ГОСУДАРСТВЕННЫЙ СТАНДАРТ
РЕСПУБЛИКИ БЕЛАРУСЬ

СТБ 34.101.60-2014

Информационные технологии и безопасность
АЛГОРИТМЫ РАЗДЕЛЕНИЯ СЕКРЕТА

Інфармацыйныя тэхналогіі і бяспека
АЛГАРЫТМЫ РАЗДЗЯЛЕННЯ САКРЭТУ

Госстандарт
Минск

УДК 004.421.056.5(083.74)(476) МКС 35.240.40 КП 05

Ключевые слова: секрет, частичный секрет, разделение секрета

Предисловие

Цели, основные принципы, положения по государственному регулированию и управлению в области технического нормирования и стандартизации установлены Законом Республики Беларусь «О техническом нормировании и стандартизации».

1 РАЗРАБОТАН учреждением Белорусского государственного университета «Научно-исследовательский институт прикладных проблем математики и информатики» (НИИ ППМИ)

ВНЕСЕН Оперативно-аналитическим центром при Президенте Республики Беларусь

2 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ постановлением Госстандарта Республики Беларусь от 28 января 2014 г. № 5

3 ВЗАМЕН СТБ П 34.101.60-2011

Издан на русском языке

Содержание

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РЕСПУБЛИКИ БЕЛАРУСЬ

Информационные технологии и безопасность
АЛГОРИТМЫ РАЗДЕЛЕНИЯ СЕКРЕТА

Інфармацыйныя тэхналогіі і бяспека
АЛГАРЫТМЫ РАЗДЗЯЛЕННЯ САКРЭТУ

Information technology and security

Secret sharing algorithms

Дата введения 2014-09-01

1 Область применения

Настоящий стандарт определяет криптографические алгоритмы разделения и восстановления секрета (ключа) между пользователями, а также алгоритмы генерации параметров, необходимых при разделении / восстановлении секрета. Секрет разделяется между пользователями таким образом, что лишь заранее определенные подмножества пользователей могут восстановить его.

Настоящий стандарт применяется при разработке средств криптографической защиты информации.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие технические нормативные правовые акты в области технического нормирования и стандартизации (далее — ТНПА):

СТБ 34.101.31-2011 Информационные технологии. Защита информации. Криптографические алгоритмы шифрования и контроля целостности

СТБ 34.101.47-2012 Информационные технологии и безопасность. Криптографические алгоритмы генерации псевдослучайных чисел

ГОСТ 34.973-91 (ИСО 8824-87) Информационная технология. Взаимосвязь открытых систем. Спецификация абстрактно-синтаксической нотации версии 1 (АСН.1)

Примечание — При пользовании настоящим стандартом целесообразно проверить действие ТНПА по каталогу, составленному по состоянию на 1 января текущего года, и по соответствующим информационным указателям, опубликованным в текущем году.

Если ссылочные ТНПА заменены (изменены), то при пользовании настоящим предстандартом следует руководствоваться замененными (измененными) ТНПА. Если ссылочные ТНПА отменены без замены, то положение, в котором дана ссылка на них, применяется в части, не затрагивающей эту ссылку.

3 Термины и определения

В настоящем стандарте применяют термины, установленные в СТБ 34.101.31, а также следующие термины с соответствующими определениями:

3.1 (t, n)**-пороговое разделение секрета**: Разделение секрета между n пользователями, при котором только подмножества из $r \geqslant t$ пользователей могут восстановить секрет.

3.2 дилер: Сторона, выполняющая разделение секрета.

3.3 открытый ключ (разделения секрета): Открытые данные пользователя или общие открытые данные, используемые для разделения / восстановления секрета.

3.4 пороговое число: Число пользователей t, достаточное для восстановления секрета.

3.5 секрет: Конфиденциальные данные.

3.6 частичный секрет: Конфиденциальные данные пользователя, используемые для восстановления секрета.

4 Обозначения

4.1 Список обозначений

$\{0,1\}^n$	множество всех слов длины n в алфавите $\{0,1\}$;
$\{0,1\}^*$	множество всех слов конечной длины в алфавите $\{0,1\}$ (включая пустое слово длины 0);
$\{0,1\}^{n*}$	множество всех слов из $\{0,1\}^*$, длина которых кратна n;
$\langle u \rangle_m$	для $u \in \{0,1\}^*$ слово из первых m символов u, если длина u не меньше m, и само слово u в противном случае;
$u \parallel v$	конкатенация $u_1 u_2 \dots u_n v_1 v_2 \dots v_m$ слов $u = u_1 u_2 \dots u_n$ и $v = v_1 v_2 \dots v_m$;
$\mathtt{01234\dots}_{16}$	представление $u \in \{0,1\}^{4*}$ шестнадцатеричным словом, при котором последовательным четырем символам u соответствует один шестнадцатеричный символ (например, $10100010 = \mathtt{A2}_{16}$);
$\bar{u}$	а) для $u = u_1 u_2 \dots u_8 \in \{0,1\}^8$ число $2^7 u_1 + 2^6 u_2 + \dots + u_8$ и б) для $u = u_1 \parallel u_2 \parallel \dots \parallel u_n$, $u_i \in \{0,1\}^8$, число $\bar{u}_1 + 2^8 \bar{u}_2 + \dots + 2^{8(n-1)} \bar{u}_n$;
$\mathbb{F}_2$	поле из двух элементов 0 и 1;
$\mathbb{F}_2[x]$	кольцо многочленов над полем $\mathbb{F}_2$;
$u(x)$	а) для $u = u_1 u_2 \dots u_8 \in \{0,1\}^8$ многочлен $u_1 x^7 + u_2 x^6 + \dots + u_8$ и б) для $u = u_1 \parallel u_2 \parallel \dots \parallel u_n$, $u_i \in \{0,1\}^8$, многочлен $u_1(x) + x^8 u_2(x) + \dots + x^{8(n-1)} u_n(x)$;
$u(0)$	для многочлена $u(x) \in \mathbb{F}_2[x]$ его значение в точке $x = 0$, свободный член многочлена;
$\deg u(x)$	степень многочлена $u(x) \in \mathbb{F}_2[x]$;
$u(x) \bmod f(x)$	для $u(x) \in \mathbb{F}_2[x]$ и ненулевого $f(x) \in \mathbb{F}_2[x]$ остаток от деления $u(x)$ на $f(x)$;
$u(x) \operatorname{div} f(x)$	для $u(x) \in \mathbb{F}_2[x]$ и ненулевого $f(x) \in \mathbb{F}_2[x]$ частное от деления $u(x)$ на $f(x)$;

$\text{НОД}(f(x), g(x))$	наибольший общий делитель многочленов $f(x), g(x) \in \mathbb{F}_2[x]$;
l	длина секрета, число из множества $\{128, 192, 256\}$;
$\lfloor z \rfloor$	для вещественного числа z максимальное целое число, не превосходящее z;
$a \leftarrow u$	присвоение переменной a значения u;
$a \xleftarrow{R} U$	случайный равновероятный (или псевдослучайный) выбор a из множества U;
$a \leftrightarrow b$	перестановка значений переменных a и b.

