Постквантовая криптография. Критический обзор и дополнения

С.Ф. Кренделев

Новосибирский Государственный Университет

2016 г.

Cryptographic Algorithm	Type	Purpose	Impact from large-scale quantum computer
AES-256	Symmetric key	Encryption	Larger key sizes needed
SHA-256, SHA-3	Hash functions		Larger output needed
RSA	Public key	Signatures, key establishment	No longer secure
ECDSA, ECDH (Elliptic Curve Cryptography)	Public key	Signatures, key exchange	No longer secure
DSA (Finite Field Cryptography, Digital Signature Algorithm))	Public key	Signatures, key exchange	No longer secure

- Методы шифрования, используемые в постквантовой криптографии, основаны на задачах, про которые точно известно, что они являются NP-сложными. Основное требование для использования этих задач, это возможность эффективно использовать эти задачи для цифровых устройств.
- Два примера, в которых нужные требования удовлетворены.

Задача А. Найти решение системы линейных диофантовых уравнений в целых числах

$$\sum_{j=1}^{n} a_{ij} x_j = d_i \quad i=1,2,...,m$$

$$a_{ij}, d_i \in Z \quad i=1,2,...,m \quad j=1,2,...,n$$

При наличии ограничений. Например, $x_j \geq 0$ $j=1,2,...,n$ или $x_j \in \{0,1\}$ $j=1,2,...,n$ - задача целочисленного программирования. Особенно интересен для шифрования случай когда $m<n$ (сильно недоопределенная система линейных уравнений). В частности, если $m=1$ и $x_j \in \{0,1\}$ $j=1,2,...,n$ то эта задача есть задача об укладке рюкзака (knapsack problem) или subset-sum problem.

Задача Б. Найти решение системы полиномиальных уравнений в целых числах.

$$f_i(x_1,x_2,...,x_n)=0 \quad i=1,2,...,m$$

Здесь так же самый сложный случай когда $m<n$, то есть число уравнений меньше, чем число неизвестных.

Если система уравнений рассматривается над конечным полем (кольцом), то задача принадлежит, вообще говоря, к классу NP, и если $NP \neq P$, то это задача трудно решаема.

Однако если $R=Z$, $m<n$ и степени многочленов ≥ 3, то задача является алгоритмически неразрешимой в целых числах. Это следует из решения 10 проблемы Гильберта.

В принципе пост квантовая криптография подразделяется на четыре направления:

Code based cryptography

(основано на кодах исправляющих ошибки)

Multivariate, quadratic equations cryptography

(основано на многочленах в конечных полях)

Lattice based cryptography

(основано на теории решеток)

Hash-based cryptography

(основано на представлении хэш функций для больших данных)

Коды, исправляющие ошибки.

Речь идет о решении системы линейных уравнений над конечным полем F.

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} \quad (A)$$

Где $\mathbf{A}$ некоторая $n \times m$ $n > m$ матрица с элементами из поля F, вектор результат $\mathbf{y} \in F^n$ известен, вектор ошибок $\mathbf{e} \in F^n$, вектор сообщения $\mathbf{x} \in F^m$. Столбцы матрицы $\mathbf{A}$ называется кодовыми векторами. Задача заключается в том, чтоб из уравнения (А) определить вектор сообщения $\mathbf{x} \in F^m$, откуда вообще говоря, определяется вектор ошибок $\mathbf{e} \in F^n$.

Нетрудно заметить, что система уравнений (А) содержит n уравнений относительно $n + m$ неизвестных.

В этой ситуации рассматриваются конечные поля (кольца). В этом случае есть только алгебраическая структура, что означает, что можно сравнивать только на равенство двух элементов.

Открытым ключом шифрования являются - поле F, матрица $\mathbf{A}$ и количество ненулевых компонент в векторе ошибок $\mathbf{e} \in F^n$

Напомним, как решаются системы переопределенных уравнений. Все вычисления будем писать в координатной форме.

Пусть $\mathbf{H}$ $n \times k$ $k<n$ матрица над полем F. Будем представлять ее в двух видах.

Столбцами, есть k векторов $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k \in F^n$ тогда

$\mathbf{H} = (\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k)$ следовательно, уравнение

$\mathbf{Hx} = \mathbf{b}$ приобретает вид

$x_1\mathbf{u}_1 + x_2\mathbf{u}_2 + ... + x_k\mathbf{u}_k = \mathbf{b}$ считаем, что $\mathbf{x} = (x_1, x_2, ..., x_k)$.

Строками, есть n векторов $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n \in F^k$, в этом случае матрица имеет вид

$$\mathbf{H} = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ . \\ . \\ . \\ \mathbf{v}_n \end{pmatrix}$$ а уравнение $\mathbf{Hx} = \mathbf{b}$ приобретает вид,

$$\begin{aligned} (\mathbf{v}_1, \mathbf{x}) &= b_1 \\ (\mathbf{v}_2, \mathbf{x}) &= b_2 \\ &..... \\ &..... \\ &..... \\ (\mathbf{v}_n, \mathbf{x}) &= b_n \end{aligned}$$

считаем, что

$\mathbf{b} = (b_1, b_2, ..., b_n)$.

Пусть дан набор целых чисел $1 \le i_1 < i_2 < \ldots < i_{k+1} \le n$, рассмотрим подматрицу

$$\mathbf{H}[i_1, i_2, \ldots, i_{k+1}] = \begin{pmatrix} \mathbf{v}_{i_1} \\ \mathbf{v}_{i_2} \\ . \\ . \\ . \\ \mathbf{v}_{i_{k+1}} \end{pmatrix}$$ и свяжем с подматрицей систему уравнений

$$\mathbf{H}[i_1, i_2, \ldots, i_{k+1}]\mathbf{x} = \begin{pmatrix} b_{i_1} \\ b_{i_2} \\ . \\ . \\ . \\ b_{i_{k+1}} \end{pmatrix} \qquad (C)$$

Для всякого набора целых чисел $1 \le j_1 < j_2 < \ldots < j_k \le n$ обозначим

$p[j_1, j_2, \ldots, j_k] = \det \mathbf{H}[j_1, j_2, \ldots, j_k]$ - это определители соответствующих миноров матрицы $\mathbf{H}$.

