УДК 004.056.2

*А. В. СИДОРЕНКО, И. В. ШАКИНКО*

# АЛГОРИТМ ХЕШИРОВАНИЯ НА ОСНОВЕ ДВУМЕРНЫХ ХАОТИЧЕСКИХ ОТОБРАЖЕНИЙ

*Белорусский государственный университет*

*Предложен алгоритм хеширования на основе динамического хаоса. Благодаря использованию хаотических отображений, алгоритм является необратимым, а поиск двух сообщений с одинаковыми хеш-значениями становится вычислительно затруднительным. Предлагаемый алгоритм включает в себя следующие этапы: выбор значений переменных и параметров двумерных хаотических отображений; реализацию итераций хаотических отображений с добавлением элементов исходного сообщения к переменным; реализацию итераций хаотических отображений без добавления элементов исходного сообщения к переменным; формирование хеш-значения. Предлагается формировать два хеш-значения $h_1$ и $h_2$, в которых используется различный порядок переменных. Результирующее хеш-значение получается при применении операции «сложение по модулю два» к хеш-значениям $h_1$ и $h_2$. Проведено тестирование предлагаемого алгоритма. Из полученных данных следует, что для рассматриваемого алгоритма характерен лавинный эффект. Статистические характеристики последовательности, сформированной из хеш-значений, схожи со статистическими характеристиками последовательности, значения элементов которой получены случайным образом, что свидетельствует о работоспособности предлагаемого алгоритма. Вычислительный эксперимент проведен с использованием отображений Чирикова, «Кота Арнольда», Эно. Установлено, что для сообщений с размером превышающим 4 Кб, при использовании отображений Эно и «Кот Арнольда» предлагаемый алгоритм справляется с задачей более чем на 20% быстрее, чем алгоритм «Keccak». Предлагаемый алгоритм хеширования может быть использован при решении задач контроля целостности данных при передаче информации в современных телекоммуникационных системах.*

***Ключевые слова:*** *динамический хаос, хаотическое отображение, хеширование, целостность данных, информационная безопасность.*

## Введение

При передаче информации в телекоммуникационных системах одной из основных задач защиты информации является обеспечение и контроль ее целостности. Традиционно для этих целей используются алгоритмы хеширования [1]. Алгоритм хеширования преобразуя входные данные позволяет поставить в соответствие сообщению произвольного размера хеш-значение, представляющее собой последовательность бит фиксированного размера [2]. Контроль целостности данных осуществляется путем использования сравнительного анализа хеш-значений, переданного и принятого сообщений. При равенстве хеш-значений сообщение идентично переданному, в противном случае – сообщение модифицировано в канале передачи.

В данной работе предложен алгоритм хеширования, основанный на использовании двумерных хаотических отображений. Одной из особенностей хаотических отображений является чувствительность к начальным условиям [3], то есть изменения в исходном сообщении приводят к существенным изменениям в формируемом хеш-значении. А это является необходимым требованием для алгоритмов хеширования.

## Двумерные хаотические отображения

Двумерные дискретные хаотические отображения записываются в итерационной форме

$$\begin{cases} x_{i+1} = f_1(x_i, y_i) \\ y_{i+1} = f_2(x_i, y_i) \end{cases}, \quad (1)$$

где $x_i$ и $y_i$ – переменные хаотического отображения на $i$-той итерации, $f_1$ и $f_2$ – некоторые функции от двух переменных.

Под начальными условиями хаотических отображений понимают исходные значения пе-

ременных. В данной работе использовались следующие дискретные двумерные хаотические отображения: отображение Чирикова, «Кот Арнольда» и отображение Эно. Данные отображения приведены, соответственно, ниже

$$\begin{cases} x_{i+1} = \mathrm{mod}(y_{i+1} + x_i, 2\partial) \\ y_{i+1} = \mathrm{mod}(y_i + K \cdot \sin(x_i), 2\partial) \end{cases}, \quad (2)$$

$$\begin{cases} x_{i+1} = \mathrm{mod}(x_i + a \cdot y_i, 1) \\ y_{i+1} = \mathrm{mod}(b \cdot x_i + (a \cdot b + 1) \cdot y_i, 1) \end{cases}, \quad (3)$$

$$\begin{cases} x_{i+1} = \mathrm{mod}(1 - a \cdot x_i^{\,2} - b \cdot y_i, 1) \\ y_{i+1} = \mathrm{mod}(x_i, 1) \end{cases}. \quad (4)$$

где $x_i$ и $y_i$ – переменные хаотического отображения на $i$-той итерации, $a$, $b$, $K$ – параметры отображения.