4.2 Пояснения к обозначениям

Двоичные слова представляют собой последовательности символов из алфавита $\{0, 1\}$. Символы нумеруются слева направо от единицы. В настоящем подразделе в качестве примера рассматривается слово

$$w = 10110001100101001011101011001000.$$

В этом слове первый символ — 1, второй — 0, ..., последний — 0.

Слова разбиваются на тетрады из четверок последовательных двоичных символов. Тетрады кодируются шестнадцатеричными символами по следующим правилам (см. таблицу 1):

Таблица 1

тетрада	символ	тетрада	символ	тетрада	символ	тетрада	символ
0000	$\texttt{0}_{16}$	0001	$\texttt{1}_{16}$	0010	$\texttt{2}_{16}$	0011	$\texttt{3}_{16}$
0100	$\texttt{4}_{16}$	0101	$\texttt{5}_{16}$	0110	$\texttt{6}_{16}$	0111	$\texttt{7}_{16}$
1000	$\texttt{8}_{16}$	1001	$\texttt{9}_{16}$	1010	$\texttt{A}_{16}$	1011	$\texttt{B}_{16}$
1100	$\texttt{C}_{16}$	1101	$\texttt{D}_{16}$	1110	$\texttt{E}_{16}$	1111	$\texttt{F}_{16}$

Пары последовательных тетрад образуют октеты. Последовательные октеты слова w имеют вид:

$$10110001 = \texttt{B1}_{16},\ 10010100 = \texttt{94}_{16},\ 10111010 = \texttt{BA}_{16},\ 11001000 = \texttt{C8}_{16}.$$

4.2.1 Слова как числа

Октету $u = u_1 u_2 \dots u_8$ ставится в соответствие байт — число $\bar{u} = 2^7 u_1 + 2^6 u_2 + \dots + u_8$. Например, октетам w соответствуют байты

$$177 = 2^7 + 2^5 + 2^4 + 1,\ 148 = 2^7 + 2^4 + 2^2,\ 186 = 2^7 + 2^5 + 2^4 + 2^3 + 2^1,\ 200 = 2^7 + 2^6 + 2^3.$$

Число ставится в соответствие не только октетам, но и любому другому двоичному слову, длина которого кратна 8. При этом используется распространенное для многих современных процессоров соглашение «от младших к старшим» (little-endian): считается, что первый байт является младшим, последний — старшим. Например, слову w соответствует число

$$\bar{w} = 177 + 2^8 \cdot 148 + 2^{16} \cdot 186 + 2^{24} \cdot 200 = 3367670961.$$

4.2.2 Слова как многочлены

Октету $u = u_1u_2 \dots u_8$ ставится в соответствие многочлен $u(x) = u_1x^7 + u_2x^6 + \dots + u_8$. Многочлен ставится в соответствие также любому непустому двоичному слову из целого числа октетов. Используется соглашение «от младших к старшим»: первому октету соответствует многочлен $u_1(x)$, второму — $x^8u_2(x)$, третьему — $x^{16}u_3(x)$ и т. д.

Многочлены $u(x)$ считаются многочленами над полем $\mathbb{F}_2$. Это значит, что при сложении и умножении многочленов операции над их коэффициентами выполняются по модулю 2 (подробнее см. [1]). Деление $u(x)$ на ненулевой $f(x)$ состоит в определении многочленов $q(x)$, $r(x)$ таких, что $u(x) = q(x)f(x) + r(x)$ и степень $r(x)$ меньше степени $f(x)$. Многочлен $r(x)$ является остатком от деления, многочлен $q(x)$ — частным. Если $r(x) = 0$, то $f(x)$ делит $u(x)$.

Наибольший общий делитель многочленов $f(x)$ и $g(x)$ — это многочлен максимальной степени, который делит и $f(x)$, и $g(x)$. Многочлены $f(x)$ и $g(x)$ взаимно просты, если $\text{НОД}(f(x), g(x)) = 1$.

Многочлен положительной степени является неприводимым, если его нельзя представить в виде произведения многочленов меньших степеней. Постоянный многочлен (константа) не является неприводимым.

5 Общие положения

5.1 Назначение

Настоящий стандарт определяет криптографические алгоритмы разделения секрета между n пользователями. Разделение секрета выполняется так, что для выбранного порогового числа $t \in \{1, 2, \dots, n\}$ только подмножества из $r \geqslant t$ пользователей могут восстановить секрет.

Перед разделением секрета пользователи договариваются об использовании общего открытого ключа M_0. Алгоритм генерации M_0 определен в 6.4. После выбора M_0 пользователи строят свои открытые ключи $M_1, M_2, \dots, M_n$ и сообщают их друг другу. Алгоритм 6.5 предназначен для одновременной генерации открытых ключей всех пользователей. Алгоритм 6.6 предназначен для индивидуальной генерации открытого ключа пользователя по его идентификатору — двоичному слову конечной длины. В качестве идентификатора может выступать кодовое представление имени пользователя, адреса его электронной почты, личного номера и т.д. В приложении А определяются стандартные открытые ключи, которые можно использовать напрямую, без повторной генерации. Одни и те же открытые ключи могут использоваться для разделения нескольких секретов.

Секрет представляет собой двоичное слово длины $l \in \{128, 192, 256\}$. С помощью алгоритма 7.3 секрет S разделяется на частичные секреты $S_1, S_2, \dots, S_n$, которые передаются различным пользователям. С помощью алгоритма 7.4 по $r \geqslant t$ частичным секретам восстанавливается первоначальный секрет S.

Для алгоритмов, определенных в настоящем стандарте, выполняются следующие свойства:

- частичные секреты S_i являются двоичными словами такой же длины, как и S;
- совокупность из $r < t$ частичных секретов не содержит никакой информации об S.

