

АВТОМАТИЗАЦІЯ ТА ІНТЕЛЕКТУАЛІЗАЦІЯ ПРИЛАДОБУДУВАННЯ

УДК 681.3 : 621

АЛГОРИТМ ХЕШ-ФУНКЦИИ С НОВЫМИ БАЗОВЫМИ ПРЕОБРАЗОВАНИЯМИ

Акбаров Д. Е., Умаров Ш. А.

Ферганский филиал Ташкентского университета информационных технологий,

г. Фергана, Узбекистан

E-mail: sht00357@gmail.com

Данный алгоритм и процедура вычисления хэш-функции (ХФ) для любой последовательности двоичных символов, которые применяются в криптографических методах обработки и защиты информации, в том числе для реализации процедур электронной цифровой подписи (ЭЦП) при передаче, обработке и хранении информации в автоматизированных системах. Предлагаемый алгоритм konstruирован на основе новых крипто стойких преобразований: S-блок преобразований байтов, таблица сжатия, логические функции, которые по свойствам обеспечат эффективные распространения и рассеивания блоков и под блоков хешируемых сообщений.

Ключевые слова: *алгоритм, хеш-функция, электронно цифровая подпись, S-блок преобразований байтов, таблица сжатия, логическая функция.*

Введение. Постановка проблемы

Расширяющееся применение информационных технологий при создании, обработке, передаче и хранении документов требует в определенных случаях сохранения конфиденциальности их содержания, обеспечения полноты и достоверности – целостности.

Одним из эффективных направлений защиты информации является криптография (криптографическая защита), широко применяемая в различных сферах деятельности в государственных и коммерческих структурах.

Криптографические методы защиты информации являются объектом серьезных научных исследований и стандартизации на национальных и международных уровнях.

Предлагаемый алгоритм с новыми преобразованиями определяет процедуру вычисления хэш-функции для любой последовательности двоичных символов.

Постановка задачи

Функции хеширования заключается в сопоставлении произвольного набора данных в виде последовательности двоичных символов и его образа фиксированной небольшой длины, что позволяет использовать эту функцию в процедурах электронной цифровой подписи для сокращения времени подписания и проверки подписи. Эффект сокращения времени, достигается за счет вычисления подписи только под образом

подписываемого набора данных.

Предлагаемая хэш-функция предназначена для проверки полноты информации криптографическим методом, и её значение используется как входное данное вместе с секретным ключом для генерации электронной цифровой подписи по соответствующему электронному документу. Этот алгоритм хэш-функции обрабатывает сообщение произвольной длины блоками размером 256 битов и вырабатывает блок хэш - значения длиной 256 битов.

Данный алгоритм и процедура вычисления хэш-функции (ХФ) для любой последовательности двоичных символов, которые применяются в криптографических методах обработки и защиты информации, в том числе для реализации процедур электронной цифровой подписи (ЭЦП) при передаче, обработке и хранении информации в автоматизированных системах.

Решение задачи. В приводимом описании предлагаемого алгоритма процедура вычисления хэш - значение состоит из следующих шагов:

Шаг 1. Добавление битов заполнения.

Хешируемое сообщение произвольной длины разбивается на блоки длиной 256 битов. Если длина последнего блока меньше, чем 256 бита, то последний блок дополняется нулями до 256 битов.

Шаг 2. Добавление значение длины сообщения.

К результату выполнения шага 1 добавляется 256-битовое значение длины в битах исходного сообщения. Битовая длина хешируемого

сообщения получается от значения длины исходного сообщения по mod 2^{256} .

Шаг 3. Добавление значение контрольной суммы.

К результату выполнения шага 2 добавляется 256-битовое значение контрольной суммы. Блок контрольной суммы вычисляется после добавления битов заполнения, как суммы по mod 2^{256} всех блоков дополненного сообщения.

Шаг 4. Обработка сообщение блоками по 256 битов.

После этих шагов расширенное сообщение разбивается на блоки длиной 256 бит. Предположим, количество блоков равно N . Обозначим эти блоки через M_1, M_2, \dots, M_N .

Обработка сообщения блоками по 256 битов выполняется следующим образом:

1. Вводится блок заданных чисел $S = S_0, \dots, S_{255}$,

где $0 \leq S_l \leq 255, l = 0, 255$ (S -блок будет единым для группы пользователей). Вводим 256 битовое стартовое хэш-значение $H=0$ и положим $n=1$.

