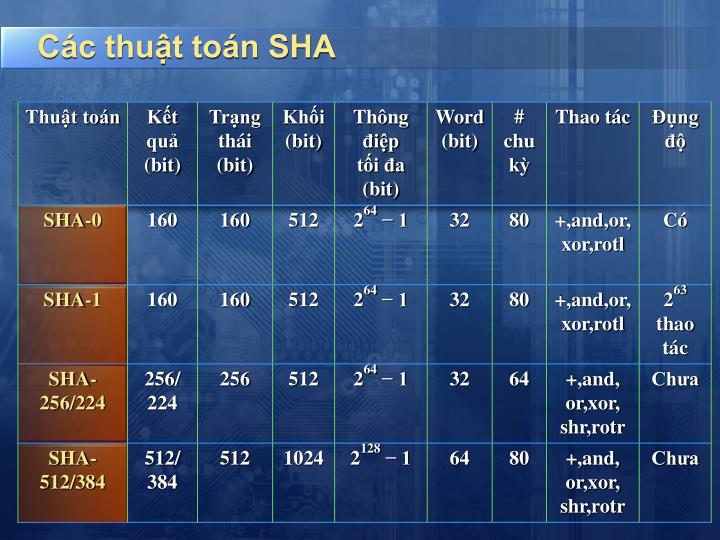
**SHA** (Secure Hash Algorithm hay [thuật giải băm](https://vi.wikipedia.org/wiki/H%C3%A0m_b%C4%83m) an toàn) là năm [thuật giải](https://vi.wikipedia.org/wiki/Thu%E1%BA%ADt_to%C3%A1n) được chấp nhận bởi [FIPS](https://vi.wikipedia.org/w/index.php?title=FIPS&action=edit&redlink=1) dùng để chuyển một đoạn dữ liệu nhất định thành một đoạn dữ liệu có chiều dài không đổi với xác suất khác biệt cao

Secure Hash Algorithm (SHA) phát triển bởi National Institute of Standard and Technology (NIST)

Đầu vào: thông điệp với độ dài tối đa 264 bits

Đầu ra: giá trị băm (message digest) có độ dài 160 bits

Giải thuật gồm 5 bước thao tác trên các khối 512 bits

Về cơ bản SHA1 rất giống với SHA0 nhưng sửa chữa một lỗi trong đặc tả kỹ thuật băm SHA ban đầu dẫn đến những điểm yếu đáng kể. các thuật toán SHA0 không được chấp nhận bởi nhiều ứng dụng.

Giải thuật SHA-1 – Nguyên lý

Bước 1: nhồi thêm dữ liệu

.Thông điệp được nhồi thêm các bits sao cho độ dài l ≡ 448 mod 512 hay l = n \* 512 + 448 (n,l nguyên)

.Thông điệp luôn luôn được nhồi thêm dữ liệu

.Số bits nhồi thêm nằm trong khoảng 1 đến 512

.Phần dữ liệu nhồi thêm bao gồm một bit 1 v à theo sau l à c ác bit 0

Bước 2: thêm vào độ dài

Độ dài của khối dữ liệu ban đầu được biểu diễn dưới dạng nhị phân 64-bit và được thêm vào cuối chuỗi nhị phân kết quả của bước 1

Độ dài được biểu diễn dưới dạng nhị phân 64-bit không dấu

Kết quả có được từ 2 bước đầu là một khối dữ liệu có độ dài là bội số của 512. Khối dữ liệu được biểu diễn:

Bằng một dãy L khối 512-bit Y­­­­­­­­­­­­0, Y1,…, YL-1

Bằng một dãy N từ (word) 32-bit M0, M1, MN-1. Vậy N = L x 16

Bước 3: khởi tạo bộ đệm MD (MD buffer)

Một bộ đệm 160-bit được dùng lưu trữ các giá trị băm trung gian và kết quả. Bộ đệm được biểu diễn bằng 5 thanh ghi 32- bit với các giá trị khởi tạo ở dạng big-endian (byte có trọng số lớn nhất trong từ nằm ở địa chỉ thấp nhất) như sau:

A = 01 23 45 67

B = 89 AB CD EF

C = FE DC BA 98

D = 76 54 32 10

E = C3 D2 E1 F0

Các giá trị này tương đương với các từ 32-bit sau:

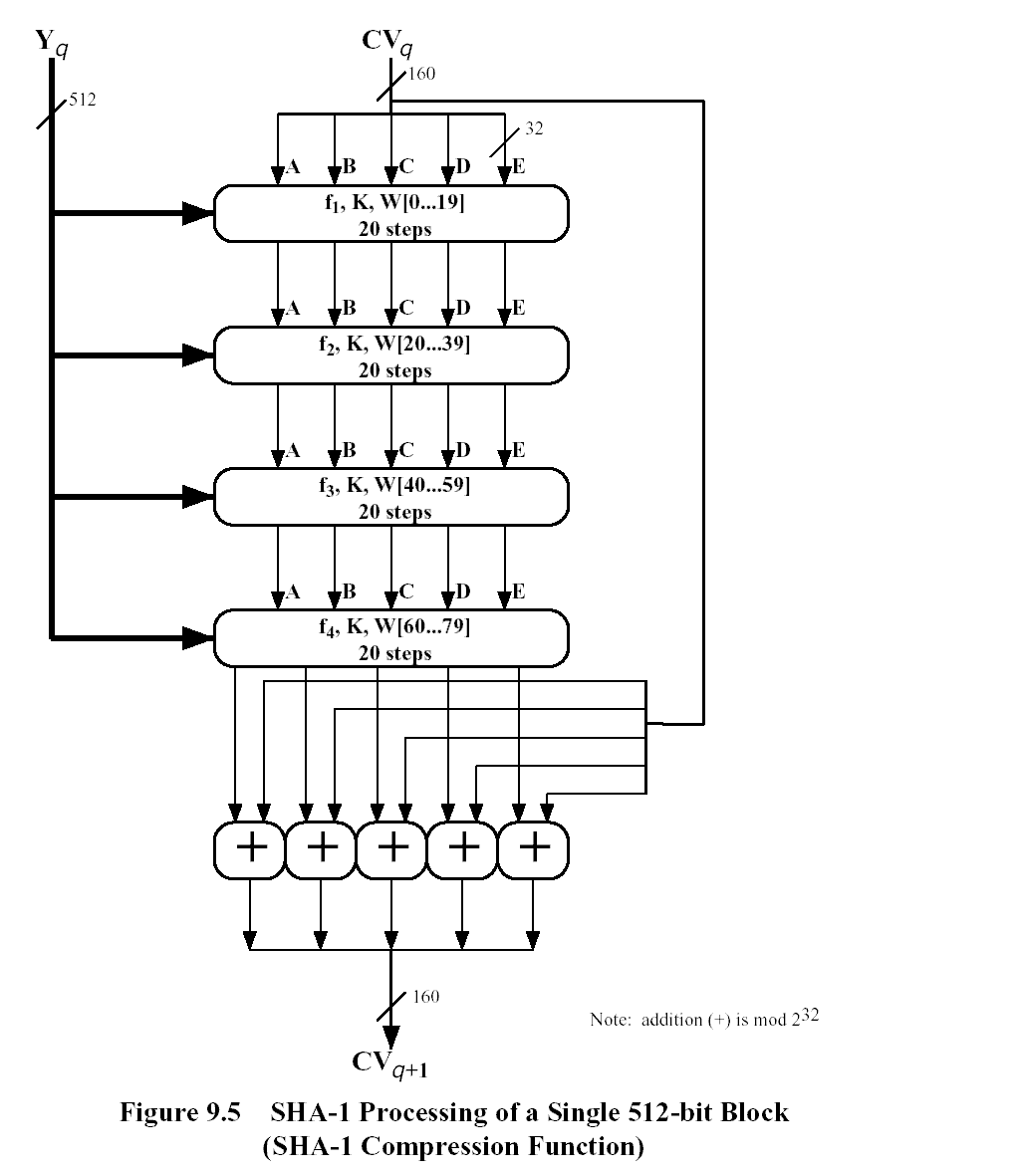
A = 01 23 45 67

B = 89 AB CD EF

C = FE DC BA 98

D = 76 54 32 10

E = C3 D2 E1 F0

Bước 4: xử lý các khối dữ liệu 512-bit 

Trọng tâm của giải thuật bao gồm 4 vòng lặp thực hiện tất cả 80 bước.

4 vòng lặp có cấu trúc như nhau, chỉ khác nhau ở các hàm logic f1, f2, f3, f4

Mỗi vòng có đầu vào gồm khối 512-bit hiện thời và một bộ đệm 160-bit ABCDE. Các thao tác sẽ cập nhật giá trị bộ đệm

Mỗi bước sử dụng một hằng số Kt (0 ≤ t ≤ 79)

Kt = 5A827999 (0 ≤ t ≤ 19)

Kt = 6ED9EBA1 (20 ≤ t ≤ 39)

Kt = 8F1BBCDC (40 ≤ t ≤ 59)

Kt = CA62C1D6 (60 ≤ t ≤ 79)

Đầu ra của 4 vòng (bước 80) được cộng với đầu ra của bước CVq để tạo ra CVq+1

Bước 5: xuất kết quả

Sau khi thao tác trên toàn bộ L blocks. Kết quả của khối thứ L là bảng băm 160-bit

Giải thuật được tóm tắt như sau:

CV 0 = IV

CVq+1 = SUM32(CV q ,ABCDE q )

MD = CV L

Với:

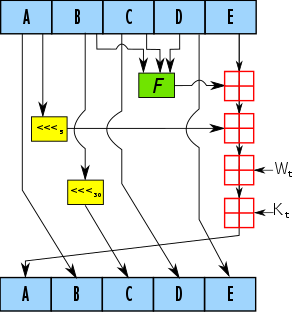
