ТЕОРИЯ ЧИСЕЛ И КРИПТОГРАФИЯ

К. Конрад

1. Введение

Криптография занимается изучением секретных (тайных) сообщений. На протяжении всей человеческой истории криптография интересовала в основном военных и дипломатов (поищите в сети "телеграмма Циммермана" — это пример того, как их интересы переплелись). Но с развитием интернета в конце 20 века криптография вошла в жизнь каждого человка. Всякий раз, когда мы посылаем текстовое сообщение, покупаем что-то в интернет-магазине или переводим деньги в электронном банке, криптография обеспечивает — по крайней мере, мы надеемся на это — что никто, кроме вашего адресата, не получит доступ к передаваемой информации.

Основные криптографические протоколы, используемые сейчас повсеместно миллионами людей, опираются на теорию чисел. В Разделе 2 мы обсудим некоторые криптографические приёмы, бывшие в ходу до компьютерной эры. Они используют арифметику остатков и линейную алгебру. В Разделах 3-5 мы опишем один из наиболее распространённых сейчас криптографических протоколов RSA, названный по фамилиям его создателей: Ривест, Шамир и Адлеман (Rivest, Shamir, Adleman) [8]. Он основан на свойствах операции возведения в степень по модулю и использует алгоритм Евклида, малую теорему Ферма и алгоритмы проверки чисел на простоту. Раздел 6 посвящён обсуждению истории открытия этого протокола.

С криптографией, целью которой является сохранение в секрете от посторонних передаваемой информации, тесно связана теория кодирования. Но задачи у неё другие — например, как надёжно передать информацию по каналу связи с помехами. Подобные задачи часто возникают, например, в телефонных сетях, при обработке цифровых аудиофайлов, обмене сообщениями между NASA на Земле и спутниками в космосе. Теория чисел играет важную роль и в теории кодирования, но последнюю мы здесь обсуждать не будем.

2. Шифр Цезаря и шифр Хилла

Один из самых старых известных способов шифрования обычно приписывается Юлию Цезарю. Устроен он так: мы сдвигаем каждую букву сообщения на фиксированную величину. Например, если сдвинем на 3 буквы вправо, то получим

$$\mathrm{A} \mapsto \mathrm{D}, \mathrm{B} \mapsto \mathrm{E}, \mathrm{C} \mapsto \mathrm{F}, \ldots, \mathrm{Z} \mapsto \mathrm{C}.$$

Сообщение

DOODLE

в зашифрованном виде выглядит так:

GRRGOH.

Зашифрованное сообщение

XKFJXI

дешифруется сдвигом на три буквы влево, получается такое сообщение

ANIMAL.

Такой алгоритм шифрования со сдвигом букв на фиксированное расстояние называется *шифром Цезаря* или *шифром сдвига*. У него есть по меньшей мере три недостатка:

(1) Знание алгоритма шифрования раскрывает алгоритм дешифрования: если известно, что при шифровании все буквы были сдвинуты на пять позиций вправо, то для дешифрования сообщения их надо сдвинуть на пять позиций влево. Таким образом, для повышения надёжности шифра надо хранить значение сдвига в тайне, хотя следующий недостаток показывает, что идея, лежащая в основе шифра Цезаря, ущербна по своей природе.
(2) Даже если мы не знаем точное значение сдвига — выбор невелик: их всего 25. Так что если нам известно, что сообщение зашифровано шифром Цезаря, то можно вооружиться терпением и попробовать все сдвиги, пока не получим осмысленный текст.
(3) Поскольку все буквы сдвигаются на одно и то же значение, то как только мы узнали зашифрованный вариант одной буквы, мы узнали сдвиг всех букв. Например, если мы определили, что буква B шифруется буквой F, это значит буква B сдвинулась на 4 позиции, а значит и все буквы сдвинулись на 4 позиции.

Переводя на язык математики, можно сказать, что шифр Цезаря это сложение букв сообщения с фиксированным числом по модулю 26, где A = 1, B = 2, и так далее до Z = 26 = 0.

Сдвиг на 3 буквы вправо это шифрующая функция $E(x) = x+3 \bmod 26$. Дешифруется сообщение функцией $D(y) = y-3 \bmod 26$.[1] Мы можем немного усложнить алгоритм, умножая значение буквы перед сдвигом: $E(x) = ax + b \bmod 26$, где $(a, 26) = 1$. В шифре Цезаря $a = 1$. Требование $(a, 26) = 1$ нам нужно для обращения функции $E(x)$, то есть для дешифровки сообщения. Пусть $E(x) = 5x + 3 \bmod 26$. Подставляя значения $x = 1, 2, 3, \ldots, 25, 0$ получим такое шифрование

$$\mathrm{A} \mapsto \mathrm{H}, \mathrm{B} \mapsto \mathrm{M}, \ldots, \mathrm{Z} \mapsto \mathrm{C}.$$

Если $E(x) = x + 3 \bmod 26$, то слово DOODLE после шифрования станет GRRGOH. Если $E(x) = 5x + 3 \bmod 26$, то слово DOODLE станет WZZWKB (проверьте!) и если $E(x) = 9x + 3 \bmod 26$, то слово DOODLE станет MHHMGV (проверьте!).

Для дешифровки сообщения нам надо найти функцию, обратную к линейной. В обычной алгебре, если $y = 5x + 3$, то $x = (1/5)(y - 3)$. Это работает и для вычислений по модулю 26, надо только заменить $1/5$ на число, обратное 5 по модулю 26. Это 21 (проверьте!). Таким образом если $E(x) = 5x+3 \bmod 26$, то соответствующая дешифрующая функция это $D(y) = 21(y-3) = 21y+15$ (потому что $21(-3) = -63 \equiv 15 \bmod 26$).

