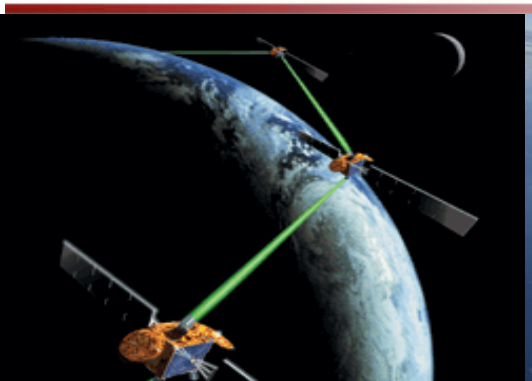
普通高等教育“十一五”国家级规划教材
教育部2011年精品教材

网络安全—技术与实践（第2版）

刘建伟 王育民 编著

清华大学出版社




课件制作人声明

- 本课件总共有17个文件，版权属于刘建伟所有，仅供选用此教材的教师和学生参考。
- 本课件严禁其他人员自行出版销售，或未经作者允许用作其他社会上的培训课程。
- 对于课件中出现的缺点和错误，欢迎读者提出宝贵意见，以便及时修订。

课件制作人：刘建伟

2016年10月25日

第7章 数字签名

一 数字签名的基本概念

二 RSA签名体制

三 Rabin签名体制

四 ElGamal签名体制

五 Schnorr签名体制

六 DSS签名体制

七 ESIGN签名体制

第7章 数字签名

八 Okamoto签名体制

九 OSS签名体制

十 离散对数签名体制

十一 其它签名体制简介

十二 消息认证码的基本用途

十三 杂凑算法/加密/签名结合应用方案

一、数字签名的基本概念

类似于手书签名，数字签名也应满足以下要求：

1. 收方能够确认或证实发方的签名，但不能伪造，简记为R1-条件。
2. 发方发出签名的消息给收方后，就不能再否认他所签发的消息，简记为S-条件。
3. 收方对已收到的签名消息不能否认，即有收报认证，简记为R2-条件。
4. 第三者可以确认收发双方之间的消息传送，但不能伪造这一过程，简记为T-条件。

1. 数字签名与消息认证的区别

消息认证

- 当收发者之间没有利害冲突时，这对于防止第三者的破坏已经足够了。
  - 收方能够验证消息发送者身份是否被篡改；
  - 收方能够验证所发消息内容是否被篡改。

数字签名

- 当收发双方存在利害冲突时，单纯用消息认证技术就无法解决他们之间的纠纷。必须采用数字签名技术。
  - 数字签名能确定消息来源的真实性
  - 数字签名能保证实体身份的真实性
  - 数字签名是不可否认的。

2. 数字签名与公钥加密的区别

公钥加密

- A采用B的公开密钥对信息加密，A将密文发给B；
- B用自己的私钥对收到的密文解密，恢复出明文。

数字签名

- A采用自己的私钥对消息m签名，A将m和签名发给B；
- B收到A的签名后，采用A的公钥来验证签名的有效性。
  - 一个签名的消息很可能在多年之后才验证其真实性
  - 数字签名可能需要多次验证
  - 对数字签名的安全性和防伪造要求很高
  - 要求签名速度比验证速度更快

3. 数字签名的分类

按照消息是否被压缩分类

- 对整体消息进行签名；
- 对压缩的消息进行签名。

按照消息/签名的对应关系划分

- 确定性（deterministic）数字签名：消息与签名一一对应，对同一消息的签名永不变化，如RSA和Rabin算法；
- 随机化（randomized）或概率式数字签名：对同一消息的签名是变化的。因此，此类签名取决于算法中的随机参数的取值，如ElGamal算法。

4. 签名体制的构成

两个组成部分

- 签名算法（signature algorithm）
- 验证算法（verification algorithm）

安全性约定

- 签名密钥是秘密的，只有签名人掌握此密钥；
- 验证算法应当公开，以便于他人进行验证。

5. 签名体制的数学表示

签名体制的数学表示

- 一个签名体制可由量(M, S, K, V)来表示
  - M是明文空间
  - S是签名的集合
  - K是密钥空间
  - V是验证函数的值域，由真、伪组成。
- 对于每一个$k \in K$，$m \in M$
  - 签名算法：$s = \mathrm{Sig}_{k1}(m) \in S$（$k1$为签名者私钥）
  - 验证算法：$\mathrm{Ver}_{k2}(s, m) \in$ {真，伪}（$k2$为签名者公钥）

签名体制的安全性

- 从m和s难于推出签名者的私钥k;
- 很难伪造另外一个消息m’，使$\mathrm{Ver}_k(s, m') \in$ {真}。

二、RSA数字签名体制

体制参数

- 令$n = p_1 \times p_2$，p_1和p_2是大素数；
- 令$m, s \in Z_n$（整数域）
- 选e，并计算出d，使 $ed \equiv 1 \bmod \varphi(n)$
- 将n, e公开（公钥），将p_1、p_2和d保密（私钥）。

签名过程

- 对$m \in Z_n$，签名：$s = Sig_k(m) = m^d \bmod n$

验证过程

- 给定m, s，验证：$m \equiv s^e \bmod n$?

RSA签名体制的安全性

讨　论

- 显然，由于只有签名者知道私钥d，根据RSA体制知，其他人不可能伪造签名；
- 易于证实$\{m, s\}$是否是合法的$\{m, s\}$对，

  只要计算 $m \equiv s^e \bmod n$ 即可。

安全性

- RSA体制的安全性依赖于$n=p_1 \times p_2$分解的困难性。

三、Rabin签名体制

体制参数

- 令$N=p\times q$，p和q是大素数
- 令$m, s \in QR_p \cap QR_q$，QR为二次剩余集
- 选p, q为秘密钥
- N为公钥

签名过程

- 明文消息m，$0<m<N$，设$m \in QR_p \cap QR_q$
- 求m的平方根，$s = Sig_k(m) = m^{1/2} \bmod p\times q$

验证过程

- 给定m, s，可验证 $m \equiv s^2 \bmod N$?

