ICS 35.040
L 80

中华人民共和国国家标准

GB/T 17901.3—2021

信息技术　安全技术　密钥管理
第3部分:采用非对称技术的机制

Information technology—Security techniques—Key management—Part 3:Mechanisms using asymmetric techniques

(ISO/IEC 11770-3:2015, MOD)

2021-03-09 发布　　2021-10-01 实施

国家市场监督管理总局
国家标准化管理委员会　发布

目　　次

前　言

GB/T 17901《信息技术　安全技术　密钥管理》拟分为6个部分：

——第1部分：框架；

——第2部分：采用对称技术的机制；

——第3部分：采用非对称技术的机制；

——第4部分：基于弱秘密的机制；

——第5部分：群组密钥管理；

——第6部分：密钥派生。

本部分为GB/T 17901的第3部分。

本部分按照GB/T 1.1—2009给出的的规则起草。

本部分使用重新起草法修改采用ISO/IEC 11770-3:2015《信息技术　安全技术　密钥管理　第3部分：采用非对称技术的机制》。

本部分与ISO/IEC 11770-3:2015的技术性差异及其原因如下：

——关于规范性引用文件，本部分做了具有技术性差异的调整，以适应我国的技术条件，调整的情况集中反映在第2章“规范性引用文件”中，具体调整如下：

- 删除了对ISO/IEC 10118、ISO/IEC 11770-1以及ISO/IEC 15946-1的引用；
- 增加了对GB/T 15843.3—2016、GB/T 17901.1—2020、GB/T 25069、GB/T 32905以及GB/T 32918系列标准的引用(见第2章)。

——删除了ISO/IEC 11770-3:2015中3.1～3.18和3.20～3.43的术语和定义，增加了3.1、3.2和3.4的术语和定义(见第3章)。

——增加了“本部分涉及使用椭圆曲线签名验证、公钥加解密的算法见GB/T 32918.2和GB/T 32918.4。”(见第5章)。

——修改了对附录C、附录D、附录E和附录F的引用关系(见第11章、第12章和第13章)。

——修改或删除了ISO/IEC 11770-3:2015中的附录C、附录D和附录E中与密钥管理具体实例相关的内容，并在附录C中增加对GB/T 32918.3—2016中规范的密钥派生函数的引用，在附录E中增加对GB/T 32918.3—2016规范的SM2密钥协商机制的引用。

——删除了ISO/IEC 11770-3:2015中的附录F和附录G。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本部分由全国信息安全标准化技术委员会(SAC/TC 260)提出并归口。

本部分起草单位：西安西电捷通无线网络通信股份有限公司、无线网络安全技术国家工程实验室、中关村无线网络安全产业联盟、北京大学深圳研究生院、国家密码管理局商用密码检测中心、国家无线电监测中心检测中心、中国电子技术标准化研究院、中国通用技术研究院、天津市无线电监测站、北京计算机技术及应用研究所、天津市电子机电产品检测中心、重庆邮电大学。

本部分主要起草人：杜志强、王月辉、朱跃生、周国良、陶洪波、李琴、铁满霞、张变玲、井经涛、李志勇、李冰、彭潇、刘科伟、黄振海、许玉娜、于光明、郎元、郑骊、颜湘、张国强、刘景莉、李冬、朱正美、商钧、王莹、赵慧、高德龙、方华、熊克琦、李玉娇、龙昭华、吴冬宇。

信息技术　安全技术　密钥管理
第3部分:采用非对称技术的机制

1　范围

GB/T 17901的本部分定义了基于非对称密码技术的密钥管理机制的要求、密钥派生函数、余子式乘法、密钥承诺、密钥确认、密钥管理框架、密钥协商、密钥传递、公钥传递。

本部分拟达到如下目的:

a)　通过密钥协商建立一个共享密钥,用于实体A和实体B间的对称加密。在密钥协商机制中,密钥通过实体A和实体B交换的数据计算得到,任何实体一方不能预先确定共享密钥值。

b)　通过密钥传递建立一个共享密钥,用于实体A和实体B间的对称加密。在密钥传递机制中,密钥由实体A选择,采用非对称密码保护技术,传给实体B。

c)　通过密钥传递将实体A的公钥传给其他实体。在公钥传递机制中,实体A的公钥经鉴别后传给其他实体,但不需保密。

本部分定义的一些机制基于GB/T 15843.3—2016相对应的鉴别机制。

本部分不包含以下密钥管理内容:

a)　密钥生存期管理;

b)　产生或确定非对称密钥对的机制;

c)　密钥的存储、存档、删除等机制。

本部分适用于采用非对称技术实现密钥管理的系统的研制,也可指导该类系统的检测。

注:本部分定义的机制不涉及实体私钥的分发,由公钥签名系统对密钥交换消息进行签名进行操作。

2　规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件,仅注日期的版本适用于本文件。凡是不注日期的引用文件,其最新版本(包括所有的修改单)适用于本文件。

GB/T 15843.3—2016　信息技术　安全技术　实体鉴别　第3部分:采用数字签名技术的机制

GB/T 17901.1—2020　信息技术　安全技术　密钥管理　第1部分:框架

GB/T 25069　信息安全技术　术语

GB/T 32905　信息安全技术　SM3密码杂凑算法

GB/T 32918.1　信息安全技术　SM2椭圆曲线公钥密码算法　第1部分:总则

GB/T 32918.2　信息安全技术　SM2椭圆曲线公钥密码算法　第2部分:数字签名算法

GB/T 32918.3—2016　信息安全技术　SM2椭圆曲线公钥密码算法　第3部分:密钥交换协议

GB/T 32918.4　信息安全技术　SM2椭圆曲线公钥密码算法　第4部分:公钥加密算法

3　术语和定义

GB/T 17901.1—2020和GB/T 25069界定的以及下列术语和定义适用于本文件。

3.1

椭圆曲线密码算法　elliptic curves cryptosystem

利用椭圆曲线上的有理点构成 Abel 加法群上椭圆离散对数的计算困难性的公钥密码算法。

3.2

权标　token

由与特定的通信相关的数据字段构成的消息，包含使用密码技术进行变换后的信息。

3.3

密钥承诺　key commitment

向对方确认己方公钥或者密钥权标的过程。

3.4

密钥确认　key confirmation

某实体确信另一个已识别的实体拥有正确的密钥的过程。

4　符号和缩略语

4.1　符号

下列符号适用于本文件。

A、B、C：实体 A、B、C 的区分标识。

BE：加密的数据块。

BS：带符号的数据块签名的数据块。

$Cert_A$：实体 A 的公钥证书。

$D_A()$：实体 A 的私有解密变换函数。

d_A：实体 A 的私有解密密钥。

E：椭圆曲线，给定在域 GF(p^m)上的方程 $Y^2=X^3+aX+b$，当 $p>3$ 并且 m 为正整数，给定在域 GF(2^m)上方程 $Y^2+XY=X^3+aX^2+b$，或者在域 GF(3^m)上的方程 $Y^2=X^3+aX^2+b$，有一额外的参考点 O_E 称为无穷远点，分别记为 E/GF(p^m)，E/GF(2^m)，或 E/GF(3^m)。

$E_A()$：实体 A 的公开加密变换函数。

e_A：实体 A 的公开加密密钥。

F()：密钥协商函数。

FP()：基于对的密钥协商函数。

F(h，g)：使用参数 h 和公共参数 g 作为输入的密钥协商函数。

G：以 n 为顺序 E 上的点。

GF(p^m)、GF(2^m)、GF(3^m)：对素数 $p>3$ 并且 m 为正整数 p^m、2^m、3^m 的有限域。

g：密钥协商函数 F 使用的所有实体共享的公共参数。

Hash()：杂凑函数。

h_A：实体 A 的私钥协商密钥。

j：用于余子式乘法中的余子式。

K：对称加密系统的密钥。

KT：密钥权标。

KT_A：实体 A 的密钥权标。

KT_{Ai}：实体 A 处理第 i 次交互后发送的密钥权标。

K_{AB}：在实体 A 与实体 B 间共享的密钥。

kdf()：密钥派生函数。

l：余子式乘法采用的值。

M：数据消息。

$MAC_K(Z)$：以密钥 K 和随机数据串 Z 作为输入的 MAC 算法输出。

n：椭圆曲线 E 在有限域中的序列的素因数。

O_E：椭圆曲线的无限远点。

P：椭圆曲线 E 上的点。

P_X：实体 A 在椭圆曲线 E 上的公开密钥协商密钥。

PKI_A：实体 A 的公开密钥信息。

parameter：密钥派生数函数使用的参数。

p_A、p_B：实体 A、B 的公开密钥协商密钥。

q：素数 $p \neq 3$ 且 $m \geqslant 1$ 的素数幂 p^m。

r：随机生成数。

r_A：实体 A 的密钥协商机制的随机数。

S_1、S_2、S_3：元素集合。

S_A：实体 A 的私有签名变换函数。

s_A：实体 A 的私有签名密钥。

T：可信第三方。

TVP：随时间变化的参数，如一随机数、一时间戳，或一序列号。

$Text_i$：包括在一数据块中第 i 个文本、数据或其他信息。

$V_A()$：实体 A 的公开验证变换函数。

v_A：实体 A 的公开验证密钥。

w()：单向函数。

$X(P)$：点 P 的 x 坐标。

$Y(P)$：点 P 的 y 坐标。

$\sqrt{q}$：一个正数 q 的平方根。

$\sharp E$：椭圆曲线 E 的顺序(或基数)。

$\|$：两数据元素连接。

$\lceil x \rceil$：大于或等于实数 x 的最小整数。

$\sum$：数字签名。

$\pi(P)$：$(X(P) \bmod 2^{\lceil \rho/2 \rceil}) + 2^{\lceil \rho/2 \rceil}$，其中 $\rho = \lceil \log_2 n \rceil$，$X(P)$为点 P 的 x 坐标。

4.2 缩略语

下列缩略语适用于本文件。

CA：认证机构(Certificate Authority)

MAC：消息鉴别码(Message Authentication Code)

5 要求

本部分要求所涉及的实体彼此知道其所声称的身份，这可通过两个实体间信息交换时所包含的标识符来实现，或从该机制所使用的上下文明显看出。身份验证意味着检查一个收到的标识符与某个已知的(可信的)或预期值是否相同。本部分所规范的机制的对象标识符见附录 A。

如果一个公钥属于某实体，那么该实体应确保它拥有相应的私钥(密钥注册见 GB/T 17901.1—2020)。

本部分中凡涉及采用密码技术解决机密性、完整性、真实性、不可否认性需求的应遵循密码相关国家标准和行业标准。其中，涉及使用椭圆曲线签名验证、公钥加解密的算法按 GB/T 32918.2 和GB/T 32918.4。

6 密钥派生函数

对于对称密码系统，不推荐使用由第 10 章派生的共享秘密而不做任何的进一步处理。通过密钥派生函数可导出共享密钥，建议使用一个单向函数作为密钥派生函数，譬如使用杂凑函数。

密钥派生函数产生的密钥与随机产生的密钥很难区分。但密钥派生函数需输入一个共享秘密和一组密钥派生参数，并产生输出所需长度的密钥。

为了让参与同一个密钥建立机制的双方能够协商出共同的密钥，密钥派生函数应事先商定。达成这一协定的方法超出了本部分的范围。

密钥派生函数实例参见附录 C。

7 余子式乘法

本章只适用使用椭圆曲线密码算法的机制。第 11 章描述的密钥协商机制和第 12 章、第 13 章描述的密钥传送机制都要求用户的私钥或者密钥权标和另一个实体的公钥或者密钥权标混合使用。在椭圆曲线密码算法中，如果另一个实体的公钥或者密钥权标不是有效的(即它不是椭圆曲线上的一个点，或者不在 n 阶子群中)，那么该操作的执行可能会导致私钥中的某些位泄露给攻击者，例如，“小子群攻击”。

有两种选择方法，可防止“小子群攻击”和类似的攻击：

一种方法是采用公钥鉴别法对从另一方接收到的公钥和密钥权标进行验证(见本部分的规定)。另一种方法是采用余子式乘法对公钥进行验证，余子式乘法将在第 11 章中具体规范。下面定义的 j 和 l 的值，将在余子式乘法中用到。

使用余子式乘法时有两种选择：

a) 如果不要求与未使用余子式相乘的实体兼容，则设 $j=\sharp E/n$ 和 $l=1$。在这种情况下，要求参与双方都应同意使用该选项，否则该机制不起作用。

b) 如果要求与未使用余子式相乘的实体兼容，则设 $j=\sharp E/n$ 和 $l=j^{-1} \bmod n$。

注：由于要求 $n>\sqrt[4]{q}$，因此 $\gcd(n,j)=1$，所以 $j^{-1} \bmod n$ 的值始终存在。

如果不需要用到余子式相乘，则 $j=l=1$。

无论是否使用了余子式相乘法，如果共享密钥(或者共享密钥的一个组成部分)的计算结果为无穷远点(O_E)，则用户认为密钥协商过程失败。

公钥验证法或余子式乘法特别适合下列情况：

a) 实体的公钥未经验证；

b) 密钥权标未经验证；

c) 用户的公钥希望长期使用。

如果另一实体的公钥已被验证且余子式较小，可被泄露的信息相当有限，则可不需完成这些测试。

8 密钥承诺

通过对秘密的密钥协商协议的密钥进行单向函数来建立密钥的机制在第 11 章描述。然而，某实体

可能在选择自己的私钥前就知道另一实体的公钥或密钥权标。所以，在该实体发现其他实体的公钥和选定自己私钥的间隔内，可以以生成 2^s 个私钥协商密钥候选值为代价，控制其所要建立的密钥中 s 个二进制比特位的值。

解决问题的方法是使用密钥承诺，这需要在协议中增加一个附加信息或协议交互来完成。密钥承诺的执行可以通过让第一实体对自己的公钥或者密钥权标进行杂凑变换并将杂凑码送给第二实体；第二实体接着回复自己的公钥或者密钥权标，然后第一实体回复自己的公钥或者密钥权标给第二实体，第二个实体可对它进行杂凑变换并验证杂凑码是否等于之前第一实体所发送的值。

9 密钥确认

显式密钥确认过程是通过添加附加消息到一个提供隐式密钥鉴别的密钥建立协议中，因此提供了显式密钥鉴别和实体鉴别。显式密钥确认可以被添加到任何不包含它本身的方法中。密钥确认通常是通过交换一个值来实现，这个值在很大的概率下只有密钥建立计算成功时才能正确计算出来。从实体 A 到实体 B 的密钥确认是通过实体 A 计算一个值并发送给实体 B 以确认实体 A 的计算。如果要求相互密钥确认，则每个实体需发送一个不同的值给对方。

隐式密钥确认是在已建立的密钥的后续使用中提供密钥确认，当发生某种错误时，就立即检测到。在这种情况下，不需显式密钥确认。如果一个实体不在线[如在单轮通信协议中，存储和转发(电子邮件)的场景]，另一个实体就不可能获得密钥确认。然而，有时建立的密钥在后来才使用，或者密钥建立过程的实体并不知道所产生的密钥是否会被立即使用。在这些情况下，通常需使用显式密钥确认方法，否则一旦错误检测出来纠正它就可能太晚。显式密钥确认可看作是密钥建立过程中具有“坚固”安全属性的一种方式，且可认作为是一个保守的协议设计。

