Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Лабораторная работа № 8**

«Исследование потоковых шифров»

Выполнил:

Студент: Лэ Н.З.

ФИТ 2 курс 4 группа

Преподаватель: Берников О.В.

Минск 2020

1. **Теоретические сведения**

Потоковый шифр (иногда говорят «поточный») – симметричный шифр, преобразующий каждый символ mi открытого текста в символ шифрованного, ci, зависящий от ключа и расположения символа в тексте.

Термин «потоковый шифр» обычно используется в том случае, когда шифруемые символы открытого текста представляются одной буквой, битом или реже – байтом.

Основной задачей потоковых шифров является выработка некоторой последовательности (гаммы) для зашифрования, т.е. выходная гамма является ключевым потоком (ключом) для сообщения.

Синхронные потоковые шифры (СПШ) характеризуются тем поток ключей генерируется независимо от открытого текста и шифртекста. Главное свойство СПШ – нераспространение ошибок. Ошибки отсутствуют, пока работают синхронно шифровальное и дешифровальное устройства отправителя и получателя информации. Один из методов борьбы с рассинхронизацией – разбить отрытый текст на отрезки, начало и конец которых выделить вставкой контрольных меток (специальных маркеров).

Синхронные потоковые шифры уязвимы к атакам на основе изменения отдельных бит шифртекста.

В самосинхронизирующихся потоковых шифрах символы ключевой гаммы зависят от исходного секретного ключа шифра и от конечного числа последних знаков зашифрованного текста. Основная идея заключается в том, что внутреннее состояние генератора потока ключей является функцией фиксированного числа предыдущих битов шифртекста. Поэтому генератор потока ключей на приемной стороне, приняв фиксированное число битов, автоматически синхронизируется с генератором гаммы.

Недостаток этих потоковых шифров – распространение ошибок, так как искажение одного бита в процессе передачи шифртекста приведет к искажению нескольких битов гаммы и, соответственно, расшифрованного сообщения.

Потоковый шифр максимально должен имитировать одноразовый блокнот. В соответствии с этим ключ должен по своим свойствам максимально походить на случайную числовую последовательность.

Ключевые последовательности (случайные последовательности, СП, либо псевдослучайные последовательности, ПСП) вырабатываются специальными блоками систем потокового шифрования – генераторами.

Стандарт устанавливает криптографические алгоритмы генерации псевдослучайных чисел. Алгоритмы стандарта могут применяться для построения ключей, синхропосылок, одноразовых паролей, других непредсказуемых или уникальных параметров криптографических алгоритмов и протоколов. Стандарт применяется при разработке, испытаниях и эксплуатации средств криптографической защиты информации.

Указанный стандарт определяет базовые понятия в рассматриваемой предметной области:

* случайные числа (последовательности) – последовательность элементов, каждый из которых не может быть предсказан (вычислен) только на основе знания предшествующих ему элементов данной последовательности;
* псевдослучайные числа – последовательность элементов, полученная в результате выполнения некоторого алгоритма и используемая в конкретном случае вместо последовательности случайных чисел.

Часто используемый алгоритм генерирования (программно или аппаратно) ПСП реализуется на основе так называемого линейного конгруэнтного генератора, описываемого следующим рекуррентным соотношением:

xt+1 = (a\*xt + c) mod n (1.1)

где: xt и xt+1 – соответственно t-й (предыдущий) и (t+1)-й (текущий, вычисляемый) члены числовой последовательности; а, с и n – константы. Период такого генератора (период ПСП) не превышает n.

Если параметры a, b и c выбраны правильно, то генератор будет порождать случайные числа с максимальным периодом, равным c. При программной реализации значение с обычно устанавливается равным 2b-1 или 2b , где b – длина слова в битах.

Достоинством линейных конгруэнтных генераторов псевдослучайных чисел является их простота и высокая скорость получения псевдослучайных значений. Линейные конгруэнтные генераторы находят применение при решении задач моделирования и математической статистики, однако в криптографических целях их нельзя рекомендовать к использованию, так как специалисты по криптоанализу научились восстанавливать всю последовательность ПСЧ по нескольким ее значениям.

Генератор практически не используются в криптографии в силу низкой криптостойкости. Тем не менее, полезны для решения задач моделирования.

