Министерство образования Республики Беларусь

Учреждение образование

«Белорусский государственный технологический университет»

Кафедра информационных систем и технологий

**Отчет к лабораторной работе**:

«Исследование асимметричных шифров RSA и Эль-Гамаля»

Выполнил:

студент 3 курса 8 группы ФИТ

Романович Никита Руслановичы

Проверил:

Берников В. О.

Минск 2020

1. **Теоретические сведения**
   1. **Математические основы ассиметричных шифров**

Как отмечалось выше, асимметричная криптография основана на сложности решения некоторых математических задач. По существу, таких задач две: разложение больших чисел на простые сомножители (задача факторизации), и вычисление дискретного логарифма в конечном поле, вычислительные операции над точками эллиптической кривой.

Эти задачи объединяет то, что они используют операцию получения остатка от целочисленного деления.

В силу этого практически все системы асимметричного зашифрования/расшифрования основаны либо на проблеме факторизации (среди них – RSA), либо на проблеме дискретного логарифмирования (среди них – Эль-Гамаля).

Базовые элементы перечисленных проблем мы рассмотрели с практической точки зрения при выполнении лабораторной работы № 1. Алгебраическая теория рассматриваемого класса криптосистем подробно рассмотрена в. Мы же здесь остановимся лишь на нескольких основных элементах этой теории.

*Теорема* 1. *Основная теорема арифметики*. Всякое натуральное число *N*, кроме 1, можно представить как произведение простых множителей.

*Определение* 1. *Задача дискретного логарифмирования* формулируется так: для данных целых чисел *а* и *b*, 1 < *а*, *b* < *n* найти логарифм – такое целое число *х*, что *ax ≡ b (mod n*), если такое число существует.

По аналогии с вещественными числами используется обозначение *х* = loga*b*.

*Теорема* 2. *Китайская теорема об остатках*. В общем случае, если разложение числа *N* на простые множители представляет собой *p1\*p2\*…\*pt* (некоторые простые числа могут встречаться несколько раз), то система уравнений (*x* mod *pi*) = *ai*, где *i*= 1,2…, *t* имеет единственное решение: *x*, меньшее *N*.

Иными словами, число (меньшее, чем произведение нескольких простых чисел) однозначно определяется своими вычетами по модулю от этих простых чисел. Китайской теоремой об остатках можно воспользоваться для решения полной системы уравнений в том случае, если известно разложение числа *N* на простые множители.

**1.2 Алгоритм RSA**

Рассматриваемый алгоритм появился (1977 г.) после алгоритма рюкзака Меркла. Он стал первым полноценный алгоритмом с открытым ключом, который впоследствии стал одним из основных для шифрования и для электронных цифровых подписей.

Из всех предложенных алгоритмов с открытыми ключами RSA проще всего понять и реализовать. Названный в честь трех его создателей: Рона Ривеста (Ron**R**ivest), Ади Шамира (Adi Shamir) и Леонарда Эдлемана (Leonard **A**dleman).

Как было отмечено, безопасность RSA основана на трудности разложения на множители больших чисел. Открытый и закрытый ключи являются функциями двух больших простых чисел. Предполагается, что восстановление открытого текста по шифртексту и открытому ключу эквивалентно разложению на множители двух больших чисел.

Для генерации двух ключей: тайного и открытого (а по сути – двух взаимосвязанных частей одного ключа, т. е. ключа, принадлежащего одному физическому лицу (или группе лиц), либо одному юридическому лицу) используются два больших случайных простых числа, *p* и *q*. Для максимальной большей криптостойкости нужно выбирать *p* и *q* равной длины. Рассчитывается произведение: *n* = *pq*. Этой есть один из трех компонент ключа, состоящего из чисел *n*, *e*, *d*.

