разностного линейного и билинейного методов криптоанализа. // Труды седьмой общероссийской научной конфиренции ***7.*** *ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. – М.: Госстандарт СССР, 1989. – 28 с.* ***8.*** *Biham E, Shamir A. Differential cryptanalysis of DES-like cryptosystems // Journal of Cryptology. – 1991. – V. 4. – № 1. – P. 3 – 72.* ***9.*** *Lai X., Massey J. L., Murphy S. Markov ciphers and differential cryptanalysis // Advances in Cryptology – EUROCRYPT'91, Proceedings. – Springer Verlag, 1991. – P. 17 – 38.* ***10.*** *S. K. Langford, M. E. Hellman. Differential-linear cryptoanalisis // Advanced in Cryptology. – CRYPTO'94 (LNCS 839). – 1994. – P. 17-25.* ***11.*** *Vaudenay S. On the security of CS-cipher // Fast Software Encryption. – FSE'99, Proceedings. – Springer Verlag, 1999. – P. 260 – 274.* ***12.*** *Kanda M., Takashima Y., Matsumoto T., Aoki K., Otha K. A strategy for constructing fast raund functions with practical security against differential and linear cryptanalysis // Selected Areas in Cryptography. – SAC 1998, Proceedings. – Springer Verlag, 1999. – P. 264 – 279.* ***13.*** *Алексейчук А. Н., Ковальчук Л. В. Верхние границы максимальных значений вероятностей дифференциальных и линейных характеристик шифра Фейстеля, содержащего сумматор по модулю* 2^m *// Прикладная радиоэлектроника. – 2006. – Т. 5. – № 1. – С. 74 – 82.* ***14.*** *Скрипник Л. В., Ковальчук Л. В., Верхние оценки средних вероятностей дифференциалов булевых отображений // Захист інформації – 2006. – №3. – С. 7-13.* ***15.*** *Олексійчук А. Н., Ковальчук Л. В., Кальченко С. В. Криптографічні параметри вузлів заміни, що характеризують стійкість ГОСТ-подібних блокових шифрів відносно методів лінійного та різницевого криптоаналізу // Захист інформації. – 2007. – № 2. – С. 12 – 23.* ***16.*** *Ковальчук Л. Обобщённые марковские шифры: оценка практической стойкости к методу дифференциального криптоанализа // Труды Пятой Общероссийской научной Конференции "Математика и безопасность информационных технологий" – (МаБИТ-06), 25-27 октября 2006. – С. 595 – 599* ***17.*** *Бондаренко М. Ф., Горбенко І. Д. та ін. Обґрунтування вимог та розробка основних рішень з побудови та властивості перспективного БСШ «Мухомор» // Прикладна радіоелектроніка. – 2007. – т. 6. №2. – С.174 – 185.* ***18.*** *Горбенко И. Д., Долгов В. І. та ін. Криптостійкість шифра «Мухомор» // Прикладна радіоелектроніка. – 2007. – т.6. №2. – С.186-194.* ***19.*** *Олейников Р. В., Лисицкая И. В. Исследование свойств подстановок ГОСТ 28147-89, построенных на основе анализа свойств координатных функций // Правове, нормативне та метрологічне забезпечення системи захисту інформації в Україні. – Київ, 2002. – Вип. 3. – С. 123 – 130.* ***20.*** *Олейников Р. В. Дифференциальный криптоанализ алгоритма шифрования ГОСТ 28147-89 // Радиотехника. – 2001. – Вып. 119. – С. 146 – 152.* ***21.*** *Лисицкая И. В., Руженцев В. И. Цепи Фейстеля и дифференциальный криптоанализ // Радиотехника. – 2002. – Вып. 126. – С. 158 – 165.* ***22.*** *Долгов В. И., Лисицкая И. В., Олейников Р. В., Голованич С. Д., Коряк А. С. Дополнительные требования к отбору таблиц подстановок для ГОСТ 28147-89 // Радиотехника. – 2001. – Вып. 119. – С. 153 – 159.* ***23.*** *Долгов В. И., Лисицкая И. В., Олейников Р. В., Шумов А. И. "Слабые" ключи в алгоритме шифрования ГОСТ 28147-89 // Радиотехника. – 2000. – Вып. 114. – С. 63 – 69.* ***24.*** *Лисицкая И. В. Противоречивые подстановки в алгоритме ГОСТ 28147-89 // Информационные системы: Сборник научных трудов. – Харьков, 1995. – НАНУ, ПАНУ, ХВУ. – 9 с.* ***25.*** *Seki H., Toshinobu K. Differential cryptanalysis of redused round of GOST // Selected Areas in Cryptography. – SAC 2000, Proceedings. – Springer Verlag, 2001. – P. 315 – 323.* ***26.*** *Лисицкая И. В., Цепурит Т. В., Лесняк В. В., Пинчук М. В., Мелецкий А. П. Исследование возможностей модернизации шифра ГОСТ 28147-89 с целью дальнейшего повышения его безопасности // Радиотехника. – 2001. – Вып. 119. – С. 160 – 165.*

УДК 003.26 : 004.424.47

ПІДХОДИ ДО ПОБУДОВИ ШВИДКИХ АЛГОРИТМІВ ХЕШУВАННЯ

Володимир Лужецький, Юрій Баришев
Вінницький національний технічний університет

Анотація: **Розглянуто конструкції хешування та підходи до їх розпаралелення. Запропоновано узагальнену конструкцію паралельного хешування, стійку до відомих атак. Визначено оцінки тривалості хешування для різних реалізацій цієї конструкції. Дані оцінки були порівняні з аналогічними оцінками для відомих конструкцій.**

Summary: **Hash constructions and approaches of their parallel computation are considered. The generalized construction of parallel hashing, that is infeasible to known attacks, is proposed. The hash computation durations of this construction different implementations are evaluated. The results of the evaluations were compared with ones of the known constructions.**

Ключові слова: **конструкція хешування, паралельні обчислення, атака Жукса.**

I Вступ

Проблеми, пов'язані з процесом хешування, почали розглядатися криптографією порівняно нещодавно. Однією з найбільш актуальних серед них є пошук найкращого рішення відповідно критерію швидкість/стійкість. Особливо це стає очевидним, коли брати до уваги, що відомі криптографічні перетворення, які покликані забезпечувати стійкість алгоритму хешування, виконуються протягом тривалого часу з використанням сучасних апаратних засобів.

