УДК 004.021

Тільняк Ю.Я.
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

МОДИФІКАЦІЯ АЛГОРИТМУ MD5 ДЛЯ КОНТРОЛЮ ЦІЛІСНОСТІ ЕЛЕКТРОННОГО ДОКУМЕНТООБІГУ ДЛЯ БЛОКЧЕЙН

У статті розглянуто спосіб застосування дискретних відображень для генерації детерміновано хаотичних числових рядів із метою підвищення криптостійкості хеш-функцій для блокчейн. Проведено порівняльний аналіз розробленої хеш-функції й аналога у вигляді методу MD5. За результатами аналізу встановлено, що хеш-функція є стійкою до інформаційних атак.

Ключові слова: *блокчейн, захист інформації, хеш-функція, алгоритм MD5, електронний контроль документообігу.*

Постановка проблеми. Сучасний рівень розвитку інформаційних технологій характеризується тенденцією домінування електронних документів над традиційними паперовими носіями інформації. Сьогодні важливо приділити особливу увагу захисту інформації, що в них міститься. Потреба введення електронного документообігу є невід'ємною частиною для блокчейн, через це виникає проблема достовірності електронних документів. Фахівці у галузі керування й обміну електронними документами зацікавлені у вирішенні проблем, пов'язаних із запобіганням і несвоєчасним виявленням несанкціонованих змін у змісті електронних документів або комунікацій в умовах зростання кількості комп'ютерних злочинів.

Аналіз останніх досліджень і публікацій. Традиційно для безпеки електронної інформації застосовують хеш-функції (hash-function). Нині популярними функціями хеш є MD5 і SHA-1. У лютому 2005 р. було опубліковано перший звіт про потенційну вразливість хеш-функції, що вперше виявлено в алгоритмі SHA-1. Насамперед цю проблему розглянув Національний інститут стандартів і технологій (NIST), видавши специфікації SHA-1 як FIPS 180-2 – офіційний федеральний стандарт США для обробки інформації. Новий захищений алгоритм для розробки даних перебуває на стадії тестування та перевірки (NIST 2012) [1]. Зважаючи, що SHA-1 є незадовільною з погляду практики застосування на сучасному рівні методів криптоаналізу та реалізації інформаційних атак, у дослідженні безпеки електронної інформації для блокчейн ми розглянемо модифікацію MD5.

Постановка завдання. Метою дослідження є застосування алгоритму MD5 для контролю цілісності електронного документообігу для блокчейн.

Виклад основного матеріалу дослідження. Розглянемо прототип SecureHashAlgorithm, який був розроблений Національним інститутом стандартів і технологій (NIST) і опублікований як федеральний інформаційний стандарт (FIPS PUB 180) у 1993 р. SHA-1, а також MD5 на алгоритмі MD4 [1; 2].

Розглянемо алгоритм виконання SHA-1. На вхід надсилається максимальна довжина 264 біт. На виході створюється дайджест повідомлення 160 біт.

Робота алгоритму SHA-1 заснована на виконанні такої послідовності дій. Повідомлення доповнюється таким чином, що його довжина є кратною 448 по модулю – 512. Кількість доданих бітів знаходиться у діапазоні від 1 до 512. Додавання є однією одиницею з необхідною кількістю нулів.

Потім до повідомлення додається блок із 64 біт. Це 64-бітове ціле число без знака відображає довжину вихідного повідомлення.

На цьому етапі генерується повідомлення, довжина якого перевищує 512 біт. Збільшене повідомлення становить послідовність 512-бітових блоків довжини L, тобто загальна довжина розширеного повідомлення $L \times 512$ біт і кратна шістнадцяти 32-бітовим словам.

