消息认证与哈希函数

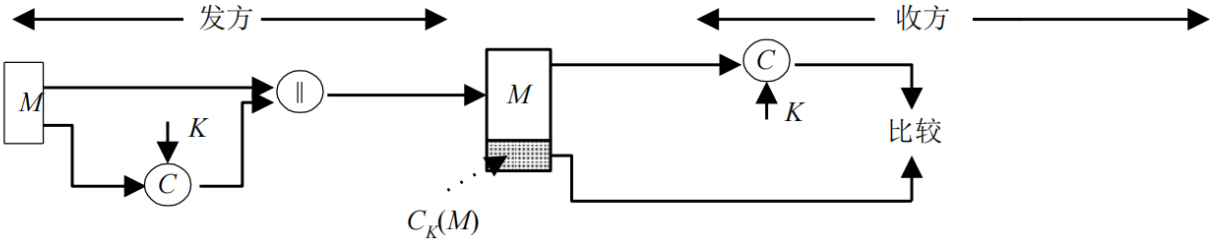
1 消息认证

消息认证码MAC

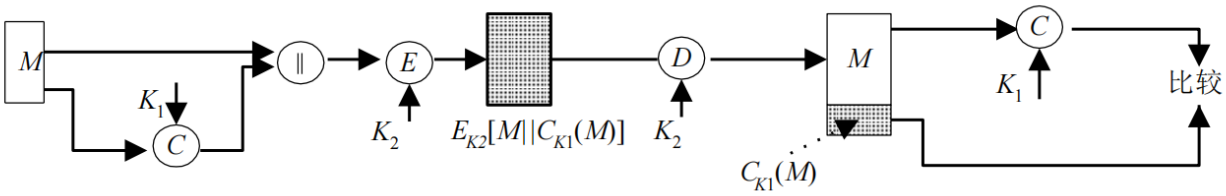
消息被一个密钥（通信双方共享）控制的公开函数作用后产生的、用作认证符的、固定长度的数值。

产生过程

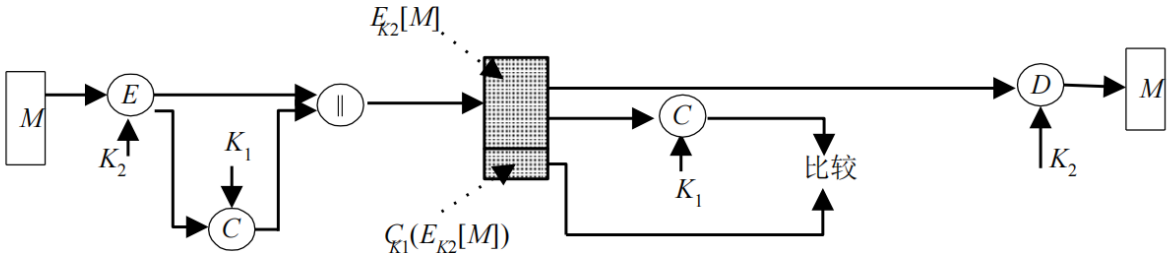
假设A打算发送给B消息M，通信双方共享密钥K，A计算$$MAC=C_K(M)$$，然后向B发送$M||MAC$，B收到后做相同计算，得到另一个MAC，对比二者是否相同。



(a) 消息认证



(b) 认证性和保密性：对明文认证



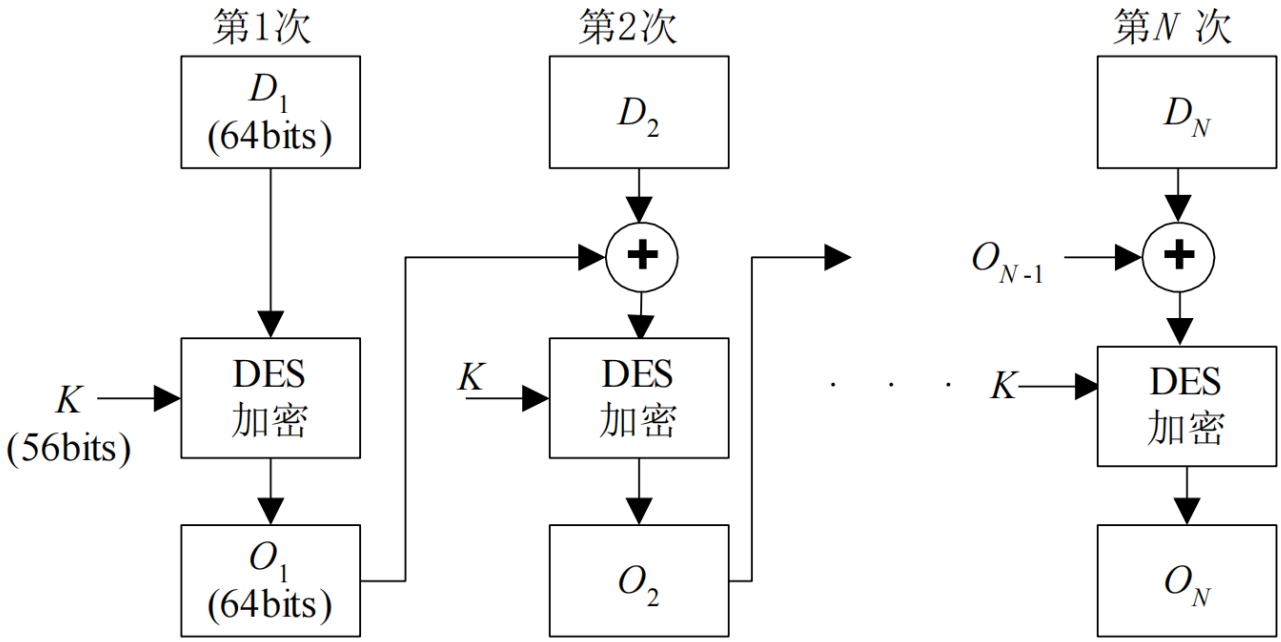
(c) 认证性和保密性：对密文认证

MAC函数**不必是可逆的**，一般都为多到一映射，即已知MAC无法还原出M。

数据认证算法

算法基于**CBC模式的DES算法**，初始向量为零向量，将消息分为64比特长的分组$D_1,\dots,D_N$按以下过程计

算数据认证码：$$O_1=E_K(D_1)\O_2=E_K(D_2\oplus O_1)\ \dots \O_N=E_K(D_N\oplus O_{N-1})$$