Вводя в алгоритм шифрования умножение по модулю 26, мы увеличиваем количество шифрующих функций более, чем в 10 раз: количество функций вида $E(x) = x + b \bmod 26$ равно 26 (точнее 25, потому что функция $E(x) = x \bmod 26$ как шифр не

[1] Выбор букв для названия функций и показателя степени неслучаен: **E**ncrypt — шифровать, **D**ecrypt — дешифровать. — *Прим. перев.*

имеет смысла), в то время как количество функций вида $E(x) = ax + b \bmod 26$ равно $\varphi(26) \cdot 26 = 12 \cdot 26 = 312$.[2]

Хотя это обобщение шифра Цезаря решает проблему сдвига всех букв на одно и то же значение, в нём остаётся изъян, нами пока не упомянутый: буква шифруется одним и тем же образом вне зависимости от того, в каком месте текста она встречается. Например, когда DOODLE при шифровании превращается в GRRGOH функцией $E(x) = x + 3 \bmod 26$, буква D превращается в букву G оба раза, равно как и двойная O становится двойной R. Если DOODLE превращается в WZZWKB функцией $E(x) = 5x + 3 \bmod 26$, то D становится W оба раза, в двойная O становится двойной Z.

Хорошо известно распределение частот букв, встречающихся в текстах на английском языке: самая частая, например, буква E, вторая по частоте буква A. Поэтому зная, что достаточно большой текст зашифрован функцией вида $E(x) = ax + b \bmod 26$, мы можем с достаточной степенью уверенности определить какие зашифрованные буквы соответствуют буквам E и A (в может и другим частотным буквам). Точно так же, как по двум точкам мы можем восстановить уравнение прямой, информации о значениях функции $E(x) = ax + b \bmod 26$ в двух точках достаточно для того, чтобы определить a и b и узнать тем самым шифрующую функцию.

Мы можем пойти дальше в усложнении шифра Цезаря, заменив числа и линейные функции векторами и матрицами. Такие шифры называются *блочными*, потому что элементом шифрования является не одна буква, а группа букв. Покажем, как это работает на примере блоков из 2 букв и матриц 2×2. Например,

DOODLE

разбивается на блоки из 2 букв

DO OD LE

и каждый блок можно рассматривать, как вектор чисел по модулю 26 (A = 1, B = 2, и так далее):

$$\begin{pmatrix}\mathrm{D}\\ \mathrm{O}\end{pmatrix} = \begin{pmatrix}4\\ 15\end{pmatrix}, \quad \begin{pmatrix}\mathrm{O}\\ \mathrm{D}\end{pmatrix} = \begin{pmatrix}15\\ 4\end{pmatrix}, \quad \begin{pmatrix}\mathrm{L}\\ \mathrm{E}\end{pmatrix} = \begin{pmatrix}12\\ 5\end{pmatrix}.$$

Выберем матрицу 2×2 со значениями по модулю 26, например такую:

$$\mathcal{A} = \begin{pmatrix}3 & 2\\ 1 & 11\end{pmatrix}.$$

Чтобы зашифровать блок из двух букв, рассматривая его как вектор значений по модулю 26, умножим этот вектор на матрицу $\mathcal{A}$, выполняя все вычисления по модулю 26:

$$\mathcal{A}\begin{pmatrix}4\\ 15\end{pmatrix} = \begin{pmatrix}16\\ 13\end{pmatrix}, \quad \mathcal{A}\begin{pmatrix}15\\ 4\end{pmatrix} = \begin{pmatrix}1\\ 7\end{pmatrix}, \quad \mathcal{A}\begin{pmatrix}12\\ 5\end{pmatrix} = \begin{pmatrix}20\\ 15\end{pmatrix}.$$

После перевода каждой компоненты вектора обратно в букву получим в зашифрованном DOODLE первую букву 16 = P, вторую букву 13 = M, третью букву 1 = A, и так далее, собирая таким образом зашифрованное слово

PMAGTO.

Обратите внимание — обе буквы D и обе буквы O в слове DOODLE были зашифрованы разными буквами. Такой шифр уже не получится взломать, используя распределение

[2] Дотошный читатель заметит, что мы здесь не учли бессмысленную функцию при $a = 1$ и $b = 0$ и будет прав. — *Прим. перев.*

частот отдельных букв в английском языке, хотя он может поддаться, если знать распределение частот двухбуквенных комбинаций.

Вернёмся к алгоритму и займёмся дешифровкой. Предположим мы получили сообщение

UPFOOU

и мы знаем, что оно было получено блочным шифрованием с указанной матрицей $\mathcal{A} = \left(\begin{smallmatrix} 3 & 2 \\ 1 & 11 \end{smallmatrix}\right)$. Поскольку функция шифрования $E(\mathbf{v}) = \mathcal{A}\mathbf{v}$, дешифровка получается умножением на матрицу, обратную $\mathcal{A}$: $D(\mathbf{v}) = \mathcal{A}^{-1}\mathbf{v}$. Когда матрица обратима? Из курса линейной алгебры известно, что (квадратная) матрица с вещественными элементами обратима тогда и только тогда, когда её определитель не равен нулю. В нашем случае 2×2 матрица $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ имеет обратную

$$\frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}.$$

Для матриц, элементы которых целые числа по модулю m, ненулевой определитель *не является* критерием обратимости. Рассмотрим, например, при $m = 26$ матрицу $\left(\begin{smallmatrix} 1 & 2 \\ 3 & 4 \end{smallmatrix}\right)$, её определитель равен $-2 \equiv 24 \bmod 26$, что не равно $0 \bmod 26$. Тем не менее эта матрица необратима по модулю 26: не существует такой 2×2 матрицы M что $\left(\begin{smallmatrix} 1 & 2 \\ 3 & 4 \end{smallmatrix}\right)M \equiv \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right) \bmod 26$.

Теорема 2.1. *Матрица 2×2 с целыми элементами по модулю m имеет обратную тогда и только тогда, когда её определитель обратим по модулю m, при этом обратная к ней матрица вычисляется по той же формуле, что и в случае матрицы с вещественными элементами.*

Доказательство. Пусть A — матрица. Если A имеет обратную матрицу B, то $AB \equiv \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right) \bmod m$, после взятия определителя левой и правой частей имеем $(\det A)(\det B) \equiv 1 \bmod m$. Значит $\det A$ обратим по модулю m.

