УДК 004.056:004.424.47

МЕТОД ХЕШУВАННЯ ДАНИХ НА ОСНОВІ ЇХ ХАРАКТЕРИСТИЧНИХ ОЗНАК

Лужецький В.А,
проф., д.т.н., завідувач кафедри захисту інформації,
Вінницький національний технічний університет,
lva_zi@mail.ru

Кисюк Д.В.,
асист. кафедри обчислювальної техніки,
Вінницький національний технічний університет,
kneimad @gmail.com

Анотація. *Розглянуто особливості, переваги та недоліки основних сучасних конструкцій криптографічних хеш-функцій. Запропоновано принципово новий підхід до побудови хеш-функцій без використання ітеративної процедури. В основі цього підходу є використання характеристичних ознак даних. Дані розглядаються як послідовність байтів. Ознаками є кількість байтів, що мають певний числовий еквівалент та номери позицій на яких ці байти розташовані. Наведено узагальнену схему неітеративного процесу хешування за запропонованим методом. Описана реалізація окремих етапів обчислень.*

Вступ

Криптографічні хеш-функції є одними з найважливіших видів криптографічних перетворень. 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 обчислювальними 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 ітеративний принцип обчислень, що передбачає поділ повідомлення на блоки, знаходження чергового хеш-значення шляхом спільного оброблення попереднього хеш-значення і блоку даних. Найвідомішими конструкціями хешування є схема Меркеля–Дамгарда [2], структура ітеративного режиму хешування Hash Iterative Framework (HAIFA), конструкція «Криптографічна губка» (Sponge) [6, 7]. Ці конструкції використано при створенні хеш-функцій MD5 і SHA-1, Wide pipe, Double pipe, 3C, BLAKE та Keccak.

Недоліками таких конструкцій є [3, 8]:

1) різний вплив блоків даних на остаточний результат хешування (значення першого блоку бере участь у формуванні всіх проміжних хеш-значень через ітеративність процедури, а значення останнього блоку враховується лише на останній ітерації) [4, 5];

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

Метод неітеративного хешування даних

Вхідне повідомлення M розбивається на послідовність байтів $M = \{ m_1, m_2, \ldots, m_L \}$. 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 K та S:

$K = (k_0, k_1, \dots, k_{255})$, $S = (s_0, s_1, \dots, s_{255})$,

елементи яких визначаються за формулами:

$$k_n^{(l)} = \begin{cases} k_n^{(l-1)} + 1, \text{якщо } m_l = n \\ k_n^{(l-1)}, \text{інакше} \end{cases}, \quad (1)$$

де $k_n^{(0)} = 0$,

$l = 1 \div L$.

$$s_n = \sum_{j=1}^{k_n} l_j^{(n)}. \quad (2)$$

Передбачається хешування даних, максимальний розмір яких 2^{32} байт, а довжина хеш-значення – 256 біт. З урахуванням цього, елементи масиву K представляються набором з 4 байтів:

$k_n = (b_{n,3}, b_{n,2}, \dots, b_{n,0})$.

Масив S складається з 256 елементів по 8 байт кожен:

$$s_n = (b_{n,7}, b_{n,6}, \dots, b_{n,0}).$$

Для підрахунку хеш-коду, спочатку масиви K та S ущільнюються до 128 біт кожен, а потім застосовується певна функція до цих двох значень:

$h = f(C_1(K), C_2(S))$.

Узагальнена схема неітеративного хешування за даним методом наведена на рис. 1. Тут f_K і f_S позначають обчислення за формулами (1) та (2).

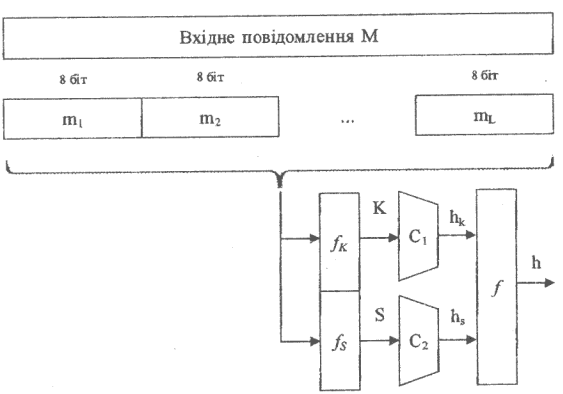

Рис. 1. Узагальнена схема неітеративного процесу хешування

Функція ущільнення C_1 передбачає виконання таких дій.

Масив K перетворюється у масив K*, кожен елемент якого представляється одним байтом:

$K^* = (k_0^*, k_1^*, \dots, k_{255}^*)$,

де $k_n^* = \left(\sum_{i=0}^{3} b_{n,i}\right) mod\ 2^8$.

Далі виконується послідовність процедур ущільнення $K^* \to K^{(128)} \to K^{(64)} \to K^{(32)} \to K^{(16)}$, результатом якої є h_k.

Ці процедури полягають у виконанні обчислень:

$$k_i^{(128)} = (k_{2i}^* \oplus k_{2i+1}^*), (i = 0 \div 127);$$
$$k_j^{(64)} = \left(k_{2j}^{(128)} \oplus k_{2j+1}^{(128)}\right), (j = 0 \div 63);$$
$$k_l^{(32)} = \left(k_{2l}^{(64)} \oplus k_{2l+1}^{(64)}\right), (l = 0 \div 31);$$
$$k_m^{(16)} = \left(k_{2m}^{(32)} \oplus k_{2m+1}^{(32)}\right), (m = 0 \div 15).$$

Функція ущільнення C_2 передбачає виконання таких дій.

Масив S перетворюється у масив S^* кожен елемент якого представляється одним байтом: $S^* = (s_0^*, s_1^*, \dots, s_{255}^*)$, де $s_n^* = \left(\sum_{i=0}^{7} b_{n,i}\right) mod\ 2^8$.

Далі, масив S^*ущільнюється до значення h_s, шляхом виконання такої послідовності процедур: $S^* \to S^{(128)} \to S^{(64)} \to S^{(32)} \to S^{(16)}$.

Функція $f(h_k, h_s)$ передбачає виконання звичайного арифметичного множення 128-розрядних кодів. Таким чином, значення хеш-функції дорівнює: $h = h_k \cdot h_s$.

Результати досліджень запропонованого методу неітеративного хешування з використанням тестів NIST показали, що цей метод забезпечує виконання вимог щодо криптографічної стійкості хеш-функції.

Отримані оцінки алгоритмічної складності реалізації функції хешування свідчать про те, що такий підхід дозволяє прискорити процес хешування до 3,2 рази порівняно з відомими функціями хешування.

Література:

1. Алферов А. П. Основы криптографии / А. П. Алферов, А. Ю. Зубов, А. С. Кузьмин, А. В. Черемушкин // –М.: Гелиос АРВ, 2001. – 479 с.
2. Лужецький В. А. Новий підхід до побудови криптографічних хеш-функцій / В. А. Лужецький, Д. В. Кисюк // «Інформаційні технології та комп'ютерна інженерія»; матеріали статей п'ятої міжнародної науково-практичної конференції, м. Івано-Фран-ківськ, 27-29 травня 2015 року. – Івано-Франківськ: Супрун В.П., 2015 р. – с. 206-208.
3. Лужецький В. А. Узагальнений метод хешування байтової форми представлення інформації / В. А. Лужецький, Д. В. Кисюк // Тези доповідей Четвертої Міжнародної науково-практичної конференції «Інформаційні технології та комп'ютерна інженерія» м. Вінниця, 28-30 травня 2014 року. – Вінниця: ВНТУ, 2014., -275 с.
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.