什么是二次剩余集？

- 设p是正整数，若一个整数a满足：
  - $\gcd(a, p)=1$且$x^2=a \bmod p$有解
  - 则称a是mod p的二次剩余，记为$a \in QR_p$。否则，为非二次剩余。
- 例如：取$p=7$，则：
  - $x^2=1 \bmod 7$有解：$x=1$，$x=6$
  - $x^2=2 \bmod 7$有解：$x=3$，$x=4$
  - $x^2=3 \bmod 7$无解
  - $x^2=4 \bmod 7$有解：$x=2$，$x=5$
  - $x^2=5 \bmod 7$无解
  - $x^2=6 \bmod 7$无解

故{1,2,4}是mod 7的二次剩余集合，且每个方程都有2个平方根。

记为：$QR_7=\{1,2,4\}$

而{3,5,6}是mod 7的非二次剩余集合。

四、ElGamal签名体制

体制参数

- p：一个大素数，可使Z_p中求解离散对数为困难问题；
- g：是群Z_P^*的一个生成元或本原元素；
- M：消息空间，为Z_P^*；
- S：签名空间，为Z_{P-1}；
- x：用户秘密钥，$x \in Z_P^*$
- $y \equiv g^x \bmod p$
- p, g, y为公钥，x为秘密钥。

签名过程

- 选择秘密随机数$k \in Z_P^*$，$m \in M$
- 计算：$H(m)$
- 计算：$r = g^k \bmod p$
- 计算：$s = [H(m) - xr]k^{-1} \bmod (p-1)$
- 签名为$Sig_k(m) = (r, s)$，将m和(r, s)送给对方。

ElGamal签名体制

验证过程

- 收信人收到m和(r, s)；
- 计算：$H(m)$;
- 验证：$Ver_k(H(m), (r, s))$=真$\Leftrightarrow y^r r^s \equiv g^{H(m)} \bmod p$;

  左边：$y^r r^s \equiv g^{xr} g^{sk} \bmod p \equiv g^{(rx+sk)} \bmod p$

  $(rx+sk) \equiv H(m) \bmod p-1$

  $y^r r^s \equiv g^{H(m)} \bmod p$

例子

- 选择p=467, g=2, x=127, 则有$y \equiv g^x \equiv 2^{127} \equiv 132 \bmod 467$
- 若待送消息m的杂凑值$H(m)$=100，选随机数k=213

  注意：(213, 466)=1, 且$213^{-1} \bmod 466 = 431$
- 则：$r=2^{213}=29 \bmod 467$, s=(100-127*29)431=51 mod 466。
- 验证：(1)收信人计算$H(m)$=100,

  (2)验证：$132^{29} 29^{51} = 189 \bmod 467$

  $2^{100} = 189 \bmod 467$

ElGamal签名体制的安全性

讨论

- 在不知{消息，签名}数对时，伪造签名相当于求离散对数；
- 如果攻击者掌握了同一随机数k下的两个消息m_1，m_2的合法签名（r_1, s_1）（r_2, s_2），就会构造如下的方程：

$m_1=r_1x+s_1k$

$m_2=r_2x+s_2k$

可见：攻击者解此方程可以求出x和k。

安全性

- 要确保此签名体制的安全性，就必须保证每次签名时，选择不同的随机数k。

五、Schnorr签名体制

体制参数

- p, q：大素数, $q|p-1$, 确保Z_p中求解离散对数为困难问题；
- g：是Z_p中乘群Z_P^*的一个元素，且$g^q \equiv 1 \bmod p$；
- M：消息空间，为Z_P^*；
- S：签名空间，为$Z_P^* \times Z_{P-1}$；
- x：用户秘密钥，$1<x<q$
- y：用户公钥，$y \equiv g^x \bmod p$
- p, q, g, y为公钥，x为秘密钥。

签名过程

- 用户选择秘密随机数$k \in Z_q$， $m \in M$
- 计算：$w=g^k \bmod p$
- 计算：$r=H(w||m)$
- 计算：$s=k+xr \bmod p$
- 签名：$Sig_k(m)=(r, s)$作为签名，将m和(r, s)送给对方。

五、Schnorr签名体制（续）

签名验证过程

- 收信人收到消息m和签名(r, s)
- 计算：$w'=g^s y^{-r} \bmod p$
- 计算：$H(w'\|m)$
- 验证$H(w'\|m)=r$？即 $Ver_y\{(r, s), m\}$=真

Schnorr签名体制的安全性

Schnorr与ElGamal的区别

- 在ElGamal体制中，g为Z_p的本原元素；在Schnorr体制中，g为Z_p^*中的子集Z_q^*的本原元，它不是Z_p^*的本原元。
- Schnorr的签名长度要比ElGamal短，由$|q|$及$|H(m)|$决定。
- $w=g^k \bmod p$可以预先计算，签名只需1次乘法和1次加法，所以签名速度非常快，适用于智能卡应用。

安全性

- 由于Schnorr签名较短，其安全性要比ElGamal签名差 。

六、DSS签名体制

DSS概况

- 1991年8月由NIST公布
- 1994年5月19日由NIST正式公布
- 1994年12月1日正式成为美国联邦信息处理标准
- 它是基于ElGamal和Schnorr签名体制设计的
- 该签名体制有较好的兼容性和适用性，已经成为网络安全体系的基本构件之一。

什么是DSA

- DSA是DSS签名标准中所采用的数字签名算法；
- 此算法由D. W. Kravitz设计。

DSS签名算法——DSA

体制参数

- p：大素数，$2^L-1<p<2^L$，512≦L≦1024；
- q：(p-1)的素因子，且$2^{159}<q<2^{160}$，即字长160b
- g：$g\equiv h^{(p-1)/q} \bmod p$，且$1<h<(p-1)$，$h^{(p-1)/q} \bmod p>1$
- x：选择用户私钥，$1<x<q$
- y：计算用户公钥，$y\equiv g^x \bmod p$
- p, q, g, y为公钥，x为私钥。

签名过程

- 用户选择秘密随机数k，$0<k<q$
- 计算：$H(m)$
- 计算：$r=(g^k \bmod p) \bmod q$
- 计算：$s=[k^{-1}(H(m)+xr)] \bmod q$
- 签名：$Sig_k(m)=(r, s)$，将m和(r, s)送给对方。

DSS签名算法——DSA

验证过程

- 收信人收到m和(r, s)；
- 计算：$H(m)$;
- 计算：$w=s^{-1} \bmod q$
- 计算：$u1=[H(m)w] \bmod q$
- 计算：$u2=rw \bmod q$
- 计算：$v=[(g^{u1}y^{u2}) \bmod p] \bmod q$
- 验证：$v \equiv r$?

