Оглавление

Постановка задачи

В данной работе будет рассмотрен и описан алгоритм RQC, участник конкурса NIST (конкурс национального института стандартов и технологий) по выбору квантово-устойчивых криптоалгоритмов для стандартизации.

Описание алгоритма RQC

RQC – алгоритм, проходящий на конкурс NIST, по программе IND-CCA2, в категории "постквантовая схема шифрования с открытым ключом". Данный шифр – шифр с открытым ключом шифрования, обладающий нескольким привлекательными свойствами:

- Доказанное соответствие стандартам IND-CPA, предполагающее трудность декодирования синдромной проблемы на структурных кодах;

- Алгоритм декодирования детерменирован, поэтому коэффициент отказов при расшифровке (DFR) равен нулю;

- Он имеет более привлекательные параметры, нежели большинство предложений, базирующихся на основе Хемминга.

Общие определения

q – мощность простого числа p

$\mathbb{F}_q$ – конечное поле с q элементами

$\mathbb{F}_{q^m}$ – более общий случай конечного поля для любого целого положительного числа m с q элементами. Так же часто будет рассматриваться $\mathbb{F}_{q^m}$ как m-мерное векторное пространство над $\mathbb{F}_q$.

Пусть $P \in F_q[X]$ - многочлен степени n. Мы можем определить векторное пространство $\mathbb{F}_{q^m}^n$ с кольцом $E_q m[x]/\langle p \rangle$, где p обозначает идеал $\mathbb{F}_q m[x]$, порожденный p.

$$\psi : F_{q^m}^n \simeq F_{q^m}[x]/\langle P \rangle$$
$$(v_0, \ldots, v_{n-1}) \to \sum_{i=0}^{n-1} v_i X^i$$

Для $u_i, v \in F_m^n$ определим их произведение аналогично $F_{q^m}[X]/\langle P \rangle$: $w = uv \in F_{q^m}^n$ единственный вектор, такой, что $\psi(w) = \psi(u)\psi(v)$. Для того, чтобы облегчить формулу, в дальнейшем будем опускать символ Ψ.

С вектором $v \in F_{q^m}^n$ можно связать квадратную матрицу $n \times n$ с записями в $F_{q^m}^n$, соответствующий произведению на v. Действительно,

$$u \cdot v = u(X)v(X) (\mathrm{mod}\, P) = \sum_{i=0}^{n-1} u_i X^i v(X) (\mathrm{mod}\, P) = \sum_{i=0}^{n-1} u_i (X^i v(X) (\mathrm{mod}\, P) = (u_0, ..., u_{n-1}) \begin{pmatrix} v(X) \bmod P \\ Xv(X) \bmod P \\ \vdots \\ X^{n-1} v(X) \bmod P \end{pmatrix}$$

Такая матрица называется идеальной матрицей, порожденной v и P, или просто v, когда нет неоднозначности в выборе P.

Определение 1. Идеальная матрица

Пусть $P \in F_q[X]$- многочлен степени n и $v \in F_{q^m}^n$. Идеальная матрица, порожденная v, — это квадрат $n \times n$, обозначенная $\mathrm{I}M(v)$ и имеет форму:

$$\mathrm{I}M(v) = \begin{pmatrix} v \\ X_v (\mathrm{mod}\, P) \\ \vdots \\ X^{n-1} v (\mathrm{mod}\, P) \end{pmatrix}$$

Как следствие, произведение двух элементов $F_{q^m}[X]/P$ эквивалентно обычному векторно-матричному произведению:

$$u \cdot v = u\mathrm{I}M(v) = \mathrm{I}M(u)^T v = v \cdot u$$

Определение 2. Ранговая метрика над $\mathbb{F}_{q^m}^n$

Пусть $X = (x_1, ..., x_n) \in \mathbb{F}_{q^m}^n$ и $(\beta_1, ..., \beta_m) \in \mathbb{F}_{q^m}^n$, рассматриваемый, как m-мерное векторное пространство над $\mathbb{F}_{q^m}^n$. Каждая координата x_i ассоциируется с вектором $\mathbb{F}_{q^m}^n$ в этом базисе: $x_i = \sum_{i=1}^{m} x_{ij} \beta_i$. Матрица размерности $m \times n$ задается для x как:

$$\| x \| \stackrel{def}{=} RankM(x)$$

Ассоциативное расстояние $d(x,y)$ между элементами x и y в $\mathbb{F}_{q^m}^n$ определяется, как $d(x,y) = \| x - y \|$.

