ИЗМЕРЕНИЕ СКОРОСТИ РАБОТЫ ТРАНСПОЗИЦИОННОГО ШИФРОВАНИЯ НА ПРИМЕРЕ PYTHON-ПРИЛОЖЕНИЯ

Ахмедова Ирода Нурмухаммедовна[1], Валимов Мунир Данияровичч[2]

[1,2] Ташкентский Международный университет образования (TIUE)
E-mail: [1] akhmedovairoda11@gmail.com; [2] munirvalimov10@gmail.com
DOI: 10.61587/mmit.tiue.uz.v1i1.391

Аннотация. В данной статье рассматривается практическая реализация процесса шифрования методом табличной перестановки с использованием языка программирования Python. В программной реализации алгоритма особое внимание уделено особенностям его функционирования при обработке текстовой информации различного объёма, а также анализу зависимости скорости выполнения операций шифрования от объёма входных данных и параметров ключа, определяющих структуру перестановки. В работе исследуется влияние изменения указанных параметров на производительность алгоритма, что позволяет выявить закономерности его поведения при различных условиях использования.

Ключевые слова: *криптография, шифрование, табличная перестановка, Python, кибербезопасность, замеры производительности.*

MEASURING THE SPEED OF TRANSPOSITION ENCRYPTION USING A PYTHON APPLICATION

Akhmedova Iroda Nurmukhamedovna[1], Valimov Munir Daniyarovich[2]

[1,2] Tashkent International University of Education (TIUE)
E-mail: [1] akhmedovairoda11@gmail.com; [2] munirvalimov10@gmail.com

Abstract. This article examines the practical implementation of a tabular permutation encryption process using the Python programming language. The software implementation of the algorithm focuses on its performance when processing text data of varying sizes, as well as on analyzing the dependence of encryption speed on the input data size and the key parameters that determine the permutation structure. The paper examines the impact of varying these parameters on the algorithm's performance, allowing us to identify patterns in its behavior under various operating conditions.

Keywords: *cryptography, encryption, table permutation, Python, cybersecurity, performance benchmarking.*

Introduction (Введение)

Стремительное развитие информационных технологий создает особую актуальность в обеспечении защиты данных при их хранении и передаче. Важность защиты данных с точки зрения В.В.Арутюнова, изложена как, опасность нарушения конфиденциальности - существует, когда ценность информации теряется при ее раскрытии [1,с.18]. Это подтверждает о связанности нарушения конфиденциальности с несанкционированным доступом к данным. Наиболее востребованным направлением защиты информации является криптография – наука о методах и способах преобразования информации для предотвращения от несанкционированного доступа.

На ряду современных криптографических стандартов, классические методы шифрования используются в образовательной и исследовательской практике. Одним из них является методика перестановки букв в сообщении, представляет собой один из самых старинных способов скрытия смысла текста от посторонних. Несмотря на слабую защищённость по сравнению с современными стандартами, такие алгоритмы активно применяются в обучении студентов IT-специальностей благодаря наглядности и лёгкости понимания.

Materials and Methods (Материалы и методы)

В своих трудах Бабаш А.В., Шанкин Г.П. приводят интересные исследования из истории создания криптографии, как «Одно из основных понятий криптографии - шифр - имеет корни в арабском слове «цифра». Одна из первых крупных книг, в которой содержательно описывается криптография - это труд, созданный Абу Вакр Ахмед бен Али бен Вахшия ан-Набати - «Книга о большом стремлении человека разгадать загадки древней письменности». В ней описано несколько систем шифров» [2, с.23-24].

Метод шифрования **«Шифр табличной маршрутной перестановки»** основана на изменении порядка символов в сообщении без изменения самих символов. Тип данного алгоритма является примером симметричного шифрования и используется для изучения принципов преобразования текстовой информации. В своем труде К.Бабенко дает определение «*Шифр* - это множество обратимых преобразований формы сообщения с целью его защиты от несанкционированного прочтения. Исходное сообщение, которое подвергается шифрованию, называется *открытым текстом*, а результат, полученный применением преобразования шифра к исходному сообщению, называется *шифртекстом* или криптограммой. Переход от открытого текста к шифртексту называется *зашифрованием*, а обратный переход – *расшифрованием*»[3,с.6].

Алгоритмы шифрования используются по методам симметричного и асимметричного шифрования. Метод **«Шифр табличной маршрутной перестановки»** числится в метод с одним ключом. В изложении Панасенко С.П. дает определение «Алгоритмы симметричного шифрования — алгоритмы шифрования, в которых для зашифровывания и расшифровывания используется один и тот же ключ, или ключ расшифровывания легко вычисляется из ключа зашифровывания и наоборот[4,с.4]».

«Простейший вариант перестановки — прямоугольная таблица с секретным размером столбца, куда исходный текст записывается по столбцам, а шифрсообщение считывается по строкам»[3,с.11]. В этом сказанном есть интересный момент о размере столбца, который может достичь достаточно больших размеров в связи с объемом данных информации. Таким образом, в задаче исследования было поставлено, чтобы в разработанной программе на Питоне, которая шифрует сообщения путём перемешивания колонок в таблице, провести серию практических замеров её быстродействия. Особой важностью представляло выяснение того, как меняется время работы при росте размера текста и увеличении числа колонок в ключе. Дополнительно требовалось проверить качество кода и найти возможные ошибки.

