Правительство Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Санкт-Петербургский государственный университет»

Кафедра информационно-аналитических систем

Терехов Антон Юрьевич

Ранцевая криптосистема с открытым ключом

Бакалаврская работа

Научный руководитель:
д. т. н., профессор Крук Е. А.

Рецензент:
тьютор Ханов А. Р.

Санкт-Петербург
2016

SAINT-PETERSBURG STATE UNIVERSITY

Sub-Department of Analytical Information Systems

Anton Terekhov

Knapsack public key cryptosystem

Graduation Thesis

Scientific supervisor:
professor Eugine Krouk

Reviewer:
tutor Arthur Khanov

Saint-Petersburg
2016

Оглавление

Введение

В данной работе будут рассмотрены альтернативы для используемых в наше время криптосистем с открытым ключом, а именно криптосистемы, основанные на задаче о рюкзаке. Один из вариантов такой системы предложили Ральф Меркл и Мартин Хеллман [5]. В данных системах открытым ключом является последовательность объектов (весов), в самом простом случае это последовательность натуральных чисел. Отправитель вычисляет сумму только тех объектов, которые соответствуют единицам в сообщении, представленном в бинарном виде, далее эта сумма пересылается адресату. В общем случае задача восстановления сообщения по вышеуказанной сумме NP-полная задача. Однако секретным ключом являются параметры прямого и обратного преобразований весов, преобразование строится таким образом, чтобы в обычном виде задача по расшифровке была сложна, а в преобразованном виде решалась быстро.

1. Проблемы ранцевых криптосистем

Указанный выше вид криптосистем был изобретён уже давно. Однако для системы Меркла-Хеллмана был найден полиномиальный алгоритм нахождения секретного ключа (параметров преобразования) Ади Шамиром [7]. Данная атака системы использует метод решения целочисленного линейного программирования, рассмотренного Хендриком Ленстра [3]. Стоит заметить, что и вышеуказанная атака, и алгоритм целочисленного линейного программирования были рассмотрены лишь с теоретической точки зрения, полиномиальная сложность этих алгоритмов доказана, однако не были рассмотрены реализации этих алгоритмов. Также существует обобщение системы, при котором объектами являются вектора чисел, варианты таких криптосистем рассмотрели Роберт Мак-Элис [4] и Харальд Нидеррайтер [6]. Эти способы появились из теории кодов, исправляющих ошибки. Слабым местом этих систем являются используемые в них методы кодирования и декодирования, определённые свойства которых позволяют провести атаку второго рода, то есть по открытому ключу получить закрытый. Для препятствования этим проблемам, в данных системах используют довольно большой по памяти открытый ключ, что мешает их активному использованию.

2. Криптосистема Меркла-Хеллмана

Идея криптосистемы заключается в следующем. Открытым ключом является последовательность из n натуральных чисел $a_1, a_2, a_3, \dots a_n$. Сторона A для передачи сообщения в битовом виде $x_1, x_2, x_3, \dots x_n, x_i \in \{0, 1\}$ вычисляет сумму соответствующих произведений $sum = \sum_{i=1}^{n} a_i x_i$ (можно считать, выполняется скалярное произведение векторов) и посылает по открытому каналу результат. Для расшифровки сообщения сторона B должна решить обратную задачу, известную, как задача о ранце: по сумме весов некоторых объектов, а также по весам всех объектов определить объекты, участвующие в сумме. В общем виде данная задача является NP-полной, а при некоторых наборах весов ответов может быть даже несколько. (Например, в случае набора весов $\{2, 3, 5\}$). В случае, когда для каждого $1 < i \leqslant n$ выполняется строгое неравенство $\sum_{j=1}^{i-1} a'_j < a'_i$ (последовательность $a'_1, a'_2, a'_3, \dots a'_n$ в таком случае называется супервозрастающей), то существует следующее решение, работающее за полиномиальное время. Изначально инициализируется переменная sum', равная полученной по каналу связи сумме. В ходе алгоритма из неё будут вычитаться веса, соответствующие единицам в дешифрованном сообщении. Рассматриваются веса a'_i в порядке убывания, в том случае, когда текущая сумма sum' меньше, чем a'_i, то, очевидно, i-й объект не входит в ответ. Если же сумма sum' больше или равна текущего веса a'_i, то он обязан входить в ответ, иначе $\sum_{j=1}^{i-1} a'_j < a'_i \leqslant sum'$, и даже все оставшиеся объекты не обеспечат нужную сумму. Для перехода к следующему весу нужно добавить текущий объект в ответ и вычесть из суммы текущий вес.

Однако для несанкционированной стороны C решение данной задачи должно быть сформулировано в общем виде. Для этого случайным образом берутся два числа: $M > \sum_{i=1}^{n} a'_i$ и $W : (W, M) = 1$, а также число W^{-1}, являющееся обратным к W по модулю M. При замене $a_i = a'_i W^{-1} \bmod M$ веса a_i оказываются равномерно распределёнными по модулю M, а операции на стороне A практически не усложняют-

ся (просто изменяются используемые веса, причём не изменяется порядок максимального веса, которым оценивается сложность алгоритма дешифрования). На стороне B полученная сумма sum' умножается на W по модулю M, в итоге получается $sum = sumW \bmod M = \sum_{i=1}^{n} a'_i W^{-1} W \bmod M = \sum_{i=1}^{n} a'_i$. Таким образом с помощью секретного ключа задача свелась к полиномиальной. При реализации, кроме домножения на W^{-1} используется также перестановка весов, при расшифровке применяется обратная.

Остаётся указать способ построения чисел W и M, а также способ построения супервозрастающей последовательности. Пусть необходимо построить ключи для некоторого количества объектов n. Вес a'_1 выбирается случайным образом из равномерного распределения целых чисел по отрезку $[1, 2^n]$. Следующий вес a'_2 выбирается равномерно из отрезка $[1 * 2^n + 1, 2 * 2^n]$, следующий за ним a'_3 выбирается из $[3 * 2^n + 1, 4 * 2^n]$ и так далее; a'_i выбирается из отрезка $[(2^{i-1} - 1) * 2^n + 1, 2^{i-1} * 2^n]$. При таких распределениях условие супервозрастаемости, очевидно, будет выполнятся, так как нижняя граница каждого отрезка в точности равна увеличенной на единицу сумме всех правых границ предыдущих отрезков. Число M при этом выбирается из отрезка $[2^{2n+1} + 1, 2^{2n+2} - 1]$, а число W выбирается случайно из отрезка $[2, M - 2]$ до тех пор, пока не будет взаимно просто с M.

