Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Отчет по лабораторной работе №7**

**«Исследование асимметричных шифров»**

Выполнил:

Cтудент 3 курса 1 группы

Парибок И. А.

Вариант 5

Минск 2023

**Практическая часть**

Разработать авторское оконное приложение в соответствии с целью лабораторной работы. При этом можно воспользоваться доступными библиотеками либо программными кодами.

В основе вычислений – кодировочные таблицы Base64 и ASCII. Приложение должно реализовывать следующие операции:

• генерация сверхвозрастающей последовательности (тайного ключа); старший член последовательности – 100-битное число; в простейшем случае принимается z = 6 (для кодировки Base64) и z = 8 (для кодировки ASCII);

•вычисление нормальной последовательности (открытого ключа);

•зашифрование сообщения, состоящего из собственных фамилии, имени и отчества;

•расшифрование сообщения;

•оценка времени выполнения операций зашифрования и расшифрования.

2. Проанализировать время выполнения операций зашифрования/расшифрования при увеличении числа членов ключевой последовательности; при использовании разных таблиц кодировки.

3. Результаты оформить в виде отчета по установленным правилам.

**Выполнение задач**

Для реализации генерации сверхвозрастающей последовательности была разработана функция, представленная на рисунке 1. Данная функция Generate реализует алгоритм генерации сверхвозрастающей целочисленной последовательности. Функция принимает единственный аргумент z - длину генерируемой последовательности, и возвращает новый массив целых чисел соответствующей длины.

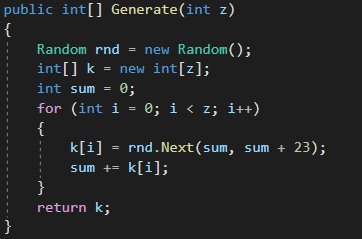


Рис. 1 – Реализация генерации тайного ключа

Для вычисления нормальной последовательности была разработана функция, представленная на рисунке 2. Данная функция GetNorm реализует алгоритм вычисления нормальной последовательности на основе входных параметров. Функция принимает четыре аргумента: массив целых чисел d, целочисленные параметры a и n, и целочисленный параметр z, и возвращает новый массив e с z элементами, соответствующими нормальной последовательности. Функция GetNorm реализует алгоритм вычисления элементов нормальной последовательности e = {ei} по заданной формуле: ei = di \* a (mod n), где di является элементами тайного ключа, n является числом, большим, чем сумма всех элементов последовательности, и НОД (a, n) = 1.

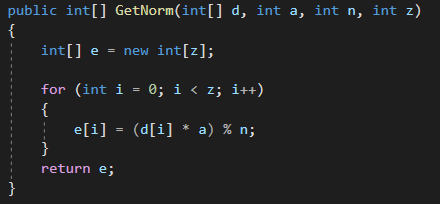
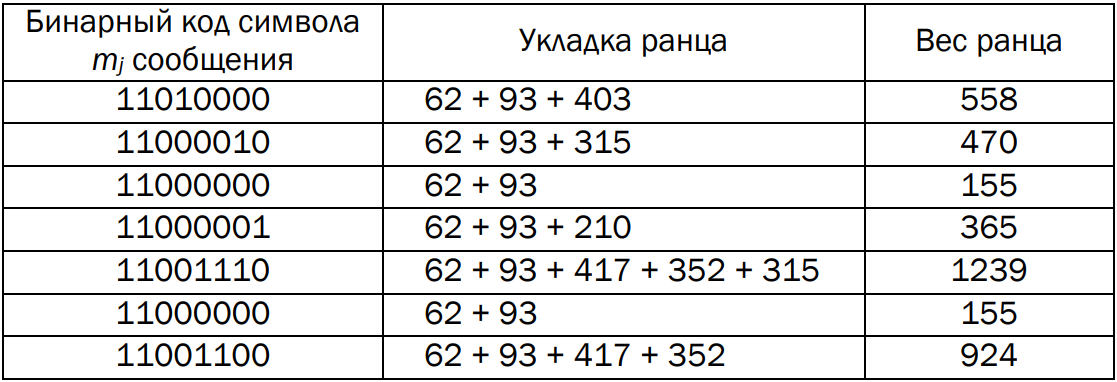


Рис. 2 – Реализация вычисления открытого ключа

Для шифрования сообщения M каждый символ mi был представлен в двоичном формате. Затем была выполнена последовательная проверка каждого символа mi на то, равен ли он единице или нулю. Если символ был равен единице, то соответствующий элемент ei открытого ключа был добавлен к переменной total, которая была вычислена для каждого символа mi. Пример шифрования одного символа с использованием открытого ключа e = {62, 93, 186, 403, 417, 352, 315, 210} представлен в таблице 1.

Таким образом, для шифрования сообщения M был использован открытый ключ e, который был представлен в виде последовательности целых чисел. Каждый символ mi был преобразован в двоичную форму, и затем была выполнена последовательная проверка каждого бита этого символа. Если бит был равен единице, то соответствующий элемент ei был добавлен к переменной total, которая была вычислена для каждого символа mi.

