ГОСУДАРСТВЕННЫЙ СТАНДАРТ
РЕСПУБЛИКИ БЕЛАРУСЬ

СТБ 34.101.31-2020

Информационные технологии и безопасность

АЛГОРИТМЫ ШИФРОВАНИЯ И КОНТРОЛЯ ЦЕЛОСТНОСТИ

Інфармацыйныя тэхналогіі і бяспека

АЛГАРЫТМЫ ШЫФРАВАННЯ І КАНТРОЛЮ ЦЭЛАСНАСЦІ

Госстандарт
Минск

УДК 004.056.55(083.74)(476) МКС 35.240.40

Ключевые слова: криптографический алгоритм, шифрование, имитозащита, аутентифицированное шифрование, хэширование, управление ключами

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»

ВНЕСЕН Оперативно-аналитическим центром при Президенте Республики Беларусь

2 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ постановлением Госстандарта Республики Беларусь от 1 октября 2020 г. № 56

3 ВЗАМЕН СТБ 34.101.31-2011

Издан на русском языке

Содержание

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РЕСПУБЛИКИ БЕЛАРУСЬ

Информационные технологии и безопасность
АЛГОРИТМЫ ШИФРОВАНИЯ И КОНТРОЛЯ ЦЕЛОСТНОСТИ

Інфармацыйныя тэхналогіі і бяспека
АЛГАРЫТМЫ ШЫФРАВАННЯ І КАНТРОЛЮ ЦЭЛАСНАСЦІ

Information technology and security
Encryption and integrity control algorithms

Дата введения 2021-09-01

1 Область применения

Настоящий стандарт устанавливает криптографические алгоритмы шифрования и контроля целостности, а также служебные алгоритмы управления ключами.

Настоящий стандарт применяется при разработке средств криптографической защиты информации.

2 Нормативные ссылки

В настоящем стандарте использована ссылка на следующий технический нормативный правовой акт в области технического нормирования и стандартизации (далее — ТНПА):

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 ссылочный ТНПА заменен (изменен), то при пользовании настоящим стандартом следует руководствоваться действующим взамен ТНПА. Если ссылочный ТНПА отменен без замены, то положение, в котором дана ссылка на него, применяется в части, не затрагивающей эту ссылку.

3 Термины и определения

В настоящем стандарте применяются следующие термины с соответствующими определениями:

3.1 аутентифицированное шифрование: Одновременные шифрование и имитозащита.

3.2 блок: Двоичное слово длины 128.

Примечание — При разбиении двоичного слова на блоки последний блок может быть неполным.

3.3 заголовок ключа: Блок, содержащий открытые атрибуты ключа.

3.4 зашифрование: Преобразование сообщения, направленное на обеспечение его конфиденциальности, которое выполняется с использованием ключа.

3.5 имитовставка: Двоичное слово, которое определяется по сообщению с использованием ключа и служит для контроля целостности и подлинности сообщения.

3.6 имитозащита: Контроль целостности и подлинности сообщений, который реализуется путем выработки и проверки имитовставок.

3.7 конфиденциальность: Гарантия того, что сообщения доступны для понимания или использования только тем сторонам, которым они предназначены.

3.8 октет: Двоичное слово длины 8.

3.9 подлинность: Гарантия того, что сторона действительно является владельцем, создателем или отправителем определенного сообщения.

3.10 преобразование ключа: Построение по исходному ключу набора новых ключей с различными заголовками.

3.11 расширение ключа: Дополнение ключа новыми символами до получения ключа определенной длины.

3.12 расшифрование: Преобразование, обратное зашифрованию.

3.13 (секретный) ключ: Параметр, который управляет операциями шифрования и имитозащиты и который известен только определенным сторонам.

3.14 синхропосылка: Открытые входные данные криптографического алгоритма, которые обеспечивают уникальность результатов криптографического преобразования на фиксированном ключе.

3.15 снятие защиты: Проверка имитовставок и расшифрование.

3.16 сообщение: Двоичное слово конечной длины.

3.17 установка защиты: Зашифрование и вычисление имитовставок.

3.18 хэш-значение: Двоичное слово фиксированной длины, которое определяется по сообщению без использования ключа и служит для контроля целостности сообщения и для представления сообщения в (необратимо) сжатой форме.

3.19 хэширование: Выработка хэш-значений.

3.20 целостность: Гарантия того, что в сообщение не внесены изменения при его хранении, передаче и обработке.

3.21 шифрование: Зашифрование или расшифрование.

4 Обозначения

4.1 Список обозначений

$\perp$	специальный объект или ситуация: пустое слово, игнорируемая переменная, ошибка;
Σ^n	множество всех слов длины n в алфавите Σ;
Σ^*	множество всех слов конечной длины в алфавите Σ (включая пустое слово $\perp$ длины 0);
$\lvert u\rvert$	длина слова $u \in \Sigma^*$;
Σ^{n*}	множество всех слов из Σ^*, длина которых кратна n;
α^n	для $\alpha \in \Sigma$ слово из n экземпляров α;
$\texttt{Lo}(u, m)$	для $u \in \Sigma^*$ и $m \leqslant \lvert u\rvert$ слово из первых m символов u;
$u \parallel v$	для $u = u_1u_2\ldots u_n \in \Sigma^n$ и $v = v_1v_2\ldots v_m \in \Sigma^m$ слово $u_1u_2\ldots u_nv_1v_2\ldots v_m$ (конкатенация);
$\texttt{Split}(u, m)$	для $u \in \Sigma^*$ и натурального m представление u в виде набора $(u_1, u_2, \ldots, u_n)$ фрагментов $u_i \in \Sigma^*$ таких, что $u = u_1 \parallel u_2 \parallel \ldots \parallel u_n$, $\lvert u_1\rvert = \lvert u_2\rvert = \ldots = \lvert u_{n-1}\rvert = m$ и $0 < \lvert u_n\rvert \leqslant m$, причем набор пуст $(n = 0)$, если $u = \perp$;
$\texttt{Split1}(u, m)$	$\texttt{Split}(u, m)$, если $u \neq \perp$, и одноэлементный набор $(\perp)$ в противном случае;
$U \bmod m$	для целого U и натурального m остаток от деления U на m;

$\mathbb{Z}_m$	алфавит $\{0, 1, \ldots, m-1\}$, $m \geqslant 2$;
$u \oplus v$	для $u = u_1 u_2 \ldots u_n \in \mathbb{Z}_m^n$ и $v = v_1 v_2 \ldots v_n \in \mathbb{Z}_m^n$ слово $w = w_1 w_2 \ldots w_n \in \mathbb{Z}_m^n$ из символов $w_i = (u_i + v_i) \bmod m$;
$u \ominus v$	для $u, v \in \mathbb{Z}_m^n$ слово $w \in \mathbb{Z}_m^n$ такое, что $u = v \oplus w$ (при $m = 2$ символы $\oplus$ и $\ominus$ эквивалентны);
$\lfloor u \rceil_m$	для $u = u_1 u_2 \ldots u_n \in \mathbb{Z}_m^n$ число $U = u_1 + m u_2 + \ldots + m^{n-1} u_n$;
$\lfloor u \rceil$	а) для октета $u = u_1 u_2 \ldots u_8 \in \{0,1\}^8$ число $u_1 2^7 + u_2 2^6 + \ldots + u_8$, б) для $u = u_1 \parallel u_2 \parallel \ldots \parallel u_n$, $u_i \in \{0,1\}^8$, число $\lfloor \lfloor u_1 \rceil \lfloor u_2 \rceil \ldots \lfloor u_n \rceil \rceil_{256}$;
$\langle U \rangle_{m,n}$	для неотрицательного целого U и натуральных m, n слово $u \in \mathbb{Z}_m^n$ такое, что $\lfloor u \rceil_m = U \bmod m^n$;
$\langle U \rangle_{8n}$	для неотрицательного целого U и натурального n слово $u \in \{0,1\}^{8n}$ такое, что $\lfloor u \rceil = U \bmod 2^{8n}$;
$\mathtt{01234...}_{16}$	представление $u \in \{0,1\}^{4*}$ шестнадцатеричным словом, при котором последовательным четырем символам u соответствует один шестнадцатеричный символ (например, $10100010 = \mathtt{A2}_{16}$);
$u \boxplus v$	для $u, v \in \{0,1\}^{8n}$ слово $\langle \lfloor u \rceil + \lfloor v \rceil \rangle_{8n}$;
$u \boxminus v$	для $u, v \in \{0,1\}^{8n}$ слово $w \in \{0,1\}^{8n}$ такое, что $u = v \boxplus w$;
$\lfloor z \rfloor$	для вещественного z максимальное целое, не превосходящее z;
$\lceil z \rceil$	для вещественного z минимальное целое, не меньшее z;
$\mathtt{ShLo}(u)$	для $u \in \{0,1\}^{8n}$ слово $\langle \lfloor \lfloor u \rceil / 2 \rfloor \rangle_{8n}$;
$\mathtt{ShHi}(u)$	для $u \in \{0,1\}^{8n}$ слово $\langle 2 \lfloor u \rceil \rangle_{8n}$;
$\varphi^r(u)$	для слова u и преобразования φ результат r-кратного действия φ на u (например, $\mathtt{ShLo}^r(u)$ — результат r-кратного действия $\mathtt{ShLo}$);
$\mathtt{RotHi}(u)$	для $u \in \{0,1\}^{8n}$ слово $\mathtt{ShHi}(u) \oplus \mathtt{ShLo}^{8n-1}(u)$;
$\mathbb{F}_2$	поле из двух элементов 0 и 1;
$\mathbb{F}_2[x]$	кольцо многочленов над полем $\mathbb{F}_2$;
$u(x)$	а) для октета $u = u_1 u_2 \ldots u_8 \in \{0,1\}^8$ многочлен $u_1 x^7 + u_2 x^6 + \ldots + u_8$, б) для $u = u_1 \parallel u_2 \parallel \ldots \parallel u_n$, $u_i \in \{0,1\}^8$, многочлен $u_1(x) + x^8 u_2(x) + \ldots + x^{8(n-1)} u_n(x)$;
$u(x) \bmod f(x)$	для $u(x) \in \mathbb{F}_2[x]$ и ненулевого $f(x) \in \mathbb{F}_2[x]$ остаток от деления $u(x)$ на $f(x)$;
$u * v$	для $u, v \in \{0,1\}^{128}$ слово $w \in \{0,1\}^{128}$ такое, что $w(x) = u(x)v(x) \bmod x^{128} + x^7 + x^2 + x + 1$;
$\mathtt{alg}(u_1, u_2, \ldots)$	вызов алгоритма $\mathtt{alg}$ с входными данными $u_1, u_2, \ldots$;
$a \leftarrow u$	присвоение переменной a значения u;
$a \leftrightarrow b$	перестановка значений переменных a и b;
$(a_1, a_2) \leftarrow (u_1, u_2)$	присвоение переменной a_1 значения u_1, переменной a_2 значения u_2;
$(\perp, a_2) \leftarrow (u_1, u_2)$	то же самое, что $a_2 \leftarrow u_2$ (например, u_1 и u_2 — выходы алгоритма и выход u_1 игнорируется).

4.2 Пояснения к обозначениям

4.2.1 Слова

Слово в алфавите Σ представляет собой последовательность символов этого алфавита. Символы нумеруются слева направо от единицы. Примеры алфавитов: $\mathbb{Z}_2 = \{0, 1\}$ (двоичный), $\mathbb{Z}_{10}$ (десятичный), $\mathbb{Z}_{16}$ (шестнадцатеричный), $\mathbb{Z}_{256}$ (байтовый).

Символы шестнадцатеричного алфавита (числа от 0 до 15) обозначаются знаками $\mathtt{0}_{16}, \mathtt{1}_{16}, \ldots, \mathtt{F}_{16}$. При записи шестнадцатеричного слова индекс 16 после всех символов, кроме последнего, исключается.

Слово $u = u_1 u_2 \ldots u_n$ в алфавите $\mathbb{Z}_m$ является записью числа $U = \lfloor u \rceil_m$ в системе счисления по основанию m. При этом первый символ слова является младшим, последний — старшим. Таким образом, используется соглашение «от младших к старшим» (little-endian), распространенное для многих современных процессоров при $m = 256$.

4.2.2 Двоичные слова

В настоящем подразделе в качестве примера рассматривается двоичное слово

$$w = 10110001100101001011101011001000.$$

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

Двоичное слово разбивается на тетрады из четверок последовательных двоичных символов. Тетрады кодируются шестнадцатеричными символами по правилам, заданным в таблице 1.

Таблица 1

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

Например, слово w кодируется следующим образом:

$$\mathtt{B194BAC8}_{16}.$$

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

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

Октету $u = u_1 u_2 \ldots u_8$ ставится в соответствие байт — число $\lfloor u \rceil = 2^7 u_1 + 2^6 u_2 + \ldots + u_8$. 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: сначала строится слово из байтов, затем применяется функция $\lfloor \cdot \rceil_{256}$. Например:

$$\lfloor w \rceil = \lfloor 177, 148, 186, 200 \rceil_{256} = 177 + 2^8 \cdot 148 + 2^{16} \cdot 186 + 2^{24} \cdot 200 = 3367670961.$$

При отождествлении слов с числами удобно представить себе гипотетический регистр, разрядность которого совпадает с длиной слова. В самый правый октет регистра загружается первый октет слова, во второй справа октет регистра — второй октет слова и

т. д., пока, наконец, в самый левый октет регистра не загружается последний октет слова. Например, для w содержимое регистра имеет вид:

$$\texttt{C8BA94B1}_{16} = 11001000101110101001010010110001.$$

При таком представлении операции ShLo, ShHi, RotHi состоят в сдвигах содержимого регистра: ShLo — вправо (в сторону младших разрядов), ShHi — влево (в сторону старших разрядов) и RotHi — циклически влево, причем при сдвигах ShLo и ShHi в освободившиеся разряды регистров записываются нули. Например, предыдущий регистр изменяется при сдвигах следующим образом:

$$\begin{aligned} \texttt{ShLo}: &\ \texttt{645D4A58}_{16} = 01100100010111010100101001011000,\\ \texttt{ShHi}: &\ \texttt{91752962}_{16} = 10010001011101010010100101100010,\\ \texttt{RotHi}: &\ \texttt{91752963}_{16} = 10010001011101010010100101100011. \end{aligned}$$

Выгружая из регистра октеты слева направо, получаем следующие результаты:

$$\begin{aligned} \texttt{ShLo}(w) &= \texttt{584A5D64}_{16},\\ \texttt{ShHi}(w) &= \texttt{62297591}_{16},\\ \texttt{RotHi}(w) &= \texttt{63297591}_{16}. \end{aligned}$$

Перестановки октетов при загрузке слова в регистр и при выгрузке из регистра в современных процессорах выполняются неявно.