3. Атака Шамира на криптосистему Меркла-Хеллмана

Атака основана на нахождении чисел W и M по открытому ключу, переводящих веса $a_1, a_2, a_3, \dots a_n$ в супервозрастающую последовательность (с точностью до перестановки), причём не обязательно в исходную. Реализуется это с учётом того, что свойство супервозрастаемости последовательности $\{a'_i\}_{i=1}^n$ оставляет свои следы в последовательности $\{a_i\}_{i=1}^n$. Очевидно, что каждая из сумм $sum'_i = \sum_{j=1}^{i} a_j$ не менее, чем в два раза больше предыдущей $2sum'_i < sum'_{i+1}$ из-за свойства супервозрастаемости. Отсюда следует, что $a'_1 = sum'_1 < \frac{sum'_n}{2^{n-1}} < \frac{M}{2^{n-1}}$. Аналогично получаются оценки для остальных элементов последовательности $a'_i < sum'_i < \frac{M}{2^{n-i}}$. Так как $a_i = a'_i W^{-1} \bmod M$ и $a'_i < M$, то $a'_i = a_i W \bmod M$, и, подставляя в наши неравенства и разделив их на M, получаем $a_i \frac{W}{M} \bmod 1 < \frac{1}{2^{n-i}}$ (здесь под операцией модуля рассматривается нецелочисленное взятие по модулю). При рассмотрении функции вещественного аргумента $f_i(x) = a_i x \bmod 1$ при $x \in (0, 1)$, мы увидим пилообразный график, состоящий из a_i склонов, расстояние между которыми будет $\frac{1}{a_i}$ (рис. 1). Легко заметить, что производная

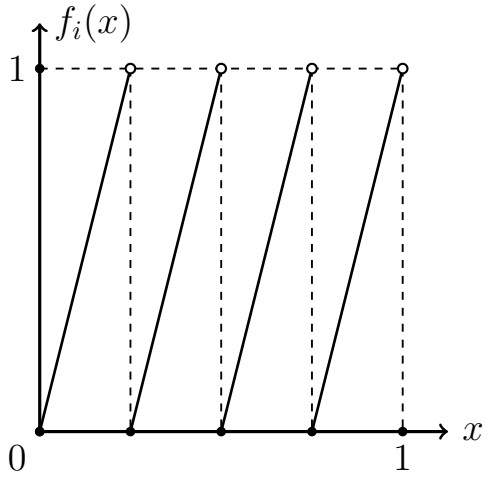

Рис. 1: График функции $f_i(x)$

функции $f_i(x)$ равна a_i во всех точках, кроме точек разрыва. Поэтому

для функции $f_1(x)$, соответствующей минимальному числу a'_1 в супервозрастающей последовательности, неизвестное при атаке значение $\frac{W}{M}$ будет близко к одному из минимумов этой функции (например, к i-му минимуму с координатой $\frac{i}{a_1}$): $\frac{W}{M} - \frac{i}{a_1} = \frac{f(\frac{W}{M})}{a_1} = \frac{a_i \frac{W}{M} \bmod 1}{a_1} < \frac{1}{2^{n-1}a_1}$. Аналогично неравенства будут выполняться и для a_2, a_3 и a_4 (номера минимумов для соответствующих функций назовём j, k и l). Стоит заметить, что из-за перестановки, описанной в предыдущем разделе, при атаке неизвестно, каким элементам открытого ключа соответствуют a'_1, a'_2, a'_3 и a'_4. Эта проблема решается перебором всех четвёрок весов объектов из открытого ключа. Таким образом далее будем считать, что a_1, a_2, a_3 и a_4 соответствуют первым четырём элементам супервозрастающей последовательности, причём пусть $\frac{i}{a_1}$ будет больше, чем $\frac{j}{a_2}$, $\frac{k}{a_3}$ и $\frac{l}{a_4}$, а a_2, a_3 и a_4 будут отсортированы по возрастанию соответствующих им весов в закрытом ключе (рис. 2). Тем самым, минимум функции f_1 будет ближайшим к искомому значению $\frac{W}{M}$, а минимум $\frac{j}{a_2}$ соответствует либо первому, либо второму элементу супервозрастающей последовательности, следовательно расстояние между ним и $\frac{W}{M}$ будет меньше $\frac{1}{2^{n-2}a_2}$, а так как минимум $\frac{i}{a_1}$ ближе всех к $\frac{W}{M}$, то и расстояние между этими двумя минимумами будет меньше $\frac{1}{2^{n-2}a_2}$. В таком случае можно написать неравенства для i, j, k и l:

$$
\begin{gathered}
0 \leqslant \frac{i}{a_1} - \frac{j}{a_2} \leqslant \frac{1}{2^{n-2}a_2} \\
0 \leqslant \frac{i}{a_1} - \frac{k}{a_3} \leqslant \frac{1}{2^{n-3}a_3} \\
0 \leqslant \frac{i}{a_1} - \frac{j}{a_4} \leqslant \frac{1}{2^{n-4}a_4} \\
1 \leqslant i \leqslant a_1 - 1 \\
1 \leqslant j \leqslant a_2 - 1 \\
1 \leqslant k \leqslant a_3 - 1 \\
1 \leqslant l \leqslant a_4 - 1
\end{gathered}
$$

Решая данную систему при всех размещениях весов $a_1, a_2, a_3, \ldots a_n$,

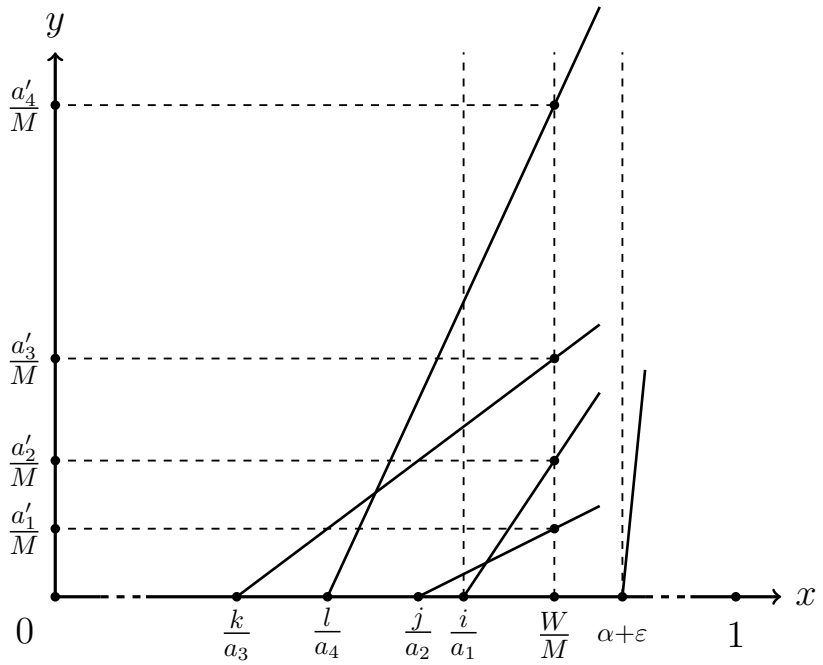

Рис. 2: Графики функций $f_i(x)$ вблизи точки $\frac{W}{M}$

будут получены точки, рядом с которыми может находится точка $\frac{W}{M}$, причём одна из них точно будет рядом с $\frac{W}{M}$, а другие могут находится рядом с другими, но подходящими для атаки значениями $\frac{W'}{M'}$.

