

# Математические основы защиты информации

## Лабораторная работа №6

### Криптосистема RSA

БГУ, ММФ, Каф. ДУиСА,  
доцент Чергинец Д.Н.

## Односторонние функции

В 1976 г. американские математики У.Диффи и М.Э.Хеллман ввели новый тип криптосистем — криптосистемы с *открытым ключом*. В ее основе лежит идея использования для шифрования односторонних функций.

**Определение.** Отображение  $f: X \rightarrow Y$  называется *односторонней* (однонаправленной) функцией, если существует полиномиальный алгоритм вычисления  $f(x)$  для любых  $x \in X$ , но не существует полиномиального алгоритма вычисления  $f^{-1}(y)$  для большинства случайно выбранных  $y$  из области значений  $Y$ .

Для криптографии представляют интерес специальные односторонние функции — односторонние функции с лазейкой (или секретом) (one-way trap-door function). Эти функции являются односторонними, если некоторая информация о функции  $f$  остается в секрете.

Точнее, функция с лазейкой  $f_k: X \rightarrow Y$  зависит от параметра  $k$  (секретный ключ),  $f_k(x)$  вычисляется за полиномиальное время, независимо от того, знаем мы  $k$  или нет;

если  $k$  известно, то  $f_k^{-1}(y)$  вычисляется за полиномиальное время;

если же  $k$  неизвестно, то не существует (на данное время) полиномиального алгоритма, вычисляющего  $f_k^{-1}(y)$ .

В криптографии множество  $X$  — это множество открытых сообщений,  $Y$  — множество зашифрованных текстов,

$f_k(x)$  - алгоритм шифрования,  $f_k^{-1}(y)$  - алгоритм дешифрования.

Условия на функцию с лазейкой  $f_k$  означают, что любой пользователь  $A$  может

послать по открытому каналу связи сообщение  $y = f_K(x)$  любому пользователю **B**. Способ шифрования содержится в доступном для всех справочнике.

Пользователь **B**, используя свой секретный ключ  $k$ , легко дешифрует сообщение  $y = f_K(x)$ , т.е. найдет  $x = f^{-1}_K(y)$  за полиномиальное время. Однако, никто другой без знания  $k$  не сможет за полиномиальное время восстановить  $x$ .

Так как существование односторонних функций до настоящего времени не доказано (их существование эквивалентно тому, что  $P \neq NP$ ), то в качестве функций с лазейкой  $f_K$  берутся функции, для которых вычисление  $f^{-1}_K(y)$  без знания дополнительной информации  $k$  является трудной математической задачей на данный момент времени.

## Криптосистема RSA

Рассмотрим функцию  $f(x) = x^e \pmod n$ ,  $e$  и  $n$  – некоторые натуральные числа.

Какими бы не были числа  $x$ ,  $e$ ,  $n$ , значение функции  $f$  в точке  $x$ , как мы видели в предыдущих лабораторных работах, считается за полиномиальное время. В Mathematica это делает

функция PowerMod. Рассмотрим обратную задачу, найти такое  $x$ , что  $x^e = y \pmod n$ . Алгоритма, который решал бы данную задачу за полиномиальное время, еще не найдено, но и не доказано, что его не существует (что наиболее вероятно).

Воспользуемся теоремой

Эйлера. Пусть мы знаем  $\varphi(n)$ . Тогда мы можем найти

$$d = e^{-1} \pmod{\varphi(n)}$$

и вычислить  $x$

$$y^d = x^{e \cdot d} = x^{1 + a \cdot \varphi(n)} = x \pmod n$$

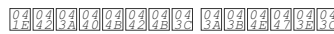
Таким образом, если мы знаем  $d$  (секретный ключ),

то мы можем дешифровать шифротекст  $y$ . На этой идее основан алгоритм RSA.

Алгоритм шифрования RSA имеет вид:

### Генерация ключей G[L]

1. Случайным образом выбираем простые числа  $p$  и  $q$  длины  $L$  бит,  $p \neq q$ .
2. Находим число  $n := p \cdot q$ .
3. Вычисляем функцию Эйлера  $\varphi(n) := (p - 1)(q - 1)$ .
4. Выбираем случайное число  $e$ , удовлетворяющее условиям  $1 < e < \varphi(n)$ ,  $\text{НОД}(e, \varphi(n)) = 1$ .
5. При помощи расширенного алгоритма Евклида находим число  $d = e^{-1} \pmod{\varphi(n)}$ .

 (который не скрывают от злоумышленников и используют при шифровании сообщения) являются числа **e** и **n**. **PublicKey = {e,n}**.  
Закрытым (секретным) ключом является число **d**. **PrivateKey = {d}**.

