Описание алгоритмов

Использование решёток для анализа рюкзачной криптосистемы Меркла–Хеллмана

Описание криптосистемы

Криптосистема Меркла–Хеллмана предложена Ральфом Мерклом и Мартином Хеллманом в [MH78] и основана на вычислительной сложности задачи о рюкзаке.

Пусть сообщение разбито на блоки $P^{(i)}, i = 1,\ldots,N$ длины n битов каждый. Каждый блок $P^{(i)}$ представляет собой некоторое число в двоичной записи:

$$P^{(i)} = \sum_{j=1}^{n} p_j^{(i)} \cdot 2^{j-1},\ p_j^{(i)} \in \{0,1\}.$$

Рассмотрим некоторый рюкзак (n-вектор) с целыми положительными весами $A = (a_1, a_2, \ldots, a_n) \in \mathbf{Z}^n$. Если выполнено условие

$$a_j > \sum_{k=1}^{j-1} a_k,\ j = 2,\ldots,n, \tag{1}$$

то такой рюкзак называется сверхвозрастающим. В случае сверхвозрастающих рюкзаков при заданном целом значении $0 \le S \le \sum_{j=1}^{n} a_j$ легко определить вектор коэффициентов $X = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ такой, что

$$S = \sum_{j=1}^{n} x_j a_j, \tag{2}$$

либо установить невозможность такого представления. (Данная задача для произвольных значений a_j называется задачей о рюкзаке.)

В самом деле,

$$x_n = \begin{cases} 1, & a_n \le S \\ 0, & a_n > S \end{cases}, \quad x_j = \begin{cases} 1, & a_j \le S - \sum\limits_{k=j+1}^{n} x_k a_k \\ 0, & a_j > S - \sum\limits_{k=j+1}^{n} x_k a_k \end{cases}, \quad j = n-1,\ldots,1,$$

при этом равенство $S = \sum_{k=1}^{n} x_k a_k$ означает, что вектор X найден, а неравенство — невозможность представления (2).

Используя линейное модулярное преобразование

$$b_j = l a_j \bmod m, \tag{3}$$

из сверхвозрастающего рюкзака A можно получить нормальный (т.е. не обладающий свойством (1)) рюкзак $B = (b_1, b_2, \ldots, b_n) \in \mathbf{Z}^n$, для которого решение задачи (2) с точки зрения вычислительной сложности не является столь тривиальным. Если при этом выполняются соотношения

$$(l, m) = 1, \quad m > \sum_{j=1}^{n} a_j,$$

то преобразование (3) является обратимым. Данное свойство используется в крптосистеме Меркла–Хеллмана.

В несколько упрощённой форме криптосистема Меркла–Хеллмана в качестве секретного ключа использует совокупность величин $a_1, a_2, \ldots, a_n, l, m$, а в качестве открытого — величины $b_1, b_2, \ldots, b_n$. Процесс зашифрования блока $P^{(i)}$ открытого текста выглядит следующим образом

$$C^{(i)} = \sum_{j=1}^{n} p_j^{(i)} \cdot b_j\,, \tag{4}$$

где $C^{(i)}$ — блок шифртекста, соответствующий $P^{(i)}$. Расшифрование $C^{(i)}$ происходит при помощи обращения преобразования (3):

$$\widetilde{C}^{(i)} = l^{(-1)} C^{(i)} \bmod m = \sum_{j=1}^{n} p_j^{(i)} \cdot \left(l^{(-1)} b_j\right) \bmod m = \sum_{j=1}^{n} p_j^{(i)} \cdot a_j \bmod m = \sum_{j=1}^{n} p_j^{(i)} \cdot a_j\,,$$

где $l^{(-1)}$ — элемент обратный l в кольце $\mathbf{Z}/m\mathbf{Z}$ вычетов по модулю m. При этом неизвестные значения $p_j^{(i)}$ находятся как решения задачи (2) при $S = \widetilde{C}^{(i)}$.

Алгоритм анализа

В работе [CJL+92] было показано, как можно свести задачу о рюкзаке для некоторого достаточно широкого класса рюкзаков к задаче поиска кратчайшего вектора решётки.