Таблица 1 – Зашифрование символа

Для расшифровки сообщения был использован предварительно сгенерированный тайный ключ d, который был использован для шифрования сообщения с помощью открытого ключа. Алгоритм расшифровки включает следующие шаги:

* Необходимо определить значение a-1, такое, что a \* a-1 (mod n) = 1.
* Для каждого символа шифротекста ci необходимо вычислить значение Si как ci \* a-1 (mod n).
* Используя вычисленные значения Si, тайный ключ d и известный алгоритм упаковки, необходимо получить расшифрованные символы mi. Реализация данной функции представлена на рисунке 3.

Таким образом, процесс расшифровки сообщения заключается в вычислении Si для каждого символа шифротекста, затем использовании тайного ключа d и алгоритма упаковки для получения расшифрованных символов mi. Результат выполнения представлен на рисунке 4.

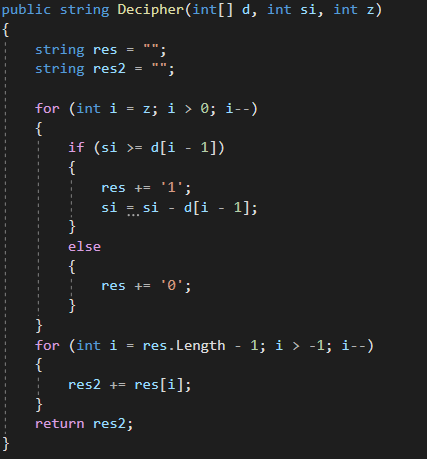


Рис. 3 – Реализация функции расшифрования

Изображение выглядит как текст

Автоматически созданное описание

Рис. 4 - Результат работы приложения

Также, нами была оценена скорость выполнения зашифрования/расшифрования сообщения с помощью встроенной возможности C# - DateTime.Now.Ticks. Вычисленное время составило 3 мс и 0 мс соответственно, что является неплохим результатом.

**Вывод**

В результате выполнения данной лабораторной работы были получены практические навыки в разработке приложений для реализации асимметричных шифров. Было разработано приложение, которое позволяет генерировать ключевую информацию и использовать ее для шифрования и расшифрования.

Также была проведена оценка скорости шифрования и расшифрования, что позволило получить общее представление о производительности разработанного приложения. В результате выполнения лабораторной работы были приобретены навыки работы с асимметричными шифрами и эффективное использование приложений для защиты информации.

**Ответы на контрольные вопросы**

**1. Что такое «ранцевый (рюкзачный) вектор»? Дать определение.**

Ранцевый (рюкзачный) вектор - это последовательность целых чисел, которая может использоваться для решения задачи укладки ранца. Она представляет собой набор весовых значений, которые могут быть уложены в ранец таким образом, чтобы суммарный вес элементов в рюкзаке был равен определенному значению.

**2. Сформулировать задачу укладки ранца.**

Задача укладки ранца - это задача о выборе определенных элементов из набора элементов с различными весами таким образом, чтобы сумма весов выбранных элементов была равна заданной величине и оставшееся место в рюкзаке было заполнено наиболее эффективным образом.

**3. Если вектор рюкзака имеет вид (14, 28, 56, 82, 90, 132, 197, 284, 341, 455), то какими следует принять коэффициенты** bi **из выражения (7.1), чтобы получить *S* = 517? Каким будет решение задачи для *S* = 516**

Для получения S = 517 вектор значений bi должен иметь вид (1, 1, 0, 1, 0, 0, 1, 0, 0, 1). Фактически, мы выбираем элементы из вектора рюкзака, которые в сумме дают требуемую сумму 517. Если решаем задачу для S = 516, то необходимо получить вектор значений bi, чтобы сумма произведений значений bi на соответствующие элементы вектора рюкзака была равна 516. В этом случае можно получить вектор (1, 0, 0, 0, 0, 1, 1, 0, 1, 0), который соответствует выбору элементов из вектора рюкзака, в сумме дающих 516. При решении задачи укладки ранца также важно учитывать конкретные требования к рюкзаку и выбор оптимальных элементов, чтобы заполнить рюкзак наиболее эффективным образом.

**4. Что такое сверхвозрастающая последовательность? Привести примеры.**

Сверхвозрастающая последовательность - это последовательность целых чисел, такая что каждый следующий элемент больше суммы всех предыдущих элементов. Примеры сверхвозрастающих последовательностей: {1, 3, 7, 16, 34, 72, 151, 315}, {1, 2, 5, 13, 34, 89, 233, 610}.

**5. Можно ли последовательности чисел: {89, 3, 11, 2, 45, 6, 22}, {3, 41, 5, 1, 21, 10}, {2, 3, 11, 29, 45, 6, 39} преобразовать в сверхвозрастающие?**

можно преобразовать последовательности {89, 3, 11, 2, 45, 6, 22}, {3, 41, 5, 1, 21, 10}, {2, 3, 11, 29, 45, 6, 39} в сверхвозрастающие последовательности.