В этом случае для разрешимости уравнения (С) необходимо и достаточно, что бы определитель расширенной матрицы равнялся 0

$$\det\begin{pmatrix} b_{i_1} & \mathbf{v}_{i_1} \\ b_{i_2} & \mathbf{v}_{i_2} \\ . & \\ . & \\ . & \\ b_{i_{k+1}} & \mathbf{v}_{i_{k+1}} \end{pmatrix} = 0$$ расписывая определитель по первому столбцу, получаем

$$b_{i_1} p[i_2,...,i_{k+1}] - b_{i_2} p[i_1,i_3,...,i_{k+1}] + ... + (-1)^{k+1} b_{i_{k+1}} p[i_1,i_2,...,i_k] = 0 \quad (*)$$

Первое очевидное следствие. Если рассмотреть вектор $\mathbf{w}$, у которого на позициях $i_1, i_2, ..., i_{k+1}$ стоят элементы

$p[i_2,...,i_{k+1}], -p[i_1,i_3,...,i_{k+1}], ..., (-1)^{k+1} p[i_1,i_2,...,i_k]$, а на остальных позициях стоят 0, то этот вектор ортогонален всем векторам столбцам $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k \in \mathrm{F}^n$. Набор таких векторов, продолженных нулями в теории кодов исправляющих ошибки называется проверочной матрицей.

Если выполнено уравнение (*) то это означает, что вектор

$b_{i_1}, b_{i_2}, ..., b_{i_{k+1}}$ является линейной комбинацией векторов $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k$ суженой на подпространство F^{k+1}. Отсюда следует, что имеет место векторное уравнение

$$\mathbf{v}_{i_1} p[i_2, ..., i_{k+1}] - \mathbf{v}_{i_2} p[i_1, i_3, ..., i_{k+1}] + ... + (-1)^{k+1} \mathbf{v}_{i_{k+1}} p[i_1, i_2, ..., i_k] = 0 \ (**)$$

Второе следствие заключается в том, что появляется возможность конструировать матрицы с предопределенными свойствами. Если условие (**) выполнено, то это означает что вектор $\mathbf{v}_{i_{k+1}}$ линейно зависим с векторами $\mathbf{v}_{i_1}, \mathbf{v}_{i_2}, ..., \mathbf{v}_{i_k}$, причем линейная зависимость связана с определителями соответствующих миноров. Можно выбирать эти миноры произвольным образом и затем строить вектор. Таким образом, если выбрать набор из k векторов, то можно выбрать оставшиеся вектора, таким образом, чтоб получились некоторые миноры с заданными свойствами. Это путь построения кодовых матриц, с заданным распределением миноров. Например, таким образом, устроены циклические коды.

Если выполнено уравнение (*) то это означает, что вектор

$b_{i_1}, b_{i_2}, ..., b_{i_{k+1}}$ является линейной комбинацией векторов $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_k$ суженой на подпространство F^{k+1}. Отсюда следует, что имеет место векторное уравнение

$$\mathbf{v}_{i_1} p[i_2, ..., i_{k+1}] - \mathbf{v}_{i_2} p[i_1, i_3, ..., i_{k+1}] + ... + (-1)^{k+1} \mathbf{v}_{i_{k+1}} p[i_1, i_2, ..., i_k] = 0 \ (**)$$

Второе следствие заключается в том, что появляется возможность конструировать матрицы с предопределенными свойствами. Если условие (**) выполнено, то это означает что вектор $\mathbf{v}_{i_{k+1}}$ линейно зависим с векторами $\mathbf{v}_{i_1}, \mathbf{v}_{i_2}, ..., \mathbf{v}_{i_k}$, причем линейная зависимость связана с определителями соответствующих миноров. Можно выбирать эти миноры произвольным образом и затем строить вектор. Таким образом, если выбрать набор из k векторов, то можно выбрать оставшиеся вектора, таким образом, чтоб получились некоторые миноры с заданными свойствами. Это путь построения кодовых матриц, с заданным распределением миноров. Например, таким образом, устроены циклические коды.

Выбирать произвольно значения всех миноров не получится, поскольку между минорами есть связь.

Эта связь называется квадратичным соотношением между минорами и выглядит так

$$p[i_1, i_2, \ldots, i_k]\, p[j_1, j_2, \ldots, j_k] =$$

$$\sum_{t=1}^{k} p[i_1, i_2, \ldots, i_{s-1}, j_t, i_{s-1}, \ldots, i_k]\, p[j_1, j_2, \ldots, j_{t-1}, i_s, j_{t+1}, \ldots, j_k]$$

Из квадратичных соотношений следует, что если известно значение некоторых миноров, то можно найти значение всех миноров не вычисляя определителей.

Таким образом, есть возможность генерировать кодовую матрицу и все возможные проверочные условия. В теории кодов исправляющих ошибки обычно используются два варианта линейные коды и циклические коды, причем они как правило не используют полный набор проверочных условий.

Атака на систему МакЭлиса.

В качестве матрицы $\mathbf{H}$ выбирается матрица размера 1024×524 над полем Z_2, исправляющая 50 ошибок. После шифрования получается вектор $\mathbf{b}$, $\mathbf{b} = \mathbf{Hx} + \mathbf{e}$ причем вектор ошибок $\mathbf{e}$ содержит не более 50 единиц, $\mathbf{x}$ - вектор сообщения.

Стандартная атака заключается в том, что выбирается произвольная квадратная подматрица $\mathbf{Y}$ матрицы $\mathbf{H}$. Если ее определитель 0 то выбирается другая матрица, если 1 то решается уравнение $\mathbf{Yx} = \mathbf{d}$, $\mathbf{d}$ - сужение вектора $\mathbf{b}$ на соответствующее подпространство. Если вектор $\mathbf{b} - \mathbf{Hx}$ содержит меньше или равно 50 единиц, то $\mathbf{x}$ - искомое сообщение.