**Предлагаемый алгоритм хеширования**

В основу предлагаемого алгоритма хеширования положено использование двумерных дискретных хаотических отображений. Вследствие высокой чувствительности к начальным условиям, вычислительно трудно найти начальные условия отображения по текущим значениям переменных, а определить значения переменных на выбранной итерации оказывается возможным только путем осуществления всех необходимых итераций хаотического отображения. Это приводит к необратимости алгоритма хеширования при его построении с использованием хаотических отображений. Для злоумышленника оказывается вычислительно сложной задачей поиск двух сообщений с одинаковым хеш-значениями.

Предлагаемый алгоритм хеширования включает в себя следующие этапы.

1. Выбор значений переменных и параметров двумерных хаотических отображений.

На первом этапе алгоритма осуществляется выбор исходных значений переменных $x_i$ и параметров хаотических отображений, $i = 1, 2, k$, $k$ – количество переменных, зависящее от размера (в битах) формируемого хеш-значения. Значения переменных $x_i$ могут выбираться случайным либо псевдослучайным образом.

Стоит отметить, что хаотическое поведение наблюдается только при определенных значениях параметров у рассматриваемых отображений.

2. Реализация итераций хаотических отображений с добавлением элементов исходного сообщения.

Элементы исходного сообщения добавляются к значениям переменных хаотических отображений. Для этого исходное сообщение разбивается на блоки размером $m$ байт, где $m$ – размер формируемого хеш-значения в байтах. Каждый блок последовательно обрабатывается следующим образом.

Блок делится на фрагменты, размером по $p$ байт. Каждый $i$-ый фрагмент добавляется к значению переменной $x_i$. При этом $s$ байт фрагмента добавляются к $p$ байтам значения переменной с использованием операции «сложение по модулю 2». Выбор размера фрагментов $p$, а также номеров байтов переменной $x_i$, к которым добавляются байты фрагмента, зависит от вида представления чисел в памяти вычислительной машины. Например, в случае, когда для представления вещественного числа используется 8 байт, то рекомендуемое значения для $p$ – 4 байта, а номера байтов, к которым происходит добавления элементов, равны 2, 3, 4 и 5. После этого к полученным значениям $x_i$ применяется двумерное хаотическое отображение. При этом в качестве первой переменной выступает $x_i$, а в качестве второй переменной $x_{i+1}$. $i = 1, 2, k - 1$. После этого хаотическое отображение применяется к переменным $x_k$ и $x_1$ ($x_k$ выступает в качестве первой переменной, $x_1$ – в качестве второй). Это позволяет распространить изменения элементов на все значения переменных $x$ за меньшее количество итераций.

3. Реализация итераций хаотических отображений без добавления элементов исходного сообщения.

В случае, если был изменен последний байт сообщения, то после предыдущего этапа изменениям подвергнуться лишь 2 байта. Для распространения изменений на оставшиеся байты требуется проведение дополнительных итераций $q$. Данный этап аналогичен второму этапу за исключением того, что к значениям переменных $x_i$ не добавляются элементы исходного сообщения. Количество дополнительных итераций $q$ выбрано, равным 50.

4. Формирование хеш-значения.