Пусть после решения линейной системы неравенств в целых числах $\alpha = \frac{i}{a_1}$, тогда искомое число $v = \frac{W}{M}$ должно находиться в интервале $(\alpha, \alpha+\varepsilon)$, где $\alpha+\varepsilon$ — самая близкая точка разрыва к α среди всех точек разрыва всех функций f_i, больших α. На этом интервале все функции принимают вид отрезков, их количество равно n, поэтому существует не более $\frac{n(n-1)}{2}$ точек их пересечения. Если отсортировать абсциссы всех точек пересечения, то они разбивают интервал $(\alpha, \alpha+\varepsilon)$ на интервалы (l_k, r_k), количество которых не превышает n^2, а значения $a'_i = f_j(v)$ на их протяжении идут в одном и том же порядке (так как отрезки в этих точках не пересекаются). На каждом из отрезков можно записать порядка n неравенств для обеспечения свойства супервозрастаемости,

а именно:

$$f_{\pi(i)}(v) > \sum_{j=1}^{i-1} f_{\pi(j)}(v), i \in 2..n$$

$$\sum_{i=1}^{n} f_i(v) < 1,$$

где $\pi(j)$ — перестановка, соответствующая порядку значений $f_j(v)$ на текущем интервале (l_k, r_k), а сами функции $f_j(v)$ представляются в виде $f_j(v) = a_j v - c_j$, где c_j — номер наибольшего минимума функции $f_j(v)$, меньшего α.

Заметим, что данные неравенства будут линейно зависеть от v. Дополнительное ускорение процесса можно получить, заметив, что последнее неравенство одинаково для всех интервалов (l_k, r_k), а это значит, что $\alpha + \varepsilon$ можно уменьшить до решения этого неравенства, тем самым уменьшив количество интервалов. После решения этой простой системы на данном отрезке, получается отрезок для значений $\frac{W}{M}$, причём если отношение любых двух целых чисел W и M попадает в этот отрезок, то веса $\{a_i\}_{i=1}^{n}$ после преобразования $a'_i = a_i W \bmod M$ будут образовывать супервозрастающую последовательность. Для нахождения таких чисел можно разложить границы отрезка в цепные дроби, взять цепную дробь наименьшей длины, которая будет между двумя получившимися, и преобразовать её в обычную дробь, числителем и знаменателем которой будут соответственно W и M. При этом часто получаются значения W и M по длине меньше, чем исходные.

4. Целочисленное решение системы линейных неравенств

При выполнении атаки наибольшую часть процессорного времени занимает решение системы линейных неравенств. В этом разделе будет разобран алгоритм решения, предложенный Ленстра, а так же его оптимизации при применении к поставленной задаче.

4.1. Обзор метода

Задача ставится следующим образом. Пусть n — количество неизвестных. Система линейных неравенств задаётся целочисленной матрицей A размера $m \times n$ и целочисленным вертикальным вектором b длины m. Решениями данной системы являются целочисленные вектора x, удовлетворяющие неравенству $Ax \leqslant b$. Стоит заметить, что коэффициенты матриц A и b могут быть и рациональными, поведение алгоритма от этого не изменится.

По сути, решением системы является пересечение многогранника, задаваемого матрицами A и b, и целочисленной сетки в n-мерном пространстве. Строки матрицы A при этом задают направление нормальных векторов к граням, то есть задают поворот грани, а элементы вектора b задают смещение граней вдоль этих направлений от центра координат. В процессе решения находится невырожденное линейное преобразование пространства τ, при котором многогранник приобретает так называемую сферическую форму (имеется в виду тот факт, что существует два шара с общим центром, один из которых содержит многогранник, а другой содержится внутри многогранника, причём отношение радиусов этих шаров зависит только от размерности n). В том случае, если многогранник имеет нулевой объём, такого τ не существует, вместо этого производится сведение задачи к меньшей задаче, по размерности равной размерности многогранника. При преобразовании τ целочисленная сетка и её базис деформируются, углы между базисными векторами становятся отличными от прямых. Далее этот базис

преобразовывается к почти ортогональному виду с помощью операций замены векторов местами и целочисленного прибавления одного вектора к другому, что соответствует умножению на унимодулярную матрицу. "Почти ортогональность" здесь означает то, что отношение детерминанта матрицы, состоящей из базисных векторов, к произведению их длин больше некоторой константы, зависящей только от размерности n; по сути это отношение объёма параллелепипеда, натянутого на базисные вектора, к объёму такого параллелепипеда при условии, что все вектора ортогональны между собой.

Таким образом, исходная задача про пересечение многогранника и целочисленной сетки переходит в эквивалентную ей задачу пересечения сферического многогранника и сетки с почти ортогональным базисом. В этом случае если самая близкая к центру шаров точка сетки попадает в многогранник, то задача решена; пересечение сетки и многогранника найдено. В противном случае сетка не пересекается с внутренним шаром, поэтому она достаточно крупна и может пересекать внешний шар в зависящем только от размерности n константном количестве гиперплоскостей, образованных фиксированным коэффициентом при самом длинном базисном векторе. Это сводит задачу к нескольким аналогичным задачам более низкой размерности. При реализации атаки необходимо найти *все* решения линейной системы неравенств, так что сводить задачу к меньшим нужно и в случае попадания ближайшей точки сетки в многогранник.

4.2. Сведение задачи к меньшим

Рассмотрим сведение задачи к меньшим более формально. Обозначим за $K = \{x \in \mathbb{R}^n : Ax \leqslant b\}$ исходный многогранник, в методе решения линейных уравнений имеется условие на его ограниченность, но при рассматриваемой атаке оно выполняется. Пусть также $L = \tau\mathbb{Z}^n$ — целочисленная сетка после преобразования. Решение задачи сводится к нахождению пересечения множеств τK и L. Очевидно, что единичная матрица I — матрица базисных векторов сетки $\mathbb{Z}^n$, поэтому τI —

матрица базисных векторов модуля L над кольцом $\mathbb{Z}$. Любой другой базис этой сетки может быть представлен в виде $M(\tau I)$, где M — унимодулярная матрица, так как она обязана быть целочисленной и обратимой, обратная матрица также должна быть целочисленной. Модуль детерминанта такой матрицы равен единице, из этого следует, что для каждого из базисов L, модуль детерминанта матрицы, состоящей из его векторов, зависит только от L и не зависит от базиса; обозначим эту величину за $d(L)$. Пусть базис $(b_1, b_2, \ldots, b_n)$ сетки L будет найденным почти ортогональным базисом:

$$\prod_{i=1}^{n} |b_i| \leqslant c_1 d(L).$$

Также, для всех базисов выполняется обратное неравенство:

$$\prod_{i=1}^{n} |b_i| \geqslant d(L).$$

Докажем следующую лемму:

Лемма 1 *Для любой точки $x \in \mathbb{R}^n$ найдётся такая ближайшая к ней точка $y \in L$, что выполняется неравенство:*

$$|x - y|^2 \leqslant \left(\left| \frac{b_1}{2} \right|^2 + \left| \frac{b_2}{2} \right|^2 + \ldots + \left| \frac{b_n}{2} \right|^2 \right)$$

Доказательство. Доказательство леммы проводится индукцией по n. Для $n = 1$ доказательство очевидно.