Затем случайным образом выбирается второй компонент ключа(открытый ключ или ключ зашифрования, *e*, такой что *e* и (*p*-1)(*q*-1) являются взаимно простыми числами; вспомним, что (*p*-1)(*q*-1) = φ(*n*) – функция Эйлера. Б. Шнайер рекомендует число е выбирать из ряда: 3, 17, 216 + 1.

Наконец расширенный алгоритм Евклида используется для вычисления третьего компонента ключа: ключа расшифрования, *d*, такого, что выполняется условие: *ed* = 1 (mod φ(n)).

Таким образом, сформирован ключ, состоящий из трех чисел, которые, в свою очередь, образуют две вышеупомянутые взаимосвязанные части: открытый (публичный) ключ, (*e*, *n*), и тайный ключ, (*d*, *n*; на самом деле, как видим, тайным здесь является лишь первое из пары чисел).

Примеры генерации ключевой информации, как и ее использования, можно найти.

Использование ключа.

Для зашифрования/расшифрования используется ключ получателя: отправитель шифрует сообщение открытым ключом, а получатель расшифровывает шифртекст своим тайным ключом.

*Зашифрование.* Если шифруется сообщение *М*, состоящее из *r* блоков: *m1, m2 , …, mi*,…,*mr,* то шифртекст *С* будет состоять из такого же числа (*r*) блоков, представляемых числами:

*ci* = (*mi*)*e* mod *n*.

*Расшифрование*. Для расшифрования каждого зашифрованного блока производится вычисление вида:

*mi* = (*ci*)*d* mod *n*.

Разработаны несколько версий стандарта рассматриваемого алгоритма. Среди прочего, в этих документах обсуждаются размеры безопасного ключа. Доступна одна из последних версий стандарта RSA: RFC 3447.

Размер ключа в алгоритме RSA связан с размером модуля, *n*. Два числа *p* и *q*, произведение которых равно *n*, должны иметь приблизительно

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.

**1.3 Алгоритм Эль-Гамаля**

Предложен Эль-Гамалем (T. El-Gamal) в 1985 г. Он может быть использован для решения трех основных криптографических задач: для зашифрования/расшифрования данных, для формирования цифровой подписи и для согласования общего ключа. Кроме того, возможны модификации алгоритма для схем проверки пароля, доказательства идентичности сообщения и другие варианты.

Как подчеркивалось выше, безопасность алгоритма Эль-Гамаля, как и безопасность алгоритма Диффи-Хеллмана, основана на трудности вычисления дискретных логарифмов. Алгоритм Эль-Гамаля фактически использует схему Диффи-Хеллмана, чтобы сформировать общий секретный ключ для абонентов, передающих друг другу сообщение, и затем сообщение шифруется путем умножения его на этот ключ.

И в случае шифрования, и в случае формирования цифровой подписи каждому пользователю необходимо сгенерировать пару ключей.

Рассматриваемый алгоритм отличается от алгоритма RSA несколькими параметрами и особенностями:

1) генерацией ключевой информации и числом компонент, составляющих ключ;

2) каждому блоку (символу) открытого сообщения в шифртексте на основе алгоритма Эль-Гамаля соответствуют 2 блока (в RSA – один-один);

3) в алгоритме Эль-Гамаля при зашифровании используется число (обозначим его *k*), которое практически никак не связано с ключевой информацией получателя и которое принимает (по определению) различные значения при зашифровании различных блоков сообщения.

***Генерация ключевой информации***. Выбирается простое число, *р*. Выбирается число (*g, g* < *p*), являющееся первообразным корнем числа *р* – очень важный элемент с точки зрения безопасности алгоритма (см. ниже).

Далее выбирается число х (х < *p*) и вычисляется последний компонент ключевой информации:

*y =g*х mod *р*.

Владельцу сформированной ключевой информации, состоящей из 4 чисел, может посылаться некоторый шифртекст, созданный с использованием открытого ключа получателя: *p*, *g*, *y*. Расшифрование шифртекста получатель производит своим тайным ключом: *p*, *g*, х.