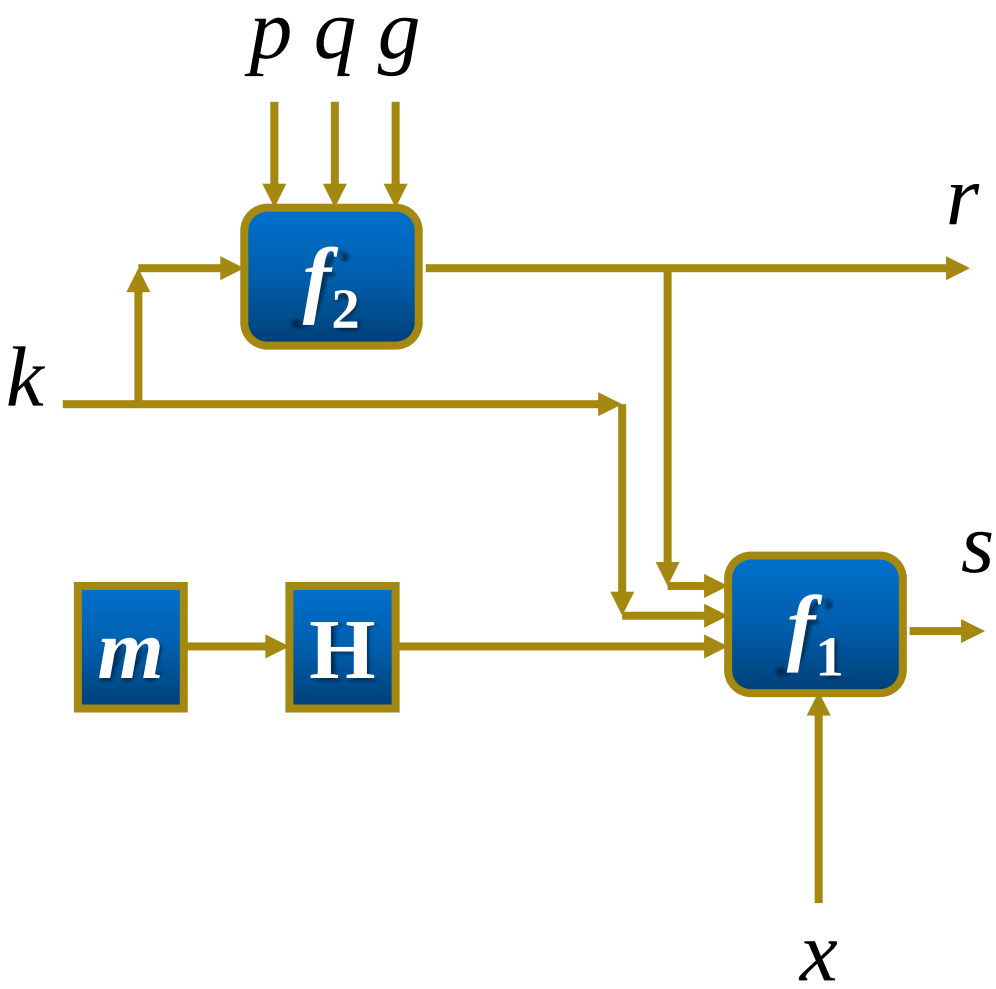
证明：

$$v=[(g^{u1}y^{u2}) \bmod p] \bmod q$$
$$=[g^{H(m)w}y^{rw} \bmod p] \bmod q$$
$$=[g^{H(m)w}g^{xrw} \bmod p] \bmod q$$
$$=[g^{[H(m)+xr]w} \bmod p] \bmod q$$

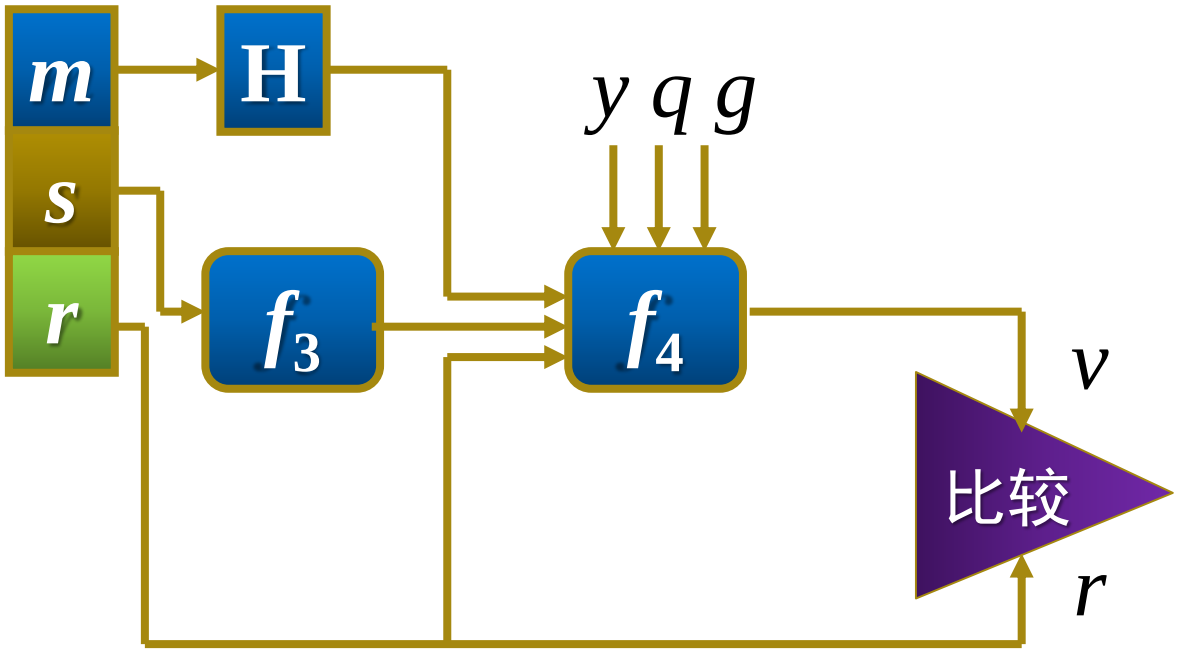
而：$[H(m)+xr]w = [H(m)+xr]s^{-1}=k \bmod q$