Разделение секрета между пользователями выполняет дилер. Обычно, он же генерирует открытые ключи, если они не заданы. В качестве дилера может выступать лицо, программа, программно-аппаратное устройство. В настоящем стандарте полагается, что дилер строго следует алгоритму разделения секрета, не меняет параметры и ключи, т. е. является доверенной стороной.

Для разделения секретных данных большого объема рекомендуется выбрать секретный ключ случайно равновероятно из $\{0,1\}^l$, зашифровать данные на этом ключе, а затем разделить ключ на частичные секреты. Другой способ обработки данных большого объема заключается в их разбиении на фрагменты — двоичные слова длины l — и разделении каждого фрагмента по отдельности. При таком подходе следует предусмотреть контроль длины и целостности исходных данных, контроль порядка следования фрагментов и порядок обработки последнего, возможно неполного, фрагмента.

В приложении Б приводятся примеры выполнения алгоритмов стандарта. Примеры можно использовать для проверки корректности реализаций алгоритмов.

В приложении В определяется способ проверки корректности секрета после его восстановления.

В приложении Г приводится модуль абстрактно-синтаксической нотации версии 1 (АСН.1), определенной в ГОСТ 34.973. Модуль задает идентификаторы алгоритмов и других объектов настоящего стандарта, описывает структуры данных для хранения открытых ключей и частичных секретов. Рекомендуется использовать модуль при встраивании алгоритмов, определенных в настоящем стандарте, в информационные системы, в которых также используется АСН.1.

В приложении Д определяется протокол разделения секрета без участия дилера. Секрет изначально не задается, а вырабатывается пользователями совместно в ходе выполнения протокола.

Вспомогательные алгоритмы, используемые в настоящем стандарте, приведены в приложении Е.

5.2 Ключи, частичные секреты

Открытые ключи $M_0, M_1, \ldots, M_n$ представляют собой двоичные слова длины l. При распространении и хранении открытых ключей должен обеспечиваться контроль их целостности.

Открытым ключам соответствуют многочлены $f_i(x) = x^l + M_i(x)$, $i = 0, 1, \ldots, n$. Для корректного восстановления секрета многочлены $f_i(x)$ должны быть попарно взаимно простыми. При использовании алгоритма 6.5 условие взаимной простоты будет автоматически выполнено. Однако условие может нарушаться при генерации открытых ключей по алгоритму 6.6. В последнем случае открытые ключи пользователей с различными идентификаторами могут совпасть. Совпадения начнут происходить только тогда, когда число пользователей n приблизится к $2^{l/2}/\sqrt{l}$. Поэтому при небольших n возможностью совпадения можно пренебречь.

При разделении секрета используется одноразовый ключ k. Одноразовый ключ должен вырабатываться без возможности предсказания и уничтожаться после использования.

Для создания одноразовых ключей может быть использован физический генератор случайных чисел, удовлетворяющий ТНПА, или алгоритм генерации псевдослучайных чисел, определенный в СТБ 34.101.47 или в другом ТНПА. Входные данные алгоритма должны включать секретный ключ и уникальную синхропосылку. Длина ключа алгоритма генерации должна быть не меньше l.

Частичные секреты являются двоичными словами длины l. Они должны храниться и распространяться с соблюдением мер конфиденциальности и контроля целостности. Утрата частичных секретов в количестве, меньшем порогового числа t, не влияет на безопасность исходного секрета и не влечет необходимость его замены.

Поскольку одни и те же открытые ключи могут использоваться при разделении различных секретов, рекомендуется выбирать уникальный номер секрета и распространять его вместе с частичными секретами. При восстановлении секрета по этому номеру можно судить о соответствии частичных секретов друг другу.

6 Алгоритмы генерации параметров

6.1 Входные и выходные данные

Входными данными алгоритма генерации общего открытого ключа является длина секрета $l \in \{128, 192, 256\}$. Выходными данными является общий открытый ключ $M_0 \in \{0,1\}^l$.

Входными данными алгоритма генерации открытых ключей пользователей являются число пользователей n, длина секрета $l \in \{128, 192, 256\}$ и общий открытый ключ $M_0 \in \{0,1\}^l$. Выходными данными являются слова $M_1, \ldots, M_n \in \{0,1\}^l$ — открытые ключи пользователей.

Входными данными алгоритма генерации открытого ключа пользователя по идентификатору являются длина секрета $l \in \{128, 192, 256\}$, общий открытый ключ $M_0 \in \{0,1\}^l$ и идентификатор пользователя $Id \in \{0,1\}^*$. Выходными данными является слово $M \in \{0,1\}^l$ — открытый ключ пользователя с идентификатором Id.

6.2 Вспомогательные алгоритмы и переменные

Алгоритм `belt-hash`. Используется алгоритм хэширования `belt-hash`, определенный в СТБ 34.101.31 (пункт 6.9.3). Входными данными алгоритма является слово $X \in \{0,1\}^*$, выходными — хэш-значение $Y \in \{0,1\}^{256}$.

Проверка неприводимости. На шаге 2 алгоритма 6.4 проверяется неприводимость многочлена. Для проверки рекомендуется использовать алгоритмы Рабина [2] или Бен-Ора [3], описанные в Е.1. При обработке случайных многочленов рекомендуется отдавать предпочтение последнему алгоритму.

Алгоритм `BuildIrred`. На шаге 2.2 алгоритма 6.5 и на шаге 2 алгоритма 6.6 используется алгоритм `BuildIrred`, определенный в 6.3. Входными данными алгоритма являются слово $u \in \{0,1\}^l$ и общий открытый ключ $M_0 \in \{0,1\}^l$. Выходными данными является многочлен $f(x) \in \mathbb{F}_2[x]$, степень которого не превосходит l и который является либо неприводимым, либо постоянным.

Переменные. Используются переменная u со значениями в $\{0,1\}^l$ и переменные $a(x), b(x), g(x), q(x), r(x)$ со значениями в $\mathbb{F}_2[x]$.

6.3 Алгоритм `BuildIrred`

Для генерации многочлена $f(x)$ выполняются следующие шаги:

1 Установить $a(x) \leftarrow u(x)$, $b(x) \leftarrow a(0)$.

2 Для $i = 1, 2, \ldots, 2l - 1$:

  1) $a(x) \leftarrow a(x)u(x) \bmod \big(x^l + M_0(x)\big)$;

  2) $b(x) \leftarrow xb(x) + a(0)$.

