第3章 数据完整性与散列函数

南京大学计算机系黄皓教授

2011年10月19日 星期三

参考文献

1. Wenbo Mao(毛文波)，现代密码学理论与实践，电子工业出版社，2004年7月。

2. W. Stallings（杨明等译），编码密码学与网络安全-原理与实践，电子工业出版社。2001年4月。

内容

1. 数据完整性的基本概念；
2. 散列函数；
3. 密码散列报文鉴别码

1. 基本概念

数据完整性需求

- 我们关于开放通信网络的脆弱性做了一个理想的、标准的假设：

  所有的通信都经受一个称为Malice的攻击者，他可以随意地窃听、截取、重发、修改、伪造或插入消息。

  当Malice插入了修改过的或伪造的消息，他将试图欺骗目标接收者，使其相信该消息来自某个其他合法的主体。

  我们需要一种机制，使得消息的接收者可以验证该消息是否是来自所声称的消息源，且在传输的过程中是否受到未授权方式修改。数据完整性就是抗击对消息未授权修改的安全服务。

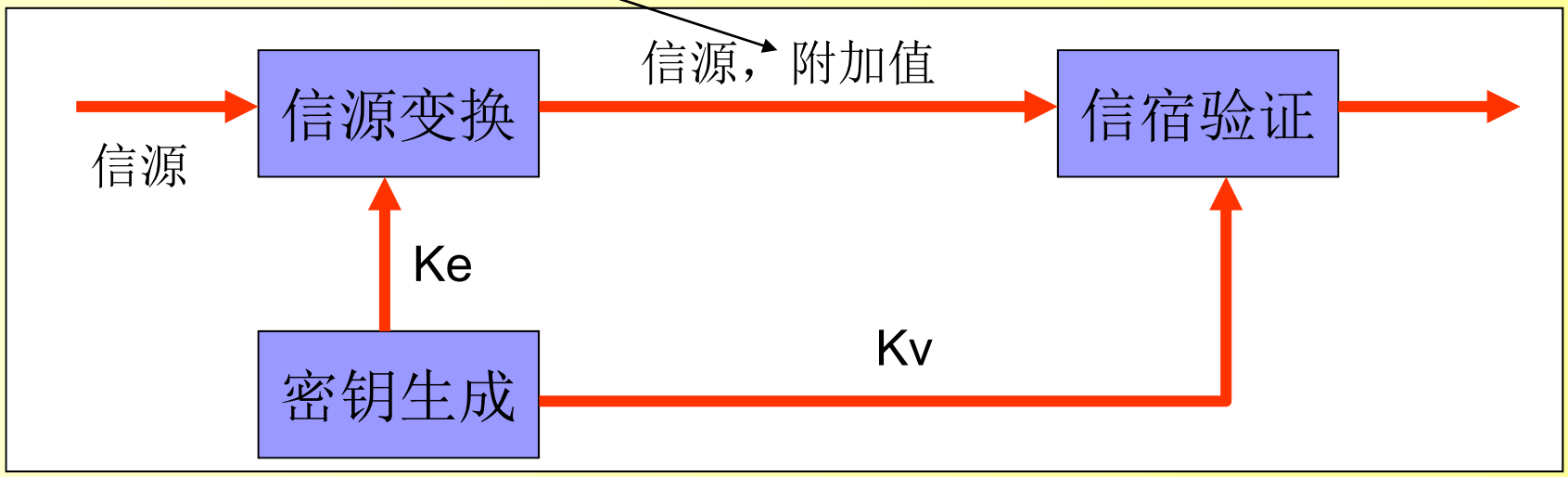
差错控制与数据完整性

- 检错码
  - 检测由于通信的缺陷而导致消息发生错误的方法。
  - 通过这种编码加入的冗余度使得消息的接收者可以采用极大似然检测器来判定接收到的码字应该译为哪条消息。
- 数据完整性
  - 消息的发送者通过编码为消息增加一些冗余来生成一个“校验值”，并将该校验值附在消息之后；
  - 消息的接收者根据与发送者协商好的一系列规则，利用附加的校验值来检验所接收到的消息的正确性。
  - 使得加入的校验值在整个校验值空间中尽可能地均匀分布，这就使得攻击者伪造一个有效校验值的概率达到最小。

篡改检测码

- 设M是任意的信息，Ke为编码密钥，Kv是验证密钥，
- 篡改检测码的生成：MDC＝f(Ke, M) (Manipulation Detection Code)
- 篡改检测码的验证：

$$G(Kv, M, MDC) = \begin{cases} \text{true}, & \text{概率为 1}, \quad \text{如果 } MDC=f(Ke, M) \\ \text{false}, & \text{压倒性概率}, \quad \text{如果 } MDC \neq f(Ke, M) \end{cases}$$



完整性的保护

- 完整性保护的应用
  - 数据通信
  - 数据存储
- 完整性保护的机制
  - 对称技术
    - 密码散列函数
    - 密码函数
  - 非对称技术：数字签名

消息摘要

存储问题：

- 假定M是要存储的消息。
- h是一个Hash函数，y=h(M)称为消息M的摘要。
- 安全地保存h，
  - 我们希望如果消息M改变成了M’，则h(M) ≠h(M’)。
  - 如果这样，我们可以用消息摘要h(M)来验证消息M是否被改变。

通信问题：

- 假定M是要发送的消息。
- 消息M的摘要y随着消息一起发送M||y，接收者验证y是否等于h(M)。
  - 问题：M||h(M) → M’ || h(M’)，h是公开的，攻击者可以完成这样的篡改，接收者无法发现消息的改变。
  - 带密钥hash：y=h(k, M)。
  - 攻击者无法完成篡改：M||h(k, M) → M’ || h(k, M’) ，因为没有密钥k。

