**Министерство науки и высшего образования РФ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»**

**(ВлГУ)**

СМК 08/03-18

Срок хранения 2 года

ВЛГУ.10.05.04.04.05.00 ПЗ

**ОТЧЕТ О НИР**

«Разработка утилиты для управления шифрованными хранилищами на основе взаимодействия с Arduino»

Специальность 10.05.04 – «Информационно-аналитические системы безопасности»

СПИСОК ИСПОЛНИТЕЛЕЙ

|  |  |  |
| --- | --- | --- |
| Руководитель НИР |  | к.т.н. доцент кафедры ИЗИ Ю.М. Монахов |
|  |
| Исполнитель |  | студент группы ИСБ-120 Г.С. Сухецкий |
|  |  |  |  |

Владимир 2024 г

## СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc167318462)

[ВВЕДЕНИЕ 3](#_Toc167318463)

[1 LINUX UNIFIED KEY SETUP 4](#_Toc167318464)

[2 ОСНОВНЫЕ ЭТАПЫ РАЗРАБОТКИ УТИЛИТЫ 5](#_Toc167318465)

[3 ОПИСАНИЕ РАБОТЫ УТИЛИТЫ 6](#_Toc167318466)

[ЗАКЛЮЧЕНИЕ 10](#_Toc167318467)

[СПИСОК ЛИТЕРАТУРЫ 11](#_Toc167318468)

[ПРИЛОЖЕНИЕ A 12](#_Toc167318469)

## ВВЕДЕНИЕ

Отчет 12 страниц.

Объект исследования:

*Система по созданию криптографический контейнеров.*

Предмет исследования:

*Проверка возможности генерирования паролей на микроконтроллере Arduino и дальнейшее использование этих паролей для работы с зашифрованными хранилищами данных.*

Цель исследования:

*Создание утилиты для управления зашифрованными хранилищами данных*.

В процессе прохождения практики требовалось решить следующие задачи:

1. Подготовка окружения и установка необходимых библиотек
2. Разработка части приложения для взаимодействия с Arduino и генерации паролей.
3. Разработка части приложения для взаимодействия с диском и созданию крипто-контейнеров.
4. Объединение частей приложения
5. Тестирование

Проект имеет высокую актуальность с точки зрения инновационности и научного интереса, а также обладает практической значимостью и потенциалом для повышения безопасности данных и оптимизации процесса управления зашифрованными хранилищами.

## LINUX UNIFIED KEY SETUP

LUKS (Linux Unified Key Setup) — это стандарт для шифрования блочных устройств в Linux. Он был разработан для обеспечения безопасности данных на уровне дисков и широко используется в различных дистрибутивах Linux. LUKS предоставляет удобный и безопасный способ управления ключами и шифрованием, что делает его идеальным для защиты конфиденциальной информации на жестких дисках, SSD и внешних накопителях.

Основные особенности LUKS:

Унифицированный формат: LUKS определяет универсальный формат для хранения метаданных шифрования на блочных устройствах, что обеспечивает совместимость между различными системами.

Многоуровневая защита: LUKS использует мастер-ключ для шифрования данных и набор ключевых слотов для хранения ключей шифрования. Это позволяет легко управлять доступом и изменять ключи.

Поддержка нескольких ключей: LUKS позволяет хранить несколько ключей шифрования, что обеспечивает гибкость в управлении доступом к зашифрованным данным.

Прозрачность для пользователя: После успешной аутентификации LUKS-диск монтируется как обычное блочное устройство, что делает его использование прозрачным для конечного пользователя.

Принципы работы LUKS:

LUKS работает на основе нескольких ключевых компонентов и процессов:

Мастер-ключ: Основной ключ, который используется для шифрования и дешифрования данных на диске. Этот ключ никогда не сохраняется в открытом виде и всегда защищен одним из ключей аутентификации.

Ключевые слоты: Места хранения ключей аутентификации (паролей или ключевых файлов), которые шифруют мастер-ключ. LUKS поддерживает до 8 ключевых слотов, что позволяет использовать несколько паролей для доступа к одному и тому же зашифрованному разделу.

Шифрование данных: Данные на диске шифруются с использованием симметричных алгоритмов шифрования (например, AES), что обеспечивает высокую производительность и безопасность.