Определение 3. $\mathbb{F}_{q^m}^n$ **- линейный код**

$\mathbb{F}_{q^m}^n$ - лиGнейный код C размерности k и длины n - это подпространство размерности k из $\mathbb{F}_{q^m}^n$, дополненное метрикой ранга. Оно обозначается $[n,k]_{q^m}$

Такой же код C может быть представлен двумя эквивалентными способами:

- Порождающая матрица $G \in \mathbb{F}_{q^m}^{k \times n}$. Каждая строка G является элементом базиса C: $C = \{xG, x \in \mathbb{F}_{q^m}^n\}$
- Матрицей проверки на четность $H \in \mathbb{F}_{q^m}^{(n-k)\times n}$. Каждая строка H определяет уравнение проверки на четность, проверяемое элементами C:

$$C = \{x \in \mathbb{F}_{q^m}^n : Hx^T = 0\}$$

H_{v^T} называется синдромом v (относительно H).

Утверждается, что G (соответственно H) находится в систематической форме тогда и только тогда, когда она имеет вид $(I_k \mid A)(соответственно(I_{n-k} \mid B))$

Определение 4. Поддержка слова

Пусть $x = (x_1, ..., x_n) \in \mathbb{F}_{q^m}^n$. Поддержка E слова x, обозначаемое $Supp(x)$, является $\mathbb{F}_q$-подпространством $\mathbb{F}_{q^m}^n$ порожденным координатами x:

$$E = \langle x_1, ..., x \rangle \mathbb{F}_q$$

И имеется в итоге $\dim(E) = \| x \|$.

Количество поддержек размерности ω в $\mathbb{F}_{q^m}$ задается благодаря Гауссовскому коэффициенту:

$$\begin{bmatrix} m \\ n \end{bmatrix}_q = \prod_{i=0}^{w-1} \frac{q^m - q^i}{q^w - q^i}$$

Шифрование

Условные обозначения

$S_w^n(\mathbb{F}_{q^m}) = \{x \in \mathbb{F}_{q^m}^n : \| x \| = w\}$ - обозначает множество векторов длины n и рангового веса w над $\mathbb{F}_{q^m}^n$

$S_{1,w}^n(\mathbb{F}_{q^m}) = \{x \in \mathbb{F}_{q^m}^n : \| x \| = w, 1 \in Supp(x)\}$ - обозначает множество векторов длины n и рангового веса w, таких что их поддержка содержит 1

$S_{w_1,w_2}^{3n}(\mathbb{F}_{q^m}) = \{x = (x_1, x_2, x_3) \in \mathbb{F}_{q^m}^{3n} : \| (x_1, x_3 \| = w_1, \| x_2 \| = w_1 + w_2, \in Supp(x_1, x_3) \subset Supp(x_2)\}$ - обозначает множество векторов длины 3n с неоднородным ранговым весом (w_1, w_2) над $\mathbb{F}_{q^m}^n$.

Версия шифрования с открытым ключом (RQC.PKE)

- $Setup(1^\lambda)$: генерирует и выдает глобальные параметры $param = (n, k, \delta, w, w_1, w_2, P)$ где $P \in \mathbb{F}_q[X]$ - неприводимый многочлен степени n
- $KeyGen(param)$: образцы $h \xleftarrow{\$} \mathbb{F}_{q^m}^n$, $g \xleftarrow{\$} S_n^n(\mathbb{F}_{q^m})$ и $(x, y) \xleftarrow{\$} S_{1,w}^{2n}(\mathbb{F}_{q^m})$, вычисляет порождающую матрицу $G \in \mathbb{F}_{q^m}^{k \times n}$ из $G_g(n, k, m)$, устанавливает $pk = (g, h, s = x + h \cdot y(\text{mod\_}P))$ и возвращает (pk, sk)
- $Encrypt(pk, m, \theta)$: использует случайную θ для генерации $(r_1, e, r_2) \xleftarrow{\$} S_{(w_1,w_2)}^{3n}(\mathbb{F}_{q^m})$, устанавливает $u = r_1 + h \cdot r_2(\text{mod\_}P)$ и $v = mG + s + r_2 + e(\text{mod\_}P)$, возвращает $c = (u, v)$
- $Decrypt(sk, c)$: возвращает G_g. Декодирует $(v - u \cdot y(\text{mod}\, P))$

Схема 1. Схема шифрования RQC.PKE

Следует обратить внимание, что порождающая матрица G кода G_g общеизвестна, поэтому безопасность схемы и возможность расшифровки не зависят от знания используемого кода и коррекцией ошибок G_g.

