УДК 004.056

В. В. Казміревський
Н. Р. Кондратенко

МЕТОД ОЦІНКИ СТАТИСТИЧНИХ ВЛАСТИВОСТЕЙ У ДОСЛІДЖЕННІ СТІЙКОСТІ ДО ДИФЕРЕНЦІЙНОГО КРИПТОАНАЛІЗУ ФУНКЦІЙ ГЕШУВАННЯ

Вінницький національний технічний університет

Анотація

Досліджено коректність та стійкість розробки криптографічних геш-функцій мовою Dart. В результаті аналізу методів дослідження функцій гешування обрано критерії перевірки: коректність реалізації, статистичні властивості та стійкість до диференційного криптоаналізу. Перевірено низку безключових геш-функцій, зокрема, і легких криптографічних геш-функцій.

Ключові слова: *геш-функція, легкі геш-функції, криптоаналіз, прообраз, алгоритм, диференційний криптоаналіз, статистичні властивості, лавинний ефект, x^2 критерій.*

Abstract

The correctness and stability of the development of cryptographic hash functions in the Dart language have been studied. As a result of the analysis of hashing function research methods, verification criteria were selected: implementation correctness, statistical properties, and resistance to differential cryptanalysis. A number of keyless hash functions, in particular, and light cryptographic hash functions have been tested.

Keywords: *hash function, easy hash functions, cryptanalysis, prototype, algorithm, differential cryptanalysis, statistical properties, avalanche effect, x^2 criterion.*

Вступ

Сьогодні великої популярності набули легкі криптографічні геш-функції, зокрема для використання в мобільних пристроях. Аналіз відповідних інформаційних джерел показав, що легкі криптографічні функції гешування, попри значну кількість тестів, є все ще не достатньо дослідженими, оскільки, швидко замінюють одне одного в першості використання у програмних продуктах. Часто не всі показники продуктивності легких криптографічних геш-функцій є протестованими. Також більшість із досліджень не надають результатів, що підтверджують криптографічні властивості даних алгоритмів гешування. Тому, дослідження статистичних властивостей для відомих та нових криптоалгоритмів є дуже актуальним, так як вчасно можна сказати, що даний криптоалгоритм є нестійким, і що його потрібно вдосконалити або замінити новим. Метою даного дослідження є покращення стійкості програмних бібліотек за рахунок криптоаналізу геш-функція, які вони реалізовують [1].

Для досягнення мети було розв’язано такі задачі:

- проаналізовано поняття легких криптографічних геш-функції;
- проаналізовано методи диференційного криптоаналізу щодо їх застосування для геш-функцій та запропоновано власний метод;
- запропоновано спосіб перевірки статистичних властивостей функцій гешування;
- розроблено засіб, який складається з набору тестів для аналізу стійкості та статистичних властивостей геш-функцій;
- проаналізовано результати тестування.

Криптографічні легкі геш-функції

Найпопулярнішими алгоритмами гешування на сьогоднішній день є такі: SHA-2, SHA-3, RIPEMD-160. Також великої популярності набули легкі криптографічні геш-функції, зокрема для використання в мобільних пристроях. Однак, на відміну від вище згаданих алгоритмів гешування, дані геш-функції дуже швидко замінюють один одного, з’являються нові алгоритми, які можуть суттєво відрізнятися один від одного. Такі функції гешування потребують якісного тестування. Було проведено багато досліджень щодо прогресу досліджень безпеки Інтернету речей (IoT), до яких відносять і мобільні пристрої, опублікованих в останні роки. Дослідники в основному зосереджені на рішеннях проблем безпеки IoT. Проблеми безпеки загалом представлені як компоненти кожного дослідження та