Для ініціалізації SHA-1 використовується 160-бітовий буфер для зберігання проміжних і кінцевих результатів хеш-функції. Буфер може бути представлений як п'ять 32-бітових регістрів *A, B, C, D* і *E*. Ці регістри ініціалізуються такими шістнадцятковими числами:

A = 67452301
B = EFCDAB89
C = 98BADCFE
D = 10325476
E = C3D2E1F0

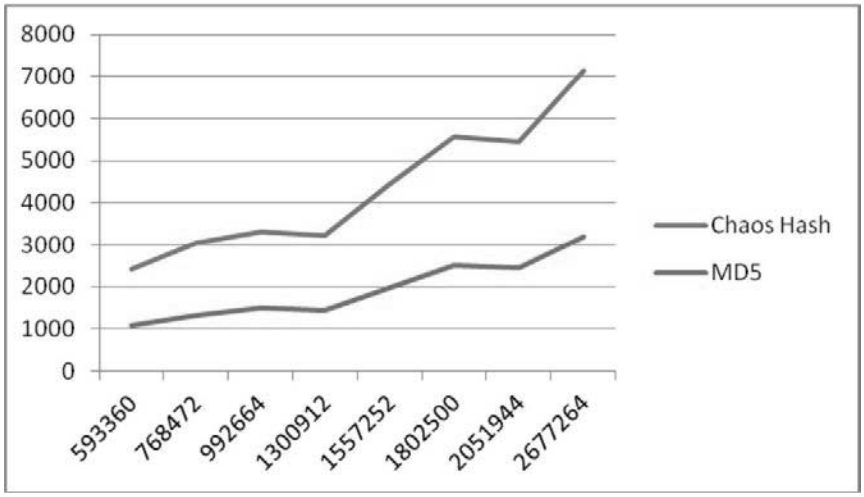

Рис. 1. Графік порівнянь результатів роботи MD5 і даних алгоритму

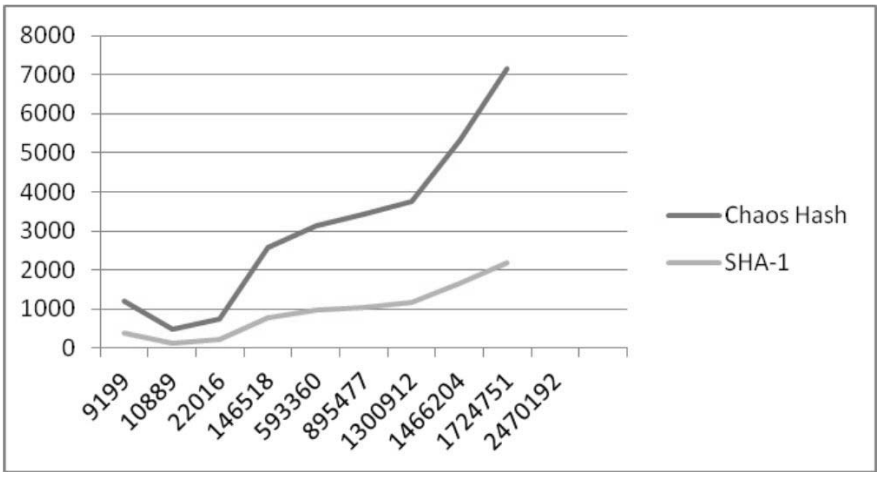

Рис. 2. Графік порівнянь показників продуктивності SHA-1 і розробленого алгоритму

Після того виконується обробка у вигляді 512-бітових блоків.

В основі алгоритму лежить модуль, що складається з 80 циклічних обробок, позначених як H_{SHA}. Всі 80 циклів мають однакову структуру.

Кожен цикл приймає поточний оброблений 512-бітний блок і 160-бітове значення буфера ABCDE і змінює вміст цього буфера.

У кожному циклі використовується додаткова константа K^t, яка набуває тільки чотирьох різних значень:

0 <= t <= 19 K^t = 5A827999
(ціле число [230 × 21/2]);
20 <= t <= 39 K^t = 6ED9EBA1
(ціла частина числа [230 × 31/2]);
40 <= t <= 59 K^t = 8F1BBCDC
(ціле число [230 × 51/2]);
60 <= t <= 79 K^t = CA62C1D6
(ціла частина числа [230 × $10^{1/2}$]) [2].

Щоб отримати хеш наступну частину повідомлення, вихід 80-го циклу є хеш-значенням попереднього. Модуль 2^{32} додає незалежно для кожного з п'яти слів у буфері з кожним із відповідних слів у попередньому хеші.

Після обробки всіх 512-бітових блоків вихід L-етапу є дайджестом 160-бітового повідомлення.