3 Установить $a(x) \leftarrow x^{2l}$.

4 Установить $g(x) \leftarrow 0$, $f(x) \leftarrow 1$.

5 Пока $b(x) \neq 0$ и $\deg b(x) \geqslant l$ выполнить:

  1) $q(x) \leftarrow a(x) \operatorname{div} b(x)$;

  2) $a(x) \leftarrow a(x) \bmod b(x)$;

  3) $a(x) \leftrightarrow b(x)$;

  4) $g(x) \leftarrow g(x) + f(x)q(x)$;

  5) $g(x) \leftrightarrow f(x)$.

6 Возвратить $f(x)$.

6.4 Алгоритм генерации общего открытого ключа

Для генерации общего открытого ключа выполняются следующие шаги:

1 Выбрать произвольным образом $M_0 \in \{0,1\}^l$.

2 Если многочлен $x^l + M_0(x)$ не является неприводимым, то вернуться к шагу 1.

3 Возвратить M_0.

6.5 Алгоритм генерации открытых ключей пользователей

Для генерации открытых ключей пользователей выполняются следующие шаги:

1 Установить $i \leftarrow 1$.

2 Пока $i \leqslant n$ выполнить:

  1) выбрать произвольным образом $u \in \{0,1\}^l$;

  2) $f(x) \leftarrow \texttt{BuildIrred}(u, M_0)$;

  3) если $\deg f(x) \neq l$, то вернуться к шагу 2.1;

  4) определить $M_i \in \{0,1\}^l$, исходя из условия $f(x) = x^l + M_i(x)$;

  5) если M_i совпадает с некоторым из слов $M_0, \ldots, M_{i-1}$, то вернуться к шагу 2.1;

  6) $i \leftarrow i + 1$.

3 Возвратить $M_1, \ldots, M_n$.

6.6 Алгоритм генерации открытого ключа пользователя по идентификатору

Для генерации открытого ключа пользователя по его идентификатору выполняются следующие шаги:

1 Установить $u \leftarrow \langle \texttt{belt-hash}(Id) \rangle_l$.

2 Установить $f(x) \leftarrow \texttt{BuildIrred}(u, M_0)$.

3 Если $\deg f(x) \neq l$ или $f(x) = x^l + M_0(x)$, то

  1) $u \leftarrow \langle \bar{u} + 1 \rangle_l$;

  2) вернуться к шагу 2.

4 Определить $M \in \{0,1\}^l$, исходя из условия $f(x) = x^l + M(x)$.

5 Возвратить M.

7 Основные алгоритмы

7.1 Входные и выходные данные

Входными данными алгоритма разделения секрета являются число пользователей n, пороговое число t, длина секрета $l \in \{128, 192, 256\}$, секрет $S \in \{0,1\}^l$ и открытые ключи $M_0, M_1, \ldots, M_n \in \{0,1\}^l$.

Выходными данными алгоритма разделения секрета являются слова $S_1, S_2, \ldots, S_n \in \{0,1\}^l$ — частичные секреты пользователей.

При восстановлении секрета используются данные пользователей с номерами $i_1, i_2, \ldots, i_r \in \{1, 2, \ldots, n\}$. Входными данными алгоритма восстановления секрета являются длина секрета $l \in \{128, 192, 256\}$, частичные секреты $S_{i_1}, S_{i_2}, \ldots, S_{i_r} \in \{0,1\}^l$, общий открытый ключ $M_0 \in \{0,1\}^l$ и открытые ключи $M_{i_1}, M_{i_2}, \ldots, M_{i_r} \in \{0,1\}^l$.

Выходными данными алгоритма восстановления секрета является либо секрет $S \in \{0,1\}^l$, либо признак `ОШИБКА`. Возврат признака `ОШИБКА` означает, что открытые ключи некорректны.

Примечание — Восстановленный секрет не обязательно будет совпадать с первоначально разделенным. Для гарантированного совпадения требуется сохранение целостности открытых ключей и частичных секретов, а также выполнение условия $r \geqslant t$.

7.2 Вспомогательные алгоритмы и переменные

Расширенный алгоритм Евклида. Для определения многочленов $d(x), u(x), v(x)$ на шаге 2.1 алгоритма 7.4 может использоваться расширенный алгоритм Евклида. Варианты алгоритма представлены в [4], а также в приложении Е.

Решение системы сравнений. На шагах 1, 2 алгоритма 7.4 определяется решение $C(x)$ системы сравнений

$$C(x) \bmod (x^l + M_{i_j}(x)) = S_{i_j}(x), \quad j = 1, 2, \ldots, r,$$

которая удовлетворяет китайской теореме об остатках [4]. Допустимы другие способы решения системы, например, с помощью алгоритма Гарнера [5].

Одноразовый ключ k. При разделении секрета используется одноразовый ключ $k \in \{0,1\}^{(t-1)l}$. Требования по управлению k определены в 5.2.

Переменные. Используется переменная $C(x)$ со значениями в $\mathbb{F}_2[x]$. Значение $C(x)$ должно быть уничтожено после использования. При восстановлении секрета дополнительно используются переменные $d(x), u(x), v(x), g(x)$ со значениями в $\mathbb{F}_2[x]$.

7.3 Алгоритм разделения секрета

Для разделения секрета выполняются следующие шаги:

1 Выработать $k \xleftarrow{R} \{0,1\}^{(t-1)l}$ (в соответствии с требованиями пункта 5.2).

2 Установить $C(x) \leftarrow (x^l + M_0(x))k(x) + S(x)$.

3 Для $i = 1, 2, \ldots, n$ выполнить:

1) $S_i(x) \leftarrow C(x) \bmod (x^l + M_i(x))$.

4 Возвратить $S_1, S_2, \ldots, S_n$.

7.4 Алгоритм восстановления секрета

Для восстановления секрета выполняются следующие шаги:

1 Установить $C(x) \leftarrow S_{i_1}(x)$, $g(x) \leftarrow x^l + M_{i_1}(x)$.

2 Для $j = 2, 3, \ldots, r$ выполнить:

1) найти $d(x), u(x), v(x)$ такие, что $d(x) = \text{НОД}(x^l + M_{i_j}, g(x))$ и $u(x)(x^l + M_{i_j}) + v(x)g(x) = d(x)$;

2) если $d(x) \neq 1$, то возвратить ОШИБКА;

3) $C(x) \leftarrow u(x)(x^l + M_{i_j}(x))C(x) + v(x)g(x)S_{i_j}(x)$;

4) $g(x) \leftarrow (x^l + M_{i_j}(x))g(x)$;

5) $C(x) \leftarrow C(x) \bmod g(x)$.

