ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ / INFORMATICS AND INFORMATION PROCESSES

DOI: https://doi.org/10.60797/IRJ.2024.146.31

ПАРАЛЛЕЛЬНЫЙ МОДУЛЬ ДЛЯ КРИПТОГРАФИЧЕСКОЙ ЗАЩИТЫ ФАЙЛОВ

Научная статья

Николаев В.А.[1, *], Гуляев И.А.[2], Лаптева М.Г.[3], Хафизова А.Ш.[4], Пикулева Н.И.[5]

[1, 2, 4, 5] Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ, Казань, Российская Федерация

[3] Казанский национальный исследовательский технологический университет, Казань, Российская Федерация

* Корреспондирующий автор (vanikolaev2002[at]yandex.ru)

Аннотация

В данной статье рассматривается разработка параллельного модуля для криптографической защиты файлов. Основной целью исследования является повышение производительности и уровня безопасности систем обработки данных посредством распределения вычислительной нагрузки между несколькими ядрами или узлами. В статье проведен анализ существующих решений и алгоритмов криптографической защиты, таких как AES, TripleDES и Twofish. Особое внимание уделено реализации параллельных вычислений для шифрования и дешифрования данных, что позволяет значительно ускорить процесс обработки информации. Методология тестирования включала планирование и разработку тестовых сценариев, проведение тестов на реальных данных и анализ полученных результатов. Тестирование проводилось на аппаратно-программной платформе, включающей процессор Intel Core i5 10400, 4 модуля оперативной памяти AMD Radeon R7 Performance Series R748G2606U2S-U DDR4 – 8 ГБ 2666 и хранилище данных Samsung SSD 860 EVO 500GB M.2. Результаты показали, что параллельное шифрование и дешифрование работают в среднем в 2,81 раза быстрее, чем последовательное шифрование, а параллельное дешифрование – в 2,57 раза быстрее. Анализ производительности и безопасности подтверждает, что разработанный модуль обеспечивает надежную криптографическую защиту и соответствует современным требованиям к системам обработки данных. Использование параллельных вычислений в криптографии открывает новые возможности для повышения эффективности работы с данными и обеспечения их безопасности.

Ключевые слова: параллельный модуль, криптографическая защита файлов, производительность, безопасность, шифрование, дешифрование, AES, TripleDES, Twofish, параллельные вычисления.

PARALLEL MODULE FOR CRYPTOGRAPHIC FILE PROTECTION

Research article

Nikolaev V.A.[1, *], Gulyaev I.A.[2], Lapteva M.G.[3], Khafizova A.S.[4], Pikuleva N.I.[5]

[1, 2, 4, 5] Kazan National Research Technical University named after A.N. Tupolev – KAI, Kazan, Russian Federation

[3] Kazan National Research Technological University, Kazan, Russian Federation

* Corresponding author (vanikolaev2002[at]yandex.ru)

Abstract

This article examines the development of a parallel module for cryptographic protection of files. The main objective of the research is to improve the performance and security level of data processing systems by distributing the computational load among several cores or nodes. The work analyses existing solutions and algorithms of cryptographic protection such as AES, TripleDES and Twofish. Particular attention is paid to the implementation of parallel computing for data encryption and decryption, which allows to significantly accelerate the information processing. The testing methodology included planning and development of test scenarios, conducting tests on real data and analysing the obtained results. The testing was conducted on a hardware and software platform including Intel Core i5 10400 processor, 4 AMD Radeon R7 Performance Series R748G2606U2S-U DDR4 – 8 GB 2666 RAM modules and Samsung SSD 860 EVO 500GB M.2 data storage. The results show that parallel encryption and decryption perform on average 2.81 times faster than serial encryption and parallel decryption performs 2.57 times faster. Performance and security analysis confirms that the developed module provides reliable cryptographic protection and meets modern requirements to data processing systems. The use of parallel computing in cryptography opens up new opportunities for increasing the efficiency of data handling and ensuring its security.

Keywords: parallel module, cryptographic file protection, performance, security, encryption, decryption, AES, TripleDES, Twofish, parallel computing.

Введение

Криптографическая защита информации является одной из самых важных задач в современном мире, где информация стала важнейшим ресурсом. Благодаря разработке параллельного модуля для криптографической защиты файлов можно значительно улучшить производительность и обеспечить высокий уровень безопасности данных. Это особенно важно в современном мире, где утечка информации может привести к серьезным последствиям как для частных лиц, так и для организаций и государств.

