Моделирование и анализ информационных систем. Т. 24, № 2 (2017), с. 239–252
Modeling and Analysis of Information Systems. Vol. 24, No 2 (2017), pp. 239–252

DOI: 10.18255/1818-1015-2017-2-239-252

УДК 517.9

Декодирование тензорного произведения MLD-кодов и приложения к кодовым криптосистемам

Деундяк В. М., Косолапов Ю. В., Лелюк Е. А.

получена 7 апреля 2017

Аннотация.

Для практического применения кодовой криптосистемы типа Мак-Элиса необходимо, чтобы используемый в основе криптосистемы код имел быстрый алгоритм декодирования. С другой стороны, используемый код должен быть таким, чтобы нахождение секретного ключа по известному открытому ключу было практически неосуществимо при относительно небольшом размере ключа. В связи с этим в настоящей работе предлагается в криптосистеме типа Мак-Элиса использовать тензорное произведение $C_1 \otimes C_2$ групповых MLD-кодов C_1 и C_2. Алгебраическая структура кода $C_1 \otimes C_2$ в общем случае отличается от структуры кодов C_1 и C_2, поэтому представляется возможным построение стойких криптосистем типа Мак-Элиса даже на основе кодов C_i, для которых известны успешные атаки на ключ. Однако на этом пути возникает проблема декодирования кода $C_1 \otimes C_2$. Основной результат настоящей работы – построение и обоснование набора необходимых для декодирования этого кода быстрых алгоритмов. Процесс построения декодера существенно опирается на групповые свойства кода $C_1 \otimes C_2$. В качестве приложения в работе построена криптосистема типа Мак-Элиса на коде $C_1 \otimes C_2$ и приводится оценка ее стойкости к атаке на ключ в предположении, что для кодовых криптосистем на кодах C_i возможна эффективная атака на ключ. Полученные результаты численно проиллюстрированы в случае, когда C_1, C_2 – коды Рида–Маллера–Бермана, для которых соответствующая кодовая криптосистема взломана Л. Миндером и А. Шокроллахи (2007 г.).

Ключевые слова: мажоритарный декодер, коды Рида–Маллера–Бермана, тензорное произведение кодов

Для цитирования: Деундяк В. М., Косолапов Ю. В., Лелюк Е. А., "Декодирование тензорного произведения MLD-кодов и приложения к кодовым криптосистемам", *Моделирование и анализ информационных систем*, **24**:2 (2017), 239–252.

Об авторах:
Деундяк Владимир Михайлович, orcid.org/0000-0001-8258-2419, канд. физ.-мат. наук, доцент,
ФГНУ НИИ "Спецвузавтоматика",
пер. Газетный, 51, г. Ростов-на-Дону, 344002 Россия,
Южный Федеральный Университет,
ул. Большая Садовая, 105/42, г. Ростов-на-Дону, 344006 Россия, e-mail: vl.deundyak@gmail.com,

Косолапов Юрий Владимирович, orcid.org/0000-0002-1491-524X, канд. техн. наук,
Южный Федеральный Университет,
ул. Большая Садовая, 105/42, г. Ростов-на-Дону, 344006 Россия, e-mail: itaim@mail.ru,

Лелюк Евгений Андреевич, orcid.org/0000-0001-6560-2561, магистрант,
Южный Федеральный Университет,
ул. Большая Садовая, 105/42, г. Ростов-на-Дону, 344006 Россия, e-mail: lelukevgeniy@mail.ru,

Введение

Стойкость применяемых в настоящее время на практике асимметричных криптосистем основана на сложности задач факторизации целых чисел или дискретного логарифмирования в конечной группе. Однако в [1] показано, что эти задачи могут быть решены за полиномиальное время на квантовом компьютере. Криптографические системы, в основе которых лежит применение помехоустойчивых кодов (далее — кодовые криптосистемы), рассматриваются в настоящее время как одна из альтернатив используемым в настоящее время асимметричным криптографическим системам [2]. Недостатком кодовых криптосистем является большой размер ключа. В частности, размер ключа для первой кодовой криптосистемы на основе кодов Гоппы, предложенной Робертом Мак-Элисом в [3], составляет порядка 65 Кбайт. Попытки уменьшить размер ключа за счет использования кодов, отличных от кодов Гоппы, не дали должного результата, так как предложенные системы оказались нестойкими. К нестойким относятся такие известные системы, как криптосистема Нидеррайтера [4], криптосистема Габидулина–Парамонова–Третьякова [5], криптосистема Сидельникова [6]. Для перечисленных криптосистем имеются эффективные структурные атаки, то есть атаки, направленные на нахождение подходящего секретного ключа по известному открытому ключу (см. [7]– [11]).

Представляется, что усилить стойкость кодовых криптосистем к структурным атакам возможно путем использования помехоустойчивых кодов, для которых, с одной стороны, имеется быстрый алгоритм декодирования, а с другой стороны, которые не обладают явно выраженной алгебраической структурой. Такой подход применен, например, в [12], где предлагается в криптосистеме типа Мак-Элиса использовать коды, индуцированные групповыми кодами. В настоящей работе приводится одно обобщение этого подхода: в качестве помехоустойчивого кода предлагается применять тензорное произведение $C_1 \otimes C_2$ двух групповых мажоритарно-декодируемых кодов C_1 и C_2 (MLD-кодов). Но на этом пути возникает задача декодирования кода $C_1 \otimes C_2$, решение которой в общем случае не известно даже тогда, когда известны эффективные декодеры для кодов C_1, C_2.

Целью настоящей работы является построение и обоснование набора необходимых для декодирования кода $C_1 \otimes C_2$ быстрых алгоритмов, когда C_1 и C_2 — групповые MLD-коды на группах $\mathcal{G}$ и $\mathcal{H}$ соответственно. Отметим, что в этом случае код $C_1 \otimes C_2$ является групповым кодом на прямом произведении $\mathcal{G} \times \mathcal{H}$. Процесс построения декодера существенно опирается на групповые свойства кода $C_1 \otimes C_2$ и на результаты работы [13], в которой построены алгоритмы декодирования для индуцированных групповых кодов. В качестве приложения построена криптосистема типа Мак-Элиса на коде $C_1 \otimes C_2$ и приводится теоретическая оценка ее стойкости к атаке на ключ в предположении, что для кодовых криптосистем на кодах C_i возможна эффективная атака на ключ. В работе полученные результаты о стойкости численно проиллюстрированы в случае, когда C_i — код Рида–Маллера–Бермана, определенный на аддитивной группе поля $\mathbb{F}_{2^{m_i}}$, $1 \leq m_i \leq 8$, для которого криптосистема типа Мак-Элиса взломана Л. Миндером и А. Шокроллахи (2007 г.).

Результаты работы представлены в первом и втором разделах. В первом разделе приводятся необходимые сведения о групповых MLD-кодах и далее строятся и обосновываются конструктивные алгоритмы для декодирования тензорного произ-

ведения таких кодов. Во втором разделе проводится анализ стойкости криптосистемы типа Мак-Элиса на коде $C_1 \otimes C_2$ к нахождению подходящего секретного ключа, если известна аналогичная структурная атака хотя бы для одного из кодов C_1 и C_2.

