# Проблемы при использовании кодовой хэш-функции для схемы подписи CFS, построенной на кодах Гоппы

А. С. Илюхина

***Аннотация*—В 2001 году была предложена схема подписи CFS, основанная на криптосистеме Нидеррайтера. Алгоритм подписи строится на базе кодовой криптографии, что делает подпись стойкой для постквантовых атак. Однако существуют определенные трудности в ее реализации. Одна из них заключается в сложности построения подписи из-за малой вероятности получения допустимого легко декодируемого синдрома. Настоящая статья рассматривает известный способ модификации оригинальной схемы подписи для решения этой проблемы. В работе изучается использование кодовой хэш-функции для быстрого получения декодируемого синдрома. При рассмотрении функции сжатия функции хэширования была найдена ошибка в ее построении и доказана небезопасность схемы подписи, построенной на такой хэш-функции.**

***Ключевые слова*—схема подписи CFS, кодовая хэш-функция**

## Введение

Электронная подпись (ЭП) — это реквизит электронного документа, позволяющий подтвердить принадлежность владельцу, а также защитить документ от изменений с момента подписания. Он получается в результате криптографического преобразования информации с использованием закрытого ключа подписи и позволяет проверить отсутствие искажения информации в электронном документе с момента формирования подписи, принадлежность подписи владельцу сертификата ключа подписи, а в случае успешной проверки — подтвердить факт подписания электронного документа. ЭП можно построить на основе криптографического протокола электронной цифровой подписи (ЭЦП). Подпись с помощью криптографических преобразований связывается не только с автором, но и с самим документом, поэтому не может быть подделана с помощью копирования.

В зависимости от алгоритма, функция, формирующая подпись, может быть детерминированной или вероятностной. Детерминированные функции всегда вычисляют одинаковую подпись для одного и того же документа. Вероятностные функции вносят в подпись элемент случайности, что усиливает криптостойкость алгоритмов ЭЦП. Однако для вероятностных схем необходимы либо аппаратный генератор шума, либо криптографически надёжный генератор псевдослучайных битов, что усложняет их использование. Различают одноразовые схемы ЭЦП, в которых после проверки подписи надо провести замену ключей, и многоразовые схемы, которые этого не требуют.

Схемы электронной цифровой подписи относятся к криптографическим алгоритмам с открытым ключом. Стартом в развитии таких алгоритмов стала работа Уитфилда Диффи и Мартина Хеллмана [1], опубликованная в 1976 году. Описанные в ней алгоритмы и те алгоритмы, которые появились после этой работы, строились на базе функций, которые сложно обратить за полиномиальное время. Это функции, для которых легко вычисляется значение по заданному аргументу, но сложно вычисляется значение аргумента по заданному значению функции. Именно такую функцию несколькими годами позже использовали в своей работе Рон Ривест, Ади Шамир и Леонард Адлеман, разработав систему RSA [2], первую практическую криптосистему с открытым ключом, стойкость которой основана на проблеме факторизации больших простых чисел. Так, нахождение произведения больших простых чисел в вычислительном отношении осуществляется легко, однако разложение на множители произведения таких чисел в вычислительном отношении представляется невыполнимым. В настоящее время алгоритм RSA принят в качестве международных стандартов ISO/IEC/DIS 9594-8 и X.509. Также Международная сеть электронного перечисления платежей SWIFT требует от банковских учреждений, пользующихся ее услугами, применения именно этого алгоритма для шифрования информации. Помимо схемы аутентификации в статье [2] представлена схема подписи, основанная на криптографическом алгоритме RSA. Так, если в асимметричной криптосистеме шифрование выполняют на открытом ключе, а расшифрование – на закрытом ключе получателя, то в схеме ЭЦП подпись производят с помощью закрытого ключа, а проверку – с помощью открытого ключа пользователя, передающего сообщения. После RSA были разработаны другие ЭЦП, такие, как алгоритмы цифровой подписи Рабина [3], Меркля [4].

Стойкость современных алгоритмов ЭЦП основана на вычислительной сложности решения различных математических задач, в число которых входят дискретное логарифмирование, факторизация и другие. Существует множество различных классов сложности таких задач, отражающих уровень вычислительных затрат для получения решения. Пример такого класса — класс P. В него входят все задачи, которые может решить классический компьютер за полиномиальное время. Задача же факторизации, на которой основан алгоритм RSA, предположительно не принадлежит этому классу, так как является более

Статья получена 16 мая 2022

Анастасия Сергеевна Илюхина, МГУ им. М.В. Ломоносова, (email: iluhina.enot@gmail.com).

сложной. Другой известный класс — класс NP. В него входят те задачи распознавания, которые классический компьютер может выполнить при наличии некоторых дополнительных сведений. Для решения таких задач на данный момент неизвестны быстрые алгоритмы, для которых сложность решения полиномиально зависела бы от размера входных данных. Существующие алгоритмы имеют экспоненциальную зависимость — это значит, что даже на мощных классических компьютерах такие задачи не будут решаться за разумное время при достаточно большой размерности задачи.

Однако уже сейчас ведутся активные разработки в области квантовых компьютеров. Первые идеи были высказаны еще в 1980 годах. Квантовый компьютер — вычислительное устройство, которое использует явления квантовой механики для передачи и обработки данных. Они используют квантовые биты — кубиты. В отличие от обычных битов, которые могут находиться либо в состоянии 0, либо в состоянии 1, кубиты существуют в смешанном состоянии, что позволяет обрабатывать все возможные состояния одновременно, достигая существенного преимущества над обычными компьютерами. Работу над квантовыми компьютерами сейчас ведут такие гиганты ИТ-индустрии, как IBM, Microsoft, Google и Intel. Это говорит о том, что прорывы в этой области могут появиться в самое ближайшее время [5].

В 1994 году был опубликован квантовый алгоритм Шора [6]. Алгоритм был разработан для решения задач факторизации целых чисел и дискретного логарифмирования в конечной группе. Он позволяет факторизовать число $N$ за полиномиальное число операций ($O(\log^3 N)$), используя $O(\log N)$ кубитов. На практике это означает, что используемые сейчас схемы окажутся не стойкими. В связи с этим появляется необходимость в создании таких алгоритмов ЭЦП, которые были бы устойчивы к взлому с применением квантовых вычислений.

В США существует Национальный институт стандартов и технологий (НИСТ). Этот институт занимается стандартизацией различных технологий, в том числе криптографических алгоритмов. По оценке специалистов НИСТ уже к 2030 году взлом алгоритма RSA, имеющего длину ключа 2000 бит, будет занимать несколько часов. В 2016 году НИСТ инициировал запрос [7] на выдвижение кандидатур постквантовых криптографических алгоритмов. Все предложенные алгоритмы должны были рассматриваться в качестве постквантовых стандартов криптографических механизмов с открытым ключом. Этап сбора заявок на участие в конкурсе закончился в 2017 году. В июне 2020 года были объявлены кандидаты, прошедшие в третий раунд. На данный момент алгоритмы оцениваются и анализируются для рассмотрения на предмет стандартизации по завершении третьего раунда.

Среди постквантовых механизмов особый интерес представляет кодовая криптография [8]. Основой стойкости кодовой криптографии является задача декодирования линейных кодов (общая задача декодирования является NP-трудной) [9]. В 1978 году была представлена первая кодовая криптосистема — криптосистема Мак-Элиса [10]. Основная идея построения криптосистемы состоит в маскировке некоторого линейного кода, имеющего эффективные алгоритмы декодирования, под код, не обладающий видимой алгебраической и комбинаторной структурой. Предполагается, что декодирование такого кода является трудной задачей. Алгоритм не получил широкого распространения на практике из-за большого размера ключа, но в то же время является кандидатом для постквантовой криптографии [11]. Целью постквантовой криптографии является разработка криптографических систем, защищенных от атак, выполненных на квантовых и классических компьютерах, и способных взаимодействовать с существующими протоколами связи и сетями. Квантовые компьютеры не дают каких-либо значительных улучшений в атаках на кодовые криптосистемы, за исключением улучшения поиска секретного ключа методом грубой силы, поэтому можно говорить о том, что алгоритм, на котором построена криптосистема Мак-Элиса, усточив к атакам, в которых бы использовался квантовый вычислитель.