И наоборот, если $\det A$ обратим по модулю m, то непосредственным вычислением можно убедиться в том, что уже знакомая формула для обратной матрицы с вещественными элементами даёт нам верный ответ. □

Определитель матрицы $\mathcal{A} = \left(\begin{smallmatrix} 3 & 2 \\ 1 & 11 \end{smallmatrix}\right)$ по модулю 26 равен $31 \equiv 5 \bmod 26$. Обратный элемент по модулю 26 равен 21. Значит

$$\mathcal{A}^{-1} = 21\begin{pmatrix} 11 & -2 \\ -1 & 3 \end{pmatrix} \equiv \begin{pmatrix} 23 & 10 \\ 5 & 11 \end{pmatrix}.$$

Обозначим эту матрицу $\mathcal{B}$. (Проверьте: $\mathcal{A}\mathcal{B} = I_2$ и $\mathcal{B}\mathcal{A} = I_2$.) Зашифрованное сообщение UPFOOU разбивается на блоки

$$\begin{pmatrix} \mathrm{U} \\ \mathrm{P} \end{pmatrix} = \begin{pmatrix} 21 \\ 16 \end{pmatrix}, \quad \begin{pmatrix} \mathrm{F} \\ \mathrm{O} \end{pmatrix} = \begin{pmatrix} 6 \\ 15 \end{pmatrix}, \quad \begin{pmatrix} \mathrm{O} \\ \mathrm{U} \end{pmatrix} = \begin{pmatrix} 15 \\ 21 \end{pmatrix},$$

дешифруем это сообщение, умножая (по модулю 26) каждый вектор на матрицу $\mathcal{B}$ слева:

$$\mathcal{B}\begin{pmatrix} 21 \\ 16 \end{pmatrix} = \begin{pmatrix} 19 \\ 21 \end{pmatrix}, \quad \mathcal{B}\begin{pmatrix} 6 \\ 15 \end{pmatrix} = \begin{pmatrix} 2 \\ 13 \end{pmatrix}, \quad \mathcal{B}\begin{pmatrix} 15 \\ 21 \end{pmatrix} = \begin{pmatrix} 9 \\ 20 \end{pmatrix}.$$

Собирая получившиеся блоки вместе и преобразуя их в буквы, получим дешифрованное сообщение:

SUBMIT.

Используя матрицу размера 3×3 мы можем шифровать сообщения, разбивая их на блоки по 3 буквы. При использовании матриц размера 6×6 и больше попытки взлома шифра, опирающиеся на распределение соответствующих частот уже затруднительны. Такая техника шифрования, использующая матрицы и арифметику остатков была предложена Лестером Хиллом (Lester Hill) в конце 1920-х годов и называется *шифром Хилла*. Несмотря на то, что шифр Хилла смог избежать некоторых недостатков шифра Цезаря (например, одинакового шифрования одинаковых букв), он по-прежнему обладает одним из дефектов шифра Цезаря: знание функции шифрования и функции дешифрования по сути эквивалентны. Любой, кто знает матрицу шифрования в шифре Хилла может вычислить обратную к ней матрицу дешифрования. В немецкой шифровальной машине Энигма (см. Рис 1, фото из каталога Computer History Museum) шифрование и дешифрование были буквально одной и той же операцией. Можно посмотреть видео [4], [5] и [7] с иллюстрацией работы этой шифровальной машины.

Рис. 1. Немецкая шифровальная машина Энигма

Этот недостаток шифров Цезаря, Хилла и шифровальной машины Энигма был неотъемлемой чертой криптографии на протяжении всей истории человечества: функции шифрования и дешифрования всегда являлись симметричными процессами в том смысле, что зная способ шифрования сообщений можно было легко реконструировать алгоритм дешифрования. Да и как можно иначе?

В 1970-х годах было установлено, что это *может* быть иначе: существует криптографический алгоритм, основанный на теории чисел, способ шифрования которого можно сообщить всем, оставив операцию дешифрования надёжной. Одной из первых реализаций этой идеи стал алгоритм RSA. Функция шифрования в нём это функция

возведения в степень по модулю: $E(x) = x^e \bmod m$ для подходящих m и e. Дешифрование это тоже возведение в степень: $D(y) = y^d \bmod m$ для того же модуля m и другого показателя d. Надёжность алгоритма обеспечивается с одной стороны простотой операции умножения (возведение в степень при шифровании), с другой — сложностью разложения на множители (для определения показателя степени при дешифровании). Но обо всём по порядку.

3. Прелюдия к RSA: решение $a^n \equiv b \bmod m$ относительно a

Чтобы понять принцип работы алгоритма RSA, разберёмся как вычислять дискретный корень по данному модулю, а именно: как по известному модулю m, положительному целому n и степени $a^n \bmod m$ найти $a \bmod m$.

Пример 3.1. Найдём a, удовлетворяющее отношению $a^3 \equiv 14 \bmod 55$. Конечно, мы могли бы перебрать все значения $a = 0, 1, 2, \ldots, 54$ пока не найдём ответ, но хочется подойти к этой задаче более грамотно, учитывая, что 55 скорее прототип гораздо большего на практике модуля (слишком большого для простой иллюстрации идеи).

Правая часть нашего сравнения, 14, обратима по модулю 55, так что искомое значение a, если оно вообще существует, должно быть обратимо по модулю 55 (а значит взаимно просто с 55). Следовательно, пользуясь теоремой Эйлера, в выражении $a^n \bmod 55$ уместно рассматривать n по модулю $\varphi(55) = \varphi(5)\varphi(11) = (5-1)(11-1) = 40$. Значит для каждого целого $t \geq 1$

$$a^{n+40t} = a^n(a^{40})^t \equiv a^n \bmod 55$$

поскольку $a^{40} \equiv 1 \bmod 55$ по теореме Эйлера.

Кроме того, если $a^3 \equiv 14 \bmod 55$, то после возведения обеих частей в степень $k \geq 1$ имеем

$$a^{3k} \equiv 14^k \bmod 55.$$

Если мы найдём такое положительное целое k, что $3k \equiv 1 \bmod 40$, то $a^{3k} \equiv a^1 \equiv$ $\equiv a \bmod 55$. Решение сравнения $3k \equiv 1 \bmod 40$ сводится к нахождению числа, обратного 3 по модулю 40. Поскольку $(3, 40) = 1$, обратный элемент можно найти, используя, например, расширенный алгоритм Евклида: это 27 mod 40. Таким образом

$$a^3 \equiv 14 \bmod 55 \Longrightarrow a^{3\cdot 27} \equiv 14^{27} \bmod 55 \Longrightarrow a \equiv 14^{27} \bmod 55.$$

Посчитаем: $14^{27} \equiv 9 \bmod 55$, значит $a \equiv 9 \bmod 55$. Проверим результат:

$$9^3 = 729 = 14 + 715 = 14 + 55 \cdot 13 \equiv 14 \bmod 55.$$

В решении сравнения $a^3 \equiv 14 \bmod 55$ было принципиальным, что 3 взаимно просто с $\varphi(55) = 40$. Изменив показатель с 3 на 6, который не взаимно прост с 40, получим сравнение $a^6 \equiv 14 \bmod 55$, которое имеет не одно, а четыре решения $\pm 3, \pm 8 \bmod 55$, а сравнение $a^6 \equiv 19 \bmod 55$ не имеет решений вовсе.