Предположим, что нам необходимо для рюкзака $B = (b_1, b_2, \ldots, b_n)$ и значения S решить следующую задачу

$$S = \sum_{j=1}^{n} x_j b_j\,, \quad X = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n\,.$$

В силу (4) её решение для $S = C^{(i)}$ приводит к расшифрованию $P^{(i)}$.

Плотностью рюкзака $B = (b_1, b_2, \ldots, b_n)$ называется величина

$$d = \frac{n}{\log_2\left(\max_j b_j\right)}.$$

Рассмотрим решётку $L = L(M)$, порождённую вектор-стобцами следующей $(n+1)\times(n+1)$-матрицы

$$M = \begin{pmatrix} Kb_1 & Kb_2 & \cdots & Kb_n & KS \\ 2 & 0 & \cdots & 0 & 1 \\ 0 & 2 & \cdots & 0 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 2 & 1 \end{pmatrix},$$

где K — достаточно большое целое.

Теорема ([CJL+92]).

Пусть $b_1, b_2, \ldots, b_n$ случайные неотрицательные целые числа, ограниченные сверху некоторой величиной D. Пусть также $X = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ — произвольный бинарный n-вектор и пусть $S = \sum_{j=1}^{n} x_j b_j$. Если для плотности d рюкзака $B = (b_1, b_2, \ldots, b_n)$ выполнено неравенство $d < 0.9408\ldots$, то задачу о рюкзаке для B можно «почти всегда» решить за полиномиальное время, совершив лишь одно обращение к оракульному алгоритму поиска кратчайшего вектора решётки $L(M)$.

Замечания:

1. Под оракульным алгоритмом понимается абстрактная возможность получить решение некоторой задачи (по которой «специализируется оракул») за постоянное (независящее от входных данных) время.
2. Вместо оракульного алгоритма в данном случае можно использовать приближённый алгоритм поиска кратчайшего вектора решётки, например, LLL-алгоритм или его модификацию.
3. Наряду с решёткой $L(M)$ можно использовать решётку $L(M')$, порождённую вектор-столбцами $(n+2)\times(n+1)$-матрицы M':

$$M' = \begin{pmatrix} Kb_1 & Kb_2 & \dots & Kb_n & -KS \\ n+1 & -1 & \dots & -1 & -1 \\ -1 & n+1 & \dots & -1 & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -1 & -1 & \dots & n+1 & -1 \\ -1 & -1 & \dots & -1 & n+1 \end{pmatrix}.$$

Таким образом, чтобы расшифровать i-й блок шифртекста $C^{(i)}$, необходимо выполнить следующие действия.

1. Положить $S = C^{(i)}$.
2. Для известного открытого ключа $B = (b_1, b_2, \dots, b_n)$ построить матрицу M (либо M').
3. Запустить LLL-алгоритм для решётки $L(M)$ (соответственно $L(M')$) и получить, таким образом, представление $\tilde{M} = MU$ (соответственно $\tilde{M}' = M'U$). Здесь столбцы матрицы $\tilde{M}$ ($\tilde{M}'$) — LLL-приведённый базис решётки $L(M)$ ($L(M')$).
4. Используя полученное на предыдущем шаге представление, проверить, что $|U_{j,1}| \le 1$, $j = 1,\dots,n$ и $\sum_{j=1}^{n} |U_{j,1}| b_j = S$.
5. Если проверяемые условия выполнены, то положить i-й блок сообщения равным $P^{(i)} = \sum_{j=1}^{n} |U_{j,1}| \cdot 2^{j-1}$. В противном случае считаем, что i-й блок шифртекста $C^{(i)}$ расшифровать не удалось. (При этом, если выполнено только первое из условий, то необходимо увеличить значение K.)

Примечание. *При использовании библиотеки NTL следует помнить, что LLL-процедуры из данной библиотеки работают с вектор-строками матриц. Поэтому перед использованием LLL-процедуры матрицу M (M') следует транспонировать (равно как и найденную матрицу U после работы процедуры).*

Список источников

[CJL+92] M.J.Coster, A.Joux, D.A.LaMacchia, A.Odlyzko, C.–P.Schnorr, and J.Stern, "Improved Low-Density Subset Sum Algorithms", Comput. Complexity, 2:11–28, 1992.