В 1986 году была представлена еще одна кодовая криптосистема - криптосистема Нидеррайтера [0], которая также является кандидатом для постквантовой криптографии. Одним из отличий криптосистемы Нидеррайтера от криптосистемы Мак-Элиса является то, что в криптосистеме Нидеррайтера не используется источник случайности. Благодаря этому криптосистему можно использовать для создания электронной подписи. В 2001 году была разработана схема электронной подписи Куртуа-Файниаса-Сендриера (CFS) [12], базирующаяся на криптосистеме Нидеррайтера. Стойкость этого алгоритма основана на сложности решения задачи синдромного декодирования. Однако существуют трудности в реализации такой схемы подписи. В схеме CFS для построения подписи необходимо декодировать синдром ошибки, вес которой не превосходит значения исправляющей способности кода. Синдром при этом строится как случайный вектор пространства, а вероятность получить декодируемый синдром растет как факториал от исправляющей способности кода. Таким образом вероятность попадания в допустимый синдром очень мала.

В настоящей работе исследовано одно из известных решений проблемы поиска декодируемого синдрома. Решение заключается в использовании кодовых хэш-функций. Метод построения кодовой хеш-функции был предложен в работе [13] в 2005 году. Он основан на структуре Меркла-Дамгора [14], [15], методе построения криптографических хеш-функций, предусматривающем разбиение входных сообщений произвольной длины на блоки фиксированной длины и работающем с ними по очереди с помощью функции сжатия. В 2017 году в работе [16] был предложен модифицированный алгоритм схемы подписи CFS на кодах Гоппы, в котором использовалась кодовая хэш-функция. Именно эта функция была рассмотрена в настоящей работе. В ней была обнаружена ошибка при построении функции сжатия, из-за которой схема подписи становится небезопасной.

В разделе §2 приведены некоторые базовые понятия и определения теории кодирования, которые будут использованы в работе.

Раздел §3 описывает схему подписи CFS. В разделе приведены алгоритмы создания и проверки электронной цифровой подписи на основе криптосистемы Нидеррайтера. Приводятся общие понятия стойкости схемы подписи, а также технические характеристики схемы и следующие из характеристик проблемы.

Раздел §4 описывает известную хэш-функцию, выбранную для работы алгоритма формирования подписи. Раздел содержит анализ параметров хэш-функции, доказательство ее нестойкости, а также влияние небезопасности хэш-функции на всю схему подписи.

## II. Основные понятия и термины

### A. Теория кодирования

**Определение II.1.** [8] *Двоичным линейным $[n, k]$-кодом $C$ над полем $\mathbb{F}_2$ называется произвольное $k$-мерное подпространство линейного пространства $\mathbb{F}_2^n$. Параметр $n$ называется длиной кода, а $k$ — размерностью кода. Значение $r = n - k$ является коразмерность кода $C$.*

**Определение II.2.** [8] *Весом Хемминга вектора $x = x_1x_2...x_n$, $x \in \mathbb{F}_2^n$, называется число его ненулевых координат $x_i$. Вес обозначается через* $\mathrm{wt}(x)$.

**Определение II.3.** [8] *Расстоянием Хемминга* $\mathrm{dist}(x, y)$ *между двумя векторами $x = x_1x_2...x_n$ и $y = y_1y_2...y_n$, $x, y \in \mathbb{F}_2^n$ называется число позиций, в которых соответствующие координаты двух векторов различны. Оно равно* $\mathrm{dist}(x, y) = \mathrm{wt}(x \oplus y)$.

**Определение II.4.** [8] *Наименьшее расстояние между различными кодовыми словами кода $C$ называется минимальным расстоянием кода:*

$$d_{min}(C) = \min_{z \in C,\, z \neq 0} \mathrm{wt}(z).$$

Элементами линейного $[n, k]$-кода $C$ являются кодовые слова. Линейный код может быть определен либо порождающей матрицей, либо матрицей проверки на четность (проверочной матрицей).

**Определение II.5.** [8] *Порождающая матрица $G$ линейного кода $C$ представляет собой матрицу размера $k \times n$, строками которой являются базисные векторы кода $C$. Любая линейная комбинация базисных векторов является кодовым вектором.*

Если взять некоторое сообщение $m = (m_1, ...m_k)$, то соответствующее ему кодовое слово $c = (c_1, ...c_n)$, принадлежащее коду $C$ можно вычислить по формуле $c = m \cdot G$, где $G$ — порождающая матрица кода $C$.

**Определение II.6.** [0] *Проверочной матрицей $H$ для линейного кода $C$ называется матрица размера $(n-k) \times n$, такая, что векторы $c \in \mathbb{F}_2^n$, являющиеся решениями уравнения $H \cdot c^T = 0$, являются кодовыми словами $C$.*

**Определение II.7.** [8] *Пусть $H$ — проверочная $(r \times n)$-матрица кода $C$, а $y \in \mathbb{F}_2^n$ — произвольный вектор. Тогда синдромом вектора $y$ называется вектор $s \in \mathbb{F}^r$: $s = y \cdot H^T$.*

**Определение II.8.** [17] *Алгоритм декодирования синдрома $dec()$ для линейного кода $C$, определяемого проверочной матрицей $H$ размера $r \times n$ — это процесс, который позволяет для заданного вектора $s \in \mathbb{F}_2^r$ найти вектор $e = dec(s) \in \mathbb{F}_2^n$, удовлетворяющий уранению $e \cdot H^T = s$, $wt(e) \leq t$, где $t$ — количество ошибок, которое способен исправить декодер.*

Вектор $e$ рассматривается как ошибка, а элемент $s \in \mathbb{F}_2^r$ является его синдромом.

## III. Схема электронной подписи CFS

**Определение III.1.** [18] *Полиномиальный вероятностный алгоритм — это алгоритм, который работает за полиномиальное время и может использовать источник случайности для получения недетерминированных результатов. Полиномиальным считается время, оцениваемое сверху выражением вида $C \cdot N^K$, где $C$ и $K$ — некоторые константы, а $N$ — размер входа алгоритма.*

**Определение III.2.** [19] *Схема цифровой подписи состоит из тройки алгоритмов* $\Sigma = (\mathbf{KGen}, \mathbf{Sign}, \mathbf{Verify})$*:*

- **KGen***: полиномиальный вероятностный алгоритм генерации ключей принимает на вход параметр безопасности $1^\lambda$ и выдает пару $(pk, sk)$ — открытый и секретный ключи.*
- **Sign***: полиномиальный вероятностный алгоритм генерации подписи принимает сообщение $m$ и закрытый ключ $sk$ и выводит подпись $\sigma$ сообщения $m$.*
- **Verify***: полиномиальный детерминированный алгоритм проверки подписи принимает в качестве входных данных открытый ключ $pk$, сообщение $m$ и подпись $\sigma$ и выводит бит, обозначающий принятие или отклонение подписи.*

### A. Описание алгоритма

CFS [12] — это схема цифровой подписи, опубликованная в 2001 году. Протокол ЭЦП основан на криптосистеме Нидеррайтера [0] на кодах Гоппы, и его стойкость сводится к сложности решения задачи декодирования синдрома и задачи различимости двоичных кодов Гоппы от случайных.

Введем следующие параметры для схемы подписи, которые позволят представить ее в соответствии с нотаций III.2:

- $\Lambda$ — множество параметров безопасности.
- $C = \{\mathcal{C}_\lambda\}_{\lambda \in \Lambda}$ — некоторое семейство линейных $[n, k]$-кодов Гоппы над полем $\mathbb{F}_2$, в котором каждый код $\mathcal{C} \in C$ имеет эффективный алгоритм декодирования $D$, исправляющий $t$ ошибок.
- $Sample(\lambda)$ — эффективный алгоритм, который по значению $\lambda \in \Lambda$ выдает проверочную матрицу $H_\lambda$ кода $\mathcal{C}_\lambda$, число ошибок $t_\lambda$ и алгоритм декодирования $D_\lambda$, исправляющий $t_\lambda$ ошибок.
- Хэш-функция $h : \{0, 1\}^* \to \mathbb{F}_2^{n-k}$.

