Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Исследование ассиметричных шифров

Студент: Валдайцев А. Д.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М. Г.

Минск 2023

# Криптоалгоритм на основе задачи об укладке ранца

Ранцевый (рюкзачный) вектор ***S*** = (*s1*, ..., *sz*) – это упорядоченный набор из *z*, *z* ≥ 3, различных натуральных чисел *si*. Входом задачи о ранце (рюкзаке) называем пару (***S***, *S*), где ***S*** – рюкзачный вектор, а *S* – натуральное число.

Решением для входа (***S****, S*) будет такое подмножество из ***S***, сумма элементов которого равняется *S*.

В наиболее известном варианте задачи о ранце требуется выяснить, обладает или нет данный вход (***S****, S*) решением. В варианте, используемом в криптографии, нужно для данного входа (***S****, S*) построить решение, зная, что такое решение существует. Оба эти варианта являются NP-полными. Имеются также варианты этой задачи, которые не лежат даже в классе NP.

Суть метода для шифрования состоит в том, что существуют две различные задачи укладки ранца: одна из них решается легко и характеризуется линейным ростом трудоемкости, а другая решается трудно. Легкий для укладки ранец можно трансформировать в трудный.

Трудный для укладки ранец применяется в качестве открытого ключа, который легко использовать для зашифрования, но невозможно – для расшифрования. В качестве закрытого ключа применяется легкий для укладки ранец, который предоставляет простой способ расшифрования сообщения.

# Генерация сверхвозрастающей последовательности

Сверхвозрастающей называется последовательность, в которой каждый последующий член больше суммы всех предыдущих.

В качестве закрытого ключа ***d*** (легкого для укладки ранца) используется сверхвозрастающая последовательность, состоящая из *z* элементов: *d1*, *d2*, …, *dz*: ***d*** = {*di*}, *i* = 1, …, *z*.

Для генерации сверхвозрастающей последовательности реализована следующая функция, отображённая на рисунке 1.1.

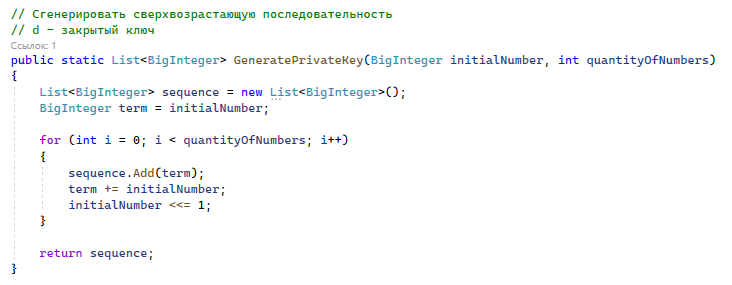


Рисунок 1.1 – Код генерации сверхвозрастающей последовательности

# Генерация нормальной последовательности

Открытый ключ ***e*** представляет собой нормальную (не сверхвозрастающую) последовательность. Он формируется на основе закрытого ключа и не позволяет легко решить задачу об укладке ранца.

Для получения открытого ключа ***e*** (***e*** = {*ei*}, *i* = 1, …, *z*) все значения закрытого ключа умножаются на некоторое число a по модулю *n*:



Значение модуля *n* должно быть больше суммы всех чисел последовательности; кроме того, НОД (*а*, *n*) = 1.

Код функции для генерации нормальной последовательности представлен на рисунке 1.2.

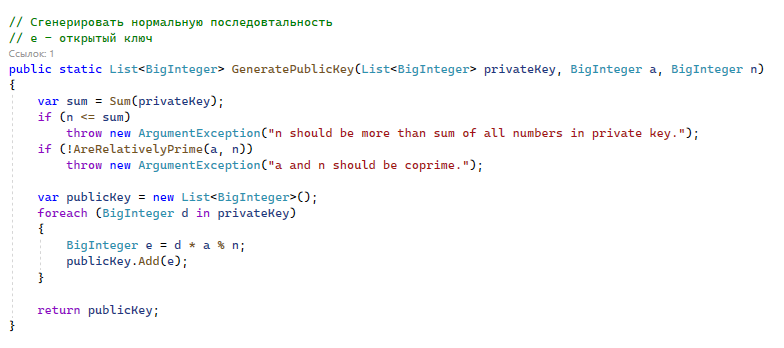


Рисунок 1.2 – Код функции генерации нормальной последовательности

# Зашифрование

Для зашифрования сообщения (*М*) оно сначала разбивается на блоки, по размерам равные числу (*z*) элементов последовательности в ранце. Затем, считая, что 1 указывает на присутствие элемента последовательности в ранце, а 0 – на его отсутствие, вычисляются полные веса рюкзаков (*Si*, *i* = 1, …, z*):* по одному ранцу для каждого блока сообщения с использованием открытого ключа получателя ***e***.

Код функции для зашифрования представлен на рисунке 1.3.