[MH78] R.C.Merkle and M.Hellman, "Hiding Information and Signatures in Trapdoor Knapsacks", IEEE Transactions on Information Theory, v.24, n.5, Sep 1978, pp.525–530.

Использование решёток для анализа усечённого линейного конгруэнтного генератора

Описание генератора

Линейный конгруэнтный генератор с параметрами x_0, a, b, m порождает последовательность

$$x_{i+1} = ax_i + b \bmod m, i \geq 0 .$$

Усечённый линейный конгруэнтный генератор с параметрами x_0, a, b, m, α, где $0 < \alpha < 1$ порождает последовательность

$$y_i = \lfloor x_i / 2^{\beta\nu} \rfloor, \beta = 1 - \alpha , \ i \geq 0 ,$$

при этом величина $\alpha\nu$ является целым числом, где $\nu = \lceil \log_2(m+1) \rceil$ — число знаков в двоичном представлении m.

Алгоритм анализа

Задача анализа состоит в восстановлении неизвестных параметров x_0, a, b, m по значениям последовательности $y_i, i > 0$. Мы условно поделим алгоритм анализа на три этапа. На первом этапе будет получен набор многочленов $P_i(x)$, обладающих свойством $P_i(a) \equiv 0 \bmod m$. На втором этапе, располагая набором многочленов $P_i(x)$, мы найдём неизвестные модуль m и множитель a. И, наконец, на третьем этапе для упрощённого случая $b = 0$ по значениям m и a мы определим начальное состояние x_0.

В работе [JS98] был предложен метод восстановления неизвестных значений m и a при помощи построения вспомогательной последовательности многочленов $P_i(x)$.

Этап 1

Располагая набором значений $y_i, i > 0$, построим последовательность из n векторов размерности t каждый (n и t — некоторые величины, выбираемые криптоаналитиком):

$$V_i = \begin{pmatrix} y_{i+1} - y_i \\ y_{i+2} - y_{i+1} \\ \cdots \\ y_{i+t} - y_{i+t-1} \end{pmatrix}, i = 1, \ldots, n .$$

Рассмотрим решётку $L = L\left(M^{(1)}\right)$, порождаемую вектор-столбцами $(n+t) \times n$ матрицы $M^{(1)}$, имеющей следующий блочный вид:

$$M^{(1)} = \begin{pmatrix} KV_1 & KV_2 & \cdots & KV_n \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix},$$

где K —некоторое целое число. Применим LLL-алгоритм для $L = L\left(M^{(1)}\right)$. Если K достаточно велико, то короткий вектор, найденный LLL-алгоритмом (первый столбец LLL-приведённой матрицы $\widetilde{M}^{(1)}$, если под LLL-приведённой матрицей понимать матрицу, столбцами которой являются векторы LLL-приведённого базиса решётки), будет иметь вид: $\widetilde{M}_1^{(1)} = \lambda_1 M_1^{(1)} + \lambda_2 M_2^{(1)} + \ldots + \lambda_n M_n^{(1)}$. (Здесь M_i обозначает i-й вектор-столбец некоторой матрицы M.)

Кроме этого, введём в рассмотрение ненаблюдаемые векторы

$$W_i = \begin{pmatrix} x_{i+1} - x_i \\ x_{i+2} - x_{i+1} \\ \ldots \\ x_{i+t} - x_{i+t-1} \end{pmatrix}, i = 1, \ldots, n.$$

В [JS98] показано, что если модуль m для произвольного простого p не делится на p^2, то при определённых значениях параметров n и t для найденных значений λ_i и ненаблюдаемых векторов W_i выполняется равенство

$$\lambda_1 W_1 + \lambda_2 W_2 + \ldots + \lambda_n W_n = 0. \tag{5}$$

Несложно убедиться, что для последовательности x_i справедливо соотношение

$$x_{i+j+1} - x_{i+j} \equiv a^j (x_{i+1} - x_i) \bmod m.$$

Таким образом, если рассмотреть первую координату нулевого вектора (5), то можно увидеть, что по модулю m она будет равна

$$(x_1 - x_0) \sum_{i=1}^{n} \lambda_i a^{i-1} \bmod m.$$

Поэтому в случае, когда $x_1 - x_0$ и m взаимнопросты, будет найден многочлен $P(x) = \sum_{i=1}^{n} \lambda_i x^{i-1}$ со свойством $P(a) \equiv 0 \bmod m$.