Применение параллельных вычислений для шифрования и дешифрования файлов позволит эффективно использовать ресурсы многопроцессорных и многоядерных систем, что приведет к увеличению скорости работы с данными. Это также повысит уровень защиты информации, так как параллельные алгоритмы криптографической защиты могут быть более надежными и сложными для взлома. Таким образом, разработка параллельного модуля для

криптографической защиты файлов является важным шагом в обеспечении безопасности информации и повышении производительности систем обработки данных.

Криптография – наука о методах защиты информации от несанкционированного доступа. С ее помощью защищают данные, используя различные методы шифрования. История криптографии насчитывает тысячелетия, начиная с простых методов замены символов и перестановки букв и заканчивая современными алгоритмами, такими как AES и RSA. Развитие криптографии тесно связано с развитием человеческой цивилизации и с появлением новых технологий, таких как компьютеры и информационные технологии. Криптография остается актуальной и важной для обеспечения безопасности данных в современном мире.

Современные методы криптографической защиты файлов играют важную роль в обеспечении безопасности информации. Основные методы включают в себя симметричное и асимметричное шифрование. Симметричное шифрование использует один ключ для шифрования и расшифрования, например, AES [1]. Асимметричное шифрование использует пару ключей: открытый для шифрования и закрытый для расшифровки, например, RSA. Также применяется гибридное шифрование, объединяющее оба метода.

Хэширование является еще одним важным методом, основанным на хэш-функциях, которые преобразуют данные в фиксированную строку. Это позволяет проверять целостность файлов и обнаруживать подделки, а также использоваться для хранения паролей и проверки подлинности. Различные хэш-функции, такие как MD5 [2], SHA-1 [3] и SHA-256 [4], имеют уникальные особенности и применения в зависимости от требований безопасности.

Цифровые подписи используются для подтверждения подлинности и целостности файлов путем хеширования и шифрования. Они обеспечивают непротиворечивость информации и защиту от изменений в процессе передачи.

Контейнеры с защищенным доступом позволяют шифровать и ограничивать доступ к конфиденциальной информации, обеспечивая безопасность данных. Один из распространенных методов защиты контейнеров – симметричное шифрование с использованием одного ключа для шифрования и расшифрования данных.

Параллельные вычисления позволяют выполнять операции над данными одновременно, используя множество вычислительных ресурсов. Это ускоряет обработку данных и выполнение сложных вычислительных задач. Важно разделить задачу на подзадачи, распределить их между вычислительными узлами или ядрами процессора. Параллельные вычисления могут быть использованы для ускорения шифрования файлов. Для эффективного управления задачами используются параллельные программные библиотеки.

Однако возникают сложности с синхронизацией данных и доступом к общим ресурсам, для решения которых применяются методы синхронизации. Важно разработать эффективные алгоритмы распределения нагрузки между вычислительными узлами, учитывая особенности вычислительных ресурсов и требования к производительности. Применение параллельных вычислений требует анализа задачи и выбора подходящих методов и инструментов.

В современном мире, где информация является важнейшим ресурсом, обеспечение ее безопасности становится первостепенной задачей. Разработка параллельного модуля для криптографической защиты файлов позволяет значительно улучшить производительность и надежность систем обработки данных. Использование параллельных вычислений в криптографии открывает новые возможности для защиты информации и повышения эффективности работы с данными.

Структура разработанного модуля

Разработка параллельного модуля для криптографической защиты файлов с использованием таких алгоритмов, как AES, TripleDES [5] и Twofish [6], позволяет существенно повысить производительность и уровень безопасности систем обработки данных.

AES (Advanced Encryption Standard) является симметричным алгоритмом шифрования, который обеспечивает высокий уровень защиты и быструю обработку данных. Он работает с блоками данных размером 128 бит и поддерживает различные длины ключей (128, 192 и 256 бит). Алгоритм AES использует несколько раундов преобразований для шифрования данных, что делает его устойчивым к различным видам атак. В нашем модуле реализация AES основана на использовании режима шифрования CBC и генерации уникального вектора инициализации для каждого шифрования.

Twofish – это один из самых надежных и устойчивых к атакам алгоритмов шифрования. Он поддерживает ключи различной длины и блочные размеры данных 128 бит. Алгоритм Twofish использует преобразования Галуа и подстановки байтов для повышения стойкости к криптоанализу, что обеспечивает высокую производительность при шифровании файлов различных размеров и типов.

TripleDES является модификацией оригинального алгоритма DES (Data Encryption Standard). Он использует три последовательных прохода DES для шифрования данных, что значительно увеличивает стойкость алгоритма. Несмотря на появление более мощных алгоритмов, TripleDES остается надежным методом шифрования, обеспечивающим защиту данных на высоком уровне.