密码散列函数

- 由对称密码技术生成的MDC常称为消息认证码(MAC，Message Authentication Code)。
- 散列函数h是一个确定的函数，它将任意长的比特串映射为定长|h|比特串的杂凑值。
- 我们希望h具有以下性质：
  - 混合变换　对于任意的输入x，输出的杂凑值h(x)应当和区间[0, $2^{|h|}$]中均匀的二进制串在计算上是不可区分的。
  - 抗碰撞攻击(强冲突)　找两个输入x和y，且$x \neq y$，使得h(x)=h(y)，在计算上应当是不可行 的。为使这个假设成立，要求h的输出空间应当足够的大。|h|最小为128，而典型的值为160。　(例如：合同欺骗)
  - 抗原像攻击 (弱冲突)　已知一个杂凑值h，找一个输入串x，使得h=h(x)，在计算上是不可行的。(例如：篡改)

RSA签名体制

- 密钥建立
  - 用户Alice的公钥为(N，e)，(N, p, q, d)为私钥。
  - 其中N=pg，p和g是两个长度差不多的大素数，e是满足gcd(e，φ(N))=1的整数。整数d，满足ed = 1 mod (φ(N))。
- 签名生成
  - 消息的m的签名：
  - $S = Sign_d(m) \leftarrow m^d \bmod (N)$
- 签名验证
  - 设Bob是验证者，他知道公钥(N，e)属于Alicec，给定一个消息-签名对(m，s)。
  - Bob的验证过程为

$$Verify_{(N, e)}(m, s) = True, \text{ 如果} m = s^e \pmod N$$

ElGamal签名体制

- 生成密钥对
  - 生成一个大的随机素数p和整数mod p的乘法群Z_p^*的生成元α;
  - 选取一个随机整数s (1≤ s ≤p-2), 计算$\beta = \alpha^s \pmod p$;
  - 公钥(p, α, β), 私钥s。
- 对信息m签名
  - 选取一个随机整数 k (1≤k≤p-2)，计算$X=\alpha^k \bmod p$
  - 从方程 $m = (s \cdot X + k \cdot Y) \bmod (p-1)$ 中求解Y;
  - 签名为(X,Y);
- 验证签名
  - 验证等式： $\beta^X \cdot X^Y \stackrel{?}{=} \alpha^m \bmod p$
  - $(\alpha^s)^X \cdot (\alpha^k)^Y = \alpha^{s \cdot X + k \cdot Y} \bmod p$

    $= \alpha^{m + t \cdot (p-1)} \bmod p$

    $= \alpha^m \cdot \alpha^{t \cdot (p-1)} \bmod p$

    $= \alpha^m \bmod p$

椭圆曲线密码体制 — 签名与验证ECSA

对信息m签名

I. 将m表示成一个二进制串
II. 计算hash值 **e= H(m);**
III. 在区间**[1，n-1]**内选取一个随机数k,
IV. 计算点 (x_1,y_1)**:=k·P** (k个P相⊕)
V. 计算$r=(x_1+e) \bmod q$;
VI. 利用私钥***d***计算
$s=(k-d \cdot r) \bmod n$
VII. 签名**(r, s)** 。

验证签名(r, s)

I. 查找公钥(E(Fq),P,n,Q),
II. 计算点 $(x_1,y_1)=sP+rQ$
III. 计算hash值 e= H(m);
IV. 计算$R=(x_1+e) \bmod q$;
V. 如果R=r，则接受签名。

2. 散列函数

散列函数的目标

- 散列函数的目的是为文件、报文或其他的分组数据产生“指纹”。要用于报文鉴别，散列函数H必须具有如下性质：
  1. H能用于任何大小的数据分组。
  2. H产生定长输出。
  3. 对任何给定的x，H(x)要相对易于计算，使得硬件和软件实现成为实际可行。
  4. 对任何给定的码H(x)，从H(x)计算x，在计算上是不可行的。这就是所谓的单向性质。
  5. 对任何给定的分组x，寻找不等于x的y，使得H(y)=H(x)在计算上是不可行的。
  6. 寻找对任何的(x,y)对使得H(x)=H(y)在计算上是不可行的。

合同欺骗的例子

- Alice准备一份合同的两个版本，一份对Bob有利，一份将使他破产；
- Alice对这两种版本的每一份都作一个细微的改变（例如，在回车之前加一个或二个空格，通过在n行中作修改，则可以得到2^n种不同的文件）。
- Alice比较这两种版本的散列值，找出相匹配的一对(N,M)， 其中对M对Bob有利,N将使他破产，并且H(M)=H(N)。
- Alice请求Bob对合同M签名：$M \,||\, E(K_{RBob}, H(M))$
- Alice 在适当的时候向法官证明Bob签署过合同：

  $N \,||\, E(K_{RBob}, H(N))$（ $= N \,||\, E(K_{RBob}, H(M))$ ）

构造意义相同的冲突报文

Dear Anthony,

This letter is		**you to**	**Mr.**	**P.**
	to introduce		**Alfred**	
I am writing		**to you**	--	**--**

---

	new	**chief**			
Barton, the			**jewellery**	**buyer for**	
	newly appointed	**senior**			

Alice 要作多少次修改？

生日攻击

- 给定**x,y, H(x)=H(y)**的概率为**1/n; n**为**H**的可能的函数值的个数；
  如果H(x)是m位的整数，则n=2^m。
- 给定**$H(x_1)$, $H(x_2)$,…, $H(x_k)$, $H(x_1)$**至少与某个**$H(x_i)$** 相同的概率为
  [1-（1-1/n) ^{k}] ≈k/n；
- 如果**H(x)**的可能的函数值个数为**2^m**，则**$H(x_1)$**至少与某个**$H(x_i)$ (i=2,..,k)** 相同的概率大于**1/2**的最小的**k**值为**$2^{m/2}$**。