розглядаються як загальні поняття, а безпека та конфіденційність часто розглядаються разом як одне поняття. На жаль, немає достатньо детальних та актуальних досліджень, що описують глибокі проблеми безпеки мобільних пристроїв, пов'язані з легкими криптографічним геш-функціями [2]. Дизайн легких криптографічних геш-функцій також побудований на трьох популярних конструкціях: конструкції Merkle-Damgard, губчатій конструкції та на конструкції на основі блочного шифру. Деякі алгоритми використовують змішану конструкцію, наприклад Merkle-Damgard або губчата, як основну конструкцію та інші конструкції (наприклад, на основі блочного шифру) як будівельні блоки для розробки функцій стиснення або перестановок [2]. До легких криптографічних геш-функцій відносять алгоритми сімейства ARMADILLO побудованих на основі конструкції Merkle-Damgard з різними варіантами довжини вихідного гешу, а саме: 80 біт, 128 біт, 160 біт, 192 біт та 256 біт. Першою геш-функцією на основі легкої губчастої конструкції була QUARK. Алгоритм був розроблений на основі потокового шифру Grain і блокового шифру KATAN. Для перестановки бітів використовувалися два регістри зсуву з нелінійним зворотним зв'язком і регістр зсуву з лінійним зворотним зв'язком. Ще одним прикладом легкої функції, побудованій на губчастій конструкції, є геш-функція SPONGENT. SPONGENT було розроблено як сімейство геш- функцій із 88, 128, 160, 224, 256-бітним геш-значенням. Автори даного алгоритму стверджують, що алгоритм стійкий до атак, спрямованих на геш-функції. Також до алгоритмів, побудованих на губчатій конструкції, відносять такі: PHOTON, GLUON, SPN-Hash, SipHash, LHash, Neeva-hash, Hash-One, Gimli- Hash, sLiSCP-hash, XOODYAK, ASCON-HASH, KNOT-Hash, DryGascon-Hash, PHOTON-Beetle-Hash, HVH та багато інших. DM-PRESENT є легкою геш-функцію, яка використовує блоковий шифр PRESENT і конструкцію DaviesMeyer. Є два типи DM-PRESENT геш-функції: DM-PRESENT-80 і DM-PRESENT-128. TWISH розроблено на базі алгоритму TWINE-128 блокового шифрування і використовує конструкцію DM. TWISH – це геш-функція довжини одного блоку, яка приймає 128-бітне вхідне повідомлення та повертає 64-бітове геш-значення. Введення повідомлення в схемі DM діє як ключ [2].

Отже, легких криптографічних геш-функцій є дуже багато з різноманітними конструкціями побудови. А їх аналіз та огляд відповідних літературних джерел показав, що легкі криптографічні функції гешування, попри значну кількість тестів, є все ще не достатньо дослідженими, оскільки, дуже швидко замінюють одне одного. Часто не всі показники продуктивності легких криптографічних геш-функцій є протестованими. Також більшість із досліджень не надають підтверджуючих результатів криптографічних властивостей даних алгоритмів гешування.

Диференційний криптоаналіз

Диференціальний криптоаналіз, як правило, є атакою на вибраний відкритий текст, що означає, що зловмисник повинен мати можливість отримати зашифровані тексти для певного набору відкритих текстів на свій вибір [1]. Статистична інформація про ключ виводиться з блоків зашифрованого тексту, отриманих шляхом шифрування пар блоків відкритого тексту з певною побітовою різницею A' під цільовим ключем. Коефіцієнт роботи атаки критично залежить від найбільшої ймовірності $P(B' \mid A')$ з різницею B' на деякому фіксованому проміжному етапі криптографічної функції, наприклад, на вході останнього раунду. У першому наближенні ймовірності $P(B' \mid A')$ для DES вважаються незалежними від конкретного значення ключа [3-5]. Атака є успішною, якщо правильне значення підключа пропонується значно частіше, ніж будь-яке інше значення. Для DES неправильні пропозиції можна вважати рівномірно розподіленими серед можливих ключових значень, якщо значення $P(B' \mid A')$ значно більше, ніж $P(C' \mid A')$ для будь-якого $C' \neq B'$ [4-7]. За даних умов є сенс розрахувати співвідношення між кількістю пропозицій правильного значення та середньою кількістю пропозицій на запис, тобто відношення сигнал/шум. Якщо розмір таблиці дорівнює 2^l, а середня кількість запропонованих підключів на пару дорівнює γ, це співвідношення дорівнює $P(B' \mid A')2^l / S/N$. Відношення сигнал/шум сильно впливає на кількість правих пар, необхідних для однозначної ідентифікації правильного значення підключа [5-9].