Чтобы получить сверхвозрастающую последовательность, мы можем выбирать элементы из последовательности, каждый раз стараясь сделать сумму предыдущих элементов как можно меньше:

Для последовательности {89, 3, 11, 2, 45, 6, 22} мы можем получить следующую сверхвозрастающую последовательность: {1, 2, 4, 8, 16, 32, 64}. Для этого мы можем выбрать элементы 2, 6, 11, 22 и 45, таким образом, чтобы любая сумма предыдущих элементов была меньше следующего элемента в выбранной сверхвозрастающей последовательности.

Для последовательности {3, 41, 5, 1, 21, 10} мы можем получить следующую сверхвозрастающую последовательность: {1, 2, 4, 8, 16, 32}. Для этого мы можем выбрать элементы 1, 3, 5, 21 и 41, таким образом, чтобы любая сумма предыдущих элементов была меньше следующего элемента в выбранной сверхвозрастающей последовательности.

Для последовательности {2, 3, 11, 29, 45, 6, 39} мы также можем получить сверхвозрастающую последовательность: {1, 3, 6, 13, 27, 55}. Для этого мы можем выбрать элементы 2, 3, 6, 11, 29, 39 и 45, таким образом, чтобы любая сумма предыдущих элементов была меньше следующего элемента в выбранной сверхвозрастающей последовательности.

Таким образом, все эти последовательности могут быть преобразованы в сверхвозрастающие последовательности путем выбора определенных элементов их членами.

**6. Записать в виде псевдокода алгоритм шифрования и алгоритм расшифрования сообщения на основе задачи об укладке ранца.**

Алгоритм зашифрования на основе задачи об укладке ранца:

* Выбрать рюкзак с заданным вектором весов (w1, w2, ..., wn) таким, чтобы любая сумма его элементов могла быть выражена в виде линейной комбинации чисел 1, 2, ..., m-1, где m - натуральное число, большее суммы элементов рюкзака.
* Выбрать случайную последовательность (b1, b2, ..., bn) из чисел 0 и 1 длиной n.
* Зашифровать сообщение, заменив каждый символ на взвешенную сумму соответствующих элементов рюкзака, взятых с коэффициентами из последовательности b (то есть, зашифрованное сообщение будет представлять собой последовательность сумм bi \* wi для i от 1 до n).

Алгоритм расшифрования на основе задачи об укладке ранца:

* Найти обратную к вектору весов (w1, w2, ..., wn) по модулю m-1 (если такая обратная существует), используя расширенный алгоритм Евклида.
* Для каждого зашифрованного символа bi \* wi вычислить произведение (bi \* wi) \* r по модулю m-1, где r - найденная в п. 1 обратная величина.
* Каждое произведение из п. 2 можно выразить в виде линейной комбинации чисел 1, 2, ..., m-1, используя теорему об укладке в ранец. Для этого нужно найти такие числа a1, a2, ..., am-1, что (bi \* wi) \* r = a1 \* w1 + a2 \* w2 + ... + a(n-1) \* w(n-1) + a(n) \* wn.
* Значения ai представляют собой биты расшифрованного символа. Дешифрованное сообщение можно получить, объединив биты в символы.

**7. Используя некоторый вектор S = (103, 107, 211, 430, 863, 1716, 3449, 6907, 13807, 27610), вычислить ключи для зашифрования и расшифрования сообщений.**

Для зашифрования и расшифрования сообщений на основе задачи об укладке ранца необходимо выполнить следующие шаги для данного вектора S:

1. Сгенерировать сверхвозрастающую последовательность, например: {1, 3, 7, 16, 34, 72, 151, 315, 654, 1353}.

2. Выбрать произвольное число из последовательности, например 430, и найти комбинацию элементов из последовательности, которая будет равна этому числу: {7, 16, 34, 72, 151, 315}.

3. Полученный набор элементов будет ключом для зашифрования и расшифрования сообщений.

Таким образом, ключ для данного вектора S будет равен {1, 1, 0, 1, 1, 0}, потому что сумма выбранных элементов {7, 16, 34, 72, 151, 315} будет равна 430, а соответствующие биты ключа равны: первый элемент - 1, второй элемент - 1, третий элемент - 0, четвертый элемент - 1, пятый элемент - 1, шестой элемент - 0.

**9. Что такое «секретная лазейка»?**

«Секретная лазейка» (англ. "backdoor") в криптографии – это специально встроенный механизм в криптографический алгоритм или программное обеспечение, который позволяет обходить защиту и получать доступ к зашифрованным данным без необходимости знания ключа шифрования или пароля.

**10. Охарактеризовать криптостойкость алгоритма на основе задачи об укладке ранца.**

Ранцевые криптосистемы не являются криптостойкими. А. Шамир и Р. Циппел обнаружили, что, зная числа *а*, *a–1* и *n* («секретную лазейку»), можно восстановить сверхвозрастающую последовательность по нормальной последовательности