使用 MAC 提供密钥确认的步骤如下：

实体 A 和 B 首先执行本部分中第 11 章和第 12 章中采用的其中一个密钥建立过程，期望共享一个的 MAC 密钥 K_{AB}，执行的步骤如下：

a) 实体 B 产生一个八位位组串的消息 M，包括：消息标识符八位位组 0x02，实体 B 的标识符，实体 A 的标识符，对应于实体 B 的密钥权标的八位位组串 KT_B(如不存在，省略)，对应于实体 A 的密钥权标的八位位组串 KT_A(如不存在，省略)，对应于实体 B 的密钥建立公钥 p_B(如不存在，省略)，对应于实体 A 的密钥建立公钥 p_A(如不存在，省略)，及可选的附加文本 Text1(如存在的话)，即：$M=02\|B\|A\|KT_B\|KT_A\|p_B\|p_A\|$Text1，其中 0x02 是消息序号。

b) 实体 B 计算 $K_B=\mathrm{kdf}(K_{AB})$，采用共享密钥 K_B，用一个合适的 MAC 机制对消息 M 计算出 $\mathrm{MAC}_{K_B}(M)$。

c) 实体 B 将消息 M 和 $\mathrm{MAC}_{K_B}(M)$发送给实体 A。

d) 实体 A 计算 $K_A=\mathrm{kdf}(K_{AB})$，用收到的消息 M 计算 $\mathrm{MAC}_{K_A}(M)$，并验证 $\mathrm{MAC}_{K_B}(M)=\mathrm{MAC}_{K_A}(M)$。

e) 假设 MAC 验证，实体 A 已收到实体 B 的密钥确认(即实体 A 知道 $K_A=K_B$)。如需相互密钥确认，则实体 A 继续执行协议，产生八位位组串的消息 M'，由几部分组成：消息标识符八位位组 0x03，实体 A 的标识符，实体 B 的标识符，对应于实体 A 的密钥权标的八位位组串 KT_A(如不存在，省略)，对应于实体 B 的密钥权标的八位位组串 KT_B(如不存在，省略)，对应于实体 A 的密钥建立公钥 p_A(如不存在，省略)，对应于实体 B 的密钥建立公钥 p_B(如不存在，省略)，以及可选的附加文本 Text2(如存在的话)，即：$M'=03\|A\|B\|KT_A\|KT_B\|p_A\|p_B\|$Text2，其中 0x03 是消息序号。

f) 实体 A 采用共享密钥 K_A，用一个合适的 MAC 机制计算 $\mathrm{MAC}_{K_A}(M')$。

g) 实体 A 将 M'和 $\mathrm{MAC}_{K_A}(M')$发送给实体 B。

h) 对消息 M'，实体 B 用 K_B 验证 $MAC_{K_A}(M')$。假设 MAC 验证，实体 B 已收到实体 A 的密钥确认（即实体 B 知道 $K_A=K_B$）。

也可采用其他确认密钥方法，如果共享密钥用于数据加密，那么一实体可将另一实体已知的一些特定的明文进行加密，并发送给该实体，如全 0 或全 1 的二进制块，但应注意后续使用该密钥时，不再加密用于密钥确认时使用过的同一明文。

10 密钥管理框架

10.1 概要

本章包含了对密钥建立机制框架的高级描述，定义了四种机制（双方密钥协商、三方密钥协商、私钥传送以及公钥传送）以及各自的使用要求，这些机制的特性参见附录 B。

10.2 双方密钥协商

本条适用于 11.1～11.11 描述的双方密钥协商的密钥协商机制。双方密钥协商是一个在 A 与 B 两实体间建立共享密钥的过程，任何一方不能预先确定共享密钥的值。密钥协商机制可以提供隐式密钥鉴别；在密钥建立的条件下，隐式密钥鉴别意味着机制实施后只有经确定的实体才可拥有正确的共享密钥。

A 与 B 两实体间的密钥协商发生在双方共享一段内容的条件下。该内容包括两个集合 S_1、S_2 和一个密钥协商函数 F。函数 F 应满足以下规定：

a) $F:S_1\times S_2\rightarrow S_2$ 将元素 $(h,g)\in S_1\times S_2$ 映射到 S_2 的元素，写成 $y=F(h,g)$；

b) F 满足可交换：

$F(h_A,F(h_B,g))=F(h_B,F(h_A,g))$；

c) 从 $F(h_1,g)$，$F(h_2,g)$ 和 g 计算得到 $F(h_1,F(h_2,g))$ 是困难的，意味着 $F(.,g)$ 是单向函数；

d) 实体 A 和 B 共享 S_2 中的一个公共元素 g，该元素是公开的；

e) 此设置下的实体可以高效计算函数值 $F(h,g)$，并有效地产生 S_1 中的随机元素。在特定的密钥协商中，可进一步包括一些条件。

注 1：附录 D 和附录 E 给出密钥协商函数 F 的例子。

注 2：如第 6 章中描述，在密钥协商机制的实际实现时对共享密钥做进一步处理。

注 3：一般来讲，若收到的函数值 $F(h,g)$ 为弱值，该协议将终止。

10.3 三方密钥协商

本条适用于描述 11.12 描述的三方之间密钥协商的密钥协商机制。三方密钥协商在三个实体 A，B，C 间建立一个共享密钥，三方中的任何一方不能预先确定共享密钥的值。三个实体 A，B，C 间的密钥协商发生在三方共享一段内容的条件下，该内容包括三个集合 S_1，S_2，S_3，密钥协商函数 F 和 FP。密钥协商函数 F 和 FP 应满足以下规定：

a) $F:S_1\times S_2\rightarrow S_2$ 将元素 $(h,g)\in S_1\times S_2$ 映射到 S_2 的元素，写成 $y=F(h,g)$；

b) F 满足可交换：

$F(h_A,F(h_B,g))=F(h_B,F(h_A,g))$；

c) 从 $F(h_1,g)$，$F(h_2,g)$ 和 g 计算 $F(h_1,F(h_2,g))$ 是困难的，意味着 $F(.,g)$ 是单向函数；

d) $FP:S_1\times S_2\times S_2\rightarrow S_3$ 将元素 $(h_C,F(h_A,g),F(h_B,g))\in S_1\times S_2\times S_2$ 映射到 S_3 的元素，写成：

$z=FP(h_C,F(h_A,g),F(h_B,g))$；

e) FP 满足可交换性：

$FP(h_C,F(h_A,g),F(h_B,g))=FP(h_C,F(h_B,g),F(h_A,g))=FP(h_B,F(h_A,g),F(h_C,g))$

$=FP(h_A,F(h_B,g),F(h_C,g))=FP(h_A,F(h_C,g),F(h_B,g))=FP(h_B,F(h_C,g),F(h_A,g))$；

f) 从 $F(h_A,g)$,$F(h_B,g)$,$F(h_C,g)$和 g 计算得到 $FP(h_C,F(h_A,g),F(h_B,g))$是困难的,意味着 $F(.,p_A,p_B)$是单向函数；

g) 实体 A,B,C 共享一个 S_2中的一个公共元素 g,该元素是公开的；

h) 此设置下的实体可以高效计算函数值 $F(h,g)$和 $FP(h_C,F(h_B,g),F(h_A,g))$,并有效地产生在 S_1中的随机元素。在特定的密钥协商中,可进一步包括一些条件。

注：如第 6 章描述,在密钥协商机制的实际实现中对共享密钥做进一步处理,共享派生密钥通常采用以下方法计算：

a) 直接从共享密钥 K_{ABC}提取一些比特位；或

b) 将共享密钥 K_{ABC}和选用一些非秘密数据经过一个单向函数计算,从输出中提取一些比特位。

10.4 秘密密钥传送

秘密密钥传送(也称密钥传递)是一个传递秘密密钥的过程,由一实体(或可信任中心)选择,传给另一实体,并通过非对称密码算法加密保护。

10.5 公钥传送

公钥传送使一实体的公钥可让其他实体以鉴别方式得到,公钥鉴别分发是一个基本的安全需求。分发主要通过以下两种方式：

a) 没有可信第三方的公钥分发；

b) 通过可信第三方的公钥分发,例如证书颁发机构。

实体 A 的公钥是实体 A 公钥信息的一部分,公钥信息至少包括实体 A 的区分标识符和实体 A 的公钥。

11 密钥协商

11.1 密钥协商机制 1

该密钥协商机制以非交互方式在实体 A 和实体 B 间建立包含相互隐式密钥鉴别的共享密钥,应满足以下要求：

a) 每个实体 X 在 S_1中有一个密钥协商私钥 h_X和一个密钥协商公钥 $p_X=F(h_X,g)$；

b) 每个实体可得到另一实体的已鉴别的密钥协商公钥副本,可用第 13 章描述的机制来实现。

密钥协商机制 1 如图 1 所示。该机制用于以离线方式在两个实体之间协商共享密钥。

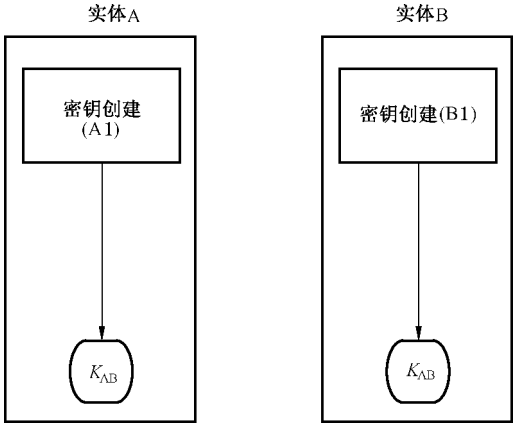

图 1 密钥协商机制 1

密钥创建(A1)：实体 A 用其私钥协商密钥 h_A和实体 B 的公钥协商密钥 p_B计算共享秘密密钥

$K_{AB}=F(h_A,p_B)$。

密钥创建(B1):实体 B 用其私钥协商密钥 h_B 和实体 A 的公钥协商密钥 p_A 计算共享私钥 $K_{AB}=F(h_B,p_A)$。在第 10 章中对 F 进行了规定,以保证密钥 K_{AB} 的两个计算值一致。

注 1:交互次数为 0。

注 2:该机制提供了相互隐式密钥鉴别。该非交互协议将产生相同的密钥。解决该问题的方法是确保密钥只用一次。也可用唯一的初始化矢量与密钥一起使用来解决该问题。

注 3:该机制不提供密钥确认。

注 4:该机制是一个密钥协商机制,它建立的密钥是实体 A 和实体 B 的私钥 h_A 和 h_B 的单向函数。然而,一实体可能在选择私钥前就知道另一实体的公钥。因此,如第 8 章所描述,一实体可选择所建立密钥的 s 位,代价是在发现另一个实体的公钥和选择私钥的间隔中,为私钥产生 2^s 个备选值。

11.2 密钥协商机制 2

该机制适用于参与协商密钥的两个实体中仅有一方知道对方身份,且只需在线交互一次的场景,采用实体 B 到实体 A 的隐式密钥鉴别方式,在实体 A 和实体 B 间经一次交互建立一个共享私钥,但实体 B 不鉴别实体 A(即实体 B 不知道和谁建立共享密钥)。该机制应满足以下要求:

a) 实体 B 在 S_1 中有一个密钥协商私钥 h_B 和一个密钥协商公钥 $p_B=F(h_B,g)$;

b) 实体 A 可以通过第 13 章描述的机制获得实体 B 已鉴别的密钥协商公钥 p_B 的副本。

密钥协商机制 2 如图 2 所示。

密钥权标创建(A1):实体 A 随机并秘密产生 r(r 在 S_1 中),计算 $F(r,g)$,并将密钥权标:$KT_{A1}=F(r,g)\|\text{Text}$ 发送给实体 B。

密钥创建(A2):实体 A 计算共享密钥 $K_{AB}=F(r,p_B)$。

密钥创建(B1):实体 B 从接收到的密钥权标 KT_{A1} 中提取 $F(r,g)$,并计算共享密钥:$K_{AB}=F(h_B,F(r,g))$。在第 10 章中对 F 进行了规定,以保证密钥 K_{AB} 的两个计算值一致。

注 1:交互次数为 1。

注 2:该机制提供从实体 B 到实体 A 的隐式鉴别(实体 B 是除了实体 A 之外唯一可计算出共享密钥的实体)。

注 3:该机制不提供密钥确认。

注 4:该机制是一个密钥协商机制,它建立的密钥是实体 A 提供的随机值 r 和实体 B 的密钥协商私钥的单向函数。因此,如第 8 章所描述,实体 A 可在选取 r 的值之前就得到实体 B 的公钥,实体 A 可选择所建立密钥的 s 位,代价是在发现实体 B 的公钥和发送 KT_{A1} 的间隔中,为 r 产生 2^s 个备选值。

注 5:由于实体 B 是从未鉴别的实体 A 收到可计算出密钥 K_{AB} 的信息,实体 B 对 K_{AB} 的使用被限制在不需实体 A 鉴别的范围内,例如,解密和消息鉴别码的生成。

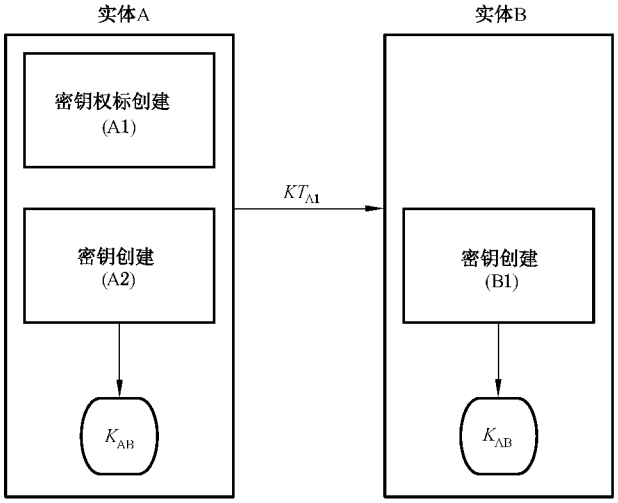

图 2 密钥协商机制 2 及机制 8

11.3 密钥协商机制 3

该机制适用于参与协商密钥的两个实体彼此知道对方身份，且只需在线交互一次的场景，通过相互隐式密钥鉴别以及实体 A 到实体 B 的实体鉴别，在实体 A 和实体 B 间经一次交互建立共享秘密密钥。应满足以下要求：

a) 实体 A 拥有非对称签名系统(S_A,V_A)。

b) 实体 B 可得到已鉴别的公开验证变换 V_A 的副本，可通过第 13 章所描述的机制实现。

c) 实体 B 有密钥(h_B,p_B)的密钥协商方案。

d) 实体 A 可获取已鉴别的实体 B 密钥协商公钥 p_B 的副本，可通过第 13 章所描述的机制实现。

e) (可选)如果被使用，TVP 可为时间戳或序列号。如果使用时间戳，需有安全和同步的时钟；如果使用序列号，则需维护和校验双边计数器。

f) 实体 A 和实体 B 在 MAC 函数和使用 K_{AB} 作为该 MAC 函数密钥的方法上应一致。MAC 函数参见 ISO/IEC 9797。

密钥协商机制 3 如图 3 所示。

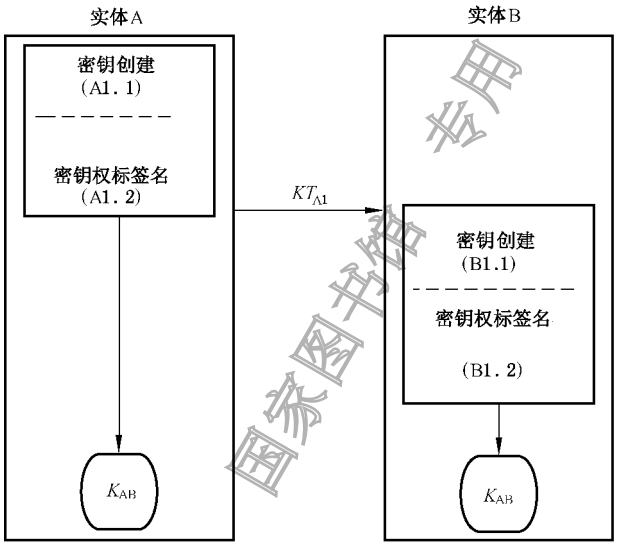

图 3 密钥协商机制 3

密钥创建(A1.1)：实体 A 随机并秘密产生 r(r 在 S_1 中)，计算 F(r,g)，按 K_{AB}=F(r,p_B)计算共享秘密密钥。实体 A 使用共享秘密密钥 K_{AB} 对发送者实体 A 的区分标识符及可选的 TVP(时间戳或序列号)的级联，计算出 MAC。

密钥权标签名(A1.2)：实体 A 用其私有签名转换 S_A 对 MAC 进行签名。实体 A 形成密钥权标，它由发送者实体 A 的区分标识符、密钥输入 F(r,g)、TVP(可选)、已签名的 MAC 和一些可选数据组成，即

$$KT_{A1}=A\|F(r,g)\|TVP\|S_A(MAC_{K_{AB}}(A\|TVP))\|Text1$$

然后将它发送给实体 B。

密钥创建(B1.1)：实体 B 从接收到的密钥权标中提取出 F(r,g)，并用其私钥协商密钥 h_B 计算共享秘密密钥，K_{AB}=F(h_B,F(r,g))。

实体 B 使用共享秘密密钥 K_{AB} 对发送者实体 A 的区分标识符和(可选)TVP 计算出 MAC。

签名验证(B1.2)：实体 B 使用发送者的公开验证转换 V_A 验证实体 A 的签名以及接收到的密钥权标 KT_{A1} 的完整性和来源。然后验证权标的及时性[通过检查(可选)TVP]。