- 假设一年365天， n=365
- k个人中没有人生日相同的概率。

$$p=\frac{n\cdot(n-1)\cdot\cdots\cdot(n-k+1)}{n^k}$$

$$\approx 1-e^{-k(k-1)/2n}$$

- p>1/2 的最小k是多少？

$$k=(2(ln2)n)^{1/2}$$

$$\approx 1.18\, n^{1/2}$$

≈23 （如果n=365)

散列函数的强行攻击

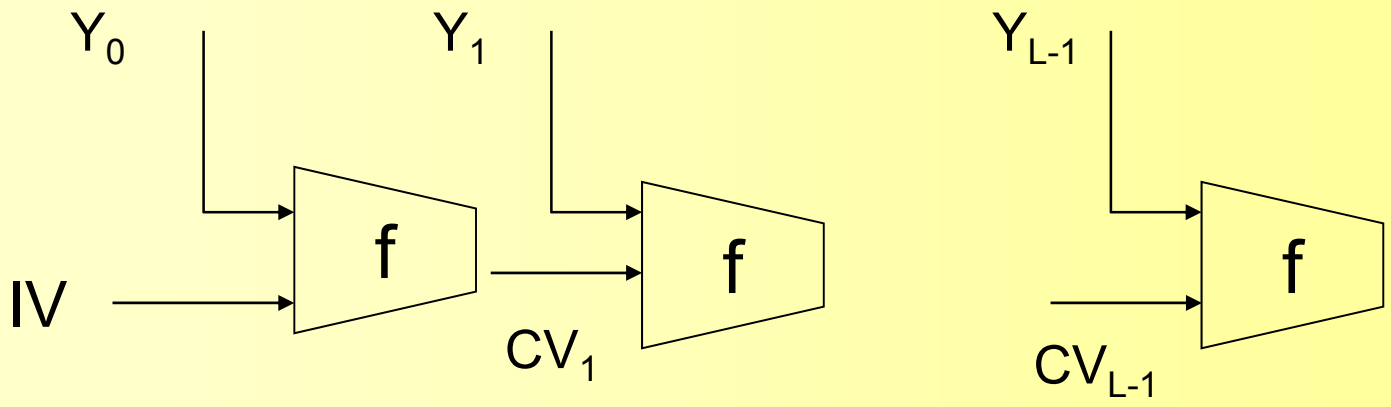
- 单向：　　给定h，　　求x使得H(x)=h;　　2^n
- 弱抗冲突：给定H(x)，求y使得H(y)=H(x);　　2^n
- 强抗冲突：求x, y，　　使得H(y) = H(x);　　2^{n-1}