При сдвигах на число позиций, кратное 8, операции ShLo, ShHi, RotHi интерпретируются намного проще и состоят в сдвиге октетов исходного слова: при ShLo — в сторону первых октетов, при ShHi — в сторону последних октетов, при RotHi — циклически в сторону последних октетов. Например:

$$\begin{aligned} \texttt{ShLo}^8(w) &= \texttt{94BAC800}_{16},\\ \texttt{ShHi}^8(w) &= \texttt{00B194BA}_{16},\\ \texttt{RotHi}^8(w) &= \texttt{C8B194BA}_{16}. \end{aligned}$$

4.2.3 Двоичные 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. Как и при представлении слов числами 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. 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.

Операция $*$ состоит в умножении слов как многочленов с заменой результата умножения на его остаток от деления на $f(x) = x^{128} + x^7 + x^2 + x + 1$. Выбранный многочлен $f(x)$ является неприводимым (его нельзя представить в виде произведения многочленов меньших степеней). Поэтому операция $*$ задает умножение слов как элементов поля из 2^{128} элементов (подробнее см. [1]).

4.3 Запись перечислений

При записи последовательности $u_1, u_2, \ldots, u_n$ допускается, если не оговорено иное, выполнение неравенства $n < 2$. При $n = 0$ идет речь о пустой последовательности, а при $n = 1$ — об одноэлементной последовательности u_1.

Аналогичные соглашения распространяются на запись конкатенации нескольких слов, суммы нескольких слагаемых, итератора цикла. Например:

– слово $u_1 \parallel u_2 \parallel \ldots \parallel u_n$ является пустым при $n = 0$ и состоит из единственного фрагмента u_1 при $n = 1$;

– сумма $u_1 \oplus u_2 \oplus \ldots \oplus u_n$ равняется u_1 при $n = 1$;

– тело цикла «для $i = 1, 2, \ldots, n$ выполнить ...» не выполняется ни разу, если $n = 0$, и выполняется один раз, если $n = 1$.

5 Общие положения

5.1 Назначение

Настоящий стандарт определяет семейство криптографических алгоритмов, предназначенных для обеспечения конфиденциальности и контроля целостности данных. Обрабатываемыми данными являются двоичные слова (сообщения).

Криптографические алгоритмы стандарта построены на основе базовых алгоритмов шифрования блока, шифрования широкого блока и криптографического сжатия. Базовые алгоритмы определяются в 6.

Криптографические алгоритмы шифрования и контроля целостности делятся на десять групп:

1) алгоритмы шифрования в режиме простой замены (см. 7.1);
2) алгоритмы шифрования в режиме сцепления блоков (см. 7.2);
3) алгоритмы шифрования в режиме гаммирования с обратной связью (см. 7.3);
4) алгоритмы шифрования в режиме счетчика (см. 7.4);
5) алгоритм выработки имитовставки (см. 7.5);
6) алгоритмы аутентифицированного шифрования данных (см. 7.6);
7) алгоритмы аутентифицированного шифрования ключа (см. 7.7);
8) алгоритм хэширования (см. 7.8);
9) алгоритмы дискового шифрования (см. 7.9);
10) алгоритмы шифрования с сохранением формата (см. 7.10).

Первые четыре группы предназначены для обеспечения конфиденциальности сообщений. Каждая группа включает алгоритм зашифрования и алгоритм расшифрования. Стороны, располагающие общим ключом, могут организовать конфиденциальный обмен сообщениями путем их зашифрования перед отправкой и расшифрования после получения. В режимах простой замены и сцепления блоков шифруются сообщения, которые содержат хотя бы один блок, а в режимах гаммирования с обратной связью и счетчика — сообщения произвольной длины. В режиме простой замены следует зашифровывать только высокоэнтропийные данные: ключи, случайные или псевдослучайные числа.

Пятый алгоритм предназначен для контроля целостности сообщений с помощью имитовставок — контрольных слов, которые определяются с использованием ключа. Стороны, располагающие общим ключом, могут организовать контроль целостности при обмене сообщениями путем добавления к ним имитовставок при отправке и проверки имитовставок при получении. Проверка имитовставок дополнительно позволяет стороне-получателю убедиться в том, что сторона-отправитель знает ключ, т. е. позволяет проверить подлинность сообщений.

Шестая и седьмая группы предназначены для обеспечения конфиденциальности и контроля целостности сообщений. Контроль целостности снова сопровождается контролем подлинности. Каждая группа включает алгоритмы установки и снятия защиты.

В алгоритмах шестой группы контролируется целостность сообщения в паре с ассоциированными открытыми данными. При установке защиты вычисляется имитовставка пары и одновременно выполняется зашифрование сообщения. При снятии защиты имитовставка проверяется и, если проверка прошла успешно, сообщение расшифровывается.

В шестой группе реализованы две схемы аутентифицированного шифрования. Алгоритмы первой схемы хорошо совместимы с алгоритмами шифрования в режиме счетчика, фактически образуя расширение данных алгоритмов. Алгоритмы второй схемы более эффективны: при обработке одних и тех же данных требуется на одно шифрование блока меньше.

В алгоритмах седьмой группы длина защищаемого сообщения должна быть сразу известна, эти алгоритмы следует применять для защиты ключей. Защищаемый ключ сопровождается открытым заголовком, который содержит открытые атрибуты ключа и одновременно является контрольным значением при проверке целостности. Могут использоваться фиксированные постоянные заголовки, которые служат только для контроля целостности. При установке защиты ключ зашифровывается вместе со своим заголовком и формируется слово, которое является одновременно защищенным ключом и имитовставкой ключа. При снятии защиты выполняется обратное преобразование и расшифрованный заголовок сравнивается с контрольным.

Восьмой алгоритм предназначен для вычисления хэш-значений — контрольных слов, которые определяются без использования ключа. Стороны могут организовать контроль целостности сообщений путем сравнения их хэш-значений с достоверными контрольными хэш-значениями. Изменение сообщения с высокой вероятностью приводит к изменению соответствующего хэш-значения, и поэтому хэш-значения могут использоваться вместо самих сообщений, например в системах электронной цифровой подписи.

Алгоритмы девятой группы предназначены для шифрования содержимого дисков и других накопителей данных. Диск разбивается на секторы, состоящие из полного числа блоков. Каждый сектор имеет уникальный номер, который учитывается при шифровании. Содержимое сектора может обновляться и зашифровываться многократно. Объем сектора при шифровании не меняется.

Шифрование может быть организовано двумя способами. При блоковом шифровании блоки секторов обрабатываются независимо друг от друга, способ обработки зависит от номера сектора и номера блока в секторе. При секторном шифровании все блоки сектора обрабатываются вместе, способ обработки зависит от номера сектора. При выборе способа шифрования следует учитывать, что блоковое шифрование выполняется примерно в 2 раза быстрее секторного, однако при повторном шифровании того же блока в той же позиции того же сектора будет получен тот же результат.

Алгоритмы десятой группы шифруют слова длины $n \geqslant 2$ в алфавите из m символов. Символы алфавита кодируются числами от 0 до $m - 1$ и, таким образом, алфавит представляет собой множество $\mathbb{Z}_m$. Правила кодирования определяются за рамками настоящего стандарта. Алгоритмы сохраняют формат: результатом шифрования слова в алфавите $\mathbb{Z}_m$ является слово в том же алфавите той же длины.

Дополнительно в разделе 8 определяются служебные алгоритмы расширения и преобразования ключа, предназначенные для создания и модификации ключей шифрования и имитозащиты.

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

В приложении Б приводится модуль абстрактно-синтаксической нотации версии 1 (АСН.1), определенной в ГОСТ 34.973. Модуль задает идентификаторы алгоритмов стандарта и описывает форматы параметров алгоритмов. Рекомендуется использовать модуль при встраивании алгоритмов в информационные системы, в которых также используется АСН.1.

5.2 Ключ

В алгоритмах шифрования и имитозащиты используется ключ $K \in \{0,1\}^{256}$. Ключ должен вырабатываться без возможности предсказания, распространяться с соблюдением мер конфиденциальности и храниться в секрете.

Разрешается использовать ключ K, полученный в результате расширения короткого ключа длины 128 или 192. При этом должен использоваться алгоритм расширения, заданный в 8.1.

Один и тот же ключ не должен использоваться в алгоритмах различных групп. Ключ аутентифицированного шифрования данных по схеме 1 не должен использоваться как ключ схемы 2 и наоборот. Ключ блокового дискового шифрования не должен использоваться как ключ секторного и наоборот.

Ключи шифрования данных должны применяться в соответствии с квотами, определенными в приложении В. Ключ шифрования с сохранением формата не должен использоваться более m^n раз для зашифрования (даже с разными синхропосылками) слов длины n в алфавите размера m.

В 8.2 определяется алгоритм преобразования ключа, с помощью которого по исходному ключу можно строить наборы новых ключей, которые, в свою очередь, также можно преобразовывать. Алгоритм преобразования может применяться для создания семейств ключей различного назначения, в том числе для использования в алгоритмах шифрования и имитозащиты различных групп. Кроме этого, алгоритм преобразования позволяет организовать обновление ключей при исчерпании лимитов времени их использования или объема обработанных на ключах данных.

Ключам, которые требуется получить в результате преобразования, ставятся в соответствие заголовки $I \in \{0,1\}^{128}$, содержащие открытые атрибуты ключей, например, тип или назначение. Кроме этого, ключам назначаются уровни $D \in \{0,1\}^{96}$. Исходному ключу назначается уровень $\langle 0 \rangle_{96}$. Алгоритм преобразования по ключу уровня D и заголовку I строит новый ключ уровня $D \boxplus \langle 1 \rangle_{96}$ с заголовком I. Многократное применение алгоритма к одному ключу с различными заголовками I соответствует генерации семейства ключей различного назначения. Последовательное применение алгоритма к одному ключу с сохранением заголовка I соответствует обновлению ключа.

5.3 Синхропосылка

При шифровании в режимах сцепления блоков, гаммирования с обратной связью и счетчика, аутентифицированном шифровании данных, дисковом шифровании, шифровании с сохранением формата используется синхропосылка $S \in \{0,1\}^{128}$.

Синхропосылка не является секретным параметром, может добавляться к зашифрованному сообщению и передаваться вместе с ним.

При шифровании в режимах гаммирования с обратной связью и счетчика, а также при аутентифицированном шифровании данных должны использоваться уникальные синхропосылки. Уникальность означает, что при зашифровании или установке защиты

на одном и том же ключе либо используются заведомо различные синхропосылки, либо вероятность совпадения синхропосылок пренебрежимо мала.

В режиме сцепления блоков синхропосылка должна быть не только уникальной, но и непредсказуемой. Непредсказуемость означает, что синхропосылки формируются случайно или по секретным правилам и вероятность угадать, какая синхропосылка будет использоваться, пренебрежимо мала.

Синхропосылки можно вырабатывать случайным или псевдослучайным методом, строить по отметкам времени, значениям монотонного счетчика, неповторяющимся номерам сообщений и др. В режиме сцепления блоков предсказуемые значения не должны использоваться напрямую для построения синхропосылок, а должны предварительно зашифроваться на том же ключе, который используется для шифрования сообщений.

При дисковом шифровании синхропосылка строится по номеру того сектора, который зашифровывается. Всякий раз при шифровании сектора используется одна и та же синхропосылка. Синхропосылки различных секторов должны различаться.

При шифровании с сохранением формата синхропосылка по умолчанию нулевая: $S = 0^{128}$. Уникальные синхропосылки следует использовать в ситуациях, когда размер алфавита m и длина шифруемых слов n невелики. В этих ситуациях у противника имеется возможность накопления результатов зашифрования значительной части из m^n возможных открытых сообщений. Уникальные синхропосылки не позволяют использовать накопленные данные для расшифрования будущих сообщений.

5.4 Имитовставка

В алгоритме выработки имитовставки и в алгоритмах аутентифицированного шифрования данных вычисляется либо проверяется имитовставка $T \in \{0,1\}^{64}$.

Если требуются не все, а $n < 64$ символов имитовставки, то должны использоваться первые n символов. При выборе n следует учитывать, что при навязывании ложного сообщения вероятность угадать с одной попытки его имитовставку, не зная ключ, равняется 2^{-n}.

5.5 Хэш-значение

В алгоритме хэширования вычисляется хэш-значение $Y \in \{0,1\}^{256}$.

Если требуются не все, а $n < 256$ символов хэш-значения, то должны использоваться первые n символов. При выборе n следует учитывать, что для определения сообщения с заданным хэш-значением требуется выполнить порядка 2^n операций, а для определения двух различных сообщений с одинаковыми хэш-значениями требуется выполнить порядка $2^{n/2}$ операций.

5.6 Интерфейсы

Определение группы алгоритмов начинается с назначения алгоритмам коротких имен и описания соглашений о входных и выходных данных алгоритмов. В совокупности такая вводная информация называется интерфейсом. Например, алгоритму шифрования блока, определенному в 6.1, назначается имя `belt-block` и объявляется, что входными данными являются блок $X \in \{0,1\}^{128}$ и ключ $K \in \{0,1\}^{256}$, а выходными — блок $Y \in \{0,1\}^{128}$.

Как правило, интерфейс каждой группы описывает два алгоритма: зашифрования или установки защиты и расшифрования или снятия защиты. Например, алгоритм зашифрования `belt-block` сопровождается алгоритмом расшифрования $\texttt{belt-block}^{-1}$.

С помощью интерфейсов можно лаконично и однозначно описывать вызов одних алгоритмов в других. Например, вызов `belt-block` записывается как $Y \leftarrow \texttt{belt-block}(X, K)$.

Алгоритм может вызываться в других алгоритмах настоящего стандарта или в алгоритмах других стандартов. И наоборот, алгоритм может вызывать другие алгоритмы. В последнем случае интерфейс содержит перечень задействованных алгоритмов.

