# лабораторная работа №2 “Атаки на шифрование методом RSA. закладки”

## Цель работы

Целью работы является знакомство с классическими атаками на шифрование с использованием алгоритма RSA.

## Основные сведения

Для дешифрации необходимо по известным *N*, *e* и шифртексту *y* найти такое , что .

Попытаемся решить сравнение при конкретных *y*, затем использовать гомоморфность отображения *D*(*x*).

Один из возможных способов следующий: пусть имеется набор пар  с условием, что , 1 < *y* < *N*, (*y*, *N*) = 1. Если каким-либо образом удалось представить *y* в виде  с целыми *sk*, то  будет решением сравнения 

*Пример*. В наличии имеется открытый ключ *N* = 31459, *e* = 5 и набор пар соответствующих друг другу исходных и зашифрованных сообщений: (23, 18707), (755, 26871), (631, 6384). Требуется расшифровать шифртекст *y* = 11 638. Для этого представим *y* в виде  Отсюда легко вычислить исходное сообщение: .

*Замечание*. Этот подход не менее трудоемок, чем поиск алгоритма решения сравнения 

**Взлом RSA при неудачном выборе параметров криптосистемы**

Само по себе использование RSA не обеспечивает безопасности. Дело еще в деталях реализации. Приведем ряд примеров. Для простоты вычислений будем работать с небольшими числами. Цель – показать особенности, не зависящие от размера.

*Пример*. Пусть пользователь выбрал *N =*2047, *e* = 179, *d =*411. Так как 2047 = 2389, а  имеют наименьшее общее кратное 88, то любой обратный к 179 по модулю 88, например 59, будет действовать как *d*.

*Пример*. Число *N* = 536813567 является произведением простого числа Мерсенна 8191 и простого числа Ферма 65537. Это очень плохой выбор.

*Пример*. Число 23360947609 является очень плохим выбором для *N* из-за того, что два его простых делителя слишком близки к друг другу. Пусть *p* > *q*, тогда имеем . Обозначим: . Так как *S* мало, то *t* – целое число, лишь немного большее  причем *t*2 – *N* является полным квадратом. Проверяем подряд целые числа  В нашем примере *t*1 = 152843, *t*2 = 152844, *t*3 = 152845 и  тогда *р =*152845 – 804. Таким образом, мы с третьей попытки нашли *p* и *q*. Количество попыток, необходимых для факторизации *N*, можно при известных *p* и *q* вычислить по следующей формуле: , где [*x*] – операция округления *x* до ближайшего целого числа.

**Атака Винера**

В некоторых приложениях требуется ускорить процесс расшифровывания в алгоритме RSA. Поэтому выбирается небольшая расшифровывающая экспонента. В случае когда расшифровывающая экспонента можно определить *d* за полиномиальное время с помощью атаки Винера, опирающейся на [непрерывные дроби](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%BD%D0%B0%D1%8F_%D0%B4%D1%80%D0%BE%D0%B1%D1%8C).

По вещественному числу определим последовательности:

Целые числа называются [непрерывной дробью](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%BD%D0%B0%D1%8F_%D0%B4%D1%80%D0%BE%D0%B1%D1%8C), представляющей , а рациональные числа — подходящими дробями. Каждая из подходящих дробей несократима, а скорость роста их знаменателей сравнима с показательной. Один из важных результатов теории [непрерывных дробей](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%BD%D0%B0%D1%8F_%D0%B4%D1%80%D0%BE%D0%B1%D1%8C):

Если несократимая дробь удовлетворяет неравенству:

То одна из подходящих дробей в разложении  \alpha в [непрерывную дробь](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%BD%D0%B0%D1%8F_%D0%B4%D1%80%D0%BE%D0%B1%D1%8C).

Пусть у нас есть модуль , причём . Допустим нападающему известна шифрующая экспонента *e*, обладающая свойством

,

где . Будем также считать, что , поскольку это выполнено в большинстве приложений. Из предположений следует, что . Следовательно,

Отсюда видно, что — довольно хорошее приближение для . Действительно,

Так как , очевидно, . Кроме того, так как предполагалось, что . Значит,

Поскольку , то — подходящая дробь в разложении дроби в [непрерывную](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BF%D1%80%D0%B5%D1%80%D1%8B%D0%B2%D0%BD%D0%B0%D1%8F_%D0%B4%D1%80%D0%BE%D0%B1%D1%8C). Таким образом, можно узнать расшифровывающую экспоненту, поочерёдно подставляя знаменатели подходящих дробей в выражение:

для некоторого случайного числа . Получив равенство, найдём . Общее число подходящих дробей, которое придётся проверить оценивается как .