Для методу перебору бітів пропонується використовувати коди Грея. Коди Грея є таким порядком двійкової системи числення, що два послідовних значення відрізняються лише одним бітом. Зокрема, коди Грея широко використовують для запобігання помилковим виходам від електромеханічних перемикачів і для полегшення виправлення помилок у цифрових комунікаціях.

Тому варто було б застосувати коди Грея і для дослідження геш-функцій. Зокрема, пропонується в подальшому використовувати також Грея подібні коди для диференційного криптоаналізу алгоритмів гешування, таких як:

- збалансований код Грея;
- довгострокові коди Грея;
- монотонні коди Грея;
- одноколінний код Грея;
- код Беккета-Грея.

Спосіб статистичної перевірки диференційної ентропії

Для проведення статистичного дослідження диференційної ентропії функції гешування H(m) вхідної послідовності $m(M_1, \ldots, M_i)$ необхідно спостерігати за змінами вихідної послідовності внаслідок зміни всіх бітів за допогомою кодів Грея. Для досягнення мети необхідно виконати гешування m випадкових наборів M_i, i = 1 … n. Для всіх наборів M_i необхідно підрахувати значення функції на даному наборі - $m(M_i)$, та значень функції на наборах, у яких виконано зміну одного із бітів послідовності - $m(M_{i1})$, $m(M_{i2})$, … , $m(M_{ij})$. Потім стає можливим підрахувати статистичну частоту G_i таким чином:

$$G_j = (\Sigma^n_{i=1} m(M_j) + m(M_{ij})) / n$$

Припущення стосовно того, що $m(M_1, \ldots, M_n)$ задовольняє умови диференційної ентропії по і-тій змінній полягає в тому, що внаслідок зміни біта вхідної послідовності геш-функції значення вихідної послідовності змінюється з імовірністю ½.

Порядок тестування окремої двійкової послідовності M_i має наступний вид:

- висувається нульова гіпотеза H_0 – припущення про те, що дана двійкова послідовність M_i є випадковою;
- за послідовністю M_i розраховується статистика тесту $c(M_i)$;
- з використанням спеціальної функції та статистики тесту розраховується значення ймовірності $P = f(c(M_i))$, $P \in [0; 1]$.;

значення ймовірності P порівнюється з пороговим значенням $\alpha \in [0{,}001; 0{,}01]$. Якщо $P \geq \alpha$, то гіпотеза H_0 приймається, тобто, тест пройдено. В іншому випадку приймається альтернативна гіпотеза [4]. Для «якісних» послідовностей ймовірність такої події α вкрай мала (~ 0,001).

Даний спосіб статистичної оцінки нелінійності та відповідності умови диференційної ентропії функцій гешування дозволяє робити довільний вибір зміни бітів вхідної послідовності та значень вихідної послідовності, які їм відповідають і входять до статистичної вибірки, що дозволяє зменшити витрати на обчислювальні розрахунки та збільшити функціональні можливості.

Аналіз статистичних властивостей геш-функцій

Для дослідження пропонується взяти наступні алгоритми гешування: геш- функції сімейства SHA-2, SHA-3, легкі геш-функції сімейства ARMADILO-80, які побудовані на конструкції Merkle-Damgard, H-PRESENT-128, TWISH-128, які побудовані на основі блокових шифрів, PHOTON-80, PHOTON-128, PHOTON- 160, PHOTON-224, PHOTON-256, QUARK-136, QUARK-176, QUARK-256, які побудовані на губчастій конструкції, та власна розроблена геш-функція, яка генеруватиме звичайну послідовність 0 та 1, розроблених на мові програмування Dart.

При проведенні статистичних досліджень кожен алгоритм формує 100 послідовностей по 10^8 байт, тобто, розмір статистичної вибірки кожного алгоритму досягає 10^{10} байт. Кожен тест для кожної з 100 послідовностей вважається незалежним спостереженням, тобто на кожну геш-функцію припадатиме 100 тестів. У таблиці 1 наведені результати статистичних тестів для кожного досліджуваного алгоритму.