Интерфейсы описывают алгоритмы, в которых используются 256-битовые ключи. Для указания на использование ключей других длин к короткому имени алгоритма следует добавить длину ключа: `belt-block128`, `belt-block192`$^{-1}$ и т. д. Имена с суффиксом `256` (например, `belt-block256`) являются синонимами первоначальных имен и могут быть использованы вместо них.

5.7 Переменные

В настоящем стандарте переменные алгоритма, которые явно объявляются перед определением его шагов, могут содержать критические данные, например, фрагмент ключа или промежуточный результат вычислений, который упрощает определение этого фрагмента. Речь идет в том числе о переменных алгоритма хэширования, который может использоваться для обработки критических данных.

При реализации алгоритма объявленные переменные следует очищать после использования. При организации очистки необходимо учитывать особенности реализации, например ситуации, когда переменная алгоритма представляется несколькими переменными реализации.

Очистка переменных алгоритма может не выполняться, если алгоритм все-таки не обрабатывает и не возвращает критические данные. Переменные могут также не очищаться, если алгоритм выполняется в защищенной среде и доступ к переменным блокируется аппаратными или другими способами.

6 Базовые алгоритмы

6.1 Шифрование блока

6.1.1 Интерфейс

Шифрование блока задается алгоритмами зашифрования `belt-block` и расшифрования `belt-block`$^{-1}$.

Входными данными `belt-block` являются блок $X \in \{0,1\}^{128}$ и ключ $K \in \{0,1\}^{256}$. Выходными данными является зашифрованный блок $Y \in \{0,1\}^{128}$.

Входными данными `belt-block`$^{-1}$ являются зашифрованный блок $Y \in \{0,1\}^{128}$ и ключ $K \in \{0,1\}^{256}$. Выходными данными является расшифрованный блок $X \in \{0,1\}^{128}$.

6.1.2 Вспомогательные преобразования и переменные

Подстановка H**.** Подстановка $H\colon \{0,1\}^8 \to \{0,1\}^8$ задается таблицей 2. В таблице входы (прообразы) и выходы (образы) H записываются в шестнадцатеричном виде. Для входного октета $u = \mathtt{IJ}_{16}$ соответствующий выходной октет $H(u)$ находится на пересечении строки `I` и столбца `J`. Например, $H(\mathtt{A2}_{16}) = \mathtt{9B}_{16}$.

Преобразования G_r $(r = 5, 13, 21)$**.** Преобразование $G_r\colon \{0,1\}^{32} \to \{0,1\}^{32}$ ставит в соответствие слову $u = u_1 \parallel u_2 \parallel u_3 \parallel u_4$, $u_i \in \{0,1\}^8$, слово

$$G_r(u) = \mathtt{RotHi}^r\left(H(u_1) \parallel H(u_2) \parallel H(u_3) \parallel H(u_4)\right).$$

Переменные. Используются переменные $a, b, c, d, e \in \{0,1\}^{32}$.

6.1.3 Алгоритм зашифрования

Зашифрование `belt-block`(X, K) выполняется следующим образом:

Таблица 2 — Подстановка H

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	B1	94	BA	C8	0A	08	F5	3B	36	6D	00	8E	58	4A	5D	E4
1	85	04	FA	9D	1B	B6	C7	AC	25	2E	72	C2	02	FD	CE	0D
2	5B	E3	D6	12	17	B9	61	81	FE	67	86	AD	71	6B	89	0B
3	5C	B0	C0	FF	33	C3	56	B8	35	C4	05	AE	D8	E0	7F	99
4	E1	2B	DC	1A	E2	82	57	EC	70	3F	CC	F0	95	EE	8D	F1
5	C1	AB	76	38	9F	E6	78	CA	F7	C6	F8	60	D5	BB	9C	4F
6	F3	3C	65	7B	63	7C	30	6A	DD	4E	A7	79	9E	B2	3D	31
7	3E	98	B5	6E	27	D3	BC	CF	59	1E	18	1F	4C	5A	B7	93
8	E9	DE	E7	2C	8F	0C	0F	A6	2D	DB	49	F4	6F	73	96	47
9	06	07	53	16	ED	24	7A	37	39	CB	A3	83	03	A9	8B	F6
A	92	BD	9B	1C	E5	D1	41	01	54	45	FB	C9	5E	4D	0E	F2
B	68	20	80	AA	22	7D	64	2F	26	87	F9	34	90	40	55	11
C	BE	32	97	13	43	FC	9A	48	A0	2A	88	5F	19	4B	09	A1
D	7E	CD	A4	D0	15	44	AF	8C	A5	84	50	BF	66	D2	E8	8A
E	A2	D7	46	52	42	A8	DF	B3	69	74	C5	51	EB	23	29	21
F	D4	EF	D9	B4	3A	62	28	75	91	14	10	EA	77	6C	DA	1D

1 Определить $(X_1, X_2, X_3, X_4) = \texttt{Split}(X, 32)$.
2 Определить $(K_1, K_2, \ldots, K_8) = \texttt{Split}(K, 32)$.
3 Обозначить $k[i] = K_{(i-1) \bmod 8+1}$, $i = 1, 2, \ldots, 56$.
4 Установить $a \leftarrow X_1$, $b \leftarrow X_2$, $c \leftarrow X_3$, $d \leftarrow X_4$.
5 Для $i = 1, 2, \ldots, 8$ выполнить (см. рисунок 1):
  1) $b \leftarrow b \oplus G_5(a \boxplus k[7i-6])$;
  2) $c \leftarrow c \oplus G_{21}(d \boxplus k[7i-5])$;
  3) $a \leftarrow a \boxminus G_{13}(b \boxplus k[7i-4])$;
  4) $e \leftarrow G_{21}(b \boxplus c \boxplus k[7i-3]) \oplus \langle i \rangle_{32}$;
  5) $b \leftarrow b \boxplus e$;
  6) $c \leftarrow c \boxminus e$;
  7) $d \leftarrow d \boxplus G_{13}(c \boxplus k[7i-2])$;
  8) $b \leftarrow b \oplus G_{21}(a \boxplus k[(7i-1])$;
  9) $c \leftarrow c \oplus G_5(d \boxplus k[7i])$;
  10) $a \leftrightarrow b$;
  11) $c \leftrightarrow d$;
  12) $b \leftrightarrow c$.
6 Установить $Y \leftarrow b \parallel d \parallel a \parallel c$.
7 Возвратить Y.

6.1.4 Алгоритм расшифрования

Расшифрование $\texttt{belt-block}^{-1}(Y, K)$ выполняется следующим образом:
1 Определить $(Y_1, Y_2, Y_3, Y_4) = \texttt{Split}(Y, 32)$.
2 Определить $(K_1, K_2, \ldots, K_8) = \texttt{Split}(K, 32)$.
3 Обозначить $k[i] = K_{(i-1) \bmod 8+1}$, $i = 1, 2, \ldots, 56$.
4 Установить $a \leftarrow Y_1$, $b \leftarrow Y_2$, $c \leftarrow Y_3$, $d \leftarrow Y_4$.
5 Для $i = 8, 7, \ldots, 1$ выполнить:
  1) $b \leftarrow b \oplus G_5(a \boxplus k[7i])$;
  2) $c \leftarrow c \oplus G_{21}(d \boxplus k[7i-1])$;
  3) $a \leftarrow a \boxminus G_{13}(b \boxplus k[7i-2])$;

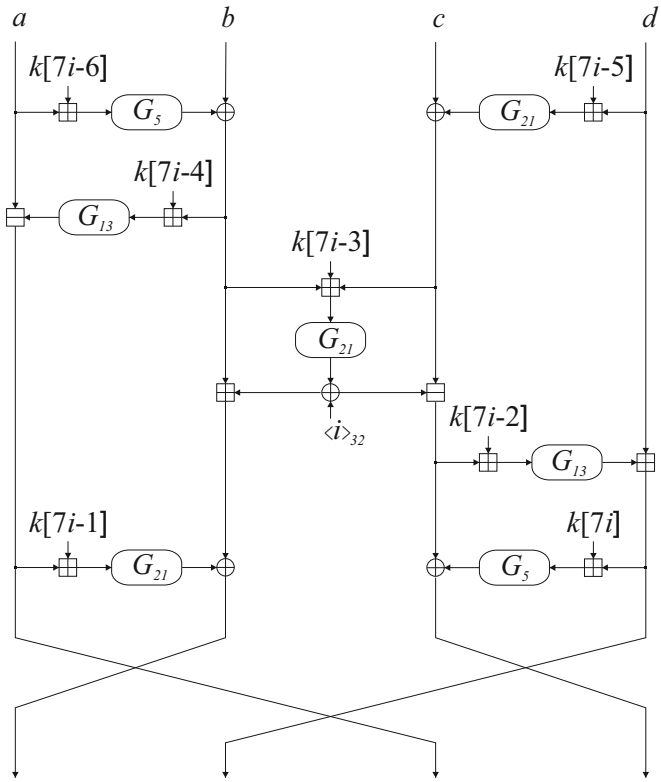

Рисунок 1 — Вычисления на i-м такте зашифрования

4) $e \leftarrow G_{21}(b \boxplus c \boxplus k[7i-3]) \oplus \langle i \rangle_{32}$;
5) $b \leftarrow b \boxplus e$;
6) $c \leftarrow c \boxminus e$;
7) $d \leftarrow d \boxplus G_{13}(c \boxplus k[7i-4])$;
8) $b \leftarrow b \oplus G_{21}(a \boxplus k[7i-5])$;
9) $c \leftarrow c \oplus G_{5}(d \boxplus k[7i-6])$;
10) $a \leftrightarrow b$;
11) $c \leftrightarrow d$;
12) $a \leftrightarrow d$.

6 Установить $X \leftarrow c \parallel a \parallel d \parallel b$.

7 Возвратить X.

6.2 Шифрование широкого блока

6.2.1 Интерфейс

Шифрование широкого блока задается алгоритмами зашифрования `belt-wblock` и расшифрования `belt-wblock`$^{-1}$.

Входными данными `belt-wblock` являются слово $X \in \{0,1\}^{8*}$ и ключ $K \in \{0,1\}^{256}$. Длина X должна быть не меньше 256. Выходными данными является зашифрованное слово $Y \in \{0,1\}^{|X|}$.

Входными данными $\texttt{belt-wblock}^{-1}$ являются зашифрованное слово $Y \in \{0,1\}^{8*}$ и ключ $K \in \{0,1\}^{256}$. Длина Y должна быть не меньше 256. Выходными данными является расшифрованное слово $X \in \{0,1\}^{|Y|}$.

Используется алгоритм `belt-block`, определенный в 6.1.

6.2.2 Переменные

Используются переменные $r \in \{0,1\}^{8*}$ и $s \in \{0,1\}^{128}$. Длина r совпадает с длиной входного слова X или Y.

6.2.3 Алгоритм зашифрования

Зашифрование $\texttt{belt-wblock}(X, K)$ выполняется следующим образом:

1 Установить $r \leftarrow X$.
2 Представить r двумя способами:
   1) в виде набора блоков $(r_1, r_2, \ldots, r_n) = \texttt{Split}(r, 128)$;
   2) в форме $r = r^{**} \parallel r^*$, где $|r^*| = 128$.
3 Для $i = 1, 2, \ldots, 2n$ выполнить:
   1) $s \leftarrow r_1 \oplus r_2 \oplus \ldots \oplus r_{n-1}$;
   2) $r^* \leftarrow r^* \oplus \texttt{belt-block}(s, K) \oplus \langle i \rangle_{128}$;
   3) $r \leftarrow \texttt{ShLo}^{128}(r)$;
   4) $r^* \leftarrow s$.
4 Установить $Y \leftarrow r$.
5 Возвратить Y.

6.2.4 Алгоритм расшифрования

Расшифрование $\texttt{belt-wblock}^{-1}(Y, K)$ выполняется следующим образом:

1 Установить $r \leftarrow Y$.
2 Представить r двумя способами:
   1) в виде набора блоков $(r_1, r_2, \ldots, r_n) = \texttt{Split}(r, 128)$;
   2) в форме $r = r^{**} \parallel r^*$, где $|r^*| = 128$.
3 Для $i = 2n, \ldots, 2, 1$ выполнить:
   1) $s \leftarrow r^*$;
   2) $r \leftarrow \texttt{ShHi}^{128}(r)$;
   3) $r^* \leftarrow r^* \oplus \texttt{belt-block}(s, K) \oplus \langle i \rangle_{128}$;
   4) $r_1 \leftarrow s \oplus r_2 \oplus \ldots \oplus r_{n-1}$.
4 Установить $X \leftarrow r$.
5 Возвратить X.

6.3 Сжатие

6.3.1 Интерфейс

Сжатие задается алгоритмом `belt-compress`.

Входными данными `belt-compress` является слово $X \in \{0,1\}^{512}$. Выходными данными являются слова $S \in \{0,1\}^{128}$ и $Y \in \{0,1\}^{256}$. Слово S является промежуточным результатом сжатия X, этот выход может игнорироваться. Слово Y является окончательным результатом сжатия.

Используется алгоритм `belt-block`, определенный в 6.1.

6.3.2 Алгоритм сжатия

Сжатие $\texttt{belt-compress}(X)$ выполняется следующим образом:

1 Определить $(X_1, X_2, X_3, X_4) = \texttt{Split}(X, 128)$.

2 Установить $S \leftarrow \texttt{belt-block}(X_3 \oplus X_4, X_1 \,\|\, X_2) \oplus X_3 \oplus X_4$.
3 Установить $Y_1 \leftarrow \texttt{belt-block}(X_1, S \,\|\, X_4) \oplus X_1$.
4 Установить $Y_2 \leftarrow \texttt{belt-block}(X_2, (S \oplus 1^{128}) \,\|\, X_3) \oplus X_2$.
5 Возвратить (S, Y), где $Y = Y_1 \,\|\, Y_2$.

7 Алгоритмы шифрования и контроля целостности

7.1 Шифрование в режиме простой замены

7.1.1 Интерфейс

Шифрование в режиме простой замены задается алгоритмами зашифрования `belt-ecb` и расшифрования $\texttt{belt-ecb}^{-1}$.