Если $(a, m) = 1$, то выражение $a^n \bmod m$ зависит от основания a и показателя n *по-разному*: основание a мы вычисляем по модулю m, а показатель n по модулю $\varphi(m)$. *Это важно понять!* Отличающиеся модули для основания (mod m) и показаталя (mod $\varphi(m)$), наряду со сложностями при вычислении $\varphi(m)$, если мы не знаем разложения m на множители, составляют в общих чертах идею, которая делает RSA надёжным алгоритмом шифрования.

4. Как работает RSA

Первый компонент в алгоритме RSA это модуль m, равный произведению двух различных простых чисел p и q. На практике используются числа из нескольких сотен разрядов, но для примера выберем p и q поменьше. Функцию Эйлера $\varphi(m) = (p-1)(q-1)$ легко вычислить, зная p и q.

Второй компонент это целое число $e \geq 1$ такое, что $(e, \varphi(m)) = 1$. Это показатель степени в функции шифрования. Для проверки взаимной простоты $(e, \varphi(m)) = 1$ можно использовать алгоритм Евклида.

Третий компонент это целое число $d \geq 1$ такое, что $ed \equiv 1 \bmod \varphi(m)$. Это показатель степени в дешифрующей функции. Зная взаимно простые e и $\varphi(m)$ можно найти d — это взаимно обратное числу e по модулю $\varphi(m)$.

Теперь всё готово к реализации алгоритма RSA. Выбрав простые p and q, положим $m = pq$, и выберем $e \geq 1$ такое, что $(e, \varphi(m)) = 1$. Затем найдём $d \geq 1$ из отношения $ed \equiv 1 \bmod \varphi(m)$. Сообщим всем числа m and e.

Числа m and e называются вашим *публичным ключом.*

Простые числа p, q и число d это ваш *приватный (секретный) ключ.*

Отправляемые вам сообщения должны быть целыми числами x, причём $0 \leq x \leq m-1$ (более длинные сообщения можно разбить на куски, меньшие m; такие примеры будут приведены ниже). Тот, кто хочет отправить вам сообщение x, должен вместо этого отправить зашифрованное сообщение $y = E(x) = x^e \bmod m$. Чтобы его дешифровать, вычислите $D(y) \equiv y^d \bmod m$.

Пример 4.1. Если $p = 43$ и $q = 97$, тогда $m = pq = 4171$ и $\varphi(m) = 42 \cdot 96 = 4032$. Поскольку $(5, \varphi(m)) = 1$ возьмём $e = 5$. Из сравнения $5d \equiv 1 \bmod \varphi(m)$ получим $d = 1613$. Получим таким образом пару функций, шифрующую и дешифрующую:

$$E(x) = x^5 \bmod m$$

$$D(y) = y^{1613} \bmod m$$

Например, если $x = 24$, то $E(x) = 24^5 \equiv 185 \bmod m$ и $D(185) = 185^{1613} \equiv 24 \bmod m$.

Покажем, что процессы шифрования и дешифрования взаимно обратны. Для этого вычислим результат применения функций шифрования и дешифрования к произвольному числу x:

$$D(E(x)) \equiv E(x)^d \bmod m \equiv (x^e)^d \bmod m \equiv x^{ed} \bmod m,$$

поэтому нам надо проверить, что

$$ed \equiv 1 \bmod \varphi(m) \Longrightarrow x^{ed} \equiv x \bmod m$$

вне зависимости от выбора x. Зная $x \bmod m$, мы знаем и сам x, если $0 \leq x \leq m-1$.

Теорема 4.2. *Пусть p и q различные простые числа и $m = pq$. Если положительные целые e и d выбраны таким образом, что $ed \equiv 1 \bmod \varphi(m)$, то для всех $x \in \mathbf{Z}$,*

$$x^{ed} \equiv x \bmod m.$$

Доказательство. По предположению $ed = 1 + \varphi(m)t$ где $t \geq 0$. Для каждого целого x,

$$x^{ed} = x^{1+\varphi(m)t} = x(x^{\varphi(m)})^t.$$

Если $(x, m) = 1$, то $x^{\varphi(m)} \equiv 1 \bmod m$ по теореме Эйлера, так что $x^{ed} = x(x^{\varphi(m)})^t \equiv$ $\equiv x \bmod m$.

Если $(x, m) > 1$, то $x^{\varphi(m)} \not\equiv 1 \bmod m$ (степень числа x не может быть сравнима с $1 \bmod m$, если у x и m есть общий множитель больше 1). Тем не менее мы покажем, что сравнение $x^{ed} \equiv x \bmod m$, или эквивалентное ему $x \cdot x^{\varphi(m)t} \equiv x \bmod m$ выполняется, используя представление m в виде произведения двух различных простых p и q (мы ещё ни разу не использовали этот факт). Приведённое ниже доказательство будет работать при всех $x \in \mathbf{Z}$, и предположение $(x, m) = 1$ о взаимной простоте x и m становится избыточным.