散列函数的密码分析

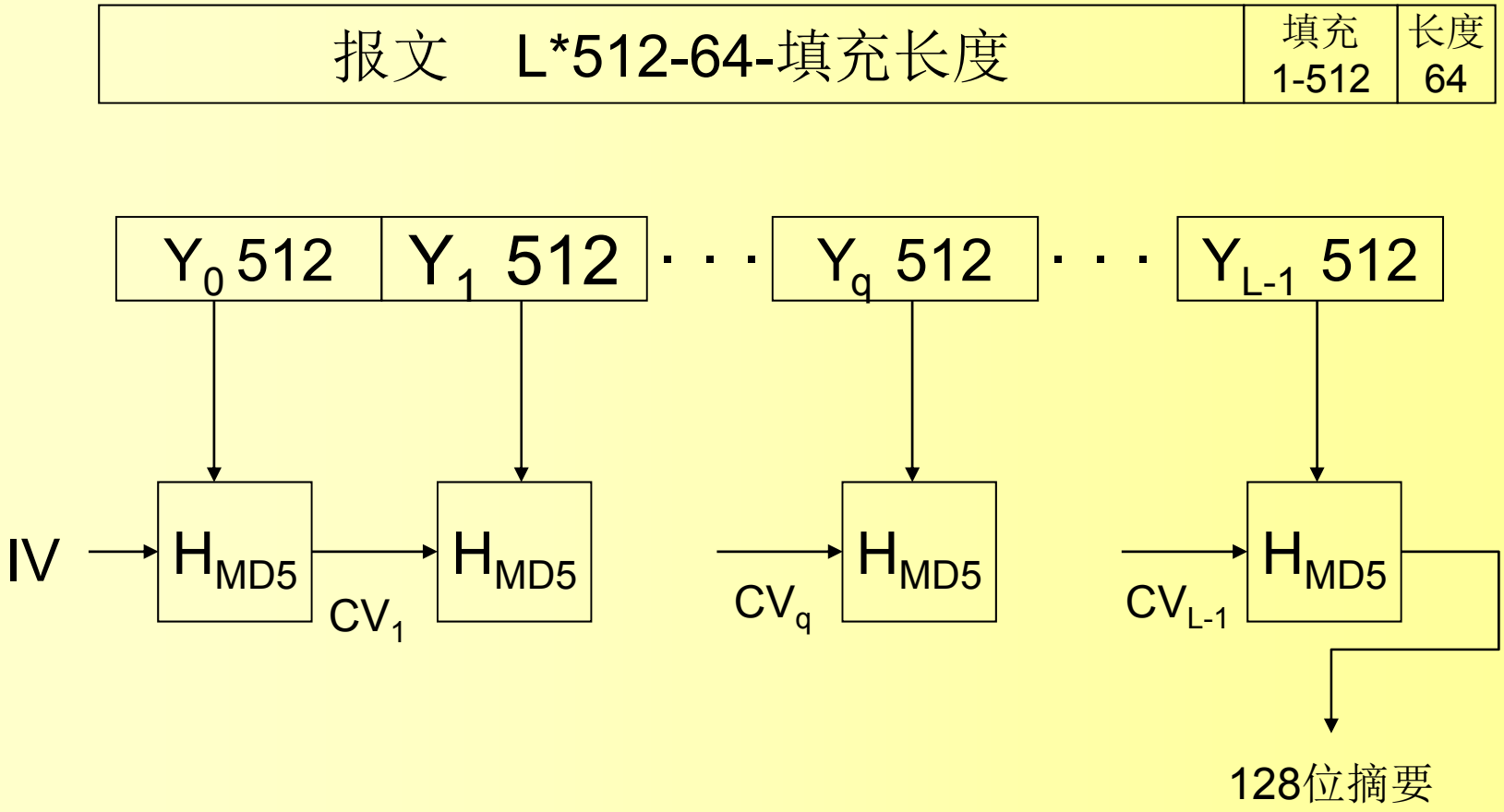
- 着重分析压缩函数f的内部结构，寻找对单次运行后就能产生冲突的高效技术；

散列函数的一般结构

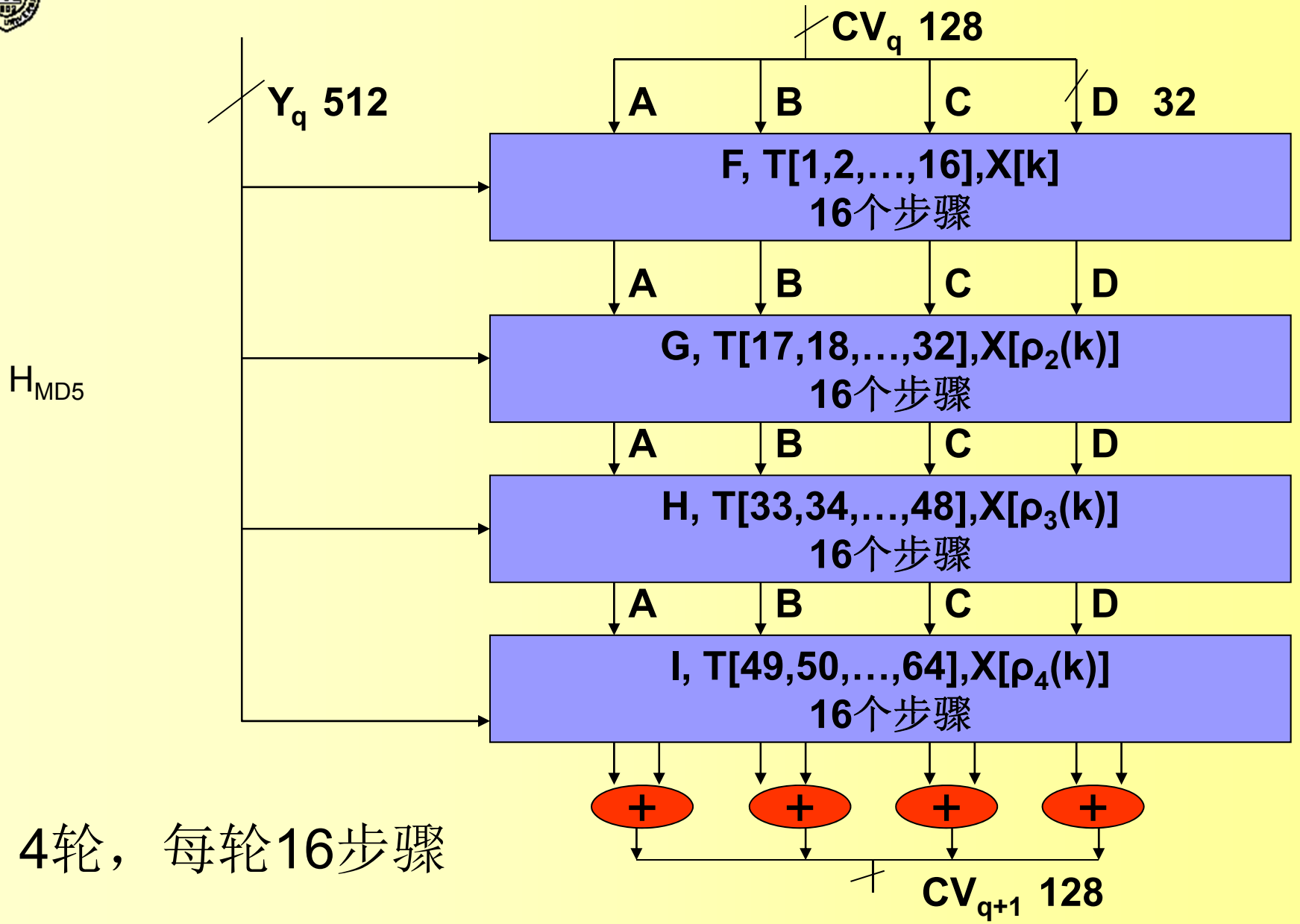
- CV_0=IV;
- CV_i=f(CV_{i-1}, Y_{i-1}); 1≤i≤L
- H(M) = CV_L



