SHA-256算法描述

**输入**：sha256的输入是最大长度小于2^64位的消息

**输出**：输出为256位的消息摘要

算法以512位的分组为单位进行处理，结构如下图：

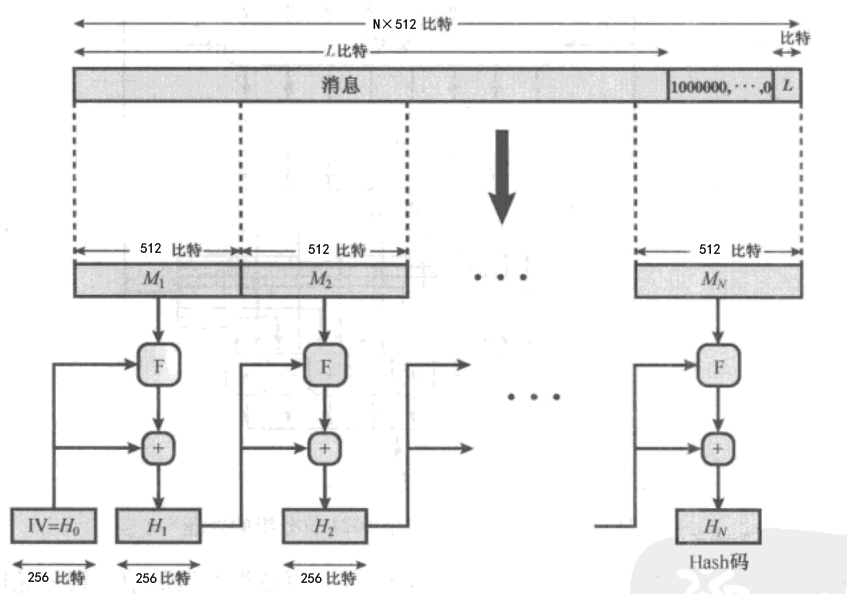


图1

步骤1：**附加填充位。**填充消息使其长度模512与448同余（除以512余数为448），即使消息已经满足上述长度要求，仍然需要进行填充，因此填充位数在1~512之间。填充由一个1和后续的0组成。

步骤2：**附加长度。**在消息后附加一个64为的块，这64bits记录的是填充前消息的长度。附加长度以后，消息的总长度刚好是512的倍数。

步骤3：**初始化hash缓冲区。**hash函数的中间结果和最终结果保存于512位的缓冲区中，缓冲区用8个32位bit的寄存器（a,b,c,d,e,f,g,h）表示，并将这些寄存器初始化为下列32为的整数（十六进制值）：//sha256固定初始值

a = 0x6a09e667, b = 0xbb67ae85,

c = 0x3c6ef372, d = 0xa54ff53a;

e = 0x510e527f, f = 0x9b05688c,

g = 0x1f83d9ab, h = 0x5be0cd19;

这些值以高位在前格式存储，也就是说，字的最高有效字节存于底地址字节位置（最左边）。这些字由前8个质数的平方根的小数部分的前32位组成（设计者使然）。

步骤4：**F模块。**以512位的分组（64字节）为单位处理消息。算法的核心是一个记为F的模块，F有64轮运算，如上图1所示，每处理一个分组均调用一次F模块。F模块的内容如图2所示。

初始化后的缓冲区abcdefgh的值即为，每轮都把缓冲区abcdefgh作为输入，并更新缓冲区。每一轮，如第t轮，使用一个32位的值，该值由当前模块被处理的512位消息分组导出（导出算法将放在后边讨论），每一轮还将使用附加的轮常数,其中0<=t<=63。附加轮常数如图3所示。

第64轮的输出和第1轮的输入相加产生。缓冲区中的8个字和中对应的字分别进行模2^32的加法运算。（最高32位，超出部分抹去）

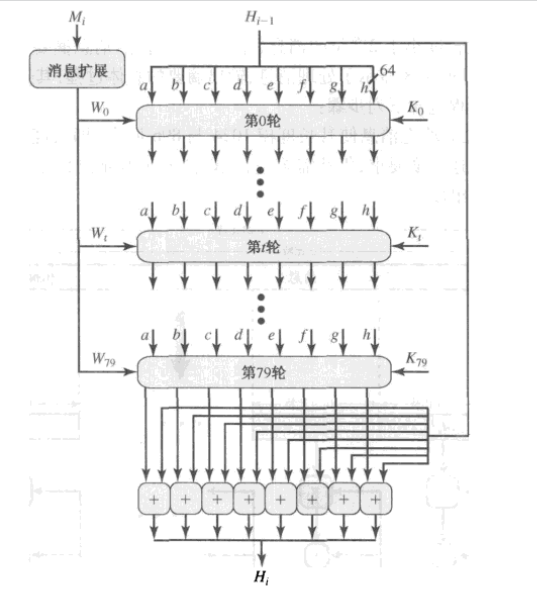


图2

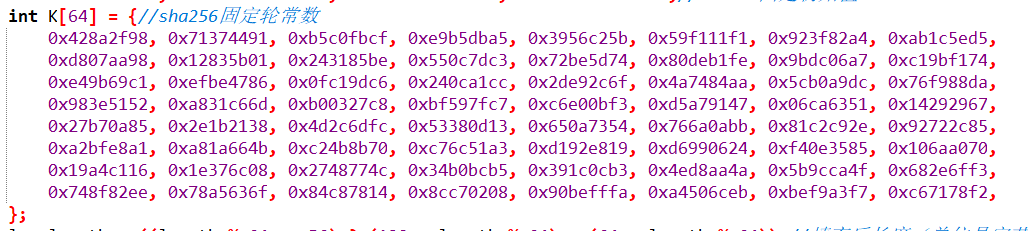


图3

步骤5：**输出。**所有的N个分组都处理完后，第N个阶段输出的是256位的信息摘要。

**sha256的F模块：**

**{**

**//轮函数：**

For t=0 to 63:

{

T1 = h + E1(e) + CH(e,f,g) + Kt + Wt

T2 =E0(a) + MAJ(a,b,c)

h = g

g = f

f = e

e = d + T1

d = c

c = b

b = a

a = T1 + T2

}**end轮函数**

**//计算中间散列值**

0 = a + 0

1 = b + 1

2 = c + 2

3 = d + 3

4 = e + 4

5 = f + 5

6 = g + 6

7 = h + 7

**//** CH( x, y, z) = (x ∧ y) ⊕ ( ¬ x ∧ z) 条件函数：如果e，则f，否则g

**//** MAJ( x, y, z) = (x ∧ y) ⊕ (x ∧ z) ⊕ (y ∧ z) 函数为真当且进档变量的多数（2或3个）为真。

**//** E1(x) = ROTR(x,6) ⊕ ROTR(x,11) ⊕ ROTR(x,25)

**//** E0(x) = ROTR(x,2) ⊕ ROTR(x,13) ⊕ ROTR(x,22)

**//**ROTR(a,b), 把a循环右移b位

**//**“+”为模2^32的加法（最高32位，超出部分抹去）

**}end F模块**

**导出算法：**

该值由当前模块被处理的512位消息分组导出64个，0<=t<=63。

512位消息可以分为16个32位的字，依次赋给前16个~

从开始，有：

**//**O1(x) = ROTR(x,17) ⊕ ROTR(x,19) ⊕ SHR(x,10)

**//** O0(x) = ROTR(x,7) ⊕ ROTR(x,18) ⊕ SHR(x,3)

**//** SHR(a,b)把a右移b位，左端补0

**//**“+”为模2^32的加法（最高32位，超出部分抹去）