Корректность. Корректность схемы шифрования явно зависит от возможности декодирования кода G_g. В частности, если предположить, что $G_g.Decode$ правильно декодирует $v - u \cdot y$, имеем:

$$Decrypt(sk, Encrypt(pk, m, \theta)) = m$$

Сбоя при расшифровке не будет, или, если быть более точным, вероятность того, что произойдет сбой равна нулю.

Версия шифрования KEM/DEM (RQC.KEM)

Пусть ε экземпляр криптосистемы RQC.PKE, описанной выше. Пусть G, H, K - хеш-функции.

- $Setup(1^{\lambda})$: как и в RQC.PKE, за исключением того, что пространство открытого текста имеет размер $k \times m \geq 256$, как того требует стандарт NIST
- $KeyGen(param)$: образцы $h \xleftarrow{\$} \mathbb{F}_{q^m}^n$, $g \xleftarrow{\$} S_n^n(\mathbb{F}_{q^m})$ и $(x, y) \xleftarrow{\$} S_{1,w}^{2n}(\mathbb{F}_{q^m})$, вычисляет порождающую матрицу $G \in \mathbb{F}_{q^m}^{k \times n}$ из $G_g(n, k, m)$, устанавливает $pk = (g, h, s = x + h \cdot y(\text{mod\_}P))$ и возвращает (pk, sk)
- $Encapsulate(pk)$: генерируется $m \xleftarrow{\$} \mathbb{F}_{q^m}^n$ (это будет служить основой для получения будущего ключа). Получить случайность $\theta \leftarrow G(m)$. Сгенерировать шифротекст $c \leftarrow (u, v) = \varepsilon$. $Encrypt(pk, m, \theta)$ и получить симметричный ключ $K \leftarrow \kappa(m, c)$. Пусть $d \leftarrow H(m)$, и отправить (c, d).
- $Decapsulate(sk, c, d)$: расшифровать $m' \leftarrow \varepsilon . Decrypt(sk, c)$,вычислить $\theta' \leftarrow G(m')$ и снова зашифровать m', чтобы получить $c \leftarrow \varepsilon . Encrypt(pk, m', \theta')$. Если $c \neq c'$ или $d \neq H(m')$, то прервать действия. В противном случае получить общий ключ $K \leftarrow K(m, c)$

Схема 2. Схема шифрования RQC.KEM

Следует обратить внимание, что несмотря на то, что NIST рекомендует использовать в качестве хеш-функции SHA512, преобразование будет опасным с точки зрения безопасности, если задать $G = H$. Поэтому лучше воспользоваться SHA3-512 для G и SHA512 для H.

Дешифрование

Алгоритм, используемый для декодирования. Пусть G_g обозначает код над $\mathbb{F}_{q^m}^n$ длины n и размерности k, порожденный вектором $g \in \mathbb{F}_{q^m}^n$. Алгоритм декодирования формируется следующим образом:

1. Найти решение (V, N) задачи $\mathrm{Re}construction(y, g, k, t)$:
   a. Шаг инициализации: вычисляются две пары q-полиномов (N_0, V_0) и (N_1, V_1), удовлетворяющих $V_0(y_i) = N_0(g_i)$ и $V_1(y_i) = N_1(g_i)$ для $1 \le i \le k$. Для этого N_0 определяется, как аннигиляторный q-полином по $g_1, ..., g_k$, N_1 определяется как q-полином, интерполирующий $y_1, ..., y_k$ на $g_1, ..., g_k$ при $V_0(X) = 0$ и $V_1(X) = 0$
   b. Шаг интерполяции: с каждой итерацией, q-степени пар (N_0, V_0) и (N_1, V_1) увеличиваются с помощью рекуррентного соотношения, гарантирующего, что если условия интерполяции $V(y_i) = N(g_i)$ для $1 \le i \le j$ удовлетворяются на шаге *j*, то они так же удовлетворяются на шаге *j*+1. Кроме того, данная конструкция гарантирует, что по крайней мере одна из пар удовлетворяет конечной степени $\deg_q(V) \le t$ и $\deg_q(N) \le k + t - 1$ когда q-полином инициализируется с вышеупомянутым шагом инициализации
2. Найти f, вычислив левое Евклидово деление N на V (остаток от деления);
3. Получить кодовое слово C путем вычисления f в g.