2.2 MD5报文摘要算法 (Rivest, RFC1321)



- MD5: RFC1321
- MD4: RFC1320
- MD2: RFC1319



H_{MD5}

4轮，每轮16步骤

H_{MD5}

1. **X[k] 是分组的第k个32比特,**

   k=1,2,…, 16

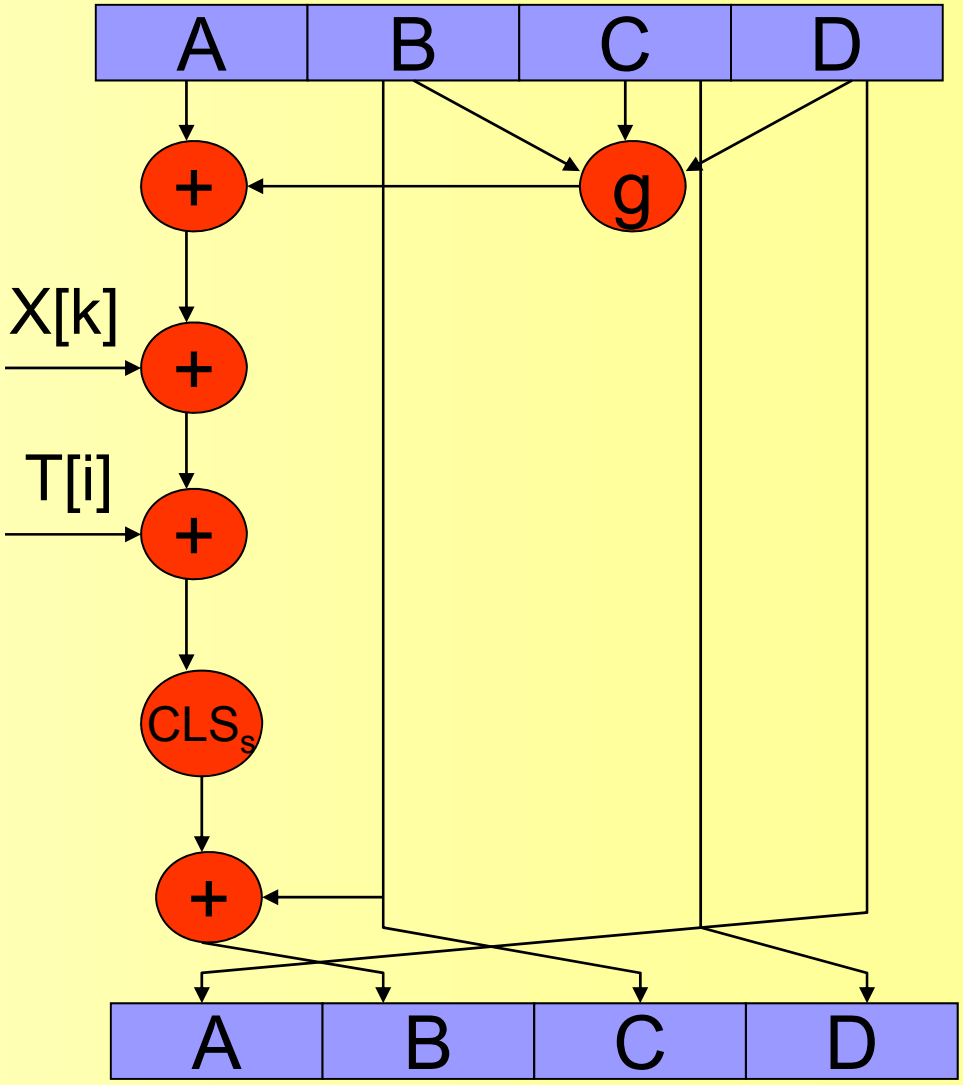
2. **T[i]= (2^{32} abs(sin(i))) 的整数部分**

3. **b=b+((a+g(b,c,d)+X[i]+T[i])<<<s)**

   循环左移s各比特。

4. **函数g**
   第1轮 F(b,c,d)= (b∧ c) ∨ (¬ b∧ d)
   第2轮 G(b,c,d)= (b∧ d) ∨ (c∧ ¬ d)
   第3轮 H(b,c,d)= b ⊕ c ⊕ d
   第4轮 I(b,c,d)= b ⊕ (b∧¬ d)

5. **$\rho_2(k)$= (1+5k) mod 16**
   $\rho_3(k)$= (5+3k) mod 16
   $\rho_4(k)$= 7k mod 16



MD5的强度

- Rivest猜想MD5
  - 给定H(x)，求y使得H(y)=H(x)需要2^{128}数量级的操作
  - 求x,y，使得H(y)=H(x)的操作，需要2^{64}数量级的操作
- 1996年Dobbertin 提出了针对MD5单轮压缩函数的攻击。
- 2004年8月17日在美国加州圣巴巴拉国际密码学会议(Crypto'2004)上，山东大学王小云教授等报告了MD5的破解方法。

2.3 SHA-1

- 安全散列算法(SHA)由美国国家标准和技术协会(NIST)提出，作为联邦信息处理标准(FIPS PUB 180) 在1993年公布。
- 1995年发布了一个修订版 FIPS PUB 180-1通常称为SHA-1
- SHA也是基于MD4的。
- 最大报文长度2^{64-1}，散列码长度160bit。
- 结构与MD5类似，抗攻击能力比MD5强；

2.4 RIPEMD-160

- 欧共体的RIPE项目研制的；
- MD4的变种，为抵抗已知的关于MD4、MD5的攻击而设计的；
- 摘要长度为160，报文长度不受限制;