1. Тензорное произведение MLD-кодов

1.1. MLD-коды

Для натурального n символом $\underline{n}$ будем обозначать множество $\{1, ..., n\}$. Пусть V – векторное пространство над конечным полем $\mathbb{F}$. Зафиксируем в V базис B и символом (V, d_B) обозначим метрическое пространство V с метрикой Хэмминга d_B, построенной относительно базиса B. Для вектора $\mathbf{x}(\in V)$ множество базисных векторов, коэффициенты при которых в разложении $\mathbf{x} = \sum_{\mathbf{b}\in B} x_{\mathbf{b}}\mathbf{b}$ ненулевые, называется носителем вектора $\mathbf{x}$ относительно базиса B и обозначается $\mathrm{supp}_B(\mathbf{x})$; коэффициенты $x_{\mathbf{b}}$ будем называть значением $\mathbf{b}$-координаты вектора $\mathbf{x}$. Вес $w_B(\mathbf{x})$ вектора $\mathbf{x}$ определяется как $|\mathrm{supp}_B(\mathbf{x})|$. (Здесь и далее символом $|A|$ обозначается мощность множества A.) Всякое линейное подпространство C метрического пространства (V, d_B) называется линейным кодом. Размерность и длину кода будем обозначать соответственно $k(C)$ и $n(C)$, а минимальное кодовое расстояние кода C обозначим $\mathrm{dist}_B(C)$. Двойственный код к коду C обозначим $C^\perp$. Множество векторов $\mathcal{M}_{\mathbf{v}} = \{\mathbf{v}^{(1)}, ..., \mathbf{v}^{(r)}\}(\subset V)$ называется M-ортогональным вектору $\mathbf{v}(\in V)$, если $|\mathrm{supp}_B(\mathbf{v}^{(i)})| > |\mathrm{supp}_B(\mathbf{v})|$ для всех $i = 1, ..., r$ и

$$\forall\ i \neq j: \quad \mathrm{supp}_B(\mathbf{v}^{(i)}) \cap \mathrm{supp}_B(\mathbf{v}^{(j)}) = \mathrm{supp}_B(\mathbf{v}). \tag{1}$$

Пусть $\mathbf{c}(\in C)$ — кодовый вектор, $\mathbf{x} = \mathbf{c} + \mathbf{e}$ — принятый из канала вектор, $w_B(\mathbf{e}) \leq \lfloor(\mathrm{dist}_B(C) - 1)/2\rfloor$. Рассмотрим разложение $\sum_{\mathbf{b}\in B} e_{\mathbf{b}}\mathbf{b}$ вектора ошибок $\mathbf{e}$ по базису B. Если для $\mathbf{b}$-координаты существует *декодирующее дерево* $\mathrm{WB}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}}$, такое, что $\lfloor r_{\mathbf{b}}/2\rfloor \geq w_B(\mathbf{e})$, то значение $e_{\mathbf{b}}$ для $\mathbf{b}$-координаты вектора $\mathbf{e}$ находится однозначно с помощью мажоритарного декодера (см. [13], алгоритм 3 Decoder2). Декодирующим деревом $\mathrm{WB}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}} = \mathrm{WB}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}}[C]$ для $\mathbf{b}$-координаты здесь и далее будем называть в соответствии с [13] помеченное дерево с корнем $\mathbf{b}$, обладающее следующими свойствами:

1) множество вершин этого дерева состоит из $L_{\mathbf{b}} + 1$ уровня; корень с меткой $\mathbf{b}(\in B)$ находится на уровне 0, а листья – на уровне $L_{\mathbf{b}}$; метки вершин i-го уровня образуют набор V_i, $i = 0, ..., L_{\mathbf{b}}$;

2) листья дерева помечены элементами из $C^\perp$;

3) каждая вершина, не являющаяся листом, имеет не менее $r_{\mathbf{b}}(\in \mathbb{N})$ непосредственно следующих за ней вершин;

4) с меткой $\mathbf{p}$ каждой вершины дерева связывается числовое значение $l(\mathbf{p})(\in \mathbb{F})$ метки, вычисляемое в зависимости от значения принятого из канала вектора $\mathbf{x}$: для каждого листа $\mathbf{p}$ дерева значение $l(\mathbf{p})$ равно скалярному произведению $(\mathbf{p}, \mathbf{x})$ векторов $\mathbf{p}$ и $\mathbf{x}$, а для вершин на уровне $i(0 \leq i \leq L_{\mathbf{b}} - 1)$ значение $l(\mathbf{p})$ вычисляется в соответствии с построенным в [13] алгоритмом MajorVote (для полноты изложения алгоритм MajorVote приведен ниже);

5) метки вершин, непосредственно следующих из произвольной вершины $\mathbf{p}$, находящейся на уровне $i(0 \le i < L_{\mathbf{b}})$, образуют в совокупности множество $\mathcal{M}_{\mathbf{p}}$, M-ортогональное $\mathbf{p}$; символом $l[\mathcal{M}_{\mathbf{p}}]$ обозначается набор $(l(\mathbf{q}))_{\mathbf{q}\in\mathcal{M}_{\mathbf{p}}}$.

Исходные параметры: $\mathcal{A}$ – последовательность чисел из $\mathbb{F}$
Результат: элемент $v(\in \mathbb{F})$, который в последовательности $\mathcal{A}$ встречается наибольшее число раз
для каждого $a \in \mathbb{F}$ **выполнять**
  вычислить величину $\mathrm{count}(a)$, равную числу появления элемента a в последовательности $\mathcal{A}$
конец цикла
если *найдется только один* $a'(\in \mathbb{F})$, *что* $\mathrm{count}(a') \ge \lceil |\mathcal{A}|/2 \rceil$ **тогда**
  $v := a'$
иначе
  $v := 0$
конец условия
возвратить v

Алгоритм 1: MajorVote

Если для кода C существует такой набор

$$\mathcal{WB}(C) = \{\mathrm{WB}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}}\}_{\mathbf{b}\in B}, \tag{2}$$

для которого $\mathrm{dmaj}_B(C) = \min_{\mathbf{b}(\in B)}\{r_{\mathbf{b}}\} = \mathrm{dist}_B(C) - 1$, то код C называют MLD-кодом (Majority Logic Decodable). Заметим, что $\mathrm{dmaj}_B(C) \le \mathrm{dist}_B(C) - 1$, иначе в противном случае получили бы, что код может гарантированно исправлять более $\lfloor(\mathrm{dist}_B(C) - 1)/2\rfloor$ ошибок.

Построение набора $\mathcal{WB}(C)$ представляется в общем случае сложной задачей. С одной стороны, сложной представляется задача построения дерева для фиксированной координаты, а с другой стороны, деревья для разных координат строятся независимо. В то же время вторая задача решается просто для групповых кодов, если имеется декодирующее дерево хотя бы для одной координаты. Необходимые сведения о групповых кодах и построении множества $\mathcal{WB}(C)$ для группового кода C приводятся ниже.

1.2. Групповые MLD-коды

Пусть $\mathcal{G} = \{g_1 = \hat{1}, ..., g_{|\mathcal{G}|}\}$ — конечная группа с зафиксированным линейным порядком на множестве ее элементов, $\hat{1}$ – нейтральный элемент группы; зафиксированный порядок на группе будем обозначать $\mathrm{ord}(\mathcal{G})$. Рассмотрим групповую алгебру $\mathbb{F}\mathcal{G}$, элементами которой являются формальные суммы (функции):

$$\sum_{g\in\mathcal{G}} a_g g, a_g \in \mathbb{F}. \tag{3}$$

В конечномерной групповой алгебре $\mathbb{F}\mathcal{G}$ зафиксируем базис $B = B^{\mathbb{F}\mathcal{G}} := \{\mathbf{g} = \delta_g\}_{g\in\mathcal{G}}$, где $\delta_g = 1g$ – функция Дирака; $\mathbf{1} := 1\hat{1}$. Это позволяет рассматривать в $\mathbb{F}\mathcal{G}$ метрику Хэмминга d_B. Отметим, что в категории конечномерных линейных пространств

$\mathbb{F}\mathcal{G}$ и $\mathbb{F}^{|\mathcal{G}|}$ изоморфны; соответствующий изоморфизм обозначим $\nu_{\mathcal{G}}$. Также отметим, что произведение функций δ_x и δ_y в $\mathbb{F}\mathcal{G}$ равно $\delta_x\delta_y = \delta_{xy}$. Поэтому элементы групповой алгебры можно записывать в виде: $\sum_{g\in\mathcal{G}} a_g\delta_g, a_g \in \mathbb{F}$. Для удобства значение функции $\phi(\in \mathbb{F}\mathcal{G})$ в точке $g(\in \mathcal{G})$ будем обозначать $\phi(g)$.