Пусть лемма верна для $n-1$. Докажем её для n. Для этого рассмотрим сетку меньшей размерности

$$L' = \left\{ x \in \mathbb{R}^n : x = \sum_{i=1}^{n-1} \alpha_i b_i, \alpha_i \in \mathbb{Z} \right\},$$

а также гиперплоскость в которой она находится:

$$H' = \left\{ x \in \mathbb{R}^n : x = \sum_{i=1}^{n-1} \beta_i b_i, \beta_i \in \mathbb{R} \right\}.$$

Обозначим за h расстояние от точки b_n до гиперплоскости H, причём $h \leqslant |b_n|$. Заметим, что сетка L содержится в объединении гиперплоскости H' и всех параллельных ей, находящихся на расстоянии, кратном h. Из этого следует, что существует такое число m, что расстояние от $x - mb_n$ до гиперплоскости H' не превосходит $\frac{h}{2}$. Этот вектор $x - mb_n$ можно разложить на сумму $x = x_1 + x_2$, где $x_1 \in H', x_2 \in H'^{\perp}$. В качестве искомой точки y возьмём сумму mb_n и ближайшей точки y' сетки L' к точке x_1, воспользовавшись индукционным предположением. В итоге имеем:

$$|x_1 - y'|^2 \leqslant \left(\left|\frac{b_1}{2}\right|^2 + \left|\frac{b_2}{2}\right|^2 + \ldots + \left|\frac{b_{n-1}}{2}\right|^2\right);$$

$$|x - y|^2 = |x_1 - y'|^2 + |x_2|^2 \leqslant$$

$$\leqslant \left(\left|\frac{b_1}{2}\right|^2 + \left|\frac{b_2}{2}\right|^2 + \ldots + \left|\frac{b_{n-1}}{2}\right|^2\right) + \left(\frac{h}{2}\right)^2 \leqslant$$

$$\leqslant \left(\left|\frac{b_1}{2}\right|^2 + \left|\frac{b_2}{2}\right|^2 + \ldots + \left|\frac{b_n}{2}\right|^2\right),$$

что и требовалось доказать. ■

Для удобства перенумеруем номера базисных векторов так, чтобы последний базисный вектор b_n был самым длинным. В этом случае доказанное неравенство можно ослабить и упростить, а именно:

$$|x - y| \leqslant \frac{1}{2}\sqrt{n}|b_n|.$$

Также легко показать, что выполняется неравенство $d(L) = hd(L')$. Из него можно вывести следующее:

$$\prod_{i=1}^{n} |b_i| \leqslant c_1 d(L) = c_1 h d(L') \leqslant c_1 h \prod_{i=1}^{n-1} |b_i|,$$

из чего следует, что

$$|b_n| \leqslant c_1 h.$$

Для оценки количества задач, к которым сводится исходная, осталось рассмотреть полученный при преобразовании τ сферический мно-

гогранник. Его сферичность означает, что существуют точка p и радиусы r и R такие, что:

$$B(p,r) \subset \tau K \subset B(p,R),$$

причём $\frac{r}{R} \geqslant c_2$.

С помощью результатов полученных выше можно доказать теорему:

Теорема 1 *Количество меньших задач, к которым сводится исходная, зависит только от размерности n и не превышает $\frac{c_1\sqrt{n}}{c_2} + 1$.*

Доказательство. При работе алгоритма возникает два случая. Первый — когда ближайшая к p точка y сетки L попадает во внутренний шар, а следовательно и в многогранник. В данном случае задача считается решённой. Второй случай — когда точка y не попадает во внутренний шар, а это значит, что $|y-p| \geqslant r$. Используя ранее выведенные неравенства получаем:

$$c_2 R \leqslant r \leqslant |y-p| \leqslant \frac{1}{2}\sqrt{n}|b_n| \leqslant \frac{1}{2}\sqrt{n}c_1 h;$$
$$2R \leqslant \frac{c_1\sqrt{n}}{c_2}h.$$

Из этого следует, что внешний шар, а следовательно и многогранник, пересекаются не более чем с $\frac{c_1\sqrt{n}}{c_2} + 1$ указанными выше гиперплоскостями, в которых находится сетка L, так как расстояния между ними равны h; таким образом задача сводится к $\frac{c_1\sqrt{n}}{c_2} + 1$ меньшим, а так как коэффициенты c_1 и c_2 зависят только от n, то и количество меньших задач зависит только от n, что и требовалось доказать. ■

При реализации данного алгоритма для атаки стоит отметить несколько моментов.

Первый момент — при нахождении преобразования τ не обязательно вычислять радиусы вышеуказанных шаров. При нахождении ближайшей точки y можно проверять её принадлежность многограннику, а не внутреннему шару. Остаётся проблемой нахождение количества гиперплоскостей, которые необходимо рассмотреть. Заметим, что объём се-

чения многогранника гиперплоскостью как функция от сдвига секущей гиперплоскости является выпуклой. При нахождении коэффициента m при b_n в разложении по базису точки y, m задаёт смещение *центральной* гиперплоскости, центральной в том смысле, что либо точка p лежит в этой гиперплоскости, либо она находится между центральной гиперплоскостью и одной из её соседних. Вместе с фактом положительности объёма многогранника и выпуклости функции это означает, что если некоторое сечение нецентральной гиперплоскости в объёме даёт ноль, то пересечение с многогранником всех следующих гиперплоскостей за рассматриваемой в том же направлении от центральной будет пустым. Эта оптимизация позволяет не вычислять радиусы, а так же не решать заведомо пустые задачи.

Второй момент — для реализации атаки необходимо не только выдавать ответ на вопрос, имеет ли данная система линейных неравенств целочисленное решение, но и находить все эти решения. В этом случае ничего не остаётся, кроме как переходить ко всем меньшим задачам и в случае попадания точки y в многогранник, притом оценки на количество задач становятся неверными. Однако эти системы линейных неравенств построены таким образом, что подавляющее большинство из них не имеет решения.

4.3. Поиск преобразования

В этой главе будет рассмотрен способ нахождения такого линейного обратимого преобразования τ, что многогранник переходит в сферический вид. Общая идея состоит в том, чтобы найти достаточно большой по объёму симплекс внутри многогранника, и преобразовать его к правильной форме. Из того, что симплекс большой (формально это будет показано далее) будет следовать, что многогранник ограничен пересечением двух подобных симплексов, а искомыми шарами будут шар вписанный в первый симплекс, и шар, описанный около пересечения двух симплексов.

В первой части алгоритма находится произвольный симплекс, вер-

шинами которого являются вершины многогранника. Легко показать, что любой такой симплекс будет содержаться в многограннике, в силу выпуклости последнего. В качестве первой вершины v_0 выбирается любая вершина многогранника (находится с помощью обычного линейного программирования с произвольной целевой функцией), а далее запускается итеративный процесс по поиску остальных вершин. Пусть уже выбраны $d+1$ вершина: $v_0, v_1, v_2, \ldots, v_d$, причём вектора $v_1 - v_0$, $v_2 - v_0$, $\ldots$, $v_d - v_0$ линейно независимы и образуют подпространство

$$V = \left\{ \sum_{i=1}^{d} \alpha_i (v_i - v_0), \alpha_i \in \mathbb{R} \right\}.$$

Для следующей вершины симплекса подходят только те v, для которых указанные вектора, а также новый вектор $v - v_0$ будут тоже линейно независимы, то есть такие v, что $(v - v_0) \notin V$. Для нахождения таких v строятся $n - d$ пар линейно независимых линейных функций (в каждой паре одна равняется другой, умноженной на -1), равных нулю на V; далее запускается процесс линейного программирования по ним. Например, можно взять $n - d$ векторов, дополняющих вектора $v_1 - v_0$, $v_2 - v_0$, $\ldots$, $v_d - v_0$ до базиса, и взять функции как скалярные произведения на эти вектора.