Процес хешування отримав узагальнення і формалізацію в [1], де були запропоновані основні вимоги до хеш-функцій та математична модель хешування, що згодом отримала назву конструкції Меркля-Дамгаарда, які незалежно один від одного формалізували та описали її. Дана конструкція набула достатньо широкого розповсюдження з часу її створення, проте її дослідження виявило ряд недоліків, до яких, зокрема, належить і нездатність цієї конструкції ефективно протистояти пошуку мультиколізій у випадку розпаралелення процесу хешування [2, 3]. Наслідуючи роботу [1], всі наступні математичні моделі хешування в західній літературі почали називатися конструкціями. В даній статті будемо дотримуватись цієї традиції.

Було виконано ряд спроб усунути недоліки конструкції Меркля-Дамгаарда. В [4] описана конструкція Девіса-Мейера. Основним недоліком цієї конструкції стала її вразливість до атаки Келсі з фіксованою точкою [4, С. 7], що дало змогу зловмисникам знаходити другий праобраз зі складністю в найгіршому випадку $O\left(2^{n/2}\right)$, де n – довжина вихідного хеш-значення. Спроба запропонувати конструкцію, яка принципово ширша та відмінна від конструкції Меркля-Дамгаарда, була зроблена в [2]. Проте і ця конструкція не дозволила виконати розпаралелення процесу хешування без втрати стійкості, оскільки основним досягненням роботи [2] є ускладнення атак на хеш-значення, які передбачають попередню підготовку зловмисника. Більше того, конструкція, запропонована в [2], є менш швидкою, ніж Меркля-Дамгаарда, оскільки передбачає обробку достатньо великого масиву (як мінімум, на 25% від повідомлення, що хешується) додаткових вхідних даних.

Найбільш вдалою спробою побудови процесу хешування, стійкого до мультиколізій, з точки зору авторів, була конструкція Люкса, запропонована в [4]. Дана конструкція дозволяє протистояти атакам Келсі, Жукса та подібним до них. До недоліків цієї конструкції можна віднести збільшення вдвічі початкового заповнення порівняно з іншими конструкціями, а отже і виникнення проблем при реалізації ключового варіанту хешування. Крім того в [4] Люкс не зміг узагальнити та інтерпретувати отримані ним результати, проте його конструкція передбачає стійке до мультиколізій розпаралелення обчислень, хоча і варіант розпаралелення, запропонований в [4], не дозволяє досягти збільшення швидкості хешування та, за рівних умов, вимагає вдвічі більших обчислювальних ресурсів порівняно з іншими конструкціями, описаними вище.

Метою даного дослідження є підвищення швидкості хешування при забезпеченні заданого рівня стійкості за рахунок конкатенації частин хеш-значення та розпаралелення обчислень для визначення цих частин.

Для досягнення цієї мети необхідно розв'язати такі задачі:

розробити узагальнену конструкцію процесу паралельного хешування, яка б забезпечувала таку ж стійкість, що і нерозпаралелене хешування великої розрядності;

сформулювати рекомендації щодо практичної реалізації процесу паралельного хешування.

II Огляд відомих конструкцій хешування

Для розробки конструкції паралельного хешування необхідно попередньо розглянути основні відомі конструкції хешування.

Процес хешування відповідно до конструкції Меркля-Дамгаарда передбачає розбиття масиву інформаційних даних $\mathbf{M}$ на блоки рівної довжини $\mathbf{M} = \{m_1, m_2, ..., m_l\}$. Для ускладнення зламу результату хешування зловмисником, дана конструкція також передбачає доповнення даних, що хешуються, блоком, що містить довжину недоповненого повідомлення [4, С. 5]. Оскільки для подальшого опису конструкції ця особливість не є принциповою, то для спрощення будемо вважати, що $\mathbf{M}$ – це вже доповнений цим блоком масив даних. Після того, як дані були підготовлені відповідним чином, відбувається хешування за ітеративним правилом:

$$h_i = f\left(m_i, h_{i-1}\right), \tag{1}$$

де h_i – результат хешування i-го блоку інформаційних даних $\left(i \in [1;t]\right)$;

$f(\cdot)$ – функція ущільнення, яка забезпечує фіксовану розрядність результату.

Хеш-значенням повідомлення h буде h_l, а h_0 є початковим заповненням і може бути використано як ключ за необхідності ключового хешування.

З метою утруднення зламу конструкції Меркля-Дамгаарда була запропонована конструкція Девіса-Мейера [2, С. 3], яка ще відома як однойменний принцип для побудови хеш-функції з блокового шифру. Фактично вона принципово відрізняється від попередньої конструкції тим, що вводиться додаткова групова операція "+" з нейтральним елементом в ітеративному процесі [2, С. 3]:

$$h_i = E_{m_i}(h_{i-1}) + h_{i-1} = f^*(m_i, h_{i-1}) + h_{i-1} = f(m_i, h_{i-1}), \tag{2}$$

де $E_y(x)$ – функція шифрування блоку даних x на ключі y.

Для зламу цієї конструкції Келсі використав властивість групової операції "+" мати нейтральний елемент [4, С. 7]. Так, знаходячи випадок – "фіксовану точку", коли результат $f^*(m_i, h_{i-1})$ дорівнює нейтральному елементу, Келсі описав алгоритм знаходження другого праобразу в найгіршому випадку зі складністю $O(2^{n/2})$.