Входными данными `belt-ecb` являются сообщение $X \in \{0,1\}^*$ и ключ $K \in \{0,1\}^{256}$. Длина X должна быть не меньше 128. Выходными данными является зашифрованное сообщение $Y \in \{0,1\}^{|X|}$.

Входными данными $\texttt{belt-ecb}^{-1}$ являются зашифрованное сообщение $Y \in \{0,1\}^*$ и ключ $K \in \{0,1\}^{256}$. Длина Y должна быть не меньше 128. Выходными данными является расшифрованное сообщение $X \in \{0,1\}^{|Y|}$.

Используются алгоритмы `belt-block` и $\texttt{belt-block}^{-1}$, определенные в 6.1.

7.1.2 Переменные

Пусть $m = |X| \bmod 128$ при зашифровании и $m = |Y| \bmod 128$ при расшифровании. Если $m \neq 0$, то используется переменная $r \in \{0,1\}^{128-m}$.

7.1.3 Алгоритм зашифрования

Зашифрование $\texttt{belt-ecb}(X, K)$ выполняется следующим образом:
1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split}(X, 128)$.
2 Если $|X_n| = 128$, то:
  1) для $i = 1, 2, \ldots, n$ выполнить: $Y_i \leftarrow \texttt{belt-block}(X_i, K)$.
3 Иначе, если $|X_n| < 128$, то:
  1) для $i = 1, 2, \ldots, n-2$ выполнить: $Y_i \leftarrow \texttt{belt-block}(X_i, K)$;
  2) $(Y_n \,\|\, r) \leftarrow \texttt{belt-block}(X_{n-1}, K)$;
  3) $Y_{n-1} \leftarrow \texttt{belt-block}(X_n \,\|\, r, K)$.
4 Возвратить $Y = Y_1 \,\|\, Y_2 \,\|\, \ldots \,\|\, Y_n$.

7.1.4 Алгоритм расшифрования

Расшифрование $\texttt{belt-ecb}^{-1}(Y, K)$ выполняется следующим образом:
1 Определить $(Y_1, Y_2, \ldots, Y_n) = \texttt{Split}(Y, 128)$.
2 Если $|Y_n| = 128$, то:
  1) для $i = 1, 2, \ldots, n$ выполнить: $X_i \leftarrow \texttt{belt-block}^{-1}(Y_i, K)$.
3 Иначе, если $|Y_n| < 128$, то:
  1) для $i = 1, 2, \ldots, n-2$ выполнить: $X_i \leftarrow \texttt{belt-block}^{-1}(Y_i, K)$;
  2) $(X_n \,\|\, r) \leftarrow \texttt{belt-block}^{-1}(Y_{n-1}, K)$;
  3) $X_{n-1} \leftarrow \texttt{belt-block}^{-1}(Y_n \,\|\, r, K)$.
4 Возвратить $X = X_1 \,\|\, X_2 \,\|\, \ldots \,\|\, X_n$.

7.2 Шифрование в режиме сцепления блоков

7.2.1 Интерфейс

Шифрование в режиме сцепления блоков задается алгоритмами зашифрования `belt-cbc` и расшифрования $\texttt{belt-cbc}^{-1}$.

Входными данными $\texttt{belt-cbc}$ являются сообщение $X \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Длина X должна быть не меньше 128. Выходными данными является зашифрованное сообщение $Y \in \{0,1\}^{|X|}$.

Входными данными $\texttt{belt-cbc}^{-1}$ являются зашифрованное сообщение $Y \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Длина Y должна быть не меньше 128. Выходными данными является расшифрованное сообщение $X \in \{0,1\}^{|Y|}$.

Используются алгоритмы $\texttt{belt-block}$ и $\texttt{belt-block}^{-1}$, определенные в 6.1.

7.2.2 Переменные

Пусть $m = |X| \bmod 128$ при зашифровании и $m = |Y| \bmod 128$ при расшифровании. Если $m \neq 0$, то используется переменная $r \in \{0,1\}^{128-m}$.

7.2.3 Алгоритм зашифрования

Зашифрование $\texttt{belt-cbc}(X, K, S)$ выполняется следующим образом:

1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split}(X, 128)$.
2 Обозначить $Y_0 = S$.
3 Если $|X_n| = 128$, то:
  1) для $i = 1, 2, \ldots, n$ выполнить: $Y_i \leftarrow \texttt{belt-block}(X_i \oplus Y_{i-1}, K)$.
4 Иначе, если $|X_n| < 128$, то:
  1) для $i = 1, 2, \ldots, n-2$ выполнить: $Y_i \leftarrow \texttt{belt-block}(X_i \oplus Y_{i-1}, K)$;
  2) $(Y_n \parallel r) \leftarrow \texttt{belt-block}(X_{n-1} \oplus Y_{n-2}, K)$;
  3) $Y_{n-1} \leftarrow \texttt{belt-block}((X_n \oplus Y_n) \parallel r, K)$.
5 Возвратить $Y = Y_1 \parallel Y_2 \parallel \ldots \parallel Y_n$.

7.2.4 Алгоритм расшифрования

Расшифрование $\texttt{belt-cbc}^{-1}(Y, K, S)$ выполняется следующим образом:

1 Определить $(Y_1, Y_2, \ldots, Y_n) = \texttt{Split}(Y, 128)$.
2 Обозначить $Y_0 = S$.
3 Если $|Y_n| = 128$, то:
  1) для $i = 1, 2, \ldots, n$ выполнить: $X_i \leftarrow \texttt{belt-block}^{-1}(Y_i, K) \oplus Y_{i-1}$.
4 Иначе, если $|Y_n| < 128$, то:
  1) для $i = 1, 2, \ldots, n-2$ выполнить: $X_i \leftarrow \texttt{belt-block}^{-1}(Y_i, K) \oplus Y_{i-1}$;
  2) $(X_n \parallel r) \leftarrow \texttt{belt-block}^{-1}(Y_{n-1}, K) \oplus (Y_n \parallel 0^{128-m})$;
  3) $X_{n-1} \leftarrow \texttt{belt-block}^{-1}(Y_n \parallel r, K) \oplus Y_{n-2}$.
5 Возвратить $X = X_1 \parallel X_2 \parallel \ldots \parallel X_n$.

7.3 Шифрование в режиме гаммирования с обратной связью

7.3.1 Интерфейс

Шифрование в режиме гаммирования с обратной связью задается алгоритмами зашифрования $\texttt{belt-cfb}$ и расшифрования $\texttt{belt-cfb}^{-1}$.

Входными данными $\texttt{belt-cfb}$ являются сообщение $X \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является зашифрованное сообщение $Y \in \{0,1\}^{|X|}$.

Входными данными $\texttt{belt-cfb}^{-1}$ являются зашифрованное сообщение $Y \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является расшифрованное сообщение $X \in \{0,1\}^{|Y|}$.

Используется алгоритм $\texttt{belt-block}$, определенный в 6.1.

7.3.2 Алгоритм зашифрования

Зашифрование $\texttt{belt-cfb}(X, K, S)$ выполняется следующим образом:
1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split}(X, 128)$.
2 Обозначить $Y_0 = S$.
3 Для $i = 1, 2, \ldots, n$ выполнить: $Y_i \leftarrow X_i \oplus \texttt{Lo}(\texttt{belt-block}(Y_{i-1}, K), |X_i|)$.
4 Возвратить $Y = Y_1 \| Y_2 \| \ldots \| Y_n$.

7.3.3 Алгоритм расшифрования

Расшифрование $\texttt{belt-cfb}^{-1}(Y, K, S)$ выполняется следующим образом:
1 Определить $(Y_1, Y_2, \ldots, Y_n) = \texttt{Split}(Y, 128)$.
2 Обозначить $Y_0 = S$.
3 Для $i = 1, 2, \ldots, n$ выполнить: $X_i \leftarrow Y_i \oplus \texttt{Lo}(\texttt{belt-block}(Y_{i-1}, K), |Y_i|)$.
4 Возвратить $X = X_1 \| X_2 \| \ldots \| X_n$.

7.4 Шифрование в режиме счетчика

7.4.1 Интерфейс

Шифрование в режиме счетчика задается алгоритмами зашифрования $\texttt{belt-ctr}$ и расшифрования $\texttt{belt-ctr}^{-1}$.

Входными данными $\texttt{belt-ctr}$ являются сообщение $X \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является зашифрованное сообщение $Y \in \{0,1\}^{|X|}$.

Входными данными $\texttt{belt-ctr}^{-1}$ являются зашифрованное сообщение $Y \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является расшифрованное сообщение $X \in \{0,1\}^{|Y|}$.

Используется алгоритм $\texttt{belt-block}$, определенный в 6.1.

7.4.2 Переменные

Используется переменная $s \in \{0,1\}^{128}$.

7.4.3 Алгоритм зашифрования

Зашифрование $\texttt{belt-ctr}(X, K, S)$ выполняется следующим образом:
1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split}(X, 128)$.
2 Установить $s \leftarrow \texttt{belt-block}(S, K)$.
3 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $s \leftarrow s \boxplus \langle 1 \rangle_{128}$;
  2) $Y_i \leftarrow X_i \oplus \texttt{Lo}(\texttt{belt-block}(s, K), |X_i|)$.
4 Возвратить $Y = Y_1 \| Y_2 \| \ldots \| Y_n$.

7.4.4 Алгоритм расшифрования

Расшифрование $\texttt{belt-ctr}^{-1}(Y, K, S)$ выполняется следующим образом:
1 Определить $(Y_1, Y_2, \ldots, Y_n) = \texttt{Split}(Y, 128)$.
2 Установить $s \leftarrow \texttt{belt-block}(S, K)$.
3 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $s \leftarrow s \boxplus \langle 1 \rangle_{128}$;
  2) $X_i \leftarrow Y_i \oplus \texttt{Lo}(\texttt{belt-block}(s, K), |Y_i|)$.
4 Возвратить $X = X_1 \| X_2 \| \ldots \| X_n$.

7.5 Выработка имитовставки

7.5.1 Интерфейс

Выработка имитовставки задается алгоритмом `belt-mac`.

Входными данными `belt-mac` являются сообщение $X \in \{0,1\}^*$ и ключ $K \in \{0,1\}^{256}$. Выходными данными является имитовставка $T \in \{0,1\}^{64}$.

Используется алгоритм `belt-block`, определенный в 6.1.

7.5.2 Вспомогательные преобразования и переменные

Преобразования φ_1 и φ_2. Преобразования $\varphi_1, \varphi_2 \colon \{0,1\}^{128} \to \{0,1\}^{128}$ действуют на слово $u = u_1 \parallel u_2 \parallel u_3 \parallel u_4$, $u_i \in \{0,1\}^{32}$, по правилам:

$$\varphi_1(u) = u_2 \parallel u_3 \parallel u_4 \parallel (u_1 \oplus u_2),$$
$$\varphi_2(u) = (u_1 \oplus u_4) \parallel u_1 \parallel u_2 \parallel u_3.$$

Отображение ψ. Отображение ψ ставит в соответствие двоичному слову u, длина которого меньше 128, слово $\psi(u) = u \parallel 1 \parallel 0^{127-|u|}$ длины 128.

Переменные. Используются переменные $r, s \in \{0,1\}^{128}$.

7.5.3 Алгоритм выработки имитовставки

Выработка имитовставки $\texttt{belt-mac}(X, K)$ выполняется следующим образом:

1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split1}(X, 128)$.
2 Установить $s \leftarrow 0^{128}$, $r \leftarrow \texttt{belt-block}(s, K)$.
3 Для $i = 1, 2, \ldots, n-1$ выполнить: $s \leftarrow \texttt{belt-block}(s \oplus X_i, K)$.
4 Если $|X_n| = 128$, то $s \leftarrow s \oplus X_n \oplus \varphi_1(r)$.
5 Иначе, если $|X_n| < 128$, то $s \leftarrow s \oplus \psi(X_n) \oplus \varphi_2(r)$.
6 Установить $T \leftarrow \texttt{Lo}(\texttt{belt-block}(s, K), 64)$.
7 Возвратить T.

Примечание — Если $X = \perp$, то $n = 1$ и $X_1 = \perp$.

7.6 Аутентифицированное шифрование данных

7.6.1 Интерфейс

Аутентифицированное шифрование данных задается алгоритмами установки и снятия защиты. Первая схема аутентифицированного шифрования представлена алгоритмами `belt-dwp` и $\texttt{belt-dwp}^{-1}$, вторая — алгоритмами `belt-che` и $\texttt{belt-che}^{-1}$.

Входными данными алгоритма установки защиты (`belt-dwp` или `belt-che`) являются сообщение $X \in \{0,1\}^*$, ассоциированные данные $I \in \{0,1\}^*$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Длины X и I должны быть меньше 2^{64}. Выходными данными являются зашифрованное сообщение $Y \in \{0,1\}^{|X|}$ и имитовставка $T \in \{0,1\}^{64}$.

Входными данными алгоритма снятия защиты ($\texttt{belt-dwp}^{-1}$ или $\texttt{belt-che}^{-1}$) являются зашифрованное сообщение $Y \in \{0,1\}^*$, ассоциированные данные $I \in \{0,1\}^*$, имитовставка $T \in \{0,1\}^{64}$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Длины Y и I должны быть меньше 2^{64}. Выходными данными является либо признак ошибки $\perp$, либо расшифрованное сообщение $X \in \{0,1\}^{|Y|}$. Возврат $\perp$ означает нарушение целостности входных данных.

Используется алгоритм `belt-block`, определенный в 6.1.

7.6.2 Переменные и константы

Используются переменные $s, t \in \{0,1\}^{128}$.

В алгоритмах схемы 1 дополнительно используется переменная $r \in \{0,1\}^{128}$.

В алгоритмах схемы 2 используется слово $C = \mathtt{02}_{16} \parallel 0^{120}$. Слово представляет многочлен $C(x) = x$ (см. 4.2.3).