Таблиця 1 – Результати статистичного тестування алгоритмів гешування

№	Назва геш-функції	Кількість пройдених тестів	№	Назва геш-функції	Кількість пройдених тестів
1	SHA-224	70	13	ARMADILO-256	62
2	SHA-256	68	14	H-PRESENT-128	51
3	SHA-384	78	15	TWISH-128	51
4	SHA-512	85	16	PHOTON-80	37
5	SHA3-224	71	17	PHOTON-128	48
6	SHA3-256	79	18	PHOTON-160	43
7	SHA3-384	84	19	PHOTON-224	59
8	SHA3-512	86	20	PHOTON-256	67
9	ARMADILO-80	36	21	QUARK-136	41
10	ARMADILO-128	45	22	QUARK-176	57
11	ARMADILO-160	41	23	QUARK-256	62
12	ARMADILO-192	48	24	Власна геш-функція	5

Відповідно до таблиці 1 для кожної геш-функції тести запускалися зі 100 різними згенерованими послідовностями. Отже, таблиця 1 відображає скільки разів із 100 той чи інший алгоритм гешування пройшов певний тест.

Отримані результати у таблиці 1 можна інтерпретувати таким чином:

- 50 і менше – геш-функція має проблеми з реалізацією та стійкістю;
- 51-70 – геш-функція доволі стійка та може мати дещо некоректну реалізацію;
- 70 і більше – геш-функція стійка та коректно реалізована.

Результати тестування алгоритмів гешування зобразимо на графіку (рис. 1), по осі Y – кількість пройдених тестів під час аналізу, по осі X – порядковий номер геш-функції.

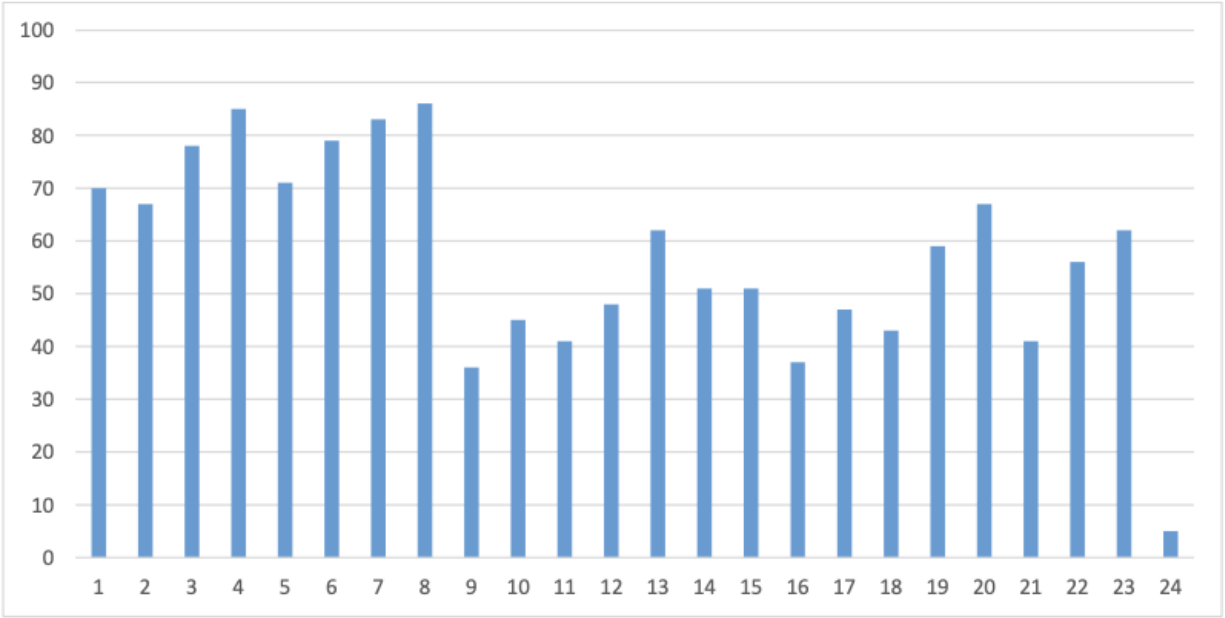

Рисунок 1 – Зображення графіку порівняння статистичних алгоритмів гешування

Дослідження засобу тестування коректності та стійкості алгоритмів гешування показало, що найстійкішими алгоритмами гешування є SHA-384, SHA-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512.

Функції SHA2-224, SHA3-256, ARMADILO-256, H-PRESENT-128, TWISH-128, PHOTON-224, PHOTON-256, QUARK-176, QUARK-256 є достатньо стійкими, але з високою ймовірність мають проблеми із реалізацією.