В некоторых случаях данный итеративный процесс прекращается до того, как будут набраны все $n+1$ вершин симплекса, а именно тогда, когда на некотором шаге ни одна функция не даст вершины v такой, что $(v - v_0) \notin V$. Это произойдёт в том и только том случае, когда весь многогранник содержится в множестве $V + v_0$, а это значит, что его размерность меньше n, и его объём является нулевым. Как говорилось ранее, в этом случае совершается преобразование к задаче с размерностью, равной размерности многогранника, или, что то же самое, с размерностью пространства V, равной d. Для этого берётся матрица, состоящая из векторов $v_1 - v_0$, $v_2 - v_0$, $\ldots$, $v_d - v_0$, и каждый из её столбцов домножается на некоторое число так, что матрица становится целочисленной, обозначим её за W; заметим, что её размерность равна $n \times d$, а все её столбцы линейно независимы. Далее с помощью построения

Эрмитовой нормальной формы [1] можно получить матрицы U и K такие, что $UW = K$, где U — унимодулярная матрица, а матрица K — целочисленная матрица размерности $n \times d$, причём под её диагональю (при $i > j$) все элементы нулевые. Так как матрица U унимодулярная, то и матрица U^{-1} тоже унимодулярная, а это значит, что столбцы u_1, u_2, …, u_n матрицы U^{-1} образуют базис сетки $\mathbb{Z}^n$. Более того, первые d столбцов матрицы U^{-1} являются базисом пространства V, так как $W = U^{-1}K$, все строки матрицы K ниже строки с номером d являются нулевыми, а V содержит все вектора-столбцы матрицы W.

Возвращаясь к исходной задаче, необходимо найти пересечение целочисленной сетки с многогранником, что в данном случае можно записать в следующем виде: найти такие x, что

$$x = \sum_{i=1}^{n} \alpha_i u_i \in K \subset v_0 + V, \alpha_i \in \mathbb{Z}.$$

Здесь x можно разложить в виде суммы $x = v_0 + x'$, где $x' \in V$. Так как столбцы u_1, u_2, …, u_d образуют базис пространства V, то коэффициенты разложения вектора x' при векторах u_{d+1}, u_{d+2}, …, u_n нулевые, поэтому коэффициенты при разложении x при этих же векторах равны аналогичным коэффициентам у вектора v_0, которые можно легко найти. Если хоть один из них не целый, то множество $v_0 + V$ не пересекается с целочисленной сеткой. Иначе, подставляя в систему неравенств $Ax \leqslant b$ $x = \sum_{i=1}^{d} y_i u_i + \sum_{i=d+1}^{n} \alpha_i u_i$, где α_i — полученные целые коэффициенты, получаем целочисленную систему линейных неравенств относительно d целочисленных переменных y_1, y_2, …, y_d.

Вторая часть алгоритма увеличивает найденный в первой части симплекс, причём увеличение происходит не до максимального по объёму симплекса, а пока его можно увеличить хотя бы в полтора раза. Симплекс увеличивается итеративно, путём замены некоторой его вершины на одну из вершин многогранника. Легко заметить, что не изменяющиеся на конкретной итерации вершины симплекса лежат в одной гиперплоскости, а объём симплекса с этими фиксированными верши-

нами линейно зависит от расстояния между меняющейся вершиной и указанной гиперплоскостью. То есть при выбранных фиксированных n вершинах симплекса максимальное увеличение его объёма достигается при выборе на место последней вершины самой дальней вершины многогранника от гиперплоскости с фиксированными вершинами. На каждой итерации перебирается вершина, которая может меняться, находится нормаль к гиперплоскости, запускается линейное программирование вдоль и против этой нормали, вершина заменяется на найденную, если увеличение объёма не меньше, чем в полтора раза. Очевидно, что данный процесс конечен, более того, количество итераций полиномиально, так как объём симплекса растёт не менее, чем экспоненциально, и ограничен конечным объёмом многогранника. После конца итеративного процесса имеем финальный симплекс. Также легко заметить, что все вершины многогранника, а следовательно и все точки многогранника, находятся от гиперплоскости каждой грани не дальше, чем в полтора раза увеличенная соответствующая высота. Данные условия показывают, что многогранник находится внутри пересечения n пар полупространств, в каждой паре полупространства ограничены параллельными гиперплоскостями, которые также параллельны соответствующей грани финального симплекса, а полупространства направлены друг на друга. Это пересечение можно представить в виде пересечения двух больших (и не обязательно равных) симплексов, подобных финальному, а коэффициенты подобия зависит только от размерности пространства (рис. 3). А это значит, что после преобразования, переводящего финальный симплекс к правильному виду (а в месте с ним и два больших, так как они подобны), отношение c_2 радиусов вписанного шара в финальный симплекс и описанного шара около пересечения двух больших, будет зависеть только от n. Остаётся только составить само преобразование τ. Очевидно, его можно получить перемножив матрицу векторов-рёбер правильного симплекса на обратную матрицу $\begin{pmatrix} v_1 - v_0 & v_2 - v_0 & \dots & v_n - v_0 \end{pmatrix}^{-1}$ векторов-рёбер финального симплекса.