7.6.3 Алгоритм установки защиты (схема 1)

Установка защиты $\mathtt{belt\text{-}dwp}(X, I, K, S)$ выполняется следующим образом:

1 Определить:
  1) $(X_1, X_2, \ldots, X_n) = \mathtt{Split}(X, 128)$;
  2) $(I_1, I_2, \ldots, I_m) = \mathtt{Split}(I, 128)$.
2 Установить:
  1) $s \leftarrow \mathtt{belt\text{-}block}(S, K)$;
  2) $r \leftarrow \mathtt{belt\text{-}block}(s, K)$;
  3) $t \leftarrow \mathtt{B194BAC80A08F53B366D008E584A5DE4}_{16}$ (см. строку 1 таблицы 2).
3 Для $i = 1, 2, \ldots, m$ выполнить:
  1) $t \leftarrow t \oplus (I_i \parallel 0^{128-|I_i|})$;
  2) $t \leftarrow t * r$.
4 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $s \leftarrow s \boxplus \langle 1 \rangle_{128}$;
  2) $Y_i \leftarrow X_i \oplus \mathtt{Lo}(\mathtt{belt\text{-}block}(s, K), |X_i|)$;
  3) $t \leftarrow t \oplus (Y_i \parallel 0^{128-|Y_i|})$;
  4) $t \leftarrow t * r$.
5 Установить $t \leftarrow t \oplus (\langle |I| \rangle_{64} \parallel \langle |X| \rangle_{64})$.
6 Установить $t \leftarrow \mathtt{belt\text{-}block}(t * r, K)$.
7 Установить $T \leftarrow \mathtt{Lo}(t, 64)$.
8 Возвратить (Y, T), где $Y = Y_1 \parallel Y_2 \parallel \ldots \parallel Y_n$.

7.6.4 Алгоритм снятия защиты (схема 1)

Снятие защиты $\mathtt{belt\text{-}dwp}^{-1}(Y, I, T, K, S)$ выполняется следующим образом:

1 Определить:
  1) $(Y_1, Y_2, \ldots, Y_n) = \mathtt{Split}(Y, 128)$;
  2) $(I_1, I_2, \ldots, I_m) = \mathtt{Split}(I, 128)$.
2 Установить:
  1) $s \leftarrow \mathtt{belt\text{-}block}(S, K)$;
  2) $r \leftarrow \mathtt{belt\text{-}block}(s, K)$;
  3) $t \leftarrow \mathtt{B194BAC80A08F53B366D008E584A5DE4}_{16}$.
3 Для $i = 1, 2, \ldots, m$ выполнить:
  1) $t \leftarrow t \oplus (I_i \parallel 0^{128-|I_i|})$;
  2) $t \leftarrow t * r$.
4 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $t \leftarrow t \oplus (Y_i \parallel 0^{128-|Y_i|})$;
  2) $t \leftarrow t * r$.
5 Установить $t \leftarrow t \oplus (\langle |I| \rangle_{64} \parallel \langle |Y| \rangle_{64})$.
6 Установить $t \leftarrow \mathtt{belt\text{-}block}(t * r, K)$.
7 Если $T \neq \mathtt{Lo}(t, 64)$, то возвратить $\perp$.
8 Для $i = 1, 2, \ldots, n$ выполнить:
  1) $s \leftarrow s \boxplus \langle 1 \rangle_{128}$;

2) $X_i \leftarrow Y_i \oplus \texttt{Lo}(\texttt{belt-block}(s, K), |Y_i|)$.

9 Возвратить $X = X_1 \| X_2 \| \ldots \| X_n$.

7.6.5 Алгоритм установки защиты (схема 2)

Установка защиты $\texttt{belt-che}(X, I, K, S)$ выполняется следующим образом:

1 Определить:
   1) $(X_1, X_2, \ldots, X_n) = \texttt{Split}(X, 128)$;
   2) $(I_1, I_2, \ldots, I_m) = \texttt{Split}(I, 128)$.
2 Установить:
   1) $s \leftarrow \texttt{belt-block}(S, K)$;
   2) $r \leftarrow s$;
   3) $t \leftarrow \texttt{B194BAC80A08F53B366D008E584A5DE4}_{16}$.
3 Для $i = 1, 2, \ldots, m$ выполнить:
   1) $t \leftarrow t \oplus (I_i \| 0^{128-|I_i|})$;
   2) $t \leftarrow t * r$.
4 Для $i = 1, 2, \ldots, n$ выполнить:
   1) $s \leftarrow (s * C) \oplus \langle 1 \rangle_{128}$;
   2) $Y_i \leftarrow X_i \oplus \texttt{Lo}(\texttt{belt-block}(s, K), |X_i|)$;
   3) $t \leftarrow t \oplus (Y_i \| 0^{128-|Y_i|})$;
   4) $t \leftarrow t * r$.
5 Установить $t \leftarrow t \oplus (\langle |I| \rangle_{64} \| \langle |X| \rangle_{64})$.
6 Установить $t \leftarrow \texttt{belt-block}(t * r, K)$.
7 Установить $T \leftarrow \texttt{Lo}(t, 64)$.
8 Возвратить (Y, T), где $Y = Y_1 \| Y_2 \| \ldots \| Y_n$.

7.6.6 Алгоритм снятия защиты (схема 2)

Снятие защиты $\texttt{belt-che}^{-1}(Y, I, T, K, S)$ выполняется следующим образом:

1 Определить:
   1) $(Y_1, Y_2, \ldots, Y_n) = \texttt{Split}(Y, 128)$;
   2) $(I_1, I_2, \ldots, I_m) = \texttt{Split}(I, 128)$.
2 Установить:
   1) $s \leftarrow \texttt{belt-block}(S, K)$;
   2) $r \leftarrow s$;
   3) $t \leftarrow \texttt{B194BAC80A08F53B366D008E584A5DE4}_{16}$.
3 Для $i = 1, 2, \ldots, m$ выполнить:
   1) $t \leftarrow t \oplus (I_i \| 0^{128-|I_i|})$;
   2) $t \leftarrow t * r$.
4 Для $i = 1, 2, \ldots, n$ выполнить:
   1) $t \leftarrow t \oplus (Y_i \| 0^{128-|Y_i|})$;
   2) $t \leftarrow t * r$.
5 Установить $t \leftarrow t \oplus (\langle |I| \rangle_{64} \| \langle |Y| \rangle_{64})$.
6 Установить $t \leftarrow \texttt{belt-block}(t * r, K)$.
7 Если $T \neq \texttt{Lo}(t, 64)$, то возвратить $\perp$.
8 Для $i = 1, 2, \ldots, n$ выполнить:
   1) $s \leftarrow (s * C) \oplus \langle 1 \rangle_{128}$;
   2) $X_i \leftarrow Y_i \oplus \texttt{Lo}(\texttt{belt-block}(s, K), |Y_i|)$.
9 Возвратить $X = X_1 \| X_2 \| \ldots \| X_n$.

Примечание 1 — Если длина имитовставки, вычисляемой при установке защиты, сокращается (см. 5.4), то на вход алгоритма снятия защиты должна подаваться сокращенная ими-

товставка, а проверка $T \neq \texttt{Lo}(t, 64)$ на шаге 7 этого алгоритма должна быть изменена на проверку $T \neq \texttt{Lo}(t, |T|)$.

Примечание 2 — В алгоритме снятия защиты расшифрование на шаге 8 может выполняться одновременно с вычислением имитовставки на шаге 4. Однако расшифрование может оказаться бесполезным, если на шаге 7 имитовставка окажется некорректной.

Примечание 3 — При вычислении $(Y, T) = \texttt{belt-dwp}(X, I, K, S)$ разрешается выдавать один или несколько промежуточных результатов $(Y', T') = \texttt{belt-dwp}(X', I, K, S)$. Здесь X' — префикс X, а Y' будет префиксом Y. Выдача промежуточных результатов позволяет дискретизировать процесс защиты, что оказывается полезным при передаче сообщений X большой длины. При снятии защиты промежуточная пара (Y', T') обрабатывается с помощью алгоритма $\texttt{belt-dwp}^{-1}$ обычным образом. Если $\texttt{belt-dwp}^{-1}(Y', I, T', K, S) = \perp$, то снятие защиты должно быть прервано. Сказанное относится также к алгоритмам `belt-che` и $\texttt{belt-che}^{-1}$.

7.7 Аутентифицированное шифрование ключа

7.7.1 Интерфейс

Аутентифицированное шифрование ключа задается алгоритмами установки защиты `belt-kwp` и снятия защиты $\texttt{belt-kwp}^{-1}$.

Входными данными `belt-kwp` являются защищаемый ключ $X \in \{0,1\}^{8*}$, его заголовок $I \in \{0,1\}^{128}$ и ключ защиты $K \in \{0,1\}^{256}$. Длина X должна быть не меньше 128. Выходными данными является защищенный ключ $Y \in \{0,1\}^{|X|+128}$.

Входными данными $\texttt{belt-kwp}^{-1}$ являются защищенный ключ $Y \in \{0,1\}^*$, его заголовок $I \in \{0,1\}^{128}$ и ключ защиты $K \in \{0,1\}^{256}$. Выходными данными является либо признак ошибки $\perp$, либо исходный ключ $X \in \{0,1\}^{|Y|-128}$. Возврат $\perp$ означает нарушение целостности входных данных.

Используются алгоритмы `belt-wblock`, $\texttt{belt-wblock}^{-1}$, определенные в 6.2.

7.7.2 Переменные

При снятии защиты используется переменная $r \in \{0,1\}^{128}$.

7.7.3 Алгоритм установки защиты

Установка защиты $\texttt{belt-kwp}(X, I, K)$ выполняется следующим образом:

1 $Y \leftarrow \texttt{belt-wblock}(X \parallel I, K)$.

2 Возвратить Y.

7.7.4 Алгоритм снятия защиты

Снятие защиты $\texttt{belt-kwp}^{-1}(Y, I, K)$ выполняется следующим образом:

1 Если длина Y не кратна 8 или $|Y| < 256$, то возвратить $\perp$.

2 $(X \parallel r) \leftarrow \texttt{belt-wblock}^{-1}(Y, K)$.

3 Если $r \neq I$, то возвратить $\perp$.

4 Возвратить X.

7.8 Хэширование

7.8.1 Интерфейс

Хэширование задается алгоритмом `belt-hash`.

Входными данными `belt-hash` является сообщение $X \in \{0,1\}^*$. Выходными данными является хэш-значение $Y \in \{0,1\}^{256}$.

Используется алгоритм `belt-compress`, определенный в 6.3.

7.8.2 Переменные

Используются переменные $r, s, t \in \{0,1\}^{128}$ и $h \in \{0,1\}^{256}$.

7.8.3 Алгоритм хэширования

Хэширование $\texttt{belt-hash}(X)$ выполняется следующим образом:

1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split1}(X, 256)$.

2 Установить

1) $r \leftarrow \langle |X| \rangle_{128}$;
2) $s \leftarrow 0^{128}$;
3) $h \leftarrow \texttt{B194BAC80A08F53B366D008E584A5DE48504FA9D1BB6C7AC252E72C202FDCE0D}_{16}$ (см. первые две строки таблицы 2);
4) $X_n \leftarrow X_n \,\|\, 0^{256-|X_n|}$.

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

1) $(t, h) \leftarrow \texttt{belt-compress}(X_i \,\|\, h)$;
2) $s \leftarrow s \oplus t$.

4 Установить $(\perp, Y) \leftarrow \texttt{belt-compress}(r \,\|\, s \,\|\, h)$.

5 Возвратить Y.

Примечание — Если $X = \perp$, то $n = 1$ и $X_1 = \perp$.

7.9 Дисковое шифрование

7.9.1 Интерфейс

Блоковое дисковое шифрование задается алгоритмами зашифрования $\texttt{belt-bde}$ и расшифрования $\texttt{belt-bde}^{-1}$. Секторное дисковое шифрование задается алгоритмами зашифрования $\texttt{belt-sde}$ и расшифрования $\texttt{belt-sde}^{-1}$.

Входными данными $\texttt{belt-bde}$, $\texttt{belt-sde}$ являются сообщение $X \in \{0,1\}^{128*}$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. При блоковом шифровании длина X должна быть не меньше 128, при секторном — не меньше 256. Выходными данными является зашифрованное сообщение $Y \in \{0,1\}^{|X|}$.

Входными данными $\texttt{belt-bde}^{-1}$, $\texttt{belt-sde}^{-1}$ являются зашифрованное сообщение $Y \in \{0,1\}^{128*}$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. При блоковом шифровании длина Y должна быть не меньше 128, при секторном — не меньше 256. Выходными данными является расшифрованное сообщение $X \in \{0,1\}^{|Y|}$.

Слова X и Y представляют собой содержимое дискового сектора до и после зашифрования. Синхропосылка S строится по номеру сектора. Например, $S = \langle D \rangle_{128}$ для сектора номер D.

Используются алгоритмы $\texttt{belt-block}$ и $\texttt{belt-block}^{-1}$, определенные в 6.1. При секторном шифровании дополнительно используются алгоритмы $\texttt{belt-wblock}$ и $\texttt{belt-wblock}^{-1}$, определенные в 6.2.

7.9.2 Переменные и константы

Используется переменная $s \in \{0,1\}^{128}$.

В алгоритмах $\texttt{belt-bde}$, $\texttt{belt-bde}^{-1}$ используется слово $C = \texttt{02}_{16} \,\|\, 0^{120}$. Слово представляет многочлен $C(x) = x$ (см. 4.2.3).

7.9.3 Алгоритм блокового зашифрования

Зашифрование $\texttt{belt-bde}(X, K, S)$ выполняется следующим образом:

1 Определить $(X_1, X_2, \ldots, X_n) = \texttt{Split}(X, 128)$.

2 Установить $s \leftarrow \texttt{belt-block}(S, K)$.

3 Для $i = 1, 2, \ldots, n$ выполнить:
   1) $s \leftarrow s * C$;
   2) $Y_i \leftarrow \texttt{belt-block}(X_i \oplus s, K) \oplus s$.
4 Возвратить $Y = Y_1 \| Y_2 \| \ldots \| Y_n$.

7.9.4 Алгоритм блокового расшифрования

Расшифрование $\texttt{belt-bde}^{-1}(Y, K, S)$ выполняется следующим образом:
1 Определить $(Y_1, Y_2, \ldots, Y_n) = \texttt{Split}(Y, 128)$.
2 Установить $s \leftarrow \texttt{belt-block}(S, K)$.
3 Для $i = 1, 2, \ldots, n$ выполнить:
   1) $s \leftarrow s * C$;
   2) $X_i \leftarrow \texttt{belt-block}^{-1}(Y_i \oplus s, K) \oplus s$.
4 Возвратить $X = X_1 \| X_2 \| \ldots \| X_n$.