Ниже представлен псевдокод алгоритма генерации ключей подписи:

**Algorithm III.1: KGen:**

**Data:** $\lambda$ — значение параметра.
**Result:** Секретный ключ $sk = (S, H, P, D)$;
Открытый ключ $pk = (t, H_{pub})$.

$(H, D, t) \leftarrow Sample(\lambda)$;
$S \xleftarrow{R} GL(n - k, \mathbb{F}_2)$; // $S$ - случайная невырожденная матрица над полем $\mathbb{F}_2$ размера $(n - k) \times (n - k)$;
$P \xleftarrow{R} S_n$; // $P$ - случайная перестановочная матрица над полем $\mathbb{F}_2$ размера $n \times n$;
$H_{pub} = S \cdot H \cdot P$;

Пусть имеется сообщение $m$, которое необходимо подписать.

**Алгоритм генерации подписи:**

1) Выбрать хэш-функцию $h()$, выдающую $n-k$ символов на выходе. Результат этой функции будет представляться в виде синдрома и декодироваться.
2) Вычислить хэш $s = h(m)$.
3) Вычислить $s_i = h(s|i)$ для каждого $i = 0, 1, 2, ....$
4) Найти $i_0$ — наименьшее значение $i$ такое, что $s_i$ можно декодировать.
5) Декодировать $s_i$. Пусть $z$ — результат декодирования, т.е. $H \cdot z^T = s_{i_0}$.
6) Подписью документа является пара $(z, i_0)$.

Так как вес вектора $z$ равен значению $t$, в статье [12] для уменьшения длины подписи предлагается проиндексировать все слова веса $t$ и использовать в качестве подписи не сам вектор $z$, а его индекс. Индексирование проводят по формуле $I_z = 1 + \binom{i_1}{1} + \binom{i_2}{2} + ... + \binom{i_t}{t}$, где $i_1 < i_2 < ... < i_t$ — позиции ненулевых битов $z$.

Ниже представлен псевдокод алгоритма генерации подписи с учетом сжатия.

**Algorithm III.2: Sign**

**Data:** Секретный ключ $sk = (S, H, P, D)$, сообщение $m \in \mathbb{F}_2^n$
**Result:** Подпись $\sigma$

$s \leftarrow h(m)$;
$i \leftarrow 0$;
**while** *true* **do**
  $s_i \leftarrow S^{-1} \cdot h(s \parallel i)$;
  **if** $D(s_i) \neq error$ **then**
    $e \leftarrow D(s_i)$;
    break;
  **else**
    $i \leftarrow i + 1$;
$e \leftarrow e \cdot P$;
// ниже происходит сжатие подписи
$k \leftarrow 1$;
$I_e \leftarrow 1$ ;
**for** $j \leftarrow 1$ **to** $n$ **do**
  **if** $e_j = 1$ **then**
    $I_e \leftarrow I_e + \binom{j}{k}$;
    $k \leftarrow k + 1$
$\sigma \leftarrow (i, I_e)$;
**return** $\sigma$

Обратное преобразование из $I_z$ в $z$ выполняется с помощью жадного алгоритма [20]. Сначала по значениям $I_z$ и $t$ необходимо найти максимальное значение $i_t$, для которого $\binom{i_t}{t} \leq I_z - 1$. Затем необходимо найти максимальное значение $i_{t-1}$, для которого $\binom{i_{t-1}}{t-1} \leq I_z - 1 - \binom{i_t}{t}$. Получаем итерационный процесс, на выходе которого будут получены значения $i_1, i_2, ...i_t$.

Шаги алгоритма проверки подписи будут следующими:

1) Восстановить $z$ из индекса $I_z$.
2) Вычислить $s_1 = H \cdot z^T$, где $H$ — открытый ключ.
3) Вычислить $s_2 = h(h(m)|i_0)$, где $h()$ — публичная хэш-функция.
4) Сравнить $s_1$ и $s_2$. Подпись верна, если эти значения совпадают.

Опишем полученный алгоритм проверки подписи с помощью псевдокода:

**Algorithm III.3: Verify**

**Data:** Открытый ключ $pk = (t, H_{pub})$, подпись $\sigma = (i, I_e)$, сообщение $m \in \mathbb{F}_2^n$
**Result:** $true$, $false$

$e \leftarrow \{0\}^n$ ; // $e = (e_1, e_2, ...e_n)$
$I_z \leftarrow I_z - 1$;
**for** $k \leftarrow 0$ **to** $t - 1$ **do**
  $ind \leftarrow 1$;
  **while** $\binom{ind}{t-k} \leq I_z$ **do**
    $ind \leftarrow ind + 1$;
  $ind \leftarrow ind - 1$ ;
  $e_{ind} \leftarrow 1$;
  $I_z \leftarrow I_z - \binom{ind}{t-k}$;
**if** $wt(e) > t$ **then**
  **return** $error$
$a \leftarrow h(h(m) \parallel i)$;
$b \leftarrow H_{pub} \cdot e^T$;
**if** *a=b* **then**
  **return** $true$
**else**
  **return** $false$

**Доказательство корректности:**

1) $a = h(m \parallel i) = d_i$
2) Так как $S^{-1} \cdot d_i$ — декодируемый синдром, существует вектор ошибки $e$, для которого выполняется: $S^{-1} \cdot d_i = H \cdot e^T$.
3) $d_i = S \cdot H \cdot e^T = S \cdot H \cdot P \cdot P^{-1} \cdot e^T = H_{pub} \cdot u^T$.
4) $H_{pub} \cdot u^T = b$. Получили равенство $a = b$, при условии $H_{pub} = S \cdot H \cdot P$.

### *B. Общие понятия стойкости схемы подписи*

**Определение III.3.** [21] *Атака — совокупность предположений о противнике и об информации, получаемой противником в ходе взаимодействия с исследуемым криптографическим объектом, в рамках которых анализируется стойкость этого объекта.*

**Определение III.4.** [21] *Угроза — задача, стоящая перед противником.*

В работе [22], опубликованной в 1987 году, авторы вводят понятие стойкости схемы электронной подписи в следующем виде. Рассматриваются два основных типа атак:

- Атаки с использованием открытого ключа, в которых противник знает только открытый ключ подписывающего лица.
- Атаки на основе сообщений, когда противник может изучить некоторые подписи, соответствующие либо

известным, либо выбранным сообщениям, прежде чем он попытается взломать схему.

Среди атак на основе сообщений выделяют следующие типы ($A$ — подписывающий):

- Атака на основе известных сообщений: противнику предоставляется доступ к подписям для набора сообщений $m_1, ..., m_t$. Сообщения известны противнику, но не выбираются им.
- Атака на основе выбранных сообщений: противнику разрешается получить от $A$ действительные подписи для выбранного списка сообщений $m_1, ..., m_t$, прежде чем противник попытается взломать схему подписи.
- Атака на основе адаптивно выбранных сообщений: противнику разрешено использовать $A$ в качестве «оракула»; противник может запрашивать подписи сообщений, зависящих от ранее полученных подписей.

Вместе с описанными атаками рассматриваются следующие угрозы:

- Полное вскрытие: получение противником закрытого ключа $A$, что означает полный взлом схемы подписи.
- Универсальная подделка: нахождение эффективного алгоритма подписи, аналогичного алгоритму подписи $A$, позволяющего подделывать подписи для любых сообщений.
- Выборочная подделка: возможность подделывать подписи для сообщений, выбранных противником.
- Экзистенциальная подделка: возможность получения допустимой подписи для какого-то сообщения, не выбираемого противником.

Схема цифровой подписи считается безопасной [23], если доказано, что она не подвержена экзистенциальной подделке при атаке с выбранным сообщением (EUF-CMA). В работе [24] Даллотом было доказано, что схема подписи CFS не является устойчивой к экзистенциальной подделке при атаке с выбранным сообщением из-за того, что высокоскоростной код Гоппы можно отличить от случайного кода [25] (скорость кодирования становится высокой при выборе малого значения $t$). В той же работе предложена модифицированная схема подписи mCFS и доказана ее устойчивость в нотации EUF-CMA.