В данной работе предлагается формировать два хеш-значения $h_1$ и $h_2$. При формирова-

нии второго хеш-значения $h_2$ на втором этапе проводится перестановка переменных $x$ следующим образом. Номер переменной $j$ при формировании значения $h_2$ равен $d_i$, где $i$ – номер переменной при формировании значения $h_1$, а $d$ – некоторая последовательность чисел. Последовательность чисел $d$ рекомендуется формировать таким образом, чтобы для каждого $x_j$ соседние переменные были отличными от случая формирования первого хеш-значения $h_1$. Примером последовательности $d$, содержащей 16 элементов, является следующая последовательность чисел: {2, 4, 6, 8, 10, 12, 14, 16, 5, 3, 1, 7, 9, 11, 13, 15}. Итоговое хеш-значение $h$ представляет собой результат выполнения операции «сложение по модулю 2», примененной к двум полученным хеш-значениям $h_1$ и $h_2$.

### Тестирование предлагаемого алгоритма хеширования

При тестировании предлагаемого алгоритма в данной работе определялось, характерен ли лавинный эффект для данного алгоритма. Проводилось также определение и сравнение статистических параметров двух последовательностей, первая из которых состоит из хеш-значений, а во-второй значения элементов получены случайным образом.

Лавинный эффект для алгоритмов хеширования заключается в том, что изменение значения одного бита в исходном сообщении приводит к изменению значений в среднем половины бит хеш-значения [4]. Тестирование на наличие лавинного эффекта проводилось следующим образом. Для сообщения $m_0$ размером 50 000 байт было вычислено хеш-значение $h_0$ размером 512 бит. После этого формировалось $N$ = 400 000 сообщений таким образом, что $i$-ое сообщение $m_i$ отличалось от исходного сообщения $m_0$ значением $i$-го бита, $i$ = 1, $N$. Для каждого сообщения $m_i$ вычислялось хеш-значение $h_i$. По полученным данным определялось количество бит $r$, значения которых отличались в хеш-значениях $h_0$ и $h_i$ (табл. 1). Для сравнения также приводятся результаты, полученные для алгоритма хеширования «Keccak». Указанный алгоритм используется для хеширования федеральным стандартом обработки информации США «SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions» [5].

Таблица 1. **Результаты тестирования по критерию «лавинный эффект»**

| | Предлагаемый алгоритм с использованием различных хаотических отображений | | | Алгоритм «Keccak» |
|---|---|---|---|---|
| | «Кот Арнольда» | Чирикова | Эно | |
| Среднее значение $r$ | 256.00 | 256.00 | 255.99 | 255.98 |
| Стандартное отклонение $r$ | 11.32 | 11.32 | 11.31 | 11.29 |
| Минимальное значение $r$ | 203 | 200 | 205 | 202 |
| Максимальное значение $r$ | 308 | 310 | 306 | 313 |

Установлено, что при использовании всех рассматриваемых хаотических отображений среднее количество изменившихся бит $r$ примерно равно 256 и составляет половину от общего количества бит хеш-значения, что соответствует условию критерия «лавинного эффекта». Минимальные и максимальные значения $r$ находятся в диапазоне от 200 до 310, что практически совпадает со значениями, полученными для алгоритма «Keccak». Значения стандартного отклонения $r$ при использовании алгоритма «Keccak» и предлагаемого алгоритма отличаются не более чем на 0.3%.

Статистические характеристики последовательности хеш-значений, полученных от неповторяющихся сообщений, должны быть схожи с характеристиками последовательности, значения которой получены случайным образом [6]. Для проверки данного свойства использовались статистические тесты Национального института стандартов и технологий США (National Institute of Standards and Technology, NIST) [7]. Данный набор тестов ориентирован на выявление различных дефектов случайностей во входной последовательности. Для 400 000 сообщений размером 256 байт были вычислены хеш-значения размером 64 байта. Полученные хеш-значения были записаны последовательно в файл, из которого было сформировано 200 последовательностей размером 1 000 000 бит каждая. После этого данные последовательности были подвергнуты тестированию с использованием пакета статистических тестов NIST.