7.9.5 Алгоритм секторного зашифрования

Зашифрование $\texttt{belt-sde}(X, K, S)$ выполняется следующим образом:
1 Установить $Y \leftarrow X$.
2 Записать $Y = Y_1 \| Y_2$, где $|Y_1| = 128$.
3 Установить $s \leftarrow \texttt{belt-block}(S, K)$.
4 Установить $Y_1 \leftarrow Y_1 \oplus s$.
5 Установить $Y \leftarrow \texttt{belt-wblock}(Y, K)$.
6 Установить $Y_1 \leftarrow Y_1 \oplus s$.
7 Возвратить Y.

7.9.6 Алгоритм секторного расшифрования

Расшифрование $\texttt{belt-sde}^{-1}(Y, K, S)$ выполняется следующим образом:
1 Установить $X \leftarrow Y$.
2 Записать $X = X_1 \| X_2$, где $|X_1| = 128$.
3 Установить $s \leftarrow \texttt{belt-block}(S, K)$.
4 Установить $X_1 \leftarrow X_1 \oplus s$.
5 Установить $X \leftarrow \texttt{belt-wblock}^{-1}(X, K)$.
6 Установить $X_1 \leftarrow X_1 \oplus s$.
7 Возвратить X.

7.10 Шифрование с сохранением формата

7.10.1 Интерфейс

Шифрование с сохранением формата задается алгоритмами зашифрования `belt-fmt` и расшифрования $\texttt{belt-fmt}^{-1}$.

Входными данными `belt-fmt` являются слово $X \in \mathbb{Z}_m^n$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является зашифрованное слово $Y \in \mathbb{Z}_m^n$.

Входными данными $\texttt{belt-fmt}^{-1}$ являются зашифрованное слово $Y \in \mathbb{Z}_m^n$, ключ $K \in \{0,1\}^{256}$ и синхропосылка $S \in \{0,1\}^{128}$. Выходными данными является расшифрованное слово $X \in \mathbb{Z}_m^n$.

Размер алфавита $\mathbb{Z}_m$ и длина n входного и выходного слов должны принадлежать интервалу $\{2, 3, \ldots, 65536\}$.

Используются алгоритмы `belt-block` и `belt-wblock`, определенные в 6.1 и 6.2.

7.10.2 Подготовка входных данных

Длина n записывается в виде суммы $n_1 + n_2$, где $n_1 = \lceil n/2 \rceil$, $n_2 = \lfloor n/2 \rfloor$. По слагаемому n_i, $i = 1, 2$, определяется минимальное натуральное b_i такое, что:

1) b_i кратно 64;
2) $2^{b_i} \geqslant m^{n_i}$.

Синхропосылка S разбивается на фрагменты $(S_1, S_2, S_3, S_4) = \texttt{Split}(S, 32)$. Дополнительно строятся слова S_0 и S_5, описывающие формат: $S_0 = S_5 = \langle m \rangle_{16} \parallel \langle n \rangle_{16}$.

7.10.3 Вспомогательные алгоритмы, переменные и обозначения

Алгоритм `belt-32block`. Используется алгоритм `belt-32block`, который принимает на вход слово $t \in \{0,1\}^{192}$ и ключ $K \in \{0,1\}^{256}$ и возвращает преобразованное слово t. В слове t выделяются фрагменты $(t_1, t_2, t_3) = \texttt{Split}(t, 64)$.

Шаги алгоритма:

1 Для $i = 1, 2, 3$:
   1) $(t_2 \parallel t_3) \leftarrow \texttt{belt-block}(t_2 \parallel t_3, K) \oplus \langle i \rangle_{64}$;
   2) $t \leftarrow t_2 \parallel t_3 \parallel (t_1 \oplus t_2)$.
2 Возвратить t.

Переменная r. Используется переменная $r \in \mathbb{Z}_m^n$. Переменная записывается в виде $r = r_1 \parallel r_2$, $|r_i| = n_i$.

Переменные t_1, t_2. Используются переменные $t_1 \in \{0,1\}^{64+b_1}$ и $t_2 \in \{0,1\}^{64+b_2}$.

Обозначение `str2bin`. Для $j \in \{1, 2\}$ и $u \in \mathbb{Z}_m^*$ через $\texttt{str2bin}(u, b_j)$ обозначается двоичное слово $\langle \lfloor u \rceil_m \rangle_{b_j}$.

Обозначение `bin2str`. Для $j \in \{1, 2\}$ и $t \in \{0,1\}^{64*}$ через $\texttt{bin2str}(t, n_j)$ обозначается слово $\langle \lfloor t \rceil \rangle_{m, n_j}$.

Обозначение `roundf`. Пусть $t \in \{0,1\}^{64*}$, причем $|t| \geqslant 128$. Обозначение $\texttt{roundf}(t, K)$ раскрывается как:

1) $\texttt{belt-block}(t, K)$ при $|t| = 128$;
2) $\texttt{belt-32block}(t, K)$ при $|t| = 192$ и
3) $\texttt{belt-wblock}(t, K)$ при $|t| \geqslant 256$.

7.10.4 Константы

В алгоритмах используются константы $C_0, C_1, \ldots, C_5 \in \{0,1\}^{32}$. Константы определяются по первым двум строкам таблицы 2 и имеют следующий вид:

$$C_0 = \texttt{B194BAC8}_{16}, \quad C_1 = \texttt{0A08F53B}_{16}, \quad C_2 = \texttt{366D008E}_{16}, \quad C_3 = \texttt{584A5DE4}_{16},$$
$$C_4 = \texttt{8504FA9D}_{16}, \quad C_5 = \texttt{1BB6C7AC}_{16}.$$

7.10.5 Алгоритм зашифрования

Зашифрование $\texttt{belt-fmt}(X, K, S)$ выполняется следующим образом:

1 Установить $r \leftarrow X$.
2 Для $i = 1, 2, 3$:
   1) $t_2 \leftarrow \texttt{roundf}(\texttt{str2bin}(r_2, b_2) \parallel C_{2i-2} \parallel S_{2i-2}, K)$;
   2) $r_1 \leftarrow r_1 \oplus \texttt{bin2str}(t_2, n_1)$;
   3) $t_1 \leftarrow \texttt{roundf}(\texttt{str2bin}(r_1, b_1) \parallel C_{2i-1} \parallel S_{2i-1}, K)$;
   4) $r_2 \leftarrow r_2 \oplus \texttt{bin2str}(t_1, n_2)$.
3 Установить $Y \leftarrow r_1 \parallel r_2$.
4 Возвратить Y.

7.10.6 Алгоритм расшифрования

Расшифрование $\texttt{belt-fmt}^{-1}(Y, K, S)$ выполняется следующим образом:

1 Установить $r \leftarrow Y$.
2 Для $i = 3, 2, 1$:
   1) $t_1 \leftarrow \texttt{roundf}(\texttt{str2bin}(r_1, b_1) \parallel C_{2i-1} \parallel S_{2i-1}, K)$;
   2) $r_2 \leftarrow r_2 \ominus \texttt{bin2str}(t_1, n_2)$;
   3) $t_2 \leftarrow \texttt{roundf}(\texttt{str2bin}(r_2, b_2) \parallel C_{2i-2} \parallel S_{2i-2}, K)$;
   4) $r_1 \leftarrow r_1 \ominus \texttt{bin2str}(t_2, n_1)$.
3 Установить $X \leftarrow r_1 \parallel r_2$.
4 Возвратить X.

8 Служебные алгоритмы

8.1 Расширение ключа

8.1.1 Интерфейс

Расширение ключа задается алгоритмом `belt-keyexpand`.

Входными данными `belt-keyexpand` является расширяемый ключ $K_1 \parallel K_2 \parallel \ldots \parallel K_n$, где $K_i \in \{0,1\}^{32}$, $n \in \{4, 6, 8\}$. Выходными данными является расширенный ключ $K \in \{0,1\}^{256}$.

8.1.2 Алгоритм расширения ключа

Расширение ключа $\texttt{belt-keyexpand}(K_1 \parallel K_2 \parallel \ldots \parallel K_n)$ выполняется следующим образом:

1 Если $n = 4$, то выполнить:
   1) $K_5 \leftarrow K_1$;
   2) $K_6 \leftarrow K_2$;
   3) $K_7 \leftarrow K_3$;
   4) $K_8 \leftarrow K_4$.
2 Если $n = 6$, то выполнить:
   1) $K_7 \leftarrow K_1 \oplus K_2 \oplus K_3$;
   2) $K_8 \leftarrow K_4 \oplus K_5 \oplus K_6$.
3 Установить $K \leftarrow K_1 \parallel K_2 \parallel \ldots \parallel K_8$.
4 Возвратить K.

8.2 Преобразование ключа

8.2.1 Интерфейс

Преобразование ключа задается алгоритмом `belt-keyrep`.

Входными данными `belt-keyrep` являются преобразуемый ключ $X \in \{0,1\}^n$, его уровень $D \in \{0,1\}^{96}$, заголовок $I \in \{0,1\}^{128}$ нового ключа и его длина m. Должны соблюдаться ограничения: $m, n \in \{128, 192, 256\}$, $m \leqslant n$. Выходными данными является преобразованный ключ $Y \in \{0,1\}^m$.

При преобразовании ключа Y его уровень следует полагать равным $D \boxplus \langle 1 \rangle_{96}$.

Используются алгоритмы `belt-compress` и `belt-keyexpand`, определенные в 6.3 и 8.1.

8.2.2 Переменные

Используются переменные $r \in \{0,1\}^{32}$ и $s \in \{0,1\}^{256}$.

8.2.3 Алгоритм преобразования ключа

Преобразование ключа $\texttt{belt-keyrep}(X, D, I, m)$ выполняется следующим образом:

1 Присвоить переменной r значение:
   1) $\texttt{B194BAC8}_{16}$, если $n = m = 128$;
   2) $\texttt{5BE3D612}_{16}$, если $n = 192$ и $m = 128$;
   3) $\texttt{5CB0C0FF}_{16}$, если $n = m = 192$;
   4) $\texttt{E12BDC1A}_{16}$, если $n = 256$ и $m = 128$;
   5) $\texttt{C1AB7638}_{16}$, если $n = 256$ и $m = 192$;
   6) $\texttt{F33C657B}_{16}$, если $n = m = 256$.
2 Установить $s \leftarrow \texttt{belt-keyexpand}(X)$.
3 Установить $(\perp, s) \leftarrow \texttt{belt-compress}(r \parallel D \parallel I \parallel s)$.
4 Установить $Y \leftarrow \texttt{Lo}(s, m)$.
5 Возвратить Y.

Приложение А
(справочное)
Проверочные примеры

А.1 Шифрование блока

В таблице А.1 представлен пример зашифрования блока. Значения переменных a, b, c, d после выполнения тактов зашифрования указаны в таблице А.2. Значения переменных a, b, c, d после выполнения шагов первого такта зашифрования представлены в таблице А.3. Дополнительная переменная e после выполнения шага 4) принимает значение $\texttt{20072EC1}_{16}$.

Таблица А.1 — Зашифрование блока

X	$\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4}_{16}$
K	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}$
Y	$\texttt{69CCA1C9 3557C9E3 D66BC3E0 FA88FA6E}_{16}$

Таблица А.2 — Такты зашифрования

Номер такта i	Переменные			
	a	b	c	d
1	$\texttt{FB56C62C}_{16}$	$\texttt{CA8EEEB7}_{16}$	$\texttt{09BAD702}_{16}$	$\texttt{CC4E441D}_{16}$
2	$\texttt{7280A094}_{16}$	$\texttt{47BB9CD6}_{16}$	$\texttt{5BD130B1}_{16}$	$\texttt{ADA525A4}_{16}$
3	$\texttt{00AB0E4D}_{16}$	$\texttt{4B4A6113}_{16}$	$\texttt{73D9CD18}_{16}$	$\texttt{57E54345}_{16}$
4	$\texttt{A50D12EF}_{16}$	$\texttt{8CD05085}_{16}$	$\texttt{99A672B7}_{16}$	$\texttt{D9A0C0E4}_{16}$
5	$\texttt{21C32063}_{16}$	$\texttt{44712C59}_{16}$	$\texttt{EC21160A}_{16}$	$\texttt{DE08AAB9}_{16}$
6	$\texttt{B5279D32}_{16}$	$\texttt{D4579966}_{16}$	$\texttt{251E3B2D}_{16}$	$\texttt{F8EF6A0F}_{16}$
7	$\texttt{26349022}_{16}$	$\texttt{08C5172E}_{16}$	$\texttt{705A63C6}_{16}$	$\texttt{5CA6AD61}_{16}$
8	$\texttt{D66BC3E0}_{16}$	$\texttt{69CCA1C9}_{16}$	$\texttt{FA88FA6E}_{16}$	$\texttt{3557C9E3}_{16}$

Таблица А.3 — Первый такт зашифрования

Шаг вычислений	Переменные			
	a	b	c	d
1) $b \leftarrow b \oplus G_5(a \boxplus k[1])$	$\texttt{B194BAC8}_{16}$	$\texttt{66DC9868}_{16}$	$\texttt{366D008E}_{16}$	$\texttt{584A5DE4}_{16}$
2) $c \leftarrow c \oplus G_{21}(d \boxplus k[2])$	$\texttt{B194BAC8}_{16}$	$\texttt{66DC9868}_{16}$	$\texttt{F95E6998}_{16}$	$\texttt{584A5DE4}_{16}$
3) $a \leftarrow a \boxminus G_{13}(b \boxplus k[3])$	$\texttt{09BAD702}_{16}$	$\texttt{66DC9868}_{16}$	$\texttt{F95E6998}_{16}$	$\texttt{584A5DE4}_{16}$
4) $e \leftarrow G_{21}(b \boxplus c \boxplus k[4]) \oplus \langle 1 \rangle_{32}$	$\texttt{09BAD702}_{16}$	$\texttt{66DC9868}_{16}$	$\texttt{F95E6998}_{16}$	$\texttt{584A5DE4}_{16}$
5) $b \leftarrow b \boxplus e$	$\texttt{09BAD702}_{16}$	$\texttt{86E3C629}_{16}$	$\texttt{F95E6998}_{16}$	$\texttt{584A5DE4}_{16}$
6) $c \leftarrow c \boxminus e$	$\texttt{09BAD702}_{16}$	$\texttt{86E3C629}_{16}$	$\texttt{D9573BD7}_{16}$	$\texttt{584A5DE4}_{16}$
7) $d \leftarrow d \boxplus G_{13}(c \boxplus k[5])$	$\texttt{09BAD702}_{16}$	$\texttt{86E3C629}_{16}$	$\texttt{D9573BD7}_{16}$	$\texttt{CA8EEEB7}_{16}$
8) $b \leftarrow b \oplus G_{21}(a \boxplus k[6])$	$\texttt{09BAD702}_{16}$	$\texttt{FB56C62C}_{16}$	$\texttt{D9573BD7}_{16}$	$\texttt{CA8EEEB7}_{16}$
9) $c \leftarrow c \oplus G_5(d \boxplus k[7])$	$\texttt{09BAD702}_{16}$	$\texttt{FB56C62C}_{16}$	$\texttt{CC4E441D}_{16}$	$\texttt{CA8EEEB7}_{16}$
10) $a \leftrightarrow b$	$\texttt{FB56C62C}_{16}$	$\texttt{09BAD702}_{16}$	$\texttt{CC4E441D}_{16}$	$\texttt{CA8EEEB7}_{16}$
11) $c \leftrightarrow d$	$\texttt{FB56C62C}_{16}$	$\texttt{09BAD702}_{16}$	$\texttt{CA8EEEB7}_{16}$	$\texttt{CC4E441D}_{16}$
12) $b \leftrightarrow c$	$\texttt{FB56C62C}_{16}$	$\texttt{CA8EEEB7}_{16}$	$\texttt{09BAD702}_{16}$	$\texttt{CC4E441D}_{16}$

В таблице А.4 представлен пример расшифрования блока. Значения переменных a, b, c, d после выполнения тактов расшифрования указаны в таблице А.5.