Если не так, то выбираем следующую подматрицу.
Согласно выше сказанному, этот процесс можно существенно ускорить.

Прежде всего, начиная с некоторого количества полученных определителей миноров, следующие миноры можно вычислять без определителей, используя квадратичное соотношение между минорами. Кроме того начиная с некоторого количества определителей можно вычислять условия разрешимости. Нетрудно заметить, что эта схема легко распараллеливается.

Мораль, хорошо бы разработать схему, в которой все миноры имели определитель 0.

Как бы я реализовывал шифрование типа кодов исправляющих ошибки.

Прямая схема.

Проще всего объяснить на игрушечном примере. Пусть зафиксировано некоторое число p. Рассмотрим матрицу $\mathbf{T}$ размера 3×3 обратимую по модулю p. Очевидно, что уравнение $\mathbf{b}=\mathbf{Tx}$ разрешимо при любой правой части. Перепишем это уравнение в виде

$$x_1\mathbf{u}_1+x_2\mathbf{u}_2+x_3\mathbf{u}_3=\mathbf{b}$$

Отметим, что это уравнение благополучно решается. Теперь добавим к векторам некоторое количество переменных. Пусть добавили 2.

Введем вектора

$$\mathbf{w}_1 = \begin{pmatrix} \mathbf{u}_1 \\ a_1 \\ b_1 \end{pmatrix}; \mathbf{w}_2 = \begin{pmatrix} \mathbf{u}_2 \\ a_2 \\ b_2 \end{pmatrix}; \mathbf{w}_3 = \begin{pmatrix} \mathbf{u}_3 \\ a_3 \\ b_3 \end{pmatrix}; \mathbf{w}_4 = \begin{pmatrix} \mathbf{0} \\ a_4 \\ b_4 \end{pmatrix}; ...; \mathbf{w}_n = \begin{pmatrix} \mathbf{0} \\ a_n \\ b_n \end{pmatrix}$$

Лучше было бы написать вот в таком виде

$$\mathbf{w}_1 = \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{s}_1 \end{pmatrix}; \mathbf{w}_2 = \begin{pmatrix} \mathbf{u}_2 \\ \mathbf{s}_2 \end{pmatrix}; \mathbf{w}_3 = \begin{pmatrix} \mathbf{u}_3 \\ \mathbf{s}_3 \end{pmatrix}; \mathbf{w}_4 = \begin{pmatrix} \mathbf{0} \\ \mathbf{s}_4 \end{pmatrix}; ...; \mathbf{w}_n = \begin{pmatrix} \mathbf{0} \\ \mathbf{s}_n \end{pmatrix}$$

Где $\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_n$ некоторые вектора (не обязательно размерности 2, как в данном случае). Для секретности необходимо, что бы ранг набора векторов был меньше чем 5). Новая система имеет вид

$$x_1\mathbf{w}_1 + x_2\mathbf{w}_2 + x_3\mathbf{w}_3 + x_4\mathbf{w}_4 + ... + x_n\mathbf{w}_n = \mathbf{d}$$

Очевидно, что всегда из этой системы находится первые три компоненты.

Выберем 5×5 матрицу $\mathbf{H}$ обратимую по модулю p. Согласно этой матрице построим новые вектора $\mathbf{v}_i = \mathbf{H}\mathbf{w}_i$ i=1,2,...,n. Все вычисления происходят по модулю p. Сделаем перестановку векторов, и рассмотрим систему уравнений

$$x_1\mathbf{v}_1 + x_2\mathbf{v}_2 + x_3\mathbf{v}_3 + x_4\mathbf{v}_4 + ... + x_n\mathbf{v}_n = \mathbf{e} \quad \text{(E 1)}$$

Набор $\mathbf{v}_i$ i=1,2,...,n является открытым ключом. Модуль p не передается. Передается только ограничение на количество битов на символ. Ранг системы не максимальный. Злоумышленник в лучшем случае знает набор векторов $\mathbf{v}_i = \mathbf{H}\mathbf{w}_i$ i=1,2,...,n и выходной вектор $\mathbf{e}$. Тем самым система уравнений (E 1) является системой уравнений в целых числах с ограничениями, следовательно, является задачей класса NP.

Из этой системы уравнений, можно определить три компоненты. Вопрос в том, как сообщить в открытом ключе, где они находятся?
Предлагается следующая схема. Предположим, что заданы три числа k_1, k_2, k_3 таких, что $k_1+k_2+k_3=n$. Символы x_i i=1,2,...,n разбиваются на три группы

$$x_1, x_2, \ldots, x_{k_1}$$

$$x_{k_1+1}, x_{k_1+2}, \ldots, x_{k_1+k_2}$$

$$x_{k_1+k_2+1}, x_{k_1+k_2+2}, \ldots, x_{k_1+k_2+k_3=n}$$

Сортировка осуществляется так, что в каждой группе ровно один определяемый символ, и на какой позиции он стоит, хранится в секретном ключе. Предположим, что шифруются числа 12 битные. Это значит, что $p>4096$. В открытом ключе сообщается набор битов, которые игнорируются. Например, биты 2,7,9,10 это означает, что в реальности передаются данные размером 8 бит. На указанные позиции можно биты ставить случайным образом .

Таким образом, каждая группа кодируется так, что элементы из одной группы совпадают во всех битах кроме 2,7,9,10 где можно ставить все что угодно. Тем самым, реальная позиция, на которой находится зашифрованный символ, сохраняется в тайне. В связи с тем, что при умножениях и сложениях происходит перенос, этот набор бит не приносит дополнительную информацию.