Резюмируя сказанное, алгоритм для построения аннулирующего многочлена $P(x)$ будет следующим.

1. По последовательности $\{y_i, i > 0\}$ построить векторы V_i, $i = 1, \ldots, n$.
2. Для векторов V_i составить матрицу $M^{(1)}$ и применить LLL-алгоритм, получив, таким образом, представление $\widetilde{M}^{(1)} = M^{(1)} U^{(1)}$, где столбцы $\widetilde{M}^{(1)}$ — составляют LLL-приведённый базис решётки $L = L\left(M^{(1)}\right)$.
3. Проверить, что $\widetilde{M}_{i,1} = 0$, $i = 1, \ldots, t$. Если хотя бы одно из этих равенств не верно, то необходимо увеличить значение K (при условии, что $n > t$) и перейти к шагу 1.
4. Построить многочлен $P(x) = \sum_{i=1}^{n} \lambda_i x^{i-1}$, положив $\lambda_i = U_{i,1}^{(1)}$.

Для нахождения нескольких аннулирующих многочленов необходимо повторить данную процедуру несколько раз.

Этап 2

Пусть теперь имеется набор аннулирующих многочленов $P_i(x) = \sum_{j=1}^{n} \lambda_{ji} x^{j-1}$, обладающих свойством $P_i(a) \equiv 0 \bmod m$ и необходимо найти модуль m и множитель a.

Из набора аннулирующих многочленов $P_i(x)$ выберем поднабор $P_i(x) = \sum_{j=1}^{n} \lambda_{ji} x^{j-1}$, $i = 1, \ldots, n$ такой, чтобы матрица

$$\Lambda = \begin{pmatrix} \lambda_{11} & \lambda_{12} & \cdots & \lambda_{1n} \\ \lambda_{21} & \lambda_{22} & \cdots & \lambda_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{n1} & \lambda_{n2} & \cdots & \lambda_{nn} \end{pmatrix} \qquad (6)$$

являлась матрицей полного ранга (другими словами, мы выбираем поднабор из n многочленов линейно независимых над полем действительных чисел). В [JS98] приведены эвристические аргументы в пользу возможности это сделать. Рассмотрим решётку $L(\Lambda)$, порождаемую столбцами Λ. Заметим, что многочлен $P(x) = \sum_{j=1}^{n} \lambda_j x^{j-1}$ можно представить в виде $P(x) = \sum_{j=2}^{n} \lambda_j \left(x^{j-1} - a^{j-1}\right) + \sum_{j=2}^{n} \lambda_j a^{j-1} + \lambda_1$. Но поскольку $P(a) \equiv 0 \bmod m$, то $\sum_{j=2}^{n} \lambda_j a^{j-1} + \lambda_1 = km$ для некоторого целого k. Следовательно, любой аннулирующий многочлен представляет собой целочисленную $(k, \lambda_2, \ldots, \lambda_n \in \mathbf{Z})$ линейную комбинацию многочленов $m, x-a, x^2-a^2, \ldots, x^{n-1}-a^{n-1}$. Значит, найдутся такие целые $\alpha_1, \alpha_2, \ldots, \alpha_n$, что столбцы матрицы

$$\Lambda' = \begin{pmatrix} \alpha_1 m & -\alpha_2 a & -\alpha_3 a^2 & \cdots & -\alpha_n a^{n-1} \\ 0 & \alpha_2 & 0 & \cdots & 0 \\ 0 & 0 & \alpha_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_n \end{pmatrix}$$

порождают решётку в точности равную $L(\Lambda)$. Из этого вытекает, что определитель матрицы Λ, совпадающий с определителем Λ', равен $\widetilde{m} = m\prod_{i=1}^{n} \alpha_i$ и, следовательно, делится на m. Повторив вышеописанную процедуру несколько раз для других наборов многочленов $P_i(x)$, мы получим некоторое количество чисел $\widetilde{m}_i$, кратных модулю m. В [JS98] приводятся эвристические аргументы в пользу того, что при достаточном количестве $\widetilde{m}_i$ их наибольший общий делитель будет совпадать с m.