2 哈希函数

2.1 **定义**

哈希函数H为公开函数，用于**将任意长的消息M**映射为**固定长度**、较短的一个值H(M)，作为**认证符**，称H(M)为**哈希值**或**消息摘要。**
提供**完整性**保障，即改变消息中的任何比特都会使哈希码发生改变。（类似“指纹”）

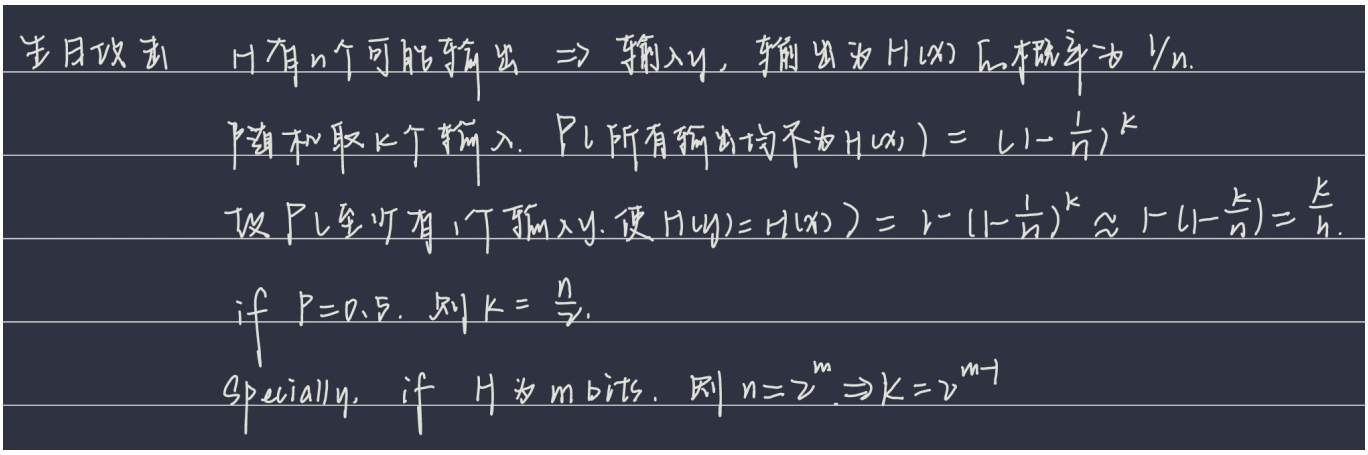
2.2 哈希函数应满足的条件

为了实现对数据的认证，哈希函数应满足以下条件：

- 输入任意长
- 输出定长
- 求H(x)容易
- **单向性**：已知h，求x使得$H(x)=h$在计算上不可行，称H(x)为单向哈希函数。 **（抗原像攻击）**
- **弱单向哈希函数**：已知x，找出y（$y\neq x$）使得$H(y)=H(x)$在计算上不可行。 **（抗第二原像攻击）**
- **强单向哈希函数**：找出任意两个不同的输入x、y，使得$H(y)=H(x)$在计算上不可行。 **（抗碰撞攻击/生日攻击）**
  碰撞性：函数对不同的输入产生相同的输出。

2.3 生日攻击

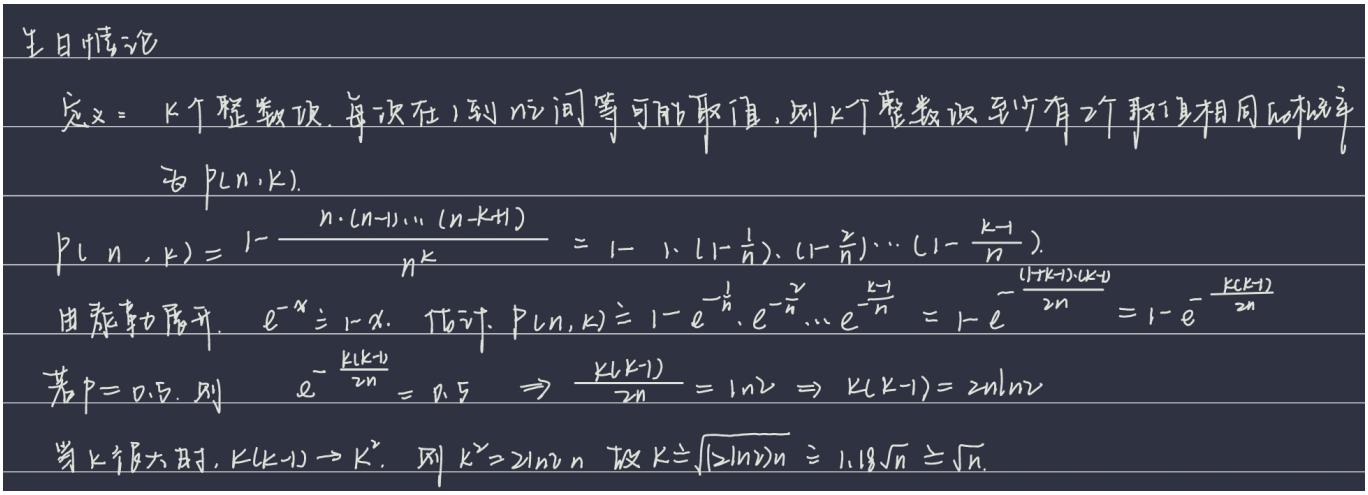
问题描述
已知一哈希函数H有n个可能的输出，H(x)是一个特定的输出，如果对H随机取k个输入，则至少有一个输入y使得H(y)=H(x)的概率为0.5时，k有多大？ 称对哈希函数H寻找上述y的攻击为第I类生日攻击。