Схема 3. Схема алгоритма декодирования

Сложность декодирования с помощью алгоритма, описанного на схеме 3, составляет $O(n^2)$ операций в $\mathbb{F}_{q^m}$.

Примечание: Декодирование было реализовано с использованием оптимизации «Полином с более низкой степенью».

Выбор параметров

Таблица 1 Наборы параметров для RQC. Безопасность выражается в битах

Параметры криптосистемы RQC								
Экземпляр	q	m	n	k	w	w_1	w_2	Безопасность
RQC-1	2	127	113	3	7	7	6	128
RQC-2	2	151	149	5	8	8	8	192
RQC-3	2	181	179	3	9	9	2	256

Таблица 2 Полиномы, рассматриваемые для RQC. P – полином, используемый для определения $\mathbb{F}_{q^m}^n$ как $\mathbb{F}_{q^m}[x]/\langle P\rangle$. $П$ - полином, используемый для определения $\mathbb{F}_{q^m}$ как $\mathbb{F}_q[X]/\langle П\rangle$.

Экземпляр	P	П
RQC-1	$X^{113}+X^9+1$	$X^{127}+X+1$
RQC-2	$X^{149}+X^{10}+X^9+X^7+1$	$X^{151}+X^3+1$
RQC-3	$X^{179}+X^4+X^2+X+1$	$X^{181}+X^7+X^6+X+1$

Таблица 3 Размеры в байтах для RQC. Безопасность выражается в битах

Экземпляр	pk размер	sk размер	ct размер	ss размер	Безопасность
RQC-1	1834	40	3652	64	128
RQC-2	2853	40	5690	64	192
RQC-3	4090	40	8164	64	256

Анализ производительности

Платформа для бенчмарков.

Бенчмарки были выполнены на машине, которая имеет 16 ГБ памяти и процессором Intel® Core™ i7-7820X CPU @ 3,6 ГГц, для которого были отключены функции HyperThreading, Turbo Boost и SpeedStep. Схема была скомпилирована с gcc (версия 9.2.0) и использует библиотеку openssl (версия 1.1.1d) в качестве провайдера для SHA2. Для каждого набора параметров результаты были получены путем вычисления среднего значения из 1000 случайных экземпляров. Для того чтобы минимизировать влияние фоновых задач, выполняемых на эталонной платформе, каждый пример повторялся 100 раз и усреднялся.

Постоянное время.

Представленные реализации были реализованы в постоянном времени и, таким образом, время их работы не должно передавать никакой информации в отношении значимых данных. Например, такие умные данные включают секретные ключи, а также вес ошибки, которая должна быть декодирована кодом.

Оптимизированная реализация

Была представлена оптимизированная реализация, использующая инструкции AVX и CLMUL. Ее производительность на вышеупомянутой эталонной платформе описана в таблице 4. При компиляции использовались следующие флаги оптимизации: -O3 -flto -mavx2 -mpclmul -msse4.2 -maes.

Таблица 4 Миллионы процессорных циклов эталонной реализации RQC

Экземпляр	Генерация ключа	Инкапсуляция	Декапсуляция
RQC-128	0.37	0.53	2.58
RQC-192	0.76	1.16	5.65
RQC-256	1.15	1.71	9.35

Безопасность

Чтобы доказать безопасность схемы, мы построим последовательность игр, переходящих от противника, получающего шифр сообщения m_0, к противнику, получающему шифрование сообщения m_1 и покажем, что если противнику удается отличить одно от другого, то можно построить симулятор, работающий примерно за то же время, который нарушает предположение DIRSD для 2-идеальных кодов или предположение DNHIRSD для 3-идеальных кодов.