注 1：交互次数为 1。

注 2：该机制提供从实体 A 到实体 B 的显式密钥鉴别和从实体 B 到实体 A 的隐式密钥鉴别。

注 3：该机制提供从实体 A 到实体 B 的密钥确认。

注 4：该机制是一个密钥协商机制，它建立的密钥是实体 A 提供的随机值 r 和实体 B 的私钥协商密钥的单向函数，如第 8 章所描述，实体 A 可在选取 r 的值之前就得到实体 B 的公钥，实体 A 可选择所建立密钥的 s 位，代价是在发现实体 B 的公钥和发送 KT_{A1} 的间隔中，为 r 产生 2^s 个备选值。

注 5：(可选)TVP 防止从实体 A 向实体 B 的密钥权标重放。

注 6：如果 Text1 被用来传送实体 A 的公钥证书，那么 11.3 的 b)可放宽到实体 B 拥有一个已鉴别的 CA 公开密钥副本。

11.4 密钥协商机制 4

该机制适用于参与协商密钥的两个实体彼此不知道对方身份，且在线交互两次的场景，实体 A 和实体 B 事先没有交换密钥信息，通过采用联合密钥控制方法，经两次交互建立共享秘密密钥。该机制既不提供实体鉴别也不提供密钥鉴别。密钥协商机制 4 如图 4 所示。

密钥权标创建(A1)：实体 A 随机并秘密产生 r_A(r_A在 S_1 中)，计算 $F(r_A,g)$，构造密钥权标 $KT_{A1}=F(r_A,g)\|Text1$，然后发给实体 B。

密钥权标创建(B1)：实体 B 随机并秘密产生 r_B(r_B在 S_1 中)，计算 $F(r_B,g)$，构造密钥权标 $KT_{B1}=F(r_B,g)\|Text2$，然后发给实体 A。

密钥创建(A2)：实体 A 从收到的密钥权标 KT_{B1} 中提取 $F(r_B,g)$，并计算共享密钥 $K_{AB}=F(r_A,F(r_B,g))$。

密钥创建(B2)：实体 B 从收到的密钥权标 KT_{A1} 中提取 $F(r_A,g)$，并计算共享密钥 $K_{AB}=F(r_B,F(r_A,g))$。

注 1：交互次数为 2。

注 2：该机制不提供显式或隐式密钥鉴别，但该机制可用于通过其他方式核实密钥权标的真实性的情况。例如，可通过第二通信信道在实体间交换密钥权标的杂凑码。参见公钥传送机制 2。

注 3：可采用单独的信道或方法核实密钥权标。

注 4：该机制不提供密钥确认。

注 5：该机制是一个密钥协商机制，它建立的密钥是实体 A 和实体 B 提供的随机值 r_A 和 r_B 的单向函数，如第 8 章所描述，实体 B 可在选择 r_B 的值之前就得到 $F(r_A,g)$，实体 B 可选择所建立密钥的 s 位，代价是在接收 KT_{A1} 和发送 KT_{B1} 的间隙中，为 r_B 产生 2^s 个备选值。

密钥协商机制 4 如图 4 所示。

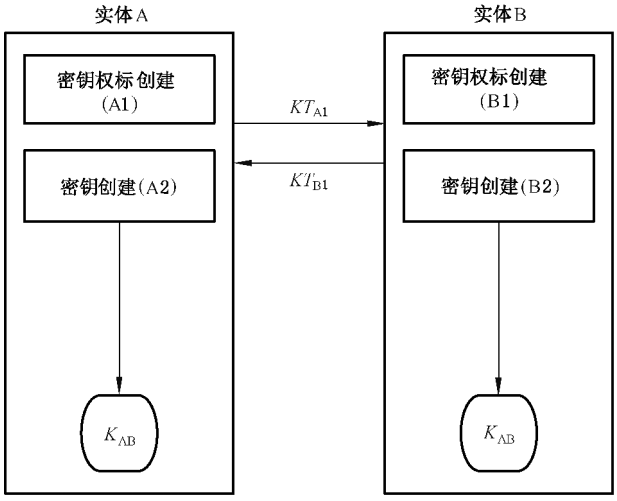

图 4 密钥协商机制 4,5,9

11.5 密钥协商机制 5

该机制适用于参与协商密钥的两个实体彼此知道对方身份，且在线交互两次的场景，通过相互显式密钥鉴别和联合密钥控制方法，经两次交互在实体 A 和实体 B 间建立一个共享秘密密钥。应满足以下要求：

a) 每个实体 X 有一个私钥协商密钥 h_X（h_X 在 S_1 中）和一个公钥协商密钥 $p_X=\mathrm{F}(h_X,\mathrm{g})$；

b) 每个实体可获取其他实体的已鉴别的公钥协商密钥副本，可通过第 13 章所描述的机制实现；

c) 两个实体有共同的单向函数 w。

密钥协商机制 5 如图 4 所示。

密钥权标创建(A1)：实体 A 随机和秘密地产生 r_A（r_A 在 S_1 中），计算 $\mathrm{F}(r_A,g)$，并将密钥权标 $KT_{A1}=\mathrm{F}(r_A,g)\|\mathrm{Text1}$ 发送给实体 B。

密钥权标创建(B1)：实体 B 随机和秘密地产生 r_B（r_B 在 S_1 中），计算 $\mathrm{F}(r_B,g)$，并将密钥权标 $KT_{B1}=\mathrm{F}(r_B,g)\|\mathrm{Text2}$ 发送给实体 A。

密钥创建(B2)：实体 B 从收到的密钥权标 KT_{A1} 中提取 $\mathrm{F}(r_A,g)$，并计算共享秘密密钥：

$K_{AB}=\mathrm{w}(\mathrm{F}(h_B,\mathrm{F}(r_A,g))\|\mathrm{F}(r_B,p_A))$，其中 w 为单向函数。

密钥创建(A2)：实体 A 从收到的密钥权标 KT_{B1} 中提取 $\mathrm{F}(r_B,g)$，并计算共享秘密密钥：

$K_{AB}=\mathrm{w}(\mathrm{F}(r_A,p_B)\|\mathrm{F}(h_A,\mathrm{F}(r_B,g)))$。

注 1：交互次数为 2。

注 2：该机制提供相互隐式密钥鉴别。如果数据域 Text2 含有一个由密钥 K_{AB} 对已知数据计算得到的 MAC，则该机制提供实体 B 到实体 A 的显式密钥鉴别。

注 3：如果数据域 Text2 含有一个由密钥 K_{AB} 对已知数据计算得到，则该机制提供实体 B 到实体 A 的密钥确认。

注 4：该机制是一个密钥协商机制，它建立的密钥是实体 A 和实体 B 提供的随机值 r_A 和 r_B 的单向函数。

注 5：如果 Text1 和 Text2 分别包含实体 A 和实体 B 的密钥协商密钥的公钥证书，则 11.5 的 b)替代为每个实体拥有一个已鉴别的 CA 公开验证密钥副本。

注 6：在某种情况下该机制可能会遭受源替换攻击，考虑到这种情况，可以这样来避免这种攻击：作为提交公钥给 CA 获得证书的一部分，提交者应证明其持有相应私钥。该类攻击对于基于椭圆曲线的协议更为严重。

11.6 密钥协商机制 6

该机制适用于参与协商密钥的两个实体彼此鉴别对方身份，且在线交互两次的场景，基于同时使用非对称加密方案和签名系统，通过相互隐式密钥鉴别和联合密钥控制方法，经两次交互在实体 A 和实体 B 间建立一个共享密钥。应满足以下要求：

a) 实体 A 具有非对称加密系统及变换($\mathrm{E_A}$，$\mathrm{D_A}$)；

b) 实体 B 具有非对称签名系统及变换($\mathrm{S_B}$，$\mathrm{V_B}$)；

c) 实体 A 可得到经鉴别的实体 B 公钥验证变换 $\mathrm{V_B}$ 副本，可使用第 13 章的机制来实现；

d) 实体 B 可得到经鉴别的实体 A 的公钥加密变换 $\mathrm{E_A}$ 副本，可使用第 13 章的机制来实现。

密钥协议机制 6 如图 5 所示。

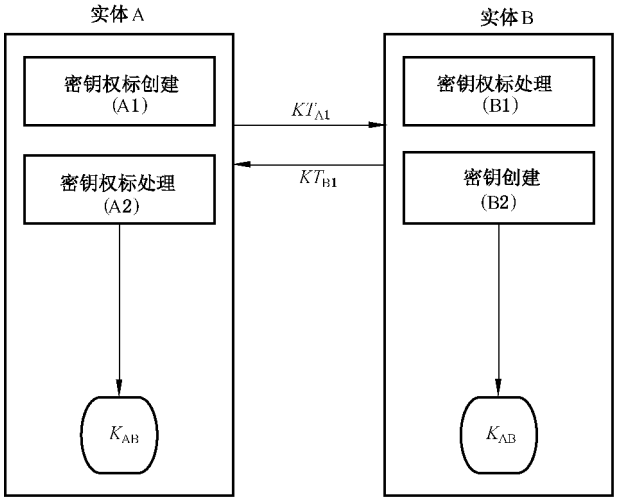

图 5　密钥协商机制 6

密钥权标创建(A1)：实体 A 产生随机数 r_A，创建密钥权标 $KT_{A1}=r_A \| \text{Text1}$，并将其发送给实体 B。

密钥权标处理(B1)：实体 B 产生随机数 r_B，使用其私有签名变换 S_B，对数据块签名，该数据块的组成是：实体 A 的区分标识符、随机数 r_A、随机数 r_B 和可选数据文本 Text2，得到 $BS=S_B(A \| r_A \| r_B \| \text{Text2})$。

然后实体 B 用实体 A 的基于公钥加密变换 E_A 对数据块加密，该数据块的组成是：区分标识符(可选)、签名块 BS 及可选的数据 Text3，实体 B 将密钥权标 $KT_{B1}=E_A(BS \| \text{Text3}) \| \text{Text4}$ 返回给实体 A，或实体 B 可包括其标识符 B：$KT_{B1}=E_A(B \| BS \| \text{Text3}) \| \text{Text4}$。

密钥创建(B2)：经密钥派生函数，共享秘密密钥由签名块 BS 中实体 B 所有或其中一部分签名 $\sum$ 组成。

密钥权标处理(A2)：实体 A 使用其私有解密变换 D_A 解密密钥权标 KT_{B1}，检查发送者的标识(可选)，并采用实体 B 的公钥验证变换 V_B 验证签名块 BS 的数字签名。然后实体 A 检查接收者标识符和签名块 BS 中随机数 r_A 是否等于在发送的权标 KT_{A1} 中的随机数 r_A。如果所有检查都成功，实体 A 接受所有采用密钥派生函数的签名块 BS 中实体 B 所有或其中一部分签名作为共享秘密密钥。

注 1：交互次数为 2。

注 2：将签名 $\sum$ 部分作为实体 A 和实体 B 间建立的秘密密钥的基础，需提前商定。

注 3：该机制提供从实体 A 与实体 B 的隐式密钥鉴别和从实体 B 向实体 A 的显式密钥鉴别。

注 4：如果 Text3 包含用密钥 K_{AB} 对已知数据计算出的 MAC，则该机制提供实体 B 向实体 A 密钥确认。

注 5：该机制是一个密钥协商机制，它建立的密钥是由实体 A 和实体 B 提供的随机值 r_A 和 r_B 的单向函数，如第 8 章所讨论，实体 B 可在选择 r_B 的值之前就得到 $F(r_A, g)$，实体 B 可选择所建立密钥的 s 位，代价是在接收 KT_{A1} 和发送 KT_{B1} 的间隙中，为 r_B 产生 2^s 个备选值。

注 6：如果 Text1 和 Text4 分别包含有实体 A 加密密钥的公钥证书的和实体 B 验证密钥的公钥证书，则 11.6 的 c)和 d)可放宽到每个实体拥有已鉴别的 CA 公开验证密钥副本。

注 7：该机制有一个显著特征，实体 B 的身份可对窃听者匿名，这是无线通信环境中有潜在意义的属性。

11.7　密钥协商机制 7

该机制适用于参与协商密钥的两个实体采用 GB/T 15843.3—2016 规定的三轮交互鉴别机制彼此鉴别对方身份，且在线交互两次的场景，通过相互鉴别在实体 A 和 B 间建立共享秘密密钥。应满足以下要求：

a) 每个实体 X 有一个非对称签名系统(S_X,V_X);

b) 实体双方均可得到另一实体已鉴别的公开验证变换副本,可使用第 13 章的机制来实现;

c) 两实体使用同一 MAC 函数。

密钥协商机制 7 如图 6 所示。

密钥权标创建(A1):实体 A 随机及秘密生成 r_A(r_A在 S_1中),计算 F(r_A,g),构建密钥权标KT_{A1}=F(r_A,g)‖Text1,并将其发送给实体 B。

密钥权标处理与密钥创建(B1):实体 B 随机及秘密生成 r_B(r_B在 S_1中),计算 F(r_B,g),计算共享密钥:K_{AB}=F(r_B,F(r_A,g)),并构建签名密钥权标 $KT_{B1}=S_B(DB_1)\|MAC_{K_{AB}}(DB_1)\|$Text3,其中 DB_1=F(r_B,g)‖F(r_A,g)‖A‖Text2,并送回给实体 A。

通过在 KT_{B1}中包含 $MAC_{K_{AB}}(DB_1)$提供密钥确认。如果双方有一个共同的对称加密系统,用 $KT_{B1}=F(r_B,g)\|E_{K_{AB}}(S_B(DB_1))$取代 KT_{B1}提供密钥确认,其中 E 是对称加密函数。

密钥权标处理(A2):实体 A 使用实体 B 的公开验证密钥验证实体 B 对密钥权标 KT_{B1}的签名,然后验证 A 的区分标识符以及步骤(A1)中 F(r_A,g)值。如果检查成功,实体 A 计算共享秘密密钥:

K_{AB}=F(r_A,F(r_B,g))。

实体 A 使用 K_{AB},验证 $MAC_{K_{AB}}(DB_1)$,构造经签名的密钥权标:$KT_{A2}=S_A(DB_2)\|MAC_{K_{AB}}(DB_2)\|$Text5,其中 DB_2=F($r_{A,}g$)‖F(r_B,g)‖B‖Text4,并发送给实体 B。

通过在 KT_{A2}中包含 $MAC_{K_{AB}}(DB_2)$提供密钥确认。也可用 $KT_{A2}=E_{K_{AB}}(S_A(DB_2))$取代 KT_{A2}提供密钥确认。

密钥权标处理(B2):实体 B 使用实体 A 的公开验证密钥验证实体 A 对密钥权标 KT_{A2}的签名,然后验证实体 B 的区分标识符以及 F(r_A,g)和 F(r_B,g)的值是否与之前协商的一致。如果验证成功,实体 B 使用 K_{AB}=F(r_B,F(r_A,g))验证 $MAC_{K_{AB}}(DB_2)$。

注 1:交互次数为 3。

注 2:该机制提供相互显式密钥鉴别和相互实体鉴别。

注 3:该机制提供相互密钥确认。

注 4:该机制是一个密钥协商机制,它建立的密钥是实体 A 和实体 B 提供的随机值 r_A和 r_B的单向函数,如第 8 章所讨论实体 B 可在选择 r_B的值之前就得到 F(r_A,g),实体 B 可选择所建立密钥的 s 位,代价是在接收 KT_{A1}和发送 KT_{B1}的间隙中,为 r_B产生 2^s个备选值。

注 5:该机制符合 GB/T 15843.3—2016。KT_{A1},KT_{B1}和 KT_{A2}与 GB/T 15843.3—2016 的 5.2.2 描述的三轮交互鉴别机制的权标相同。TVP 也相同,在下列使用上做些改变:将 R_A设置为 F(r_A,g);以及将 R_B设置为 F(r_B,g)。

注 6:如果数据域 Text1 和 Text3(或 Text5 和 Text3)分别包含实体 A 和实体 B 的公钥证书,则 11.7 的 b)可放宽要求。

注 7:每个实体拥有一份已鉴别的 CA 公开验证密钥副本。

注 8:如果使用带有文本杂凑的签名机制,则 F(r_A,g)及/或 F(r_B,g)不需在密钥权标 KT_{B1}中发送。同样,F(r_A,g),F(r_B,g)不需在密钥权标 KT_{A2}中发送,但在各签名的计算中包含随机数。

