МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образование «Белорусский государственный технологический университет»

Кафедра информационных систем и технологий

**«Исследование асимметричных шифров RSA и Эль-Гамаля»**

Студент:

Агапкина Диана Сергеевна

Вариант 7

Преподаватель:

Блинова Евгения Александровна

Минск 2020

**ЗАДАНИЕ 1.** Алгоритм RSA

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.

Реализация функции генерации открытого и закрытого ключей представлена ниже.



Для генерации двух ключей: тайного и открытого 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.

Наконец расширенный алгоритм Евклида используется для вычисления третьего компонента ключа: ключа расшифрования, *d*, такого, что выполняется условие:



Другими словами:



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.

Функция шифрования представлена ниже.



Для зашифрования каждого зашифрованного блока производится вычисление вида:



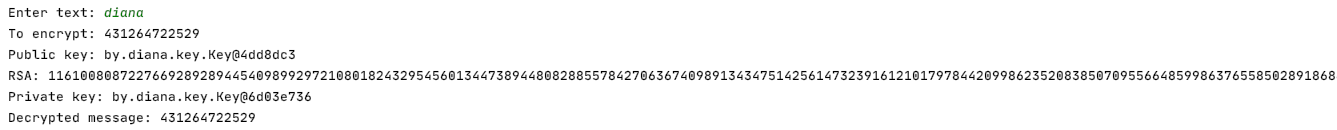
Функция расшифрования представлена ниже.



Для зашифрования каждого зашифрованного блока производится вычисление вида:



Результат выполнения программы представлен ниже.



**ЗАДАНИЕ 2.** Алгоритм Эль-Гамаля

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.

Реализация функции генерации открытого и закрытого ключей представлена ниже.



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:



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

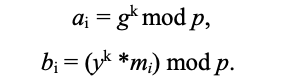
Как видим, на самом деле тайным является лишь одно число (как и в RSA): х.

Зашифрование отправителем (каждого отдельного блокаmi 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:



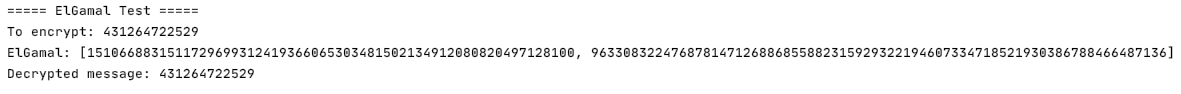
Здесь стал очевидный упомянутый недостатком алгоритма шифрования Эль-Гамаля: удвоение (реально – примерно в 1,5 раза) длины зашифрованного текста по сравнению с начальным текстом.

Случайное число k должно сразу после вычисления уничтожаться. Расшифрование ci выполняется по следующей формуле:





Результат представлен ниже.



**ЗАДАНИЕ 3.** Генерация ключей с помощью OpenSSL

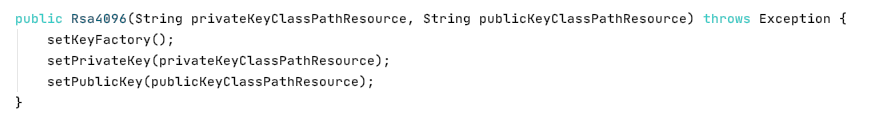
Сначала генерируем приватный ключ. В Java класс PKCS8EncodedKeySpec ожидает закрытый ключ RSA с кодировкой PKCS8. (Java-код, н.д.). Сгенерируем с помощью OpenSSL.



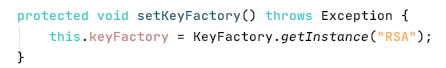
Теперь открытый. В Java класс X509EncodedKeySpec ожидает открытый ключ RSA с кодировкой X509. (Java-код, н.д.). Открытый ключ генерируется из закрытого ключа, поэтому сначала необходимо иметь закрытый ключ.



Конструктор.



Конструктор прост и принимает 2 параметра. Первый параметр – это полное имя пути к классу файла личного ключа, сгенерированного OpenSSL. Второй параметр одинаков для файла открытого ключа.



Метод setKeyFactory () создает экземпляр класса KeyFactory для алгоритма RSA. Затем будет использоваться для генерации приватного и публичного ключей.

Метод установки приватного ключа.



Метод установки публичного ключа.

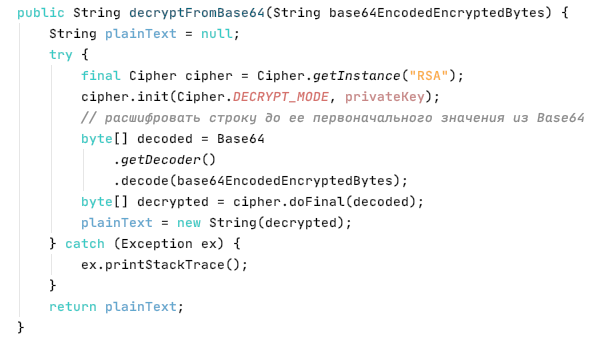


Метод шифрования представлен ниже.



Принимает в качестве параметра строку для шифрования. Возвращаемый тип говорит, что будет возвращена строка в кодировке Base64. Для зашифрования нужен только открытый ключ.

Метод расшифрования.



Тест и результат представлен ниже.

