密 码 学

（第十三讲）

HASH函数

张焕国

武汉大学计算机学院

目　录

目　录

一、HASH函数的概念

1、 Hash的作用

- *Hash码也称报文摘要。*
- *它具有极强的错误检测能力。*
- *用Hash码作MAC，可用于认证。*
- *用Hash码辅助数字签名。*
- *Hash函数可用于保密。*

一、HASH函数的概念

2、Hash函数的定义

① *Hash函数将任意长的数据M变换为定长的码h，记为：h=HASH(M)或 h=H(M)。*

②实用性：*对于给定的数据M，计算h=HASH(M)是高效的。*

③安全性：

- *单向性：对给定的Hash值h，找到满足H(x)＝h的x在计算上是不可行的。*

  否则，设传送数据为C=＜M，H(M||K)＞，K是密钥。攻击者可以截获C，求出Hash函数的逆，从而得出M||S＝H^{-1}(C)，然后从M和M||K即可得出K。

一、HASH函数的概念

2、Hash函数的定义

③安全性：

- *抗弱碰撞性：对任何给定的x，找到满足y≠x且H(x)=H(y)的y在计算上是不可行的。*

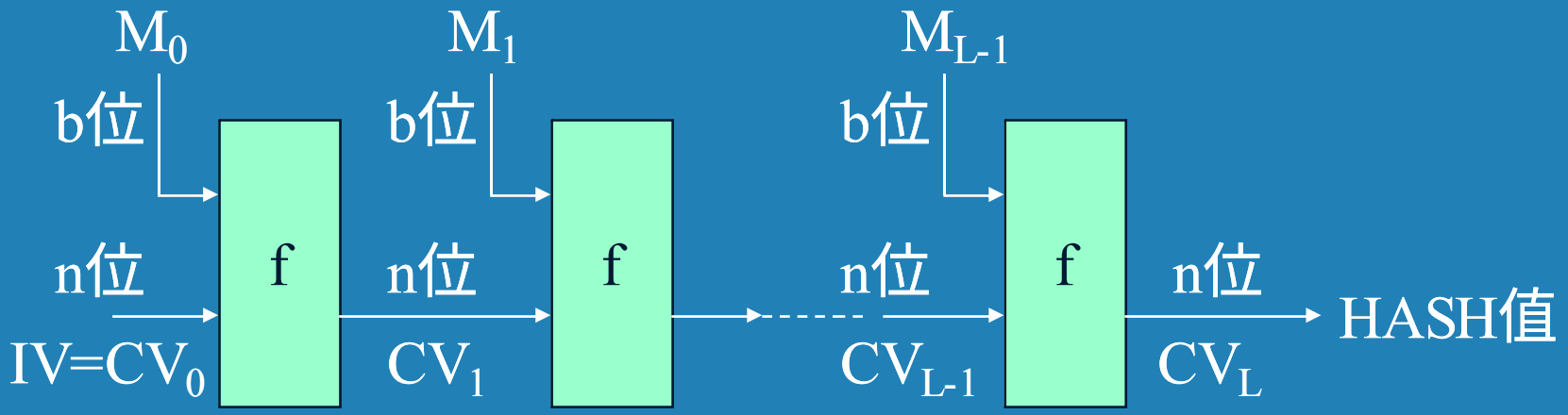
  否则，攻击者可以截获报文M及其Hash函数值H(M)，并找出另一报文M′使得H(M′)=H(M)。这样攻击者可用M′去冒充M，而收方不能发现。

- *抗强碰撞性：找到任何满足H(x)=H(y)的偶对(x,y)在计算上是不可行的。*

一、HASH函数的概念

3、安全Hash函数的一般结构

- Merkle提出了安全Hash函数主处理的一般结构
- 对数据压缩，产生Hash码。



*b*位分组，f为压缩函数，*L*轮链接迭代，*n*位输出。

一、HASH函数的概念

3、安全Hash函数的一般结构

- *分组：将输入分为L-1个大小为b位的分组。*
- *填充：若第L-1个分组不足b位，则将其填充为b位。*
- *附加：再附加上一个表示输入的总长度分组。*
- *共L个大小为b位的分组。*
- *由于输入中包含长度，所以攻击者必须找出具有相同Hash值且长度相等的两条报文，或者找出两条长度不等但加入报文长度后Hash值相同的报文，从而增加了攻击的难度。*
- 目前大多数Hash函数均采用这种结构。

二、SHA-1 HASH函数

1、 SHA系列Hash函数

- *SHA系列Hash函数是由美国标准与技术研究所(NIST)设计的。*
- *1993年公布了SHA-0(FIPS PUB 180)，后来发现它不安全。*
- *1995年又公布了SHA-1（FIPS PUB 180-1）。*
- *2002年又公布了SHA-2（FIPS PUB 180-2）。*