В соответствии с [14], с. 39, всякий отличный от $\{\mathbf{0}\}$ *левый* идеал C в групповой алгебре $\mathbb{F}\mathcal{G}$ называется *групповым кодом* ($\mathbb{F}\mathcal{G}$–кодом) длины $n(C) = |\mathcal{G}|$. Идеал в групповой алгебре $\mathbb{F}\mathcal{G}$ является подпространством пространства функций $\mathbb{F}\mathcal{G}$, размерность $k(C)$ кода C — это размерность этого подпространства. Пусть $B^C = \{\epsilon_1, ..., \epsilon_{k(C)}\}(\subseteq \mathbb{F}\mathcal{G})$ — базис идеала C. Заметим, что порядок $\mathrm{ord}(\mathcal{G})$ индуцирует порядок на базисе $B^{\mathbb{F}\mathcal{G}}$ групповой алгебры $\mathbb{F}\mathcal{G}$, что позволяет выписать порождающую матрицу $G(C)$ группового кода C:

$$G(C) = \begin{pmatrix} \nu_{\mathcal{G}}(\epsilon_1) \\ \dots \\ \nu_{\mathcal{G}}(\epsilon_{k(C)}) \end{pmatrix} \tag{4}$$

Группа $\mathcal{G}$ действует слева на групповой алгебре $\mathbb{F}\mathcal{G}$ следующим естественным образом (см. [14], с. 32):

$$\mathcal{G} \times \mathbb{F}\mathcal{G} \ni (g, \phi = \sum_{h\in\mathcal{G}} \phi_h h) \mapsto \phi g^{-1} := \sum_{h\in\mathcal{G}} \phi_{hg^{-1}} h \in \mathbb{F}\mathcal{G}. \tag{5}$$

Отметим, что $C^\perp$ — также групповой код [14], т.е. левый идеал. В силу этого действие группы $\mathcal{G}$ по правилу (5) на элементах кода $C^\perp$ не выводит за код $C^\perp$. С другой стороны, группа $\mathcal{G}$ действует транзитивно на элементах из $B^{\mathbb{F}\mathcal{G}}$ и не нарушает M-ортогональности [13]. Это позволяет построить набор (2) по одному из декодирующих деревьев (соответствующие алгоритмы построены в [13]). В частности, если для базисной функции $\mathbf{1} = \delta_{\hat{1}} = 1\hat{1}$ удалось построить декодирующее дерево $\mathrm{WB}_{\mathbf{1},r_\mathbf{1},L_\mathbf{1}}$, то дерево $\mathrm{WB}_{\mathbf{g},r_\mathbf{g},L_\mathbf{g}}$ для базисной функции $\mathbf{g} = \delta_g = 1g$ может быть построено путем действия элементов $g^{-1}(\in \mathcal{G})$ на узлы дерева $\mathrm{WB}_{\mathbf{1},r_\mathbf{1},L_\mathbf{1}}$ по правилу (5). При этом $r_\mathbf{1} = r_\mathbf{g}$ и $L_\mathbf{1} = L_\mathbf{g}$ для всех $g \in \mathcal{G}$.

1.3. Тензорное произведение групповых кодов

Пусть $\mathcal{G} = \{g_1, ..., g_{|\mathcal{G}|}\}$, $\mathcal{H} = \{h_1, ..., h_{|\mathcal{H}|}\}$ — конечные группы с зафиксированными на них линейными порядками $\mathrm{ord}(\mathcal{G})$ и $\mathrm{ord}(\mathcal{H})$. В групповых алгебрах $\mathbb{F}\mathcal{G}$ и $\mathbb{F}\mathcal{H}$ зафиксируем базисы $B^{\mathbb{F}\mathcal{G}} = \{\delta_{g_1}, ..., \delta_{g_{|\mathcal{G}|}}\}$ и $B^{\mathbb{F}\mathcal{H}} = \{\delta_{h_1}, ..., \delta_{h_{|\mathcal{H}|}}\}$ соответственно. Рассмотрим тензорное произведение $\mathbb{F}\mathcal{G}\otimes_{\mathbb{F}}\mathbb{F}\mathcal{H}$ групповых алгебр $\mathbb{F}\mathcal{G}$ и $\mathbb{F}\mathcal{H}$ над полем $\mathbb{F}$ (см. [15], с.79). Отметим, что $\mathbb{F}\mathcal{G} \otimes_{\mathbb{F}} \mathbb{F}\mathcal{H} = \mathbb{F}(\mathcal{G} \times \mathcal{H})$.

В групповых алгебрах $\mathbb{F}\mathcal{G}$ и $\mathbb{F}\mathcal{H}$ рассмотрим групповые коды C_1 ($\subseteq \mathbb{F}\mathcal{G}$) и C_2 ($\subseteq \mathbb{F}\mathcal{H}$) с соответствующими базисами $B^{C_1} = \{\epsilon_1, ..., \epsilon_{k(C_1)}\}$ и $B^{C_2} = \{\phi_1, ..., \phi_{k(C_2)}\}$, где $\epsilon_i = \sum_{g\in\mathcal{G}} a_{i,g}\delta_g, a_{i,g} \in \mathbb{F}$ и $\phi_j = \sum_{h\in\mathcal{H}} b_{j,h}\delta_h, b_{j,h} \in \mathbb{F}$. Тензорным произведением кодов C_1 и C_2 будем называть код $C_1 \otimes C_2(\subseteq \mathbb{F}(\mathcal{G} \times \mathcal{H}))$ с базисом $B^{C_1\otimes C_2} = \{\epsilon_i \otimes \phi_j | i = 1, ..., k(C_1), j = 1, ..., k(C_2)\}$, где $(\epsilon_i \otimes \phi_j)(g, h) = \epsilon_i(g)\phi_j(h)$, $(g, h) \in \mathcal{G} \times \mathcal{H}$. Под тензорным произведением $A \otimes B$ матрицы $A = (a_{i,j})$ размера $(r \times s)$ и матрицы B

будем понимать, как обычно, матрицу вида:

$$A \otimes B = \begin{pmatrix} a_{1,1}B & \dots & a_{1,s}B \\ a_{2,1}B & \dots & a_{2,s}B \\ \dots & \dots & \dots \\ a_{r,1}B & \dots & a_{r,s}B \end{pmatrix}.$$

Тогда $G(C_1 \otimes C_2) = G(C_1) \otimes G(C_2)$ — порождающая матрица кода $C_1 \otimes C_2$, где, согласно (4), порождающие матрицы кодов C_1 и C_2 могут быть представлены в виде соответственно:

$$G(C_1) = \begin{pmatrix} a_{1,g_1} & \dots & a_{1,g_{|\mathcal{G}|}} \\ a_{2,g_1} & \dots & a_{2,g_{|\mathcal{G}|}} \\ \dots & \dots & \dots \\ a_{k(C_1),g_1} & \dots & a_{k(C_1),g_{|\mathcal{G}|}} \end{pmatrix}, \quad G(C_2) = \begin{pmatrix} b_{1,h_1} & \dots & b_{1,h_{|\mathcal{H}|}} \\ b_{2,h_1} & \dots & b_{2,h_{|\mathcal{H}|}} \\ \dots & \dots & \dots \\ b_{k(C_2),h_1} & \dots & b_{k(C_2),h_{|\mathcal{H}|}} \end{pmatrix}.$$

Заметим, $k(C_1 \otimes C_2) = k(C_1)k(C_2)$, $n(C_1 \otimes C_2) = n(C_1)n(C_2)$ и

$$\operatorname{dist}_{B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}}(C_1 \otimes C_2) = \operatorname{dist}_{B^{\mathbb{F}\mathcal{G}}}(C_1)\operatorname{dist}_{B^{\mathbb{F}\mathcal{H}}}(C_2), \tag{6}$$

где $B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})} = \{\delta_{(g_1,h_1)}, \dots, \delta_{(g_1,h_{|\mathcal{H}|})}, \delta_{(g_2,h_1)}, \dots, \delta_{(g_{|\mathcal{G}|},h_{|\mathcal{H}|})}\}$ — базис групповой алгебры $\mathbb{F}(\mathcal{G} \times \mathcal{H})$ на группе $\mathcal{G} \times \mathcal{H}$ с линейным порядком, индуцированным линейными порядками $\operatorname{ord}(\mathcal{G})$ и $\operatorname{ord}(\mathcal{H})$.