3. 密码散列报文鉴别码HMAC

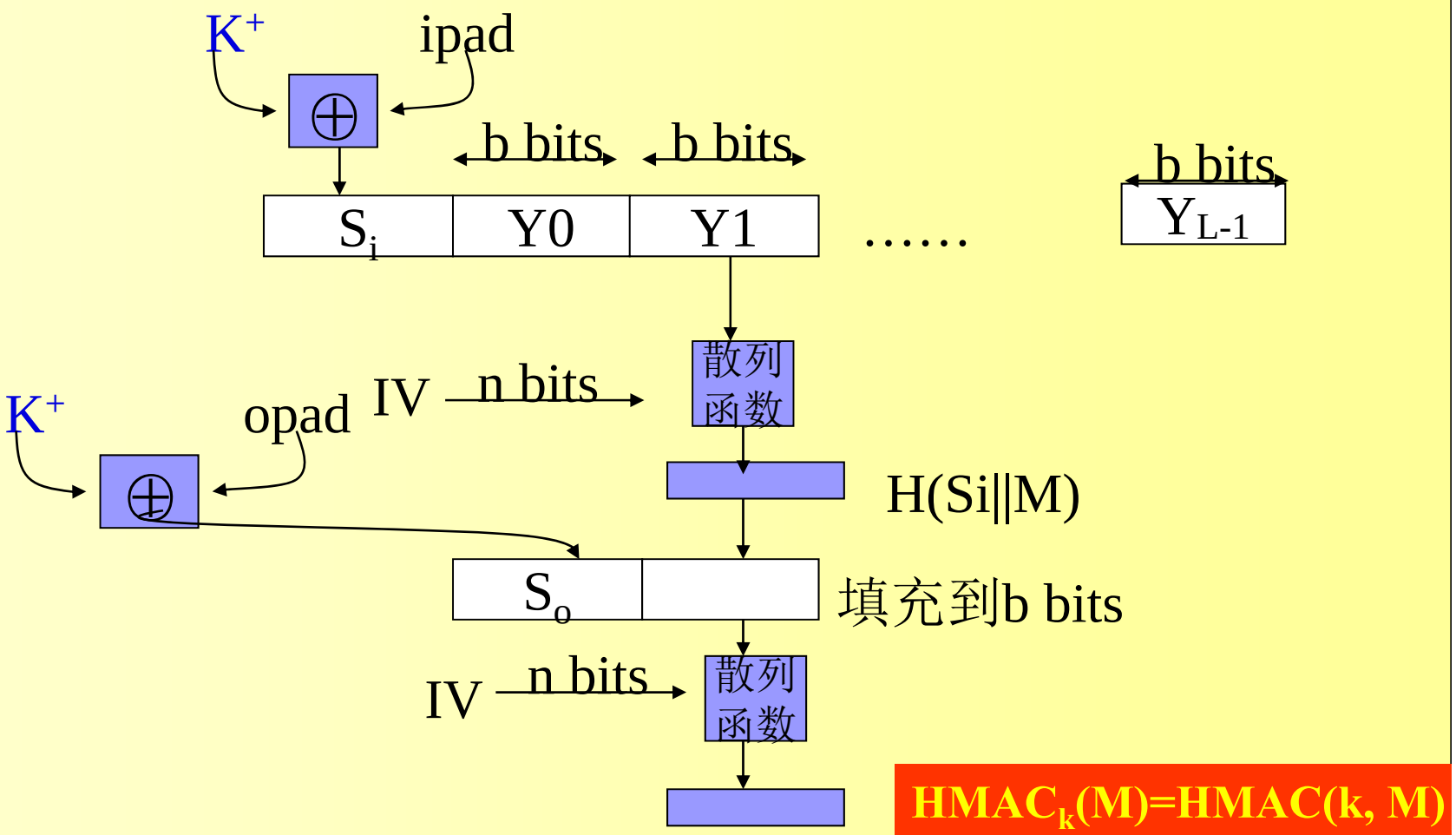
HMAC RFC2104

- **keyed-Hashing for Message Authentication Code**
- 在最近几年，研究的热点转向由密码散列码导出MAC。这样的目的在于：
  - 密码散列函数如MD5和SHA-1的软件执行速度比对称分组密码如DES的快。
  - 很容易获得密码散列函数的库代码。
  - 美国或其他国家对密码散列函数没有出口限制，而对称分组密码，即便用作MAC也是限制的。

HMAC设计目标

- 无需修改地使用现有的散列函数。特别是，散列函数的软件实现执行很快，且程序代码码是公开的和容易获得的。
- 当出现或获得更快的或更安全的散列函数时，对算法中嵌入的散列函数要能轻易地进行替换。
- 保持散列函数的原有性能不会导致算法性能的降低。
- 使用和处理密钥的方式很简单。
- 基于对嵌入散列函数合理的假设，对鉴别机制的强度有一个易懂的密码编码分析。

HMAC 算法　RFC2104



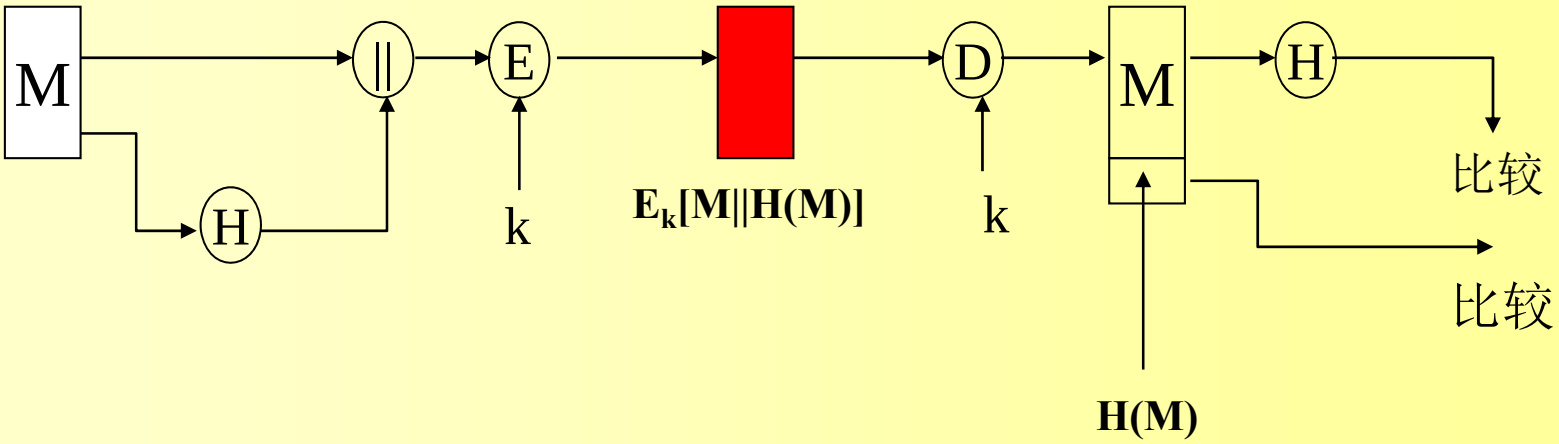
HMAC算法

1. 对密钥K的左端填充一些0生成一个b bits的串K^+。
2. ipad: b/8个00110110，opad: 01011010。
3. 将K^+ 与中ipad按比特异或(XOR)产生一个b bits 分组S_i。
4. 将报文M附加到S_i后。
5. 使用H计算第3步产生流的散列值。
6. 将K^+ 与opad按比特异或产生一个b bits 的分组S_0.
7. 将第4步产生的散列值附加到S_0后。
8. 使用H计算第6步产生流的散列值，并输出这个结果。