## ОСНОВНЫЕ ЭТАПЫ РАЗРАБОТКИ УТИЛИТЫ

1. Подготовка окружения и установка необходимых библиотек

Для разработки утилиты был использован язык программирования Python и ряд дополнительных библиотек, таких как serial для работы с последовательным портом и cryptsetup для управления шифрованными томами. Также потребовалась настройка Arduino IDE для программирования микроконтроллера.

2. Разработка кода на Python

Основной функционал утилиты реализованый на языке Python. Включает следующие ключевые компоненты:

* Инициализация соединения с Arduino.
* Передача фразы для генерации пароля.
* Получение сгенерированного пароля.
* Создание, открытие, закрытие и изменение размера шифрованных хранилищ.

3. Разработка кода для Arduino

Программа на Arduino реализует следующие функции:

* Чтение фразы из последовательного порта.
* Генерация пароля с использованием алгоритма SHA-512.
* Передача сгенерированного пароля обратно на компьютер.

## ОПИСАНИЕ РАБОТЫ УТИЛИТЫ

Утилита, выполняет следующие основные функции:

* Создание нового хранилища (new\_vault)

Пользователь инициирует создание нового шифрованного хранилища, вызывая команду new\_vault(). Процесс создания хранилища включает несколько шагов:

* Утилита запрашивает у пользователя ввод фразы для генерации пароля.
* Фраза передается на Arduino через последовательный порт.
* Arduino генерирует пароль на основе хэширования SHA-512 и отправляет его обратно на утилиту.
* Утилита создает новый файл хранилища и форматирует его с использованием утилиты cryptsetup для шифрования.
* Создается файловая система на шифрованном разделе, после чего хранилище закрывается и готово к использованию.

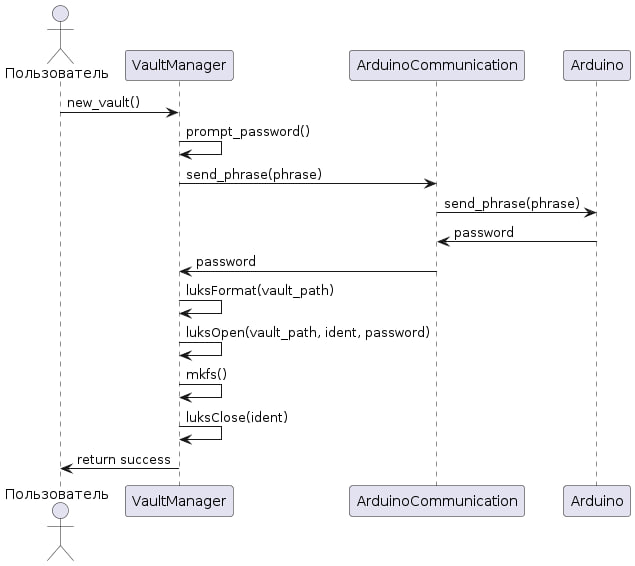


Рисунок 1

* Открытие существующего хранилища (open\_vault)

Для доступа к уже существующему шифрованному хранилищпользователь использует команду open\_vault(). Процесс открытия хранилища включает следующие шаги:

* Утилита запрашивает у пользователя ввод фразы для генерации пароля.
* Фраза передается на Arduino, который генерирует и возвращает пароль.
* Утилита использует пароль для открытия шифрованного раздела с помощью cryptsetup.
* После успешного открытия раздела утилита монтирует файловую систему, делая данные доступными для пользователя.

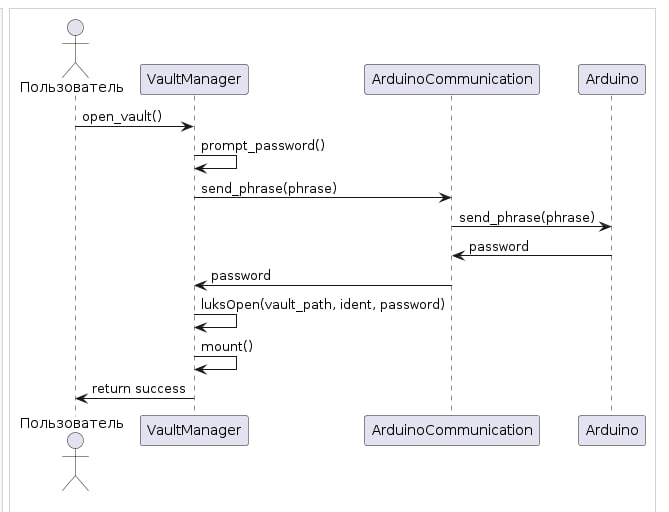


Рисунок 2

* Закрытие хранилища (close\_vault):