Проведемо покрокову реалізацію алгоритму формування хеш-функції з використанням детермінованої хаотичної послідовності чисел. Генеруючи дискретну хаотичну послідовність і регулюючи її дані, отримуємо дайджест, що відповідає цим даним.

Як генератор детерміновано хаотичної послідовності приймається дискретна карта Лоренца:

$$x + 1 = 1 - 2 \times | xi |.$$

Всі операції виконуються над числами з рухомою комою з подвійною точністю.

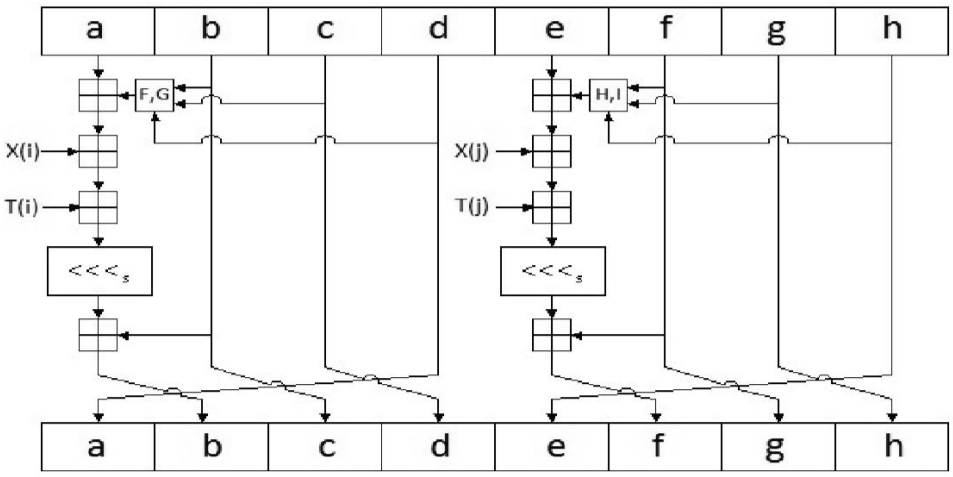

Рис 3. Блок-схема зміненого алгоритму MD5

Крок 1. Присвоюємо значення ініціалізації (початкового) хаотичної послідовності кількості, отриманого з такої операції:

xi = 0xFFFFFFFFFFFFFFFF / 0x10000000000000.

Крок 2. Створюємо значення хаотичної послідовності.

Крок 3. Беремо наступний байт із даних і ділимо його значення на 1 000.

Крок 4. Ми виводимо отримане значення зі значення визначеної детермінованої послідовності; якщо згенероване значення хаотичної послідовності більше нуля, тоді додаємо.

Крок 5. Ми робимо 10 послідовних поколінь хаотичної послідовності без змішування.

Крок 6. Якщо всі дані передані, то дайджест є знаком і мантією останнього елемента хаотичної послідовності. В іншому разі він повертається до кроку 2.

Для тесту продуктивності програми було створено файл інциденту, що записується для тесту продуктивності, а також розмір отриманого файлу і час оброблення алгоритмом файлу (рис. 1, 2).

На рис. 1, 2 видно: на горизонтальній осі – кількість даних у байтах, а на вертикальній – час обробки цього обсягу в мільйонах.

Як видно з графіків, використовуються дискретні індикатори, що призводить до детермінованих і хаотичних процесів для створення функцій, які заважають індивідам, здійснюють інформаційний напад, вибирають електронні файли і здійснюють контроль. Непередбачуваність поведінки хаотичної системи і велика кількість дискретних відображень забезпечують основні вимоги, попередні й ефективні функції [1–2]:

1. необоротність, що виникає з властивостей дискретного відображення Лоренца;

2. опір до зіткнення першого роду: для повідомлення завдання *M* неможливо вибрати інше повідомлення *N*, для якого *H (N) = H (M)*, що забезпечує сильну залежність дискретної дисплейної орбіти від великих об'єктів, визначених значенням кожного файлу електронного документа;

3. стійкість до зіткнення другого роду: практична нездатність подолати пару повідомлень, які мають той самий дайджест, що і результат хешування.