Game G_1: Это и есть настоящая игра, которую алгоритмически можно представить следующим образом:

$\mathbf{Game}^{1}_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^{\lambda})$
2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4. $\mathbf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m_0}, \theta)$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
6. $\mathtt{RETURN}\ \mathbf{b}'$

Рисунок 1 Game1 - оригинальная игра

Game G_2: В этой игре мы начинаем с того, что забываем ключ дешифрования sk и берем s наугад, а затем честно продолжаем:

$\mathbf{Game}^2_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $\mathbf{s} \xleftarrow{\$} \mathbb{F}_{q^m}^n$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4. $\mathbf{c}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathbf{m_0}, \theta)$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
6. $\mathtt{RETURN}\ \mathbf{b}'$

Рисунок 2 Game2 - забыли ключ дешифрования и берем s наугад

Противник имеет доступ к pk и c*. Поскольку у него есть доступ к pk и функции Encrypt то все, что можно вычислить из pk и c*, можно вычислить и из pk. Более того, распределение $c*$ не зависит от игры, в которой мы находимся, и поэтому мы можем предположить, что единственным входом противника является pk. Предположим, что у него есть алгоритм D_λ, принимающий pk в качестве входных данных, который с преимуществом различает игры G_1 и G_2, для некоторого параметра безопасности λ. Тогда он также может построить алгоритм D_0 E, D_λ который решает задачу DIRSD (n, w) для параметров (n, w).

Game G_3: теперь, когда мы больше не знаем ключ дешифрования, мы можем начать генерировать случайные шифротексты. Итак, вместо того, чтобы выбирать правильно взвешенные r_1, r_2, e, симулятор теперь выбирает случайные векторы в полном пространстве

$\mathbf{Game}^3_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $\mathbf{s} \xleftarrow{\$} \mathbb{F}_{q^m}^n$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4a. Use randomness θ to generate $\mathbf{e} \xleftarrow{\$} \mathbb{F}_{q^m}^n$, $\mathbf{r} = (\mathbf{r}_1, \mathbf{r}_2) \xleftarrow{\$} \mathbb{F}_{q^m}^{2n}$ uniformly at random
4b. $\mathbf{u} \leftarrow \mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r_2}$ and $\mathbf{v} \leftarrow \mathbf{m}_0\mathbf{G} + \mathbf{s} \cdot \mathbf{r}_2 + \mathbf{e}$
4c. $\mathbf{c}^* \leftarrow (\mathbf{u}, \mathbf{v})$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
6. $\mathtt{RETURN}\ \mathbf{b}'$

Рисунок 3 Game3 - забыли ключ дешифрования и берем векторы наугад

Заметим, что противник не в состоянии получить c* из pk, поскольку в зависимости от того в какой игре мы находимся c* генерируется по-разному. Таким образом, на вход для различных игр будет подаваться (pk, c*). Поскольку он должен взаимодействовать с

претендентом, как обычно, мы предполагаем, что он имеет два режима доступа FIND и GUESS для обработки сначала pk, а затем c*

Game G4: теперь мы зашифруем другой открытый текст. Мы выбрали r'_1, r'_2, e' случайным образом и установили $u = r'_1 + h \cdot r'_1$ и $v = m_1 G + s \cdot r'_2 + e'$

$\mathbf{Game}^4_{\mathcal{E},\mathcal{A}}(\lambda)$
1. $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$
2a. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{param})$ with $\mathsf{pk} = (\mathbf{g}, \mathbf{h}, \mathbf{s} = \mathbf{x} + \mathbf{h} \cdot \mathbf{y})$ and $\mathsf{sk} = (\mathbf{x}, \mathbf{y})$
2b. $\mathbf{s} \xleftarrow{\$} \mathbb{F}_{q^m}^n$
2c. $(\mathsf{pk}, \mathsf{sk}) \leftarrow ((\mathbf{g}, \mathbf{h}, \mathbf{s}), \mathbf{0})$
3. $(\mathbf{m}_0, \mathbf{m}_1) \leftarrow \mathcal{A}(\mathtt{FIND} : \mathsf{pk})$
4a. Use randomness θ to generate $\mathbf{e}' \xleftarrow{\$} \mathbb{F}_{q^m}^n$ and $\mathbf{r}' = (\mathbf{r}'_1, \mathbf{r}'_2) \xleftarrow{\$} \mathbb{F}_{q^m}^{2n}$
4b. $\mathbf{u} \leftarrow \mathbf{r}'_1 + \mathbf{h} \cdot \mathbf{r}'_2$ and $\mathbf{v} \leftarrow \mathbf{m}_1\mathbf{G} + \mathbf{s} \cdot \mathbf{r}'_2 + \mathbf{e}'$
4c. $\mathbf{c}^* \leftarrow (\mathbf{u}, \mathbf{v})$
5. $\mathbf{b}' \leftarrow \mathcal{A}(\mathtt{GUESS} : \mathbf{c}^*)$
6. $\mathtt{RETURN}\ \mathbf{b}'$