  SHA-2包括3个Hash函数： SHA-256， SHA-384，SHA-512
- *2005年王小云给出一种攻击SHA-1的方法，用2^{69}操作找到一个碰撞，以前认为是2^{80}。*
- *NIST打算2010年废弃SHA-1。*

二、SHA-1 HASH函数

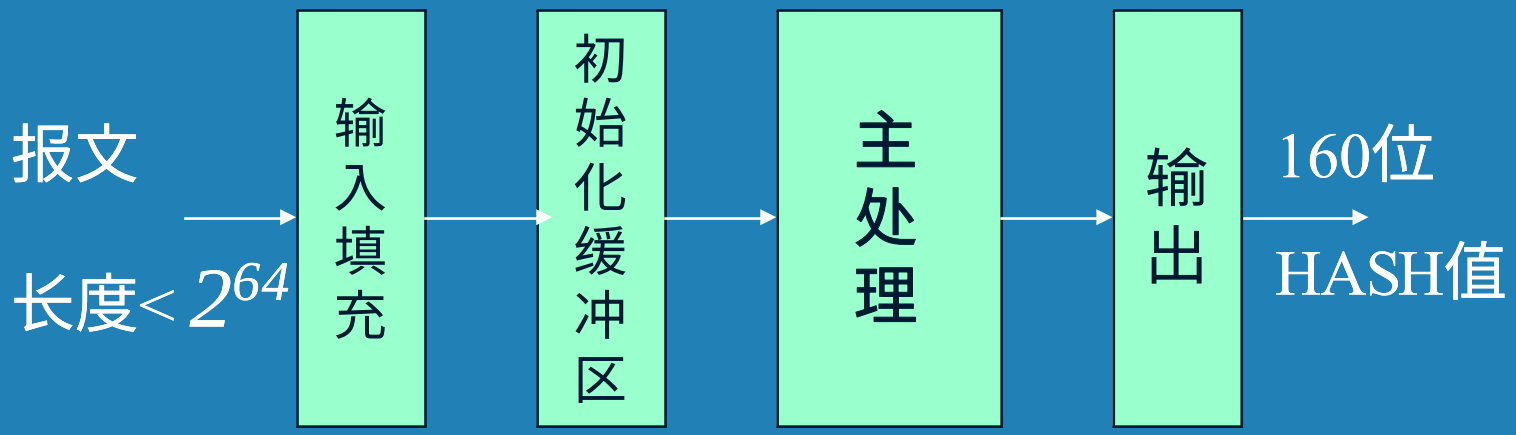
1、 SHA系列Hash函数

- *SHA-1是在MD5的基础上发展起来的。它采用Merkle提出了安全Hash结构。已被美国政府和许多国际组织采纳作为标准。*
- *SHA-1的输入为长度小于2^{64}位的报文，输出为160位的报文摘要，该算法对输入按512位进行分组，并以分组为单位进行处理。*

