SM9 标识密码算法

第1 部分：总则

目　　次

SM9 标识密码算法

第 1 部分：总则

本部分描述了必要的数学基础知识与相关密码技术，以帮助实现本文其它各部分所规定的密码机制。

1 术语

1.1

标识 identity

可唯一确定一个实体身份的信息。标识应由实体无法否认的信息组成，如实体的可识别名称、电子邮箱、身份证号、电话号码等。

1.2

主密钥 master key

处于标识密码密钥分层结构最顶层的密钥，包括主私钥和主公钥，其中主公钥公开，主私钥由KGC秘密保存。KGC用主私钥和用户的标识生成用户的私钥。在标识密码中，主私钥一般由KGC通过随机数发生器产生，主公钥由主私钥结合系统参数产生。

本文中，签名系统的主密钥与加密系统的主密钥不同。数字签名算法属于签名系统，其主密钥为签名主密钥，密钥交换协议、密钥封装机制和公钥加密算法属于加密系统，其主密钥为加密主密钥。

1.3

密钥生成中心 key generation center；KGC

在SM9标识密码中，负责选择系统参数、生成主密钥并产生用户私钥的可信机构。

2 符号和缩略语

下列符号和缩略语适用于本部分。

cf：椭圆曲线阶相对于 N 的余因子。

cid：用一个字节表示的曲线识别符，用以区分所用曲线的类型。

DLP：有限域上离散对数问题。

$deg(f)$：多项式 $f(x)$的次数。

d_1、d_2：k 的两个因子。

E：定义在有限域上的椭圆曲线。

$ECDLP$：椭圆曲线离散对数问题。

$E(F_q)$：有限域 F_q 上椭圆曲线 E 的所有有理点(包括无穷远点 O)组成的集合。

$E(F_q)[r]$：$E(F_q)$上 r-扭点的集合(即曲线 $E(F_q)$上的 r 阶扭子群)。

e：从 $G_1 \times G_2$ 到 G_T的双线性对。

eid：用一个字节表示的双线性对 *e* 的识别符，用以区分所用双线性对的类型。
F_p：包含 *p* 个元素的素域。
F_q：包含 *q* 个元素的有限域。
F_q^*：由 F_q 中所有非零元构成的乘法群。
F_{q^m}：有限域 F_q 的 *m* 次扩域。
G_T：阶为素数 *N* 的乘法循环群。
G_1：阶为素数 *N* 的加法循环群。
G_2：阶为素数 *N* 的加法循环群。
$\gcd(x, y)$：*x* 和 *y* 的最大公因子。
k：曲线 $E(F_q)$相对于 *N* 的嵌入次数，其中 *N* 是$\#E(F_q)$的素因子。
m：有限域 F_{q^m}关于 F_q 的扩张次数。
mod *f*(*x*)：模多项式 *f*(*x*)的运算。
mod *n*：模 *n* 运算。例如，23 mod 7=2。
N：循环群 $\mathbb{G}_1$、$\mathbb{G}_2$ 和 $\mathbb{G}_T$ 的阶，为大于 2^{191} 的素数。
O：椭圆曲线上的一个特殊点，称为无穷远点或零点，是椭圆曲线加法群的单位元。
P：$P=(x_P, y_P)$ 是椭圆曲线上除 *O* 之外的一个点，其坐标 x_P，y_P 满足椭圆曲线方程。
P_1：$\mathbb{G}_1$ 的生成元。
P_2：$\mathbb{G}_2$ 的生成元。
P+*Q*：椭圆曲线 *E* 上两个点 *P* 与 *Q* 的和。
p：大于 2^{191} 的素数。
q：有限域 F_q 中元素的数目。
x_P：点 *P* 的 *x* 坐标。
x||*y*：*x* 与 *y* 的拼接，其中 *x* 和 *y* 是比特串或字节串。
$x \equiv y \pmod q$：*x* 与 *y* 模 *q* 同余。亦即，$x \bmod q = y \bmod q$。
y_P：点 *P* 的 *y* 坐标。
$\#E(K)$：*E*(*K*)上点的数目，称为椭圆曲线群 *E*(*K*)的阶，其中 *K* 为有限域（包括 F_q 和 F_{q^k}）。
<*P*>：由椭圆曲线上点 *P* 生成的循环群。
[*u*]*P*：椭圆曲线上点 *P* 的 *u* 倍点。
$[x, y]$：不小于 *x* 且不大于 *y* 的整数的集合。
$\lceil x \rceil$：顶函数，不小于 *x* 的最小整数。例如，$\lceil 7 \rceil = 7, \lceil 8.3 \rceil = 9$。
$\lfloor x \rfloor$：底函数，不大于 *x* 的最大整数。例如，$\lfloor 7 \rfloor = 7, \lfloor 8.3 \rfloor = 8$。
β：扭曲线参数。
ψ：$\mathbb{G}_2$ 到 $\mathbb{G}_1$ 的同态映射，满足 $P_1 = \psi(P_2)$。
⊕：长度相等的两个比特串按比特的模2加运算。

3 有限域和椭圆曲线

3.1 有限域

3.1.1 概述

域由一个非空集合**F**和两种运算共同组成，这两种运算分别为加法(用“+”表示)和乘法(用“·”表示)，并且满足下列算术特性：

a) $(\mathbf{F}, +)$对于加法运算构成加法交换群，单位元用0表示。
b) $(\mathbf{F}\backslash\{0\}, \cdot)$对于乘法运算构成乘法交换群，单位元用1表示。

c) 分配律成立：对于所有的$a,b,c\in\mathbf{F}$，都有$(a+b)\cdot c=a\cdot c+b\cdot c$。

若集合$\mathbf{F}$是有限集合，则称域为有限域。有限域的元素个数称为有限域的阶。

3.1.2 素域 F_p

阶为素数的有限域是素域。

设 p 是一个素数，则整数模 p 的全体余数的集合$\{0,1,2,...,p-1\}$关于模 p 的加法和乘法构成一个 p 阶素域，用符号 F_p 表示。

F_p 具有如下性质：

a) 加法单位元是 0；

b) 乘法单位元是 1；

c) 域元素的加法是整数的模 p 加法，即若 $a,b\in F_p$，则 $a+b=(a+b) \bmod p$；

d) 域元素的乘法是整数的模 p 乘法，即若 $a,b\in F_p$，则 $a\cdot b=(a\cdot b) \bmod p$。

3.1.3 有限域 F_{q^m}

设 q 是一个素数或素数方幂，$f(x)$是多项式环 $F_q[x]$上的一个 m $(m>1)$次不可约多项式(称为约化多项式或域多项式)，商环 $F_q[x]/(f(x))$是含 q^m 个元素的有限域(记为 F_{q^m})，称 F_{q^m} 是有限域 F_q 的扩域，域 F_q 为域 F_{q^m} 的子域，m 为扩张次数。F_{q^m} 可以看成 F_q 上的 m 维向量空间。F_{q^m} 的每一个元可以唯一地写成 $a_0\beta_0+a_1\beta_1+\cdots+a_{m-1}\beta_{m-1}$ 的形式，其中 $a_i\in F_q$，而$\beta_0,\beta_1,...,\beta_{m-1}$ 是向量空间 F_{q^m} 在 F_q 上的一组基。

F_{q^m}中的元素可以用多项式基或正规基表示。在本文中，如果不作特别说明，F_{q^m}中元素均采用多项式基表示。

不可约多项式$f(x)$可取为首一的多项式$f(x)=x^m+f_{m-1}x^{m-1}+\cdots+f_2x^2+f_1x+f_0$ (其中$f_i\in F_q$, $i=0,1,...,m-1$)，F_{q^m}中的元素由多项式环 $F_q[x]$中所有次数低于 m 的多项式构成。多项式集合$\{x^{m-1},x^{m-2},...,x,1\}$是 F_{q^m} 在 F_q 上的一组基，称为多项式基。域 F_{q^m} 上的任意一个元素 $a(x)=a_{m-1}x^{m-1}+a_{m-2}x^{m-2}+\cdots+a_1x+a_0$ 在 F_q 上的系数恰好构成了一个 m 维向量，用 $a=(a_{m-1},a_{m-2},...,a_1,a_0)$表示，其中分量 $a_i\in F_q$，$i=0,1,...,m-1$。

F_{q^m} 具有如下性质：

a) 零元 0 用 m 维向量(0,...,0,0)表示；

b) 乘法单位元 1 用 m 维向量(0,...,0,1)表示；

c) 两个域元素的加法为向量加法，各个分量用域 F_q 的加法；

d) 域元素 a 和 b 的乘法定义如下：设 a 和 b 对应的 F_q 上多项式为 $a(x)$和 $b(x)$，则 $a\cdot b$ 定义为多项式$(a(x)\cdot b(x)) \bmod f(x)$对应的向量。

e) 逆元：设 a 对应的 F_q 上多项式为 $a(x)$，a 的逆元 a^{-1} 对应的 F_q 上多项式为 $a^{-1}(x)$，那么有 $a(x)\cdot a^{-1}(x)\equiv 1 \bmod f(x)$。

关于有限域的扩域F_{q^m}更多细节，参见附录A.1。

3.2 有限域上的椭圆曲线

有限域 $F_{q^m}(m\geq 1)$上的椭圆曲线是由点组成的集合。在仿射坐标系下，椭圆曲线上点 P (非无穷远点)用满足一定方程的两个域元素 x_P 和 y_P 表示，x_P,y_P 分别称为点 P 的 x 坐标和 y 坐标，并记 $P=(x_P,y_P)$。

本部分描述特征为大素数 p 的域上的曲线。

本部分如果不作特别说明，椭圆曲线上的点均采用仿射坐标表示。

定义在 F_{p^m} 上的椭圆曲线方程为：

$$y^2=x^3+ax+b，\ a,b\in F_{p^m}，\text{且}\ 4a^3+27b^2\neq 0。 \tag{1}$$

椭圆曲线 $E(F_{p^m})$定义为：

$$E(F_{p^m})=\{(x,y)|\ x,y\in F_{p^m}，\text{且满足方程(1)}\}\cup\{O\}，\text{其中 } O \text{ 是无穷远点。}$$

椭圆曲线 $E(F_{p^m})$上的点的数目用$\#E(F_{p^m})$表示，称为椭圆曲线 $E(F_{p^m})$的阶。

本文规定素数 $p>2^{191}$。

设 E 和 E'是定义在 F_q 上的椭圆曲线，如果存在一个同构映射 ϕ_d：$E'(F_{q^d})\to E(F_{q^d})$，其中 d 是使映射存在的最小整数，则称 E'为 E 的 d 次扭曲线。当 $p\geq 5$ 时，d 的取值有三种情况：

a) 若 $a=0, b\neq 0$, 那么 $d=6$，E': $y^2=x^3+\beta b$，$\phi_6 : E' \to E : (x,y) \mapsto (\beta^{-1/3}x, \beta^{-1/2}y)$；

b) 若 $b=0, a\neq 0$, 那么 $d=4$，E': $y^2=x^3+\beta a x$，$\phi_4 : E' \to E : (x,y) \mapsto (\beta^{-1/2}x, \beta^{-3/4}y)$；

c) 若 $a\neq 0, b\neq 0$, 那么 $d=2$，E': $y^2=x^3+\beta^2 a x+\beta^3 b$，$\phi_2 : E' \to E : (x,y) \mapsto (\beta^{-1}x, \beta^{-3/2}y)$。

3.3 椭圆曲线群

椭圆曲线 $E(F_{p^m})$ $(m\geq 1)$上的点按照下面的加法运算规则，构成一个交换群：

a) $O+O=O$；

b) $\forall P=(x,y)\in E(F_{p^m})\backslash\{O\}$，$P+O=O+P=P$；

c) $\forall P=(x,y)\in E(F_{p^m})\backslash\{O\}$，$P$ 的逆元素$-P=(x,-y)$，$P+(-P)=O$；

d) 两个非互逆的不同点相加的规则：

设 $P_1=(x_1,y_1)\in E(F_{p^m})\backslash\{O\}$，$P_2=(x_2,y_2)\in E(F_{p^m})\backslash\{O\}$，且 $x_1\neq x_2$，

设 $P_3=(x_3,y_3)=P_1+P_2$，则

$$\begin{cases} x_3=\lambda^2-x_1-x_2, \\ y_3=\lambda(x_1-x_3)-y_1, \end{cases}$$

其中

$$\lambda=\frac{y_2-y_1}{x_2-x_1}；$$

e) 倍点规则：

设 $P_1=(x_1,y_1)\in E(F_{p^m})\backslash\{O\}$，且 $y_1\neq 0$，$P_3=(x_3,y_3)=P_1+P_1$，

则

$$\begin{cases} x_3=\lambda^2-2x_1, \\ y_3=\lambda(x_1-x_3)-y_1, \end{cases}$$

其中

$$\lambda=\frac{3x_1^2+a}{2y_1}。$$

3.4 椭圆曲线多倍点运算

椭圆曲线上同一个点的重复相加称为该点的多倍点运算。设 u 是一个正整数，P 是椭圆曲线上的点，其 u 倍点 $Q=[u]P=\underbrace{P+P+\cdots+P}_{u个}$。

多倍点运算可以扩展到 0 倍点运算和负数倍点运算：$[0]P=O$，$[-u]P=[u](-P)$。

多倍点运算可以通过一些技巧有效地实现，参见附录A.2。

3.5 椭圆曲线子群上点的验证

输入：定义 F_{q^m} 上(q 为奇素数，$m\geq 1$)椭圆曲线方程的参数 a、b，椭圆曲线 $E(F_{q^m})$上子群 $\boldsymbol{G}$ 的阶 N，F_{q^m} 上的一对元素(x, y)。

输出：若(x, y)是群 $\mathbb{G}$ 中的元素，则输出“有效”；否则输出“无效”。

a) 在 F_{q^m} 上验证(x, y)是否满足椭圆曲线方程 $y^2 = x^3 + ax + b$；

b) 令 $Q=(x, y)$，验证$[N]Q = O$；

若以上任何一项验证失败，则输出“无效”；否则，输出“有效”。

3.6 离散对数问题

3.6.1 有限域上离散对数问题(*DLP*)

有限域 F_{q^m} (q 为奇素数，$m\geq 1$)的全体非零元素构成一个乘法循环群，记为 $F_{q^m}^*$。$F_{q^m}^*$中存在元素 g，使得 $F_{q^m}^*=\{g^i \mid 0\leq i\leq q^m-2\}$，称 g 为生成元。$F_{q^m}^*$中元素 a 的阶是满足 $a^t = 1$ 的最小正整数 t。群 $F_{q^m}^*$的阶为 q^m-1，因此 $t \mid q^m-1$。

设乘法循环群$F_{q^m}^*$的生成元为g，$y\in F_{q^m}^*$，有限域上离散对数问题是指确定整数$x\in[0, q^m-2]$，使得 $y = g^x$ 在$F_{q^m}^*$上成立。

3.6.2 椭圆曲线离散对数问题(*ECDLP*)

已知椭圆曲线$E(F_{q^m})$ ($m\geq 1$)，阶为n的点$P\in E(F_{q^m})$及$Q\in\langle P\rangle$，椭圆曲线离散对数问题是指确定整数$l\in[0, n-1]$，使得$Q = [l]P$成立。

4 双线性对及安全曲线

4.1 双线性对

设$(\mathbb{G}_1, +)$、$(\mathbb{G}_2, +)$和$(\mathbb{G}_T, \cdot)$是三个循环群，$\mathbb{G}_1$、$\mathbb{G}_2$ 和 $\mathbb{G}_T$ 的阶均为素数 N，P_1 是 $\mathbb{G}_1$ 的生成元，P_2 是 $\mathbb{G}_2$ 的生成元，存在 $\mathbb{G}_2$ 到 $\mathbb{G}_1$ 的同态映射 ψ使得 $\psi(P_2)=P_1$；

双线性对 e 是 $\mathbb{G}_1\times\mathbb{G}_2\rightarrow\mathbb{G}_T$的映射，满足如下条件：

a) 双线性性：对任意的 $P\in\mathbb{G}_1$，$Q\in\mathbb{G}_2$，$a, b\in\mathbb{Z}_N$，有 $e([a]P, [b]Q)=e(P, Q)^{ab}$；

b) 非退化性：$e(P_1, P_2)\neq 1_{\mathbb{G}_T}$；

c) 可计算性：对任意的 $P\in\mathbb{G}_1$，$Q\in\mathbb{G}_2$，存在有效的算法计算 $e(P, Q)$。

本部分所用的双线性对定义在椭圆曲线群上，主要有Weil对、Tate对、Ate对、R-ate对等。

4.2 安全性

双线性对的安全性主要建立在以下几个问题的难解性基础之上：

问题 1 (双线性逆 DH(BIDH)) 对 $a, b\in[1, N-1]$，给定$([a]P_1, [b]P_2)$，计算 $e(P_1, P_2)^{b/a}$ 是困难的。

问题 2 (判定性双线性逆 DH(DBIDH)) 对 $a, b, r\in[1, N-1]$，区分$(P_1, P_2, [a]P_1, [b]P_2, e(P_1, P_2)^{b/a})$和$(P_1, P_2, [a]P_1, [b]P_2, e(P_1, P_2)^r)$是困难的。

问题 3 (τ-双线性逆 DH(τ-BDHI)) 对正整数 τ和 $x\in[1, N-1]$，给定$(P_1, [x]P_1, P_2, [x]P_2, [x^2]P_2, ..., [x^\tau]P_2)$，计算 $e(P_1, P_2)^{1/x}$ 是困难的。

问题 4 (τ-Gap-双线性逆 DH(τ-Gap-BDHI)) 对正整数 τ和 $x\in[1, N-1]$，给定$(P_1, [x]P_1, P_2, [x]P_2, [x^2]P_2, \ldots, [x^\tau]P_2)$和 DBIDH 确定算法，计算 $e(P_1, P_2)^{1/x}$ 是困难的。

上述问题的难解性是SM9标识密码的安全性的重要基础，这些问题的难解性都意味着$\mathbb{G}_1$、$\mathbb{G}_2$和$\mathbb{G}_T$上的离散对数问题难解，选取的椭圆曲线应首先使得离散对数问题难解。

4.3 嵌入次数及安全曲线

设G是椭圆曲线$E(F_q)$的N阶子群，使$N|q^k-1$成立的最小正整数k称为子群G相对于N的嵌入次数，也称为曲线$E(F_q)$相对于N的嵌入次数。

设G_1是$E(F_{qd_1})$(d_1整除k)的N阶子群，G_2是$E(F_{qd_2})$(d_2整除k)的N阶子群，则椭圆曲线双线性对的值域G_T是$F_q{}^{k*}$的子群，因此椭圆曲线双线性对可将椭圆曲线离散对数问题转化为有限域$F_q{}^{k*}$上离散对数问题。嵌入次数越大安全性越高，但双线性对的计算越困难，因而需要采用嵌入次数适中且达到安全性标准的椭圆曲线。本文规定$q^k>2^{1536}$。

本文规定选用如下的曲线：

a) 基域q为大于2^{191}的素数、嵌入次数$k=2^i3^j$的常曲线，其中$i>0, j\geq 0$；

b) 基域q为大于2^{768}的素数、嵌入次数$k=2$的超奇异曲线；

对小于2^{360}的N，建议：

c) $N-1$含有大于2^{190}的素因子；

d) $N+1$含有大于2^{120}的素因子。

5 数据类型及其转换

5.1 数据类型

数据类型包括比特串、字节串、域元素、椭圆曲线上的点和整数。

比特串：有序的 0 和 1 的序列。

字节串：有序的字节序列，其中 8 比特为 1 个字节，最左边的比特为最高位。

域元素：有限域 F_{q^m} ($m\geq 1$)中的元素。

椭圆曲线上的点：椭圆曲线 $E(F_{q^m})(m\geq 1)$上的点 P 或者是无穷远点 O，或者是一对域元素(x_P, y_P)，其中域元素 x_P 和 y_P 满足椭圆曲线方程。

点的字节串表示有多种形式，用一个字节 PC 加以标识。无穷远点 O 的字节串表示是单一的零字节 $PC=00$。非无穷远点 $P=(x_P, y_P)$有如下三种字节串表示形式：

a) 压缩表示形式，$PC=02$ 或 03；

b) 未压缩表示形式，$PC=04$；

混合表示形式，$PC=06$或07。

注：混合表示形式既包含压缩表示形式又包含未压缩表示形式。在实现中，它允许转换到压缩表示形式或者未压缩表示形式。对于椭圆曲线上点的压缩表示形式和混合表示形式，本文定为可选形式。椭圆曲线上点的压缩表示形式参见附录A.4。

5.2 数据类型转换

5.2.1 数据类型转换关系

图1表示了各种数据类型之间的转换关系，线上的标志是相应数据转换方法所在的条。

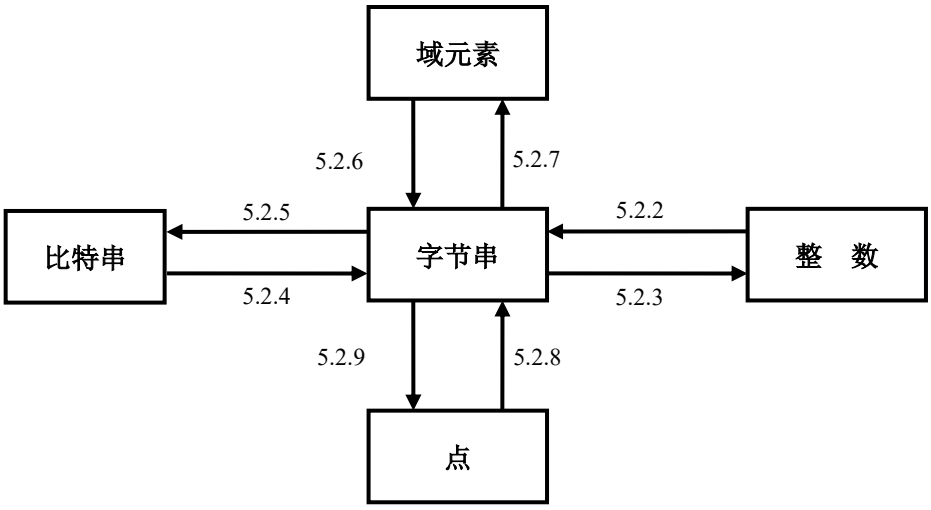

图 1　数据类型和转换约定

5.2.2　整数到字节串的转换

输入：非负整数 x，以及字节串的目标长度 l (其中 l 满足 $2^{8l}>x$)。

输出：长度为 l 的字节串 M。

a)　设 $M_{l-1}, M_{l-2}, \ldots, M_0$ 是 M 的从最左边到最右边的字节；

b)　M 的字节满足：

$$x=\sum_{i=0}^{l-1}2^{8i}M_i \text{。}$$

5.2.3　字节串到整数的转换

输入：长度为 l 的字节串 M。

输出：整数 x。

a)　设 $M_{l-1}, M_{l-2}, \ldots, M_0$ 是 M 的从最左边到最右边的字节；

b)　将 M 转换为整数 x：

$$x=\sum_{i=0}^{l-1}2^{8i}M_i \text{。}$$

5.2.4　比特串到字节串的转换

输入：长度为 n 的比特串 s。

输出：长度为 l 的字节串 M，其中 $l=\lceil n/8\rceil$。

a)　设 $s_{n-1}, s_{n-2}, \ldots, s_0$ 是 s 从最左边到最右边的比特；

b)　设 $M_{l-1}, M_{l-2}, \ldots, M_0$ 是 M 从最左边到最右边的字节，则

$M_i=s_{8i+7}\, s_{8i+6} \ldots s_{8i+1}\, s_{8i}$，其中$0\le i<l$，当$8i+j\ge n$，$0<j\le 7$时，$s_{8i+j}=0$。

5.2.5　字节串到比特串的转换

输入：长度为 l 的字节串 M。

输出：长度为 n 的比特串 s，其中 $n=8l$。

a)　设 $M_{l-1}, M_{l-2}, \ldots, M_0$ 是 M 从最左边到最右边的字节；

b)　设 $s_{n-1}, s_{n-2}, \ldots, s_0$ 是 s 从最左边到最右边的比特，则 s_i 是 M_j 右起第 $i-8j+1$ 比特，其中 $j=\lfloor i/8\rfloor$。

5.2.6　域元素到字节串的转换

输入：$F_{q^m}(m\geq 1)$中的元素$\alpha=(a_{m-1}, a_{m-2}, \ldots, a_1, a_0)$，$q=p$。

输出：长度 l 的字节串 S，其中 $l=\lceil \log_2 q/8 \rceil \times m$。

a) 若 $m=1$，则$\alpha=a_0(q=p)$，α必为区间$[0, q-1]$中的整数，按 6.2.2 的细节把α转换成长度为 l 的字节串 S；

b) 若 $m>1$，则$\alpha=(a_{m-1}, a_{m-2}, \ldots, a_1, a_0)$ $(q=p)$，其中 $a_i \in F_q$，$i=0,1,\ldots,m-1$；

1) 置 $r=\lceil \log_2 q/8 \rceil$；
2) 对 i 从 $m-1$ 到 0 执行：
按 6.2.2 的细节把 $a_i(q=p)$转换成长度为 r 的字节串 s_i；
3) $S=s_{m-1} \| s_{m-2} \| \ldots \| s_0$。

5.2.7 字节串到域元素的转换

情形 1：转换为基域中元素

输入：域 F_q，$q=p$，长度为 l 的字节串 S，$l=\lceil \log_2 q/8 \rceil$。

输出：F_q 中的元素α。

若 $q=p$，则按 6.2.3 的细节将 S 转换为整数α，若$\alpha \notin [0, q-1]$，则报错；

情形 2：转换为扩域中元素

输入：域 $F_{q^m}(m\geq 2)$，$q=p$，长度为 l 的字节串 S，其中 $l=\lceil \log_2 q/8 \rceil \times m$。

输出：F_{q^m} 中的元素α。

a) 将字节串 S 平均分成 m 段，每段长度为 l/m，记作 $S=(S_{m-1}, S_{m-2}, \ldots, S_1, S_0)$；
b) 对 i 从 $m-1$ 到 0 执行：
按 6.2.3 的细节将 S_i 转换为整数 a_i，若 $a_i \notin [0, q-1]$，则报错；
c) 若$q=p$，输出$\alpha=(a_{m-1}, a_{m-2}, \ldots, a_1, a_0)$。

5.2.8 点到字节串的转换

点到字节串的转换分为两种情形：一种是在计算过程中，将椭圆曲线点转换为字节串后才能作为某个函数(如杂凑函数)的输入，这种情况下只需直接将点转换为字节串；一种是在传输或存储椭圆曲线点时，为了减少传输的量或存储空间，可采用点的压缩或混合压缩表示形式，这种情况下需要加入一个字节的识别符 PC 来指示点的表示形式。下面分两种情况说明详细的转换过程。

情形 1：直接转换

输入：椭圆曲线 $E(F_{q^m})(m\geq 1)$上的点 $P=(x_P, y_P)$，且 $P \neq O$。

输出：长度为 $2l$ 的字节串 $X_1 \| Y_1$。(当 $m=1$ 时，$l=\lceil \log_2 q/8 \rceil$；当 $m>1$ 时，$l=\lceil \log_2 q/8 \rceil \times m$。)

a) 按 6.2.6 中的细节把域元素 x_P 转换成长度为 l 的字节串 X_1；
b) 按 6.2.6 中的细节把域元素 y_P 转换成长度为 l 的字节串 Y_1；
c) 输出字节串 $X_1 \| Y_1$。

情形 2：添加一字节识别符 *PC* 的转换

输入：椭圆曲线 $E(F_{q^m})(m\geq 1)$上的点 $P=(x_P, y_P)$，且 $P\neq O$。

输出：字节串 PO。若选用未压缩表示形式或混合表示形式，则输出字节串长度为 $2l+1$；若选用压缩表示形式，则输出字节串长度为 $l+1$。(当 $m=1$ 时，$l=\lceil \log_2 q/8 \rceil$；当 $m>1$ 时，$l=\lceil \log_2 q/8 \rceil \times m$。)

a) 按 6.2.6 中的细节把域元素 x_P 转换成长度为 l 的字节串 X_1；

b) 若选用压缩表示形式，则：

1) 计算比特 $\tilde{y}_P$；(参见附录 A.4。)

2) 若 $\tilde{y}_P=0$，则令 $PC=02$；若 $\tilde{y}_P=1$，则令 $PC=03$；

3) 字节串 $PO=PC\|X_1$；

c) 若选用未压缩表示形式，则：

1) 按 6.2.6 的细节把域元素 y_P 转换成长度为 l 的字节串 Y_1；

2) 令 $PC=04$；

3) 字节串 $PO=PC\|X_1\|Y_1$；

d) 若选用混合表示形式，则：

1) 按 6.2.6 的细节把域元素 y_P 转换成长度为 l 的字节串 Y_1；

2) 计算比特 $\tilde{y}_P$；(参见附录 A.4。)

3) 若 $\tilde{y}_P=0$，则令 $PC=06$；若 $\tilde{y}_P=1$，则令 $PC=07$；

4) 字节串 $PO=PC\|X_1\|Y_1$。

5.2.9 字节串到点的转换

字节串到点的转换是 6.2.8 的逆过程。下面也分两种情况加以说明。

情形 1：直接转换

输入：定义 $F_{q^m}(m\geq 1)$上椭圆曲线的域元素 a、b，长度为 $2l$ 的字节串 $X_1\|Y_1$，X_1、Y_1 的长度均为 l (当 $m=1$ 时，$l=\lceil \log_2 q/8 \rceil$；当 $m>1$ 时，$l=\lceil \log_2 q/8 \rceil \times m$)。

输出：椭圆曲线上的点 $P=(x_P, y_P)$，且 $P\neq O$。

a) 按 6.2.7 的细节把字节串 X_1 转换成域元素 x_P；

b) 按 6.2.7 的细节把字节串 Y_1 转换成域元素 y_P。

情形 2：包含一字节识别符 *PC* 的字节串的转换

输入：定义 $F_{q^m}(m\geq 1)$上椭圆曲线的域元素 a、b，字节串 PO。若选用未压缩表示形式或混合表示形式，则字节串 PO 长度为 $2l+1$；若选用压缩表示形式，则字节串 PO 长度为 $l+1$(当 $m=1$ 时，$l=\lceil \log_2 q/8 \rceil$；当 $m>1$ 时，$l=\lceil \log_2 q/8 \rceil \times m$)。

输出：椭圆曲线上的点 $P=(x_P, y_P)$，且 $P\neq O$。

a) 若选用压缩表示形式，则 $PO=PC\|X_1$；若选用未压缩表示形式或混合表示形式，则 $PO=PC\|X_1\|Y_1$，其中 PC 是单一字节，X_1 和 Y_1 都是长度为 l 的字节串；

b) 按 6.2.7 的细节把字节串 X_1 转换成域元素 x_P；

c) 若选用压缩表示形式，则：

1) 检验 $PC=02$ 或者是 $PC=03$，若不是这种情形，则报错；

2) 若 $PC=02$，则令 $\tilde{y}_P=0$；若 $PC=03$，则令 $\tilde{y}_P=1$；

3) 将 $(x_P, \tilde{y}_P)$ 转换为椭圆曲线上的一个点 (x_P, y_P)；(参见附录 A.4。)

d) 若选用未压缩表示形式，则：

1) 检验 $PC=04$，若不是这种情形，则报错；
2) 按 6.2.7 的细节把字节串 Y_1 转换成域元素 y_P；

e) 若选用混合表示形式，则：
1) 检验 $PC=06$ 或者 $PC=07$，若不是这种情形，则报错；
2) 执行步骤 e.2.1)或者 e.2.2)：
- 按 6.2.7 的细节把字节串 Y_1 转换成域元素 y_P；
- 若 $PC=06$，则令 $\tilde{y}_P=0$，否则令 $\tilde{y}_P=1$；

将 $(x_P,\tilde{y}_P)$ 转换为椭圆曲线上的一个点(x_P,y_P)；(参见附录 A.4。)

f) 验证(x_P,y_P)是否满足曲线方程，若不满足，则报错；

g) $P=(x_P,y_P)$。

6 系统参数及其验证

6.1 系统参数

系统参数包括：

a) 曲线的识别符 *cid*，用一个字节表示：0x10 表示 F_q(素数 $q>3$)上常曲线，0x11 表示 F_q 上超奇异曲线，0x12 表示 F_q 上常曲线及其扭曲线；
b) 椭圆曲线基域 F_q 的参数：基域参数为大于 3 的素数 q；
c) F_q 中的两个元素 a 和 b，它们定义椭圆曲线 E 的方程：$y^2=x^3+ax+b$；扭曲线参数β(若 *cid* 的低 4 位为 2)；
d) 余因子 *cf* 和素数 N，其中 $cf\times N=\#E(F_q)$，规定 $N>2^{191}$ 且 N 不整除 *cf*，如果 N 小于 2^{360}，建议 $N-1$ 含有大于 2^{190} 的素因子，$N+1$ 含有大于 2^{120} 的素因子；
e) 曲线 $E(F_q)$相对于 N 的嵌入次数 k (N 阶循环群$(\mathbb{G}_T,\cdot)\subset F_q^*$)，规定 $q^k>2^{1536}$；
f) N 阶循环群$(\mathbb{G}_1,+)$的生成元 $P_1=(x_{P_1},y_{P_1})$，$P_1\neq O$；
g) N 阶循环群$(\mathbb{G}_2,+)$的生成元 $P_2=(x_{P_2},y_{P_2})$，$P_2\neq O$；
h) 双线性对 e：$\mathbb{G}_1\times\mathbb{G}_2\to\mathbb{G}_T$，用一个字节的识别符 *eid* 表示：0x01 表示 Tate 对，0x02 表示 Weil 对，0x03 表示 Ate 对，0x04 表示 R-ate 对；
i) (选项)参数 d_1,d_2，其中 d_1,d_2 整除 k；
j) (选项)$\mathbb{G}_2$ 到 $\mathbb{G}_1$ 的同态映射 ψ，使得 $P_1=\psi(P_2)$。
k) (选项)BN 曲线的基域特征 q，曲线阶 r，Frobenius 映射的迹 tr 可通过参数 t 来确定，t 至少达到 63 比特。