所以：$v=g^k \bmod q= r$

DSS签名框图



DSS签名验证框图



DSS签名算法的公众反应

RSA公司反应强烈

- RSA公司想以RSA算法为标准
- RSA指出：RSA可以加密，而DSA不能用于加密；
- DSA是由NSA开发的，可能设有陷门；
- DSA的签名验证速度比RSA慢，不适合联机在线验证；
- RSA是一个事实上的标准，而DSS与现行标准不相容；
- DSA未经公开选择，还没有足够的时间进行分析证明；
- DSA可能侵犯了其他专利，如Schnorr签名算法，Diffie-Hellman的密钥分配算法；
- DSS中模数为512b所限定的密钥量太小，现已经改为凡是512-1024b中可被64除尽的数，均可供使用。

DSS签名算法的实现速度

	DSA	RSA
密钥生成	14s	Off card
预计算	14s	N/A
签名	0.035s	15s
证实	16s	1.5s

注：PC机80386/33M，模皆为512b

十、离散对数签名体制

体制参数

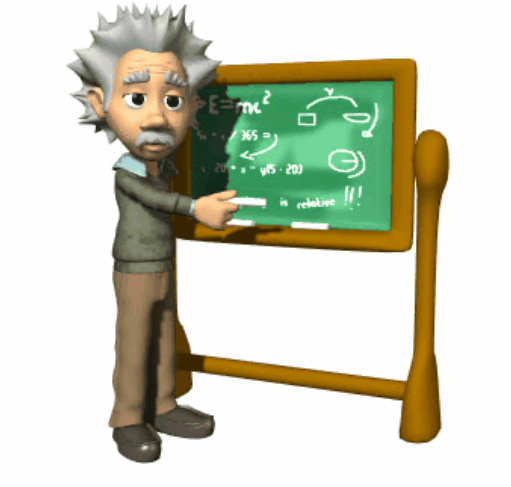
- p：大素数
- q: (p-1)的大素因子
- g：$g \in Z_p^*$，且满足$g^q=1 \bmod p$
- M：消息空间，$M \in Z_p^*$；
- S：签名空间；
- x：用户秘密钥，$1<x<q$
- y：用户公钥，$y \equiv g^x \bmod p$
- p, q, g, y为公钥，x为秘密钥。

签名过程

- 用户选择一次性秘密随机数k，$0<k<q$
- 计算：H(m)
- 计算：$r=g^k \bmod p$
- 签字方程：$ak = b + cx \bmod q$
- $\mathrm{Sig}_k(m)=(r, s)$作为签名，将m和(r, s)送给对方。

签名的验证过程

- 收端收到消息m和签名(r, s)后，可以按照以下验证方程检验：
- $\mathrm{Ver}(M, r, s)$=真 $\Leftrightarrow$ $r^a \equiv g^b y^c \bmod q$



所有以上的签名体制均可看成其一个特例！

签名算法中a, b, c的取值

	a	b	c	签字方程式
1	$\pm r$	$\pm s$	$\pm$ H(m)	$rk=s+\mathrm{H}(m)x \bmod q$
2	$\pm r$	$\pm\mathrm{H}(m)$	$\pm s$	$rk=\mathrm{H}(m)+sx \bmod q$
3	$\pm s$	$\pm r$	$\pm$ H(m)	$sk=r+H(m)x \bmod q$
4	$\pm$ s	H(m)	$\pm r$	$sk=\mathrm{H}(m)+rx \bmod q$
5	$\pm\mathrm{H}(m)$	$\pm s$	$\pm r$	$\mathrm{H}(m)k=s+rx \bmod q$
6	$\pm\mathrm{H}(m)$	$\pm r$	$\pm s$	$\mathrm{H}(m)k=r+sx \bmod \mathrm{q}$

签名方程$ak=b+cx$对应的签名算法

方案	签名方程	验证方程	对应体制
1	$rk=s+\mathrm{H}(m)x \bmod q$	$r^r=g^s y^{\mathrm{H}(m)} \bmod p$	Yen, Laih
2	$rk=\mathrm{H}(m)+sx \bmod q$	$r^r=g^{\mathrm{H(m)}}\mathrm{y}^s \bmod p$	Agnew, Yen
3	$rk=r+\mathrm{H}(m)\mathrm{x} \bmod \mathrm{q}$	$r^r=\mathrm{g}^{\mathrm{H}(m)}y^s \bmod p$	
4	$sk=\mathrm{H}(m)+rx \bmod q$	$r^s=g^{\mathrm{H}(m)}y^r \bmod p$	ElGamal, DSA
5	$\mathrm{H}(m)\mathrm{k}=s+rx \bmod q$	$r^{\mathrm{H}(m)}=g^s y^r \bmod p$	Schnorr, Nyberg
6	$\mathrm{H}(m)k=r+sx \bmod q$	$r^{\mathrm{H}(M)}=g^r y^s \bmod p$	

十一、其他签名体制——不可否认签名

1989年Chaum和Antwerpen提出不可否认签名的概念

- 为什么需要不可否认签名？普通签名可以精确地被复制，适合公开声明之类文件的散发；但是对于个人或公司信件，特别是有价值文件的签名，如果也可以随意复制和散发，就可能造成灾难。
- 这类签名要求在签名者合作下，才能验证签名。无签名者合作，不可能验证签名，从而可以防止复制和散布他所签的文件。
- 这一性质可以可以用于知识产权的保护等，在电子出版物的知识产权保护中将大显身手。产权拥有者可以控制产品的发布。

防失败签名（Fail-Stop Signature）

1991年由Pfitzmann和Waidner提出

- 这是一种强化安全性的数字签名，可防范有充足计算资源的攻击者。
- 当A的签名受到攻击，甚至在分析出A的私钥的情况下，攻击者也难以伪造A的签名。
- 同时，A也难以对自己的签名进行抵赖。
- 它是一次性签名方案，即给定密钥只能签署一个消息。

盲签名（Blind Signature）

1983年由Chuam最先提出

- 对于一般的数字签名来说，签名者总是要先知道文件内容后才签名，这是正常的应用情形。
- 但是有时需要某人对一个文件签名，但又不想让他知道所签署的文件内容。
- 在选举投票和数字货币协议中，会用到此签名体制。
- 盲签名在电子商务系统中，有重要的应用。

群签名（Group Signature）

1991年由Chaum和van Heyst最先提出

- 只有群中的成员才能代表群体签名。
- 接收到签名的人可以用公钥验证群签名，但不可能知道由群体中的那个成员所签。
- 发生争议时，由群体中的成员或可信赖机构识别群签名的签名者。
- 这类签名可以用于项目投标。

例如：所有参有投标的公司组成一个群体，且每个公司都匿名地采用群签名对自己的标书签名。当选中了一个满意的标书后，招标方就可以识别出签名的公司，而其他标书仍保持匿名。中标方若想反悔已无济于事，因为在没有他参与的情况下，仍可以正确地识别出签名人是谁。