1.4. Декодирование тензорного произведения MLD-кодов

Пусть $\mathbf{c}(\in C_1 \otimes C_2)$ – кодовый вектор, который на выходе из канала принимает вид

$$\mathbf{x} = \mathbf{c} + \mathbf{e}, \quad \mathbf{e} \in \mathbb{F}(\mathcal{G} \times \mathcal{H}). \tag{7}$$

В этом разделе строятся алгоритмы, позволяющие правильно находить значение вектора ошибок $\mathbf{e}$, если вес этого вектора удовлетворяет следующему неравенству (см. (6)):

$$w_{B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}}(\mathbf{e}) \le \left\lfloor \frac{\operatorname{dist}_{B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}}(C_1 \otimes C_2) - 1}{2} \right\rfloor. \tag{8}$$

Прежде всего сформулируем вспомогательную лемму.

Лемма 1. *Пусть* $\mathbf{c}_1 \in \mathbb{F}^{n_1}$, $\mathcal{M}_{\mathbf{c}_1}$ – *M-ортогональное множество для вектора* $\mathbf{c}_1$, $\mathbf{c}_2 \in \mathbb{F}^{n_2}$, $\mathcal{M}_{\mathbf{c}_2}$ – *M-ортогональное множество для вектора* $\mathbf{c}_2$, *тогда M-ортогональное множество для вектора* $\mathbf{c}_1 \otimes \mathbf{c}_2$ *состоит из векторов вида:*

$$1)\mathbf{c}_1 \otimes \mathbf{w},$$
$$2)\mathbf{v} \otimes \mathbf{c}_2,$$
$$3)\mathbf{c}_1 \otimes \mathbf{w} + \mathbf{v} \otimes \mathbf{c}_2 + \mathbf{v} \otimes \mathbf{w},$$

где $\mathbf{v} \in \mathcal{M}_{\mathbf{c}_1}$, $\mathbf{w} \in \mathcal{M}_{\mathbf{c}_2}$.

Доказательство. Эта лемма вытекает из [14], с. 121–122. □

Для векторов $\mathbf{c}_1 \in C_1(\subseteq \mathbb{F}\mathcal{G})$ и $\mathbf{c}_2 \in C_2(\in\subseteq \mathbb{F}\mathcal{H})$ рассмотрим соответствующие им M-ортогональные множества $\mathcal{M}_{\mathbf{c}_1}$ и $\mathcal{M}_{\mathbf{c}_2}$. Ниже построен алгоритм M_orth, который конструирует для каждого вектора $\mathbf{c}_1 \otimes \mathbf{c}_2 \in C_1 \otimes C_2(\subseteq \mathbb{F}(\mathcal{G} \times \mathcal{H}))$ такое M-ортогональное множество $\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}$, что

$$|\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}| = |\mathcal{M}_{\mathbf{c}_1}| + |\mathcal{M}_{\mathbf{c}_1}|. \tag{9}$$

Исходные параметры: Векторы $\mathbf{c}_1$,$\mathbf{c}_2$ и соответствующие им
$\mathcal{M}$-ортогональные множества $\mathcal{M}_{\mathbf{c}_1}$, $\mathcal{M}_{\mathbf{c}_2}$.
Результат: $\mathcal{M}$-ортогональное множество $\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}$ для вектора $\mathbf{c}_1 \otimes \mathbf{c}_2$.
$\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2} := \emptyset$
для каждого $\mathbf{w} \in \mathcal{M}_{\mathbf{c}_2}$ **выполнять**
| к $\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}$ добавить $\mathbf{c}_1 \otimes \mathbf{w}$
конец цикла
для каждого $\mathbf{v} \in \mathcal{M}_{\mathbf{c}_1}$ **выполнять**
| к $\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}$ добавить $\mathbf{v} \otimes \mathbf{c}_2$
конец цикла
возвратить $\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}$

Алгоритм 2: M_orth

С применением алгоритма M_orth построен алгоритм MakeTensorTree, который для корня с меткой $\mathbf{1}_\mathcal{G} \otimes \mathbf{1}_\mathcal{H}$ по декодирующим деревьям $\mathrm{WB}_{\mathbf{1}_\mathcal{G},r_{\mathbf{1}_\mathcal{G}},L_{\mathbf{1}_\mathcal{G}}}[C_1]$ и $\mathrm{WB}_{\mathbf{1}_\mathcal{H},r_{\mathbf{1}_\mathcal{H}},L_{\mathbf{1}_\mathcal{H}}}[C_2]$ строит некоторое *вспомогательное декодирующее дерево*, которое обозначим следующим образом:

$$\mathrm{WB}^{\otimes}{}_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H},r_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}},L_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}}[C_1 \otimes C_2]. \tag{10}$$

Поясним, почему дерево, строящееся в алгоритме MakeTensorTree, имеет $L_{\mathbf{1}_\mathcal{G}} + L_{\mathbf{1}_\mathcal{H}} - 1$ уровней. В этом алгоритме с помощью алгоритма M_orth для каждой вершины $\mathbf{v} = \mathbf{c}_1^i \otimes \mathbf{c}_2^j$ на уровне k дерева (10), где $i \in \{0, ..., L_{\mathbf{1}_\mathcal{G}} - 1\}$ – i-й уровень декодирующего дерева для кода C_1, $j \in \{0, ..., L_{\mathbf{1}_\mathcal{H}} - 1\}$ – j-й уровень декодирующего дерева для кода C_2, строится множество векторов на уровне $k + 1$ дерева (10), состоящее из подмножеств двух типов:

1) $\mathbf{c}_1^i \otimes \mathbf{c}_2^{j+1}$,
2) $\mathbf{c}_1^{i+1} \otimes \mathbf{c}_2^j$.

Таким образом, максимальный уровень дерева (10) будет достигаться, например, если сначала поочередно пройти $L_{\mathbf{1}_\mathcal{G}} - 1$ векторов второго типа, каждый из которых находится на следующем уровне дерева, затем пройти $L_{\mathbf{1}_\mathcal{H}}$ векторов первого типа. Тогда глубина дерева (10) будет равна $L_{\mathbf{1}_\mathcal{G}} + L_{\mathbf{1}_\mathcal{H}} - 1$.