注 9:密钥确认也可通过加密签名中的一部分来实现,在这种情况下,在 11.7 中的 c)可不采用。

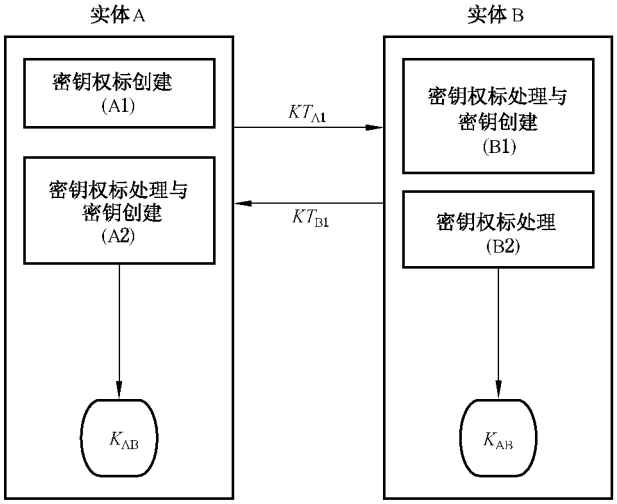

图 6　密钥协商机制 7

11.8　密钥协商机制 8

该机制适用于参与协商密钥的两个实体采用椭圆曲线密码算法，且在线交互一次的场景，通过相互隐式密钥鉴别及一次交互在实体 A 和实体 B 间建立一个共享秘密密钥。应满足以下要求：

a)　每个实体 X 在 S_1 中都有一个私钥协商密钥 h_X 和一个公钥协商密钥 $p_X=\mathrm{F}(h_X,g)$；

b)　每个实体可得到另一个实体已鉴别的公钥协商密钥副本，可使用第 13 章的机制来实现。

密钥协商机制 8 如图 2 所示。

l 和 j 值用于余子式乘法(见第 7 章)。还需采用一个函数将一椭圆点 P 转换为整数，例如，函数 $\pi(P)=(\mathrm{X}(P)\bmod 2^{\lceil \rho/2 \rceil})+2^{\lceil \rho/2 \rceil}$，其中 $\rho=\lceil \log_2 n \rceil$，$\mathrm{X}(P)$ 为点 P 的 x 坐标。

密钥权标创建(A1)：实体 A 随机并秘密地产生 r_A(r_A 在 S_1 中)，计算 $\mathrm{F}(r_A,g)$，构建密钥权标 $KT_{A1}=\mathrm{F}(r_A,g)$，并将其发送给实体 B。

密钥创建(A2)：实体 A 计算共享秘密密钥：$K_{AB}=((r_A+\pi(KT_{A1})h_A)\cdot l)(j\cdot(P_B+\pi(P_B)P_B))$

密钥创建(B1)：实体 B 计算共享秘密密钥：$K_{AB}=((h_B+\pi(P_B)h_B)\cdot l)(j\cdot(KT_{A1}+\pi(KT_{A1})P_A))$

注 1：交互次数为 1。

注 2：该机制提供相互隐式密钥鉴别。

11.9　密钥协商机制 9

该机制适用于参与协商密钥的两个实体采用椭圆曲线密码算法，且在线交互两次的场景，通过相互隐式密钥鉴别，经两轮交互在实体 A 和实体 B 间建立一个共享秘密密钥。应满足以下要求：

a)　每个实体 X 在 S_1 中有一个私钥协商密钥 h_X 和一个公钥协商密钥 $p_X=\mathrm{F}(h_X,g)$；

b)　每个实体可得到另一个实体已鉴别的公钥协商密钥副本，可使用第 13 章的机制来实现。

密钥协商机制 9 如图 4 所示。

l 和 j 值用于余子式乘法(见第 7 章)。还需采用一个函数将一椭圆点 P 转换为整数，例如，函数 $\pi(P)=(X(P)\bmod 2^{\lceil \rho/2 \rceil})+2^{\lceil \rho/2 \rceil}$，其中 $\rho=\lceil \log_2 n \rceil$，$\mathrm{X}(P)$ 为点 P 的 x 坐标。

密钥权标创建(A1)：实体 A 随机及秘密地产生 r_A(r_A 在 S_1 中)，计算 $\mathrm{F}(r_A,g)$，构建密钥权标 $KT_{A1}=\mathrm{F}(r_A,g)$，然后发给实体 B。

密钥权标创建(B1)：实体 B 随机及秘密地产生 r_B(r_B 在 S_1 中)，计算 $\mathrm{F}(r_B,g)$，构建密钥权标 KT_{B1}

$=\mathrm{F}(r_B,g)$，然后发给实体 A。

密钥创建(A2)：实体 A 计算共享密钥：$K_{AB}=((r_A+\pi(KT_{A1})h_A)\cdot l)(j\cdot(KT_{B1}+\pi(KT_{B1})P_B))$

密钥创建(B2)：实体 B 计算共享密钥：$K_{AB}=((r_B+\pi(KT_{B1})h_B)\cdot l)(j\cdot(KT_{A1}+\pi(KT_{A1})P_A))$

注 1：交互次数为 2。

注 2：该机制提供相互隐式密钥鉴别。

注 3：在某些情况下，该机制可能会受到源替换攻击，但可以通过增加延迟检测来避免这种攻击。

11.10 密钥协商机制 10

该机制适用于参与协商密钥的两个实体采用 SM2 椭圆曲线密码算法，且在线交互三次的场景，见 GB/T 32918.3—2016。该密钥协商机制采用椭圆曲线密码体制，通过三次交互采用相互隐式密钥鉴别，在实体 A 和实体 B 间建立一个共享密钥。应满足以下要求：

a) 每个实体 X 在 S_1 中有一个私钥协商密钥 h_X 和一个公钥协商密钥 $p_X=\mathrm{F}(h_X,g)$；

b) 每个实体可得到另一个实体已鉴别的公钥协商密钥副本，可使用第 13 章的机制来实现。

密钥协商机制 10 如图 7 所示。

l 和 j 值用于余子式乘法(见第 7 章)。还需采用一个函数将一椭圆点 P 转换为整数，例如，函数 $\pi(P)=(\mathrm{X}(P)\bmod(2^{\lceil\rho/2\rceil})+2^{\lceil\rho/2\rceil}$，其中 $\rho=\lceil\log_2 n\rceil$，$\mathrm{X}(P)$ 为点 P 的 x 坐标。

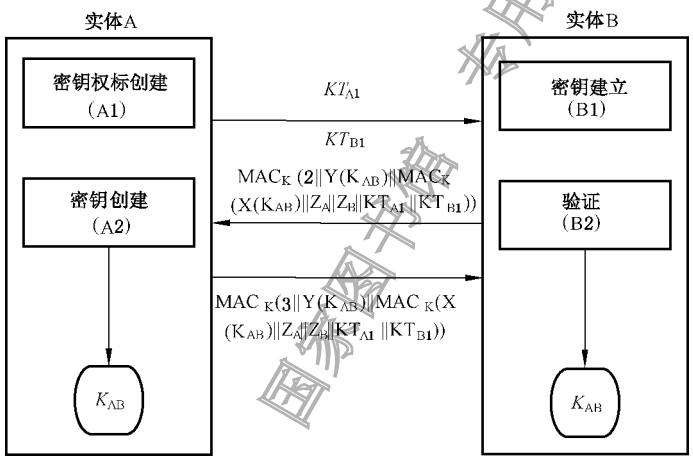

图 7 密钥协商机制 10

密钥权标创建(A1)：实体 A 随机及秘密地产生 r_A(r_A 在 S_1 中)，计算 $\mathrm{F}(r_A,g)$，构建密钥权标 $KT_{A1}=\mathrm{F}(r_A,g)$，然后发给实体 B。

密钥创建(B1)：实体 B 随机及秘密地产生 r_B(r_B 在 S_1 中)，计算 $\mathrm{F}(r_B,g)$，构建密钥权标 $KT_{B1}=\mathrm{F}(r_B,g)$，计算共享秘密密钥：

$K_{AB}=((r_B+\pi(KT_{B1})h_B)\cdot l)(j\cdot(KT_{A1}+\pi(KT_{A1})P_A))$

实体 B 计算密钥 $K=\mathrm{kdf}(K_{AB}\|Z_A\|Z_B)$。并进一步构建 $\mathrm{MAC}_K(2\|\mathrm{Y}(K_{AB})\|\mathrm{MAC}_K(\mathrm{X}(K_{AB})\|Z_A\|Z_B\|KT_{A1}\|KT_{B1}))$，然后将 KT_{B1} 及 $\mathrm{MAC}_K(2\|\mathrm{Y}(K_{AB})\|\mathrm{MAC}_K(\mathrm{X}(K_{AB})\|Z_A\|Z_B\|KT_{A1}\|KT_{B1}))$ 发给实体 A。

密钥创建(A2)：实体 A 计算共享秘密密钥：

$K_{AB}=((r_A+\pi(KT_{A1})h_A)\cdot l)(j\cdot(KT_{B1}+\pi(KT_{B1})P_B))$

计算密钥 $K=\mathrm{kdf}(K_{AB}\|Z_A\|Z_B)$ 及 $\mathrm{MAC}_K(2\|\mathrm{Y}(K_{AB})\|\mathrm{MAC}_K(\mathrm{X}(K_{AB})\|Z_A\|Z_B\|KT_{A1}\|KT_{B1}))$ 并验证实体 B 发送的内容。然后计算：

$\mathrm{MAC}_K(3\|\mathrm{Y}(K_{AB})\|\mathrm{MAC}_K(\mathrm{X}(K_{AB})\|Z_A\|Z_B\|KT_{A1}\|KT_{B1}))$，然后将其发送给实体 B。

验证(B2):实体 B 计算 $MAC_K(3\|Y(K_{AB})\|MAC_K(X(K_{AB})\|Z_A\|Z_B\|KT_{A1}\|KT_{B1}))$。

注 1:交互次数为 3。

注 2:该机制提供相互显式密钥鉴别。

注 3:附录 E 中的 E.3 给出该机制的一个例子。

11.11 密钥协商机制 11

该机制适用于参与协商密钥的两个实体中仅有一方知道对方身份,且在线交互四次的场景,通过四次交互在实体 A 和实体 B 间建立一个共享密钥,应满足以下要求:

a) 实体 B 有一个采用(E_B,D_B)变换的非对称加密系统;

b) 实体 A 可得到公开验证变换的一个已鉴别副本,用于验证 $Cert_B$;

c) 双方实体使用同一密钥派生函数 kdf。

密钥协商机制 11 如图 8 所示。

实体确认(A1):实体 A 选择一个随机整数 r_A,发送一个消息 $M_1=(r_A\|Text1)$给实体 B。

实体确认(B1):实体 B 选择一个随机整数 r_B,发送一个消息 $M_2=(r_B\|Cert_B\|Text2)$给实体 A。

密钥权标与密钥创建(A2):实体 A 验证 $Cert_B$以得到实体 B 公钥的可信副本,并产生一个随机整数 r'_A,计算共享密钥 $K_{AB}=kdf(r_A,r_B,r'_A)$。然后发送密钥权标 $KT_{A2}=E_B(r'_A)$和 $MAC_{K_{AB}}(M_1\|KT_{A2})$给实体 B。

密钥创建(B2):实体 B 解密 KT_{A2},计算共享密钥 $K_{AB}=kdf(r_A,r_B,r'_A)$。计算 $MAC_{K_{AB}}(M_1\|KT_{A2})$,并与接收到的 MAC 值进行比较。实体 B 把 $MAC_{K_{AB}}(M_2)$发给实体 A。

密钥验证(A3):实体 A 计算 $MAC_{K_{AB}}(M_2)$,并与接收到的 MAC 值进行比较。

注 1:交互次数为 4。

注 2:该机制提供实体 B 到实体 A 的隐式密钥鉴别。

注 3:该机制源于传输层安全协议(TLS),在 TLS 中密钥协商过程称为 TLS 握手阶段。在 TLS 中,每个实体有一个“密码套件”,即实体支持的一系列算法。作为“密码套件协商”过程的一部分,Text1 和 Text2 用来交换这些密码套件。

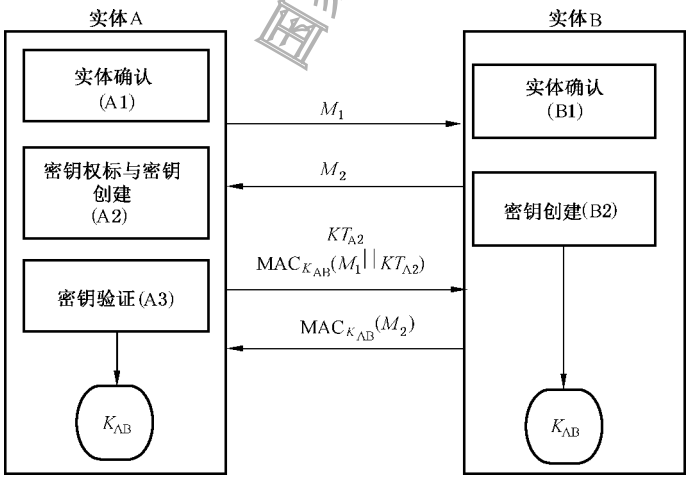

图 8 密钥协商机制 11

11.12 密钥协商机制 12

该机制用于以离线方式在三个实体之间协商共享密钥,非交互式地采用相互隐式密钥鉴别在实体 A、实体 B 和实体 C 间建立一个共享密钥,应满足以下要求:

a) 每个实体 X 在 S_1 中有一个密钥协商私钥 h_X 和一个密钥协商公钥 $p_X=\mathrm{F}(h_X,g)$；

b) 每个实体可得到其他实体已鉴别的密钥协商公钥副本，可使用第 13 章的机制来实现。

密码协商机制 12 如图 9 所示。

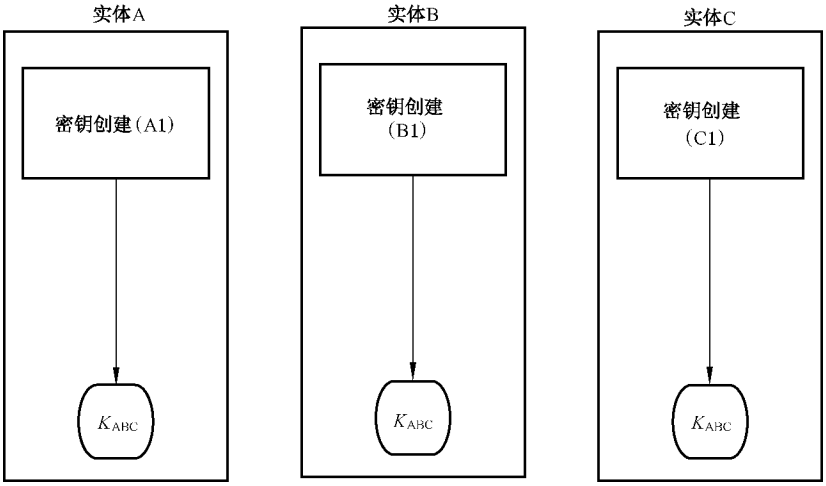

图 9 密钥协商机制 12

密钥创建(A1)：实体 A 使用自己的私钥协商密钥 h_A、实体 B 的公钥协商密钥 p_B、实体 C 的公钥协商密钥 p_C 计算共享秘密密钥 $K_{ABC}=\mathrm{FP}(h_A,p_B,p_C)$。

密钥创建(B1)：实体 B 用自己的私钥协商密钥 h_B、实体 A 的公钥协商密钥 p_A、实体 C 的公钥协商密钥 p_C 计算共享秘密密钥 $K_{ABC}=\mathrm{FP}(h_B,p_C,p_A)$。

密钥创建(C1)：实体 C 用自己的私钥协商密钥 h_C、实体 A 的公钥协商密钥 p_A、实体 B 的公钥协商密钥 p_B 计算共享秘密密钥 $K_{ABC}=\mathrm{FP}(h_C,p_A,p_B)$。

根据第 10 章中规定的函数 F 和 FP 条件，密钥 K_{ABC} 的三个计算值相等。

注 1：交互次数为 0。

注 2：该机制提供相互隐式密钥鉴别。然而，这种零交互协议总是产生相同的密钥，一种消除该问题带来隐患的办法是确保该密钥只被使用一次。此外，每次使用该密钥时并用一个唯一初始向量也可以解决该问题。