代理签名（Proxy Signature）

1995年由Mambo等最先提出

- 代理签名就是某人授权其代理进行的签名，在委托签名时，签名密钥不交给代理人。代理签名有如下几个特性：

1. **不可区分性：**代理签名与委托人的签名不可区分；
2. **不可伪造性：**只有委托人和代理人可以建立合法的签名；
3. **代理签名的差异：**代理人若想伪造一个合法的代理签名，是不可行的，即伪造的代理签名可以被检测出来；
4. **可证实性：**签名验证人可以相信委托签名就是委托人认可的签名消息；
5. **可识别性：**委托人可从代理签名中确定出代理签名人的身份；
6. **不可抵赖性：**代理签名人不能抵赖已被接受的代理签名。

指定证实人签名（Designated Confirmer Signature）

此类签名于 1994年由Okamoto等最先提出

- 在一个机构中，指定一个人负责证实所有人的签名。
- 任何成员所签的文件都具有不可否认性，但证实工作均由指定人完成。
- 这种签名有助于防止签名失效。例如，在签名人的签名密钥确实丢失，或在他休假、病倒或去世时，都能对其签名进行验证。

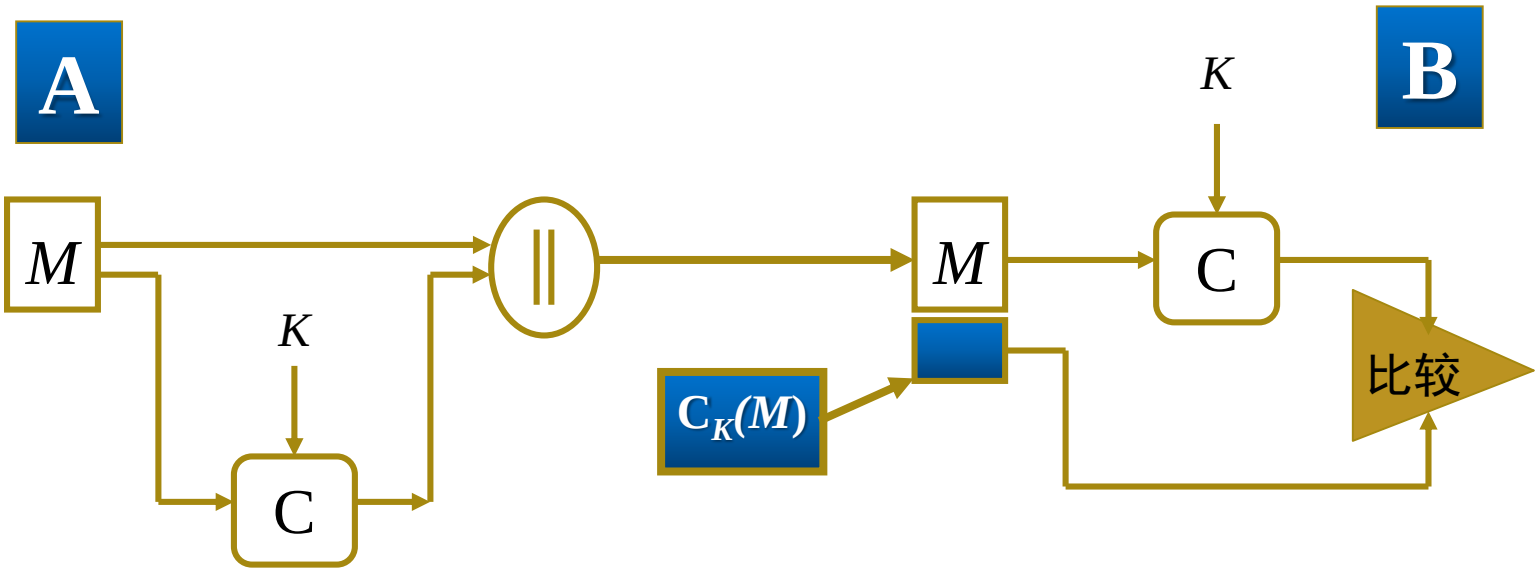
一次性签名（One-Time Signature）

1978年由Rabin最先提出

- 签名者至多只能对一个消息进行签名，否则签名就可能被伪造。
- 在公钥签名体制中，它要求对每个签名消息都要采用一个新的公钥作为验证参数。
- 一次性签名的优点是产生和验证速度都非常快，特别适用于计算能力比较低的芯片和智能卡实现。

