Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный технологический университет»

Факультет информационных технологий

Кафедра информационных систем и технологий

**Отчет к лабораторной работе**:

«Исследование асимметричных шифров RSA и Эль-Гамаля»

Выполнила:

студентка 3 курса 5 группы

специальности ПОИТ

Кубик Е.В.

Минск 2021

1. **Теоретические сведения**

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);
* 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).

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

N = p1\* p2\* p3\* ... \* pz, z > 1. (1.1)

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

a^x ≡ b (mod n), (1.2)

если такое число существует.

По аналогии с вещественными числами используется обозначение х = logab.

Теорема 2. Китайская теорема об остатках. В общем случае, если разложение числа N на простые множители представляет собой p1\*p2\*…\*pt (некоторые простые чис.ла могут встречаться несколько раз), то система уравнений

(x mod pi) = ai, (1.3)

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.

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 (RonRivest), Ади Шамира (Adi Shamir) и Леонарда Эдлемана (Leonard Adleman).

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)). (1.4)

Другими словами:

d(-1) = e(mod φ(n)). (1.5)

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).

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 шифротекст С будет состоять из такого же числа (r) блоков, представляемых числами:

ci = (mi)^e mod n. (1.6)

Расшифрование. Для расшифрования каждого зашифрованного блока производится вычисление вида:

mi = (ci)^d mod n. (1.7)

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 одинаковую длину, поскольку в этом случае найти сомножители (факторы) сложнее, чем в случае, когда длина чисел значительно различается. Например, если предполагается использовать 768-битный модуль, то каждое число должно иметь длину приблизительно 384 бита. В 1999 году 512-битный ключ был вскрыт за семь месяцев. Это означает, что 512-битные ключи уже не обеспечивают достаточную криптостойкость. Сейчас в критических системах применяются ключи длиной 1024 и 2048 бит.

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. (1.8)

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.

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 (8.6). 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 блокаmi исходного сообщения) предусматривает использование, как это особо подчеркивалось выше, некоторого случайного числа k (1 < k).

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:

ai = g^k mod p, (1.9)

bi = (y^k \*mi) mod p. (1.10)

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:

mi = (bi \*(ai)^р-x-1 ) mod p (1.11)

Еще раз возвратимся к криптостойкости рассмотренного алгоритма.

Если для зашифрования двух разных блоков (m1и m2) некоторого сообщения использовать одинаковые k, то для соответствующих шифртекстов c1 = (a1, b1) и c2 = (a2, b2) выполняется соотношение b1(b2) -1 = m1(m2) -1. Из этого выражения можно легко вычислить m2, если известно m1.

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

1. **Практическая часть**

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

1. С помощью простого консольного приложения составить табличную или графическую форму зависимости времени вычисления параметра у, функционально заданного выражением вида:

у = a^x mod n,

от параметров: а (десятичные числа от 5 до 35; можно взять 1 или 2 числа), х (числа, желательно – простые, из диапазона от 103 до 10100; для примера взять 5-10 чисел, равномерно распределенных в указанном диапазоне), n (для примера взять числа, в двоичном виде состоящие из 1024 и 2048 бит).

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

Приложение должно реализовывать следующие операции:

* зашифрование и расшифрование текстовых документов на основе алгоритмов RSA и Эль-Гамаля;
* 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.

Ключевую информацию для обоих алгоритмов можно сгенерировать самостоятельно либо воспользоваться, например, одной из утилит криптографической библиотеки OpenSSL, с помощью которой, в частности, можно сгенерировать ключевую информацию для алгоритма RSA.

1. Используя примерно одинаковый порядок ключевой информации, оценить производительность обоих алгоритмов и относительное изменение объемов криптотекстов (по отношению к объемам открытых текстов).

Для выполнения первого задания была написана функция, которая позволяет произвести расчеты времени вычисления у = a^x mod n.