3 $S(x) \leftarrow C(x) \bmod (x^l + M_0(x))$.

4 Возвратить S.

Приложение А
(рекомендуемое)
Стандартные параметры

В таблице А.1 представлены стандартные общие открытые ключи M_0 для различных уровней стойкости. Этим ключам соответствуют неприводимые многочлены $x^{128} + x^7 + x^2 + x + 1$, $x^{192} + x^7 + x^2 + x + 1$ и $x^{256} + x^{10} + x^5 + x^2 + 1$.

В таблицах А.2 –– А.4 представлены стандартные открытые ключи пользователей. Открытые ключи M_0 и M_i соответствуют требованиям 5.2 и могут использоваться при реализации алгоритмов разделения и восстановления секрета.

Таблица А.1 — Стандартные общие открытые ключи

l	M_0
128	87000000 00000000 00000000 00000000_{16}
192	87000000 00000000 00000000 00000000 00000000 00000000_{16}
256	25040000 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}

Таблица А.2 — Стандартные открытые ключи пользователей ($l = 128$)

i	M_i
1	85020000 00000000 00000000 00000000_{16}
2	410C0000 00000000 00000000 00000000_{16}
3	21180000 00000000 00000000 00000000_{16}
4	15800000 00000000 00000000 00000000_{16}
5	01830000 00000000 00000000 00000000_{16}
6	81020200 00000000 00000000 00000000_{16}
7	81200200 00000000 00000000 00000000_{16}
8	01A00200 00000000 00000000 00000000_{16}
9	41010800 00000000 00000000 00000000_{16}
10	05020800 00000000 00000000 00000000_{16}
11	01280800 00000000 00000000 00000000_{16}
12	01A00800 00000000 00000000 00000000_{16}
13	41801000 00000000 00000000 00000000_{16}
14	25002000 00000000 00000000 00000000_{16}
15	05042000 00000000 00000000 00000000_{16}
16	010C2000 00000000 00000000 00000000_{16}

Таблица А.3 — Стандартные открытые ключи пользователей ($l = 192$)

i	M_i
1	09120000 00000000 00000000 00000000 00000000 00000000_{16}
2	41120000 00000000 00000000 00000000 00000000 00000000_{16}
3	01860000 00000000 00000000 00000000 00000000 00000000_{16}
4	21880000 00000000 00000000 00000000 00000000 00000000_{16}
5	05C00000 00000000 00000000 00000000 00000000 00000000_{16}
6	49000200 00000000 00000000 00000000 00000000 00000000_{16}
7	85000200 00000000 00000000 00000000 00000000 00000000_{16}
8	09100200 00000000 00000000 00000000 00000000 00000000_{16}
9	01080600 00000000 00000000 00000000 00000000 00000000_{16}
10	01020900 00000000 00000000 00000000 00000000 00000000_{16}
11	81000A00 00000000 00000000 00000000 00000000 00000000_{16}
12	11042000 00000000 00000000 00000000 00000000 00000000_{16}
13	01802200 00000000 00000000 00000000 00000000 00000000_{16}
14	09024000 00000000 00000000 00000000 00000000 00000000_{16}
15	01084200 00000000 00000000 00000000 00000000 00000000_{16}
16	01048100 00000000 00000000 00000000 00000000 00000000_{16}

Таблица А.4 — Стандартные открытые ключи пользователей ($l = 256$)

i	M_i
1	0B000100 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
2	0D000100 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
3	01A00100 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
4	61000200 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
5	85000400 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
6	81012000 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
7	05402000 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
8	11002800 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
9	01028100 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
10	01048200 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
11	0B000001 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
12	01280001 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
13	09002001 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
14	29000002 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
15	09200002 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}
16	0B000008 00000000 00000000 00000000 00000000 00000000 00000000 00000000_{16}

Приложение Б
(справочное)
Проверочные примеры

Б.1 Генерация открытого ключа пользователя по идентификатору

В таблице Б.1 представлены примеры генерации открытого ключа пользователя по идентификатору. Используются стандартные общие открытые ключи M_0, заданные в таблице А.1.

Таблица Б.1 — Генерация открытого ключа пользователя по идентификатору

Id	$\texttt{416C696365}_{16}$
$\texttt{belt-hash}(Id)$	$\texttt{B60BA9A6 5593973E 88C1B779 B7D0DC64 4BE6034B 6950EDAE 1FE6D22B 0A8B1836}_{16}$
M $(l = 128)$	$\texttt{F9D6F31B 5DB0BB61 F00E17EE F2E6007F}_{16}$
M $(l = 192)$	$\texttt{09EA7929 7F94A3E4 3A3885FC 0D1BB8FD D0DF86FD 313CEF46}_{16}$
M $(l = 256)$	$\texttt{D53CC51B E1F976F1 032A00D9 CD0E190E 62C37FFD 233E8A9D F14C85F8 5C51A045}_{16}$

Б.2 Разделение секрета

В таблицах Б.2 — Б.4 представлены примеры $(3, 5)$-порогового разделения секрета. Используются стандартные общие открытые ключи M_0, заданные в таблице А.1, и стандартные открытые ключи пользователей $M_1, M_2, \ldots, M_5$, заданные в таблицах А.2 — А.4.