Поскольку p и q взаимно просты, для всех $x \in \mathbf{Z}$ имеем

$$x \cdot x^{\varphi(m)t} \equiv x \bmod pq \Longleftrightarrow x \cdot x^{\varphi(m)t} \equiv x \bmod p \text{ и } x \cdot x^{\varphi(m)t} \equiv x \bmod q.$$

Поскольку $\varphi(m) = \varphi(p)\varphi(q) = (p-1)(q-1)$, то $x \cdot x^{\varphi(m)t} = x \cdot x^{(p-1)(q-1)t}$. Для доказательства сравнения

$$x \cdot x^{(p-1)(q-1)t} \stackrel{?}{\equiv} x \bmod p$$

для всех $x \in \mathbf{Z}$, рассмотрим два случая:

1) если $x \equiv 0 \bmod p$, то обе части $0 \bmod p$, следовательно они сравнимы.
2) если $x \not\equiv 0 \bmod p$, то $x^{p-1} \equiv 1 \bmod p$ по малой теореме Ферма, значит $x^{(p-1)(q-1)t} \equiv 1 \bmod p$. Получаем $x \cdot x^{(p-1)(q-1)t} \equiv x \cdot 1 \equiv x \bmod p$.

Обоснование сравнения $x \cdot x^{(p-1)(q-1)t} \equiv x \bmod q$ для всех $x \in \mathbf{Z}$ доказывается так же, с заменой p на q (рассматривая случаи, когда $x \equiv 0 \bmod q$ и $x \not\equiv 0 \bmod q$).

Доказательство закончено. □

Доказательство Теоремы 4.2 мы начали со случая $(x, m) = 1$ и применили теорему Эйлера, но в оставшейся части доказательства вместо неё использовали только малую теорему Ферма. На практике случайно выбранное $x \bmod m$ будет скорее всего взаимно просто с m, когда m является произведением двух больших простых чисел, но всё же полезно знать, что Теорема 4.2 верна для всех x без исключений, даже когда $(x, m) > 1$, причём теорема Эйлера для её доказательства не нужна вовсе.

Довольно распространённое заблуждение заключается в том, что именно теорема Эйлера объясняет взаимную обратность функций шифрования и дешифрования в алгоритме RSA. *В основании алгоритма RSA лежит малая теорема Ферма, а не теорема Эйлера.* Работа [8, §VI], в которой Ривест, Шамир и Адлеман изложили математическое обоснование алгоритма RSA, начинается с напоминания формулировки теоремы Эйлера $x^{\varphi(m)} \equiv 1 \bmod m$, когда $(x, m) = 1$, но *они ни разу её не использовали.* Вместо неё в доказательстве использовалась только малая теорема Ферма, ровно как изложено выше.

Замечание 4.3. Теорема 4.2 верна не только для m, равного произведению двух различных простых, но и для произвольного количества различных простых множителей: $x^{ed} \equiv x \bmod m$ для всех x, когда $ed \equiv 1 \bmod \varphi(m)$. Если станет известен эффективный способ разложения на множители чисел вида $m = pq$, то мы можем использовать в качестве модуля произведение трёх и большего количества простых сомножителей (если, конечно, техника разложения на множители чисел вида pq не будет работать на таких числах тоже).

Пример 4.4. Пусть ваш публичный ключ $m = 2823907$ и $e = 3$. Чтобы отправить вам сообщение $x = 71520$, зашифруем его функцией $E(x) = x^3 \equiv 83246 \bmod m$ и отправим вместо него число 83246.

Предположим, шпион перехватил число 83246. Как ему дешифровать это сообщение? Шпиону надо решить уравнение $x^3 \equiv 83246 \bmod m$. Чтобы сделать это без полного перебора всех чисел от 2 до $m-1$ (на практике число m состоит из тысяч цифр, а не 7, как в этом примере[3]), шпиону надо найти обратное числу 3 по модулю $\varphi(m)$: если $3d \equiv 1 \bmod \varphi(m)$, то

$$D(y) = y^d \bmod m,$$

так что $x \equiv D(E(x)) = 83246^d \bmod m$. Для вычисления d ему надо знать модуль $\varphi(m)$, по которому найти обратное числу 3. Всё, что знает шпион — это число $m = 2823907$. И в этом всё дело: в то время как вам не составит труда найти d, потому что это вы выбрали простые числа p и q (и которых никто другой не знает) и это позволило вам вычислить $\varphi(m) = (p-1)(q-1)$, пока не известно как в общем случае найти $\varphi(m)$, зная только само m, но не разложение его на простые множители.

Покажем, как вычислить дешифрующую функцию, зная разложение m на множители: $m = 1223 \cdot 2309$. Были выбраны $p = 1223$ и $q = 2309$. Тогда $\varphi(m) = (p-1)(q-1) = = 2820376$. Сравнение $3d \equiv 1 \bmod \varphi(m)$ имеет решение $d = 1880251$, так что дешифрующая функция это $D(y) \equiv y^{1880251} \bmod m$. Проверим: $82346^{1880251} \equiv 71520 \bmod m$, что соответствует начальному выбору передаваемого сообщения $x = 71520$.

Предположим вы получили новое сообщение 230748, т.е. $E(x) = 230748$ для какого-то неизвестного x. Можно найти x, вычислив $D(230748) = 230748^{1880251} \equiv 270513 \bmod m$. Поэтому $x = 270513$. Пока шпиону неизвестно разложение m на множители, он не может вычислить $\varphi(m)$, а значит и d, и таким образом не сможет дешифровать перехваченное сообщение.

Краткое изложение алгоритма RSA: Выберем два различных простых p и q. Положим $m = pq$ и выберем $e \in \mathbf{Z}^+$ такое, что $(e, \varphi(m)) = 1$. Пара чисел (m, e) это ваш *публичный ключ*, который можно сообщить любому, кто хочет отправить вам сообщение $x \bmod m$ в зашифрованном виде $E(x) = x^e \bmod m$. Тройка чисел (p, q, d), где $d \in \mathbf{Z}^+$ определяется из соотношения $ed \equiv 1 \bmod \varphi(m)$ — ваш *приватный ключ*. Зная его, можно дешифровать сообщения с помощью функции $D(y) \equiv y^d \bmod m$. Такая схема работает, поскольку $x^{ed} \equiv x \bmod m$ для всех x по Теореме 4.2, поэтому $D(E(x)) \equiv x \bmod m$ для всех $x \bmod m$.