抗第二原像攻击：找1人与某给定人的生日相同，时间复杂度为n。

生日悖论

在k个人中至少有两个人的生日相同的概率大于0.5时，k至少多大？ 称寻找函数H的具有相同输出的两个任意输入的攻击方式为第II类生日攻击。



抗碰撞攻击：找任意两人生日相同，复杂度为$\sqrt n$。

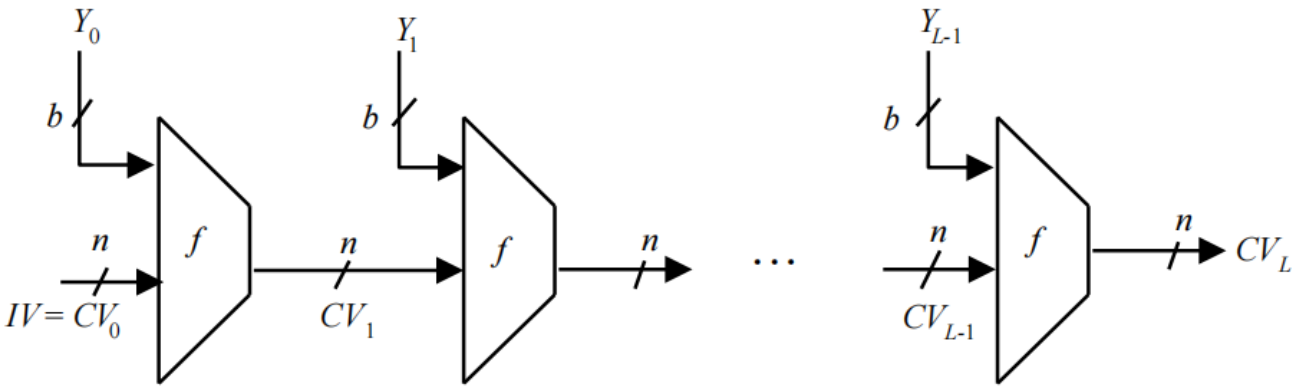
生日攻击

生日攻击基于以下结论：

> 若哈希函数输出长为m比特，如果H的k个随机输入中至少有两个产生相同输出的概率大于0.5，则$k=\sqrt {2^m}=2^{m/2}$

2.4 迭代型哈希函数的一般结构

函数的输入M被分为L个分组$Y_0,\dots,Y_{L-1}$，每个分组的长为b比特，最后一组长度不够需要进行填充。最后一个分组中包括**整个函数输入的长度值。**



算法表达如下：$$CV_0=IV=n\ CV_i=f(CV_{i-1},Y_{i-1})\quad 1\leq i\leq L\ H(M)=CV_L$$
核心：设计**无碰撞**的压缩函数f。

3 MD5哈希算法

3.1 算法描述

采用迭代型哈希函数的一般结构
输入：任意长比特
分组：512比特
输出：128比特

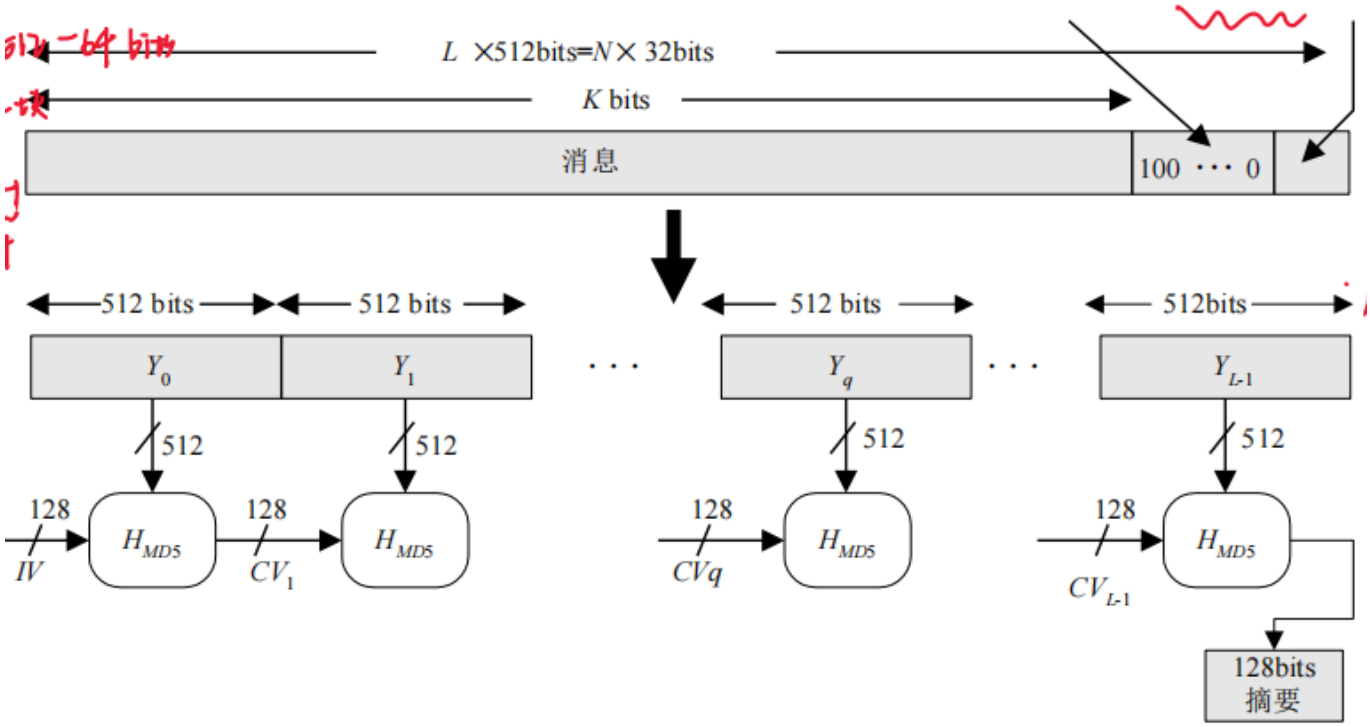
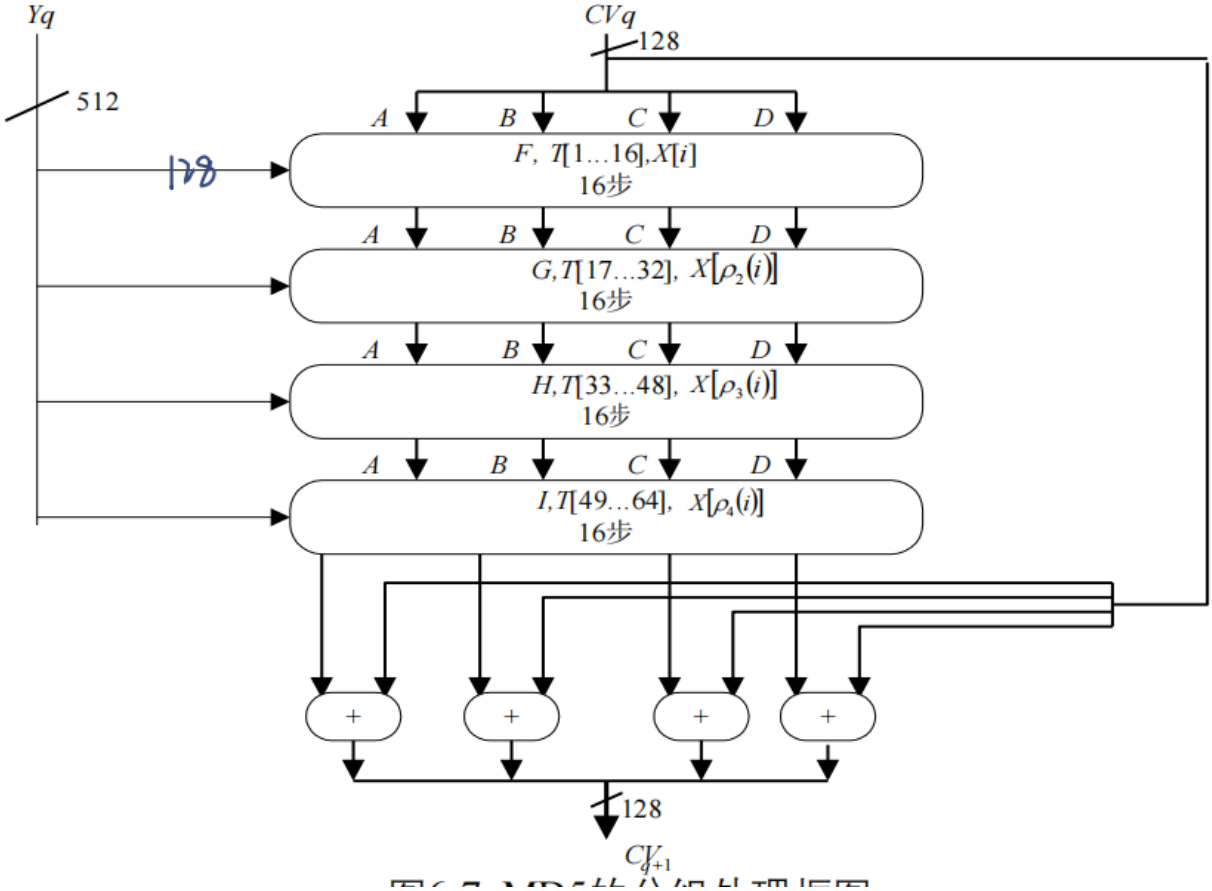