2. Вводится блок хэшируемого сообщения M_n :

$$\dot{I}_n = m_0(0)m_1(0)\dots m_{255}(0) \text{ (256 битов).}$$

3. Если $n=1$, то $T(0) = \dot{I}_n$, иначе $T(0) = H$. Тогда генерируется первые 32 байта ключа следующим образом:

$$k_0 = t_0(0)t_1(0)\dots t_7(0), \quad k_1 = t_8(0)t_9(0)\dots t_{15}(0), \dots,$$

$$k_{31} = t_{248}(0)t_{249}(0)\dots t_{255}(0).$$

Блок $T(0)$ циклически сдвигается налево 29 битов и его результат обозначается через $T(1)$:

$$T(1) = T(0) \lll 29 = t_{29}(0)t_{30}(0)\dots t_{255}(0)t_0(0)\dots t_{28}(0) = t_0(1)t_1(1)\dots t_{255}(1).$$

От блока $T(1)$ генерируются следующие 32 байта ключа следующим образом:

$$k_{0+32.1} = t_0(1)t_1(1)\dots t_7(1), \quad k_{1+32.1} = t_8(1)t_9(1)\dots t_{15}(1), \dots,$$

$$k_{31+32.1} = t_{248}(1)t_{249}(1)\dots t_{255}(1).$$

Блок $T(1)$ циклически сдвигается налево 29 битов и результат обозначается через $T(2)$:

$$T(2) = T(1) \lll 29 = t_{29}(1)t_{30}(1)\dots t_{255}(1)t_0(1)\dots t_{28}(1) = t_0(2)t_1(2)\dots t_{255}(2).$$

От блока $T(2)$ генерируются последующие 32 байта ключа следующим образом:

$$k_{0+32.2} = t_0(2)t_1(2)\dots t_7(2),$$

$$k_{1+32.2} = t_8(2)t_9(2)\dots t_{15}(2), \dots,$$

$$k_{31+32.2} = t_{248}(2)t_{249}(2)\dots t_{255}(2).$$

Аналогично, из выражений $T(3), T(4), T(5), T(6)$ генерируются следующие ключи:

$$k_{0+32.3}, \dots, k_{31+32.3}, k_{0+32.4}, \dots, k_{31+32.4},$$

$$k_{0+32.5}, \dots, k_{31+32.5}, k_{0+32.6}, \dots, k_{31+32.6}.$$

Блок $T(6)$ циклически сдвигается налево 29 битов и результат обозначается через $T(7)$:

$$T(7) = T(6) \lll 29 = t_{29}(6)t_{30}(6)\dots t_{255}(6)t_0(6)\dots t_{28}(6) = t_0(7)t_1(7)\dots t_{255}(7).$$

От блока $T(7)$ генерируется последние 32 байта ключа следующим образом:

$$k_{0+32.7} = t_0(7)t_1(7)\dots t_7(7), \quad k_{1+32.7} = t_8(7)t_9(7)\dots t_{15}(7), \dots,$$

$$k_{31+32.7} = t_{248}(7)t_{249}(7)\dots t_{255}(7).$$

4. Используя, генерированных ключей и с помощью S-блока на основе несложных преобразований генерируется новый 256 байтовый ключ KR :

4.1. $i=0; j=0;$

4.2. $j=(j + S_i + k_i) \bmod 256;$

4.3. $P=S_i; Q=S_j;$

4.4. $S_i=Q; S_j=P;$

4.5. Проверяется условие $i < 255$. Если выполняется это условие, то полагая $i=i+1$ переходит к шагу 4.2., иначе переход к следующему шагу;

4.6. $i=0; j=0; l=0;$

4.7. $i=(i+1) \bmod 256;$

4.8. $j=(j+S_i) \bmod 256;$

4.9. $P=S_i; Q=S_j;$

4.10. $S_i=Q; S_j=P;$

4.11. $t=(S_i+S_j) \bmod 256;$

4.12. $kr_l=S_i;$

4.13. $l=l+1;$

4.14. Проверяется условие $l < 256$. Если выполняется это условие, то переходит к шагу 4.7., иначе заканчивается генерации ключа $KR = kr_0, kr_1, \dots, kr_l, \dots, kr_{255}$.

5. Полагается $i=0$ и $T(0) = M_n$.

6. Ключ $KRI = kr_{0+32.i}, \dots, kr_{31+32.i}$ ($\hat{a} \hat{e} \hat{o} \hat{i} \hat{a}$) и блоком $T(0) = t_0(0)\dots t_7(0)t_8(0)\dots t_{15}(0)\dots t_{255}(0)$ выполняем операцию XOR и обозначим результат этой операции через $H(0) = h_0(0)h_1(0)\dots h_{255}(0)$, т. е. $KRI \oplus T(0) = H(0)$.