6.2 系统参数的验证

下面的条件应由系统参数的生成者加以验证。这些条件也能由系统参数的用户验证。

输入：系统参数集合。

输出：若所有参数有效，则输出“有效”；否则输出“无效”。

a) 验证 q 是大于 3 的素数(参见附录 C.1.5)；
b) 验证 a,b 是区间[0，$q-1$]中的整数；
c) 验证在 F_q 上 $4a^3+27b^2\neq 0$；若 *cid* 的低 4 位为 2，验证β是非平方元(参见附录 C.1.4.3.1)；
d) 验证 N 为大于 2^{191} 的素数且 N 不整除 *cf*，如果 N 小于 2^{360}，验证 $N-1$ 含有大于 2^{190} 的素因子，$N+1$ 含有大于 2^{120} 的素因子；
e) 验证$|q+1-cf\times N|<2q^{1/2}$；

f) 验证 $q^k > 2^{1536}$，且 k 为使 $N|(q^m-1)$成立的最小正整数 m；

g) 验证(x_{P_1}, y_{P_1})是群 $\mathbb{G}_1$ 中的元素；

h) 验证(x_{P_2}, y_{P_2})是群 $\mathbb{G}_2$ 中的元素；

i) 验证 $e(P_1, P_2)\in F_{q^k}^*\backslash\{1\}$，且 $e(P_1, P_2)^N=1$；

j) （选项）验证 d_1, d_2 整除 k；

k) （选项）验证 $P_1=\psi(P_2)$；

l) （选项）验证 t 至少达到 63 比特。

若以上任何一项验证失败，则输出“无效”；否则，输出“有效”。

附 录 A
关于椭圆曲线的背景知识

A.1 有限域

A.1.1 素域F_p

设 p 是一个素数，整数模 p 的全体余数的集合{0,1,2,...,p−1}关于模 p 的加法和乘法构成一个 p 阶素域，用符号 F_p 表示。加法单位元是 0，乘法单位元是 1，F_p 的元素满足如下运算法则：

——**加法**：设 $a, b \in F_p$，则 $a + b = r$，其中 $r = (a + b) \bmod p$, $r \in [0, p-1]$。

——**乘法**：设 $a, b \in F_p$，则 $a \cdot b = s$, 其中 $s = (a \cdot b) \bmod p$, $s \in [0, p-1]$。

记F_p^*是由F_p中所有非零元构成的乘法群，由于F_p^*是循环群，所以在F_p中至少存在一个元素g，使得F_p中任一非零元都可以由g的一个方幂表示，称g为F_p^*的生成元(或本原元)，即$F_p^*=\{g^i \mid 0 \le i \le p-2\}$。设$a=g^i \in F_p^*$，其中$0 \le i \le p-2$，则$a$的乘法逆元为：$a^{-1}= g^{p-1-i}$

示例 1：素域 F_{19}，$F_{19}=\{0, 1, 2, \ldots, 18\}$。

F_{19} 中加法的示例：$10, 14 \in F_{19}$，10+14=24，24 mod 19 ＝5，则 10+14=5。

F_{19} 中乘法的示例：$7, 8 \in F_{19}$，7×8=56，56 mod 19 ＝18，则 7 · 8=18。

13 是 F_{19}^* 的一个生成元，则 F_{19}^* 中元素可由 13 的方幂表示出来：

$13^0 = 1$，$13^1 = 13$，$13^2 = 17$，$13^3 = 12$，$13^4 = 4$，$13^5 = 14$，$13^6 = 11$，$13^7 = 10$，$13^8 = 16$，$13^9 = 18$，

$13^{10} = 6$，$13^{11} = 2$，$13^{12} = 7$，$13^{13} = 15$，$13^{14} = 5$，$13^{15} = 8$，$13^{16} = 9$，$13^{17} = 3$，$13^{18} = 1$。

A.1.2 有限域F_{q^m}

设 q 是一个素数或素数方幂，$f(x)$是多项式环 $F_q[x]$上的一个 m (m＞1)次不可约多项式(称为约化多项式或域多项式)，商环 $F_q[x]/(f(x))$是含 q^m 个元素的有限域(记为 F_{q^m})，称 F_{q^m} 是有限域 F_q 的扩域，域 F_q 为域 F_{q^m} 的子域，m 为扩张次数。F_{q^m} 可以看成 F_q 上的 m 维向量空间，也就是说，在 F_{q^m} 中存在 m 个元素 $\alpha_0, \alpha_1, \ldots, \alpha_{m-1}$，使得 $\forall a \in F_{q^m}$，a 可以唯一表示为：$a = a_{m-1}\alpha_{m-1} + \cdots + a_1\alpha_1 + a_0\alpha_0$，其中 $a_i \in F_q$，称{α_{m-1}, …,α_1, α_0}为 F_{q^m} 在 F_q 上的一组基。给定这样一组基，就可以由向量(a_{m-1},a_{m-2},…,a_1,a_0)来表示域元素 a。

F_{q^m}在F_q上的基有多种选择：多项式基和正规基等。

不可约多项式$f(x)$可取为首一的多项式$f(x)=x^m+f_{m-1}x^{m-1}+ \cdots + f_2x^2+ f_1x+ f_0$ (其中$f_i \in F_q$, i=0,1,…,m−1)，F_{q^m}中的元素由多项式环$F_q[x]$中所有次数低于m的多项式构成，即$F_{q^m}= \{a_{m-1}x^{m-1}+ a_{m-2}x^{m-2}+ \cdots + a_1x+ a_0 \mid a_i \in F_q, i = 0,1, \ldots,m\text{-}1\}$。多项式集合$\{x^{m-1},x^{m-2},\ldots, x,1\}$是$F_{q^m}$作为向量空间在$F_q$上的一组基，称为多项式基。当$m$含有因子$d$($1<d<m$)时，$F_{q^m}$可以由$F_{q^d}$扩张生成，从$F_{q^d}[x]$中选取一个合适的$m/d$次不可约多项式作为$F_{q^m}$在$F_{q^d}$上的约化多项式，$F_{q^m}$可以由塔式扩张方法(towering method)得到，这种扩张的基本形式仍是由F_q中元素组成的向量。例如当m=6时，可以先由F_q经过3次扩张得扩域F_{q^3}，再由F_{q^3}经过2次扩张得到扩域F_{q^6}；也可以先由F_q经过2次扩张得扩域F_{q^2}，再由F_{q^2}经过3次扩张得到扩域F_{q^6} 。

F_{q^m}在F_q上形如$\{\beta,\beta^q,\beta^{q^2},\ldots,\beta^{q^{m-1}}\}$的一组基称为正规基，其中$\beta \in F_{q^m}$ 。$\forall a \in F_{q^m}$，a可以唯一表示为：$a=a_0\beta+a_1\beta^q+\cdots+a_{m-1}\beta^{q^{m-1}}$，其中$a_i \in F_q$，$i$=0,1,…,$m$−1。对于任意有限域$F_q$及其扩域$F_{q^m}$，这样的基总是存在的。

如果不作特别说明，F_{q^m} 中元素均采用多项式基表示。

域元素 $a_{m-1}x^{m-1}+ a_{m-2}x^{m-2}+ \cdots + a_1x+ a_0$ 相对于多项式基可以由向量(a_{m-1},a_{m-2},…,a_1,a_0)表示，所以$F_{q^m}= \{(a_{m-1},a_{m-2},\ldots,a_1,a_0) \mid a_i \in F_q, i = 0,1,\ldots,m\text{-}1\}$。

乘法单位元 1 由(0,…,0,1)表示，零元由(0,…,0,0)表示。域元素的加法和乘法定义如下：

——加法运算

$\forall(a_{m-1},a_{m-2},\ldots,a_1,a_0)$，$(b_{m-1},b_{m-2},\ldots,b_1,b_0)\in F_{q^m}$，则$(a_{m-1},a_{m-2},\ldots,a_1,a_0)+(b_{m-1},b_{m-2},\ldots,b_1,b_0)=(c_{m-1},c_{m-2},\ldots,c_1,c_0)$，其中$c_i=a_i+b_i\in F_q$，$i=0,1,\ldots,m-1$，亦即，加法运算按分量执行域$F_q$的加法运算。

——乘法运算

$\forall(a_{m-1},a_{m-2},\ldots,a_1,a_0)$，$(b_{m-1},b_{m-2},\ldots,b_1,b_0)\in F_{q^m}$，则$(a_{m-1},a_{m-2},\ldots,a_1,a_0)\cdot(b_{m-1},b_{m-2},\ldots,b_1,b_0)=(r_{m-1},r_{m-2},\ldots,r_1,r_0)$，其中多项式$(r_{m-1}x^{m-1}+r_{m-2}x^{m-2}+\cdots+r_1x+r_0)$是$(a_{m-1}x^{m-1}+a_{m-2}x^{m-2}+\cdots+a_1x+a_0)\cdot(b_{m-1}x^{m-1}+b_{m-2}x^{m-2}+\cdots+b_1x+b_0)$在$F_q[x]$中模$f(x)$的余式。

F_{q^m}恰包含q^m个元素。记$F_{q^m}^*$是由F_{q^m}中所有非零元构成的乘法群，$F_{q^m}^*$是循环群，在F_{q^m}中至少存在一个元素g，使得F_{q^m}中任一非零元都可以由g的一个方幂表示，称g为$F_{q^m}^*$的生成元(或本原元)，即：$F_{q^m}^*=\{g^i|0\le i\le q^m-2\}$。 设$a=g^i\in F_{q^m}^*$，其中$0\le i\le q^m-2$，则$a$的乘法逆元为：$a^{-1}=g^{q^m-1-i}$。

示例 2：F_{3^2}的多项式基表示

取 F_3 上的一个不可约多项式$f(x)=x^2+1$，则 F_{3^2}中的元素是：

(0,0)，(0,1)，(0,2)，(1,0)，(1,1)，(1,2)，(2,0)，(2,1)，(2,2)

加法：(2,1)+(2,0)=(1,1)

乘法：(2,1)·(2,0)=(2,2)

$$(2x+1)\cdot 2x=4x^2+2x$$
$$=x^2+2x$$
$$\equiv 2x+2 \pmod{f(x)}$$

即 $2x+2$ 是$(2x+1)\cdot 2x$ 除以$f(x)$的余式。

乘法单位元是(0,1)，$\alpha=x+1$ 是 $F_{3^2}^*$的一个生成元，则α的方幂为：

$\alpha^0=(0,1)$，$\alpha^1=(1,1)$，$\alpha^2=(2,0)$，$\alpha^3=(2,1)$，$\alpha^4=(0,2)$，$\alpha^5=(2,2)$，

$\alpha^6=(1,0)$，$\alpha^7=(1,2)$，$\alpha^8=(0,1)$。

A.1.3 有限域上的椭圆曲线

A.1.3.1 概述

有限域上椭圆曲线常用的表示形式有两种：仿射坐标表示和射影坐标表示。

A.1.3.2 仿射坐标表示

设 p 是大于 3 的素数，F_{p^m}上椭圆曲线方程在仿射坐标系下可以简化为$y^2=x^3+ax+b$，其中 $a,b\in F_{p^m}$，且使得$4a^3+27b^2\neq 0$。椭圆曲线上的点集记为$E(F_{p^m})=\{(x,y)|\,x,y\in F_{p^m}$，且满足曲线方程$y^2=x^3+ax+b\}\cup\{O\}$，其中 O 是椭圆曲线的无穷远点，又称为零点。

$E(F_{p^m})$ $(m\ge 1)$上的点按照下面的加法运算规则，构成一个交换群：

a) $O+O=O$;

b) $\forall P=(x,y)\in E(F_{p^m})\backslash\{O\}$，$P+O=O+P=P$;

c) $\forall P=(x,y)\in E(F_{p^m})\backslash\{O\}$，$P$ 的逆元素$-P=(x,-y)$，$P+(-P)=O$;

d) 点 $P_1=(x_1,y_1)\in E(F_{p^m})\backslash\{O\}$，$P_2=(x_2,y_2)\in E(F_{p^m})\backslash\{O\}$，$P_3=(x_3,y_3)=P_1+P_2\neq O$，则

$$\begin{cases} x_3=\lambda^2-x_1-x_2, \\ y_3=\lambda(x_1-x_3)-y_1, \end{cases}$$

其中

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & \text{若 } x_1 \neq x_2, \\ \dfrac{3x_1^{\,2} + a}{2y_1}, & \text{若 } x_1 = x_2 \text{且 } P_2 \neq -P_1 \text{。} \end{cases}$$

示例 3：有限域 F_{19} 上一条椭圆曲线

F_{19} 上方程：$y^2=x^3+x+1$，其中 $a=1$，$b=1$。则 F_{19} 上曲线的点为：

(0 , 1)，(0, 18)，(2, 7)，(2, 12)，(5, 6)，(5, 13)，(7, 3)，(7, 16)，(9, 6)，(9, 13)，(10, 2)，(10, 17)，(13, 8)，(13, 11)，(14, 2)，(14, 17)，(15, 3)，(15, 16)，(16, 3)，(16, 16)

则 $E(F_{19})$有 21 个点(包括无穷远点 O)。

a) 取 P_1=(10,2)，P_2=(9,6)，计算 $P_3=P_1+P_2$：

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1} = \frac{6-2}{9-10} = \frac{4}{-1} = -4 \equiv 15 \ (\mathrm{mod}\ 19),$$

$x_3=15^{\,2}-10-9=225-10-9\equiv 16-10-9=-3\equiv 16 \pmod{19}$,

$y_3=15\times(10-16)-2=15\times(-6)-2\equiv 3 \pmod{19}$,

所以 P_3=(16,3)。

b) 取 P_1=(10,2)，计算[2]P_1：

$$\lambda = \frac{3x_1^{\,2} + a}{2y_1} = \frac{3\times 10^2 + 1}{2\times 2} = \frac{3\times 5 + 1}{4} = \frac{16}{4} = 4 \ (\mathrm{mod}\ 19),$$

$x_3=4^2-10-10=-4\equiv 15 \pmod{19}$,

$y_3=4\times(10-15)-2=-22\equiv 16 \pmod{19}$,

所以[2]P_1=(15,16)。

A.1.3.3 射影坐标表示

A.1.3.3.1 标准摄影坐标系

设 p 是大于 3 的素数，F_{p^m} 上椭圆曲线方程在标准射影坐标系下可以简化为 $y^2z=x^3+axz^2+bz^3$，其中 $a,b\in F_{p^m}$，且 $4a^3+27b^2\neq 0$。椭圆曲线上的点集记为 $E(F_{p^m})=\{(x,y,z)|x,y,z\in F_{p^m}$ 且满足曲线方程 $y^2z=x^3+axz^2+bz^3\}$。对于(x_1,y_1,z_1)和(x_2,y_2,z_2)，若存在某个 $u\in F_{p^m}$ 且 $u\neq 0$，使得：$x_1=ux_2$，$y_1=uy_2$，$z_1=uz_2$，则称这两个三元组等价，表示同一个点。

若 $z\neq 0$，记 $X=x/z$，$Y=y/z$，则可从标准射影坐标表示转化为仿射坐标表示：$Y^2=X^3+aX+b$；

若 $z=0$，(0,1,0)对应的仿射坐标系下的点即无穷远点 O。

标准射影坐标系下，$E(F_{p^m})$上点的加法运算定义如下：

a) $O+O=O$；

b) $\forall P=(x,y,z)\in E(F_{p^m})\backslash\{O\}$，$P+O=O+P=P$；

c) $\forall P=(x,y,z)\in E(F_{p^m})\backslash\{O\}$，$P$ 的逆元素$-P=(ux,-uy,uz)$，$u\in F_{p^m}$ 且 $u\neq 0$，$P+(-P)=O$；

d) 设点 $P_1=(x_1,y_1,z_1)\in E(F_{p^m})\backslash\{O\}$，$P_2=(x_2,y_2,z_2)\in E(F_{p^m})\backslash\{O\}$，$P_3=P_1+P_2=(x_3,y_3,z_3)\neq O$，

若 $P_1\neq P_2$，则：

$$\lambda_1=x_1z_2,\quad \lambda_2=x_2z_1,\quad \lambda_3=\lambda_1-\lambda_2,\quad \lambda_4=y_1z_2,\quad \lambda_5=y_2z_1,\quad \lambda_6=\lambda_4-\lambda_5,\quad \lambda_7=\lambda_1+\lambda_2,\quad \lambda_8=z_1z_2,$$

$$\lambda_9=\lambda_3^2,\quad \lambda_{10}=\lambda_3\lambda_9,\quad \lambda_{11}=\lambda_8\lambda_6^2-\lambda_7\lambda_9,\quad x_3=\lambda_3\lambda_{11},\quad y_3=\lambda_6(\lambda_9\lambda_1-\lambda_{11})-\lambda_4\lambda_{10},\quad z_3=\lambda_{10}\lambda_8\ ;$$

若 $P_1=P_2$，则：

$$\lambda_1=3x_1^2+az_1^2,\quad \lambda_2=2y_1z_1,\quad \lambda_3=y_1^2,\quad \lambda_4=\lambda_3x_1z_1,\quad \lambda_5=\lambda_2^2,\quad \lambda_6=\lambda_1^2-8\lambda_4,$$

$x_3=\lambda_2\lambda_6$，$y_3=\lambda_1(4\lambda_4-\lambda_6)-2\lambda_5\lambda_3$，$z_3=\lambda_2\lambda_5$。

A.1.3.3.2 Jacobian加重射影坐标系

设 p 是大于 3 的素数，F_{p^m} 上椭圆曲线方程在 Jacobian 加重射影坐标系下可以简化为 $y^2=x^3+axz^4+bz^6$。其中 $a,b\in F_{p^m}$，且 $4a^3+27b^2\neq 0$。椭圆曲线上的点集记为 $E(F_{p^m})=\{(x,y,z)|x,y,z\in F_{p^m}$ 且满足曲线方程 $y^2=x^3+axz^4+bz^6\}$。对于(x_1,y_1,z_1)和(x_2,y_2,z_2)，若存在某个 $u\in F_{p^m}$ 且 $u\neq 0$，使得：$x_1=u^2x_2$，$y_1=u^3y_2$，$z_1=uz_2$，则称这两个三元组等价，表示同一个点。

若 $z\neq 0$,记 $X=x/z^2$,$Y=y/z^3$,则可从 Jacobian 加重射影坐标表示转化为仿射坐标表示：$Y^2=X^3+aX+b$；

若 $z=0$，(1,1,0)对应的仿射坐标系下的点即无穷远点 O。

Jacobian 加重射影坐标系下，$E(F_{p^m})$上点的加法运算定义如下：

a) $O+O=O$；

b) $\forall P=(x,y,z)\in E(F_{p^m})\backslash\{O\}$，$P+O=O+P=P$；

c) $\forall P=(x,y,z)\in E(F_{p^m})\backslash\{O\}$，$P$ 的逆元素$-P=(u^2x,-u^3y,uz)$，$u\in F_{p^m}$ 且 $u\neq 0$，$P+(-P)=O$；

d) 设点 $P_1=(x_1,y_1,z_1)\in E(F_{p^m})\backslash\{O\}$，$P_2=(x_2,y_2,z_2)\in E(F_{p^m})\backslash\{O\}$，$P_3=P_1+P_2=(x_3,y_3,z_3)\neq O$，

若 $P_1\neq P_2$，则：

$$\lambda_1=x_1z_2^2,\quad \lambda_2=x_2z_1^2,\quad \lambda_3=\lambda_1-\lambda_2,\quad \lambda_4=y_1z_2^3,\quad \lambda_5=y_2z_1^3,\quad \lambda_6=\lambda_4-\lambda_5,\quad \lambda_7=\lambda_1+\lambda_2,$$

$$\lambda_8=\lambda_4+\lambda_5,\ \lambda_9=\lambda_7\lambda_3^2,\ x_3=\lambda_6^2-\lambda_9,\ \lambda_{10}=\lambda_9^2-2x_3,\quad y_3=(\lambda_{10}\lambda_6-\lambda_8\lambda_3^3)/2,\ z_3=z_1z_2\lambda_3;$$

若 $P_1=P_2$，则：

$$\lambda_1=3x_1^2+az_1^4,\ \lambda_2=4x_1y_1^2,\ \lambda_3=8y_1^4,\ x_3=\lambda_1^2-2\lambda_2,\ y_3=\lambda_1(\lambda_2-x_3)-\lambda_3,\ z_3=2y_1z_1。$$

A.1.4 有限域上椭圆曲线的阶

有限域 F_{q^m} 上一条椭圆曲线的阶是指点集 $E(F_{q^m})$中元素的个数，记为$\#E(F_{q^m})$。由 Hasse 定理知：$q^m+1-2\,q^{m/2}\leq\#E(F_{qm})\leq q^m+1+2q^{m/2}$，即$\#E(F_{q^m})=q^m+1-t$，其中 t 称为 Frobenius 迹且$|t|\leq 2q^{m/2}$。

若 F_{q^m} 的特征整除 Frobenius 迹 t，则称此曲线为超奇异的，否则为非超奇异的。

设$E(F_{q^m})$是F_{q^m}上的椭圆曲线，r是与q^m互素的整数，则$E(F_{q^m})$的r阶扭子群$E(F_{q^m})[r]=\{P\in E(F_{q^m})\mid [r]P=O\}$，$E(F_{q^m})[r]$中的点称为$r$-扭点。

A.2 椭圆曲线多倍点运算

椭圆曲线上同一个点的重复相加称为该点的多倍点运算。设 u 是一个正整数，P 是椭圆曲线上的点，其 u 倍点 $Q=[u]P=\underbrace{P+P+\cdots+P}_{u个}$。

多倍点运算可以扩展到 0 倍点运算和负数倍点运算：$[0]P=O$，$[-u]P=[u](-P)$。

椭圆曲线多倍点运算的实现有多种方法，这里只介绍最基本的三种方法，以下都假设$1\leqslant u<N$。

算法一：二进制展开法

输入：点P，l 比特的整数$u=\sum_{j=0}^{l-1}u_j2^j$，$u_j\in\{0,1\}$。

输出：$Q=[u]P$。

a) 置$Q=O$；

b) 对j 从$l-1$降至0执行：

b.1) $Q=[2]Q$；

b.2) 若u_j=1 则$Q = Q + P$；

c) 输出Q。

算法二：加减法

输入：点P，l 比特的整数$u=\sum_{j=0}^{l-1}u_j 2^j$，$u_j \in \{0,1\}$。

输出：Q=[u]P。

a) 设3u的二进制表示是$h_r h_{r-1}\cdots h_1 h_0$，其中最高位$h_r$为1，显然 $r=l$ 或 l+1；

b) 设u的二进制表示是$u_r u_{r-1}\cdots u_1 u_0$；

c) 置Q =P；

d) 对i从r −1降至1执行：

d.1) Q =[2]Q；

d.2) 若h_i = 1，且u_i = 0，则 $Q = Q + P$；

d.3) 若h_i = 0，且u_i = 1，则 $Q = Q – P$；

e) 输出Q。

注：减去点(x, y)，只要加上$(x, –y)$。有多种不同的变种可以加速这一运算。

算法三：滑动窗法

输入：点P，l 比特的整数$u=\sum_{j=0}^{l-1}u_j 2^j$，$u_j \in \{0,1\}$。

输出：Q=[u]P。

设窗口长度 r >1。

预计算

a) P_1=P，P_2 = [2]P；

b) i从 1 到 2^{r-1}−1 计算 P_{2i+1}=P_{2i-1}+P_2；

c) 置 j = l−1，Q = O。

主循环

d) 当 $j \geq 0$ 执行：

d.1) 若u_j = 0，则Q = [2]Q，j = j −1；

d.2) 否则

d.2.1) 令 t 是使$j – t + 1 \leq r$且u_t = 1的最小整数；

d.2.2) $h_j=\sum_{i=0}^{j-t}u_{t+i}2^i$；

d.2.3) Q=[2^{j-t+1}]Q+P_{h_j}；

d.2.4) 置 j =t−1；

e) 输出Q。

A.3 离散对数问题

A.3.1 求解有限域上离散对数问题的方法

有限域 F_q 的全体非零元素构成一个乘法循环群，记为 F_q^*。F_q^*中存在一个元素 g，g 称为生成元，

使得 $F_q^*=\{g^i \mid 0\le i\le q-2\}$。$a\in F_q$ 的阶是满足 $a^t=1$ 的最小正整数 t。循环群 F_q^*的阶为 $q-1$，因此 $t \mid q-1$。

设乘法循环群 F_q^*的生成元为 g，$y\in F_q^*$，有限域上离散对数问题是指确定整数 $x\in[0, q-2]$，使得 $y=g^x \bmod q$ 成立。

有限域上离散对数问题现有攻击方法有：

a) Pohlig-Hellman方法：设l是$q-1$的最大素因子，则时间复杂度为$O(l^{1/2})$；

b) BSGS方法：时间复杂度与空间复杂度均为$(\pi q/2)^{1/2}$；

c) Pollard方法：时间复杂度为$(\pi q/2)^{1/2}$；

d) 并行Pollard方法：设s为并行处理器个数，时间复杂度为$(\pi q/2)^{1/2}/s$；

e) 线性筛法(对素域F_q)：时间复杂度为 $\exp((1+o(1))(\log q)^{1/2}(\log\log q)^{1/2})$；

f) Gauss整数法(对素域F_q)：时间复杂度为 $\exp((1+o(1))(\log q)^{1/2}(\log\log q)^{1/2})$；

g) 剩余列举筛法(对素域F_q)：时间复杂度为 $\exp((1+o(1))(\log q)^{1/2}(\log\log q)^{1/2})$；

h) 数域筛法(对素域F_q)：时间复杂度为 $\exp(((64/9)^{1/3}+o(1))(\log q(\log\log q)^2)^{1/3})$；

i) 函数域筛法（对小特征域）：时间复杂度为 $\exp(c(\log q(\log\log q)^2)^{1/4+o(1)})$ 和拟多项式时间。

从以上列举的求解离散对数问题的方法及其时间复杂度可知：对于一般的大特征域上的离散对数问题，存在亚指数级计算复杂度的攻击方法，对小特征域上的离散对数问题，目前已经有拟多项式时间的攻击方法。

A.3.2 求解椭圆曲线离散对数问题的方法

已知椭圆曲线$E(F_q)$，阶为n的点$P\in E(F_q)$及$Q\in<P>$，椭圆曲线离散对数问题是指确定整数$u\in[0, n-1]$，使得$Q=[u]P$成立。

*ECDLP*现有攻击方法有：

a) Pohlig-Hellman方法：设l是n的最大素因子，则时间复杂度为$O(l^{1/2})$；

b) BSGS方法：时间复杂度与空间复杂度均为$(\pi n/2)^{1/2}$；

c) Pollard方法：时间复杂度为$(\pi n/2)^{1/2}$；

d) 并行Pollard方法：设r为并行处理器个数，时间复杂度为$(\pi n/2)^{1/2}/r$；

e) MOV−方法： 把超奇异椭圆曲线及具有相似性质的曲线的*ECDLP*降到F_q的小扩域上的离散对数问题(亚指数级计算复杂度算法)；

f) Anomalous方法： 对Anomalous曲线(#$E(F_q)=q$的曲线)的有效攻击方法(多项式级计算复杂度算法)；

g) GHS−方法：利用Weil下降技术求解扩张次数为合数的二元扩域上椭圆曲线离散对数问题，将*ECDLP*转化为超椭圆曲线离散对数问题，而求解高亏格的超椭圆曲线离散对数存在亚指数级计算复杂度算法。

h) DGS−点分解方法：对低次扩域上的椭圆曲线离散对数利用的指标计算方法，在某些特殊情况下，其求解复杂度低于平方根时间复杂度。

从上述对椭圆曲线离散对数问题解法的描述与分析可知：对于一般曲线的离散对数问题，目前的求解方法都为指数级计算复杂度，未发现亚指数级计算复杂度的一般攻击方法；而对于某些特殊曲线的离散对数问题，存在多项式级或者亚指数级计算复杂度算法。

A.4 点的压缩

A.4.1 概述

对于椭圆曲线$E(F_q)$上的任意非无穷远点$P=(x_P, y_P)$，该点能由坐标x_P及由x_P和y_P导出的一个特定比特简洁地表示，称为点的压缩表示。

A.4.2 F_p上椭圆曲线点的压缩与解压缩方法

设$P=(x_P, y_P)$是定义在F_p上椭圆曲线E：$y^2=x^3+ax+b$上的一个点，$\tilde{y}_P$为y_P的最右边的一个比特，则点P可由x_P和比特$\tilde{y}_P$表示。

由x_P和$\tilde{y}_P$恢复y_P的方法如下：

a) 在F_p上计算域元素$\alpha=x_P^3+ax_P+b$；

b) 计算α在F_p上的平方根β(参见附录C.1.4)，若输出是“不存在平方根”，则报错；

c) 若β的最右边比特等于$\tilde{y}_P$，则置$y_P=\beta$；否则置$y_P=p-\beta$。

A.4.3 F_{q^m} (q为奇素数，$m\geq 2$)上椭圆曲线点的压缩与解压缩方法

设$P=(x_P, y_P)$是定义在F_{q^m}上椭圆曲线E：$y^2=x^3+ax+b$上的一个点，则y_P可表示为$(y_{m-1}, y_{m-2}, \ldots, y_1, y_0)$，$\tilde{y}_P$为$y_0$的最右边的一个比特，则点$P$可由$x_P$和比特$\tilde{y}_P$表示。

由x_P和$\tilde{y}_P$恢复y_P的方法如下：

a) 在F_{q^m}上计算域元素$\alpha=x_P^3+ax_P+b$；

b) 计算α在F_{q^m}上的平方根β(参见附录C.1.4)，若输出是“不存在平方根”，则报错；

若β的表示$(\beta_{m-1}, \beta_{m-2}, \ldots, \beta_1, \beta_0)$中$\beta_0$的最右边比特等于$\tilde{y}_P$，则置$y_P=\beta$；否则置$y_P=(\beta'_{m-1}, \beta'_{m-2}, \ldots, \beta'_1, \beta'_0)$，其中$\beta'_i=(q-\beta_i)\in F_q$，$i=0,1,\ldots,m-1$。

附 录 B
椭圆曲线上双线性对的计算

B.1 概述

设有限域 F_q 上椭圆曲线为 $E(F_q)$，若$\#E(F_q)=cf\times r$，r 是素数且 gcd(r, q)=1，cf 为余因子，则使 $r|q^k-1$ 的最小正整数 k 称为椭圆曲线相对于 r 的嵌入次数。若 $\mathbb{G}$ 是 $E(F_q)$的 r 阶子群，则 $\mathbb{G}$ 的嵌入次数也是 k。

设 $\overline{F}_q$ 是有限域F_q的代数闭包，$E[r]$表示 $E(\overline{F}_q)$ 中所有r阶点的集合。

B.2 Miller算法

设F_{q^k}上椭圆曲线$E(F_{q^k})$的方程为$y^2=x^3+ax+b$，定义过$E(F_{q^k})$上点U和V的直线为$g_{U,V}$：$E(F_{q^k})\to F_{q^k}$，若过U, V两点的直线方程为$\lambda x+\delta y+\tau=0$，则令函数$g_{U,V}(Q)=\lambda x_Q+\delta y_Q+\tau$，其中$Q=(x_Q, y_Q)$。当$U=V$时，$g_{U,V}$定义为过点$U$的切线；若$U$和$V$中有一个点为无穷远点$O$，$g_{U,V}$就是过另一个点且垂直于$x$轴的直线。一般用$g_U$作为$g_{U,-U}$的简写。

记 $U=(x_U, y_U)$，$V=(x_V, y_V)$，$Q=(x_Q, y_Q)$，$\lambda_1=(3x_V^2+a)/(2y_V)$，$\lambda_2=(y_U-y_V)/(x_U-x_V)$，则有以下性质：

a) $g_{U,V}(O)=g_{U,O}(Q)=g_{O,V}(Q)=1$；

b) $g_{V,V}(Q)=\lambda_1(x_Q-x_V)-y_Q+y_V$，$Q\neq O$；

c) $g_{U,V}(Q)=\lambda_2(x_Q-x_V)-y_Q+y_V$，$Q\neq O$，$U\neq\pm V$；

d) $g_{V,-V}(Q)=x_Q-x_V$，$Q\neq O$。

Miller 算法是计算双线性对的有效算法。

Miller 算法

输入：曲线 E，E 上两点 P 和 Q，整数 c。

输出：$f_{P,c}(Q)$。

a) 设 c 的二进制表示是 $c_j\ldots c_1c_0$，其最高位 c_j 为 1；

b) 置 $f=1$, $V=P$；

c) 对 i 从 $j-1$ 降至 0，执行：

c.1) 计算 $f=f^2\cdot g_{V,V}(Q)/g_{2V}(Q)$, $V=[2]V$；

c.2) 若 $c_i=1$，令 $f=f\cdot g_{V,P}(Q)/g_{V+P}(Q)$, $V=V+P$。

d) 输出 f。

一般，称$f_{P,c}(Q)$为Miller函数。

B.3 Weil对的计算

设 E 是 F_q 上的椭圆曲线，r 是与 q 互素的正整数，设μ_r是 r 次单位根集合，k 是相对于 r 的嵌入次数，即 $r\mid q^k-1$，则$\mu_r\subset F_{q^k}$。

令 $\mathbb{G}_1=E[r]$，$\mathbb{G}_2=E[r]$，$\mathbb{G}_T=\mu_r$，则 Weil 对是从 $\mathbb{G}_1\times\mathbb{G}_2$ 到 $\mathbb{G}_T$的双线性映射，记为 e_r。

设 $P\in\mathbb{G}_1$，$Q\in\mathbb{G}_2$，若 $P=O$ 或 $Q=O$，则 $e_r(P,Q)=1$；如果 $P\neq O$ 且 $Q\neq O$，随机选取非无穷远点 $T\in\mathbb{G}_1$, $U\in\mathbb{G}_2$，使得 $P+T$ 和 T 均不等于 U 或 $U+Q$，则 Weil 对为：

$$e_r(P,Q)=\frac{f_{P+T,r}(Q+U)f_{T,r}(U)f_{U,r}(P+T)f_{Q+U,r}(T)}{f_{T,r}(Q+U)f_{P+T,r}(U)f_{Q+U,r}(P+T)f_{U,r}(T)}。$$

$f_{P+T,r}(Q+U)$，$f_{T,r}(Q+U)$，$f_{P+T,r}(U)$，$f_{T,r}(U)$，$f_{Q+U,r}(P+T)$，$f_{Q+U,r}(T)$，$f_{U,r}(P+T)$和$f_{U,r}(T)$均可用Miller算法计算。在计算过程中，若出现分母为0的情况，则更换点T或U重新计算。

B.4 Tate对的计算

设 E 是 F_q 上的椭圆曲线，r 是与 q 互素的正整数，k 是相对于 r 的嵌入次数。设 Q 是 $E(F_{q^k})[r]$上的 r 阶点，由 Q 生成的循环群记为<Q>。$(F_{q^k}^*)^r$ 为 $F_{q^k}^*$中每一个元素的 r 次幂构成的集合，$(F_{q^k}^*)^r$ 是 $F_{q^k}^*$的子群，$F_{q^k}^*$关于$(F_{q^k}^*)^r$ 的商群记为 $F_{q^k}^*/(F_{q^k}^*)^r$。

