.NET Core, Entity Framework Core, менеджер паролей, шифрование AES, PBKDF2, мастер-пароль, безопасное хранение, ролевой доступ, аудит, веб-приложение.*

В малых организациях повсеместно распространена практика хранения корпоративных паролей (учётные данные от сервисов, сетевого оборудования, общих почтовых ящиков) в текстовых файлах, электронных таблицах или системах обмена сообщениями. Подобный подход создаёт высокие риски утечки информации, не позволяет разграничить доступ сотрудников и полностью исключает возможность аудита действий с учётными данными [3, с. 45]. При увольнении сотрудника или компрометации рабочей станции все пароли, хранящиеся в незащищённом виде, оказываются под угрозой. Целью настоящей работы является разработка веб-ориентированного менеджера паролей на платформе ASP.NET Core, обеспечивающего централизованное хранение учётных записей в зашифрованном виде, доступ на основе ролевой модели и мастер-пароля, а также полное протоколирование операций. Приложение ориентировано на использование в небольших коллективах и не требует приобретения коммерческих продуктов.

Функциональные требования к системе включают: создание, редактирование и удаление записей, содержащих название сервиса, логин, пароль и примечание; объединение записей в смысловые группы (проекты, отделы); назначение прав доступа пользователей к группам; ведение журнала всех операций с паролями. С точки зрения безопасности критически важными являются шифрование паролей в базе данных с использованием алгоритма AES-256, двухуровневая схема управления ключами (мастер-пароль и ключ шифрования данных), защита от перебора мастер-пароля и ролевое разграничение доступа к административным функциям [4, с. 312]. Аутентификация сотрудников в системе осуществляется с помощью встроенной подсистемы ASP.NET Core Identity, поддерживающей хеширование паролей и настраиваемые политики блокировки. Доступ к расшифрованным паролям предоставляется только после ввода мастер-пароля, который является общим для всех пользователей системы и задаётся на этапе развёртывания администратором.

В качестве технологической основы выбран фреймворк ASP.NET Core версии 8, функционирующий на платформе .NET 8 с длительной поддержкой. Выбор обусловлен кроссплатформенностью, высокой производительностью и наличием развитых встроенных средств безопасности [1, с. 567]. Для взаимодействия с реляционной базой данных применяется объектно-реляционное отображение Entity Framework Core, использующее провайдер PostgreSQL. Данная СУБД выбрана из соображений надёжности, поддержки сложных запросов и транзакций, а также возможности развёртывания в контейнеризованной среде. Криптографические операции реализуются исключительно штатными классами пространства имён System.Security.Cryptography без привлечения сторонних библиотек, что повышает доверие к решению и упрощает аудит кода. Пользовательский интерфейс строится на основе технологии Razor Pages с применением CSS-фреймворка Bootstrap 5, обеспечивающего адаптивную вёрстку и корректное отображение на различных устройствах.

Модель данных разработанной системы включает пять основных сущностей. Пользователи, расширяющие стандартную модель IdentityUser дополнительным полем FullName, хранят учётные записи сотрудников. Группы паролей (PasswordGroup) характеризуются названием и описанием и служат для логической группировки хранимых записей. Связь между пользователями и группами с признаком права на просмотр реализована через сущность UserGroupPermission, что позволяет гибко назначать и отзывать доступ к подмножествам паролей. Непосредственно хранимые пароли (StoredPassword) содержат название сервиса, логин, зашифрованные данные пароля, вектор инициализации (IV), дату создания и внешний ключ на создавшего пользователя. Поле с зашифрованным паролем представлено массивом байт, что исключает случайное попадание открытых данных в логи или дампы базы данных. Журнал аудита (AuditLog) фиксирует временную метку, идентификатор пользователя, тип действия (просмотр, создание, изменение, удаление) и ссылку на объект операции. Целостность данных на уровне базы

обеспечивается внешними ключами и настроенным каскадным удалением для связанных записей, где это обусловлено бизнес-логикой.