Параллельный модуль для криптографической защиты файлов, названный «ParallelEncryptionModule», предоставляет возможность параллельного шифрования и дешифрования файлов с использованием алгоритмов AES, TripleDES и Twofish. Модуль также включает модульные тесты для проверки его функциональности.

Структура модуля «ParallelEncryptionModule» обеспечивает четкое разделение компонентов и функциональности, что позволяет эффективно разрабатывать, поддерживать и тестировать систему. Основные компоненты модуля включают:

1. Algorithms: реализация алгоритмов шифрования AES, TripleDES и Twofish.
2. EncryptionManager: управление процессами шифрования и дешифрования файлов.
3. KeyManagement: управление ключами шифрования и дешифрования.
4. ParallelProcessing: механизмы параллельной обработки данных.

Интеграция параллельного модуля с файловым менеджером является ключевым аспектом его архитектуры. Файловый менеджер предоставляет удобный пользовательский интерфейс для работы с файлами и управляет операциями копирования, перемещения и удаления файлов.

Архитектура файлового менеджера построена на принципах модульности и разделения ответственности. В его структуре присутствуют следующие компоненты:

1. Commands: реализация пользовательских команд.
2. Converter: преобразование данных.
3. DB: управление базой данных.
4. Icons: иконки для отображения файлов и папок.
5. Models: модели данных файловой системы.
6. Services: дополнительные сервисы.
7. View: файлы пользовательского интерфейса.
8. ViewModels: связь моделей и представлений.
9. app.xaml: главный объект приложения.
10. readme.md: документация по файловому менеджеру.

Файловый менеджер взаимодействует с параллельным модулем для шифрования и дешифрования файлов. Пользователь выбирает файлы и ключ шифрования для зашифровки или зашифрованные файлы и ключ дешифрования для расшифровки. Параллельный модуль выполняет операции с использованием выбранного алгоритма, обеспечивая криптографическую защиту данных. Файловый менеджер предоставляет удобный интерфейс для работы с зашифрованными и расшифрованными файлами, что делает управление данными простым и эффективным.

Диаграмма класса EncryptionDialogueViewModel представлена на рисунке 1.