十二、消息认证码MAC的基本用途

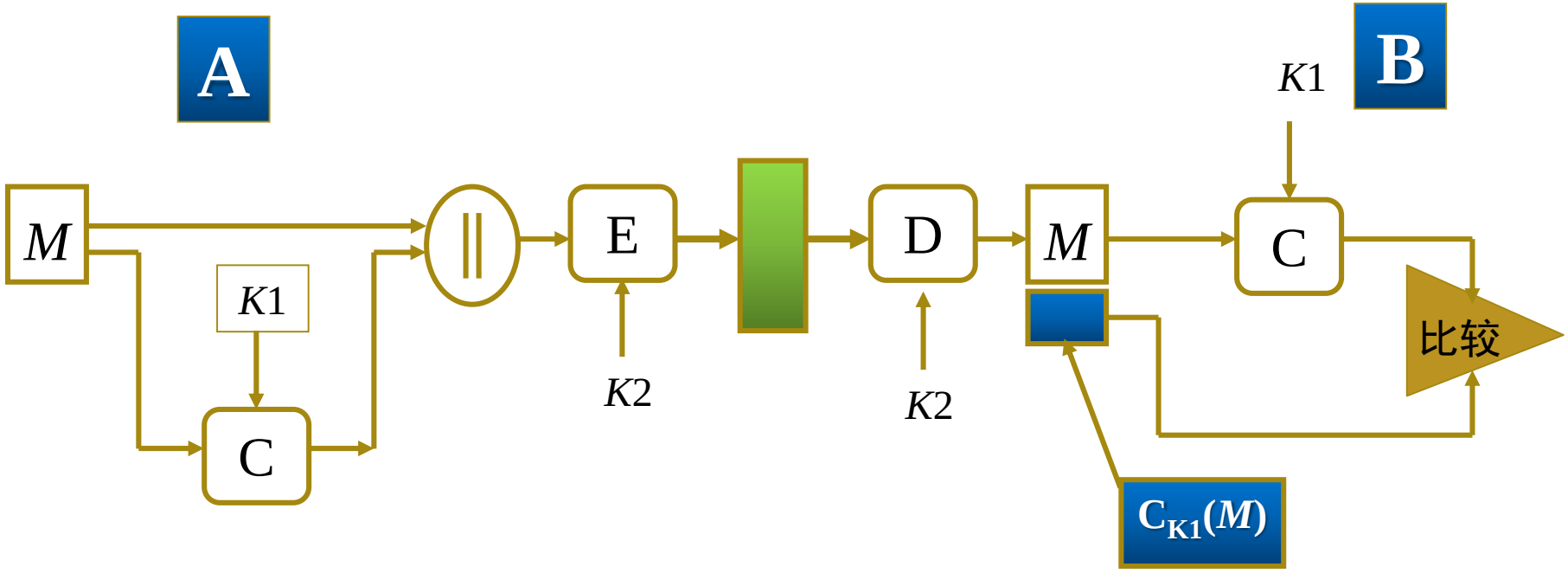
1 消息认证



注： *M*为消息，C为MAC函数，*K*共享的密钥

消息认证码MAC的基本用途

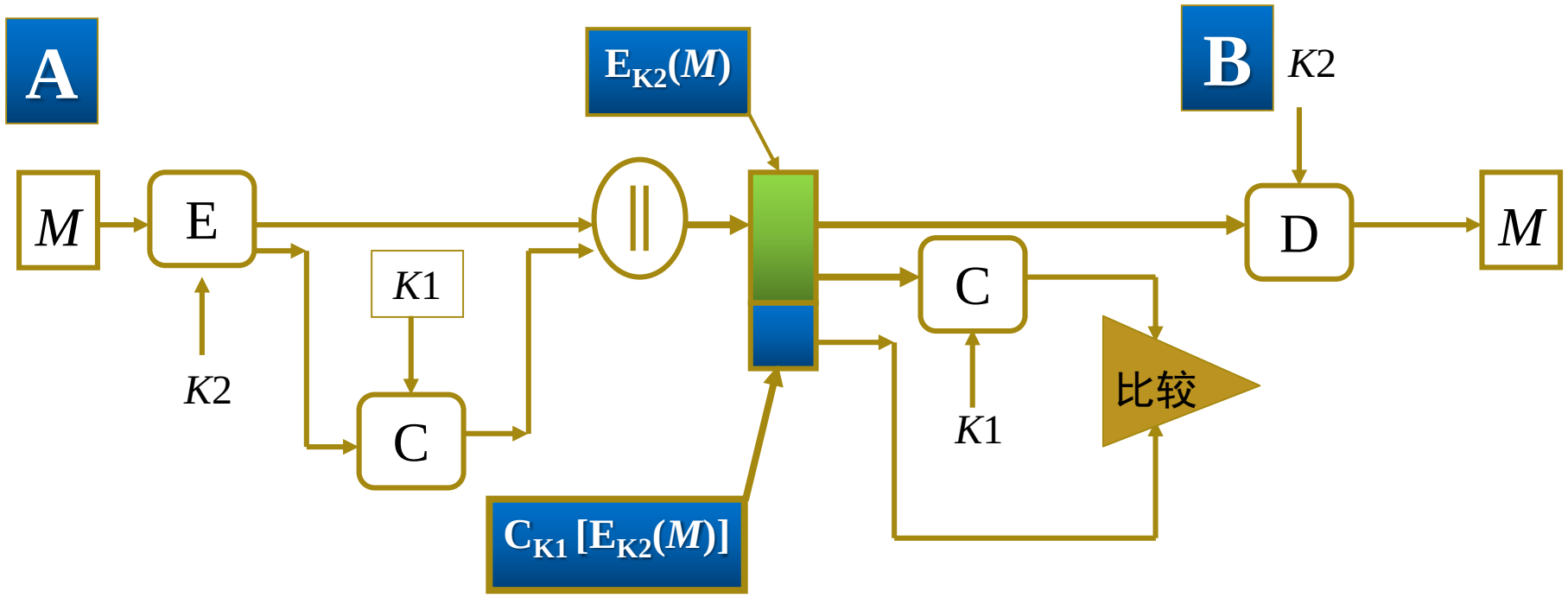
2 消息认证和保密：计算MAC后，再加密



注：这种认证方式最常用