图6-6 MD5的算法框图

处理过程

1. 填充消息
   填充后消息的长度为512的某一倍数减64。
   任何一个消息都需要填充
2. 附加消息的长度
   将填充前消息的长度，用**小端**方式表示，若长度大于64比特，则模2^{64}，放在1中留的64比特处。
   分组，512比特一组，$Y_0,\dots,Y_{L-1}$

3. 对MD缓冲区初始化
   128比特长的缓冲区——4个32比特（1字）的寄存器（A,B,C,D），每个寄存器存入A=01234567，B=89ABCDEF，C=FEDCBA98，D=76543210，实际上A为67452301
4. 以分组为单位对消息进行处理
   每一组经压缩函数H_{MD5}处理。
   H_{MD5}包括4轮处理，每轮对ABCD进行16次迭代。 4轮的流程图如下：



   - 每轮的输入为当前处理的消息分组和缓冲区的当前值A、B、C、D，输出仍放在缓冲区中以产生新的A、B、C、D。
   - 每轮处理过程还需加上常数表T中四分之一个元素，分别为$T[1..16], T[17..32], T[33..48], T[49..64]$。
   - 第4轮的输出再与第1轮的输入**按字模2^{32}相加**，结果即为压缩函数的输出。$$CV_0=IV\ CV_{q+1}=CV_{q}+RF_I[Y_q,RF_G[Y_q,RF_g[Y_q,RF_F[Y_q,CV_q]]]]\ MD=CV_L$$

3.2 压缩函数

迭代过程的流程图如下：

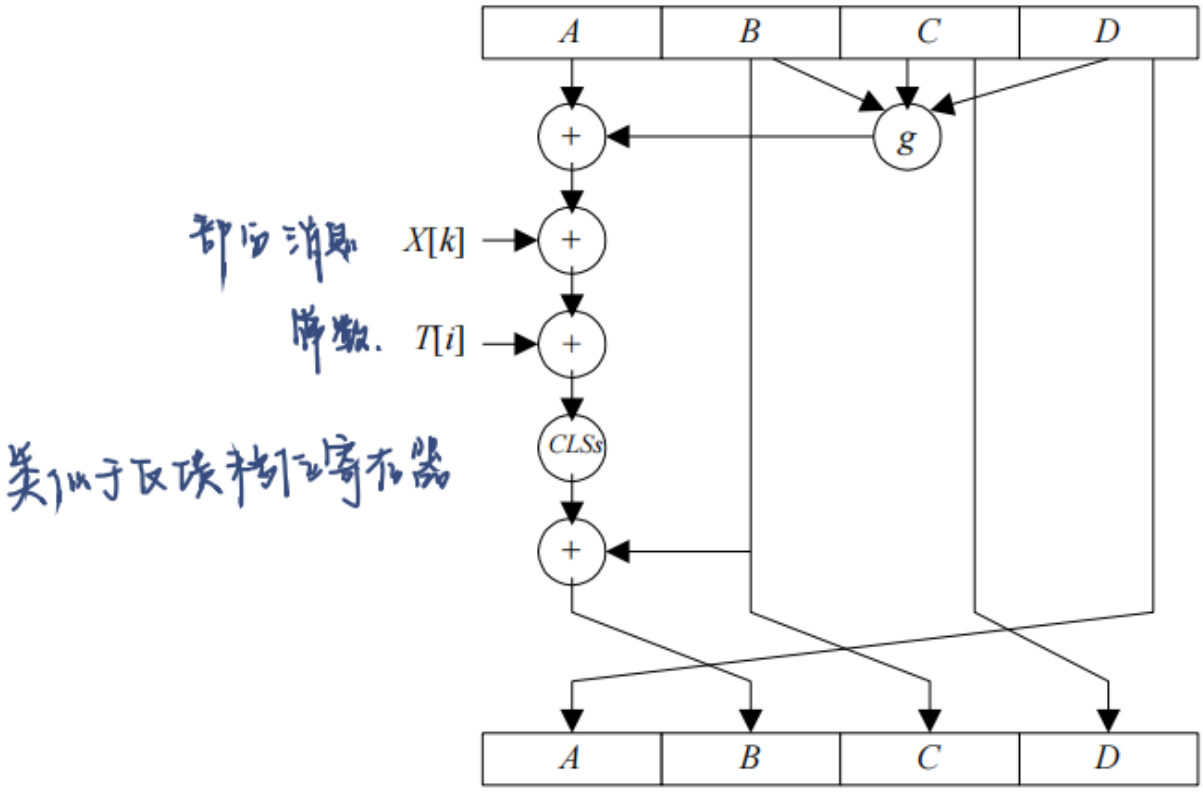


图6-8 压缩函数中的一步迭代示意图

$X[k]=M[q\times 16 + k]$即消息第q个分量中的第k个字。
在4轮处理过程中，每轮以不同的次序使用16个字，在第一轮以字的初始次序使用。第二轮到第四轮对字的次序i做置换后得到一个新次序，然后以新次序使用16个字。
4轮处理过程，使用不同的基本逻辑函数F,G,H,I，每个逻辑函数的输入为3个32比特的字（bcd），输出为1个32比特的字。其中的运算定义为**逐比特**的。