Рисунок 4 Game4 - Зашифровали другой открытый текст

Выходы игр Game3 и Game4 имеют одинаковое распределение, и поэтому эти две игры неразличимы с информационно-теоретической точки зрения. Действительно, для каждого набора (r, e) из игры Game3, приводящего к данному (u, v), существует один к одному отображение на пару (r', e'), которая приводит к Game4 в том же (u, v), а именно $r' = r$ и $e' = e + m_0 G - m_1 G$. Из этого следует, что равномерный выбор (r, e) в игре Game3 и равномерный выбор (r', e') в Game4 приводит к такому же распределению (u, v).

Game G5: в этой игре теперь выбираем r'_1, r'_2, e' с правильным весом.

Game G6: Теперь мы завершаем игру сменой открытого ключа на честно сгенерированный.

Игры Game4 и Game5 являются эквивалентами игр Game3 и Game5, за исключением того, что вместо m_0 используется m_1. Различительная черта между этими двумя играми нарушает предположение DNHIRSD. Аналогично игры Game5 и Game6 являются эквивалентами игр Game2 и Game1, и различительная черта между этими двумя играми нарушает предположение DIRSD.

В итоге получилось построить последовательность игр, позволяющих симулятору преобразовать шифротекст сообщения m_0 в шифротекст сообщения m_1. Следовательно, преимущество противника против эксперимента IND-CPA ограничено как:

$$Adv^{ind}_{\varepsilon,A}(\lambda) \leq 2(Adv^{DIRSD}(\lambda) + Adv^{DNHIRSD}(\lambda))$$

Известные атаки

Далее будут рассмотрены наиболее известные атаки на задачи DIRSD и DNHIRSD, на которых основан RQC.

Хотя эти проблемы являются решающими, лучшие атаки в текущем состоянии исследований основаны на решении их вычислительной версии. Существует два типа атак на эти проблемы:

- комбинаторные атаки, где цель состоит в том, чтобы найти поддержку ошибки или кодового слова;

- алгебраические атаки, когда противник пытается решить алгебраическую систему, например. используя вычисление базиса Грёбнера.

Эти атаки являются общими атаками на проблемы RSD и NHRSD, поскольку не существует известного улучшения, использующего идеальную структуру кодов.

Комбинаторные атаки

Лучшая комбинаторная атака для решения проблемы RSD для [sn, n]-кода над $\mathbb{F}_{q^m}$ с ошибкой ранга весом r имеет сложность:

$$O(((s-1)nm)^w q^{r\left\lceil\frac{(n+1)m}{sn}\right\rceil - m})$$

операций в $\mathbb{F}_q$, где w - экспонента сложности нахождения решения линейной системы.

Общая идея атаки заключается в адаптации атаки информационного декодирования для расстояния Хэмминга. Для ранговой метрики атакующий пытается угадать подпространство, которое содержит поддержку ошибки, а затем решает линейную систему, полученную из уравнений проверки четности, чтобы проверить, был ли выбор правильным.

Алгебраическая атака на RSD

Второй способ решения экземпляра RSD - записать его в виде системы уравнений, называемой моделированием, тогда, если решить эту систему, то есть найти одно решение, то это будет решением экземпляра RSD.