В [3] було показано, що як конструкція Меркля-Дамгаарда, так і конструкція Девіса-Мейера є вразливими до мультиколізій у випадку їх каскадування, запропонованого в [5, С. 39], для збільшення швидкості хешування.

З метою ускладнення атак, що використовують пошук мультиколізій, описаний в [3], авторами [2] була запропонована конструкція HAIFA (HAsh Iterative FrAmework). Дана конструкція передбачала розширення аргументів ітеративної функції за рахунок введення до аргументів довжини вже оброблених інформаційних даних та псевдовипадкового числа:

$$h_i = f(m_i, h_{i-1}, \#bits, c), \tag{3}$$

де $\#bits$ – кількість вже захешованих бітів повідомлення;
c – псевдовипадкове число.

Введення додаткових аргументів функції ущільнення дозволило утруднити криптоаналітику задачу заздалегідь підготовленої атаки, але в той же час, функція (3) не забезпечує нерозв'язуваність цієї задачі. Крім того, з точки зору швидкості, конструкція HAIFA поступається конструкціям, які розглядалися раніше, оскільки додаткові аргументи потребують додаткової обробки.

Найбільш стійкою до мультиколізій є конструкція Люкса [4], тому вона має найбільший потенціал для розпаралелення порівняно з іншими конструкціями. Конструкція, запропонована Люксом, виплила з очевидної відповіді на атаку Жукса [3] – якщо при розпаралеленні стійкість *n*-розрядного хеш-значення $O(2^{n/2})$, то необхідно вдвічі збільшити розрядність обчислюваних даних в конструкції [4, С. 8]. Таке збільшення розрядності очевидно буде негативно вливати на швидкість та обчислювальні ресурси. Для цього Люксом була запропонована конструкція "подвоєного каналу", що передбачає наявність двох обчислювачів, за допомогою кожного з яких визначається хеш-значення розрядності *n*, при цьому інформаційні дані розбиваються на блоки по *k* біт ($k \geq n$) [4, С. 11]:

$$\begin{cases} h'_i = f(h'_{i-1}, h''_{i-1} \,\|\, m_i) \\ h''_i = f(h''_{i-1}, h'_{i-1} \,\|\, m_i) \end{cases}, \tag{4}$$

де h'_i та h''_i – хеш-значення, отримані на першому та другому каналах відповідно.

Після останньої ітерації передбачається раунд хешування, який ущільнює отримані значення до *n* розрядів [4, С. 11]:

$$h = f(h^\circ, h'_l \,\|\, h''_l \,\|\, 0^{k-n}), \tag{5}$$

де h° – початкове заповнення, аналогічне до h'_0 та h''_0; 0^{k-n} – доповнення до повного $(k+n)$-розрядного блоку.

Основними недоліками конструкції "подвоєного каналу" є подвійні апаратні витрати для реалізації хешування порівняно з іншими конструкціями; велика кількість векторів ініціалізації – 3*n*, що ускладнює розробку ключового варіанту хешування на основі такої конструкції. Варто відзначити, що останній крок негативно впливає на стійкість всієї конструкції загалом та порушує однорідність процесу хешування. Крім того, структура Люкса залишається вразливою до атак, які використовують попередню підготовку криптоаналітика.

ІІІ Загальні теоретичні засади розпаралелення процесу хешування

Однією з найбільших проблем алгоритмів хешування є стійкість до атаки дня народження [3, С. 306]. Її існування стало суттєвим викликом криптографії. Можливість такої атаки фактично стала вимагати збільшення розрядності вихідного хеш-значення вдвічі. Таке збільшення розрядності суттєво збільшує час обчислення хеш-значення та вимагає достатньо великого обчислювального ресурсу. Для виходу із ситуації Бартом Пренілом в [5, С. 39] було запропоновано каскадування хеш-функцій, тобто обчислення двох або більше хеш-значень малої розрядності для одного відкритого тексту та їх подальша конкатенація для досягнення заданої розрядності вихідного хеш-значення. Зауважимо, що на час виходу роботи [5] поняття хеш-функції було дещо звужене конструкцією Меркля-Дамгаарда, а тому каскадування в [5] пропонувалося саме для хеш-функцій такого виду.

Розглянемо математичну модель процесу хешування, яка використовує каскадування для того, щоб в подальшому визначити шляхи її вдосконалення.

Нехай є дві функції ущільнення $f^{(1)}(\cdot)$ та $f^{(2)}(\cdot)$. Кожна з цих функцій використовує як вхідні данні блок повідомлення, що хешується, сталої довжини $m_i\left(i=\overline{1,l}\right)$ та хеш-значення, отримане на попередній ітерації алгоритму h_{i-1}, тобто *i*-те значення буде розраховуватись за допомогою кожної функції відповідно до формули (1).

Каскадування може мати різний вигляд, але, зазвичай, воно передбачає отримання вихідного хешу шляхом конкатенації результатів хешування повідомлення **M** за допомогою кожної з функцій $h=h_l^{(1)}\,||\,h_l^2$, де $h_l^{(1)}$, $h_l^{(2)}$ – вихідні значення функцій $f^{(1)}(\cdot)$ та $f^{(2)}(\cdot)$ відповідно.

Після публікації [3] використання "чистого" каскадування стало неактуальним у зв'язку з виявленими недоліками. Проте даний підхід отримав подальший розвиток. В [6, С. 2] розглядається розпаралелена конструкція Меркля-Дамгаарда з подвійним змішуванням, що була запропонована як основа для майбутнього стандарту SHA-3 на конкурсі, організованим NIST. Ця конструкція використовує 2*s* різних функцій ущільнення, які послідовно застосовуються до блоків повідомлення, та дві функції перемішування отриманих результатів. Реалізується два паралельних процеси хешування, які оброблюють *s* блоків повідомлення, після чого застосовується функція перемішування отриманих результатів [6, С. 2]. Попри те, що дана конструкція є більш стійкою, ніж класичне каскадування, в [6] показано, що вона все ще є набагато вразливішою до пошуку мультиколізій та другого праобразу, ніж це вимагається від сучасних хеш-функцій.