Центральным элементом архитектуры является двухуровневая схема управления ключами шифрования, реализованная в соответствии с принципом разделения ключей [2, с. 234]. На этапе первоначальной настройки системы генерируется случайный 256-битный ключ шифрования данных (Data Encryption Key, DEK) с использованием криптографически стойкого генератора RandomNumberGenerator. Администратор задаёт мастер-пароль, из которого посредством функции формирования ключа PBKDF2 (алгоритм Rfc2898DeriveBytes) [8] с уникальной случайной солью и ста тысячами итераций хеширования вырабатывается ключ шифрования ключа (Key Encryption Key, KEK). DEK зашифровывается алгоритмом AES-256 в режиме CBC с использованием KEK, после чего полученный шифротекст вместе с солью сохраняется в служебной таблице настроек системы. Открытое значение DEK нигде в базе данных не присутствует. При начале активной сессии пользователь вводит мастер-пароль; система извлекает соль и зашифрованный DEK, повторно вычисляет KEK и расшифровывает DEK, помещая его в защищённый сессионный кэш на базе IDistributedCache. Для каждого хранимого пароля при создании генерируется уникальный 128-битный вектор инициализации, и пароль шифруется алгоритмом AES-256 в режиме CBC с использованием DEK и этого вектора. При запросе на просмотр пароля DEK извлекается из сессионного кэша, после чего пароль расшифровывается и возвращается пользователю. Таким образом, компрометация файла базы данных без знания мастер-пароля не позволяет злоумышленнику восстановить ни один пароль, а смена мастер-пароля требует перешифрования только зашифрованного DEK, не затрагивая все хранимые записи.

Ролевая модель системы включает две роли: «Администратор» и «Пользователь». Администратор имеет доступ к управлению пользователями, группами и правами, а также к журналу аудита. Пользователь может просматривать только те группы и пароли, которые разрешены ему через

соответствующие записи в таблице UserGroupPermission. Проверка ролей реализована через политики авторизации ASP.NET Core с применением атрибутов [Authorize(Roles = "Администратор")] на соответствующих страницах Razor. Дополнительно на уровне бизнес-логики при каждом запросе проверяется членство пользователя в группе, что исключает несанкционированный доступ даже в случае ошибок конфигурации маршрутизации. Все операции с паролями, включая просмотр, создание, редактирование и удаление, протоколируются через вызов сервиса аудита. Журнал доступен исключительно администратору и защищён от редактирования, что обеспечивает неотказуемость и полную прослеживаемость изменений.

Пользовательский интерфейс приложения построен на основе Razor Pages и использует компоненты Bootstrap 5 для создания единообразного визуального оформления. Главная страница после аутентификации отображает список групп, доступных текущему пользователю. При входе в группу выводится таблица с хранимыми записями, где пароли по умолчанию скрыты. Для отображения пароля система запрашивает мастер-пароль, если срок сессионного кэша истёк, после чего пароль временно показывается в открытом виде. Формы ввода данных снабжены валидацией как на стороне клиента, так и на сервере, что предотвращает попадание в базу некорректных записей. Административный раздел содержит страницы управления пользователями, группами, назначениями прав и журнал аудита с возможностью фильтрации по дате и типу события. Визуальные индикаторы состояний (активная запись, просроченный доступ) помогают оперативно оценивать текущую конфигурацию.

Апробация разработанной системы проводилась в лабораторной среде, имитирующей инфраструктуру малого предприятия численностью до двадцати сотрудников. Приложение было развёрнуто на виртуальной машине под управлением Ubuntu Server 22.04 в контейнере Docker, в качестве обратной базы данных использовался PostgreSQL 16. В ходе тестирования

моделировались сценарии добавления новых сотрудников, назначения им прав на определённые группы паролей, просмотра и изменения записей, а также попытки несанкционированного доступа к паролям чужой группы. Результаты показали, что время расшифровки одного пароля составляет менее ста миллисекунд, что незаметно для пользователя, а производительность системы при одновременной работе до десяти пользователей остаётся стабильной. Журнал аудита позволил полностью восстановить хронологию действий каждого участника. Применение ролевой модели исключило возможность просмотра чужих паролей даже при прямой манипуляции URL-адресами. Резервное копирование базы данных вместе с зашифрованным DEK обеспечивает возможность восстановления системы на новом сервере после повторного ввода мастер-пароля.