7. Полученный 256-битовой блок разделяется на четыре 64-битовые блоки $X = h_0(0)\dots h_{63}(0)$,

$$Y = h_{64}(0)\dots h_{127}(0), \quad Z = h_{128}(0)\dots h_{191}(0),$$

$W = h_{192}(0)\dots h_{255}(0)$ и вычисляются значения следующих логических функций:

$$F(X; Y; Z; W) = (\bar{X} \wedge Y) \wedge (Z \oplus W) \oplus W;$$

$$G(X; Y; Z; W) = \bar{W} \wedge ((X \wedge Z) \oplus \bar{Y}) \oplus (Z \wedge W);$$

$$R(X; Y; Z; W) = (X \wedge Y \wedge \bar{Z}) \oplus (Z \wedge \bar{W}) \oplus (\bar{X} \wedge W) \oplus (Y \wedge W);$$

$$V(X; Y; Z; W) = (Y \wedge W \wedge \bar{Z}) \oplus (Z \wedge \bar{W}) \oplus \bar{X}.$$

Здесь побитовые логические операции AND, OR, NOT, XOR обозначены соответственно знаками $\wedge, \vee, \bar{}, \oplus$. Значения этих логических функций будет 64-битовыми блоками.

8. Обозначается через $L(0)$ 256-битовой блок, полученный в результате конкатенации четырёх 64-битовых блоков значения логических функций, т.е.

$$L(0) = F(X; Y; Z; W) \parallel G(X; Y; Z; W) \parallel R(X; Y; Z; W) \parallel V(X; Y; Z; W) = l_0(0)l_1(0)\dots l_{255}(0).$$

9. Обозначается через $\dot{A}(0)$ 512-битовой блок, результат конкатенации $H(0)$ и $L(0)$, т.е.,

$$A(0) = H(0) \parallel L(0) = h_0(0)h_1(0)\dots h_{255}(0)l_0(0)l_1(0)\dots l_{255}(0) = a_0(0)a_1(0)\dots a_{511}(0)$$

10. Разделяется блок $\dot{A}(0)$ по 4 бита следующим образом:

$$\begin{aligned} a_0(0)a_1(0)a_2(0)a_3(0) &= x_0, \\ a_4(0)a_5(0)a_6(0)a_7(0) &= y_0, \\ a_8(0)a_9(0)a_{10}(0)a_{11}(0) &= x_1, \\ a_{12}(0)a_{13}(0)a_{14}(0)a_{15}(0) &= y_1, \dots \end{aligned}$$

$$\begin{aligned} a_{0+8\alpha}(0)a_{1+8\alpha}(0)a_{2+8\alpha}(0)a_{3+8\alpha}(0) &= x_\alpha, \\ a_{4+8\alpha}(0)a_{5+8\alpha}(0)a_{6+8\alpha}(0)a_{7+8\alpha}(0) &= y_\alpha, \dots, \\ a_{0+8\cdot 63}(0)a_{1+8\cdot 63}(0)a_{2+8\cdot 63}(0)a_{3+8\cdot 63}(0) &= x_{63}, \\ a_{4+8\cdot 63}(0)a_{5+8\cdot 63}(0)a_{6+8\cdot 63}(0)a_{7+8\cdot 63}(0) &= y_{63}. \end{aligned}$$

Обозначается: $x_\alpha \parallel y_\alpha = b_\alpha$.

11. По данной таблице сжатия, в которой определенном порядке расположены числа от 0 до 15, однобайтовый блок b_α преобразуется в полубайтовый блок, расположенный в пересечении строки x_α и столбца y_α (табл. 1).

Таблица 1.

	0	1	2	...	y_α	...	15
0	$d_0(0)$	$d_1(0)$	$d_2(0)$...	$d_{y_\alpha}(0)$...	$d_{15}(0)$
1	$d_0(1)$	$d_1(1)$	$d_2(1)$...	$d_{y_\alpha}(1)$...	$d_{15}(1)$
2	$d_0(2)$	$d_1(2)$	$d_2(2)$...	$d_{y_\alpha}(2)$...	$d_{15}(2)$
...
x_α	$d_0(x_\alpha)$	$d_1(x_\alpha)$	$d_2(x_\alpha)$...	$d_{y_\alpha}(x_\alpha)$...	$d_{15}(x_\alpha)$
...
15	$d_0(15)$	$d_1(15)$	$d_2(15)$...	$d_{y_\alpha}(15)$...	$d_{15}(15)$

Если обозначить через $\dot{\tilde{N}}$ преобразование таблицы сжатия, то через $\dot{O}(1)$ обозначается 256-битовой блок полученный в результате этого преобразования:

$$\begin{aligned} \dot{\tilde{N}}(\dot{a}_0, \dot{y}_0, \dot{a}_1, \dot{y}_1, \dots, \dot{a}_{63}, \dot{y}_{63}) (512 \text{ áèò}) &= \\ = d_{y_{63}}(x_0)d_{y_0}(x_{63})d_{y_{62}}(x_1)d_{y_1}(x_{62})\dots d_{y_{32}}(x_{31})d_{y_{31}}(x_{32}) (256 \text{ áèò}) &= \\ = t_0(1)t_1(1)\dots t_{255}(1) = T(1). \end{aligned}$$

12. $i=i+1$.