令 $\mathbb{G}_1= E(F_q)[r]$，$\mathbb{G}_2=$ <Q>，$\mathbb{G}_T= F_{q^k}^*/(F_{q^k}^*)^r$，则 Tate 对是从 $\mathbb{G}_1\times\mathbb{G}_2$ 到 $\mathbb{G}_T$ 的双线性映射，记为 t_r。

设 $P\in\mathbb{G}_1$，$Q\in\mathbb{G}_2$，若 $P=O$ 或 $Q=O$ ，则 $t_r=1$；若 $P\neq O$ 且 $Q\neq O$，随机选择非无穷远点 $U\in E(F_{q^k})$，使得 $P\neq Q, P\neq Q+U, U\neq -Q$，则 Tate 对为：

$$t_r(P,Q)=\frac{f_{P,r}(Q+U)}{f_{P,r}(U)}。$$

$f_{P,r}(Q+U)$和$f_{P,r}(U)$可通过 Miller 算法计算。在计算过程中，若出现分母为 0 的情况，则更换点 U 重新计算。

在实际应用中，一般使用约化 Tate 对：

$$t_r(P,Q)=\begin{cases} f_{P,r}(Q)^{(q^k-1)/r}, & Q\neq O, \\ 1, & Q=O。\end{cases}$$

约化 Tate 对比一般 Tate 对的计算量减少了一半。若相对于 r 的嵌入次数 k 是偶数时，约化 Tate 对的计算方法可以进一步优化。算法 1 描述的是一般约化 Tate 对的计算方法，算法 2、3、4 均指 $k=2d$ 的情况。

算法 1

输入：与 q 互素的整数 r，$P\in E(F_q)[r]$，$Q\in E(F_{q^k})[r]$。

输出：$t_r(P, Q)$。

a) 设 r 的二进制表示是 $r_j\ldots r_1r_0$，其最高位 r_j 为 1；

b) 置 $f=1$, $V=P$；

c) 对 $i=j-1$ 降至 0，执行：

c.1) 计算 $f=f^2\cdot g_{V,V}(Q)/g_{2V}(Q)$, $V=[2]V$；

c.2) 若 $r_i=1$，则计算 $f=f\cdot g_{V,P}(Q)/g_{V+P}(Q)$, $V=V+P$；

d) 计算 $f=f^{(q^k-1)/r}$；

e) 输出 f。

算法 2

输入：与 q 互素的整数 r，$P\in E(F_q)[r]$，$Q\in E(F_{q^k})[r]$。

输出：$t_r(P, Q)$。

a) 设 r 的二进制表示是 $r_j\ldots r_1r_0$，其最高位 r_j 为 1；

b) 置 $f=1$, $V=P$；

c) 对 $i=j-1$ 降至 0，执行：

c.1) 计算 $f=f^2\cdot g_{V,V}(Q)/g_{2V}(Q)$, $V=[2]V$；

c.2) 若 $r_i=1$，则计算 $f=f\cdot g_{V,P}(Q)/g_{V+P}(Q)$, $V=V+P$；

d) 计算 $f=f^{q^d-1}$ ；

e) 计算 $f=f^{(q^d+1)/r}$ ；

f) 输出 f。

算法 3

如果将 F_{q^k}($k=2d$)看成 F_{q^d}的二次扩域，则 F_{q^k}上元素可表示成 $w=w_0+iw_1$ 的形式，其中 $w_0, w_1\in F_{q^d}$，则 w 的共轭 $\overline{w}=w_0-iw_1$，此时算法 1 中的求逆运算可用共轭代替。

输入：与 q 互素的整数 r，$P\in E(F_q)[r]$，$Q\in E(F_{q^k})[r]$。

输出：$t_r(P, Q)$。

a) 设 r 的二进制表示是 $r_j\ldots r_1r_0$，其最高位 r_j 为 1；

b) 置 $f=1$, $V=P$；

c) 对 i 从 $j-1$ 降至 0，执行：

c.1) 计算 $f = f^2 \cdot g_{V,V}(Q) \cdot \overline{g}_{2V}(Q),\ V = [2]V$；

c.2) 若 r_i=1，令 $f = f \cdot g_{V,P}(Q) \cdot \overline{g}_{V+P}(Q),\ V = V + P$；

d) 计算 $f = f^{q^d-1}$；

e) 计算 $f = f^{(q^d+1)/r}$；

f) 输出 f。

算法 4

当 q 为大于 3 的素数时，点 $Q \in E'$， E'是 E 的扭曲线，此时算法可进一步优化。

输入： $P \in E(F_q)[r]$，$Q \in E'(F_{q^d})[r]$，整数 r。

输出： $t_r(P, Q)$。

a) 设 r 的二进制表示是 $r_j \ldots r_1 r_0$，其最高位 r_j 为 1；

b) 置 f=1, V=P；

c) 对 i 从 j–1 降至 0，执行：

c.1) 计算 $f = f^2 \cdot g_{V,V}(Q),\ V = [2]V$；

c.2) 若 r_i=1，则计算 $f = f \cdot g_{V,P}(Q),\ V = V + P$；

d) 计算 $f = f^{q^d-1}$；

e) 计算 $f = f^{(q^d+1)/r}$；

f) 输出 f。

B.5 Ate对的计算

设 π_q 为 Frobenius 自同态，即 π_q：$E \to E$，$(x,y) \mapsto (x^q, y^q)$； $[q]$为映射：$E \to E$，$Q \mapsto [q]Q$； $[1]$为单位映射； π_q 的对偶为 π_q'，满足 $\pi_q \cdot \pi_q' = [q]$；Ker( )表示映射的核；设椭圆曲线 $E(F_q)$的 Frobenius 迹为 t,令 $T = t - 1$。

下面给出不同结构下的Ate对的计算方法。

B.5.1 定义在$\mathbb{G}_2 \times \mathbb{G}_1$上Ate对的计算

设 $\boldsymbol{G}_1 = E[r] \cap \mathrm{Ker}(\pi_q - [1])$，$\mathbb{G}_2 = E[r] \cap \mathrm{Ker}(\pi_q - [q])$，$P \in \boldsymbol{G}_1$，$Q \in \boldsymbol{G}_2$。定义 $\boldsymbol{G}_2 \times \boldsymbol{G}_1$ 上 Ate 对：

$$Ate: \boldsymbol{G}_2 \times \boldsymbol{G}_1 \to F_{q^k}^* / (F_{q^k}^*)^r$$

$$(Q, P) \mapsto f_{Q,T}(P)^{(q^k-1)/r}。$$

下面给出 $\boldsymbol{G}_2 \times \boldsymbol{G}_1$ 上 Ate 对的计算方法：

输入： $\boldsymbol{G}_1 = E[r] \cap \mathrm{Ker}(\pi_q - [1])$，$\boldsymbol{G}_2 = E[r] \cap \mathrm{Ker}(\pi_q - [q])$，$P \in \boldsymbol{G}_1$，$Q \in \boldsymbol{G}_2$，整数 $T = t$-1。

输出： $Ate(Q, P)$。

a) 设 T 的二进制表示是 $t_j \ldots t_1 t_0$，其最高位 t_j 为 1；

b) 置 f=1, V=Q；

c) 对 i 从 j–1 降至 0，执行：

c.1) 计算 $f = f^2 \cdot g_{V,V}(P),\ V = [2]V$；

c.2) 若 t_i=1，计算 $f = f \cdot g_{V,Q}(P) / g_{V+Q}(P),\ V = V + Q$；

d) 计算 $f = f^{(q^k-1)/r}$；

e) 输出 f。

B.5.2 定义在$\mathbb{G}_1 \times \mathbb{G}_2$上Ate对的计算

对于超奇异椭圆曲线来说，以上Ate对的定义与技术可以直接应用；而对于常曲线来说，需要把$\mathbb{G}_2$转换到扭曲线上才可以定义Ate对。

B.5.2.1 超奇异椭圆曲线上Ate对

设 E 为定义在 F_q 上的超奇异椭圆曲线，$\boldsymbol{G}_1 = E[r] \cap \mathrm{Ker}(\pi_q' - [q])$，$\boldsymbol{G}_2 = E[r] \cap \mathrm{Ker}(\pi_q' - [1])$，$\boldsymbol{G}_T = F_{q^k}^* / (F_{q^k}^*)^r$，

$P\in\boldsymbol{G}_1$，$Q\in\boldsymbol{G}_2$。定义 $\boldsymbol{G}_1\times\boldsymbol{G}_2$ 上的 Ate 对：

$$Ate：\boldsymbol{G}_1\times\boldsymbol{G}_2\to F_{qk}^{*}/(F_{qk}^{*})^{r}$$
$$(P,Q)\mapsto f_{P,T}(Q)^{(q^k-1)/r}。$$

下面给出 $\boldsymbol{G}_1\times\boldsymbol{G}_2$ 上 Ate 对的计算方法：

输入： $\boldsymbol{G}_1=E[r]\cap\mathrm{Ker}(\pi_q'-[q])$，$\boldsymbol{G}_2=E[r]\cap\mathrm{Ker}(\pi_q'-[1])$，$P\in\boldsymbol{G}_1$，$Q\in\boldsymbol{G}_2$，整数 $T=t-1$。

输出： $Ate(P,Q)$。

a) 设 T 的二进制表示是 $t_j\ldots t_1t_0$，其最高位 t_j 为 1；

b) 置 $f=1$, $V=P$；

c) 对 i 从 $j-1$ 降至 0，执行：

c.1) 计算 $f=f^2\cdot g_{V,V}(Q)$, $V=[2]V$；

c.2) 若 $t_i=1$，计算 $f=f\cdot g_{V,P}(Q)/g_{V+P}(Q)$, $V=V+P$；

d) 计算 $f=f^{(q^k-1)/r}$；

e) 输出f。

B.5.2.2 常曲线上的Ate对

对于常曲线来说，存在一个整数 e，使得$(\pi_q')^e$成为 $\mathbb{G}_1$ 上的自同构，这样可以用扭曲线理论在 $Ate(P,Q)$ 和 $f_{P,T^e}(Q)$ 之间建立起联系，其中 $T=t-1$，t 为迹。

设 E 是定义在 F_q 上的椭圆曲线，E'为 E 的 d 次扭曲线。k 为嵌入次数，$m=\gcd(k,d)$，$e=k/m$，ζ_m 是 m 次本原单位根，当 $p\geq 5$ 时，d 的取值有三种情况：

a) $d=6$，$\beta=\zeta_m^{-6}$，E': $y^2=x^3+\beta b$，$\phi_6:E'\to E:(x,y)\mapsto(\beta^{-1/3}x,\beta^{-1/2}y)$，$\mathbb{G}_1=E[r]\cap\mathrm{Ker}(\pi_q-[1])$，$\mathbb{G}_2=E'[r]\cap\mathrm{Ker}([\beta^{-1/6}]\pi_q^e-[1])$；

b) $d=4$，$\beta=\zeta_m^{-4}$，E': $y^2=x^3+\beta ax$，$\phi_4:E'\to E:(x,y)\mapsto(\beta^{-1/2}x,\beta^{-3/4}y)$，$\mathbb{G}_1=E[r]\cap\mathrm{Ker}(\pi_q-[1])$，$\mathbb{G}_2=E'[r]\cap\mathrm{Ker}([\beta^{-1/4}]\pi_q^e-[1])$；

c) $d=2$，$\beta=\zeta_m^{-2}$，E': $y^2=x^3+\beta^2ax+\beta^3b$，$\phi_2:E'\to E:(x,y)\mapsto(\beta^{-1}x,\beta^{-3/2}y)$，$\mathbb{G}_1=E[r]\cap\mathrm{Ker}(\pi_q-[1])$，$\mathbb{G}_2=E'[r]\cap\mathrm{Ker}([\beta^{-1/2}]\pi_q^e-[1])$。

设 $P\in\mathbb{G}_1$，$Q\in\mathbb{G}_2$。定义 $\mathbb{G}_1\times\mathbb{G}_2$ 上 Ate 对：

$$Ate：\mathbb{G}_1\times\mathbb{G}_2\to F_{qk}^{*}/(F_{qk}^{*})^{r}$$
$$(P,Q)\mapsto f_{P,T^e}(Q)^{(q^k-1)/r}。$$

下面给出具体算法描述：

输入： $\mathbb{G}_1$，$\mathbb{G}_2$，$P\in\mathbb{G}_1$，$Q\in\mathbb{G}_2$，整数 $T=t-1$。

输出： $Ate(P,Q)$。

a) 计算 $u=T^e$；

b) 设 u 的二进制表示是 $t_j\ldots t_1t_0$，其最高位 t_j 为 1；

c) 置 $f=1$, $V=P$；

d) 对 i 从 $j-1$ 降至 0，执行：

d.1) 计算 $f=f^2\cdot g_{V,V}(Q)$, $V=[2]V$；

d.2) 若 $t_i=1$，计算 $f=f\cdot g_{V,P}(Q)/g_{V+P}(Q)$, $V=V+P$；

e) 计算 $f=f^{(q^k-1)/r}$；

f) 输出f。

如果定义在$\mathbb{G}_1\times\mathbb{G}_2$上的Ate对所基于的椭圆曲线是超奇异的，则容易看出它比Tate对有更高的效率。但对于常曲线来说，只有当$|T^k|\le r$时它的运算效率才会比Tate对高，所以只有在t值较小时才推荐使用Ate对。

B.6 R-ate 对的计算

B.6.1 R-ate 对的定义

R-ate 对中的“R”可视为两个对的比值，也可以看成是 Tate 对的某固定幂次。

令 $A, B, a, b \in Z$， $A = aB + b$. Miller 函数$f_{Q,A}(P)$有如下性质：

$f_{Q,A}(P)=f_{Q,aB+b}(P)=f_{Q,aB}(P)\cdot f_{Q,b}(P)\cdot g_{[aB]Q,[b]Q}(P)/g_{[A]Q}(P)$

$$=f_{Q,B}^{a}(P)\cdot f_{[B]Q,a}(P)\cdot f_{Q,b}(P)\cdot\frac{g_{[aB]Q,[b]Q}(P)}{g_{[A]Q}(P)}$$

定义 R-ate 对为

$$R_{A,B}(Q,P)=\left(f_{[B]Q,a}(P)\cdot f_{Q,b}(P)\cdot\frac{g_{[aB]Q,[b]Q}(P)}{g_{[A]Q}(P)}\right)^{(q^k-1)/n}$$

$$=\left(\frac{f_{Q,\mathrm{A}}(P)}{f_{Q,B}^{a}(P)}\right)^{(q^k-1)/n}$$

如果$f_{Q,A}(P)$和$f_{Q,B}(P)$是非退化对的 Miller 函数，则 $R_{A,B}(Q,P)$也是非退化对。

令 $L_1, L_2, M_1, M_2\in$ Z，使得 $e_n^{L_1}(Q,P)=\left(f_{Q,A}(P)\right)^{M_1\cdot(q^k-1)/n}$

$$e_n^{L_2}(Q,P)=\left(f_{Q,B}(P)\right)^{M_2\cdot(q^k-1)/n}$$

令 $M=lcm(M_1,M_2)$ ， $m=(M/M_1)\cdot L_1-a\cdot(M/M_2)\cdot L_2$.

为了非退化，n 不能整除 m. 我们有

$$e_n^m(Q,P)=e_n^{\frac{M}{M_1}L_1-a\frac{M}{M_2}L_2}(Q,P)=\frac{e_n(Q,P)^{L_1\frac{M}{M_1}}}{e_n(Q,P)^{aL_2\frac{M}{M_2}}}=\left(\frac{f_{Q,A}(P)}{f_{Q,B}(P)^a}\right)^{M\cdot(q^k-1)/n}$$

易见$e_n^m(Q,P)=R_{A,B}(Q,P)^M$.

一般来说，不是任意整数对(A,B)都能给出非退化对，(A,B)有四种选择：

1. $(A;B)=(q^i;n)$
2. $(A;B)=(q;T_1)$
3. $(A;B)=(T_i;T_j)$
4. $(A;B)=(n;T_i)$

其中 $T_i\equiv q^i(\mathrm{mod}\ n)$ ， $i\in$ Z ， $0<i<k$.

情形 1： $(A;B)=(q^i;n)$，由于 $A=aB+b$, 即 $q^i=an+b$. 因此 $b\equiv q^i(\mathrm{mod}\ n)$，

又$\left(\frac{f_{Q,q^i}(P)}{f_{Q,n}^{a}(P)}\right)^{(q^k-1)/n}=R_{A,B}(Q,P)=\left(f_{[n]Q,a}(P)f_{Q,b}(P)\frac{g_{[an]Q,[b]Q}(P)}{g_{[q^i]Q}(P)}\right)^{(q^k-1)/n}$

因为 $b \equiv q^i \pmod n$, 所以 $g_{[an]Q,[b]Q}(P) = g_{[q^i]Q}(P)$. 更进一步， $f_{[n]Q,a}(P) = 1$. 因此

$$R_{A,B}(Q,P) = f_{Q,q^i}(P)^{(q^k-1)/n} \tag{1}$$

情形 2：$(A;B) = (q;T_1)$ ，即 $q = aT_1 + b$， 则：

$$\left(\frac{f_{Q,q}(P)}{f^a_{Q,T_1}(P)}\right)^{(q^k-1)/n} = R_{A,B}(Q,P) = \left(f_{[T_1]Q,a}(P) f_{Q,b}(P) \frac{g_{[aT_1]Q,[b]Q}(P)}{g_{[q]Q}(P)}\right)^{(q^k-1)/n}$$

由于 $f_{[T_1]Q,a}(P) = f^q_{Q,a}(P)$，因此

$$R_{A,B}(Q,P) = \left(f^q_{Q,a}(P) f_{Q,b}(P) \frac{g_{[aT_1]Q,[b]Q}(P)}{g_{[q]Q}(P)}\right)^{(q^k-1)/n} \tag{2}$$

情形 3：$(A;B) = (T_i;T_j)$， 即 $T_i = aT_j + b$. 有：

$$\left(\frac{f_{Q,T_i}(P)}{f^a_{Q,T_j}(P)}\right)^{(q^k-1)/n} = R_{A,B}(Q,P) = \left(f_{[T_j]Q,a}(P) f_{Q,b}(P) \frac{g_{[aT_j]Q,[b]Q}(P)}{g_{[q^i]Q}(P)}\right)^{(q^k-1)/n}$$

同样，因为 $f_{[T_j]Q,a}(P) = f^{q_j}_{Q,a}(P)$，因此：

$$R_{A,B}(Q,P) = \left(f^{q_j}_{Q,a}(P) f_{Q,b}(P) \frac{g_{[aT_j]Q,[b]Q}(P)}{g_{[q^i]Q}(P)}\right)^{(q^k-1)/n} \tag{3}$$

情形 4： $(A;B) = (n;T_i)$，即 $n = aT_i + b$. 因此：

$$\left(\frac{f_{Q,n}(P)}{f^a_{Q,T_i}(P)}\right)^{(q^k-1)/n} = R_{A,B}(Q,P) = \left(f_{[T_i]Q,a}(P) f_{Q,b}(P) \frac{g_{[aT_i]Q,[b]Q}(P)}{g_{[n]Q}(P)}\right)^{(q^k-1)/n}$$

同样，由 $f_{[T_i]Q,a}(P) = f^{q_i}_{Q,a}(P)$ 得

$$R_{A,B}(Q,P) = \left(f^{q_i}_{Q,a}(P) f_{Q,b}(P) \frac{g_{[aT_i]Q,[b]Q}(P)}{g_{[n]Q}(P)}\right)^{(q^k-1)/n} \tag{4}$$

情形 1 的 R-ate 对也称 Ate_i 对。情形 2、3、4 的对计算需要两个长度为 $\log a$ 和 $\log b$ 的 Miller 循环。情形 2 和情形 4 只能改变一个参数 i 来获得有效对，情形 3 可以改变两个参数。因此，一般都选择情形 3 的 R-ate 对，这时$(A;B) = (T_i;T_j)$。

为了降低Miller循环次数，可以尝试不同的i和j，使整数a和b足够小，从而使Miller循环次数减至$\log(r^{1/\Phi(k)})$。

B. 6. 2 BN曲线上R-ate对的计算

Barreto 和 Naehrig 提出了一种构造素域 F_q 上适合对的常曲线的方法，通过此方法构造的曲线称为 BN 曲线。BN 曲线方程为 E：$y^2 = x^3 + b$, 其中 $b \neq 0$. 嵌入次数 k=12，曲线阶 r 也是素数。

基域特征 q，曲线阶 r，Frobenius 映射的迹 tr 可通过参数 t 来确定：

$q(t) = 36t^4 + 36t^3 + 24t^2 + 6t + 1$

$r(t) = 36t^4 + 36t^3 + 18t^2 + 6t + 1$

$tr(t) = 6t^2 + 1$

其中 $t\in Z$ 是任意使得 $q=q(t)$ 和 $r=r(t)$ 均为素数的整数，为了达到一定的安全级别，t 必须足够大，至少达到 63 比特。

BN 曲线存在定义在 F_{q^2} 上的 6 次扭曲线 E'： $y^2=x^3+\beta b$，其中 $\beta\in F_{q^2}$，并且在 F_{q^2} 上既不是二次元也不是三次元，选择 β 使得 $r|\#E'(F_{q^2})$，G_2 中点可用扭曲线 E' 上的点来表示，$\phi_6:E'\to E:(x,y)\mapsto(\beta^{-1/3}x,\beta^{-1/2}y)$。因此对的计算限制在 $E(F_q)$上点 P 和 $E'(F_{q^2})$上点 Q'。

π_q 为 Frobenius 自同态，π_q：$E\to E$，$\pi_q(x,y)=(x^q,y^q)$。

$\pi_{q^2}:E\to E,\quad \pi_{q^2}(x,y)=(x^{q^2},y^{q^2})$。

R-ate 对的计算：

输入: $P\in E(F_q)[r], Q\in E'(F_{q^2})[r]$，$a=6t+2$ 。

输出： $R_a(Q,P)$ 。

a) 设 $a=\sum_{i=0}^{L-1}a_i2^i$， $a_{L-1}=1$；

b) 置 $T=Q$， $f=1$；

c) 对 i 从 $L-2$ 降至 0，执行：

c.1) 计算 $f=f^2\cdot g_{T,T}(P)$， $T=[2]T$；

c.2) 若 $a_i=1$，计算 $f=f\cdot g_{T,Q}(P)$，$T=T+Q$；

d) 计算 $Q_1=\pi_q(Q)$，$Q_2=\pi_{q^2}(Q)$；

e) 计算 $f=f\cdot g_{T,Q_1}(P)$，$T=T+Q_1$ ；

f) 计算 $f=f\cdot g_{T,-Q_2}(P)$，$T=T-Q_2$ ；

g) 计算 $f=f^{(q^{12}-1)/r}$；

h) 输出 f。

B.7 适合对的椭圆曲线

对于超奇异曲线，双线性对的构造相对容易，但对于随机生成的曲线，构造可计算的双线性对比较困难，因此采用常曲线时，需要构造适合对的曲线。

假设 E 是定义在 F_q 上的椭圆曲线，如果以下三个条件成立，则称 E 是适合对的曲线：

a) $\#E(F_q)$有一个不小于 $\sqrt{q}$ 的素因子 r ；

b) E 相对于 r 的嵌入次数小于 $\log_2(r)/8$ ；

c) $r\pm1$ 的最大素因子的规模与 r 相当。

构造适合对的椭圆曲线的步骤如下：

步骤 1：选定 k，计算整数 t、r、q，使得存在一条椭圆曲线 $E(F_q)$，其迹为 t，具有一个素数阶 r 的子群且嵌入次数为 k；

步骤 2：利用复乘方法在 F_q 上计算该曲线的方程参数。

附 录 C
数论算法

C.1 有限域中的运算

C.1.1 有限域中的指数运算

设a是正整数，g是域F_q上的元素，指数运算是计算g^a的运算过程。通过以下的二进制方法可以有效地执行指数运算。

输入：正整数a，域F_q，域元素g。

输出：g^a。

a) 置$e=a \bmod (q-1)$，若$e=0$，则输出1；

b) 设e的二进制表示是$e_r e_{r-1}\ldots e_1 e_0$，其最高位$e_r$为1；

c) 置$x=g$；

d) 对 i 从$r-1$降至 0 执行：

d.1) 置$x=x^2$；

d.2) 若$e_i=1$，则置$x=g\cdot x$；

e) 输出x。

C.1.2 有限域中的逆运算

设g是域F_q上的非零元素，则逆元素g^{-1}是使得$g\cdot c=1$成立的域元素c。由于$c=g^{q-2}$，因此求逆可通过指数运算实现。若q是素数，g是满足$1\le g\le q-1$的整数，则g^{-1}是整数c，$1\le c\le q-1$，且$g\cdot c\equiv 1 \pmod q$。

输入：域F_q，F_q中的非零元素g。

输出：逆元素g^{-1}。

a) 计算$c=g^{q-2}$(参见附录C.1.1)；

b) 输出c。

更为有效的方法是扩展的欧几里德(Euclid)算法。

C.1.3 Lucas序列的生成

令X和Y是非零整数，X和Y的Lucas序列U_k, V_k的定义如下：

$U_0=0$, $U_1=1$, 当$k\ge 2$时，$U_k=X\cdot U_{k-1}-Y\cdot U_{k-2}$；

$V_0=2$, $V_1=X$, 当$k\ge 2$时，$V_k=X\cdot V_{k-1}-Y\cdot V_{k-2}$。

上述递归式适于计算k值较小的U_k和V_k。对大整数k，下面的算法可有效地计算$U_k \bmod q$和$V_k \bmod q$。

输入：奇素数q，整数X和Y，正整数k。

输出：$U_k \bmod q$和$V_k \bmod q$。

a) 置$\Delta=X^2-4Y$；

b) 设k的二进制表示是$k=k_r k_{r-1}\ldots k_1 k_0$，其中最高位$k_r$为1；

c) 置$U=1$，$V=X$；

d) 对i从$r-1$降至0执行：

d.1) 置$(U,V)=(\ (U\cdot V)\bmod q,\ ((V^2+\Delta\cdot U^2)/2) \bmod q\ )$；

d.2) 若$k_i=1$，则置$(U,V)=(\ ((X\cdot U+V)/2)\bmod q,\ ((X\cdot V+\Delta\cdot U)/2)\bmod q\ )$；

e) 输出U和V。

C.1.4 平方根的求解

C.1.4.1 F_q上平方根的求解

设q是奇素数，g是满足$0 \le g < q$的整数，g的平方根(mod q)是整数y，即$y^2 \bmod q = g$，$0 \le y < p$。

若$g=0$，则只有一个平方根，即$y=0$；若$g \neq 0$，则g有零个或两个平方根，若y是其中一个平方根，则另一个平方根就是$q-y$。

下面的算法可以确定g是否有平方根，若有，就计算其中一个根。

输入：奇素数q，整数g，$0<g<q$。

输出：若存在g的平方根，则输出一个平方根，否则输出“不存在平方根”。

算法1：对$q \equiv 3 \pmod 4$，即存在正整数u，使得$q=4u+3$。

a) 计算$y = g^{u+1} \bmod q$(参见附录C.1.1)；

b) 计算$z = y^2 \bmod q$；

c) 若$z=g$，则输出 y ；否则输出“不存在平方根”。

算法2：对$q \equiv 5 \pmod 8$，即存在正整数u，使得$q=8u+5$。

a) 计算$z = g^{2u+1} \bmod q$(参见附录C.1.1)；

b) 若$z \equiv 1 \pmod q$，计算$y = g^{u+1} \bmod q$，输出y，终止算法；

c) 若$z \equiv -1 \pmod q$，计算$y = (2g \cdot (4g)^u) \bmod q$，输出$y$，终止算法；

d) 输出“不存在平方根”。

算法3：对$q \equiv 1 \pmod 8$，即存在正整数u，使得$q=8u+1$。

a) 置$Y=g$；

b) 生成随机数X，$0<X<q$；

c) 计算Lucas序列元素(见附录C.1.3)：$U = U_{4u+1} \bmod q$，$V = V_{4u+1} \bmod q$；

d) 若$V^2 \equiv 4Y \pmod q$，则输出 $y = (V/2) \bmod q$，并终止；

e) 若$U \bmod q \neq 1$且$U \bmod q \neq q-1$，则输出“不存在平方根”，并终止；

f) 返回步骤b)。

C.1.4.2 F_{q^2}上平方根的求解

设q是奇素数，对于二次扩域F_{q^2}，假设约化多项式为$f(x)=x^2+n$，$n \in F_q$，则F_{q^2}中元素β可表示成$a+bx$的形式，$a,b \in F_q$，则β的平方根为：

$$\sqrt{\beta} = \sqrt{a+bx} = \pm\left(\sqrt{\frac{a+\sqrt{a^2-nb^2}}{2}} + \frac{xb}{2\sqrt{\frac{a+\sqrt{a^2-nb^2}}{2}}}\right) \text{ 或}$$

$$\pm\left(\sqrt{\frac{a-\sqrt{a^2-nb^2}}{2}} + \frac{xb}{2\sqrt{\frac{a-\sqrt{a^2-nb^2}}{2}}}\right)$$

下面的算法可以确定β是否有平方根，若有，就计算其中一个根。

输入：F_{q^2}中元素$\beta=a+bx$且$\beta \neq 0$，q为奇素数。

输出：若存在β的平方根，则输出一个平方根z，否则输出“不存在平方根”。

a) 计算$U = a^2 - nb^2$；

b) 利用C.1.4.1的方法求$U \bmod q$的平方根，若$U \bmod q$的平方根存在，记作w_i，即$w_i^2 = U \bmod q$，$i=1,2$，转步骤c)；否则输出“不存在平方根”，并终止；

c) 对i从1至2执行：

c.1) 计算$V=(a+w_i)/2$ ；

c.2) 利用C.1.4.1的方法求$V \bmod q$的平方根，若$V \bmod q$的平方根存在，任取一个根y，即$y^2=V \bmod q$，转步骤d)；若$V \bmod q$的平方根不存在且i=2，输出“不存在平方根”，并终止算法；

d) 计算 $z_1 = b/2y \pmod q$，令$z_0=y$；

e) 输出 $z = z_0+z_1x$。

C.1.4.3 F_{q^m}上平方根的求解

C.1.4.3.1 F_{q^m}上平方元检测

设q是奇素数，且$m\geq2$，g是域F_{q^m}中非零元素，下面算法给出g是否为一个平方元的检测。

输入：域元素g。

输出：若g是平方元则输出“是平方元”，否则输出“不是平方元”。

a) 计算$B= g^{(q^m-1)/2}$；(参见C.1.1)

b) 若B=1，则输出“是平方元”；

c) 若B= −1，则输出“不是平方元”。

C.1.4.3.2 F_{q^m}上平方根的求解

设q是奇素数，且$m\geq2$。

输入：域元素g。

输出：若g是平方元则输出平方根B，否则输出“没有平方根”。

a) 随机选取非平方元Y；

b) 计算$q^m-1=2^u\times k$ ；（其中k为奇数。）

c) 计算$Y= Y^k$；

d) 计算$C= g^k$；

e) 计算$B= g^{(k+1)/2}$；

f) 若$C^{2^{u-1}}\neq1$，则输出“没有平方根”，终止算法；

g) 当$C\neq1$执行：

g.1) 设i是使$C^{2^i}=1$成立的最小正整数；

g.2) 计算$C= C\times Y^{2^{u-i}}$；

g.3) 计算$B= B\times Y^{2^{u-i-1}}$；

h) 输出B。

C.1.5 概率素性检测

设u是一个大的正整数，下面的概率算法(Miller-Rabin检测)将确定u是素数还是合数。

输入：一个大的奇数 u 和一个大的正整数T。

输出：“概率素数”或“合数”。

a) 计算 v 和奇数w，使得$u-1=2^v\cdot w$；

b) 对j从1到T 执行：

b.1) 在区间[2，$u-1$]中选取随机数a；

b.2) 置$b=a^w \bmod u$；

b.3) 若b=1或$u-1$，转到步骤b.6)；

b.4) 对i从1到$v-1$执行：

b.4.1) 置$b = b^2 \bmod u$；

b.4.2) 若$b = u-1$, 转到步骤b.6)；

b.4.3) 若$b = 1$，输出“合数”并终止；

b.4.4) 下一个i；

b.5) 输出“合数”, 并终止；

b.6) 下一个j；

c) 输出“概率素数”。

若算法输出“合数”，则u是一个合数。若算法输出“概率素数”， 则u是合数的概率小于2^{-2T}。这样，通过选取足够大的T，误差可以忽略。

C.2 有限域上的多项式

C.2.1 最大公因式

若$f(x)\neq 0$和$g(x)\neq 0$是系数在域F_q中的两个多项式，则唯一地存在次数最高的首一多项式$d(x)$，其系数在域F_q中且同时整除$f(x)$和$g(x)$。多项式$d(x)$称为$f(x)$和$g(x)$的最大公因子, 记为$\gcd(f(x), g(x))$。利用下面的算法(欧几里德算法)可计算出两个多项式的最大公因子。