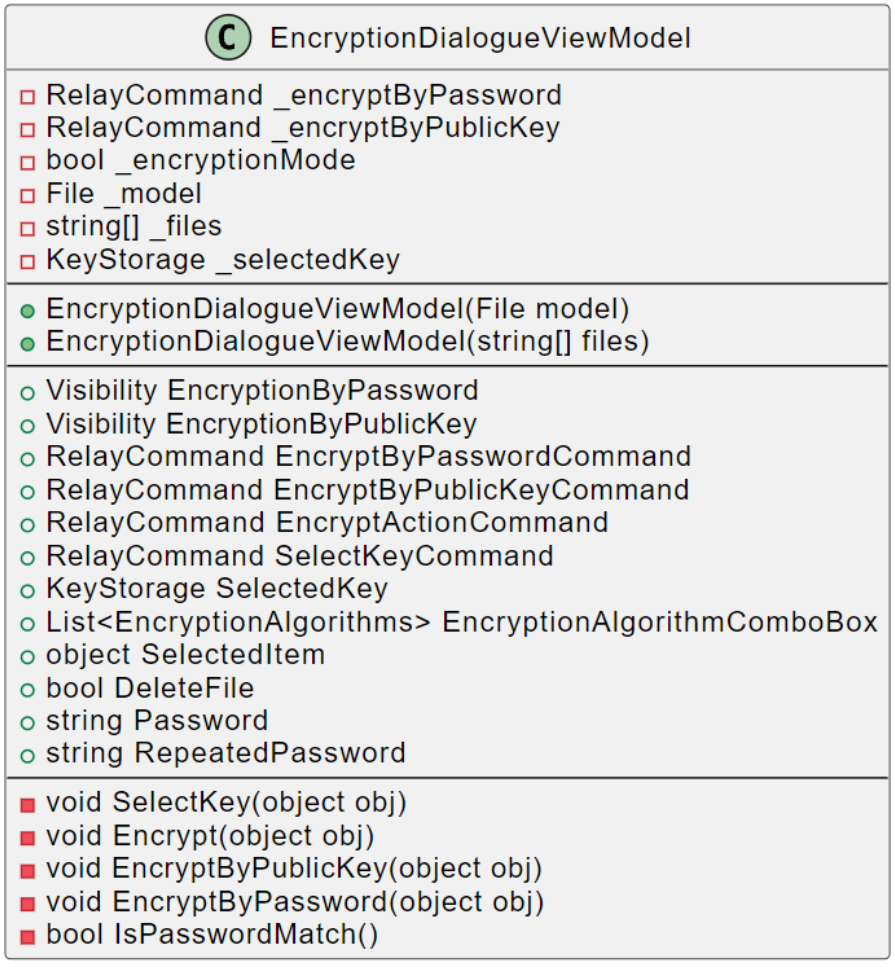

Рисунок 1 - Диаграмма класса программного модуля

DOI: https://doi.org/10.60797/IRJ.2024.146.31.1

Диаграмма классов «EncryptionDialogueViewModel» представляет структуру и функциональность класса «EncryptionDialogueViewModel», который отвечает за обработку и управление операциями шифрования и дешифрования файлов в приложении «ParallelEncryptionModule».

Основные элементы класса «EncryptionDialogueViewModel» включают поля, конструкторы, свойства и методы. Поля включают «private RelayCommand _encryptByPassword» для выполнения шифрования по паролю, «private RelayCommand _encryptByPublicKey» для выполнения шифрования по открытому ключу, «private bool _encryptionMode» для режима шифрования (по умолчанию включен), «private readonly File _model» для модели файла,

который подлежит шифрованию, «private readonly string[] _files» для массива файлов, предназначенных для шифрования, и «private KeyStorage _selectedKey» для выбранного ключа шифрования.

Конструкторы включают «public EncryptionDialogueViewModel(File model)» для инициализации с моделью файла и «public EncryptionDialogueViewModel(string[] files)» для инициализации с массивом файлов. Свойства включают «public Visibility EncryptionByPassword { get; }» для видимости опции шифрования по паролю, «public Visibility EncryptionByPublicKey { get; }» для видимости опции шифрования по открытому ключу, «public RelayCommand EncryptByPasswordCommand { get; }» для команды шифрования по паролю, «public RelayCommand EncryptByPublicKeyCommand { get; }» для команды шифрования по открытому ключу, «public RelayCommand EncryptActionCommand { get; }» для команды выполнения шифрования, «public RelayCommand SelectKeyCommand { get; }» для команды выбора ключа, «public KeyStorage SelectedKey { get; set; }» для выбранного ключа шифрования, «public List<EncryptionAlgorithms> EncryptionAlgorithmComboBox { get; }» для списка доступных алгоритмов шифрования, «public object SelectedItem { get; set; }» для выбранного элемента (файл или ключ), «public bool DeleteFile { get; set; }» для флага удаления файла после шифрования, «public string Password { get; set; }» для пароля шифрования, и «public string RepeatedPassword { get; set; }» для подтверждения пароля.

Методы включают «private void SelectKey(object obj)» для выбора ключа, «private void Encrypt(object obj)» для выполнения шифрования, «private void EncryptByPublicKey(object obj)» для шифрования по открытому ключу, «private void EncryptByPassword(object obj)» для шифрования по паролю и «private bool IsPasswordMatch()» для проверки совпадения паролей.

Класс «EncryptionDialogueViewModel» обеспечивает функциональность для управления операциями шифрования файлов в диалоговом окне приложения. Он предоставляет команды и свойства, необходимые для взаимодействия с пользователем и выполнения различных операций шифрования. Этот класс используется в диалоговом окне шифрования файлов, позволяя пользователям выбрать метод шифрования (по паролю или открытому ключу), задать необходимые параметры (ключ или пароль), и выполнить операцию шифрования. Также он обеспечивает обработку ошибок и проверку корректности введенных данных, таких как совпадение паролей.

Диаграмма иллюстрирует структуру класса, показывая его поля, конструкторы, свойства и методы, что помогает понять внутреннюю логику и функциональность «EncryptionDialogueViewModel» в контексте приложения «ParallelEncryptionModule».

На рис. 2 и 3 представлены элементы пользовательского интерфейса приложения.