Шляхами покращення методу каскадування було використання конструкцій побудови хеш-алгоритмів, відмінних від конструкції Меркля-Дамгаарда [3]. Одна з найбільш цікавих конструкцій Tandem-DM була заснована на конструкції Девіса-Мейера та блокового шифру як функції ущільнення [7, С. 3]. Нехай функції $f^{(1)}(\cdot)$ та $f^{(2)}(\cdot)$ є функціями, що використовують конструкцію Девіса-Мейера, тобто аналогічні $f(\cdot)$ у виразі (2). Tandem-DM передбачає ітеративний процес обробки вхідних, організований таким чином:

$$\begin{cases} h_i^{(1)} = f^{(1)}\left(h_{i-1}^{(1)}, h_{i-1}^{(2)}, m_i\right) \\ h_i^{(2)} = f^{(2)}\left(h_i^{(1)}, h_{i-1}^{(1)}, h_{i-1}^{(2)}, m_i\right) \end{cases}. \tag{6}$$

Стійкість цієї конструкції ще не досліджена досконально, але її попередні дослідження, наведені в [7], не виявили вразливих місць у цій конструкції. Основним недоліком такої конструкції, навіть, при її теоретично доведеній стійкості та використанні ідеального блокового шифру, є необхідність попереднього обчислення $h_i^{(1)}$ для визначення $h_i^{(2)}$. Відповідно принципового розпаралелення обчислень не відбувається, а збільшення швидкості може бути досягнуто лише за умови зменшення розрядності операцій та суттєво залежить від обраного блокового шифру.

Із запропонованих моделей найбільшу увагу наукової громадськості привернула конструкція Люкса, яка була розглянута вище. Підхід Люкса може покращити стійкість, але в зв'язку з тим, що він збільшує витрати ресурсів і не зменшує показник час/розрядність порівняно з іншими конструкціями, він також не може вважатися кращим виходом із ситуації.

Для збільшення швидкості пропонується виконувати розпаралелення процесу хешування на кожній ітерації. Для пояснення ідеї, що пропонується, використаємо поняття каналу, введене в [4]. У загальному випадку будемо розглядати *q* каналів обчислення. З метою спрощення й уніфікації реалізації хешування бажано, щоб ці канали реалізовували однакові функції або подібні, що відрізняються певним параметром,

але для загального випадку будемо вважати, що різні канали хешування реалізують різні функції ущільнення $f^{(1)}(\cdot), f^{(2)}(\cdot),..., f^{(q)}(\cdot)$, але з тією особливістю, що вони виконують перетворення приблизно за однаковий час на одній й тій самій апаратній платформі.

Оскільки більшості реалізацій паралельного хешування загрожує атака Жукса [3], то, з метою унеможливлення її реалізації для конструкції хешування, що пропонується, передбачимо кореспонденцію всіх каналів обчислення після кожної ітерації. Іншими словами, результат хешування, який обчислюється в кожному каналі, має залежати від результату хешування, отриманого в інших каналах після попередньої ітерації. Таким чином, зловмиснику доведеться знаходити колізії для повної розрядності вихідного хеш-значення, а не зменшеної в q раз, як це можливо при каскадуванні [3, С. 313]. Очевидно, що для ефективної реалізації таких ідей, є бажаною однакова швидкість обчислення однієї ітерації в усіх каналах, оскільки при такій конструкції загальна швидкість хешування буде дорівнювати швидкості хешування в найповільнішому каналі обчислення.

Таким чином конструкція хешування набуде такого вигляду:

$$\begin{cases} h_i^{(1)} = f^{(1)}\left(h_{i-1}^{(1)}, h_{i-1}^{(2)},..., h_{i-1}^{(q)}, m_i\right) \\ h_i^{(2)} = f^{(2)}\left(h_{i-1}^{(1)}, h_{i-1}^{(2)},..., h_{i-1}^{(q)}, m_i\right) \\ ... \\ h_i^{(q)} = f^{(q)}\left(h_{i-1}^{(q)}, h_{i-1}^{(q)},..., h_{i-1}^{(q)}, m_i\right) \end{cases}, \tag{7}$$

де $h_{i-1}^{(1)}, h_{i-1}^{(2)},..., h_{i-1}^{(q)}$ – проміжні хеш-значення, отримані на попередньому кроці, розрядності n/q для очікуваного n-розрядного результату хешування.

При потребі ключового хешування конструкція (7) може використовувати як ключ вектор ініціалізації $h_0^{(1)}, h_0^{(2)},..., h_0^{(q)}$. Таким чином, результатом хешування повідомлення **M** буде $h = h_l^{(1)} \| h_l^{(2)} \| ... \| h_l^{(q)}$.

Для конструкції (7) атака Жукса неможлива, оскільки, знайшовши колізію в одному каналі на певній ітерації, зловмисник не може використати її з метою підміни одного інформаційного блоку на інший, не перевіривши, чи викличе це ж повідомлення колізію в інших $q-1$ каналах з урахуванням їх взаємного впливу одне на одного на кожній ітерації.