注 3：该机制不提供密钥确认。

注 4：这是一个密钥协商机制，它建立的密钥分别是实体 A、实体 B、实体 C 私钥协商密钥的单向函数。

12 密钥传递

12.1 密钥传递机制 1

该机制适用于参与密钥传递的两个实体中仅有一方知道对方身份，且只需在线交互一次的场景，通过实体 B 到实体 A 的隐式密钥鉴别，经一次交互实体 A 将一个秘密密钥传给实体 B，应满足以下要求：

a) 实体 B 有一个非对称加密系统(E_B，D_B)；

b) 实体 A 能访问实体 B 已鉴别的公开加密变换 E_B 副本，可使用第 13 章的机制来实现；

c) 可选的 TVP 可为一时间戳或一序列号，如果使用时间戳，那么实体 A 和实体 B 应保持时钟同步；如果使用序列号，那么实体 A 和实体 B 应维护双边计数器。

密钥传递机制 1 如图 10 所示。

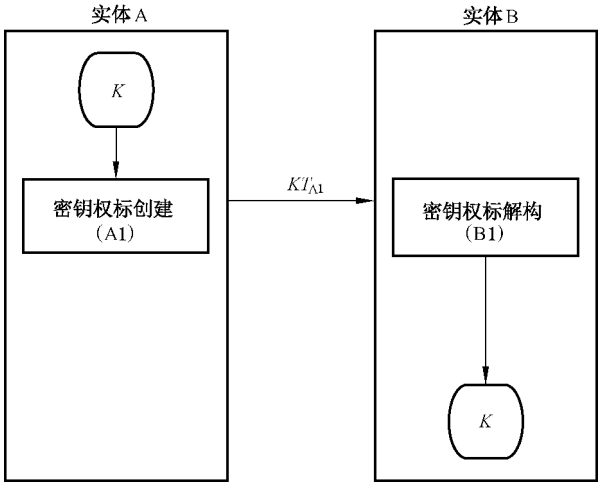

图 10　密钥传递机制 1

密钥权标创建(A1):假设 K 是实体 A 希望安全传递给实体 B 的一个秘密密钥。实体 A 创建一个密钥数据块,该数据块包含其可区分标识符(可选)、密钥 K、一个可选的 TVP 和一个可选的数据域 Text1。实体 A 使用接收者的公开加密变换 E_B 对该密钥数据块加密,然后将密钥权标 $KT_{A1}=E_B(A\|K\|TVP\|Text1)\|Text2$ 发送给实体 B。

密钥权标解构(B1):实体 B 使用自己的私有解密变换 D_B 将接收到的密钥权标加密部分进行解密,恢复密钥 K,检查可选的 TVP,将恢复的密钥 K 与声称是发送方的实体 A 相关联。

注 1:交互次数为 1。

注 2:由于只有实体 B 才能恢复密钥 K,该机制提供从实体 B 到实体 A 的隐式密钥鉴别。

注 3:该机制不提供密钥确认。

注 4:实体 A 可选择密钥。

注 5:由于实体 B 从未经鉴别的实体 A 接收到密钥 K,实体 B 对密钥 K 的安全使用限于不需要对实体 A 做鉴别的功能。例如,可执行解密和产生消息鉴别码,但不执行加密和验证消息鉴别码。

12.2　密钥传递机制 2

该密钥传递机制是 GB/T 15843.3—2016 一次交互实体鉴别机制的扩展,适用于参与密钥传递的两个实体只需在线交互一次的场景,它通过实体 A 到实体 B 的显式密钥鉴别,实体 B 到实体 A 的隐式密钥鉴别,从实体 A 传递一个经加密和签名的秘密密钥给实体 B,应满足以下要求:

a)　实体 A 有一个非对称签名系统(S_A,V_A);

b)　实体 B 有一个非对称加密系统(E_B,D_B);

c)　实体 A 能够访问实体 B 已鉴别的公开加密变换 E_B 副本,可使用第 13 章的机制来实现;

d)　实体 B 能够访问实体 A 已鉴别的公开验证变换 V_A 副本,可使用第 13 章的机制来实现;

e)　可选的 TVP 是一时间戳或一序列号,如果使用时间戳,那么实体 A 和实体 B 应保持时钟同步,或使用一可信第三方时间戳机构;如使用序列号,那么实体 A 和实体 B 应维护双边计数器。

密钥传递机制 2 如图 11 所示。

密钥加密(A1.1):假设 K 是实体 A 希望安全传递给实体 B 的一个密钥,实体 A 形成一个密钥数据块,该数据块包含发送方可区分识别符、密钥 K 和一个可选数据域 Text1。实体 A 用实体 B 的公开加密变换 E_B 对密钥数据块进行加密,形成加密数据块 $BE=E_B(A\|K\|Text1)$。

密钥权标创建(A1.2):实体 A 创建权标数据块,包括接收者的可区分识别符、可选的 TVP(时间戳

或序列号),经加密的数据块 BE 和可选的数据域 Text2。实体 A 用其私有签名变换 S_A 签署数据,附加上可选的 Text3,然后形成密钥权标 $KT_{A1}=S_A(B\|TVP\|BE\|Text2)\|Text3$,发送给实体 B。

密钥权标验证(B1.1):实体 B 用发送方的公开验证变换 V_A 来验证接收的密钥权标中的数字签名,然后实体 B 检查 KT_{A1} 中的识别符和可选的 TVP。

密钥解密(B1.2):实体 B 用其私有解密变换 D_B 解密块 BE,然后将包含在块 BE 中的实体 A 的识别符和签名实体的身份进行比较,如果所有的检查成功,那么实体 B 将接受密钥 K。

注 1:交互次数为 1。

注 2:该机制提供了实体 A 到实体 B 的实体鉴别,该机制还提供从实体 B 到实体 A 的隐式密钥鉴别。

注 3:该机制提供了实体 A 到实体 B 的密钥确认。实体 B 确信它与实体 A 分享正确的密钥,但实体 A 在它收到实体 B 用密钥 K 加密答复之后,才能确信实体 B 已经收到该密钥。

注 4:可选的 TVP 提供实体 A 到实体 B 的实体鉴别,并防止密钥权标重放。为了防止密钥数据块 BE 的重放,需在 Text1 中增加一个 TVP。

注 5:实体 A 可选择密钥 K_A,因为它是发起实体。类似地,实体 B 也能选择密钥 K_B。联合密钥控制可以通过将实体 A 和实体 B 的密钥 K_A 和 K_B 结合起来(可通过这个机制的两个例子传递),形成一个共享秘密密钥 K_{AB}。联合密钥控制需额外的一次交互。这种结合应是单向的,否则实体 A 就能够选择共享秘密密钥。这种机制可被分类为密钥协商机制。

注 6:实体 A 的可区分标识符包括在被加密的块 BE 中,以防止实体 A 的加密密钥块被另一实体盗用。抵抗该攻击的方法是要求实体 B 比较实体 A 的标识符和实体 A 对权标的签名。

注 7:与 GB/T 15843.3—2016 一致,采用公钥算法 KT_{A1} 的实体鉴别和单次鉴别机制的权标发送兼容。通过使用 $BE\|Text2$ 取代 GB/T 15843.3—2016 机制中的 Text 1,该权标能够帮助密钥 K 的传送。

注 8:数据域 Text 3 可用来传递实体 A 的公钥证书,在这种情况下,12.2 的 d)可放宽为实体 B 拥有 CA 一个已鉴别的公开验证密钥副本。

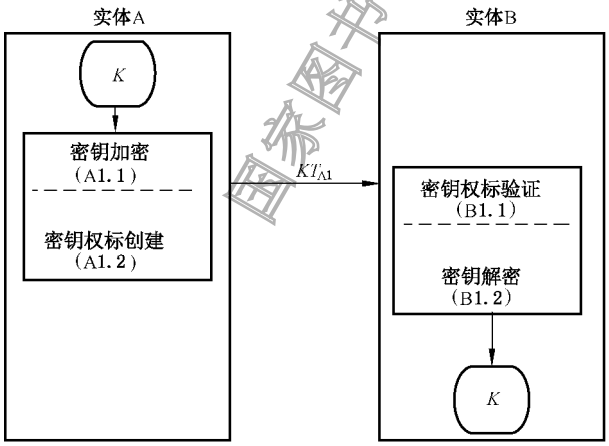

图 11 密钥传递机制 2

12.3 密钥传递机制 3

该密钥传递机制中,实体 A 采用单边密钥确认通过一次交互将经签名及加密的秘密密钥传给实体 B。该机制适用于参与密钥传递的两个实体只需在线交互一次的场景。应满足以下要求:

a) 实体 A 有一个非对称的签名系统(S_A,V_A);

b) 实体 B 有一个非对称的加密系统(E_B,D_B);

c) 实体 A 得到实体 B 已鉴别的公开加密变换 E_B 副本,可使用第 13 章的机制来实现;

d) 实体 B 得到实体 A 已鉴别的公开鉴别变换 V_A 副本,可使用第 13 章的机制来实现;

e) 可选的 TVP 是一时间戳或一序列号,如果使用时间戳,那么实体 A 和实体 B 应保持同步时

钟；如果使用序列号，那么实体 A 和实体 B 应维护双边计数器。

密钥传递机制 3 如图 12 所示。

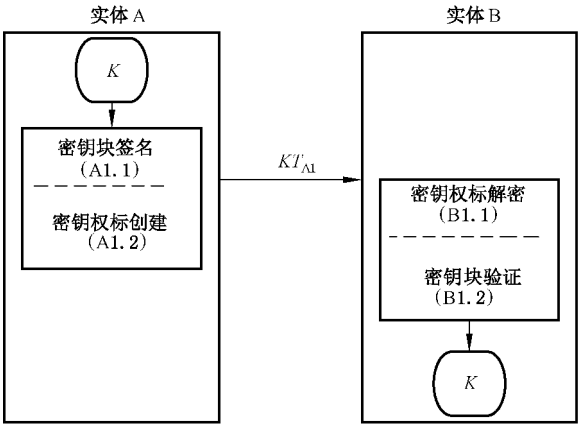

图 12　密钥传递机制 3

密钥块签名(A1.1)：假设实体 A 希望将一个秘密密钥 K 安全传递给实体 B，实体 A 形成一个密钥数据块，该数据块包含接收方的可区分识别符、密钥 K、可选的 TVP(序列号或时间戳)和可选数据，然后用其私有签名变换 S_A 对密钥块进行签名，产生签名后的数据块 $BS=S_A(B\|K\|TVP\|Text1)$。

密钥权标创建(A1.2)：实体 A 构建权标数据块，它包括已签名的块 BS 和可选项 Text2。实体 A 用接收者的公开加密变换 E_B 加密权标数据块，附上可选的 Text3，然后形成密钥权标 $KT_{A1}=E_B(BS\|Text2)\|Text3$，并送给实体 B。

密钥权标解密(B1.1)：实体 B 使用其私有解密变换 D_B 解出接收到的密钥权标加密部分。

密钥块验证(B1.2)：实体 B 使用发送方的公开验证变换 V_A 验证 BS 的完整性和来源，确认它确实是权标的接收者(通过检查 BS 中的标识符)，并可选择地确认 TVP 在可接受的范围内(验证权标的时间性)，如所有的验证都成功，实体 B 将接受该密钥 K。

注 1：交互次数为 1。

注 2：该机制提供实体 A 到实体 B 的实体鉴别和从实体 B 到实体 A 的隐式密钥鉴别。

注 3：该机制提供实体 A 到实体 B 的密钥确认，实体 B 确信它与实体 A 分享正确的密钥，但实体 A 在它收到实体 B 用密钥 K 加密答复之后，才能确信实体 B 已经收到该密钥。

注 4：实体 A 可选择密钥。

注 5：实体 B 的可区分标识符包括在被签名的密钥块 BS 中来明确表明密钥的接收者，从而防止由实体 B 签名的块 BS 被滥用。

注 6：数据域 Text3 能用来传递实体 A 的公钥证书，在这种情况下，12.3 的 d)可放宽为实体 B 拥有 CA 已鉴别的公开验证密钥副本。

注 7：如果将该密钥传递机制的两种实施方法结合起来(从实体 A 到实体 B，从实体 B 到实体 A)，那么就可以提供相互实体鉴别和联合密钥控制(取决于可选项 TVP 的使用)。

12.4　密钥传递机制 4

该机制适用于参与密钥传递的两个实体彼此鉴别对方身份，且在线交互两次的场景，基于 GB/T 15843.3—2016 中的两次交互鉴别机制，将密钥从实体 B 传递到实体 A。应满足以下要求：

a)　实体 A 有一个非对称加密系统(E_A，D_A)；

b)　实体 B 有一个非对称签名系统(S_B，V_B)；

c)　实体 A 可得到实体 B 已鉴别的公钥验证变换 V_B 副本，可使用第 13 章的机制来实现；

d)　实体 B 可得到实体 A 已鉴别的公钥加密变换 E_A 副本，可使用第 13 章的机制来实现。

密钥传递机制 4 如图 13 所示。

密钥权标创建(**A1**):实体 A 产生一个随机数 r_A,将 r_A 和可选的数据域 Text1 组成密钥权标 KT_{A1},$KT_{A1}=r_A\|$ Text 1 并把它送给实体 B。

密钥块加密(**B1.1**):假设实体 B 希望将一个秘密密钥 K 安全传递到实体 A,实体 B 构建一个由发送者的区分标识符、密钥 K 和一可选的数据域 Text2 组成的密钥数据块。实体 B 用实体 A 的公钥变换 E_A 将密钥数据块加密,构成加密块 $BE=E_A(B\|K\|\text{Text2})$。

密钥权标创建(**B1.2**):实体 B 可选地生成随机数 r_B,并形成权标数据块,该权标数据块由接收者的区分标识符、步骤 A1 中接收的随机数 r_A、新的随机数 r_B(可选)、加密块 BE 以及可选的数据域 Text3 组成,然后实体 B 用其私钥签名变换 S_B 形成签名,附加可选的 Text4,产生密钥权标 $KT_{B1}=S_B(A\|r_A\|r_B\|BE\|\text{Text3})$,并送给实体 A。

密钥权标验证(**A2.1**):实体 A 使用发送者的公钥验证变换 V_B 验证接收到的公钥权标 KT_{B1} 中的数字签名,然后实体 A 检查 KT_{B1} 中的区分标识符,并检查接收到的值 r_A 与步骤 A1 发送的随机数是否一致。

密钥块解密(**A2.2**):实体 A 用其私钥解密变换 D_A 解密加密块 BE,然后验证 BE 中的发送者区分标识符。如所有检查都成功,实体 A 接受该密钥 K。

注 1:交互次数为 2。

注 2:该机制提供从实体 A 到实体 B 的隐式密钥鉴别。

注 3:该机制提供实体 B 到实体 A 的密钥确认,实体 A 确信它与实体 B 分享正确的密钥,但实体 B 在它收到实体 A 用密钥 K 加密答复之后,才能确信实体 A 已经收到该密钥。

注 4:实体 B 可选择密钥。

注 5:权标 KT_{A1} 和 KT_{B1} 与 GB/T 15843.3—2016 中的 5.1.2 中所描述的两次交互鉴别机制里发送的权标一致(注意此时实体 A 和实体 B 的角色互换),通过使用 $BE\|$ Text3 取代 GB/T 15843.3—2016 机制中的 Text2,该权标能够帮助密钥 K 的传送。

注 6:如果该密钥传递机制在两个实体间并行执行两次,得到的相互密钥传递机制与 GB/T 15843.3—2016 中的 5.2.3所描述的机制一致。

注 7:为了与 GB/T 15843.3—2016 保持一致性,将数据域 r_B 包含在内。但由于 KT_{B1} 中存在 BE,r_B 在此机制中将是可选的。

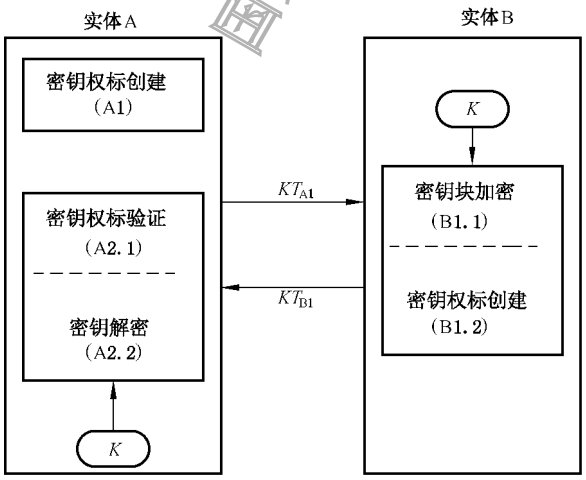

图 13 密钥传递机制 4

12.5 密钥传递机制 5

该机制适用于参与密钥传递的两个实体采用 GB/T 15843.3—2016 规定的三次交互鉴别机制彼此鉴别对方身份,且在线交互三次的场景,通过相互实体鉴别的和密钥确认传递两个共享密钥,一个从实