Если C_1 и C_2 — групповые MLD-коды, то есть $\mathrm{dmaj}_{B^{\mathbb{F}\mathcal{G}}}(C_1) = \mathrm{dist}_{B^{\mathbb{F}\mathcal{G}}}(C_1) - 1$ и $\mathrm{dmaj}_{B^{\mathbb{F}\mathcal{G}}}(C_2) = \mathrm{dist}_{B^{\mathbb{F}\mathcal{G}}}(C_2) - 1$, то в общем случае вспомогательное декодирующее дерево, построенное по алгоритму MakeTensorTree, не позволяет найти с помощью мажоритарного декодера (см. [13], алгоритм 3 Decoder2) значения ошибок $\mathbf{e}$, вес которых удовлетворяет неравенству (8). Дело в том, что из сравнения равенств (6) и

Исходные параметры: $\mathrm{WB}_{\mathbf{1}_\mathcal{G},r_{\mathbf{1}_\mathcal{G}},L_{\mathbf{1}_\mathcal{G}}}[C_1]$, $\mathrm{WB}_{\mathbf{1}_\mathcal{H},r_{\mathbf{1}_\mathcal{H}},L_{\mathbf{1}_\mathcal{H}}}[C_2]$.
Результат: дерево (10).
$V_1^{\otimes} := \mathrm{M\_orth}(\mathbf{1}_\mathcal{G}, \mathbf{1}_\mathcal{H}, \mathcal{M}_{\mathbf{1}_\mathcal{G}}, \mathcal{M}_{\mathbf{1}_\mathcal{H}})$
цикл $1 \le k \le L_{\mathbf{1}_\mathcal{G}} + L_{\mathbf{1}_\mathcal{H}} - 2$ **выполнять**
  для каждого $\mathbf{v}^k = (\mathbf{c}_1^k \otimes \mathbf{c}_2^k) \in V_k^{\otimes}$ **выполнять**
    если $\mathbf{v}^k \notin (C_1 \otimes C_2)^{\perp}$ **тогда**
      на уровень $V_{k+1}^{\otimes}$ добавить $|\mathcal{M}_{\mathbf{c}_1^k}| \cdot |\mathcal{M}_{\mathbf{c}_2^k}|$ вершин и соединить их с вершиной, имеющей метку $\mathbf{v}^k$ на уровне $V_k^{\otimes}$;
      пометить добавленные вершины метками из множества

$$\mathcal{M}_{\mathbf{v}^k} = \mathrm{M\_orth}(\mathbf{c}_1^k, \mathbf{c}_2^k, \mathcal{M}_{\mathbf{c}_1^k}, \mathcal{M}_{\mathbf{c}_2^k}).$$

    конец условия
  конец цикла
конец цикла
возвратить $\mathrm{WB}^{\otimes}_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H},r_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}},L_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}}[C_1 \otimes C_2]$

Алгоритм 3: MakeTensorTree

(9) вытекает, что мощность $|\mathcal{M}_{\mathbf{c}_1}|+|\mathcal{M}_{\mathbf{c}_2}|$ построенного алгоритмом MakeTensorTree M-ортогонального множества $\mathcal{M}_{\mathbf{c}_1\otimes\mathbf{c}_2}$ меньше

$$\mathrm{dist}_{B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}}(C_1 \otimes C_2) - 1 = |\mathcal{M}_{\mathbf{c}_1}||\mathcal{M}_{\mathbf{c}_2}| + |\mathcal{M}_{\mathbf{c}_1}| + |\mathcal{M}_{\mathbf{c}_2}|$$

для каждого узла с меткой $\mathbf{c}_1 \otimes \mathbf{c}_2$.

Вспомогательное декодирующее дерево (10) может быть достроено до полного декодирующего дерева MLD-кода $C_1 \otimes C_2$ на основании конструкции 3) леммы 1 путем добавления недостающих вершин, однако представляется удобным работать только со значениями меток недостающих вершин. Алгоритм AddVals для каждого узла $\mathbf{v} = \mathbf{c}_1 \otimes \mathbf{c}_2$ по принятому из канала вектору $\mathbf{x}$ и вспомогательному декодирующему дереву (10) вычисляет дополнительные $|\mathcal{M}_{\mathbf{c}_1}||\mathcal{M}_{\mathbf{c}_2}|$ значений меток недостающих вершин, но, подчеркнем, к дереву (10) при выполнении алгоритма AddVals дополнительные вершины не добавляются.

Алгоритм AddVals применяется в алгоритме декодирования DecodeTensorBit, который по принятому вектору $\mathbf{x}$ находит значение $e_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}$ вектора ошибок $\mathbf{e}$ в координате, соответствующей базисной функции $\mathbf{1}_\mathcal{G} \otimes \mathbf{1}_\mathcal{H}$. (Отметим, что в алгоритме DecodeTensorBit используется операция конкатенации наборов чисел, которая обозначается символом $\uplus$.) Таким образом, алгоритм DecodeTensorBit в случае тензорного произведения кодов выполняет функцию упомянутого выше алгоритма мажоритарного декодирования и правильно находит значение координаты вектора ошибок $\mathbf{e}$, когда вес ошибки удовлетворяет неравенству (8).

Заметим, что алгоритм MakeTensorTree строит вспомогательное декодирующее дерево для координаты, соответствующей элементу $\widehat{1}_\mathcal{G}\times\widehat{1}_\mathcal{H}(\in \mathcal{G}\times\mathcal{H})$. В [13] для групповых кодов построен приведенный ниже вспомогательный алгоритм CloneTree, позволяющий по вспомогательному декодирующему дереву с одной меткой у корня построить вспомогательноее декодирующее дерево для корня с любой другой меткой.

Исходные параметры: $\mathbf{x}$ – вектор вида (7), $\mathcal{M}_{\mathbf{v}^k}$ – M - ортогональное множество для $\mathbf{v}^k = \mathbf{c}_1^k \otimes \mathbf{c}_2^k$, $\mathrm{WB}^{\otimes}{}_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}, r_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}, L_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}}[C_1 \otimes C_2]$ – декодирующее дерево, k – текущий уровень дерева
Результат: Набор чисел l_k мощности $|\mathcal{M}_{\mathbf{c}_1^k}| \cdot |\mathcal{M}_{\mathbf{c}_2^k}|$
если $k = L_{\mathbf{1}_\mathcal{G}} + L_{\mathbf{1}_\mathcal{H}} - 2$ **тогда**
  цикл $1 \le i \le |\mathcal{M}_{\mathbf{c}_2^k}|$ **выполнять**
    цикл $|\mathcal{M}_{\mathbf{c}_2^k}| + 1 \le j \le |\mathcal{M}_{\mathbf{c}_2^k}| + |\mathcal{M}_{\mathbf{c}_1^k}|$ **выполнять**
      к l_k добавить $l(\mathbf{v}_i = (\mathbf{c}_1^i \otimes \mathbf{c}_2^i)) + l(\mathbf{v}_j = (\mathbf{c}_1^j \otimes \mathbf{c}_2^j)) + < \mathbf{c}_1^j \otimes \mathbf{c}_2^i, \mathbf{x} >$, где $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{M}_{\mathbf{v}^k}, \mathbf{c}_1^j \otimes \mathbf{c}_2^i \in (C_1 \otimes C_2)^\perp$
    конец цикла
  конец цикла
иначе
  цикл $1 \le i \le |\mathcal{M}_{\mathbf{c}_2^k}|$ **выполнять**
    цикл $|\mathcal{M}_{\mathbf{c}_2^k}| + 1 \le j \le |\mathcal{M}_{\mathbf{c}_2^k}| + |\mathcal{M}_{\mathbf{c}_1^k}|$ **выполнять**
      к l_k добавить $l(\mathbf{v}_i = (\mathbf{c}_1^i \otimes \mathbf{c}_2^i)) + l(\mathbf{v}_j = (\mathbf{c}_1^j \otimes \mathbf{c}_2^j)) + l(\mathbf{c}_1^j \otimes \mathbf{c}_2^i)$, где $\mathbf{v}_i, \mathbf{v}_j \in \mathcal{M}_{\mathbf{v}^k}, \mathbf{c}_1^j \otimes \mathbf{c}_2^i \in V^{\otimes}{}_{k+2}$
    конец цикла
  конец цикла
конец условия
возвратить l_k