4 SHA安全哈希函数

SHA(Secure Hash Algorithm)

4.1 算法描述

输入：小于2^{64}比特长的任意消息
分组：512比特
输出：160比特

处理过程与MD5相似
消息填充、附加长度（大端序）、缓冲区初始化（5个寄存器）、分组处理（4轮，每轮20次迭代）、输出

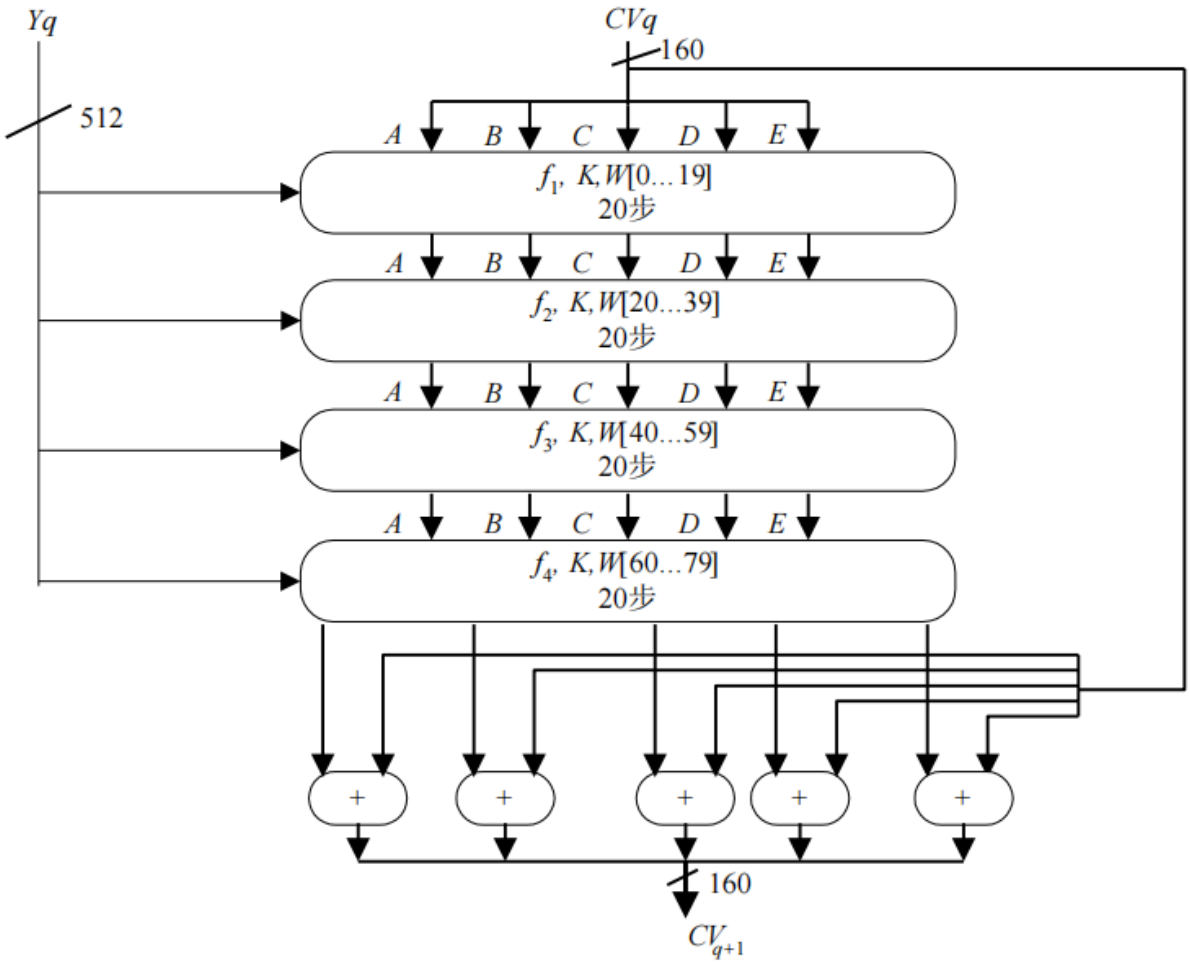


图6-9 SHA的分组处理框图

4.2 压缩函数

$$A,B,C,D,E \leftarrow (E+f_t(B,C,D)+CLS_5(A)+W_t+K_t,A,CLS_30(B),C,D,$$
其中，t为迭代步数$0\leq t\leq 79$，CLS_s为循环左移s位，W_t是由当前分组导出的一个32比特长的字。