体 A 传给实体 B,另一个从实体 B 传给实体 A。应满足以下要求:

a) 每个实体 X 有一个非对称的签名系统(S_X,V_X);

b) 每个实体 X 有一个非对称的加密系统(E_X,D_X);

c) 每个实体可得到其他实体已鉴别的公开验证变换副本,可使用第 13 章的机制来实现;

d) 每个实体可得到其他实体已鉴别的公开加密变换副本,可使用第 13 章的机制来实现。

密钥传递机制 5 如图 14 所示。

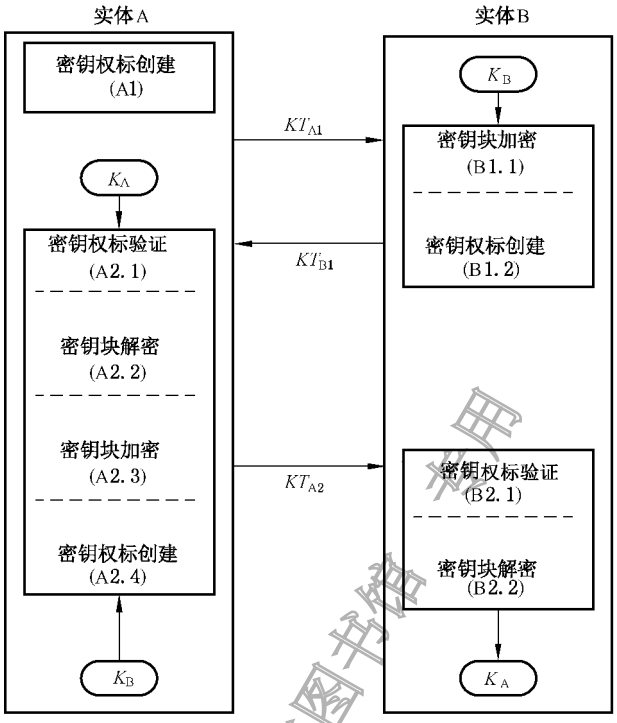

图 14 密钥传递机制 5

密钥权标创建(A1):实体 A 随机生成 r_A,构建密钥权标 $KT_{A1}=r_A\|\text{Text1}$。并将其发送给实体 B。

密钥块加密(B1.1):假设实体 B 希望将一个秘密密钥 K 安全传递到实体 A,实体 B 构建一个由它自己的区分标识符、密钥 K_B 以及可选的 Text2 组成的块,然后用接收者的公钥加密变换 E_A 加密这个块:

$BE_1=E_A(B\|K_B\|\text{Text2})$

密钥权标创建(B1.2):实体 B 随机生成 r_B,构建一个包含 r_B、r_A、接收者标识、加密的密钥块 BE_1 以及可选的 Text3 数据块。实体 B 用其私钥签名变换 S_B 签名这个数据块,附加可选的 Text4,形成密钥权标块 KT_{B1},$KT_{B1}=S_B(r_B\|r_A\|A\|BE_1\|\text{Text3})\|\text{Text4}$,并发给实体 A。

密钥权标验证(A2.1):实体 A 使用实体 B 的公钥验证变换 V_B 来验证实体 B 对密钥权标 KT_{B1} 的签名。检查它在 KT_{B1} 中的区分标识符,并检查接收到的 r_A 是否与步骤 A1 发送的随机数一致。

密钥块解密(A2.2):实体 A 使用其私钥解密变换 D_A 解密加密块 BE_1,并检查实体 B 的区分标识符,如果所有的检查结果都成功,实体 A 接受密钥 K_B。

密钥块加密(A2.3):实体 A 构建一个包含其区分标识符、其私钥 K_A 及可选的 Text5 的数据块,并使用接收者的公开加密变换 E_B 加密这个加密块:$BE_2=E_B(A\|K_A\|\text{Text5})$。

密钥权标创建(A2.4):实体 A 构建一个包含随机数 r_A、随机数 r_B、接收者的区分标识符、加密的密钥块 BE_2 和可选的 Text6 数据块。实体 A 用其私钥签名变换 S_A 来签名这个数据块,附上可选的 Text7,并发送这个密钥权标 $KT_{A2}=S_A(r_A\|r_B\|B\|BE_2\|\text{Text6}\|\text{Text7})$ 给实体 B。

密钥权标验证(B2.1):实体 B 使用实体 A 的公钥验证变换 V_A 验证实体 A 的签名,检查 KT_{A2} 中它

自己的区分标识符，并检查接收到的 r_B 是否与步骤 B1.2 发送的随机数一致，此外，实体 B 还检查接收到的 r_A 是否与 KT_{A1} 中的数值是一致。

密钥块解密(**B2.2**)：实体 B 用其私钥变换 D_B 解密加密数据块 BE_2，验证实体 A 的区分标识符，如果所有的检查成功，那么实体 B 接受密钥 K_A，如果只是要求单方密钥传输，则 BE_1 或 BE_2 可省略。

注 1：交互次数为 3。

注 2：该机制提供相互实体鉴别，即实体 B 到实体 A 的 K_A 隐式密钥鉴别和实体 A 到实体 B 的 K_B 隐式密钥鉴别。

注 3：该机制对 K_A 和 K_B 都进行从发送端到接收端的密钥确认，此外如果实体 A 在 K_B 中包含一个 MAC，那么就 K_B 而言，这种机制可以进行相互密钥确认。

注 4：由于实体 A 是发送实体，它可选择密钥 K_A。类似地，实体 B 也可选择密钥 K_B。联合密钥控制可通过将两个密钥 K_A 和 K_B 组合在一起构成一个共享秘密密钥 K_{AB} 来实现。组合函数应为单向的，否则实体 A 可选择共享秘密密钥。这种机制可分类为一种密钥协商机制。

注 5：KT_{A1}、KT_{B1} 和 KT_{A2} 与 GB/T 15843.3—2016 中的 5.2.2 描述的三次交互鉴别机制中发送的密钥权标兼容。通过使用 $BE_1 \| Text3$ 取代 GB/T 15843.3—2016 机制中的 Text2，第二个权标能够帮助密钥 K_B 的传送。通过使用 $BE_2 \| Text6$ 取代 GB/T 15843.3—2016 中的 Text4，第三个权标能够帮助密钥 K_A 的传送，第三个权标还能够帮助 Text6 里的 MAC 的传送。

注 6：如果数据域 Text1 和 Text4(或 Text7 和 Text4)包含实体 A 和实体 B 的公钥证书，那么 12.5 的 c)和 d)可放宽为两个实体都拥有 CA 经鉴别的公开验证密钥副本。

12.6 密钥传递机制 6

该机制适用于参与密钥传递的两个实体采用以零知识技术为基础，且在线交互三次的场景，经三次交互安全传递两个密钥，一个从实体 A 到实体 B，另一个从实体 B 到实体 A。此外，该机制还实现相互实体鉴别及对各密钥分别进行相互密钥确认。该机制应满足以下要求：

a) 每个实体 X 有一个非对称的加密系统(E_X，D_X)；

b) 每个实体可得到其他实体已鉴别的公开加密变换副本。可使用第 13 章的机制来实现。

密钥传递机制 6 如图 15 所示。

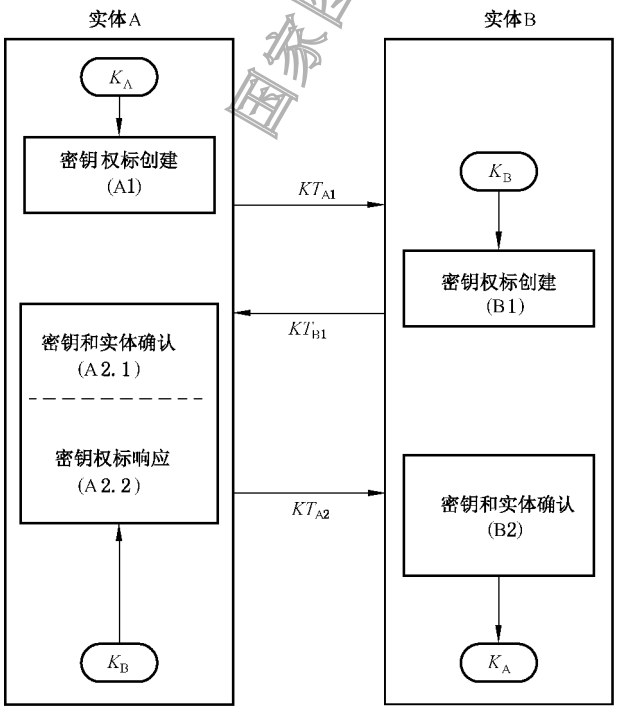

图 15 密钥传递机制 6

密钥权标创建(A1):实体 A 获得一个密钥 K_A,并希望把它安全地传送给实体 B。实体 A 选择一个随机数 r_A并构造一个包含其区分标识符、密钥 K_A、随机数 r_A以及可选的数据块 Text1 的密钥数据块,然后用实体 B 的公钥加密变换 E_B将密钥块加密,产生加密数据块 $BE_1=E_B(A\|K_A\|r_A\|Text1)$。

实体 A 构造一个密钥权标 KT_{A1},它由加密数据块 BE_1及可选数据域 Text2 组成,$KT_{A1}=BE_1\|Text2$,并发送给实体 B。

密钥权标创建(B1):实体 B 从接收到的密钥权标 KT_{A1}中提取加密密钥块 BE_1,并用其私钥解密变换 D_B来将它解密,然后检查解密后的密钥块 BE_1中包含的实体 A 标识符。

实体 B 获得一个密钥 K_B,并希望把它安全地传送给实体 A。实体 B 选择一个随机数 r_B,并构造一个包含其区分标识符、密钥 K_B、随机数 r_B、随机数 r_A(从解密块提取)及可选的数据域 Text3 的密钥数据块。实体 B 用实体 A 的公钥加密变换 E_A将密钥块加密,构造加密数据块 $BE_2=E_A(B\|K_B\|r_A\|r_B\|Text3)$。

实体 B 构造一个密钥权标 KT_{B1},它由加密数据块 BE_2及可选数据域 Text4 组成,$KT_{B1}=BE_2\|Text4$,并发送给实体 A。

密钥和实体确认(A2.1):实体 A 从接收到的密钥权标 KT_{B1}中提取加密密钥块 BE_2,并使用其私钥解密变换 D_A将它解密。实体 A 通过比较随机数 r_A与包含在加密块 BE_2中的随机数 r_A检查密钥权标的有效性。如果验证成功,那么实体 A 确认密钥 K_A已安全到达实体 B。

密钥权标响应(A2.2):实体 A 从解密的密钥块中提取随机数 r_B,并用随机数 r_B和可选的数据域 Text5 构建密钥权标 $KT_{A2}=r_B\|Text5$。然后实体送给实体 B。

密钥和实体确认(B2):实体 B 验证从 KT_{A2}提取的响应 r_B与 KT_{B1}中的随机数 r_B是否一致,如果验证成功,那么实体 B 就实现对实体 A 的鉴别,同时确认密钥 K_B已安全到达实体 A。

注 1:交互次数是 3。

注 2:该机制提供从实体 B 到实体 A 的 K_A隐式密钥鉴别,以及实体 A 到实体 B 的 K_B隐式密钥鉴别。

注 3:由于实体 A 是发送实体,实体 A 可选择密钥 K_A。类似地,实体 B 也可选择密钥 K_B。联合密钥控制可通过将两个密钥 K_A和 K_B组合在一起构成一个共享秘密密钥 K_{AB}来实现。组合函数是单向的,否则实体 B 可选择共享秘密密钥。这种机制可归类为一种密钥协商机制。

注 4:该机制使用非对称技术相互传递两个秘密密钥,K_A从实体 A 传到实体 B,K_B从实体 B 传到实体 A。该机制可用于如下密钥功能分离:实体 A 用密钥 K_A加密传输给实体 B 的消息以及验证来自实体 B 的认证码。实体 B 使用得到的密钥 K_A解密 A 传来的消息并生成给实体 A 的认证码。密钥 K_B的密钥功能也采用类似的方式分离。这样,非对称密钥传递机制就可拓展为密钥使用。

注 5:该机制以零知识技术为基础,通过使用该机制,双方实体除了自己能够计算得到的信息之外,无法获得任何其他信息。

13 公钥传递

13.1 公钥传递机制 1

如果实体 A 经一个受保护通道到实体 B,(例如提供数据源鉴别及数据完整性的通道),如速递、注册邮件等,那么实体 A 可将其公钥信息直接通过该受保护通道传递给实体 B。这是传递公钥最基本的形式。该机制应满足以下要求:

a) 实体 A 的公钥信息 PKI_A至少要包含实体 A 的可区分标识符和实体 A 的公钥。此外,它还可以包含序列号、有效期、时间戳和其他数据元素。

b) 由于公钥信息不包含任何秘密数据,所以通信信道不需提供保密性。

公钥传递机制 1 如图 16 所示。

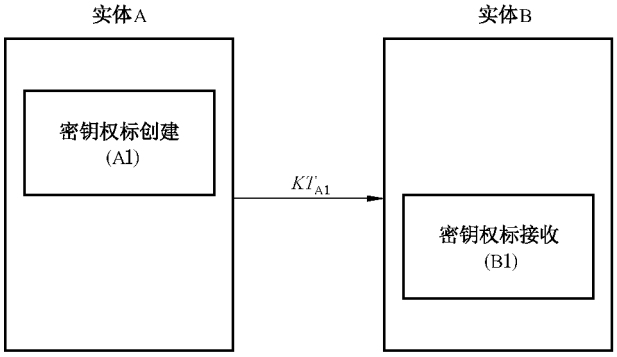

图 16　公钥传输机制 1

密钥权标创建(A1)：实体 A 生成密钥权标 KT_{A1}，它包含实体 A 的公钥信息和可选的数据字段 Text，并通过受保护的通道将 $KT_{A1}=PKI_A \| \text{Text}$ 传给实体 B。

密钥权标接收(B1)：实体 B 得到从受保护通道传递的实体 A 的密钥权标，得到实体 A 的公钥 PKI_A 并将实体 A 的公钥保存在有效的公钥列表中(该列表需防止被篡改)。

注 1：该机制可用于传递公开验证密钥(对于非对称签名系统)，或公开加密密钥(对于非对称加密系统)，或公钥协商密钥。

注 2：鉴别包括数据完整性和数据源鉴别(如 ISO 7498-2:1989 所定义)。

13.2　公钥传递机制 2

该传递机制通过不受保护的通道将实体 A 的公钥信息传递给实体 B。为了确认完整性以及接收到公钥信息的来源，需使用另外一条鉴别通道。该机制有效地用于：当公钥信息 PKI 以电子形式通过高带宽通道传输，而公钥信息的鉴别经低带宽通道执行，如电话、速递或者注册邮件等。作为一种附加要求，如 GB/T 32905 所定义，实体间需共享通用的杂凑函数。应满足以下要求：

a)　实体 A 的公钥信息 PKI_A 至少要包含实体 A 的可区分标识符和实体 A 的公钥。此外，它还可包含序列号、有效期、时间戳和其他数据元素。

b)　由于公钥信息不包含任何秘密数据，所以通信信道不需提供保密性。

公钥传递机制 2 如图 17 所示。

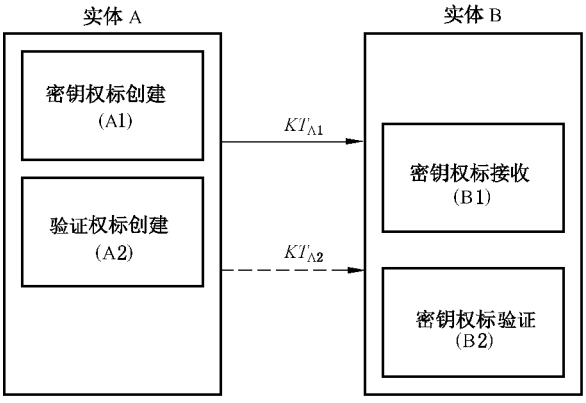

图 17　公钥传递机制 2

密钥权标创建(A1)：实体 A 生成密钥权标 KT_{A1}，它包含实体 A 的公钥信息和可选的数据字段 Text，并通过不受保护的通道将 $KT_{A1}=PKI_A \| \text{Text}$ 传给实体 B。