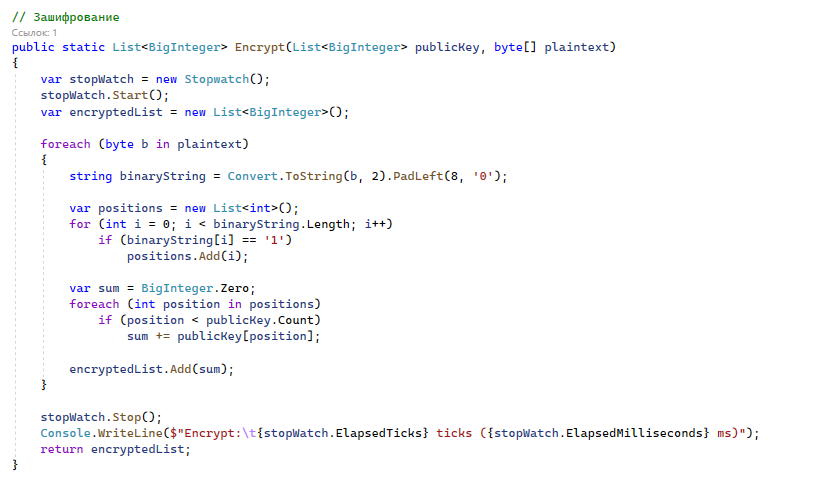


Рисунок 1.3 – Код функции зашифрования

# Расшифрование

Для расшифрования сообщения получатель (используя свой тайный ключ ***d***: сверхвозрастающую последовательность) должен сначала определить такое обратное к *а* число *а–1*, что



После определения обратного числа каждое значение шифрограммы (*ci*) преобразуется в соответствии со следующим соотношением:



Полученное на основании последней формулы для каждого блока число далее рассматривается как заданный вес ранца, который следует упаковать, используя сверхвозрастающую последовательность, то есть тайный ключ получателя.

Код функции расшифрования представлен на рисунке 1.4.



Рисунок 1.4 – Код функции расшифрования

# Время выполнения зашифрования и расшифрования

Для оценки времени выполнения операций зашифрования и расшифрования была проведена оценка с разными таблицами кодировки, и, соответственно, размером блоков шифрования, а также при подаче на вход алгоритма генерации ключей разных чисел z, которые отвечают за количество членов в ключевой последовательности.

Оценку времени операций зашифрования и расшифрования при использовании разных таблиц кодировки можно увидеть на рисунке 2.1. Разница между ними несущественная, однако, Base64 имеет тенденцию к немного более медленному зашифрованию и расшифрованию.

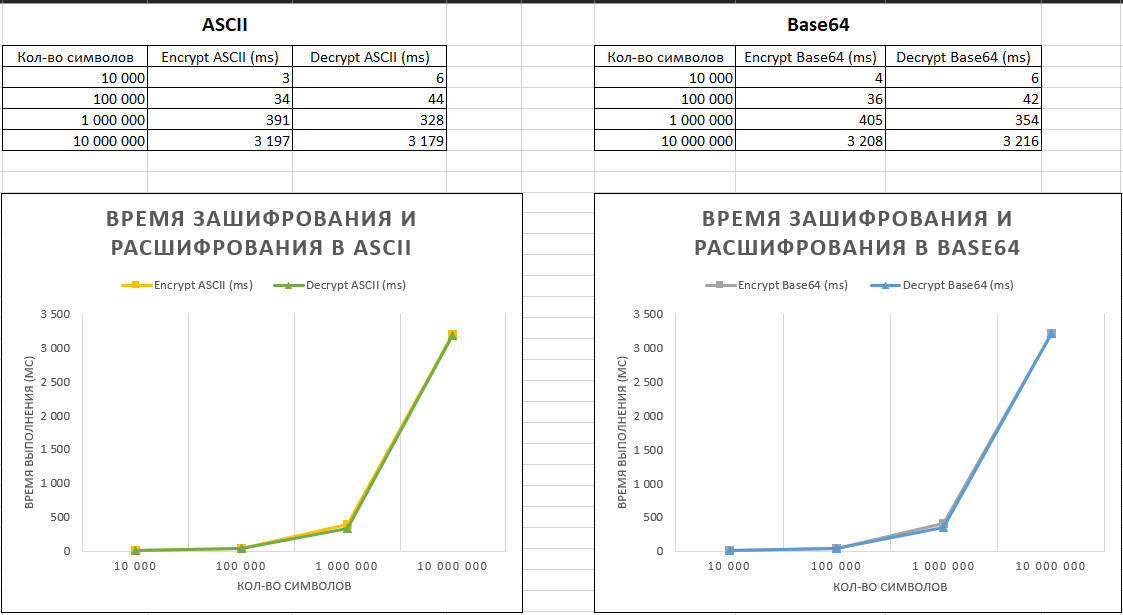


Рисунок 2.1 – Время зашифрования в ASCII и Base64

График, отображающий изменение времени зашифрования и расшифрования при разном количестве членов ключевой последовательности, представлен на рисунке 2.2.



Рисунок 2.2 – Время зашифрования при разных числах *z*

Анализируя этот график, можно заметить, что при увеличении числа членов ключевой последовательности, время зашифрования и расшифрования значительно увеличивается. Однако, необходимо учитывать, что если будет использоваться длинная ключевая последовательность и короткое сообщение, то влияние параметра *z* можно и не ощутить.

# Вывод

В данный лабораторной работе были изучены и приобретены практические навыки разработки и использования ассиметричных шифров.