二、SHA-1 HASH函数

2、SHA-1的结构

- *采用Merkle提出了安全Hash结构*



二、SHA-1 HASH函数

3、运算算法

(1)输入填充

- *目的是使填充后的报文长度满足：*

$$长度 = 448 \bmod 512$$

。

①填充方法是在报文后附加一个1和若干个0。

②然后附上表示填充前报文长度的64位数据(最高有效位在前)。

- *若报文本身已经满足上述长度要求，仍然需要进行填充（例如，若报文长度为448位，则仍需要填充512位使其长度为960位），因此填充位数在1到512之间。*

二、SHA-1 HASH函数

3、运算算法

⑵初始化缓冲区

- *缓冲区由5个32位的寄存器(A，B，C，D，E)组成，用于保存160位的中间结果和最终结果。*
- *将寄存器初始化为下列32位的整数：*

A： 67452301　　B： EFCDAB89

C： 98BADCFE　　D： 10325476

E： C3D2E1F0

注意：高有效位存于低地址。

二、SHA-1 HASH函数

3、运算算法

(3)主处理

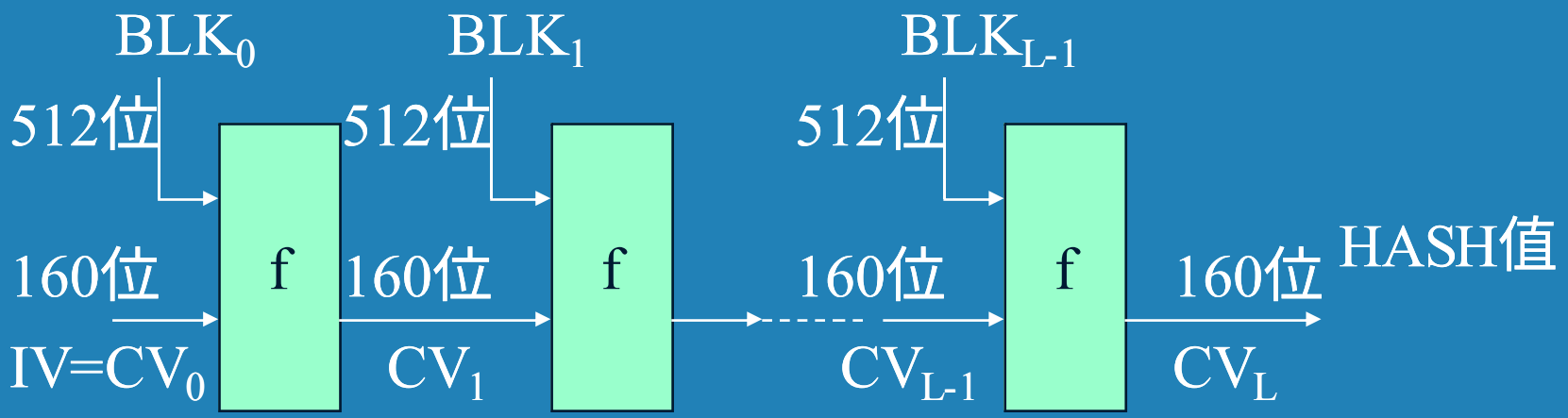
- *主处理是SHA-1 HASH函数的核心。*
- *每次处理一个512位的分组，循环次数为填充后报文的分组数L。*