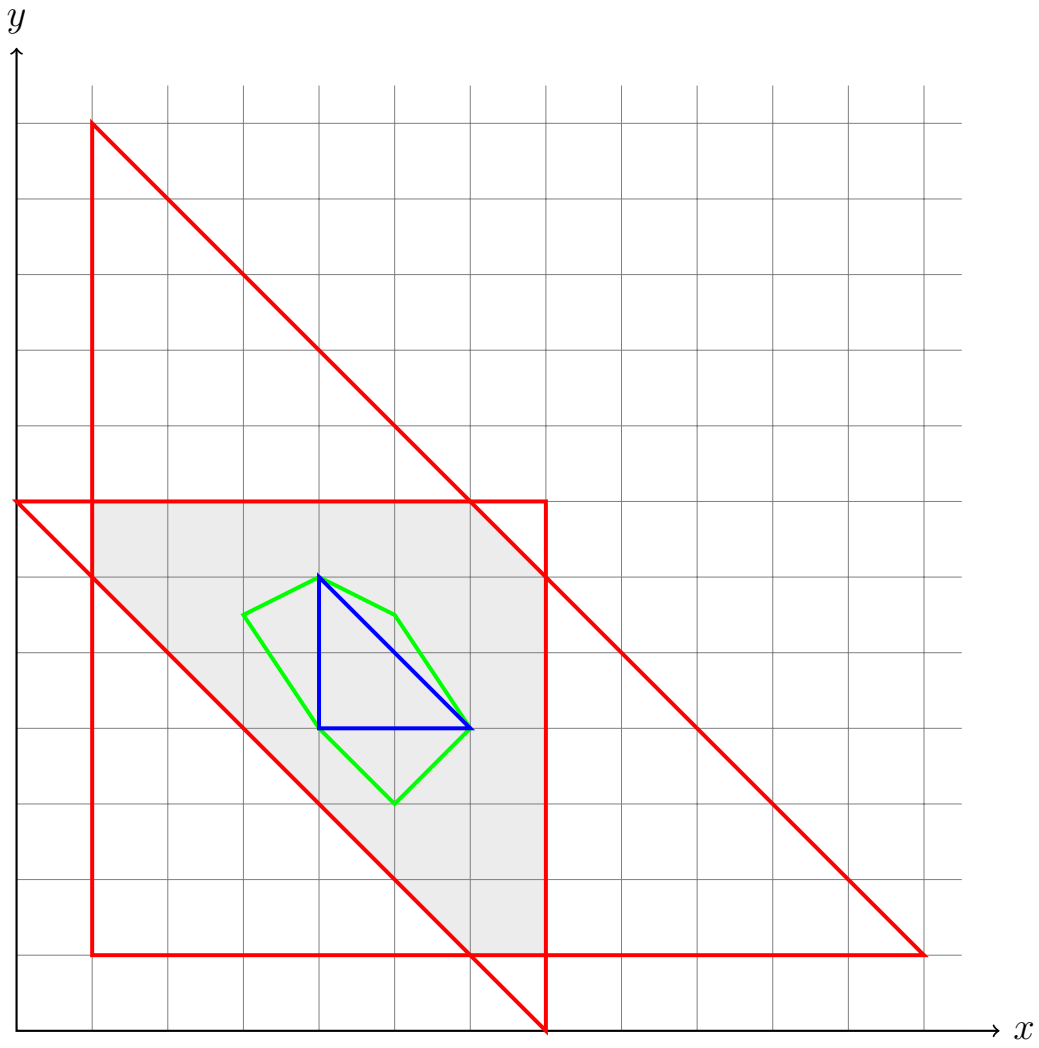

Рис. 3: Зелёным отмечен исходный многогранник, синим — финальный симплекс, красным — два больших симплекса, серым — их пересечение.

4.4. Оптимизация поиска преобразования

При использовании данного алгоритма поиска преобразования для атаки, указанные выше итеративные процессы в обеих частях алгоритма являются довольно громоздкими, так как на каждой из n итераций необходимо искать (или пересчитывать) дополняющие до базиса вектора, а также хотя бы раз запускать линейное программирование. В данной конкретной задаче можно существенно упростить алгоритм, взяв во внимание вид многогранника.

Для понимания формы исходного многогранника рассмотрим систе-

му линейных уравнений, его задающих.

$$0 \leqslant \frac{i}{a_1} - \frac{j}{a_2} \leqslant \frac{1}{2^{n-2}a_2}$$
$$0 \leqslant \frac{i}{a_1} - \frac{k}{a_3} \leqslant \frac{1}{2^{n-3}a_3}$$
$$0 \leqslant \frac{i}{a_1} - \frac{j}{a_4} \leqslant \frac{1}{2^{n-4}a_4}$$

$$1 \leqslant i \leqslant a_1 - 1$$
$$1 \leqslant j \leqslant a_2 - 1$$
$$1 \leqslant k \leqslant a_3 - 1$$
$$1 \leqslant l \leqslant a_4 - 1$$

Последние четыре пары неравенств задают легко представляемую коробку; затем она пересекается с фигурой, заданной первыми тремя парами неравенств. Для определения формы второй фигуры рассмотрим её сечения гиперплоскостью $i = t$ при разных значениях t. После переноса слагаемого с i через знаки неравенств в каждой паре и домножения на отрицательный коэффициент, данные пары неравенств принимают следующий вид:

$$\frac{a_2}{a_1}i - \frac{1}{2^{n-2}} \leqslant j \leqslant \frac{a_2}{a_1}i$$
$$\frac{a_3}{a_1}i - \frac{1}{2^{n-3}} \leqslant k \leqslant \frac{a_3}{a_1}i$$
$$\frac{a_4}{a_1}i - \frac{1}{2^{n-4}} \leqslant l \leqslant \frac{a_4}{a_1}i.$$

Как видно из неравенств, в сечении получается достаточно маленький прямоугольный параллелепипед, причём при изменении i на Δi, всё сечение сдвигается на $\frac{a_2}{a_1}\Delta i$ по j, на $\frac{a_3}{a_1}\Delta i$ по k и на $\frac{a_4}{a_1}\Delta i$ по l. Таким образом первые три пары неравенств задают узкую, бесконечную призму, направляющей которой является вектор $(a_1, a_2, a_3, a_4)^T$. Для представления всего многогранника в целом, необходимо пересечь данную призму с коробкой. Рассмотрим более детально пересечение бесконечной приз-

мы и четырёх гиперплоскостей, соответствующих левым неравенствам в последних четырёх парах (для других четырёх неравенств ситуация аналогичная). Координаты по i точек грани, создаваемой отсечением от бесконечной призмы такой гиперплоскостью, лежат на довольно небольшом отрезке. Для того, чтобы это понять, перепишем неравенства выше уже для i, подставив $j = 1, k = 1, l = 1$:

$$\frac{a_1}{a_2} \leqslant i \leqslant \frac{a_1}{a_2}\left(1 + \frac{1}{2^{n-2}}\right)$$
$$\frac{a_1}{a_3} \leqslant i \leqslant \frac{a_1}{a_3}\left(1 + \frac{1}{2^{n-3}}\right)$$
$$\frac{a_1}{a_4} \leqslant i \leqslant \frac{a_1}{a_4}\left(1 + \frac{1}{2^{n-4}}\right)$$

Также, отсечение гиперплоскостью $i = 1$ даёт отрезок нулевой длины: $1 \leqslant i \leqslant 1$. Если самый правый из этих четырёх отрезков не пересекается со всеми остальными, то в итоге многогранником является четырёхмерный параллелепипед, в ином случае будет почти он, только для одной пары сторон вместо одной будет несколько граней. Остаётся заметить, что первый исход наиболее вероятен, так как для пересечения указанных отрезков необходимо, чтобы разность между некоторыми двумя из значений a_1, a_2, a_3 и a_4 была порядка 2^4 (считая, что порядок этих чисел — 2^n), и вероятность этого события крайне мала. Во втором случае после преобразования многогранника к сферической форме получится многогранник, близкий к параллелепипеду, так как его стороны, состоящие из нескольких граней будут почти плоскими из-за сильной вытянутости исходного многогранника.