**Определение III.5.** [26]*Рассмотрим схему цифровой подписи* $\Sigma = (\mathbf{KGen}, \mathbf{Sign}, \mathbf{Verify})$. *Схема* $\Sigma$ *не подвержена экзистенциальной подделке при атаке с выбранным сообщением (EUF-CMA) тогда и только тогда, когда для любого полиномиального вероятностного алгоритма* $\mathcal{A}$ *вероятность успеха следующего эксперимента незначительна:*

$$\mathbf{P}_{\text{adv}}^{\text{EUF-CMA}}(\mathcal{A}) =$$

$$\Pr\left[\begin{array}{c|c} \mathbf{Verify}^{pk}(M^*, \sigma^*) = 1 \\ \wedge\, M^* \notin \Omega \end{array} \middle| \begin{array}{c} (pk, sk) \leftarrow \mathbf{KGen}(1^\lambda), \\ (M^*, \sigma^*) \leftarrow \mathcal{A}^{\mathbf{Sign}^{sk}}(pk) \end{array}\right].$$

*Здесь* $q \in \mathbb{N}$ *— количество запросов к подписывающему оракулу* $\boldsymbol{Sign}^{sk}$*, выполненных* $\mathcal{A}$*, и* $\Omega = \{M_i\}_{i=1,...,q}$ *— множество сообщений, предоставленных оракулу для подписи.*

Сформулируем задачу экзистенциальной подделки для схемы CFS, описанной в разделе 3.1. Не нарушая общности, опустим алгоритм сжатия вектора $e$ и в качестве выхода задачи будем использовать сам вектор $e$ веса, не превышающего значение $t$.

**Задача экзистенциальной подделки подписи:**
*Вход:* $m \in \mathbb{F}_2^*$, $pk = (t, H_{pub})$.
*Выход:* $e \in \mathbb{F}_2^n$: $H_{pub} \cdot e^T = h(h(m)|i), \text{wt}(e) \leq t$.

В контексте этой задачи достаточно найти такое $m' \neq m$, для которого $h(m') = h(m)$ для получения подделки. Исходя из описанной задачи и определения EUF-CMA можно сформулировать следующее утверждение:

**Утверждение III.1.** *Устойчивость схемы подписи CFS к экзистенциальной подделке при атаке с выбранным сообщением основана на двух предположениях: сложности задачи декодирования кода* $C$*, на котором построена схема подписи, и стойкости используемой в алгоритме хэш-функции.*

### *C. Характеристики схемы подписи*

Коды Гоппы, на которых строится схема, впервые были представлены в 1970 году в работе [27]. Приведем ниже небольшое введение в эти коды.

Пусть $g(x) \in GF(2^m)[x]$ — многочлен степени $t$ над полем $GF(2^m)$. И пусть $L = \{\alpha_1, ..., \alpha_n\}$ — множество элементов $GF(2^m)$, таких, что $g(\alpha_i) \neq 0$. Занумеруем координаты произвольного вектора $a \in (GF(2^m))^n$ элементами $L$ следующим образом: $a = (a_{\alpha_1}, ..., a_{\alpha_n})$.

**Определение III.6.** [28] *Кодом Гоппы* $\Gamma(L, g)$ *с порождающим многочленом* $g(x)$ *и определяющим множеством* $L = \{\alpha_1, ..., \alpha_n\}$ *называется множество двоичных векторов* $a = (a_{\alpha_1}, ..., a_{\alpha_n})$*, таких, что* $H \cdot a^T = 0$*, где*

$$H = \begin{pmatrix} \frac{1}{g(\alpha_1)} & \cdots & \frac{1}{g(\alpha_n)} \\ \frac{\alpha_1}{g(\alpha_1)} & \cdots & \frac{\alpha_n}{g(\alpha_n)} \\ \cdots & \cdots & \cdots \\ \frac{\alpha_1^{t-1}}{g(\alpha_1)} & \cdots & \frac{\alpha_n^{t-1}}{g(\alpha_n)} \end{pmatrix}.$$

Матрица $H$ по определению является проверочной матрицей кода $\Gamma(L, g)$.

**Теорема III.1.** *[28] Пусть* $g(x) \in GF(2^m)[x]$ *— многочлен степени* $t$*, свободный от квадратов и не имеющий корней в поле* $GF(2^m)$*, и* $\Gamma(L, g)$ *— соответствующий код Гоппы длины* $n$ *и размерности* $k$*. Если* $t < 2^{(m/2)-1}$*, то* $k = n - mt$*.*

В оригинальной статье рассматривается семейство двоичных кодов Гоппы длины $n = 2^m$, исправляющих $t$-ошибок. Эти коды имеют размерность $k = n - tm$. Открытый ключ криптосистемы — матрица размера $(n-k) \times n = tm \times 2^m$. Потребительские характеристики схемы CFS, построенной на таких кодах, представлены в таблице ниже.

| | |
|---|---|
| Сложность вычисления подписи | $t!t^2m^3$ |
| Длина подписи | $2^m$ |
| Сложность подтверждения подписи [12] | $t^2m$ |
| Размер открытого ключа | $tm2^m$ |
| Сложность лучшей атаки декодирования [29] | $2^{tm(1/2+o(1))}$ |
| Сложность лучшей структурной атаки [30] | $tm2^{m(t-2)}$ |

Сложность вычисления подписи следует из двух утверждений:

**Утверждение III.2.** *[12] Пусть для некоторого $m \in \mathbb{Z}$ задан двоичный код Гоппы $\Gamma$, исправляющий $t$ ошибок, с длиной $n = 2^m$ и размерностью $k = n - mt$. Тогда вероятность успешной генерации подписи в схеме CFS, построенной на коде $\Gamma$, будет равна $1/t!$.*

**Утверждение III.3.** *[12] Пусть для некоторого $m \in \mathbb{Z}$ задан двоичный код Гоппы $\Gamma$, исправляющий $t$ ошибок, с длиной $n = 2^m$ и размерностью $k = n - mt$. Сложность алгоритма синдромного декодирования Берлекэмпа-Месси для кода $\Gamma$ в поле $\mathbb{F}_2$ составляет $\mathcal{O}(t^2m^3)$.*

В оригинальной статье предлагается использовать следующие параметры для схемы подписи CFS: $m = 16,\ t = 9$. Однако в материалах конференции ASIACRYPT 2009 авторы [31] показывают атаку на схему CFS, основаную на «несбалансированной» атаке дней рождения, предложенной Даниэлем Блейхенбахером. Данная атака на схему CFS доказывает нестойкость схемы, построенной для оригинальных параметров, в связи с чем значения $m$ и $t$ были изменены. Для уровня безопасности, требующего более $2^{80}$ бинарных операций, в [31] предложены новые параметры: $m = 21$ и $t = 10$; $m = 19$ и $t = 11$; или $m = 15$ и $t = 12$.

Из данных, представленных в таблице выше, можно вывести следующую проблему построения схемы подписи: большая сложность вычисления подписи, связанная с итерационным поиском декодируемого синдрома. Решение этой проблемы было предложено в работе [16]. Оно заключается в использовании кодовой хэш-функции, вместо случайной. Выходом функции сжатия такой хэш-функции являются только декодируемые синдромы, что позволяет сократить количество итераций до одной.

## IV. Функция хэширования

**Определение IV.1.** [32] *Хеш-функция — это функция, которая отображает строки произвольной длины в строки фиксированной длины, называемые хеш-значениями.*

**Определение IV.2.** [32] *Коллизией для хэш-функции $h : \mathbb{F}_2^p \to \mathbb{F}_2^r$ называется пара векторов $x_1, x_2 \in \mathbb{F}_2^p$ для которых выполняется соотношение $h(x_1) = h(x_2)$ при $x_1 \neq x_2$.*

### A. Алгоритм оригинальной хэш-функции

Идея построения хэш-функции, которая выдает только декодируемые синдромы, была представлена в статье [16] в 2017 году. Рассмотрим предложенную функцию $h_c$ подробнее. Стандартный способ получения синдрома — умножение проверочной матрицы $H$ на вектор ошибок $e$. На этом основано построение хэш-функции $h_c$. Хэш-функция является итерационной. Сообщение $m$, которое необходимо подписать, делится на части, каждая часть размера $p - r$ бит, где $p$ — длина входного вектора для функции сжатия $f()$. Обозначим каждую часть за $m_i$ ($i = 1,\ 2,\ ...,\ N = \lceil \frac{\text{len}(m)}{p-r} \rceil$). Для первой итерации формируется случайный начальный вектор IV длины $r$.