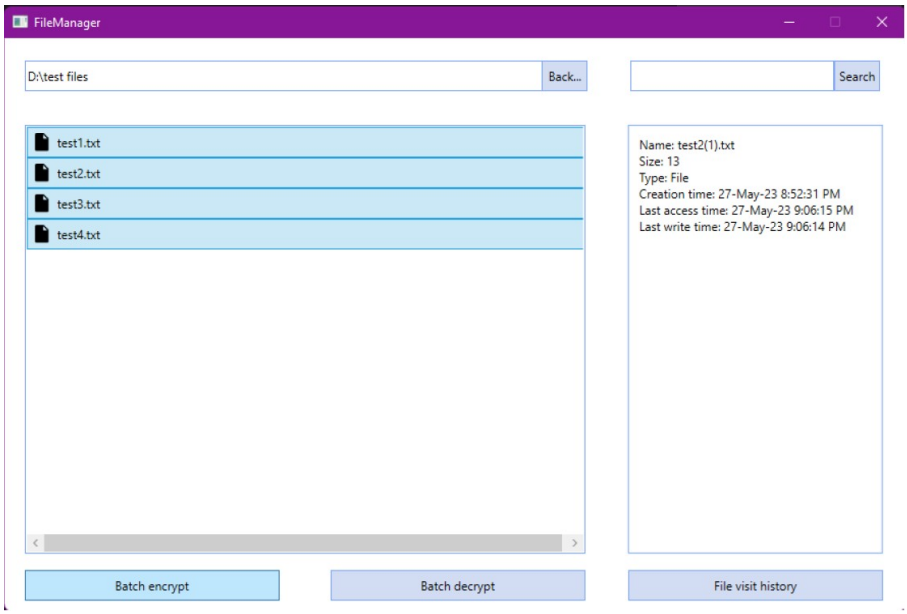

Рисунок 2 - Главное окно файлового менеджера с отображением контекстного меню шифрования и дешифрования

DOI: https://doi.org/10.60797/IRJ.2024.146.31.2

Рисунок 3 - Окно шифрования файла
DOI: https://doi.org/10.60797/IRJ.2024.146.31.3

Тестирование и анализ результатов

Тестирование разработанного параллельного модуля для криптографической защиты файлов проводилось с целью оценки его производительности, надежности и безопасности. Методология тестирования включала несколько этапов. На этапе планирования тестирования были определены цели и задачи тестирования, а также составлен план тестирования. Далее были разработаны тестовые случаи, представляющие собой сценарии тестирования для проверки функциональности модуля. На этапе проведения тестов тестовые сценарии были выполнены на реальных данных. Затем проведен анализ результатов, включающий сравнение полученных результатов с ожидаемыми значениями. В случае выявления дефектов проводилось их устранение и повторное тестирование.

Основные цели тестирования включали проверку функциональности, надежности, производительности и безопасности разработанного модуля.

Загрузка и отображение файлов и папок проводилось с использованием различных наборов данных. Модуль успешно загружал и корректно отображал файлы и папки, что подтверждало его функциональность и надежность.

Навигация по файловой системе была протестирована на различных структурах каталогов. Модуль обеспечивал быструю и корректную навигацию, что подтверждало удобство использования и эффективность интерфейса.

В проекте «ParallelEncryptionModule» для реализации параллельного модуля шифрования файлов использованы различные подходы для распараллеливания. Задачи шифрования и дешифрования файлов разделены на независимые потоки выполнения. Каждый поток отвечает за обработку определенного участка данных или выполнение конкретной операции, что позволяет использовать преимущества многопоточности и распараллеливать вычисления, увеличивая производительность и эффективность модуля. Для предотвращения возможных конфликтов и гонок при доступе к общим ресурсам, таким как файлы или память, применяются механизмы синхронизации, такие как мьютексы, семафоры или блокировки, что гарантирует корректное взаимодействие между потоками и защиту общих ресурсов от некорректного доступа.

Данные и задачи распределяются между потоками таким образом, чтобы сбалансировать нагрузку и обеспечить равномерное использование ресурсов. Например, при шифровании нескольких файлов они разделяются между потоками, чтобы каждый поток обрабатывал свой блок данных. Используются механизмы управления потоками и пулом потоков для эффективной работы с потоками, что позволяет контролировать создание, выполнение и завершение потоков, а также повторное использование потоков для минимизации накладных расходов на создание и уничтожение потоков. При параллельной обработке данных предусмотрена обработка ошибок и исключений. В случае возникновения ошибок, таких как неправильные данные или ошибки в работе алгоритмов шифрования, они корректно обрабатываются и предоставляется информация об ошибке пользователю.

Применение метода «Parallel.ForEach» для параллельного выполнения операций шифрования и дешифрования файлов (рис. 4). Метод «Parallel.ForEach» позволяет выполнять итерации параллельно, используя все доступные процессорные ресурсы [7], [8], [9]. В коде задается максимальная степень параллелизма, равная количеству процессоров в системе, что позволяет максимально эффективно использовать доступные вычислительные ресурсы.