**Атака повторным шифрованием**

Строим последовательность: . Итак, , а так как , то существует такое натуральное число m, что . Но тогда , отсюда следует, что , значит, *y*m-1 – решение сравнения .

*Пример*. Пусть у нас имеется открытый ключ *N* =84517 , *e* = 397 и зашифрованное им сообщение *y* = 8646. Необходимо найти исходный текст *x*. Возведем *y* в степень *e* и получим *y*2 = 37043. Будем повторять операцию до тех пор, пока не получим *yn* = *y. yn*-1 – искомое сообщение: *y*3 = 5569, *y*4 = 61833, *y*5 = 83891, *y*6 = 16137, *y*7 = 8646. *y*6 является решением сравнения , а, следовательно, искомым сообщением *x*.

*Замечание*. Анализ метода повторного шифрования хорошо показывает необходимость соблюдения требований на выбор *p* и *q* для обеспечения стойкости. В данном примере *d* = 82 225. Неудачный выбор криптосистемы привел к тому, что атака методом повторного шифрования дала результат почти сразу, тогда как нахождение *d* потребовало бы на порядок больших вычислений.

**Атака на основе Китайской теоремы об остатках.**

Как отмечалось ранее, системы шифрования с открытыми ключами работают сравнительно медленно. Для повышения скорости шифрования RSA на практике используют малую экспоненту зашифрования.

Если выбрать число *е* небольшим или таким, чтобы в его двоичной записи было мало единиц, то процедуру шифрова­ния можно значительно ускорить. Например, выбрав *е* = 3 (при этом ни *р* – 1, ни *q –* 1 не должны делиться на 3), мы сможем реализовать шифрование с помощью одного возведе­ния в квадрат по модулю *N* и одного перемножения. Выбрав  65 537 – число, двоичная запись которого содержит только две единицы, мы сможем реализовать шифрование с помощью 16 возведений в квадрат по модулю *N* и одного перемножения. Если экспонента *е* выбирается случайно, то реализация шифрования по алгоритму RSA потребует *s* воз­ведений в квадрат по модулю *N* и в среднем *s*/2 умножений по тому же модулю, где 5 – длина двоичной записи числа *N.* Вместе с тем выбор небольшой экспоненты *е* может привести к негативным последствиям. Дело в том, что у нескольких корреспондентов могут оказаться одинаковые экспоненты *е*.

Пусть, например, три корреспондента имеют попарно взаимно простые модули *N*1, *N*2, *N*3 и общую экспоненту *е =*3. Если еще один пользователь посылает им некое цирку­лярное сообщение *x*, то криптоаналитик противника может получить в свое распоряжение три шифрованных текста *i =*1,2, 3. Далее он может найти решение системы сравнений, лежащее в интервале 0 < *y* < *N*1∙*N*2∙*N*3



По китайской теореме об остатках такое решение единственно, а так как , то *y = x*3. Значение *х* можно найти, вычислив кубический корень .

Отметим, что выбор малой экспоненты расшифрования *d* также нежелателен в связи с возможностью определения *d* простым перебором. Известно также что если , то экспоненту *d* легко найти, используя непрерывные дроби.

*Пример*. Три пользователя имеют модули *N*1 = 26549, *N*2 = 45901, *N*3 = 25351. Все пользователи используют экспоненту *e* = 3. Всем пользователям было послано некое сообщение *x*, причем пользователи получили сообщения *y*1 = 5366, *y*2 = 814, *y*3 = 4454. Найдем *M*0 = *N*1∙*N*2∙*N*3 = 30893378827799. Далее находим

*m*1 = *N*2∙*N*3 = 1163636251

*m*2 = *N*1∙*N*3 = 673043699

*m*3 = *N*1∙*N*2 = 1218625649

*n*1 = *m*1-1 mod *N*1 = 13533

*n*2 = *m*2-1 mod *N*2 = 27930

*n*3 = *m*3-1 mod *N*3 = 22354

*S* = *y*1∙*n*1∙*m*1 + *y*2∙*n*2∙*m*2 + *y*3∙*n*3∙*m*3 = 84501028038745578 + 15301661957638980 + 121332116653000684 = 221134806649385242

*S* mod *M*0 = 1000000000

*x* = (*S* mod *M*0)1/3 = 1000 – исходное сообщение, отправленное пользователям.