Попри цю сукупність факторів, збіг яких потрібен зловмиснику, варто зауважити, що повідомлення, які мають логічний зміст з точки зору людини, далеко не використовують всі можливі символи, крім того повідомлення можуть бути "шаблонними" і незначно відрізнятися, скажімо, наприклад, лише вартістю контракту. Такі особливості кодового представлення інформації в обчислювальній техніці збільшують шанси зловмисника на можливість багатократного використання однієї колізії. Відповідно зловмисник може "підготуватися" до атаки заздалегідь. Прикладом таких атак може бути атака Келсі-Кохо [2, С. 4]. Для ускладнення криптоаналізу цієї задачі пропонується ввести до входів каналів ще й псевдовипадкові числа. З урахуванням цього фактору конструкція (7) набуде такого вигляду:

$$\begin{cases} h_i^{(1)} = f^{(1)}\left(h_{i-1}^{(1)}, h_{i-1}^{(2)},..., h_{i-1}^{(q)}, m_i, c_i^{(1)}\right) \\ h_i^{(2)} = f^{(2)}\left(h_{i-1}^{(1)}, h_{i-1}^{(2)},..., h_{i-1}^{(q)}, m_i, c_i^{(2)}\right) \\ ... \\ h_i^{(q)} = f^{(q)}\left(h_{i-1}^{(1)}, h_{i-1}^{(2)},..., h_{i-1}^{(q)}, m_i, c_i^{(q)}\right) \end{cases}, \tag{8}$$

де $c_i^{(1)}, c_i^{(2)},..., c_i^{(q)}$ – псевдовипадкові числа.

За допомогою конструкції (8) можна побудувати паралельний процес хешування, стійкий до атак, що базуються на мультиколізіях та попередній підготовці, проте можна запропонувати ще багато й інших конструкцій, подібних до (8), які будуть зберігати ці властивості стійкості, тому узагальнимо конструкцію (8). У загальному випадку, кожний канал обчислення може реалізовувати функцію, яка залежить не від одного попереднього значення, а від усіх попередніх значень, визначених на всіх каналах обчислень. Також можуть використовуватись декілька блоків інформаційних даних та декілька псевдовипадкових чисел, що покращить стійкість хешування. Існує велика кількість можливих варіантів реалізації цього підходу, але при будь-якому варіанті хешування буде стійким лише за умови зав'язки хеш-значення від блоку інформаційних даних, хеш-значень, отриманих на попередніх ітераціях на всіх каналах,

псевдовипадкового числа. Введемо функції групування $H(\cdot), M(\cdot), C(\cdot)$, за допомогою яких визначатимуться ці аргументи:

$$\begin{cases} h_i^{*(jv)} = H_i^{(j)}\left(h_0^{(1)}, h_1^{(1)}, \ldots, h_{i-1}^{(1)}, h_0^{(2)}, h_2^{(2)}, \ldots, h_{i-1}^{(q)}, m_1, m_2, \ldots, m_l, c_1^{(j)}, c_2^{(j)}, \ldots, c_l^{(j)}\right) \\ m_i^{*(j)} = M_i^{(j)}\left(h_0^{(1)}, h_1^{(1)}, \ldots, h_{i-1}^{(1)}, h_0^{(2)}, h_2^{(2)}, \ldots, h_{i-1}^{(q)}, m_1, m_2, \ldots, m_l, c_1^{(j)}, c_2^{(j)}, \ldots, c_l^{(j)}\right) \\ c_i^{*(j)} = C_i^{(j)}\left(h_0^{(1)}, h_1^{(1)}, \ldots, h_{i-1}^{(1)}, h_0^{(2)}, h_2^{(2)}, \ldots, h_{i-1}^{(q)}, m_1, m_2, \ldots, m_l, c_1^{(j)}, c_2^{(j)}, \ldots, c_l^{(j)}\right) \end{cases}. \quad (9)$$

Підставимо аргументи, визначені в (9) в конструкцію (8):

$$\begin{cases} h_i^{(1)} = f^{(1)}\left(h_i^{*(11)}, h_i^{*(12)}, \ldots, h_i^{*(1q)}, m_i^{*(1)}, c_i^{*(1)}\right) \\ h_i^{(2)} = f^{(2)}\left(h_i^{*(21)}, h_i^{*(22)}, \ldots, h_i^{*(2q)}, m_i^{*(2)}, c_i^{*(2)}\right) \\ \ldots \\ h_i^{(q)} = f^{(q)}\left(h_i^{*(q1)}, h_i^{*(q2)}, \ldots, h_i^{*(qq)}, m_i^{*(q)}, c_i^{*(q)}\right) \end{cases}. \quad (10)$$

В конструкції (10) функції ущільнення є фіксованими в кожному каналі обчислення, але вони можуть бути і різними та обиратись на кожній ітерації за певним правилом:

$$r_i^{(j)} = R_i^{(j)}\left(h_0^{(1)}, h_1^{(1)}, \ldots, h_{i-1}^{(1)}, h_0^{(2)}, h_2^{(2)}, \ldots, h_{i-1}^{(q)}, m_1, m_2, \ldots, m_l, c_1^{(j)}, c_2^{(j)}, \ldots, c_l^{(j)}\right), \quad (11)$$

де $r_i^{(j)}$ – номер функції ущільнення, що буде використовуватись в *j*-му каналі обчислення на *i*-ій ітерацій.

З урахуванням (11), конструкція (10) матиме такий вигляд:

$$\begin{cases} h_i^{(1)} = f_{r_i^{(1)}}^{(1)}\left(h_i^{*(11)}, h_i^{*(12)}, \ldots, h_i^{*(1q)}, m_i^{*(1)}, c_i^{*(1)}\right) \\ h_i^{(2)} = f_{r_i^{(2)}}^{(2)}\left(h_i^{*(21)}, h_i^{*(22)}, \ldots, h_i^{*(2q)}, m_i^{*(2)}, c_i^{*(2)}\right) \\ \ldots \\ h_i^{(q)} = f_{r_i^{(q)}}^{(q)}\left(h_i^{*(q1)}, h_i^{*(q2)}, \ldots, h_i^{*(qq)}, m_i^{*(q)}, c_i^{*(q)}\right) \end{cases}. \quad (12)$$

Розглянемо особливості реалізації узагальненої конструкції паралельного хешування (12).