Так как в случае четырёхмерного параллелепипеда всего 16 вершин, а во всех остальных случаях вершин не сильно больше, обе части алгоритма поиска преобразования τ можно модернизировать, используя предподсчёт вершин. При известных вершинах найти $n + 1$ из них, которые не лежат в одной гиперплоскости — задача простая. Вторая часть алгоритма также упрощается. Пусть $p_1, p_2, \ldots, p_{verts}$ — все вершины исходного многогранника, их количество равно $verts$; пусть также

$R = \begin{pmatrix} p_1 - v_0 & p_2 - v_0 & \dots & p_{verts} - v_0 \end{pmatrix}$, а S —это n столбцов из матрицы R, соответствующих рёбрам текущего симплекса. Легко показать, что столбцы матрицы ${S^{-1}}^T$ — являются нормалями к гиперплоскостям соответствующих граней, а их модуль такой, что скалярное произведение этой нормали с произвольным вектором-ребром равно отношению текущей высоты симплекса и расстояния от конца этого ребра до гиперплоскости, содержащей сторону; как раз это и используется на каждой итерации во второй части алгоритма. Поэтому на каждой итерации вычисляется $S^{-1}R$, далее находится самое большое по модулю число в этой матрице, заменяется соответственная вершина, пересчитывается матрица S. Когда все элементы матрицы $S^{-1}R$ будут по модулю меньше полутора, необходимо попытаться поменять вершину v_0, для этого в качестве нулевой вершины выбирается другая, пересчитываются матрицы S и R, запускается далее итерационный процесс.

Отдельно стоит рассмотреть процесс предподсчёта вершин. Все неравенства в поставленной задаче — парные, более того, при всех трансформациях исходной задачи (преобразование τ, сведение задачи к меньшим, сведение задачи к меньшей в случае нулевого объёма многогранника) они также остаются парными. Ввиду этого можно решать подругому поставленную задачу: необходимо найти целочисленные решения x системы $c \leqslant Ax \leqslant b$; число строк матрицы A уменьшается вдвое. Для нахождения всех вершин n-мерного многогранника необходимо перебрать все сочетания из n граней, построить матрицу A' из строк матрицы A, являющихся нормалями к этим граням, обратить A' и домножить её на соответствующий вектор b' (при стандартной задаче $Ax \leqslant b$). При парной постановке задачи не только уменьшается количество строк, из которых надо брать сочетания, но также отсутствуют повторяющиеся вычисления, так как для пары параллельных граней есть общая нормаль, которую не надо подставлять несколько раз в матрицу A'. Более того, в исходной задаче, когда её размерность максимальна и равна четырём, обычно достаточно обращать матрицу всего четыре раза, так как три направления граней при нахождении вершин точно известны (направления из первых трёх пар неравенств).

Перед предподсчётом вершин после трансформаций исходной задачи может сложиться ситуация, когда две строки матрицы A отличаются только домножением на скаляр. В этом случае их можно объединить, при этом также снизится число сочетаний строк матрицы A в переборе. Также после нахождения всех вершин может оказаться так, что некоторые пары неравенств выполняются строго для всех вершин, а значит, и для всех точек многогранника. Такие неравенства в дальнейшем рассмотрении не нужны, так как не изменяют многогранник. Для эффективности перебора на следующих стадиях алгоритма такие пары неравенств нужно удалять.

Так как часто многогранником будет являться параллелепипед, можно уменьшить количество меньших задач, к которым сводится исходная. Дело в том, что при указанном преобразовании τ, к правильному виду преобразовывается некоторый симплекс внутри параллелепипеда, центр шаров при этом оказывается не в его середине (что уменьшает радиус внутреннего шара и увеличивает радиус внешнего). Дополнительной проблемой преобразования симплекса к правильному виду является то, что при некоторых размерностях (например при $n = 2$) его вершины имеют иррациональные координаты, что не подходит для нашей задачи; приходится использовать приближение координат симплекса, например с помощью подходящих дробей. Есть альтернативный вариант, решающий вышеуказанные проблемы: преобразовывать к правильному виду не симплекс, а весь многогранник. Конечно, это стоит делать, когда в него вписан не только симплекс, но и соответствующий параллелепипед, что легко проверяется путём подстановки в систему неравенств точки $v_0 + \sum_{i=1}^{n} p_i - v_0$. В этом случае многогранник превращается в гиперкуб, координаты которого, очевидно, рациональные, а отношения радиусов внутреннего и внешнего шаров лучше.

4.5. Изменение базиса

Нераскрытой осталась только одна часть алгоритма — преобразование базиса к почти ортогональному виду. Общий план данной ча-

сти таков. Пусть $b_1, b_2, \ldots, b_n$ — текущий базис сетки. Преобразование любого базиса сетки к любому другому, как отмечалось ранее, эквивалентно линейному преобразованию, которому соответствует унимодулярная матрица. А это, в свою очередь, значит, что в базисе можно только менять местами вектора и прибавлять один базисный вектор, умноженный на целое число, к другому.

Итак, рассмотрим обычную ортогонализацию Грамма-Шмидта для рассматриваемого базиса. Обозначим за b'_i проекцию вектора b_i на ортогональное дополнение пространства, образованного предшествующими векторами $b_1, b_2, \ldots, b_{i-1}$. Также обозначим используемые в ортогонализации коэффициенты

$$\mu_{ij} = \frac{(b_i, b'_j)}{(b'_j, b'_j)},$$

причём

$$b'_i = b_i - \sum_{j=1}^{i-1} \mu_{ij} b'_j.$$

В статье Ленстра [2] доказано, что для базиса условие почти ортогональности выполняется, если выполняются следующие неравенства:

$$|\mu_{ij}| \leqslant 0.5, \quad 1 \leqslant j < i \leqslant n$$
$$|b'_i + \mu_{ii-1} b'_{i-1}|^2 \geqslant \frac{3}{4} |b'_{i-1}|^2, \quad 1 < i \leqslant n.$$

Второе неравенство здесь означает, что вектор $b'_i + \mu_{ii-1} b'_{i-1}$, являющийся проекцией b_i на ортогональное дополнения пространства векторов $b_1, b_2, \ldots, b_{i-2}$, превосходит по длине три четверти проекции вектора b_{i-1} на то же пространство.

Достигаются данные условия путём итеративного процесса, также указанного в статье [2]: на каждом его шаге неравенства выполняются при $i, j \leqslant k$, при некотором k. Если для следующего k не выполняется первое неравенство при $i = k, j = k - 1$, то к вектору b_k прибавляется столько векторов b_{k-1}, чтобы оно выполнялось. Если далее второе неравенство выполняется, то исправляется первое неравенство для всех остальных j, иначе вектора b_k и b_{k-1} меняются местами. Одновременно

со всеми операциями пересчитываются текущие значения для коэффициентов μ_{ij}, векторов b_i, а так же квадратов длин векторов b'_i. Однако для задачи линейного целочисленного программирования (в части нахождения ближайшей точки сетки к центру шаров) необходимо также знать вектора b'_i, поэтому в реализации они также пересчитываются.