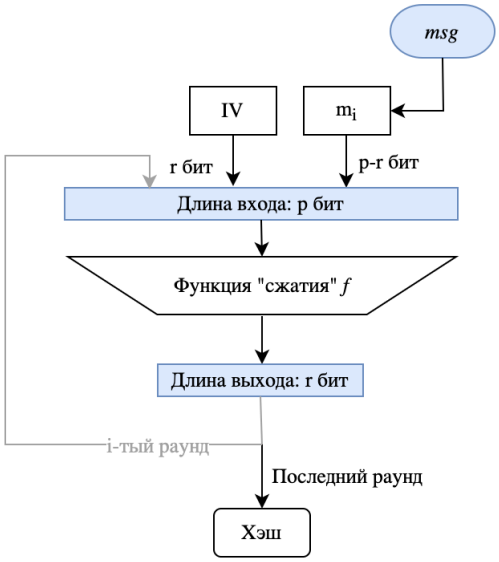


Рис. 1.

Алгоритм хэширования представлен на Рис.1 и выглядит следующим образом:

$$F_1 = f(\text{IV}|m_1),$$
$$F_2 = f(F_1|m_2),$$
$$...$$
$$F_i = f(F_{i-1}|m_i),$$
$$...$$
$$F_N = f(F_{N-1}|m_N).$$

$F_N$ — итоговое значение хэш-функции $h_c(m)$, $h_c : \mathbb{F}_2^* \to \mathbb{F}_2^r$. Функция сжатия $f()$ преобразует векторы длины $p$ в векторы длины $r$, $f : \mathbb{F}_2^p \to \mathbb{F}_2^r$.

Матрица $H_{pub}$ — публичный ключ криптосистемы, следовательно, ее можно использовать в качестве параметра в хэш-функции. Она имеет размер $r \times n$. Основная идея заключается в том, чтобы разделить матрицу $H_{pub}$

по столбцам на некоторое количество блоков, а затем суммировать определенные столбцы матрицы, по одному из каждого блока. Это будет равносильно умножению матрицы на вектор. Количество блоков при этом равно весу вектора, на который умножается матрица. В нашем случае он должен быть близок к значению корректирующей способности кода $t$, чтобы получать декодируемые синдромы.

Число блоков обозначим через $w$, $w \leq t$. Матрица $H_{pub}$ делится на $w$ блоков: $H_{pub} = [H_{pub_1},\ H_{pub_2},\ ...\ H_{pub_w}]$. Каждый блок $H_{pub_i}$ имеет размер $r \times l$ и следующий вид: $H_{pub_i} = [h_{(i-1)\cdot\ell+1},\ \ h_{(i-1)\cdot\ell+2},\ \ ...,\ \ h_{(i-1)\cdot\ell+l}]$, $i = 1, 2..., w$, где $h_1, h_2, ...h_n$ — столбцы матрицы $H_{pub}$ (Рис. 2).

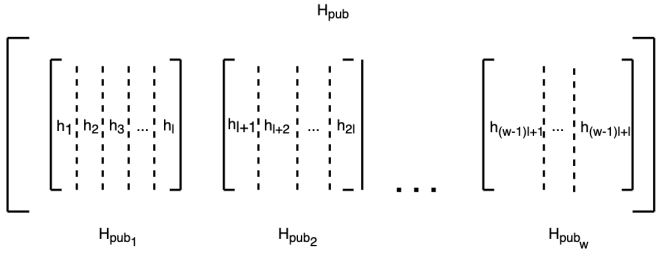


Рис. 2.

Пусть $x$ — вектор, к которому применяется функция сжатия $f()$, $x \in \mathbb{F}_2^p$. Значение $p$ берется равным $w \cdot \log_2 l$. Тогда вектор $x$ делится нацело на $w$ блоков: $x = (x_1,\ x_2,\ ...,\ x_w)$. При этом каждый блок $x_i \in \mathbb{F}_2^{\log_2 l}$, и его можно представить в виде десятичного числа от 0 до $l$. Значение функции сжатия вычисляется по следующей формуле:

$$f(x) = \sum_{i=1}^{w} h_{(i-1)l+x_i+1}.$$

Здесь суммируются $w$ разных столбцов матрицы $H_{pub}$. Если брать значение $w$ меньшим или равным корректирующей способности кода $t$, то в качестве выхода функции сжатия на каждой итерации получается значение декодируемого синдрома.

В статье [16] доказана теорема, подтверждающая корректность построения хэш-функции:

**Теорема IV.1.** *Вычисление функции сжатия $f$ эквивалентно вычислению синдрома вектора длины $n$ и веса $w$, т. е. для любого вектора $x \in \mathbb{F}_2^p$ найдется вектор $c$, для которого $H \cdot c^T = f(x)$, $wt(c) = w$.*

Сложность обращения приведенной хэш-функции основана на проблеме синдромного декодирования, считающейся NP-трудной [9].

### *B. Ошибка в хэш-функции*

В статье [16] авторы предлагают использовать коды Гоппы для построения хэш-функции. Эти коды имеют следующие параметры:

- $n = 2^m$;
- $k = n - mt$;
- $r = n - k = mt$.

Авторы берут некое $w$, чтобы разделить матрицу на $w$ блоков и сложить столбцы матрицы, по одному столбцу из каждого блока. Длина полученного после всех суммирований вектора — $r$ . Какой именно столбец необходимо брать из каждого блока, авторы определяют по вектору $x$ — части подписываемого сообщения длины $p$. Значение $n$ должно делиться на $w$. Так как $n = 2^m$, значение $w$ будет равным некоторому $2^{m'}$, $m' < m$. Полученное от деления $n$ на $w$ число $l$ — число столбцов в каждом блоке, $l = n/w = 2^{m-m'}$.

Входной вектор функции сжатия также делится на $w$ блоков, но уже по $\log_2 l$ элементов в каждом блоке. Это делается для того, чтобы, переведя каждый блок в десятичную систему, получить номер одного из $l$ столбцов блока матрицы $H$. Если $p$ — длина входного вектора, то $p = w \log_2 l = w \cdot (m - m')$.

В алгоритме подписи значение $w$ берется меньшим или равным $t$ ($w \leq t$), чтобы полученный на выходе синдром можно было декодировать. Авторы утверждают [16], что для параметров функции сжатия $f : \mathbb{F}_2^p \to \mathbb{F}_2^r$ выполняется соотношение: $p > r$.

**Утверждение IV.1.** *Пусть $C$ — произвольный код Гоппы, исправляющий $t$ ошибок, размерности $n-tm$ и длины $n = 2^m$. Пусть $f$ — предложенная функция сжатия $f : \mathbb{F}_2^p \to \mathbb{F}_2^r$, $f(x) = \sum_{i=1}^{w} h_{(i-1)l+x_i+1}$. В ней $h_j$ — столбцы некоторой проверочной матрицы $H$ размера $r \times n$ для произвольного кода Гоппы $C$, $x_i$ — координаты входного вектора $x$ длины $p$, значение $w$ не превосходит $t$. Тогда для значений $p$ и $r$ выполняется соотношение $p < r$.*

*Доказательство.* Код $C$ для построения функции хэширования задается своей проверочной матрицей $H$. Для кодов Гоппы имеем следующие параметры:

- длина кода: $n = 2^m$ ;
- корректирующая способность кода: $t$;
- размерность кода: $k = n - mt$;
- длина выходной последовательности функции сжатия: $r = n - k = mt$;
- количество блоков, на которые будет разделена матрица $H$: $w$, $w|n, w \leq t$;
- размер каждого блока: $l = n/w = 2^{m-m'}$;
- длина входной последовательности функции сжатия: $p$, $p = w \log_2 l = w \cdot (m - m')$;
- функция сжатия: $f : \mathbb{F}_2^p \to \mathbb{F}_2^r$ .

Проведем доказательство от противного:

1) Пусть $r < p$. Тогда $mt < w \cdot (m - m')$, т.к. $r = n - k = mt$.
2) $m' = \log_2 w \Rightarrow mt < mw - w \log_2 w$ .
3) Так как $w \leq t$, получаем $mw \leq mt \Rightarrow mw < mw - w \log_2 w$.
4) Вычитая $mw$ слева и справа, получим $0 < -w \log_2 w \ \Rightarrow\ w \log_2 w < 0$ . Если принять, что $w > 0$, неравенство верно только при $w < 1$. Тогда $w \in (0,\ 1)$, а значит не является целым числом. Так как $w$ — это количество блоков матрицы, оно должно быть целым числом. Получено противоречие.