4. 散列函数与数字签名的应用

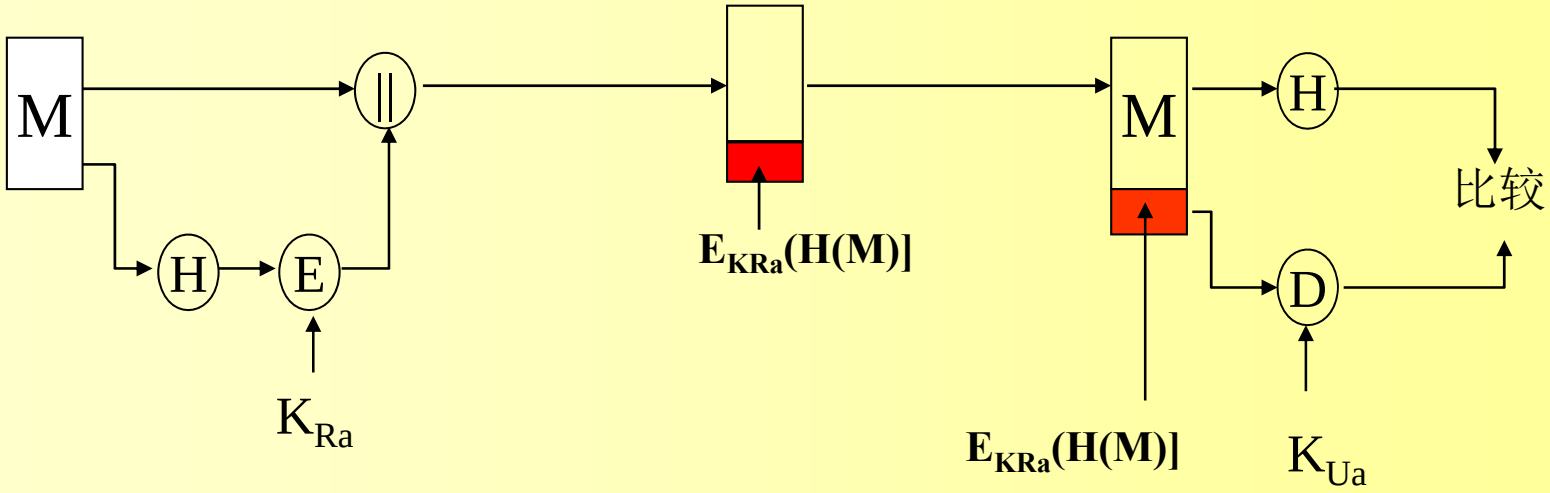
散列函数的使用 (a)

a) 提供了保密和鉴别。



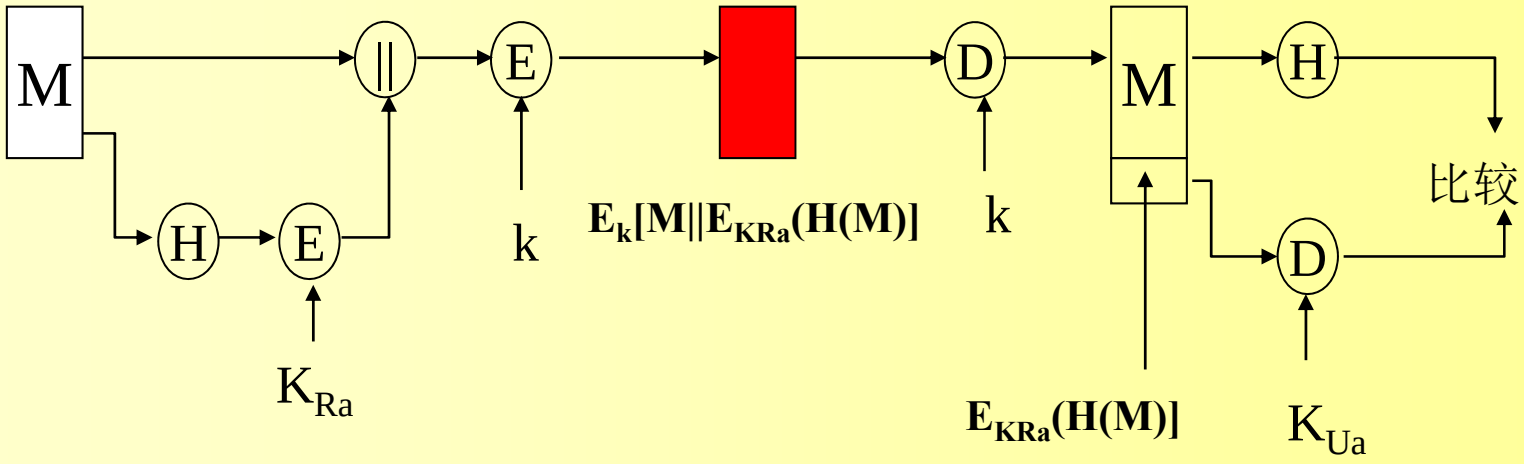
散列函数的使用 (c)