13. Проверяется условие $i < 8$. Если выполняется это условие, то полагается $T(0) = T(1)$, и осуществляется переход к шагу б, иначе переход к следующему шагу.

14. $H = T(1)$.

15. $n=n+1$.

16. Проверяется условие $n \leq N$. Если выполняется это условие, то осуществляется переход к шагу 2, иначе полученное значение H принимается за хэш - значением сообщения M .

Анализ результатов

Вычисление значения хэш-функции одного j -го блока представляется в виде:

$$T_i(1) = TC\{T_i(0) \oplus KRI_i \mid F[T_i(0) \oplus KRI_i]G[T_i(0) \oplus KRI_i]R[T_i(0) \oplus KRI_i]V[T_i(0) \oplus KRI_i]\}, (1)$$

где $T_i(0)$ – j -ый блок хешируемого сообщения;

$$T_i(0) = T_i(1), i=1,2,\dots,8; H_j = T_8(1).$$

Для следующего блока хешируемого сообщения, ключи KRI_i генерируются заново по

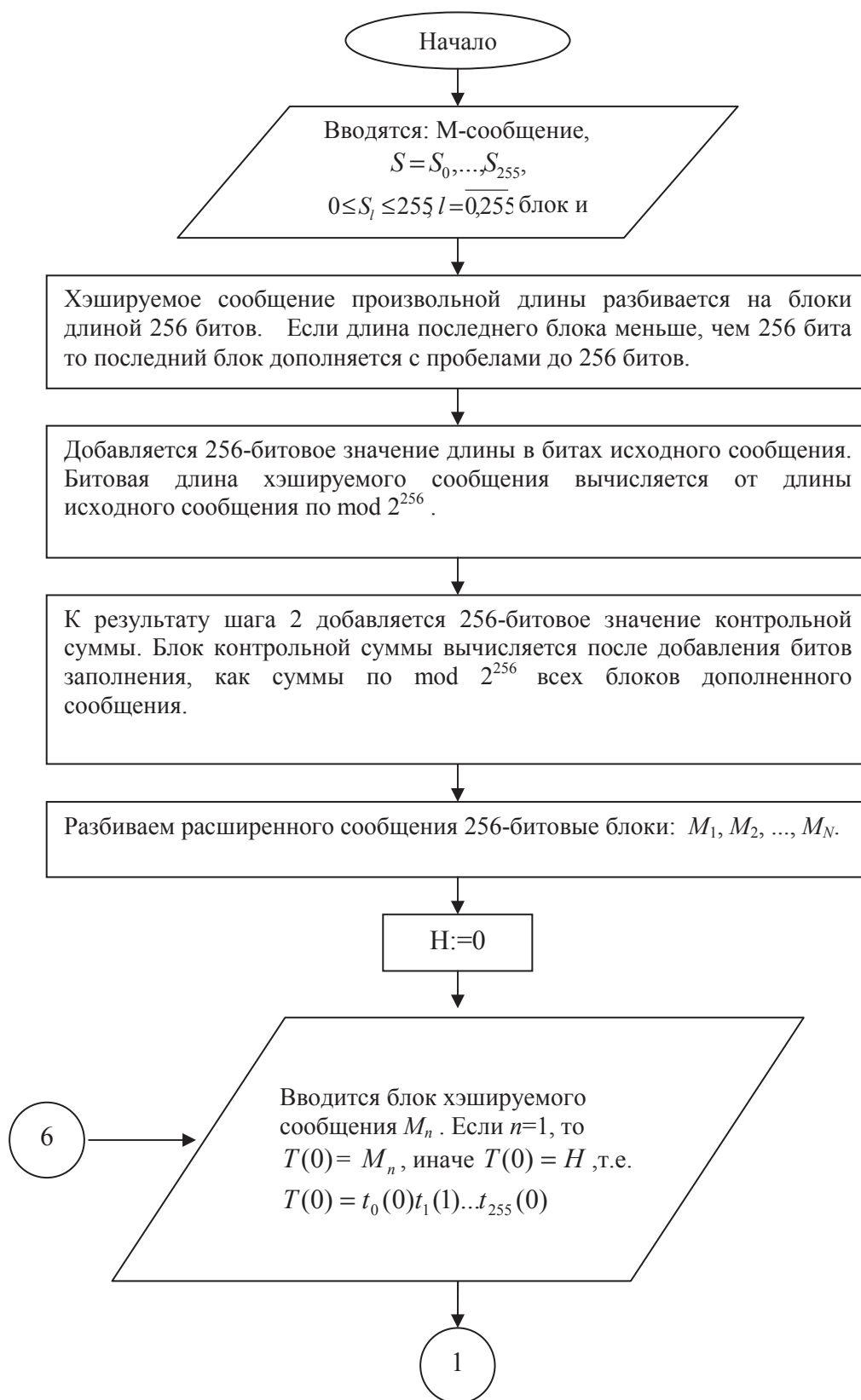
хэш значению предыдущего блока $H_j = T_8(1)$, а для первого блока ключи KRI_i генерируются от начального блока хешируемого сообщения. В правой 256-битовой части конкатенации выражения (1) функции $F[H_i(0)]$, $G[H_i(0)]$, $R[H_i(0)]$, $V[H_i(0)]$ дают перемешивание и рассеивание блока $H_i(0)$, который представляет собой левую 256-битовую часть 512-битовой конкатенации. Кроме того, конструкции этих функции позволяют преобразовать разные 256 – битовых блоков на разные 256 – битовые блоки, что после 8-и раундового цикла, не снижает стойкости хэш-функции.