二、SHA-1 HASH函数

3、运算算法

(3)主处理

- 主处理的结构：



f为SHA-1的压缩函数。

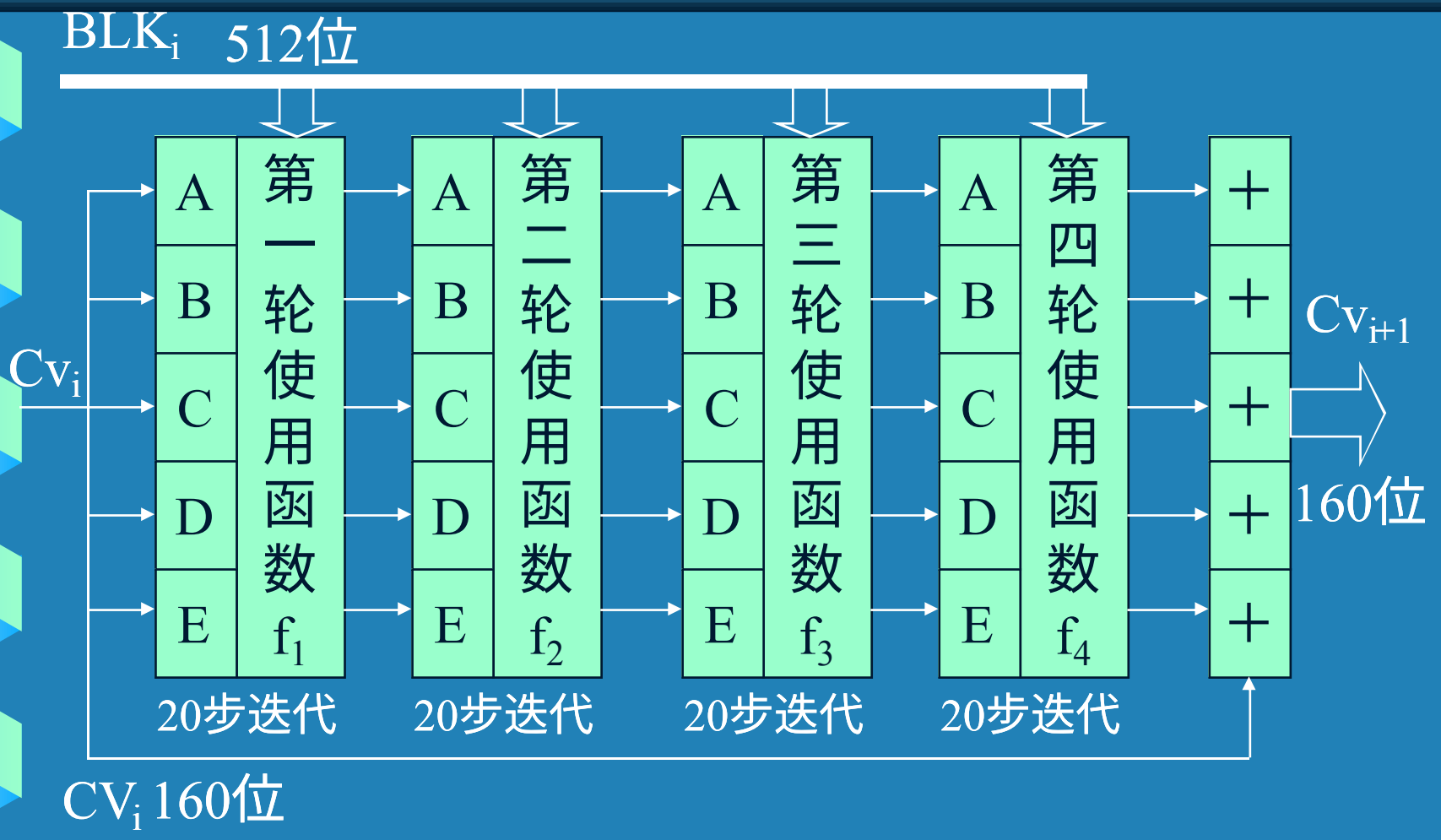
二、SHA-1 HASH函数

3、运算算法

⑶主处理

- *压缩函数是主处理的核心。*
- *它由四层运算（每层20步迭代）组成，四层的运算结构相同。*
- *每轮的输入是当前要处理的512位的分组BLK和160位缓冲区ABCDE的内容，每层都对ABCDE的内容更新，而且每轮使用的逻辑函数不同，分别为f_1, f_2, f_3和f_4。*
- *第四层的输出与第一层的输入相加得到压缩函数的输出。*

二、SHA-1 HASH函数



二、SHA-1 HASH函数

3、运算算法

⑷输出

- *所有的**L**个**512**位的分组处理完后，第**L**个分组的输出即是**160**位的报文摘要。*

二、SHA-1 HASH函数

3、运算算法

⑸归纳

- $CV_0 = IV$（ABCDE的初值）
- $CV_{i+1} = CV_i \qquad 0 \le i \le L-1$

$$\begin{cases} CV_{i+1}(0) = CV_i(0) + A_i \\ CV_{i+1}(1) = CV_i(1) + B_i \\ CV_{i+1}(2) = CV_i(2) + C_i \\ CV_{i+1}(3) = CV_i(3) + D_i \\ CV_{i+1}(4) = CV_i(4) + E_i \end{cases}$$

+ 为 模2^{32}加法

- $h = CV_L$

二、SHA-1 HASH函数

3、运算算法

(6)压缩函数

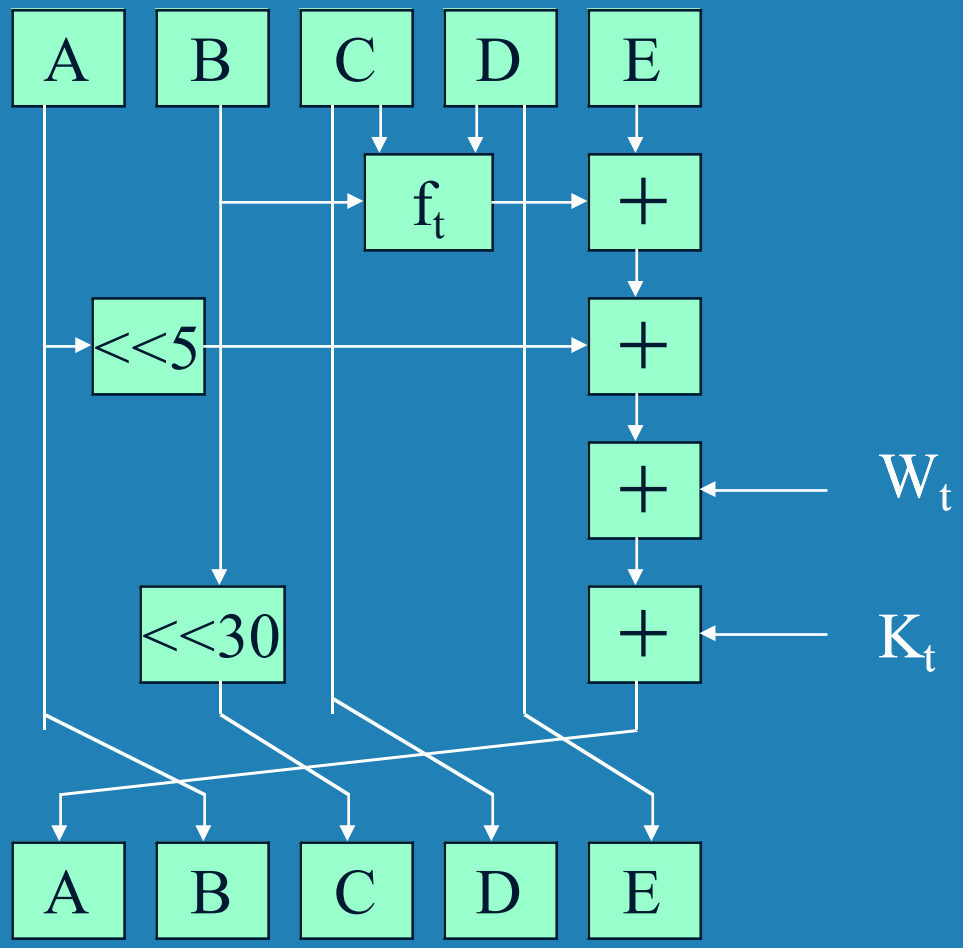
缺点：

输出B=输入A

输出D=输入C

输出E=输入D

A、C、D没有运算



二、SHA-1 HASH函数

3、运算算法

(6)压缩函数

- 每步具有下述形式：

$$A,B,C,D,E \leftarrow (E+f_t(B,C,D)+(A<<5)+W_t+K_t),A,(B<<30),C,D$$

- 每轮对A,B,C,D,E进行20次迭代，四轮共80次迭代。t 为迭代次数编号，所以 $0 \le t \le 79$ 。
- 其中，$f_t(B,C,D)$ = 第t步使用的基本逻辑函数；