Комбинации нескольких (чаще двух) линейных конгруэнтных генераторов позволяют значительно повысить период ПСП. Б. Шнайер, например, приводит данные о том, как на 32-разрядных ПК реализовать генератор в виде комбинации двух, каждый из которых обеспечивает период соответственно 231 – 85 и 231 – 249, а комбинированный генератор позволяет достичь периода ПСП, равного произведению указанных чисел.

Достаточно распространенным является использование регистров сдвига (РС) в качестве генераторов ПСП в силу простоты реализации на основе цифровой логики. РС с линейной обратной связью (РСЛОС) состоит из двух частей: собственно РС и функции обратной связи. На рис. 6.2 представлена общая схема РС с линейной обратной связью. Функция обратной связи реализуется с помощью сумматоров сложения по модулю два.

РСЛОС строятся на основе примитивных порождающих полиномов (многочленов. Если многочлен является неприводимым, то период ПСП при ненулевом начальном условии (ненулевом состоянии) регистра будет максимально возможным: 2L – 1.

Генератор ПСП на основе RSA устроен следующим образом. Последовательность генерируется с использованием соотношения:

xt = (xt-1) е mod n (1.2)

Начальными параметрами служат n, большие простые числа p и q (причем n = p\*q), целое число е, взаимно простое с произведением (р – 1)\*( q –1), а также некоторое случайное начальное значение, x0.

Выходом генератора является на t-м шаге является младший бит числа xt.

Безопасность генератора опирается на сложности взлома алгоритма RSA, т. е. на разложении числа n на простые сомножители.

Широкое распространение получил алгоритм генерации ПСП, называемый алгоритмом BBS (от фамилий авторов: L. Blum, M. Blum, M. Shub) или генератором на основе квадратичных вычетов. Для целей криптографии этот метод предложен в 1986 г.

Начальное значение x0 генератора вычисляется на основе соотношения:

x0 = x^2 mod n (1.3)

где n, как и в генераторе на основе RSA, является произведением простых чисел p и q, однако в нашем случае эти простые числа должны быть сравнимы с числом 3 по модулю 4, т. е. при делении p и q на 4 должен получаться одинаковый остаток: 3; число x должно быть взаимно простым с n; число n называют числом Блюма.

Выходом генератора на t-м шаге является младший бит числа xt:

xt = (xt-1)^2 mod n (1.4)

Алгоритм является сравнительно медленном. Для ускорения можно использовать не последний бит числа, а несколько последних бит. Однако, понятно, что при этом алгоритм является менее криптостойким.

Алгоритм RC4 разработан Р. Ривестом в 1987 г. Представляет собой потоковый шифр с переменным размером ключа. Здесь гамма не зависит от открытого текста.

Алгоритм RC4, как и любой потоковый шифр, строится на основе генератора псевдослучайных битов (генератора ПСП). На вход генератора записывается ключ, а на выходе читаются псевдослучайные биты. Длина ключа может составлять от 40 до 2048 бит.

Ядро алгоритма состоит из функции генерации ключевого потока. Другая часть алгоритма – функция инициализации, которая использует ключ переменной длины Ki для создания начального состояния генератора ключевого потока.

В основе алгоритма – размер блока или слова, определяемый параметром n. Обычно n = 8, но можно использовать и другие значения. Внутренне состояние шифра определяется массивом слов (S-блоком) размером 2 n. При n = 8 элементы блока представляют собой перестановку чисел от 0 до 255, а сама перестановка зависит от ключа переменной длины. Другими элементами внутреннего состояния являются 2 счетчика (каждый размером в одно слово; обозначим их i и j) с нулевыми начальными значениями. В основе вычислений лежит операция по mod 2^n

Генератор ключевого потока RC4 переставляет значения, хранящиеся в S, и каждый раз выбирает различное значение из S в качестве результата. В одном цикле RC4 определяется одно n-битное слово K из ключевого потока, которое в последующем суммируется с исходным текстом для получения зашифрованного текста. Эта часть алгоритма называется генератором ПСП.

Байт K используется в операции XOR с открытым текстом для получения 8-битного шифртекста или для его расшифрования.