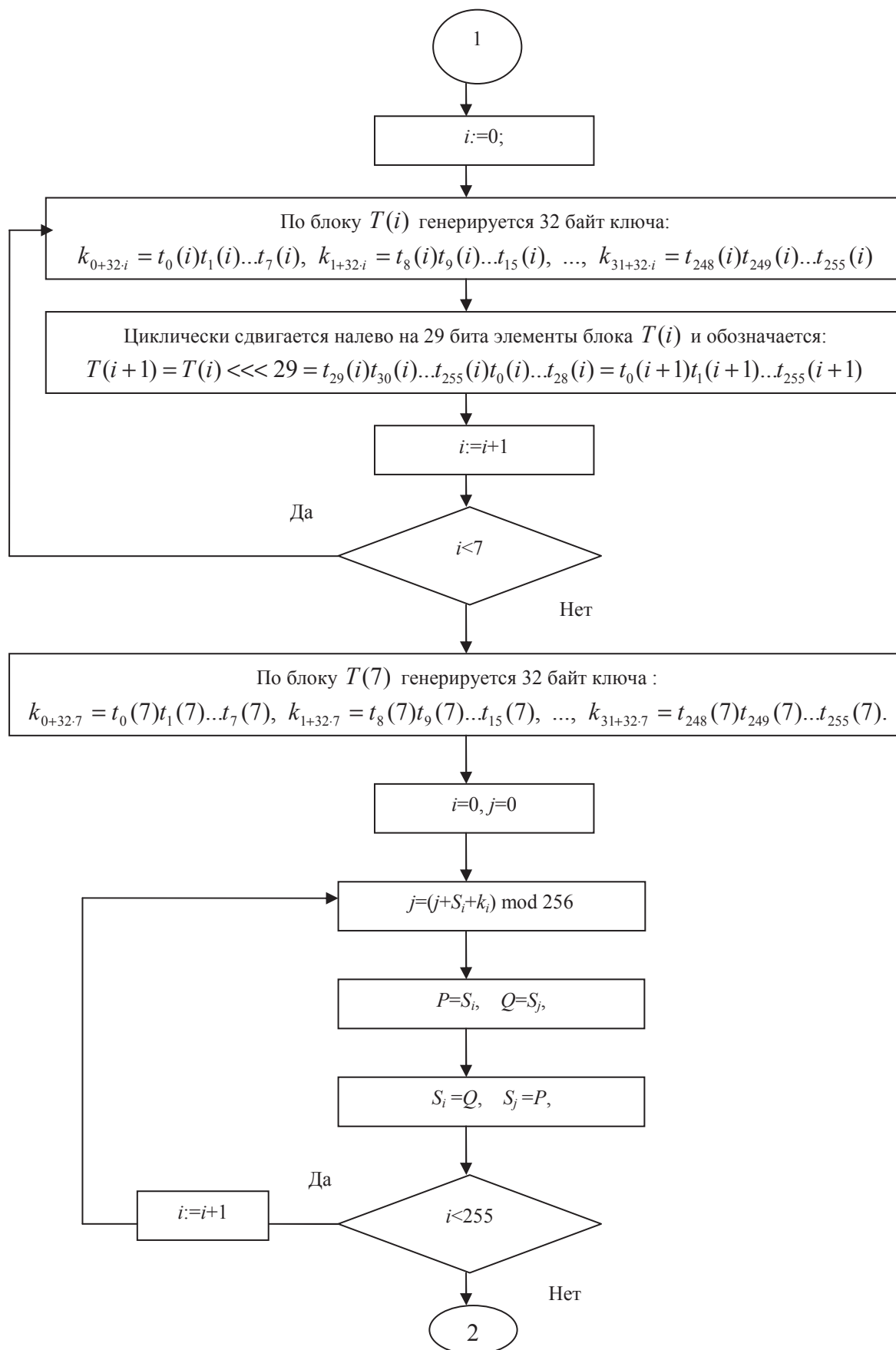
Заключение

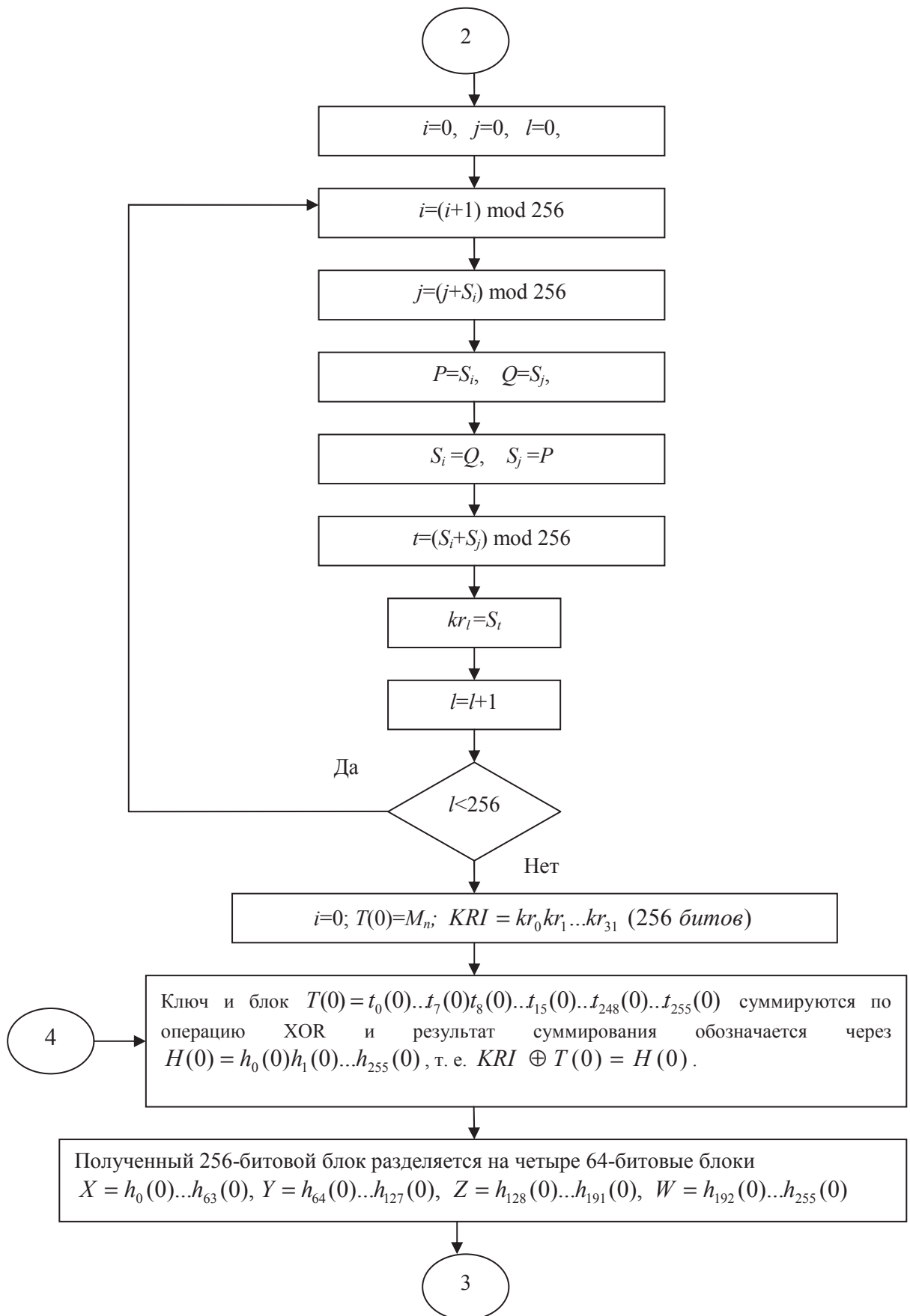
На основе идеи разработки предлагаемого алгоритма хэш-функции используются комбинацию преобразований: побитное сложение по mod 2, S-блок, таблица сжатия, логические функции. Генерация базовых преобразования с указанными свойствами со стороны (по усмотрению) группы пользователей является один из основных преимуществ в приложениях.