Дешифрование. Получен вектор $\mathbf{e}$. Вычисляем вектор $\mathbf{H}^{-1}\mathbf{e}$ по модулю p, согласно конструкции получаем вектор $\mathbf{d}$. От вектора $\mathbf{d}$ берем первые три компоненты, получаем вектор $\mathbf{b}$. Решаем уравнение $x_1\mathbf{u}_1 + x_2\mathbf{u}_2 + x_3\mathbf{u}_3 = \mathbf{b}$.

Полученные решения переставляем согласно перестановке. Отсюда получаем расшифрованный текст.

Двойственная схема.

Рассмотрим однородную систему уравнений (Е 1)

$x_1\mathbf{v}_1 + x_2\mathbf{v}_2 + x_3\mathbf{v}_3 + x_4\mathbf{v}_4 + \ldots + x_n\mathbf{v}_n = \mathbf{0}$ (Е 2)

Вообще говоря, эта система имеет (в данном примере) не меньше чем $n-3$ решений. Оформим решение данного уравнения как вектора строки.

$\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_r$ r<n-3. Системе сопоставим вектора строки $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_5$. Условие того что вектора являются решением однородной системы запишем как условие ортогональности. $(\mathbf{s}_i, \mathbf{u}_j) = (\mathbf{u}_j, \mathbf{s}_i) = 0 \bmod(p)$ i=1,2,...,r;j=1,2,...,5.

Для усиления системы шифрования, необходимо сделать так, чтоб система векторов из ядра уравнения (Е 2) не имела максимального ранга. Это достигается следующим образом. Строятся вектора $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_s$ (n>s>r), по правилу $\mathbf{w}_i = \sum_{j=1}^{r} \lambda_j \mathbf{s}_j$ i=1,2,...,s. Тем самым все миноры матрицы, для которой вектора $\mathbf{w}_i$ являются столбцами, обращаются в 0. Набор векторов $\mathbf{w}_i$ i=1,2,...,s является открытым ключом. Естественно компоненты векторов переставляются.

Шифрование заключается в том, что выбираются произвольные числа $\mu_1, \mu_2, ..., \mu_s$, согласно правилам из реализации **Прямая схема** строится вектор сообщения $\mathbf{e}$, на выход отправляется вектор

$$\mathbf{b} = \sum_{j=1}^{s} \mu_j \mathbf{w}_j + \mathbf{e}$$

Дешифрование заключается в том, что надо вектор $\mathbf{b}$ надо скалярно умножить на вектора $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_5$ и свести задачу к задаче из **Прямой схемы.**

Замечание. Вовсе не обязательно шифровать 3 числа по этой схеме. Можно и 2. Это позволяет усилить стойкость шифрования.

Multivariate Cryptography

В этой части рассматривается вариант построения криптографической системы основанной на решении системы полиномиальной системы вида

$$y_1 = f_1(x_1, x_2, ..., x_n)$$
$$y_2 = f_2(x_1, x_2, ..., x_n)$$
..... (M),
.....
$$y_m = f_m(x_1, x_2, ..., x_n)$$

Где $f_i(x_1, x_2, ..., x_n)$ i=1,2,...,m многочлены от n переменных, которые принадлежат некоторому кольцу R .

Если R конечное поле, то задача (M) принадлежит, вообще говоря, к классу NP, и если $NP \neq P$, то это задача трудно решаема.

Однако если R=Z, m<n и степени многочленов ≥ 3, то задача (M) является алгоритмически неразрешимой в целых числах. Это следует из решения 10 проблемы Гильберта.

Как строить систему полиномиальных уравнений (M), которые можно эффективно решать. В постквантовой криптографии фактически используются два подхода, которые будут описаны.

Треугольные отображения.

Этот вариант предложил Диффи в 1986 г. Там фактически ничего нового с точки зрения математики нет, поскольку все эти конструкции были известны в математике и связаны с проблемой Якобиана. Очень полный обзор на эту тему был опубликован в 1982 г. Где описаны результаты за 50 лет на эту тему. Треугольные преобразования (De Jonquières groups) образуют подгруппу группы Луиджи Кремоны.(1863г.) в шифровании этот вариант – сети Фейстеля.

Пусть задано некоторое кольцо R и модуль над кольцом R^n. Элементы модуля R^n будем записывать как вектора $(x_1, x_2, ..., x_n)$ $x_i \in R$ $i = 1, 2, ..., n$.

Треугольные преобразования записывают в виде.

$$y_1 = x_1 + f_1(x_2, ..., x_n)$$
$$y_2 = x_2 + f_2(x_3, ..., x_n)$$
$$.....$$ верхнетреугольное.
$$y_{n-1} = x_2 + f_n(x_n)$$
$$y_n = x_n$$

$$y_1 = x_1$$
$$y_2 = x_2 + g_1(x_1)$$
$$y_3 = x_3 + g_2(x_1, x_2)$$ нижнетреугольное.
$$.....$$
$$y_n = x_n + g_n(x_1, x_2, ..., x_{n-1})$$

Функции f_i, g_i могут быть абсолютно произвольными, обычно это многочлены. Данные преобразования обратимы, следовательно, любые суперпозиции этих преобразований также обратимы.

Построение полиномиальных отображений в конечных полях.

Пусть задано некоторое кольцо R и модуль над кольцом R^n. Элементы модуля R^n будем записывать как вектора
$\mathbf{u} = (x_1, x_2, ..., x_n)$ $x_i \in R$ $i = 1, 2, ..., n$.

Предположим, что в модуле R^n задан некоторый базис $\mathbf{e}_0, \mathbf{e}_1, ..., \mathbf{e}_{n-1}$ (любой элемент из R^n можно представить в виде линейной комбинации базисных векторов). Снабдим модуль R^n, структурой линейной алгебры определив умножение базисных элементов по правилу

$$\mathbf{e}_i \times \mathbf{e}_j = \sum_{k=0}^{n-1} \gamma_{ijk} \mathbf{e}_k \quad \text{i,j=0,1,...,n-1} \quad \gamma_{ijk} \in R$$

В зависимости от свойств структурных констант γ_{ijk}, линейная алгебра может быть ассоциативной, коммутативной и иметь единичный элемент. Если структурные константы определяют коммутативную, ассоциативную линейную алгебру с единицей, то на векторах определено вычисление любого многочлена.