После завершения работы с данными в хранилище пользователь может закрыть его, вызвав команду close\_vault(). Процесс закрытия включает:

* Проверку, что никакие процессы не используют файлы в хранилище.
* Завершение всех процессов, использующих файлы, если они есть.
* Отмонтирование файловой системы и закрытие шифрованного раздела с помощью cryptsetup.

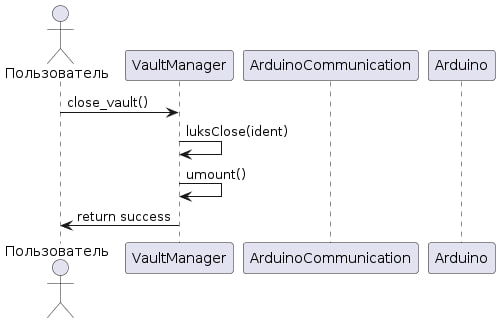


Рисунок 3

* Изменение размера хранилища (resize\_vault)

Для увеличения размера существующего хранилища пользователь вызывает команду resize\_vault(). Процесс изменения размера включает:

* Утилита запрашивает у пользователя ввод фразы для генерации пароля.
* Фраза передается на Arduino, который генерирует и возвращает пароль.
* Утилита копирует текущее хранилище для создания резервной копии.
* Утилита изменяет размер файла хранилища с помощью qemu-img.
* Открытие шифрованного раздела с помощью пароля, увеличение размера файловой системы и закрытие раздела.

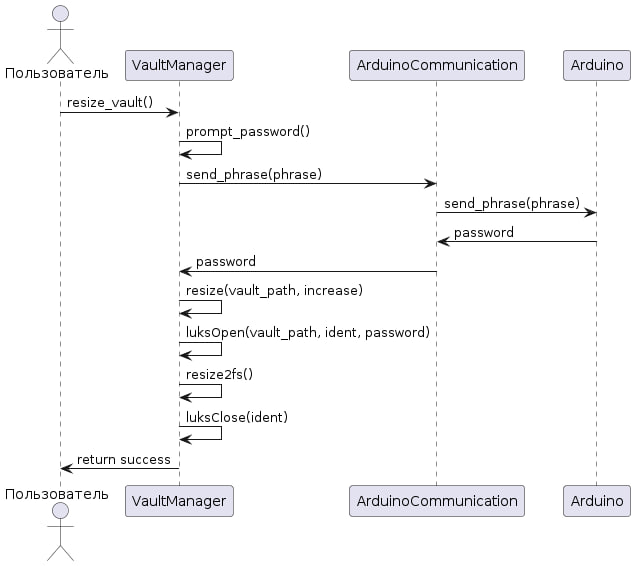


Рисунок 4.

## ЗАКЛЮЧЕНИЕ

В ходе проведенного исследования была разработана утилита для управления зашифрованными хранилищами данных, использующая микроконтроллер Arduino для генерации паролей. Это позволяет значительно повысить уровень безопасности хранения конфиденциальной информации за счет использования аппаратных методов создания сложных паролей.

## СПИСОК ЛИТЕРАТУРЫ

1. Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. Pearson Education.

2. Schneier, B. (2015). Applied Cryptography: Protocols, Algorithms, and Source Code in C. John Wiley & Sons.

3. Ferguson, N., Schneier, B., & Kohno, T. (2010). Cryptography Engineering: Design Principles and Practical Applications. John Wiley & Sons.

4. Tanenbaum, A. S., & Bos, H. (2014). Modern Operating Systems. Pearson

5. Goyal, P. (2018). Mastering Python Networking. Packt Publishing.

6. Paar, C., & Pelzl, J. (2010). Understanding Cryptography: A Textbook for Students and Practitioners. Springer.

## ПРИЛОЖЕНИЕ A

Ссылка на github проекта: https://github.com/Krim-code/PyVault-LuksBased.git