Алгоритм 4: AddVals

Исходные параметры: $\mathbf{x}$ – вектор вида (7), $\mathrm{WB}^{\otimes}[C_1 \otimes C_2] = \mathrm{WB}^{\otimes}{}_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}, r_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}, L_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}}[C_1 \otimes C_2]$ – декодирующее дерево
Результат: $e_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}$
цикл $1 \le k \le L_{\mathbf{1}_\mathcal{G}} + L_{\mathbf{1}_\mathcal{H}} - 1$ **выполнять**
  для каждого $\mathbf{v}_k \in V^{\otimes}{}_k$ **выполнять**
    если $\mathbf{v}_k \in (C_1 \otimes C_2)^\perp$ **тогда**
      $l(\mathbf{v}_k) :=< \mathbf{v}_k, \mathbf{x} >$
    иначе
      $l(\mathbf{v}_k) := \mathrm{MajorVote}(l[\mathcal{M}_{v^k}] \uplus \mathrm{AddVals}(\mathbf{x}, \mathcal{M}_{v^k}, \mathrm{WB}^{\otimes}[C_1 \otimes C_2], k))$
    конец условия
  конец цикла
конец цикла
$e_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}} := \mathrm{MajorVote}(l[\mathcal{M}_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}] \uplus \mathrm{AddVals}(\mathbf{x}, \mathcal{M}_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}, \mathrm{WB}^{\otimes}[C_1 \otimes C_2], 0))$
возвратить $e_{\mathbf{1}_\mathcal{G}\otimes\mathbf{1}_\mathcal{H}}$

Алгоритм 5: DecodeTensorBit

Таким образом, набор вспомогательных декодирующих деревьев

$$\mathcal{WB}^{\otimes}(C_1 \otimes C_2) = \{\mathrm{WB}^{\otimes}_{\delta_{(g,h)}, r_{\delta_{(g,h)}}, L_{\delta_{(g,h)}}}\}_{\delta_{(g,h)} \in B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}}$$

для группового кода $C_1 \otimes C_2$ может быть построен по дереву (10). Именно, для

Исходные параметры: $\mathcal{G}$, $\mathrm{WB}^{\otimes}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}}[C]$, $\mathbf{b}'$
Результат: $\mathrm{WB}^{\otimes}_{\mathbf{b}',r'_{\mathbf{b}},L_{\mathbf{b}'}}[C]$
$\mathrm{WB}^{\otimes}_{\mathbf{b}',r'_{\mathbf{b}},L_{\mathbf{b}'}}[C] := \mathrm{WB}^{\otimes}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}}[C]$;
Найти $g(\in \mathcal{G})$ такой, что $(g,\mathbf{b}) = \mathbf{b}'$;
для каждой *метки* $\mathbf{p}$ *дерева* $\mathrm{WB}^{\otimes}_{\mathbf{b}',r'_{\mathbf{b}},L_{\mathbf{b}'}}[C]$ **выполнять**
  // Действие элементом g на $\mathbf{p}$ по правилу (5);
  $\mathbf{p} := (g,\mathbf{p})$;
конец цикла
возвратить $\mathrm{WB}^{\otimes}_{\mathbf{b}',r'_{\mathbf{b}},L_{\mathbf{b}'}}[C]$

Алгоритм 6: CloneTree

любой базисной функции $\delta_{(g,h)} \in B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}$:

$$\mathrm{WB}^{\otimes}_{\delta_{(g,h)},r_{\delta_{(g,h)}},L_{\delta_{(g,h)}}} = \mathrm{CloneTree}(\mathcal{G}\times\mathcal{H}, \mathrm{WB}^{\otimes}{}_{\mathbf{1}_{\mathcal{G}}\otimes\mathbf{1}_{\mathcal{H}},r_{\mathbf{1}_{\mathcal{G}}\otimes\mathbf{1}_{\mathcal{H}}},L_{\mathbf{1}_{\mathcal{G}}\otimes\mathbf{1}_{\mathcal{H}}}}, \delta_{(g,h)}).$$

По аналогии с алгоритмом Decoder3 из [13] построен алгоритм DecodeTensorVector декодирования принятого вектора $\mathbf{x}$, в котором каждая координата декодируется с помощью алгоритма DecodeTensorBit.

Исходные параметры: принятый вектор $\mathbf{x} = \mathbf{c} + \mathbf{e}$, набор вспомогательных декодирующих деревьев
$\mathcal{WB}^{\otimes}(C_1 \otimes C_2) = \{\mathrm{WB}^{\otimes}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}}\}_{\mathbf{b}\in B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}}$
Результат: вектор $\mathbf{c}'$ – результат декодирования
для каждого $\mathbf{b} \in B^{\mathbb{F}(\mathcal{G}\times\mathcal{H})}$ **выполнять**
  $a := x_{\mathbf{b}} - \mathrm{DecodeTensorBit}(\mathbf{x}, \mathbf{b}_i, \mathrm{WB}^{\otimes}_{\mathbf{b},r_{\mathbf{b}},L_{\mathbf{b}}})$
конец цикла
возвратить $\mathbf{c}'$

Алгоритм 7: DecodeTensorVector

Таким образом справедлива следующая теорема.

Теорема 1. *Пусть* C_1 *и* C_2 *— групповые* MLD*-коды с наборами декодирующих деревьев* $\mathcal{WB}(C_1)$ *и* $\mathcal{WB}(C_2)$ *соответственно,* $\mathcal{WB}^{\otimes}(C_1 \otimes C_2)$ *– набор вспомогательных декодирующих деревьев кода* $C_1 \otimes C_2$*, построенный с помощью алгоритмов* MakeTensorTree *и* CloneTree *с использованием* $\mathcal{WB}(C_1)$ *и* $\mathcal{WB}(C_2)$*. Тогда, если* $\mathbf{c}(\in C_1 \otimes C_2)$ *– кодовый вектор, который на выходе из канала принимает вид* $\mathbf{x} = \mathbf{c} + \mathbf{e}$*, где вес вектора ошибок* $\mathbf{e}$ *удовлетворяет условию* (8)*, то*

$$\mathrm{DecodeTensorVector}(\mathbf{x}, \mathcal{WB}^{\otimes}(C_1 \otimes C_2)) = \mathbf{c}.$$

2. Криптосистема типа Мак-Элиса на основе произведения кодов

2.1. Криптосистема типа Мак-Элиса

Пусть $C(\subseteq \mathbb{F}^n)$ – линейный $[n,k,d]$-код длины $n = n(C)$, размерности $k = k(C)$, с кодовым расстоянием $d = \mathrm{dist}_B(C)$, $G(C)$ – порождающая матрица кода C. Под крип-

тосистемой типа Мак-Элиса на основе $[n, k, d]$-кода C здесь понимается асимметричная криптосистема, в которой открытый ключ $\mathbf{k}_{\text{pub}}$ – это пара $(\widetilde{G}, t = \lfloor (d-1)/2 \rfloor)$, а секретный ключ $\mathbf{k}_{\text{sec}}$ – пара матриц (S, P), где S – случайная невырожденная $(k \times k)$-матрица, P – случайная перестановочная $(n \times n)$-матрица, причем $\widetilde{G} = S \cdot G(C) \cdot P$. Правило шифрования произвольного сообщения $\mathbf{s} (\in \mathbb{F}^k)$ имеет вид:

$$\mathbf{z} = \mathbf{s}\widetilde{G} + \mathbf{e}, \tag{11}$$

где вес Хэмминга добавляемой ошибки $\mathbf{e} = (e_1, ..., e_n)$ удовлетворяет неравенству: $w_B(\mathbf{e}) \leq t$. Для расшифрования $\mathbf{c}$ секретный ключ $\mathbf{k}_{\text{sec}}$ используется по правилу: $\mathbf{s} = \text{Dec}_C(\mathbf{z}P^{-1})S^{-1}$, где $\text{Dec}_C : \mathbb{F}^n \to \mathbb{F}^k$ — декодер кода C, гарантированно исправляющий t и менее ошибок и восстанавливающий вектор $\mathbf{s}$. Далее предполагается, что вектор ошибок $\mathbf{e}$ выбирается случайно и равновероятно из множества $\mathbb{F}^{n,t} = \mathbb{F}_q^{n,t} (\subseteq \mathbb{F}_q^n)$, состоящего из векторов веса t, $|\mathbb{F}^{n,t}| = C_n^t (q-1)^t$.