IV Особливості реалізації процесу паралельного хешування

Практична реалізація конструкції (12) може бути здійснена за такими варіантами:

- на одному універсальному процесорі реалізуються всі канали обчислення;
- кожному каналу обчислення відповідає окремий універсальний процесор;
- на одному спеціалізованому процесорі реалізуються всі канали обчислення;
- кожному каналу обчислення відповідає окремий спеціалізований процесор.
- Відмінність між спеціалізованим і універсальним процесорами полягає в тому, що в спеціалізованому процесорі архітектура спроектована для вирішення конкретної задачі і тому, деякі операції відбуваються паралельно в межах каналу.
- Визначимо тривалість обчислення хеш-значення в одному каналі протягом ітерації на універсальному процесорі. Відповідно конструкції (12) ітерація передбачає:
- вибір функції ущільнення;
- генерування псевдовипадкового числа;
- групування псевдовипадкових чисел;
- групування блоків інформаційних даних;
- групування попередніх значень хешу (*q* раз);
- обчислення нового значення хешу.

Таким чином, час одної ітерації хешування в одному каналі на універсальному процесорі становитиме:

$$t_1 = t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h} + t_f^{(n/q)}, \quad (13)$$

де t_r – час вибору функції ущільнення;

t_{gen} – час генерування псевдовипадкового числа;

t_{gr_c} – час групування псевдовипадкових чисел;

t_{gr_m} – час групування блоків інформаційних даних;

t_{gr_h} – час групування попередніх значень хешу;

$t_f^{(n/q)}$ – час обчислення хеш-значення розрядності n/q за допомогою функції $f_{r_i^{(j)}}^{(j)}(\cdot)$.

Отже, ітерація на одному універсальному процесорі буде виконуватись за час $q \cdot t_1$, а на паралельних універсальних процесорах – за t_1. При визначенні хеш-значення за допомогою спеціалізованого процесора буде можливим розпаралелення обчислень в каналі. Таким чином, можна буде виконувати всі операції групування паралельно. Розглянемо найгірший з точки зору швидкодії випадок, коли генерування псевдовипадкових чисел відбувається під час самого хешування. В такому разі, групування псевдовипадкових чисел можливо лише після завершення процесу генерації. Відповідно, тривалість однієї ітерації при реалізації одного каналу обчислення за допомогою спеціалізованого процесора буде визначатись таким чином:

$$t_1' = \max\left(t_r; t_{gen} + t_{gr\_c}; t_{gr\_m}; t_{gr\_h}\right) + t_f^{(n/q)}. \tag{14}$$

Час групування та час вибору форми функції багато в чому залежать від способу їх здійснення. За умови подібної реалізації цих операцій вираз (14) можна спростити:

$$t_1' = t_{gen} + t_{gr\_c} + t_f^{(n/q)}. \tag{15}$$

Отже на одному спеціалізованому процесорі обчислення будуть тривати $q \cdot t_1'$, а на q паралельних спеціалізованих процесорах t_1'.

Час хешування *n* біт повідомлення за допомогою конструкції, що пропонується, буде тривати *q* ітерацій та становитиме:

- для реалізації на одному універсальному процесорі:

$$t_{u1} = q^2 \cdot \left(t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h}\right) + q^2 \cdot t_f^{(n/q)}; \tag{16}$$

- для реалізації на q універсальних процесорах :

$$t_{uq} = q \cdot \left(t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h}\right) + q \cdot t_f^{(n/q)}; \tag{17}$$

- для реалізації на одному спеціалізованому процесорі:

$$t_{s1} = q^2 \cdot \left(t_{gen} + t_{gr\_c}\right) + q^2 \cdot t_f^{(n/q)}; \tag{18}$$

- для реалізації на q спеціалізованих процесорах

$$t_{sq} = q \cdot \left(t_{gen} + t_{gr\_c}\right) + q \cdot t_f^{(n/q)}. \tag{19}$$

Очевидно, що для інших конструкцій, крім HAIFA, час виконання становитиме $t_f^{(n)}$. Для HAIFA цей час дещо більший внаслідок використання псевдовипадкових чисел та номерів блоків інформаційних даних цією конструкцією. Якщо функція ущільнення використовуватиме лише побітові та унарні операції, то отримаємо співвідношення:

$$t_f^{(n)} = q \cdot t_f^{(n/q)}. \tag{20}$$

Розділивши час хешування (16) – (19) на час (20), отримаємо формули для визначення програшу/виграшу від використання конструкції, що пропонується, порівняно з відомими конструкціями для різних реалізацій:

- на одному універсальному процесорі:

$$\delta_{u1} = \frac{t_{u1}}{t_f^{(n)}} = \frac{q \cdot (t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h})}{t_f^{(n/q)}} + q; \tag{21}$$

- на q універсальних процесорах:

$$\delta_{uq} = \frac{t_{uq}}{t_f^{(n)}} = \frac{t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h}}{t_f^{(n/q)}} + 1; \tag{22}$$

- на одному спеціалізованому процесорі:

$$\delta_{s1} = \frac{t_{s1}}{t_f^{(n)}} = \frac{q \cdot \left(t_{gen} + t_{gr\_c}\right)}{t_f^{(n/q)}} + q \; ; \tag{23}$$

- на q спеціалізованих процесорах:

$$\delta_{sq} = \frac{t_{sq}}{t_f^{(n)}} = \frac{t_{gen} + t_{gr\_c}}{t_f^{(n/q)}} + 1 . \tag{24}$$