  <<s = 32位的变量循环左移s位

  W_t = 从当前分组BLK导出的32位的字

  K_t = 加法常量，共使用4个不同的加法常量。

  +为 模2^{32}加法

二、SHA-1 HASH函数

3、运算算法

(6)压缩函数

- 逻辑函数f_t

  每层使用一个逻辑函数，其输入均为B,C,D(每个32位)，输出为一个32位的字。定义分别为：

  第一层$0 \le t \le 19$　　$f_1=f_t(B,C,D)=(B \wedge C) \vee (\neg B \wedge D)$

  第二层$20 \le t \le 39$　　$f_2=f_t(B,C,D)=B \oplus C \oplus D$

  第三层$40 \le t \le 59$　　$f_3=f_t(B,C,D)=(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$

  第三层$60 \le t \le 79$　　$f_4=f_t(B,C,D)=B \oplus C \oplus D$

- 缺点：f_2和f_4都是线性函数。

二、SHA-1 HASH函数

3、运算算法

(6)压缩函数

- 加法常量K_t

  每层使用一个加法常量。

- 各轮中使用的加法常量:

第一层K_t	0≤t≤19	5A827999
第二层 K_t	20≤t≤39	6ED9EBA1
第三层 K_t	40≤t≤59	8F1BBCDC
第四层 K_t	60≤t≤79	CA62C1D6

- **缺点**：*压缩字K_t的作用范围太小，只影响输出A，不影响B、C、D、E。*

二、SHA-1 HASH函数

3、运算算法

(6)压缩函数

- 压缩字W_t

  每步使用从512位的报文分组BLK导出的一个32位的字W_t。因共有80步，所以共需要80个32位字W_t（$0 \le t \le 79$）。

- 将BLK 划分为16个32位的字(M_0至M_{15})，再扩展为80个32位的字(M_0至M_{79})。

  扩展过程为：

若$0 \le t \le 15$,则$W_t = M_t$

若$16 \le t \le 79$,则$W_t = (W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3}) <<< 1$

二、SHA-1 HASH函数

- 前16步迭代中W_t的值等于报文分组的第t个字，其余64步迭代中W_t等于前面四个W_t值异或后循环左移一位的结果。
- **缺点：**
  - *压缩函数是线性函数*
  - *压缩字W_t的作用范围太小，只影响输出A，不影响B、C、D、E。*

二、SHA-1 HASH函数

注意：

- *SHA-1是美国及许多国际组织的标准。*
- *SHA-1目前还可以应用。*
- *MD-5被我国山东大学王小云教授攻击，已不安全。不能应用。*

三、SHA-2 HASH函数

1、 SHA-2的概况

- *2002年公布了**SHA-2**（FIPS PUB 180-2）。*
  SHA-2包括3个Hash函数： SHA-256， SHA-384，SHA-512
- *目的：*
  与AES配套
  增强安全性
- *与SHA-1比较：*
  结构相同
  逻辑函数相同
  摸算术相同

三、SHA-2 HASH函数

1、 SHA-2的概况

SHA参数比较

	SHA-1	SHA-256	SHA-384	SHA-512
消息摘要长度	160	256	384	512
消息长度	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$
分组长度	512	512	1024	1024
字长度	32	32	64	64
步骤数	80	64	80	80
安全性	80	128	192	256