IV = giá trị khởi tạo của bộ đệm ABCDE

ABCDEq = đầu ra của hàm nén trên khối thứ q

L = số khối 512-bit của thông điệp

SUM32 = phép cộng modulo 232 trên từng từ (32 bits) của đầu vào

MD = giá trị băm



Giải thuật thực hiện tất cả 80 bước, mỗi bước được mô tả như sau:

A ← E + f(t, B, C, D) + S 5(A) + Wt + Kt

B ← A

C ← S30(B)

D ← C

E ← D

Trong đó

A,B,C,D,E = các từ trong bộ đệm

t = số thứ tự của bước

F(t,B,C,D) = làm logic tại bước t

Sk = dịch vòng trái k bits

Wt = từ thứ t của khối dữ liệu

Kt = hằng số

+ = phép cộng modulo 232

Các hàm f

|  |  |  |
| --- | --- | --- |
| Bước | Hàm | Giá trị |
| (0 ≤ t ≤ 19) | f1 = f(t,B,C,D) | (B ˄ C) ˅ ( ̚ B ˄ D) |
| (20 ≤ t ≤ 39) | f2 = f(t,B,C,D) | B xor C xor D |
| (40 ≤ t ≤ 59) | f3 = f(t,B,C,D) | (B ˄ C) ˅ (B ˄ D) ˅ (C ˄ D) |
| (60 ≤ t ≤ 79) | f4 = f(t,B,C,D) | B xor C xor D |

• Từ 16 từ 32-bit từ khối dữ liệu đầu vào, mở rộng thành 80 từ Wt

– Với 0 ≤ t ≤ 15, giá trị Wt lấy trực tiếp từ khối dữ liệu

– Với t > 15 : Wt = S 1(Wt-16 xor Wt-14 xor Wt-8 xor Wt-3)