Количество последовательностей, прошедших тестирование по каждому тесту представлены в табл. 2. Согласно рекомендациям NIST,

Таблица 2. **Количество последовательностей, прошедших тесты NIST**

| Хаотическое отображение | Номер теста | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Отображение «Кот Арнольда» | 198 | 198 | 198 | 200 | 200 | 198 | 199 | 195 |
| Отображение Чирикова | 199 | 195 | 199 | 197 | 200 | 200 | 198 | 194 |
| Отображение Эно | 198 | 194 | 198 | 197 | 199 | 199 | 199 | 194 |
| Хаотическое отображение | Номер теста | | | | | | | |
| | 9 | 10 | 11 | 12 | 13 | 14 | | 15 |
| Отображение «Кот Арнольда» | 200 | 198 | 197 | 121 из 124 | 122 из 124 | 196 | | 199 |
| Отображение Чирикова | 198 | 198 | 197 | 126 из 128 | 126 из 128 | 197 | | 199 |
| Отображение Эно | 198 | 199 | 199 | 121 из 122 | 120 из 122 | 198 | | 199 |

для того, чтобы тест считался успешно пройденным, необходимо, что бы минимальное количество последовательностей, прошедших тест составляло 193 последовательности из 200 последовательностей, 123 из 128, 121 из 126, 119 из 124, 117 из 122, 116 из 121 последовательности. Анализ показал, успешно пройденными являются все тесты NIST при использовании хаотических отображений «Кот Арнольда», Чирикова, Эно.

### Анализ времени формирования хеш-значения

Для проведения оценки вычислительной сложности проведен расчет количества операций $z$, необходимых для обработки одного блока сообщения размером 576 бит предлагаемого алгоритма и алгоритма «Keccak» (табл. 3).

Таблица 3. **Количество операций $z$, необходимое для обработки одного блока сообщения размером 576 бит**

| Алгоритм «Keccak» | | Предлагаемый алгоритм с использованием отображения «Кот Арнольда» | |
|---|---|---|---|
| Количество операций | Наименование операции | Количество операций | Наименование операции |
| 1752 | «сложение по модулю 2» | 36 | «сложение по модулю 2» |
| 384 | «логическое или» | 144 | «умножение» |
| 192 | «логическое и» | 72 | «сложение» |
| 192 | «логическое не» | 72 | «дробная часть числа» |
| 768 | «циклический сдвиг» | | |

Если принять количество тактов на выполнение логических операций, операции «сложение по модулю 2», операции «циклический сдвиг» равным 1, на выполнение операций «сложение» и «дробная часть числа» равным 5, а на выполнение операции «умножение» – 9 [8], то общее количество тактов, необходимое алгоритму «Keccak» равно 3288, а предлагаемому алгоритму – 2052. При использовании в предлагаемом алгоритме отображения Эно общее количество операций существенно не меняется. При использовании отображения Чирикова требуется проведение операций «синус».

Также осуществлен анализ времени формирования хеш-значений на компьютере с центральным процессором AMD A4–4300M APU с частотой 2.5 ГГц, с установленной 64-разрядной операционной системой Windows 7. Исходные размеры сообщений $s$ были выбраны равными $2^j$ байт, где $j = 7, 8, ..., 18$. Графики зависимости времени $t_h$ формирования хеш-значения от размера сообщения $s$ при использовании различных хаотических отображений представлены на рис. 1.

Как следует из полученных данных, наименьшее время формирования хеш-значения достигается при использовании отображения «Кот Арнольда», наибольшее – при использовании отображения «Чирикова». При использовании отображения Эно данный показатель превышает приблизительно на 20% показатель, полученный при использовании отображения «Кот Арнольда».

Для сообщений с размером менее 2 Кб, алгоритм «Keccak» превосходит по быстродействию предлагаемый алгоритм. Однако, для сообщений с размером большим, чем 4 Кб, при использовании отображений Эно и «Кот Арнольда» предлагаемый алгоритм справляется с задачей за меньшее время, чем «Keccak». Так, при размере файла, равном 256 Кбайт алгоритму «Keccak» требуется на 25% больше времени, чем предлагаемому алгоритму