2.2. Анализ стойкости $\text{McE}(C_1 \otimes C_2)$ к атакам на ключ

Рассмотрим криптосистему типа Мак-Элиса $\text{McE}(C)$, где $C = C_1 \otimes C_2$ – тензорное произведение $[n_1, k_1, d_1]$-кода C_1 и $[n_2, k_2, d_2]$-кода C_2. В качестве модели нарушителя рассмотрим противника, целью которого является нахождение подходящего секретного ключа для правильного расшифрования криптограмм. Предполагается, что наблюдатель имеет алгоритм Attack, с помощью которого может быть эффективно найден подходящий секретный ключ для криптосистемы $\text{McE}(C_2)$.

Порождающая матрица кода $C = C_1 \otimes C_2$ имеет вид $G(C) = G(C_1) \otimes G(C_2)$, а размерность K кода C равна $k_1 k_2$. Тогда $(k_1 k_2 \times k_1 k_2)$-матрица S (часть секретного ключа $\mathbf{k}_{\text{sec}}$) может быть представлена в блочном виде:

$$S = \left(\begin{array}{c|c|c|c} S_{0,0} & S_{0,1} & ... & S_{0,k_1-1} \\ \hline S_{1,0} & S_{1,1} & ... & S_{1,k_1-1} \\ \hline ... & ... & ... & ... \\ \hline S_{k_1-1,0} & S_{k_1-1,1} & ... & S_{k_1-1,k_1-1} \end{array} \right), \tag{12}$$

где S_{ij} — $(k_2 \times k_2)$ -матрица, $i, j = 0, ..., k_1 - 1$. Поэтому для матрицы $S \cdot G(C)$ имеет место представление:

$$S \cdot G(C) = \left(\begin{array}{c|c|c} \sum\limits_{j=0}^{k_1-1} S_{0,j} g_{j,1}^1 G(C_2) & ... & \sum\limits_{j=0}^{k_1-1} S_{0,j} g_{j,n_1}^1 G(C_2) \\ \hline \sum\limits_{j=0}^{k_1-1} S_{1,j} g_{j,1}^1 G(C_2) & ... & \sum\limits_{j=0}^{k_1} S_{1,j} g_{j,n_1}^1 G(C_2) \\ \hline ... & ... & ... \\ \hline \sum\limits_{j=0}^{k_1-1} S_{k_1-1,j} g_{j,1}^1 G(C_2) & ... & \sum\limits_{j=0}^{k_1-1} S_{k_1-1,j} g_{jn_1}^1 G(C_2) \end{array} \right). \tag{13}$$

Для каждого $i \in \{1, ..., n_1\}$ блочный столбец матрицы (13) представим в виде:

$$\left(\begin{array}{c} \sum\limits_{j=0}^{k_1-1} S_{0,j} g_{j,i}^1 G(C_2) \\ \hline \sum\limits_{j=0}^{k_1-1} S_{1,j} g_{j,i}^1 G(C_2) \\ \hline \ldots \\ \hline \sum\limits_{j=0}^{k_1-1} S_{v_1,j} g_{j,i}^1 G(C_2) \end{array}\right) = \mathbf{S}_i G(C_2), \quad \mathbf{S}_i = \left(\begin{array}{c} S_{0,0} \\ \hline S_{1,0} \\ \hline \ldots \\ \hline S_{k_1-1,0} \end{array}\right) g_{0,i}^1 + \ldots + \left(\begin{array}{c} S_{0,k_1-1} \\ \hline S_{1,k_1-1} \\ \hline \ldots \\ \hline S_{k_1-1,k_1-1} \end{array}\right) g_{k_1-1,i}^1. \tag{14}$$

Непосредственно проверяется, что для каждого $i \in \{1, ..., n_1\}$ матрица $\mathbf{S}_i$ имеет ранг k_2. Тогда сложность нахождения подходящего секретного ключа не превышает соответствующей сложности применения алгоритма AttackInduced из [12] для криптосистемы типа Мак-Элиса на индуцированном коде с порождающей матрицей $I_{n_1} \otimes G_{k_2-1}^2$. Именно, если Q – вычислительная сложность алгоритма Attack для $\mathrm{McE}(C_2)$, то

$$\mathcal{O}\left(\left(\frac{n_1^{n_2-1}}{e}\right)^{n_1} (n_1 Q + k_1 k_2)^3\right) \tag{15}$$

– оценка сверху на сложность нахождения ключа для $\mathrm{McE}(C_1 \otimes C_2)$.

Таблица 1. Значения величины $\lceil \log_2(K(n_1, n_2)) \rceil$, где $K(n_1, n_2)$ – количество перебираемых ключей в атаке AttackInduced на криптосистему $\mathrm{McE}(C_1 \otimes C_2)$, здесь C_i – код Рида–Маллера $\mathcal{RM}(r_i, m_i)$, $n_i = 2^{m_i}$, $r_i \le m_i$, $m_i = 1, ..., 8$, $i \in \{1; 2\}$
Table 1. Values of $\lceil \log_2(K(n_1, n_2)) \rceil$, where $K(n_1, n_2)$ is number of probed keys in attack AttackInduced for $\mathrm{McE}(C_1 \otimes C_2)$ cryptosystem, where here C_i – Reed-Muller code $\mathcal{RM}(r_i, m_i)$, $n_i = 2^{m_i}$, $r_i \le m_i$, $m_i = 1, ..., 8$, $i \in \{1; 2\}$

	n_2							
n_1	*2*	*4*	*8*	*16*	*32*	*64*	*128*	*256*
2	0	0	8	25	58	122	**250**	**506**
4	0	16	48	112	**240**	**496**	**1008**	**2032**
8	8	48	**152**	**344**	**728**	**1496**	**3032**	**6104**
16	25	112	**344**	**928**	**1952**	**4000**	**8096**	**16288**
32	52	**240**	**728**	**1952**	**4896**	**10016**	**20256**	**40000**
64	122	**496**	**1496**	**4000**	**10016**	**24064**	**48640**	**96000**
128	**250**	**1008**	**3032**	**8096**	**20256**	**48640**	**113536**	**237000**
256	**506**	**2032**	**6104**	**16288**	**40000**	**96000**	**237000**	**512000**

В таблице 1 приведен пример расчета сложности атаки на ключ для криптосистемы $\mathrm{McE}(C_1 \otimes C_2)$, где C_i – код Рида–Маллера, $i \in \{1; 2\}$. В каждой ячейке таблицы приведено число $\lceil \log_2(K(n_1, n_2)) \rceil$, где $K(n_1, n_2)$ – количество перебираемых ключей в атаке AttackInduced (множитель $(n_1^{n_2-1} e^{-1})^{n_1}$ в (15)), где n_1 и n_2 – длины кодов C_1 и C_2 соответственно. Жирным выделены те ячейки таблицы, для которых $\lceil \log_2(K(n_1, n_2)) \rceil \ge 128$, так как перебор ключей длины 128 бит и более

в настоящее время является вычислительно неосуществимой задачей [16]. Заметим также, что $A \otimes B = Q_1 \cdot (B \otimes A) \cdot Q_2$, где Q_1 и Q_2 – перестановочные матрицы подходящего размера. Поэтому парам (n_1, n_2) и (n_2, n_1) в таблице соответствуют ячейки с одинаковым значением $\min\{\lceil \log_2(K(n_1, n_2)) \rceil ; \lceil \log_2(K(n_2, n_1)) \rceil\}$. На основании результатов, представленных в таблице 1, можно сделать вывод, что для кодов Рида–Маллера C_1 и C_2 криптосистема $\mathrm{McE}(C_1 \otimes C_2)$ представляется стойкой к структурным атакам на ключ уже при $n_i \geq 8$, где n_i – длина кода C_i, $i = 1, 2$.