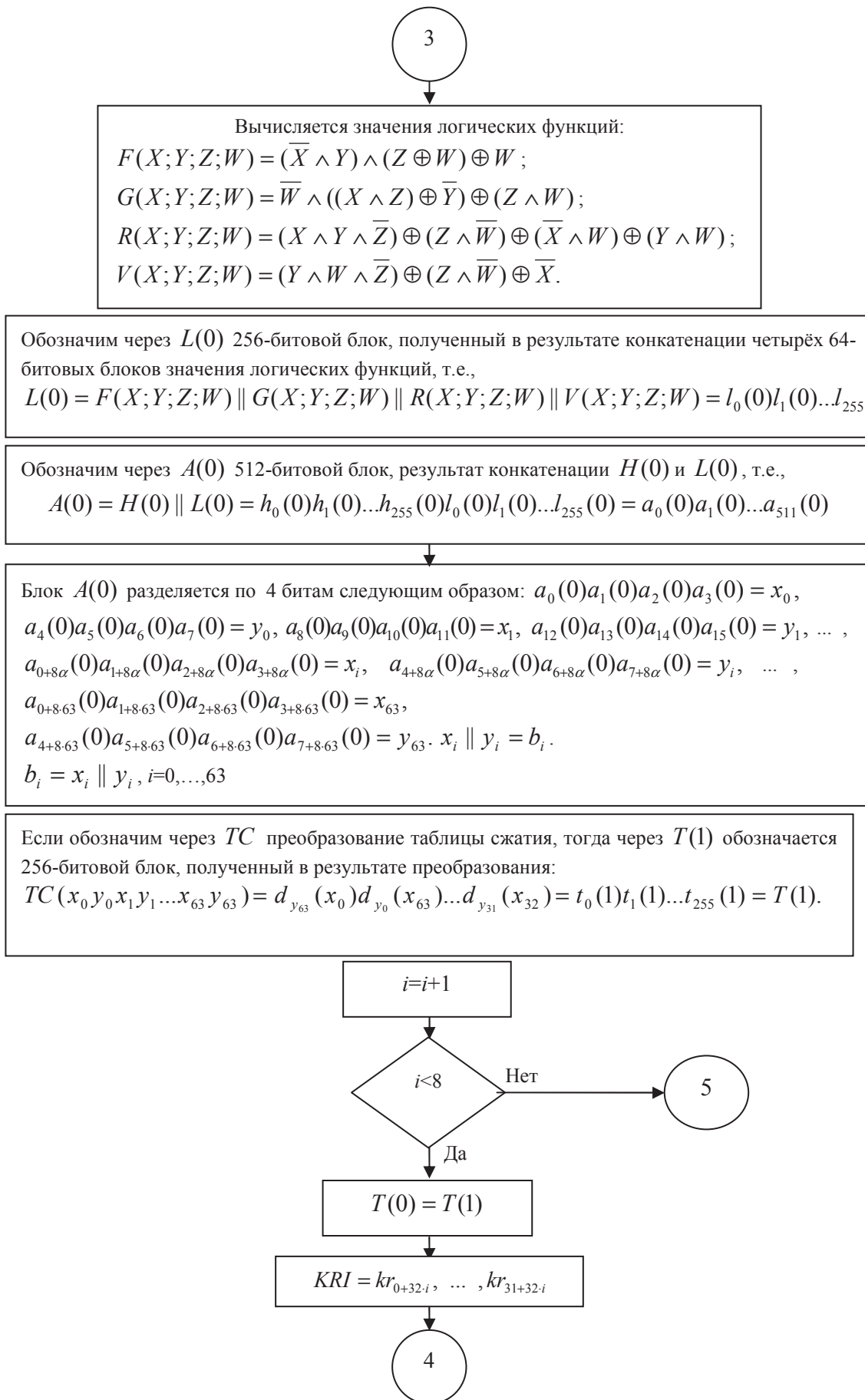
И конструкция и преобразования алгоритма имеют перспективного приложения так как их основы являются универсальными и гибкими к приложениям.