Для построения полей Галуа необходимо зафиксировать структурные константы и кольцо R. Фиксируем простое число p. Положим $R = Z_p$, тогда $R^n = Z_p^n$. Отметим, что Z_p - поле. Выберем произвольный базис в векторном пространстве Z_p^n, который обозначим $\mathbf{e}_0, \mathbf{e}_1, ..., \mathbf{e}_{n-1}$. Введем в векторном пространстве структуру линейной алгебры

$\mathbf{e}_0$ - единичный элемент

$\mathbf{e}_1$ - порождающий элемент

$\mathbf{e}_1 \times \mathbf{e}_1 = \mathbf{e}_2$

$\mathbf{e}_1 \times \mathbf{e}_2 = \mathbf{e}_3$

$\mathbf{e}_1 \times \mathbf{e}_3 = \mathbf{e}_4$

.....

$\mathbf{e}_1 \times \mathbf{e}_{n-2} = \mathbf{e}_{n-1}$

$\mathbf{e}_1 \times \mathbf{e}_{n-1} = \mathbf{e}_n = \lambda_0 \mathbf{e}_0 + \lambda_1 \mathbf{e}_1 + \lambda_2 \mathbf{e}_2 + ... + \lambda_{n-1} \mathbf{e}_{n-1} \quad \lambda_i \in Z_p$ i=0,1,...,n-1

Это другое представление факторкольца $Z_p[x]/(x^n + r(x))$. Если $x^n + r(x)$ неприводимый многочлен над полем Z_p, то полученная линейная алгебра является конечным полем.

Всякий элемент поля Z_p^n можно рассматривать как вектор $\boldsymbol{\eta} \in Z_p^n$. Тем самым его можно разложить по базису $\mathbf{e}_0, \mathbf{e}_1, ..., \mathbf{e}_{n-1}$. Это означает, что

$$\boldsymbol{\eta} = \mu_0 \mathbf{e}_0 + \mu_1 \mathbf{e}_1 + ... + \mu_{n-1} \mathbf{e}_{n-1} \quad \mu_i \in Z_p \quad \text{i=0,1,2,...,n-1}$$

Отсюда следует, что есть возможность вычислять любой многочлен с коэффициентами из поля Z_p от произвольного вектора. Другое обозначение для полученного поля $GF(p^n)$

В качестве стандартного базиса можно брать любой набор линейно независимых векторов над полем Z_p. Таким образом, есть произвол, связанный с тем, что в первых можно брать произвольный неприводимый многочлен для некоторого расширения, во вторых выбирать произвольный базис.

Следующий шаг заключается в том, что бы построить полиномиальное взаимно однозначное отображение (перестановочное отображение). В принципе можно любую перестановку в конечных полях реализовать в виде полиномиального отображения, однако это ведет к недопустимому увеличению размеров открытого ключа. В принципе можно использовать многочлены Диксона, но это весьма специфический класс отображений. В некоторых работах такой вариант предлагается.

Основной вариант заключается в том, что бы действовать как в RSA и возводить в подходящую степень. В данном случае допустимая степень d должна удовлетворять условию $\gcd(d,p^n-1)=1$. Здесь опять необходимо потребовать, что бы число d было достаточно маленьким.

В стандартных приложениях **Multivariate Cryptography** выбирают в качестве p простое число 2. Следовательно, рассматривается поле Галуа вида $GF(2^n)$. Для такого сорта полей имеют место два полезных обстоятельства.

1. Возведение в квадрат является линейной операцией. Это означает, что для всякой пары векторов $\mathbf{u}, \mathbf{v} \in GF(2^n)$, и пары элементов $x, y \in Z_2$ имеет место

$$(\mathbf{u}x + \mathbf{v}y)^2 = \mathbf{u}^2 x^2 + 2\mathbf{uv}xy + \mathbf{v}^2 y^2 = \mathbf{u}^2 x + \mathbf{v}^2 y$$

2. Если $\gcd(2^k + 1, 2^n - 1) = 1$, то возведение в степень $2^k + 1$ суть взаимно однозначное квадратичное отображение.

Пример $2^5 = 32, 2^5 - 1 = 31$ степени, которые дают квадратичные многочлены суть $3, 5, 9, 17$.

$2^8 = 256$, $2^8 - 1 = 255 = 3 \times 5 \times 17$ здесь нет подходящих степеней.

Как использовать Multivariate Cryptography?

Принцип использования такой. Сначала строится некоторая линейная система уравнений вида

$$\mathbf{u}_1 x_1 + \mathbf{u}_2 x_2 + ... + \mathbf{u}_n x_n = \mathbf{b} \ (\mathrm{C}) \quad \mathbf{u}_i \in Z_p^n$$

Теперь применяем методы построения полиномиальных отображений указанные выше и получаем систему уравнений вида

$$y_1 = f_1(x_1, x_2, ..., x_n)$$
$$y_2 = f_2(x_1, x_2, ..., x_n)$$
$$.....$$
$$.....$$
$$y_n = f_n(x_1, x_2, ..., x_n)$$

Отметим, что число уравнений совпадает с числом неизвестных. Набор многочленов $f_1(x_1, x_2, ..., x_n)$, $f_2(x_1, x_2, ..., x_n)$,..., $f_n(x_1, x_2, ..., x_n)$ и простое число p являются открытым ключом. Для шифрования вычисляются значения многочленов в точке, которая описывает передаваемое сообщение.

В результате получаем зашифрованное сообщение $(y_1, y_2, ..., y_n)$. Для дешифрования получаем правую часть уравнения (C), после чего решаем линейное уравнение (C).

Как строить систему полиномиальных уравнений (М), которые можно эффективно решать. Приведем простейший пример.