Список литературы / References

[1] Shor P. W., “Algorithms for quantum computation: Discrete logarithms and factoring”, *Proceedings 35th Annual Symposium on Foundations of Computer Science*, IEEE Computer Society Press, 1994, 124–134.

[2] Sendrier N., Tillich J.-P., “Code-Based Cryptography: New Security Solutions Against a Quantum Adversary”, *ERCIM News, ERCIM, 2016, Special Theme Cybersecurity (106). hal-01410068.*

[3] McEliece R. J., “A Public-Key Cryptosystem Based on Algebraic Coding Theory”, *JPL Deep Space Network Progress Report*, 1978, № 42, 114–116.

[4] Niederreiter H., “Knapsack-Type Cryptosystem and Algebraic Coding Theory”, *Probl. Control and Inform. Theory*, **15** (1986), 94–34.

[5] Gabidulin E. M. et al., “Ideals Over a Non-Commutative Ring and Their Application in Cryptology”, *Advances in Cryptology–EUROCRYPT’91 / Ed. by D. W. Davies. Lect. Notes in Comp. Sci.*, **547** (1991), 482–489.

[6] Сидельников В. М., “Открытое шифрование на основе двоичных кодов Рида–Маллера”, *Дискретная математика*, **6**:2 (1994), 3–20; [Sidel’nikov V. M., “Open coding based on Reed–Muller binary codes”, *Diskr. Mat.*, **6**:2 (1994), 3–20, (in Russian).]

[7] Сидельников В. М., Шестаков С. О., “О системе шифрования, основанной на обобщенных кодах Рида–Соломона”, *Дискретная математика*, **3**:3 (1992), 57–63; [Sidel’nikov V. M., Shestakov S. O., “O sisteme shifrovanija, osnovannoj pa obobshhennyh kodah Rida–Solomona”, *Diskr. Mat.*, **3**:3 (1992), 57–63, (in Russian).]

[8] Деундяк В. М. и др., “Модификация криптоаналитического алгоритма Сидельникова–Шестакова для обобщенных кодов Рида–Соломона и ее программная реализация”, *Известия высших учебных заведений. Северо-Кавказский регион. Технические науки*, 2006, № 4, 15–20; [Deundyak V. M. et al., “Modifikatsiya kriptoanaliticheskogo algoritma Sidel’nikova–Shestakova dlya obobshchennykh kodov Rida-Solomona i ee programmnaya realizatsiya”, *Izvestiya vysshikh uchebnykh zavedeniy. Severo-Kavkazskiy region. Tekhnicheskie nauki*, 2006, № 4, 15–20, (in Russian).]

[9] Overbeck R., “Structural Attacks for Public Key Cryptosystems based on Gabidulin Codes”, *Journal of Cryptology*, **21**:2 (2008), 280–301.

[10] Minder L., Shokrollahi A., “Cryptanalysis of the Sidelnikov cryptosystem”, *Lecture Notes in Computer Science*, **4515** (2007), 347–360.

[11] Чижов И. И., Бородин М. А., “Эффективная атака на криптосистему Мак-Элиса, построенную на основе кодов Рида–Маллера”, *Дискрет. матем.*, **26**:1 (2014), 10–20; [Chizhov I. I., Borodin M. A., “Jeffektivnaja ataka na kriptosistemu Mak-Jelisa, postroennuju na osnove kodov Rida–Mallera”, *Diskr. Mat.*, **26**:1 (2014), 10–20, (in Russian).]

[12] Деундяк В. М., Косолапов Ю. В., “Криптосистема на индуцированных групповых кодах”, *Модел. и анализ информ. систем*, **23**:2 (2016), 137–152; [Deundyak V. M., Kosolapov Yu. V., “Cryptosystem Based on Induced Group Codes”, *Modeling and Analysis of Information Systems*, **23**:2 (2016), 137–152, (in Russian).]

[13] Деундяк В. М., Косолапов Ю. В., "Алгоритмы для мажоритарного декодирования групповых кодов", *Модел. и анализ информ. систем*, **22**:4 (2015), 464–482; [Deundyak V. M., Kosolapov Yu. V., "Algorithms for Majority Decoding of Group Codes", *Modeling and Analysis of Information Systems*, **22**:4 (2015), 464–482, (in Russian).]

[14] Циммерман К.-Х., *Методы теории модулярных представлений в алгебраической теории кодирования*, МЦНМО, М., 2011; [Tsimmerman K.-Kh., *Metody teorii modulyarnykh predstavleniy v algebraicheskoy teorii kodirovaniya*, MTsNMO, M., 2011, (in Russian).]

[15] Curtis C. W., Reiner I., *Representation Theory of Finite Groups and Associative Algebras*, Intersclence Publishers, New York, 1962.

[16] Lenstra A. K., Verheul E. R., "Selecting Cryptographic Key Sizes", *Journal of Cryptology*, **14** (2001), 255–293.

Deundyak V. M., Kosolapov Y. V., Lelyuk E. A., "Decoding the Tensor Product of MLD Codes and Applications for Code Cryptosystems", *Modeling and Analysis of Information Systems*, **24**:2 (2017), 239–252.

DOI: 10.18255/1818-1015-2017-2-239-252

Abstract. For the practical application of code cryptosystems such as McEliece, it is necessary that the code used in the cryptosystem should have a fast decoding algorithm. On the other hand, the code used must be such that finding a secret key from a known public key would be impractical with a relatively small key size. In this connection, in the present paper it is proposed to use the tensor product $C_1 \otimes C_2$ of group MLD codes C_1 and C_2 in a McEliece-type cryptosystem. The algebraic structure of the code $C_1 \otimes C_2$ in the general case differs from the structure of the codes C_1 and C_2, so it is possible to build stable cryptosystems of the McEliece type even on the basis of codes C_i for which successful attacks on the key are known. However, in this way there is a problem of decoding the code $C_1 \otimes C_2$. The main result of this paper is the construction and justification of a set of fast algorithms needed for decoding this code. The process of constructing the decoder relies heavily on the group properties of the code $C_1 \otimes C_2$. As an application, the McEliece-type cryptosystem is constructed on the code $C_1 \otimes C_2$ and an estimate is given of its resistance to attack on the key under the assumption that for code cryptosystems on codes C_i an effective attack on the key is possible. The results obtained are numerically illustrated in the case when C_1, C_2 are Reed–Muller–Berman codes for which the corresponding code cryptosystem was hacked by L. Minder and A. Shokrollahi (2007).

Keywords: majority decoder, Reed–Muller–Berman codes, tensor product codes

About the authors:
Deundyak Vladimir Mikhailovich, orcid.org/0000-0001-8258-2419, PhD,
FGNU NII "Specvuzavtomatika",
51 Gazetniy lane, Rostov-on-Don 344002, Russia
South Federal University,
105/42 Bolshaya Sadovaya Str., Rostov-on-Don 344006, Russia, e-mail: vl.deundyak@gmail.com,

Kosolapov Yury Vladimirovich, orcid.org/0000-0002-1491-524X, PhD,
South Federal University,
105/42 Bolshaya Sadovaya Str., Rostov-on-Don 344006, Russia, e-mail: itaim@mail.ru,

Leluk Evgeniy Andreevich, orcid.org/0000-0001-6560-2561,
South Federal University,
105/42 Bolshaya Sadovaya Str., Rostov-on-Don 344006, Russia, e-mail: lelukevgeniy@mail.ru