Как видим, на самом деле тайным является лишь одно число (как и в RSA): *х*.

*Определение* 2. *Первообразный корень* (primary (residual ) root ) по модулю *р* является таким числом, что его степени (*gi*, 1 ≤*i*≤*p*-1 ) дают все возможные по модулю *р* вычеты (остатки), которые взаимно просты с *p*.

*Пример* 1. Следующие остатки по модулю 5 (*p* = 5) от 2i: 2, 4, 3, 1. Они дают все возможные остатки. Число 2 является первообразным корнем по модулю 5.

*Пример*2. Следующие остатки по модулю 7 от 2i: 2, 4, 1, 2, ... (они не дают всех возможных остатков). Число 2 не является первообразным корнем по модулю 7. *Пример* 3. Для *р* = 17 и *g*= 3: остатки по модулю 17 от 3i: 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6, 1.

Число 3 является первообразным корнем по модулю 17.

Можно также отметить, что, например, 5 является первообразным корнем числа 97, 19 – числа 191, 11 – числа 839, 7 – числа 997.

Понятно, что для больших значений *р* количество всех неповторяющихся остатков (*р* – 1) будет также большим. А поскольку в равнении (8.6) мы используем модуль *р* большого простого числа и находим первообразным корень от *р*, который имеет важное свойство: при использовании разных степеней (*а*i = *а*х) решение будет равномерно распределяться от 0 до *р* – 1, то нахождение криптоаналитиком нужного *х* чрезвычайно затруднено. В этом заключается односторонность функции, задаваемой. И на этом основывается криптостойкость шифра Эль-Гамаля.

Для схемы вероятностного шифрования само сообщение и ключ не определяют шифртекст однозначно.

***Зашифрование сообщения***. Как ранее, предположим, что сообщение *М* = {*mi*}, где – *mi* – *i*-й блок сообщения.

Зашифрование отправителем (каждого отдельного блока*mi* исходного сообщения) предусматривает использование, как это особо подчеркивалось выше, некоторого случайного числа *k* (1 < k <p – 1).

В силу использования случайной величины *k* шифр Эль-Гамаля называют также *шифром многозначной замены*, а также *схемой вероятностного шифрования*.

Вероятностный характер шифрования является преимуществом для схемы Эль-Гамаля по сравнению, например, с алгоритмом RSA.

Блок шифртекста (*ci*) состоит из двух чисел: аi и *bi*:

*a*i *= g*k mod *p*, *b*i = (*y*k \**mi*) mod *p*. Здесь стал очевидный упомянутый недостатком алгоритма шифрования Эль-Гамаля: удвоение (реально – примерно в 1,5 раза) длины зашифрованного текста по сравнению с начальным текстом.

Случайное число *k* должно сразу после вычисления уничтожаться.

***Расшифрование*** *ci* выполняется по следующей формуле:

*m*i = (*b*i \*(*a*i)x)-1) mod *p* или *mi = (bi \*(ai)*р-x-1) mod *p* где (*a*x)-1 – обратное значение числа *a*x по модулю *p*.

Нетрудно проверить, что (*ai*)x)-1) = *gk*х mod *p*.

Еще раз возвратимся к криптостойкости рассмотренного алгоритма.

Если для зашифрования двух разных блоков (*m*1и *m*2) некоторого сообщения использовать одинаковые *k*, то для соответствующих шифртекстов *c*1 = (*a*1, *b*1) и *c*2 = (*a*2, *b*2) выполняется соотношение *b*1(*b*2)-1 = *m*1(*m*2)-1. Из этого выражения можно легко вычислить *m*2, если известно *m*1.

При примерно одинаковой размерности ключей рассмотренные алгоритмы обеспечивают примерно одинаковый уровень криптостойкости.

**2 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: зашифрование и расшифрование текстовых документов на основе алгоритмов RSA и Эль-Гамаля, а также определение времени выполнения операций. Исходный текст для зашифрования – собственные фамилия, имя, отчество. Для численного представления блоков текста можно, в том числе, пользоваться указанными выше кодировочными таблицами.