一次迭代的流程图如下：

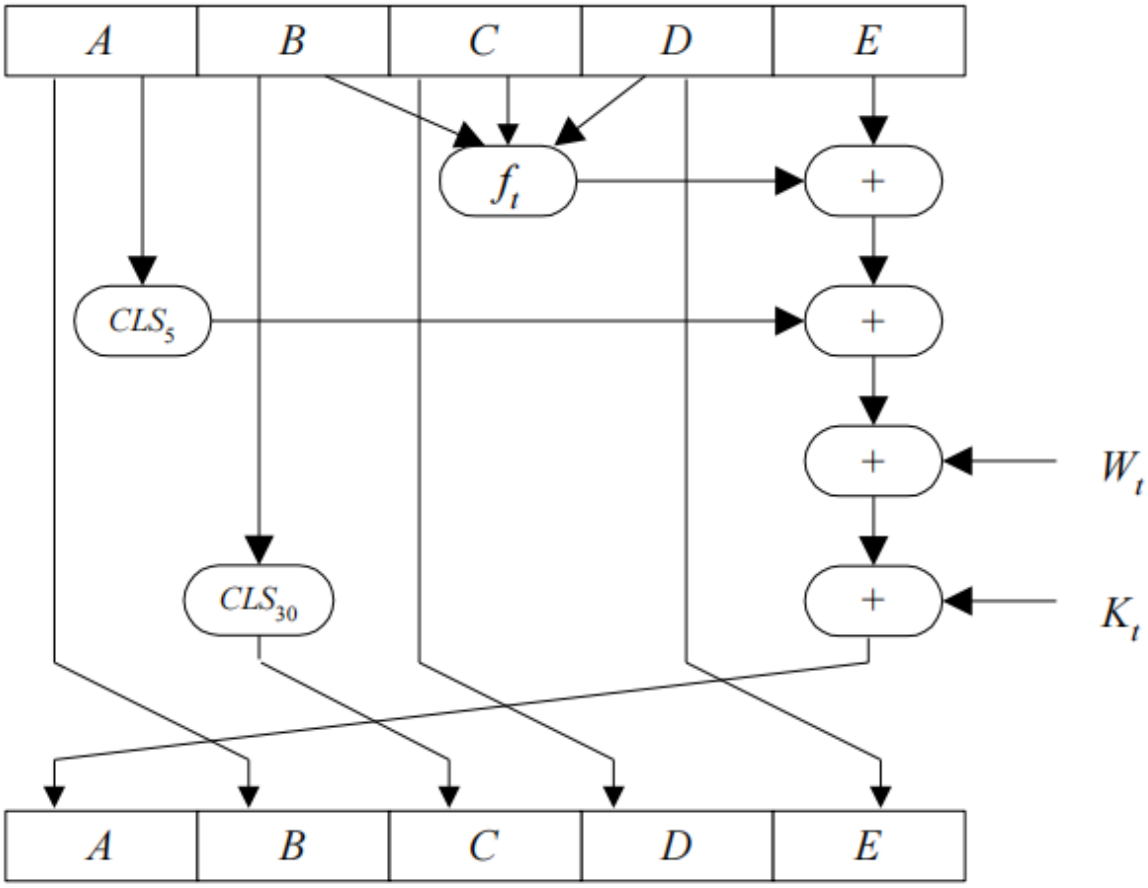


图6-10 SHA的压缩函数中一步迭代示意图

导出W_t方法——按LFSR方式

前16个值：$W_0,W_1,\dots,W_{15}$；

其余值：$W_t = CLS_1(W_{t-16}\oplus W_{t-14}\oplus W_{t-8}\oplus W_{t-3})$

产生过程如下：

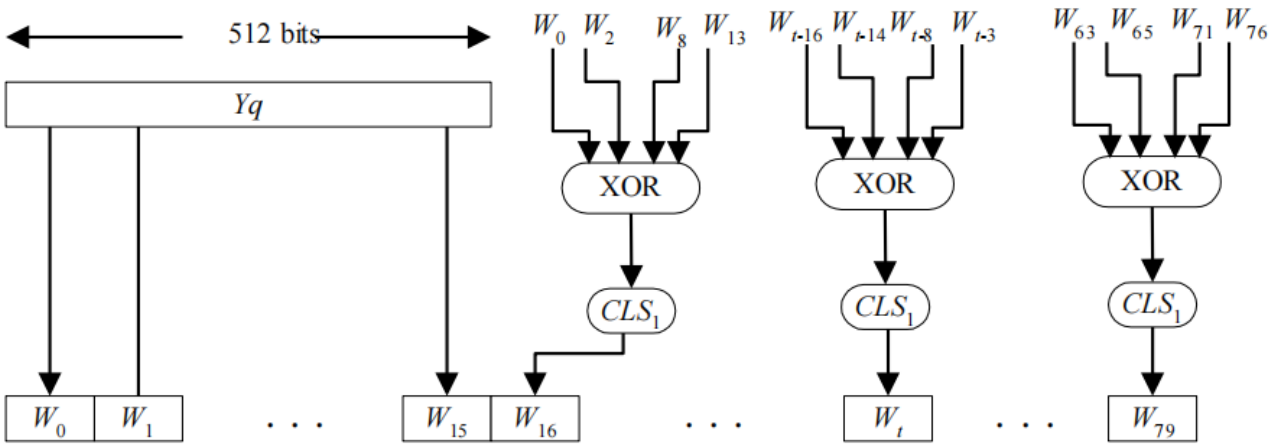


图6-11 SHA分组处理所需的80个字的产生过程

相较于MD5直接用一个消息分组的16个字作为每步迭代的输入，SHA将消息分组的16个字**扩展为80个字**以供压缩函数使用，使得寻找具有相同压缩值的不同消息更加困难。

4.3 SHA与MD5比较

抗穷搜索攻击的强度：MD5<SHA

攻击类型	MD5	SHA
寻找具有给定消息摘要的消息	2^{128}	2^{160}
找出具有相同消息摘要的两个不同消息	2^{64}	2^{80}

速度：MD5>SHA

5 HMAC

使用带密钥的哈希函数生成MAC

5.1 算法描述

分组：b比特
分组数：L
输出：n比特
密钥：K
如果密钥长度大于b，则将密钥输入哈希函数产生一个n比特的密钥
K^+是左边经填充0后的K，长度为b
$ipad$为$b/8$个00110110，$opad$为$b/8$个01011010

