Łamanie MD5

Obliczenia równoległe na kartach graficznych CUDA

Rafał Leja (UWr)

29 stycznia 2026

Plan

1. Tło: MD5 i problem do rozwiązania.
2. Implementacja: CPU vs GPU.
3. Benchmarki.
4. Podsumowanie.

MD5

Funkcja skrótu MD5 (Message-Digest Algorithm 5)
$h : 0,1^* \to 0,1^{128}$ 128-bitowy skrót (hash) z dowolnej długości wejścia. 128-bitowy stan wewnętrzny

MD5 do przechowywania haseł

- W latach 1990 - 2000 standard do haszowania haseł
- Główna zaleta - szybkość obliczeń
- Około 2004 odkryto, że MD5 jest podatne na kolizje.
- W rezultacie, wiele systemów zaczęło migrować do bezpieczniejszych algorytmów, takich jak SHA-1* i SHA-256.
- *SHA-1 również okazało się podatne na kolizje.

MD5 do przechowywania haseł

W 2019 ponad ćwierć CMS-ów używało MD5 do haszowania haseł użytkowników.

MD5 - padding

- Dane wejściowe są dzielone na bloki 512-bitowe
- Jeśli ostatni blok jest krótszy niż 512 bitów, stosowany jest padding
  - Dodanie bitu '1' na końcu danych
  - Dodanie bitów '0' aż do osiągnięcia długości 448 bitów
  - Dodanie 64-bitowej reprezentacji długości oryginalnych danych

MD5 - inicjalizacja

- MD5 używa czterech 32-bitowych słów jako stanu wewnętrznego:
  - A = 0x67452301
  - B = 0xEFCDAB89
  - C = 0x98BADCFE
  - D = 0x10325476
- Te wartości są inicjalizowane na początku procesu haszowania.

MD5 - funkcje pomocnicze

MD5 używa czterech nieliniowych funkcji bitowych:

- F(X, Y, Z) = (X AND Y) OR (NOT X AND Z)
- G(X, Y, Z) = (X AND Z) OR (Y AND NOT Z)
- H(X, Y, Z) = X XOR Y XOR Z
- I(X, Y, Z) = Y XOR (X OR NOT Z)

MD5 - główna pętla

- Dla każdego 512-bitowego bloku danych:
  - Podziel blok na szesnaście 32-bitowych słów M[0..15]
  - Wykonaj 64 rundy operacji, podzielone na cztery fazy po 16 rund każda
  - W każdej rundzie użyj jednej z funkcji F, G, H, I oraz odpowiedniego słowa M[i] i stałej K[i]

MD5 - aktualizacja stanu

- Po przetworzeniu każdego bloku, zaktualizuj stan wewnętrzny:
  - $A = A + AA$
  - $B = B + BB$
  - $C = C + CC$
  - $D = D + DD$
- Gdzie AA, BB, CC, DD to wartości stanu przed przetworzeniem bieżącego bloku.

MD5 - wynik końcowy

- Po przetworzeniu wszystkich bloków, wynikowy hash to konkatenacja A, B, C, D
- Wynik jest reprezentowany jako 32-znakowy ciąg szesnastkowy.

Implementacja - MD5

```
for (int i = 0; i < 64; i++) {
    uint32_t F, g;
    switch (i / 16) {
    case 0:
        F = (b & c) | (~b & d);
        g = i;
        break;
    case 1:
        F = (d & b) | (~d & c);
        g = (5 * i + 1) % 16;
        break;
    case 2:
        F = b ^ c ^ d;
        g = (3 * i + 5) % 16;
        break;
    case 3:
        F = c ^ (b | ~d);
        g = (7 * i) % 16;
        break;
    }

    F = F + a + K_d[i] + M[g];
    a = d;
    d = c;
    c = b;
    b = b + ((F << S_d[i]) | (F >> (32 - S_d[i])));
}
```

Implementacja - padding

implementacja - generowanie haseł

Implementacja - CUDA

Źródła I