Реализация алгоритма RSA приведена на рисунке (Рисунок 2.1)

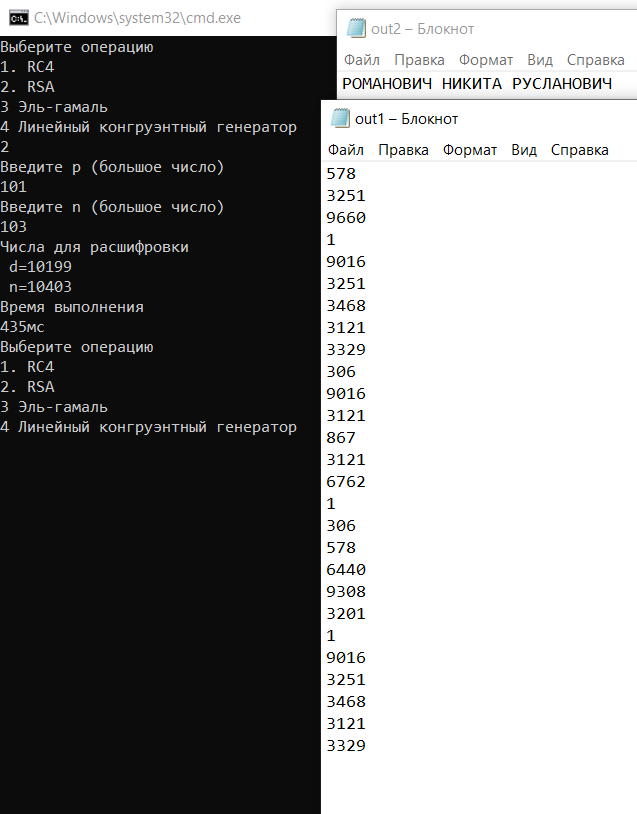


Рисунок 2.1 – Пример работы алгоритма RSA

Пример работы алгоритма Эль-Гамаля приведен на рисунке (Рисунок 2.2)

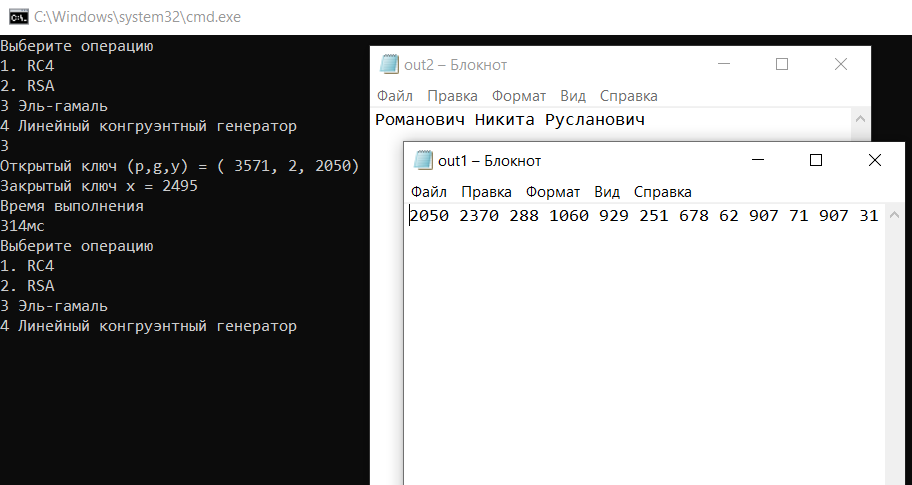


Рисунок 2.2 – Пример работы алгоритма Эль-Гамаля

Вывод: В данной лабораторной работе были изучены теоретические сведения об ассиметричных а также разработано программное средство в соответствии с заданием реализующее работу алгоритма RSA, а также программно реализован алгоритм Эль–Гамаля.