Любой, кто знает одно из чисел p или q знает оба числа, поскольку $m = pq$, а число m известно всем, т.к. содержится в публичном ключе. По числам p, q и e можно вычислить d и тем самым дешифровать отправленное вам зашифрованное сообщение.[4]

[3] В 1874, Уильям Стэнли Джэвенс (William Stanley Jevons) написал [6, p. 141] “произведение каких двух чисел даст нам 8,616,460,799? Вряд ли кто-нибудь кроме меня это узнает; потому что это два больших простых числа...”. Сейчас при помощи компьютера можно мгновенно разложить на множители и получить ответ: $89681 \cdot 96079$. Джэвенс продолжал “Для хорошего компьютера эта работа займёт много недель,” но он имел в виду не машину. В 1800-е словом “компьютер” называли *человека*, выполнявшего вычисления.

[4] Возведение в степень e и d коммутативно (их можно выполнять в любом порядке с одинаковым результатом), т.е. $E(D(x)) \equiv x^{ed} \equiv x^{de} \equiv D(E(x)) \bmod m$ для всех x. Это наблюдение важно при изготовлении цифровой подписи: при помощи степени d из дешифрующей функции *зашифруйте* подтверждение получения сообщения. Это подтверждение можно дешифровать с помощью степени e из публичного ключа. Без знания числа d из приватного ключа вряд ли кто-то сможет зашифровать осмысленное сообщение.

Создание публичного ключа начинается с выбора двух достаточно больших простых чисел. Как проверить большое число p на простоту? Можно 20 раз выполнить тест Ферма. Если ни в одном из тестов не будет обнаружен *свидетель составности* (т.е. такое a от 1 до $p-1$, что $a^{p-1} \not\equiv 1 \bmod p$), то можно с достаточной уверенностью считать p простым и использовать его. Правда, p может оказаться числом Кармайкла.[5] Существуют и более качественные вероятностные тесты простоты, например тест Миллера-Рабина, который не имеет подобных дефектов.

На втором шаге создания публичного ключа, после получения модуля $m = pq$, надо выбрать e — показатель шифрующей функции, по которому затем вычислить показатель дешифрующей функции d из соотношения $ed \equiv 1 \bmod \varphi(m)$. Здесь требуется аккуратность: если d (которое хранится в тайне, как часть приватного ключа) меньше, чем примерно $\sqrt[4]{m}$, то для его отыскания можно использовать разложение дроби e/m в непрерывную дробь. Этот способ называется атакой Винера (Wiener's continued fraction attack) [9].

5. Примеры шифрования алгоритмом RSA

Чтобы приблизить наше изложение алгоритма к реальности, применим RSA к текстовому сообщению. Непросто впечатлиться шифрованием 16-разрядного числа вроде 4932123409876578, хотя примерно это и происходит, когда при покупке в интернет-магазине пересылается номер банковской карточки. Закодируем буквы строками из двух цифр: A = 01, B = 02, и так далее до Z = 26, пробел = 27.

Пример 5.1. Зашифруем строчку MOM = 131513. Возьмём $m = 2823907$ и $e = 3$ из Примера 4.4, $E(\text{MOM}) = 131513^3 \equiv 1842379 \bmod m$. Можно передавать последовательность в таком виде, а можно найти способ перекодировать эту последовательность цифр обратно в текст.

Если сообщение (это число) больше модуля — *разбейте его на части*. Например, для модуля $m = 2823907$, длина которого 7 цифр, блок из букв не может быть длиннее 3 (а зашифрованное сообщение не больше 6 цифр). Модуль $m = 12964553$ состоит из 8 цифр, но первая пара цифр 12, что меньше 27, поэтому снова мы должны использовать блок не более чем из 3 букв. Дело в том, что поскольку буква N кодируется числом 14 ($14 > 12$) и блок из 4 букв, начинающийся с N, будет закодирован бóльшим числом, чем выбранный модуль.

Пример 5.2. Если $m = 2823907$ и $e = 3$, то $E(x) = x^3 \bmod m$ и как было показано в Примере 4.4 $D(y) = y^{1880251} \bmod m$.

Сообщение STOP NOW кодируется так: 1920151627141523 (напомним, 27 это пробел). Разобъём его на блоки длиной не больше 3 букв: $(192015, 162714, 1523)$. Применим $E(x) = x^3 \bmod m$ к каждому блоку: первый блок шифруется так $192015^3 \equiv$ $\equiv 859833 \bmod m$, оставшиеся два аналогичным образом.

Вот зашифрованное сообщение $(859833, 1395490, 2758917)$. В обратную сторону, если было получено зашифрованное сообщение $(1294545, 1214153)$, то исходное сообщение получается применением дешифрующей функции к каждому блоку:

$$D(1294545) = 1294545^{1880251} \bmod m \equiv 11209 \bmod m$$

и

$$D(1214153) = 1214153^{1880251} \bmod m \equiv 2205 \bmod m.$$

[5] См. https://kconrad.math.uconn.edu/blurbs/ugradnumthy/fermattest.pdf.

Полученные дешифрованные числа должны быть чётной длины (каждая буква — две цифры), но первое состоит из 5 цифр, что означает пропущенный ноль в начале. Получаем такое исходное сообщение (011209,2205) = (ALI,VE) = ALIVE.

Замечание 5.3. Ещё раз: следите за длиной блоков и величиной модуля. Например, в Примере 5.2 не разбивайте сообщение на 4-буквенные блоки, потому что модуль записывается 7 цифрами. Попробуем тем не менее, используя модуль из Примера 5.2 зашифровать STOP = 19201516 как единый блок цифр, получим $19201516^5 \equiv 2086151 \bmod m$, но после дешифрования $y = 2086151$ получим $D(y) = y^{1880251} \equiv 2722781 \bmod m$. Поскольку 2722781 = 02722781, не получится преобразовать обратно в строку, поскольку "72" и "81" не принадлежат множеству $\{01, 02, \ldots, 27\}$. Таким образом, для выбранного в Примере 5.2 модуля мы можем разбивать сообщения на блоки длины не больше 3.

Приведём ещё некоторые соображения, касающиеся алгоритма RSA.

1. Если $(e, \varphi(m)) > 1$, алгоритм не стоит использовать.

   Например, если $m = 217 = 7 \cdot 31$ и $e = 3$, то получаем $\varphi(m) = 6 \cdot 30$ и $(e, \varphi(m)) > 1$. Подставив в функцию $E(x) = x^3 \bmod m$, получим $8^3 \equiv 78 \bmod 217$ и $9^3 \equiv 78 \bmod 217$, таким образом $E(8) \equiv E(9) \bmod m$. Это плохо — разные сообщения могут совпасть после шифрования.
2. В замечании 4.3 мы упомянули, что RSA может работать, если модуль m является произведением больше, чем двух простых чисел. Но важно, чтобы они были попарно различны.