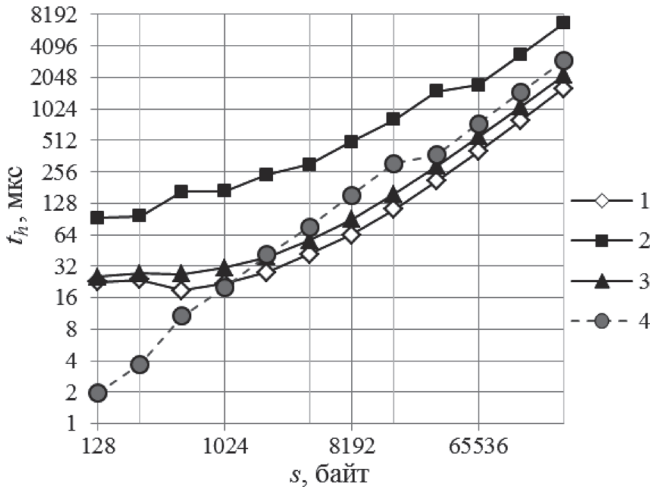


Рис. 1. Зависимость времени формирования хеш-значения $t_h$ от размера сообщения $s$: *1* – «Кот Арнольда», *2* – Чирикова, *3* – Эно; *4* – алгоритм «Keccak»

при использовании отображения Эно, и на 40% больше по сравнению со случаем использования отображения «Кот Арнольда». При использовании отображения Чирикова предлагаемый алгоритм формирует хеш-значение примерно в 2 раза медленнее, чем алгоритм «Keccak».

## Заключение

Предложен алгоритм хеширования на основе двумерных дискретных хаотических отображений. Благодаря использованию хаотических отображений, предлагаемый алгоритм является необратимым, а поиск двух сообщений с одинаковыми хеш-значениями становится вычислительно затруднительным.

Тестирование предлагаемого алгоритма показало, что для него характерным является наличие лавинного эффекта. Статистические характеристики последовательности, сформированной из хеш-значений, схожи со статистическими характеристиками последовательности, значения элементов которой получены случайным образом, что свидетельствует о работоспособности предлагаемого алгоритма.

Вычислительный эксперимент проведен с использованием отображений Чирикова, «Кот Арнольда» и Эно. Установлено, что для сообщений с размером, превышающим 4 Кб, при использовании отображений Эно и «Кота Арнольда» предлагаемый алгоритм справляется с задачей более чем на 20% быстрее, чем алгоритм «Keccak».

Рассмотренный алгоритм хеширования может быть использован при решении задач контроля целостности данных при передаче информации в современных телекоммуникационных системах.

## Литература

1. **Криптология**: учебник / Ю. С. Харин [и др.]. – Минск: БГУ, 2013. – 511 с.

2. **Sobti, R.** Cryptographic hash functions: a review / R. Sobti, G. Geetha // International journal of computer science issues. – 2012. – Vol. 2, № 2. – P. 461–479.

3. **Птицын, Н.** Приложение теории детерминированного хаоса в криптографии / Н. Птицын. – М: МГТУ им. Н. Э. Баумана – 2002. – 80 с.

4. **One-Way hash function based on cascade chaos / F. Xiang** [et al.] // The open cybernetics & systemics journal. – 2015. – Vol. 9. – P. 573–580.

5. **SHA-3 Standard**: Permutation-Based Hash and Extendable-Output Functions: FIPS 202. – Publ. 2015-08-01. – Gaithersburg: National Institute of Standards and Technology, 2015. – 29 p.

6. **Мао, В**. Современная криптография: теория и практика / В. Мао (под ред. Клюшиной Д. А.). – М: издательский дом Вильямс – 2005. – 768 с.

7. **On the interpretation of results from the NIST statistical test suite / M. Sys** [et al.] // Romanian Journal of information science and technology. – 2015. – Vol. 18, № 1. – P. 18–32.

8. **Fog, A.** Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs [Electronic resource] / A. Fog. – Technical University of Denmark, 2016. – Mode of access: http://www.agner.org/optimize/instruction_tables.pdf. – Date of access: 27.03.2017.

## References

1. **Cryptology**: textbook / Yu. S. Kharin [et al.]. – Minsk: BSU, 2013. – 511 p.