□

### *C. Поиск коллизий*

**Утверждение IV.2.** *Пусть дана функция сжатия $f$ : $\mathbb{F}_2^p \to \mathbb{F}_2^r$, $f(x) = \sum_{i=1}^{w} h_{(i-1)l+x_i+1}$. В ней $h_j$ — столбцы*

*некоторой проверочной матрицы $H$ размера $r \times n$ для произвольного кода $C$, $x_i$ — координаты случайного вектора $x$ длины $p$, значение $w$ не превосходит $t$. Если для этой функции $p < r$, то построенная по ней хэш-функция $h$ не является безопасной, так как для нее за полиномиальное время находятся коллизии.*

*Доказательство.* Рассмотрим алгоритм хэширования функции $h$:

$$F_1 = f(\text{IV}|m_1),$$
$$F_2 = f(F_1|m_2),$$
$$...$$
$$F_i = f(F_{i-1}|m_i),$$
$$...$$
$$F_N = f(F_{N-1}|m_N).$$

1. Хэш-функция является итерационной, следовательно, выход функции сжатия $f$ предыдущей итерации, конкатенированный с частью сообщения, будет подаваться на вход функции сжатия $f$ следующей итерации.
2. Так как $p < r$, перед каждой новой итерацией выход предыдущей будет обрезаться перед подачей. Вне зависимости от того, как именно это будет происходить, этот процесс повлечет за собой возможность построения коллизий. Так как функция сжатия построена на матричных операциях, уменьшение выходного значения эквивалентно "удалению"из матрицы $H$ части строк, что приводит к линейной зависимости столбцов. Линейная зависимость столбцов при этом обеспечивает возможность построения коллизий.
3. Выбор способа уменьшения размера выходного значения функции сжатия влияет только на номера строк, которые будут "удалены".
4. Для построения коллизии $h(m) = h(m'), m \neq m'$ необходимо выполнить следующие шаги:
   1) Пусть $y$ — вектор, полученный после применения функции сжатия к первой части $m$ на первой итерации алгоритма функции хэширования. В этом векторе перед подачей на вторую итерацию будут удалены определенные элементы.
   2) В матрице $H$ необходимо найти такие столбцы (по одному из каждого блока), сумма которых совпадала бы с вектором $y$ в координатах, которые не будут удалены.
   3) Номера найденных столбцов необходимо перевести в двоичную запись и подать в качестве IV и первой части $m'$ на первой итерации функции хэширования.
   4) Остальную часть $m'$ необходимо взять эквивалентной $m$.

В этом случае для разных входных значений будут получаться одинаковые выходные значения функции сжатия, что повлечет за собой одинаковые выходные значения всей функции хэширования. □

Опишем процесс поиска коллизий на конкретном примере. Возьмем код Гоппы [33]. Проверочная матрица для этого кода:

$$\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0
\end{bmatrix}$$

Параметры кода:

- $n = 2^4$;
- $m = 4$;
- $k = n - mt = 8$;
- $r = n - k = 8$;
- $t = 2$.

Тогда параметры хэш-функции будут следующими:

- $w = 2$;
- $l = n/w = 8$;
- $p = w \cdot log_2 l = 6$.

Для уменьшения длины выходного значения функции сжатии будем отрезать от выхода функции сжатия часть, оставляя $p-q$ битов, где $q$ — количество битов сообщения, участвующих в итерации. Такой способ уменьшения не ограничивает общности, так как влияет только на номера строк, которые будут "удалены"из проверочной матрицы. Пусть $q = 3$, $p - q = 3$. Для первой итерации возьмем случайный двоичный инициализирующий вектор IV длины 3: $(i_1\ i_2\ i_3)$. Пусть он равен $(0\ 0\ 1)$. Будем хэшировать случайное двоичное сообщение $m = (m_1, m_2)$, где $m_1, m_2$ — двоичные векторы длины 3, $m = (x_1\ x_2\ x_3\ x_4\ x_5\ x_6)$. Пусть $m = (1\ 0\ 1\ 1\ 1\ 1)$. Рассмотрим процесс хэширования $m$:

- **1 итерация:** $(\text{IV}|m_1) = (0\ 0\ 1|1\ 0\ 1) = (0\ 0\ 1\ 1\ 0\ 1)$. Так как $w = 2$, $001 \to 1$, $101 \to 5$, из первого блока будет взят первый столбец, а из второго пятый (отсчет с нулевого столбца в каждом блоке). После суммирования столбцов получим $y = (0\ 0\ 1\ 1\ 0\ 1\ 0\ 1)$. Отрежем от вектора последние 5 элементов. Получим для входа на следующей итерации вектор $y' = (0\ 0\ 1)$.
- **2 итерация:** $(y'|m_2) = (0\ 0\ 1|1\ 1\ 1) = (0\ 0\ 1\ 1\ 1\ 1)$. Так как $001 \to 1$, $111 \to 7$, из первого блока будет взят первый столбец, а из второго седьмой. Получим $y = (0\ 1\ 1\ 1\ 1\ 1\ 1\ 1)$. Это значение будет являться выходом функции хэширования.

Чтобы найти коллизию $m' = (x'_1\ x'_2\ x'_3\ x'_4\ x'_5\ x'_6)$ для $m$, необходимо в матрице $H$ найти два столбца из разных блоков, которые при суммировании дадут вектор $(0\ 0\ 1\ *\ *\ *\ *\ *)$, отличные от столбцов из первой итерации. В нем первые три координаты совпадают с первыми тремя координатами вектора $y$, полученного на первой итерации хэширования $m$. Подходят четвертый столбец из первого блока и третий столбец из второго. Тогда $4 \to 100$, $3 \to 011$. Значения $x'_4$, $x'_5$, $x'_6$ возьмем равными $x_4$, $x_5$, $x_6$, соответственно. В этом случае $\text{IV}' = (1\ 0\ 0)$, $m' = (0\ 1\ 1\ 1\ 1\ 1)$. Проверим, что в результате получается коллизия. Рассмотрим процесс хэширования $m'$:

- **1 итерация:** $(\text{IV'}|m'_1) = (1\ 0\ 0|0\ 1\ 1) = (1\ 0\ 0\ 0\ 1\ 1)$. Из первого блока будет взят четвертый столбец, а из второго третий (отсчет с нулевого столбца в каждом блоке). После суммирования столбцов получим

$y = (0\ 0\ 1\ 0\ 1\ 1\ 1\ 0)$. Отрежем от вектора последние 5 элементов. Получим для входа на следующей итерации вектор $y' = (0\ 0\ 1)$.

- **2 итерация:** $(y'|m'_2) = (0\ 0\ 1|1\ 1\ 1) = (0\ 0\ 1\ 1\ 1\ 1)$. На вход подается тот же самый вектор, который подавался на вход второй итерации при хэшировании $m$, значит и выход будет таким же, следовательно, получена коллизия.

### *D. Влияние найденной ошибки на стойкость CFS*

Рассмотрим подробнее алгоритм формирования подписи схемы CFS (без сжатия).

**Algorithm IV.1: Sign**

**Data:** Секретный ключ $sk = (S, H, P, D)$, сообщение $m \in \mathbb{F}_2^n$
**Result:** Подпись $\sigma$

$s \leftarrow h(m)$;
$i \leftarrow 0$;
**while** *true* **do**
  $s_i \leftarrow S^{-1} \cdot h(s \parallel i)$;
  **if** $D(s_i) \neq error$ **then**
    $e \leftarrow D(s_i)$;
    break;
  **else**
    $i \leftarrow i + 1$;
$e \leftarrow e \cdot P$;
$\sigma \leftarrow (i, e)$;
**return** $\sigma$

Из утверждения III.1 следует, что, если алгоритм хэш-функции не является стойким к построению коллизий, то вся схема подписи не является устойчивой к экзистенциальной подделке при атаке с выбранным сообщением.