Сьогодні питання дослідження шляхів подальшого зниження вразливості криптографічних хеш-функцій для різних додатків актуальні, оскільки за постійного зростання потужності та швидкодії сучасних обчислювальних засобів ймовірність успішної атаки також зростає. Алгоритми хешування, які використовуються у сучасних стандартах на електронний підпис в інших суміжних додатках [1–3], мають досить високу стійкість до колізій, однак ускладнення алгоритму часто призводить до збільшення обчислювальної складності і витрат на його реалізацію. Тому відносно прості хеш-алгоритми, використовувані в додатках, які не потребують високого рівня стійкості, можуть залишатися практично корисними. Наприклад, хеш-функції MD5 і SHA-1 досі продовжують використовуватися у деяких практичних додатках, незважаючи на виявлені вразливості, оскільки володіють високою швидкодією. Таким чином, дослідження можливості модифікації таких криптографічних протоколів із метою підвищення їх стійкості продовжує викликати не тільки науковий, але і практичний інтерес [3–5].

У роботі пропонується варіант модифікації алгоритму MD5. Якщо в наявному протоколі вихідна хеш-сума становить 128 біт, то пропонований алгоритм хешує текст довільної довжини, перетворюючи його в 256-бітну послідовність. На рис. 3 показана блок-схема модифікованого алгоритму.

Під час тестування програмного коду було показано, що модифікований алгоритм має той самий порядок обчислювальної складності, що і стандартний, тобто час обчислення хеш-суми одного і того ж тексту за допомогою обох алгоритмів практично не відрізняється (табл. 1). Це є важливим фактором, оскільки основною перевагою хеш-функцій такого виду і є швидкодія.

Таблиця 1

Час обчислення хешу

Кількість вхідних даних (кількість символів)	Час обчислення стандартного MD5 на комп'ютері з частотою 2 Ггц	Час обчислення модифікованого MD5 на комп'ютері з частотою 2 ГГц
10	0.038	0.041
50	0.045	0.044

Також хеш-функція повинна задовольняти таку умову, що за зміни одного біта досягається «лавинний ефект», тобто хеш істотно відрізняється для хешування вхідних даних. Приклад порівняння таких повідомлень і відповідних згорток наведено в табл. 2.

Таблиця 2

Вихідні дані модифікованого MD5

Вхідний текст	Отриманий хеш
Hello	cbd02904ff8ad1f3dee3c92bc2418c2f7b21 4cc3769299558fb5a423f74b2bb2
Iello	e39b1e6cd4f6e7e18103e0cae8eb667ec4a 3bdc4970887048e6c45209d4814cb

У підвищенні стійкості MD5 після процедури модифікації можна переконатися, розглянувши основні атаки на хеш-функції і порівнявши основний і вдосконалений алгоритми. Атака «днів народження» [4] – один із методів пошуку колізій хеш-функцій на основі парадоксу днів народження. Для заданої криптографічної хеш-функції f є метою атаки для пошуку колізії другого роду. Для цього обчислюються значення f для випадково вибраних блоків вхідних даних доти, поки не будуть знайдені два блоки, що мають однаковий хеш. Таким чином, якщо f має N різних рівноймовірних вихідних значень і N є досить великим, то з парадоксу днів народження випливає, що в середньому після перебору $1,25 \times \sqrt{N}$ різних вхідних значень буде знайдена шукана колізія. Якщо ж хеш-функція генерує n-бітне значення, то число випадкових вхідних даних, для яких хеш-коди з великою ймовірністю дадуть колізію, однаково не 2n, а тільки близько $2^{\frac{n}{2}}$

Таким чином, для MD5 різної розмірності хешу існує різна кількість вхідних даних для порівнянної ймовірності колізії (табл. 3).

Атака «грубою силою» [5] може бути виконана для знаходження першого прообразу за заданим хеш-значенням або для знаходження другого прообразу, що дає таке ж хеш-значення, як і сигнальне повідомлення. Суть атаки полягає в послідовному або випадковому переборі вхідних повідомлень і порівнянні результату виконання хеш-функції з її заданим значенням [4–7]. Складність такої атаки оцінюється 2^{n-1} операцій обчислення хеш-значення, де n – довжина хеш-значення. З табл. 3 показано, що модифікований алгоритм вимагає в $3,5 \times 1038$ разів більше операцій, ніж стандартний [8].