5. Численный эксперимент

Так как подавляющую часть вычислительного процесса занимает решение системы линейных неравенств, будет рассмотрена эффективность именно этого алгоритма. Ввиду большого количества работы с матрицами, а также из-за своей сложности, алгоритм был реализован на языке Python 2.6. Эксперимент проходил следующим образом: вычислялось среднее время работы для каждого из рассматриваемых длин ключа. Для каждой длины было взято три случайных публичных ключа, для каждого из них десять раз случайно выбирались четыре элемента и для них решалась система линейных неравенств. В итоге для каждого n было получено среднее время решения t. Так как при атаке необходимо перебрать все четвёрки весов из n (с учётом порядка), то общее время атаки оценивается временем $T = n^4 t$.

Для исследования асимптотики T предположим, что $T = \gamma n^\alpha$ (так как алгоритм является полиномиальным). Если взять натуральный логарифм от обеих частей, получим равенство:

$$ln(T) = \alpha ln(n) + ln(\gamma),$$

а учитывая оценку для T получаем:

$$\begin{gathered} ln(n^4 t) = \alpha ln(n) + ln(\gamma) \\ 4ln(n) + ln(t) = \alpha ln(n) + ln(\gamma) \\ ln(t) = (\alpha - 4) ln(n) + ln(\gamma). \end{gathered}$$

Стоит заметить, что нельзя брать логарифм от размерной величины, однако здесь вместо $ln(t)$ имеется в виду $ln(\frac{t}{t_0})$, где t_0 равняется одной секунде. Аналогичные рассуждения применяются и к T, а коэффициент γ при таких условностях безразмерен. Таким образом логарифм времени решения системы линейных неравенств должен линейно зависеть от логарифма длины ключа. Поэтому интересующий нас коэффициент α легко найти с помощью линейной аппроксимации графика $ln(t)$ от $ln(n)$ (рис. 4).

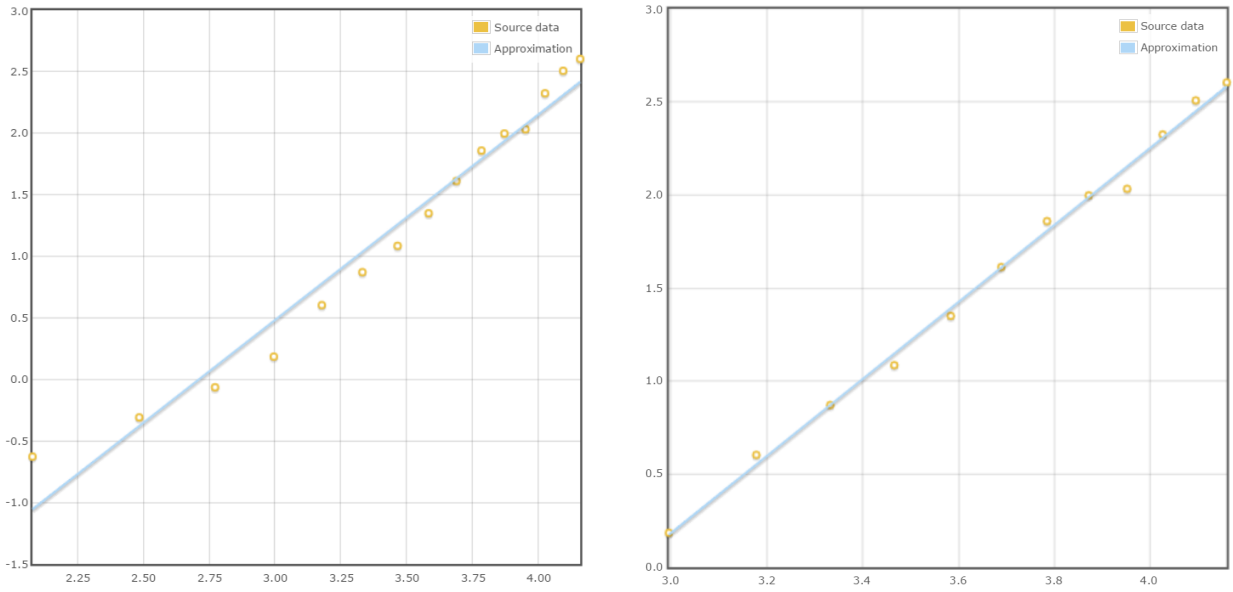

Рис. 4: Графики $ln(t)$ от $ln(n)$ и их линейные аппроксимации

Линейная аппроксимация выдала коэффициент наклона $a = (1.67 \pm 0.18)$, а также смещение прямой $b = (-4.5 \pm 0.6)$. Стоит заметить, что при отбросе из рассмотрения первых трёх точек (отклонение могло внести то, что эти точки соответствуют небольшим значениям n), остальные гораздо лучше ложатся на прямую. Второй раз полученные коэффициенты равны $a = (2.07 \pm 0.10)$, $b = (-6.0 \pm 0.4)$. Таким образом решение системы линейных неравенств выполняется за квадратичное время, а показатель у самой атаки равен 6. Остаётся отметить, что данная атака легко распараллеливается: решения систем линейных неравенств независимы и могут вычисляться на разных ядрах процессора. Для предлагаемых Мерклом и Хеллманом $n = 100$ при количестве ядер порядка 10^4 вычисления займут порядка 10^8 операций, что выполняется сравнительно быстро.

Заключение

В итоге было показано, что система Меркла-Хеллмана действительно ненадёжна на практике. При продолжении её изучения можно рассматривать модификации, например, когда множество весов разбито на несколько частей, в каждой из которых веса образуют супервозрастающую последовательность только по своему модулю. Однако гораздо более широкий простор действий открывают ранцевые криптосистемы с объектами-векторами. Например, ключ в системе Нидеррайтера (одной из таких систем) обязан быть большим по памяти для обеспечения стойкости системы, это происходит вследствие малого веса используемых ошибок, и это возможно исправить. Для дальнейших исследований можно сформулировать следующую задачу: изучить основанные на теории кодирования ранцевые криптосистемы с объектами-векторами при разных способах кодирования и различных заданиях множества допустимых ошибок.

Список литературы

[1] Kannan Ravindran, Bachem Achim. Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix // SIAM Journal on Computing. — 1979. — Vol. 8, no. 4. — P. 499–507.

[2] Lenstra A. K., Lenstra H. W. Jr., Lovász Lászlo. Factoring polynomials with rational coefficients // Math. Ann. — 1982. — Vol. 261. — P. 515–534.

[3] Lenstra H. W. Jr. Integer programming with a fixed number of variables // Math. Oper. Res. — 1983. — Vol. 8. — P. 538–548.

[4] McEliece R. J. A Public-Key Cryptosystem Based On Algebraic Coding Theory // Deep Space Network Progress Report. — 1978. — Vol. 44. — P. 114–116.

[5] Merkle Ralph C., Hellman Martin E. Hiding information and signatures in trapdoor knapsacks // IEEE Transactions on Information Theory. — 1978. — Vol. 24, no. 5. — P. 525–530.

[6] Niederreiter Harald. Knapsack-type cryptosystems and algebraic coding theory // Problems of Control and Information Theory. — 1986. — Vol. 15, no. 2. — P. 159–166.

[7] Shamir Adi. A polynomial-time algorithm for breaking the basic Merkle-Hellman cryptosystem // IEEE Transactions on Information Theory. — 1984. — Vol. 30, no. 5. — P. 699–704.