密钥权标接收(B1)：实体 B 得到从受保护信道传递的实体 A 的密钥权标，得到实体 A 的公钥 PKI_A 并且将实体 A 的公钥保存并防止被篡改，以便接下来的鉴别和使用。

验证权标创建(**A2**):实体 A 用其公钥计算检验杂凑值 hash(PKI_A),并将这个检验值和可选的实体 A 和实体 B 的可区分标识符通过第二条单独的已鉴别信道(如速递或注册邮件)传输给实体 B,其中 KT_{A2}=A‖B‖hash(PKI_A)‖Text2。

密钥权标验证(**B2**):一旦收到验证权标 KT_{A2},实体 B 选择性的检验实体 A 和实体 B 的可区分标识符,根据实体 A 收到的 KT_{A1} 里的公钥信息计算出检验值,并于 KT_{A2} 中的检验值做比较,如果校验成功,实体 B 将实体 A 的公钥保存在有效的公钥列表中(该列表需防止被篡改)。

注 1:该机制可用于传递公开验证密钥(对于非对称签名系统),或公开加密密钥(对于非对称加密系统),或公钥协商密钥。

注 2:鉴别包括数据完整性和数据源鉴别。

注 3:如果传递的公钥用于非对称数字签名系统,而不是消息恢复,那么实体 A 可用相应的私有签名密钥对 KT_{A1} 签名。这样,步骤 B1 中使用接收到的公开验证密钥进行的实体 A 的签名验证可确保实体 A 知道对应的私有签名密钥,因此可以推测,实体 A 是在权标创建时知道相应的私有签名密钥的唯一实体。如果 PKI_A 中使用时间戳,那么验证可确保实体 A 当前知道相应的私有签名密钥。

注 4:来自实体 A 的人工签名信可用于验证权标。

13.3 公钥传递机制 3

该机制通过可信第三方的身份鉴别方式从实体 A 传递一个公钥到实体 B。使用公钥证书形式交换公钥可确保证实体的公钥鉴别。实体 A 的公钥证书包括公钥信息以及可信第三方(证书认证机构)的数字签名。引入证书认证机构将减少鉴别用户公钥的分发问题,转为证书认证机构(CA)公钥分发问题见 GB/T 17901.1—2020。

该机制基于这样假设:一个带有实体 A 公钥信息 PKI_A 的有效公钥证书 $Cert_A$ 由某 CA 机构发布,且实体 B 可获取发布公钥证书的 CA 机构已鉴别的公开验证变换 V_{CA} 副本。

公钥传递机制 3 如图 18 所示。

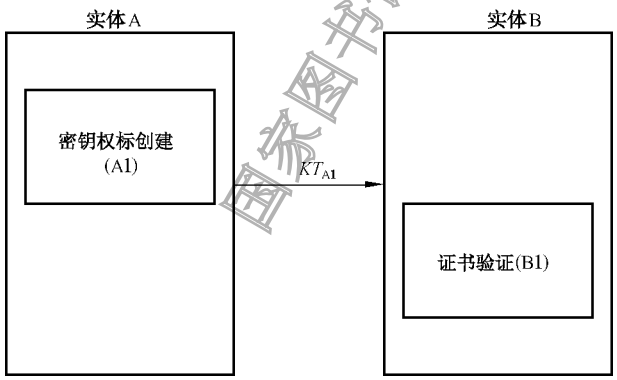

图 18 公钥传递机制 3

密钥权标创建(**A1**):实体 A 生成密钥权标 KT_{A1},它包含实体 A 的公钥证书,并将 KT_{A1}=PKI_A‖Text 传递给实体 B。

证书验证(**B1**):实体 B 一旦接收到公钥证书,就通过 CA 的公开验证变换 V_{CA} 验证公钥信息,并校验实体 A 公钥的有效性。

如果实体 B 希望确认实体 A 的公钥证书最近是否被取消,那么实体 B 应通过某种可信途径咨询可信第三方(如 CA 机构)。

注 1:交互次数为 1,但实体 B 可以要求实体 A 传输公钥证书。这个额外的交互是可选的,这里不显示。实体 A 的公钥证书也可通过目录分发,在这种情况下,需在目录和实体 B 间执行公钥传输机制。

注 2:该机制不提供实体鉴别。

注 3:收到的公钥证书确认了公钥经 CA 鉴别。

注 4:实体 B 可通过鉴别方式得到 CA 机构的公开验证密钥 V_{CA}。可使用第 13 章的机制来实现。

附 录 A
（规范性附录）
对象标识符

密钥管理机制中使用的对象标识符按本附录进行规范。

```
KeyManagement-AsymmetricTechniques {
iso(1) standard(0) keyManagement(11770)
asymmetricTechniques(3) asn1-module(0) object-identifiers(0) }
DEFINITIONS EXPLICIT TAGS ::= BEGIN
-- EXPORTS All; --
-- IMPORTS None; --
OID ::= OBJECT IDENTIFIER -- Alias
-- Synonyms --
id-km-at OID ::= {
   iso(1) standard(0) keyManagement(11770) asymmetricTechniques(3) }
-- Assignments --
id-km-at-kAM-1 OID ::= { id-km-at keyAgreementMechanism(1) }
id-km-at-kAM-2 OID ::= { id-km-at keyAgreementMechanism(2) }
id-km-at-kAM-3 OID ::= { id-km-at keyAgreementMechanism(3) }
id-km-at-kAM-4 OID ::= { id-km-at keyAgreementMechanism(4) }
id-km-at-kAM-5 OID ::= { id-km-at keyAgreementMechanism(5) }
id-km-at-kAM-6 OID ::= { id-km-at keyAgreementMechanism(6) }
id-km-at-kAM-7 OID ::= { id-km-at keyAgreementMechanism(7) }
id-km-at-kAM-8 OID ::= { id-km-at keyAgreementMechanism(8) }
id-km-at-kAM-9 OID ::= { id-km-at keyAgreementMechanism(9) }
id-km-at-kAM-10 OID ::= { id-km-at keyAgreementMechanism(10) }
id-km-at-kAM-11 OID ::= { id-km-at keyAgreementMechanism(11) }
id-km-at-kAM-12 OID ::= { id-km-at keyAgreementMechanism(21) }
id-km-at-kTM-1 OID ::= { id-km-at keyTransportMechanism(12) }
id-km-at-kTM-2 OID ::= { id-km-at keyTransportMechanism(13) }
id-km-at-kTM-3 OID ::= { id-km-at keyTransportMechanism(14) }
id-km-at-kTM-4 OID ::= { id-km-at keyTransportMechanism(15) }
id-km-at-kTM-5 OID ::= { id-km-at keyTransportMechanism(16) }
id-km-at-kTM-6 OID ::= { id-km-at keyTransportMechanism(17) }
id-km-at-pKT-1 OID ::= { id-km-at publicKeyTransportMechanism(18) }
id-km-at-pKT-2 OID ::= { id-km-at publicKeyTransportMechanism(19) }
id-km-at-pKT-3 OID ::= { id-km-at publicKeyTransportMechanism(20) }
-- Key Agreement Mechanism 1 --
keyConstruction-1a OID ::= {? id-km-at-kAM-1 keyConstructionFunction(1) }
keyConstruction-1b OID ::= {? id-km-at-kAM-1 keyConstructionFunction(2) }
-- Key Agreement Mechanism 2 --
```

```
keyTokenConstruction-2 OID ::= {
  id-km-at-kAM-2 keyTokenConstructionFunction(1) }
keyConstruction-2a OID ::= {
  id-km-at-kAM-2 keyConstructionFunction(2) }
keyConstruction-2b OID ::= {
  id-km-at-kAM-2 keyConstructionFunction(3) }
-- Key Agreement Mechanism 3 --
keyConstruction-3a OID ::= {
  id-km-at-kAM-3 keyConstructionFunction(1) }
keyTokenSignature-3 OID ::= {
  id-km-at-kAM-3 keyTokenSignatureFunction(2) }
keyConstruction-3b OID ::= {
  id-km-at-kAM-3 keyConstructionFunction(3) }
signatureVerification-3 OID ::= {
  id-km-at-kAM-3 signatureVerificationFunction(4) }
-- Key Agreement Mechanism 4 --
keyTokenConstruction-4a OID ::= {
  id-km-at-kAM-4 keyTokenConstructionFunction(1) }
keyTokenConstruction-4b OID ::= {
  id-km-at-kAM-4 keyTokenConstructionFunction(2) }
keyConstruction-4a OID ::= {
  id-km-at-kAM-4 keyConstructionFunction(3) }
keyConstruction-4b OID ::= {
  id-km-at-kAM-4 keyConstructionFunction(4) }
-- Key Agreement Mechanism 5 --
keyTokenConstruction-5a OID ::= {
  id-km-at-kAM-5 keyTokenConstructionFunction(1) }
keyTokenConstruction-5b OID ::= {
  id-km-at-kAM-5 keyTokenConstructionFunction(2) }
keyConstruction-5a OID ::= {
  id-km-at-kAM-5 keyConstructionFunction(3) }
keyConstruction-5b OID ::= {
  id-km-at-kAM-5 keyConstructionFunction(4) }
-- Key Agreement Mechanism 6 --
keyTokenConstruction-6 OID ::= {
  id-km-at-kAM-6 keyTokenConstructionFunction(1) }
keyTokenProcessing-6b OID ::= {
  id-km-at-kAM-6 keyTokenProcessingFunction(2) }
keyConstruction-6 OID ::= {
  id-km-at-kAM-6 keyConstructionFunction(3) }
keyTokenProcessing-6a OID ::= {
  id-km-at-kAM-6 keyTokenProcessingFunction(4) }
-- Key Agreement Mechanism 7 --
```

```
keyTokenConstruction-7 OID ::= {
  id-km-at-kAM-7 keyTokenConstructionFunction(1) }
keyTokenProcessingAndKeyConstruction-7 OID ::= {
  id-km-at-kAM-7 keyTokenProcessingAndKeyConstructionFunction(2) }
keyTokenProcessing-7a OID ::= {
  id-km-at-kAM-7 keyTokenProcessingFunction(4) }
keyTokenProcessing-7b OID ::= {
  id-km-at-kAM-7 keyTokenProcessingFunction(5) }
-- Key Agreement Mechanism 8 --
keyTokenConstruction-8 OID ::= {
  id-km-at-kAM-8 keyTokenConstructionFunction(1) }
keyConstruction-8a OID ::= {
  id-km-at-kAM-8 keyConstructionFunction(2) }
keyConstruction-8b OID ::= {
  id-km-at-kAM-8 keyConstructionFunction(3) }
-- Key Agreement Mechanism 9 --
keyTokenConstruction-9a OID ::= {
  id-km-at-kAM-9 keyTokenConstructionFunction(1) }
keyTokenConstruction-9b OID ::= {
  id-km-at-kAM-9 keyTokenConstructionFunction(2) }
keyConstruction-9a OID ::= {
  id-km-at-kAM-9 keyConstructionFunction(3) }
keyConstruction-9b OID ::= {
  id-km-at-kAM-9 keyConstructionFunction(4) }
-- Key Agreement Mechanism 10 --
keyTokenConstruction-10a OID ::= {
  id-km-at-kAM-10 keyTokenConstructionFunction(1) }
keyConstruction-10b OID ::= {
  id-km-at-kAM-10 keyConstructionFunction(2) }
keyConstruction-10a OID ::= {
  id-km-at-kAM-10 keyConstructionFunction(3) }
verification-10b OID ::= {
  id-km-at-kAM-10 verificationFunction(4) }
-- Key Agreement Mechanism 11 --
entityConfirmation-11a OID ::= {
  id-km-at-kAM-11 entityConfirmationFunction(1) }
entityConfirmation-11b OID ::= {
  id-km-at-kAM-11 entityConfirmationFunction(2) }
keyTokenAndKeyConstruction-11 OID ::= {
  id-km-at-kAM-11 keyTokenProcessingAndKeyConstructionFunction(3) }
keyConstruction-11 OID ::= {
  id-km-at-kAM-11 keyConstructionFunction(4) }
keyVerification-11 OID ::= {
```

```
    id-km-at-kAM-11 keyVerificationFunction(5) }
    -- Key Transport Mechanism 1 -
    keyTokenConstruction-1 OID ::= {
      id-km-at-kTM-1 keyTokenConstructionFunction(1) }
    keyTokenDeconstruction-1 OID ::= {
      id-km-at-kTM-1 keyTokenDeconstructionFunction(2) }
    -- Key Transport Mechanism 2 --
    keyEncryption-2 OID ::= {
      id-km-at-kTM-2 keyEncryptionFunction(1) }
    keyTokenConstruction-2a OID ::= {
      id-km-at-kTM-2 keyTokenConstructionFunction(2) }
    keyTokenVerification-2 OID ::= {
      id-km-at-kTM-2 keyTokenVerificationFunction(3) }
    keyDecryption-2 OID ::= {
      id-km-at-kTM-2 keyDecryptionFunction(4) }
    -- Key Transport Mechanism 3 -
    keyBlockSignature-3 OID ::= {
      id-km-at-kTM-3 keyBlockSignatureFunction(1) }
    keyTokenConstruction-3 OID ::= {
      id-km-at-kTM-3 keyTokenConstructionFunction(2) }
    keyTokenDecryption-3 OID ::= {
      id-km-at-kTM-3 keyTokenDecryptionFunction(3) }
    keyBlockVerification-3 OID ::= {
      id-km-at-kTM-3 keyBlockVerificationFunction(4)
    -- Key Transport Mechanism 4 --
    keyTokenConstruction-4c OID ::= {
      id-km-at-kTM-4 keyTokenConstructionFunction(1)
    keyBlockEncryption-4 OID ::= {
      id-km-at-kTM-4 keyBlockEncryptionFunction(2) }
    keyTokenConstruction-4d OID ::= {
      id-km-at-kTM-4 keyTokenConstructionFunction(3)
    keyTokenVerification-4 OID ::= {
      id-km-at-kTM-4 keyTokenVerificationFunction(4)
    keyBlockDecryption-4 OID ::= {
      id-km-at-kTM-4 keyBlockDecryptionFunction(5) }
    -- Key Transport Mechanism 5 -
    keyTokenConstruction-5c OID ::= {
      id-km-at-kTM-5 keyTokenConstructionFunction(1)
    keyBlockEncryption-5b OID ::= {
      id-km-at-kTM-5 keyBlockEncryptionFunction(2) }
    keyTokenConstruction-5d OID ::= {
      id-km-at-kTM-5 keyTokenConstructionFunction(3)
    keyTokenVerification-5a OID ::= {
```

```
    id-km-at-kTM-5 keyTokenVerificationFunction(4)
keyBlockDecryption-5a OID ::= {
    id-km-at-kTM-5 keyBlockDecryptionFunction(5) }
keyBlockEncryption-5a OID ::= {
    id-km-at-kTM-5 keyBlockEncryptionFunction(6) }
keyTokenConstruction-5e OID ::= {
    id-km-at-kTM-5 keyTokenConstructionFunction(7)}
keyTokenVerification-5b OID ::= {
    id-km-at-kTM-5 keyTokenVerificationFunction(8) }
keyBlockDecryption-5b OID ::= {
    id-km-at-kTM-5 keyBlockDecryptionFunction(9) }
-- Key Transport Mechanism 6 --
keyTokenConstruction-6a OID ::= {
    id-km-at-kTM-6 keyTokenConstructionFunction(1) }
keyTokenConstruction-6b OID ::= {
    id-km-at-kTM-6 keyTokenConstructionFunction(2) }
keyEntityConfirmation-6a OID ::= {
id-km-at-kTM-6 keyEntityConfirmationFunction(3) }
keyTokenResponse-6 OID ::= {
id-km-at-kTM-6 keyResponseFunction(4) }
keyEntityConfirmation-6b OID ::= {
    id-km-at-kTM-6 keyEntityConfirmationFunction(5) }
-- Public Key Transport Mechanism 1 -
keyTokenConstruction-1a OID ::= {
    id-km-at-pKT-1 keyTokenConstructionFunction(1) }
keyTokenReception-1 OID ::= {
    id-km-at-pKT-1 keyTokenReceptionFunction(2) }
-- Public Key Transport Mechanism 2 -
keyTokenConstruction-2b OID ::= {
    id-km-at-pKT-2 keyTokenConstructionFunction(1) }
keyTokenReception-2 OID ::= {
    id-km-at-pKT-2 keyTokenReceptionFunction(2) }
keyTokenVerification-2a OID ::= {
    id-km-at-pKT-2 keyTokenVerificationFunction(3) }
-- Public Key Transport Mechanism 3 -
keyTokenConstruction-3a OID ::= {
    id-km-at-pKT-3 keyTokenConstructionFunction(1) }
certificationVerification-3 OID ::= {
    id-km-at-pKT-3 certificationVerificationFunction(2) }
    END - KeyManagemen
```