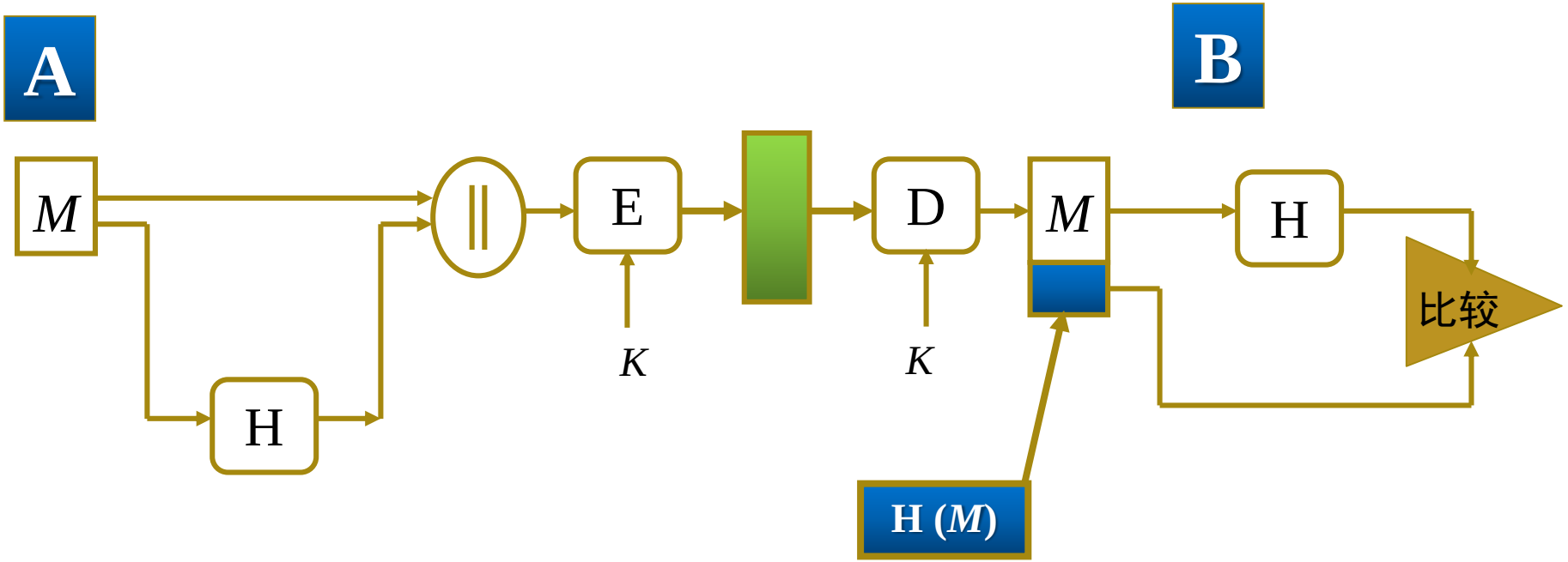
消息认证码MAC的基本用途

3 消息认证和保密：先加密，后计算MAC



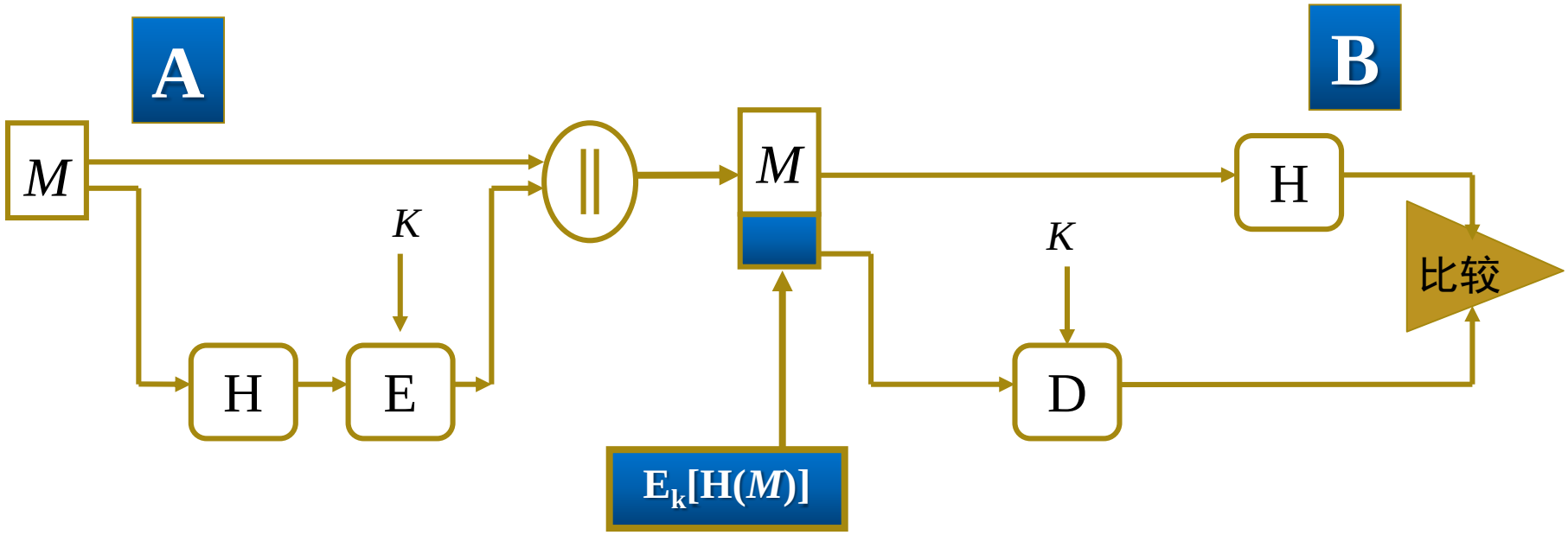
十三、杂凑算法/加密/签名结合应用方案

1 既提供保密性，又提供消息认证



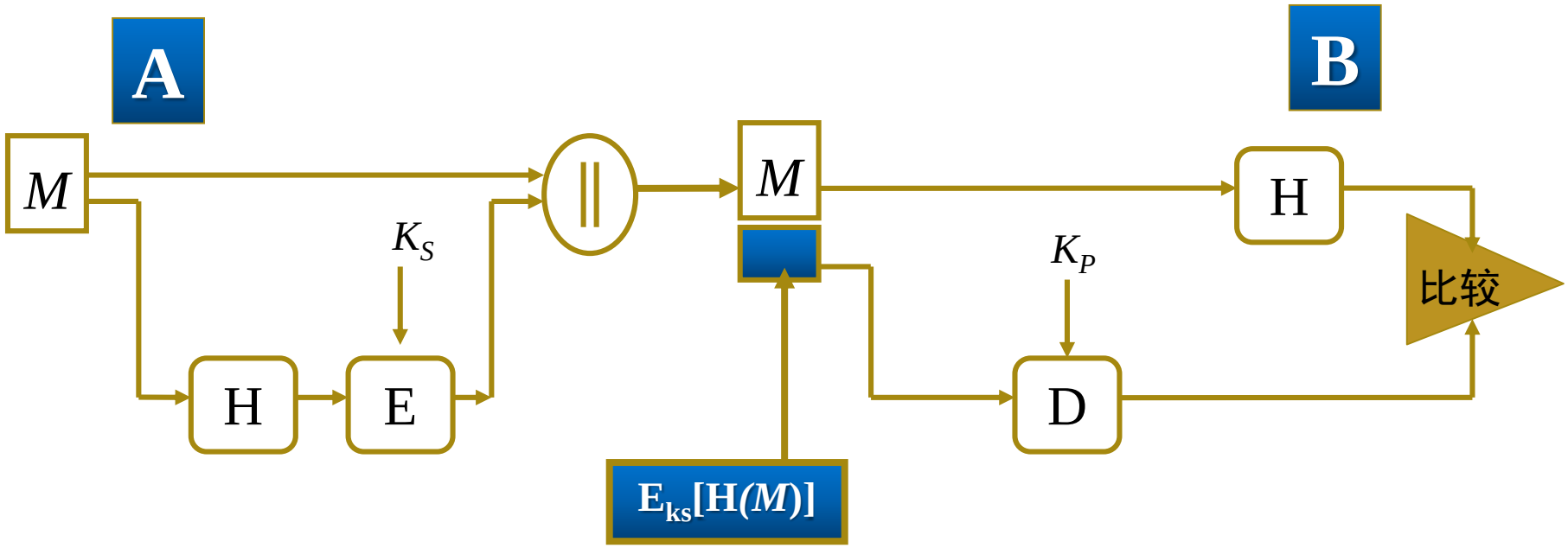
杂凑算法/加密/签名结合应用方案