注:1、所有的长度以比特为单位。

2、安全性是指对输出长度为n比特hash函数的生日攻击产生碰撞的工作量大约为$2^{n/2}$。

三、SHA-2 HASH函数

2、 SHA-512

⑴ SHA-512概况

- *输入长度<2^{128}*
- *数据分组长度1024位*
- *输出长度512位*

三、SHA-2 HASH函数

2、 SHA-512

⑵运算算法

① 填充

- *使填充后的长度=896 mod 1024。*
- *即使消息长度已满足上述要求，也要填充。*
- *填充由1个1和后续若干个0组成。*

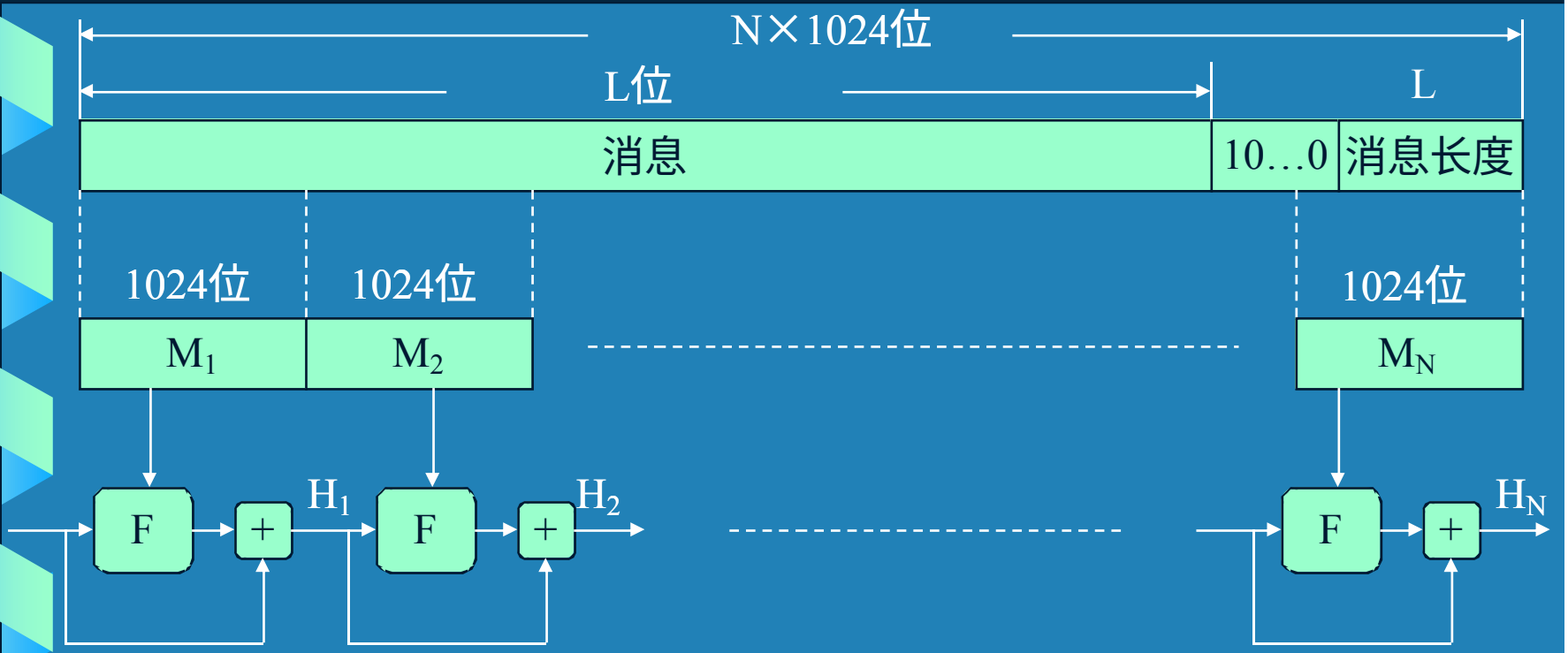
② 附加长度

- *填充后，再附加128位的块，表示原消息的长度。*

注意：*在①、②步后，数据长度为1024的N倍。*

- *将数据分成N块，每块1024位，进行迭代处理。*

三、SHA-2 HASH函数



- F块处理
- +为摸2^{64}加

SHA-2框图

三、SHA-2 HASH函数

2、 SHA-512

③初始化缓冲区

- 运算的中间结果和最终结果保存于512比特的缓冲区中，缓冲区用8个64比特的寄存器(A,B,C,D,E,F,G,H)表示，并将这些寄存器初始化为下列64比特的整数。
- A＝6A09E667F3BCC908　E=510E527FADE682D1
  B＝BB67AE8584CAA73B　F=9B05688C2B3E6C1F
  C＝3C6EF372FE94F82B　G=1F83D9ABFB41BD6B
  D＝A54FF53A5F1D36F1　H=5BE0CD19137E2179
- 获得方式：前8个素数取平方根，取小数部分的前64比特。
- 存储方式：最高有效字节存于低地址字节位置。