2. **Sobti, R.** Cryptographic hash functions: a review / R. Sobti, G. Geetha // International journal of computer science issues. – 2012. – Vol. 2, № 2. – P. 461–479.

3. **Pticyn, N**. Deterministic chaos theory application to cryptography / N. Pticyn. – Moscow: Bauman MSTU – 2002. – 80 p.

4. **One-Way hash function based on cascade chaos / F. Xiang** [et al.] // The open cybernetics & systemics journal. – 2015. – Vol. 9. – P. 573–580.

5. **SHA-3 Standard**: Permutation-Based Hash and Extendable-Output Functions: FIPS 202. – Publ. 2015-08-01. – Gaithersburg: National Institute of Standards and Technology, 2015. – 29 p.

6. **Mao, W**. Modern Cryptography: Theory and Practice / W. Mao (edited by Kljushina D. A.). – Moscow: Williams Publishing House. – 2005. – 768 p.

7. **On the interpretation of results from the NIST statistical test suite / M. Sys** [et al.] // Romanian journal of information science and technology. – 2015. – Vol. 18, № 1. – P. 18–32.

8. **Fog, A.** Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs [Electronic resource] / A. Fog. – Technical University of Denmark, 2016. – Mode of access: http://www.agner.org/optimize/instruction_tables.pdf. – Date of access: 27.03.2017.



| *Поступила* | *После доработки* | *Принята к печати* |
|---|---|---|
| *04.03.2017* | *07.04.2017* | *10.06.2017* |


*Sidorenko A. V., Shakinko I. V.*

# HASHING ALGORITHM BASED ON TWO-DIMENSIONAL CHAOTIC MAPPINGS

*Belarusian State University*

*A new hashing algorithm based on dynamic chaos is proposed. Owing to the use of chaotic mappings, this algorithm is irreversible and a search for two messages with identical hash-values becomes computationally difficult. The proposed algorithm consists of the stages: selection of the variables and of the parameters of two-dimensional chaotic mappings; realization of iterations of the chaotic mappings with the addition of the original-message elements to the variables; realization of iterations of the chaotic mappings without the addition of the original-message elements to the variables; the hash-value formation. The formation of the two hash-values $h_1$ and $h_2$ realized with different orders of the variables. The resultant hash-value is obtained by the modulo-2 addition operation applied to the hash-values $h_1$ and $h_2$. The proposed algorithm has been tested. It has been found that this algorithm is characterized by the avalanche effect. The statistical characteristics of the sequence formed of hash-values are identical to those of the sequence with the randomly obtained values of the elements, pointing to the adequate performance of this algorithm. The computational experiment has been realized using the Chirikov, «Arnold's cat» and Henon maps. It is demonstrated that, with the use of Henon and «Arnold's cat» maps for the messages exceeding 4 KB, the proposed algorithm outperforms «Keccak» algorithm, being faster by 20% and more.*

*The proposed hashing algorithm may be used in solving the problems of data integrity in modern telecommunication systems.*

***Keywords:*** *dynamic chaos, chaotic mapping, hashing, data integrity, information security.*

**Сидоренко Алевтина Васильевна.** Профессор кафедры физики и аэрокосмических технологий. Научные интересы: защита информации в телекоммуникационных системах, динамический хаос, теоретическая информатика, радиофизика, биофизика.

**Sidorenko Alevtina Vasilevna** – professor, doctor of technical sciencies. Department of Radiophysics and Computer technologies. Belarusian State University. Scientific interests: information security, dynamic chaos, theoretic informatics, radiophysics, biophysics. E-mal: sidorenkoa@yandex.ru

**Шакинко Иван Владимирович.** Аспирант кафедры телекоммуникаций и информационных технологий БГУ. Научные интересы: защита информации в телекоммуникационных системах, динамический хаос и его применение.

**Shakinko Ivan Vladimirovich** – post graduate. Scientific interests: information security in telecommunication systems, dynamic chaos and it's application.

Данная работа выполняется в рамках ГПНИ «Информатика, космос и безопасность».