Таблица Б.2 — Разделение секрета $(l = 128)$

S	$\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4}_{16}$
k	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}$ $\texttt{06075316 ED247A37 39CBA383 03A98BF6}_{16}$
S_1	$\texttt{E27D0CFD 31C557BC 37C3897D CFF2C7FC}_{16}$
S_2	$\texttt{50BB9EEC BAEF52DD B811BCDE 1495441D}_{16}$
S_3	$\texttt{A92473F6 79668353 4AD11581 2A3F9950}_{16}$
S_4	$\texttt{9A8331FD 945D58E6 D8723E47 44FB1DA9}_{16}$
S_5	$\texttt{51913D18 C8625C5A B0812133 FB643D66}_{16}$

Таблица Б.3 — Разделение секрета $(l = 192)$

S	$\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC}_{16}$
k	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37}$ $\texttt{39CBA383 03A98BF6 92BD9B1C E5D14101 5445FBC9 5E4D0EF2}_{16}$
S_1	$\texttt{8D0EBB0C 67A315C2 14B34A5D 68E9712A 12F7B432 87E3138A}_{16}$
S_2	$\texttt{2506EB82 83D85553 18479D27 8A752B04 E9B5E6CC 43543403}_{16}$
S_3	$\texttt{E5B885E6 5E69ADD3 30D08268 EC3D0A44 B04B8E14 2CDDDD5C}_{16}$
S_4	$\texttt{E85B368A 66489AFE 0E73D3D0 EEB6A210 CF0629C2 75AB1E94}_{16}$
S_5	$\texttt{ED6CD8B5 6C37C03E E4FF04AE 2A975AAA 748AA0E9 7AA0DE20}_{16}$

Таблица Б.4 — Разделение секрета ($l = 256$)

S	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D$_{16}$
k	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6 92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
S_1	27EC2268 C7A06E7C C54F66FC 3D357298 4D4D4EF6 9916EB8D 1EAFDFA4 20217ADC$_{16}$
S_2	20E06235 E355CC43 3E2AF2F4 100C636F 3BFAB861 A4390614 E42BC175 77BCBE42$_{16}$
S_3	1E14B1E7 95CED216 AAC5BB52 6EFC786C 5BCE1F18 65D3886E D4DD7D9E FEF77F39$_{16}$
S_4	62EFAD25 44718293 262E2CB7 4A396B50 B6D8843D F5E2F0EE FFFE6CD1 8722765E$_{16}$
S_5	71ADE959 FC88CCBB 1C521FA9 A1168C18 4619832A B66265E0 8A65DD48 EE406418$_{16}$

Б.3 Восстановление секрета

В таблицах Б.5 — Б.7 представлены примеры восстановления секрета. Используются открытые ключи и частичные секреты, описанные в Б.2.

Таблица Б.5 — Восстановление секрета ($l = 128$)

Подмножество пользователей	Восстановленный секрет
{1, 2}	6380669C A508058F A9AADF98 6C77C175$_{16}$
{1, 3}	ABD72A83 5739A358 DD954BEF 7A923AEC$_{16}$
{1, 4}	225E2DF0 E4AE6532 D5A74198 1410A83C$_{16}$
{1, 5}	E0C4268A C9C5FE35 C15334E4 D01417BE$_{16}$
{2, 3}	E8BA8376 76967C5C 939DBF51 72C9AB4F$_{16}$
{2, 4}	6CB93B8C F600A746 F8520860 901E36FA$_{16}$
{2, 5}	E4FCC7E2 4E448324 367F4003 26954776$_{16}$
{3, 4}	81C498D5 5DC506E8 58DE632A 079C2C31$_{16}$
{3, 5}	E685CC72 5DDE29E6 09275639 12CBBEA4$_{16}$
{4, 5}	40F629F9 A4487DBC BF53192E A4A49EAA$_{16}$
3 и более пользователей	B194BAC8 0A08F53B 366D008E 584A5DE4$_{16}$

Таблица Б.6 — Восстановление секрета ($l = 192$)

Подмножество пользователей	Восстановленный секрет
{1, 2}	1E9811BD 520C56E1 2B5B0E51 7756FA1A EE3CACC1 $3B6313E9_{16}$
{1, 3}	A2E3B51A FBD7AFD5 52048DD6 444416E0 7F2D9FA9 $2D726920_{16}$
{1, 4}	2B65B8D1 BEF2EA07 9F6C45DF 5877EAA1 8F118853 $9B0AEF32_{16}$
{1, 5}	7E880E3E 89CE5FD4 E8452256 BD66E42D 18D88C0C $F85FDC26_{16}$
{2, 3}	AF8AB830 4FEBD5CF 89D643A8 50C77165 7310CA0E $8EDF9C60_{16}$
{2, 4}	6D542544 073C04C1 C417ABDC 292755A2 861B4EB5 $90B65841_{16}$
{2, 5}	EF5CE43C 8AE6F4E4 41CE1C2D 16ACC662 D6CC1D8B $AF937320_{16}$
{3, 4}	21B6A467 511CD2CE 6AE671E1 D0992538 BFB4EAE9 $27F70991_{16}$
{3, 5}	F2E19395 8DB1D339 1D54C410 244C151D BC267D6F $5182DEC4_{16}$
{4, 5}	1C0E2B99 D81134E0 EB9AD402 79D09786 CA3CDA79 $B2E5D385_{16}$
3 и более пользователей	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D $1BB6C7AC_{16}$

Таблица Б.7 — Восстановление секрета ($l = 256$)

Подмножество пользователей	Восстановленный секрет
{1, 2}	C39C8FA8 590A7855 914AED9B 05940D9E 8A119B13 0D939B87 99889C93 $8D1E078D_{16}$
{1, 3}	70EDE256 F46BDC35 EEE39361 921EE8A3 94E8E67F 3F56ABFB A65329D1 $46DA185B_{16}$
{1, 4}	7C2D5033 F0F10CC6 9065B13B B53BE7D1 9D61CF86 4CF1578E 8325F105 $64F995A3_{16}$
{1, 5}	00DD41CD 32684FE7 564F67FC 51B0AD87 003EEBDF 90E803BA 37CBA4FF $8D9A724F_{16}$
{2, 3}	31C06C2B F7AF38C2 A6870A7F 1B7BA9CC 1A741DD9 6374A4D1 7A1F7016 $66C9A777_{16}$
{2, 4}	44FC1DE6 84980BE2 660BB7BC E50728A1 25A81D3B 71B8D4AC D74E0319 $0ADA473B_{16}$
{2, 5}	264FD3BE 92984957 58B24463 63616A38 75D15EB9 6F95A122 332597A8 $7B2CCCBC_{16}$
{3, 4}	3ACC00A6 DF80BC31 4A708A19 D467F954 40B21435 6D4666B4 075E384B $87BEB86C_{16}$
{3, 5}	B3C2EDAD 484A5A86 4575721D 10B9D0C0 9AE32C97 2C74857B A423D045 $02EE0066_{16}$
{4, 5}	3F5F33C7 78D77A4F ADC0BB51 BE9F0153 2627D1E8 3D023DA7 2255CC82 $6B05213B_{16}$
3 и более пользователей	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 $02FDCE0D_{16}$

Приложение В
(рекомендуемое)
Проверка секрета

После восстановления секрета может потребоваться проверка того, что он совпадает с первоначальным. Такая проверка обеспечивает защиту от предоставления некорректных частичных секретов по ошибке или злонамеренно.