Так же достаточно проста и инициализация S-блока. Этот алгоритм использует ключ, который подается на вход пользователем. Сначала S-блок заполняется линейно: S0 = 0, S1 = 1, …, S255 = 255. Затем заполняется секретным ключом другой 256-байтный массив. Если необходимо, ключ повторяется многократно, чтобы заполнить весь массив: K0, K1, …, K255. Далее массив S перемешивается путем перестановок, определяемых ключом.

Другими известными потоковыми шифрами являются, например, SEAL и WAKE.

Наилучшие характеристики будут иметь генераторы случайных чисел, основанные на «естественных случайностях», свойственных, например, процессам в радиоэлектронной аппаратуре, в системах телекоммуникаций, в приемах работы с клавиатурой операторов и др.

1. **Практическая часть**

В данной лабораторной работе необходимо разработать пользовательское приложение, которое должно реализовывать следующие операции:

1. Приложение 1 должно реализовывать генерацию ПСП в соответствии с вариантом:

* вариант задания – 6;
* алгоритм генерации ПСП - Линейный конгруэнтный генератор;
* параметры - а = 430, с = 2531, n = 11979.

1. Приложение 2 должно реализовывать алгоритм RC4 в соответствии с вариантом, а также дополнительно выполнять оценку скорости выполнения операций генерации ПСП.

* вариант задания – 9;
* n - 6;
* ключ (в виде десятичных чисел) - 61, 60, 23, 23, 23, 20.

В качестве шифруемого сообщения может быть выбран произвольный текст.

Для выполнения первого задания была написана функция, которая позволяет имитировать работу линейного конгруэнтного генератора. Функция получает 3 параметра: x – стартовый элемент последовательности, a – мультипликатор, n – основание модуля. Необходимый параметр с (сумматор) высчитывается по формуле 2^b, где b – количество битов в n.

Согласно введенным параметрам по формуле y = (x\*a+c) mod n высчитывается следующий элемент последовательности.

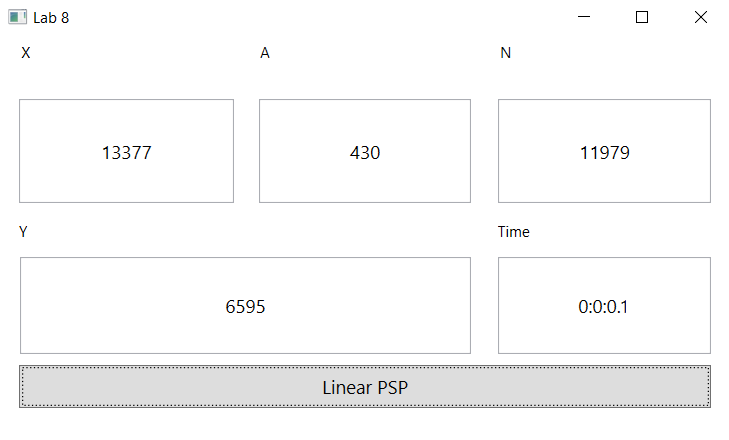


Рисунок 2.1 – Линейный конгруэнтный генератор

Для выполнения следующего задания (алгоритм RC4) было разработано приложение, предоставленное на рисунке 2.2. Приложение поддерживает ввод символов латинского алфавита (больших и малых). Ключевую последовательность можно ввести в

В приложении реализованы функции EncryptData и DecryptData, которые позволяют получить шифропоследовательность или же изначальную последовательность соответственно.

На вход функции EncryptData подаются два байтовых массива – массив ключа и массив даты. N является строго зафиксированным числом – 8. Создаются два массива – key и box размерностью в 2^n = 64. В массив key мы циклично помещаем байты ключа. Массив box заполняется числами от 0 до 63.

После формирования массивов мы поэлементно проходим массив box. Здесь основную роль играют счетчики i и j, которые отвечают за перемешивание блока box.

* j = 0; i =0;
* j = (j + Si + Ki) mod 64;
* поменять местами Si и Sj;
* i = i +1;
* если i < 64, то перейти во 2 пункт.

Далее нужно произвести генерации случайных 8-битных слов. Делается это согласно формулам:

* i = (i + 1) mod 64;
* j = (j + Si) mod 64;
* поменять местами Si и Sj;
* a = (Si + Sj) mod 64;
* K = Sa.

После чего отбираем из массива box все k-ые элементы.