Таблица А.4 — Расшифрование блока

Y	$\texttt{E12BDC1A E28257EC 703FCCF0 95EE8DF1}_{16}$
K	$\texttt{92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511}_{16}$
X	$\texttt{0DC53006 00CAB840 B38448E5 E993F421}_{16}$

Таблица А.5 — Такты расшифрования

Номер такта i	Переменные			
	a	b	c	d
8	$\texttt{A174D6FC}_{16}$	$\texttt{377EB086}_{16}$	$\texttt{BA7C2D07}_{16}$	$\texttt{0DAA044B}_{16}$
7	$\texttt{B01E75B3}_{16}$	$\texttt{0F53A46F}_{16}$	$\texttt{8893A01F}_{16}$	$\texttt{A4E35989}_{16}$
6	$\texttt{B5B85383}_{16}$	$\texttt{33D8BC0E}_{16}$	$\texttt{9A46CD5F}_{16}$	$\texttt{F8D778D4}_{16}$
5	$\texttt{07234634}_{16}$	$\texttt{723B48FC}_{16}$	$\texttt{04690666}_{16}$	$\texttt{ADB565F3}_{16}$
4	$\texttt{3141A829}_{16}$	$\texttt{2AD3FB40}_{16}$	$\texttt{D30032B1}_{16}$	$\texttt{4D336185}_{16}$
3	$\texttt{ADA2EC35}_{16}$	$\texttt{DADBC720}_{16}$	$\texttt{3421AC22}_{16}$	$\texttt{22EC7943}_{16}$
2	$\texttt{9DAC9289}_{16}$	$\texttt{89A2E5ED}_{16}$	$\texttt{9253A0F0}_{16}$	$\texttt{3B871FA3}_{16}$
1	$\texttt{00CAB840}_{16}$	$\texttt{E993F421}_{16}$	$\texttt{0DC53006}_{16}$	$\texttt{B38448E5}_{16}$

А.2 Шифрование широкого блока

В таблицах А.6, А.7 представлены примеры шифрования широкого блока.

Таблица А.6 — Зашифрование широкого блока

K	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}$
X	$\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}$ $\texttt{5BE3D612 17B96181 FE6786AD 716B890B}_{16}$
Y	$\texttt{49A38EE1 08D6C742 E52B774F 00A6EF98 B106CBD1 3EA4FB06 80323051 BC04DF76}$ $\texttt{E487B055 C69BCF54 1176169F 1DC9F6C8}_{16}$
X	$\texttt{B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D}$ $\texttt{5BE3D612 17B96181 FE6786AD 716B89}_{16}$
Y	$\texttt{F08EF22D CAA06C81 FB127219 74221CA7 AB82C628 56FCF2F9 FCA006E0 19A28F16}$ $\texttt{E5821A51 F5735946 25DBAB8F 6A5C94}_{16}$

Таблица А.7 — Расшифрование широкого блока

K	$\texttt{92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511}_{16}$
Y	$\texttt{E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F}$ $\texttt{F33C657B 637C306A DD4EA779 9EB23D31}_{16}$
X	$\texttt{92632EE0 C21AD9E0 9A39343E 5C07DAA4 889B03F2 E6847EB1 52EC99F7 A4D9F154}$ $\texttt{B5EF68D8 E4A39E56 7153DE13 D72254EE}_{16}$
Y	$\texttt{E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F}$ $\texttt{F33C657B}_{16}$
X	$\texttt{DF3F8822 30BAAFFC 92F05660 32117231 0E3CB218 2681EF43 102E6717 5E177BD7}$ $\texttt{5E93E4E8}_{16}$

А.3 Сжатие

В таблице А.8 представлен пример сжатия с помощью алгоритма `belt-compress`.

Таблица А.8 — Сжатие

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B 5CB0C0FF 33C356B8 35C405AE D8E07F99$_{16}$
S	46FE7425 C9B181EB 41DFEE3E 72163D5A$_{16}$
Y	ED2F5481 D593F40D 87FCE37D 6BC1A2E1 B7D1A2CC 975C82D3 C0497488 C90D99D8$_{16}$

А.4 Шифрование в режиме простой замены

В таблицах А.9, А.10 представлены примеры шифрования в режиме простой замены.

Таблица А.9 — Зашифрование в режиме простой замены

K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
Y	69CCA1C9 3557C9E3 D66BC3E0 FA88FA6E 5F23102E F1097107 75017F73 806DA9DC 46FB2ED2 CE771F26 DCB5E5D1 569F9AB0$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B89$_{16}$
Y	69CCA1C9 3557C9E3 D66BC3E0 FA88FA6E 36F00CFE D6D1CA14 98C12798 F4BEB207 5F23102E F1097107 75017F73 806DA9$_{16}$

Таблица А.10 — Расшифрование в режиме простой замены

K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779 9EB23D31$_{16}$
X	0DC53006 00CAB840 B38448E5 E993F421 E55A239F 2AB5C5D5 FDB6E81B 40938E2A 54120CA3 E6E19C7A D750FC35 31DAEAB7$_{16}$
Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B$_{16}$
X	0DC53006 00CAB840 B38448E5 E993F421 5780A6E2 B69EAFBB 258726D7 B6718523 E55A239F$_{16}$

А.5 Шифрование в режиме сцепления блоков

В таблицах А.11, А.12 представлены примеры шифрования в режиме сцепления блоков.

Таблица А.11 — Зашифрование в режиме сцепления блоков

K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
Y	10116EFA E6AD58EE 14852E11 DA1B8A74 5CF2480E 8D03F1C1 9492E53E D3A70F60 657C1EE8 C0E0AE5B 58388BF8 A68E3309$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612$_{16}$
Y	10116EFA E6AD58EE 14852E11 DA1B8A74 6A9BBADC AF73F968 F875DEDC 0A44F6B1 5CF2480E$_{16}$

Таблица А.12 — Расшифрование в режиме сцепления блоков

K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
S	7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$
Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779 9EB23D31$_{16}$
X	730894D6 158E17CC 1600185A 8F411CAB 0471FF85 C8379239 8D8924EB D57D03DB 95B97A9B 7907E4B0 20960455 E46176F8$_{16}$
Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B$_{16}$
X	730894D6 158E17CC 1600185A 8F411CAB B6AB7AF8 541CF857 55B8EA27 239F08D2 166646E4$_{16}$

А.6 Шифрование в режиме гаммирования с обратной связью

В таблицах А.13, А.14 представлены примеры шифрования в режиме гаммирования с обратной связью.

Таблица А.13 — Зашифрование в режиме гаммирования с обратной связью

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1$_{16}$
Y	C31E490A 90EFA374 626CC99E 4B7B8540 A6E48685 464A5A06 849C9CA7 69A1B0AE 55C2CC59 39303EC8 32DD2FE1 6C8E5A1B$_{16}$

Таблица А.14 — Расшифрование в режиме гаммирования с обратной связью

Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779 9EB23D31$_{16}$
K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
S	7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$
X	FA9D107A 86F375EE 65CD1DB8 81224BD0 16AFF814 938ED39B 3361ABB0 BF0851B6 52244EB0 6842DD4C 94AA4500 774E40BB$_{16}$

А.7 Шифрование в режиме счетчика

В таблицах А.15, А.16 представлены примеры шифрования в режиме счетчика.

Таблица А.15 — Зашифрование в режиме счетчика

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1$_{16}$
Y	52C9AF96 FF50F644 35FC43DE F56BD797 D5B5B1FF 79FB4125 7AB9CDF6 E63E81F8 F0034147 3EAE4098 33622DE0 5213773A$_{16}$

Таблица А.16 — Расшифрование в режиме счетчика

Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779$_{16}$
K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
S	7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$
X	DF181ED0 08A20F43 DCBBB936 50DAD34B 389CDEE5 826D40E2 D4BD80F4 9A93F5D2 12F63331 66456F16 9043CC5F$_{16}$

А.8 Выработка имитовставки

В таблице А.17 представлены примеры выработки имитовставки.

Таблица А.17 — Выработка имитовставки

K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
X	B194BAC8 0A08F53B 366D008E 58$_{16}$
Y	7260DA60 138F96C9$_{16}$
X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
Y	2DAB5977 1B4B16D0$_{16}$

А.9 Аутентифицированное шифрование данных

В таблице А.18 представлены примеры применения операции $*$. В таблицах А.19, А.20 представлены примеры установки и снятия защиты данных.

Таблица А.18 — Операция $*$

u	34904055 11BE3297 1343724C 5AB793E9$_{16}$
v	22481783 8761A9D6 E3EC9689 110FB0F3$_{16}$
$u * v$	0001D107 FC67DE40 04DC2C80 3DFD95C3$_{16}$
u	703FCCF0 95EE8DF1 C1ABF8EE 8DF1C1AB$_{16}$
v	2055704E 2EDB48FE 87E74075 A5E77EB1$_{16}$
$u * v$	4A5C9593 8B3FE8F6 74D59BC1 EB356079$_{16}$

Таблица А.19 — Установка защиты данных

I	8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1$_{16}$
	belt-dwp
X	B194BAC8 0A08F53B 366D008E 584A5DE4$_{16}$
Y	52C9AF96 FF50F644 35FC43DE F56BD797$_{16}$
T	3B2E0AEB 2B91854B$_{16}$
	belt-che
X	B194BAC8 0A08F53B 366D008E 584A5D$_{16}$
Y	BF3DAEAF 5D18D2BC C30EA62D 2E70A4$_{16}$
T	548622B8 44123FF7$_{16}$

Таблица А.20 — Снятие защиты данных

I	C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779 9EB23D31$_{16}$
K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
S	7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$

belt-dwp^{-1}	
Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1$_{16}$
T	6A2C2C94 C4150DC0$_{16}$
X	DF181ED0 08A20F43 DCBBB936 50DAD34B$_{16}$

belt-che^{-1}	
Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638$_{16}$
T	7D9D4F59 D40D197D$_{16}$
X	2BABF43E B37B5398 A9068F31 A3C758B7 62F44AA9$_{16}$

А.10 Аутентифицированное шифрование ключа

В таблицах А.21, А.22 представлены примеры установки и снятия защиты ключа.

Таблица А.21 — Установка защиты ключа

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D$_{16}$
I	5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
Y	49A38EE1 08D6C742 E52B774F 00A6EF98 B106CBD1 3EA4FB06 80323051 BC04DF76 E487B055 C69BCF54 1176169F 1DC9F6C8$_{16}$

Таблица А.22 — Снятие защиты ключа

Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779 9EB23D31$_{16}$
I	B5EF68D8 E4A39E56 7153DE13 D72254EE$_{16}$
K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
X	92632EE0 C21AD9E0 9A39343E 5C07DAA4 889B03F2 E6847EB1 52EC99F7 A4D9F154$_{16}$

А.11 Хэширование

В таблице А.23 представлены примеры хэширования.

Таблица А.23 — Хэширование

X	B194BAC8 0A08F53B 366D008E 58$_{16}$
Y	ABEF9725 D4C5A835 97A367D1 4494CC25 42F20F65 9DDFECC9 61A3EC55 0CBA8C75$_{16}$

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D$_{16}$
Y	749E4C36 53AECE5E 48DB4761 227742EB 6DBE13F4 A80F7BEF F1A9CF8D 10EE7786$_{16}$

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
Y	9D02EE44 6FB6A29F E5C982D4 B13AF9D3 E90861BC 4CEF27CF 306BFB0B 174A154A$_{16}$

А.12 Дисковое шифрование

В таблицах А.24, А.25 представлены примеры дискового шифрования.