Отметим, что числа **p, q, φ(n)** при дальнейших расчетах не нужны, но являются секретными, так как с их помощью можно определить секретный ключ **d**.

### Шифрование

Для того чтобы зашифровать число **m**,  $1 < m < n$ , необходимо вычислить шифртекст по формуле  $c = m^e \pmod n$

### Дешифрование

Для того чтобы получить открытое сообщение, необходимо воспользоваться секретным ключом **d**  
 $m = c^d \pmod n$ .

### Обоснование

Почему же  $c^d = m \pmod n$ ?

Если **НОД(m,n)=1**, то равенство следует из теоремы

Эйлера:

$$c^d = m^{ed} = m^{1+\varphi(n)} = m \pmod n.$$

Теперь покажем, что  $m = c^d \pmod n$  и для **m**, не взаимно простых с **n**, т.е.  $m=ps$  или  $m=q s$ .  
Пусть, для определенности,  $m=ps$  ( $(m,q)=1$ ).

Тогда

$$c^d = p^{ed} s^{ed} = 0 = m \pmod p$$

$$c^d = m^{ed} = m^{1+b\varphi(q)} = m \pmod q$$

Следовательно,

$$c^d = m \pmod n.$$

### Задание 1.

Написать функцию генерации ключей **GenerationKey[ L ]**, которая возвращает ключи **PublicKey**, **PrinateKey** криптосистемы RSA, где  $n = p q$ , простые числа **p** и **q** длины **L** бит генерируются при помощи изученного нами ранее алгоритма Миллера-Рабина или теоремы Диемитко.

### Задание 2.

Задать функцию **EnRSA**[**m\_Integer**, **PublicKey\_**], которая шифрует число **m**,  $0 < m < n$ , при помощи открытого ключа **PublicKey** = {**e**,**n**}.

### Задание 3.

Определить дешифрующую функцию **DeRSA**[**c\_Integer**, **PublicKey\_**, **PrivateKey\_**], возвращающую число **m**.

## Криптоанализ RSA

**Алгоритм. Разложение модуля на множители по известным показателям RSA.**

**Вход:**  $n, e, d \in \mathbb{N}$

**Выход:**  $p, q$ .

1. При помощи последовательного деления на 2 представляем число  $e \cdot d - 1$  в виде  $2^s t$ , где  $t$  – нечетное.
2. Выбираем случайное  $a \in \mathbb{N}$ ,  $1 < a < n - 1$ .
3. Если  $\text{НОД}(a, n) > 1$ , то  $p := \text{НОД}(a, n)$ ,  $q := \frac{n}{p}$ , конец алгоритма.
4. Вычисляем  $u := a^t \pmod{n}$ ,  $v := u^2 \pmod{n}$ .
5. Если  $u = 1$ , то переходим к шагу 2.
6. Пока  $v \neq 1$  вычисляем  $u := v$ ,  $v := u^2 \pmod{n}$ .
7. Если  $u \equiv -1 \pmod{n}$ , то переходим к шагу 2, иначе вычисляем  $p := \text{НОД}(u + 1, n)$ ,  $q := \text{НОД}(u - 1, n)$ , конец алгоритма.

### Задание 4.

По известным открытому **PublicKey** = {**e**,**n**} и секретному **PrivateKey** = **d** ключу найти разложение на множители числа **n**. Представить число **e** в

двоичной системе счисления. Почему в криптосистеме RSA зачастую берут именно такое число  $e = 65\,537$ ? Является ли описанный выше алгоритм, вычисляющий  $p, q$  по  $e, d, n$ ,

- полиномиальным?
- вероятностным?
- всегда ли выдает правильный ответ?

Вариант 1.

Вариант 2.

Вариант 3.

Вариант 4.

Вариант 5.

Вариант 6.

Вариант 7.

Вариант 8.

Вариант 9.

Вариант 10.

Вариант 11.

Вариант 12.

Вариант 13.

Вариант 14.

Вариант 15.

Вариант 16.

Вариант 17.

Вариант 18.

Вариант 19.

Вариант 20.

Вариант 21.

Вариант 22.

Вариант 23.

Вариант 24.

Вариант 25.

Вариант 26.

Вариант 27.

Вариант 28.

Вариант 29.

Вариант 30.

### Задание 5.

Ознакомиться со встроенной функцией `GenerateAsymmetricKeyPair[]`.

Из скольких простых чисел состоит модуль  $n$  в открытом ключе `PublicKey` криптосистемы RSA? Делители модуля состоят из одинакового количества бит или могут иметь различную длину в битах?