Дані показники передбачають застосування лише побітових та унарних операцій, але зазначені операції не можуть забезпечити стійкість хешування, тому при хешуванні також використовують і нелінійні операції або лише нелінійні, як це зроблено в [8] для хешування теоретично доведеної стійкості. Оскільки сучасне хешування передбачає $n \geq 256$ біт, відповідно ці нелінійні операції будуть суттєво довше виконуватись на 32- та 64-розрядних універсальних процесорах, ніж декілька операцій малої розрядності. Отже, якщо обрати q таким, що $n/q \leq 32(64)$, то співвідношення (20) зміниться і буде мати такий вигляд:

$$t_f^{(n)} = q^2 \cdot t_f^{(n/q)} . \tag{25}$$

Враховуючи співвідношення (25), формули (21) – (24) мають бути скореговані так:

- для реалізації запропонованої конструкції на одному універсальному процесорі:

$$\delta_{u1} = \frac{t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h}}{t_f^{(n/q)}} + 1 ; \tag{26}$$

- для реалізації запропонованої конструкції на q універсальних процесорах:

$$\delta_{uq} = \frac{t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h}}{q \cdot t_f^{(n/q)}} + \frac{1}{q} \; ; \tag{27}$$

- для реалізації запропонованої конструкції на одному спеціалізованому процесорі:

$$\delta_{s1} = \frac{t_{gen} + t_{gr\_c}}{t_f^{(n/q)}} + 1 ; \tag{28}$$

- для реалізації запропонованої конструкції на q спеціалізованих процесорах:

$$\delta_{sq} = \frac{t_{gen} + t_{gr\_c}}{q \cdot t_f^{(n/q)}} + \frac{1}{q} . \tag{29}$$

Введемо таке позначення для часу обчислень, що передують обчисленню функції ущільнення:

$$t_{pr} = t_r + t_{gen} + t_{gr\_c} + t_{gr\_m} + q \cdot t_{gr\_h} . \tag{30}$$

Використовуючи формули (21) – (24) та (26) – (29), зведемо в табл. 1 оцінки програшу/виграшу від використання хешування на основі конструкції, що пропонується, порівняно з відомими конструкціями для реалізації за допомогою універсального процесора, враховуючи позначення (30) та особливості функції ущільнення (20) і (25).

Таблиця 1 – Оцінки програшу/виграшу залежно від особливостей реалізації хеш-функції

Кількість процесорів	Використовувані операції	δ	
		$t_f^{(n/q)} \sim t_{pr}$	$t_f^{(n/q)} >> t_{pr}$
1	побітові	$\sim 2q$	$\approx q$
	регістрові	~ 2	≈ 1
q	побітові	~ 2	≈ 1
	регістрові	$\sim 2/q$	$\approx 1/q$

З табл. 1 видно, що хешування за допомогою конструкції, що пропонується, буде швидшим лише при його паралельній реалізації за умов наявності у функції ущільнення операцій, відмінних від побітових. Також, з табл. 1 випливає, що за умов, коли тривалість попередніх обчислень t_{pr} є суттєво меншою за час

обчислення значення функції ущільнення, чому відповідає випадок хешування теоретично доведеної стійкості [8], виграш в часі стає більшим при паралельній реалізації та меншим при реалізації на одному процесорі, ніж у випадку, коли вони порівнювані. Тому пропонується реалізовувати функції групування, вибору функції ущільнення та генерування псевдовипадкових чисел за допомогою операцій, які швидко виконуються на апаратних платформах. Останнє можна зробити без втрати стійкості конструкції загалом, оскільки стійкість забезпечується, в першу чергу, за допомогою функції ущільнення.

У табл. 2 наведемо програш/виграш у швидкості при вихідній довжині хеш-значення 256 біт, за умови, що тривалість попередніх обчислень є порівнюваною з часом обчислення значення функції ущільнення.

Таблиця 2 – Програш/виграш при обчисленні 256-розрядного хеш-значення

Кількість процесорів	Використовувані операції	Розрядність процесорів			
		8	16	32	64
1	побітові	64	32	16	8
	регістрові	2	2	2	2
q	побітові	2	2	2	2
	регістрові	1/16	1/8	1/4	1/2

З табл. 2 видно, що чим меншої розрядності обираються канали хешування, тим більшим є програш при реалізації запропонованої конструкції на одному процесорі і тим більше виграш при реалізації на *q* паралельних процесорах. Аналогічні результати будуть і при збільшенні розрядності вихідного хеш-значення. Відповідно, конструкція, що пропонується, може ефективно впроваджуватись в пристроях з невеликою розрядністю процесорів, до яких, наприклад, відносяться смарт-картки.

Якщо замість позначення (30) підставити таке позначення

$$t'_{pr} = t_{gen} + t_{gr\_c}, \tag{31}$$

то оцінки для реалізації запропонованої конструкції за допомогою спеціалізованих процесорів будуть аналогічні наведеним в табл. 1 та 2.

V Висновки

У статті запропоновано узагальнену конструкцію хешування з розпаралеленням обчислень, яка є криптографічно стійкою до відомих атак. Особливість цієї конструкції полягає в тому, що процес хешування складається з *q* паралельних процесів обчислення хеш-значень в *q* раз меншої розрядності, ніж потрібна розрядність хеш-значення. Остаточний результат хешування є конкатенацією результатів паралельних процесів хешування. Завдяки такому розпаралеленню досягається зменшення часу хешування порівняно з відомими підходами.

Стійкість запропонованої конструкції до відомих атак досягається за рахунок того, що кожен з паралельних процесів як вхідні дані використовує певні набори блоків даних, що хешуються, певні набори псевдовипадкових чисел, що характеризують блоки даних, певні набори результатів хешування, отриманих на попередніх ітераціях іншими процесами хешування.

Одержано оцінки часу хешування на основі запропонованої конструкції, які свідчать про те, що практичне використання конструкції є доцільним лише в разі паралельної реалізації хешування на *q* процесорах $(q > 1)$ і при реалізації функції ущільнення на основі регістрових операцій. Наприклад, для одержання хеш-значення розрядності 256 біт з використанням шістнадцяти 16-розрядних процесорів, час хешування у 8 разів менший, ніж час хешування за допомогою відомих конструкцій.