输入：有限域F_q，F_q上的两个非零多项式 $f(x)\neq 0, g(x)\neq 0$。

输出：$d(x)=\gcd(f(x), g(x))$。

a) 置$a(x)=f(x)$，$b(x)=g(x)$；

b) 当$b(x)\neq 0$时，循环执行：

b.1) 置$c(x)=a(x) \mod b(x)$；

b.2) 置$a(x)=b(x)$；

b.3) 置$b(x)=c(x)$；

设α是$a(x)$的首项系数并输出$\alpha^{-1}a(x)$。

C.2.2 F_q上多项式不可约性的检测

设$f(x)$是F_q上的多项式，利用下面的算法可以有效地检测$f(x)$的不可约性。

输入：F_q上的首一多项式$f(x)$，素数q。

输出：若$f(x)$在F_q上不可约，则输出“正确”；否则，输出“错误”。

a) 置$u(x)=x$，$m=\deg(f(x))$；

b) 对i从1到$\lfloor m/2 \rfloor$执行：

b.1) 计算$u(x)=u^q(x) \mod f(x)$；

b.2) 计算$d(x)=\gcd(f(x)，u(x)-x)$；

b.3) 若$d(x)\neq 1$，则输出“错误”，并终止算法；

c) 输出“正确”。

C.3 椭圆曲线算法

C.3.1 椭圆曲线点的寻找

给定有限域上的椭圆曲线，利用下面的算法可有效地找出曲线上任意一个非无穷远点。

C.3.1.1 $E(F_p)$上点的寻找

输入：素数p, F_p上一条椭圆曲线E的参数a，b。

输出：$E(F_p)$上一个非无穷远点。

a) 选取随机整数x，$0\leq x<p$；

b) 置$\alpha=(x^3+ax+b) \mod p$；

c) 若$\alpha = 0$，则输出$(x, 0)$并终止算法；

d) 求$\alpha \bmod p$的平方根y (参见附录C.1.4.1)；

e) 若步骤 d) 的输出是“不存在平方根”，则返回步骤a)；

f) 输出(x, y)。

C.3.1.2 $E(F_{q^m})$ $(m \geq 2)$上点的寻找

输入：有限域F_{q^m}(q为奇素数)， F_{q^m}上的椭圆曲线E的参数a, b。

输出：E上一个非无穷远点。

a) 随机选取F_{q^m}上元素x；

b) 在F_{q^m}上计算$\alpha = x^3 + ax + b$ ；

c) 若$\alpha = 0$，则输出$(x, 0)$并终止算法；

d) 在F_{q^m}上求α 的平方根y (参见附录C.1.4.3)；

e) 若步骤 d) 的输出是“不存在平方根”，则返回步骤a)；

f) 输出(x, y)。

C.3.2 椭圆曲线上l阶点的寻找

本算法可用于椭圆曲线l阶子群生成元的求取。

输入：椭圆曲线$E(F_q)$的参数a、b，曲线阶$\#E(F_q) = n = l \cdot r$，其中l为素数。

输出：$E(F_q)$上一个l阶点。

a) 用C.3.1的方法随机选取曲线上点Q；

b) 计算$P=[r]Q$；

c) 若$P=O$ ，返回步骤a)；

d) 输出P。

C.3.3 扭曲线上l阶点的寻找

设F_{q^m}上椭圆曲线E的方程：$y^2=x^3+ax+b$,其阶$\#E(F_{q^m})=q^m+1-t$,设其扭曲线E'的方程：$y^2=x^3+\beta^2 \cdot ax+ \beta^3 \cdot b$，$\beta$为$F_{q^m}$上非平方元，$E'(F_{q^m})$的阶$\#E'(F_{q^m})=q^m+1+t$。

输入：椭圆曲线$E(F_{q^m})$的扭曲线$E'(F_{q^m})$的参数a、b和β，扭曲线阶$\#E'(F_{q^m}) = n' = l \cdot r$，其中$l$为素数。

输出：$E'(F_{q^m})$上一个l阶点。

a) 用C.3.1的方法随机选取$E'(F_{q^m})$上点Q；

b) 计算$P=[r]Q$；

c) 若$P=O$ ，返回步骤a)；

否则，P是l阶点；

d) 输出P。

SM9 标识密码算法
第 2 部分：数字签名算法

目　　次

SM9 标识密码算法

第 2 部分：数字签名算法

本部分规定了用椭圆曲线对实现的基于标识的数字签名算法，包括数字签名生成算法和验证算法，并给出了数字签名与验证算法及其相应的流程。

1 术语和定义

1.1

消息 message

任意有限长度的比特串。

1.2

签名消息 signed message

由消息以及该消息的数字签名部分所组成的一组数据元素。

1.3

签名密钥 signature key

在数字签名生成过程中由签名者专用的秘密数据元素，即签名者的私钥。

1.4

签名主密钥 signature master key

处于标识密码密钥分层结构最顶层的密钥，包括签名主私钥和签名主公钥，其中签名主公钥公开，签名主私钥由 KGC 秘密保存。KGC 用签名主私钥和用户的标识生成用户的签名私钥。在标识密码中，签名主私钥一般由 KGC 通过随机数发生器产生，签名主公钥由签名主私钥结合系统参数产生。

1.5

标识 identity

可唯一确定一个实体身份的信息。标识应由实体无法否认的信息组成，如实体的可识别名称、电子邮箱、身份证号、电话号码、街道地址等。

1.6

密钥生成中心 key generation center；KGC

在本部分中，负责选择系统参数、生成签名主密钥并产生用户签名私钥的可信机构。

2 符号

下列符号适用于本部分。

A, B：使用标识密码系统的两个用户。

cf：椭圆曲线阶相对于 N 的余因子。

cid：用一个字节表示的曲线的识别符，其中 0x10 表示 F_p(素数 $p>2^{191}$) 上常曲线(即非超奇异曲线)，0x11 表示 F_p 上超奇异曲线，0x12 表示 F_p 上常曲线及其扭曲线。

ds_A：用户 A 的签名私钥。

e：从 $\mathbb{G}_1\times\mathbb{G}_2$ 到 $\mathbb{G}_T$ 的双线性对。

eid：用一个字节表示的双线性对 e 的识别符，其中 0x01 表示 Tate 对，0x02 表示 Weil 对，0x03 表示 Ate 对，0x04 表示 R-Ate 对。

$\mathbb{G}_T$：阶为素数N的乘法循环群。

$\mathbb{G}_1$：阶为素数N的加法循环群。

$\mathbb{G}_2$：阶为素数N的加法循环群。

g^u：乘法群 $\mathbb{G}_T$ 中元素 g 的 u 次幂，即 $g^u = \underbrace{g\cdot g\cdot ....\cdot g}_{u个}$，$u$ 是正整数。

$H_v(\ )$：密码杂凑函数。

$H_1(\ ), H_2(\ )$：由密码杂凑函数派生的密码函数。

hid：在本部分中，用一个字节表示的签名私钥生成函数识别符，由 KGC 选择并公开。

(h, S)：发送的签名。

(h', S')：收到的签名。

ID_A：用户 A 的标识，可以唯一确定用户 A 的公钥。

M：待签名消息。

M'：待验证消息。

mod n：模 n 运算。例如，23 mod 7=2。

N：循环群$\mathbb{G}_1$、$\mathbb{G}_2$和$\mathbb{G}_T$的阶，为大于2^{191}的素数。

$P_{pub\text{-}s}$：签名主公钥。

P_1：群$\mathbb{G}_1$的生成元。

P_2：群$\mathbb{G}_2$的生成元。

ks：签名主私钥。

$<P>$：由元素 P 生成的循环群。

$[u]P$：加法群 $\mathbb{G}_1$、$\mathbb{G}_2$ 中元素 P 的 u 倍。

$\lceil x\rceil$：顶函数，不小于 x 的最小整数。例如，$\lceil 7\rceil=7, \lceil 8.3\rceil=9$。

$\lfloor x\rfloor$：底函数，不大于 x 的最大整数。例如，$\lfloor 7\rfloor=7, \lfloor 8.3\rfloor=8$。

$x \| y$：x 与 y 的拼接，x 和 y 是比特串或字节串。

$[x, y]$：不小于 x 且不大于 y 的整数的集合。

β：扭曲线参数。

3 算法参数与辅助函数

3.1 总则

本部分规定了一个用椭圆曲线对实现的基于标识的数字签名算法。该算法的签名者持有一个标识和一个相应的签名私钥，该签名私钥由密钥生成中心通过签名主私钥和签名者的标识结合产生。签名者用自身签名私钥对数据产生数字签名，验证者用签名者的标识验证签名的可靠性。

在签名的生成和验证过程之前，都要用密码杂凑函数对待签消息M和待验证消息M'进行压缩。

3.2 系统参数组

系统参数组包括曲线识别符 cid；椭圆曲线基域 F_q 的参数；椭圆曲线方程参数 a 和 b；扭曲线参数 β(若 cid 的低 4 位为 2)；曲线阶的素因子 N 和相对于 N 的余因子 cf；曲线 $E(F_q)$相对于 N 的嵌入次数 k；$E(F_{qd_1})$(d_1 整除 k)的 N 阶循环子群 $\mathbb{G}_1$ 的生成元 P_1；$E(F_{qd_2})$ (d_2 整除 k)的 N 阶循环子群 $\mathbb{G}_2$ 的生成元 P_2；双线性对 e 的识别符 eid；（选项）$\mathbb{G}_2$ 到 $\mathbb{G}_1$ 的同态映射 ψ。

双线性对 e 的值域为 N 阶乘法循环群 $\mathbb{G}_T$。

3.3 系统签名主密钥和用户签名密钥的产生

KGC产生随机数$ks\in[1, N-1]$作为签名主私钥，计算$\mathbb{G}_2$中的元素$P_{pub\text{-}s}=[ks]P_2$作为签名主公钥，则签名主密钥对为$(ks, P_{pub\text{-}s})$。KGC秘密保存ks，公开$P_{pub\text{-}s}$。

KGC选择并公开用一个字节表示的签名私钥生成函数识别符hid。

用户A的标识为ID_A，为产生用户A的签名私钥ds_A，KGC首先在有限域F_N上计算$t_1=H_1(ID_A\|hid,N)+ks$，若$t_1=0$则需重新产生签名主私钥，计算和公开签名主公钥，并更新已有用户的签名私钥；否则计算$t_2=ks\cdot t_1^{-1} \bmod N$，然后计算$ds_A=[t_2]P_1$。

3.4 辅助函数

3.4.1 概述

在本部分规定的基于标识的数字签名算法中，涉及到两类辅助函数：密码杂凑函数与随机数发生器。

3.4.2 密码杂凑函数

3.4.2.1 密码杂凑函数 $H_v(\ )$

密码杂凑函数$H_v(\ )$的输出是长度恰为v比特的杂凑值。本部分规定使用国家密码管理主管部门批准的密码杂凑函数，如SM3密码杂凑算法。

3.4.2.2 密码函数 $H_1(\ )$

密码函数$H_1(Z, n)$的输入为比特串Z和整数n，输出为一个整数$h_1\in[1, n-1]$。 $H_1(Z, n)$需要调用密码杂凑函数$H_v(\ )$。关于密码杂凑函数$H_v(\ )$，应符合本部分3.4.2.1的规定。

密码函数 $H_1(Z, n)$：

输入：比特串 Z，整数 n。

输出：整数$h_1\in[1, n-1]$。

步骤 1：初始化一个 32 比特构成的计数器 ct=0x00000001；

步骤 2：计算 $hlen=8\times\lceil(5\times(\log_2 n))/32\rceil$；

步骤 3：对 i 从 1 到$\lceil hlen/v\rceil$执行：

步骤 3.1：计算 $Ha_i=H_v(0x01\|Z\|ct)$；

步骤 3.2：ct++；

步骤 4：若 $hlen/v$ 是整数，令 $Ha!_{\lceil hlen/v\rceil}=Ha_{\lceil hlen/v\rceil}$，

否则令 $Ha!_{\lceil hlen/v\rceil}$为 $Ha_{\lceil hlen/v\rceil}$最左边的$(hlen-(v\times\lfloor hlen/v\rfloor))$比特；

步骤 5：令 $Ha=Ha_1\|Ha_2\|\cdots\|Ha_{\lceil hlen/v\rceil-1}\|Ha!_{\lceil hlen/v\rceil}$，按本文第 1 部分 6.2.4 和 6.2.3 给出的细节将 Ha 的数据类型转换为整数；

步骤6：计算$h_1=(Ha \bmod (n-1))+1$。

3.4.2.3 密码函数 $H_2(\ )$

密码函数$H_2(Z, n)$的输入为比特串Z和整数n，输出为一个整数$h_2\in[1, n-1]$。$H_2(Z, n)$需要调用密码杂凑函数$H_v(\ )$。关于密码杂凑函数$H_v(\ )$，应符合本部分3.4.2.1的规定。

密码函数 $H_2(Z, n)$：

输入：比特串 Z，整数 n。

输出：整数$h_2\in[1, n-1]$。

步骤 1：初始化一个 32 比特构成的计数器 ct=0x00000001；

步骤 2：计算 $hlen$=8×⌈(5×($\log_2 n$))/32⌉；

步骤 3：对 i 从 1 到⌈$hlen/v$⌉执行：

步骤 3.1：计算 $Ha_i=H_v(0x02||Z||ct)$；

步骤 3.2：ct++；

步骤 4：若 $hlen/v$ 是整数，令 $Ha!_{\lceil hlen/v\rceil}=Ha_{\lceil hlen/v\rceil}$，

否则令 $Ha!_{\lceil hlen/v\rceil}$为 $Ha_{\lceil hlen/v\rceil}$最左边的($hlen$–(v×⌊$hlen/v$⌋))比特；

步骤 5：令 $Ha=Ha_1\,||\,Ha_2\,||\cdots||\,Ha_{\lceil hlen/v\rceil-1}\,||\,Ha!_{\lceil hlen/v\rceil}$，按本标准第 1 部分 6.2.4 和 6.2.3 给出的细节将 Ha 的数据类型转换为整数；

步骤6：计算h_2=(Ha mod (n-1))+1。

3.4.3 随机数发生器

本部分规定使用国家密码管理主管部门批准的随机数发生器。

4 数字签名生成算法及流程

4.1 数字签名生成算法

设待签名的消息为比特串 M，为了获取消息 M 的数字签名(h, S)，作为签名者的用户 A 应实现以下运算步骤：

A1： 计算群 $\mathbb{G}_T$ 中的元素 $g=e(P_1, P_{pub\text{-}s})$；

A2： 产生随机数 $r\in[1, N-1]$；

A3： 计算群 $\mathbb{G}_T$ 中的元素 $w=g^r$，将 w 的数据类型转换为比特串；

A4： 计算整数 $h=H_2(M||w, N)$；

A5： 计算整数 $l=(r-h) \bmod N$，若 $l=0$ 则返回 A2；

A6： 计算群 $\mathbb{G}_1$ 中的元素 $S=[l]ds_A$；

A7： 消息 M 的签名为(h, S)。

4.2 数字签名生成算法流程

数字签名生成算法流程如图1。

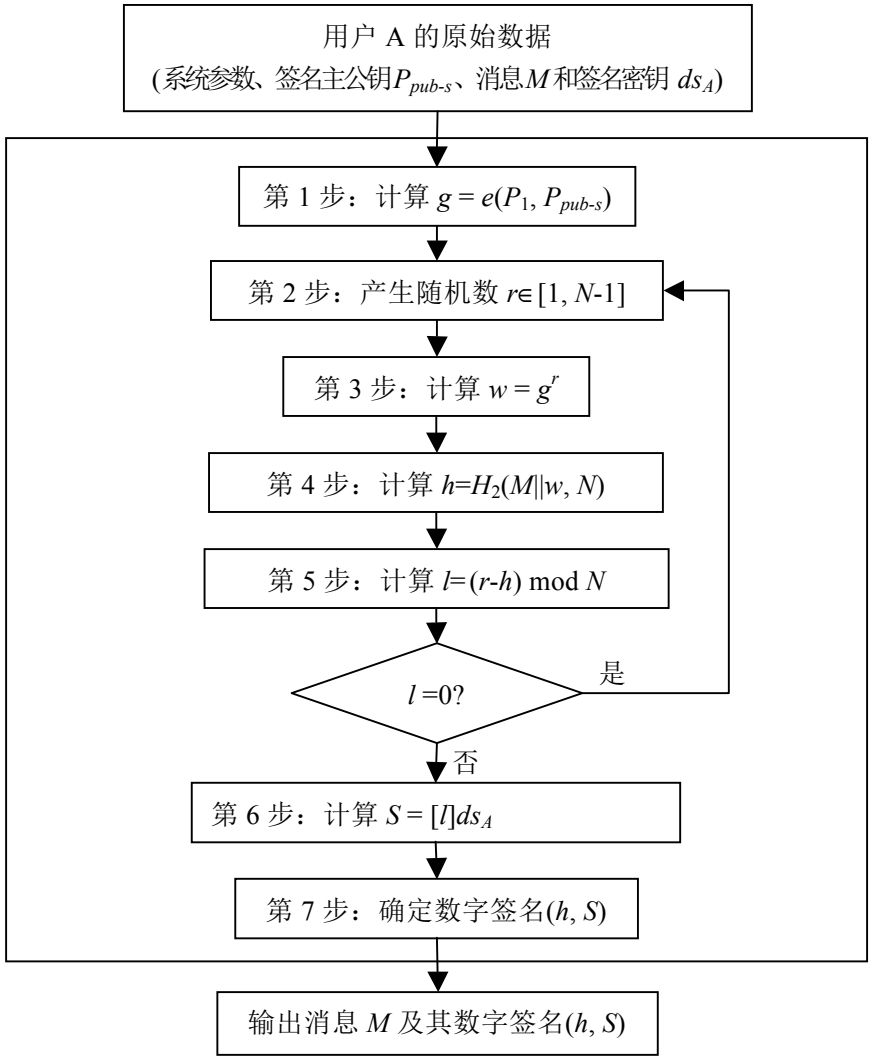

图1　数字签名生成算法流程

5　数字签名验证算法及流程

5.1　数字签名验证算法

为了检验收到的消息 M' 及其数字签名(h', S')，作为验证者的用户 B 应实现以下运算步骤：

B1：　检验 $h' \in [1, N-1]$ 是否成立，若不成立则验证不通过；

B2：　将 S'的数据类型转换为椭圆曲线上的点，检验 $S' \in \mathbb{G}_1$ 是否成立，若不成立则验证不通过；

B3：　计算群 $\mathbb{G}_T$ 中的元素 $g=e(P_1, P_{pub-s})$；

B4：　计算群 $\mathbb{G}_T$ 中的元素 $t=g^{h'}$；

B5：　计算整数 $h_1=H_1(ID_A||hid, N)$；

B6：　计算群 $\mathbb{G}_2$ 中的元素 $P=[h_1]P_2+P_{pub-s}$；

B7：　计算群 $\mathbb{G}_T$ 中的元素 $u = e(S', P)$；

B8：　计算群 $\mathbb{G}_T$ 中的元素 $w' = u \cdot t$，将 w'的数据类型转换为比特串；

B9：　计算整数 $h_2 = H_2(M'||w', N)$，检验 $h_2 = h'$ 是否成立，若成立则验证通过；否则验证不通过。

5.2　数字签名验证算法流程

数字签名验证算法流程如图2。

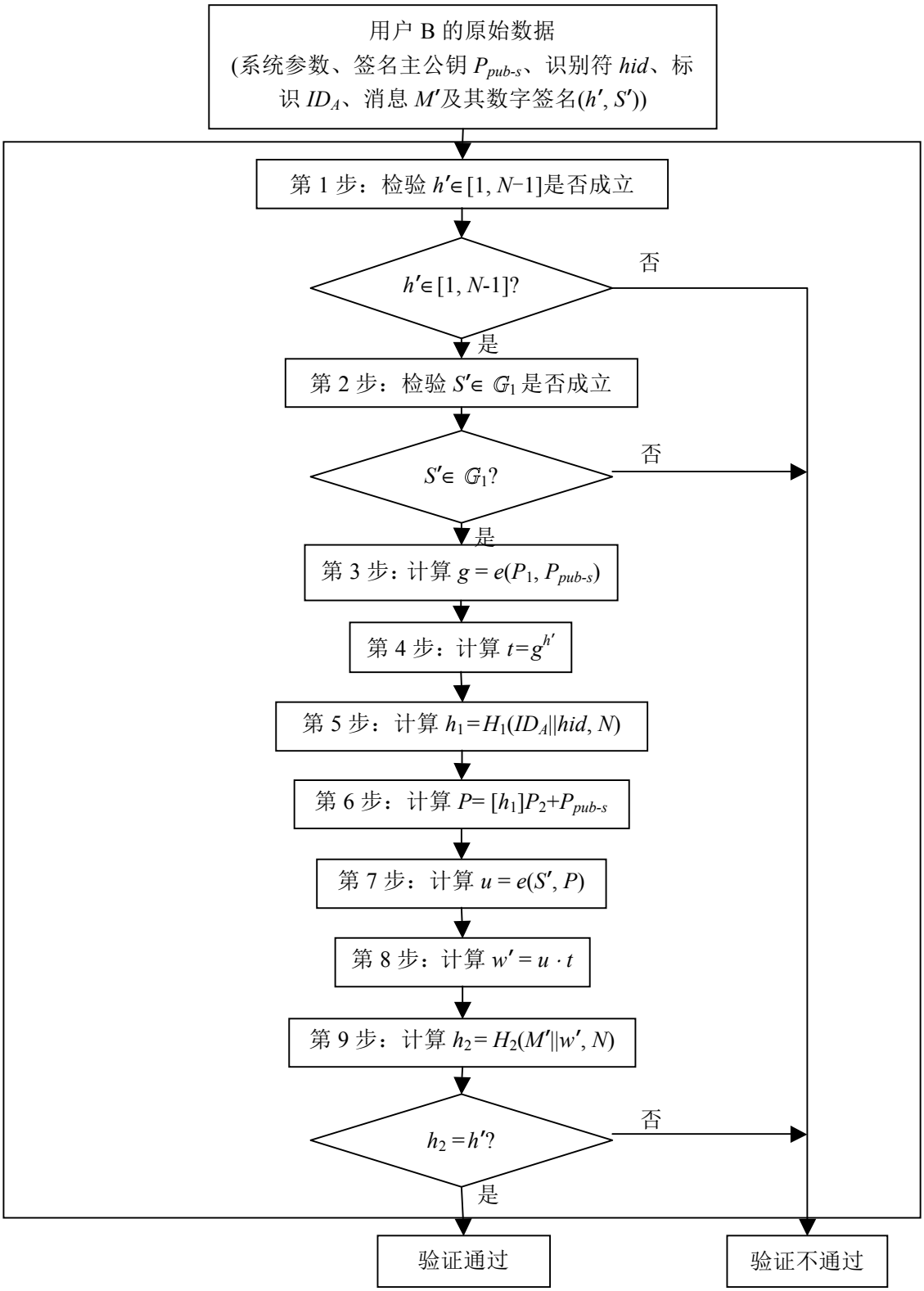

图2　数字签名验证算法流程

SM9 标识密码算法
第 3 部分：密钥交换协议

目　　次

SM9 标识密码算法

第 3 部分：密钥交换协议

本部分规定了用椭圆曲线对实现的基于标识的密钥交换协议，并提供了相应的流程。该协议可以使通信双方通过对方的标识和自身的私钥经两次或可选三次信息传递过程，计算获取一个由双方共同决定的共享秘密密钥。该秘密密钥可作为对称密码算法的会话密钥。协议中选项可以实现密钥确认。

1 术语

1.1

密钥交换 key exchange

在通信实体之间安全地交换密钥的方案，可以使通信双方在非安全通信线路上为信息传送安全地交换密钥。

1.2

密钥协商 key agreement

多个用户之间建立一个共享秘密密钥的过程，并且其中的任何一个用户都不能预先确定该密钥的值。

1.3

从A到B的密钥确认 key confirmation from A to B

使用户 B 确信用户 A 拥有特定秘密密钥的保证。

1.4

密钥派生函数 key derivation function

通过作用于共享秘密和双方都知道的其它参数，产生一个或多个共享秘密密钥的函数。

1.5

发起方 initiator

在一个协议的操作过程中发送首轮交换信息的用户。

1.6

响应方 responder

在一个协议的操作过程中不是发送首轮交换信息的用户。

1.7

加密主密钥 encryption master key

处于标识密码密钥分层结构最顶层的密钥，包括加密主私钥和加密主公钥，其中加密主公钥公开，加密主私钥由 KGC 秘密保存。KGC 用加密主私钥和用户的标识生成用户的加密私钥。在标识密码中，加密主私钥一般由 KGC 通过随机数发生器产生，加密主公钥由加密主私钥结合系统参数产生。

1.8

标识 identity

可唯一确定一个实体身份的信息。标识应由实体无法否认的信息组成，如实体的可识别名称、电子邮箱、身份证号、电话号码、街道地址等。

1.9

密钥生成中心 key generation center；KGC

在本部分中，负责选择系统参数、生成加密主密钥并产生用户加密私钥的可信机构。

2 符号

下列符号适用于本部分。

A, B：使用公钥密码系统的两个用户。

cf：椭圆曲线阶相对于 N 的余因子。

cid：用一个字节表示的曲线的识别符，其中 0x10 表示 F_p(素数 $p>2^{191}$) 上常曲线(即非超奇异曲线)，0x11 表示 F_p 上超奇异曲线，0x12 表示 F_p 上常曲线及其扭曲线。

de_A：用户A的加密私钥。

de_B：用户 B 的加密私钥。

e：从 $\mathbb{G}_1\times\mathbb{G}_2$ 到 $\mathbb{G}_T$ 的双线性对。

eid：用一个字节表示的双线性对 e 的识别符，其中 0x01 表示 Tate 对，0x02 表示 Weil 对，0x03 表示 Ate 对,0x04 表示 R-ate 对。

$\mathbb{G}_T$：阶为素数N的乘法循环群。

$\mathbb{G}_1$：阶为素数N的加法循环群。

$\mathbb{G}_2$：阶为素数N的加法循环群。

g^u：乘法群 $\mathbb{G}_T$ 中元素 g 的 u 次幂，即 $g^u=\underbrace{g\cdot g\cdot\ldots\cdot g}_{u个}$，$u$ 是正整数。

$H_v(\,),Hash(\,)$：密码杂凑函数。

$H_1(\,)$：由密码杂凑函数派生的密码函数。

hid：在本部分中，用一个字节表示的加密私钥生成函数识别符，由 KGC 选择并公开。

ID_A：用户 A 的标识，可以唯一确定用户 A 的公钥。

ID_B：用户 B 的标识，可以唯一确定用户 B 的公钥。

$KDF(\,)$：密钥派生函数。

N：循环群$\mathbb{G}_1$、$\mathbb{G}_2$和$\mathbb{G}_T$的阶，为大于2^{191}的素数。

$P_{pub\text{-}e}$：加密主公钥。

P_1：群 $\mathbb{G}_1$ 的生成元。

P_2：群 $\mathbb{G}_2$ 的生成元。

r_A：密钥交换中用户 A 产生的临时密钥值。

r_B：密钥交换中用户 B 产生的临时密钥值。

SK_A, SK_B：密钥交换协议商定的共享秘密密钥。

ke：加密主私钥。

$<P>$：由元素 P 生成的循环群。

$[u]P$：加法群 $\mathbb{G}_1$、$\mathbb{G}_2$ 中元素 P 的 u 倍。

$\lceil x \rceil$：顶函数，不小于 x 的最小整数。例如，$\lceil 7 \rceil = 7, \lceil 8.3 \rceil = 9$。

$\lfloor x \rfloor$：底函数，不大于 x 的最大整数。例如，$\lfloor 7 \rfloor = 7, \lfloor 8.3 \rfloor = 8$。

$x||y$：x 与 y 的拼接，x 和 y 是比特串或字节串。

$[x, y]$：不小于x且不大于y的整数的集合。

β：扭曲线参数。

3 算法参数与辅助函数

3.1 总则

本部分规定了一个用椭圆曲线对实现的基于标识的密钥交换协议。参与密钥交换的发起方用户A和响应方用户B各自持有一个标识和一个相应的加密私钥，加密私钥均由密钥生成中心通过加密主私钥和用户的标识结合产生。用户A和B通过交互的信息传递，用标识和各自的加密私钥来商定一个只有他们知道的秘密密钥，用户双方可以通过可选项实现密钥确认。这个共享的秘密密钥通常用在某个对称密码算法中。该密钥交换协议能够用于密钥管理和协商。

3.2 系统参数组

系统参数组包括曲线识别符 cid；椭圆曲线基域 F_q 的参数；椭圆曲线方程参数 a 和 b；扭曲线参数 β(若 cid 的低 4 位为 2)；曲线阶的素因子 N 和相对于 N 的余因子 cf；曲线 $E(F_q)$相对于 N 的嵌入次数 k；$E(F_{q^{d_1}})$(d_1 整除 k)的 N 阶循环子群 $\mathbb{G}_1$ 的生成元 P_1；$E(F_{q^{d_2}})$ (d_2 整除 k)的 N 阶循环子群 $\mathbb{G}_2$ 的生成元 P_2；双线性对 e 的识别符 eid；（选项）$\mathbb{G}_2$ 到 $\mathbb{G}_1$ 的同态映射 ψ。

双线性对 e 的值域为 N 阶乘法循环群 $\mathbb{G}_T$。

3.3 系统加密主密钥和用户加密密钥的产生

KGC产生随机数$ke \in [1, N-1]$作为加密主私钥，计算$\mathbb{G}_1$中的元素$P_{pub\text{-}e}=[ke]P_1$作为加密主公钥，则加密主密钥对为$(ke, P_{pub\text{-}e})$。KGC秘密保存ke，公开$P_{pub\text{-}e}$。

KGC选择并公开用一个字节表示的加密私钥生成函数识别符hid。

用户A和B的标识分别为ID_A和ID_B。为产生用户A的加密私钥de_A，KGC首先在有限域F_N上计算$t_1=H_1(ID_A||hid,N)+ke$，若$t_1=0$则需重新产生加密主私钥，计算和公开加密主公钥，并更新已有用户的加密私钥；否则计算$t_2 = ke \cdot t_1^{-1}$，然后计算$de_A=[t_2]P_2$。为产生用户B的加密私钥de_B，KGC首先在有限域F_N上计算$t_3=H_1(ID_B||hid,N)+ke$，若$t_3=0$则需重新产生加密主私钥，计算和公开加密主公钥，并更新已有用户的加密私钥；否则计算$t_4 = ke \cdot t_3^{-1}$，然后计算$de_B=[t_4]P_2$。

3.4 辅助函数

3.4.1 概述

在本部分规定的基于标识的密钥交换协议中，涉及到三类辅助函数：密码杂凑函数、密钥派生函数与随机数发生器。这三类辅助函数的强弱直接影响密钥交换协议的安全性。

3.4.2 密码杂凑函数

3.4.2.1 密码杂凑函数 $H_v(\ )$

密码杂凑函数$H_v(\ )$的输出是长度恰为v比特的杂凑值。本部分规定使用国家密码管理主管部门批准的密码杂凑函数，如SM3密码杂凑算法。

3.4.2.2 密码函数 $H_1(\ )$

密码函数$H_1(Z, n)$的输入为比特串Z和整数n，输出为一个整数$h_1 \in [1, n-1]$。 $H_1(Z, n)$需要调用密码杂凑函数$H_v(\ )$。关于密码杂凑函数$H_v(\ )$，应符合本部分5.4.2.1的规定。

密码函数 $H_1(Z, n)$：

输入：比特串 Z，整数 n。

输出：整数$h_1 \in [1, n-1]$。

步骤 1：初始化一个 32 比特构成的计数器 ct=0x00000001；

步骤 2：计算 $hlen=8\times\lceil(5\times(\log_2 n))/32\rceil$；

步骤 3：对 i 从 1 到$\lceil hlen/v\rceil$执行：

步骤 3.1：计算 $Ha_i=H_v(0x01\|Z\|ct)$；

步骤 3.2：ct++；

步骤 4：若 $hlen/v$ 是整数，令 $Ha!_{\lceil hlen/v\rceil}=Ha_{\lceil hlen/v\rceil}$，

否则令 $Ha!_{\lceil hlen/v\rceil}$为 $Ha_{\lceil hlen/v\rceil}$最左边的$(hlen-(v\times\lfloor hlen/v\rfloor))$比特；

步骤 5：令 $Ha = Ha_1 \| Ha_2 \| \cdots \| Ha_{\lceil hlen/v\rceil-1} \| Ha!_{\lceil hlen/v\rceil}$，将 Ha 的数据类型转换为整数；

步骤6：计算$h_1=(Ha \bmod (n-1))+1$。

3.4.2.3 密码函数 $H_2(\ )$

密码函数$H_2(Z, n)$的输入为比特串Z和整数n，输出为一个整数$h_2 \in [1, n-1]$。$H_2(Z, n)$需要调用密码杂凑函数$H_v(\ )$。关于密码杂凑函数$H_v(\ )$，应符合本部分5.4.2.1的规定。

密码函数 $H_2(Z, n)$：

输入：比特串 Z，整数 n。

输出：整数$h_2 \in [1, n-1]$。

步骤 1：初始化一个 32 比特构成的计数器 ct=0x00000001；

步骤 2：计算 $hlen=8\times\lceil(5\times(\log_2 n))/32\rceil$；

步骤 3：对 i 从 1 到$\lceil hlen/v\rceil$执行：

步骤 3.1：计算 $Ha_i=H_v(0x02\|Z\|ct)$；

步骤 3.2：ct++；

步骤 4：若 $hlen/v$ 是整数，令 $Ha!_{\lceil hlen/v\rceil}=Ha_{\lceil hlen/v\rceil}$，

否则令 $Ha!_{\lceil hlen/v\rceil}$为 $Ha_{\lceil hlen/v\rceil}$最左边的$(hlen-(v\times\lfloor hlen/v\rfloor))$比特；

步骤 5：令 $Ha = Ha_1 \| Ha_2 \| \cdots \| Ha_{\lceil hlen/v\rceil-1} \| Ha!_{\lceil hlen/v\rceil}$，将 Ha 的数据类型转换为整数；

步骤6：计算$h_2=(Ha \bmod (n-1))+1$。

3.4.3 密钥派生函数

密钥派生函数的作用是从一个共享的秘密比特串中派生出密钥数据。在密钥协商过程中，密钥派生函数作用在密钥交换所获共享的秘密比特串上，从中产生所需的会话密钥或进一步加密所需的密钥数据。

密钥派生函数需要调用密码杂凑函数。

设密码杂凑函数为 $H_v(\ )$，其输出是长度恰为 v 比特的杂凑值。

密钥派生函数 $KDF(Z, klen)$：

输入：比特串 Z(双方共享的数据)，整数 *klen*(表示要获得的密钥数据的比特长度，要求该值小于 $(2^{32}-1)v$)。

输出：长度为 *klen* 的密钥数据比特串 *K*。

步骤 1：初始化一个 32 比特构成的计数器 *ct*=0x00000001；

步骤 2：对 *i* 从 1 到⌈*klen*/*v*⌉执行：

步骤 2.1：计算 $Ha_i=H_v(Z\|ct)$；

步骤 2.2：*ct*++；

步骤 3：若 *klen*/*v* 是整数，令 $Ha!_{\lceil klen/v\rceil}=Ha_{\lceil klen/v\rceil}$，

否则令 $Ha!_{\lceil klen/v\rceil}$为 $Ha_{\lceil klen/v\rceil}$最左边的$(klen-(v\times\lfloor klen/v\rfloor))$比特；

步骤4：令$K=Ha_1\|Ha_2\|\cdots\|Ha_{\lceil klen/v\rceil-1}\|Ha!_{\lceil klen/v\rceil}$。