Покажем теперь как, зная m (m = НОД $\widetilde{m}_i$), определить a. Умножим строки с номерами $3, 4, \ldots, n$ матрицы Λ на некоторое достаточно большое целое число K. Получим матрицу

$$M^{(2)} = \begin{pmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} & \cdots & \lambda_{1n} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} & \cdots & \lambda_{2n} \\ K\lambda_{31} & K\lambda_{32} & K\lambda_{33} & \cdots & K\lambda_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ K\lambda_{n1} & K\lambda_{n2} & K\lambda_{n3} & \cdots & K\lambda_{nn} \end{pmatrix}.$$

Применим к решётке $L\left(M^{(2)}\right)$ LLL-алгоритм и получим, таким образом, матрицу $\widetilde{M}^{(2)} = M^{(2)} U^{(2)}$, столбцы которой составляют LLL-приведённый базис решётки $L\left(M^{(2)}\right)$. Если K достаточно велико, то первые два столбца матрицы

$\widetilde{M}^{(2)}$ будут являться линейной комбинацией двух первых столбцов матрицы Λ', т.е. будут иметь вид:

$$\begin{pmatrix} \beta_1 m - \gamma_1 a & \beta_2 m - \gamma_2 a \\ \gamma_1 & \gamma_2 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{pmatrix}. \qquad (7)$$

Это легко видеть, если заметить, что столбцы матрицы

$$\Lambda'' = \begin{pmatrix} \alpha_1 m & -\alpha_2 a & -\alpha_3 a^2 & \cdots & -\alpha_n a^{n-1} \\ 0 & \alpha_2 & 0 & \cdots & 0 \\ 0 & 0 & K\alpha_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & K\alpha_n \end{pmatrix}$$

порождают решётку $L\left(M^{(2)}\right)$. В таком случае присутствие в линейной комбинации любого столбца Λ'' с номером, начиная с 3, приводит к тому, что длина этой линейной комбинации будет не меньше K, т.е. не будет короткой. В то же время линейные комбинации (7) являются достаточно короткими и поэтому будут найдены LLL-алгоритмом.

Если наибольший общий делитель чисел $\gamma_1 = \widetilde{M}_{21}^{(2)}, \gamma_2 = \widetilde{M}_{22}^{(2)}, m$ равен 1, то возможно представление $\kappa_1\gamma_1 + \kappa_2\gamma_2 + \kappa m = 1$ (его можно получить, используя алгоритм Евклида). Тогда a находится по формуле

$$-\kappa_1 \widetilde{M}_{11}^{(2)} - \kappa_2 \widetilde{M}_{12}^{(2)} \bmod m = -\kappa_1\left(\beta_1 m - \gamma_1 a\right) - \kappa_2\left(\beta_2 m - \gamma_2 a\right) \bmod m = a. \qquad (8)$$

Итак, подытожим сказанное. Для нахождения значений m и a по последовательности аннулирующих многочленов $P_i(x)$ необходимо выполнить следующие действия.

1. Выбрать из последовательности $P_i(x)$ поднабор из n линейно независимых над полем действительных чисел многочленов.
2. Из коэффициентов выбранных многочленов составить матрицу Λ (6).
3. Вычислить $\widetilde{m} = \det\Lambda$. Полученное число $\widetilde{m}$ будет кратно m.
4. Повторить шаги 1 – 3 для нахождения нескольких значений $\widetilde{m}_i$.
5. Найти наибольший общий делитель чисел $\widetilde{m}_i$ и взять его в качестве значения модуля m. (Естественно, для нахождения m данным способом количество $\widetilde{m}_i$ не должно быть слишком мало.)
6. Для набора линейно независимых над полем действительных чисел многочленов $P_i(x)$ составить матрицу $M^{(2)}$ и применить к ней LLL-алгоритм для нахождения матрицы $\widetilde{M}^{(2)} = M^{(2)}U^{(2)}$.
7. Положить $\gamma_1 = \widetilde{M}_{21}^{(2)}, \gamma_2 = \widetilde{M}_{22}^{(2)}$ и проверить, что НОД$(\gamma_1, \gamma_2, m) = 1$.
8. Если γ_1, γ_2, m не взаимнопросты, то вывести сообщение об ошибке: «МНОЖИТЕЛЬ НЕ НАЙДЕН». Иначе при помощи алгоритма Евклида найти представление $\kappa_1\gamma_1 + \kappa_2\gamma_2 + \kappa m = 1$ и вычислить множитель a по формуле (8).