**Бесключевое чтение**

Пусть два пользователя выбрали одинаковый модуль *N* и разные экспоненты *e*1 и *e*2. Если один пользователь посылает им некое циркулярное сообщение *x*, то криптоаналитик противника может получить в свое распоряжение два шифрованных текста  и  В таком случае криптоаналитик может получить исходное сообщение, используя расширенный алгоритм Евклида, находим  такие, что . Отсюда получаем: 

*Пример*. Два пользователя применяют общий модуль *N* = 137759, но разные взаимно простые экспоненты *e*1 = 191 и *e*2 = 233. Пользователи получили шифртексты *y*1 = 60197 и *y*2 = 63656, которые содержат одно и то же сообщение. Найдем исходное сообщение методом бесключевого чтения. Так как *e*1 и *e*2 взаимно просты, то найдем такие *r* и *s*, что  С помощью расширенного алгоритма Евклида находим *r* = 61, *s* = –50. Искомое сообщение 

**Закладки в RSA.**

Реализовать расшифровку сообщений, зашифрованных ключами с закладкой, согласно статье [«Встраиваем бэкдор в публичный ключ RSA»](http://habrahabr.ru/post/248269/)

## Задание

Последовательно изучить приемы взлома RSA при неудачном выборе параметров криптосистемы (случай близко расположенных *p* и *q*), атаки: Винера, повторным шифрованием, бесключевым чтением. Разработать и реализовать алгоритмы, проводящие указанные атаки. Оценить производительность методов атаки, построить сравнительные графики зависимости скорости работы от размера ключа.

По статье [«Встраиваем бэкдор в публичный ключ RSA»](http://habrahabr.ru/post/248269/) реализовать расшифровку сообщений, зашифрованных ключами с закладкой. Оценить накладные расходы на расшифровку данных при внедренной закладке.

## Порядок выполнения

1. Изучить примеры взлома RSA при неудачном выборе параметров криптосистемы.
2. Разработать и реализовать алгоритм, факторизацией *N* при близко расположенных *p* и *q* (отличающихся не менее чем на порядок).
3. Изучить атаку на алгоритм шифрования RSA методом Винера.
4. Разработать и реализовать алгоритм, позволяющий по известным значениям модуля *N*, и экспоненты *e*, с учетом ограничений метода Винера, получить исходный текст.
5. Изучить атаку на алгоритм шифрования RSA посредством повторного шифрования.
6. Разработать и реализовать алгоритм, позволяющий по известным значениям модуля *N*, и экспоненты *e*, используя определение порядка *e* в конечном поле, получать исходный текст методом перешифрования.
7. Изучить атаку на алгоритм шифрования RSA посредством метода бесключевого чтения.
8. Разработать и реализовать алгоритм, позволяющий по известным значениям модуля *N*, и экспонент *e*1 и *e*2 определить значения *r* и *s* при условии, чтобы *e*1∙*r – e*2∙*s*=1. Для этого необходимо использовать расширенный алгоритм Евклида. Используя полученные значения r и s, получить исходный текст.
9. Практически оценить производительность методов атаки, построить сравнительные графики зависимости скорости работы от размера ключа.
10. Реализовать расшифровку сообщений, зашифрованных ключами с закладкой, согласно статье [«Встраиваем бэкдор в публичный ключ RSA»](http://habrahabr.ru/post/248269/).
11. Практически оценить накладные расходы на расшифровку данных при внедренной закладке.

## Содержание отчета

1. Цель работы.
2. Задание.
3. Краткая теория.
4. Выдержки из текста программы, реализующие критичные для работы алгоритмов атаки участки.
5. Примеры выбора значений криптосистемы и результаты работы соответствующих этим примерам методов атаки.
6. Графики производительности.
7. Выводы по работе.