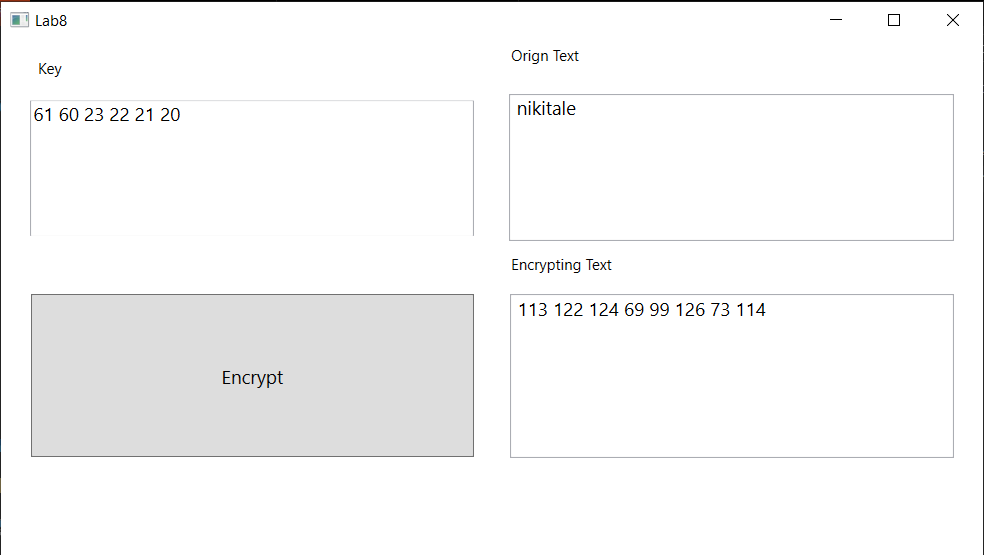
****

Рисунок 2.2 – Алгоритм RC4 (шифрование)

Для того, чтобы дешифровать сообщение, используется функция DecryptData, в которой вызывается функция EncryptData с параметрами: ключ (должен быть тот же, что и при шифровании) и сгенерированная последовательность. Функция произведет обратные преобразования и вернет индексы введенных символов.

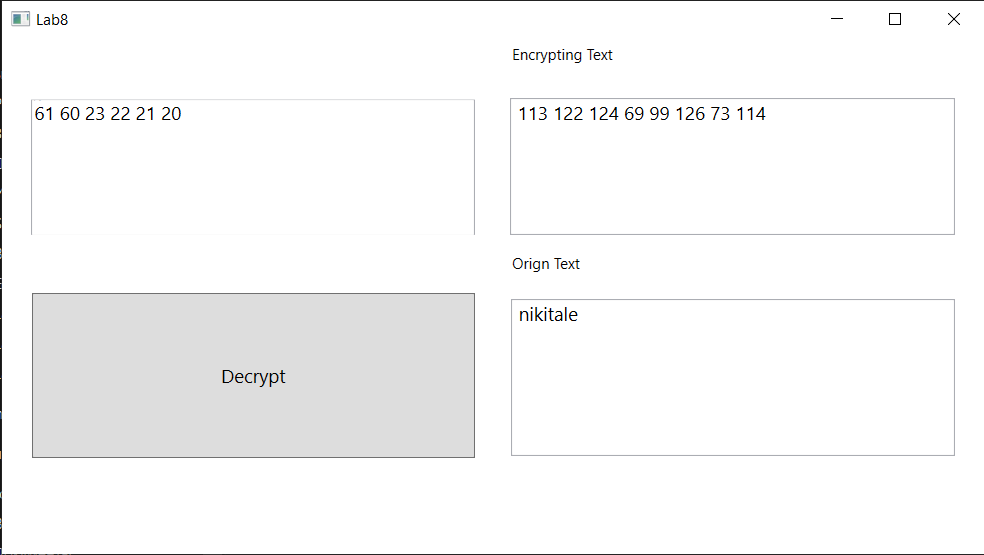
****

Рисунок 2.3 – Алгоритм RC4 (расшифрование)

**Вывод**

В данной лабораторной работе я закрепил теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости потоковых шифров. Разработал приложения для реализации указанных преподавателем методов генерации ключевой информации и ее использования для потокового зашифрования/расшифрования. Выполнил анализ криптостойкости потоковых шифров. А также оценил скорость зашифрования/расшифрования реализованных шифров.