   Если m имеет в разложении на простые кратный множитель, алгоритм не стоит использовать даже когда $(e, \varphi(m)) = 1$. Пусть, например $m = 275 = 5^2 \cdot 11$ и $e = 3$. Тогда $\varphi(m) = 200$, значит $(e, \varphi(m)) = 1$ и решением отношения $3d \equiv 1 \bmod \varphi(m)$ является $d = 67$. Взяв функции $E(x) = x^3 \bmod m$ и $D(y) = y^{67} \bmod m$, для $x = 15$ получим $D(E(x)) = (x^3)^{67} = x^{201} = 15^{201} \equiv 125 \bmod m \not\equiv 15 \bmod m$. Получается, что последовательное применение шифрующей и дешифрующей функций не приводит к исходному сообщению.

Зная разложение на множители $m = pq$, мы можем вычислить $\varphi(m)$ по формуле $(p-1)(q-1)$ и решить соотношение $ed \equiv 1 \bmod \varphi(m)$ относительно d, получая тем самым дешифрующую функцию $D(y) = y^d \bmod m$. В конце Разделов 2 и 3 было сказано, что надёжность шифра RSA обусловлена сложностью вычисления $\varphi(m)$ по числу m не зная его разложения на простые множители. *Но насколько точно это утверждение?* Проницательный читатель может поинтересоваться — не существует ли способа по публичному ключу (m, e) вычислить $\varphi(m)$, не раскладывая m на множители? Если так, то мы можем решить соотношение $ed \equiv 1 \bmod \varphi(m)$ относительно d и получить дешифрующую функцию $D(y) = y^d \bmod m$, не раскладывая m на простые множители.

Следующая теорема показывает, как по $\varphi(m)$ вычислить p и q. Таким образом, знание $\varphi(m)$ и знание факторизации числа m эквивалентны.

Теорема 5.4. *Пусть $m = pq$, где p и q различные простые числа. Зная m и $\varphi(m)$ мы можем вычислить p и q.*

Доказательство. Раскроем скобки: $\varphi(m) = (p-1)(q-1) = pq-(p+q)+1 = m-(p+q)+1$, значит $p+q = m+1-\varphi(m)$. Следовательно, если нам известны m и $\varphi(m)$, то мы знаем как произведение pq, так и сумму $p+q$, которые с точностью до знака совпадают с коэффициентами квадратного многочлена с корнями p and q:

$$(X-p)(X-q) = X^2 - (p+q)X + pq = X^2 - (m+1-\varphi(m))X + m.$$

Если m и $\varphi(m)$ известны, то известны и коэффициенты этого многочлена, корни которого можно вычислить по известной формуле. Таким образом, мы можем вычислить p и q, зная m и $\varphi(m)$ в случае, когда m равно произведению двух простых сомножителей. □

Пример 5.5. Пусть $m = 3297523$ и нам известно значение $\varphi(m) = 3292840$. Если $m = pq$ для различных простых p and q, то из доказанной Теоремы 5.4 значения p и q это корни квадратного многочлена

$$X^2 - (m + 1 - \varphi(m))X + m = X^2 - 4684X + 3297523.$$

Корни этого многочлена 863 и 3821.

Чтобы взломать RSA, нам в конечном счёте нужно знать не $\varphi(m)$, а показатель d. Быть может есть способ найти d по публичному ключу (m, e), обойдясь без факторизации модуля m или, что то же самое, знания $\varphi(m)$?

Поскольку $ed \equiv 1 \bmod \varphi(m)$ и $ed \neq 1$ (никто не будет использовать алгоритм с $e = d = 1$), то значение $ed - 1$ является *положительным кратным* числа $\varphi(m)$.

Следующая теорема даёт вероятностный алгоритм разложения m на простые делители по известным m и положительному кратному числа $\varphi(m)$.

Теорема 5.6. *Пусть m равно произведению двух различных нечётных простых чисел и пусть положительное целое N — кратное числа $\varphi(m)$. Запишем N в виде $N = 2^r k$ где $r \geq 1$ и k нечётное.*[6] *Более половины всех $a \in \{1, 2, \ldots, m-1\}$ удовлетворяют одному из следующих условий:*

(1) $1 < (a, m) < m$,

(2) $(a, m) = 1$, $a^{2^k} \not\equiv 1 \bmod m$, *а среди $i \in \{1, \ldots, r\}$ найдётся такое, что*
- $a^{2^i k} \equiv 1 \bmod m$
- $1 < (a^{2^{i-1}k} - 1, m) < m$

Доказательство Теоремы 5.6 мы опустим[7], приведём лишь пример её использования.

Пример 5.7. Мы знаем модуль $m = 9{,}330{,}443$ и каким-то образом нам стало известно, что $N = 11{,}347{,}658{,}496$ является кратным $\varphi(m)$. Выделив из числа N наибольшую степень 2, получим $N = 2^8 \cdot 44{,}326{,}791 = 2^r k$. Выберем случайное положительное целое число, не превосходящее $m - 1$, например $a = 5{,}662{,}037$.

Используя алгоритм Евклида убедимся, что $(a, m) = 1$. Вычисляя затем $a^{2^i k} \bmod m$ для всех $i = 0, \ldots, 8$, мы обнаружим, что при $i = 6$ выполняются следующие соотношения $a^{2^5 k} \equiv 4{,}259{,}023 \bmod m$ и $a^{2^6 k} \equiv 1 \bmod m$. Снова используя алгоритм Евклида, найдём $(a^{2^5 k} - 1, m) = (4{,}259{,}022, m) = 2699$. Таким образом нетривиальный делитель числа m это 2699, второй делитель равен $m/2699 = 3457$.

Теорема 5.6 даёт нам *вероятностный алгоритм* нахождения делителей m по известному m и какому-то кратному $\varphi(m)$, потому что случайный выбор числа $a \in \{1, \ldots, m-1\}$ более, чем в половине случаев удовлетворяет одному из двух условий

[6] Неравенство $r \geq 1$ выполняется, поскольку раз $\varphi(m)$ чётное, то и его кратное N тоже чётное.