Функція гешування власної розробки показала найменший результат, вона пройшла тільки тест на вектори, що показує те, що розроблений засіб працює коректно.

Інші ж алгоритми гешування є нестійкими або мають суттєві проблеми із коректністю реалізації.

Висновки

Отже, розглянуто та застосовано метод перевірки статистичної стійкості з використанням кодів Грея геш-функцій. Таким чином, отримані результати статистичної безпеки (табл. 1 та рис. 1) свідчать, що більшість функцій гешування задовольняють встановленим критеріям, тобто за різними показниками вихідні послідовності (геш-значення) не відрізняються (у статистичному сенсі) від реалізації випадкового процесу. Це стосуються, переважно, відомих та стандартизованих алгоритмів, які застосуються в різних криптографічних додатках та вже були суттєво досліджені та вивчені при попередніх випробуваннях. Але серед алгоритмів із табл. 1 є і такі, показники статистичної безпеки яких є незадовільними, або зовсім неприйнятними.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Казміревський В. В. Дослідження стійкості до лінійного та диференційного криптоаналізу функцій гешування: матеріали LI науково-технічної конференції факультету інформаційних технологій та комп'ютерної інженерії, м. Вінниця, 2022. 4 с. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki- 2022/paper/view/ 15844/13341 (дата звернення: 03.03.2023).

2. Harshvardhan Tiwari. Merkle-Damgård Construction Method and Alternatives: A Review. Centre for Incubation, Innovation, Research and Consultancy (CIIRC) Jyothy Institute of Technology. 2017. pp. 284-304. URL: https://sites.google.com/site/sashadkeem1502/home/mobilni-os (дата звернення: 03.03.2023).

3.Лужецький В. А., Баришев Ю. В. Конструкції гешування стійкі до мультиколізій. Наукові праці ВНТУ. – 2010. No1. – 8 с. URL: https://praci.vntu.edu.ua/index.php/praci/article/view/191/189. (дата звернен-ня: 03.03.2023).

4. NIST computer security resource center: Hash Functions SHA-3 Project. 2020. URL: https://csrc.nist.gov/projects/hash-functions/sha-3-project (дата звернення: 03.03.2023).

5. Patrick Nohe. Re-Hashed: The Difference Between SHA-1, SHA-2 and SHA-256 Hash Algorithms. Hashedout by the ssl store. 2018. 11 p. URL: https://www.thesslstore.com/blog/difference-sha-1-sha-2-sha256-hash-algorithms/ (дата звернення: 03.03.2023).

6. Guido Bertoni, Joan Daemen, Seth Hoffert, Michaël Peeters, Gilles Van Assche, Ronny Van Keer. The sponge and duplex constructions. URL: https://keccak.team/sponge_ duplex.html (дата звернення: 03.09.2022).

7. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. Sponge Functions. Presented at ECRYPT Hash Workshop. 2007. 22 p.. URL: http://sponge.noekeon.org/ SpongeFunctions.pdf (дата звернення: 03.03.2023).

8. G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. Keccak specifications. SHA-3 Algorithm Submission. 2008. 14 p.. URL: https://keccak.team/files/ Keccak- submission-3.pdf (дата звернення: 03.09.2022).

9. Hans Dobbertin, Antoon Bosselaers, Bart Preneel. RIPEMD-160: A Strengthened Version of RIPEMD // Katholieke Universiteit Leuven, ESAT-COSIC K. Belgium. pp. 71-82. URL: https://core.ac.uk/download/pdf/ 191296638.pdf (дата звернення: 03.03.2023).

Казміревський Віталій Віталійович — аспірант 1-го курсу кафедри захисту інформації, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: kazmirevskiy1999@gmail.com

Науковий керівник: ***Кондратенко Наталія Романівна*** – к. т. н., професор кафедри захисту інформації, Вінницький національний технічний університет, Вінниця. email: kondrn2014@gmail.com

Vitaliy Kazmirevs'kiy — Faculty for Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, email: kazmirevskiy1999@gmail.com

Scientific supervisor: ***Natalia Kondtratenko*** – PhD (Eng), Professor of Information Protection Department, Vinnytsia National Technical University, Vinnytsia. email: kondrn2014@gmail.com