算法框图：

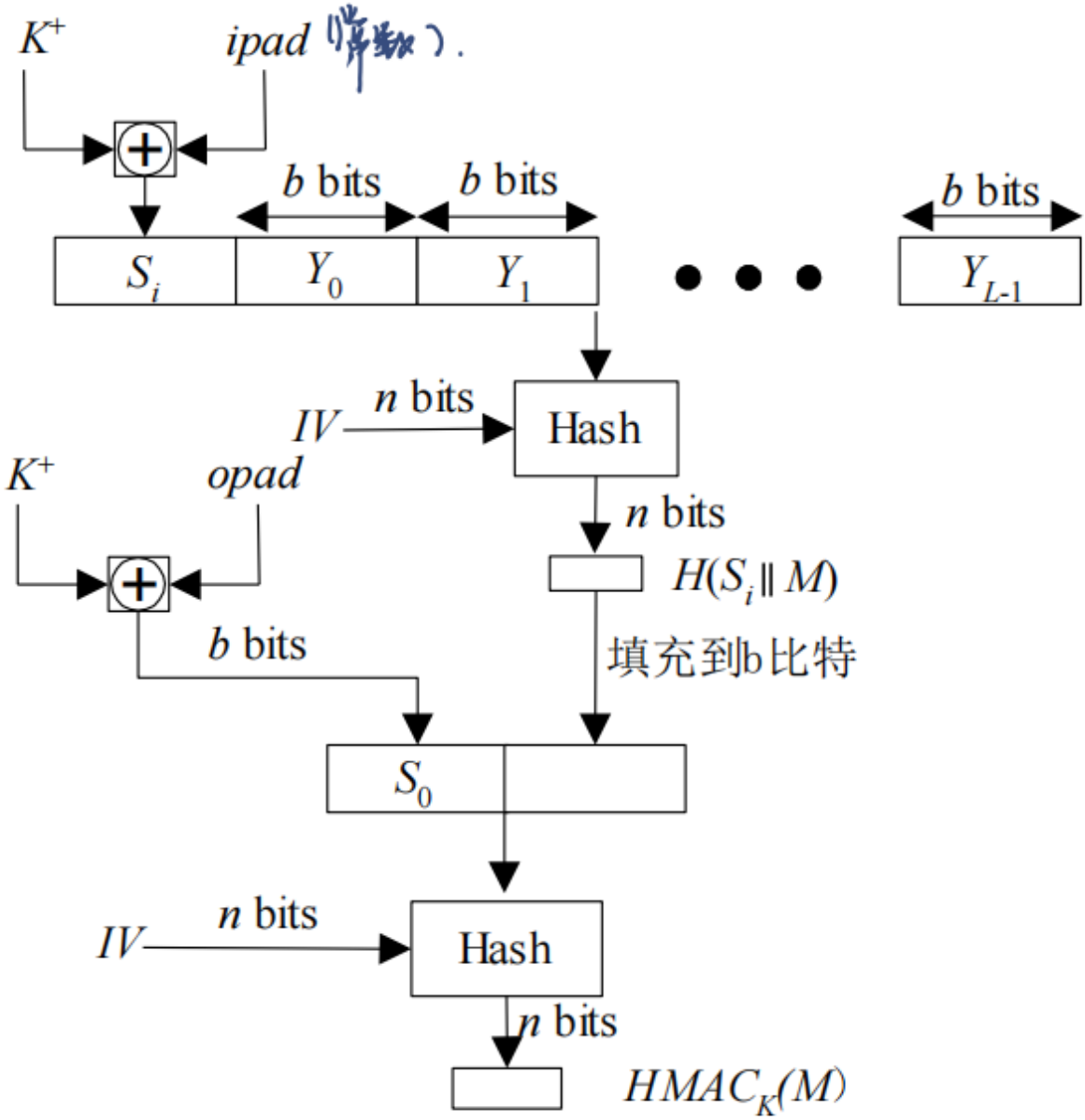


图6-12 HMAC的算法框图

表达式：$$HMAC_k=H[(K^+\oplus opad) || H[(K^+\oplus ipad)|| M]]$$
在实现HMAC时，可以预先求出以下两个量$$f(IV,(K^+\oplus ipad)\f(IV,(K^+\oplus opad)$$
$f(cv,block)$为压缩函数，输入n比特链接变量、b比特分组，输出n比特链接变量。

上述两个值仅在更换密钥时进行修改，用于作为哈希函数的初值IV。

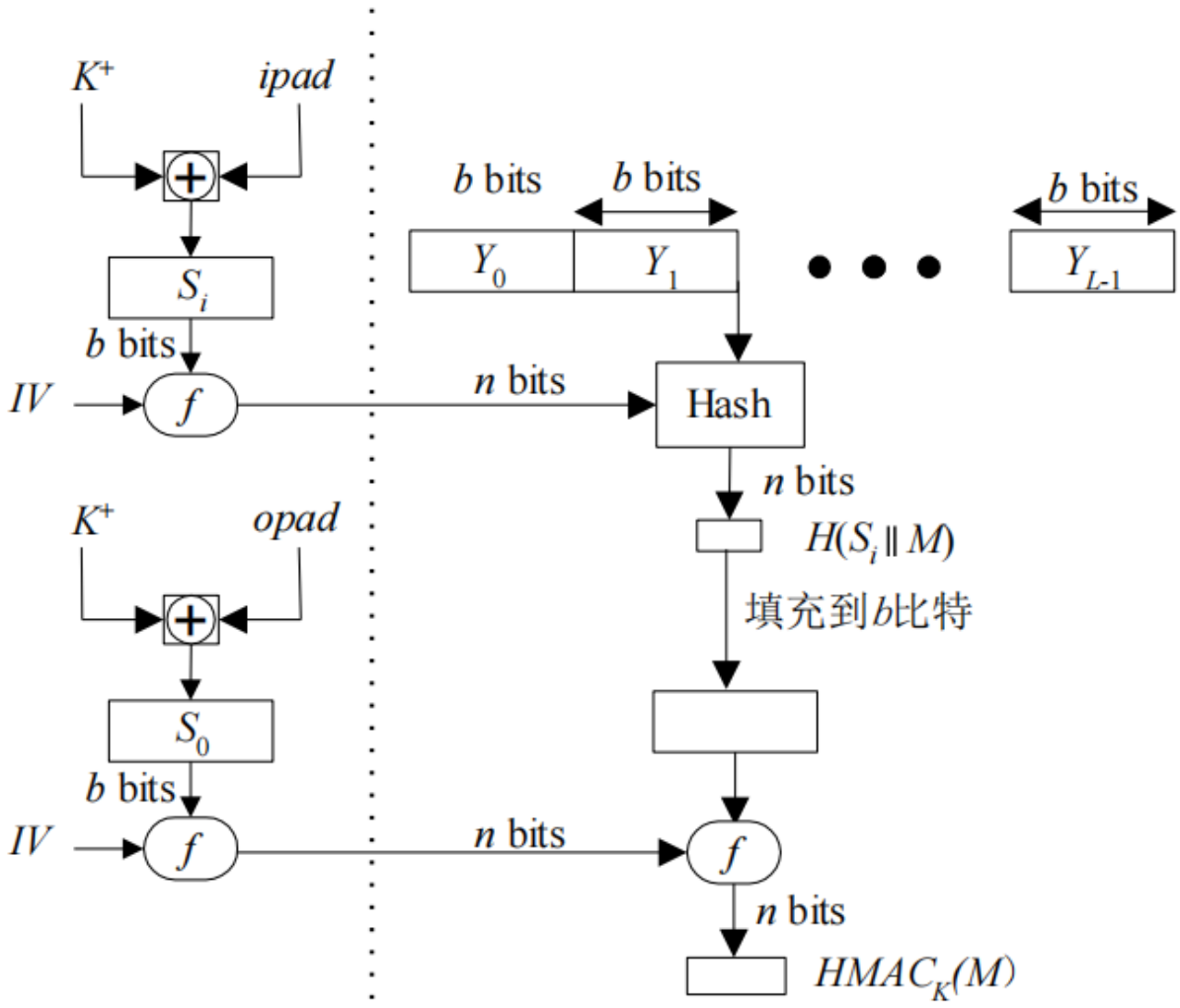


图6-13 HMAC的有效实现

5.2 HMAC的安全性

安全性取决于镶嵌的哈希函数的安全性
已证明对HMAC的攻击等价于对内嵌哈希函数的攻击：

- 攻击者能够计算压缩函数的一个输出，即使IV是随机的和秘密的
- 攻击者能够找出哈希函数的碰撞，即使IV是随机的和秘密的

第一种攻击中，n比特长的IV可视为密钥，即对密钥的穷搜索攻击，攻击的复杂度为$O(2^n)$
第二种攻击中，生日攻击，攻击复杂度为$O(2^{n/2})$，但前提是得到HMAC**在同一密钥下**产生的一系列消息。