Зафиксируем некоторое число p. Будем считать, что возведение в степень 3 является взаимно однозначным отображением в кольце Z_p. Введем две матрицы размера $n \times n$, $\mathbf{H}, \mathbf{G}$. Шифруем n чисел $x_1, x_2, ..., x_n$. Ограничением будет количество бит на число (или некоторое число $q < p$). Составим n однородных многочленов степени 3.

$$\varphi_i(x_1, x_2, ..., x_n) = (\sum_{j=1}^{n} h_{ij} x_j)^3 \bmod(p) \text{ i=1,2,...,n}$$

Приведение по модулю проводится после возведения в степень. Теперь введем новые многочлены по правилу

$$\phi_i(x_1, x_2, ..., x_n) = \sum_{j=1}^{n} g_{ij} \varphi_j(x_1, x_2, ..., x_n) \bmod(p) \text{ i=1,2,...,n}$$

Этот шаг необходим для того, что бы замаскировать степени коэффициентов.

Открытый ключ – набор многочленов $\phi_i(x_1, x_2, ..., x_n)$ i=1,2,...,n, ограничение q. Передавать p не обязательно.

Проблемы и варианты Multivariate Cryptography

1. Система многочленов в открытом ключе обладает тем свойством, что число неизвестных совпадает с числом уравнений. Тем самым для вскрытия необходимо уметь решать системы полиномиальных уравнений над конечным полем. Известно, что у данной системы уравнений всегда существует единственное решение. Следовательно, для решения такого сорта уравнений можно использовать базисы Гребнера. В случае, когда используются конечные поля, не происходит роста коэффициентов многочлена и этот подход становится достаточно эффективным.

2. Для усиления стойкости предлагаются следующие варианты.

Добавление бессмысленных многочленов.

Фиксация некоторых переменных.

В любом таком варианте получается переопределенная система уравнений, которая, тем не менее, всегда имеет единственное решение, следовательно, метод базисов Гребнера работает и здесь.

3. Для усиления стойкости мы предлагаем не сообщать число p, а только сообщать ограничение на размер переменных. В этом случае решать систему уравнений придется исходя из того, что решения ищутся в целых числах. Это означает, что при использовании базисов Гребнера происходит не контролируемым образом.

Вывод такой – в качестве системы криптографии с открытым ключом данный подход недостаточно стойкий.

Для чего используется Multivariate Cryptography

Для того чтобы задача решения полиномиальных уравнений была сложно решаема необходимо, чтоб число уравнений было меньше чем число неизвестных. Однако такие уравнения имеют много решений и не позволяют расшифровать сообщение. Стандартный подход заключается в том, что часть уравнений убирается. В результате получается сильно недоопределенная система. Данная система уравнений трактуется как отображение из поля Z_p^n в модуль Z_p^m где $m<n$. Согласно определению это хэш-функция. Есть ограничения на число переменных n связанные с описанием многочленов, поэтому данный подход применяется только для цифровой подписи. В данном случае обратимость полиномиальных отображений не является обязательной, и используются другие специфические отображения.

Вариант системы с открытым ключом для Multivariate Cryptography

В примере, относящемся к шифрованию с открытым ключом, использующим коды исправляющие ошибки, была построена система линейных уравнений от n переменных и 5 уравнений.

$$x_1\mathbf{w}_1 + x_2\mathbf{w}_2 + x_3\mathbf{w}_3 + x_4\mathbf{w}_4 + \ldots + x_n\mathbf{w}_n = \mathbf{d}$$

Из этой системы однозначно восстанавливалось значение трех переменных.

Применим к этой системе методы, описанные выше, получим систему пяти полиномиальных уравнений от n неизвестных. Например, можно использовать пример с многочленами приведенный ранее. В результате получаем сильно неопределённое полиномиальное уравнение. Если не сообщать модуль p то полученная задача является алгоритмически неразрешимой.

Lattice-based cryptography.

Собственно теория кодов исправляющих ошибки и криптография на решетках отличаются только тем, что в первом случае рассматривается линейная алгебра на конечных полях (конечных кольцах), во втором случае линейная алгебра на бесконечных кольцах в которых есть структура упорядочивания, согласно которой есть алгоритма Евклида. Стандартные примеры использования это применение к построению криптосистем с открытым ключом.

Напомним, что такое решетки. Предположим, что в некотором модуле R^n над кольцом R, задано k векторов $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_k$. Решеткой назовем множество линейных комбинаций $\lambda_1 \mathbf{v}_1 + \lambda_2 \mathbf{v}_2 + ... + \lambda_k \mathbf{v}_k$, где коэффициенты λ_i i=1,2,...,k лежат в подкольце кольца R. Стандартное кольцо для приложений это кольцо целых чисел Z. Важно отметить, что это кольцо Евклидово.

Список систем шифрования

1. Рюкзачные схемы
2. GGH/HNF - Goldreich, Goldwasser and Halevi HNF - Эрмитова нормальная форма.
3. NTRU (1998г.).
4. LWE (R-LWE)

Речь опять идет о решении уравнения аналогичного уравнению из кодов исправляющих ошибки, только вместо конечных полей выступают кольца.

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} \text{ (Д)}$$

Где $\mathbf{A}$ некоторая $n \times m$ матрица с элементами из кольца Z, $\mathbf{y} \in Z^n$ вектор ошибок $\mathbf{e} \in Z^n$, вектор $\mathbf{x} \in Z^m$. Нетрудно заметить, что система уравнений (Д) содержит n уравнений относительно $n+m$ неизвестных. В зависимости от типа шифрования из этой системы можно определить либо $\mathbf{x}$, либо $\mathbf{e}$, либо оба вектора. В зависимости от того, какой вектор определяется, этот вектор объявляется вектором сообщения.

На самом деле слово решетки, связано с вопросом о разрешимости уравнения $\mathbf{A}\mathbf{x} = \mathbf{b}$ в целых числах. В этом случае условие того что ранг расширенной матрицы совпадает с рангом исходной является необходимым, но не является достаточным. В том случае, когда матрица $\mathbf{A}$ квадратная, то в силу правила Крамера любой минор расширенной матрицы должен делиться на определитель матрицы $\mathbf{A}$. Это является условием на разрешимость уравнения в целых числах. В том случае, когда матрица определяет переопределенную систему линейных уравнений, то это свойство должно выполняться для любой квадратной подматрицы, к этому надо добавить условие разрешимости в целом.