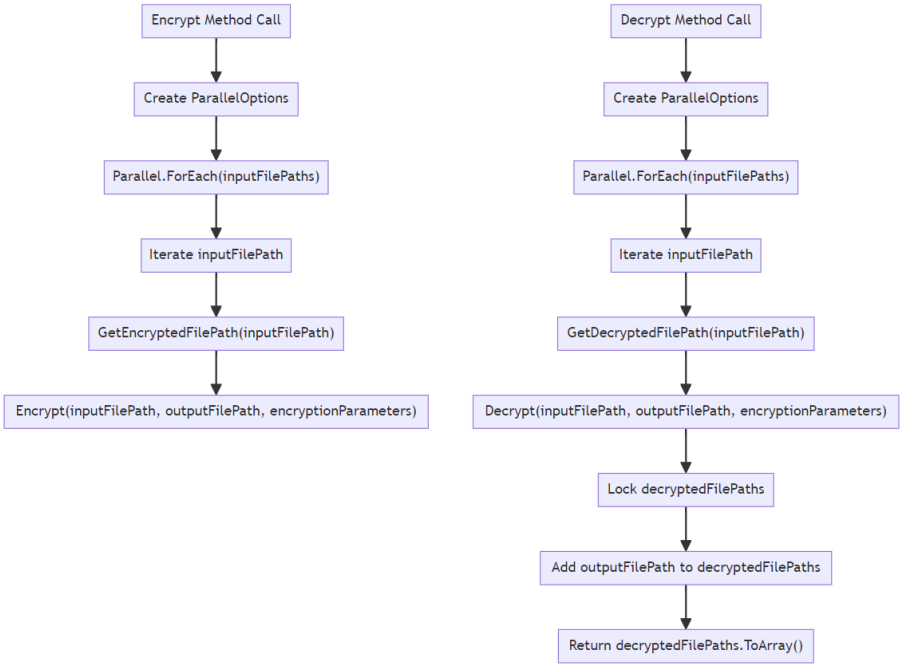

Рисунок 4 - Параллельные процессы шифрования и дешифрования
DOI: https://doi.org/10.60797/IRJ.2024.146.31.4

Эти подходы обеспечивают высокую производительность и надежность модуля, а также позволяют эффективно обрабатывать большие объемы данных.

Тестирование шифрования и дешифрования файлов проводилось с использованием алгоритмов AES, TripleDES и Twofish. Результаты показали высокую производительность и надежность модуля. Параллельное шифрование и дешифрование работало в среднем в 2.81 раза быстрее, чем последовательное шифрование, а параллельное дешифрование – в 2.57 раза быстрее.

Функция удаления файлов была протестирована на различных наборах данных. Модуль успешно удалял файлы, что подтверждало его функциональность и удобство использования.

Для сравнения разработанного модуля с существующими решениями были рассмотрены несколько популярных программ для шифрования файлов. Модуль продемонстрировал конкурентоспособные характеристики по следующим критериям: функциональность, удобство использования, безопасность и производительность. Модуль обладает всеми необходимыми функциональными возможностями для шифрования и дешифрования файлов, а также расширенными функциями. Он имеет интуитивно понятный интерфейс и удобные средства навигации. Модуль использует современные алгоритмы шифрования и методы защиты ключей. Он обеспечивает высокую производительность при шифровании и дешифровании файлов.

Тестирование проводилось на аппаратно-программной платформе, включающей процессор Intel Core i5 10400 [10], 4 модуля оперативной памяти AMD Radeon R7 Performance Series R748G2606U2S-U DDR4 – 8 ГБ 2666 и хранилище данных Samsung SSD 860 EVO 500GB M.2.

Анализ производительности и безопасности показал, что время выполнения последовательного и параллельного шифрования и дешифрования файлов различного размера значительно улучшилось при использовании параллельного модуля. Приложение эффективно использовало системные ресурсы при выполнении операций шифрования и дешифрования пяти файлов объемом 1 ГБ каждый, что подтверждало эффективность работы параллельных операций. Оценка стойкости алгоритмов шифрования показала, что используемые алгоритмы обладают высокой стойкостью и не содержат известных уязвимостей. Это гарантировало надежную защиту шифрованных данных от несанкционированного доступа и восстановления исходных данных без соответствующего ключа.

Результаты анализа производительности и безопасности подтверждают, что разработанный модуль обладает высокой производительностью и надежной криптографической защитой, соответствуя требованиям к современным системам обработки данных.

