Новий підхід до побудови криптографічних хеш-функцій

Лужецький В.А.[1], Кисюк Д.В.[2]

[1]Проф., д.т.н., завідувач кафедри захисту інформації, Вінницький національний технічний університет
вул. Хмельницьке шосе 95, м. Вінниця, Україна, lva_zi@mail.ru
[2]Асист. кафедри обчислювальної техніки, Вінницький національний технічний університет
вул. Хмельницьке шосе 95, м. Вінниця, Україна, kneimad @gmail.com

***Анотація* — Розглянуто основні сучасні конструкції побудови криптографічних хеш-функцій та проаналізовано їх переваги та недоліки. Запропоновано принципово новий підхід до побудови хеш-функцій без використання ітеративної процедури. Наведено математичну модель запропонованого методу.**

Ключові слова: хешування, гешування, хеш-функція, неітеративне хешування.

New method of hash-function creation

Luzhetskyi V.A.[1], Kysiuk D.V. [2]

[1] Prof., Head of Department of Information Protection, Vinnytsia National Technical University
Khmelnytske shose str., 95, Vinnytsia, Ukraine, lva_zi@mail.ru
[2]Asist., Department of Computer Science, Vinnytsia National Technical University
Khmelnytske shose str., 95, Vinnytsia, Ukraine, kneimad @gmail.com

***Abstract* — Basic methods of cryptographic hash-function creation was considered. The advantages and disadvantages of these methods also was analyzed. Authors offered the new method of non-iteration hash-function creation. The mathematical and schematic models of this method were presented.**

Keywords: hashing, hash - function, noniterrate hashing.

I. Вступ

Криптографічні хеш-функції є одними з найважливіших видів криптографічних перетво-рень. Вони широко застосовуються у задачах криптографічного захисту інформації. Натепер існує велика кількість різноманітних хеш-функцій. Проте, зростаючі вимоги, що висуваються до швидкості хешування даних, а також необхідність реалізації у пристроях з невеликими обчис-лювальними можливостями, приводять до необхідності розробки нових методів хешування, з можливою їх спеціалізацією для певних пристроїв чи повідомлень особливого виду [1, 2].

Більшість сучасних хеш-функцій реалізують ітераційний принцип обчислень, що передбачає поділ повідомлення на блоки, знаходження чергового хеш-значення шляхом спільного оброблення попереднього хеш-значення і блоку даних. Найпоширенішою конструкцією хешування є схема Меркеля – Дамгарда, яка використовується у хеш-функціях MD5 і SHA-1, Wide pipe, Double pipe, 3C та інші [2].

Близькою за структурою до схеми Меркеля – Дамгарда є структура ітеративного режиму хешування Hash Iterative Framework (HAIFA), яка порівняно з нею має покращену стійкість до атак, і була використана при побудові хеш-функції BLAKE [3, 4].

Відмінність схеми HAIFA полягає у тому, що на кожній ітерації у процесі ущільнення беруть участь не тільки попереднє проміжне хеш-значення h_{i-1} і значення попереднього блоку m_i, але й псевдовипадкове число (*salt*) та номер початкового блоку L_i [5].

Ще одним важливим представником ітеративних конструкцій є конструкція «Криптографічна губка» (Sponge), що використана при побудові хеш-функції Keccak, яка є переможцем конкурсу на новий стандарт хешування SHA-3. На відміну від алгоритмів, що застосовують схему Меркеля – Дамгарда, конструкція «Губка» працює з розширеним блоком вхідних даних [6, 7].

Усі описані схеми успішно застосовуються для створення хеш-функцій з використанням ітераційних процедур. Проте, у зв'язку з тим, що питання про лавиноподібний ефект з початковим заповненням при великій кількості ітерацій недостатньо досліджений, і, відповідно, використання цих схем є недостатньо обґрунтованим. Також, до недоліків слід віднести такі особливості цих підходів [3, 8]:

1) різний вплив блоків даних на остаточний результат хешування (значення першого блоку бере участь у формуванні усіх проміжних хеш-значень через ітеративність процедури, а значення останнього блоку враховується лише на останній ітерації);

2) існує потенційна можливість за результатами кожної ітерації відновити блок даних і попереднє хеш-значення, тому зазвичай дані методи

намагаються ускладнити процедуру такого відновлення за рахунок ускладнення обчислень на кожній ітерації [9].

Для усунення вказаних недоліків автори пропонують принципово новий підхід до побудови хеш-функцій.

II. Метод хешування на основі характеристчних ознак послідовної стурктури даних

Вхідне повідомлення M розбивається на послідовність байтів:

$M = \{ m_1, m_2, ..., m_L \}$.

Кожен байт розглядається як число n, що відповідає ASCII – коду символу представленого байтом m_l ($l = 1 \div L$), тобто $n = f(m_l)$.