Results (Результаты)

В работе рассмотрено исследование производительности приложения при обработке текстов различного объёма в реализации программного кода по алгоритму «**Шифр табличной маршрутной перестановки**» на языке Python. Проведен анализ влияния длины ключа и размера входных данных на скорость выполнения операций шифрования.

Алгоритм метода приведен на примере шифрования фразы "HELLO WORLD!" Определяется число, например, 4 — это будет ширина таблицы, который представляет количество столбцов. Буквы вписывают в таблицу построчно слева направо (Рис.1,а).

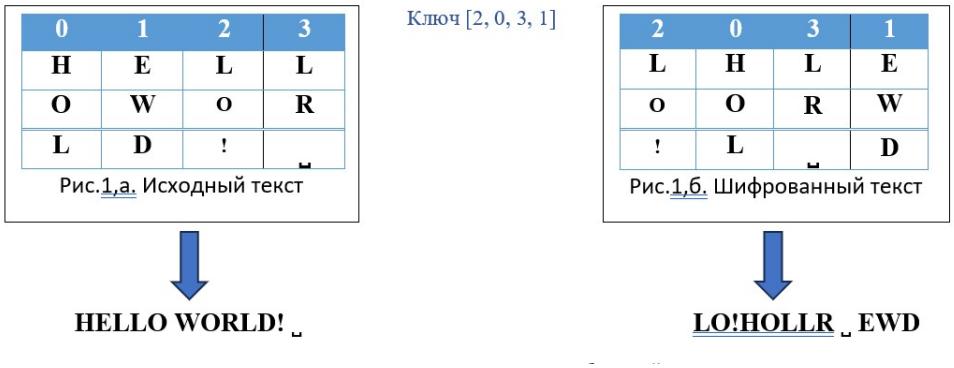

Рисунок 1. Визуализация процесса табличной перестановки

В данном алгоритме КЛЮЧ - порядок чтения колонок. Например, [2, 0, 3, 1]. В переставленных столбцах текст нужно читать сверху вниз и записать в строку зашифрованный текст (Рис.1,б).

В исходном коде программы на Python присутствуют три главные функции. Первая (**chunk_text**) делит входной текст на куски нужной длины, дописывая пробелы - " ␣ ", если последний кусок получается короче. Вторая (**columnar_encrypt**) проводит само шифрование: проходит по ключу и собирает буквы из соответствующих колонок. Третья (**columnar_decrypt**) выполняет обратную операцию — восстанавливает исходный текст из зашифрованного набора символов.

В целях исключения влияния разных конфигураций компьютеров на результаты, и для обеспечения их достоверности, все измерения выполнены в Python версии 3.12 в операционной системе Ubuntu 24.04. Подсчёт времени осуществляется с применением встроенной библиотеки **time**. Каждый тест повторяется многократно (от 50 до 100 раз в зависимости от размера данных), после окончания испытаний статистически определяется среднее значение, таким образом можно снизить случайные погрешности.

В исследовании проведены экспериментальные замеры производительности, включающие оценку времени выполнения операций при варьировании входных данных. Полученные результаты позволили определить наиболее эффективные подходы к программной реализации криптографических функций, а также выявить факторы, оказывающие наибольшее влияние на скорость работы алгоритма.

В ходе экспериментов было исследовано влияние двух переменных: длины ключа и объема текста. Результаты первого эксперимента (текст 10 000 символов, ключ от 5 до 30 колонок) показали, что ширина таблицы практически не влияет на скорость шифрования. Время колебалось в диапазоне от 0,43 до 0,59 мс. Это объясняется тем, что основные затраты ресурсов в Python уходят не на итерацию по ключу, а на операции конкатенации строк. Результаты второго эксперимента (фиксированный ключ, текст от 100 до 20 000 символов) продемонстрировали линейный рост времени обработки (Табл.3).

Линейная зависимость подтверждает эффективность алгоритма для обработки больших массивов текстовых данных.

Таблица 3. Результаты

Количество символов	Среднее время (мс)	Множитель
100	0.006	×1
500	0.024	×4
1000	0.060	×10
5000	0.230	×38
10000	0.947	×158
20000	0.910	×152

Discussion (Обсуждение/Интерпретация)

Анализ показал, что алгоритм шифрования перестановкой обладает высокой скоростью работы и линейной сложностью. Наилучшая производительность достигается при минимизации количества операций пересоздания строк в памяти. Полученные данные могут быть использованы студентами при изучении основ криптографии и разработчиками при создании легковесных систем защиты данных.

Список литература

1. Арутюнов В.В. Основы информационной безопасности: Учебное пособие. — М.: МФА, 2008. — 178 с.
2. Бабаш А.В., Шанкин Г.П. Криптография. Под редакцией В.П. Шерстюка, ЭЛ. Применко / А.В. Бабаш, Г.П. Шанкин. — М.: СОЛОН-ПРЕСС, 2007. — 512 с. — (Серия книг «Аспекты защиты»). 15ВЫ 5-93455-135-3
3. Составитель Л.К.Бабенко. Введение в специальность «Организация и технология защиты информации». Таганрог: Изд-во ТРТУ, 1999. 54с.
4. Панасенко С.П. Алгоритмы шифрования. Специальный справочник. — СПб.: БХВ-Петербург, 2009. — 576 с.: ил. ISBN 978-5-9775-0319-8
5. Официальная документация Python 3.12 — раздел модуля time. URL:https://docs.python.org/3/library/time.html