Сравнение разработанного менеджера паролей с существующими аналогами, такими как KeePass (клиент-серверные модификации), Bitwarden в режиме самостоятельного развёртывания и коммерческая система Passwork, выявляет ряд преимуществ и ограничений [5, с. 312]. К достоинствам следует отнести полный контроль над данными и их физическим размещением, отсутствие лицензионных отчислений, простоту установки благодаря контейнеризации, а также прозрачную интеграцию с корпоративной системой учётных записей на базе ASP.NET Core Identity. Возможность гибкой доработки под специфические требования организации выгодно отличает данное решение от коробочных продуктов с фиксированной функциональностью. Вместе с тем, текущая версия не поддерживает двухфакторную аутентификацию для мастер-пароля, не имеет механизма автономного доступа к паролям при отсутствии соединения с сервером и не включает браузерное расширение для автоматической подстановки учётных данных в веб-формы. Перспективы развития системы связаны с реализацией второго фактора на основе TOTP, созданием REST API для интеграции с внешними сервисами и разработкой мобильного клиента.

Проведённая работа позволила создать защищённый веб-ориентированный менеджер паролей, отвечающий потребностям малых организаций в безопасном хранении и совместном использовании учётных данных. Благодаря двухуровневому шифрованию с разделением ключей и применению мастер-пароля обеспечивается конфиденциальность информации даже при компрометации базы данных. Ролевая модель и аудит действий гарантируют разграничение доступа и полную прослеживаемость операций. Использование платформы ASP.NET Core и встроенных криптографических средств подтвердило эффективность современных технологий .NET для построения безопасных веб-приложений. Разработанное решение может быть адаптировано для использования в организациях различного профиля, где существует потребность в централизованном и контролируемом управлении паролями.

Список использованных источников

1. Лок, Э. ASP.NET Core в действии / Э. Лок ; пер. с англ. – М.: ДМК Пресс, 2021. – 832 с.

2. Прайс, М. C# 10 и .NET 6. Современная кросс-платформенная разработка / М. Прайс ; пер. с англ. – СПб.: Питер, 2022. – 848 с.

3. Хоффман, Э. Безопасность веб-приложений. Разведка, защита, нападение / Э. Хоффман ; пер. с англ. – СПб.: Питер, 2021. – 384 с.

4. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке С / Б. Шнайер ; пер. с англ.–М.: Триумф, 2002.–816 с.

5. Моргунов, Е.П. PostgreSQL. Основы языка SQL / Е.П. Моргунов. – СПб.: БХВ-Петербург, 2023. – 336 с.

6. Смит, Дж. Entity Framework Core в действии / Дж. Смит ; пер. с англ. – М.: ДМК Пресс, 2022. – 520 с.

7. Официальная документация Microsoft. Безопасность в ASP.NET Core [Электронный ресурс]. – URL: https://learn.microsoft.com/ru-ru/aspnet/core/security/ (дата обращения: 26.06.2026).

8. Официальная документация Microsoft. Класс Rfc2898DeriveBytes [Электронный ресурс]. – URL: https://learn.microsoft.com/ru-ru/dotnet/api/system.security.cryptography.rfc2898derivebytes (дата обращения: 26.06.2026).

A WEB-BASED PASSWORD MANAGER FOR SMALL ORGANIZATIONS ON THE ASP.NET CORE PLATFORM WITH A MASTER PASSWORD AND ENCRYPTED STORAGE

Molotkov Daniil Yurievich

Student, Department of "Computer Systems and Information Security", Don State Technical University

Gazizov Andrey Ravilevich

Associate Professor, Candidate of Pedagogical Sciences, Head of the Department of "Computer Systems and Information Security", Don State Technical University

In small organizations, corporate passwords are often stored in unprotected forms such as spreadsheets or text files, leading to high risks of data leakage and a lack of access control and auditing. The aim of this work is to develop a web-based password manager on the ASP.NET Core platform that provides centralized storage of credentials in encrypted form, access based on a role model and a master password, and full activity logging. The article presents the architecture of the system, a two-level key management scheme using AES-256 and PBKDF2, and a data model implemented with Entity Framework Core and PostgreSQL. Role-based access control and an audit log ensure secure and transparent handling of sensitive data. The system was tested in a laboratory environment simulating a small enterprise infrastructure and demonstrated high performance and resistance to unauthorized access.

Keywords: password manager, ASP.NET Core, encryption, AES, PBKDF2, master password, role-based access, audit, PostgreSQL, secure storage