Этап 3
Считаем, что значения m и a известны. Для упрощения рассуждений будем также считать, что параметр генератора $b = 0$. По последовательности значений $y_i, i > 0$ нам необходимо найти начальное состояние генератора x_0.

Следуя [FHK+88], изложим общую идею определения значений x_i по известным y_i $1 \le i \le k$, где k — некоторое целое, выбираемое криптоаналитиком.

Поскольку $b = 0$, то для последовательности $x_i, i > 0$ линейного конгруэнтного генератора будут справедливы соотношения $x_{i+1} = ax_i \bmod m$. Следовательно $a^{i-1}x_1 - x_i \equiv 0 \bmod m$, $2 \le i \le k$.

Поскольку $mx_1 \equiv 0 \bmod m$, то можно рассмотреть систему сравнений (векторы сравниваются покоординатно)

$$XM^{(3)} \equiv \mathbf{0}_k \bmod m, \tag{9}$$

где $\mathbf{0}_k$ — нулевой k-вектор-строка, $X = (x_1, x_2, \ldots, x_k)$, а $k \times k$-матрица $M^{(3)}$ имеет следующий вид

$$M^{(3)} = \begin{pmatrix} m & a & a^2 & \cdots & a^{k-1} \\ 0 & -1 & 0 & \cdots & 0 \\ 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -1 \end{pmatrix}.$$

Применим к столбцам матрицы $M^{(3)}$ LLL-алгоритм и получим, таким образом, представление $\widetilde{M}^{(3)} = M^{(3)}U^{(3)}$. В силу свойств LLL-алгоритма столбцы матрицы $\widetilde{M}^{(3)}$ будут иметь достаточно короткую евклидову норму. Предположим, что для вектор-столбцов $\widetilde{m}_i^{(3)}$ матрицы $\widetilde{M}^{(3)}$ выполнено соотношение

$$\max_{1 \le i \le k} \left\| \widetilde{m}_i^{(3)} \right\| \le \frac{m}{2^{\beta\nu+1}\sqrt{k}} \tag{10}$$

(заметим, что достаточно проверить данное соотношение только для $\widetilde{m}_k^{(3)}$, поскольку LLL-алгоритм упорядочивает вектор-столбцы $\widetilde{M}^{(3)}$ по возрастанию длин). Тогда можно предложить следующий способ восстановления x_i по y_i.

Величины x_i представим в виде $x_i = y_i 2^{\beta\nu} + z_i$, где y_i — известные $\alpha\nu$ старших битов последовательности x_i, а z_i — неизвестные $\beta\nu$ младших битов. Введём в рассмотрение следующие вектор-строки $Y = (y_1, y_2, \ldots, y_k)$, $Z = (z_1, z_2, \ldots, z_k)$, $C = (c_1, c_2, \ldots, c_k) \equiv -2^{\beta\nu} Y\widetilde{M}^{(3)}$, причём c_i выбираются таким образом, чтобы выполнялось соотношение $-m/2 < c_i \le m/2$. Тогда сравнения (9) запишутся в виде

$$XM^{(3)} \equiv \mathbf{0}_k \bmod m \Leftrightarrow \left(2^{\beta\nu} Y + Z\right)\left(M^{(3)}U^{(3)}\right) \equiv \mathbf{0}_k \bmod m \Leftrightarrow Z\widetilde{M}^{(3)} \equiv C \bmod m.$$