Література: ***1.*** *R. C. Merkle. Secrecy, Authentifications and Public Key Systems. A Dissertation Submitted to the Department of Electrical Engineering and the Committee on Graduate Studies of Stanford University in Particial Fulfillment of the Requirements. / Режим доступу до ресурсу – http://www.merkle.com/papers/Thesis1979.pdf* ***2.*** *E. Biham, O. Dunkelman. A Framework for Iterative Hash Functions: HAIFA. / Режим доступу до ресурсу - http://csrc.nist.gov/pki/HashWorkshop/2006/Papers/DUNKELMAN_NIST3.pdf* ***3.*** *A. Joux. Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions. In M. K. Franklin, editor, CRYPTO'04, volume 3152 of Lecture Notes in Computer Science, pages 306-316. Springer, 2004.* ***4.*** *S. Lucks. Design Principles for Iterated Hash Functions. E-print (September 29, 2004) // Режим доступу до ресурсу - http://eprint.iacr.org/2004/253.pdf* ***5.*** *B. Preneel. Analysis and Design of Cryptographic Hash Functions. PhD thesis, Katholieke Universiteit Leuven, 1993 // Режим доступу до ресурсу – http://homes.esat.kuleuven.be/~preneel/phd_preneel_feb1993.pdf* ***6.*** *N. Fergusson, S.*

Lucks. Attacks on AURORA-512 and the Double-MIX Merkle-Damgard Transform // Режим доступу до ресурсу – http://eprint.iacr.org/2009/113.pdf **7.** E. Fleischmann, M. Gorski, S. Lucks. On the Security Of Tandem-DM. Full version of paper (2009/02/03) // Режим доступу до ресурсу – http://eprint.iacr.org/2009/054.pdf **8.** Патент України на корисну модель № 18693 МПК G 09 C 1/00. Спосіб ключового хешування теоретично доведеної стійкості / Стасєв Ю. В., Кузнецов О. О., Євсеєв С. П., Чевардін В. Є., Малахов С. В., Гришко А. В.; заявник та патентовласник Харківський університет повітряних сил. – №u200605734 ; заявл. 25.05.06 ; опубл. 15.11.06, Бюл. № 11.

УДК 621.391

ИСПОЛЬЗОВАНИЕ BDS-СТАТИСТИКИ ДЛЯ ОЦЕНКИ ПАРАМЕТРОВ ОДНОМЕРНЫХ ОТОБРАЖЕНИЙ ПО НАБЛЮДЕНИЮ ХАОТИЧЕСКОГО ВРЕМЕННОГО РЯДА

Константин Васюта
Харьковский университет Воздушных Сил

Аннотация: **Для анализа структурной скрытности систем передачи информации с хаотической несущей, формируемой одномерным отображением, предложен новый метод оценки его параметров с использованием непараметрической BDS-статистики. В работе показано, что, опираясь на различие в топологических свойствах в фазовом пространстве хаотических, регулярных и случайных процессов с независимыми значениями можно решать задачу оценки параметров отображения по наблюдению хаотического временного ряда на фоне белого шума. Приведены результаты численного моделирования предложенного метода оценки параметров хаотических отображений для различного характера шума.**
Summary: **For the analysis of structural stealthiness of transmission systems of the information with random bearing, shaped one-dimensional representation, offers a new method of an estimate of its parameters with use nonparametric BDS-statistics. It is in-process shown, that, leaning against difference in absolute concepts in a phase space random, regular and random processes with independent values it is possible to solve a problem estimates of parameters representation on observation of random time series against a white noise. Outcomes of numerical modeling of the offered method of an estimate of parameters of random representations for various character of noise are resulted.**
Ключевые слова: **Скрытность, хаотические сигналы, BDS-статистика, оценка параметров.**

Некоторые фундаментальные свойства динамического хаоса вызвали естественный интерес исследователей к их использованию для обеспечения скрытности работы радиосистем (излучаемых ими сигналов) [1]. Достоинства таких радиосистем определяются использованием для передачи сообщений случайно-подобных во временной (спектральной) области хаотических процессов (последовательностей), которые маскируют сигнал под шум. Однако, в отличие от случайных, «малоразмерные» хаотические процессы и последовательности, при их анализе на фазовой плоскости (фазовом пространстве), имеют регулярную структуру и это свойство снижает их скрытность.

В дальнейшем под скрытностью будем понимать [2] способность противостоять мерам радиотехнической разведки: обнаружению сигнала и определению его структуры на основе оценки ряда его параметров без учета возможности раскрытия смысла информации.

Традиционно при оценке вероятности разведки (вероятности обнаружения и раскрытия структуры сигнала) используется энергетический критерий и не учитывается “форма” сигнала. Оптимальный обнаружитель представляет собой измеритель мощности процесса, позволяющий выявлять энергетические приращения над мощностью шумов при наличии сигнала в анализируемом диапазоне частот. В тоже время, понятие “форма” процесса рассматриваемое как лингвистическая характеристика, которая косвенно определяет зависимость его элементов, может быть формализована (представлена численной мерой) и способна дать более объективную оценку вероятности разведки [3]. Такая формализация может быть выполнена, например, с помощью следующей последовательности: “форма” процесса $\rightarrow$ структурированность аттрактора процесса $\rightarrow$ зависимость значений процесса $\rightarrow$ критерий зависимости (динамический или статистический) $\rightarrow$ мера зависимости (например, динамические инварианты: показатели Ляпунова, корреляционная размерность или энтропия). Корреляционной размерностью можно характеризовать структурированность аттрактора (устойчивую упорядоченность его элементов и связей), связанного с анализируемым процессом. Например, случайный I.I.D (independent and identically distributed) процесс неструктурирован, его аттрактор полностью “заполняет” пространство вложения, а