**Исследовательская работа**

**на тему:** Алгоритм шифрования на основе элементарного клеточного автомата

**Автор работы:**

**Обучающийся 8 класса**

**ГБОУ Гимназия №1538**

Чистобаев Андрей Андреевич

**Руководитель работы:**

**Консультант**

Носкова Людмила Николаевна,

Преподаватель математики, высшая категория

**Москва**

**2017 год**

# Содержание

## Введение…………………………………………………………………………3

## Критерии оценки……………………………………………………………….4

## Концепция……………………………………………………………………….5

## Ресурсы…………………………………………………………………………..7

## План выполнения………………………………………………………………8

## Реализация………………………………………………………………………9

## Оценка итогов работы…………………………………………………….….11

## Литература……………………………………………………………………..12

## Приложение. О работе элементарных клеточных автоматов……..……13

# Постановка проблемы

Сохранение конфиденциальности данных пользователей очень важно в нашем информационном обществе. Для обеспечения безопасности используются криптографические алгоритмы и хэш-функции. Но почему мы должны доверять этим инструментам, которыми подчистую пользуются крупные компании, не спросив нашего мнения?

Доверие к используемому инструменту зависит от его надёжности, которая определяется множеством тестов, созданных профессионалами. В криптографии для описания надёжности используется термин криптостойкость.

Поэтому, перед использованием алгоритма необратимого шифрования, основанного на элементарном клеточном автомате, на своём сайте, я решил провести полноценное тестирование самостоятельно. Для того чтобы тесты были действительно полноценны и всесторонне описывали алгоритм, я ознакомился с основами криптографии и криптоанализа при помощи замечательной книги Брюса Шнайера, «Прикладная криптография».

Если понимание некоторых терминов вызвало у вас затруднение, то милости прошу в приложение 2. В первом приложении вы найдёте информацию о работе элементарных клеточных автоматов. В третьем описана работа самого алгоритма. Четвёртое содержит список всех литературных источников.

Для полноценного понимания хода исследования рекомендуется читать текст работы последовательно, от начала до конца. Приятного чтения.

# Критерии оценки

Как понять, что работа выполнена успешно? Что для нас есть успех? Нужно дать определение.

Моя работа считается успешно выполненной, когда в её итоге выбранный мною алгоритм шифрования будет использован на сайте. То есть я собираюсь не только протестировать его, но и при необходимости модифицировать.

При криптоанализе стоит считать, что злоумышленнику известен наш алгоритм. Поэтому в программе-тестировщике я допускаю использование своей реализации алгоритма. Но имеются и другие данные: открытый текст и шифротекст. В соответствии с известными данными выделяют 4 основных метода криптоанализа:

1. Атака на основе шифротекста
2. Атака на основе открытых текстов и соответствующих шифротекстов
3. Атака на основе подобранного открытого текста
4. Атака на основе адаптивно подобранного открытого текста

Моя реализация должна быть подвергнута всем этим видам атак. На основе результатов попыток взлома я решу, что стоит переделать, а что добавить. Криптостойкость конечной версии я сравню с криптостойкостью других аналогичных алгоритмов шифрования.

Так же реализация должна быть переносимой и производительной. Производительность конечной версии будет сравнена как с производительностью других алгоритмов, так и с производительностью предыдущих версий. Переносимости я добьюсь при помощи использования лишь стандартных средств, поддерживаемых на всех устройствах, от умных утюгов до персональных компьютеров.

# Концепция

И так, что же предстоит сделать? Я должен реализовать алгоритм необратимого шифрования, основанный на элементарном клеточном автомате, и использовать его. Но перед использованием нужно убедиться, что он обладает приемлемой криптостойкостью.

Для её определения необходимо написать тест. Этот тест я решил разбить на 4 этапа, в соответствии с основными методами криптоанализа.

Тест будет проведён как на персональном компьютере с Windows 7, так и на смартфоне под управлением Android, чем при успешном запуске будет доказана переносимость реализации.

Для создания программы мною был выбран язык программирования C++, так как он удовлетворяет всем вышеописанным требованиям. Правильно написанные программы на этом языке как переносимы, так и производительны, но самое главное то, что они наглядны. Чтобы достичь ещё большей наглядности, программа написана в объектно-ориентированном стиле, что почти не сказалось на её быстродействии.

Примерный план каждого этапа:

1. Сохранение текущего процессорного времени
2. Попытка взлома (подбором)
3. Сохранение текущего процессорного времени
4. Вычисление и вывод времени работы
5. Вывод результатов взлома (возможные ключи)

Конечно же, весь код, нафаршированный комментариями, будет выложен в интернете на всеобщее обозрение. Тем самым я хочу узнать мнение простых людей и профессионалов.