Опишем подробнее, почему это верно для приведенной схемы. Выход функции хэширования $h()$ после умножения слева на матрицу $S^{-1}$ (матрица $S$ — случайная невырожденная матрица, полученная при генерации ключей схемы) проверяется на декодируемость. Так как функция $h()$ всегда выдает декодируемые синдромы, подписью будет являться вектор ошибки, вычисленный при первой попытке декодирования, умноженный справа на матрицу $P$ (матрица $P$ — случайная перестановочная матрица, полученная при генерации ключей схемы).

Так как для функции $h()$ возможно построение коллизий, переменной $s_i$ может быть присвоено одно и то же значение при разных входных сообщениях $m_1$ и $m_2$. Исходя из этого, для сообщений $m_1$ и $m_2$ будут найдены одинаковые вектора ошибок $e_1$ и $e_2$, а значит будут получены одинаковые подписи.

Таким образом вся схема подписи, построенная с использованием предложенной хэш-функции, является небезопасной.

## V. Выводы

В работе была рассмотрена схема подписи CFS, выделены ее ключевые недостатки. В их числе большая сложность построения подписи, связанная с поиском декодируемого синдрома.

Для решения проблемы в работе был изучен вопрос выбора хэш-функции для схемы электронной цифровой подписи. Для быстрого декодирования выходного значения функции хэширования необходимо использовать кодовые функции хэширования. Одна из таких функций была рассмотрена в работе. При изучении в ней была найдена ошибка, из-за которой для хэш-функции становится возможным нахождение коллизий. Успешный поиск был продемонстрирован на примере кодов Гоппы с малыми параметрами. Также в работе было показано влияние найденной ошибки на всю схему подписи CFS.

## Библиография


[1] W. Diffie и M. Hellman, «New directions in cryptography,» *IEEE Transactions on Information Theory*, т. 22, № 6, с. 644—654, 1976. doi: 10.1109/TIT.1976.1055638.

[2] R. L. Rivest, A. Shamir и L. Adleman, «A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,» *Commun. ACM*, т. 21, № 2, с. 120—126, февр. 1978. doi: 10.1145/359340.359342. url: https://doi.org/10.1145/359340.359342.

[3] M. O. Rabin, «DIGITALIZED SIGNATURES AND PUBLIC-KEY FUNCTIONS AS INTRACTABLE AS FACTORIZATION,» USA, тех. отч., 1979.

[4] R. C. Merkle, «One Way Hash Functions and DES,» в *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, ред., New York, NY: Springer New York, 1990, с. 428—446.

[5] H. M. Grumbling E., *Quantum Computing: Progress and Prospects*. 2019.

[6] P. Shor, «Algorithms for quantum computation: discrete logarithms and factoring,» в *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, с. 124—134. doi: 10.1109/SFCS.1994.365700.

[7] N. I. of Standards и T. (NIST), «Announcing Request for Nominations for Public-Key Post-Quantum Cryptographic Algorithms,» 2016.

[8] H. Singh, «Code based Cryptography: Classic McEliece,» *CoRR*, т. abs/1907.12754, 2019. url: http://arxiv.org/abs/1907.12754.

[9] E. Berlekamp, R. McEliece и H. van Tilborg, «On the inherent intractability of certain coding problems (Corresp.),» *IEEE Transactions on Information Theory*, т. 24, № 3, с. 384—386, 1978. doi: 10.1109/TIT.1978.1055873.

[10] R. J. McEliece, «A public-key cryptosystem based on algebraic,» т. 4244, 1978, с. 114—116.

[11] D. J. Bernstein, «Introduction to post-quantum cryptography,» в *Post-Quantum Cryptography*, D. J. Bernstein, J. Buchmann и E. Dahmen, ред. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, с. 1—14. doi: 10.1007/978-3-540-88702-7_1. url: https://doi.org/10.1007/978-3-540-88702-7_1.

[12] N. T. Courtois, M. Finiasz и N. Sendrier, «How to Achieve a McEliece-Based Digital Signature Scheme,» в *Advances in Cryptology — ASIACRYPT 2001*, C. Boyd, ред., Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, с. 157—174.

[13] D. Augot, M. Finiasz и N. Sendrier, «A Family of Fast Syndrome Based Cryptographic Hash Functions,» в *Progress in Cryptology – Mycrypt 2005*, E. Dawson и S. Vaudenay, ред., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, с. 64—83.

[14] I. B. Damgård, «A Design Principle for Hash Functions,» в *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, ред., New York, NY: Springer New York, 1990, с. 416—427.

[15] R. C. Merkle, «Secrecy, Authentication, and Public Key Systems.,» AAI8001972, дис. … док., Stanford, CA, USA, 1979.

[16] F. Ren, D. Zheng и W. Wang, «An Efficient Code Based Digital Signature Algorithm,» *Int. J. Netw. Secur.*, т. 19, с. 1072—1079, 2017.

[17] P.-L. Cayrel, A. Otmani и D. Vergnaud, «On Kabatianskii-Krouk-Smeets Signatures,» в *Arithmetic of Finite Fields*, C. Carlet и B. Sunar, ред., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, с. 237—251.

[18] M. T. Banday, *Cryptographic Security Solutions for the Internet of Things*. янв. 2019.

[19] G. D'Alconzo, A. Meneghetti и P. Piasenti, «Security issues of CFS-like digital signature algorithms,» *CoRR*, т. abs/2112.00429, 2021. url: https://arxiv.org/abs/2112.00429.

[20] T. H. Cormen, C. E. Leiserson, R. L. Rivest и C. Stein, *Introduction to Algorithms, Third Edition*, 3rd. The MIT Press, 2009.

[21] E. K. Alekseev, I. B. Oshkin, V. O. Popov и S. V. Smyshlyaev, «On the cryptographic properties of algorithms accompanying the applications of standards GOST R 34.11-2012 and GOST R 34.10-2012,» *Mat. Vopr. Kriptogr.*, т. 7, № 1, с. 5—38, 2016.

[22] S. Goldwasser, S. Micali и R. L. Rivest, «A Digital Signature Scheme Secure Against Adaptive Chosen Message Attack,» в *Discrete Algorithms and Complexity*, D. S. Johnson, T. Nishizeki, A. Nozaki и H. S. Wilf, ред., Academic Press, 1987, с. 287—310. doi: https://doi.org/10.1016/B978-0-12-386870-1.50022-8. url: https://www.sciencedirect.com/science/article/pii/B9780123868701500228.

[23] T. G. Tan, P. Szalachowski и J. Zhou, *Challenges of Post-Quantum Digital Signing in Real-world Applications: A Survey*, Cryptology ePrint Archive, Report 2019/1374, https://ia.cr/2019/1374, 2019.

[24] L. Dallot, «Towards a Concrete Security Proof of Courtois, Finiasz and Sendrier Signature Scheme,» в *Research in Cryptology*, S. Lucks, A.-R. Sadeghi и C. Wolf, ред., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, с. 65—77.

[25] J.-C. Faugère, V. Gauthier, A. Otmani, L. Perret и J.-P. Tillich, «A Distinguisher for High-Rate McEliece Cryptosystems,» *IACR Cryptology ePrint Archive*, т. 2010, с. 331, июль 2010. doi: 10.1109/ITW.2011.6089437.

[26] K. Morozov, P. Roy, R. Steinwandt и R. Xu, «On the security of the Courtois-Finiasz-Sendrier signature,» *Open Mathematics*, т. 16, с. 161—167, март 2018. doi: 10.1515/math-2018-0011.

[27] V. D. Goppa, «A New Class of Linear Correcting Codes,» *Problems of Information Transmission*, т. 6, № 3, с. 207—212, 1970.

[28] P. Loidreau, «Codes Derived from Binary Goppa Codes,» *Problems of Information Transmission - PROBL INF TRANSM*, т. 37, апр. 2001. doi: 10.1023/A:1010406807141.

[29] A. Canteaut и F. Chabaud, «A new algorithm for finding minimum-weight words in a linear code: application to McEliece's cryptosystem and to narrow-sense BCH codes of length 511,» *IEEE Transactions on Information Theory*, т. 44, № 1, с. 367—378, 1998. doi: 10.1109/18.651067.