3.4.4 随机数发生器

本部分规定使用国家密码管理主管部门批准的随机数发生器。

4 密钥交换协议及流程

4.1 密钥交换协议

设用户 A 和 B 协商获得密钥数据的长度为 *klen* 比特，用户 A 为发起方，用户 B 为响应方。

用户 A 和 B 双方为了获得相同的密钥，应实现如下运算步骤：

用户 A：

A1： 计算群 $\mathbb{G}_1$ 中的元素 $Q_B=[H_1(ID_B\|hid, N)]P_1+P_{pub\text{-}e}$；

A2： 产生随机数 $r_A\in[1, N-1]$；

A3： 计算群 $\mathbb{G}_1$ 中的元素 $R_A=[r_A]Q_B$；

A4： 将 R_A 发送给用户 B；

用户 B：

B1： 计算群 $\mathbb{G}_1$ 中的元素 $Q_A=[H_1(ID_A\|hid, N)]P_1+P_{pub\text{-}e}$；

B2： 产生随机数 $r_B\in[1, N-1]$；

B3： 计算群 $\mathbb{G}_1$ 中的元素 $R_B=[r_B]Q_A$；

B4： 验证 $R_A\in\mathbb{G}_1$ 是否成立，若不成立则协商失败；否则计算群 $\mathbb{G}_T$ 中的元素 $g_1=e(R_A, de_B)$，$g_2=e(P_{pub-e}, P_2)^{r_B}$，$g_3=g_1^{r_B}$，将 g_1, g_2, g_3 的数据类型转换为比特串；

B5： 把 R_A 和 R_B 的数据类型转换为比特串，计算 $SK_B=KDF(ID_A\|ID_B\|R_A\|R_B\|g_1\|g_2\|g_3, klen)$；

B6： (选项)计算 $S_B=Hash(0x82\|g_1\|Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B))$；

B7： 将 R_B、(选项 S_B)发送给用户 A；

用户 A：

A5： 验证 $R_B\in\mathbb{G}_1$ 是否成立，若不成立则协商失败；否则计算群 $\mathbb{G}_T$ 中的元素 $g_1'=e(P_{pub-e}, P_2)^{r_A}$，$g_2'=e(R_B, de_A)$，$g_3'=(g_2')^{r_A}$，将 g_1', g_2', g_3'的数据类型转换为比特串；

A6： 把 R_A 和 R_B 的数据类型转换为比特串，(选项)计算 $S_1=Hash(0x82\|g_1'\|Hash(g_2'\|g_3'\|ID_A\|ID_B\|R_A\|R_B))$，并检验 $S_1=S_B$ 是否成立，若等式不成立则从 B 到 A 的密钥确认失败；

A7： 计算 $SK_A=KDF(ID_A\|ID_B\|R_A\|R_B\|g_1'\|g_2'\|g_3', klen)$；

A8： (选项)计算 $S_A=Hash(0x83\|g_1'\|Hash(g_2'\|g_3'\|ID_A\|ID_B\|R_A\|R_B))$，并将 S_A 发送给用户 B。

用户 B：

B8： (选项)计算 $S_2=Hash(0x83\|g_1\|Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B))$，并检验 $S_2=S_A$ 是否成立，若等式不成立则从 A 到 B 的密钥确认失败。

4.2 密钥交换协议流程

密钥交换协议流程如图1。

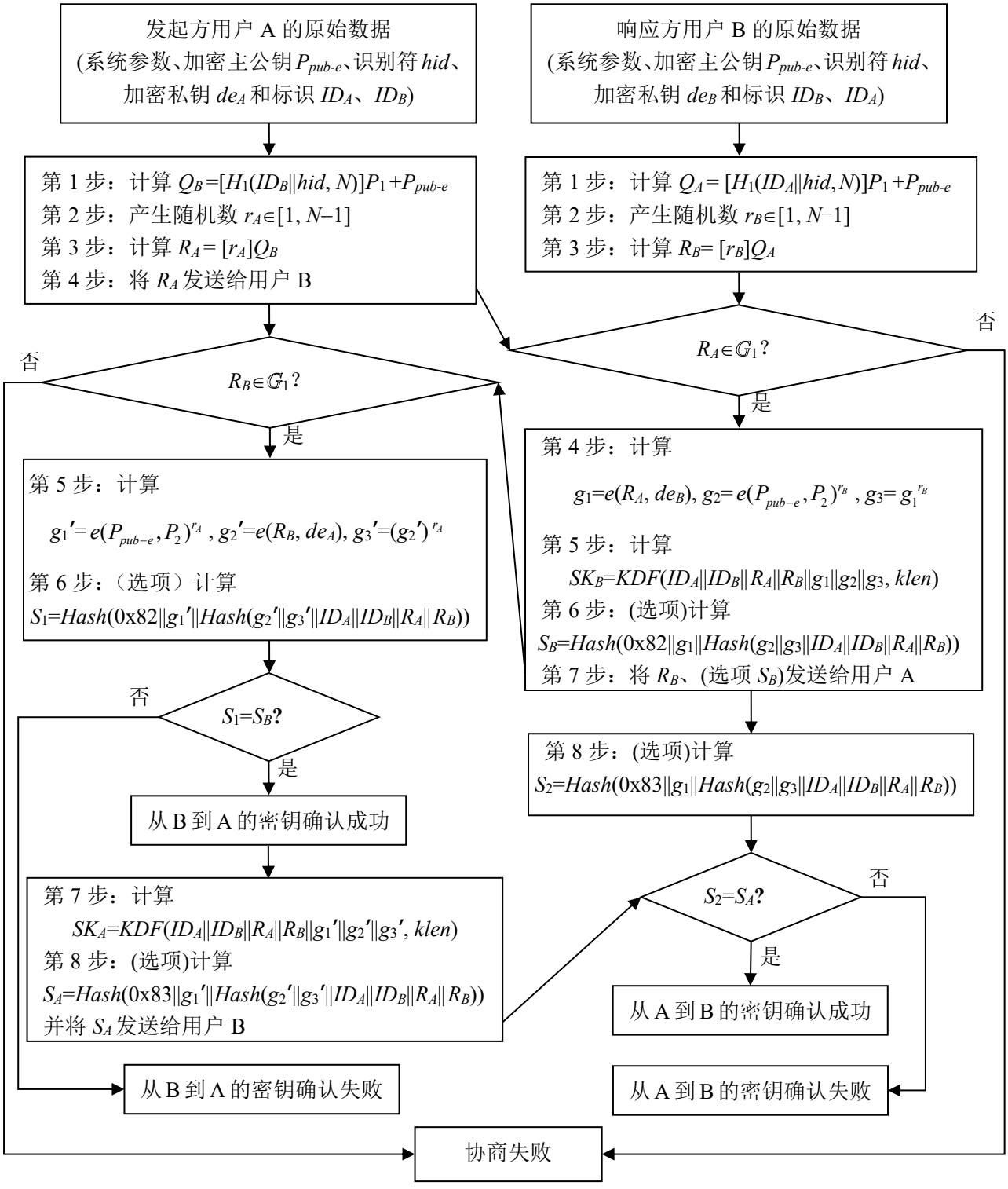

图 1 密钥交换协议流程

SM9 标识密码算法
第 4 部分：密钥封装机制和公钥加密算法

目　　次

SM9 标识密码算法

第 4 部分：密钥封装机制和公钥加密算法

本部分规定了用椭圆曲线对实现的基于标识的密钥封装机制和公钥加密与解密算法，并提供相应的流程。利用密钥封装机制可以封装密钥给特定的实体。公钥加密与解密算法即基于标识的非对称密码算法，该算法使消息发送者可以利用接收者的标识对消息进行加密，唯有接收者可以用相应的私钥对该密文进行解密，从而获取消息。

1 术语

1.1

秘密密钥 secret key
在密码体制中收发双方共同拥有的、而第三方不知道的一种密钥。

1.2

消息 message
任意有限长度的比特串。

1.3

明文 plaintext
未加密的信息。

1.4

密文 ciphertext
经过变换、信息内容被隐藏起来的数据。

1.5

加密 encipherment
为了产生密文，即隐藏数据的信息内容，由密码算法对数据进行(可逆)变换。

1.6

解密 decipherment
加密对应的逆过程。

1.7

密钥派生函数 key derivation function
通过作用于共享秘密和双方都知道的其它参数，产生一个或多个共享秘密密钥的函数。

1.8

消息认证码 message authentication code；MAC

一种认证算法作用于特定的密钥和消息比特串所得出的一段码字，可用来鉴别数据的来源和检验数据的完整性。用于求取消息认证码的函数称作消息认证码函数。

1.9

加密主密钥 encryption master key

处于标识密码密钥分层结构最顶层的密钥，包括加密主私钥和加密主公钥，其中加密主公钥公开，加密主私钥由 KGC 秘密保存。KGC 用加密主私钥和用户的标识生成用户的加密私钥。在标识密码中，加密主私钥一般由 KGC 通过随机数发生器产生，加密主公钥由加密主私钥结合系统参数产生。

1.10

标识 identity

可唯一确定一个实体身份的信息。标识应由实体无法否认的信息组成，如实体的可识别名称、电子邮箱、身份证号、电话号码、街道地址等。

1.11

密钥生成中心 key generation center；KGC

在本部分中，负责选择系统参数、生成加密主密钥并产生用户加密私钥的可信机构。

2 符号

下列符号适用于本部分。

A, B：使用公钥密码系统的两个用户。

cf：椭圆曲线阶相对于 N 的余因子。

cid：用一个字节表示的曲线的识别符，其中 0x10 表示 F_p(素数 $p>2^{191}$) 上常曲线(即非超奇异曲线)，0x11 表示 F_p 上超奇异曲线，0x12 表示 F_p 上常曲线及其扭曲线。

Dec( )：分组解密算法。

de_B：用户B的私钥。

Enc( )：分组加密算法。

e：从 $\mathbb{G}_1\times\mathbb{G}_2$ 到 $\mathbb{G}_T$ 的双线性对。

eid：用一个字节表示的双线性对 *e* 的识别符，其中 0x01 表示 Tate 对，0x02 表示 Weil 对，0x03 表示 Ate 对，0x04 表示 R-ate 对。

$\mathbb{G}_T$：阶为素数N的乘法循环群。

$\mathbb{G}_1$：阶为素数N的加法循环群。

$\mathbb{G}_2$：阶为素数N的加法循环群。

g^u：乘法群 $\mathbb{G}_T$ 中元素 g 的 u 次幂，即 $g^u=\underbrace{g\cdot g\cdot\ldots\cdot g}_{u个}$，$u$ 是正整数。

H_v( )：密码杂凑函数。

H_1( )：由密码杂凑函数派生的密码函数。

hid：在本部分中，用一个字节表示的私钥生成函数识别符，由 KGC 选择并公开。

ID_B：用户B的标识，可以唯一确定用户B的公钥。

$KDF(\ )$：密钥派生函数。

M：待加密的消息。

M'：解密得到的消息。

$MAC(\ )$：消息认证码函数。

N：循环群$\mathbb{G}_1$、$\mathbb{G}_2$和$\mathbb{G}_T$的阶，为大于2^{191}的素数。

$P_{pub\text{-}e}$：加密主公钥。

P_1：群 $\mathbb{G}_1$ 的生成元。

P_2：群 $\mathbb{G}_2$ 的生成元。

ke：加密主私钥。

$<P>$：由元素 P 生成的循环群。

$[u]P$：加法群 $\mathbb{G}_1$、$\mathbb{G}_2$ 中元素 P 的 u 倍。

$x\|y$：x 与 y 的拼接，x 和 y 是比特串或字节串。

$[x, y]$：不小于 x 且不大于 y 的整数的集合。

$\oplus$：长度相等的两个比特串按比特的模2加运算。

β：扭曲线参数。

3 算法参数与辅助函数

3.1 总则

在现代密码系统中，密钥是控制密码变换的重要参数，而且密码的安全性极大地依赖于对密钥的安全保护。密钥封装机制使得封装者可以产生和加密一个秘密密钥给目标用户，而唯有目标用户可以解封装该秘密密钥，并把它作为进一步的会话密钥。

本部分规定了一个用椭圆曲线对实现的基于标识的密钥封装机制。解封装用户持有一个标识和一个相应的加密私钥，该加密私钥由密钥生成中心通过加密主私钥和解封装用户的标识结合产生。封装者利用解封装用户的标识产生并加密一个秘密密钥给对方，解封装用户则用相应的加密私钥解封装该秘密密钥。

本部分还规定了一个用椭圆曲线对实现的基于标识的公钥加密算法。该公钥加密算法是上述密钥封装机制和消息封装机制的结合，消息封装机制包括基于密钥派生函数的序列密码以及结合密钥派生函数的分组密码算法两种类型，该算法可提供消息的机密性。在基于标识的加密算法中，解密用户持有一个标识和一个相应的加密私钥，该加密私钥由密钥生成中心通过加密主私钥和解密用户的标识结合产生。加密用户用解密用户的标识加密数据，解密用户用自身加密私钥解密数据。

3.2 系统参数组

系统参数组包括曲线识别符 cid；椭圆曲线基域 F_q 的参数；椭圆曲线方程参数 a 和 b；扭曲线参数 β(若 cid 的低 4 位为 2)；曲线阶的素因子 N 和相对于 N 的余因子 cf；曲线 $E(F_q)$相对于 N 的嵌入次数 k；$E(F_{q^{d_1}})$ (d_1 整除 k)的 N 阶循环子群 $\mathbb{G}_1$ 的生成元 P_1；$E(F_{q^{d_2}})$ (d_2 整除 k)的 N 阶循环子群 $\mathbb{G}_2$ 的生成元 P_2；双线性对 e 的识别符 eid；（选项）$\mathbb{G}_2$ 到 $\mathbb{G}_1$ 的同态映射 ψ。

双线性对 e 的值域为 N 阶乘法循环群 $\mathbb{G}_T$。

3.3 系统加密主密钥和用户加密密钥的产生

KGC产生随机数$ke \in [1, N-1]$作为加密主私钥，计算$\mathbb{G}_1$中的元素$P_{pub\text{-}e}=[ke]P_1$作为加密主公钥，则加密主密钥对为$(ke, P_{pub\text{-}e})$。KGC秘密保存ke，公开$P_{pub\text{-}e}$。

KGC选择并公开用一个字节表示的加密私钥生成函数识别符hid。

用户B的标识为ID_B，为产生用户B的加密私钥de_B，KGC首先在有限域F_N上计算$t_1=H_1(ID_B||hid,N)+ke$，若$t_1=0$则需重新产生加密主私钥，计算和公开加密主公钥，并更新已有用户的加密私钥；否则计算$t_2 = ke \cdot t_1^{-1}$，然后计算$de_B=[t_2]P_2$。

3.4 辅助函数

3.4.1 概述

在本部分规定的基于标识的密钥封装机制和公钥加密算法中，涉及到五类辅助函数：密码杂凑函数、密钥派生函数、消息认证码函数、随机数发生器和分组密码算法。这五类辅助函数的强弱直接影响密钥封装机制和公钥加密算法的安全性。

3.4.2 密码杂凑函数

3.4.2.1 密码杂凑函数 $H_v(\ )$

密码杂凑函数$H_v(\ )$的输出是长度恰为v比特的杂凑值。本部分规定使用国家密码管理主管部门批准的密码杂凑函数，如SM3密码杂凑算法。

3.4.2.2 密码函数 $H_1(\ )$

密码函数$H_1(Z, n)$的输入为比特串Z和整数n，输出为一个整数$h_1\in[1, n-1]$。 $H_1(Z, n)$需要调用密码杂凑函数$H_v(\ )$。关于密码杂凑函数$H_v(\ )$，应符合本部分5.4.2.1的规定。

密码函数 $H_1(Z, n)$：

输入：比特串 Z，整数 n。

输出：整数$h_1\in[1, n-1]$。

步骤 1：初始化一个 32 比特构成的计数器 ct=0x00000001；

步骤 2：计算 $hlen=8\times\lceil(5\times(\log_2 n))/32\rceil$；

步骤 3：对 i 从 1 到$\lceil hlen/v\rceil$执行：

步骤 3.1：计算 $Ha_i=H_v(0x01||Z||ct)$；

步骤 3.2：ct++；

步骤 4：若 $hlen/v$ 是整数，令 $Ha!_{\lceil hlen/v\rceil}=Ha_{\lceil hlen/v\rceil}$，

否则令 $Ha!_{\lceil hlen/v\rceil}$为 $Ha_{\lceil hlen/v\rceil}$最左边的$(hlen-(v\times\lfloor hlen/v\rfloor))$比特；

步骤 5：令 $Ha = Ha_1 \| Ha_2 \| \cdots \| Ha_{\lceil hlen/v\rceil-1} \| Ha!_{\lceil hlen/v\rceil}$，将 Ha 的数据类型转换为整数；

步骤6：计算$h_1=(Ha \bmod (n\text{-}1))+1$。

3.4.2.3 密码函数 $H_2(\ )$

密码函数$H_2(Z, n)$的输入为比特串Z和整数n，输出为一个整数$h_2\in[1, n-1]$。$H_2(Z, n)$需要调用密码杂凑函数$H_v(\ )$。关于密码杂凑函数$H_v(\ )$，应符合本部分5.4.2.1的规定。

密码函数 $H_2(Z, n)$：

输入：比特串 Z，整数 n。

输出：整数$h_2\in[1, n-1]$。

步骤 1：初始化一个 32 比特构成的计数器 ct=0x00000001；

步骤 2：计算 $hlen=8\times\lceil(5\times(\log_2 n))/32\rceil$；

步骤 3：对 i 从 1 到$\lceil hlen/v\rceil$执行：

步骤 3.1：计算 $Ha_i=H_v(0x02||Z||ct)$；

步骤 3.2：ct++；

步骤 4：若 $hlen/v$ 是整数，令 $Ha!_{\lceil hlen/v\rceil}=Ha_{\lceil hlen/v\rceil}$，

否则令 $Ha!_{\lceil hlen/v\rceil}$为 $Ha_{\lceil hlen/v\rceil}$最左边的($hlen-(v\times\lfloor hlen/v\rfloor)$)比特；

步骤 5：令 $Ha = Ha_1 \| Ha_2 \| \cdots \| Ha_{\lceil hlen/v\rceil-1} \| Ha!_{\lceil hlen/v\rceil}$，将 Ha 的数据类型转换为整数；

步骤6：计算$h_2=(Ha \bmod (n-1))+1$。

3.4.3 密钥派生函数

本部分采用的密钥派生函数$KDF(\ )$。

3.4.4 分组密码算法

分组密码算法包括加密算法$Enc(K_1, m)$和解密算法$Dec(K_1, c)$。$Enc(K_1, m)$表示用密钥K_1对明文m进行加密，其输出为密文比特串c；$Dec(K_1, c)$表示用密钥K_1对密文c进行解密，其输出为明文比特串m或“错误”。密钥K_1的比特长度记为K_1_len。

本部分规定使用国家密码管理主管部门批准的分组密码算法，如SM4分组密码算法。

3.4.5 消息认证码函数

消息认证码函数 $MAC(K_2, Z)$的作用是防止消息数据 Z 被非法篡改，它在密钥 K_2 的控制下，产生消息数据比特串 Z 的认证码，密钥 K_2 的比特长度记为 K_2_len。在本部分的基于标识的加密算法中，消息认证码函数使用密钥派生函数生成的密钥对密文比特串求取消息认证码，从而使解密者可以鉴别消息的来源和检验数据的完整性。

消息认证码函数需要调用密码杂凑函数。

设密码杂凑函数为 $H_v(\ )$，其输出是长度恰为 v 比特的杂凑值。

消息认证码函数 $MAC(K_2, Z)$：

输入：比特串 K_2(比特长度为 K_2_len 的密钥)，比特串 Z(待求取消息认证码的消息)。

输出：长度为 v 的消息认证码数据比特串 K。

步骤1：$K=H_v(Z\|K_2)$。

3.4.6 随机数发生器

本部分规定使用国家密码管理主管部门批准的随机数发生器。

4 密钥封装机制及流程

4.1 密钥封装算法及流程

4.1.1 密钥封装算法

为了封装比特长度为$klen$的密钥给用户B，作为封装者的用户A需要执行以下运算步骤：

A1： 计算群 $\mathbb{G}_1$ 中的元素 $Q_B=[H_1(ID_B\|hid, N)]P_1+P_{pub\text{-}e}$；

A2： 产生随机数 $r\in[1, N-1]$；

A3： 计算群 $\mathbb{G}_1$ 中的元素 $C=[r]Q_B$，将 C 的数据类型转换为比特串；

A4： 计算群 $\mathbb{G}_T$ 中的元素 $g=e(P_{pub\text{-}e}, P_2)$；

A5： 计算群 $\mathbb{G}_T$ 中的元素 $w=g^r$，将 w 的数据类型转换为比特串；

A6： 计算 $K=KDF(C\|w\|ID_B, klen)$，若 K 为全 0 比特串，则返回 A2。

A7： 输出(K, C)，其中 K 是被封装的密钥，C 是封装密文。

4.1.2 密钥封装算法流程

密钥封装算法流程如图1。

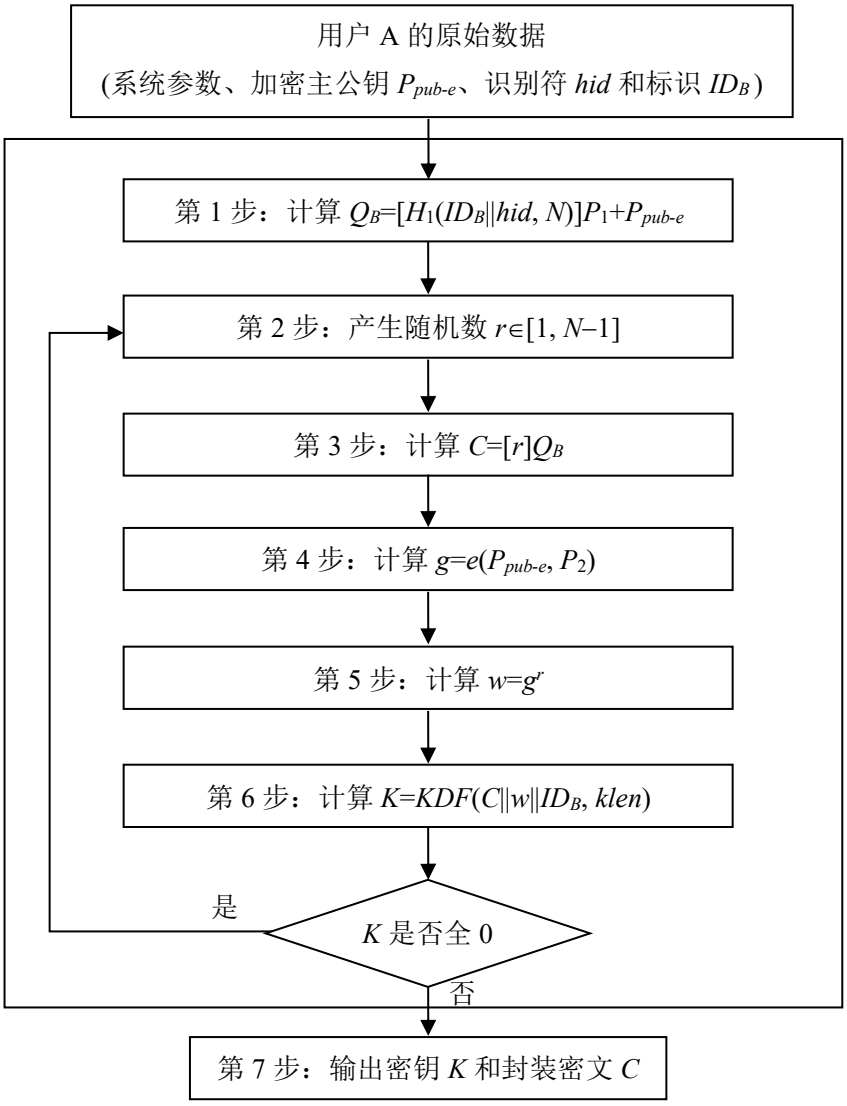

图 1　密钥封装算法流程

4.2　解封装算法及流程

4.2.1　解封装算法

用户B收到封装密文C后，为了对比特长度为$klen$的密钥解封装，需要执行以下运算步骤：

B1：　验证 $C\in\mathbb{G}_1$ 是否成立，若不成立则报错并退出；

B2：　计算群 $\mathbb{G}_T$ 中的元素 $w'=e(C, de_B)$，将 w'的数据类型转换为比特串；

B3：　将 C 的数据类型转换为比特串，计算封装的密钥 $K'=KDF(C||w'||ID_B, klen)$，若 K'为全 0 比特串，则报错并退出；

B4：　输出密钥 K'。

4.2.2　解封装算法流程

解封装算法流程如图2。

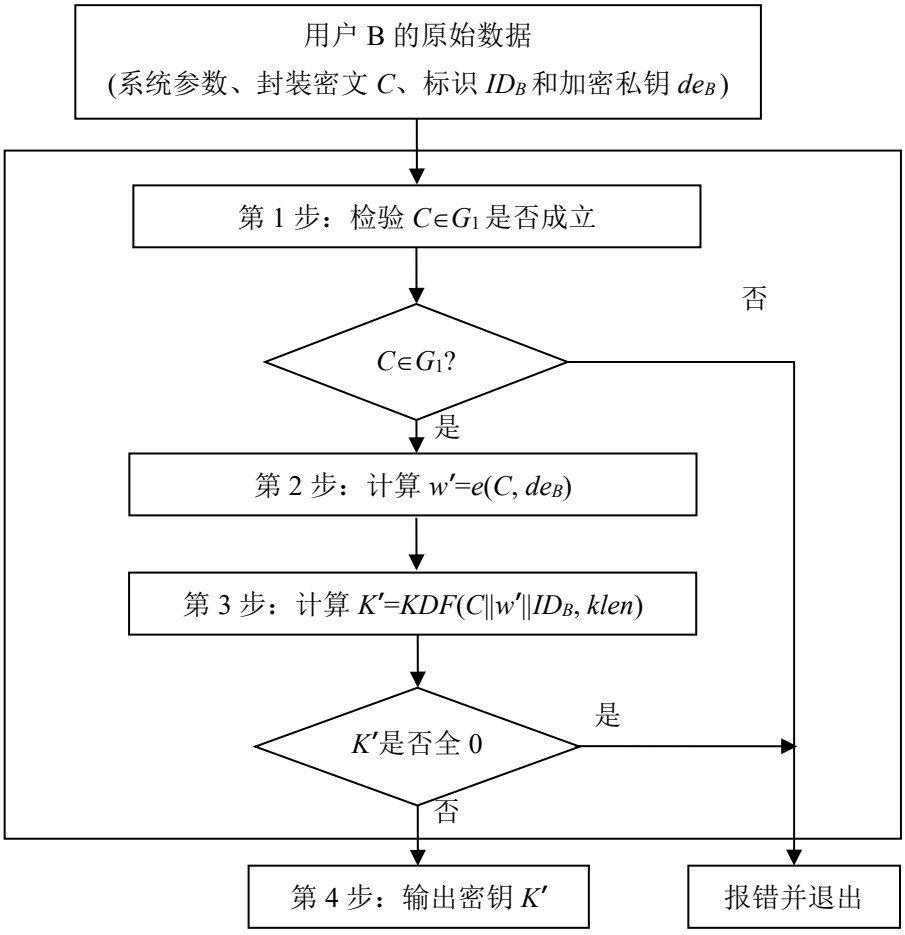

图 2　解封装算法流程

5　公钥加密算法及流程

5.1　加密算法及流程

5.1.1　加密算法

设需要发送的消息为比特串 M，$mlen$ 为 M 的比特长度，K_1_len 为分组密码算法中密钥 K_1 的比特长度，K_2_len 为函数 $MAC(K_2, Z)$中密钥 K_2 的比特长度。

为了加密明文 M 给用户 B，作为加密者的用户 A 应实现以下运算步骤：

A1：　计算群 $\mathbb{G}_1$ 中的元素 $Q_B=[H_1(ID_B\|hid, N)]P_1+P_{pub\text{-}e}$；

A2：　产生随机数 $r \in [1, N-1]$；

A3：　计算群 $\mathbb{G}_1$ 中的元素 $C_1=[r]Q_B$，将 C_1 的数据类型转换为比特串；

A4：　计算群 $\mathbb{G}_T$ 中的元素 $g=e(P_{pub\text{-}e}, P_2)$；

A5：　计算群 $\mathbb{G}_T$ 中的元素 $w=g^r$，按将 w 的数据类型转换为比特串；

A6：　按加密明文的方法分类进行计算：

a)　如果加密明文的方法是基于密钥派生函数的序列密码算法，则

1)　计算整数 $klen=mlen+K_2\_len$，然后计算 $K=KDF(C_1\|w\|ID_B, klen)$。令 K_1 为 K 最左边的 $mlen$ 比特，K_2 为剩下的 K_2_len 比特，若 K_1 为全 0 比特串，则返回 A2；

2) 计算 $C_2=M\oplus K_1$。

b) 如果加密明文的方法是结合密钥派生函数的分组密码算法，则

1) 计算整数 $klen=K_1\_len+K_2\_len$，然后计算 $K=KDF(C_1\|w\|ID_B, klen)$。令 K_1 为 K 最左边的 K_1_len 比特，K_2 为剩下的 K_2_len 比特，若 K_1 为全 0 比特串，则返回 A2；

2) 计算 $C_2=Enc(K_1, M)$。

A7： 计算 $C_3=MAC(K_2, C_2)$；

A8： 输出密文 $C=C_1\|C_3\|C_2$。

5.1.2 加密算法流程

加密算法流程如图3。

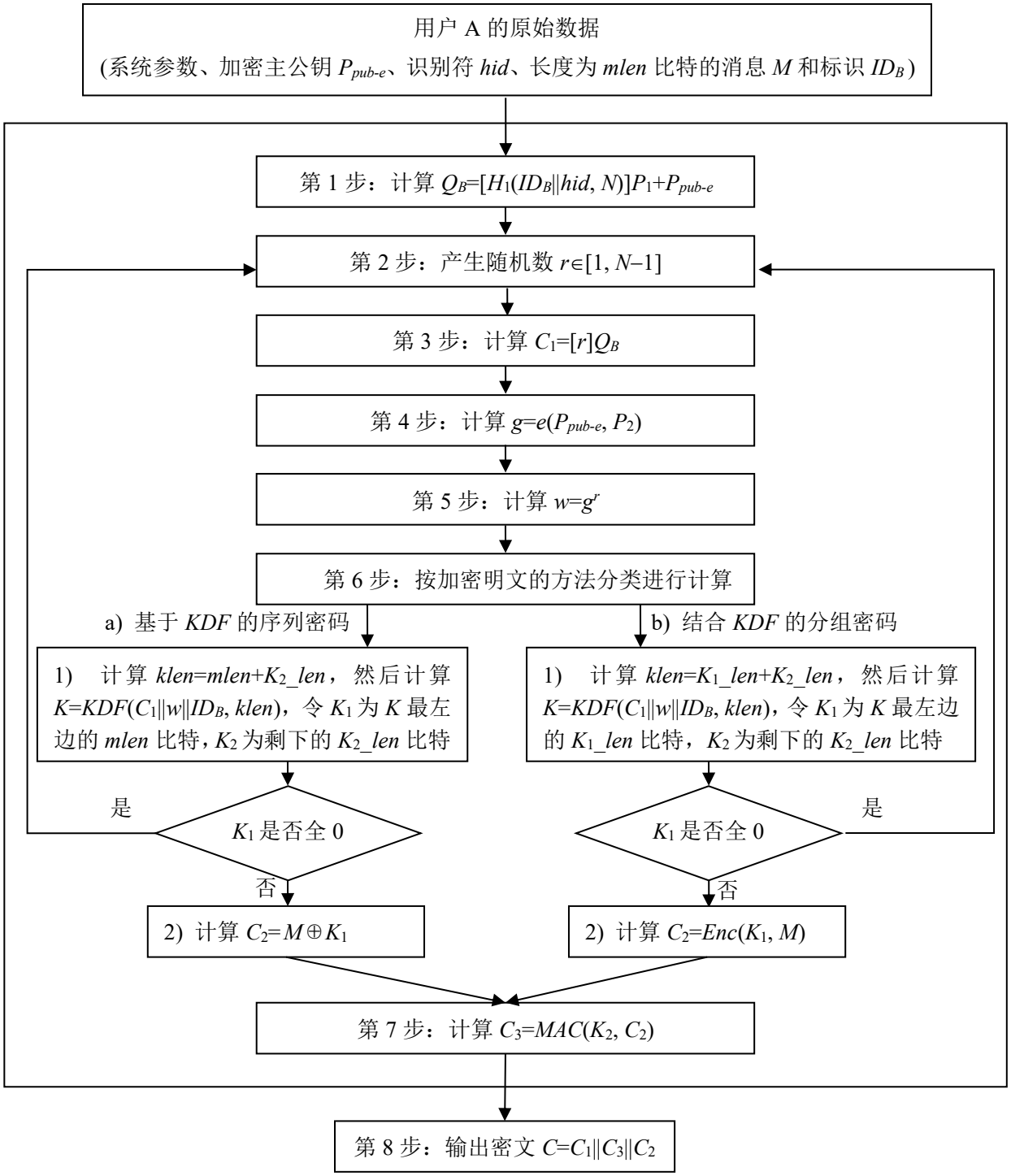

图 3　加密算法流程

5.2　解密算法及流程

5.2.1　解密算法

设 $mlen$ 为密文 $C=C_1\|C_3\|C_2$ 中 C_2 的比特长度，K_1_len 为分组密码算法中密钥 K_1 的比特长度，K_2_len

为函数 $MAC(K_2, Z)$中密钥 K_2 的比特长度。

为了对 C 进行解密，作为解密者的用户 B 应实现以下运算步骤：

B1： 从 C 中取出比特串 C_1，将 C_1 的数据类型转换为椭圆曲线上的点，验证 $C_1 \in G_1$ 是否成立，若不成立则报错并退出；

B2： 计算群 G_T 中的元素 $w'=e(C_1, de_B)$，将 w'的数据类型转换为比特串；

B3： 按加密明文的方法分类进行计算：

a) 如果加密明文的方法是基于密钥派生函数的序列密码算法，则

1) 计算整数 $klen=mlen+K_2\_len$，然后计算 $K'=KDF(C_1\|w'\|ID_B, klen)$。令 K_1'为 K'最左边的 $mlen$ 比特，K_2'为剩下的 K_2_len 比特，若 K_1'为全 0 比特串，则报错并退出；

2) 计算 $M'=C_2 \oplus K_1'$。

b) 如果加密明文的方法是结合密钥派生函数的分组密码算法，则

1) 计算整数 $klen=K_1\_len+K_2\_len$，然后计算 $K'=KDF(C_1\|w'\|ID_B, klen)$。令 K_1'为 K'最左边的 K_1_len 比特，K_2'为剩下的 K_2_len 比特，若 K_1'为全 0 比特串，则报错并退出；

