УДК 004.056

ДОСЛІДЖЕННЯ ТА АНАЛІЗ АЛГОРИТМІВ ХЕШУВАННЯ ДЖЕНКІНСА

Рисований Олександр Миколайович, к.т.н., доц.
Національний технічний університет «Харківський політехнічний інститут»
Україна, Харків
rysov81524@gmail.com

Анотація. *У тезах проаналізовані різні алгоритми Дженкінса. В роботі використовувались різні системи команд: від базової системи команд до команд паралельної обробки даних у скалярному режимі.*
Ключові слова: *алгоритм Дженкінса; алгоритм lookup; алгоритм Jenkins mix; алгоритм SpookyHash.*

Вступна частина. Хешування – це одна з ключових технологій сучасної інформатики та кібербезпеки. Актуальність хешування постійно зростає, оскільки обсяг цифрових даних, необхідність захисту та швидкість обробки постійно збільшуються.

Актуальність застосування хешування обумовлена кількома чинниками [1].

По-перше, у сучасних інформаційних системах постійно зростає потреба у швидкому та надійному способі ідентифікації та перевірки даних. Хеш-функції дозволяють ефективно порівнювати великі обсяги інформації, забезпечуючи високу швидкість доступу та пошуку.

По-друге, хешування є невід'ємною частиною криптографічних механізмів захисту інформації. Воно використовується для зберігання паролів, створення цифрових підписів, формування контрольних сум та перевірки цілісності даних під час передачі незахищеними каналами зв'язку. Сучасні хеш-алгоритми, такі як SHA-2, SHA-3, BLAKE3 та інші, забезпечують високий рівень стійкості до колізій та передбачуваності, що робить їх надійним засобом захисту інформації.

По-третє, зростання популярності блокчейн-технологій та криптовалют ще більше посилило значущість хешування. Саме хеш-функції лежать в основі побудови ланцюжків блоків, формування адрес та перевірки транзакцій, забезпечуючи незмінність та довіру до даних у розподілених системах.

Крім того, хешування активно застосовується в машинному навчанні, штучному інтелекті та аналізі великих даних, де воно використовується для оптимізації зберігання та обробки інформації, скорочення обсягів даних та підвищення продуктивності обчислень.

Таким чином, хешування є фундаментальною технологією сучасної інформатики, що забезпечує безпеку, надійність та ефективність обробки інформації. Його розвиток та вдосконалення залишаються актуальним напрямом досліджень, що мають важливе практичне значення для всіх галузей цифрової економіки.

Розвиток хешування спрямоване на підвищення швидкості, стійкості до колізій та криптографічної стійкості, що робить хешування фундаментальною частиною цифрової інфраструктури XXI століття.

Метою роботи є аналіз алгоритмів хешування Дженкінса [2-3] для архітектури мікропроцесора x64 у середовищі програмування masm64 з метою використання сучасних інструкцій SSE та AVX паралельної обробки для ще меншого часу отримання хешів.

Основна частина. Хеш (hash) – це унікальний цифровий відбиток, який можна присвоїти будь-якому файлу.

Хеш-функція має властивості:

- детермінованості: для одного і того ж входу завжди генерується той самий хеш;

– швидкість обчислення: хешування має бути ефективно обчислюваним для будь-яких вхідних даних;

– односторонності: неможливо відновити вхідні дані, знаючи лише їх хеш;

– стійкості до колізій: складно знайти два різні вхідні значення, які мають однаковий хеш.

Хеш-функція Jenkins (також відома як Jenkins One-at-a-Time Hash) має високу швидкість роботи і відмінний лавинний ефект.

Алгоритм складається з послідовних операцій зсуву в різні сторони на різне число розрядів та виконання операцій суми та суми за модулем 2. Ці операції отримані в результаті багаторазової зміни даних та аналізу отриманих результатів щодо швидкості виконання і лавинного процесу.