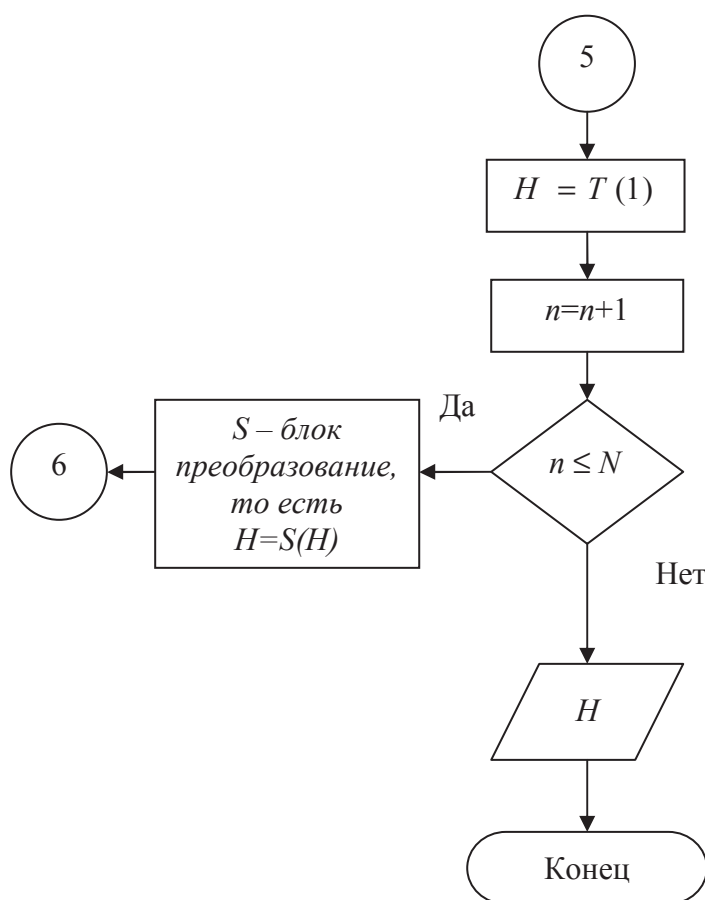
ПРИЛОЖЕНИЕ А. Блок-схема функции хеширования











Литература

1. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. –М.: издательство ТРИУМФ, 2003 - 816 с.
2. Алферов А. П., Зубов А. Ю., Кузьмин А. С., Черемушкин А. В. Основы криптографии: Учебное пособие, 2-е изд. –М.: Гелиос АРВ, 2002.-480 с.
3. Харин Ю. С., Берник В. И., Матвеев Г. В., Агиевич С. Г. «Математические и компьютерные основы криптологии» ООО «Новое знание» 2003 г. 381 с.

4. Соловьев Ю. П. Рациональные точки на эллиптических кривых /Соросовский образовательный журнал, №10, 1997, 138-143 стр5. Darrel Hankerson, Alfred T. Meners, Scott Vanstone. Guide to elliptic curve cryptography. QA76. 9. A 25 H 37 2003.- 311 p.
6. Акбаров Д. Е. Ахборот хавфсизлигини таъминлашнинг криптографик усуллари ва уларнинг қўлланилиши – Тошкент, “Ўзбекистон маркаси” 2009 – 432 б.

УДК 681.3 : 621

Д. Ё. Акбаров, Ш. А. Умаров

Ферганська філія Ташкентського університету інформаційних технологій, м. Фергана, Узбекистан

АЛГОРИТМ ХЕШ-ФУНКЦІЇ З НОВИМИ БАЗОВИМИ ПЕРЕТВОРЕННЯМИ

Обґрунтовано даний алгоритм і процедура обчислення хеш-функції (ХФ) для будь-якої послідовності двійкових символів, які застосовуються в криптографічних методах обробки та захисту інформації, у тому числі для реалізації процедур електронного цифрового підпису (ЕЦП) при передачі, обробці та зберіганні інформації в автоматизованих системах.

Пропонований алгоритм запропоновано конструювати на основі нових крипто стійких перетворень: S-блок перетворень байтів, таблиця стиснення, логічні функції, які за властивостями забезпечать ефективні поширення і розсіювання блоків і підблоків хешіруємих повідомлень.

Ключові слова: алгоритм, хеш-функція, електронно цифровий підпис, S-блок перетворень байтів, таблиця стиснення, логічна функція.

D. E. Akbarov, Sh. A. Umarov

Fergana branch of the Tashkent university information technologies, Fergana, Uzbekistan

ALGORITHM OF HASHING FUNCTION WITH NEW BASIC CONVERSIONS

This algorithm and procedure of computation of the hashing function (HF) for any sequence of binary characters which are applied in cryptography methods of processing and information security, including to implementation of procedures of the digital signature (DS) by transmission, processing and information storage in automated systems. The offered algorithm is constructed on the basis of new crypto permanent conversions: S-block conversions of bytes, the table of compression, logic functions which on properties will provide effective distribution and dispersion of units and under units of the hashed messages.

Keywords: algorithm, hashing function, electronically sign-code signature, S-block conversions of bytes, table of compression, logic function.

*Надійшла до редакції
26 квітня 2016 року*

*Рецензовано
11 травня 2016 року*

© Акбаров Д. Е., Умаров Ш. А., 2016