Заключение

В данной статье была рассмотрена разработка параллельного модуля для криптографической защиты файлов с использованием алгоритмов AES, TripleDES и Twofish. Проведенные исследования и тестирования подтвердили высокую производительность и надежность разработанного модуля. Использование параллельных вычислений позволяет значительно ускорить процесс шифрования и дешифрования данных, что особенно актуально в условиях современного мира, где защита информации становится первостепенной задачей.

Разработанный модуль успешно интегрируется с файловым менеджером, обеспечивая удобный пользовательский интерфейс и эффективное взаимодействие с файлами. Он предоставляет все необходимые функциональные возможности для шифрования и дешифрования файлов, а также дополнительные функции для удобства использования и управления ключами.

Анализ производительности и безопасности показал, что параллельное шифрование и дешифрование работают в среднем в 2.81 раза быстрее, чем последовательное шифрование, а параллельное дешифрование – в 2.57 раза быстрее. Это подтверждает эффективность использования параллельных вычислений в криптографии. Тестирование проводилось на современном аппаратно-программном обеспечении, что также свидетельствует о применимости модуля в реальных условиях.

Перспективы дальнейшего развития включают интеграцию дополнительных алгоритмов шифрования и оптимизацию существующих для еще большей производительности и надежности. Также возможно расширение функциональности модуля, включая разработку новых методов управления ключами и улучшение интерфейса пользователя. Продолжение исследований в области параллельных вычислений и их применения в криптографии позволит создать еще более эффективные и безопасные системы для защиты данных в будущем.

Конфликт интересов

Не указан.

Рецензия

Гибадуллин Р.Ф., Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ, Казань, Российская Федерация
DOI: https://doi.org/10.60797/IRJ.2024.146.31.5

Conflict of Interest

None declared.

Review

Gibadullin R.F., Kazan National Research Technical University named after A.N. Tupolev – KAI, Kazan, Russian Federation
DOI: https://doi.org/10.60797/IRJ.2024.146.31.5

Список литературы / References

1. Borhan R. Successful implementation of AES algorithm in hardware / R. Borhan, R.M. Fuad Tengku Aziz // 2012 IEEE International Conference on Electronics Design, Systems and Applications (ICEDSA). — Kuala Lumpur, 2012. — P. 27-32. — DOI: 10.1109/ICEDSA.2012.6507810.

2. Zheng X. Research for the application and safety of MD5 algorithm in password authentication / X. Zheng, J. Jin // 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery. — Chongqing, 2012. — P. 2216-2219. — DOI: 10.1109/FSKD.2012.6234010.

3. Putri Ratna A.A. Analysis and comparison of MD5 and SHA-1 algorithm implementation in Simple-O authentication based security system / A.A. Putri Ratna, P. Dewi Purnamasari, A. Shaugi [et al.] // 2013 International Conference on QiR. — Yogyakarta, 2013. — P. 99-104. — DOI: 10.1109/QiR.2013.6632545.

4. Wu R. A High-Performance Parallel Hardware Architecture of SHA-256 Hash in ASIC / R. Wu, X. Zhang, M. Wang [et al.] // 2020 22nd International Conference on Advanced Communication Technology (ICACT). — Phoenix Park, 2020. — P. 1242-1247. — DOI: 10.23919/ICACT48636.2020.9061457.

5. Dulla G.L. An Enhanced BlowFish (eBf) Algorithm for Securing x64FileMessage Content / G.L. Dulla, B.D. Gerardo, R.P. Medina // 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM). — Baguio City, 2018. — P. 1-6. — DOI: 10.1109/HNICEM.2018.8666434.

6. Dibas H. A comprehensive performance empirical study of the symmetric algorithms:AES, 3DES, Blowfish and Twofish / H. Dibas, K.E. Sabri // 2021 International Conference on Information Technology (ICIT). — Amman, 2021. — P. 344-349. — DOI: 10.1109/ICIT52682.2021.9491644.

7. Гибадуллин Р.Ф. Потокобезопасные вызовы элементов управления в обогащенных клиентских приложениях / Р.Ф. Гибадуллин // Программные системы и вычислительные методы. — 2022. — № 4. — С. 1-19. — DOI: 10.7256/2454-0714.2022.4.39029.

8. Викторов И.В. Разработка синтаксического дерева для автоматизированного транслятора последовательного программного кода в параллельный код для многоядерных процессоров / И.В. Викторов, Р.Ф. Гибадуллин // Программные системы и вычислительные методы. — 2023. — № 1. — С. 13-25. — DOI: 10.7256/2454-0714.2023.1.38483.