Реалізація алгоритму отримання хеш-функції для прикладу, коли вводиться число 0, наведено нижче:

A ← 30h = код числа 0
B := B + A ; 0000 0030
A ← 10 p ; 0000 C000
B := B + A ; 0000 C030
A → 6 p ; 0000 0300
B := B ⊕ A ; 0000 C330
A ← 3 p ; 0006 1980
B := B + A ; 0006 DCB0
A → 11 p ; 0000 00DB
B := B ⊕ A ; 0006 DC6B
A ← 15 p ; 6E35 8000
A := A + B ; 6E3C 5C6B

Блок коду на мові Асемблере в середовищі masm64, який реалізує цей алгоритм може бути таким:

```
        xor rbx, rbx ;
        lea rsi, Buf1 ;
        mov rcx, Amount ;
        dec rcx             ; ignore \n
        dec rcx             ; ignore \r
    .loop:
        lodsb               ; ax = Buf1[i]
        movzx rax, ax

        add rbx, rax        ; hash +=  Buf1[i]

        mov rdx, rbx
        shl rdx, 10         ; hash << 10
        add rbx, rdx        ; hash +=  hash << 10
```

```
    mov rdx, rbx
    shr rdx, 6          ; hash >> 6
    xor rbx, rdx        ; hash ^=  hash >> 6
  loop .loop

    mov rdx, rbx
    shl rdx, 3          ; hash << 3
    add rbx, rdx        ; hash ^=  hash << 3

    mov rdx, rbx
    shr rdx, 11         ; hash >> 11
    xor rbx, rdx        ; hash ^=  hash >> 11
   mov rdx, rbx
   shl rdx, 15          ; hash << 15
   add rbx, rdx         ; hash +=  hash << 15
 mov Result, rbx
```

Існують інші варіанти алгоритмів хешування Jenkins Hash:

– lookup2 / lookup3 / lookup8 – складніші функції для 32/64-бітних даних;

– Jenkins mix() – використовується всередині більших хеш-алгоритмів;

- SpookyHash - вдосконалений спадкоємець, також розроблений Дженкінс.

Алгоритм Jenkins mix використовує функцію змішування mix. При цьому здійснюється перемішування внутрішніх 32 або 64-бітних значення так, щоб кожен біт входу впливав на кожен біт виходу. Це забезпечує рівномірний розподіл хешів, мінімізацію колізій та стійкість до локальних змін даних.

Алгоритм SpookyHash – це сучасна, дуже швидка некриптографічна хеш-функція. Цей алгоритм є дуже швидким, має мінімальні колізії. Він призначений для хешування великих обсягів даних з високою швидкістю та рівномірним розподілом, особливо на 64-бітових архітектурах. При цьому для 64-бітових систем використовується ідея перемішування mix, але вже з 12 змінними та іншими константами обертання.

Висновки. У тезах проаналізовані різні алгоритми Дженкінса. В роботі використовувались різні системи команд: від базової системи команд до команд паралельної обробки даних у скалярному режимі. Для кожної системи команд за допомогою інструкції rdtsc отримувався час виконання основного блоку коду й зроблені відповідні висновки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Eastlake, D., & Jones, P. (2001). *US Secure Hash Algorithm 1 (SHA-1).* RFC 3174, Internet Engineering Task Force (IETF). https://www.rfc-editor.org/rfc/rfc3174.

[2] Jenkins, B. (2011). *SpookyHash: a 128-bit noncryptographic hash.* Bob Jenkins' official documentation. http://burtleburtle.net/bob/hash/spooky.html.

[3] BLAKE3 Team. (2020). *BLAKE3: One function, fast everywhere*. Official white paper. https://github.com/BLAKE3-team/BLAKE3-specs.

Rysovanyi O.M.

DEVELOPMENT AND ANALYSIS OF ALGORITHMS IN KHESHUVANNYA JENKINS

Abstract. *Theses analyzes various Jenkins algorithms. In the robot, various command systems were developed: from the basic command system to commands for parallel data processing in the scalar mode.*

Keywords: *Jenkins algorithm; lookup algorithm; Jenkins mix algorithm; SpookyHash algorithm.*