2) 计算 $M'=Dec(K_1', C_2)$。

B4： 计算 $u=MAC(K_2', C_2)$，从 C 中取出比特串 C_3，若 $u \neq C_3$，则报错并退出；

B5： 输出明文 M'。

5.2.2 解密算法流程

解密算法流程如图4。

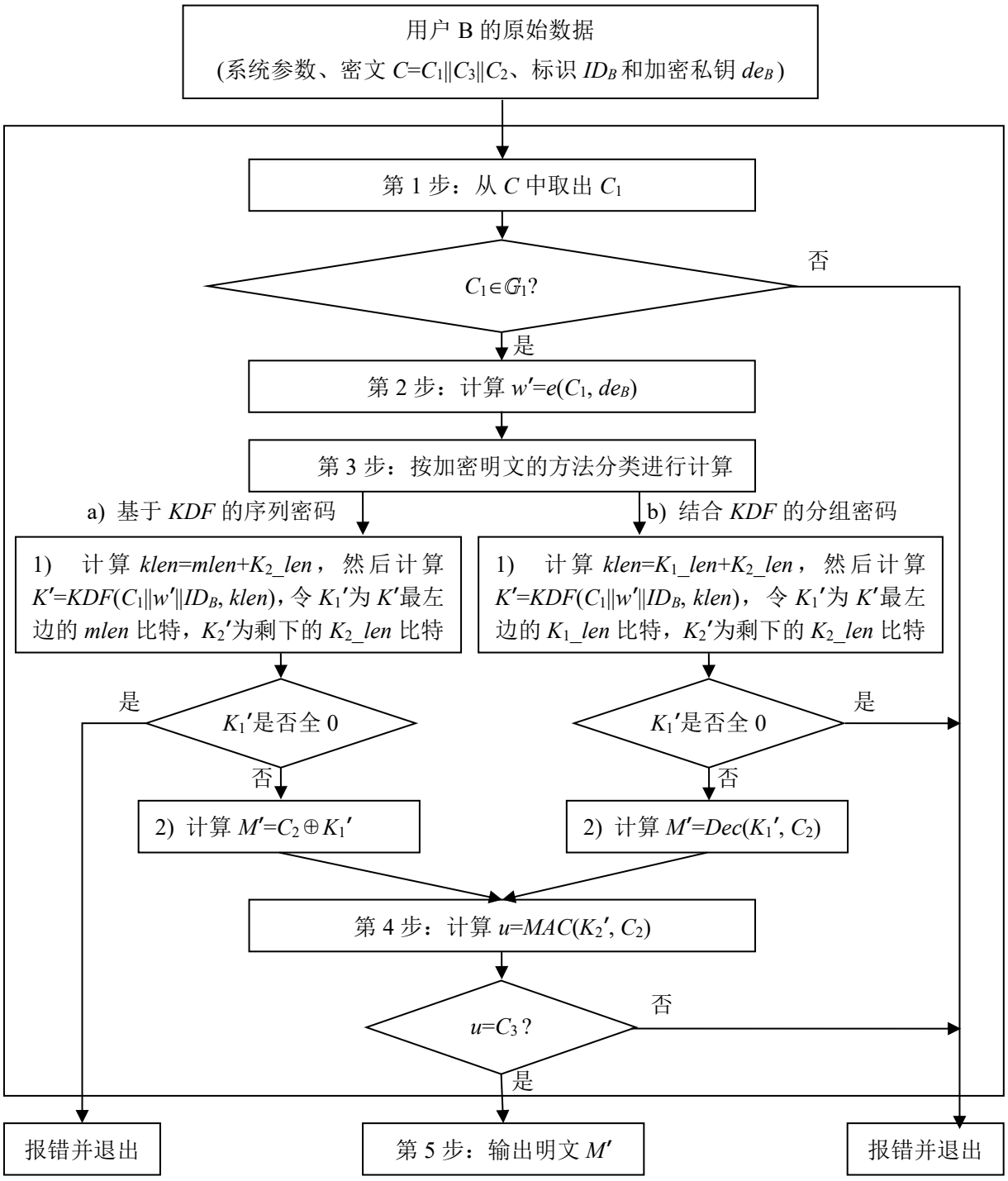
用户 B 的原始数据
(系统参数、密文 $C=C_1\|C_3\|C_2$、标识 ID_B 和加密私钥 de_B)
第 1 步：从 C 中取出 C_1
$C_1\in\mathbb{G}_1$?
否
是
第 2 步：计算 $w'=e(C_1, de_B)$
第 3 步：按加密明文的方法分类进行计算
a) 基于 KDF 的序列密码
b) 结合 KDF 的分组密码
1) 计算 $klen=mlen+K_2\_len$，然后计算 $K'=KDF(C_1\|w'\|ID_B, klen)$，令 K_1'为 K'最左边的 $mlen$ 比特，K_2'为剩下的 K_2_len 比特
1) 计算 $klen=K_1\_len+K_2\_len$，然后计算 $K'=KDF(C_1\|w'\|ID_B, klen)$，令 K_1'为 K'最左边的 K_1_len 比特，K_2'为剩下的 K_2_len 比特
是
K_1'是否全 0
否
2) 计算 $M'=C_2\oplus K_1'$
是
K_1'是否全 0
否
2) 计算 $M'=Dec(K_1', C_2)$
第 4 步：计算 $u=MAC(K_2', C_2)$
$u=C_3$?
否
是
报错并退出
第 5 步：输出明文 M'
报错并退出

图 4　解密算法流程

SM9 标识密码算法
第 5 部分：参数定义

目　　次

SM9 标识密码算法

第 5 部分：参数定义

本部分规定了SM9标识密码算法的曲线参数，并给出了数字签名算法、密钥交换协议、密钥封装机制、公钥加密算法示例。

1 系统参数

本文使用256位的BN曲线。

椭圆曲线方程：$y^2 = x^3 + b$。

曲线参数：

参数 t ：60000000 0058F98A

迹 $\mathrm{tr}(t) = 6t^2 + 1$：D8000000 019062ED 0000B98B 0CB27659

基域特征 $q(t) = 36t^4 + 36t^3 + 24t^2 + 6t + 1$：

B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27 E351457D

方程参数 b：05

群的阶 $N(t) = 36t^4 + 36t^3 + 18t^2 + 6t + 1$：

B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C D69ECF25

余因子 cf：1

嵌入次数 k：12

扭曲线的参数 β：$\sqrt{-2}$

k 的因子 $d_1=1, d_2=2$

曲线识别符 cid：0x12

群 $\mathbb{G}_1$ 的生成元 $P_1 = (x_{P_1}, y_{P_1})$：

坐标 x_{P_1}：93DE051D 62BF718F F5ED0704 487D01D6 E1E40869 09DC3280 E8C4E481 7C66DDDD

坐标 y_{P_1}：21FE8DDA 4F21E607 63106512 5C395BBC 1C1C00CB FA602435 0C464CD7 0A3EA616

群 $\mathbb{G}_2$ 的生成元 $P_2 = (x_{P_2}, y_{P_2})$：

坐标 x_{P_2}：（85AEF3D0 78640C98 597B6027 B441A01F F1DD2C19 0F5E93C4 54806C11 D8806141 ，
37227552 92130B08 D2AAB97F D34EC120 EE265948 D19C17AB F9B7213B AF82D65B ）

坐标 y_{P_2}：（17509B09 2E845C12 66BA0D26 2CBEE6ED 0736A96F A347C8BD 856DC76B 84EBEB96 ，
A7CF28D5 19BE3DA6 5F317015 3D278FF2 47EFBA98 A71A0811 6215BBA5 C999A7C7 ）

双线性对的识别符 eid：0x04

2 扩域元素的表示

$F_{q^{12}}$ 的 1-2-4-12 塔式扩张：

(1) $F_{q^2}[u] = F_q[u]/(u^2-\alpha)$, $\alpha = -2$；

(2) $F_{q^4}[v] = F_{q^2}[v]/(v^2-\xi)$, $\xi = u$；

(3) $F_{q^{12}}[w] = F_{q^4}[w]/(w^3-v)$, $v^2 = \xi$；

其中，

第(1)进行二次扩张的约化多项式为：$x^2-\alpha, \alpha = -2$；

第(2)进行二次扩张的约化多项式为：$x^2 - u, u^2 = \alpha, u = \sqrt{-2}$；

第(3)进行三次扩张的约化多项式为：$x^3 - v, v^2 = u, v = \sqrt[3]{\sqrt{-2}}$；

u 属于 F_{q^2}，表示为(1 , 0)，左边是第 1 维(高维)，右边是第 0 维（低维）。

v 属于 F_{q^4}，表示为(0 , 1 , 0 , 0)，其中左边(0 , 1)是 F_{q^4} 中元素以 F_{q^2} 表示的第 1 维（高维），右边(0 , 0)是 F_{q^4} 中元素以 F_{q^2} 表示的第 0 维（低维）。

$F_{q^{12}}$ 中元素有三种表示方法：

(1) $F_{q^{12}}$ 中元素 A 用 F_{q^4} 中元素表示：

$$A = aw^2+bw+c = (a,b,c),$$

a,b,c 用 F_{q^2} 中元素表示：

$$a = a_1v+a_0 = (a_1,a_0);$$
$$b = b_1v+b_0 = (b_1,b_0);$$
$$c = c_1v+c_0 = (c_1,c_0);$$

其中 $a_1,a_0,b_1,b_0,c_1,c_0 \in F_{q^2}$。

(2) $F_{q^{12}}$ 中元素 A 用 F_{q^2} 中的元素表示：

$$A = (a_1,a_0,b_1,b_0,c_1,c_0),$$

a_1,a_0,b_1,b_0,c_1,c_0 用基域 F_q 中的元素表示：

$$a_0=a_{0,1}u+a_{0,0}=(a_{0,1},a_{0,0});$$
$$a_1=a_{1,1}u+a_{1,0}=(a_{1,1},a_{1,0});$$
$$b_0=b_{0,1}u+b_{0,0}=(b_{0,1},b_{0,0});$$
$$b_1=b_{1,1}u+b_{1,0}=(b_{1,1},b_{1,0});$$
$$c_0=c_{0,1}u+c_{0,0}=(c_{0,1},c_{0,0});$$
$$c_1=c_{1,1}u+c_{1,0}=(c_{1,1},c_{1,0});$$

其中 $a_{0,1},a_{0,0},a_{1,1},a_{1,0},b_{0,1},b_{0,0},b_{1,1},b_{1,0},c_{0,1},c_{0,0},c_{1,1},c_{1,0} \in F_q$。

(3) $F_{q^{12}}$ 中元素 A 用基域 F_q 中的元素表示：

$$A=(a_{0,1},a_{0,0},a_{1,1},a_{1,0},b_{0,1},b_{0,0},b_{1,1},b_{1,0},c_{0,1},c_{0,0},c_{1,1},c_{1,0}),$$

其中 $a_{0,1},a_{0,0},a_{1,1},a_{1,0},b_{0,1},b_{0,0},b_{1,1},b_{1,0},c_{0,1},c_{0,0},c_{1,1},c_{1,0} \in F_q$。

F_{q^2} 中单位元的表示为(0, 1)；

F_{q^4} 中单位元的表示为(0, 0, 0, 1)；

$F_{q^{12}}$ 中单位元的表示为(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1)。

各种扩域中分量序为：左边是高维，右边是低维。

示例数据中，扩域中的元素均用基域中的元素表示。

附 录 A
数字签名算法示例

A.1 一般要求

本附录选用 SM3 算法给出的密码杂凑函数，其输入是长度小于 2^{64} 的消息比特串，输出是长度为 256 比特的杂凑值，记为 $H_{256}(\ )$。

本附录中，所有用 16 进制表示的数，左边为高位，右边为低位。

本附录中，消息采用ASCII编码。

A.2 数字签名与验证

椭圆曲线方程为：$y^2 = x^3 + b$

基域特征 q：B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27 E351457D

方程参数 b：05

群 $\mathbb{G}_1$,$\mathbb{G}_2$ 的阶 N：B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C D69ECF25

余因子 cf：1

嵌入次数 k：12

扭曲线的参数 β：$\sqrt{-2}$

群 $\mathbb{G}_1$ 的生成元 $P_1 = (x_{P_1}, y_{P_1})$：

坐标 x_{P_1}：93DE051D 62BF718F F5ED0704 487D01D6 E1E40869 09DC3280 E8C4E481 7C66DDDD

坐标 y_{P_1}：21FE8DDA 4F21E607 63106512 5C395BBC 1C1C00CB FA602435 0C464CD7 0A3EA616

群 $\mathbb{G}_2$ 的生成元 $P_2 = (x_{P_2}, y_{P_2})$：

坐标 x_{P_2}：(85AEF3D0 78640C98 597B6027 B441A01F F1DD2C19 0F5E93C4 54806C11 D8806141,
37227552 92130B08 D2AAB97F D34EC120 EE265948 D19C17AB F9B7213B AF82D65B)

坐标 y_{P_2}：(17509B09 2E845C12 66BA0D26 2CBEE6ED 0736A96F A347C8BD 856DC76B 84EBEB96,
A7CF28D5 19BE3DA6 5F317015 3D278FF2 47EFBA98 A71A0811 6215BBA5 C999A7C7)

双线性对的识别符 eid：0x04

签名主密钥和用户签名密钥产生过程中的相关值：

签名主私钥 ks：0130E7 8459D785 45CB54C5 87E02CF4 80CE0B66 340F319F 348A1D5B 1F2DC5F4

签名主公钥 $P_{pub\text{-}s} = [ks]P_2 = (x_{P_{pub\text{-}s}}, y_{P_{pub\text{-}s}})$：

坐标 $x_{P_{pub\text{-}s}}$：(9F64080B 3084F733 E48AFF4B 41B56501 1CE0711C 5E392CFB 0AB1B679 1B94C408,
29DBA116 152D1F78 6CE843ED 24A3B573 414D2177 386A92DD 8F14D656 96EA5E32)

坐标 $y_{P_{pub\text{-}s}}$：(69850938 ABEA0112 B57329F4 47E3A0CB AD3E2FDB 1A77F335 E89E1408 D0EF1C25,
41E00A53 DDA532DA 1A7CE027 B7A46F74 1006E85F 5CDFF073 0E75C05F B4E3216D)

签名私钥生成函数识别符 hid：0x01

实体 A 的标识 ID_A：Alice

ID_A 的 16 进制表示：416C6963 65

在有限域 F_N 上计算 $t_1 = H_1(ID_A \| hid, N) + ks$：

$ID_A \| hid$：416C6963 6501

$H_1(ID_A \| hid, N)$：2ACC468C 3926B0BD B2767E99 FF26E084 DE9CED8D BC7D5FBF 418027B6 67862FAB

t_1：2ACD7773 BD808842 F841D35F 87070D79 5F6AF8F3 F08C915E 760A4511 86B3F59F

在有限域 F_N 上计算 $t_2 = ks \cdot t_1^{-1}$：

t_2：291FE3CA C8F58AD2 DC462C8D 4D578A94 DAFD5624 DDC28E32 8D293668 8A86CF1A

签名私钥 $ds_A = [t_2]P_1 = (x_{ds_A}, y_{ds_A})$：

坐标 x_{ds_A}：A5702F05 CF131530 5E2D6EB6 4B0DEB92 3DB1A0BC F0CAFF90 523AC875 4AA69820
坐标 y_{ds_A}：78559A84 4411F982 5C109F5E E3F52D72 0DD01785 392A727B B1556952 B2B013D3

签名步骤中的相关值：

待签名消息 M：Chinese IBS standard
M 的 16 进制表示：4368696E 65736520 49425320 7374616E 64617264
计算群 $\mathbb{G}_T$ 中的元素 $g = e(P_1, P_{pub\text{-}s})$：
(4E378FB5 561CD066 8F906B73 1AC58FEE 25738EDF 09CADC7A 29C0ABC0 177AEA6D,
28B3404A 61908F5D 6198815C 99AF1990 C8AF3865 5930058C 28C21BB5 39CE0000,
38BFFE40 A22D529A 0C66124B 2C308DAC 92299126 56F62B4F ACFCED40 8E02380F,
A01F2C8B EE817696 09462C69 C96AA923 FD863E20 9D3CE26D D889B55E 2E3873DB,
67E0E0C2 EED7A699 3DCE28FE 9AA2EF56 83430786 0839677F 96685F2B 44D0911F,
5A1AE172 102EFD95 DF7338DB C577C66D 8D6C15E0 A0158C75 07228EFB 078F42A6,
1604A3FC FA9783E6 67CE9FCB 1062C2A5 C6685C31 6DDA62DE 0548BAA6 BA30038B,
93634F44 FA13AF76 169F3CC8 FBEA880A DAFF8475 D5FD28A7 5DEB83C4 4362B439,
B3129A75 D31D1719 4675A1BC 56947920 898FBF39 0A5BF5D9 31CE6CBB 3340F66D,
4C744E69 C4A2E1C8 ED72F796 D151A17C E2325B94 3260FC46 0B9F73CB 57C9014B,
84B87422 330D7936 EABA1109 FA5A7A71 81EE16F2 438B0AEB 2F38FD5F 7554E57A,
AAB9F06A 4EEBA432 3A7833DB 202E4E35 639D93FA 3305AF73 F0F071D7 D284FCFB)

产生随机数 r：033C86 16B06704 813203DF D0096502 2ED15975 C662337A ED648835 DC4B1CBE
计算群 $\mathbb{G}_T$ 中的元素 $w = g^r$：
(81377B8F DBC2839B 4FA2D0E0 F8AA6853 BBBE9E9C 4099608F 8612C607 8ACD7563,
815AEBA2 17AD502D A0F48704 CC73CABB 3C06209B D87142E1 4CBD99E8 BCA1680F,
30DADC5C D9E207AE E32209F6 C3CA3EC0 D800A1A4 2D33C731 53DED47C 70A39D2E,
8EAF5D17 9A1836B3 59A9D1D9 BFC19F2E FCDB8293 28620962 BD3FDF15 F2567F58,
A543D256 09AE9439 20679194 ED30328B B33FD156 60BDE485 C6B79A7B 32B01398,
3F012DB0 4BA59FE8 8DB88932 1CC2373D 4C0C35E8 4F7AB1FF 33679BCA 575D6765,
4F8624EB 435B838C CA77B2D0 347E65D5 E4696441 2A096F41 50D8C5ED E5440DDF,
0656FCB6 63D24731 E8029218 8A2471B8 B68AA993 89926849 9D23C897 55A1A897,
44643CEA D40F0965 F28E1CD2 895C3D11 8E4F65C9 A0E3E741 B6DD52C0 EE2D25F5,
898D6084 8026B7EF B8FCC1B2 442ECF07 95F8A81C EE99A624 8F294C82 C90D26BD,
6A814AAF 475F128A EF43A128 E37F8015 4AE6CB92 CAD7D150 1BAE30F7 50B3A9BD,
1F96B08E 97997363 91131470 5BFB9A9D BB97F755 53EC90FB B2DDAE53 C8F68E42)

计算 $h = H_2(M||w, N)$：
$M||w$：
4368696E 65736520 49425320 7374616E 64617264 81377B8F DBC2839B 4FA2D0E0 F8AA6853 BBBE9E9C
4099608F 8612C607 8ACD7563 815AEBA2 17AD502D A0F48704 CC73CABB 3C06209B D87142E1 4CBD99E8
BCA1680F 30DADC5C D9E207AE E32209F6 C3CA3EC0 D800A1A4 2D33C731 53DED47C 70A39D2E 8EAF5D17
9A1836B3 59A9D1D9 BFC19F2E FCDB8293 28620962 BD3FDF15 F2567F58 A543D256 09AE9439 20679194
ED30328B B33FD156 60BDE485 C6B79A7B 32B01398 3F012DB0 4BA59FE8 8DB88932 1CC2373D 4C0C35E8
4F7AB1FF 33679BCA 575D6765 4F8624EB 435B838C CA77B2D0 347E65D5 E4696441 2A096F41 50D8C5ED
E5440DDF 0656FCB6 63D24731 E8029218 8A2471B8 B68AA993 89926849 9D23C897 55A1A897 44643CEA
D40F0965 F28E1CD2 895C3D11 8E4F65C9 A0E3E741 B6DD52C0 EE2D25F5 898D6084 8026B7EF B8FCC1B2
442ECF07 95F8A81C EE99A624 8F294C82 C90D26BD 6A814AAF 475F128A EF43A128 E37F8015 4AE6CB92
CAD7D150 1BAE30F7 50B3A9BD 1F96B08E 97997363 91131470 5BFB9A9D BB97F755 53EC90FB B2DDAE53

C8F68E42

h：823C4B21 E4BD2DFE 1ED92C60 6653E996 66856315 2FC33F55 D7BFBB9B D9705ADB

计算 $l = (r - h) \bmod N$：3406F164 3496DFF8 385C82CF 5F4442B0 123E89AB AF898013 FB13AE36 D9799108

计算群 $\mathbb{G}_1$ 中的元素 $S = [l]ds_A = (x_S, y_S)$：

坐标 x_S：73BF9692 3CE58B6A D0E13E96 43A406D8 EB98417C 50EF1B29 CEF9ADB4 8B6D598C

坐标 y_S：856712F1 C2E0968A B7769F42 A99586AE D139D5B8 B3E15891 827CC2AC ED9BAA05

消息 M 的签名为(h, S)：

h： 823C4B21 E4BD2DFE 1ED92C60 6653E996 66856315 2FC33F55 D7BFBB9B D9705ADB

S：04 73BF9692 3CE58B6A D0E13E96 43A406D8 EB98417C 50EF1B29 CEF9ADB4 8B6D598C
856712F1 C2E0968A B7769F42 A99586AE D139D5B8 B3E15891 827CC2AC ED9BAA05

验证步骤中的相关值：

计算群 $\mathbb{G}_T$ 中的元素 $g = e(P_1, P_{pub\text{-}s})$：

(4E378FB5 561CD066 8F906B73 1AC58FEE 25738EDF 09CADC7A 29C0ABC0 177AEA6D ,
28B3404A 61908F5D 6198815C 99AF1990 C8AF3865 5930058C 28C21BB5 39CE0000 ,
38BFFE40 A22D529A 0C66124B 2C308DAC 92299126 56F62B4F ACFCED40 8E02380F ,
A01F2C8B EE817696 09462C69 C96AA923 FD863E20 9D3CE26D D889B55E 2E3873DB ,
67E0E0C2 EED7A699 3DCE28FE 9AA2EF56 83430786 0839677F 96685F2B 44D0911F ,
5A1AE172 102EFD95 DF7338DB C577C66D 8D6C15E0 A0158C75 07228EFB 078F42A6 ,
1604A3FC FA9783E6 67CE9FCB 1062C2A5 C6685C31 6DDA62DE 0548BAA6 BA30038B ,
93634F44 FA13AF76 169F3CC8 FBEA880A DAFF8475 D5FD28A7 5DEB83C4 4362B439 ,
B3129A75 D31D1719 4675A1BC 56947920 898FBF39 0A5BF5D9 31CE6CBB 3340F66D ,
4C744E69 C4A2E1C8 ED72F796 D151A17C E2325B94 3260FC46 0B9F73CB 57C9014B ,
84B87422 330D7936 EABA1109 FA5A7A71 81EE16F2 438B0AEB 2F38FD5F 7554E57A ,
AAB9F06A 4EEBA432 3A7833DB 202E4E35 639D93FA 3305AF73 F0F071D7 D284FCFB)

计算群 $\mathbb{G}_T$ 中的元素 $t = g^{h'}$：

(B59486D6 F3AE4649 ADF387C5 A22790E4 2B98051A 339B3403 B17B1F2B 38259EFE ,
1632C30A A86001F5 2EEFED51 7AA672D7 0F03AF3E E9197017 EDA43143 6CFBDACE ,
2F635B5B 0243F6F4 876A1D91 49EAFAB7 1060EA43 52DE6D4A 83B5F8F3 DF73EFF0 ,
3A27F33E 024339B8 3F16E58A E524A5FA A3E7FD00 9568A9FF 23752BC8 DD85B704 ,
08208E26 734BC667 31AEE530 692B3AE2 77EA70D6 BBAF8F48 5295D067 E67B3B4F ,
1DBDDD78 126E962E 950CEBB3 85C3F7A3 E0A5597F 9C3B9FB3 F5DAC3DA A85FD016 ,
189E64A3 C0A0D876 11A83AEC 8F3A3688 C0ABF2F6 4860CF33 1463ACB3 A4AABB04 ,
6E3FA26F 762D1A23 71601BE0 0DA702B1 A726273C E843D991 CE5C2EAB AB2EAC6F ,
A5BCFFD5 40EE56B5 A26CCDA5 66FD8ABC 3615CB7D EA8F240E 0BF46158 16C2B23E ,
A074A0AA 62A26C28 3F11543C ECDEA524 2113FE2E 982CCBDA 2D495EF6 C05550A6 ,
2E3F160C 96C16059 5A1034B5 15692066 8A7BEE5E 82E0B8BE 06963FDD BDEB5AAE ,
0DCF9EA2 8617B596 5313B917 D556DA0D 3A557C41 12CE1C4A 06B327D7 DC18273D)

计算 $h_1 = H_1(ID_A \| hid, N)$：

$ID_A \| hid$：416C6963 6501

h_1：2ACC468C 3926B0BD B2767E99 FF26E084 DE9CED8D BC7D5FBF 418027B6 67862FAB

计算群 $\mathbb{G}_2$ 中的元素 $P = [h_1]P_2 + P_{pub\text{-}s} = (x_P, y_P)$：

坐标 x_P：(511F2C82 3C7484DD FC16BBC5 3AAD33B7 8D2429AF CF7F8AD8 B72261B4 E1FFCF79 ,
7B234E1D 623A172A AA89164A F3E828B4 D0E49CE6 EC5C7FE9 2E657272 250CBAF6)

坐标 y_P：(4831DD31 3EC39FDA 59F3E14F EBCFF784 8D11875D 805662D2 6969CF70 5D46ED70 ,
73B542A6 9058F460 1AC19F23 72036863 68FEC436 C13C2B07 61F9F9B6 E14A36E4)

计算群 $\mathbb{G}_T$ 中的元素 $u = e(S', P)$：

(A97A1713 04A0316F C8BA21B9 11289C43 71E73B7D 2163AC5B 44F3B525 88EB69A1 ,
1838972B F0CA86E1 7147468A 869A3261 FCC27993 AA50E367 27918ED5 ABD71C0C ,
291663C4 9DF9B4A8 2B122412 B749BF14 4341F2E2 25645061 45E0B771 73496F50 ,
ABB3B115 E006FAE8 EC3CB133 F411DF05 B32CFA15 7716082D EEDF7BDB 188966DF ,
5FCC7DBD FC714FC8 989E0331 83814227 5EAE6B63 09BAD1DE FE28263A D66E6780 ,
48697F5C 62EE4342 325A9EF0 3775A52F 1C0B9D5F B08D99E8 D65A436B 8A9AF05E ,
5C53DC7E 4D8A0B75 57920B21 FA5F2E75 B38C4445 F0CF9153 AC412724 0530F5D5 ,
01BBD7B3 4565F80C CB452809 3CE9FAFD F6AD84FD 620F3B5B C324DA19 BB665151 ,
4AE8D623 18D2BA35 F9494189 100BCD82 F1B1399B 0B148677 00D3D7A2 43D02D3A ,
701409A6 6ED452DE C4586735 CF363137 9501DC75 6466F6F1 8E3BC002 722531AE ,
7B9A10CE B34F1195 6A04E306 4663D87B 844B452C 3D81C91A 8223938D 1A9ABBC4 ,
753A274B 8E9E35AF 503B7C2E 39ABB32B C8674FC8 EC012D8B EBDFFF2F E0985F85)

计算群 $\mathbb{G}_T$ 中的元素 $w' = u \cdot t$：

(81377B8F DBC2839B 4FA2D0E0 F8AA6853 BBBE9E9C 4099608F 8612C607 8ACD7563 ,
815AEBA2 17AD502D A0F48704 CC73CABB 3C06209B D87142E1 4CBD99E8 BCA1680F ,
30DADC5C D9E207AE E32209F6 C3CA3EC0 D800A1A4 2D33C731 53DED47C 70A39D2E ,
8EAF5D17 9A1836B3 59A9D1D9 BFC19F2E FCDB8293 28620962 BD3FDF15 F2567F58 ,
A543D256 09AE9439 20679194 ED30328B B33FD156 60BDE485 C6B79A7B 32B01398 ,
3F012DB0 4BA59FE8 8DB88932 1CC2373D 4C0C35E8 4F7AB1FF 33679BCA 575D6765 ,
4F8624EB 435B838C CA77B2D0 347E65D5 E4696441 2A096F41 50D8C5ED E5440DDF ,
0656FCB6 63D24731 E8029218 8A2471B8 B68AA993 89926849 9D23C897 55A1A897 ,
44643CEA D40F0965 F28E1CD2 895C3D11 8E4F65C9 A0E3E741 B6DD52C0 EE2D25F5 ,
898D6084 8026B7EF B8FCC1B2 442ECF07 95F8A81C EE99A624 8F294C82 C90D26BD ,
6A814AAF 475F128A EF43A128 E37F8015 4AE6CB92 CAD7D150 1BAE30F7 50B3A9BD ,
1F96B08E 97997363 91131470 5BFB9A9D BB97F755 53EC90FB B2DDAE53 C8F68E42)

计算 $h_2 = H_2(M' \| w', N)$：

$M' \| w'$：

4368696E 65736520 49425320 7374616E 64617264 81377B8F DBC2839B 4FA2D0E0 F8AA6853 BBBE9E9C
4099608F 8612C607 8ACD7563 815AEBA2 17AD502D A0F48704 CC73CABB 3C06209B D87142E1 4CBD99E8
BCA1680F 30DADC5C D9E207AE E32209F6 C3CA3EC0 D800A1A4 2D33C731 53DED47C 70A39D2E 8EAF5D17
9A1836B3 59A9D1D9 BFC19F2E FCDB8293 28620962 BD3FDF15 F2567F58 A543D256 09AE9439 20679194
ED30328B B33FD156 60BDE485 C6B79A7B 32B01398 3F012DB0 4BA59FE8 8DB88932 1CC2373D 4C0C35E8
4F7AB1FF 33679BCA 575D6765 4F8624EB 435B838C CA77B2D0 347E65D5 E4696441 2A096F41 50D8C5ED
E5440DDF 0656FCB6 63D24731 E8029218 8A2471B8 B68AA993 89926849 9D23C897 55A1A897 44643CEA
D40F0965 F28E1CD2 895C3D11 8E4F65C9 A0E3E741 B6DD52C0 EE2D25F5 898D6084 8026B7EF B8FCC1B2
442ECF07 95F8A81C EE99A624 8F294C82 C90D26BD 6A814AAF 475F128A EF43A128 E37F8015 4AE6CB92
CAD7D150 1BAE30F7 50B3A9BD 1F96B08E 97997363 91131470 5BFB9A9D BB97F755 53EC90FB B2DDAE53
C8F68E42

h_2：823C4B21 E4BD2DFE 1ED92C60 6653E996 66856315 2FC33F55 D7BFBB9B D9705ADB

$h_2 = h$, 验证通过！

附 录 B
密钥交换协议示例

B.1 一般要求

本附录选用 SM3 算法的密码杂凑函数，其输入是长度小于 2^{64} 的消息比特串，输出是长度为 256 比特的杂凑值，记为 $H_{256}(\ )$。

本附录中，所有用16进制表示的数，左边为高位，右边为低位。

B.2 密钥交换

椭圆曲线方程为：$y^2 = x^3 + b$

基域特征 q：B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27 E351457D

方程参数 b：05

群 $\mathbb{G}_1$,$\mathbb{G}_2$ 的阶 N：B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C D69ECF25

余因子 cf：1

嵌入次数 k：12

扭曲线的参数 β : $\sqrt{-2}$

群 $\mathbb{G}_1$ 的生成元 $P_1 = (x_{P1}, y_{P1})$：

坐标 x_{P1}：93DE051D 62BF718F F5ED0704 487D01D6 E1E40869 09DC3280 E8C4E481 7C66DDDD

坐标 y_{P1}：21FE8DDA 4F21E607 63106512 5C395BBC 1C1C00CB FA602435 0C464CD7 0A3EA616

群 $\mathbb{G}_2$ 的生成元 $P_2 = (x_{P2}, y_{P2})$：

坐标 x_{P2}：（85AEF3D0 78640C98 597B6027 B441A01F F1DD2C19 0F5E93C4 54806C11 D8806141 ，
37227552 92130B08 D2AAB97F D34EC120 EE265948 D19C17AB F9B7213B AF82D65B ）

坐标 y_{P2}：（17509B09 2E845C12 66BA0D26 2CBEE6ED 0736A96F A347C8BD 856DC76B 84EBEB96 ，
A7CF28D5 19BE3DA6 5F317015 3D278FF2 47EFBA98 A71A0811 6215BBA5 C999A7C7 ）

双线性对的识别符 eid：0x04

加密主密钥和用户加密密钥产生过程中的相关值：

加密主私钥 ke：02E65B 0762D042 F51F0D23 542B13ED 8CFA2E9A 0E720636 1E013A28 3905E31F

加密主公钥 $P_{pub\text{-}e} = [ke]P_1 = (x_{P_{pub\text{-}e}}, y_{P_{pub\text{-}e}})$：

坐标 $x_{P_{pub\text{-}e}}$：91745426 68E8F14A B273C094 5C3690C6 6E5DD096 78B86F73 4C435056 7ED06283

坐标 $y_{P_{pub\text{-}e}}$：54E598C6 BF749A3D ACC9FFFE DD9DB686 6C50457C FC7AA2A4 AD65C316 8FF74210

加密私钥生成函数识别符 hid：0x02

实体 A 的标识 ID_A：Alice

ID_A 的 16 进制表示：416C6963 65

在有限域 F_N 上计算 $t_1 = H_1(ID_A \| hid, N) + ke$：

$ID_A \| hid$：416C6963 6502

$H_1(ID_A \| hid, N)$：A9AC0FDA 7380ED8E 3325FDDC D40A7221 E3CD72F6 FFA7F27D 54AD494C EDB4E212

t_1：A9AEF635 7AE3BDD1 28450B00 2835860F 70C7A191 0E19F8B3 72AE8375 26BAC531

在有限域 F_N 上计算 $t_2 = ke \cdot t_1^{-1}$：

t_2：607CD136 1FBEA46F F5F89A0B A0C6D246 2D080452 AD2EA22F AF9FB48C AB47ECBD

计算 $de_A = [t_2]P_2 = (x_{de_A}, y_{de_A})$：

坐标 x_{de_A}：（0FE8EAB3 95199B56 BF1D75BD 2CD610B6 424F08D1 092922C5 882B52DC D6CA832A ，
7DA57BC5 0241F9E5 BFDDC075 DD9D32C7 777100D7 36916CFC 165D8D36 E0634CD7 ）