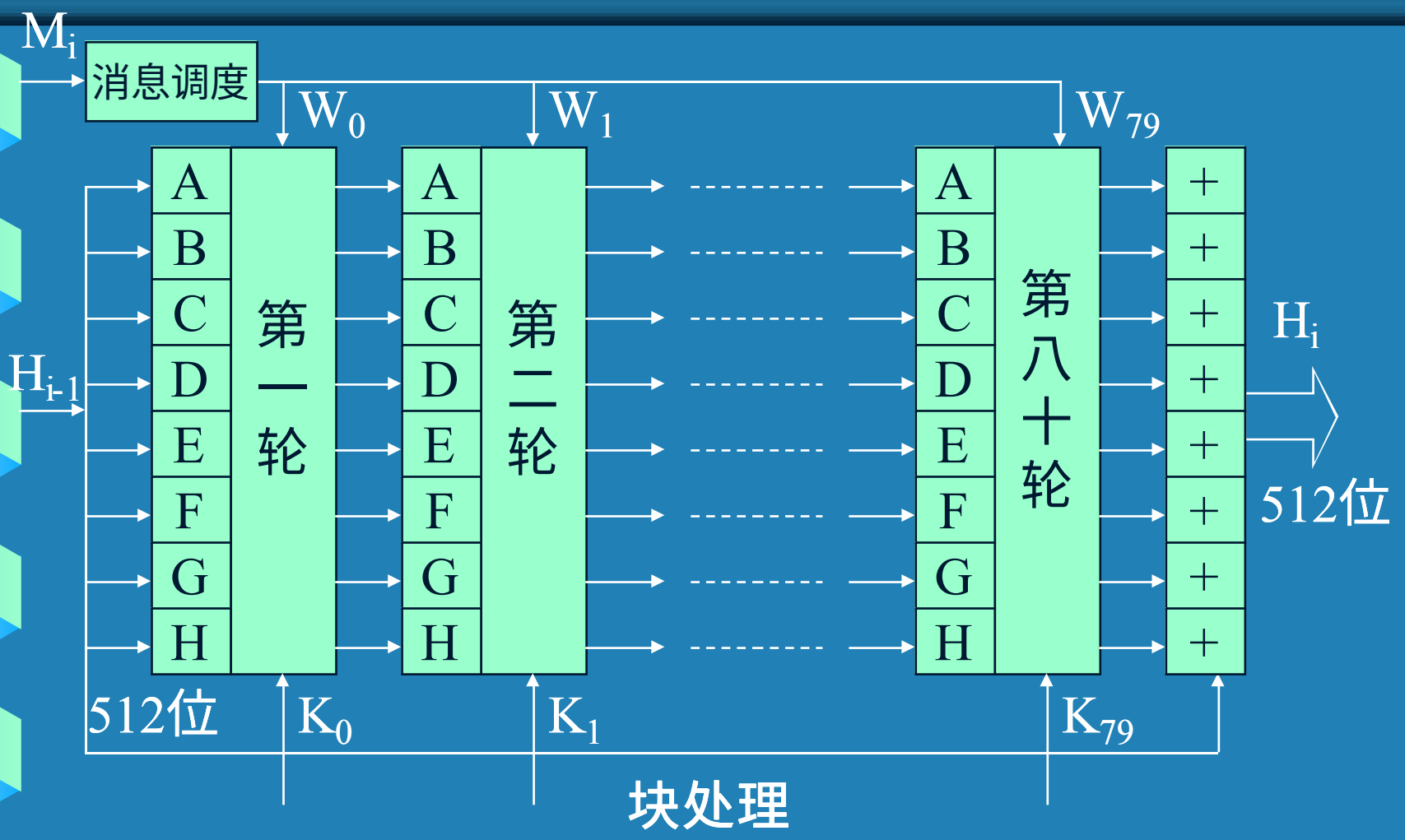
三、SHA-2 HASH函数

2、 SHA-512

④块处理

- 1024比特块的处理。
- 80轮迭代运算。
- 每一轮都把512比特缓冲区的值ABCDEFGH作为输入，并更新缓冲区的值。第一轮时，缓冲区里的值是中间的Hash值H_{i-1}。
- 每一轮，都使用一个64比特的值W_t，其中$0 \le t \le 79$。
- 每一轮还将使用附加的常数K_t，其中$0 \le t \le 79$。
- 80轮迭代后输出H_i。
- 存储方式：最高有效字节存于低地址字节位置。

三、SHA-2 HASH函数



三、SHA-2 HASH函数

2、 SHA-512

⑤轮函数

- 每一轮的处理。
- 基本逻辑函数：
  - CH（E,F,G）=（E AND F）⊕（NOT E AND G)
  - Maj（A,B,C）=(A AND B)⊕(A AND C)⊕(B AND C)
  - $\sum_0^{512}$=ROTR^{28}(A)⊕ ROTR^{34}(A)⊕ ROTR^{39}(A)
  - $\sum_1^{512}$=ROTR^{14}(E)⊕ ROTR^{18}(E)⊕ ROTR^{41}(E)

其中ROTR^i(X)表示把X循环右移i位。

注意：前2个函数与SHA-1的相同，后2个函数不同。

三、SHA-2 HASH函数

2、 SHA-512

⑤轮函数

- 附加常数K_t：
  - 80个K_t， $0 \leq t \leq 79$。
  - 最小的80个素数开立方，取根的小数部分的前64比特。
  - 这些无理数的小数部分有很好的随机性。
  - 附加K_t， 可消除数据中的规律性。