Висновки. Запропонований варіант модифікації алгоритму MD5 дозволяє за швидкодії знизити ймовірність колізії як першого, так і другого роду. Використання таких алгоритмів, наприклад, у додатках прискореного пошуку даних у великих масивах, де колізія призводить не до порушення цілісності повідомлення, а всього лише до помилки пошуку, може значно підвищити ефективність роботи таких систем за рахунок зниження ймовірності такої помилки. Застосування модифікації такого алгоритму у введенні електронного документообігу дозволить збільшити рівень безпеки у середовищі комунікації для блокчейн зокрема.

Таблиця 3

Імовірність колізії

Розмірність хеш (n)	Кількість рівноможливих вихідних значень (2n)	Кількість вхідних даних, за яких колізія буде із заданою вірогідністю				
		10-6%	0,1%	1%	25%	50%
128	3.4×1038	2.6×1016	8.3×1017	2.6×1018	1.4×1019	2.2×1019
256	1.2×1077	4.8×1035	1.5×1037	4.8×1037	2.6×1038	4.0×1038

Список літератури:

1. Wang X., Yin Y.L., Yu H. Finding collisions in the full SHA-1. Advances in cryptology – CRYPTO 2005. *Lecture Notes in Computer Science.* Berlin: Springer-Verlag, 2005. V. 3621. P. 17–36.
2. Довгаль В.М., Гордиенко В.В., Никитин М.О. Алгоритм на основе механизмов хаотической динамики для контроля целостности электронных документов. *Ученые записки. Электронный научный журнал.* Курск, 2012. С. 29–32.
3. ГОСТ Р 34.11-2012. Информационная технология. Криптографическая защита информации. Функция хеширования.
4. Халимов Г.З. Универсальное хеширование по максимальной кривой третьего рода. *Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика.* 2011. № 1 (96). Вып. 17/1. С. 137–145.
5. Халимов О.Г., Буханцов А.Д., Халимов Г.З. Построение кривых Гурвица для универсального хеширования. *Научные ведомости БелГУ. Сер. История. Политология. Экономика. Информатика.* 2014. № 1 (172). Вып. 29/1. С. 153–160.
6. Ohta K., Koyama K. Meet-in-the-Middle Attack on Digital Signature Schemes. *Abstract of AUSCRYPT'90*. 1990. P. 110–121.
7. ANSI X9.30 (PART 2), American National Standard for Financial Services – Public key cryptography using irreversible algorithms for the financial services industry – Part 2: The secure hash algorithm (SHA), ASC X9 Secretariat – American Bankers Association, 1993.
8. Буханцов А.Д., Дружкова И.В. О модификации алгоритма MD5. *Научные ведомости. Серия: Экономика. Информатика.* Белгород, 2016. № 2 (223). Вып. 37. С. 174–177.

МОДИФИКАЦИЯ АЛГОРИТМА MD5 ДЛЯ КОНТРОЛЯ ЦЕЛОСТНОСТИ ЭЛЕКТРОННОГО ДОКУМЕНТООБОРОТА ДЛЯ БЛОКЧЕЙН

В статье рассмотрен способ применения дискретных отражений для генерации детерминировано хаотических числовых рядов с целью повышения криптоустойчивости хэш-функций для блокчейн. Проведен сравнительный анализ разработанной хэш-функции и аналога в виде метода MD5. В результате анализа установлено, что хэш-функция является устойчивой к информационным атакам.

Ключевые слова: *блокчейн, защита информации, хэш-функция, алгоритм MD5, электронный контроль документооборота.*

MODIFICATION OF MD5 ALGORITHM FOR CONTROL OF THE INTEGRITY OF ELECTRONIC DOCUMENTARY CLEANING FOR BLOCKCHAIN

In the article the method of using discrete maps for generation of deterministically chaotic numerical series is considered in order to increase the cryptostability of hash functions for blockchain. A comparative analysis of the developed hash function and analog in the form of the MD5 method has been carried out. According to the results of the analysis, it has been established that the hash function is resistant to information attacks.

Key words: *blockchain, information protection, hash function, MD5 algorithm, electronic control of document circulation.*