Речь идет о линейной алгебре над целыми числами. В этом случае аналогом векторных пространств являются модули над кольцами.

Есть два основных подхода.

Первый подход является аналогом решения уравнения $ax + y = b$ в целых числах при ограничении на число y. При условии, что $a > 1$ и $0 \le y < a$ данное уравнение легко решается. Сначала, находится $y = b \bmod(a)$, вычет b по модулю a, затем $x = (b - y) / \text{a}$. Обобщение этого подхода впервые предложено Гауссом в работе про бинарные квадратичные формы, откуда следовала теория вычетов для гауссовых чисел. Гауссовы числа – это комплексные числа с целыми коэффициентами. В принципе этот вариант можно обобщать на алгебраические числа.

Кстати, изучение вычетов на комплексные (алгебраические) числа может иметь приложения в построении системы симметричного шифрования для слабых вычислительных устройств.

Конструкция такая. Предположим, что есть три квадратных, целочисленных $n \times n$ матрицы $\mathbf{H, G, R}$ таких, что $\mathbf{HG} = p\mathbf{R}$ (p>1), $\mathbf{H}$ матрица с ненулевым определителем. Рассматривается уравнение $\mathbf{Gx} + \mathbf{e} = \mathbf{y}$. По смыслу уравнение $\mathbf{Gx} = \mathbf{y} - \mathbf{e}$ разрешимо в целых числах. Умножим уравнение слева на матрицу $\mathbf{H}$, получаем

$$\mathbf{HGx} + \mathbf{He} = p\mathbf{Rx} + \mathbf{He} = \mathbf{Hy}$$

Уравнение в целых числах разрешимо, следовательно, разрешимо по любому модулю. Возьмем по модулю p. Получаем уравнение $\mathbf{He} = \mathbf{Hy} \bmod(p)$. Отсюда возникает требование - $\mathbf{He} = \mathbf{He} \bmod(p)$. Это равенство покомпонентное, и дает ограничения на вектор ошибки $\mathbf{e}$. Если это ограничение выполнено, то находится вектор ошибки $\mathbf{e}$. Если матрица $\mathbf{G}$ имеет ненулевой определитель, то находится вектор $\mathbf{x}$ из уравнения

$$\mathbf{Gx} = \mathbf{y} - \mathbf{e}$$

Основная проблема – как в простых терминах сформулировать условие на вектор ошибки $\mathbf{He} = \mathbf{He} \bmod(p)$. Это необходимо для формирования открытого ключа.

Как это реализовано в NTRU. Все матрицы $\mathbf{H,G,R}$ циркулянты. Это означает, что все они попарно коммутируют. Матрица $\mathbf{H}$ имеет ненулевой определитель p (p>1). $\mathbf{G} = ad(\mathbf{H})\mathbf{R}$ здесь $ad(\mathbf{H})$ алгебраическое дополнение матрицы $\mathbf{H}$, $\mathbf{R}$ - матрица с определителем 0. Так как $\mathbf{H}ad(\mathbf{H}) = p\mathbf{E}$ ($\mathbf{E}$-единичная матрица), то $\mathbf{HG} = p\mathbf{R}$. Далее действуем согласно указаниям выше. В реальном NTRU, фигурируют модулярные методы, но сути дела это никак не меняет. В этом подходе вектор $\mathbf{x}$ найти невозможно, поскольку матрица $\mathbf{R}$ необратима. Параметры матриц и вектора ошибок получены экспериментальным путем. Недостаток – иногда неправильно расшифровывает.

Реализация GGH/HNF. В качестве матрицы $\mathbf{G}$ выбирается произведение матриц $\mathbf{G} = \mathbf{TR}$, где $\mathbf{T}$ почти ортогональная матрица с определителем больше 1, $\mathbf{R}$ - унимодулярная. $\mathbf{H} = ad(\mathbf{T})$. Далее по образцу.

Не смотря на проблемы связанные с GGH, это шифрование можно использовать следующим образом. Пусть квадратная матрица $\mathbf{H}$ такая, что позволяет находить ошибки из некоторого диапазона. Другими словами, можно находить вектор ошибок из уравнения $\mathbf{Hx}+\mathbf{e}=\mathbf{y}$. Тогда рассмотрим аналог систем счисления $\mathbf{e}_0+\mathbf{He}_1+\mathbf{H}^2\mathbf{e}_2+\ldots+\mathbf{H}^n\mathbf{e}_n=\mathbf{y}$, где все вектора $\mathbf{e}_i$ i=0,1,...,n удовлетворяют ограничениям на вектор ошибок. Тогда все вектора ошибок по заданному вектору $\mathbf{y}$ находятся однозначно. Полученная система уравнений может быть замаскирована аналогично случаю для кодов исправляющих ошибки. Например

$\mathbf{H}=\begin{pmatrix}5 & 2\\ -1 & 3\end{pmatrix}$ допускает ошибки вида $(0,0),(0,1),(1,1),(-1,0),(0,-1),(-1,-1)$.

Тогда замаскированная система счисления выглядит так

$$\begin{pmatrix}34\\147\end{pmatrix}x_1+\begin{pmatrix}4\\9\end{pmatrix}x_2+\begin{pmatrix}12\\29\end{pmatrix}x_3+\begin{pmatrix}203\\458\end{pmatrix}x_4+\begin{pmatrix}2\\5\end{pmatrix}x_5+\begin{pmatrix}13\\30\end{pmatrix}x_6+\begin{pmatrix}53\\121\end{pmatrix}x_7+\begin{pmatrix}3\\7\end{pmatrix}x_8=\begin{pmatrix}y_1\\y_2\end{pmatrix}$$

С ограничением $x_i\in\{-1,0,1\}$. Полученная система сильно не доопределена, поэтому попадает в Задачу А. Пример не совсем удачный, однако шифрование можно усилить, если перейти к двойственной задаче. Это было описано ранее.