三、SHA-2 HASH函数

2、 SHA-512

⑤轮函数

- 压缩字W_t：
  - 80个W_t，$0 \le t \le 79$。
  - 从1024位的块中导出，将其分成16个64位的M_t。
  - $W_t = M_t$，$0 \le t \le 15$。
  - $W_t = \sigma_1^{512}(W_{t-2}) + W_{t-7} + \sigma_0^{512}(W_{t-15}) + W_{t-16}$。
  - 其中$\sigma_0^{512}(x) = ROTR^1(x) \oplus ROTR^8(x) \oplus SHR^7(x)$;
    $\sigma_1^{512}(x) = ROTR^{19}(x) \oplus ROTR^{61}(x) \oplus SHR^6(x)$;
    $SHR^i(x)$表示x左移i位，右边填0;

三、SHA-2 HASH函数

2、 SHA-512

⑤轮函数

$T_1=H+Ch(E,F,G)+(\Sigma_1^{512}E)+W_t+K_t$

$T_2=(\Sigma_0^{512}A)+Maj(A,B,C)$

$A=T_1+T_2$

$B=A$

$C=B$

$D=C$

$E=D+T_1$

$F=E$

$G=F$

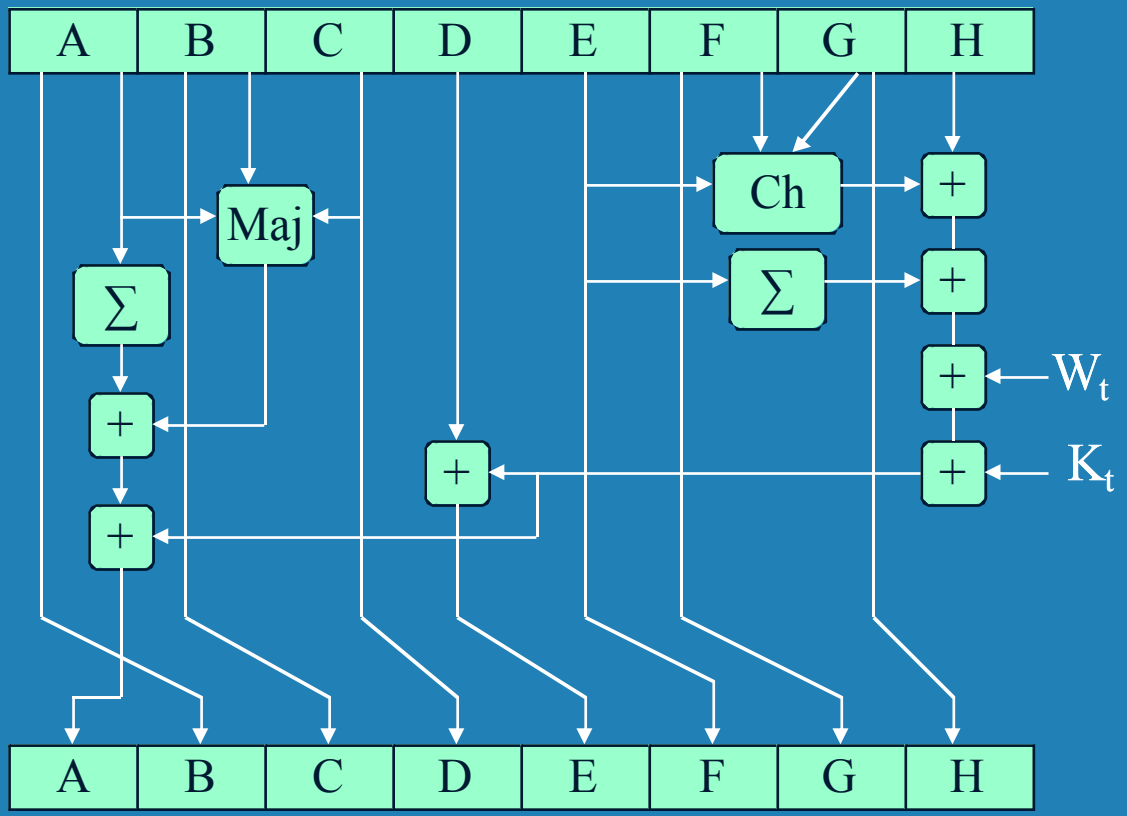
$H=G$

缺点：直接相等的太多。

三、SHA-2 HASH函数

2、 SHA-512

⑤轮函数



三、SHA-2 HASH函数

2、 SHA-512

- 评说 ：
  - ♣ SHA-512的结构与SHA-1相同；
  - ♣ SHA-512的输出长度比SHA-1长，因此抗穷举攻击能力增强；
  - ♣ SHA-512的4个逻辑函数与SHA-1变化不大；
    SHA-1每20轮使用一个逻辑函数；
    SHA-2每轮都使用四个逻辑函数；
  - ♣ 应用还未开始；
  - ♣ 安全性只有经过实践检验，才能给出正确结论。


谢 谢！