Грубо говоря, система будет состоять из $m\binom{n-k-1}{n}$ уравнений в $\binom{n}{r}$ переменных, которые являются максимальными минорами, эта система затем была решена путем вычисления ее базиса Грёбнера, другое моделирование позволяет решить ее непосредственно путем линеаризации. Более точное условие $m\binom{n-k-1}{n} \geq \binom{n}{r}$ когда она выполняется, она соответствует передетерминированному случаю, если если нет, то соответствует недоопределенному случаю.

Если условие выполняется, то решение может быть выполнено со следующими затратами:

$$O(m\binom{n-k-1}{n}\binom{n}{r}^{w-1})$$

Если условие не выполняется, затраты следующие:

$$O((B_b + C_b)A_b^{w-1})$$

Квантовое ускорение

На текущий момент не существует важного квантового ускорения для вышеупомянутых алгебраических атак.

Для комбинаторных атак квантовое ускорение легко поддается анализу. Небольшое обобщение алгоритма квантового поиска Гровера позволяет разделить в 2 раза экспоненциальную сложность атак. Таким образом, сложность квантовой комбинаторной атаки на проблему RSD составляет:

$$O(((s-1)nm)^w q^{\frac{1}{2}(r\left\lceil\frac{(n+1)m}{sn}\right\rceil)} - m)$$

Временные атаки

Эти атаки представляют теоретический интерес, тем не менее, они довольно непрактичны в реальных ситуациях, так как требуют огромного количество запросов к временному оракулу. Представленные эталонная и оптимизированная реализации реализованы таким образом, что не допускают утечки веса декодируемой ошибки, тем самым предотвращая вышеупомянутые атаки на тайминг.

Преимущества и недостатки RQC

Преимущества

Основными преимуществами RQC перед существующими криптосистемами на базе кодов являются:

- сведение его IND-CPA к хорошо изученной проблеме теории кодирования: проблеме синдрома декодирования;

- защищенность против атак, направленных на восстановление скрытой структуры используемого кода;

- он характеризуется нулевым коэффициентом неудач при расшифровке;

- он имеет более привлекательные параметры, чем большинство предложений на основе Хэмминга.

Нулевая частота отказов при дешифровании позволяет достичь строгого снижения для IND-CCA2 безопасности версии KEM-DEM.

Недостатки

Ранговая метрика имеет очень хорошие характеристики, но использование ранговой метрики для криптографических целей не очень старое (1991 год). Это может показаться ограничением, но все же в последние годы была проведена большая работа по пониманию присущей вычислительной сложности соответствующих проблем.. Комбинаторные атаки очень хорошо изучены, а недавние результаты позволяют получить четкое представление о сложности алгебраических атак.

Список используемых источников

1. Carlos Aguilar-Melchor, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Efficient encryption from random quasi-cyclic codes. IEEE Transactions on Information Theory, 64(5):3927–3943, 2018.

2. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography with constant input locality. In Alfred Menezes, editor, CRYPTO 2007, volume 4622 of LNCS, pages 92–110. Springer, Heidelberg, August 2007.

3. Nicolas Aragon, Philippe Gaborit, Adrien Hauteville, and Jean-Pierre Tillich. A new algorithm for solving the rank syndrome decoding problem. Vail, USA, 2018. IEEE.

4. Daniel Augot, Pierre Loidreau, and Gwezheneg Robert. Generalized Gabidulin codes over fields of any characteristic. Des. Codes Cryptogr., 86(8):1807–1848, Aug 2018. 12, 13, 14 35

5. Magali Bardet, Pierre Briaud, Maxime Bros, Philippe Gaborit, Vincent Neiger, Olivier Ruatta, and Jean-Pierre Tillich. An algebraic attack on rank metric code-based cryptosystems. In Advances in Cryptology - EUROCRYPT 2020 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020. Proceedings, 2020.

6. Magali Bardet, Maxime Bros, Daniel Cabarcas, Philippe Gaborit, Ray Perlner, Daniel Smith-Tone, Jean-Pierre Tillich, and Javier Verbel. Algebraic attacks for solving the rank decoding and minrank problems without Gröbner basis, 2020.

7. Slim Bettaieb, Loïc Bidoux, Philippe Gaborit, and Etienne Marcatel. Preventing timing attacks against RQC using constant time decoding of Gabidulin codes. In 10th International Conference on Post-Quantum Cryptography, PQCrypto 2019, Chongqing China, May 8-10, 2019.