[30] P. Loidreau и N. Sendrier, «Weak keys in the McEliece public-key cryptosystem,» *IEEE Transactions on Information Theory*, т. 47, № 3, с. 1207—1211, 2001. doi: 10.1109/18.915687.

[31] M. Finiasz и N. Sendrier, «Security Bounds for the Design of Code-Based Cryptosystems,» в *Advances in Cryptology – ASIACRYPT 2009*, M. Matsui, ред., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, с. 88—105.

[32] S. Goldwasser и M. Bellare, *Lecture Notes on Cryptography*, 2001.

[33] H. C. A. V. Tilborg, *Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial*, 1st. USA: Kluwer Academic Publishers, 1999.

[34] E. R. Berlekamp, «Goppa codes,» *IEEE Trans. Inf. Theory*, т. 19, с. 590—592, 1973.

# Problems in the usage of a code hash function for a CFS signature scheme built on Goppa codes

Anastasiya Ilyukhina

***Abstract*—In 2001, a CFS signature scheme based on the Niederreiter cryptosystem was proposed. The signature algorithm is based on code cryptography, which makes the signature resistant to post-quantum attacks. However, there are certain difficulties in its implementation. One of them lies in the complexity of constructing a signature due to the low probability of receiving an acceptable syndrome that can be easily decoded. This article considers a known way of modifying the original signature scheme to solve this problem. The paper studies the use of a code hash function to quickly obtain a decodable syndrome. When considering the compression function of the hash function, an error was found in its construction and the insecurity of the signature scheme built on such a hash function was proved.**

***Keywords*—CFS signature, code based hash function**

References

[1] S. Goldwasser and M. Bellare, *Lecture notes on cryptography*, 2001.

[2] S. Goldwasser, S. Micali, and R. L. Rivest, «A digital signature scheme secure against adaptive chosen message attack», in *Discrete Algorithms and Complexity*, D. S. Johnson, T. Nishizeki, A. Nozaki, and H. S. Wilf, Eds., Academic Press, 1987, pp. 287–310. doi: https://doi.org/10.1016/B978-0-12-386870-1.50022-8. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123868701500228.

[3] T. G. Tan, P. Szalachowski, and J. Zhou, *Challenges of post-quantum digital signing in real-world applications: A survey*, Cryptology ePrint Archive, Report 2019/1374, https://ia.cr/2019/1374, 2019.

[4] E. K. Alekseev, I. B. Oshkin, V. O. Popov, and S. V. Smyshlyaev, «On the cryptographic properties of algorithms accompanying the applications of standards gost r 34.11-2012 and gost r 34.10-2012», *Mat. Vopr. Kriptogr.*, vol. 7, no. 1, pp. 5–38, 2016.

[5] L. Dallot, «Towards a concrete security proof of courtois, finiasz and sendrier signature scheme», in *Research in Cryptology*, S. Lucks, A.-R. Sadeghi, and C. Wolf, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 65–77.

[6] K. Morozov, P. Roy, R. Steinwandt, and R. Xu, «On the security of the courtois-finiasz-sendrier signature», *Open Mathematics*, vol. 16, pp. 161–167, Mar. 2018. doi: 10.1515/math-2018-0011.

[7] J.-C. Faugère, V. Gauthier, A. Otmani, L. Perret, and J.-P. Tillich, «A distinguisher for high-rate mceliece cryptosystems», *IACR Cryptology ePrint Archive*, vol. 2010, p. 331, Jul. 2010. doi: 10.1109/ITW.2011.6089437.

[8] P.-L. Cayrel, A. Otmani, and D. Vergnaud, «On kabatianskii-krouk-smeets signatures», in *Arithmetic of Finite Fields*, C. Carlet and B. Sunar, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 237–251.

[9] V. D. Goppa, «A new class of linear correcting codes», *Problems of Information Transmission*, vol. 6, no. 3, pp. 207–212, 1970.

[10] P. Loidreau, «Codes derived from binary goppa codes», *Problems of Information Transmission - PROBL INF TRANSM*, vol. 37, Apr. 2001. doi: 10.1023/A:1010406807141.

[11] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*, 3rd. The MIT Press, 2009.

[12] M. T. Banday, *Cryptographic Security Solutions for the Internet of Things*. Jan. 2019.

[13] G. D'Alconzo, A. Meneghetti, and P. Piasenti, «Security issues of cfs-like digital signature algorithms», *CoRR*, vol. abs/2112.00429, 2021. [Online]. Available: https://arxiv.org/abs/2112.00429.

[14] D. Augot, M. Finiasz, and N. Sendrier, «A family of fast syndrome based cryptographic hash functions», in *Progress in Cryptology – Mycrypt 2005*, E. Dawson and S. Vaudenay, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 64–83.

[15] I. B. Damgård, «A design principle for hash functions», in *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, Ed., New York, NY: Springer New York, 1990, pp. 416–427.

[16] R. C. Merkle, «One way hash functions and des», in *Advances in Cryptology — CRYPTO' 89 Proceedings*, G. Brassard, Ed., New York, NY: Springer New York, 1990, pp. 428–446.

[17] W. Diffie and M. Hellman, «New directions in cryptography», *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976. doi: 10.1109/TIT.1976.1055638.

[18] E. R. Berlekamp, «Goppa codes», *IEEE Trans. Inf. Theory*, vol. 19, pp. 590–592, 1973.

[19] M. Finiasz and N. Sendrier, «Security bounds for the design of code-based cryptosystems», in *Advances in Cryptology – ASIACRYPT 2009*, M. Matsui, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 88–105.

[20] A. Canteaut and F. Chabaud, «A new algorithm for finding minimum-weight words in a linear code: Application to mceliece's cryptosystem and to narrow-sense bch codes of length 511», *IEEE Transactions*

*on Information Theory*, vol. 44, no. 1, pp. 367–378, 1998. doi: 10.1109/18.651067.

[21] P. Loidreau and N. Sendrier, «Weak keys in the mceliece public-key cryptosystem», *IEEE Transactions on Information Theory*, vol. 47, no. 3, pp. 1207–1211, 2001. doi: 10.1109/18.915687.

[22] R. L. Rivest, A. Shamir, and L. Adleman, «A method for obtaining digital signatures and public-key cryptosystems», *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. doi: 10.1145/359340.359342. [Online]. Available: https://doi.org/10.1145/359340.359342.

[23] M. O. Rabin, «Digitalized signatures and public-key functions as intractable as factorization», USA, Tech. Rep., 1979.

[24] H. M. Grumbling E., *Quantum Computing: Progress and Prospects*. 2019.

[25] P. Shor, «Algorithms for quantum computation: Discrete logarithms and factoring», in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134. doi: 10.1109/SFCS.1994.365700.

[26] N. I. of Standards and T. (NIST), «Announcing request for nominations for public-key post-quantum cryptographic algorithms», 2016.

[27] H. Singh, «Code based cryptography: Classic mceliece», *CoRR*, vol. abs/1907.12754, 2019. [Online]. Available: http://arxiv.org/abs/1907.12754.

[28] E. Berlekamp, R. McEliece, and H. van Tilborg, «On the inherent intractability of certain coding problems (corresp.)», *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, 1978. doi: 10.1109/TIT.1978.1055873.

[29] R. J. McEliece, «A public-key cryptosystem based on algebraic», vol. 4244, 1978, pp. 114–116.

[30] D. J. Bernstein, «Introduction to post-quantum cryptography», in *Post-Quantum Cryptography*, D. J. Bernstein, J. Buchmann, and E. Dahmen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–14. doi: 10.1007/978-3-540-88702-7_1. [Online]. Available: https://doi.org/10.1007/978-3-540-88702-7_1.

[31] N. T. Courtois, M. Finiasz, and N. Sendrier, «How to achieve a mceliece-based digital signature scheme», in *Advances in Cryptology — ASIACRYPT 2001*, C. Boyd, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 157–174.

[32] F. Ren, D. Zheng, and W. Wang, «An efficient code based digital signature algorithm», *Int. J. Netw. Secur.*, vol. 19, pp. 1072–1079, 2017.

[33] R. C. Merkle, «Secrecy, authentication, and public key systems.», AAI8001972, Ph.D. dissertation, Stanford, CA, USA, 1979.

[34] H. C. A. V. Tilborg, *Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial*, 1st. USA: Kluwer Academic Publishers, 1999.