6 SM3哈希算法

输出256比特
过程与上述两个算法并无大区别
填充消息、附加消息长度、迭代压缩、消息扩展、压缩函数

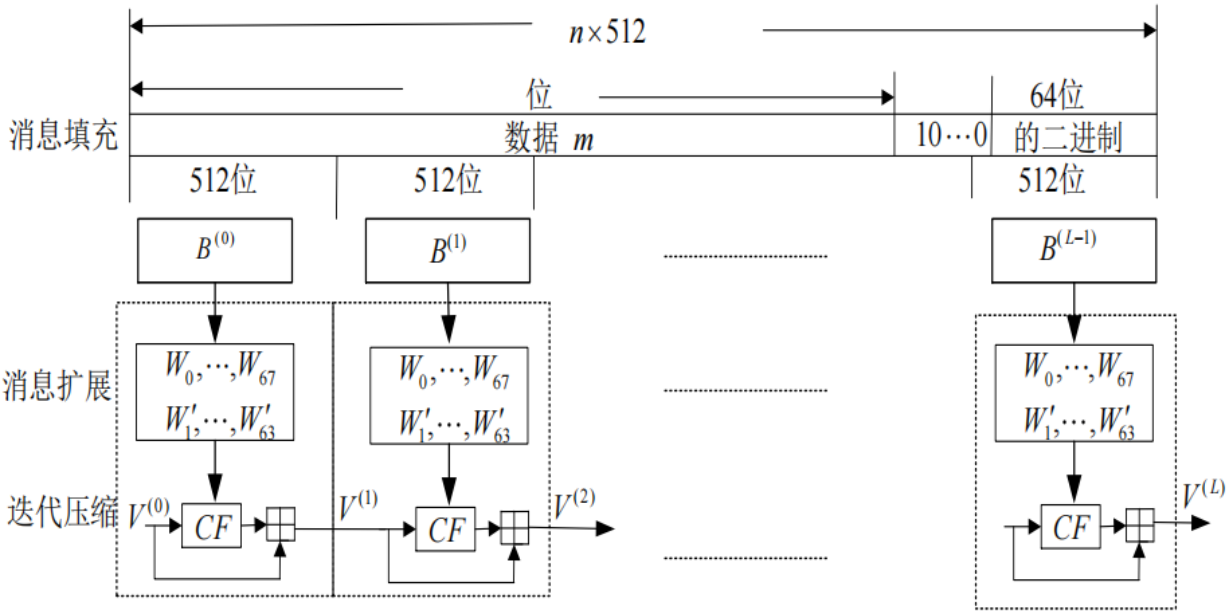


图6-14 **SM3** 产生消息哈希值的处理过程

迭代过程

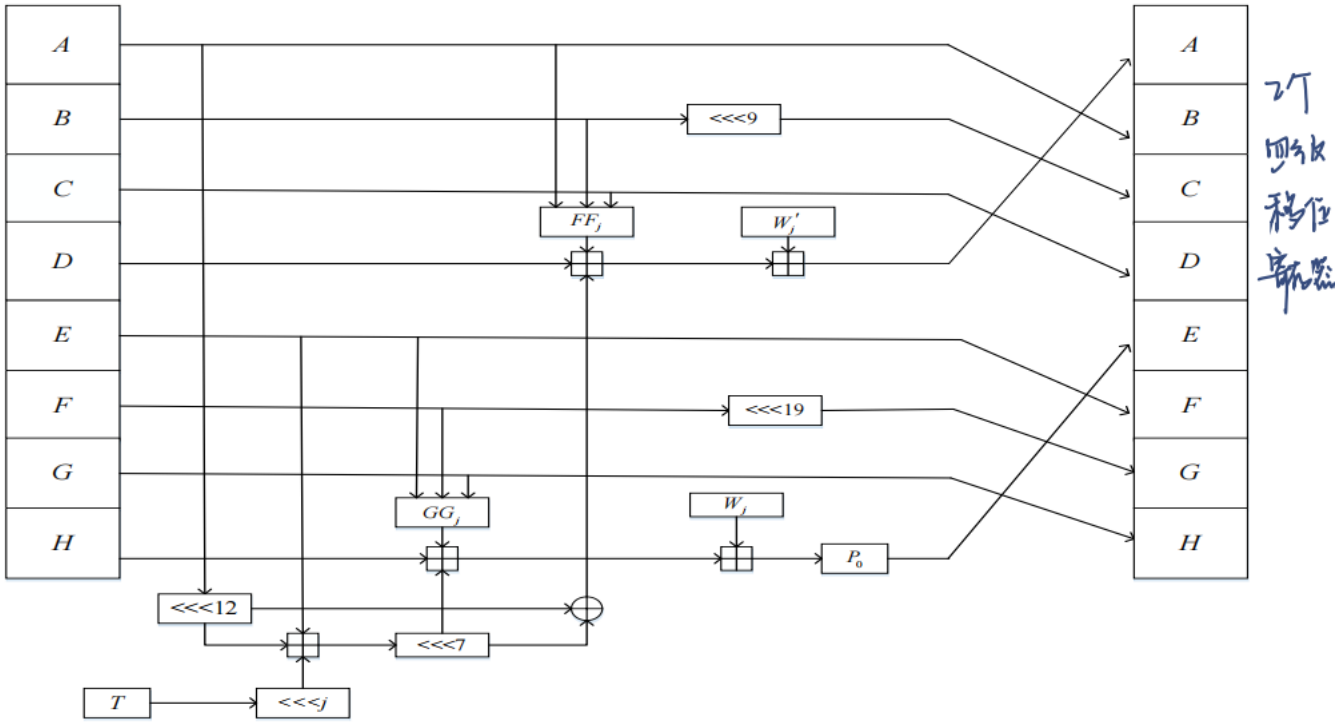


图6-13 SM3压缩函数中一步迭代示意图。

$GG_j(X,Y,Z)$非线性——混淆
置换函数$P_0(X),P_1(X)$线性——扩散