Таблица А.24 — Дисковое зашифрование

X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B$_{16}$
K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1$_{16}$

Блоковое	
Y	E9CAB32D 879CC50C 10378EB0 7C10F263 07257E2D BE2B854C BC9F3828 2D59D6A7 7F952001 C5D1244F 53210A27 C216D4BB$_{16}$

Секторное	
Y	1FCBB018 52003D60 B66024C5 08608BAA 2C21AF1E 884CF311 54D3077D 4643CF22 49EB2F5A 68E4BA01 9D90211A 81D690D9$_{16}$

Таблица А.25 — Дисковое расшифрование

Y	E12BDC1A E28257EC 703FCCF0 95EE8DF1 C1AB7638 9FE678CA F7C6F860 D5BB9C4F F33C657B 637C306A DD4EA779 9EB23D31$_{16}$
K	92BD9B1C E5D14101 5445FBC9 5E4D0EF2 682080AA 227D642F 2687F934 90405511$_{16}$
S	7ECDA4D0 1544AF8C A58450BF 66D2E88A$_{16}$

Блоковое	
X	7041BC22 6352C706 D00EA8EF 23CFE46A FAE11857 7D037FAC DC36E4EC C1F65746 09F23694 3FB809E1 BEE4A1C6 86C13ACC$_{16}$

Секторное	
X	E9FDF3F7 88657332 E6C46FCF 5251B8A6 D43543A9 3E323383 7DB15711 83A6EF4D 7FEB5CDF 999E1A3F 51A5A338 1BEB7FA5$_{16}$

А.13 Шифрование с сохранением формата

В таблице А.26 представлены примеры зашифрования с сохранением формата. В таблице символы слов $X, Y \in \mathbb{Z}_m^n$ представляются десятичными числами и разделяются запятыми.

Таблица А.26 — Зашифрование с сохранением формата

K	E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6$_{16}$
S	BE329713 43FC9A48 A02A885F 194B09A1$_{16}$

m	10
n	10
X	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Y	6, 9, 3, 4, 7, 7, 0, 3, 5, 2

m	58
n	21
X	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
Y	7, 4, 6, 21, 49, 55, 24, 23, 22, 50, 27, 39, 24, 24, 17, 32, 57, 43, 26, 5, 29

m	65536
n	17
X	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16
Y	14290, 31359, 58054, 51842, 44653, 34762, 28652, 48929, 6541, 13788, 7784, 46182, 61098, 43056, 3564, 21568, 63878,

A.14 Расширение ключа

В таблице А.27 представлены примеры расширения ключа.

Таблица А.27 — Расширение ключа

K_1	$\texttt{E9DEE72C}_{16}$	K_3	$\texttt{2DDB49F4}_{16}$	K_5	$\texttt{06075316}_{16}$
K_2	$\texttt{8F0C0FA6}_{16}$	K_4	$\texttt{6F739647}_{16}$	K_6	$\texttt{ED247A37}_{16}$

$n = 4$	
K	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 E9DEE72C 8F0C0FA6 2DDB49F4 6F739647}_{16}$

$n = 6$	
K	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 4B09A17E 8450BF66}_{16}$

A.15 Преобразование ключа

В таблице А.28 представлены примеры преобразования ключа.

Таблица А.28 — Преобразование ключа

X	$\texttt{E9DEE72C 8F0C0FA6 2DDB49F4 6F739647 06075316 ED247A37 39CBA383 03A98BF6}_{16}$
D	$\texttt{01000000 00000000 00000000}_{16}$
I	$\texttt{5BE3D612 17B96181 FE6786AD 716B890B}_{16}$

m	128
Y	$\texttt{6BBBC233 6670D31A B83DAA90 D52C0541}_{16}$

m	192
Y	$\texttt{9A2532A1 8CBAF145 398D5A95 FEEA6C82 5B9C1971 56A00275}_{16}$

m	256
Y	$\texttt{76E166E6 AB21256B 6739397B 672B8796 14B81CF0 5955FC3A B09343A7 45C48F77}_{16}$

Приложение Б
(рекомендуемое)
Модуль АСН.1

В модуле АСН.1 алгоритмам настоящего стандарта назначаются идентификаторы. Назначение выполняется указанием имени алгоритма (группы алгоритмов) и соответствующего идентификатора. Имя представляет собой объединение короткого имени алгоритма, указанного в интерфейсе, с 3-символьным суффиксом `128`, `192` или `256`. Суффикс обозначает длину используемого ключа. Суффикс не добавляется к именам `belt-compress`, `belt-hash`, `belt-keyexpand`, `belt-keyrep`.

Группа может включать два алгоритма: зашифрования (установки защиты) и расшифрования (снятия защиты). Идентификатор относится к обоим алгоритмам группы.

В модуле АСН.1 дополнительно определяются форматы следующих параметров:

`IV`	Синхропосылка в алгоритмах `belt-cbcXXX`, `belt-cfbXXX`, `belt-ctrXXX`, `belt-dwpXXX`, `belt-cheXXX`, `belt-bdeXXX`, `belt-sdeXXX`, `belt-fmtXXX`;
`KeyHeader`	Заголовок ключа в алгоритмах `belt-kwpXXX`, `belt-keyrep`;
`KeyLevel`	Уровень ключа в алгоритме `belt-keyrep`.

Если алгоритм шифрования описывается типом

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

то в компоненте `algorithm` должен указываться идентификатор алгоритма, а в компоненте `parameters` — используемая синхропосылка типа `IV`. Если синхропосылка не используется, то компонент `parameters` должен быть опущен. Отсутствие `parameters` при описании алгоритмов `belt-fmtXXX` означает, что используется нулевая синхропосылка.

Модуль АСН.1 имеет следующий вид:

```
Belt-module-v2 {iso(1) member-body(2) by(112) 0 2 0 34 101 31 module(1) ver2(2)}
DEFINITIONS ::=
BEGIN
  belt OBJECT IDENTIFIER ::= {iso(1) member-body(2) by(112) 0 2 0 34 101 31}

  belt-block128 OBJECT IDENTIFIER ::= {belt 3}
  belt-block192 OBJECT IDENTIFIER ::= {belt 4}
  belt-block256 OBJECT IDENTIFIER ::= {belt 5}
  belt-wblock128 OBJECT IDENTIFIER ::= {belt 6}
  belt-wblock192 OBJECT IDENTIFIER ::= {belt 7}
  belt-wblock256 OBJECT IDENTIFIER ::= {belt 8}
  belt-compress OBJECT IDENTIFIER ::= {belt 9}
  belt-ecb128 OBJECT IDENTIFIER ::= {belt 11}
  belt-ecb192 OBJECT IDENTIFIER ::= {belt 12}
  belt-ecb256 OBJECT IDENTIFIER ::= {belt 13}
  belt-cbc128 OBJECT IDENTIFIER ::= {belt 21}
```

```
  belt-cbc192 OBJECT IDENTIFIER ::= {belt 22}
  belt-cbc256 OBJECT IDENTIFIER ::= {belt 23}
  belt-cfb128 OBJECT IDENTIFIER ::= {belt 31}
  belt-cfb192 OBJECT IDENTIFIER ::= {belt 32}
  belt-cfb256 OBJECT IDENTIFIER ::= {belt 33}
  belt-ctr128 OBJECT IDENTIFIER ::= {belt 41}
  belt-ctr192 OBJECT IDENTIFIER ::= {belt 42}
  belt-ctr256 OBJECT IDENTIFIER ::= {belt 43}
  belt-mac128 OBJECT IDENTIFIER ::= {belt 51}
  belt-mac192 OBJECT IDENTIFIER ::= {belt 52}
  belt-mac256 OBJECT IDENTIFIER ::= {belt 53}
  belt-dwp128 OBJECT IDENTIFIER ::= {belt 61}
  belt-dwp192 OBJECT IDENTIFIER ::= {belt 62}
  belt-dwp256 OBJECT IDENTIFIER ::= {belt 63}
  belt-che128 OBJECT IDENTIFIER ::= {belt 64}
  belt-che192 OBJECT IDENTIFIER ::= {belt 65}
  belt-che256 OBJECT IDENTIFIER ::= {belt 66}
  belt-kwp128 OBJECT IDENTIFIER ::= {belt 71}
  belt-kwp192 OBJECT IDENTIFIER ::= {belt 72}
  belt-kwp256 OBJECT IDENTIFIER ::= {belt 73}
  belt-hash OBJECT IDENTIFIER ::= {belt 81}
  belt-keyexpand OBJECT IDENTIFIER ::= {belt 91}
  belt-keyrep OBJECT IDENTIFIER ::= {belt 101}
  belt-bde128 OBJECT IDENTIFIER ::= {belt 111}
  belt-bde192 OBJECT IDENTIFIER ::= {belt 112}
  belt-bde256 OBJECT IDENTIFIER ::= {belt 113}
  belt-sde128 OBJECT IDENTIFIER ::= {belt 121}
  belt-sde192 OBJECT IDENTIFIER ::= {belt 122}
  belt-sde256 OBJECT IDENTIFIER ::= {belt 123}
  belt-fmt128 OBJECT IDENTIFIER ::= {belt 131}
  belt-fmt192 OBJECT IDENTIFIER ::= {belt 132}
  belt-fmt256 OBJECT IDENTIFIER ::= {belt 133}

  IV ::= OCTET STRING (SIZE(16))
  KeyHeader ::= OCTET STRING (SIZE(16))
  KeyLevel ::= OCTET STRING (SIZE(12))
END
```

Приложение В
(обязательное)
Квоты ключей шифрования данных

Алгоритмы шифрования `belt-cbc`, `belt-cfb`, `belt-ctr`, `belt-dwp`, `belt-che` остаются надежными до тех пор, пока соблюдаются квоты для используемых в них ключей. Квота ключа — это максимальный объем данных, которые разрешается зашифровать на этом ключе. Квота задается количеством блоков зашифровываемых сообщений, которое нельзя превысить. Учитываются, в том числе, последние и поэтому возможно неполные блоки. В алгоритмах `belt-dwp`, `belt-che` каждая имитовставка засчитывается как блок.

Надежность алгоритмов шифрования означает, что гипотетическому противнику трудно отличить зашифрованные сообщения Y от случайных слов. Противник выбирает открытые сообщения X, управляет синхропосылками S по правилам, изложенным в 5.3, и получает в ответ либо штатные Y, вычисленные на неизвестном ключе K, либо случайные слова такой же длины. Противнику предлагается распознать тип преобразования, которое применяется к сообщениям X: случайное или штатное.

Качество распознавания характеризует преобладание p. Оно имеет вид $|1 - \alpha - \beta|$, где α — вероятность признать случайное преобразование штатным, β — вероятность признать штатное преобразование случайным. Величина p является оценкой сверху для вероятности успеха любой разумной атаки на алгоритм шифрования при выбираемых X. В частности, малость p означает трудность получения по Y любой информации о сообщении X, кроме его длины.

Квота ключа ограничивает преобладание p. В таблице В.1 представлены квоты, при соблюдении которых p не превышает пороги 2^{-32}, 2^{-48}, 2^{-64}. Пороги определяют уровень гарантий безопасности шифрования. Первый порог дает средние гарантии, второй — высокие, третий — максимальные.

Таблица В.1 — Квоты ключей

Уровень гарантий	`belt-cbc`	`belt-cfb`	`belt-ctr`	`belt-dwp`	`belt-che`
Средний ($p \leqslant 2^{-32}$)	2^{48}	2^{48}	$2^{48}\sqrt{\frac{2}{3}}$	$2^{48}\sqrt{\frac{2}{7D+7}}$	$2^{48}\sqrt{\frac{2}{5D+7}}$
Высокий ($p \leqslant 2^{-48}$)	2^{40}	2^{40}	$2^{40}\sqrt{\frac{2}{3}}$	$2^{40}\sqrt{\frac{2}{7D+7}}$	$2^{40}\sqrt{\frac{2}{5D+7}}$
Максимальный ($p \leqslant 2^{-64}$)	2^{32}	2^{32}	$2^{32}\sqrt{\frac{2}{3}}$	$2^{32}\sqrt{\frac{2}{7D+7}}$	$2^{32}\sqrt{\frac{2}{5D+7}}$

Величина D в последних двух столбцах таблицы — это максимальное суммарное количество блоков X и I вместе с блоком длин $\langle |X| \rangle_{64} \parallel \langle |I| \rangle_{64}$, обрабатываемых алгоритмом `belt-dwp` или `belt-che` в целевой системе защиты информации.

***Пример** — Пусть данные в системе обрабатываются пакетами. Защищенный пакет состоит из заголовка I, синхропосылки S, результата зашифрования сообщения X и имитовставки T. Пусть длина I (в октетах) равняется* 46*, а длина X (снова в октетах) не превосходит* 1408*. Тогда*

$$D = \lceil 46/16 \rceil + \lceil 1408/16 \rceil + 1 = 3 + 88 + 1 = 92.$$

Если используется алгоритм `belt-dwp`, то для обеспечения максимальных гарантий безопасности суммарное число блоков X и имитовставок T не должно

превосходить

$$2^{32}\sqrt{\frac{2}{7\cdot 92+7}} \approx 2^{27.8}.$$

Квоты не вводятся для ключей алгоритмов шифрования, в которых синхропосылка S либо не используется (`belt-ecb`), либо ее разрешается многократно использовать с выбранным ключом (`belt-bde`, `belt-sde`, `belt-fmt`). Для таких алгоритмов повтор X или (X, S) приводит к повтору Y, и описанная выше концепция надежности неприменима.

Квоты не применяются, если зашифровываются ключи и другие высокоэнтропийные данные, которыми не может манипулировать противник.

Приложение Г
(справочное)
Сведения о предыдущей редакции

Часть алгоритмов настоящего стандарта установлена в его предыдущей редакции — СТБ 34.101.31-2011. Наследуемые алгоритмы перечислены в таблице Г.1.

Алгоритмы перенесены в настоящий стандарт без функциональных изменений.

Таблица Г.1

Алгоритм (группа алгоритмов)	Пункт СТБ 34.101.31-2011
`belt-block`	6.1
`belt-ecb`	6.2
`belt-cbc`	6.3
`belt-cfb`	6.4
`belt-ctr`	6.5
`belt-mac`	6.6
`belt-dwp`	6.7
`belt-kwp`	6.8
`belt-hash`	6.9
`belt-keyexpand`	7.1
`belt-keyrep`	7.2

Библиография

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

Поправка к официальной редакции

В каком месте	Напечатано	Должно быть
Приложение А, наименование	Тестовые примеры	Проверочные примеры
Приложение А, таблица А.22, столбец 1	X I K Y	Y I K X
Приложение А, таблица А.25, столбец 1	X K S $\overline{}$ Y $\overline{}$ Y	Y K S $\overline{}$ X $\overline{}$ X
Приложение А, таблица А.8, строка X	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D$_{16}$	B194BAC8 0A08F53B 366D008E 584A5DE4 8504FA9D 1BB6C7AC 252E72C2 02FDCE0D 5BE3D612 17B96181 FE6786AD 716B890B 5CB0C0FF 33C356B8 35C405AE D8E07F99$_{16}$