Повідомлення характеризується кількістю елементів k_n, що мають числовий еквівалент n ($n = 0 \div 255$) та номерами позицій у яких розташовані ці елементи.

На основі цих характеристик утворюється два масиви $\boldsymbol{K}$ та $\boldsymbol{S}$:

$\boldsymbol{K} = (k_0, k_1, \ldots, k_{255})$,

$\boldsymbol{S} = (s_0, s_1, \ldots, s_{255})$,

де,

$$s_j^{(n)} = \begin{cases} s_{j-1}^{(n)} + l, & \text{якщо } f(m_l) = n \\ s_{j-1}^{(n)}, & \text{інакше} \end{cases},$$

$s_0^{(i)} = 0$,

$s_n = s_{k_n}^{(n)}$,

$j = 1 \div k_n$.

Для підрахунку хеш-коду, необхідно привести довжину масивів $\boldsymbol{K}$ та $\boldsymbol{S}$ до потрібного значення. Розглянемо варіанти ущільнення масивів:

1) ущільнення масивів $\boldsymbol{K}$ та $\boldsymbol{S}$ до розміру хеш-значення та застосування деякої функції до цих двох ущільнених масивів;

$h = f(C(\boldsymbol{K}), C(\boldsymbol{S}))$

2) Застосування деякої операції до двох масивів з подальшим ущільненням отриманого результату до довжини хеш-значення.

$h = C (g(\boldsymbol{K}, \boldsymbol{S}))$.

У доповіді розглядаються варіанти хешування безпосередньо повідомлення M та хешування повідомлення M, на яке накладено псевдовипадкову послідовність.

Узагальнена схема процесу хешування наведена на рис. 1.

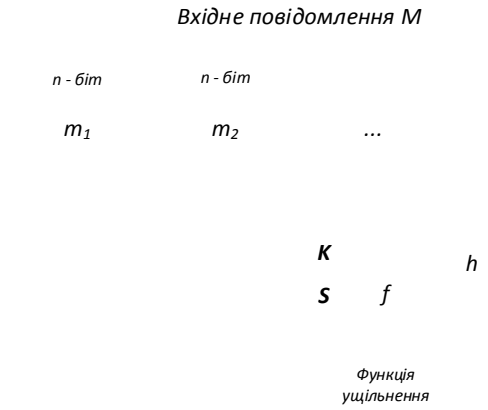

Рис. 1 – Узагальнена схема процесу хешування

III. Висновки

Запропоновано принципово новий метод побудови хеш-функцій, який не передбачає використання ітеративних процедур, а використовує характеристичні ознаки вхідних даних та їх нескладну обробку.

Хешування на основі характеристичних ознак послідовної структури даних значно спрощує та прискорює процес генерування хеш-значення, а також позбавляє отриману хеш-функцію недоліків відомих методів хешування за ітераційною процедурою.

[1] Алферов А. П. Основы криптографии / А. П. Алферов, А. Ю. Зубов, А. С. Кузьмин, А. В. Черемушкин. –М.: Гелиос АРВ, 2001. –479с.

[2] Bernstein D. J. List of SHA-3 candidates measured, indexed by machine / D. J. Bernstein, T. Lange 2011. – Режим доступу до статті: http://bench.cr.yp.to/results-sha3.html.

[3] Лужецький В. А. Узагальнений метод хешування байтової форми представлення інформації / В. А. Лужецький, Д. В. Кисюк // IV міжнародна науково-практична конференція «Інформаційні технології та комп'ютерна інженерія». – Вінниця: ВНТУ, 2014., -275с.

[4] Кутя Е. Ю. Анализ, сравнение и особенности архитектуры функции хеширования BLAKE проекта SHA-3 / Е. Ю. Кутя, И. Д. Горбенко // Прикладная радиоэлектроника: науч.- техн. журнал. – 2012. – Том 11. № 2., - 277 с.

[5] Aumasson J. P. SHA-3 proposal BLAKE / Henzen L., Meier W., Phan R. - 2010.

[6] Liangyu X. Attacks on round-reduced BLAKE / X. Liangyu, L. Ji. - 2009.

[7] Bos J. W. Performance analysis of the SHA-3 candidates on exotic multi-core architectures / J. W. Bos, D. Stefan. - 2010.

[8] Neves S. ChaCha implementation. - 2009. – Режим доступу до статті: http://eden.dei.uc.pt/sneves/chacha/chacha.html.

[9] Knezevic M. Fair and consistent hardware evaluation of fourteen round two SHA-3 candidates / M. Knezevic, K. Kobayashi, J. Ikegami, S. Matsuo, A. Satoh, U. Kocabas, J. Fan, T. Katashita, T. Sugawara, K. Sakiyama, I. Verbauwhede, K. Ohta, N. Homma, T. Aoki // April 2011.