9. Гибадуллин Р.Ф. Неоднозначность результатов при использовании методов класса Parallel в рамках исполняющей среды .NET Framework / Р.Ф. Гибадуллин, И.В. Викторов // Программные системы и вычислительные методы. — 2023. — № 2. — С. 1-14. — DOI: 10.7256/2454-0714.2023.2.39801.

10. Cebrián J.M. Improving Energy Efficiency through Parallelization and Vectorization on Intel Core i5 and i7 Processors / J.M. Cebrián, L. Natvig, J.C. Meyer [et al.] // 2012 SC Companion: High Performance Computing, Networking Storage and Analysis. — Salt Lake City, 2012. — P. 675-684. — DOI: 10.1109/SC.Companion.2012.93.

Список литературы на английском языке / References in English

1. Borhan R. Successful implementation of AES algorithm in hardware / R. Borhan, R.M. Fuad Tengku Aziz // 2012 IEEE International Conference on Electronics Design, Systems and Applications (ICEDSA). — Kuala Lumpur, 2012. — P. 27-32. — DOI: 10.1109/ICEDSA.2012.6507810.

2. Zheng X. Research for the application and safety of MD5 algorithm in password authentication / X. Zheng, J. Jin // 2012 9th International Conference on Fuzzy Systems and Knowledge Discovery. — Chongqing, 2012. — P. 2216-2219. — DOI: 10.1109/FSKD.2012.6234010.

3. Putri Ratna A.A. Analysis and comparison of MD5 and SHA-1 algorithm implementation in Simple-O authentication based security system / A.A. Putri Ratna, P. Dewi Purnamasari, A. Shaugi [et al.] // 2013 International Conference on QiR. — Yogyakarta, 2013. — P. 99-104. — DOI: 10.1109/QiR.2013.6632545.

4. Wu R. A High-Performance Parallel Hardware Architecture of SHA-256 Hash in ASIC / R. Wu, X. Zhang, M. Wang [et al.] // 2020 22nd International Conference on Advanced Communication Technology (ICACT). — Phoenix Park, 2020. — P. 1242-1247. — DOI: 10.23919/ICACT48636.2020.9061457.

5. Dulla G.L. An Enhanced BlowFish (eBf) Algorithm for Securing x64FileMessage Content / G.L. Dulla, B.D. Gerardo, R.P. Medina // 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM). — Baguio City, 2018. — P. 1-6. — DOI: 10.1109/HNICEM.2018.8666434.

6. Dibas H. A comprehensive performance empirical study of the symmetric algorithms:AES, 3DES, Blowfish and Twofish / H. Dibas, K.E. Sabri // 2021 International Conference on Information Technology (ICIT). — Amman, 2021. — P. 344-349. — DOI: 10.1109/ICIT52682.2021.9491644.

7. Gibadullin R.F. Potokobezopasnye vyzovy jelementov upravlenija v obogashhennyh klientskih prilozhenijah [Thread-safe calls of control elements in enriched client applications] / R.F. Gibadullin // Programmnye sistemy i vychislitel'nye metody [Software Systems and Computational Methods]. — 2022. — № 4. — P. 1-19. — DOI: 10.7256/2454-0714.2022.4.39029. [in Russian]

8. Viktorov I.V. Razrabotka sintaksicheskogo dereva dlja avtomatizirovannogo transljatora posledovatel'nogo programmnogo koda v parallel'nyj kod dlja mnogojadernyh processorov [Development of syntax tree for automated translator of sequential program code into parallel code for multicore processors] / I.V. Viktorov, R.F. Gibadullin // Programmnye sistemy i vychislitel'nye metody [Software Systems and Computational Methods]. — 2023. — № 1. — P. 13-25. — DOI: 10.7256/2454-0714.2023.1.38483. [in Russian]

9. Gibadullin R.F. Neodnoznachnost' rezul'tatov pri ispol'zovanii metodov klassa Parallel v ramkah ispolnjajushhej sredy .NET Framework [Ambiguity of results when using methods of Parallel class within the framework of executable environment .NET Framework] / R.F. Gibadullin, I.V. Viktorov // Programmnye sistemy i vychislitel'nye metody [Programme Systems and Computational Methods]. — 2023. — № 2. — P. 1-14. — DOI: 10.7256/2454-0714.2023.2.39801. [in Russian]

10. Cebrián J.M. Improving Energy Efficiency through Parallelization and Vectorization on Intel Core i5 and i7 Processors / J.M. Cebrián, L. Natvig, J.C. Meyer [et al.] // 2012 SC Companion: High Performance Computing, Networking Storage and Analysis. — Salt Lake City, 2012. — P. 675-684. — DOI: 10.1109/SC.Companion.2012.93.