坐标 y_{de_A}：（83A457DA F52CAD46 4C903B26 062CAF93 7BB40E37 DADED9ED A401050E 49C8AD0C ，

6970876B 9AAD1B7A 50BB4863 A11E574A F1FE3C59 75161D73 DE4C3AF6 21FB1EFB ）

实体 B 的标识 ID_B：Bob

ID_B 的 16 进制表示：426F62

在有限域 F_N 上计算 $t_3=H_1(ID_B\|\ hid,N)+ke$：

$ID_B\|\ hid$：426F6202

$H_1(ID_B\|\ hid,N)$：56AF6EF1 D2AB38F1 EE77A5D5 38DD33B4 4917F2D9 AD6AB68A 993B36C7 27ED9838

t_3：56B2554C DA0E0934 E396B2F8 8D0847A1 D6122173 BBDCBCC0 B73C70EF 60F37B57

在有限域 F_N 上计算 $t_4=ke\cdot t_3^{-1}$ ：

t_4：372C4846 2D0F0380 6B32E010 CFB1E0F6 98F50E47 2BAABF26 7D38252F 6BE5960A

计算 $de_B=[t_4]P_2=(x_{deB}, y_{deB})$：

坐标 x_{deB}：（74CCC3AC 9C383C60 AF083972 B96D05C7 5F12C890 7D128A17 ADAFBAB8 C5A4ACF7 ，
01092FF4 DE893626 70C21711 B6DBE52D CD5F8E40 C6654B3D ECE573C2 AB3D29B2 ）

坐标 y_{deB}：（44B0294A A04290E1 524FF3E3 DA8CFD43 2BB64DE3 A8040B5B 88D1B5FC 86A4EBC1 ，
8CFC48FB 4FF37F1E 27727464 F3C34E21 53861AD0 8E972D16 25FC1A7B D18D5539 ）

交换密钥的长度 $klen$：0x80

密钥交换步骤 A1-A4 中的相关值：

计算 $Q_B=[H_1(ID_B\|hid, N)]P_1+P_{pub-e}=（x_{QB}, y_{QB}）$：

$ID_B\|\ hid$：426F6202

$H_1(ID_B\|hid,N)$：56AF6EF1 D2AB38F1 EE77A5D5 38DD33B4 4917F2D9 AD6AB68A 993B36C7 27ED9838

坐标 x_{QB}：A1C5EA63 AE85302B 026C2EE8 6DC7E880 2CE30830 61571FC9 8747011C E088BBD7

坐标 y_{QB}：635385A8 F01C8E73 720CA4AD 5DE81258 10B6271C 84B27EC6 EAB182C6 266E4DA2

取 r_A 为：5879 DD1D51E1 75946F23 B1B41E93 BA31C584 AE59A426 EC1046A4 D03B06C8

计算 $R_A=[r_A]Q_B=(x_{RA}, y_{RA})$：

坐标 x_{RA}：7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80 9CF23B6D 964BB265

坐标 y_{RA}：A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599

密钥交换步骤 B1-B7 中的相关值：

计算 $Q_A=[H_1(ID_A\|\ hid, N)]P_1+P_{pub-e}=(x_{QA}, y_{QA})$：

$ID_A\|\ hid$：416C6963 6502

$H_1(ID_A\|\ hid,N)$：A9AC0FDA 7380ED8E 3325FDDC D40A7221 E3CD72F6 FFA7F27D 54AD494C EDB4E212

坐标 x_{QA}：66C68126 E6C3E197 69A203C0 C3275CF9 121A4A11 6D7851DA 9A702A3E 14F679DD

坐标 y_{QA}：52AF31F2 45EB74CD E62F99A2 B557B621 9C53C3F3 BA7B21E1 FDC62EA4 BCFF9795

取 r_B 为：018B98 C44BEF9F 8537FB7D 071B2C92 8B3BC65B D3D69E1E EE213564 905634FE

计算 $R_B=[r_B]Q_A=(x_{RB}, y_{RB})$：

坐标 x_{RB}：861E9148 5FB7623D 2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482

坐标 y_{RB}：32D906A4 69EBC121 6A802A70 52D5617C D430FB56 FBA729D4 1D9BD668 E9EB9600

计算 $g_1=e(R_A, de_B)$：

(28542FB6 954C84BE 6A5F2988 A31CB681 7BA07819 66FA83D9 673A9577 D3C0C134,
5E27C19F C02ED9AE 37F5BB7B E9C03C2B 87DE0275 39CCF03E 6B7D36DE 4AB45CD1,
A1ABFCD3 0C57DB0F 1A838E3A 8F2BF823 479C978B D1372305 06EA6249 C891049E,
34974779 13AB89F5 E2960F38 2B1B5C8E E09DE0FA 498BA95C 4409D630 D343DA40,
4FEC9347 2DA33A4D B6599095 C0CF895E 3A7B993E E5E4EBE3 B9AB7D7D 5FF2A3D1,
647BA154 C3E8E185 DFC33657 C1F128D4 80F3F7E3 F1680120 8029E194 34C733BB,
73F21693 C66FC237 24DB2638 0C526223 C705DAF6 BA18B763 A68623C8 6A632B05,
0F63A071 A6D62EA4 5B59A194 2DFF5335 D1A232C9 C5664FAD 5D6AF54C 11418B0D,

8C8E9D8D 905780D5 0E779067 F2C4B1C8 F83A8B59 D735BB52 AF35F567 30BDE5AC,
861CCD99 78617267 CE4AD978 9F77739E 62F2E57B 48C2FF26 D2E90A79 A1D86B93,
9B1CA08F 64712E33 AEDA3F44 BD6CB633 E0F72221 1E344D73 EC9BBEBC 92142765,
6BA584CE 742A2A3A B41C15D3 EF94EDEB 8EF74A2B DCDAAECC 09ABA567 981F6437)

计算 $g_2 = e(P_{pub-e}, P_2)^{r_B}$:

(1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492,
5FFEB92A D870F97D C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7,
2C5C3B37 E4F2FF83 DB33D98C 0317BCBB BBF4AC6D F6B89ECA 58268B28 0045E612,
6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C F9B43C78 434AEC38,
0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D,
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D,
00DD2B74 16BAA911 72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5,
7EBAC034 9F854446 9E60C32F 6075FB04 68A68147 FF013537 DF792FFC E024F857,
10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6 D4B651B6 4F3A3A5E,
58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859,
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79,
934FDDA6 D3AB48C8 571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6)

计算 $g_3 = g_1^{r_B}$:

(A76B6777 AD87C912 4C7D7065 F74808DB 2E80371C 70471580 B0C7C457 A79EA5E7,
242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332 4B3BDB4C 682BF9B2,
0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204,
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2,
ADC269D1 B6233258 2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01,
B1ED0650 2333B2AB 1AE697EA 34F2EF8C 6E47B043 1831706C B5AFCD75 754FA795,
28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6 6F388ED6 644AF851,
885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827,
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F,
4A40AC8F C5B7168F A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133,
35D89EAE B36F4D31 BB671306 4CDA8835 E2AA4529 F4212932 7C6F7E8A B760654D,
58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9 A17C9D11 A5A6B148)

计算 $SK_B = KDF(ID_A \| ID_B \| R_A \| R_B \| g_1 \| g_2 \| g_3, klen)$:

$ID_A \| ID_B \| R_A \| R_B \| g_1 \| g_2 \| g_3$:

416C6963 65426F62 7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80 9CF23B6D 964BB265
A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599 861E9148 5FB7623D
2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482 32D906A4 69EBC121 6A802A70 52D5617C
D430FB56 FBA729D4 1D9BD668 E9EB9600 28542FB6 954C84BE 6A5F2988 A31CB681 7BA07819 66FA83D9
673A9577 D3C0C134 5E27C19F C02ED9AE 37F5BB7B E9C03C2B 87DE0275 39CCF03E 6B7D36DE 4AB45CD1
A1ABFCD3 0C57DB0F 1A838E3A 8F2BF823 479C978B D1372305 06EA6249 C891049E 34974779 13AB89F5
E2960F38 2B1B5C8E E09DE0FA 498BA95C 4409D630 D343DA40 4FEC9347 2DA33A4D B6599095 C0CF895E
3A7B993E E5E4EBE3 B9AB7D7D 5FF2A3D1 647BA154 C3E8E185 DFC33657 C1F128D4 80F3F7E3 F1680120
8029E194 34C733BB 73F21693 C66FC237 24DB2638 0C526223 C705DAF6 BA18B763 A68623C8 6A632B05
0F63A071 A6D62EA4 5B59A194 2DFF5335 D1A232C9 C5664FAD 5D6AF54C 11418B0D 8C8E9D8D 905780D5
0E779067 F2C4B1C8 F83A8B59 D735BB52 AF35F567 30BDE5AC 861CCD99 78617267 CE4AD978 9F77739E
62F2E57B 48C2FF26 D2E90A79 A1D86B93 9B1CA08F 64712E33 AEDA3F44 BD6CB633 E0F72221 1E344D73
EC9BBEBC 92142765 6BA584CE 742A2A3A B41C15D3 EF94EDEB 8EF74A2B DCDAAECC 09ABA567 981F6437
1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492 5FFEB92A D870F97D
C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7 2C5C3B37 E4F2FF83 DB33D98C 0317BCBB

BBF4AC6D F6B89ECA 58268B28 0045E612 6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C
F9B43C78 434AEC38 0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D 00DD2B74 16BAA911
72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5 7EBAC034 9F854446 9E60C32F 6075FB04
68A68147 FF013537 DF792FFC E024F857 10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6
D4B651B6 4F3A3A5E 58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79 934FDDA6 D3AB48C8
571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6 A76B6777 AD87C912 4C7D7065 F74808DB
2E80371C 70471580 B0C7C457 A79EA5E7 242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332
4B3BDB4C 682BF9B2 0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2 ADC269D1 B6233258
2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01 B1ED0650 2333B2AB 1AE697EA 34F2EF8C
6E47B043 1831706C B5AFCD75 754FA795 28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6
6F388ED6 644AF851 885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F 4A40AC8F C5B7168F
A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133 35D89EAE B36F4D31 BB671306 4CDA8835
E2AA4529 F4212932 7C6F7E8A B760654D 58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9
A17C9D11 A5A6B148

SK_B：C5C13A8F 59A97CDE AE64F16A 2272A9E7

计算选项 $S_B=Hash(0x82\|g_1\|Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B))$ ：

$g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B$ ：

1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492 5FFEB92A D870F97D
C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7 2C5C3B37 E4F2FF83 DB33D98C 0317BCBB
BBF4AC6D F6B89ECA 58268B28 0045E612 6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C
F9B43C78 434AEC38 0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D 00DD2B74 16BAA911
72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5 7EBAC034 9F854446 9E60C32F 6075FB04
68A68147 FF013537 DF792FFC E024F857 10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6
D4B651B6 4F3A3A5E 58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79 934FDDA6 D3AB48C8
571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6 A76B6777 AD87C912 4C7D7065 F74808DB
2E80371C 70471580 B0C7C457 A79EA5E7 242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332
4B3BDB4C 682BF9B2 0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2 ADC269D1 B6233258
2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01 B1ED0650 2333B2AB 1AE697EA 34F2EF8C
6E47B043 1831706C B5AFCD75 754FA795 28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6
6F388ED6 644AF851 885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F 4A40AC8F C5B7168F
A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133 35D89EAE B36F4D31 BB671306 4CDA8835
E2AA4529 F4212932 7C6F7E8A B760654D 58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9
A17C9D11 A5A6B148 416C6963 65426F62 7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80
9CF23B6D 964BB265 A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599
861E9148 5FB7623D 2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482 32D906A4 69EBC121
6A802A70 52D5617C D430FB56 FBA729D4 1D9BD668 E9EB9600

$Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B)$：72747133 56B7479A 2D592732 0E6B888A FC4D1769 66FF841D 8F480AF0 479BFDB7

0x82$\|g_1\|Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B)$：

8228542F B6954C84 BE6A5F29 88A31CB6 817BA078 1966FA83 D9673A95 77D3C0C1 345E27C1 9FC02ED9 AE37F5BB 7BE9C03C 2B87DE02 7539CCF0 3E6B7D36 DE4AB45C D1A1ABFC D30C57DB 0F1A838E 3A8F2BF8 23479C97 8BD13723 0506EA62 49C89104 9E349747 7913AB89 F5E2960F 382B1B5C 8EE09DE0 FA498BA9 5C4409D6 30D343DA 404FEC93 472DA33A 4DB65990 95C0CF89 5E3A7B99 3EE5E4EB E3B9AB7D 7D5FF2A3 D1647BA1 54C3E8E1 85DFC336 57C1F128 D480F3F7 E3F16801 208029E1 9434C733 BB73F216 93C66FC2 3724DB26 380C5262 23C705DA F6BA18B7 63A68623 C86A632B 050F63A0 71A6D62E A45B59A1 942DFF53 35D1A232 C9C5664F AD5D6AF5 4C11418B 0D8C8E9D 8D905780 D50E7790 67F2C4B1 C8F83A8B 59D735BB 52AF35F5 6730BDE5 AC861CCD 99786172 67CE4AD9 789F7773 9E62F2E5 7B48C2FF 26D2E90A 79A1D86B 939B1CA0 8F64712E 33AEDA3F 44BD6CB6 33E0F722 211E344D 73EC9BBE BC921427 656BA584 CE742A2A 3AB41C15 D3EF94ED EB8EF74A 2BDCDAAE CC09ABA5 67981F64 37727471 3356B747 9A2D5927 320E6B88 8AFC4D17 6966FF84 1D8F480A F0479BFD B7

选项 S_B：3BB4BCEE 8139C960 B4D6566D B1E0D5F0 B2767680 E5E1BF93 4103E6C6 6E40FFEE

密钥交换步骤 A5-A8 中的相关值：

计算 $g_1' = e(P_{pub\text{-}e}, P_2)^{r_A}$：

(28542FB6 954C84BE 6A5F2988 A31CB681 7BA07819 66FA83D9 673A9577 D3C0C134,
5E27C19F C02ED9AE 37F5BB7B E9C03C2B 87DE0275 39CCF03E 6B7D36DE 4AB45CD1,
A1ABFCD3 0C57DB0F 1A838E3A 8F2BF823 479C978B D1372305 06EA6249 C891049E,
34974779 13AB89F5 E2960F38 2B1B5C8E E09DE0FA 498BA95C 4409D630 D343DA40,
4FEC9347 2DA33A4D B6599095 C0CF895E 3A7B993E E5E4EBE3 B9AB7D7D 5FF2A3D1,
647BA154 C3E8E185 DFC33657 C1F128D4 80F3F7E3 F1680120 8029E194 34C733BB,
73F21693 C66FC237 24DB2638 0C526223 C705DAF6 BA18B763 A68623C8 6A632B05,
0F63A071 A6D62EA4 5B59A194 2DFF5335 D1A232C9 C5664FAD 5D6AF54C 11418B0D,
8C8E9D8D 905780D5 0E779067 F2C4B1C8 F83A8B59 D735BB52 AF35F567 30BDE5AC,
861CCD99 78617267 CE4AD978 9F77739E 62F2E57B 48C2FF26 D2E90A79 A1D86B93,
9B1CA08F 64712E33 AEDA3F44 BD6CB633 E0F72221 1E344D73 EC9BBEBC 92142765,
6BA584CE 742A2A3A B41C15D3 EF94EDEB 8EF74A2B DCDAAECC 09ABA567 981F6437)

计算 $g_2' = e(R_B, de_A)$：

(1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492,
5FFEB92A D870F97D C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7,
2C5C3B37 E4F2FF83 DB33D98C 0317BCBB BBF4AC6D F6B89ECA 58268B28 0045E612,
6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C F9B43C78 434AEC38,
0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D,
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D,
00DD2B74 16BAA911 72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5,
7EBAC034 9F854446 9E60C32F 6075FB04 68A68147 FF013537 DF792FFC E024F857,
10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6 D4B651B6 4F3A3A5E,
58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859,
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79,
934FDDA6 D3AB48C8 571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6)

计算 $g_3' = (g_2')^{r_A}$：

(A76B6777 AD87C912 4C7D7065 F74808DB 2E80371C 70471580 B0C7C457 A79EA5E7,
242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332 4B3BDB4C 682BF9B2,
0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204,
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2,

ADC269D1 B6233258 2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01,
B1ED0650 2333B2AB 1AE697EA 34F2EF8C 6E47B043 1831706C B5AFCD75 754FA795,
28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6 6F388ED6 644AF851,
885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827,
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F,
4A40AC8F C5B7168F A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133,
35D89EAE B36F4D31 BB671306 4CDA8835 E2AA4529 F4212932 7C6F7E8A B760654D,
58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9 A17C9D11 A5A6B148)

计算选项 $S_1=Hash(0x82\|g_1'\|Hash(g_2'\|g_3'\|ID_A\|ID_B\|R_A\|R_B))$：

$g_2'\|g_3'\|ID_A\|ID_B\|R_A\|R_B$：

1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492 5FFEB92A D870F97D
C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7 2C5C3B37 E4F2FF83 DB33D98C 0317BCBB
BBF4AC6D F6B89ECA 58268B28 0045E612 6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C
F9B43C78 434AEC38 0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D 00DD2B74 16BAA911
72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5 7EBAC034 9F854446 9E60C32F 6075FB04
68A68147 FF013537 DF792FFC E024F857 10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6
D4B651B6 4F3A3A5E 58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79 934FDDA6 D3AB48C8
571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6 A76B6777 AD87C912 4C7D7065 F74808DB
2E80371C 70471580 B0C7C457 A79EA5E7 242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332
4B3BDB4C 682BF9B2 0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2 ADC269D1 B6233258
2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01 B1ED0650 2333B2AB 1AE697EA 34F2EF8C
6E47B043 1831706C B5AFCD75 754FA795 28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6
6F388ED6 644AF851 885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F 4A40AC8F C5B7168F
A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133 35D89EAE B36F4D31 BB671306 4CDA8835
E2AA4529 F4212932 7C6F7E8A B760654D 58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9
A17C9D11 A5A6B148 416C6963 65426F62 7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80
9CF23B6D 964BB265 A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599
861E9148 5FB7623D 2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482 32D906A4 69EBC121
6A802A70 52D5617C D430FB56 FBA729D4 1D9BD668 E9EB9600

$Hash(g_2'\|g_3'\|ID_A\|ID_B\|R_A\|R_B)$： 72747133 56B7479A 2D592732 0E6B888A FC4D1769 66FF841D
8F480AF0 479BFDB7

$0x82\|g_1'\|Hash(g_2'\|g_3'\|ID_A\|ID_B\|R_A\|R_B)$：

8228542F B6954C84 BE6A5F29 88A31CB6 817BA078 1966FA83 D9673A95 77D3C0C1 345E27C1 9FC02ED9
AE37F5BB 7BE9C03C 2B87DE02 7539CCF0 3E6B7D36 DE4AB45C D1A1ABFC D30C57DB 0F1A838E 3A8F2BF8
23479C97 8BD13723 0506EA62 49C89104 9E349747 7913AB89 F5E2960F 382B1B5C 8EE09DE0 FA498BA9
5C4409D6 30D343DA 404FEC93 472DA33A 4DB65990 95C0CF89 5E3A7B99 3EE5E4EB E3B9AB7D 7D5FF2A3
D1647BA1 54C3E8E1 85DFC336 57C1F128 D480F3F7 E3F16801 208029E1 9434C733 BB73F216 93C66FC2
3724DB26 380C5262 23C705DA F6BA18B7 63A68623 C86A632B 050F63A0 71A6D62E A45B59A1 942DFF53
35D1A232 C9C5664F AD5D6AF5 4C11418B 0D8C8E9D 8D905780 D50E7790 67F2C4B1 C8F83A8B 59D735BB
52AF35F5 6730BDE5 AC861CCD 99786172 67CE4AD9 789F7773 9E62F2E5 7B48C2FF 26D2E90A 79A1D86B

939B1CA0 8F64712E 33AEDA3F 44BD6CB6 33E0F722 211E344D 73EC9BBE BC921427 656BA584 CE742A2A
3AB41C15 D3EF94ED EB8EF74A 2BDCDAAE CC09ABA5 67981F64 37727471 3356B747 9A2D5927 320E6B88
8AFC4D17 6966FF84 1D8F480A F0479BFD B7

选项 S_1：3BB4BCEE 8139C960 B4D6566D B1E0D5F0 B2767680 E5E1BF93 4103E6C6 6E40FFEE

计算 $SK_A = KDF(ID_A \| ID_B \| R_A \| R_B \| g_1' \| g_2' \| g_3', klen)$：

$ID_A \| ID_B \| R_A \| R_B \| g_1' \| g_2' \| g_3'$：

416C6963 65426F62 7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80 9CF23B6D 964BB265
A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599 861E9148 5FB7623D
2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482 32D906A4 69EBC121 6A802A70 52D5617C
D430FB56 FBA729D4 1D9BD668 E9EB9600 28542FB6 954C84BE 6A5F2988 A31CB681 7BA07819 66FA83D9
673A9577 D3C0C134 5E27C19F C02ED9AE 37F5BB7B E9C03C2B 87DE0275 39CCF03E 6B7D36DE 4AB45CD1
A1ABFCD3 0C57DB0F 1A838E3A 8F2BF823 479C978B D1372305 06EA6249 C891049E 34974779 13AB89F5
E2960F38 2B1B5C8E E09DE0FA 498BA95C 4409D630 D343DA40 4FEC9347 2DA33A4D B6599095 C0CF895E
3A7B993E E5E4EBE3 B9AB7D7D 5FF2A3D1 647BA154 C3E8E185 DFC33657 C1F128D4 80F3F7E3 F1680120
8029E194 34C733BB 73F21693 C66FC237 24DB2638 0C526223 C705DAF6 BA18B763 A68623C8 6A632B05
0F63A071 A6D62EA4 5B59A194 2DFF5335 D1A232C9 C5664FAD 5D6AF54C 11418B0D 8C8E9D8D 905780D5
0E779067 F2C4B1C8 F83A8B59 D735BB52 AF35F567 30BDE5AC 861CCD99 78617267 CE4AD978 9F77739E
62F2E57B 48C2FF26 D2E90A79 A1D86B93 9B1CA08F 64712E33 AEDA3F44 BD6CB633 E0F72221 1E344D73
EC9BBEBC 92142765 6BA584CE 742A2A3A B41C15D3 EF94EDEB 8EF74A2B DCDAAECC 09ABA567 981F6437
1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492 5FFEB92A D870F97D
C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7 2C5C3B37 E4F2FF83 DB33D98C 0317BCBB
BBF4AC6D F6B89ECA 58268B28 0045E612 6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C
F9B43C78 434AEC38 0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D 00DD2B74 16BAA911
72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5 7EBAC034 9F854446 9E60C32F 6075FB04
68A68147 FF013537 DF792FFC E024F857 10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6
D4B651B6 4F3A3A5E 58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79 934FDDA6 D3AB48C8
571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6 A76B6777 AD87C912 4C7D7065 F74808DB
2E80371C 70471580 B0C7C457 A79EA5E7 242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332
4B3BDB4C 682BF9B2 0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2 ADC269D1 B6233258
2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01 B1ED0650 2333B2AB 1AE697EA 34F2EF8C
6E47B043 1831706C B5AFCD75 754FA795 28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6
6F388ED6 644AF851 885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F 4A40AC8F C5B7168F
A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133 35D89EAE B36F4D31 BB671306 4CDA8835
E2AA4529 F4212932 7C6F7E8A B760654D 58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9
A17C9D11 A5A6B148

SK_A：C5C13A8F 59A97CDE AE64F16A 2272A9E7

计算选项 $S_A = Hash(0x83 \| g_1' \| Hash(g_2' \| g_3' \| ID_A \| ID_B \| R_A \| R_B))$：

$g_2' \| g_3' \| ID_A \| ID_B \| R_A \| R_B$ ：

1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492 5FFEB92A D870F97D
C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7 2C5C3B37 E4F2FF83 DB33D98C 0317BCBB
BBF4AC6D F6B89ECA 58268B28 0045E612 6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C
F9B43C78 434AEC38 0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D 00DD2B74 16BAA911

72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5 7EBAC034 9F854446 9E60C32F 6075FB04
68A68147 FF013537 DF792FFC E024F857 10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6
D4B651B6 4F3A3A5E 58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859
2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79 934FDDA6 D3AB48C8
571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6 A76B6777 AD87C912 4C7D7065 F74808DB
2E80371C 70471580 B0C7C457 A79EA5E7 242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332
4B3BDB4C 682BF9B2 0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2 ADC269D1 B6233258
2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01 B1ED0650 2333B2AB 1AE697EA 34F2EF8C
6E47B043 1831706C B5AFCD75 754FA795 28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6
6F388ED6 644AF851 885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F 4A40AC8F C5B7168F
A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133 35D89EAE B36F4D31 BB671306 4CDA8835
E2AA4529 F4212932 7C6F7E8A B760654D 58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9
A17C9D11 A5A6B148 416C6963 65426F62 7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80
9CF23B6D 964BB265 A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599
861E9148 5FB7623D 2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482 32D906A4 69EBC121
6A802A70 52D5617C D430FB56 FBA729D4 1D9BD668 E9EB9600

$Hash(g_2' \| g_3' \| ID_A \| ID_B \| R_A \| R_B)$:72747133 56B7479A 2D592732 0E6B888A FC4D1769 66FF841D
8F480AF0 479BFDB7

$0x83 \| g_1' \| Hash(g_2' \| g_3' \| ID_A \| ID_B \| R_A \| R_B)$:
8328542F B6954C84 BE6A5F29 88A31CB6 817BA078 1966FA83 D9673A95 77D3C0C1 345E27C1 9FC02ED9
AE37F5BB 7BE9C03C 2B87DE02 7539CCF0 3E6B7D36 DE4AB45C D1A1ABFC D30C57DB 0F1A838E 3A8F2BF8
23479C97 8BD13723 0506EA62 49C89104 9E349747 7913AB89 F5E2960F 382B1B5C 8EE09DE0 FA498BA9
5C4409D6 30D343DA 404FEC93 472DA33A 4DB65990 95C0CF89 5E3A7B99 3EE5E4EB E3B9AB7D 7D5FF2A3
D1647BA1 54C3E8E1 85DFC336 57C1F128 D480F3F7 E3F16801 208029E1 9434C733 BB73F216 93C66FC2
3724DB26 380C5262 23C705DA F6BA18B7 63A68623 C86A632B 050F63A0 71A6D62E A45B59A1 942DFF53
35D1A232 C9C5664F AD5D6AF5 4C11418B 0D8C8E9D 8D905780 D50E7790 67F2C4B1 C8F83A8B 59D735BB
52AF35F5 6730BDE5 AC861CCD 99786172 67CE4AD9 789F7773 9E62F2E5 7B48C2FF 26D2E90A 79A1D86B
939B1CA0 8F64712E 33AEDA3F 44BD6CB6 33E0F722 211E344D 73EC9BBE BC921427 656BA584 CE742A2A
3AB41C15 D3EF94ED EB8EF74A 2BDCDAAE CC09ABA5 67981F64 37727471 3356B747 9A2D5927 320E6B88
8AFC4D17 6966FF84 1D8F480A F0479BFD B7

选项 S_A：195D1B72 56BA7E0E 67C71202 A25F8C94 FF824170 2C2F55D6 13AE1C6B 98215172

密钥交换步骤 B8 中的相关值：

计算选项 $S_2 = Hash(0x83 \| g_1 \| Hash(g_2 \| g_3 \| ID_A \| ID_B \| R_A \| R_B))$：

$g_2 \| g_3 \| ID_A \| ID_B \| R_A \| R_B$ ：
1052D6E9 D13E3819 09DFF7B2 B41E13C9 87D0A906 8423B769 480DACCE 6A06F492 5FFEB92A D870F97D
C0893114 DA22A44D BC9E7A8B 6CA31A0C F0467265 A1FB48C7 2C5C3B37 E4F2FF83 DB33D98C 0317BCBB
BBF4AC6D F6B89ECA 58268B28 0045E612 6CED9E2D 7C9CD3D5 AD630DEF AB0B8315 06218037 EE0F861C
F9B43C78 434AEC38 0AE7BF3E 1AEC0CB6 7A034409 06C7DFB3 BCD4B6EE EBB7E371 F0094AD4 A816088D
98DBC791 D0671CAC A12236CD F8F39E15 AEB96FAE B39606D5 B04AC581 746A663D 00DD2B74 16BAA911
72E89D53 09D834F7 8C1E31B4 483BB971 85931BAD 7BE1B9B5 7EBAC034 9F854446 9E60C32F 6075FB04
68A68147 FF013537 DF792FFC E024F857 10CC2B56 1A62B62D A36AEFD6 0850714F 49170FD9 4A0010C6
D4B651B6 4F3A3A5E 58C9687B EDDCD9E4 FEDAB16B 884D1FE6 DFA117B2 AB821F74 E0BF7ACD A2269859

2A430968 F1608606 1904CE20 1847934B 11CA0F9E 9528F5A9 D0CE8F01 5C9AEA79 934FDDA6 D3AB48C8
571CE235 4B79742A A498CB8C DDE6BD1F A5946345 A1A652F6 A76B6777 AD87C912 4C7D7065 F74808DB
2E80371C 70471580 B0C7C457 A79EA5E7 242FA31F F8E139FA E169A169 92F5F029 162664CE 78B33332
4B3BDB4C 682BF9B2 0626D64D CE603F33 2E9593F6 2B67A6B0 02DEB6DD 2E7D4FAD 3F33C38F 202DE204
53274906 11B2AE6F 849CF779 B9B74AD9 BA6CF397 F6132612 0777CE46 92F85DC2 ADC269D1 B6233258
2D823132 A9712754 77A0CF1D CCF4B2BF 096D9110 F74E2A01 B1ED0650 2333B2AB 1AE697EA 34F2EF8C
6E47B043 1831706C B5AFCD75 754FA795 28F65B36 51E184BC ED030661 EE4A8D67 0FBAE267 96E8CDB6
6F388ED6 644AF851 885C7F92 4CC7CB20 968AA50E 8230A3B3 9C2BB5DD 4D753D94 BE5DD9A4 272CF827
0DA649CB 8A63172F 8FB028CD 951E7621 5824A4EE 28405D3C 5E5DFDA6 C7CE293F 4A40AC8F C5B7168F
A54AD3D0 B81A0F8F 50C16436 6CCDEC1C 9A40DCE9 F0A31133 35D89EAE B36F4D31 BB671306 4CDA8835
E2AA4529 F4212932 7C6F7E8A B760654D 58D17E44 8F6D5CBC A66BD7E3 3810D270 DD3B9436 B1BF46B9
A17C9D11 A5A6B148 416C6963 65426F62 7CBA5B19 069EE66A A79D4904 13D11846 B9BA76DD 22567F80
9CF23B6D 964BB265 A9760C99 CB6F7063 43FED056 37085864 958D6C90 902ABA7D 405FBEDF 7B781599
861E9148 5FB7623D 2794F495 031A3559 8B493BD4 5BE37813 ABC710FC C1F34482 32D906A4 69EBC121
6A802A70 52D5617C D430FB56 FBA729D4 1D9BD668 E9EB9600

$Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B)$：72747133 56B7479A 2D592732 0E6B888A FC4D1769 66FF841D 8F480AF0
479BFDB7

0x83$\|g_1\|Hash(g_2\|g_3\|ID_A\|ID_B\|R_A\|R_B)$：

8328542F B6954C84 BE6A5F29 88A31CB6 817BA078 1966FA83 D9673A95 77D3C0C1 345E27C1 9FC02ED9
AE37F5BB 7BE9C03C 2B87DE02 7539CCF0 3E6B7D36 DE4AB45C D1A1ABFC D30C57DB 0F1A838E 3A8F2BF8
23479C97 8BD13723 0506EA62 49C89104 9E349747 7913AB89 F5E2960F 382B1B5C 8EE09DE0 FA498BA9
5C4409D6 30D343DA 404FEC93 472DA33A 4DB65990 95C0CF89 5E3A7B99 3EE5E4EB E3B9AB7D 7D5FF2A3
D1647BA1 54C3E8E1 85DFC336 57C1F128 D480F3F7 E3F16801 208029E1 9434C733 BB73F216 93C66FC2
3724DB26 380C5262 23C705DA F6BA18B7 63A68623 C86A632B 050F63A0 71A6D62E A45B59A1 942DFF53
35D1A232 C9C5664F AD5D6AF5 4C11418B 0D8C8E9D 8D905780 D50E7790 67F2C4B1 C8F83A8B 59D735BB
52AF35F5 6730BDE5 AC861CCD 99786172 67CE4AD9 789F7773 9E62F2E5 7B48C2FF 26D2E90A 79A1D86B
939B1CA0 8F64712E 33AEDA3F 44BD6CB6 33E0F722 211E344D 73EC9BBE BC921427 656BA584 CE742A2A
3AB41C15 D3EF94ED EB8EF74A 2BDCDAAE CC09ABA5 67981F64 37727471 3356B747 9A2D5927 320E6B88
8AFC4D17 6966FF84 1D8F480A F0479BFD B7

选项 S_2：195D1B72 56BA7E0E 67C71202 A25F8C94 FF824170 2C2F55D6 13AE1C6B 98215172

S_2=S_A,从A到B的密钥确认成功!