2 仅提供消息认证



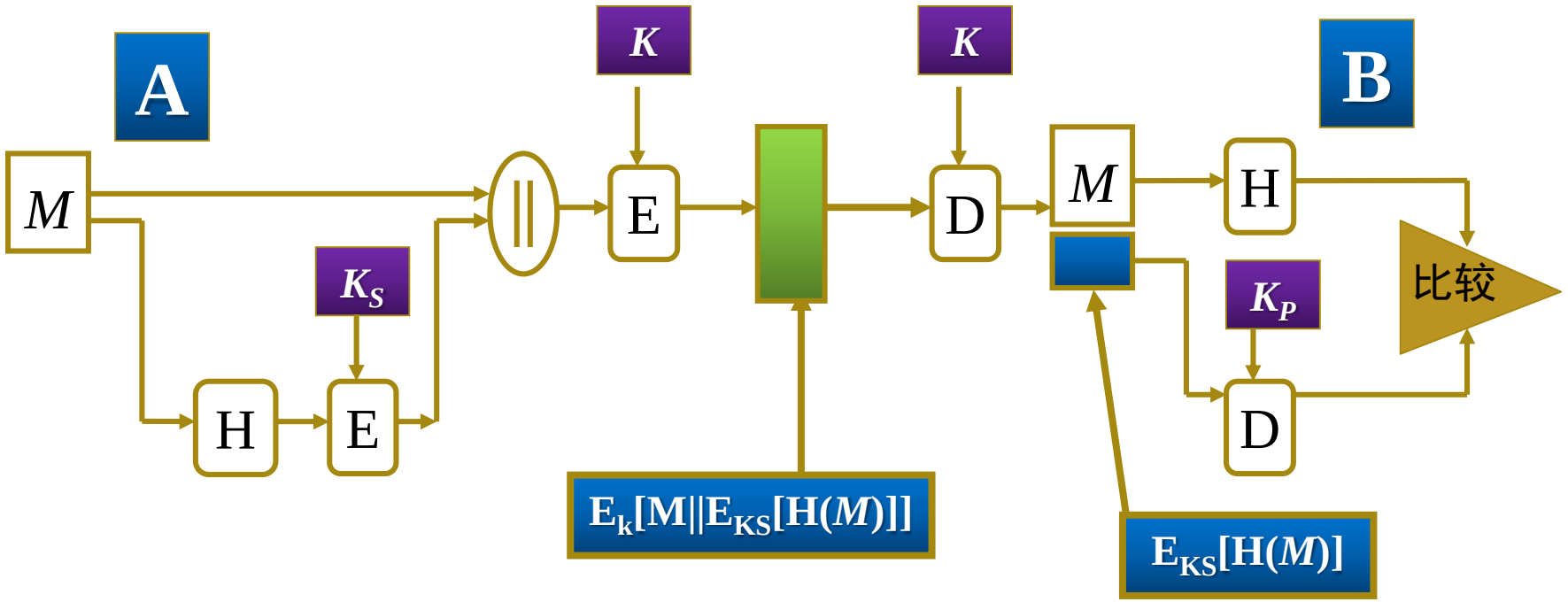
杂凑算法/加密/签名结合应用方案

3 既提供消息认证，又提供数字签名



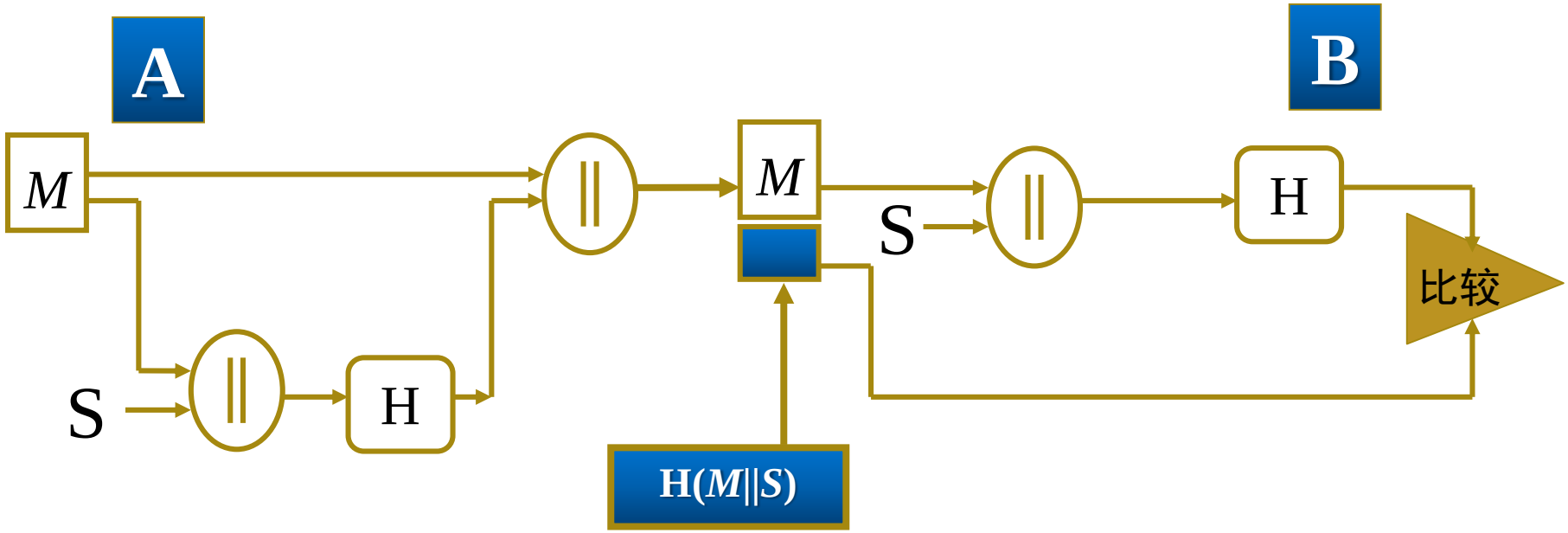
杂凑算法/加密/签名结合应用方案

4 既提供保密性，又提供消息认证和数字签名



杂凑算法/加密/签名结合应用方案

5 仅提供消息认证



谢谢！