Проверку корректности восстановленного секрета можно организовать с помощью функции хэширования h, которая ставит в соответствие двоичному слову X его хэш-значение $h(X) \in \{0,1\}^m$, $m \geqslant l$. Функция h должна быть алгоритмически определена в некотором ТНПА. Рекомендуется использовать функцию хэширования, заданную в СТБ 34.101.31.

При разделении секрета S дилер применяет h и определяет контрольное слово $H = \langle h(S) \rangle_l$. Затем с помощью алгоритма 7.3 дилер разделяет S и H на частичные секреты, используя одно и то же пороговое число t. Вместе с частичными секретами S_i от S дилер передает пользователям частичные секреты H_i от H, $i = 1, 2, \ldots, n$.

С помощью алгоритма 7.4 пользователи по своим частичным секретам восстанавливают и секрет S, и контрольное слово H. Восстановленный секрет признается корректным, если $\langle h(S) \rangle_l$ совпадает с H.

Приложение Г
(рекомендуемое)
Модуль ACH.1

Г.1 Идентификаторы

Алгоритмам стандарта присваиваются следующие идентификаторы:

`bels-genm0`	алгоритм генерации общего открытого ключа (6.4);
`bels-genmi`	алгоритм генерации открытых ключей пользователей (6.5);
`bels-genmid`	алгоритм генерации открытого ключа пользователя по идентификатору (6.6);
`bels-share`	алгоритмы разделения и восстановления секрета (7.3, 7.4).

Стандартным общим открытым ключам, заданным в таблице А.1, присваиваются следующие идентификаторы:

`bels-m0128v1`	общий открытый ключ для $l = 128$;
`bels-m0192v1`	общий открытый ключ для $l = 192$;
`bels-m0256v1`	общий открытый ключ для $l = 256$.

Г.2 Описание общего открытого ключа

Общий открытый ключ может быть представлен значением типа:

```
CommonPublicKey ::= CHOICE {
  specified OCTET STRING(SIZE(16|24|32)),
  named OBJECT IDENTIFIER
}
```

Выбор компонента `specified` означает явное задание общего открытого ключа. Компонент `named` используется для ссылки на именованные параметры, заданные в таблице А.1 или в другом ТНПА.

Г.3 Описание открытого ключа пользователя

Для описания открытого ключа пользователя используется тип

```
PublicKey ::= SEQUENCE {
  m0 CommonPublicKey,
  m OCTET STRING(SIZE(16|24|32)),
  id OCTET STRING OPTIONAL
}
```

Компонент `m0` описывает общий открытый ключ, представленный типом `CommonPublicKey`. Компонент `m` определяет открытый ключ пользователя. Длина компонента `m` должна совпадать с длиной компонента `specified` типа `CommonPublicKey`. Необязательный компонент `id` описывает идентификатор пользователя.

Г.4 Описание частичного секрета пользователя

Частичный секрет пользователя представляется следующим типом:

```
SecretShare ::= SEQUENCE{
  version INTEGER {ssVer1(1)} (ssVer1),
  publicKey PublicKey,
  threshold INTEGER,
  share OCTET STRING(SIZE(16|24|32)),
  serial OCTET STRING OPTIONAL,
  mac SecretMAC OPTIONAL
}
```

Компонент `version` указывает на версию данного типа АСН.1. Примененный синтаксис обязывает использовать версию 1, которая обозначена через `ssVer1`. Компонент `publicKey` описывает открытый ключ пользователя. В компоненте `threshold` указывается пороговое число t. Компонент `share` определяет частичный секрет пользователя, длина которого должна совпадать с длиной открытого ключа пользователя. Необязательный компонент `serial` описывает уникальный номер, который может быть задан при разделении секрета (см. 5.2). Необязательный компонент `mac` описывает параметры, необходимые для проверки корректности секрета в соответствии с приложением В, и представлен типом `SecretMAC`:

```
SecretMAC ::= SEQUENCE{
  hash AlgorithmIdentifier,
  mac OCTET STRING(SIZE(16|24|32))
}
```

Компонент `hash` описывает идентификатор функции хэширования, а компонент `mac` описывает частичный секрет H_i от контрольного слова H и имеет ту же длину, что и частичный секрет пользователя. Тип `AlgorithmIdentifier` определен следующим образом:

```
AlgorithmIdentifier ::= SEQUENCE {
  algorithm OBJECT IDENTIFIER,
  parameters ANY DEFINED BY algorithm OPTIONAL
}
```

В компоненте `algorithm` должен быть установлен идентификатор функции хэширования, удовлетворяющей требованиям, определенным в приложении В.

Г.5 Модуль АСН.1

Модуль АСН.1 имеет следующий вид:

```
Bels-module-v2 {iso(1) member-body(2) by(112) 0 2 0 34 101 60 module(1) ver2(2)}
DEFINITIONS ::=
BEGIN
  bels OBJECT IDENTIFIER ::= {iso(1) member-body(2) by(112) 0 2 0 34 101 60}
```

```
  bels-share OBJECT IDENTIFIER ::= {bels 11}
  bels-genm0 OBJECT IDENTIFIER ::= {bels 101}
  bels-genmi OBJECT IDENTIFIER ::= {bels 102}
  bels-genmid OBJECT IDENTIFIER ::= {bels 103}

  bels-m0 OBJECT IDENTIFIER ::= {bels m0(2)}
  bels-m0128v1 OBJECT IDENTIFIER ::= {bels-m0 1}
  bels-m0192v1 OBJECT IDENTIFIER ::= {bels-m0 2}
  bels-m0256v1 OBJECT IDENTIFIER ::= {bels-m0 3}

  AlgorithmIdentifier ::= SEQUENCE {
    algorithm OBJECT IDENTIFIER,
    parameters ANY DEFINED BY algorithm OPTIONAL
  }

  SecretMAC ::= SEQUENCE{
    hash AlgorithmIdentifier,
    mac OCTET STRING(SIZE(16|24|32))
  }

  CommonPublicKey ::= CHOICE {
    specified OCTET STRING(SIZE(16|24|32)),
    named OBJECT IDENTIFIER
  }

  PublicKey ::= SEQUENCE {
    m0 CommonPublicKey,
    m OCTET STRING(SIZE(16|24|32)),
    id OCTET STRING OPTIONAL
  }

  SecretShare ::= SEQUENCE {
    version INTEGER {ssVer1(1)} (ssVer1),
    publicKey PublicKey,
    threshold INTEGER,
    share OCTET STRING(SIZE(16|24|32)),
    serial OCTET STRING OPTIONAL,
    mac SecretMAC OPTIONAL
  }
END
```

Приложение Д
(справочное)
Разделение секрета без участия дилера

В некоторых случаях необходимо выработать секрет и разделить его между пользователями без участия дилера. В настоящем приложении описывается протокол решения данной задачи.