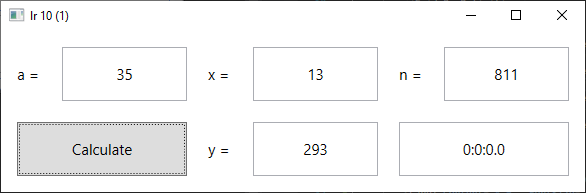


Рисунок 10.1 – Расчет времени вычисления

Для выполнения следующего задания (алгоритм RSA) было разработано приложение, предоставленное на рисунке 10.2. Приложение поддерживает ввод символов латинского алфавита (больших и малых). Первые два вводимых параметра – p и q. Должны быть простыми числами. При попытке ввода не простых чисел – выведет соответствующее сообщение.

На основе этих чисел высчитывает n = p\*q.

После чего высчитывается f(n) = (p-1)\*(q-1).

Третий параметр – число e, которое должно быть меньше n и должно быть взаимно простым с f(n).

Третий важный компонент – ключ расшифрования d, вычисляется при помощи расширенного алгоритма Евклида:

ed = 1 (mod F(n)).

После чего посимвольно вычисляется индекс каждой буквы в сообщении и подставляется в формулу:

c[i] = m[i]^e mod n.

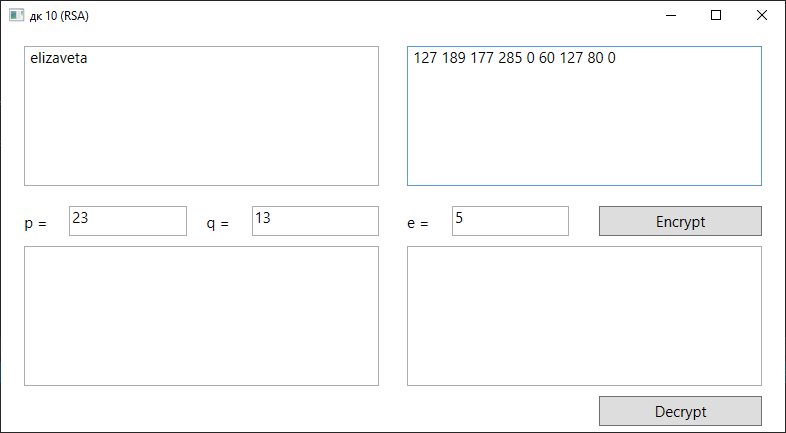


Рисунок 10.2 – Алгоритм RSA (шифрование)

Для дешифрования сообщения мы вводим все предыдущие параметры: p, q и e. По предыдущему алгоритму вычисляется параметр d.

После чего числа в сообщении подставляется в формулу:

m[i] = c[i]^d mod n.

После чего по индексам получаем соответствующие буквы.

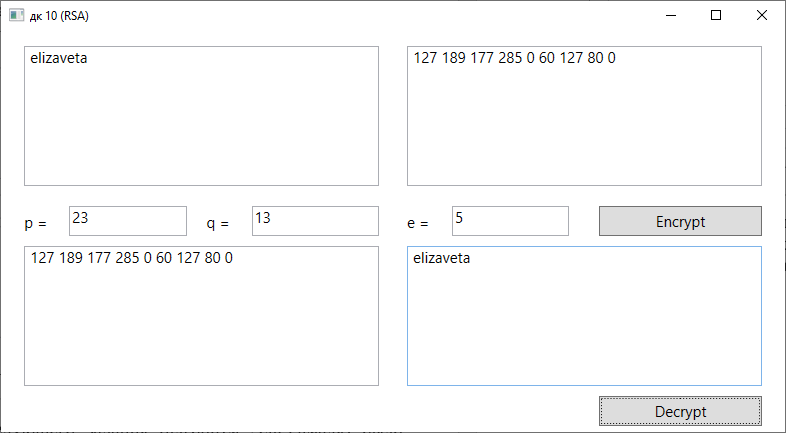


Рисунок 10.3 – Алгоритм RSA (расшифрование)

Для выполнения следующего задания (алгоритм Эль-Гамаля) было разработано приложение, предоставленное на рисунке 10.4. Приложение поддерживает ввод символов латинского алфавита (больших и малых). Первые два вводимых параметра – p и g. P должно быть простым числом, а еще g должно быть больше 1 и меньше p - 1. При попытке ввода не простых чисел – выведет соответствующее сообщение.