- 提供鉴别、防抵赖。

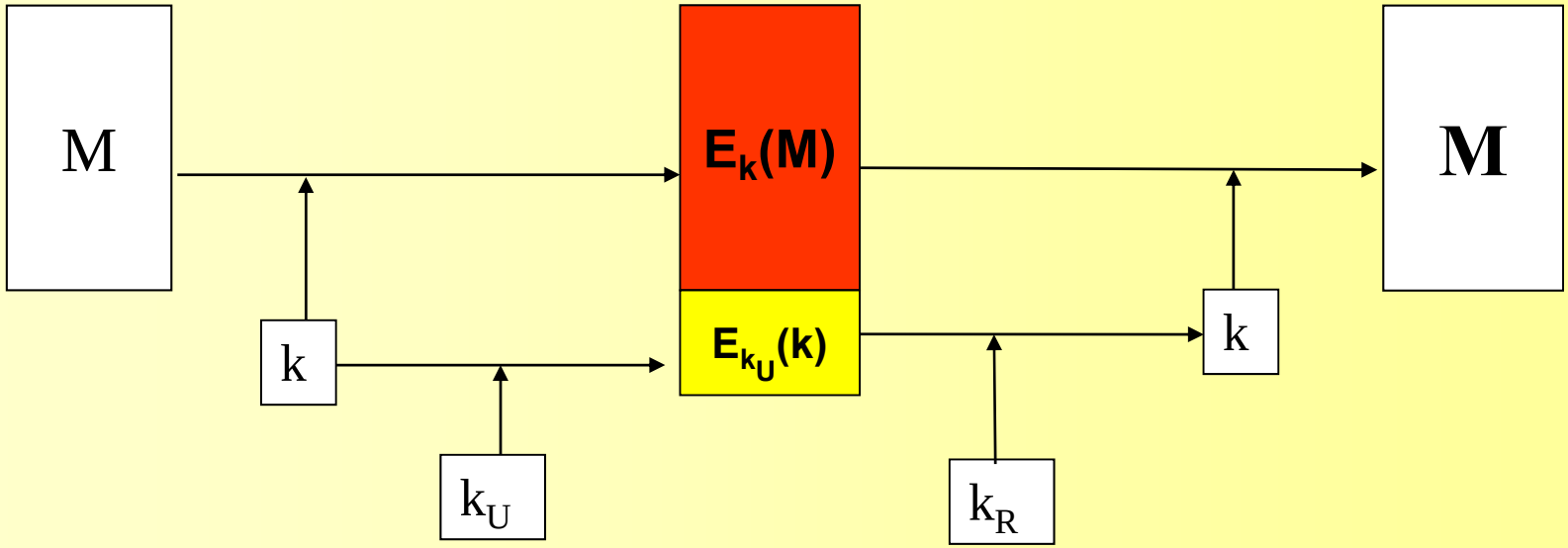


散列函数的使用 (d)

- 同时提供保密性、鉴别和抵赖。

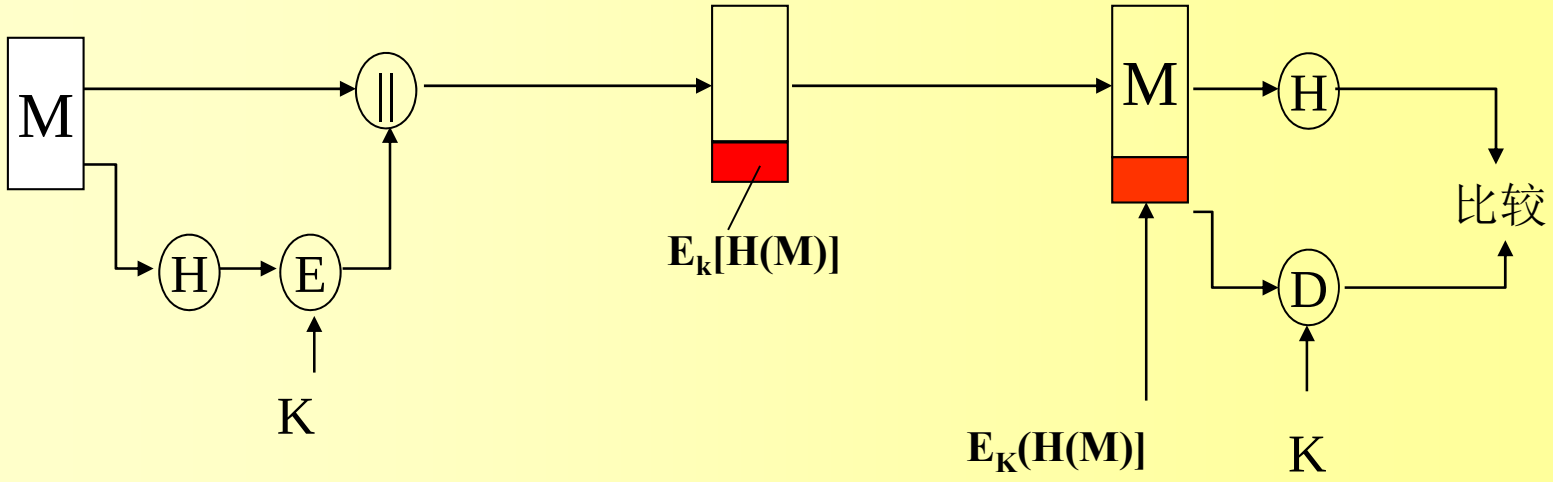


数字信封



散列函数的使用 (b)

- 散列与加密结果合并为一个整体函数实际上就是一个MAC。E[H(M)]是变长报文M和密钥K的函数值，且它生成一个定长的输出，对不知道该密钥的对手来说是安全的。
- 提供鉴别



实践题目建议

- 实现SHA-1，详细阐述其实现步骤。
- 分析Windows系统关键数据文件，利用散列函数实现一个系统完整性报告的软件。