Заметим, что в силу (10) можем записать

$$\left| \sum_{i=1}^{k} z_i \widetilde{M}_{ij}^{(3)} \right| \le \|Z\| \cdot \left\| \widetilde{m}_j^{(3)} \right\| \le \left(2^{\beta\nu} - 1\right)\sqrt{k} \cdot \frac{m}{2^{\beta\nu+1}\sqrt{k}} < m/2.$$

В силу последнего неравенства сравнения $Z\widetilde{M}^{(3)} \equiv C \bmod m$ представляют собой невырожденную (т.к. $k = \operatorname{rank} M^{(3)} = \operatorname{rank} \widetilde{M}^{(3)}$) систему линейных уравнений $Z\widetilde{M}^{(3)} = C$ от k неизвестных с k уравнениями. Решив эту систему, найдём значения z_i, $1 \le i \le k$, по которым легко найти $x_i = y_i 2^{\beta\nu} + z_i$. Зная x_1, в случае когда a и m взимнопросты, можно вычислить $x_0 = a^{(-1)} x_1 \bmod m$, где $a^{(-1)}$ такое целое число, для которого $a \cdot a^{(-1)} \equiv 1 \bmod m$. (Случай невзаимнопростых a и m встречается на практике редко, поскольку приводит к сокращению периода генератора.)
В [FHK+88] была доказана следующая теорема.

Теорема ([FHK+88]).

Предположим, что модуль m свободен от квадратов (не делится на квадрат любого целого числа). Пусть также даны число k и некоторая константа $\varepsilon > 0$. Тогда существует константа $c(\varepsilon, k)$, что для $m > c(\varepsilon, k)$ найдётся «исключительное множество» $E(m, \varepsilon, k)$ со свойствами:

1. $\left| E(m, \varepsilon, k) \right| \le m^{1-\varepsilon}$;
2. для любого $a \notin E(m, \varepsilon, k)$, порождающего x_i и y_i, значения x_i однозначно восстанавливаются по y_i описанным выше способом, если $\alpha\nu \ge (1/k + \varepsilon)\log_2 m + c_k$, где $c_k = \dfrac{k}{2} + (k-1)\log_2 3 + \dfrac{7}{2}\log_2 k + 2$.

Итак, для определения начального состояния x_0 при условии, что a и m взаинопросты, необходимо выполнить следующие действия.

1. Построить матрицу $M^{(3)}$ и применить к её вектор-столбцам LLL-алгоритм, получив, таким образом представление $\widetilde{M}^{(3)} = M^{(3)} U^{(3)}$.
2. Решить относительно вектора неизвестных $Z = (z_1, z_2, \dots, z_k)$ систему линейных алгебраических уравнений $Z\widetilde{M}^{(3)} = C$, где $C = (c_1, c_2, \dots, c_k) \equiv -2^{\beta\nu} Y\widetilde{M}^{(3)}$, $-m/2 < c_i \le m/2$.
3. Проверить, что найденные решения — целые и удовлетворяют соотношениям $0 \le z_i < 2^{\beta\nu}$.
4. Если проверяемые на предыдущем шаге условия не выполнены, вывести сообщение об ошибке: «НАЧАЛЬНОЕ СОСТОЯНИЕ НЕ НАЙДЕНО», завершить работу алгоритма.
5. Положить $x_1 = y_1 2^{\beta\nu} + z_1$ и вычислить $x_0 = a^{(-1)} x_1 \bmod m$, где $a^{(-1)}$ такое целое число, для которого $a \cdot a^{(-1)} \equiv 1 \bmod m$.

Примечание. *При использовании библиотеки NTL следует помнить, что LLL-процедуры из данной библиотеки работают с вектор-строками матриц. Поэтому перед использованием LLL-процедуры матрицу M следует транспонировать (равно как и найденную матрицу U после работы процедуры).*

Список источников

[FHK+88]A.M.Frieze, J.Hastad, R.Kannan, J.C.Lagarias, and A.Shamir, "Reconstructing Truncated Integer Variables Satisfying Linear Congruences", SIAM J. Comput., 17(2):262–280, 1988.
[JS98]A.Joux and J.Stern, "Lattice Reduction: A Toolbox for the Cryptanalyst", Journal of Cryptology, vol. 17, no. 3, pp. 161–185, 1998.