附 录 C
密钥封装机制示例

C.1 一般要求

本附录选用 SM3 算法给出的密码杂凑函数，其输入是长度小于 2^{64} 的消息比特串，输出是长度为 256 比特的杂凑值，记为 $H_{256}(\ )$。

本附录中，所有用 16 进制表示的数，左边为高位，右边为低位。

本附录中，明文采用ASCII编码。

C.2 密钥封装

椭圆曲线方程为：$y^2 = x^3 + b$

基域特征 q：B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27 E351457D

方程参数 b：05

群 $\mathbb{G}_1$,$\mathbb{G}_2$ 的阶 N：B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C D69ECF25

余因子 cf：1

嵌入次数 k：12

扭曲线的参数 β : $\sqrt{-2}$

群 $\mathbb{G}_1$ 的生成元 $P_1 = (x_{P_1}, y_{P_1})$：

坐标 x_{P_1}：93DE051D 62BF718F F5ED0704 487D01D6 E1E40869 09DC3280 E8C4E481 7C66DDDD

坐标 y_{P_1}：21FE8DDA 4F21E607 63106512 5C395BBC 1C1C00CB FA602435 0C464CD7 0A3EA616

群 $\mathbb{G}_2$ 的生成元 $P_2 = (x_{P_2}, y_{P_2})$：

坐标 x_{P_2}：(85AEF3D0 78640C98 597B6027 B441A01F F1DD2C19 0F5E93C4 54806C11 D8806141,
37227552 92130B08 D2AAB97F D34EC120 EE265948 D19C17AB F9B7213B AF82D65B)

坐标 y_{P_2}：(17509B09 2E845C12 66BA0D26 2CBEE6ED 0736A96F A347C8BD 856DC76B 84EBEB96,
A7CF28D5 19BE3DA6 5F317015 3D278FF2 47EFBA98 A71A0811 6215BBA5 C999A7C7)

双线性对的识别符 eid：0x04

加密主密钥和用户密钥产生过程中的相关值：

加密主私钥 ke：01EDEE 3778F441 F8DEA3D9 FA0ACC4E 07EE36C9 3F9A0861 8AF4AD85 CEDE1C22

加密主公钥 $P_{pub\text{-}e} = [ke]P_1 = (x_{P_{pub\text{-}e}}, y_{P_{pub\text{-}e}})$：

坐标 $x_{P_{pub\text{-}e}}$：787ED7B8 A51F3AB8 4E0A6600 3F32DA5C 720B17EC A7137D39 ABC66E3C 80A892FF

坐标 $y_{P_{pub\text{-}e}}$：769DE617 91E5ADC4 B9FF85A3 1354900B 20287127 9A8C49DC 3F220F64 4C57A7B1

加密私钥生成函数识别符 hid：0x03

实体 B 的标识 ID_B：Bob

ID_B 的 16 进制表示：426F62

在有限域 F_N 上计算 $t_1 = H_1(ID_B \| hid, N) + ke$：

$ID_B \| hid$：426F6203

$H_1(ID_B \| hid, N)$：9CB1F628 8CE0E510 43CE7234 4582FFC3 01E0A812 A7F5F200 4B85547A 24B82716

t_1：9CB3E416 C459D952 3CAD160E 3F8DCC11 09CEDEDB E78FFA61 D67A01FF F3964338

在有限域 F_N 上计算 $t_2 = ke \cdot t_1^{-1}$：

t_2：864E4D83 91948B37 535ECFA4 4C3F8D4E 545ADA50 2FF8229C 7C32F529 AF406E06

计算 $de_B=[t_2]P_2=(x_{deB}, y_{deB})$：

坐标 x_{deB}：(94736ACD 2C8C8796 CC4785E9 38301A13 9A059D35 37B64141 40B2D31E ECF41683,
115BAE85 F5D8BC6C 3DBD9E53 42979ACC CF3C2F4F 28420B1C B4F8C0B5 9A19B158)

坐标 y_{deB}：(7AA5E475 70DA7600 CD760A0C F7BEAF71 C447F384 4753FE74 FA7BA92C A7D3B55F,
27538A62 E7F7BFB5 1DCE0870 4796D94C 9D56734F 119EA447 32B50E31 CDEB75C1)

封装密钥的长度:0100

密钥封装步骤 A1-A7 中的相关值：

计算 $Q_B=[H_1(ID_B||hid, N)]P_1+P_{pub-e}=(x_{QB}, y_{QB})$：

$ID_B||hid$：426F6203

$H_1(ID_B||hid,N)$：9CB1F628 8CE0E510 43CE7234 4582FFC3 01E0A812 A7F5F200 4B85547A 24B82716

坐标 x_{QB}：709D1658 08B0A43E 2574E203 FA885ABC BAB16A24 0C4C1916 552E7C43 D09763B8

坐标 y_{QB}：693269A6 BE2456F4 33337582 74786B60 51FF87B7 F198DA4B A1A2C6E3 36F51FCC

产生随机数 r：7401 5F8489C0 1EF42704 56F9E647 5BFB602B DE7F33FD 482AB4E3 684A6722

计算 $C=[r]Q_B=(x_C, y_C)$：

坐标 x_C：1EDEE2C3 F4659144 91DE44CE FB2CB434 AB02C308 D9DC5E20 67B4FED5 AAAC8A0F

坐标 y_C：1C9B4C43 5ECA35AB 83BB7341 74C0F78F DE81A533 74AFF3B3 602BBC5E 37BE9A4C

计算 $g=e(P_{pub-e}, P_2)$：

(9746FC5B 231CEDF3 6F835C47 893D63C6 FF652BCB 92375CE3 C2AB256D 1FD56413,
232A2F80 CFBAE061 F196BB99 213D5030 6648AC33 CDC78E8F 8A1563FF BF3BD3EB,
68E8A16C 0AC905F6 92904ABC C004B1AC F12106BD 0A15B6E7 08D76E72 B9288EF2,
9436A60C 403F4F8B AC4DD3E3 93E25419 E634FC2B 3DAF247F 6092A802 F60D5C58,
A140EAEF 3893D574 CB83C01D 951A53F5 1975760B E57F3BBD 89817498 D2158352,
95A2BCCE 25359D03 3FC654BD 6A9E462E 5BD0686F F6DDD745 5F71FFF1 5AFFD3F0,
B0432019 0B1E90CE DF6AC570 147A23AE 6F0EAE45 034E6C62 124DD6E8 978F78AD,
A504E3B4 3C1DD367 94217FA1 B05AC046 C4131854 C3D3E3A5 B5967A64 A861F0A2,
897F7B35 D1C0E21D 84D75CFF AC08C73E 744A16A4 7EE76E28 A0B03849 888D10FF,
24443BB4 24B12C41 EAF6D34D 92520590 1F5CBA59 CFEBA352 24660DB3 848B0BF5,
0825403F B3F681AB 2B036DBB A25483D5 CB98BD56 F3DF95F0 A7A705A2 F6FD804B,
9CE7BC68 062182CF 5D9F4A98 C5A4ED1F 3B4CE4EA 817D19ED 7EF2CE98 E6F5864D)

计算 $w=g^r$：

(8EAB0CD6 D0C95A6B BB7051AC 848FDFB9 689E5E5C 486B1294 557189B3 38B53B1D,
78082BB4 0152DC35 AC774442 CC6408FF D68494D9 953D77BF 55E30E84 697F6674,
5AAF5223 9E46B037 3B3168BA B75C32E0 48B5FAEB ABFA1F7F 9BA6B4C0 C90E65B0,
75F6A2D9 ED54C87C DDD2EAA7 87032320 205E7AC7 D7FEAA86 95AB2BF7 F5710861,
247C2034 CCF4A143 2DA1876D 023AD6D7 4FF1678F DA3AF37A 3D9F613C DE805798,
8B07151B AC93AF48 D78D86C2 6EA97F24 E2DACC84 104CCE87 91FE90BA 61B2049C,
AAC6AB38 EA07F996 6173FD9B BF34AAB5 8EE84CD3 777A9FD0 0BBCA1DC 09CF8696,
A1040465 BD723AE5 13C4BE3E F2CFDC08 8A935F0B 207DEED7 AAD5CE2F C37D4203,
4D874A4C E9B3B587 65B1252A 0880952B 4FF3C97E A1A4CFDC 67A0A007 2541A03D,
3924EABC 443B0503 510B93BB CD98EB70 E0192B82 1D14D69C CB2513A1 A7421EB7,
A018A035 E8FB61F2 71DE1C5B 3E781C63 508C113B 3EAC5378 05EAE164 D732FAD0,
56BEA27C 8624D506 4C9C278A 193D63F6 908EE558 DF5F5E07 21317FC6 E829C242)

计算 $K=KDF(C||w||ID_B, klen)$：

$C||w||ID_B$：

1EDEE2C3 F4659144 91DE44CE FB2CB434 AB02C308 D9DC5E20 67B4FED5 AAAC8A0F 1C9B4C43
5ECA35AB 83BB7341 74C0F78F DE81A533 74AFF3B3 602BBC5E 37BE9A4C 8EAB0CD6 D0C95A6B

BB7051AC 848FDFB9 689E5E5C 486B1294 557189B3 38B53B1D 78082BB4 0152DC35 AC774442
CC6408FF D68494D9 953D77BF 55E30E84 697F6674 5AAF5223 9E46B037 3B3168BA B75C32E0
48B5FAEB ABFA1F7F 9BA6B4C0 C90E65B0 75F6A2D9 ED54C87C DDD2EAA7 87032320 205E7AC7
D7FEAA86 95AB2BF7 F5710861 247C2034 CCF4A143 2DA1876D 023AD6D7 4FF1678F DA3AF37A
3D9F613C DE805798 8B07151B AC93AF48 D78D86C2 6EA97F24 E2DACC84 104CCE87 91FE90BA
61B2049C AAC6AB38 EA07F996 6173FD9B BF34AAB5 8EE84CD3 777A9FD0 0BBCA1DC 09CF8696
A1040465 BD723AE5 13C4BE3E F2CFDC08 8A935F0B 207DEED7 AAD5CE2F C37D4203 4D874A4C
E9B3B587 65B1252A 0880952B 4FF3C97E A1A4CFDC 67A0A007 2541A03D 3924EABC 443B0503
510B93BB CD98EB70 E0192B82 1D14D69C CB2513A1 A7421EB7 A018A035 E8FB61F2 71DE1C5B
3E781C63 508C113B 3EAC5378 05EAE164 D732FAD0 56BEA27C 8624D506 4C9C278A 193D63F6
908EE558 DF5F5E07 21317FC6 E829C242 426F62

K：4FF5CF86 D2AD40C8 F4BAC98D 76ABDBDE 0C0E2F0A 829D3F91 1EF5B2BC E0695480

解封装步骤 B1-B4 中的相关值：

计算 $w'=e(C', de_B)$：

(8EAB0CD6 D0C95A6B BB7051AC 848FDFB9 689E5E5C 486B1294 557189B3 38B53B1D,
78082BB4 0152DC35 AC774442 CC6408FF D68494D9 953D77BF 55E30E84 697F6674,
5AAF5223 9E46B037 3B3168BA B75C32E0 48B5FAEB ABFA1F7F 9BA6B4C0 C90E65B0,
75F6A2D9 ED54C87C DDD2EAA7 87032320 205E7AC7 D7FEAA86 95AB2BF7 F5710861,
247C2034 CCF4A143 2DA1876D 023AD6D7 4FF1678F DA3AF37A 3D9F613C DE805798,
8B07151B AC93AF48 D78D86C2 6EA97F24 E2DACC84 104CCE87 91FE90BA 61B2049C,
AAC6AB38 EA07F996 6173FD9B BF34AAB5 8EE84CD3 777A9FD0 0BBCA1DC 09CF8696,
A1040465 BD723AE5 13C4BE3E F2CFDC08 8A935F0B 207DEED7 AAD5CE2F C37D4203,
4D874A4C E9B3B587 65B1252A 0880952B 4FF3C97E A1A4CFDC 67A0A007 2541A03D,
3924EABC 443B0503 510B93BB CD98EB70 E0192B82 1D14D69C CB2513A1 A7421EB7,
A018A035 E8FB61F2 71DE1C5B 3E781C63 508C113B 3EAC5378 05EAE164 D732FAD0,
56BEA27C 8624D506 4C9C278A 193D63F6 908EE558 DF5F5E07 21317FC6 E829C242)

计算 $K'=KDF(C'||w'||ID_B, klen)$：

$C'||w'||ID_B$：

1EDEE2C3 F4659144 91DE44CE FB2CB434 AB02C308 D9DC5E20 67B4FED5 AAAC8A0F 1C9B4C43
5ECA35AB 83BB7341 74C0F78F DE81A533 74AFF3B3 602BBC5E 37BE9A4C 8EAB0CD6 D0C95A6B
BB7051AC 848FDFB9 689E5E5C 486B1294 557189B3 38B53B1D 78082BB4 0152DC35 AC774442
CC6408FF D68494D9 953D77BF 55E30E84 697F6674 5AAF5223 9E46B037 3B3168BA B75C32E0
48B5FAEB ABFA1F7F 9BA6B4C0 C90E65B0 75F6A2D9 ED54C87C DDD2EAA7 87032320 205E7AC7
D7FEAA86 95AB2BF7 F5710861 247C2034 CCF4A143 2DA1876D 023AD6D7 4FF1678F DA3AF37A
3D9F613C DE805798 8B07151B AC93AF48 D78D86C2 6EA97F24 E2DACC84 104CCE87 91FE90BA
61B2049C AAC6AB38 EA07F996 6173FD9B BF34AAB5 8EE84CD3 777A9FD0 0BBCA1DC 09CF8696
A1040465 BD723AE5 13C4BE3E F2CFDC08 8A935F0B 207DEED7 AAD5CE2F C37D4203 4D874A4C
E9B3B587 65B1252A 0880952B 4FF3C97E A1A4CFDC 67A0A007 2541A03D 3924EABC 443B0503
510B93BB CD98EB70 E0192B82 1D14D69C CB2513A1 A7421EB7 A018A035 E8FB61F2 71DE1C5B
3E781C63 508C113B 3EAC5378 05EAE164 D732FAD0 56BEA27C 8624D506 4C9C278A 193D63F6
908EE558 DF5F5E07 21317FC6 E829C242 426F62

K'：4FF5CF86 D2AD40C8 F4BAC98D 76ABDBDE 0C0E2F0A 829D3F91 1EF5B2BC E0695480

附 录 D
公钥加密算法示例

D.1 一般要求

本附录选用 SM3 算法给出的密码杂凑函数，其输入是长度小于 2^{64} 的消息比特串，输出是长度为 256 比特的杂凑值，记为 $H_{256}(\ )$。

本附录选用 SM4 算法给出的分组密码函数，作为加密所用的分组密码算法。在此示例中，分组长度为 128 比特，填充方式遵循 PKCS#5，工作模式为 ECB。

本附录中，所有用 16 进制表示的数，左边为高位，右边为低位。

本附录中，明文采用ASCII编码。

D.2 公钥加解密

椭圆曲线方程为：$y^2 = x^3 + b$

基域特征 q：B6400000 02A3A6F1 D603AB4F F58EC745 21F2934B 1A7AEEDB E56F9B27 E351457D

方程参数 b：05

群 $\mathbb{G}_1$,$\mathbb{G}_2$ 的阶 N：B6400000 02A3A6F1 D603AB4F F58EC744 49F2934B 18EA8BEE E56EE19C D69ECF25

余因子 cf：1

嵌入次数 k：12

扭曲线的参数 β：$\sqrt{-2}$

群 $\mathbb{G}_1$ 的生成元 $P_1 = (x_{P_1}, y_{P_1})$：

坐标 x_{P_1}：93DE051D 62BF718F F5ED0704 487D01D6 E1E40869 09DC3280 E8C4E481 7C66DDDD

坐标 y_{P_1}：21FE8DDA 4F21E607 63106512 5C395BBC 1C1C00CB FA602435 0C464CD7 0A3EA616

群 $\mathbb{G}_2$ 的生成元 $P_2 = (x_{P_2}, y_{P_2})$：

坐标 x_{P_2}：(85AEF3D0 78640C98 597B6027 B441A01F F1DD2C19 0F5E93C4 54806C11 D8806141,
37227552 92130B08 D2AAB97F D34EC120 EE265948 D19C17AB F9B7213B AF82D65B)

坐标 y_{P_2}：(17509B09 2E845C12 66BA0D26 2CBEE6ED 0736A96F A347C8BD 856DC76B 84EBEB96,
A7CF28D5 19BE3DA6 5F317015 3D278FF2 47EFBA98 A71A0811 6215BBA5 C999A7C7)

双线性对的识别符 eid：0x04

加密主密钥和用户加密密钥产生过程中的相关值：

加密主私钥 ke：01EDEE 3778F441 F8DEA3D9 FA0ACC4E 07EE36C9 3F9A0861 8AF4AD85 CEDE1C22

加密主公钥 $P_{pub\text{-}e} = [ke]P_1 = (x_{P_{pub\text{-}e}}, y_{P_{pub\text{-}e}})$：

坐标 $x_{P_{pub\text{-}e}}$：787ED7B8 A51F3AB8 4E0A6600 3F32DA5C 720B17EC A7137D39 ABC66E3C 80A892FF

坐标 $y_{P_{pub\text{-}e}}$：769DE617 91E5ADC4 B9FF85A3 1354900B 20287127 9A8C49DC 3F220F64 4C57A7B1

加密私钥生成函数识别符 hid：0x03

实体 B 的标识 ID_B：Bob

ID_B 的 16 进制表示：426F62

在有限域 F_N 上计算 $t_1 = H_1(ID_B \| hid, N) + ke$：

$ID_B \| hid$：426F6203

$H_1(ID_B \| hid, N)$：9CB1F628 8CE0E510 43CE7234 4582FFC3 01E0A812 A7F5F200 4B85547A 24B82716

t_1：9CB3E416 C459D952 3CAD160E 3F8DCC11 09CEDEDB E78FFA61 D67A01FF F3964338

在有限域 F_N 上计算 $t_2 = ke \cdot t_1^{-1}$：

t_2：864E4D83 91948B37 535ECFA4 4C3F8D4E 545ADA50 2FF8229C 7C32F529 AF406E06

计算 $de_B = [t_2]P_2 = (x_{de_B}, y_{de_B})$：

坐标 x_{deB}：(94736ACD 2C8C8796 CC4785E9 38301A13 9A059D35 37B64141 40B2D31E ECF41683,
115BAE85 F5D8BC6C 3DBD9E53 42979ACC CF3C2F4F 28420B1C B4F8C0B5 9A19B158)
坐标 y_{deB}：(7AA5E475 70DA7600 CD760A0C F7BEAF71 C447F384 4753FE74 FA7BA92C A7D3B55F,
27538A62 E7F7BFB5 1DCE0870 4796D94C 9D56734F 119EA447 32B50E31 CDEB75C1)
待加密消息 M 为：Chinese IBE standard
消息 M 的 16 进制表示为：4368696E 65736520 49424520 7374616E 64617264
消息 M 的长度 $mlen$：0xA0
K_1_len：0x80
K_2_len：0x0100

加密算法步骤 A1-A8 中的相关值：

计算 $Q_B=[H_1(ID_B||hid, N)]P_1+P_{pub-e}=(x_{Q_B}, y_{Q_B})$：
$ID_B||hid$：426F6203
$H_1(ID_B||hid,N)$：9CB1F628 8CE0E510 43CE7234 4582FFC3 01E0A812 A7F5F200 4B85547A 24B827
坐标 x_{Q_B}：709D1658 08B0A43E 2574E203 FA885ABC BAB16A24 0C4C1916 552E7C43 D09763B8
坐标 y_{Q_B}：693269A6 BE2456F4 33337582 74786B60 51FF87B7 F198DA4B A1A2C6E3 36F51FCC
产生随机数 r：AAC0 541779C8 FC45E3E2 CB25C12B 5D2576B2 129AE8BB 5EE2CBE5 EC9E785C
计算 $C_1=[r]Q_B=(x_{C_1}, y_{C_1})$：
坐标 x_{C_1}：24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF
坐标 y_{C_1}：42FFCA97 D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0
计算 $g=e(P_{pub-e}, P_2)$：
(9746FC5B 231CEDF3 6F835C47 893D63C6 FF652BCB 92375CE3 C2AB256D 1FD56413 ,
232A2F80 CFBAE061 F196BB99 213D5030 6648AC33 CDC78E8F 8A1563FF BF3BD3EB ,
68E8A16C 0AC905F6 92904ABC C004B1AC F12106BD 0A15B6E7 08D76E72 B9288EF2 ,
9436A60C 403F4F8B AC4DD3E3 93E25419 E634FC2B 3DAF247F 6092A802 F60D5C58 ,
A140EAEF 3893D574 CB83C01D 951A53F5 1975760B E57F3BBD 89817498 D2158352 ,
95A2BCCE 25359D03 3FC654BD 6A9E462E 5BD0686F F6DDD745 5F71FFF1 5AFFD3F0 ,
B0432019 0B1E90CE DF6AC570 147A23AE 6F0EAE45 034E6C62 124DD6E8 978F78AD ,
A504E3B4 3C1DD367 94217FA1 B05AC046 C4131854 C3D3E3A5 B5967A64 A861F0A2 ,
897F7B35 D1C0E21D 84D75CFF AC08C73E 744A16A4 7EE76E28 A0B03849 888D10FF ,
24443BB4 24B12C41 EAF6D34D 92520590 1F5CBA59 CFEBA352 24660DB3 848B0BF5 ,
0825403F B3F681AB 2B036DBB A25483D5 CB98BD56 F3DF95F0 A7A705A2 F6FD804B ,
9CE7BC68 062182CF 5D9F4A98 C5A4ED1F 3B4CE4EA 817D19ED 7EF2CE98 E6F5864D)
计算 $w=g^r$：
(63253798 B7535975 A90F2025 61FC5457 0FEE88BF 69E3B7A5 12697069 E59E1F5D ,
42D54B98 4AF01D71 0BA0030C 18738F6B 14E4DF47 2ACAF893 99228D85 AF117904 ,
B426DFF0 40C49F9A 43BCD7FD 7D757B7D 1D8D7311 C08FC3B5 7616C5EE 137785A3 ,
28D19396 DBDFAC50 EEE62B1C 7F994BB6 F9BD9EFB 2221A1BE 1B6EB3E8 F71485B4 ,
A3EEF46E 1B99F614 D7BD7F57 574BA7EB B502AF0B DABA0787 C5C4DBC5 6A344A25 ,
A06790B6 05CEA0BB AF34776D 6B1FC019 8A02D05B BAAC6F64 A555AB2C A576F0DA ,
B405CBBF 22197B94 FD18D27D A0B0E52C 8754EE94 27963469 1FEA6E13 FFD0584E ,
AA2A94A7 E2259B67 1896302B 4275AE3E 8CF20100 98D5BEAF 19D0A6E6 0354E1C5 ,
5C97E64F 848B06D3 9BA8828F F59502C0 81D3DAE6 8F35F7E6 448DB96D 220A0FBA ,
02BE03C5 1BF062B6 F564AE0B FB42DCA3 6E71D387 512E3BCC CA3379B7 3EC47176 ,
52BE92FB 9E78BA9E 1D80A156 06580493 5742DBD2 B9675430 11AAC533 33909FBF ,

5FADEC14 A2FBD152 48E77467 442A6969 8246FB03 14C7A824 6D952219 DD2144ED ）

按加密明文的方法分类进行计算：

a) 加密明文的方法为基于 *KDF* 的序列密码：

计算 $klen=mlen+K_2\_len$：01A0

计算 $K=KDF(C_1||w||ID_B, klen)=K_1||K_2$：

$C_1||w||ID_B$ ：

24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF 42FFCA97
D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0 63253798 B7535975
A90F2025 61FC5457 0FEE88BF 69E3B7A5 12697069 E59E1F5D 42D54B98 4AF01D71 0BA0030C
18738F6B 14E4DF47 2ACAF893 99228D85 AF117904 B426DFF0 40C49F9A 43BCD7FD 7D757B7D
1D8D7311 C08FC3B5 7616C5EE 137785A3 28D19396 DBDFAC50 EEE62B1C 7F994BB6 F9BD9EFB
2221A1BE 1B6EB3E8 F71485B4 A3EEF46E 1B99F614 D7BD7F57 574BA7EB B502AF0B DABA0787
C5C4DBC5 6A344A25 A06790B6 05CEA0BB AF34776D 6B1FC019 8A02D05B BAAC6F64 A555AB2C
A576F0DA B405CBBF 22197B94 FD18D27D A0B0E52C 8754EE94 27963469 1FEA6E13 FFD0584E
AA2A94A7 E2259B67 1896302B 4275AE3E 8CF20100 98D5BEAF 19D0A6E6 0354E1C5 5C97E64F
848B06D3 9BA8828F F59502C0 81D3DAE6 8F35F7E6 448DB96D 220A0FBA 02BE03C5 1BF062B6
F564AE0B FB42DCA3 6E71D387 512E3BCC CA3379B7 3EC47176 52BE92FB 9E78BA9E 1D80A156
06580493 5742DBD2 B9675430 11AAC533 33909FBF 5FADEC14 A2FBD152 48E77467 442A6969
8246FB03 14C7A824 6D952219 DD2144ED 426F62

$K=K_1||K_2$：58373260 F067EC48 667C21C1 44F8BC33 CD304978 8651FFD5 F738003E 51DF3117
4D0E4E40 2FD87F45 81B612F7 4259DB57 4F67ECE6

计算 $C_2=M\oplus K_1$：

K_1：58373260 F067EC48 667C21C1 44F8BC33 CD304978

C_2：1B5F5B0E 95148968 2F3E64E1 378CDD5D A9513B1C

计算 $C_3=MAC(K_2, C_2)$：

K_2：8651FFD5 F738003E 51DF3117 4D0E4E40 2FD87F45 81B612F7 4259DB57 4F67ECE6

C_3：BA672387 BCD6DE50 16A158A5 2BB2E7FC 429197BC AB70B25A FEE37A2B 9DB9F367

计算 $C=C_1||C_3||C_2$：

24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF 42FFCA97
D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0 BA672387 BCD6DE50
16A158A5 2BB2E7FC 429197BC AB70B25A FEE37A2B 9DB9F367 1B5F5B0E 95148968 2F3E64E1
378CDD5D A9513B1C

b) 加密明文的方法为分组密码算法：

计算 $klen= K_1\_len +K_2\_len$：0180

计算 $K=KDF(C_1||w||ID_B, klen) =K_1||K_2$：

$C_1||w||ID_B$ ：

24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF 42FFCA97
D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0 63253798 B7535975
A90F2025 61FC5457 0FEE88BF 69E3B7A5 12697069 E59E1F5D 42D54B98 4AF01D71 0BA0030C
18738F6B 14E4DF47 2ACAF893 99228D85 AF117904 B426DFF0 40C49F9A 43BCD7FD 7D757B7D
1D8D7311 C08FC3B5 7616C5EE 137785A3 28D19396 DBDFAC50 EEE62B1C 7F994BB6 F9BD9EFB
2221A1BE 1B6EB3E8 F71485B4 A3EEF46E 1B99F614 D7BD7F57 574BA7EB B502AF0B DABA0787
C5C4DBC5 6A344A25 A06790B6 05CEA0BB AF34776D 6B1FC019 8A02D05B BAAC6F64 A555AB2C
A576F0DA B405CBBF 22197B94 FD18D27D A0B0E52C 8754EE94 27963469 1FEA6E13 FFD0584E
AA2A94A7 E2259B67 1896302B 4275AE3E 8CF20100 98D5BEAF 19D0A6E6 0354E1C5 5C97E64F
848B06D3 9BA8828F F59502C0 81D3DAE6 8F35F7E6 448DB96D 220A0FBA 02BE03C5 1BF062B6
F564AE0B FB42DCA3 6E71D387 512E3BCC CA3379B7 3EC47176 52BE92FB 9E78BA9E 1D80A156

06580493 5742DBD2 B9675430 11AAC533 33909FBF 5FADEC14 A2FBD152 48E77467 442A6969 8246FB03 14C7A824 6D952219 DD2144ED 426F62

$K=K_1\|K_2$：58373260 F067EC48 667C21C1 44F8BC33 CD304978 8651FFD5 F738003E 51DF3117 4D0E4E40 2FD87F45 81B612F7 4259DB57

计算 $C_2=Enc(K_1, M)$：

K_1：58373260 F067EC48 667C21C1 44F8BC33

M 填充为：4368696E 65736520 49424520 7374616E 64617264 0C0C0C0C 0C0C0C0C 0C0C0C0C 0C0C0C0C

C_2：E05B6FAC 6F11B965 268C994F 00DBA7A8 BB00FD60 583546CB DF464925 0863F10A

计算 $C_3=MAC(K_2, C_2)$：

K_2：CD304978 8651FFD5 F738003E 51DF3117 4D0E4E40 2FD87F45 81B612F7 4259DB57

C_3：FD3C98DD 92C44C68 332675A3 70CCEEDE 31E0C5CD 209C2576 01149D12 B394A2BE

计算 $C=C_1\|C_3\|C_2$：

24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF 42FFCA97 D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0 FD3C98DD 92C44C68 332675A3 70CCEEDE 31E0C5CD 209C2576 01149D12 B394A2BE E05B6FAC 6F11B965 268C994F 00DBA7A8 BB00FD60 583546CB DF464925 0863F10A

解密算法步骤 B1-B5 中的相关值：

计算 $w'=e(C_1', de_B)$：

(63253798 B7535975 A90F2025 61FC5457 0FEE88BF 69E3B7A5 12697069 E59E1F5D,
42D54B98 4AF01D71 0BA0030C 18738F6B 14E4DF47 2ACAF893 99228D85 AF117904,
B426DFF0 40C49F9A 43BCD7FD 7D757B7D 1D8D7311 C08FC3B5 7616C5EE 137785A3,
28D19396 DBDFAC50 EEE62B1C 7F994BB6 F9BD9EFB 2221A1BE 1B6EB3E8 F71485B4,
A3EEF46E 1B99F614 D7BD7F57 574BA7EB B502AF0B DABA0787 C5C4DBC5 6A344A25,
A06790B6 05CEA0BB AF34776D 6B1FC019 8A02D05B BAAC6F64 A555AB2C A576F0DA,
B405CBBF 22197B94 FD18D27D A0B0E52C 8754EE94 27963469 1FEA6E13 FFD0584E,
AA2A94A7 E2259B67 1896302B 4275AE3E 8CF20100 98D5BEAF 19D0A6E6 0354E1C5,
5C97E64F 848B06D3 9BA8828F F59502C0 81D3DAE6 8F35F7E6 448DB96D 220A0FBA,
02BE03C5 1BF062B6 F564AE0B FB42DCA3 6E71D387 512E3BCC CA3379B7 3EC47176,
52BE92FB 9E78BA9E 1D80A156 06580493 5742DBD2 B9675430 11AAC533 33909FBF,
5FADEC14 A2FBD152 48E77467 442A6969 8246FB03 14C7A824 6D952219 DD2144ED)

按加密明文的方法分类进行计算：

a) 加密明文的方法为基于 *KDF* 的序列密码：

计算 $klen=mlen+K_2\_len$：01A0

计算 $K'=KDF(C_1'\|w'\|ID_B, klen)=K_1\|K_2$：

$C_1'\|w'\|ID_B$：

24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF 42FFCA97 D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0 63253798 B7535975 A90F2025 61FC5457 0FEE88BF 69E3B7A5 12697069 E59E1F5D 42D54B98 4AF01D71 0BA0030C 18738F6B 14E4DF47 2ACAF893 99228D85 AF117904 B426DFF0 40C49F9A 43BCD7FD 7D757B7D 1D8D7311 C08FC3B5 7616C5EE 137785A3 28D19396 DBDFAC50 EEE62B1C 7F994BB6 F9BD9EFB 2221A1BE 1B6EB3E8 F71485B4 A3EEF46E 1B99F614 D7BD7F57 574BA7EB B502AF0B DABA0787 C5C4DBC5 6A344A25 A06790B6 05CEA0BB AF34776D 6B1FC019 8A02D05B BAAC6F64 A555AB2C A576F0DA B405CBBF 22197B94 FD18D27D A0B0E52C 8754EE94 27963469 1FEA6E13 FFD0584E AA2A94A7 E2259B67 1896302B 4275AE3E 8CF20100 98D5BEAF 19D0A6E6 0354E1C5 5C97E64F

848B06D3 9BA8828F F59502C0 81D3DAE6 8F35F7E6 448DB96D 220A0FBA 02BE03C5 1BF062B6
F564AE0B FB42DCA3 6E71D387 512E3BCC CA3379B7 3EC47176 52BE92FB 9E78BA9E 1D80A156
06580493 5742DBD2 B9675430 11AAC533 33909FBF 5FADEC14 A2FBD152 48E77467 442A6969
8246FB03 14C7A824 6D952219 DD2144ED 426F62

$K=K_1'\|K_2'$：58373260 F067EC48 667C21C1 44F8BC33 CD304978 8651FFD5 F738003E 51DF3117
4D0E4E40 2FD87F45 81B612F7 4259DB57 4F67ECE6

计算 $M'=C_2'\oplus K_1'$：

K_1'： 58373260 F067EC48 667C21C1 44F8BC33 CD304978

M'： 4368696E 65736520 49424520 7374616E 64617264

计算 $u=MAC(K_2', C_2')$ ：

K_2'：8651FFD5 F738003E 51DF3117 4D0E4E40 2FD87F45 81B612F7 4259DB57 4F67ECE6

u ：BA672387 BCD6DE50 16A158A5 2BB2E7FC 429197BC AB70B25A FEE37A2B 9DB9F367

$u=C_3'$，明文即为：Chinese IBE standard

b) 加密明文的方法为分组密码算法：

计算 $klen=K_1\_len+K_2\_len$：0180

计算 $K'=KDF(C_1'\|w'\|ID_B, klen)=K_1'\|K_2'$：

$C_1'\|w'\|ID_B$ ：

24454711 64490618 E1EE2052 8FF1D545 B0F14C8B CAA44544 F03DAB5D AC07D8FF 42FFCA97
D57CDDC0 5EA405F2 E586FEB3 A6930715 532B8000 759F1305 9ED59AC0 63253798 B7535975
A90F2025 61FC5457 0FEE88BF 69E3B7A5 12697069 E59E1F5D 42D54B98 4AF01D71 0BA0030C
18738F6B 14E4DF47 2ACAF893 99228D85 AF117904 B426DFF0 40C49F9A 43BCD7FD 7D757B7D
1D8D7311 C08FC3B5 7616C5EE 137785A3 28D19396 DBDFAC50 EEE62B1C 7F994BB6 F9BD9EFB
2221A1BE 1B6EB3E8 F71485B4 A3EEF46E 1B99F614 D7BD7F57 574BA7EB B502AF0B DABA0787
C5C4DBC5 6A344A25 A06790B6 05CEA0BB AF34776D 6B1FC019 8A02D05B BAAC6F64 A555AB2C
A576F0DA B405CBBF 22197B94 FD18D27D A0B0E52C 8754EE94 27963469 1FEA6E13 FFD0584E
AA2A94A7 E2259B67 1896302B 4275AE3E 8CF20100 98D5BEAF 19D0A6E6 0354E1C5 5C97E64F
848B06D3 9BA8828F F59502C0 81D3DAE6 8F35F7E6 448DB96D 220A0FBA 02BE03C5 1BF062B6
F564AE0B FB42DCA3 6E71D387 512E3BCC CA3379B7 3EC47176 52BE92FB 9E78BA9E 1D80A156
06580493 5742DBD2 B9675430 11AAC533 33909FBF 5FADEC14 A2FBD152 48E77467 442A6969
8246FB03 14C7A824 6D952219 DD2144ED 426F62

$K'=K_1'\|K_2'$：58373260 F067EC48 667C21C1 44F8BC33 CD304978 8651FFD5 F738003E 51DF3117
4D0E4E40 2FD87F45 81B612F7 4259DB57

计算 $M'=Dec(K_1', C_2')$：

K_1'：58373260 F067EC48 667C21C1 44F8BC33

M'：4368696E 65736520 49424520 7374616E 64617264 0C0C0C0C 0C0C0C0C 0C0C0C0C

计算 $u=MAC(K_2', C_2')$ ：

K_2'：CD304978 8651FFD5 F738003E 51DF3117 4D0E4E40 2FD87F45 81B612F7 4259DB57

u：FD3C98DD 92C44C68 332675A3 70CCEEDE 31E0C5CD 209C2576 01149D12 B394A2BE

$u=C_3'$，明文即为：Chinese IBE standard