[7] Теорема 5.6 на самом деле верна для всех нечётных $m > 1$ не являющихся степенью простого числа, не обязательно только для чисел вида pq. Доказательство теоремы опирается на те же идеи, которые используются в обосновании работы теста Миллера-Рабина. См. Следствие A.2 в https://kconrad.math.uconn.edu/blurbs/ugradnumthy/millerrabin.pdf, заменив там $\varphi(n) = 2^e k$ на $N = 2^r k$.

теоремы, и оба этих условия позволяют нам вычислить нетривиальный делитель m: (a, m) или $(a^{2^{i-1}k} - 1, m)$. Когда $m = pq$, его нетривиальный делитель это или p или q, так что зная один из них, мы знаем оба. А согласно Теореме 5.6 достаточно попробовать не так много значений a, прежде, чем мы найдём нетривиальный делитель m.

Есть модули, к которым Теорема 5.6 неприменима, это числа вида $m = 2q$ для нечётных простых q. Но это замечание не представляет никакого практического интереса, поскольку в алгоритме RSA нет смысла выбирать чётный модуль.

Если нам известен публичный ключ (m, e), то в силу Теоремы 5.6 вычисление показателя d и разложение на простые множители модуля m это эквивалентные с вычислительной точки зрения задачи:

- d можно найти, зная p и q и используя расширенный алгоритм Евклида для решения соотношения $ed \equiv 1 \bmod (p-1)(q-1)$ относительно d;
- разложить m на множители p и q можно, зная d и положительное кратное числа $\varphi(m)$ и используя вероятностный алгоритм из Теоремы 5.6, положив $N = ed - 1$.

Криптографические протоколы вроде шифра Цезаря, шифра Хилла или немецкой шифровальной машины Энигма называются *симметричными*, поскольку оба метода — шифрования и дешифрования — можно определить, зная другой, причём довольно быстро. А в Энигме процессы шифрования и дешифрования вообще одинаковы. Для использования симметричного шифра надо хранить в тайне от противника алгоритм шифрования. Напротив, алгоритм RSA называется *асимметричным* шифром, поскольку знание способа шифрования не позволяет — по крайней мере, насколько нам известно — узнать алгоритм дешифровки. Раздел криптографии, в котором параметры алгоритма шифрования могут быть опубликованы без ущерба для надёжности дешифровки, получил название “криптография с открытым ключом”.

Кажется, что асимметричный шифр по всем параметрам превосходит симметричный, но на практике используются оба, дополняя друг друга. Современные симметричные шифры гораздо быстрее асиметричных (особенно на этапе дешифрования), поэтому RSA не очень подходит для обмена сообщениями большого размера. Асимметричный алгоритм шифрования нужен адресатам для безопасного обмена ключами (относительно небольшими), которые затем используются надёжным симметричным шифром вроде AES (Advanced Encryption Standard). Таким образом при помощи асимметричного шифра функция шифрования симметричного шифра сохраняется в секрете, делая подобный одноразовый сеанс передачи информации безопасным.

6. История RSA

Алгоритм RSA был придуман в 1977 и впервые опубликован том же году в колонке Мартина Гарднера в журнале *Scientific American* [3]. В то время было неизвестно, что эта идея уже разрабатывалась британской разведкой несколькими годами ранее. Клиффорд Кокс (Clifford Cocks), изучавший теорию чисел в Кембридже, работал на Центр Правительственной Связи (GCHQ, Government Communications Headquarters, британский аналог американского NSA, National Security Agency) и в засекреченной работе описал в точности алгоритм RSA, используя $e = m$. Таким образом, Кокс выбрал в качестве показателя шифрующей функции e значение модуля m и следовательно, ему пришлось потребовать, чтобы $(m, \varphi(m)) = 1$. Иначе говоря, чтобы для простых p и q выполнялось $(pq, (p-1)(q-1)) = 1$, или что то же самое $(p, q-1) = 1$ и $(q, p-1) = 1$

поскольку очевидно, что p взаимно просто с $p-1$ и q взаимно просто с $q-1$. Этот алгоритм не такой гибкий, как RSA, но в его основе лежит та же математика.

Из-за нехватки достаточно производительных компьютеров в GCHQ не смогли довести работу до практической реализации до того, как алгоритм был заново открыт Ривестом, Шамиром и Адлеманом. Оригинал работы был рассекречен GCHQ в 1997 году, спустя 20 лет после (пере)открытия RSA.

Вместо термина "криптографии с открытым ключом" в GCHQ использовали "non-secret encryption" как его назвал Кокс в заголовке своей работы [1]. Изложение истории разработки этого протокола в GCHQ была написана Джеймсом Эллисом (James Ellis) [2].

Список литературы

[1] C. Cocks, A Note on Non-Secret Encryption, CESG report, November 20, 1973. URL http://cryptocellar.org/cesg/notense.pdf.

[2] J. H. Ellis, The Story of Non-Secret Encryption, CESG Report, 1987. URL https://cryptocellar.org/cesg/possnse.pdf.

[3] M. Gardner, A new kind of cipher that would take millions of years to break, *Scientific American* **237** (Aug. 1977), 120–124.

[4] J. Grime, Enigma Machine - Numberphile, https://www.youtube.com/watch?v=G2_Q9FoD-oQ.

[5] J. Grime, Flaw in the Enigma Code - Numberphile, https://www.youtube.com/watch?v=V4V2bpZlqx8.

[6] W. S. Jevons, "The Principles of Science: a Treatise on Logic and Scientific Method," MacMillan & Co., New York, 1874. URL https://archive.org/stream/principlesofscie00jevorich#page/n165/mode/2up.

[7] J. Owen, How did the Enigma Machine work? https://www.youtube.com/watch?v=ybkkiGtJmkM.

[8] R. L. Rivest, A. Shamir, L. Adleman, A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, *Comm. ACM* **21**, Feb. 1978. URL https://people.csail.mit.edu/rivest/pubs/RSA78.pdf.

[9] M. Wiener, Cryptanalysis of short RSA secret exponents, *IEEE Transactions on Info. Theory* **36** (1990), 553-558.