То обстоятельство, что кольцо целых чисел является Евклидовым, позволяет строить сильно не доопределенные системы уравнений, у которых переменные лежат в любом наперед заданном диапазоне. Это означает, что можно воспользоваться либо аналогом двойственной задачи, либо воспользоваться системой недоопределенных полиномиальных уравнений.

В случае, когда получена система полиномиальных недоопределенных уравнений, получаем алгоритмически неразрешимую задачу. Вот пример применения.

$$\begin{pmatrix}1\\1\end{pmatrix}x_0+\begin{pmatrix}3\\5\end{pmatrix}x_1+\begin{pmatrix}9\\25\end{pmatrix}x_2+\begin{pmatrix}27\\125\end{pmatrix}x_4=\begin{pmatrix}a\\b\end{pmatrix}$$ очевидно, что диапазон изменения переменных $0\le x_i<15$ i=0,1,2,3. Максимальное значение компонент вектора $\begin{pmatrix}a\\b\end{pmatrix}$ меньше 2340. Следовательно, можно сделать сильное модульное умножение по модулю 2351 (это простое число), а затем возвести в степень 3. Получится система двух уравнений третьей степени от 4-х переменных.

- Относительно LWE (RLWE) можно сказать, что это некоторый аналог двойственного к GGH подхода.
- Все что касается теории решеток, практически все сказанное, без изменений переноситься на Евклидовы кольца. В связи с тем, что нужно представление для этих объектов то в качестве Евклидова кольца выбирается кольцо многочленов с коэффициентами из некоторого поля.

Хэш-функции, основанные на решетках.

Ajtai предложил схему построения хэш-функции следующего вида.

Выбирается случайная целочисленная матрица $\mathbf{H}$ размера $m \times n$ (считается что $m < n$). Вектор $\mathbf{y} \in Z_d^n$. Хэш-функция $f(\mathbf{y}) : Z_d^n \to Z_p^m$ определяется по правилу $f(\mathbf{y}) = \mathbf{H}\mathbf{y} \bmod(p)$. Этот вариант, лучше представить в виде

$f(\mathbf{y}) = \sum_{i=1}^{n} \mathbf{h}_i y_i \mod(p)$ здесь $\mathbf{h}_i$ вектора столбцы матрицы $\mathbf{H}$, y_i компоненты вектора $\mathbf{y}$. Если использовать результаты из полиномиальных отображений в полях, то можно возвести в квадрат и получить результат аналогичный цифровой подписи в **Multivariate Cryptography.**

При таком представлении есть возможность генерировать вектора $\mathbf{h}_i$ на лету, что означает возможность не фиксировать n. Генерацию можно проводить способом аналогичным генерации векторов в code based подходе.

Другой приятной особенностью является то, что для вычисления хэш-функции не нужно добавлять данные для выравнивания.

Данная схема допускает нелинейные обобщения.

1. $f(\mathbf{y}) = \sum_{i=1}^{n} \mathbf{h}_i y_i y_{i+1} \bmod(p)$

2. При генерации векторов $\mathbf{h}_i$, начиная с $i = m+1$ вектор $\mathbf{h}_{i+1} = \sum_{j=1}^{m} \mathbf{h}_j y_{j-i}$.

То есть, на входе выбирается набор линейно независимых векторов $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_m$. Каждый следующий суть линейная комбинация этих векторов с предыдущими значениями данных для хеширования.

Протокол обмена ключами.

Приведем очень простой протокол обмена ключами.

Предположим, что два участника А, Б хотят совместно вычислить скалярное произведение векторов $(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} x_i y_i$ так, чтобы другой участник не знал ничего о векторе владельца. Вектор $\mathbf{x} = (x_1, x_2, ..., x_n)$ принадлежит участнику А, вектор $\mathbf{y} = (y_1, y_2, ..., y_n)$ принадлежит участнику Б. Считаем, что компоненты векторов принадлежат множеству целых чисел Z.

Для этого каждый из них делает следующее.

Участник А выбирает число k $k < n$, вектор $\mathbf{a} \in Z^k$, набор векторов $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n \in Z^k$ таких, что $x_i = (\mathbf{v}_i, \mathbf{a})$ i=1,2,...,n, и ранг матрицы образованной векторами $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n \in Z^k$ меньше чем k.

Участник Б выбирает число s s<n, вектор $\mathbf{b} \in Z^s$, набор векторов $\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n \in Z^s$ таких, что $y_i = (\mathbf{w}_i, \mathbf{b})$ i=1,2,...,n и ранг матрицы образованной векторами $\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n \in Z^s$ меньше чем s.

Такой выбор можно сделать всегда.

А посылает Б набор векторов $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n \in Z^k$

Б посылает А набор векторов $\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n \in Z^s$

А вычисляет вектор $\mathbf{e} = \sum_{i=1}^{n} x_i \mathbf{w}_i$ и отправляет Б

Б вычисляет вектор $\mathbf{d} = \sum_{i=1}^{n} y_i \mathbf{v}_i$ и оправляет А

А вычисляет число $(\mathbf{d}, \mathbf{a})$

Б вычисляет число $(\mathbf{e}, \mathbf{b})$

По построению $(\mathbf{d}, \mathbf{a}) = (\mathbf{e}, \mathbf{b})$, следовательно, оба участника имеют одинаковое число.

Из уравнений $\mathbf{e} = \sum_{i=1}^{n} x_i \mathbf{w}_i$ и $\mathbf{d} = \sum_{i=1}^{n} y_i \mathbf{v}_i$ с известными векторами $\mathbf{e}, \mathbf{d}, \mathbf{w}_i, \mathbf{v}_i$ i=1,2,...,n значения x_i, y_i i=1,2,...,n не определяются по построению.