Предполагается, что пользователи заранее договорились использовать определенные открытые ключи и выбрали пороговое число t.

Для выработки секрета и одновременного его разделения пользователь с номером $i = 1, 2, \ldots, n$ выполняет следующие действия:

1 Вырабатывает $P_i \xleftarrow{R} \{0,1\}^l$. Генерация P_i должна выполняться по тем же правилам, что и генерация одноразового ключа k (см. 5.2).

2 Разделяет P_i на частичные секреты $P_{i1}, P_{i2}, \ldots, P_{in}$ по алгоритму 7.3, используя пороговое число t.

3 Пересылает j-му пользователю частичный секрет P_{ij}. При пересылке должна обеспечиваться конфиденциальность и контролироваться целостность и подлинность данных.

4 После всех взаимных пересылок формирует свой частичный секрет $S_i \in \{0,1\}^l$, которому соответствует многочлен $S_i(x) = P_{1i}(x) + P_{2i}(x) + \ldots + P_{ni}(x)$.

По окончании протокола пользователи получают частичные секреты $S_1, S_2, \ldots, S_n$. По любым $r \geqslant t$ из них с помощью алгоритма 7.4 можно восстановить один и тот же секрет $S \in \{0,1\}^l$ такой, что

$$S(x) = P_1(x) + P_2(x) + \ldots + P_n(x).$$

Приложение Е
(справочное)
Вспомогательные алгоритмы

Е.1 Проверка неприводимости алгоритмами Рабина и Бен-Ора

Е.1.1 Входные и выходные данные

Входными данными алгоритмов Рабина и Бен-Ора является многочлен $f(x) \in \mathbb{F}_2[x]$ степени $l \geqslant 2$. Выходными данными является ответ `ДА` или `НЕТ`. Ответ `ДА` означает, что $f(x)$ неприводим. Ответ `НЕТ` означает обратное.

Е.1.2 Переменная

Используется переменная $g(x)$ со значениями в $\mathbb{F}_2[x]$, $\deg g(x) < l$.

Е.1.3 Алгоритм Рабина

Для проверки неприводимости многочлена $f(x)$ выполняются следующие шаги:

1 Установить $g(x) \leftarrow x^2$.

2 Для $i = 1, 2, \ldots, l-1$ выполнить:

  1) если $l = ip$, где p — простое число, и $\text{НОД}(f(x), g(x)+x) \neq 1$, то возвратить `НЕТ`;

  2) $g(x) \leftarrow g^2(x) \bmod f(x)$.

3 Если $g(x) \neq x$, то возвратить `НЕТ`.

4 Возвратить `ДА`.

Е.1.4 Алгоритм Бен-Ора

Для проверки неприводимости многочлена $f(x)$ выполняются следующие шаги:

1 Установить $g(x) \leftarrow x$.

2 Для $i = 1, 2, \ldots, \lfloor l/2 \rfloor$ выполнить:

  1) $g(x) \leftarrow g^2(x) \bmod f(x)$;

  2) если $\text{НОД}(f(x), g(x) + x) \neq 1$, то возвратить `НЕТ`.

3 Возвратить `ДА`.

Е.2 Расширенный алгоритм Евклида

Е.2.1 Входные и выходные данные

Входными данными расширенного алгоритма Евклида являются многочлены $f(x), g(x) \in \mathbb{F}_2[x]$, $\deg f(x) \leqslant \deg g(x)$. Выходными данными являются многочлены $d(x), u(x), v(x) \in \mathbb{F}_2[x]$ такие, что

$$d(x) = \text{НОД}(f(x), g(x)) = u(x)f(x) + v(x)g(x).$$

Е.2.2 Переменные

Используются переменные $u_1(x)$, $v_1(x)$, $d_1(x)$, $q(x)$ со значениями в $\mathbb{F}_2[x]$.

Е.2.3 Алгоритм

Для нахождения многочленов $d(x)$, $u(x)$ и $v(x)$ выполняются следующие шаги:

1 Установить $(u(x), v(x), d(x)) \leftarrow (0, 1, g(x))$.

2 Установить $(u_1(x), v_1(x), d_1(x)) \leftarrow (1, 0, f(x))$.

3 Пока $d_1(x) \neq 0$ выполнить:

1) $q(x) \leftarrow d(x) \operatorname{div} d_1(x)$;
2) $u(x) \leftarrow u(x) + q(x)u_1(x)$;
3) $v(x) \leftarrow v(x) + q(x)v_1(x)$;
4) $d(x) \leftarrow d(x) + q(x)d_1(x)$;
5) $u(x) \leftrightarrow u_1(x)$;
6) $v(x) \leftrightarrow v_1(x)$;
7) $d(x) \leftrightarrow d_1(x)$.

4 Возвратить $d(x)$, $u(x)$, $v(x)$.

Библиография

[1] Лидл Р., Нидеррайтер Г. Конечные поля
М.: Мир, 1988

[2] Rabin M. Probabilistic algorithms in finite fields
SIAM J. Comp. 9, 1980, 273–280
(Вероятностные алгоритмы в конечных полях)

[3] Ben-Or M. Probabilistic algorithms in finite fields
In Proc. 22nd IEEE Symp. Foundations Computer Science, 1981, 394–398
(Вероятностные алгоритмы в конечных полях)

[4] Cohen H. A course in computational algebraic number theory
Berlin, Heidelberg, New York: Springer, 1996
(Курс вычислительной алгебраической теории чисел)

[5] Garner H. The residue number system
IRE Transactions on Electronic Computers, EC-8, 1959, 140–147
(Системы остаточных классов)

Поправка к официальной редакции

В каком месте	Напечатано	Должно быть
Приложение А, первый абзац	$x^{256} + x^{10} + x^5 + x^2 + x + 1$	$x^{256} + x^{10} + x^5 + x^2 + 1$