# Ресурсы

Что мне понадобится для реализации идеи? Приведу подробный список с пояснениями.

Во-первых, мне понадобится персональный компьютер и смартфон с соответствующими OS, необходимые для написания теста и его запуска. Для этого на них должно быть установлено дополнительное ПО.

Во-вторых, как я напишу программу на C++ без знания языка? Для этого необходимо ознакомиться с его основами. С этим мне помогли книги замечательных авторов - Бьярне Страуструпа и Стефана Рэнди Дэвиса.

В-третьих, тест должен быть полным и всеобъемлющим. Чтобы создать его, нужно ознакомиться с основами криптоанализа. Для этого идеально подходит книга Брюса Шнайера, «Прикладная криптография».

Также стоит ознакомиться и с основами работы элементарных клеточных автоматов. В этом деле помощником мне был интернет.

# План выполнения

1. Ознакомиться с основами работы элементарных клеточных автоматов
2. Узнать основы криптографии и криптоанализа
3. Ознакомиться с алгоритмом шифрования
4. Обучиться основам языка программирования C++
5. Реализовать алгоритм в виде программы
6. Провести тестирование реализации
7. Исправить начальные недочёты
8. Провести сравнение с аналогичными алгоритмами
9. Выложить реализацию в свободный доступ
10. Начать использование алгоритма на своём сайте

В целом, получился вполне обычный цикл разработки. Но на деле всё шло не так гладко, и до 9 пункта ещё так далеко, как до Луны пешком. Какие проблемы появились в ходе работы? Узнаем далее.

# Реализация

Ещё до начала тестирования выяснились некоторые проблемы, возникшие из-за использования в алгоритме элементарных клеточных автоматов.

* **Проблема №1.** Маленький диапазон значений ключа.

Слово «элементарный» говорит само за себя. Существует лишь 256 правил перехода состояний, то есть 256 возможных значений ключа. Поэтому ключ легко найти подбором.

* **Проблема №2.** Коллизии.

При использовании практически любого ключа происходят коллизии. Есть исключения, но и они не идеальны. При использовании ключа 204 исходная строка никак не изменяется. При использовании ключа 240 код символа делится на 2. Всё гениальное просто, главное избавились от коллизий!

Эти проблемы оказались очень важными, и оставить их без внимания не удастся. Для определения проблемы №2 мне пришлось написать дополнительный тест.

Проблема первая решается очень даже просто. Нужно лишь использовать для каждого символа открытого текста новый ключ. Для достижения ещё большей надёжности стоит просчитывать несколько поколений. А ещё лучше использовать случайный ключ. Чувствуете, как медленно и аккуратно подступает шизофрения?

Вторую проблему уже никак не исправить, можно лишь использовать ту парочку бестолковых ключей, которую я описал выше. Это коренной недочёт всего алгоритма, поэтому и в правду поделать ничего уже нельзя, как бы это ни было печально.

На следующей странице перейдём к обзору одного из этапов теста. Исходный код остальных тестов можете найти по ссылке, данной в списке литературы.

**Этап 2. Атака на основе открытых текстов и соответствующих им шифротекстов.**

На самом деле же пара нам понадобится лишь одна. Из-за всё того же маленького диапазона возможных значений ключа. Рассмотрим алгоритм взлома подробнее, с учётом того, что все символы исходной строки шифруются одним ключом.

**Дано:**

* Открытый текст
* Шифротекст, ему соответствующий

**Решение:**

Решение тривиально – мы узнаем ключ простым подбором.

Каждый символ исходной строки шифруется подряд с использованием всех 256 возможных ключей. При этом нам неизвестно количество просчитанных поколений при шифровке, поэтому при помощи каждого ключа мы шифруем по 100 поколений, на каждом из которых результат сравнивается с символом из данного шифротекста. Все ключи, при которых символы совпадут, сохраняются в вектор, который у каждого символа свой. Потом мы находим пересечение этих векторов. Каждый ключ из этого пересечения равновероятен.

**Критика.**

Почему я написал «каждый ключ», если он только 1? Потому что некоторые поколения при использовании различных правил перехода состояний могут совпадать. Взгляните на дано. На этапе взлома количество просчитываемых поколений неизвестно. Из-за этого и возникает такая неопределённость результата при простом подборе.

Но если возможны 2 или более ключа, значит, они работают одинаково? Конечно, нет. Совпадение возникает лишь при таких данных.

**Зачем нам нужен ключ, если шифрование необратимое?**