附 录 B
（资料性附录）
密钥建立机制特性

表 B.1 和表 B.2 给出了本部分密钥建立/传输机制的主要特性。

其中：

A：表示该机制为实体 A 提供的特性。

B：表示该机制为实体 B 提供的特性。

A，B：表示该机制为实体 A 和实体 B 提供的特性。

No：表示该机制不提供这种特性。

Opt：表示该机制通过附加方法，提供可选特性。

(A)：表示该机制通过附加方法，为实体 A 提供可选特性。

(B)：表示该机制通过附加方法，为实体 B 提供此种特性。

MFS：表示该机制提供前向互保密特性。

表 B.1 中的公钥操作：如非对称变换“(2F，1F)”的计算次数指在密钥协商机制 2 中实体 A 需两次函数 F 计算，实体 B 只需要一次函数 F 计算。非对称变换“(1FP，1FP，1FP)”的计算次数指在密钥协商机制 12 中实体 A 需一次函数 FP 计算，实体 B 需一次函数 FP 计算，实体 *C* 需一次函数 FP 计算。

公钥在表 B.2 中的操作：实体 *X* 执行非对称变换 D_X，S_X，V_X 的计算次数：如：“($1E_B$，$1D_B$)” 指实体 A 需一次 E_B 函数计算，实体 B 需一次函数 D_B 计算。

另一个从密钥更新中得到的重要特性是预防重放攻击。密钥更新可保证两实体间几乎不可能出现重放攻击。

隐式密钥鉴别的特性由其定义确定。当表中隐式密钥鉴别为“A”时，意味着实体 B 可确信实体 A 是唯一拥有正确密钥的实体。当表中隐式密钥鉴别为“A，B”时，意味着实体 A 和实体 B 可确信只有另一实体拥有正确的密钥。

注：本部分所采用国际标准涉及的专利信息参见附录 F。

表 B.1 密钥协商机制特性

机制	交互次数	隐式密钥鉴别	密钥确认	实体鉴别	公钥操作	前向保密	密钥更新
1	0	A，B	No	No	(1F，1F)	No	No
2	1	B	No	No	(2F，1F)	A	A
3	1	A，B	B	A	($2F/1S_A$，$2F/1V_A$)	A	A
4	2	No	No	No	(2F，2F)	MFS	A，B
5	2	A，B	Opt	No	(3F，3F)	A，B	A，B
6	2	A，B	Opt	B	($1V_B/1D_A$，$1S_B/1E_A$)	B	A，B
7	3	A，B	A，B	A，B	($2F/1V_B/1S_A$，$2F/1S_B/1V_A$)	MFS	A，B
8	1	A，B	No	No	(2F，1F)	A	A
9	2	A，B	No	No	(2F，2F)	MFS	A，B
10	3	A，B	A，B	A，B	(2F，2F)	MFS	A，B
11	4	B	A，B	B	($1V_{CA}/1E_B$，$1D_B$)	MFS	A，B
12	0	A，B，C	No	No	(1FP，1FP，1FP)	No	No

表 B.2　密钥传递机制特性

机制	交互次数	隐式密钥鉴别	密钥确认	密钥控制	实体鉴别	公钥操作	前向保密	密钥更新
1	1	B	No	A	No	$(1E_B,1D_B)$	A	A
2	1	B	B	A	A	$(1E_B/1S_A,1V_A/1D_B)$	A	A
3	1	B	B	A	A	$(1S_A/1E_B,1D_B/1V_A)$	A	A
4	2	A	A	B	B	$(1V_B/1D_A,1E_A/1S_B)$	B	A
5	3	A,B	(A),B	A,B	A,B		No	A,B
6	3	A,B	No	A,B	No	$(1E_B/1D_A,1D_B/1E_A)$	No	A,B

附 录 C
（资料性附录）
密钥派生函数实例

C.1 ASN.1 密钥派生函数的语法

本附录描述密钥派生函数的语法。

密钥派生函数的输入为共享密钥“*ZZ*”和其他信息“OtherInfo”。其他信息包括发起者的信息“entityAInfo”以及应答者的信息“entityBInfo”“suppPubInfo”和“suppPrivInfo”。

```
OtherInfo ::= SEQUENCE {
    keyInfo KeySpecificInfo,
    entityAInfo [0] OCTET STRING OPTIONAL,
    entityBInfo [1] OCTET STRING OPTIONAL,
    suppPubInfo [2] OCTET STRING OPTIONAL,
    suppPrivInfo [3] OCTET STRING OPTIONAL
}
KeySpecificInfo ::= SEQUENCE {
    algorithm OBJECT IDENTIFIER,
    counter Counter
}
Counter ::= INTEGER (1...32767)
```

suppPubInfo 和 suppPrivInfo 在密钥派生中是可选的字域。这些字域可用于承载额外的、补充的、公开和私有信息，这些信息通信方相互知道，而不是只有其中一方知道。

suppPubInfo 和 suppPrivInfo 的内容由密钥管理协议定义。这些 suppPubInfo 和 suppPrivInfo 字域的定义、语法以及编码规则由密钥管理协议负责，超出在本部分的范围。

所有密钥派生杂凑函数的输入在长度上应是八位位组的整数个数。suppPrivInfo 可包含 *ZZ*。

注：用第 11 章和第 12 章中的某些机制生成共享密钥，既可是椭圆曲线上的点，也可是椭圆曲线上两点间的级联。对于前者，为了获取共享密钥整数 *ZZ*，将该点代入函数 π 得到共享密钥整数 *ZZ*。

C.2 GB/T 32918.3—2016 密钥派生函数

见 GB/T 32918.3—2016 中 5.4.3 的密钥派生函数。

附 录 D
（资料性附录）
函数 F、集合 S_1 和 S_2 实例

本附录给出函数 F 以及相应集合 S_1 和 S_2 的一个广泛使用例子，只要适当选择参数，可以推测集合 S_1 和 S_2 满足第 10 章中的 5 个特性。

设 p 为一个素数，g 是 F_P 的一个本原元。设 $S_2=\{0,1,\cdots p-1\}$，$S_1=\{2,\cdots,p-2\}$。那么

$F(h,g)=g^h \bmod p$。

F 对 h 是可交换的，其中 $(g^{h_B})^{h_A}=(g^{h_A})^{h_B}=(g^{h_A h_B}) \bmod p$。

素数 p 应足够大，以使 $F(.,g)$ 可推测为一个单向函数。每个实体 X 在集合 S_1 中有一个私钥 h_x，且该私钥只有实体 X 知道，和一个可被其他实体知道的公钥 $p_x=g^{h_x} \bmod p$。

注 1：对于以素数为模的离散对数，素数大小的选择宜使在相应的循环群中计算离散对数在计算上是不可行的。为使计算离散对数不可行，可对该素数添加其他一些条件。建议选择一个强素数 p，使得 $p-1$ 有一个大的素因子 q，选择 g 为大素数阶 q 群的生成元。

注 2：对于以复合数为模的离散对数，该模数宜选为两个不同的奇素数乘积，并加以保密。模数大小的选择宜使分解该模数在计算上不可行的。为使分解该模数在计算上不可行，可对这些素数的选择添加其他一些条件。

附 录 E
（资料性附录）
基于椭圆曲线的密钥建立机制实例

E.1 函数 F 例子

本附录给出函数 F 一个广泛使用例子，只要适当选择参数假定特定的参数，可满足第 10 章中所列出的 5 个特性。

设 E 是定义在有限域 F_q 上的一条椭圆曲线，给定一个整数 d 及点 $G\in E(\mathrm{GF}(q))$，G 为基点，则函数 F 为 $\mathrm{F}(d,G)=dG$。函数 F 满足 $d_1(d_2G)=d_2(d_1G)=d_1d_2G$。

$E(F_q)$ 应足够大，使得 F(.,G)可看成一单向函数。设每个实体 X 在 $E(F_q)$ 上有且只有实体 X 知道的私钥 h_x，及可被其他实体知道的公钥 $p_x=h_xG$。

E.2 公共信息

对所有密钥协商体制，在进行协商共享秘密过程之前，应在各参与者间建立下列公共信息，并可选择性进行验证（参见 GB/T 32918.1 对参数验证的描述）：

——与密钥对关联的椭圆曲线参数，对所有参与者密钥对，这些参数应相同，包括 p，p^m，2^m 或 3^m，对 GF(q)，GF(p^m)，GF(2^m)或 GF(3^m)及基本参数 E，n，G 的描述。

在 X9.62 中详细说明的椭圆曲线标识符，提供了一种简便方法去标识椭圆曲线域参数，也可被用于指定公共信息值的组别。

在下列定义的每一机制中，最终达成一致的密钥不应直接用于加密。而该密钥应作为密钥派生函数的输入，使得参与者双方可由它导出相同的加密密钥。因此，双方参与者应就下列信息达成一致：

——一个密钥派生函数 kdf。

——密钥派生函数的参数。

——用于运行余子式乘法的类型（若有）。

E.3 SM2 密钥协商机制

见 GB/T 32918.3—2016。

附 录 F
（资料性附录）
所采用国际标准涉及的专利信息

本部分在推进过程中已广泛地征集意见，未收到任何针对本部分的专利声明。本标准修改采用ISO/IEC 11770-3:2015，ISO/IEC 11770-3:2015在其附录H中给出了ISO/IEC 11770-3:2015可能涉及的专利信息。本附录如下内容是对ISO/IEC 11770-3:2015附录H的翻译（仅对原来标记为“申请中”的两项专利EP 0639907、JP2010-7982的状态进行了更新），本部分的发布机构对于这些专利的真实性、有效性和范围无任何立场。

在起草ISO/IEC 11770-3:2015期间，收到了在ISO/IEC 11770-3:2015应用时可能依赖的相关专利的声明信息，已识别的专利信息如表F.1所示。但是，ISO和IEC对这些专利权的证据、有效性和范围等不持任何立场。

已识别的专利权人声明将以适当的条款进行专利许可，以使本部分得以应用，其前提是寻求获得许可的人同意。更多的信息可以从已识别的专利权人那里获得。

表F.1中的专利信息引自ISO/IEC 11770-3:2008。

表 F.1 ISO/IEC 11770-3:2008 可能涉及的专利信息

范围	发明人	专利号	授权日	联系地址
密钥分发方法	Eiji Okamoto	JP 1871933 US 4876716 EP 0257585 CA 1279709	1994-09-26 1989-10-24 1992-11-25 1991-01-01	General Manager Intellectual Property Division NEC Corporation 7-1, Shiba 5-chome, minato-ku Tokyo 108-8001, Japan
Goss密钥协商	Goss	US 4956863	1990-09-11	Jones Futurex™ Inc. 3715 Atherton Road Rocklin, CA USA 95765
基于ID的DH密钥协商	Eiji Okamoto Kazue Tanaka	JP 1871933 US 4876716 EP 0257585 CA 1279709 AS 618229	1998-01-09 1991-07-02 1997-08-06 1995-01-03 1992-05-04	General Manager Intellectual Property Division NEC Corporation 7-1, Shiba 5-chome, minato-ku Tokyo 108-8001, Japan
Nyberg-Rueppel密钥协商	Nyberg-Rueppel	US 5600725	1997-02-04	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward, California USA 94545
Nyberg-Rueppel签名	Nyberg-Rueppel	EP 0639907	1997-12-08	r3 security engineering ag 6th floor, Glatt Tower CH-8301 Glattzentrum Zurich, Switzerland

由于ISO/IEC 15946-3:2002中包含的机制已转移到ISO/IEC 11770-3:2008,表F.2中的专利信息引自ISO/IEC 15946-3:2002,在ISO/IEC JTC1/SC27 N4348常备文件8(SD8)专利信息中注册。

表F.2 ISO/IEC 15946-3:2002可能涉及的专利信息

范围	发明人	专利号	授权日	联系地址
DH密钥协商	Hellman-Diffie-Merkle	US 4200770	1980-04-29	—
密钥分发系统	Kazue Tanaka	US 5029208 AU 618229 CA 2011396 EP 0385511 JP 2734726	1991-07-02 1992-05-04 1995-01-03 1997-08-06 1998-01-09	General Manager Intellectual Property Division NEC Corporation 7-1, Shiba 5-chome, minato-ku Tokyo 108-8001, Japan
生成独特且不可预测的值	Johnson, Donald B.	US 6078667	2000-06-20	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward, California USA 94545
有限域乘法的方法和装置	Lambert, Robert J. Vadekar, Ashok	US 6049815	2000-04-11	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward, California USA 94545
有限域逆环	Dworkin, James D. Glaser, P.Michael Torla, Michael John Lambert, Robert J. Vadekar, Ashok Vanstone,Scot A.	US 6009450	1999-12-28	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward, California USA 94545
智能卡的数字签名	Mullin, Ronald Vanstone,Scott A.	US 5999626	1999-12-07	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward, California USA 94545
用于椭圆曲线处理器的有限域反向电路	Dworkin, James D. Torla, Michael J.Vanstone,Scot A.	US 5982895	1997-12-24	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward, California USA 94545

表 F.2（续）

范围	发明人	专利号	授权日	联系地址
智能卡的传输验证协议	Vanstone，Scot A.	US 5955717	1997-01-30	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545
增强型公钥协议	Vanstone，Scot A. Menezes，Alfred J.Qu， Minghua	US 5933504	1999-08-03	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545
隐式签名的密钥协商和传输协议	Vanstone，Scot A. Menezes，Alfred J. Qu，Minghua	US 5896455	1999-04—20	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545
隐式签名的密钥协商和传输协议	Vanstone，Scot A. Menezes，Alfred J. Qu，Minghua	US 5889865	1999-03-30	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545
多个位乘法器	Mullin，Ronald	US 5787028	1998-07-28	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545
隐式签名的密钥协商和传输协议	Vanstone，Scot A. Menezes，Alfred J. Qu，Minghua	US 5761305	1998-06-02	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545

表 F.2（续）

范围	发明人	专利号	授权日	联系地址
有限域乘法器的计算方法和装置	Onyszchuk，Ivan M. Mullin，R.Vanstone， Scott，A.	US 4745568	1988-05-17	Executive Vice President Field Operations Certicom Corp. 25801 Industrial Blvd Hayward，California USA 94545
用于在一对用户之间建立认证后的共享秘密值的方法和装置	Matyas，Jr. Stephen M.Johnson，Donald B.	US 5953420	1999-09-14	Director of Licensing International Business Machines Corporation North Castle Drive Armonk，NY 10504 USA

表 F.3 中的专利信息引自 ISO 专利数据库和 IEC 专利数据库，截至 2014 年 11 月 29 日。

表 F.3　ISO/IEC 11770-3:2015 可能涉及的专利信息

范围	发明人	专利号	授权日	联系地址
公钥加密系统、签名系统、加密通信系统、密钥生成器、公钥生成器和计算机程序	Murata Machinery， Ltd. Ryuichi Sakai Masao Kasahara	JP 4485122	已授权	Masahiro Tsubota Intellectual Property Department Murata Machinery，Ltd.136，Takeda-Mukaishiro-cho，Fushimi-ku，Kyoto，612-8686， Japan
基于 IP 的认证密钥交换系统、认证密钥交换方法、认证密钥交换设备，以及认证密钥交换程序和记录介质	—	JP 2010-7982	已授权	Kengo Nagata Licensing Group，Intellectual Property Center Nippon Telegraph and Telephone Corporation 9-11，Midori-cho，3-Chome Musashino-Shi，Tokyo，180-8585，Japan

参 考 文 献

[1] ISO 7498-2:1989 Information processing systems—Open systems interconnection—Basic reference model—Part 2: Security architecture

[2] ISO/IEC 9797(all parts) Information technology—Security techniques—Message authentication codes (MACs)

中 华 人 民 共 和 国
国 家 标 准
信息技术 安全技术 密钥管理
第 3 部分:采用非对称技术的机制
GB/T 17901.3—2021
*
中 国 标 准 出 版 社 出 版 发 行
北京市朝阳区和平里西街甲 2 号(100029)
北京市西城区三里河北街 16 号(100045)
网址:www.spc.org.cn
服务热线:400-168-0010
2021 年 3 月第一版
*
书号:155066·1-64858

GB/T 17901.3-2021