Также необходим третий параметр – секретный ключ c. Он также должен быть больше единицы и меньше p – 1.

По формуле d = g^c mod p.

Берется случайное число k.

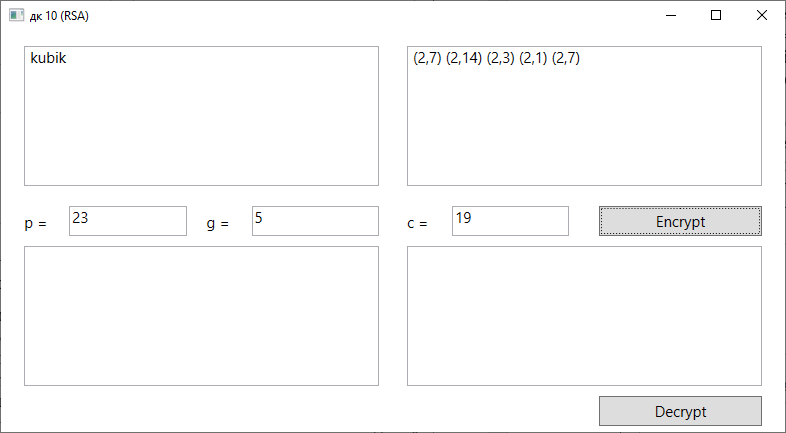
После чего по формулам вычисляются:

r = g^k mod p.

e = (m[i] \* d^k) mod p.

Число r для всех символов сообщения является одинаковым. Число e для каждой буквы получится разным.

Все, что остается сделать – сформировать пары этих чисел и вывести их.

Рисунок 10.4 – Алгоритм Эль-Гамаля (шифрование)

Для того чтобы дешифровать сообщение, нам нужно знать p и секретный ключ c, а также пары чисел r и e.

Индекс символа рассчитывается по формуле:

m[i] = (e \* r^р-c-1) mod p

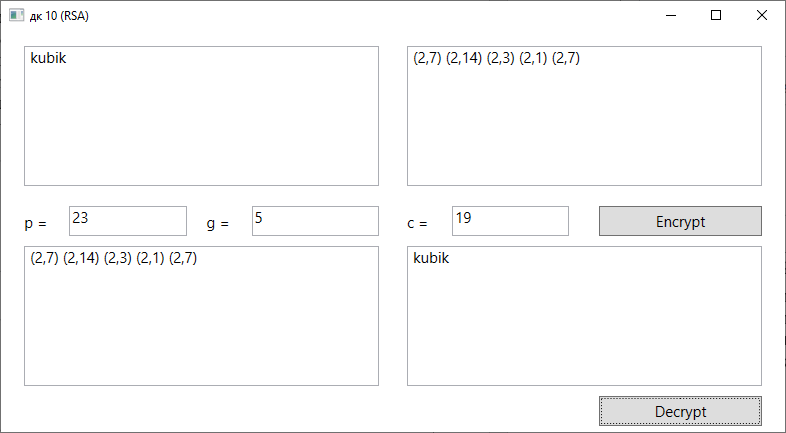


Рисунок 10.5 – Алгоритм Эль-Гамаля (расшифрование)

**Вывод**

В данной лабораторной работе я закрепила теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости асимметричных шифров RSA и Эль-Гамаля. Разработала приложения для реализации асимметричного зашифрования/расшифрования на основе этих алгоритмов. Выполнила анализ криптостойкости асимметричных шифров RSA и Эль-Гамаля. Выполнила анализ криптостойкости ассиметричных шифров RSA и Эль-Гамаля. А также оценила скорость зашифрования/расшифрования реализованных шифров.